Atmel マイクロ コントローラ

AT09331 : ASF USB階層手引書

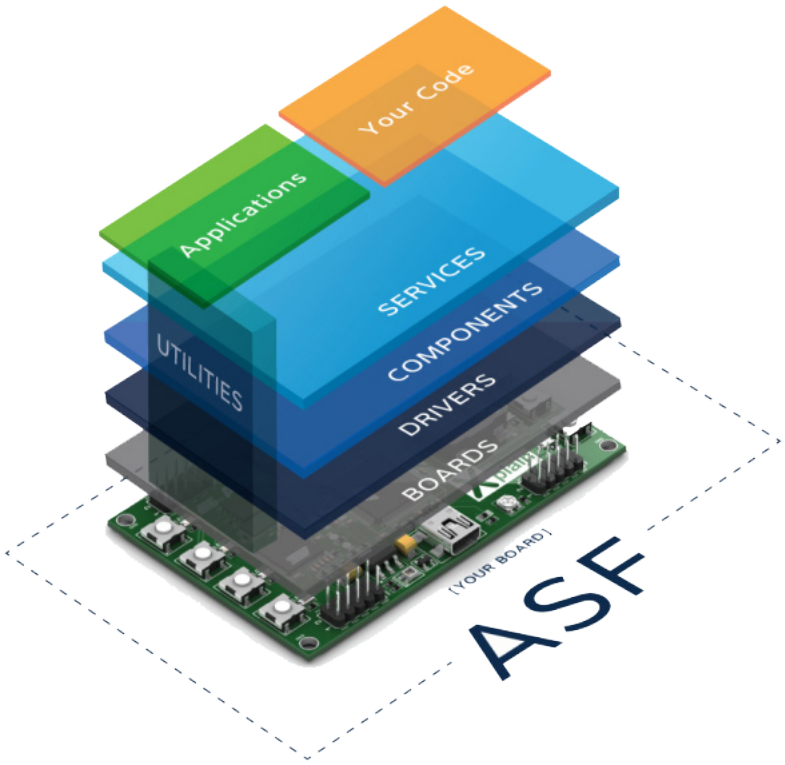
応用記述

序文

Atmel®ソフトウェア枠組み(ASF)はAtmelマイクロ コントローラ デバイス用の無料組み込みソフトウェアの集合です。これはAtmel製品の使用を簡単にし、ハードウェアに対する抽象化と高価値中間ソフトウェア(ミドルウェア)を提供します。

ASFは評価、試作、設計、製造段階に対して使うように設計されています。ASFは図形的使用者インターフェースを持つAtmel Studio IDEに統合され、または様々な商用と解放ソースのコンパイラに対する独立した一式として利用可能です。

この文書は中間ソフトウェア サービスとしてASFに含まれる、応用に関してUSB階層に対するAPIインターフェースを記述します。



ASF USB階層とASFのより多くの情報については以下のリンクでオンライン資料を参照してください。

- ASF-USB
- ASF(www.atmel.com/asf)

本書は一般の方々の便宜のため有志により作成されたもので、Atmel社とは無関係であることを御承知ください。しおりの[はじめに]での内容にご注意ください。

目次

序文 1
ソフトウェア許諾契約 5
1. USB装置制御部(UDC) 6
1.1. API概要 6
1.1.1. 関数定義 6
1.2. USB装置基本構成設定 6
1.2.1. 独自構成設定 6
1.2.2. VBUS監視 7
1.2.3. USB装置基本構成設定 8
1.2.4. conf_clock.h例 9
1.3. USB装置の高度な使用事例 10
1.3.1. USB速度変更 10
1.3.2. USB文字列の使用 11
1.3.3. USB遠隔起動機能の使用 11
1.3.4. バス給電応用推奨 12
1.3.5. USB動的通番 13
2. 通信クラス装置(CDC)用USB装置インターフェース(UDI) 14
2.1. API概要 14
2.1.1. 構造体定義 14
2.1.2. マクロ定義 14
2.1.3. 関数定義 18
2.2. USB装置通信クラス装置単位部(UDI CDC)用の即時開始の手引き 21
2.2.1. 基本的な使用事例 22
2.2.2. 高度な使用事例 23
2.2.3. 複合装置でのCDC 23
2.3. 構成設定ファイル例 25
2.3.1. conf_usb.h 25
2.3.2. conf_clock.h 30
2.3.3. conf_clocks.h 36
2.3.4. conf_board.h 39
3. 人間インターフェース標準装置(標準HID)用USB装置インターフェース(UDI) 41
3.1. API概要 41
3.1.1. 変数と型定義 41
3.1.2. 構造体定義 41
3.1.3. マクロ定義 41
3.1.4. 関数定義 42
3.2. USB装置標準単位部(UDI標準)用の即時開始の手引き 42
3.2.1. 基本的な使用事例 42
3.2.2. 高度な使用事例 43
3.2.3. 複合装置での標準HID 43
3.3. 構成設定ファイル例 45
3.3.1. conf_usb.h 45
3.3.2. conf_clock.h 51
3.3.3. conf_clocks.h 52
3.3.4. conf_board.h 55
4. 人間インターフェース装置キーボード(HIDキーボード)用USB装置インターフェース(UDI) 57
4.1. API概要 57
4.1.1. 変数と型定義 57
4.1.2. 構造体定義 57
4.1.3. マクロ定義 57
4.1.4. 関数定義 58
4.2. USB装置キーボード単位部(UDIキーボード)用の即時開始の手引き 58
4.2.1. 基本的な使用事例 58
4.2.2. 高度な使用事例 59
4.2.3. 複合装置でのHIDキーボード 59
4.3. 構成設定ファイル例 61
4.3.1. conf_usb.h 61

4.3.2. conf_clock.h ... 66
4.3.3. conf_clocks.h ... 69
4.3.4. conf_board.h ... 72
5. 人間インターフェース装置マウス(HIDマウス)用USB装置インターフェース(UDI) ... 73
5.1. API概要 ... 73
5.1.1. 変数と型定義 ... 73
5.1.2. 構造体定義 ... 73
5.1.3. マクロ定義 ... 73
5.1.4. 関数定義 ... 74
5.2. USB装置マウス単位部(UDIマウス)用の即時開始の手引き ... 75
5.2.1. 基本的な使用事例 ... 75
5.2.2. 高度な使用事例 ... 76
5.2.3. 複合装置でのHIDマウス ... 76
5.3. 構成設定ファイル例 ... 77
5.3.1. conf_usb.h ... 77
5.3.2. conf_clock.h ... 83
5.3.3. conf_clocks.h ... 86
5.3.4. conf_board.h ... 89
6. 大容量記憶装置クラス(MSC)用USB装置インターフェース(UDI) ... 91
6.1. API概要 ... 91
6.1.1. 変数と型定義 ... 91
6.1.2. 構造体定義 ... 91
6.1.3. マクロ定義 ... 91
6.1.4. 関数定義 ... 92
6.2. USB装置大容量記憶装置単位部(UDI MSC)用の即時開始の手引き ... 92
6.2.1. 基本的な使用事例 ... 92
6.2.2. 高度な使用事例 ... 94
6.2.3. 複合装置でのMSC ... 94
6.3. 構成設定ファイル例 ... 95
6.3.1. conf_usb.h ... 95
6.3.2. conf_clock.h ... 101
6.3.3. conf_clocks.h ... 107
6.3.4. conf_board.h ... 109
6.3.5. conf_access.h ... 111
6.3.6. conf_virtual_mem.h ... 118
7. 供給者クラス装置用USB装置インターフェース(UDI) ... 120
7.1. API概要 ... 120
7.1.1. 変数と型定義 ... 120
7.1.2. 構造体定義 ... 120
7.1.3. マクロ定義 ... 120
7.1.4. 関数定義 ... 121
7.2. USB装置大容量記憶装置単位部(UDI MSC)用の即時開始の手引き ... 123
7.2.1. 基本的な使用事例 ... 123
7.2.2. 高度な使用事例 ... 124
7.2.3. 複合装置での供給者 ... 125
7.3. 構成設定ファイル例 ... 126
7.3.1. conf_usb.h ... 126
7.3.2. conf_clock.h ... 132
7.3.3. conf_clocks.h ... 134
7.3.4. conf_board.h ... 137
8. USBホスト制御部(UHC) ... 139
8.1. API概要 ... 139
8.1.1. 構造体定義 ... 139
8.1.2. 関数定義 ... 139
8.1.3. 列挙定義 ... 141
8.2. USB装置基本構成設定 ... 142
8.2.1. USBホスト使用者構成設定 ... 142
8.2.2. USBホスト使用者呼び戻し ... 142
8.2.3. USBホスト構成設定段階 ... 142

8.2.4. conf_clock.h例 143
8.3. USBホストの高度な使用事例 144
8.3.1. USB高速(HS)支援許可 144
8.3.2. 複数クラス支援 145
8.3.3. 二重役割支援 145
9. 通信クラス装置(CDC)用USBホスト インターフェース(UHI) 147
9.1. API概要 147
9.1.1. マクロ定義 147
9.1.2. 関数定義 147
9.2. USBホスト通信クラス装置単位部(UHI CDC)用の即時開始の手引き 149
9.2.1. 基本的な使用事例 149
9.2.2. 高度な使用事例 151
9.3. 構成設定ファイル例 151
9.3.1. conf_usb_host.h 151
9.3.2. conf_clock.h 152
9.3.3. conf_clocks.h 156
9.3.4. conf_board.h 158
10. 人間インターフェース装置マウス(HIDマウス)用USBホスト インターフェース(UHI) 161
10.1. API概要 161
10.1.1. マクロ定義 161
10.1.2. 関数定義 161
10.2. USBホスト マウス単位部(UHIマウス)用の即時開始の手引き 162
10.2.1. 基本的な使用事例 162
10.2.2. 高度な使用事例 163
10.3. 構成設定ファイル例 163
10.3.1. conf_usb_host.h 163
10.3.2. conf_clock.h 165
10.3.3. conf_clocks.h 168
10.3.4. conf_board.h 171
11. 大容量記憶装置クラス(MSC)用USBホスト インターフェース(UHI) 173
11.1. API概要 173
11.1.1. 変数と型定義 173
11.1.2. 構造体定義 173
11.1.3. マクロ定義 173
11.1.4. 関数定義 173
11.1.5. 列挙定義 176
11.2. USBホスト大容量記憶装置単位部(UHI USC)用の即時開始の手引き 177
11.2.1. 基本的な使用事例 177
11.2.2. 高度な使用事例 178
11.3. 構成設定ファイル例 178
11.3.1. conf_usb_host.h 178
11.3.2. conf_clock.h 179
11.3.3. conf_clocks.h 183
11.3.4. conf_board.h 185
12. 供給者クラス装置用USBホスト インターフェース(UHI) 187
12.1. API概要 187
12.1.1. マクロ定義 187
12.1.2. 関数定義 187
12.2. USBホスト供給者単位部(UHI供給者)用の即時開始の手引き 190
12.2.1. 基本的な使用事例 190
12.2.2. 高度な使用事例 191
12.3. 構成設定ファイル例 191
12.3.1. conf_usb_host.h 191
12.3.2. conf_clock.h 193
12.3.3. conf_clocks.h 195
12.3.4. conf_board.h 198
文書改訂履歴 199

ソフトウェア許諾契約

変更の有りまたはなしでソースと2進の形式での使用と再配布は以下の条件に合っていれば許されます。

1. ソース コードの再配布は上の著作権通知、この条件一覧、それと以下のお断りを維持しなければなりません。
2. 2進形式での再配布はこの配布で提供された資料や他の素材で、上の著作権通知、この条件一覧、それと以下のお断りを再現しなければなりません。
3. Atmelの名称は先に書かれた許諾の指定なしにこのソフトウェアから派生した販促製品や裏書(保証)に使えないかもしれません。
4. このソフトウェアはAtmelのマイクロ コントローラ製品との関係でだけ再分配して使うことができます。

このソフトウェアは特定目的のための市場性と適合性の暗黙的な保証が明白且つ明確に放棄されることを含みますが、これに限らず、何等かの明示的または暗示的な保証と"現状そのまま"でAtmelによって提供されます。例えそのような損害賠償の可能性を通知されたとしても、このソフトウェアの使用の外の何処でも生じる契約、厳密な責任、(不注意やその他を含む)不法行為かどうかに拘わらず、発生する(代替物またはサービスの獲得、使用、データ、または利益の損失、または事業中断を含みますが、それに限らず)、如何なる直接的、間接的、偶発的、特別的、典型的、または間接的な損害に関して決してAtmelに責任がないでしょう。

1. USB装置制御部(UDC)

UDCはUSB装置の上位抽象物を提供します。主な装置状態(開始/接続/起動)を制御するのにこれらの関数を使うことができます。
USB装置階層内の全てのUSB装置インターフェース(UDI)はUSB列挙(接続認証)を支援するためのUDCに基づきます。
この資料は以下のようにUDCに基づく共通的なUSB装置の使い方を記述します。

- API概要
- USB装置基本構成設定
- USB装置の高度な使用事例

1.1. API概要

1.1.1. 関数定義

1.1.1.1. udc_attach()関数

可能な時に装置をﾊﾞｽに接続

```
void udc_attach( void )
```

警告: ﾄﾞﾗｲﾊﾞにVBUS制御が含まれる場合、ﾎｽﾄから受け入れ可能なVBUSﾚﾍﾞﾙが検出された時に装置を接続します。

1.1.1.2. udc_detach()関数

装置をﾊﾞｽから切断

```
void udc_detach( void )
```

駆動部はUSB線のD-またはD+のﾌﾟﾙｱｯﾌﾟを取り去らなければなりません。

1.1.1.3. udc_get_interface_desc()関数

現在のインターフェース記述子のﾎﾟｲﾝﾀを返します。

```
usb_iface_desc_t UDC_DESC_STORAGE * udc_get_interface_desc( void )
```

戻り値：現在のインターフェース記述子のﾎﾟｲﾝﾀ

1.1.1.4. udc_include_vbus_monitoring()関数

VBUS事象を検定

```
bool udc_include_vbus_monitoring( void )
```

戻り値：VBUS監視が可能ならば真(true)

より多くの詳細についてはVBUS監視をご覧ください。

1.1.1.5. udc_remotewakeup()関数

USBﾄﾞﾗｲﾊﾞは”上方向再開”と呼ばれる再開信号を送ります。

```
void udc_remotewakeup( void )
```

これはﾎｽﾄによって遠隔起動機能が許可されている時にだけ認可されます。

1.1.1.6. udc_start()関数

USB装置階層を開始

```
void udc_start( void )
```

1.1.1.7. udc_stop()関数

USB装置階層を停止

```
void udc_stop( void )
```

1.2. USB装置基本構成設定

1.2.1. 独自構成設定

以下のUSB装置構成設定は応用のconf_usb.hﾌｧｲﾙに含まれなければなりません。

1. USB_DEVICE_VENDOR_ID (ﾜｰﾄﾞ)
   USB orgによって提供された供給者(Vendor)ID (Atmel 0x03EB)
2. USB_DEVICE_PRODUCT_ID (ﾜｰﾄﾞ)
   (usb_atmel.hで言及される)製品(Product)ID
3. USB_DEVICE_MAJOR_VERSION (ﾊﾞｲﾄ)
   装置の主版番号

4. USB_DEVICE_MINOR_VERSION (ﾊﾞｲﾄ)
   装置の副版番号
5. USB_DEVICE_MANUFACTURE_NAME (文字列)
   製造業者(manufacture)のASCII名
6. USB_DEVICE_PRODUCT_NAME (文字列)
   製品(product)のASCII名
7. USB_DEVICE_SERIAL_NAME (文字列)
   通番を許可して設定するためのASCII名
8. USB_DEVICE_POWER (数値)
   最大装置電力 (単位 mA)
9. USB_DEVICE_ATTR (ﾊﾞｲﾄ)
   利用可能なUSB属性:
   - USB_CONFIG_ATTR_SELF_POWERED
   - USB_CONFIG_ATTR_REMOTE_WAKEUP

   注: 遠隔起動が許可された場合、これは遠隔起動呼び戻しを定義します。
10. USB_DEVICE_LOW_SPEED (定義のみ)
    USB装置に低速(LS)での走行を強制
11. USB_DEVICE_HS_SUPPORT (定義のみ)
    USB装置に高速(HS)での走行を認可
12. USB_DEVICE_MAX_EP (ﾊﾞｲﾄ)
    USB装置によって使われる最大ｴﾝﾄﾞﾎﾟｲﾝﾄ番号を定義
    これは既にUDI既定構成設定で定義されます。例えば、
    - ｴﾝﾄﾞﾎﾟｲﾝﾄ制御0x00, ｴﾝﾄﾞﾎﾟｲﾝﾄ0x01, ｴﾝﾄﾞﾎﾟｲﾝﾄ0x82が使われる時はUSB_DEVICE_MAX_EP=2
    - ｴﾝﾄﾞﾎﾟｲﾝﾄ制御0x00だけが使われる時はUSB_DEVICE_MAX_EP=0
    - ｴﾝﾄﾞﾎﾟｲﾝﾄ0x01とｴﾝﾄﾞﾎﾟｲﾝﾄ0x81が使われる時はUSB_DEVICE_MAX_EP=1 (USBBｲﾝﾀｰﾌｪｰｽで可能でない構成設定)

1.2.2. VBUS監視

VBUS監視はUSB自己給電応用にだけ使われます。

- 既定ではVBUSがHighの時、または内部VBUS監視のない装置に対してUSBが開始する時にUSB装置が自動的に接続されます。conf_usb.hﾌｧｲﾙはUSB_DEVICE_ATTACH_AUTO_DISABLE定義を含みません。

```
//#define USB_DEVICE_ATTACH_AUTO_DISABLE
```

- 独自VBUS監視を追加してください。conf_usb.hﾌｧｲﾙはUSB_DEVICE_ATTACH_AUTO_DISABLE定義を含みます。

```
#define USB_DEVICE_ATTACH_AUTO_DISABLE
```

使用者Cﾌｧｲﾙは以下を含みます。

```
// VBUS監視認可
if (!udc_include_vbus_monitoring()) {
  // 汎用入出力またはその他経由で独自VBUS監視を実装
}
Event_VBUS_present() // VBUS割り込み、または汎用入出力割り込み、またはその他
{
  // USB装置接続
  udc_attach();
}
```

- 電池充電の場合。conf_usb.hﾌｧｲﾙはUSB_DEVICE_ATTACH_AUTO_DISABLE定義を含みます。

```
#define USB_DEVICE_ATTACH_AUTO_DISABLE
```

使用者Cﾌｧｲﾙは以下を含みます。

```
Event VBUS present() // VBUS割り込み、または汎用入出力割込み、または･･･
{
  // 電池充電を認可するが、USBを開始するためにｷｰ押下待機
}
Event Key press()
{
  // 電池充電停止
```

```
    // USB開始
    udc_attach();
}
```

1.2.3. USB装置基本構成設定

1.2.3.1. USB装置制御部 (UDC) - 事前必要条件

全てのUSB装置に対する共通的な事前必要条件

この単位部は完全な割り込み駆動のUSB装置階層に基づき、sleepmgrを支援します。AVR®とAtmel®のSMART ARM®に基づくSAM3/4ﾃﾞﾊﾞｲｽについてはｸﾛｯｸ ｻｰﾋﾞｽが支援されます。SAM D21ﾃﾞﾊﾞｲｽについてはｸﾛｯｸ駆動部が支援されます。

ﾌﾟﾛｼﾞｪｸﾄを正しく構成設定するために以下の手続きが実行されなければなりません。

- ｸﾛｯｸ構成設定を指定してください。
  - XMEGA® USBﾃﾞﾊﾞｲｽは48MHzｸﾛｯｸ入力が必要です。
    XMEGA USBﾃﾞﾊﾞｲｽは12MHzよりも高いCPU周波数が必要です。
    ﾌﾚｰﾑ開始または外部発振器のどちらによってでも自動的に校正される内部RC 48MHzを使うことができます。
  - USB高速(HS)支援のないUC3とSAM3/4ﾃﾞﾊﾞｲｽは48MHzｸﾛｯｸ入力が必要です。
    PLLと外部発振器を使わなければなりません。
  - USB高速(HS)支援付きのUC3とSAM3/4ﾃﾞﾊﾞｲｽは12MHzｸﾛｯｸ入力が必要です。
    外部発振器を使わなければなりません。
  - USBCﾊｰﾄﾞｳｪｱ付きのUC3ﾃﾞﾊﾞｲｽは25MHzよりも高いCPU周波数が必要です。
  - USB高速(HS)支援のないSAM D21ﾃﾞﾊﾞｲｽは48MHzｸﾛｯｸ入力が必要です。
    USBCRMとでDFLLを使うべきです。
- conf_board.hに於いて、USB線を許可するためにCONF_BOARD_USB_PORT定義が追加されなければなりません。(全ての基板に対して必須ではありません。)
- 割り込みを許可してください。
- ｸﾛｯｸ ｻｰﾋﾞｽを初期化してください。

休止管理ｻｰﾋﾞｽの使用は任意選択ですが、消費電力を減らすために推奨されます。

- 休止管理ｻｰﾋﾞｽを初期化してください。
- 応用がｱｲﾄﾞﾙ状態の時に休止動作形態を活性にしてください。

conf_clock.h例

AVRとSAM3/4ﾃﾞﾊﾞｲｽについては以下の初期化ｺｰﾄﾞを追加してください。

```
sysclk_init();
irq_initialize_vectors();
cpu_irq_enable();
board_init();
sleepmgr_init(); // 任意選択
```

SAM D21ﾃﾞﾊﾞｲｽについては初期化ｺｰﾄﾞに以下を追加してください。

```
system_init();
irq_initialize_vectors();
cpu_irq_enable();
sleepmgr_init(); // 任意選択
```

主ｱｲﾄﾞﾙ繰り返しに以下を追加してください。

```
sleepmgr_enter_sleep(); // 任意選択
```

1.2.3.2. USB装置制御部 (UDC) - ｺｰﾄﾞ例

全てのUSB装置に対する共通的なｺｰﾄﾞ例

conf_usb.hの内容

```
#define USB_DEVICE_VENDOR_ID 0x03EB
#define USB_DEVICE_PRODUCT_ID 0xXXXX
#define USB_DEVICE_MAJOR_VERSION 1
#define USB_DEVICE_MINOR_VERSION 0
#define USB_DEVICE_POWER 100
#define USB_DEVICE_ATTR USB_CONFIG_ATTR_BUS_POWERED
```

応用Cファイルに以下を追加してください。

```
void usb_init(void)
{
  udc_start();
}
```

1.2.3.3. USB装置制御部 (UDC) - 作業の流れ

全てのUSB装置に対する共通的な作業の流れ

1. conf_usb.hが利用可能で主USB装置構成設定である以下の構成設定を含むことを確実にしてください。

```
// USB orgによって提供される供給者(Vendor)ID (Atmel 0x03EB)
#define USB_DEVICE_VENDOR_ID 0x03EB // ﾜｰﾄﾞ型
// 製品(Product)ID (usb_atmel.hで言及したAtmelのPID)
#define USB_DEVICE_PRODUCT_ID 0xXXXX // ﾜｰﾄﾞ型
// 装置の主版番号
#define USB_DEVICE_MAJOR_VERSION 1 // ﾊﾞｲﾄ型
// 装置の副版番号
#define USB_DEVICE_MINOR_VERSION 0 // ﾊﾞｲﾄ型
// 装置最大電力 (mA)
#define USB_DEVICE_POWER 100 // 9ﾋﾞｯﾄ型
// 機能を許可するためのUSB属性
#define USB_DEVICE_ATTR USB_CONFIG_ATTR_BUS_POWERED // ﾌﾗｸﾞ
```

2. 階層を許可してUSBを開始するためにUSB装置階層開始関数を呼んでください。

```
udc_start();
```

注: USB On The Go(OTG)ｺﾈｸﾀ(USB IDﾋﾟﾝ)を通して管理される二重役割USB(ﾎｽﾄと装置)の事例では、udc_start()の呼び出しが取り去られてuhc _start()によって置き換えられなければなりません。更なる情報に関して「Atmel AVR4950:ASF - USBﾎｽﾄ階層」応用記述で”二重役割”項を参照してください。

1.2.4. conf_clock.h例

XMEGAのconf_colck.hの内容

```
// 内部RCに基づく構成設定:
// 48MHzのUSBｸﾛｯｸが必要
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_RCOSC
#define CONFIG_OSC_RC32_CAL           48000000UL
#define CONFIG_OSC_AUTOCAL_RC32MHZ_REF_OSC   OSC_ID_USBSOF
// USBで動くのに12MHz以上のCPUｸﾛｯｸが必要(ここでは24MHz)
#define CONFIG_SYSCLK_SOURCE      SYSCLK_SRC_RC32MHZ
#define CONFIG_SYSCLK_PSADIV      SYSCLK_PSADIV_2
#define CONFIG_SYSCLK_PSBCDIV     SYSCLK_PSBCDIV_1_1
```

AT32UC3A0,AT32UC3A1,AT32UC3Bﾃﾞﾊﾞｲｽ(USBB)用のconf_colck.hの内容

```
// 12MHz外部発振器に基づく構成設定
#define CONFIG_PLL1_SOURCE            PLL_SRC_OSC0
#define CONFIG_PLL1_MUL               8
#define CONFIG_PLL1_DIV               2
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV             1 // Fusb = Fsys/(2 ^ USB_div)
```

AT32UC3A3とAT32UC3A4ﾃﾞﾊﾞｲｽ(高速(HS)支援付きUSBB)用のconf_colck.hの内容

```
// 12MHz外部発振器に基づく構成設定
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_OSC0
#define CONFIG_USBCLK_DIV             1 // Fusb = Fsys/(2 ^ USB_div)
```

AT32UC3C,ATUCXXD,ATUCXXL3U,ATUCXXL4Uﾃﾞﾊﾞｲｽ(USBC)用のconf_colck.hの内容

```
// 12MHz外部発振器に基づく構成設定
#define CONFIG_PLL1_SOURCE            PLL_SRC_OSC0
#define CONFIG_PLL1_MUL               8
#define CONFIG_PLL1_DIV               2
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV             1 // Fusb = Fsys/(2 ^ USB_div)
// USBCで動くために25MHz以上のCPUｸﾛｯｸが必要
```

```
#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_PLL1
```

SAM3S,SAM3SD,SAM4Sﾃﾞﾊﾞｲｽ(UPD:USB周辺機能装置)用のconf_colck.hの内容

```
// PLL1 (B)任意選択  (Fpll = (Fclk * PLL_mul) / PLL_div)
#define CONFIG_PLL1_SOURCE            PLL_SRC_MAINCK_XTAL
#define CONFIG_PLL1_MUL               16
#define CONFIG_PLL1_DIV               2
// USBｸﾛｯｸ元任意選択  (Fusb = FpllX / USB_div)
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV             2
```

SAM3Uﾃﾞﾊﾞｲｽ(UPDHS:USB周辺機能装置 高速(HS))用のconf_colck.hの内容

```
// UPLLで固定化されるUSBｸﾛｯｸ元
```

SAM3XとSAM3Aﾃﾞﾊﾞｲｽ(UOTGHS:USB OTG 高速(HS))用のconf_colck.hの内容

```
// UPLLで固定化されるUSBｸﾛｯｸ元
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_UPLL
#define CONFIG_USBCLK_DIV             1
```

SAM D21ﾃﾞﾊﾞｲｽ(USB)用のconf_colck.hの内容

```
// ｼｽﾃﾑ ｸﾛｯｸ ﾊﾞｽ構成設定
#  define CONF_CLOCK_FLASH_WAIT_STATES             2

// DFLLで固定化されるUSBｸﾛｯｸ元
// SYSTEM_CLOCK_SOURCE_DFLL構成設定 - ﾃﾞｼﾞﾀﾙ周波数固定化閉路
#  define CONF_CLOCK_DFLL_ENABLE                   true
#  define CONF_CLOCK_DFLL_LOOP_MODE                SYSTEM_CLOCK_DFLL_LOOP_MODE_USB_RECOVERY
#  define CONF_CLOCK_DFLL_ON_DEMAND                true

// clocks_init走行時にGCLKを構成設定するためにこれをtrueに設定してください。
// falseに設定した場合、clocks_init()でGCLK発振器の何も構成設定されません。
#  define CONF_CLOCK_CONFIGURE_GCLK                true

// GCLK発振器0(主ｸﾛｯｸ)構成設定
#  define CONF_CLOCK_GCLK_0_ENABLE                 true
#  define CONF_CLOCK_GCLK_0_RUN_IN_STANDBY         true
#  define CONF_CLOCK_GCLK_0_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_DFLL
#  define CONF_CLOCK_GCLK_0_PRESCALER              1
#  define CONF_CLOCK_GCLK_0_OUTPUT_ENABLE          false
```

1.3. USB装置の高度な使用事例

- USB速度変更
- USB文字列の使用
- USB遠隔起動機能の使用
- ﾊﾞｽ給電応用勧告
- USB動的通番

1.3.1. USB速度変更

この事例では、USB装置が異なるUSB速度で使われます。

1.3.1.1. 構成設定段階

この事例の実装に先だって、既に”基本的な使用事例”のUDI単位部が適用されていることを確実にしてください。

1.3.1.2. 使用段階

ｺｰﾄﾞ例

conf_usb.hの内容

```
#if // 低速(LS)
#define USB_DEVICE_LOW_SPEED
// #define USB_DEVICE_HS_SUPPORT

#elif // 全速(FS)
```

```
// #define USB_DEVICE_LOW_SPEED
// #define USB_DEVICE_HS_SUPPORT
#elif // 高速(HS)
// #define USB_DEVICE_LOW_SPEED
#define USB_DEVICE_HS_SUPPORT

#endif
```

作業の流れ

1. conf_usb.hが利用可能でUSB装置低速(LS、1.5Mbps)に必要とされる以下のパラメータを含むことを確実にしてください。

```
#define USB_DEVICE_LOW_SPEED
//#define  USB_DEVICE_HS_SUPPORT
```

2. conf_usb.hがUSB装置全速(FS、12Mbps)に必要とされる以下のパラメータを含むことを確実にしてください。

```
//#define USB_DEVICE_LOW_SPEED
//#define  USB_DEVICE_HS_SUPPORT
```

3. conf_usb.hがUSB装置高速(HS、480Mbps)に必要とされる以下のパラメータを含むことを確実にしてください。

```
//#define USB_DEVICE_LOW_SPEED
#define  USB_DEVICE_HS_SUPPORT
```

1.3.2. USB文字列の使用

この事例では、USB装置に通常のUSB文字列が追加されます。

1.3.2.1. 構成設定段階

この事例の実装に先だって、既に”基本的な使用事例”のUDI単位部が適用されていることを確実にしてください。

1.3.2.2. 使用段階

コード例

conf_usb.hの内容

```
#define  USB_DEVICE_MANUFACTURE_NAME      "Manufacture name"
#define  USB_DEVICE_PRODUCT_NAME          "Product name"
#define  USB_DEVICE_SERIAL_NAME           "12...EF"
```

作業の流れ

1. conf_usb.hが利用可能で各種のUSB文字列を許すのに必要とされる以下のパラメータを含むことを確実にしてください。

```
// 製造業者用の静的ASCII名
#define  USB_DEVICE_MANUFACTURE_NAME "Manufacture name"
```

```
// 製品用の静的ASCII名
#define  USB_DEVICE_PRODUCT_NAME "Product name"
```

```
// 通番を許可して設定するための静的ASCII名
#define  USB_DEVICE_SERIAL_NAME "12...EF"
```

1.3.3. USB遠隔起動機能の使用

この事例では、USB遠隔起動機能が許可されます。

1.3.3.1. 構成設定段階

この事例の実装に先だって、既に”基本的な使用事例”のUDI単位部が適用されていることを確実にしてください。

1.3.3.2. 使用段階

コード例

conf_usb.hの内容

```
#define  USB_DEVICE_ATTR ¥
(USB_CONFIG_ATTR_REMOTE_WAKEUP | USB_CONFIG_ATTR_...._POWERED)
#define UDC_REMOTEWAKEUP_ENABLE() my_callback_remotewakeup_enable()
extern void my_callback_remotewakeup_enable(void);
#define UDC_REMOTEWAKEUP_DISABLE() my_callback_remotewakeup_disable()
extern void my_callback_remotewakeup_disable(void);
```

応用Cﾌｧｲﾙに以下を追加してください。

```
void my_callback_remotewakeup_enable(void)
{
// 応用起動事象許可(例えば汎用入出力割り込み許可)
}
void my_callback_remotewakeup_disable(void)
{
// 応用起動事象禁止(例えば汎用入出力割り込み禁止)
}

void my_interrupt_event(void)
{
   udc_remotewakeup();
}
```

作業の流れ

1. conf_usb.hが利用可能で遠隔起動機能を許可するのに必要とされる以下のﾊﾟﾗﾒｰﾀを含むことを確実にしてください。

```
// 遠隔起動機能を認可
#define  USB_DEVICE_ATTR (USB_CONFIG_ATTR_REMOTE_WAKEUP | USB_CONFIG_ATTR_..._POWERED)
```

```
// ﾎｽﾄが遠隔起動機能を許可する時に呼ばれる呼び戻し関数を定義
#define UDC_REMOTEWAKEUP_ENABLE() my_callback_remotewakeup_enable()
extern void my_callback_remotewakeup_enable(void);
```

```
// ﾎｽﾄが遠隔起動機能を禁止する時に呼ばれる呼び戻し関数を定義
#define UDC_REMOTEWAKEUP_DISABLE() my_callback_remotewakeup_disable()
extern void my_callback_remotewakeup_disable(void);
```

2. 遠隔起動送出(USB上方向)

```
udc_remotewakeup();
```

1.3.4. ﾊﾞｽ給電応用推奨

この事例では、USB装置ﾊﾞｽ給電機能が許可されます。この機能は正しい消費電力管理が必要です。

1.3.4.1. 構成設定段階

この事例の実装に先だって、既に”基本的な使用事例”のUDI単位部が適用されていることを確実にしてください。

1.3.4.2. 使用段階

ｺｰﾄﾞ例

conf_usb.hの内容

```
#define  USB_DEVICE_ATTR (USB_CONFIG_ATTR_BUS_POWERED)
#define  UDC_SUSPEND_EVENT()          user_callback_suspend_action()
extern void user_callback_suspend_action(void)
#define  UDC_RESUME_EVENT()           user_callback_resume_action()
extern void user_callback_resume_action(void)
```

応用Cﾌｧｲﾙに以下を追加してください。

```
void user_callback_suspend_action(void)
{
   // 消費電力低減のためにﾊｰﾄﾞｳｪｱ部分を禁止
}
void user_callback_resume_action(void)
{
   // ﾊｰﾄﾞｳｪｱ部分を再許可
}
```

作業の流れ

1. conf_usb.hが利用可能で以下のﾊﾟﾗﾒｰﾀを含むことを確実にしてください。

```
// ﾊﾞｽ給電機能を認可
#define  USB_DEVICE_ATTR (USB_CONFIG_ATTR_BUS_POWERED)
```

```
// ホストがUSB線を一時停止する時に呼ばれる呼び戻し関数を定義
#define UDC_SUSPEND_EVENT() user_callback_suspend_action()
extern void user_callback_suspend_action(void);
```

```
// ホストまたは装置がUSB線を再開する時に呼ばれる呼び戻し関数を定義
#define UDC_RESUME_EVENT() user_callback_resume_action()
extern void user_callback_resume_action(void);
```

2. 一時停止動作形態で消費電力を減らしてください(VBUSで最大2.5mA)。

```
void user_callback_suspend_action(void)
{
   turn_off_components();
}
```

1.3.5. USB動的通番

この事例では、USB通番文字列が動的です。静的通番文字列についてはUSB文字列の使用を参照してください。

1.3.5.1. 構成設定段階

この事例の実装に先だって、既に”基本的な使用事例”のUDI単位部が適用されていることを確実にしてください。

1.3.5.2. 使用段階

ｺｰﾄﾞ例

conf_usb.hの内容

```
#define  USB_DEVICE_SERIAL_NAME
#define  USB_DEVICE_GET_SERIAL_NAME_POINTER serial_number
#define  USB_DEVICE_GET_SERIAL_NAME_LENGTH  12
extern uint8_t serial_number[];
```

応用Cﾌｧｲﾙに以下を追加してください。

```
uint8_t serial_number[USB_DEVICE_GET_SERIAL_NAME_LENGTH];
void init_build_usb_serial_number(void)
{
serial_number[0] = 'A';
serial_number[1] = 'B';
～
serial_number[USB_DEVICE_GET_SERIAL_NAME_LENGTH-1] = 'C';
}
```

作業の流れ

1. conf_usb.hが利用可能で動的なUSB通番文字列を許可するのに必要とされる以下のﾊﾟﾗﾒｰﾀを含むことを確実にしてください。

```
#define  USB_DEVICE_SERIAL_NAME // この空を定義
#define  USB_DEVICE_GET_SERIAL_NAME_POINTER serial_number // 通番配列ﾎﾟｲﾝﾀを与える
#define  USB_DEVICE_GET_SERIAL_NAME_LENGTH  12 // 通番配列の大きさを与える
extern uint8_t serial_number[]; // 外部通番配列宣言
```

2. USB階層を始める前に、通番配列を初期化してください。

```
uint8_t serial_number[USB_DEVICE_GET_SERIAL_NAME_LENGTH];
void init_build_usb_serial_number(void)
{
serial_number[0] = 'A';
serial_number[1] = 'B';
～
serial_number[USB_DEVICE_GET_SERIAL_NAME_LENGTH-1] = 'C';
}
```

2. 通信ｸﾗｽ装置(CDC)用USB装置ｲﾝﾀｰﾌｪｰｽ(UDI)

通信ｸﾗｽ装置(CDC)用USB装置ｲﾝﾀｰﾌｪｰｽ(UDI)はUSB CDC直列装置の構成設定と管理に関するｲﾝﾀｰﾌｪｰｽを提供します。
この資料の概要は以下のとおりです。

- API概要
- USB装置通信ｸﾗｽ装置単位部(UDI CDC)用の即時開始の手引き
- 構成設定ﾌｧｲﾙ例

Atmel®ｿﾌﾄｳｪｱ枠組み(ASF) USB装置階層とUSB装置CDCに関するより多くの詳細については以下の応用記述を参照してください。

- AVR4900：ASF – USB装置階層
- AVR4907：ASF – USB装置CDC応用
- AVR4920：ASF – USB装置階層 – 適合と性能係数
- AVR4921：ASF – USB装置階層 – ASF V1とV2間の違い

2.1. API概要

2.1.1. 構造体定義

2.1.1.1. udi_cdc_comm_desc_t構造体

CDC通信ｸﾗｽ ｲﾝﾀｰﾌｪｰｽに関連する機能とｴﾝﾄﾞﾎﾟｲﾝﾄの記述子を持つｲﾝﾀｰﾌｪｰｽ記述子

表2-1. ﾒﾝﾊﾞ

型	名前	説明
usb_cdc_acm_desc_t	acm	CDC仮想制御模式(ACM)機能記述子
usb_cdc_call_mgmt_desc_t	call_mgmt	CDC呼び出し管理機能記述子
usb_ep_desc_t	ep_notify	通知ｴﾝﾄﾞﾎﾟｲﾝﾄ記述子
usb_cdc_hdr_desc_t	header	CDC先頭部機能記述子
usb_iface_desc_t	iface	標準ｲﾝﾀｰﾌｪｰｽ記述子
usb_cdc_union_desc_t	union_desc	CDC共用体機能記述子

2.1.1.2. udi_cdc_data_desc_t構造体

CDCﾃﾞｰﾀ ｸﾗｽ ｲﾝﾀｰﾌｪｰｽに関連するｴﾝﾄﾞﾎﾟｲﾝﾄ記述子を持つｲﾝﾀｰﾌｪｰｽ記述子

表2-2. ﾒﾝﾊﾞ

型	名前	説明
usb_ep_desc_t	ep_in	ﾃﾞｰﾀINｴﾝﾄﾞﾎﾟｲﾝﾄ記述子
usb_ep_desc_t	ep_out	ﾃﾞｰﾀOUTｴﾝﾄﾞﾎﾟｲﾝﾄ記述子
usb_iface_desc_t	iface	標準ｲﾝﾀｰﾌｪｰｽ記述子

2.1.2. ﾏｸﾛ定義

2.1.2.1. ｲﾝﾀｰﾌｪｰｽ記述子の内容

USB装置には7つまでのCDCｲﾝﾀｰﾌｪｰｽを実装することができます。

UDI_CDC_IAD_STRING_ID_0 ﾏｸﾛ

```
#define UDI_CDC_IAD_STRING_ID_0
```

IADｲﾝﾀｰﾌｪｰｽに関連する文字列なし

UDI_CDC_COMM_STRING_ID_0 ﾏｸﾛ

```
#define UDI_CDC_COMM_STRING_ID_0
```

COMMｲﾝﾀｰﾌｪｰｽに関連する文字列なし

UDI_CDC_DATA_STRING_ID_0 ﾏｸﾛ

```
#define UDI_CDC_DATA_STRING_ID_0
```

DATAｲﾝﾀｰﾌｪｰｽに関連する文字列なし

UDI_CDC_IAD_DESC_0 ﾏｸﾛ

```
#define UDI_CDC_IAD_DESC_0
```

ﾎﾟｰﾄ0用IAD記述子

UDI_CDC_COMM_DESC_0 マクロ

```
#define UDI_CDC_COMM_DESC_0
```

ポート0用COMM記述子

UDI_CDC_DATA_DESC_0_FS マクロ

```
#define UDI_CDC_DATA_DESC_0_FS
```

全速(FS)装置のポート0用DATA記述子

UDI_CDC_DATA_DESC_0_HS マクロ

```
#define UDI_CDC_DATA_DESC_0_HS
```

高速(HS)装置のポート0用DATA記述子

UDI_CDC_IAD_STRING_ID_1 マクロ

```
#define UDI_CDC_IAD_STRING_ID_1
```

IADインターフェースに関連する文字列なし

UDI_CDC_COMM_STRING_ID_1 マクロ

```
#define UDI_CDC_COMM_STRING_ID_1
```

COMMインターフェースに関連する文字列なし

UDI_CDC_DATA_STRING_ID_1 マクロ

```
#define UDI_CDC_DATA_STRING_ID_1
```

UDI_CDC_IAD_DESC_1 マクロ

```
#define UDI_CDC_IAD_DESC_1
```

UDI_CDC_COMM_DESC_1 マクロ

```
#define UDI_CDC_COMM_DESC_1
```

UDI_CDC_DATA_DESC_1_FS マクロ

```
#define UDI_CDC_DATA_DESC_1_FS
```

UDI_CDC_DATA_DESC_1_HS マクロ

```
#define UDI_CDC_DATA_DESC_1_HS
```

UDI_CDC_IAD_STRING_ID_2 マクロ

```
#define UDI_CDC_IAD_STRING_ID_2
```

IADインターフェースに関連する文字列なし

UDI_CDC_COMM_STRING_ID_2 マクロ

```
#define UDI_CDC_COMM_STRING_ID_2
```

COMMインターフェースに関連する文字列なし

UDI_CDC_DATA_STRING_ID_2 マクロ

```
#define UDI_CDC_DATA_STRING_ID_2
```

UDI_CDC_IAD_DESC_2 マクロ

```
#define UDI_CDC_IAD_DESC_2
```

UDI_CDC_COMM_DESC_2 マクロ

```
#define UDI_CDC_COMM_DESC_2
```

UDI_CDC_DATA_DESC_2_FS マクロ

```
#define UDI_CDC_DATA_DESC_2_FS
```

UDI_CDC_DATA_DESC_2_HS マクロ

```
#define UDI_CDC_DATA_DESC_2_HS
```

UDI_CDC_IAD_STRING_ID_3 マクロ

```
#define UDI_CDC_IAD_STRING_ID_3
```

IADインターフェースに関連する文字列なし

UDI_CDC_COMM_STRING_ID_3 マクロ

```
#define UDI_CDC_COMM_STRING_ID_3
```

COMMインターフェースに関連する文字列なし

UDI_CDC_DATA_STRING_ID_3 マクロ

```
#define UDI_CDC_DATA_STRING_ID_3
```

UDI_CDC_IAD_DESC_3 マクロ

```
#define UDI_CDC_IAD_DESC_3
```

UDI_CDC_COMM_DESC_3 マクロ

```
#define UDI_CDC_COMM_DESC_3
```

UDI_CDC_DATA_DESC_3_FS マクロ

```
#define UDI_CDC_DATA_DESC_3_FS
```

UDI_CDC_DATA_DESC_3_HS マクロ

```
#define UDI_CDC_DATA_DESC_3_HS
```

UDI_CDC_IAD_STRING_ID_4 マクロ

```
#define UDI_CDC_IAD_STRING_ID_4
```

IADインターフェースに関連する文字列なし

UDI_CDC_COMM_STRING_ID_4 マクロ

```
#define UDI_CDC_COMM_STRING_ID_4
```

COMMインターフェースに関連する文字列なし

UDI_CDC_DATA_STRING_ID_4 マクロ

```
#define UDI_CDC_DATA_STRING_ID_4
```

UDI_CDC_IAD_DESC_4 マクロ

```
#define UDI_CDC_IAD_DESC_4
```

UDI_CDC_COMM_DESC_4 マクロ

```
#define UDI_CDC_COMM_DESC_4
```

UDI_CDC_DATA_DESC_4_FS マクロ

```
#define UDI_CDC_DATA_DESC_4_FS
```

UDI_CDC_DATA_DESC_4_HS マクロ

```
#define UDI_CDC_DATA_DESC_4_HS
```

UDI_CDC_IAD_STRING_ID_5 マクロ

```
#define UDI_CDC_IAD_STRING_ID_5
```

IADインターフェースに関連する文字列なし

UDI_CDC_COMM_STRING_ID_5 マクロ

```
#define UDI_CDC_COMM_STRING_ID_5
```

COMMインターフェースに関連する文字列なし

UDI_CDC_DATA_STRING_ID_5 マクロ

```
#define UDI_CDC_DATA_STRING_ID_5
```

UDI_CDC_IAD_DESC_5 マクロ

```
#define UDI_CDC_IAD_DESC_5
```

UDI_CDC_COMM_DESC_5 マクロ

```
#define UDI_CDC_COMM_DESC_5
```

UDI_CDC_DATA_DESC_5_FS マクロ

```
#define UDI_CDC_DATA_DESC_5_FS
```

UDI_CDC_DATA_DESC_5_HS マクロ

```
#define UDI_CDC_DATA_DESC_5_HS
```

UDI_CDC_IAD_STRING_ID_6 マクロ

```
#define UDI_CDC_IAD_STRING_ID_6
```

IADインターフェースに関連する文字列なし

UDI_CDC_COMM_STRING_ID_6 マクロ

```
#define UDI_CDC_COMM_STRING_ID_6
```

COMMインターフェースに関連する文字列なし

UDI_CDC_DATA_STRING_ID_6 マクロ

```
#define UDI_CDC_DATA_STRING_ID_6
```

UDI_CDC_IAD_DESC_6 マクロ

```
#define UDI_CDC_IAD_DESC_6
```

UDI_CDC_COMM_DESC_6 マクロ

```
#define UDI_CDC_COMM_DESC_6
```

UDI_CDC_DATA_DESC_6_FS マクロ

```
#define UDI_CDC_DATA_DESC_6_FS
```

UDI_CDC_DATA_DESC_6_HS マクロ

```
#define UDI_CDC_DATA_DESC_6_HS
```

2.1.2.2. UDI_CDC_COMM_DESC マクロ

```
#define UDI_CDC_COMM_DESC(port)
```

全ての速度に対するCDC COMMインターフェース記述子の内容

2.1.2.3. UDI_CDC_COMM_EP_SIZE マクロ

```
#define UDI_CDC_COMM_EP_SIZE
```

全ての速度に対するCDC通信ｴﾝﾄﾞﾎﾟｲﾝﾄの大きさ

2.1.2.4. UDI_CDC_DATA_DESC_COMMON マクロ

```
#define UDI_CDC_DATA_DESC_COMMON
```

CDC DATAインターフェース記述子の内容

2.1.2.5. UDI_CDC_DATA_DESC_FS マクロ

```
#define UDI_CDC_DATA_DESC_FS(port)
```

全速(FS)に対するCDC DATAインターフェース記述子の内容

2.1.2.6. UDI_CDC_DATA_DESC_HS マクロ

```
#define UDI_CDC_DATA_DESC_HS(port)
```

高速(HS)に対するCDC DATAインターフェース記述子の内容

2.1.2.7. UDI_CDC_DATA_EPS_FS_SIZE マクロ

```
#define UDI_CDC_DATA_EPS_FS_SIZE
```

全速(FS)に対するCDCﾃﾞｰﾀ ｴﾝﾄﾞﾎﾟｲﾝﾄの大きさ (8,16,32,64ﾊﾞｲﾄ)

2.1.2.8. UDI_CDC_DATA_EPS_HS_SIZE マクロ

```
#define UDI_CDC_DATA_EPS_HS_SIZE
```

高速(HS)に対するCDCﾃﾞｰﾀ ｴﾝﾄﾞﾎﾟｲﾝﾄの大きさ (512ﾊﾞｲﾄのみ)

2.1.2.9. UDI_CDC_IAD_DESC マクロ

```
#define UDI_CDC_IAD_DESC(port)
```

全ての速度に対するCDC IADインターフェース記述子の内容

2.1.3. 関数定義

2.1.3.1. 単一CDCインターフェース支援を持つ応用のためのインターフェース

udi_cdc_ctrl_signal_dcd()関数

データ搬送波検出(DCD:Data Carrier Detect)信号の状態変化を通知

```
void udi_cdc_ctrl_signal_dcd( bool b_set )
```

表2-3. パラメータ

パラメータ名	データ方向	説明
b_set	[入力]	真ならばDCDは許可、さもなければ禁止

udi_cdc_ctrl_signal_dsr()関数

データ設定準備可(DSR:Data Set Ready)信号の状態変化を通知

```
void udi_cdc_ctrl_signal_dsr( bool b_set )
```

表2-4. パラメータ

パラメータ名	データ方向	説明
b_set	[入力]	真ならばDSRは許可、さもなければ禁止

udi_cdc_signal_framing_error()関数

フレーミング異常を通知

```
void udi_cdc_signal_framing_error( void )
```

udi_cdc_signal_parity_error()関数

パリティ誤りを通知

```
void udi_cdc_signal_parity_error( void )
```

udi_cdc_signal_overrun()関数

オーバーランを通知

```
void udi_cdc_signal_overrun( void )
```

udi_cdc_get_nb_received_data()関数

受信バイト数取得

```
iram_size_t udi_cdc_get_nb_received_data( void )
```

戻り値：利用可能なデータ数

udi_cdc_is_rx_ready()関数

この関数は文字がCDC線で受信されているかを調べます。

```
bool udi_cdc_is_rx_ready(void)
```

戻り値：読まれるべきバイトが準備可の場合に1

udi_cdc_getc()関数

待機してCDC線上の値を取得します。

```
int udi_cdc_getc( void )
```

戻り値：CDC線上で読まれた値

udi_cdc_read_buf()関数

CDC線でのRAM緩衝部を読みます。

```
iram_size_t udi_cdc_read_buf( void * buf, iram_size_t size )
```

表2-5. パラメータ

パラメータ名	データ方向	説明
buf	[出力]	読んだ値
size	[入力]	読んだ値の数

戻り値：残りデータ数

udi_cdc_get_free_tx_buffer()関数

送信緩衝部内の空きﾊﾞｲﾄ数を取得

```
iram_size_t udi_cdc_get_free_tx_buffer( void )
```

戻り値：送信緩衝部内の空きﾊﾞｲﾄ数

udi_cdc_is_tx_ready()関数

この関数は新しい文字の送出が可能かを調べます。ｺﾝﾊﾟｲﾗの.libﾌｧｲﾙからのscanf再指定を支援するためにint型が使われます。

```
bool udi_cdc_is_tx_ready( void )
```

戻り値：新しい文字を送ることができる場合に1

udi_cdc_putc()関数

CDC線上にﾊﾞｲﾄを出力

```
int udi_cdc_putc( int value )
```

ｺﾝﾊﾟｲﾗの.libﾌｧｲﾙからのprintf再指定を支援するためにint型が使われます。

表2-6. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
value	[入力]	出力する値

戻り値：関数が成功裏に終了した場合に1、さもなければ0

udi_cdc_write_buf()関数

CDC線のRAM緩衝部を書きます。

```
iram_size_t udi_cdc_write_buf( const void * buf, iram_size_t size )
```

表2-7. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
buf	[入力]	書く値
size	[入力]	書く値の数

戻り値：残りﾃﾞｰﾀ数

2.1.3.2. 複数CDCｲﾝﾀｰﾌｪｰｽ支援を持つ応用のためのｲﾝﾀｰﾌｪｰｽ

udi_cdc_multi_ctrl_signal_dcd()関数

DCD信号の状態変化を通知

```
void udi_cdc_multi_ctrl_signal_dcd( uint8_t port, bool b_set )
```

表2-8. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
port	[入力]	管理する通信ﾎﾟｰﾄ番号
b_set	[入力]	真ならばDCDは許可、さもなければ禁止

udi_cdc_multi_ctrl_signal_dsr()関数

DSR信号の状態変化を通知

```
void udi_cdc_multi_ctrl_signal_dsr( uint8_t port, bool b_set )
```

表2-9. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
port	[入力]	管理する通信ﾎﾟｰﾄ番号
b_set	[入力]	真ならばDSRは許可、さもなければ禁止

udi_cdc_multi_signal_framing_error()関数

ﾌﾚｰﾐﾝｸﾞ異常を通知

```
void udi_cdc_multi_signal_framing_error( uint8_t port )
```

表2-10. パラメータ

パラメータ名	データ方向	説明
port	[入力]	管理する通信ポート番号

udi_cdc_multi_signal_parity_error()関数

パリティ誤りを通知

```
void udi_cdc_multi_signal_parity_error( uint8_t port )
```

表2-11. パラメータ

パラメータ名	データ方向	説明
port	[入力]	管理する通信ポート番号

udi_cdc_multi_signal_overrun()関数

オーバーランを通知

```
void udi_cdc_multi_signal_overrun( uint8_t port )
```

表2-12. パラメータ

パラメータ名	データ方向	説明
port	[入力]	管理する通信ポート番号

udi_cdc_multi_get_nb_received_data()関数

受信したバイト数を取得

```
iram_size_t udi_cdc_multi_get_nb_received_data( uint8_t port )
```

表2-13. パラメータ

パラメータ名	データ方向	説明
port	[入力]	管理する通信ポート番号

戻り値：利用可能なデータ数

udi_cdc_multi_is_rx_ready()関数

この関数は文字がCDC線上で受信されたかを調べます。

```
bool udi_cdc_multi_is_rx_ready( uint8_t port )
```

表2-14. パラメータ

パラメータ名	データ方向	説明
port	[入力]	管理する通信ポート番号

戻り値：バイトが読まれる準備可の場合に1

udi_cdc_multi_getc()関数

待機してCDC線で値を取得

```
int udi_cdc_multi_getc( uint8_t port )
```

表2-15. パラメータ

パラメータ名	データ方向	説明
port	[入力]	管理する通信ポート番号

戻り値：CDC線で読まれた値

udi_cdc_multi_read_buf()関数

CDC線のRAM緩衝部を読みます。

```
iram_size_t udi_cdc_multi_read_buf( uint8_t port, void * buf, iram_size_t size )
```

表2-16. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
port	[入力]	管理する通信ﾎﾟｰﾄ番号
buf	[出力]	読んだ値
size	[入力]	読んだ値の数

戻り値：残りﾃﾞｰﾀ数

udi_cdc_multi_get_free_tx_buffer()関数

送信緩衝部内の空きﾊﾞｲﾄ数を取得

```
iram_size_t udi_cdc_multi_get_free_tx_buffer( uint8_t port )
```

表2-17. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
port	[入力]	管理する通信ﾎﾟｰﾄ番号

戻り値：送信緩衝部内の空きﾊﾞｲﾄ数

udi_cdc_multi_is_tx_ready()関数

この関数は新しい文字が送出可能かを調べます。

```
bool udi_cdc_multi_is_tx_ready( uint8_t port )
```

表2-18. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
port	[入力]	管理する通信ﾎﾟｰﾄ番号

戻り値：新しい文字を送ることができる場合に1

udi_cdc_multi_putc()関数

CDC線にﾊﾞｲﾄを出力します。ｺﾝﾊﾟｲﾗLIBからのprintf再指定を支援するためにint型が使われます。

```
int udi_cdc_multi_putc( uint8_t port, int value )
```

表2-19. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
port	[入力]	管理する通信ﾎﾟｰﾄ番号
value	[入力]	送出する値

戻り値：関数が成功裏に終了した場合に1、さもなければ0

udi_cdc_multi_write_buf()関数

CDC線のRAM緩衝部に書きます。

```
iram_size_t udi_cdc_multi_write_buf( uint8_t port, const void * buf, iram_size_t size )
```

表2-20. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
port	[入力]	管理する通信ﾎﾟｰﾄ番号
buf	[入力]	書く値
size	[入力]	書く値の数

戻り値：残りﾃﾞｰﾀ数

2.2. USB装置通信ｸﾗｽ装置単位部(UDI CDC)用の即時開始の手引き

これは使用事例の選択に於いて単位部をどう構成設定して使うかを段階的に指示する、USB装置ｲﾝﾀｰﾌｪｰｽCDC単位部(UDI CDC)用の即時開始の手引きです。

使用事例は様々なｺｰﾄﾞの断片を含み、または強調します。構成設定に関する段階でのｺｰﾄﾞの断片は独自初期化関数内に複写することができ、一方使用に関する段階は例えば主応用関数内に複写することができます。

2.2.1. 基本的な使用事例

この使用事例では1つの通信ポートだけで"USB CDC (Single Interface Device)"単位部が使われます。"USB CDC (Composite Device)"単位部の使い方は高度な使用事例で記述されます。

2.2.1.1. 構成設定段階

USB装置のため、共通USB装置構成設定段階に従います。USB装置基本構成設定を参照してください。

2.2.1.2. 使用段階

ｺｰﾄﾞ例

conf_usb.hの内容

```
#define UDI_CDC_ENABLE_EXT(port) my_callback_cdc_enable()
extern bool my_callback_cdc_enable(void);
#define UDI_CDC_DISABLE_EXT(port) my_callback_cdc_disable()
extern void my_callback_cdc_disable(void);
#define  UDI_CDC_LOW_RATE

#define  UDI_CDC_DEFAULT_RATE             115200
#define  UDI_CDC_DEFAULT_STOPBITS         CDC_STOP_BITS_1
#define  UDI_CDC_DEFAULT_PARITY           CDC_PAR_NONE
#define  UDI_CDC_DEFAULT_DATABITS         8
#include "udi_cdc_conf.h"                 // conf_usb.hﾌｧｲﾙの最後で
```

応用Cﾌｧｲﾙに追加してください。

```
static bool my_flag_autorize_cdc_transfert = false;
bool my_callback_cdc_enable(void)
{
  my_flag_autorize_cdc_transfert = true;
  return true;
}

void my_callback_cdc_disable(void)
{
  my_flag_autorize_cdc_transfert = false;
}

void task(void)
{
  if (my_flag_autorize_cdc_transfert) {
  udi_cdc_putc('A');
  udi_cdc_getc();
 }
}
```

作業の流れ

1. conf_usb.hが利用可能でUSB装置CDC構成設定である以下の構成設定を含むことを確実にしてください。

```
#define USB_DEVICE_SERIAL_NAME   "12...EF"                    // CDC用ﾃﾞｨｽｸ通番
```

注：USB通番はCDCｲﾝﾀｰﾌｪｰｽが使われる時に必須です。

```
#define UDI_CDC_ENABLE_EXT(port) my_callback_cdc_enable()
extern bool my_callback_cdc_enable(void);
```

注：装置列挙(USB装置の検出と識別)後、USBﾎｽﾄは装置の構成設定を始めます。装置からのUSB CDCｲﾝﾀｰﾌｪｰｽがﾎｽﾄによって受け入れられると、USBﾎｽﾄはこのｲﾝﾀｰﾌｪｰｽを許可してUDI_CDC_ENABLE_EXT()呼び戻し関数が呼ばれ、trueが返ります。故にこの事象が受け取られた時にCDCｲﾝﾀｰﾌｪｰｽでのﾃﾞｰﾀ転送が承認されます。

```
#define UDI_CDC_DISABLE_EXT(port) my_callback_cdc_disable()
extern void my_callback_cdc_disable(void);
```

注：USB装置が接続解除されるか、またはUSBﾎｽﾄによってﾘｾｯﾄされると、USBｲﾝﾀｰﾌｪｰｽは禁止され、UDI_CDC_DISABLE_EXT()呼び戻し関数が呼ばれます。故に、ﾃﾞｰﾀ転送はCDCｲﾝﾀｰﾌｪｰｽで停止されなければなりません

```
#define  UDI_CDC_LOW_RATE
```

注：CDC緩衝部容量を減らすためにﾎｽﾄへのCDC装置転送が低速(<512000bps)の時にこれを定義してください。

```
#define  UDI_CDC_DEFAULT_RATE              115200
#define  UDI_CDC_DEFAULT_STOPBITS          CDC_STOP_BITS_1
#define  UDI_CDC_DEFAULT_PARITY            CDC_PAR_NONE
#define  UDI_CDC_DEFAULT_DATABITS          8
```

注：始動時の通信ﾎﾟｰﾄ既定構成設定

2. CDC線でﾃﾞｰﾀを送出または待機

```
// 待機してCDC線上の値を取得
int udi_cdc_getc(void);
// CDC線のRAM緩衝部読み込み
iram_size_t udi_cdc_read_buf(int* buf, iram_size_t size);
// CDC線にﾊﾞｲﾄを送出
int udi_cdc_putc(int value);
// CDC線のRAM緩衝部書き込み
iram_size_t udi_cdc_write_buf(const int* buf, iram_size_t size);
```

2.2.2. 高度な使用事例

UDI CDC単位部の複数ｲﾝﾀｰﾌｪｰｽの使用法については以下をご覧ください。

・複合装置でのCDC

UDI CDC単位部のもっと高度な使用については以下をご覧ください。

・USB装置の高度な使用事例

2.2.3. 複合装置でのCDC

USB複合装置は1つよりも多くのUSBｸﾗｽを使うUSB装置です。この使用事例ではUSB複合装置を作成するのに”USB CDC (Composite Device)”単位部を使います。故に、このUSB単位部は”USB HID Mouse (Composite Device)”のような別な”複合装置(Composite Device)”単位部と連携することができます。

また、「AVR4902:ASF - USB複合装置」応用記述を参照することもできます。

2.2.3.1. 構成設定段階

動くためのこの事例の構成設定ｺｰﾄﾞについては基本的な使用事例に従わなければなりません。

2.2.3.2. 使用段階

ｺｰﾄﾞ例

conf_usb.hの内容

```
#define USB_DEVICE_EP_CTRL_SIZE  64
#define USB_DEVICE_NB_INTERFACE (X+2)
#define USB_DEVICE_MAX_EP (X+3)

#define  UDI_CDC_DATA_EP_IN_0          (1 | USB_EP_DIR_IN)  // 送信
#define  UDI_CDC_DATA_EP_OUT_0         (2 | USB_EP_DIR_OUT) // 受信
#define  UDI_CDC_COMM_EP_0             (3 | USB_EP_DIR_IN)  // 通知ｴﾝﾄﾞﾎﾟｲﾝﾄ
#define  UDI_CDC_COMM_IFACE_NUMBER_0   X+0
#define  UDI_CDC_DATA_IFACE_NUMBER_0   X+1

#define UDI_COMPOSITE_DESC_T ¥
   usb_iad_desc_t udi_cdc_iad; ¥
   udi_cdc_comm_desc_t udi_cdc_comm; ¥
   udi_cdc_data_desc_t udi_cdc_data; ¥
   ～
#define UDI_COMPOSITE_DESC_FS ¥
   .udi_cdc_iad               = UDI_CDC_IAD_DESC_0, ¥
   .udi_cdc_comm              = UDI_CDC_COMM_DESC_0, ¥
   .udi_cdc_data              = UDI_CDC_DATA_DESC_0_FS, ¥
   ～
#define UDI_COMPOSITE_DESC_HS ¥
   .udi_cdc_iad               = UDI_CDC_IAD_DESC_0, ¥
   .udi_cdc_comm              = UDI_CDC_COMM_DESC_0, ¥
   .udi_cdc_data              = UDI_CDC_DATA_DESC_0_HS, ¥
   ～
```

```
#define UDI_COMPOSITE_API ¥
   &udi_api_cdc_comm,        ¥
   &udi_api_cdc_data,        ¥
   ～
```

作業の流れ

1. conf_usb.hが利用可能でUSB複合装置構成設定に必要とされる以下のﾊﾟﾗﾒｰﾀを含むことを確実にしてください。

```
// ｴﾝﾄﾞﾎﾟｲﾝﾄ制御容量、これは以下でなければなりません。
// - 全速(FS)装置に対して、8, 16 ,32 または 64 (RAMを節約するために8が推奨されます。)
// - 高速(HS)装置に対して、64
#define USB_DEVICE_EP_CTRL_SIZE   64
// このUSB装置での総ｲﾝﾀｰﾌｪｰｽ数
// CDC用に2を加算
#define USB_DEVICE_NB_INTERFACE  (X+2)
// このUSB装置での総ｴﾝﾄﾞﾎﾟｲﾝﾄ数
// これには各ｲﾝﾀｰﾌｪｰｽに対する各々のｴﾝﾄﾞﾎﾟｲﾝﾄを含めなければなりません。
// CDC用に3を加算
#define USB_DEVICE_MAX_EP  (X+3)
```

2. conf_usb.hが複合装置の記述子を含むことを確実にしてください。

```
// あなたが選んだCDC用のｴﾝﾄﾞﾎﾟｲﾝﾄ番号
// ｴﾝﾄﾞﾎﾟｲﾝﾄ番号は1から始まります。
#define   UDI_CDC_DATA_EP_IN_0              (1 | USB_EP_DIR_IN)   // 送信
#define   UDI_CDC_DATA_EP_OUT_0             (2 | USB_EP_DIR_OUT)  // 受信
#define   UDI_CDC_COMM_EP_0                 (3 | USB_EP_DIR_IN)   // 通知ｴﾝﾄﾞﾎﾟｲﾝﾄ
// 0から始まるｲﾝﾀｰﾌｪｰｽのｲﾝﾀｰﾌｪｰｽ指標
#define   UDI_CDC_COMM_IFACE_NUMBER_0       X+0
#define   UDI_CDC_DATA_IFACE_NUMBER_0       X+1
```

3. conf_usb.hがUSB複合装置構成設定に必要とされる以下のﾊﾟﾗﾒｰﾀを含むことを確実にしてください。

```
// USBｲﾝﾀｰﾌｪｰｽ記述子構造体
#define UDI_COMPOSITE_DESC_T ¥
  ～
  usb_iad_desc_t udi_cdc_iad; ¥
  udi_cdc_comm_desc_t udi_cdc_comm; ¥
  udi_cdc_data_desc_t udi_cdc_data; ¥
  ～
// 全速(FS)用USBｲﾝﾀｰﾌｪｰｽ記述子値
#define UDI_COMPOSITE_DESC_FS ¥
   ～
   .udi_cdc_iad                  = UDI_CDC_IAD_DESC_0, ¥
   .udi_cdc_comm                 = UDI_CDC_COMM_DESC_0, ¥
   .udi_cdc_data                 = UDI_CDC_DATA_DESC_0_FS, ¥
   ～
// 高速(HS)用USBｲﾝﾀｰﾌｪｰｽ記述子値
#define UDI_COMPOSITE_DESC_HS ¥
   ～
   .udi_cdc_iad                  = UDI_CDC_IAD_DESC_0, ¥
   .udi_cdc_comm                 = UDI_CDC_COMM_DESC_0, ¥
   .udi_cdc_data                 = UDI_CDC_DATA_DESC_0_HS, ¥
   ～
// USBｲﾝﾀｰﾌｪｰｽAPI
#define UDI_COMPOSITE_API ¥
   ～
   &udi_api_cdc_comm,        ¥
   &udi_api_cdc_data,        ¥
   ～
```

注：上の4つの一覧で与えられた記述子の順番は全てのｲﾝﾀｰﾌｪｰｽ指標によって定義された順番と同じでなければなりません。ｲﾝﾀｰﾌｪｰｽ指標の順番はUDI_X_IFACE_NUMBER定義を通して定義されます。また、CDCは複合装置用のUSBｲﾝﾀｰﾌｪｰｽ関連記述子(IAD)も必要です。

2.3. 構成設定ファイル例

2.3.1. conf_usb.h

2.3.1.1. 単一UDI CDC

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_USB_H_
#define _CONF_USB_H_

#include "compiler.h"

#warning You must refill the following definitions with a correct values

#define  USB_DEVICE_VENDOR_ID              USB_VID_ATMEL
#define  USB_DEVICE_PRODUCT_ID             USB_PID_ATMEL_ASF_CDC
#define  USB_DEVICE_MAJOR_VERSION          1
#define  USB_DEVICE_MINOR_VERSION          0
#define  USB_DEVICE_POWER                  100 // VBUS線での消費(mA)
#define  USB_DEVICE_ATTR                   ¥
    (USB_CONFIG_ATTR_SELF_POWERED)
// (USB_CONFIG_ATTR_BUS_POWERED)
// (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_SELF_POWERED)
// (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_BUS_POWERED)

// #define  USB_DEVICE_MANUFACTURE_NAME       "Manufacture name"
// #define  USB_DEVICE_PRODUCT_NAME           "Product name"
// #define  USB_DEVICE_SERIAL_NAME            "12...EF"

#if (UC3A3||UC3A4)
#define  USB_DEVICE_HS_SUPPORT
#endif

// #define  UDC_VBUS_EVENT(b_vbus_high)      user_callback_vbus_action(b_vbus_high)
// extern void user_callback_vbus_action(bool b_vbus_high);
// #define  UDC_SOF_EVENT()                  user_callback_sof_action()
// extern void user_callback_sof_action(void);
// #define  UDC_SUSPEND_EVENT()              user_callback_suspend_action()
// extern void user_callback_suspend_action(void);
// #define  UDC_RESUME_EVENT()               user_callback_resume_action()
// extern void user_callback_resume_action(void);
// Mandatory when USB_DEVICE_ATTR authorizes remote wakeup feature
// #define  UDC_REMOTEWAKEUP_ENABLE()        user_callback_remotewakeup_enable()
// extern void user_callback_remotewakeup_enable(void);
// #define  UDC_REMOTEWAKEUP_DISABLE()       user_callback_remotewakeup_disable()
// extern void user_callback_remotewakeup_disable(void);
// 製造業者、製品、通番以外の追加文字列記述子を支援しなければならない時
// #define  UDC_GET_EXTRA_STRING()

#define  UDI_CDC_PORT_NB 1

#define  UDI_CDC_ENABLE_EXT(port)          true
#define  UDI_CDC_DISABLE_EXT(port)
#define  UDI_CDC_RX_NOTIFY(port)
#define  UDI_CDC_TX_EMPTY_NOTIFY(port)
#define  UDI_CDC_SET_CODING_EXT(port,cfg)
#define  UDI_CDC_SET_DTR_EXT(port,set)
#define  UDI_CDC_SET_RTS_EXT(port,set)

  // #define UDI_CDC_ENABLE_EXT(port) my_callback_cdc_enable()
// extern bool my_callback_cdc_enable(void);
```

```
// #define UDI_CDC_DISABLE_EXT(port) my_callback_cdc_disable()
// extern void my_callback_cdc_disable(void);
// #define  UDI_CDC_RX_NOTIFY(port) my_callback_rx_notify(port)
// extern void my_callback_rx_notify(uint8_t port);
// #define  UDI_CDC_TX_EMPTY_NOTIFY(port) my_callback_tx_empty_notify(port)
// extern void my_callback_tx_empty_notify(uint8_t port);
// #define  UDI_CDC_SET_CODING_EXT(port,cfg) my_callback_config(port,cfg)
// extern void my_callback_config(uint8_t port, usb_cdc_line_coding_t * cfg);
// #define  UDI_CDC_SET_DTR_EXT(port,set) my_callback_cdc_set_dtr(port,set)
// extern void my_callback_cdc_set_dtr(uint8_t port, bool b_enable);
// #define  UDI_CDC_SET_RTS_EXT(port,set) my_callback_cdc_set_rts(port,set)
// extern void my_callback_cdc_set_rts(uint8_t port, bool b_enable);

#define  UDI_CDC_LOW_RATE
```

2.3.1.2. 複数UDI CDC（複合）

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_USB_H_
#define _CONF_USB_H_

#include "compiler.h"

#warning You must refill the following definitions with a correct values

#define  USB_DEVICE_VENDOR_ID             USB_VID_ATMEL
#define  USB_DEVICE_PRODUCT_ID            0xFFFF
#define  USB_DEVICE_MAJOR_VERSION         1
#define  USB_DEVICE_MINOR_VERSION         0
#define  USB_DEVICE_POWER                 100 // VBUS線での消費(mA)
#define  USB_DEVICE_ATTR                  ¥
        (USB_CONFIG_ATTR_SELF_POWERED)
// (USB_CONFIG_ATTR_BUS_POWERED)
// (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_SELF_POWERED)
// (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_BUS_POWERED)

// #define  USB_DEVICE_MANUFACTURE_NAME      "Manufacture name"
// #define  USB_DEVICE_PRODUCT_NAME          "Product name"
// #define  USB_DEVICE_SERIAL_NAME           "12...EF" // MSC用ﾃﾞｨｽｸ通番

//#define  USB_DEVICE_LOW_SPEED

#if (UC3A3||UC3A4)
//#define  USB_DEVICE_HS_SUPPORT
#endif

// #define  UDC_VBUS_EVENT(b_vbus_high)      user_callback_vbus_action(b_vbus_high)
// extern void user_callback_vbus_action(bool b_vbus_high);
// #define  UDC_SOF_EVENT()                  user_callback_sof_action()
// extern void user_callback_sof_action(void);
// #define  UDC_SUSPEND_EVENT()              user_callback_suspend_action()
// extern void user_callback_suspend_action(void);
// #define  UDC_RESUME_EVENT()               user_callback_resume_action()
// extern void user_callback_resume_action(void);
// #define  UDC_REMOTEWAKEUP_ENABLE()        user_callback_remotewakeup_enable()
// extern void user_callback_remotewakeup_enable(void);
// #define  UDC_REMOTEWAKEUP_DISABLE()       user_callback_remotewakeup_disable()
// extern void user_callback_remotewakeup_disable(void);
// #define  UDC_GET_EXTRA_STRING()
```

```
#define  USB_DEVICE_EP_CTRL_SIZE       64

#define  USB_DEVICE_NB_INTERFACE       1 // 1またはそれ以上

#define  USB_DEVICE_MAX_EP             1 // 0～インターフェースによって要求された最大エンドポイント

#define  UDI_CDC_PORT_NB 1

#define  UDI_CDC_ENABLE_EXT(port)             true
#define  UDI_CDC_DISABLE_EXT(port)
#define  UDI_CDC_RX_NOTIFY(port)
#define  UDI_CDC_TX_EMPTY_NOTIFY(port)
#define  UDI_CDC_SET_CODING_EXT(port,cfg)
#define  UDI_CDC_SET_DTR_EXT(port,set)
#define  UDI_CDC_SET_RTS_EXT(port,set)
/*
 * #define UDI_CDC_ENABLE_EXT(port) my_callback_cdc_enable()
 * extern bool my_callback_cdc_enable(void);
 * #define UDI_CDC_DISABLE_EXT(port) my_callback_cdc_disable()
 * extern void my_callback_cdc_disable(void);
 * #define  UDI_CDC_RX_NOTIFY(port) my_callback_rx_notify(port)
 * extern void my_callback_rx_notify(uint8_t port);
 * #define  UDI_CDC_TX_EMPTY_NOTIFY(port) my_callback_tx_empty_notify(port)
 * extern void my_callback_tx_empty_notify(uint8_t port);
 * #define  UDI_CDC_SET_CODING_EXT(port,cfg) my_callback_config(port,cfg)
 * extern void my_callback_config(uint8_t port, usb_cdc_line_coding_t * cfg);
 * #define  UDI_CDC_SET_DTR_EXT(port,set) my_callback_cdc_set_dtr(port,set)
 * extern void my_callback_cdc_set_dtr(uint8_t port, bool b_enable);
 * #define  UDI_CDC_SET_RTS_EXT(port,set) my_callback_cdc_set_rts(port,set)
 * extern void my_callback_cdc_set_rts(uint8_t port, bool b_enable);
 */

#define  UDI_CDC_LOW_RATE

#define  UDI_CDC_DEFAULT_RATE             115200
#define  UDI_CDC_DEFAULT_STOPBITS         CDC_STOP_BITS_1
#define  UDI_CDC_DEFAULT_PARITY           CDC_PAR_NONE
#define  UDI_CDC_DEFAULT_DATABITS         8

#define  UDI_CDC_DATA_EP_IN_0          (1 | USB_EP_DIR_IN)  // 送信
#define  UDI_CDC_DATA_EP_OUT_0         (2 | USB_EP_DIR_OUT) // 受信
#define  UDI_CDC_COMM_EP_0             (3 | USB_EP_DIR_IN)  // 通知エンドポイント
#define  UDI_CDC_DATA_EP_IN_2          (4 | USB_EP_DIR_IN)  // 送信
#define  UDI_CDC_DATA_EP_OUT_2         (5 | USB_EP_DIR_OUT) // 受信
#define  UDI_CDC_COMM_EP_2             (6 | USB_EP_DIR_IN)  // 通知エンドポイント
#define  UDI_CDC_DATA_EP_IN_3          (7 | USB_EP_DIR_IN)  // 送信
#define  UDI_CDC_DATA_EP_OUT_3         (8 | USB_EP_DIR_OUT) // 受信
#define  UDI_CDC_COMM_EP_3             (9 | USB_EP_DIR_IN)  // 通知エンドポイント

#define  UDI_CDC_COMM_IFACE_NUMBER_0   0
#define  UDI_CDC_DATA_IFACE_NUMBER_0   1
#define  UDI_CDC_COMM_IFACE_NUMBER_2   2
#define  UDI_CDC_DATA_IFACE_NUMBER_2   3
#define  UDI_CDC_COMM_IFACE_NUMBER_3   4
#define  UDI_CDC_DATA_IFACE_NUMBER_3   5

#define UDI_MSC_GLOBAL_VENDOR_ID            ¥
   'A', 'T', 'M', 'E', 'L', ' ', ' ', ' '
#define UDI_MSC_GLOBAL_PRODUCT_VERSION           ¥
   '1', '.', '0', '0'
```

```
#define  UDI_MSC_ENABLE_EXT()          true
#define  UDI_MSC_DISABLE_EXT()
#define  UDI_MSC_NOTIFY_TRANS_EXT()
/*
 * #define UDI_MSC_ENABLE_EXT() my_callback_msc_enable()
 * extern bool my_callback_msc_enable(void);
 * #define UDI_MSC_DISABLE_EXT() my_callback_msc_disable()
 * extern void my_callback_msc_disable(void);
 * #define  UDI_MSC_NOTIFY_TRANS_EXT()    msc_notify_trans()
 * extern void msc_notify_trans(void) {
 */

#define  UDI_MSC_EP_IN                 (1 | USB_EP_DIR_IN)
#define  UDI_MSC_EP_OUT                (2 | USB_EP_DIR_OUT)

#define  UDI_MSC_IFACE_NUMBER          0

#define  UDI_HID_MOUSE_ENABLE_EXT()       true
#define  UDI_HID_MOUSE_DISABLE_EXT()
// #define UDI_HID_MOUSE_ENABLE_EXT() my_callback_mouse_enable()
// extern bool my_callback_mouse_enable(void);
// #define UDI_HID_MOUSE_DISABLE_EXT() my_callback_mouse_disable()
// extern void my_callback_mouse_disable(void);

#define  UDI_HID_MOUSE_EP_IN           (1 | USB_EP_DIR_IN)

#define  UDI_HID_MOUSE_IFACE_NUMBER    0

#define  UDI_HID_KBD_ENABLE_EXT()       true
#define  UDI_HID_KBD_DISABLE_EXT()
// #define UDI_HID_KBD_ENABLE_EXT() my_callback_keyboard_enable()
// extern bool my_callback_keyboard_enable(void);
// #define UDI_HID_KBD_DISABLE_EXT() my_callback_keyboard_disable()
// extern void my_callback_keyboard_disable(void);
#define  UDI_HID_KBD_CHANGE_LED(value)
// #define  UDI_HID_KBD_CHANGE_LED(value) my_callback_keyboard_led(value)
// extern void my_callback_keyboard_led(uint8_t value)

#define  UDI_HID_KBD_EP_IN           (1 | USB_EP_DIR_IN)

#define  UDI_HID_KBD_IFACE_NUMBER    0
#define  UDI_HID_GENERIC_ENABLE_EXT()       true
#define  UDI_HID_GENERIC_DISABLE_EXT()
#define  UDI_HID_GENERIC_REPORT_OUT(ptr)
#define  UDI_HID_GENERIC_SET_FEATURE(f)
/*
 * #define UDI_HID_GENERIC_ENABLE_EXT() my_callback_generic_enable()
 * extern bool my_callback_generic_enable(void);
 * #define UDI_HID_GENERIC_DISABLE_EXT() my_callback_generic_disable()
 * extern void my_callback_generic_disable(void);
 * #define  UDI_HID_GENERIC_REPORT_OUT(ptr) my_callback_generic_report_out(ptr)
 * extern void my_callback_generic_report_out(uint8_t *report);
 * #define  UDI_HID_GENERIC_SET_FEATURE(f) my_callback_generic_set_feature(f)
 * extern void my_callback_generic_set_feature(uint8_t *report_feature);
 */
#define  UDI_HID_REPORT_IN_SIZE             64
#define  UDI_HID_REPORT_OUT_SIZE            64
#define  UDI_HID_REPORT_FEATURE_SIZE        4
#define  UDI_HID_GENERIC_EP_SIZE            64
```

```
#define  UDI_HID_GENERIC_EP_OUT    (2 | USB_EP_DIR_OUT)
#define  UDI_HID_GENERIC_EP_IN     (1 | USB_EP_DIR_IN)

#define  UDI_HID_GENERIC_IFACE_NUMBER    0

#define  UDI_PHDC_ENABLE_EXT()           true
#define  UDI_PHDC_DISABLE_EXT()

#define  UDI_PHDC_DATAMSG_FORMAT        USB_PHDC_DATAMSG_FORMAT_11073_20601
#define  UDI_PHDC_SPECIALIZATION        {0x2345} // 11073_20601での定義

#define  UDI_PHDC_QOS_OUT               ¥
        (USB_PHDC_QOS_MEDIUM_BETTER|USB_PHDC_QOS_HIGH_BEST)
#define  UDI_PHDC_QOS_IN                ¥
        (USB_PHDC_QOS_LOW_GOOD|USB_PHDC_QOS_MEDIUM_BETTER|USB_PHDC_QOS_MEDIUM_BEST)

#define  UDI_PHDC_METADATA_DESC_BULK_IN    {0x01,0x02,0x03}
#define  UDI_PHDC_METADATA_DESC_BULK_OUT   {0x01,0x02,0x03}
#define  UDI_PHDC_METADATA_DESC_INT_IN     {0x01,0x02,0x03}

#define  UDI_PHDC_EP_BULK_OUT           (1 | USB_EP_DIR_OUT)
#define  UDI_PHDC_EP_BULK_IN            (2 | USB_EP_DIR_IN)
#if ((UDI_PHDC_QOS_IN&USB_PHDC_QOS_LOW_GOOD)==USB_PHDC_QOS_LOW_GOOD)
// UDI_PHDC_QOS_INの場合にだけUSB_PHDC_QOS_LOW_GOODをｲﾝｸﾙｰﾄﾞ
#  define  UDI_PHDC_EP_INTERRUPT_IN       (3 | USB_EP_DIR_IN)
#endif

#define  UDI_PHDC_EP_SIZE_BULK_OUT     32
#define  UDI_PHDC_EP_SIZE_BULK_IN      32
#define  UDI_PHDC_EP_SIZE_INT_IN       8

#define  UDI_PHDC_IFACE_NUMBER          0

#define  UDI_VENDOR_ENABLE_EXT()          true
#define  UDI_VENDOR_DISABLE_EXT()
#define  UDI_VENDOR_SETUP_OUT_RECEIVED()  false
#define  UDI_VENDOR_SETUP_IN_RECEIVED()   false
/*
 * #define UDI_VENDOR_ENABLE_EXT() my_callback_vendor_enable()
 * extern bool my_callback_vendor_enable(void);
 * #define UDI_VENDOR_DISABLE_EXT() my_callback_vendor_disable()
 * extern void my_callback_vendor_disable(void);
 *
 * #define  UDI_VENDOR_SETUP_OUT_RECEIVED()  my_vendor_setup_out_received()
 * extern bool my_vendor_setup_out_received(void);
 * #define  UDI_VENDOR_SETUP_IN_RECEIVED()   my_vendor_setup_in_received()
 * extern bool my_vendor_setup_in_received(void);
 */

#define  UDI_VENDOR_EPS_SIZE_INT_FS    64
#define  UDI_VENDOR_EPS_SIZE_BULK_FS   64
#define  UDI_VENDOR_EPS_SIZE_ISO_FS   256

#define  UDI_VENDOR_EPS_SIZE_INT_HS    64
#define  UDI_VENDOR_EPS_SIZE_BULK_HS  512
#define  UDI_VENDOR_EPS_SIZE_ISO_HS    64

#define  UDI_VENDOR_EP_INTERRUPT_IN  (1 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_INTERRUPT_OUT (2 | USB_EP_DIR_OUT)
#define  UDI_VENDOR_EP_BULK_IN       (3 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_BULK_OUT      (4 | USB_EP_DIR_OUT)
```

```
#define  UDI_VENDOR_EP_ISO_IN       (5 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_ISO_OUT      (6 | USB_EP_DIR_OUT)

#define  UDI_VENDOR_IFACE_NUMBER    0

//... 最終的に他のインターフェース構成設定を追加してください。

  #define UDI_COMPOSITE_DESC_T

  #define UDI_COMPOSITE_DESC_FS

  #define UDI_COMPOSITE_DESC_HS

  #define UDI_COMPOSITE_API

  /* CDC、MSC、HIDマウス インターフェースを持つ装置の例
  #define UDI_COMPOSITE_DESC_T ¥
      usb_iad_desc_t udi_cdc_iad; ¥
      udi_cdc_comm_desc_t udi_cdc_comm; ¥
      udi_cdc_data_desc_t udi_cdc_data; ¥
      udi_msc_desc_t udi_msc; ¥
      udi_hid_mouse_desc_t udi_hid_mouse

  #define UDI_COMPOSITE_DESC_FS ¥
      .udi_cdc_iad               = UDI_CDC_IAD_DESC_0, ¥
      .udi_cdc_comm              = UDI_CDC_COMM_DESC_0, ¥
      .udi_cdc_data              = UDI_CDC_DATA_DESC_0_FS, ¥
      .udi_msc                   = UDI_MSC_DESC_FS, ¥
      .udi_hid_mouse             = UDI_HID_MOUSE_DESC

  #define UDI_COMPOSITE_DESC_HS ¥
      .udi_cdc_iad               = UDI_CDC_IAD_DESC_0, ¥
      .udi_cdc_comm              = UDI_CDC_COMM_DESC_0, ¥
      .udi_cdc_data              = UDI_CDC_DATA_DESC_0_HS, ¥
      .udi_msc                   = UDI_MSC_DESC_HS, ¥
      .udi_hid_mouse             = UDI_HID_MOUSE_DESC

  #define UDI_COMPOSITE_API  ¥
      &udi_api_cdc_comm,        ¥
      &udi_api_cdc_data,        ¥
      &udi_api_msc,             ¥
      &udi_api_hid_mouse
  */

  /* インターフェース用インクルードの例
  #include "udi_msc.h"
  #include "udi_hid_kbd.h"
  #include "udi_hid_mouse.h"
  #include "udi_cdc.h"
  #include "udi_phdc.h"
  #include "udi_vendor.h"
  */
  /* USBによって使われる呼び戻しの宣言
  #include "callback_def.h"
  */

  #endif // _CONF_USB_H_
```

2.3.2. conf_clock.h

2.3.2.1. XMEGA (USB)

```
/*
```

```
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

#define CONFIG_USBCLK_SOURCE      USBCLK_SRC_RCOSC
#define CONFIG_OSC_RC32_CAL       48000000UL

#define CONFIG_OSC_AUTOCAL_RC32MHZ_REF_OSC  OSC_ID_USBSOF

#define CONFIG_SYSCLK_SOURCE      SYSCLK_SRC_RC32MHZ
#define CONFIG_SYSCLK_PSADIV      SYSCLK_PSADIV_2
#define CONFIG_SYSCLK_PSBCDIV     SYSCLK_PSBCDIV_1_1

/*

#define CONFIG_PLL0_SOURCE        PLL_SRC_XOSC
#define CONFIG_PLL0_MUL           6
#define CONFIG_PLL0_DIV           1

#define CONFIG_USBCLK_SOURCE      USBCLK_SRC_PLL

#define CONFIG_SYSCLK_SOURCE      SYSCLK_SRC_PLL
#define CONFIG_SYSCLK_PSADIV      SYSCLK_PSADIV_2
#define CONFIG_SYSCLK_PSBCDIV     SYSCLK_PSBCDIV_1_2
*/

#endif /* CONF_CLOCK_H_INCLUDED */
```

2.3.2.2. AT32UC3A0, AT32UC3A1, AT32UC3Bﾃﾞﾊﾞｲｽ (USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== ｼｽﾃﾑ ｸﾛｯｸ元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RCSYS
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_OSC0
#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_PLL0

// ===== PLL0任意選択
#define CONFIG_PLL0_SOURCE            PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE          PLL_SRC_OSC1
#define CONFIG_PLL0_MUL               8 /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL0_DIV               2 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== PLL1任意選択
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC0
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL1_MUL             8 /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL1_DIV             2 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== ｼｽﾃﾑ ｸﾛｯｸ ﾊﾞｽ分周任意選択
//#define CONFIG_SYSCLK_CPU_DIV         0 /* Fcpu = Fsys/(2 ^ CPU_div) */
//#define CONFIG_SYSCLK_PBA_DIV         0 /* Fpba = Fsys/(2 ^ PBA_div) */
//#define CONFIG_SYSCLK_PBB_DIV         0 /* Fpbb = Fsys/(2 ^ PBB_div) */

// ===== 周辺機能ｸﾛｯｸ管理任意選択
```

```
//#define CONFIG_SYSCLK_INIT_CPUMASK   ((1 << SYSCLK_SYSTIMER) | (1 << SYSCLK_OCD))
//#define CONFIG_SYSCLK_INIT_PBAMASK   (1 << SYSCLK_USART0)
//#define CONFIG_SYSCLK_INIT_PBBMASK   (1 << SYSCLK_HMATRIX)
//#define CONFIG_SYSCLK_INIT_HSBMASK   (1 << SYSCLK_MDMA_HSB)

// ===== USBクロック元任意選択
//#define CONFIG_USBCLK_SOURCE         USBCLK_SRC_OSC0
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_PLL0
//#define CONFIG_USBCLK_SOURCE         USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV             1 /* Fusb = Fsys/(2 ^ USB_div) */

#endif /* CONF_CLOCK_H_INCLUDED */
```

2.3.2.3. AT32UC3A3, AT32UC3A4ﾃﾞﾊﾞｲｽ (高速(HS)支援付USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== システム クロック元任意選択
//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_RCSYS
//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_OSC0
#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_PLL0

// ===== PLL0任意選択
#define CONFIG_PLL0_SOURCE            PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE           PLL_SRC_OSC1
#define CONFIG_PLL0_MUL               11 /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL0_DIV               2 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== PLL1任意選択
//#define CONFIG_PLL1_SOURCE           PLL_SRC_OSC0
//#define CONFIG_PLL1_SOURCE           PLL_SRC_OSC1
//#define CONFIG_PLL1_MUL              8 /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL1_DIV              2 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== システム クロック ﾊﾞｽ分周任意選択
//#define CONFIG_SYSCLK_CPU_DIV          0 /* Fcpu = Fsys/(2 ^ CPU_div) */
//#define CONFIG_SYSCLK_PBA_DIV          0 /* Fpba = Fsys/(2 ^ PBA_div) */
//#define CONFIG_SYSCLK_PBB_DIV          0 /* Fpbb = Fsys/(2 ^ PBB_div) */

// ===== 周辺機能ｸﾛｯｸ管理任意選択
//#define CONFIG_SYSCLK_INIT_CPUMASK   ((1 << SYSCLK_SYSTIMER) | (1 << SYSCLK_OCD))
//#define CONFIG_SYSCLK_INIT_PBAMASK   (1 << SYSCLK_USART0)
//#define CONFIG_SYSCLK_INIT_PBBMASK   (1 << SYSCLK_HMATRIX)
//#define CONFIG_SYSCLK_INIT_HSBMASK   (1 << SYSCLK_MDMA_HSB)

// ===== USBクロック元任意選択
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_OSC0
//#define CONFIG_USBCLK_SOURCE         USBCLK_SRC_PLL0
//#define CONFIG_USBCLK_SOURCE         USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV             1 /* Fusb = Fsys/(2 ^ USB_div) */

#endif /* CONF_CLOCK_H_INCLUDED */
```

2.3.2.4. AT32UC3C, ATUCXXD, ATUCXXL3U, ATUCXXL4Uﾃﾞﾊﾞｲｽ (USBC)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */
```

```
#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== システム クロック元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RCSYS
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_OSC0
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_OSC1
#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_PLL0
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLL1
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RC8M

// ===== PLL0任意選択
#define CONFIG_PLL0_SOURCE            PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL0_SOURCE          PLL_SRC_RC8M
#define CONFIG_PLL0_MUL               3 /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL0_DIV               1 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== PLL1任意選択
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC0
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL1_SOURCE          PLL_SRC_RC8M
//#define CONFIG_PLL1_MUL             3 /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL1_DIV             1 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== システム クロック バス分周任意選択
//#define CONFIG_SYSCLK_CPU_DIV       0 /* Fcpu = Fsys/(2 ^ CPU_div) */
//#define CONFIG_SYSCLK_PBA_DIV       0 /* Fpba = Fsys/(2 ^ PBA_div) */
//#define CONFIG_SYSCLK_PBB_DIV       0 /* Fpbb = Fsys/(2 ^ PBB_div) */
//#define CONFIG_SYSCLK_PBC_DIV       0 /* Fpbc = Fsys/(2 ^ PBC_div) */

// ===== 周辺機能ｸﾛｯｸ管理任意選択
//#define CONFIG_SYSCLK_INIT_CPUMASK  ((1 << SYSCLK_SYSTIMER) | (1 << SYSCLK_OCD))
//#define CONFIG_SYSCLK_INIT_PBAMASK  (1 << SYSCLK_USART0)
//#define CONFIG_SYSCLK_INIT_PBBMASK  (1 << SYSCLK_HMATRIX)
//#define CONFIG_SYSCLK_INIT_HSBMASK  (1 << SYSCLK_MDMA_HSB)

// ===== USBｸﾛｯｸ元任意選択
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_OSC0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_OSC1
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_PLL0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV             1 /* Fusb = Fsys/(2 ^ USB_div) */

#endif /* CONF_CLOCK_H_INCLUDED */
```

2.3.2.5. SAM3S, SAM3SD, SAM4Sﾃﾞﾊﾞｲｽ (UPD:USB周辺機能装置)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== システム クロック(MCK)元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_BYPASS
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_4M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_8M_RC
```

```
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_12M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_BYPASS
#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLLACK
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLLBCK

// ===== システム クロック(MCK)前置分周器任意選択    (Fmck = Fsys / (SYSCLK_PRES))
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_1
#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_2
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_4
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_8
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_16
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_32
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_64
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_3

// ===== PLL0 (A)任意選択    (Fpll = (Fclk * PLL_mul) / PLL_div)
// ここにmulとdivの実効値を使ってください。
#define CONFIG_PLL0_SOURCE          PLL_SRC_MAINCK_XTAL
#define CONFIG_PLL0_MUL             32
#define CONFIG_PLL0_DIV             3

// ===== PLL1 (B)任意選択    (Fpll = (Fclk * PLL_mul) / PLL_div)
// ここにmulとdivの実効値を使ってください。
#define CONFIG_PLL1_SOURCE          PLL_SRC_MAINCK_XTAL
#define CONFIG_PLL1_MUL             16
#define CONFIG_PLL1_DIV             2

// ===== USBクロック元任意選択    (Fusb = FpllX / USB_div)
// ここにdivの実効値を使ってください。
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL0
#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV           2

// ===== 目的対象周波数 (システム クロック)
// - XTAL周波数: 12MHz
// - システム クロック元: PLLA
// - システム クロック前置分周器: 2 (2分周)
// - PLLA供給元: XTAL
// - PLLA出力: XTAL * 32 / 3
// - システム クロックは: 12 * 32 / 3 / 2 = 64MHz
// ===== 目的対象周波数 (USBクロック)
// - USBクロック元: PLLB
// - USBクロック分周器: 2 (2分周)
// - PLLB出力: XTAL * 16 / 2
// - USBクロック: 12 * 16 / 2 / 2 = 48MHz

#endif /* CONF_CLOCK_H_INCLUDED */
```

2.3.2.6. SAM3Uﾃﾞﾊﾞｲｽ (UPDHS:USB周辺機能装置 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== システム クロック(MCK)元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_BYPASS
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_4M_RC
```

```
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_8M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_12M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_BYPASS
#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLLACK
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_UPLLCK

// ===== システム クロック(MCK)前置分周器任意選択    (Fmck = Fsys / (SYSCLK_PRES))
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_1
#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_2
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_4
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_8
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_16
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_32
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_64
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_3

// ===== PLL0 (A)任意選択    (Fpll = (Fclk * PLL_mul) / PLL_div)
// ここにmulとdivの実効値を使ってください。
#define CONFIG_PLL0_SOURCE          PLL_SRC_MAINCK_XTAL
#define CONFIG_PLL0_MUL             16
#define CONFIG_PLL0_DIV             1

// ===== 480MHzで固定化されたUPLL (UTMI)ﾊｰﾄﾞｳｪｱ

// ===== UPLLで固定化されたUSBｸﾛｯｸ元

// ===== 目的対象周波数 (システム クロック)
// - XTAL周波数: 12MHz
// - システム クロック元: PLLA
// - システム クロック前置分周器: 2 (2分周)
// - PLLA供給元: XTAL
// - PLLA出力: XTAL * 16 / 1
// - システム クロックは: 12 * 16 / 1 / 2 = 96MHz
// ===== 目的対象周波数 (USBｸﾛｯｸ)
// - USBｸﾛｯｸ元: UPLL
// - UPLL周波数: 480MHz
// - USBｸﾛｯｸ: 480MHz

#endif /* CONF_CLOCK_H_INCLUDED */
```

2.3.2.7. SAM3X, SAM3Aﾃﾞﾊﾞｲｽ (UOTGHS:USB OTG 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== システム クロック(MCK)元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_BYPASS
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_4M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_8M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_12M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_BYPASS
#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLLACK
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_UPLLCK
```

```
// ===== システム クロック(MCK)前置分周器任意選択    (Fmck = Fsys / (SYSCLK_PRES))
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_1
#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_2
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_4
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_8
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_16
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_32
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_64
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_3

// ===== PLL0 (A)任意選択    (Fpll = (Fclk * PLL_mul) / PLL_div)
// ここにmulとdivの実効値を使ってください。
#define CONFIG_PLL0_SOURCE          PLL_SRC_MAINCK_XTAL
#define CONFIG_PLL0_MUL             14
#define CONFIG_PLL0_DIV             1

// ===== 480MHzで固定化されたUPLL (UTMI)ﾊｰﾄﾞｳｪｱ

// ===== USBｸﾛｯｸ元任意選択    (Fusb = FpllX / USB_div)
// ここにdivの実効値を使ってください。
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL0
#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_UPLL
#define CONFIG_USBCLK_DIV           1

// ===== 目的対象周波数 (システム クロック)
// - XTAL周波数: 12MHz
// - システム クロック元: PLLA
// - システム クロック前置分周器: 2 (2分周)
// - PLLA供給元: XTAL
// - PLLA出力: XTAL * 14 / 1
// - システム クロックは: 12 * 14 / 1 /2 = 84MHz
// ===== 目的対象周波数 (USBｸﾛｯｸ)
// - USBｸﾛｯｸ元: UPLL
// - USBｸﾛｯｸ分周器: 1 (分周なし)
// - UPLL周波数: 480MHz
// - USBｸﾛｯｸ: 480 / 1 = 480MHz

#endif /* CONF_CLOCK_H_INCLUDED */
```

2.3.3. conf_clocks.h

2.3.3.1. SAM D21ﾃﾞﾊﾞｲｽ (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#include <clock.h>

#ifndef CONF_CLOCKS_H_INCLUDED
#  define CONF_CLOCKS_H_INCLUDED

/* システム クロック ﾊﾞｽ構成設定 */
#  define CONF_CLOCK_CPU_CLOCK_FAILURE_DETECT      false
#  define CONF_CLOCK_FLASH_WAIT_STATES             2
#  define CONF_CLOCK_CPU_DIVIDER                   SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBA_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBB_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBC_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1

/* SYSTEM_CLOCK_SOURCE_OSC8M構成設定 - 内部8MHz発振器 */
#  define CONF_CLOCK_OSC8M_PRESCALER               SYSTEM_OSC8M_DIV_1
#  define CONF_CLOCK_OSC8M_ON_DEMAND               true
```

```
#  define CONF_CLOCK_OSC8M_RUN_IN_STANDBY          false

/* SYSTEM_CLOCK_SOURCE_XOSC構成設定 - 外部ｸﾛｯｸ/発振器 */
#  define CONF_CLOCK_XOSC_ENABLE                   false
#  define CONF_CLOCK_XOSC_EXTERNAL_CRYSTAL         SYSTEM_CLOCK_EXTERNAL_CRYSTAL
#  define CONF_CLOCK_XOSC_EXTERNAL_FREQUENCY       12000000UL
#  define CONF_CLOCK_XOSC_STARTUP_TIME             SYSTEM_XOSC_STARTUP_32768
#  define CONF_CLOCK_XOSC_AUTO_GAIN_CONTROL        true
#  define CONF_CLOCK_XOSC_ON_DEMAND                true
#  define CONF_CLOCK_XOSC_RUN_IN_STANDBY           false

/* SYSTEM_CLOCK_SOURCE_XOSC32K構成設定 - 外部32kHzｸﾘｽﾀﾙ/ｸﾛｯｸ発振器 */
#  define CONF_CLOCK_XOSC32K_ENABLE                false
#  define CONF_CLOCK_XOSC32K_EXTERNAL_CRYSTAL      SYSTEM_CLOCK_EXTERNAL_CRYSTAL
#  define CONF_CLOCK_XOSC32K_START?UP_TIME          SYSTEM_XOSC32K_STARTUP_65536
#  define CONF_CLOCK_XOSC32K_AUTO_AMPLITUDE_CONTROL  false
#  define CONF_CLOCK_XOSC32K_ENABLE_1KHZ_OUPUT     false
#  define CONF_CLOCK_XOSC32K_ENABLE_32KHZ_OUTPUT   true
#  define CONF_CLOCK_XOSC32K_ON_DEMAND             true
#  define CONF_CLOCK_XOSC32K_RUN_IN_STANDBY        false

/* SYSTEM_CLOCK_SOURCE_OSC32K構成設定 - 内部32kHz発振器 */
#  define CONF_CLOCK_OSC32K_ENABLE                 false
#  define CONF_CLOCK_OSC32K_STARTUP_TIME           SYSTEM_OSC32K_STARTUP_130
#  define CONF_CLOCK_OSC32K_ENABLE_1KHZ_OUTPUT     true
#  define CONF_CLOCK_OSC32K_ENABLE_32KHZ_OUTPUT    true
#  define CONF_CLOCK_OSC32K_ON_DEMAND              true
#  define CONF_CLOCK_OSC32K_RUN_IN_STANDBY         false

/* SYSTEM_CLOCK_SOURCE_DFLL構成設定 - ﾃﾞｼﾞﾀﾙ周波数固定化閉路 */
#  define CONF_CLOCK_DFLL_ENABLE                   true
#  define CONF_CLOCK_DFLL_LOOP_MODE                SYSTEM_CLOCK_DFLL_LOOP_MODE_USB_RECOVERY
#  define CONF_CLOCK_DFLL_ON_DEMAND                true

/* DFLL開路動作構成設定 */
#  define CONF_CLOCK_DFLL_FINE_VALUE               (512)

/* DFLL閉路動作構成設定 */
#  define CONF_CLOCK_DFLL_SOURCE_GCLK_GENERATOR    GCLK_GENERATOR_1
#  define CONF_CLOCK_DFLL_MULTIPLY_FACTOR          (48000000 / 32768)
#  define CONF_CLOCK_DFLL_QUICK_LOCK               true
#  define CONF_CLOCK_DFLL_TRACK_AFTER_FINE_LOCK    true
#  define CONF_CLOCK_DFLL_KEEP_LOCK_ON_WAKEUP      true
#  define CONF_CLOCK_DFLL_ENABLE_CHILL_CYCLE       true
#  define CONF_CLOCK_DFLL_MAX_COARSE_STEP_SIZE     (0x1f / 4)
#  define CONF_CLOCK_DFLL_MAX_FINE_STEP_SIZE       (0xff / 4)

/* SYSTEM_CLOCK_SOURCE_DPLL構成設定 - ﾃﾞｼﾞﾀﾙ位相固定化閉路 */
#  define CONF_CLOCK_DPLL_ENABLE                   false
#  define CONF_CLOCK_DPLL_ON_DEMAND                true
#  define CONF_CLOCK_DPLL_RUN_IN_STANDBY           false
#  define CONF_CLOCK_DPLL_LOCK_BYPASS              false
#  define CONF_CLOCK_DPLL_WAKE_UP_FAST             false
#  define CONF_CLOCK_DPLL_LOW_POWER_ENABLE         false
#  define CONF_CLOCK_DPLL_LOCK_TIME                SYSTEM_CLOCK_SOURCE_DPLL_LOCK_TIME_DEFAULT
#  define CONF_CLOCK_DPLL_REFERENCE_CLOCK          SYSTEM_CLOCK_SOURCE_DPLL_REFERENCE_CLOCK_XOSC32K
#  define CONF_CLOCK_DPLL_FILTER                   SYSTEM_CLOCK_SOURCE_DPLL_FILTER_DEFAULT
#  define CONF_CLOCK_DPLL_REFERENCE_FREQUENCY      32768
#  define CONF_CLOCK_DPLL_REFERENCE_DIVIDER        1
#  define CONF_CLOCK_DPLL_OUTPUT_FREQUENCY         48000000
```

```
/* DPLL GCLK基準構成設定 */
# define CONF_CLOCK_DPLL_REFERENCE_GCLK_GENERATOR GCLK_GENERATOR_1
/* DPLL GCLK固定化計時器構成設定 */
# define CONF_CLOCK_DPLL_LOCK_GCLK_GENERATOR GCLK_GENERATOR_1

/* clocks_init走行時にGCLKを構成設定するためにこれをtrueに設定してください。
 * falseに設定した場合、GCLK発振器はclocks_init()で何も構成設定されません。*/
#  define CONF_CLOCK_CONFIGURE_GCLK                 true

/* GCLK発振器0構成設定 (主ｸﾛｯｸ) */
#  define CONF_CLOCK_GCLK_0_ENABLE                  true
#  define CONF_CLOCK_GCLK_0_RUN_IN_STANDBY          true
#  define CONF_CLOCK_GCLK_0_CLOCK_SOURCE            SYSTEM_CLOCK_SOURCE_DFLL
#  define CONF_CLOCK_GCLK_0_PRESCALER               1
#  define CONF_CLOCK_GCLK_0_OUTPUT_ENABLE           false

/* GCLK発振器1構成設定 */
#  define CONF_CLOCK_GCLK_1_ENABLE                  false
#  define CONF_CLOCK_GCLK_1_RUN_IN_STANDBY          false
#  define CONF_CLOCK_GCLK_1_CLOCK_SOURCE            SYSTEM_CLOCK_SOURCE_XOSC32K
#  define CONF_CLOCK_GCLK_1_PRESCALER               1
#  define CONF_CLOCK_GCLK_1_OUTPUT_ENABLE           false

/* GCLK発振器2構成設定 (RTC) */
#  define CONF_CLOCK_GCLK_2_ENABLE                  false
#  define CONF_CLOCK_GCLK_2_RUN_IN_STANDBY          false
#  define CONF_CLOCK_GCLK_2_CLOCK_SOURCE            SYSTEM_CLOCK_SOURCE_OSC32K
#  define CONF_CLOCK_GCLK_2_PRESCALER               32
#  define CONF_CLOCK_GCLK_2_OUTPUT_ENABLE           false

/* GCLK発振器3構成設定 */
#  define CONF_CLOCK_GCLK_3_ENABLE                  false
#  define CONF_CLOCK_GCLK_3_RUN_IN_STANDBY          false
#  define CONF_CLOCK_GCLK_3_CLOCK_SOURCE            SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_3_PRESCALER               1
#  define CONF_CLOCK_GCLK_3_OUTPUT_ENABLE           false

/* GCLK発振器4構成設定 */
#  define CONF_CLOCK_GCLK_4_ENABLE                  false
#  define CONF_CLOCK_GCLK_4_RUN_IN_STANDBY          false
#  define CONF_CLOCK_GCLK_4_CLOCK_SOURCE            SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_4_PRESCALER               1
#  define CONF_CLOCK_GCLK_4_OUTPUT_ENABLE           false

/* GCLK発振器5構成設定 */
#  define CONF_CLOCK_GCLK_5_ENABLE                  false
#  define CONF_CLOCK_GCLK_5_RUN_IN_STANDBY          false
#  define CONF_CLOCK_GCLK_5_CLOCK_SOURCE            SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_5_PRESCALER               1
#  define CONF_CLOCK_GCLK_5_OUTPUT_ENABLE           false

/* GCLK発振器6構成設定 */
#  define CONF_CLOCK_GCLK_6_ENABLE                  false
#  define CONF_CLOCK_GCLK_6_RUN_IN_STANDBY          false
#  define CONF_CLOCK_GCLK_6_CLOCK_SOURCE            SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_6_PRESCALER               1
#  define CONF_CLOCK_GCLK_6_OUTPUT_ENABLE           false

/* GCLK発振器7構成設定 */
#  define CONF_CLOCK_GCLK_7_ENABLE                  false
#  define CONF_CLOCK_GCLK_7_RUN_IN_STANDBY          false
```

```
#  define CONF_CLOCK_GCLK_7_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_7_PRESCALER             1
#  define CONF_CLOCK_GCLK_7_OUTPUT_ENABLE         false

/* GCLK発振器8構成設定 */
#  define CONF_CLOCK_GCLK_8_ENABLE                false
#  define CONF_CLOCK_GCLK_8_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_8_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_8_PRESCALER             1
#  define CONF_CLOCK_GCLK_8_OUTPUT_ENABLE         false

#endif /* CONF_CLOCKS_H_INCLUDED */
```

2.3.4. conf_board.h

2.3.4.1. AT32UC3A0, AT32UC3A1, AT32UC3Bﾃﾞﾊﾞｲｽ (USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// UARTﾎﾟｰﾄ許可
#define CONF_BOARD_COM_PORT

#endif /* CONF_BOARD_H_INCLUDED */
```

2.3.4.2. AT32UC3A3, AT32UC3A4ﾃﾞﾊﾞｲｽ (高速(HS)支援付USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// UARTﾎﾟｰﾄ許可
#define CONF_BOARD_COM_PORT

#endif /* CONF_BOARD_H_INCLUDED */
```

2.3.4.3. AT32UC3C, ATUCXXD, ATUCXXL3U, ATUCXXL4Uﾃﾞﾊﾞｲｽ (USBC)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// UARTﾎﾟｰﾄ許可
#define CONF_BOARD_COM_PORT

#endif /* CONF_BOARD_H_INCLUDED */
```

2.3.4.4. SAM3X, SAM3Aﾃﾞﾊﾞｲｽ (UOTGHS:USB OTG 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED
```

```
// UART単位部が使われます。
#define CONF_BOARD_UART_CONSOLE
#define CONF_BOARD_USB_PORT

#endif /* CONF_BOARD_H_INCLUDED */
```

2.3.4.5. SAM D21ﾃﾞﾊﾞｲｽ (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

/* USB VBUS検出許可 */
#define CONF_BOARD_USB_VBUS_DETECT

#endif /* CONF_BOARD_H_INCLUDED */
```

3. 人間インターフェース標準装置(標準HID)用USB装置インターフェース(UDI)

人間インターフェース標準装置(標準HID)用USB装置インターフェース(UDI)はUSB標準HID装置の構成設定と管理に関するインターフェースを提供します。

この資料の概要は以下のとおりです。

- API概要
- USB標準装置単位部(標準UDI)用の即時開始の手引き
- 構成設定ファイル例

Atmel®ソフトウェア枠組み(ASF) USB装置階層とUSB装置標準HIDに関するより多くの詳細については以下の応用記述を参照してください。

- AVR4900：ASF - USB装置階層
- AVR4905：ASF - USB装置標準HID応用
- AVR4920：ASF - USB装置階層 - 適合と性能係数
- AVR4921：ASF - USB装置階層 - ASF V1とV2間の違い

3.1. API概要

3.1.1. 変数と型定義

3.1.1.1. USB装置コア(UDC)とのインターフェース

UDCによって必要とされる構造体

udi_api_hid_generic変数

```
UDC_DESC_STORAGE udi_api_t      udi_api_hid_generic
```

UDC用の標準UDI APIを含む全域構造体

3.1.2. 構造体定義

3.1.2.1. udi_hid_generic_desc_t構造体

標準HID用インターフェース記述子構造体

表3-1. メンバ

型	名前	説明
usb_ep_desc_t	ep_in	標準USBエンドポイント記述子構造体
usb_ep_desc_t	ep_out	標準USBエンドポイント記述子構造体
usb_hid_descriptor_t	hid	HID記述子
usb_iface_desc_t	iface	標準USBインターフェース記述構造体

3.1.2.2. udi_hid_generic_report_desc_t構造体

標準HID用報告記述子

表3-2. メンバ

型	名前	説明
uint8_t	array[]	詳細報告データを置く配列

3.1.3. マクロ定義

3.1.3.1. USBインターフェース記述子

以下の構造体は予め定義されたUSBインターフェース記述子を提供します。これは最終的なUSB記述子を定義するのに使われなければなりません。

UDI_HID_GENERIC_STRING_ID マクロ

```
#define UDI_HID_GENERIC_STRING_ID
```

既定によってこのインターフェースに関連する文字列はなし

UDI_HID_GENERIC_DESC マクロ

```
#define UDI_HID_GENERIC_DESC
```

全ての速度に対する標準HIDインターフェース記述子の内容

3.1.4. 関数定義

3.1.4.1. 人間インターフェース装置(HID)標準クラス用USB装置インターフェース(UDI)

このUSBクラスを使うために上位応用によって使われる共通的なAPI

udi_hid_generic_send_report_in()関数

USBホストへ報告を送るのに使われるルーチン

```
bool udi_hid_generic_send_report_in( uint8_t * data )
```

表3-3. パラメータ

パラメータ名	データ方向	説明
data	[入力]	送る報告のポインタ(大きさ=UDI_HID_REPORT_IN_SIZE)

戻り値： 関数が成功裏に終了した場合に1、さもなければ0

3.2. USB装置標準単位部(UDI標準)用の即時開始の手引き

これは使用事例の選択に於いて単位部をどう構成設定して使うかを段階的に指示する、USB装置標準単位部(UDI標準)用の即時開始の手引きです。

使用事例は様々なコードの断片を含みます。構成設定に関する段階でのコードの断片は独自初期化関数内に複写することができ、一方使用に関する段階は例えば主応用関数内に複写することができます。

3.2.1. 基本的な使用事例

この使用事例では"USB HID generic (Single Interface Device)"単位部が使われます。"USB HID generic (Composite Device)"単位部の使い方は高度な使用事例で記述されます。

3.2.1.1. 構成設定段階

USB装置のため、共通USB装置構成設定段階に従います。USB装置基本構成設定を参照してください。

3.2.1.2. 使用段階

コード例

conf_usb.hの内容

```
#define UDI_HID_generic_ENABLE_EXT() my_callback_generic_enable()
extern bool my_callback_generic_enable(void);
#define UDI_HID_generic_DISABLE_EXT() my_callback_generic_disable()
extern void my_callback_generic_disable(void);
#include "udi_hid_generic_conf.h" // conf_usb.hファイルの最後で
```

応用Cファイルに追加してください。

```
static bool my_flag_autorize_generic_events = false;
bool my_callback_generic_enable(void)
{
   my_flag_autorize_generic_events = true;
   return true;
}
void my_callback_generic_disable(void)
{
   my_flag_autorize_generic_events = false;
}

void my_button_press_event(void)
{
   if (!my_flag_autorize_generic_events) {
      return;
   }
   uint8_t report[] = {0x00,0x01,0x02...};
   udi_hid_generic_send_report_in(report);
}

void my_callback_generic_report_out(uint8_t *report)
{
   if ((report[0] == MY_VALUE_0)
```

```
        (report[1] == MY_VALUE_1)) {
      // 報告は正常
   }
}

void my_callback_generic_set_feature(uint8_t *report_feature)
{
   if ((report_feature[0] == MY_VALUE_0)
       (report_feature[1] == MY_VALUE_1)) {
      // 報告機能は正常
   }
}
```

作業の流れ

1. conf_usb.hが利用可能でUSB装置標準構成設定である以下の構成設定を含むことを確実にしてください。

```
#define UDI_HID_GENERIC_ENABLE_EXT() my_callback_generic_enable()
extern bool my_callback_generic_enable(void);
```

注：装置列挙(USB装置の検出と識別)後、USBホストは装置の構成設定を始めます。装置からのUSB標準インターフェースがホストによって受け入れられると、USBホストはこのインターフェースを許可してUDI_HID_GENERIC_ENABLE_EXT()呼び戻し関数が呼ばれ、trueが返ります。故に、この関数で標準HIDによって使われる感知器を許可することが推奨されます。

```
#define UDI_HID_GENERIC_DISABLE_EXT() my_callback_generic_disable()
extern void my_callback_generic_disable(void);
```

注：USB装置が切断されるか、またはUSBホストによってリセットされると、USBインターフェースは禁止され、UDI_HID_GENERIC_DISABLE_EXT()呼び戻し関数が呼ばれます。故に、この関数で標準HIDインターフェースによって使われる感知器を禁止することが推奨されます。

```
#define  UDI_HID_GENERIC_REPORT_OUT(ptr) my_callback_generic_report_out(ptr)
extern void my_callback_generic_report_out(uint8_t *report);
```

注：OUT報告を受け取るのに使われる呼び戻し

```
#define  UDI_HID_GENERIC_SET_FEATURE(f) my_callback_generic_set_feature(f)
extern void my_callback_generic_set_feature(uint8_t *report_feature);
```

注：SET FEATURE報告を受け取るのに使われる呼び戻し

```
#define  UDI_HID_REPORT_IN_SIZE             64
#define  UDI_HID_REPORT_OUT_SIZE            64
#define  UDI_HID_REPORT_FEATURE_SIZE        4
```

注：報告の大きさは最終的に応用によって定義されます。

```
#define  UDI_HID_GENERIC_EP_SIZE 64
```

注：割り込みｴﾝﾄﾞﾎﾟｲﾝﾄの大きさは最終的に応用によって定義されます。

2. IN報告送出

```
uint8_t report[] = {0x00, 0x01, 0x02...};
udi_hid_generic_send_report_in(report);
```

3.2.2. 高度な使用事例

UDI HID単位部の複数インターフェースの使用については以下をご覧ください。

・複合装置での標準HID

UDI標準HID単位部のもっと高度な使用については以下をご覧ください。

・USB装置の高度な使用事例

3.2.3. 複合装置での標準HID

USB複合装置は1つよりも多くのUSBｸﾗｽを使うUSB装置です。この使用事例ではUSB複合装置を作成するのに”USB HID Generic (Composite Device)”単位部が使われます。故に、このUSB単位部は”USB MSC (Composite Device)”のような別な”複合装置(Composite Device)”単位部と連携することができます。

また、「AVR4902:ASF - USB複合装置」応用記述を参照することもできます。

3.2.3.1. 構成設定段階

動くためのこの事例の構成設定ｺｰﾄﾞについては基本的な使用事例に従わなければなりません。

3.2.3.2. 使用段階

ｺｰﾄﾞ例

conf_usb.hの内容

```
#define USB_DEVICE_EP_CTRL_SIZE  64
#define USB_DEVICE_NB_INTERFACE (X+1)
#define USB_DEVICE_MAX_EP (X+2)
#define UDI_HID_GENERIC_EP_IN    (1 | USB_EP_DIR_IN)
#define UDI_HID_GENERIC_EP_OUT   (2 | USB_EP_DIR_OUT)
#define UDI_HID_GENERIC_IFACE_NUMBER  X
#define UDI_COMPOSITE_DESC_T ¥
   udi_hid_generic_desc_t udi_hid_generic; ¥
   ～
#define UDI_COMPOSITE_DESC_FS ¥
   .udi_hid_generic = UDI_HID_GENERIC_DESC, ¥
   ～
#define UDI_COMPOSITE_DESC_HS ¥
   .udi_hid_generic = UDI_HID_GENERIC_DESC, ¥
   ～
#define UDI_COMPOSITE_API ¥
   &udi_api_hid_generic, ¥
   ～
```

作業の流れ

1. conf_usb.hが利用可能でUSB複合装置構成設定に必要とされる以下のﾊﾟﾗﾒｰﾀを含むことを確実にしてください。

```
// ｴﾝﾄﾞﾎﾟｲﾝﾄ制御容量、これは以下でなければなりません。
// - 低速(LS)装置に対して、8
// - 全速(FS)装置に対して、8, 16 ,32 または 64 (RAMを節約するために8が推奨されます。)
// - 高速(HS)装置に対して、64
#define USB_DEVICE_EP_CTRL_SIZE  64
// このUSB装置での総ｲﾝﾀｰﾌｪｰｽ数
// 標準HID用に1を加算
#define USB_DEVICE_NB_INTERFACE (X+1)
// このUSB装置での総ｴﾝﾄﾞﾎﾟｲﾝﾄ数
// これには各ｲﾝﾀｰﾌｪｰｽに対する各々のｴﾝﾄﾞﾎﾟｲﾝﾄを含めなければなりません。
// 標準HID用に2を加算
#define USB_DEVICE_MAX_EP (X+2)
```

2. conf_usb.hが複合装置の記述子を含むことを確実にしてください。

```
// あなたが選んだ標準HID用のｴﾝﾄﾞﾎﾟｲﾝﾄ番号
// ｴﾝﾄﾞﾎﾟｲﾝﾄ番号は1から始まります。
#define UDI_HID_GENERIC_EP_IN    (1 | USB_EP_DIR_IN)
#define UDI_HID_GENERIC_EP_OUT   (2 | USB_EP_DIR_OUT)
// 0から始まるｲﾝﾀｰﾌｪｰｽのｲﾝﾀｰﾌｪｰｽ指標
#define UDI_HID_GENERIC_IFACE_NUMBER  X
```

3. conf_usb.hがUSB複合装置構成設定に必要とされる以下のﾊﾟﾗﾒｰﾀを含むことを確実にしてください。

```
// USBｲﾝﾀｰﾌｪｰｽ記述子構造体
#define UDI_COMPOSITE_DESC_T ¥
   ～
   udi_hid_generic_desc_t udi_hid_generic; ¥
   ～
// 全速(FS)用USBｲﾝﾀｰﾌｪｰｽ記述子値
#define UDI_COMPOSITE_DESC_FS ¥
   ～
   .udi_hid_generic = UDI_HID_GENERIC_DESC, ¥
   ～
// 高速(HS)用USBｲﾝﾀｰﾌｪｰｽ記述子値
#define UDI_COMPOSITE_DESC_HS ¥
   ～
   .udi_hid_generic = UDI_HID_GENERIC_DESC, ¥
   ～
```

```
// USBインターフェースAPI
#define UDI_COMPOSITE_API ¥
    ～
    &udi_api_hid_generic, ¥
    ～
```

注：上の4つの一覧で与えられた記述子の順番は全てのインターフェース指標によって定義された順番と同じでなければなりません。インターフェース指標の順番はUDI_X_IFACE_NUMBER定義を通して定義されます。

3.3. 構成設定ファイル例

3.3.1. conf_usb.h

3.3.1.1. 単一UDI標準HID

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_USB_H_
#define _CONF_USB_H_

#include "compiler.h"

#warning You must refill the following definitions with a correct values

#define  USB_DEVICE_VENDOR_ID             USB_VID_ATMEL
#define  USB_DEVICE_PRODUCT_ID            USB_PID_ATMEL_ASF_HIDGENERIC
#define  USB_DEVICE_MAJOR_VERSION         1
#define  USB_DEVICE_MINOR_VERSION         0
#define  USB_DEVICE_POWER                 100 // VBUS線での消費(mA)
#define  USB_DEVICE_ATTR                  ¥
    (USB_CONFIG_ATTR_SELF_POWERED)
// (USB_CONFIG_ATTR_BUS_POWERED)
//   (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_SELF_POWERED)
//   (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_BUS_POWERED)

// #define  USB_DEVICE_MANUFACTURE_NAME      "Manufacture name"
// #define  USB_DEVICE_PRODUCT_NAME          "Product name"
// #define  USB_DEVICE_SERIAL_NAME           "12...EF"

//#define  USB_DEVICE_LOW_SPEED

#if (UC3A3||UC3A4)
//#define  USB_DEVICE_HS_SUPPORT
#endif

// #define  UDC_VBUS_EVENT(b_vbus_high)      user_callback_vbus_action(b_vbus_high)
// extern void user_callback_vbus_action(bool b_vbus_high);
// #define  UDC_SOF_EVENT()                  user_callback_sof_action()
// extern void user_callback_sof_action(void);
// #define  UDC_SUSPEND_EVENT()              user_callback_suspend_action()
// extern void user_callback_suspend_action(void);
// #define  UDC_RESUME_EVENT()               user_callback_resume_action()
// extern void user_callback_resume_action(void);
// #define  UDC_REMOTEWAKEUP_ENABLE()        user_callback_remotewakeup_enable()
// extern void user_callback_remotewakeup_enable(void);
// #define  UDC_REMOTEWAKEUP_DISABLE()       user_callback_remotewakeup_disable()
// extern void user_callback_remotewakeup_disable(void);
// #define  UDC_GET_EXTRA_STRING()

#define  UDI_HID_GENERIC_ENABLE_EXT()       true
#define  UDI_HID_GENERIC_DISABLE_EXT()
#define  UDI_HID_GENERIC_REPORT_OUT(ptr)
```

```
#define  UDI_HID_GENERIC_SET_FEATURE(f)
/*
 * #define UDI_HID_GENERIC_ENABLE_EXT() my_callback_generic_enable()
 * extern bool my_callback_generic_enable(void);
 * #define UDI_HID_GENERIC_DISABLE_EXT() my_callback_generic_disable()
 * extern void my_callback_generic_disable(void);
 * #define  UDI_HID_GENERIC_REPORT_OUT(ptr) my_callback_generic_report_out(ptr)
 * extern void my_callback_generic_report_out(uint8_t *report);
 * #define  UDI_HID_GENERIC_SET_FEATURE(f) my_callback_generic_set_feature(f)
 * extern void my_callback_generic_set_feature(uint8_t *report_feature);
 */

#define  UDI_HID_REPORT_IN_SIZE             64
#define  UDI_HID_REPORT_OUT_SIZE            64
#define  UDI_HID_REPORT_FEATURE_SIZE        4

#define  UDI_HID_GENERIC_EP_SIZE            64

#include "udi_hid_generic_conf.h"

#endif // _CONF_USB_H_
```

3.3.1.2. 複数UDI標準HID（複合）

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_USB_H_
#define _CONF_USB_H_

#include "compiler.h"

#warning You must refill the following definitions with a correct values

#define  USB_DEVICE_VENDOR_ID             USB_VID_ATMEL
#define  USB_DEVICE_PRODUCT_ID            0xFFFF
#define  USB_DEVICE_MAJOR_VERSION         1
#define  USB_DEVICE_MINOR_VERSION         0
#define  USB_DEVICE_POWER                 100 // VBUS線での消費(mA)
#define  USB_DEVICE_ATTR                  ¥
        (USB_CONFIG_ATTR_SELF_POWERED)
// (USB_CONFIG_ATTR_BUS_POWERED)
// (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_SELF_POWERED)
/ (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_BUS_POWERED)

// #define  USB_DEVICE_MANUFACTURE_NAME      "Manufacture name"
// #define  USB_DEVICE_PRODUCT_NAME          "Product name"
// #define  USB_DEVICE_SERIAL_NAME           "12...EF" // MSC用ﾃﾞｨｽｸ通番

//#define  USB_DEVICE_LOW_SPEED

#if (UC3A3||UC3A4)
//#define  USB_DEVICE_HS_SUPPORT
#endif

// #define  UDC_VBUS_EVENT(b_vbus_high)      user_callback_vbus_action(b_vbus_high)
// extern void user_callback_vbus_action(bool b_vbus_high);
// #define  UDC_SOF_EVENT()                  user_callback_sof_action()
// extern void user_callback_sof_action(void);
/ #define  UDC_SUSPEND_EVENT()              user_callback_suspend_action()
// extern void user_callback_suspend_action(void);
```

```
// #define  UDC_RESUME_EVENT()               user_callback_resume_action()
// extern void user_callback_resume_action(void);
// #define  UDC_REMOTEWAKEUP_ENABLE()        user_callback_remotewakeup_enable()
// extern void user_callback_remotewakeup_enable(void);
// #define  UDC_REMOTEWAKEUP_DISABLE()       user_callback_remotewakeup_disable()
// extern void user_callback_remotewakeup_disable(void);
// #define  UDC_GET_EXTRA_STRING()

#define  USB_DEVICE_EP_CTRL_SIZE       64

#define  USB_DEVICE_NB_INTERFACE       1 // 1またはそれ以上

#define  USB_DEVICE_MAX_EP             1 // 0～インターフェースによって要求された最大エンドポイント

#define  UDI_CDC_PORT_NB 1

#define  UDI_CDC_ENABLE_EXT(port)              true
#define  UDI_CDC_DISABLE_EXT(port)
#define  UDI_CDC_RX_NOTIFY(port)
#define  UDI_CDC_TX_EMPTY_NOTIFY(port)
#define  UDI_CDC_SET_CODING_EXT(port,cfg)
#define  UDI_CDC_SET_DTR_EXT(port,set)
#define  UDI_CDC_SET_RTS_EXT(port,set)
/*
 * #define UDI_CDC_ENABLE_EXT(port) my_callback_cdc_enable()
 * extern bool my_callback_cdc_enable(void);
 * #define UDI_CDC_DISABLE_EXT(port) my_callback_cdc_disable()
 * extern void my_callback_cdc_disable(void);
 * #define  UDI_CDC_RX_NOTIFY(port) my_callback_rx_notify(port)
 * extern void my_callback_rx_notify(uint8_t port);
 * #define  UDI_CDC_TX_EMPTY_NOTIFY(port) my_callback_tx_empty_notify(port)
 * extern void my_callback_tx_empty_notify(uint8_t port);
 * #define  UDI_CDC_SET_CODING_EXT(port,cfg) my_callback_config(port,cfg)
 * extern void my_callback_config(uint8_t port, usb_cdc_line_coding_t * cfg);
 * #define  UDI_CDC_SET_DTR_EXT(port,set) my_callback_cdc_set_dtr(port,set)
 * extern void my_callback_cdc_set_dtr(uint8_t port, bool b_enable);
 * #define  UDI_CDC_SET_RTS_EXT(port,set) my_callback_cdc_set_rts(port,set)
 * extern void my_callback_cdc_set_rts(uint8_t port, bool b_enable);
 */

#define  UDI_CDC_LOW_RATE

#define  UDI_CDC_DEFAULT_RATE             115200
#define  UDI_CDC_DEFAULT_STOPBITS         CDC_STOP_BITS_1
#define  UDI_CDC_DEFAULT_PARITY           CDC_PAR_NONE
#define  UDI_CDC_DEFAULT_DATABITS         8

#define  UDI_CDC_DATA_EP_IN_0          (1 | USB_EP_DIR_IN)  // 送信
#define  UDI_CDC_DATA_EP_OUT_0         (2 | USB_EP_DIR_OUT) // 受信
#define  UDI_CDC_COMM_EP_0             (3 | USB_EP_DIR_IN)  // 通知エンドポイント
#define  UDI_CDC_DATA_EP_IN_2          (4 | USB_EP_DIR_IN)  // 送信
#define  UDI_CDC_DATA_EP_OUT_2         (5 | USB_EP_DIR_OUT) // 受信
#define  UDI_CDC_COMM_EP_2             (6 | USB_EP_DIR_IN)  // 通知エンドポイント
#define  UDI_CDC_DATA_EP_IN_3          (7 | USB_EP_DIR_IN)  // 送信
#define  UDI_CDC_DATA_EP_OUT_3         (8 | USB_EP_DIR_OUT) // 受信
#define  UDI_CDC_COMM_EP_3             (9 | USB_EP_DIR_IN)  // 通知エンドポイント

#define  UDI_CDC_COMM_IFACE_NUMBER_0   0
#define  UDI_CDC_DATA_IFACE_NUMBER_0   1
#define  UDI_CDC_COMM_IFACE_NUMBER_2   2
#define  UDI_CDC_DATA_IFACE_NUMBER_2   3
```

```
#define  UDI_CDC_COMM_IFACE_NUMBER_3   4
#define  UDI_CDC_DATA_IFACE_NUMBER_3   5

#define UDI_MSC_GLOBAL_VENDOR_ID           ¥
   'A', 'T', 'M', 'E', 'L', ' ', ' ', ' '
#define UDI_MSC_GLOBAL_PRODUCT_VERSION           ¥
   '1', '.', '0', '0'

#define  UDI_MSC_ENABLE_EXT()          true
#define  UDI_MSC_DISABLE_EXT()
#define  UDI_MSC_NOTIFY_TRANS_EXT()
/*
 * #define UDI_MSC_ENABLE_EXT() my_callback_msc_enable()
 * extern bool my_callback_msc_enable(void);
 * #define UDI_MSC_DISABLE_EXT() my_callback_msc_disable()
 * extern void my_callback_msc_disable(void);
 * #define  UDI_MSC_NOTIFY_TRANS_EXT()    msc_notify_trans()
 * extern void msc_notify_trans(void) {
 */

#define  UDI_MSC_EP_IN                 (1 | USB_EP_DIR_IN)
#define  UDI_MSC_EP_OUT                (2 | USB_EP_DIR_OUT)

#define  UDI_MSC_IFACE_NUMBER          0

#define  UDI_HID_MOUSE_ENABLE_EXT()       true
#define  UDI_HID_MOUSE_DISABLE_EXT()
// #define UDI_HID_MOUSE_ENABLE_EXT() my_callback_mouse_enable()
// extern bool my_callback_mouse_enable(void);
// #define UDI_HID_MOUSE_DISABLE_EXT() my_callback_mouse_disable()
// extern void my_callback_mouse_disable(void);

#define  UDI_HID_MOUSE_EP_IN           (1 | USB_EP_DIR_IN)

#define  UDI_HID_MOUSE_IFACE_NUMBER    0

#define  UDI_HID_KBD_ENABLE_EXT()       true
#define  UDI_HID_KBD_DISABLE_EXT()
// #define UDI_HID_KBD_ENABLE_EXT() my_callback_keyboard_enable()
// extern bool my_callback_keyboard_enable(void);
// #define UDI_HID_KBD_DISABLE_EXT() my_callback_keyboard_disable()
// extern void my_callback_keyboard_disable(void);
#define  UDI_HID_KBD_CHANGE_LED(value)
// #define  UDI_HID_KBD_CHANGE_LED(value) my_callback_keyboard_led(value)
// extern void my_callback_keyboard_led(uint8_t value)

#define  UDI_HID_KBD_EP_IN           (1 | USB_EP_DIR_IN)

#define  UDI_HID_KBD_IFACE_NUMBER    0

#define  UDI_HID_GENERIC_ENABLE_EXT()       true
#define  UDI_HID_GENERIC_DISABLE_EXT()
#define  UDI_HID_GENERIC_REPORT_OUT(ptr)
#define  UDI_HID_GENERIC_SET_FEATURE(f)
/*
 * #define UDI_HID_GENERIC_ENABLE_EXT() my_callback_generic_enable()
 * extern bool my_callback_generic_enable(void);
 * #define UDI_HID_GENERIC_DISABLE_EXT() my_callback_generic_disable()
 * extern void my_callback_generic_disable(void);
 * #define  UDI_HID_GENERIC_REPORT_OUT(ptr) my_callback_generic_report_out(ptr)
 * extern void my_callback_generic_report_out(uint8_t *report);
```

```
 * #define  UDI_HID_GENERIC_SET_FEATURE(f) my_callback_generic_set_feature(f)
 * extern void my_callback_generic_set_feature(uint8_t *report_feature);
 */
#define  UDI_HID_REPORT_IN_SIZE             64
#define  UDI_HID_REPORT_OUT_SIZE            64
#define  UDI_HID_REPORT_FEATURE_SIZE        4
#define  UDI_HID_GENERIC_EP_SIZE            64

#define  UDI_HID_GENERIC_EP_OUT    (2 | USB_EP_DIR_OUT)
#define  UDI_HID_GENERIC_EP_IN     (1 | USB_EP_DIR_IN)

#define  UDI_HID_GENERIC_IFACE_NUMBER    0

#define  UDI_PHDC_ENABLE_EXT()           true
#define  UDI_PHDC_DISABLE_EXT()

#define  UDI_PHDC_DATAMSG_FORMAT        USB_PHDC_DATAMSG_FORMAT_11073_20601
#define  UDI_PHDC_SPECIALIZATION        {0x2345} // 11073_20601での定義

#define  UDI_PHDC_QOS_OUT               ¥
        (USB_PHDC_QOS_MEDIUM_BETTER|USB_PHDC_QOS_HIGH_BEST)
#define  UDI_PHDC_QOS_IN                ¥
        (USB_PHDC_QOS_LOW_GOOD|USB_PHDC_QOS_MEDIUM_BETTER|USB_PHDC_QOS_MEDIUM_BEST)

#define  UDI_PHDC_METADATA_DESC_BULK_IN    {0x01,0x02,0x03}
#define  UDI_PHDC_METADATA_DESC_BULK_OUT   {0x01,0x02,0x03}
#define  UDI_PHDC_METADATA_DESC_INT_IN     {0x01,0x02,0x03}

#define  UDI_PHDC_EP_BULK_OUT           (1 | USB_EP_DIR_OUT)
#define  UDI_PHDC_EP_BULK_IN            (2 | USB_EP_DIR_IN)
#if ((UDI_PHDC_QOS_IN&USB_PHDC_QOS_LOW_GOOD)==USB_PHDC_QOS_LOW_GOOD)
// UDI_PHDC_QOS_INの場合にだけUSB_PHDC_QOS_LOW_GOODをｲﾝｸﾙｰﾄﾞ
#  define  UDI_PHDC_EP_INTERRUPT_IN       (3 | USB_EP_DIR_IN)
#endif

#define  UDI_PHDC_EP_SIZE_BULK_OUT      32
#define  UDI_PHDC_EP_SIZE_BULK_IN       32
#define  UDI_PHDC_EP_SIZE_INT_IN        8

#define  UDI_PHDC_IFACE_NUMBER           0
#define  UDI_VENDOR_ENABLE_EXT()           true
#define  UDI_VENDOR_DISABLE_EXT()
#define  UDI_VENDOR_SETUP_OUT_RECEIVED()  false
#define  UDI_VENDOR_SETUP_IN_RECEIVED()   false
/*
 * #define UDI_VENDOR_ENABLE_EXT() my_callback_vendor_enable()
 * extern bool my_callback_vendor_enable(void);
 * #define UDI_VENDOR_DISABLE_EXT() my_callback_vendor_disable()
 * extern void my_callback_vendor_disable(void);
 *
 * #define  UDI_VENDOR_SETUP_OUT_RECEIVED()  my_vendor_setup_out_received()
 * extern bool my_vendor_setup_out_received(void);
 * #define  UDI_VENDOR_SETUP_IN_RECEIVED()   my_vendor_setup_in_received()
 * extern bool my_vendor_setup_in_received(void);
 */

#define  UDI_VENDOR_EPS_SIZE_INT_FS    64
#define  UDI_VENDOR_EPS_SIZE_BULK_FS   64
#define  UDI_VENDOR_EPS_SIZE_ISO_FS   256

#define  UDI_VENDOR_EPS_SIZE_INT_HS    64
```

```
#define  UDI_VENDOR_EPS_SIZE_BULK_HS  512
#define  UDI_VENDOR_EPS_SIZE_ISO_HS    64

#define  UDI_VENDOR_EP_INTERRUPT_IN  (1 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_INTERRUPT_OUT (2 | USB_EP_DIR_OUT)
#define  UDI_VENDOR_EP_BULK_IN       (3 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_BULK_OUT      (4 | USB_EP_DIR_OUT)
#define  UDI_VENDOR_EP_ISO_IN        (5 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_ISO_OUT       (6 | USB_EP_DIR_OUT)

#define  UDI_VENDOR_IFACE_NUMBER     0

//... 最終的に他のｲﾝﾀｰﾌｪｰｽ構成設定を追加してください。

#define UDI_COMPOSITE_DESC_T

#define UDI_COMPOSITE_DESC_FS

#define UDI_COMPOSITE_DESC_HS

#define UDI_COMPOSITE_API

/* CDC、MSC、HIDﾏｳｽ ｲﾝﾀｰﾌｪｰｽを持つ装置の例
#define UDI_COMPOSITE_DESC_T ¥
    usb_iad_desc_t udi_cdc_iad; ¥
    udi_cdc_comm_desc_t udi_cdc_comm; ¥
    udi_cdc_data_desc_t udi_cdc_data; ¥
    udi_msc_desc_t udi_msc; ¥
    udi_hid_mouse_desc_t udi_hid_mouse

#define UDI_COMPOSITE_DESC_FS ¥
    .udi_cdc_iad               = UDI_CDC_IAD_DESC_0, ¥
    .udi_cdc_comm              = UDI_CDC_COMM_DESC_0, ¥
    .udi_cdc_data              = UDI_CDC_DATA_DESC_0_FS, ¥
    .udi_msc                   = UDI_MSC_DESC_FS, ¥
    .udi_hid_mouse             = UDI_HID_MOUSE_DESC

#define UDI_COMPOSITE_DESC_HS ¥
    .udi_cdc_iad               = UDI_CDC_IAD_DESC_0, ¥
    .udi_cdc_comm              = UDI_CDC_COMM_DESC_0, ¥
    .udi_cdc_data              = UDI_CDC_DATA_DESC_0_HS, ¥
    .udi_msc                   = UDI_MSC_DESC_HS, ¥
    .udi_hid_mouse             = UDI_HID_MOUSE_DESC

#define UDI_COMPOSITE_API  ¥
    &udi_api_cdc_comm,       ¥
    &udi_api_cdc_data,       ¥
    &udi_api_msc,            ¥
    &udi_api_hid_mouse
*/

/* ｲﾝﾀｰﾌｪｰｽ用ｲﾝｸﾙｰﾄﾞの例
#include "udi_msc.h"
#include "udi_hid_kbd.h"
#include "udi_hid_mouse.h"
#include "udi_cdc.h"
#include "udi_phdc.h"
#include "udi_vendor.h"
*/
/* USBによって使われる呼び戻しの宣言
#include "callback_def.h"
```

```
*/

#endif // _CONF_USB_H_
```

3.3.2. conf_clock.h

3.3.2.1. AT32UC3C, ATUCXXD, ATUCXXL3U, ATUCXXL4Uﾃﾞﾊﾞｲｽ (USBC)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== ｼｽﾃﾑ ｸﾛｯｸ元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RCSYS
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_OSC0
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_OSC1
#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_PLL0
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLL1
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RC8M

// ===== PLL0任意選択
#define CONFIG_PLL0_SOURCE            PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL0_SOURCE          PLL_SRC_RC8M
#define CONFIG_PLL0_MUL               3 /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL0_DIV               1 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== PLL1任意選択
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC0
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL1_SOURCE          PLL_SRC_RC8M
//#define CONFIG_PLL1_MUL             3 /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL1_DIV             1 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== ｼｽﾃﾑ ｸﾛｯｸ ﾊﾞｽ分周任意選択
//#define CONFIG_SYSCLK_CPU_DIV       0 /* Fcpu = Fsys/(2 ^ CPU_div) */
//#define CONFIG_SYSCLK_PBA_DIV       0 /* Fpba = Fsys/(2 ^ PBA_div) */
//#define CONFIG_SYSCLK_PBB_DIV       0 /* Fpbb = Fsys/(2 ^ PBB_div) */
//#define CONFIG_SYSCLK_PBC_DIV       0 /* Fpbc = Fsys/(2 ^ PBC_div) */

// ===== 周辺機能ｸﾛｯｸ管理任意選択
//#define CONFIG_SYSCLK_INIT_CPUMASK  ((1 << SYSCLK_SYSTIMER) | (1 << SYSCLK_OCD))
//#define CONFIG_SYSCLK_INIT_PBAMASK  (1 << SYSCLK_USART0)
//#define CONFIG_SYSCLK_INIT_PBBMASK  (1 << SYSCLK_HMATRIX)
//#define CONFIG_SYSCLK_INIT_HSBMASK  (1 << SYSCLK_MDMA_HSB)

// ===== USBｸﾛｯｸ元任意選択
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_OSC0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_OSC1
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_PLL0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV             1 /* Fusb = Fsys/(2 ^ USB_div) */

#endif /* CONF_CLOCK_H_INCLUDED */
```

3.3.2.2. SAM3X, SAM3Aﾃﾞﾊﾞｲｽ (UOTGHS:USB OTG 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */
```

```
#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== システム クロック(MCK)元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_BYPASS
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_4M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_8M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_12M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_BYPASS
#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLLACK
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_UPLLCK

// ===== システム クロック(MCK)前置分周器任意選択    (Fmck = Fsys / (SYSCLK_PRES))
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_1
#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_2
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_4
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_8
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_16
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_32
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_64
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_3

// ===== PLL0 (A)任意選択    (Fpll = (Fclk * PLL_mul) / PLL_div)
// ここにmulとdivの実効値を使ってください。
#define CONFIG_PLL0_SOURCE          PLL_SRC_MAINCK_XTAL
#define CONFIG_PLL0_MUL             14
#define CONFIG_PLL0_DIV             1

// ===== 480MHzで固定化されたUPLL (UTMI)ﾊｰﾄﾞｳｪｱ

// ===== USBクロック元任意選択    (Fusb = FpllX / USB_div)
// ここにdivの実効値を使ってください。
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL0
#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_UPLL
#define CONFIG_USBCLK_DIV           1

// ===== 目的対象周波数 (システム クロック)
// - XTAL周波数: 12MHz
// - システム クロック元: PLLA
// - システム クロック前置分周器: 2 (2分周)
// - PLLA供給元: XTAL
// - PLLA出力: XTAL * 14 / 1
// - システム クロックは: 12 * 14 / 1 /2 = 84MHz
// ===== 目的対象周波数 (USBクロック)
// - USBクロック元: UPLL
// - USBクロック分周器: 1 (分周なし)
// - UPLL周波数: 480MHz
// - USBクロック: 480 / 1 = 480MHz

#endif /* CONF_CLOCK_H_INCLUDED */
```

3.3.3. conf_clocks.h

3.3.3.1. SAM D21ﾃﾞﾊﾞｲｽ (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */
```

```
#include <clock.h>

#ifndef CONF_CLOCKS_H_INCLUDED
#  define CONF_CLOCKS_H_INCLUDED

/* システム クロック バス構成設定 */
#  define CONF_CLOCK_CPU_CLOCK_FAILURE_DETECT      false
#  define CONF_CLOCK_FLASH_WAIT_STATES             2
#  define CONF_CLOCK_CPU_DIVIDER                   SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBA_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBB_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBC_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1

/* SYSTEM_CLOCK_SOURCE_OSC8M構成設定 - 内部8MHz発振器 */
#  define CONF_CLOCK_OSC8M_PRESCALER               SYSTEM_OSC8M_DIV_1
#  define CONF_CLOCK_OSC8M_ON_DEMAND               true
#  define CONF_CLOCK_OSC8M_RUN_IN_STANDBY          false

/* SYSTEM_CLOCK_SOURCE_XOSC構成設定 - 外部ｸﾛｯｸ/発振器 */
#  define CONF_CLOCK_XOSC_ENABLE                   false
#  define CONF_CLOCK_XOSC_EXTERNAL_CRYSTAL         SYSTEM_CLOCK_EXTERNAL_CRYSTAL
#  define CONF_CLOCK_XOSC_EXTERNAL_FREQUENCY       12000000UL
#  define CONF_CLOCK_XOSC_STARTUP_TIME             SYSTEM_XOSC_STARTUP_32768
#  define CONF_CLOCK_XOSC_AUTO_GAIN_CONTROL        true
#  define CONF_CLOCK_XOSC_ON_DEMAND                true
#  define CONF_CLOCK_XOSC_RUN_IN_STANDBY           false

/* SYSTEM_CLOCK_SOURCE_XOSC32K構成設定 - 外部32kHzｸﾘｽﾀﾙ/ｸﾛｯｸ発振器 */
#  define CONF_CLOCK_XOSC32K_ENABLE                false
#  define CONF_CLOCK_XOSC32K_EXTERNAL_CRYSTAL      SYSTEM_CLOCK_EXTERNAL_CRYSTAL
#  define CONF_CLOCK_XOSC32K_STARTUP_TIME          SYSTEM_XOSC32K_STARTUP_65536
#  define CONF_CLOCK_XOSC32K_AUTO_AMPLITUDE_CONTROL  false
#  define CONF_CLOCK_XOSC32K_ENABLE_1KHZ_OUPUT     false
#  define CONF_CLOCK_XOSC32K_ENABLE_32KHZ_OUTPUT   true
#  define CONF_CLOCK_XOSC32K_ON_DEMAND             true
#  define CONF_CLOCK_XOSC32K_RUN_IN_STANDBY        false

/* SYSTEM_CLOCK_SOURCE_OSC32K構成設定 - 内部32kHz発振器 */
#  define CONF_CLOCK_OSC32K_ENABLE                 false
#  define CONF_CLOCK_OSC32K_STARTUP_TIME           SYSTEM_OSC32K_STARTUP_130
#  define CONF_CLOCK_OSC32K_ENABLE_1KHZ_OUTPUT     true
#  define CONF_CLOCK_OSC32K_ENABLE_32KHZ_OUTPUT    true
#  define CONF_CLOCK_OSC32K_ON_DEMAND              true
#  define CONF_CLOCK_OSC32K_RUN_IN_STANDBY         false

/* SYSTEM_CLOCK_SOURCE_DFLL構成設定 - ﾃﾞｼﾞﾀﾙ周波数固定化閉路 */
#  define CONF_CLOCK_DFLL_ENABLE                   true
#  define CONF_CLOCK_DFLL_LOOP_MODE                SYSTEM_CLOCK_DFLL_LOOP_MODE_USB_RECOVERY
#  define CONF_CLOCK_DFLL_ON_DEMAND                true

/* DFLL開路動作構成設定 */
#  define CONF_CLOCK_DFLL_FINE_VALUE               (512)

/* DFLL閉路動作構成設定 */
#  define CONF_CLOCK_DFLL_SOURCE_GCLK_GENERATOR    GCLK_GENERATOR_1
#  define CONF_CLOCK_DFLL_MULTIPLY_FACTOR          (48000000 / 32768)
#  define CONF_CLOCK_DFLL_QUICK_LOCK               true
#  define CONF_CLOCK_DFLL_TRACK_AFTER_FINE_LOCK    true
#  define CONF_CLOCK_DFLL_KEEP_LOCK_ON_WAKEUP      true
#  define CONF_CLOCK_DFLL_ENABLE_CHILL_CYCLE       true
#  define CONF_CLOCK_DFLL_MAX_COARSE_STEP_SIZE     (0x1f / 4)
```

```
#  define CONF_CLOCK_DFLL_MAX_FINE_STEP_SIZE      (0xff / 4)

/* SYSTEM_CLOCK_SOURCE_DPLL構成設定 - ﾃﾞｼﾞﾀﾙ位相固定化閉路 */
#  define CONF_CLOCK_DPLL_ENABLE                  false
#  define CONF_CLOCK_DPLL_ON_DEMAND               true
#  define CONF_CLOCK_DPLL_RUN_IN_STANDBY          false
#  define CONF_CLOCK_DPLL_LOCK_BYPASS             false
#  define CONF_CLOCK_DPLL_WAKE_UP_FAST            false
#  define CONF_CLOCK_DPLL_LOW_POWER_ENABLE        false
#  define CONF_CLOCK_DPLL_LOCK_TIME               SYSTEM_CLOCK_SOURCE_DPLL_LOCK_TIME_DEFAULT
#  define CONF_CLOCK_DPLL_REFERENCE_CLOCK         SYSTEM_CLOCK_SOURCE_DPLL_REFERENCE_CLOCK_XOSC32K
#  define CONF_CLOCK_DPLL_FILTER                  SYSTEM_CLOCK_SOURCE_DPLL_FILTER_DEFAULT
#  define CONF_CLOCK_DPLL_REFERENCE_FREQUENCY     32768
#  define CONF_CLOCK_DPLL_REFERENCE_DIVIDER       1
#  define CONF_CLOCK_DPLL_OUTPUT_FREQUENCY        48000000

/* DPLL GCLK基準構成設定 */
# define CONF_CLOCK_DPLL_REFERENCE_GCLK_GENERATOR GCLK_GENERATOR_1
/* DPLL GCLK固定化計時器構成設定 */
# define CONF_CLOCK_DPLL_LOCK_GCLK_GENERATOR GCLK_GENERATOR_1

/* clocks_init走行時にGCLKを構成設定するためにこれをtrueに設定してください。
 * falseに設定した場合、GCLK発振器はclocks_init()で何も構成設定されません。*/
#  define CONF_CLOCK_CONFIGURE_GCLK               true

/* GCLK発振器0構成設定（主ｸﾛｯｸ） */
#  define CONF_CLOCK_GCLK_0_ENABLE                true
#  define CONF_CLOCK_GCLK_0_RUN_IN_STANDBY        true
#  define CONF_CLOCK_GCLK_0_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_DFLL
#  define CONF_CLOCK_GCLK_0_PRESCALER             1
#  define CONF_CLOCK_GCLK_0_OUTPUT_ENABLE         false

/* GCLK発振器1構成設定 */
#  define CONF_CLOCK_GCLK_1_ENABLE                false
#  define CONF_CLOCK_GCLK_1_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_1_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_XOSC32K
#  define CONF_CLOCK_GCLK_1_PRESCALER             1
#  define CONF_CLOCK_GCLK_1_OUTPUT_ENABLE         false

/* GCLK発振器2構成設定（RTC） */
#  define CONF_CLOCK_GCLK_2_ENABLE                false
#  define CONF_CLOCK_GCLK_2_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_2_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC32K
#  define CONF_CLOCK_GCLK_2_PRESCALER             32
#  define CONF_CLOCK_GCLK_2_OUTPUT_ENABLE         false

/* GCLK発振器3構成設定 */
#  define CONF_CLOCK_GCLK_3_ENABLE                false
#  define CONF_CLOCK_GCLK_3_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_3_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_3_PRESCALER             1
#  define CONF_CLOCK_GCLK_3_OUTPUT_ENABLE         false

/* GCLK発振器4構成設定 */
#  define CONF_CLOCK_GCLK_4_ENABLE                false
#  define CONF_CLOCK_GCLK_4_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_4_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_4_PRESCALER             1
#  define CONF_CLOCK_GCLK_4_OUTPUT_ENABLE         false

/* GCLK発振器5構成設定 */
```

```
#  define CONF_CLOCK_GCLK_5_ENABLE                    false
#  define CONF_CLOCK_GCLK_5_RUN_IN_STANDBY            false
#  define CONF_CLOCK_GCLK_5_CLOCK_SOURCE              SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_5_PRESCALER                 1
#  define CONF_CLOCK_GCLK_5_OUTPUT_ENABLE             false

/* GCLK発振器6構成設定 */
#  define CONF_CLOCK_GCLK_6_ENABLE                    false
#  define CONF_CLOCK_GCLK_6_RUN_IN_STANDBY            false
#  define CONF_CLOCK_GCLK_6_CLOCK_SOURCE              SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_6_PRESCALER                 1
#  define CONF_CLOCK_GCLK_6_OUTPUT_ENABLE             false

/* GCLK発振器7構成設定 */
#  define CONF_CLOCK_GCLK_7_ENABLE                    false
#  define CONF_CLOCK_GCLK_7_RUN_IN_STANDBY            false
#  define CONF_CLOCK_GCLK_7_CLOCK_SOURCE              SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_7_PRESCALER                 1
#  define CONF_CLOCK_GCLK_7_OUTPUT_ENABLE             false

/* GCLK発振器8構成設定 */
#  define CONF_CLOCK_GCLK_8_ENABLE                    false
#  define CONF_CLOCK_GCLK_8_RUN_IN_STANDBY            false
#  define CONF_CLOCK_GCLK_8_CLOCK_SOURCE              SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_8_PRESCALER                 1
#  define CONF_CLOCK_GCLK_8_OUTPUT_ENABLE             false

#endif /* CONF_CLOCKS_H_INCLUDED */
```

3.3.4. conf_board.h

3.3.4.1. AT32UC3C, ATUCXXD, ATUCXXL3U, ATUCXXL4Uﾃﾞﾊﾞｲｽ (USBC)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// この例については既定基板初期化(ｽｲｯﾁ/LED)だけが必要です。

#endif /* CONF_BOARD_H_INCLUDED */
```

3.3.4.2. SAM3X, SAM3Aﾃﾞﾊﾞｲｽ (UOTGHS:USB OTG 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// USBﾋﾟﾝが使われます。
#define CONF_BOARD_USB_PORT

#endif /* CONF_BOARD_H_INCLUDED */
```

3.3.4.3. SAM D21ﾃﾞﾊﾞｲｽ (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */
```

```
#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

/* USB VBUS検出許可 */
#define CONF_BOARD_USB_VBUS_DETECT

#endif /* CONF_BOARD_H_INCLUDED */
```

4. 人間インターフェース装置キーボード(HIDキーボード)用USB装置インターフェース(UDI)

人間インターフェース装置キーボード(HIDキーボード)用USB装置インターフェース(UDI)はUSB HIDキーボード装置の構成設定と管理に関するインターフェースを提供します。

この資料の概要は以下のとおりです。

- API概要
- USB装置キーボード単位部(UDIキーボード)用の即時開始の手引き
- 構成設定ファイル例

Atmel®ソフトウェア枠組み(ASF)USB装置階層とUSB装置HIDキーボードのより多くの詳細については以下の応用記述を参照してください。

- AVR4900 : ASF - USB装置階層
- AVR4904 : ASF - USB装置HIDキーボード応用
- AVR4920 : ASF - USB装置階層 - 適合と性能係数
- AVR4921 : ASF - USB装置階層 - ASF V1とV2間の違い

4.1. API概要

4.1.1. 変数と型定義

4.1.1.1. USB装置コア(UDC)とのインターフェース

UDCによって必要とされる構造体

udi_api_hid_kbd変数

```
UDC_DESC_STORAGE      udi_api_t udi_api_hid_kbd
```

UDC用の標準UDI APIを含む全域構造体

4.1.2. 構造体定義

4.1.2.1. udi_hid_kbd_desc_t構造体

HIDキーボード用インターフェース記述子構造体

表4-1. メンバ

型	名前	説明
usb_ep_desc_t	ep	標準USBエンドポイント記述子構造体
usb_hid_descriptor_t	hid	HID記述子
usb_iface_desc_t	iface	標準USBインターフェース記述構造体

4.1.2.2. udi_hid_kbd_report_desc_t構造体

HIDキーボード用報告記述子

表4-2. メンバ

型	名前	説明
uint8_t	array[]	詳細報告データを置く配列

4.1.3. マクロ定義

4.1.3.1. USBインターフェース記述子

以下の構造体は予め定義されたUSBインターフェース記述子を提供します。これは最終的なUSB記述子を定義するのに使われなければなりません。

UDI_HID_KBD_STRING_ID マクロ

```
#define UDI_HID_KBD_STRING_ID
```

既定によってこのインターフェースに関連する文字列はなし

UDI_HID_KBD_EP_SIZE マクロ

```
#define UDI_HID_KBD_EP_SIZE
```

HIDキーボードのエンドポイントの大きさ

UDI_HID_KBD_DESC マクロ

```
#define UDI_HID_KBD_DESC
```

全ての速度に対するHIDキーボード インターフェース記述子の内容

4.1.4. 関数定義

4.1.4.1. 人間インターフェース装置(HID)キーボード クラス用USB装置インターフェース(UDI)

このUSBクラスを使うために上位応用によって使われる共通的なAPI

udi_hid_kbd_modifier_up()関数

修飾キー解放事象送出

```
bool udi_hid_kbd_modifier_up( uint8_t modifier_id )
```

表4-3. パラメータ

パラメータ名	データ方向	説明
modifier_id	[入力]	修飾キーのID

戻り値：関数が成功裏に終了した場合に1、さもなければ0

udi_hid_kbd_modifier_down()関数

修飾キー押下事象送出

```
bool udi_hid_kbd_modifier_down( uint8_t modifier_id )
```

表4-4. パラメータ

パラメータ名	データ方向	説明
modifier_id	[入力]	修飾キーのID

戻り値：関数が成功裏に終了した場合に1、さもなければ0

udi_hid_kbd_up()関数

キー解放事象送出

```
bool udi_hid_kbd_up( uint8_t key_id )
```

表4-5. パラメータ

パラメータ名	データ方向	説明
key_id	[入力]	キーのID

戻り値：関数が成功裏に終了した場合に1、さもなければ0

udi_hid_kbd_down()関数

キー押下事象送出

```
bool udi_hid_kbd_down( uint8_t key_id )
```

表4-6. パラメータ

パラメータ名	データ方向	説明
key_id	[入力]	キーのID

戻り値：関数が成功裏に終了した場合に1、さもなければ0

4.2. USB装置キーボード単位部(UDIキーボード)用の即時開始の手引き

これは使用事例の選択に於いて単位部をどう構成設定して使うかを段階的に指示する、USB装置キーボード単位部(UDIキーボード)用の即時開始の手引きです。

使用事例は様々なコードの断片を含みます。構成設定に関する段階でのコードの断片は独自初期化関数内に複写することができ、一方使用に関する段階は例えば主応用関数内に複写することができます。

4.2.1. 基本的な使用事例

この使用事例では"USB HID keyboard (Single Interface Device)"単位部が使われます。"USB HID keyboard (Composite Device)"単位部の使い方は高度な使用事例で記述されます。

4.2.1.1. 構成設定段階

USB装置のため、共通USB装置構成設定段階に従います。USB装置基本構成設定を参照してください。

4.2.1.2. 使用段階

コード例

conf_usb.hの内容

```
#define UDI_HID_KBD_ENABLE_EXT() my_callback_keyboard_enable()
extern bool my_callback_keyboard_enable(void);
#define UDI_HID_KBD_DISABLE_EXT() my_callback_keyboard_disable()
extern void my_callback_keyboard_disable(void);
#include "udi_hid_keyboard_conf.h" // conf_usb.hファイルの最後で
```

応用Cファイルに追加してください。

```
static bool my_flag_autorize_keyboard_events = false;
bool my_callback_keyboard_enable(void)
{
   my_flag_autorize_keyboard_events = true;
   return true;
}
void my_callback_keyboard_disable(void)
{
   my_flag_autorize_keyboard_events = false;
}

void my_key_A_press_event(void)
{
   if (!my_flag_autorize_keyboard_events) {
      return;
   }
   udi_hid_kbd_up(HID_A);
}
```

作業の流れ

1. conf_usb.hが利用可能でUSB装置ｷｰﾎﾞｰﾄﾞ構成設定である以下の構成設定を含むことを確実にしてください。

```
#define UDI_HID_KBD_ENABLE_EXT() my_callback_keyboard_enable()
extern bool my_callback_keyboard_enable(void);
```

注：装置列挙(USB装置の検出と識別)後、USBﾎｽﾄは装置の構成設定を始めます。装置からのUSBｷｰﾎﾞｰﾄﾞ ｲﾝﾀｰﾌｪｰｽがﾎｽﾄによって受け入れられると、USBﾎｽﾄはこのｲﾝﾀｰﾌｪｰｽを許可してUDI_HID_KBD_ENABLE_EXT()呼び戻し関数が呼ばれ、trueが返ります。故に、この関数でｷｰﾎﾞｰﾄﾞによって使われる感知器を許可することが推奨されます。

```
#define UDI_HID_KBD_DISABLE_EXT() my_callback_keyboard_disable()
extern void my_callback_keyboard_disable(void);
```

注：USB装置が切断されるか、またはUSBﾎｽﾄによってﾘｾｯﾄされると、USBｲﾝﾀｰﾌｪｰｽは禁止され、UDI_HID_KBD_DISABLE_EXT()呼び戻し関数が呼ばれます。故に、この関数でｷｰﾎﾞｰﾄﾞによって使われる感知器を禁止することが推奨されます。

2. ｷｰﾎﾞｰﾄﾞ事象送出

```
// 修飾ｷｰ解放事象送出
udi_hid_kbd_modifier_up(uint8_t modifier_id);
// 装飾ｷｰ押下事象送出
udi_hid_kbd_modifier_down(uint8_t modifier_id);
// ｷｰ解放事象送出
udi_hid_kbd_up(uint8_t key_id);
// ｷｰ押下事象送出
udi_hid_kbd_down(uint8_t key_id);
```

4.2.2. 高度な使用事例

UDI HID単位部の複数ｲﾝﾀｰﾌｪｰｽの使用については以下をご覧ください。

・複合装置でのHIDｷｰﾎﾞｰﾄﾞ

UDI HIDｷｰﾎﾞｰﾄﾞ単位部のもっと高度な使用については以下をご覧ください。

・USB装置の高度な使用事例

4.2.3. 複合装置でのHIDｷｰﾎﾞｰﾄﾞ

USB複合装置は1つよりも多くのUSBｸﾗｽを使うUSB装置です。この使用事例ではUSB複合装置を作成するのに"USB HID Keyboard (Composite Device)"単位部が使われます。故に、このUSB単位部は"USB MSC (Composite Device)"のような別な"複合装置(Composite Device)"単位部と連携することができます。

また、「AVR4902:ASF - USB複合装置」応用記述を参照することもできます。

4.2.3.1. 構成設定段階

動くためのこの事例の構成設定ｺｰﾄﾞについては基本的な使用事例に従わなければなりません。

4.2.3.2. 使用段階

ｺｰﾄﾞ例

conf_usb.hの内容

```
#define USB_DEVICE_EP_CTRL_SIZE  64
#define USB_DEVICE_NB_INTERFACE (X+1)
#define USB_DEVICE_MAX_EP (X+1)

#define UDI_HID_KBD_EP_IN  (X | USB_EP_DIR_IN)
#define UDI_HID_KBD_IFACE_NUMBER  X

#define UDI_COMPOSITE_DESC_T ¥
   udi_hid_kbd_desc_t udi_hid_kbd; ¥
   ～
#define UDI_COMPOSITE_DESC_FS ¥
   .udi_hid_kbd = UDI_HID_KBD_DESC, ¥
   ～
#define UDI_COMPOSITE_DESC_HS ¥
   .udi_hid_kbd = UDI_HID_KBD_DESC, ¥
   ～
#define UDI_COMPOSITE_API ¥
   &udi_api_hid_kbd, ¥
   ～
```

作業の流れ

1. conf_usb.hが利用可能でUSB複合装置構成設定に必要とされる以下のﾊﾟﾗﾒｰﾀを含むことを確実にしてください。

```
// ｴﾝﾄﾞﾎﾟｲﾝﾄ制御容量、これは以下でなければなりません。
// - 低速(LS)装置に対して、8
// - 全速(FS)装置に対して、8, 16 ,32 または 64 (RAMを節約するために8が推奨されます。)
// - 高速(HS)装置に対して、64
#define USB_DEVICE_EP_CTRL_SIZE  64
// このUSB装置での総ｲﾝﾀｰﾌｪｰｽ数
// HIDｷｰﾎﾞｰﾄﾞ用に1を加算
#define USB_DEVICE_NB_INTERFACE (X+1)
// このUSB装置での総ｴﾝﾄﾞﾎﾟｲﾝﾄ数
// これには各ｲﾝﾀｰﾌｪｰｽに対する各々のｴﾝﾄﾞﾎﾟｲﾝﾄを含めなければなりません。
// HIDｷｰﾎﾞｰﾄﾞ用に1を加算
#define USB_DEVICE_MAX_EP (X+1)
```

2. conf_usb.hが複合装置の記述子を含むことを確実にしてください。

```
// あなたが選んだｷｰﾎﾞｰﾄﾞ用のｴﾝﾄﾞﾎﾟｲﾝﾄ番号
// ｴﾝﾄﾞﾎﾟｲﾝﾄ番号は1から始まります。
#define UDI_HID_KBD_EP_IN  (X | USB_EP_DIR_IN)
// 0から始まるｲﾝﾀｰﾌｪｰｽのｲﾝﾀｰﾌｪｰｽ指標
#define UDI_HID_KBD_IFACE_NUMBER  X
```

3. conf_usb.hがUSB複合装置構成設定に必要とされる以下のﾊﾟﾗﾒｰﾀを含むことを確実にしてください。

```
// USBｲﾝﾀｰﾌｪｰｽ記述子構造体
#define UDI_COMPOSITE_DESC_T ¥
   ～
   udi_hid_kbd_desc_t udi_hid_kbd; ¥
   ～
// 全速(FS)用USBｲﾝﾀｰﾌｪｰｽ記述子値
#define UDI_COMPOSITE_DESC_FS ¥
   ～
   .udi_hid_kbd = UDI_HID_KBD_DESC, ¥
   ～
// 高速(HS)用USBｲﾝﾀｰﾌｪｰｽ記述子値
#define UDI_COMPOSITE_DESC_HS ¥
```

```
    ～
    .udi_hid_kbd = UDI_HID_KBD_DESC, ¥
    ～
// USBインターフェースAPI
#define UDI_COMPOSITE_API ¥
    ～
    &udi_api_hid_kbd, ¥
      ～
```

注：上の4つの一覧で与えられた記述子の順番は全てのインターフェース指標によって定義された順番と同じでなければなりません。インターフェース指標の順番はUDI_X_IFACE_NUMBER定義を通して定義されます。

4.3. 構成設定ファイル例

4.3.1. conf_usb.h

4.3.1.1. 単一UDI HID KBD

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_USB_H_
#define _CONF_USB_H_

#include "compiler.h"

#warning You must refill the following definitions with a correct values

#define  USB_DEVICE_VENDOR_ID             USB_VID_ATMEL
#define  USB_DEVICE_PRODUCT_ID            USB_PID_ATMEL_ASF_HIDKEYBOARD
#define  USB_DEVICE_MAJOR_VERSION         1
#define  USB_DEVICE_MINOR_VERSION         0
#define  USB_DEVICE_POWER                 100 // VBUS線での消費(mA)
#define  USB_DEVICE_ATTR                  ¥
    (USB_CONFIG_ATTR_SELF_POWERED)
// (USB_CONFIG_ATTR_BUS_POWERED)
//   (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_SELF_POWERED)
//   (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_BUS_POWERED)

// #define  USB_DEVICE_MANUFACTURE_NAME      "Manufacture name"
// #define  USB_DEVICE_PRODUCT_NAME          "Product name"
// #define  USB_DEVICE_SERIAL_NAME           "12...EF"

//#define  USB_DEVICE_LOW_SPEED

#if (UC3A3||UC3A4)
//#define  USB_DEVICE_HS_SUPPORT
#endif

// #define  UDC_VBUS_EVENT(b_vbus_high)      user_callback_vbus_action(b_vbus_high)
// extern void user_callback_vbus_action(bool b_vbus_high);
// #define  UDC_SOF_EVENT()                  user_callback_sof_action()
// extern void user_callback_sof_action(void);
// #define  UDC_SUSPEND_EVENT()              user_callback_suspend_action()
// extern void user_callback_suspend_action(void);
// #define  UDC_RESUME_EVENT()               user_callback_resume_action()
// extern void user_callback_resume_action(void);
// #define  UDC_REMOTEWAKEUP_ENABLE()        user_callback_remotewakeup_enable()
// extern void user_callback_remotewakeup_enable(void);
// #define  UDC_REMOTEWAKEUP_DISABLE()       user_callback_remotewakeup_disable()
// extern void user_callback_remotewakeup_disable(void);
// #define  UDC_GET_EXTRA_STRING()
```

```
#define  UDI_HID_KBD_ENABLE_EXT()       true
#define  UDI_HID_KBD_DISABLE_EXT()
// #define UDI_HID_KBD_ENABLE_EXT() my_callback_keyboard_enable()
// extern bool my_callback_keyboard_enable(void);
// #define UDI_HID_KBD_DISABLE_EXT() my_callback_keyboard_disable()
// extern void my_callback_keyboard_disable(void);
#define  UDI_HID_KBD_CHANGE_LED(value)
// #define  UDI_HID_KBD_CHANGE_LED(value) my_callback_keyboard_led(value)
// extern void my_callback_keyboard_led(uint8_t value)

#include "udi_hid_kbd_conf.h"

#endif // _CONF_USB_H_
```

4.3.1.2. 複数UDI HID KBD (複合)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_USB_H_
#define _CONF_USB_H_

#include "compiler.h"

#warning You must refill the following definitions with a correct values

#define  USB_DEVICE_VENDOR_ID             USB_VID_ATMEL
#define  USB_DEVICE_PRODUCT_ID            0xFFFF
#define  USB_DEVICE_MAJOR_VERSION         1
#define  USB_DEVICE_MINOR_VERSION         0
#define  USB_DEVICE_POWER                 100 // VBUS線での消費(mA)
#define  USB_DEVICE_ATTR                  ¥
        (USB_CONFIG_ATTR_SELF_POWERED)
// (USB_CONFIG_ATTR_BUS_POWERED)
// (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_SELF_POWERED)
// (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_BUS_POWERED)

// #define  USB_DEVICE_MANUFACTURE_NAME      "Manufacture name"
// #define  USB_DEVICE_PRODUCT_NAME          "Product name"
// #define  USB_DEVICE_SERIAL_NAME           "12...EF" // MSC用ﾃﾞｨｽｸ通番

//#define  USB_DEVICE_LOW_SPEED

#if (UC3A3||UC3A4)
//#define  USB_DEVICE_HS_SUPPORT
#endif

// #define  UDC_VBUS_EVENT(b_vbus_high)      user_callback_vbus_action(b_vbus_high)
// extern void user_callback_vbus_action(bool b_vbus_high);
// #define  UDC_SOF_EVENT()                  user_callback_sof_action()
// extern void user_callback_sof_action(void);
// #define  UDC_SUSPEND_EVENT()              user_callback_suspend_action()
// extern void user_callback_suspend_action(void);
// #define  UDC_RESUME_EVENT()               user_callback_resume_action()
// extern void user_callback_resume_action(void);
// #define  UDC_REMOTEWAKEUP_ENABLE()        user_callback_remotewakeup_enable()
// extern void user_callback_remotewakeup_enable(void);
// #define  UDC_REMOTEWAKEUP_DISABLE()       user_callback_remotewakeup_disable()
// extern void user_callback_remotewakeup_disable(void);
// #define  UDC_GET_EXTRA_STRING()
```

```
#define  USB_DEVICE_EP_CTRL_SIZE       64

#define  USB_DEVICE_NB_INTERFACE       1 // 1またはそれ以上

#define  USB_DEVICE_MAX_EP             1 // 0～ｲﾝﾀｰﾌｪｰｽによって要求された最大ｴﾝﾄﾞﾎﾟｲﾝﾄ

#define  UDI_CDC_PORT_NB 1

#define  UDI_CDC_ENABLE_EXT(port)             true
#define  UDI_CDC_DISABLE_EXT(port)
#define  UDI_CDC_RX_NOTIFY(port)
#define  UDI_CDC_TX_EMPTY_NOTIFY(port)
#define  UDI_CDC_SET_CODING_EXT(port,cfg)
#define  UDI_CDC_SET_DTR_EXT(port,set)
#define  UDI_CDC_SET_RTS_EXT(port,set)
/*
 * #define UDI_CDC_ENABLE_EXT(port) my_callback_cdc_enable()
 * extern bool my_callback_cdc_enable(void);
 * #define UDI_CDC_DISABLE_EXT(port) my_callback_cdc_disable()
 * extern void my_callback_cdc_disable(void);
 * #define  UDI_CDC_RX_NOTIFY(port) my_callback_rx_notify(port)
 * extern void my_callback_rx_notify(uint8_t port);
 * #define  UDI_CDC_TX_EMPTY_NOTIFY(port) my_callback_tx_empty_notify(port)
 * extern void my_callback_tx_empty_notify(uint8_t port);
 * #define  UDI_CDC_SET_CODING_EXT(port,cfg) my_callback_config(port,cfg)
 * extern void my_callback_config(uint8_t port, usb_cdc_line_coding_t * cfg);
 * #define  UDI_CDC_SET_DTR_EXT(port,set) my_callback_cdc_set_dtr(port,set)
 * extern void my_callback_cdc_set_dtr(uint8_t port, bool b_enable);
 * #define  UDI_CDC_SET_RTS_EXT(port,set) my_callback_cdc_set_rts(port,set)
 * extern void my_callback_cdc_set_rts(uint8_t port, bool b_enable);
 */

#define  UDI_CDC_LOW_RATE

#define  UDI_CDC_DEFAULT_RATE             115200
#define  UDI_CDC_DEFAULT_STOPBITS         CDC_STOP_BITS_1
#define  UDI_CDC_DEFAULT_PARITY           CDC_PAR_NONE
#define  UDI_CDC_DEFAULT_DATABITS         8

#define  UDI_CDC_DATA_EP_IN_0          (1 | USB_EP_DIR_IN)  // 送信
#define  UDI_CDC_DATA_EP_OUT_0         (2 | USB_EP_DIR_OUT) // 受信
#define  UDI_CDC_COMM_EP_0             (3 | USB_EP_DIR_IN)  // 通知ｴﾝﾄﾞﾎﾟｲﾝﾄ
#define  UDI_CDC_DATA_EP_IN_2          (4 | USB_EP_DIR_IN)  // 送信
#define  UDI_CDC_DATA_EP_OUT_2         (5 | USB_EP_DIR_OUT) // 受信
#define  UDI_CDC_COMM_EP_2             (6 | USB_EP_DIR_IN)  // 通知ｴﾝﾄﾞﾎﾟｲﾝﾄ
#define  UDI_CDC_DATA_EP_IN_3          (7 | USB_EP_DIR_IN)  // 送信
#define  UDI_CDC_DATA_EP_OUT_3         (8 | USB_EP_DIR_OUT) // 受信
#define  UDI_CDC_COMM_EP_3             (9 | USB_EP_DIR_IN)  // 通知ｴﾝﾄﾞﾎﾟｲﾝﾄ

#define  UDI_CDC_COMM_IFACE_NUMBER_0   0
#define  UDI_CDC_DATA_IFACE_NUMBER_0   1
#define  UDI_CDC_COMM_IFACE_NUMBER_2   2
#define  UDI_CDC_DATA_IFACE_NUMBER_2   3
#define  UDI_CDC_COMM_IFACE_NUMBER_3   4
#define  UDI_CDC_DATA_IFACE_NUMBER_3   5

#define UDI_MSC_GLOBAL_VENDOR_ID           ¥
   'A', 'T', 'M', 'E', 'L', ' ', ' ', ' '
#define UDI_MSC_GLOBAL_PRODUCT_VERSION            ¥
   '1', '.', '0', '0'
```

```
#define  UDI_MSC_ENABLE_EXT()          true
#define  UDI_MSC_DISABLE_EXT()
#define  UDI_MSC_NOTIFY_TRANS_EXT()
/*
 * #define UDI_MSC_ENABLE_EXT() my_callback_msc_enable()
 * extern bool my_callback_msc_enable(void);
 * #define UDI_MSC_DISABLE_EXT() my_callback_msc_disable()
 * extern void my_callback_msc_disable(void);
 * #define  UDI_MSC_NOTIFY_TRANS_EXT()    msc_notify_trans()
 * extern void msc_notify_trans(void) {
 */

#define  UDI_MSC_EP_IN                 (1 | USB_EP_DIR_IN)
#define  UDI_MSC_EP_OUT                (2 | USB_EP_DIR_OUT)

#define  UDI_MSC_IFACE_NUMBER          0

#define  UDI_HID_MOUSE_ENABLE_EXT()       true
#define  UDI_HID_MOUSE_DISABLE_EXT()
// #define UDI_HID_MOUSE_ENABLE_EXT() my_callback_mouse_enable()
// extern bool my_callback_mouse_enable(void);
// #define UDI_HID_MOUSE_DISABLE_EXT() my_callback_mouse_disable()
// extern void my_callback_mouse_disable(void);

#define  UDI_HID_MOUSE_EP_IN           (1 | USB_EP_DIR_IN)

#define  UDI_HID_MOUSE_IFACE_NUMBER    0

#define  UDI_HID_KBD_ENABLE_EXT()       true
#define  UDI_HID_KBD_DISABLE_EXT()
// #define UDI_HID_KBD_ENABLE_EXT() my_callback_keyboard_enable()
// extern bool my_callback_keyboard_enable(void);
// #define UDI_HID_KBD_DISABLE_EXT() my_callback_keyboard_disable()
// extern void my_callback_keyboard_disable(void);
#define  UDI_HID_KBD_CHANGE_LED(value)
// #define  UDI_HID_KBD_CHANGE_LED(value) my_callback_keyboard_led(value)
// extern void my_callback_keyboard_led(uint8_t value)
#define  UDI_HID_KBD_EP_IN           (1 | USB_EP_DIR_IN)

#define  UDI_HID_KBD_IFACE_NUMBER    0

#define  UDI_HID_GENERIC_ENABLE_EXT()       true
#define  UDI_HID_GENERIC_DISABLE_EXT()
#define  UDI_HID_GENERIC_REPORT_OUT(ptr)
#define  UDI_HID_GENERIC_SET_FEATURE(f)
/*
 * #define UDI_HID_GENERIC_ENABLE_EXT() my_callback_generic_enable()
 * extern bool my_callback_generic_enable(void);
 * #define UDI_HID_GENERIC_DISABLE_EXT() my_callback_generic_disable()
 * extern void my_callback_generic_disable(void);
 * #define  UDI_HID_GENERIC_REPORT_OUT(ptr) my_callback_generic_report_out(ptr)
 * extern void my_callback_generic_report_out(uint8_t *report);
 * #define  UDI_HID_GENERIC_SET_FEATURE(f) my_callback_generic_set_feature(f)
 * extern void my_callback_generic_set_feature(uint8_t *report_feature);
 */
#define  UDI_HID_REPORT_IN_SIZE             64
#define  UDI_HID_REPORT_OUT_SIZE            64
#define  UDI_HID_REPORT_FEATURE_SIZE        4
#define  UDI_HID_GENERIC_EP_SIZE            64

#define  UDI_HID_GENERIC_EP_OUT    (2 | USB_EP_DIR_OUT)
```

```
#define  UDI_HID_GENERIC_EP_IN    (1 | USB_EP_DIR_IN)

#define  UDI_HID_GENERIC_IFACE_NUMBER    0

#define  UDI_PHDC_ENABLE_EXT()           true
#define  UDI_PHDC_DISABLE_EXT()

#define  UDI_PHDC_DATAMSG_FORMAT       USB_PHDC_DATAMSG_FORMAT_11073_20601
#define  UDI_PHDC_SPECIALIZATION       {0x2345} // 11073_20601での定義

#define  UDI_PHDC_QOS_OUT              ¥
        (USB_PHDC_QOS_MEDIUM_BETTER|USB_PHDC_QOS_HIGH_BEST)
#define  UDI_PHDC_QOS_IN               ¥
        (USB_PHDC_QOS_LOW_GOOD|USB_PHDC_QOS_MEDIUM_BETTER|USB_PHDC_QOS_MEDIUM_BEST)

#define  UDI_PHDC_METADATA_DESC_BULK_IN   {0x01,0x02,0x03}
#define  UDI_PHDC_METADATA_DESC_BULK_OUT  {0x01,0x02,0x03}
#define  UDI_PHDC_METADATA_DESC_INT_IN    {0x01,0x02,0x03}

#define  UDI_PHDC_EP_BULK_OUT          (1 | USB_EP_DIR_OUT)
#define  UDI_PHDC_EP_BULK_IN           (2 | USB_EP_DIR_IN)
#if ((UDI_PHDC_QOS_IN&USB_PHDC_QOS_LOW_GOOD)==USB_PHDC_QOS_LOW_GOOD)
// UDI_PHDC_QOS_INの場合にだけUSB_PHDC_QOS_LOW_GOODをｲﾝｸﾙｰﾄﾞ
#  define  UDI_PHDC_EP_INTERRUPT_IN      (3 | USB_EP_DIR_IN)
#endif

#define  UDI_PHDC_EP_SIZE_BULK_OUT     32
#define  UDI_PHDC_EP_SIZE_BULK_IN      32
#define  UDI_PHDC_EP_SIZE_INT_IN       8

#define  UDI_PHDC_IFACE_NUMBER          0

#define  UDI_VENDOR_ENABLE_EXT()          true
#define  UDI_VENDOR_DISABLE_EXT()
#define  UDI_VENDOR_SETUP_OUT_RECEIVED()  false
#define  UDI_VENDOR_SETUP_IN_RECEIVED()   false
/*
 * #define UDI_VENDOR_ENABLE_EXT() my_callback_vendor_enable()
 * extern bool my_callback_vendor_enable(void);
 * #define UDI_VENDOR_DISABLE_EXT() my_callback_vendor_disable()
 * extern void my_callback_vendor_disable(void);
 *
 * #define  UDI_VENDOR_SETUP_OUT_RECEIVED()  my_vendor_setup_out_received()
 * extern bool my_vendor_setup_out_received(void);
 * #define  UDI_VENDOR_SETUP_IN_RECEIVED()   my_vendor_setup_in_received()
 * extern bool my_vendor_setup_in_received(void);
 */

#define  UDI_VENDOR_EPS_SIZE_INT_FS    64
#define  UDI_VENDOR_EPS_SIZE_BULK_FS   64
#define  UDI_VENDOR_EPS_SIZE_ISO_FS   256

#define  UDI_VENDOR_EPS_SIZE_INT_HS    64
#define  UDI_VENDOR_EPS_SIZE_BULK_HS  512
#define  UDI_VENDOR_EPS_SIZE_ISO_HS    64

#define  UDI_VENDOR_EP_INTERRUPT_IN  (1 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_INTERRUPT_OUT (2 | USB_EP_DIR_OUT)
#define  UDI_VENDOR_EP_BULK_IN       (3 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_BULK_OUT      (4 | USB_EP_DIR_OUT)
#define  UDI_VENDOR_EP_ISO_IN        (5 | USB_EP_DIR_IN)
```

```
#define  UDI_VENDOR_EP_ISO_OUT       (6 | USB_EP_DIR_OUT)

#define  UDI_VENDOR_IFACE_NUMBER     0

//... 最終的に他のｲﾝﾀｰﾌｪｰｽ構成設定を追加してください。

#define UDI_COMPOSITE_DESC_T

#define UDI_COMPOSITE_DESC_FS

#define UDI_COMPOSITE_DESC_HS

#define UDI_COMPOSITE_API

/* CDC、MSC、HIDﾏｳｽ ｲﾝﾀｰﾌｪｰｽを持つ装置の例
#define UDI_COMPOSITE_DESC_T ¥
    usb_iad_desc_t udi_cdc_iad; ¥
    udi_cdc_comm_desc_t udi_cdc_comm; ¥
    udi_cdc_data_desc_t udi_cdc_data; ¥
    udi_msc_desc_t udi_msc; ¥
    udi_hid_mouse_desc_t udi_hid_mouse

#define UDI_COMPOSITE_DESC_FS ¥
    .udi_cdc_iad               = UDI_CDC_IAD_DESC_0, ¥
    .udi_cdc_comm              = UDI_CDC_COMM_DESC_0, ¥
    .udi_cdc_data              = UDI_CDC_DATA_DESC_0_FS, ¥
    .udi_msc                   = UDI_MSC_DESC_FS, ¥
    .udi_hid_mouse             = UDI_HID_MOUSE_DESC

#define UDI_COMPOSITE_DESC_HS ¥
    .udi_cdc_iad               = UDI_CDC_IAD_DESC_0, ¥
    .udi_cdc_comm              = UDI_CDC_COMM_DESC_0, ¥
    .udi_cdc_data              = UDI_CDC_DATA_DESC_0_HS, ¥
    .udi_msc                   = UDI_MSC_DESC_HS, ¥
    .udi_hid_mouse             = UDI_HID_MOUSE_DESC

#define UDI_COMPOSITE_API  ¥
    &udi_api_cdc_comm,         ¥
    &udi_api_cdc_data,         ¥
    &udi_api_msc,              ¥
    &udi_api_hid_mouse
*/

/* ｲﾝﾀｰﾌｪｰｽ用ｲﾝｸﾙｰﾄﾞの例
#include "udi_msc.h"
#include "udi_hid_kbd.h"
#include "udi_hid_mouse.h"
#include "udi_cdc.h"
#include "udi_phdc.h"
#include "udi_vendor.h"
*/
/* USBによって使われる呼び戻しの宣言
#include "callback_def.h"
*/

#endif // _CONF_USB_H_
```

4.3.2. conf_clock.h

4.3.2.1. AT32UC3A0, ATUC3A1, ATUC3Bﾃﾞﾊﾞｲｽ (USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
```

```
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== システム クロック元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RCSYS
#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_OSC0
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLL0

// ===== PLL0任意選択
#define CONFIG_PLL0_SOURCE            PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE          PLL_SRC_OSC1
#define CONFIG_PLL0_MUL               8 /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL0_DIV               2 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== PLL1任意選択
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC0
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL1_MUL             8 /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL1_DIV             2 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== システム クロック バス分周任意選択
//#define CONFIG_SYSCLK_CPU_DIV       0 /* Fcpu = Fsys/(2 ^ CPU_div) */
//#define CONFIG_SYSCLK_PBA_DIV       0 /* Fpba = Fsys/(2 ^ PBA_div) */
//#define CONFIG_SYSCLK_PBB_DIV       0 /* Fpbb = Fsys/(2 ^ PBB_div) */

// ===== 周辺機能ｸﾛｯｸ管理任意選択
//#define CONFIG_SYSCLK_INIT_CPUMASK  ((1 << SYSCLK_SYSTIMER) | (1 << SYSCLK_OCD))
//#define CONFIG_SYSCLK_INIT_PBAMASK  (1 << SYSCLK_USART0)
//#define CONFIG_SYSCLK_INIT_PBBMASK  (1 << SYSCLK_HMATRIX)
//#define CONFIG_SYSCLK_INIT_HSBMASK  (1 << SYSCLK_MDMA_HSB)

// ===== USBｸﾛｯｸ元任意選択
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_OSC0
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_PLL0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV             1 /* Fusb = Fsys/(2 ^ USB_div) */

#endif /* CONF_CLOCK_H_INCLUDED */
```

4.3.2.2. AT32UC3C, ATUCXXD, ATUCXXL3U, ATUCXXL4Uﾃﾞﾊﾞｲｽ (USBC)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== システム クロック元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RCSYS
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_OSC0
#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_PLL0
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLL1
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RC120M

// ===== PLL0任意選択
#define CONFIG_PLL0_SOURCE            PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE          PLL_SRC_RC120M
#define CONFIG_PLL0_MUL               8 /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL0_DIV               2 /* Fpll = (Fclk * PLL_mul) / PLL_div */
```

```
// ===== PLL1任意選択
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC0
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL1_SOURCE          PLL_SRC_RC120M
//#define CONFIG_PLL1_MUL             3 /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL1_DIV             1 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== システム クロック バス分周任意選択
//#define CONFIG_SYSCLK_CPU_DIV       0 /* Fcpu = Fsys/(2 ^ CPU_div) */
//#define CONFIG_SYSCLK_PBA_DIV       0 /* Fpba = Fsys/(2 ^ PBA_div) */
//#define CONFIG_SYSCLK_PBB_DIV       0 /* Fpbb = Fsys/(2 ^ PBB_div) */

// ===== 周辺機能ｸﾛｯｸ管理任意選択
//#define CONFIG_SYSCLK_INIT_CPUMASK  ((1 << SYSCLK_SYSTIMER) | (1 << SYSCLK_OCD))
//#define CONFIG_SYSCLK_INIT_PBAMASK  (1 << SYSCLK_USART0)
//#define CONFIG_SYSCLK_INIT_PBBMASK  (1 << SYSCLK_HMATRIX)
//#define CONFIG_SYSCLK_INIT_HSBMASK  (1 << SYSCLK_MDMA_HSB)

// ===== USBｸﾛｯｸ元任意選択
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_OSC0
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_PLL0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV             1 /* Fusb = Fsys/(2 ^ USB_div) */

#endif /* CONF_CLOCK_H_INCLUDED */
```

4.3.2.3. SAM3X, SAM3Aﾃﾞﾊﾞｲｽ (UOTGHS:USB OTG 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== システム ｸﾛｯｸ(MCK)元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_BYPASS
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_4M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_8M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_12M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_BYPASS
#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLLACK
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_UPLLCK

// ===== システム ｸﾛｯｸ(MCK)前置分周器任意選択    (Fmck = Fsys / (SYSCLK_PRES))
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_1
#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_2
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_4
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_8
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_16
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_32
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_64
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_3

// ===== PLL0 (A)任意選択    (Fpll = (Fclk * PLL_mul) / PLL_div)
// ここにmulとdivの実効値を使ってください。
#define CONFIG_PLL0_SOURCE          PLL_SRC_MAINCK_XTAL
#define CONFIG_PLL0_MUL             14
```

```
#define CONFIG_PLL0_DIV              1

// ===== 480MHzで固定化されたUPLL (UTMI)ﾊｰﾄﾞｳｪｱ

// ===== USBｸﾛｯｸ元任意選択    (Fusb = FpllX / USB_div)
// ここにdivの実効値を使ってください。
//#define CONFIG_USBCLK_SOURCE         USBCLK_SRC_PLL0
#define CONFIG_USBCLK_SOURCE         USBCLK_SRC_UPLL
#define CONFIG_USBCLK_DIV            1

// ===== 目的対象周波数 (ｼｽﾃﾑ ｸﾛｯｸ)
// - XTAL周波数: 12MHz
// - ｼｽﾃﾑ ｸﾛｯｸ元: PLLA
// - ｼｽﾃﾑ ｸﾛｯｸ前置分周器: 2 (2分周)
// - PLLA供給元: XTAL
// - PLLA出力: XTAL * 14 / 1
// - ｼｽﾃﾑ ｸﾛｯｸは: 12 * 14 / 1 /2 = 84MHz
// ===== 目的対象周波数 (USBｸﾛｯｸ)
// - USBｸﾛｯｸ元: UPLL
// - USBｸﾛｯｸ分周器: 1 (分周なし)
// - UPLL周波数: 480MHz
// - USBｸﾛｯｸ: 480 / 1 = 480MHz

#endif /* CONF_CLOCK_H_INCLUDED */
```

4.3.3. conf_clocks.h

4.3.3.1. SAM D21ﾃﾞﾊﾞｲｽ (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#include <clock.h>

#ifndef CONF_CLOCKS_H_INCLUDED
#  define CONF_CLOCKS_H_INCLUDED

/* ｼｽﾃﾑ ｸﾛｯｸ ﾊﾞｽ構成設定 */
#  define CONF_CLOCK_CPU_CLOCK_FAILURE_DETECT      false
#  define CONF_CLOCK_FLASH_WAIT_STATES             2
#  define CONF_CLOCK_CPU_DIVIDER                   SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBA_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBB_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBC_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1

/* SYSTEM_CLOCK_SOURCE_OSC8M構成設定 - 内部8MHz発振器 */
#  define CONF_CLOCK_OSC8M_PRESCALER               SYSTEM_OSC8M_DIV_1
#  define CONF_CLOCK_OSC8M_ON_DEMAND               true
#  define CONF_CLOCK_OSC8M_RUN_IN_STANDBY          false

/* SYSTEM_CLOCK_SOURCE_XOSC構成設定 - 外部ｸﾛｯｸ/発振器 */
#  define CONF_CLOCK_XOSC_ENABLE                   false
#  define CONF_CLOCK_XOSC_EXTERNAL_CRYSTAL         SYSTEM_CLOCK_EXTERNAL_CRYSTAL
#  define CONF_CLOCK_XOSC_EXTERNAL_FREQUENCY       12000000UL
#  define CONF_CLOCK_XOSC_STARTUP_TIME             SYSTEM_XOSC_STARTUP_32768
#  define CONF_CLOCK_XOSC_AUTO_GAIN_CONTROL        true
#  define CONF_CLOCK_XOSC_ON_DEMAND                true
#  define CONF_CLOCK_XOSC_RUN_IN_STANDBY           false

/* SYSTEM_CLOCK_SOURCE_XOSC32K構成設定 - 外部32kHzｸﾘｽﾀﾙ/ｸﾛｯｸ発振器 */
#  define CONF_CLOCK_XOSC32K_ENABLE                false
#  define CONF_CLOCK_XOSC32K_EXTERNAL_CRYSTAL      SYSTEM_CLOCK_EXTERNAL_CRYSTAL
```

```
#  define CONF_CLOCK_XOSC32K_STARTUP_TIME          SYSTEM_XOSC32K_STARTUP_65536
#  define CONF_CLOCK_XOSC32K_AUTO_AMPLITUDE_CONTROL  false
#  define CONF_CLOCK_XOSC32K_ENABLE_1KHZ_OUPUT     false
#  define CONF_CLOCK_XOSC32K_ENABLE_32KHZ_OUTPUT   true
#  define CONF_CLOCK_XOSC32K_ON_DEMAND             true
#  define CONF_CLOCK_XOSC32K_RUN_IN_STANDBY        false

/* SYSTEM_CLOCK_SOURCE_OSC32K構成設定 - 内部32kHz発振器 */
#  define CONF_CLOCK_OSC32K_ENABLE                 false
#  define CONF_CLOCK_OSC32K_STARTUP_TIME           SYSTEM_OSC32K_STARTUP_130
#  define CONF_CLOCK_OSC32K_ENABLE_1KHZ_OUTPUT     true
#  define CONF_CLOCK_OSC32K_ENABLE_32KHZ_OUTPUT    true
#  define CONF_CLOCK_OSC32K_ON_DEMAND              true
#  define CONF_CLOCK_OSC32K_RUN_IN_STANDBY         false

/* SYSTEM_CLOCK_SOURCE_DFLL構成設定 - ﾃﾞｼﾞﾀﾙ周波数固定化閉路 */
#  define CONF_CLOCK_DFLL_ENABLE                   true
#  define CONF_CLOCK_DFLL_LOOP_MODE                SYSTEM_CLOCK_DFLL_LOOP_MODE_USB_RECOVERY
#  define CONF_CLOCK_DFLL_ON_DEMAND                true

/* DFLL開路動作構成設定 */
#  define CONF_CLOCK_DFLL_FINE_VALUE               (512)

/* DFLL閉路動作構成設定 */
#  define CONF_CLOCK_DFLL_SOURCE_GCLK_GENERATOR    GCLK_GENERATOR_1
#  define CONF_CLOCK_DFLL_MULTIPLY_FACTOR          (48000000 / 32768)
#  define CONF_CLOCK_DFLL_QUICK_LOCK               true
#  define CONF_CLOCK_DFLL_TRACK_AFTER_FINE_LOCK    true
#  define CONF_CLOCK_DFLL_KEEP_LOCK_ON_WAKEUP      true
#  define CONF_CLOCK_DFLL_ENABLE_CHILL_CYCLE       true
#  define CONF_CLOCK_DFLL_MAX_COARSE_STEP_SIZE     (0x1f / 4)
#  define CONF_CLOCK_DFLL_MAX_FINE_STEP_SIZE       (0xff / 4)

/* SYSTEM_CLOCK_SOURCE_DPLL構成設定 - ﾃﾞｼﾞﾀﾙ位相固定化閉路 */
#  define CONF_CLOCK_DPLL_ENABLE                   false
#  define CONF_CLOCK_DPLL_ON_DEMAND                true
#  define CONF_CLOCK_DPLL_RUN_IN_STANDBY           false
#  define CONF_CLOCK_DPLL_LOCK_BYPASS              false
#  define CONF_CLOCK_DPLL_WAKE_UP_FAST             false
#  define CONF_CLOCK_DPLL_LOW_POWER_ENABLE         false

#  define CONF_CLOCK_DPLL_LOCK_TIME                SYSTEM_CLOCK_SOURCE_DPLL_LOCK_TIME_DEFAULT
#  define CONF_CLOCK_DPLL_REFERENCE_CLOCK          SYSTEM_CLOCK_SOURCE_DPLL_REFERENCE_CLOCK_XOSC32K
#  define CONF_CLOCK_DPLL_FILTER                   SYSTEM_CLOCK_SOURCE_DPLL_FILTER_DEFAULT

#  define CONF_CLOCK_DPLL_REFERENCE_FREQUENCY      32768
#  define CONF_CLOCK_DPLL_REFERENCE_DIVIDER        1
#  define CONF_CLOCK_DPLL_OUTPUT_FREQUENCY         48000000

/* DPLL GCLK基準構成設定 */
# define CONF_CLOCK_DPLL_REFERENCE_GCLK_GENERATOR GCLK_GENERATOR_1
/* DPLL GCLK固定化計時器構成設定 */
# define CONF_CLOCK_DPLL_LOCK_GCLK_GENERATOR GCLK_GENERATOR_1

/* clocks_init走行時にGCLKを構成設定するためにこれをtrueに設定してください。
 * falseに設定した場合、GCLK発振器はclocks_init()で何も構成設定されません。*/
#  define CONF_CLOCK_CONFIGURE_GCLK                true

/* GCLK発振器0構成設定（主ｸﾛｯｸ） */
#  define CONF_CLOCK_GCLK_0_ENABLE                 true
#  define CONF_CLOCK_GCLK_0_RUN_IN_STANDBY         true
```

```
#  define CONF_CLOCK_GCLK_0_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_DFLL
#  define CONF_CLOCK_GCLK_0_PRESCALER             1
#  define CONF_CLOCK_GCLK_0_OUTPUT_ENABLE         false

/* GCLK発振器1構成設定 */
#  define CONF_CLOCK_GCLK_1_ENABLE                false
#  define CONF_CLOCK_GCLK_1_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_1_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_XOSC32K
#  define CONF_CLOCK_GCLK_1_PRESCALER             1
#  define CONF_CLOCK_GCLK_1_OUTPUT_ENABLE         false

/* GCLK発振器2構成設定(RTC) */
#  define CONF_CLOCK_GCLK_2_ENABLE                false
#  define CONF_CLOCK_GCLK_2_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_2_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC32K
#  define CONF_CLOCK_GCLK_2_PRESCALER             32
#  define CONF_CLOCK_GCLK_2_OUTPUT_ENABLE         false

/* GCLK発振器3構成設定 */
#  define CONF_CLOCK_GCLK_3_ENABLE                false
#  define CONF_CLOCK_GCLK_3_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_3_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_3_PRESCALER             1
#  define CONF_CLOCK_GCLK_3_OUTPUT_ENABLE         false

/* GCLK発振器4構成設定 */
#  define CONF_CLOCK_GCLK_4_ENABLE                false
#  define CONF_CLOCK_GCLK_4_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_4_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_4_PRESCALER             1
#  define CONF_CLOCK_GCLK_4_OUTPUT_ENABLE         false

/* GCLK発振器5構成設定 */
#  define CONF_CLOCK_GCLK_5_ENABLE                false
#  define CONF_CLOCK_GCLK_5_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_5_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_5_PRESCALER             1
#  define CONF_CLOCK_GCLK_5_OUTPUT_ENABLE         false

/* GCLK発振器6構成設定 */
#  define CONF_CLOCK_GCLK_6_ENABLE                false
#  define CONF_CLOCK_GCLK_6_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_6_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_6_PRESCALER             1
#  define CONF_CLOCK_GCLK_6_OUTPUT_ENABLE         false

/* GCLK発振器7構成設定 */
#  define CONF_CLOCK_GCLK_7_ENABLE                false
#  define CONF_CLOCK_GCLK_7_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_7_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_7_PRESCALER             1
#  define CONF_CLOCK_GCLK_7_OUTPUT_ENABLE         false

/* GCLK発振器8構成設定 */
#  define CONF_CLOCK_GCLK_8_ENABLE                false
#  define CONF_CLOCK_GCLK_8_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_8_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_8_PRESCALER             1
#  define CONF_CLOCK_GCLK_8_OUTPUT_ENABLE         false

#endif /* CONF_CLOCKS_H_INCLUDED */
```

4.3.4. conf_board.h

4.3.4.1. AT32UC3A0, AT32UC3A1, AT32UC3Bﾃﾞﾊﾞｲｽ (USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// この例については既定基板初期化(ｽｲｯﾁ/LED)だけが必要です。

#endif /* CONF_BOARD_H_INCLUDED */
```

4.3.4.2. AT32UC3C, ATUCXXD, ATUCXXL3U, ATUCXXL4Uﾃﾞﾊﾞｲｽ (USBC)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// USBﾎﾟｰﾄ許可
#define CONF_BOARD_USB_PORT

#endif /* CONF_BOARD_H_INCLUDED */
```

4.3.4.3. SAM3X, SAM3Aﾃﾞﾊﾞｲｽ (UOTGHS:USB OTG 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// USBﾋﾟﾝが使われます。
#define CONF_BOARD_USB_PORT

#endif /* CONF_BOARD_H_INCLUDED */
```

4.3.4.4. SAM D21ﾃﾞﾊﾞｲｽ (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

/* USB VBUS検出許可 */
#define CONF_BOARD_USB_VBUS_DETECT

#endif /* CONF_BOARD_H_INCLUDED */
```

5. 人間インターフェース装置マウス(HIDマウス)用USB装置インターフェース(UDI)

人間インターフェース装置マウス(HIDマウス)用USB装置インターフェース(UDI)はUSB HIDマウス装置の構成設定と管理に関するインターフェースを提供します。

この資料の概要は以下のとおりです。

- API概要
- USB装置マウス単位部(UDIマウス)用の即時開始の手引き
- 構成設定ファイル例

Atmel®ソフトウェア枠組み(ASF) USB装置階層とUSB装置HIDマウスのより多くの詳細については以下の応用記述を参照してください。

- AVR4900：ASF - USB装置階層
- AVR4903：ASF - USB装置HIDマウス応用
- AVR4920：ASF - USB装置階層 - 適合と性能係数
- AVR4921：ASF - USB装置階層 - ASF V1とV2間の違い

5.1. API概要

5.1.1. 変数と型定義

5.1.1.1. USB装置コア(UDC)とのインターフェース

UDCによって必要とされる構造体

udi_api_hid_mouse変数

```
UDC_DESC_STORAGE      udi_api_t udi_api_hid_mouse
```

UDC用の標準UDI APIを含む全域構造体

5.1.2. 構造体定義

5.1.2.1. udi_hid_mouse_desc_t構造体

HIDマウス用インターフェース記述子構造体

表5-1. メンバ

型	名前	説明
usb_ep_desc_t	ep	標準USBエンドポイント記述子構造体
usb_hid_descriptor_t	hid	HID記述子
usb_iface_desc_t	iface	標準USBインターフェース記述構造体

5.1.2.2. udi_hid_mouse_report_desc_t構造体

HIDマウス用報告記述子

表5-2. メンバ

型	名前	説明
uint8_t	array[]	詳細報告データを置く配列

5.1.3. マクロ定義

5.1.3.1. USBインターフェース記述子

以下の構造体は予め定義されたUSBインターフェース記述子を提供します。これは最終的なUSB記述子を定義するのに使われなければなりません。

UDI_HID_MOUSE_STRING_ID マクロ

```
#define UDI_HID_MOUSE_STRING_ID
```

既定によってこのインターフェースに関連する文字列はなし

UDI_HID_MOUSE_EP_SIZE マクロ

```
#define UDI_HID_MOUSE_EP_SIZE
```

HIDマウスのエンドポイントの大きさ

UDI_HID_MOUSE_DESC マクロ

```
#define UDI_HID_MOUSE_DESC
```

全ての速度に対するHIDマスス インターフェース記述子の内容

5.1.3.2. 釦事象用インターフェース

HID_MOUSE_BTN_DOWN マクロ

```
#define HID_MOUSE_BTN_DOWN
```

釦下降(押下)を示す値

HID_MOUSE_BTN_UP マクロ

```
#define HID_MOUSE_BTN_UP
```

釦上昇(解放)を示す値

5.1.4. 関数定義

5.1.4.1. マウス事象用インターフェース

udi_hid_mouse_moveScroll()関数

回転輪移動

```
bool udi_hid_mouse_moveScroll( int8_t pos )
```

表5-3. パラメータ

パラメータ名	データ方向	説明
pos	[入力]	移動するための符号付き値

戻り値：関数が成功裏に終了した場合に1、さもなければ0

udi_hid_mouse_moveY()関数

Y軸上のマウス ポインタ移動

```
bool udi_hid_mouse_moveY( int8_t pos_y )
```

表5-4. パラメータ

パラメータ名	データ方向	説明
pos_y	[入力]	移動するための符号付き値

戻り値：関数が成功裏に終了した場合に1、さもなければ0

udi_hid_mouse_moveX()関数

X軸上のマウス ポインタ移動

```
bool udi_hid_mouse_moveX( int8_t pos_x )
```

表5-5. パラメータ

パラメータ名	データ方向	説明
pos_x	[入力]	移動するための符号付き値

戻り値：関数が成功裏に終了した場合に1、さもなければ0

5.1.4.2. 釦事象用インターフェース

udi_hid_mouse_btnmiddle()関数

中央釦状態変化

```
bool udi_hid_mouse_btnmiddle( bool b_state )
```

表5-6. パラメータ

パラメータ名	データ方向	説明
b_state	[入力]	新しい釦の状態

戻り値：関数が成功裏に終了した場合に1、さもなければ0

udi_hid_mouse_btnright()関数

右釦状態変化

```
bool udi_hid_mouse_btnright( bool b_state )
```

表5-7. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
b_state	[入力]	新しい釦の状態

戻り値：関数が成功裏に終了した場合に1、さもなければ0

udi_hid_mouse_btnleft()関数

左釦状態変化

```
bool udi_hid_mouse_btnleft( bool b_state )
```

表5-8. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
b_state	[入力]	新しい釦の状態

戻り値：関数が成功裏に終了した場合に1、さもなければ0

5.2. USB装置ﾏｳｽ単位部(UDIﾏｳｽ)用の即時開始の手引き

これは使用事例の選択に於いて単位部をどう構成設定して使うかを段階的に指示する、USB装置ﾏｳｽ単位部(UDIﾏｳｽ)用の即時開始の手引きです。

使用事例は様々なｺｰﾄﾞの断片を含みます。構成設定に関する段階でのｺｰﾄﾞの断片は独自初期化関数内に複写することができ、一方使用に関する段階は例えば主応用関数内に複写することができます。

5.2.1. 基本的な使用事例

この使用事例では"USB HID Mouse (Single Interface Device)"単位部が使われます。"USB HID Mouse (Composite Device)"単位部の使い方は高度な使用事例で記述されます。

5.2.1.1. 構成設定段階

USB装置のため、共通USB装置構成設定段階に従います。USB装置基本構成設定を参照してください。

5.2.1.2. 使用段階

ｺｰﾄﾞ例

conf_usb.hの内容

```
#define UDI_HID_MOUSE_ENABLE_EXT() my_callback_mouse_enable()
extern bool my_callback_mouse_enable(void);
#define UDI_HID_MOUSE_DISABLE_EXT() my_callback_mouse_disable()
extern void my_callback_mouse_disable(void);
#include "udi_hid_mouse_conf.h" // conf_usb.hﾌｧｲﾙの最後で
```

応用Cﾌｧｲﾙに追加してください。

```
static bool my_flag_autorize_mouse_events = false;
bool my_callback_mouse_enable(void)
{
   my_flag_autorize_mouse_events = true;
   return true;
}
void my_callback_mouse_disable(void)
{
   my_flag_autorize_mouse_events = false;
}

void my_button_press_event(void)
{
   if (!my_flag_autorize_mouse_events) {
      return;
   }
   udi_hid_mouse_btnleft(HID_MOUSE_BTN_DOWN);
}
```

作業の流れ

1. conf_usb.hが利用可能でUSB装置ﾏｳｽ構成設定である以下の構成設定を含むことを確実にしてください。

```
#define UDI_HID_MOUSE_ENABLE_EXT() my_callback_mouse_enable()
extern bool my_callback_mouse_enable(void);
```

注：装置列挙(USB装置の検出と識別)後、USBホストは装置の構成設定を始めます。装置からのUSBマウス インターフェースがホストによって受け入れられると、USBホストはこのインターフェースを許可してUDI_HID_MOUSE_ENABLE_EXT()呼び戻し関数が呼ばれ、trueが返ります。故に、この関数でマウスによって使われる感知器を許可することが推奨されます。

```
#define UDI_HID_MOUSE_DISABLE_EXT() my_callback_mouse_disable()
extern void my_callback_mouse_disable(void);
```

注：USB装置が切断されるか、またはUSBホストによってリセットされると、USBインターフェースは禁止され、UDI_HID_MOUSE_DISABLE_EXT()呼び戻し関数が呼ばれます。故に、この関数でマウスによって使われる感知器を禁止することが推奨されます。

2. マウス事象送出

```
// 回転輪での値を送出
udi_hid_mouse_moveScroll(int8_t pos);
// マウス ポインタでのY軸値を送出
udi_hid_mouse_moveY(int8_t pos_y);
// マウス ポインタでのX軸値を送出
udi_hid_mouse_moveX(int8_t pos_x);
// 中央釦クリック事象を送出
udi_hid_mouse_btnmiddle(bool b_state);
// 右釦クリック事象を送出
udi_hid_mouse_btnright(bool b_state);
// 左釦クリック事象を送出
udi_hid_mouse_btnleft(bool b_state);
```

5.2.2. 高度な使用事例

UDI HID単位部の複数インターフェースの使用については以下をご覧ください。

・複合装置でのHIDマウス

UDI HIDマウス単位部のもっと高度な使用については以下をご覧ください。

・USB装置の高度な使用事例

5.2.3. 複合装置でのHIDマウス

USB複合装置は1つよりも多くのUSBクラスを使うUSB装置です。この使用事例ではUSB複合装置を作成するのに”USB HID Mouse (Composite Device)”単位部が使われます。故に、このUSB単位部は”USB MSC (Composite Device)”のような別な”複合装置(Composite Device)”単位部と連携することができます。

また、「AVR4902:ASF – USB複合装置」応用記述を参照することもできます。

5.2.3.1. 構成設定段階

動くためのこの事例の構成設定ｺｰﾄﾞについては基本的な使用事例に従わなければなりません。

5.2.3.2. 使用段階

ｺｰﾄﾞ例

conf_usb.hの内容

```
#define USB_DEVICE_EP_CTRL_SIZE  64
#define USB_DEVICE_NB_INTERFACE (X+1)
#define USB_DEVICE_MAX_EP (X+1)

#define UDI_HID_MOUSE_EP_IN  (X | USB_EP_DIR_IN)
#define UDI_HID_MOUSE_IFACE_NUMBER  X

#define UDI_COMPOSITE_DESC_T ¥
   udi_hid_mouse_desc_t udi_hid_mouse; ¥
   ～
#define UDI_COMPOSITE_DESC_FS ¥
   .udi_hid_mouse = UDI_HID_MOUSE_DESC, ¥
   ～
#define UDI_COMPOSITE_DESC_HS ¥
   .udi_hid_mouse = UDI_HID_MOUSE_DESC, ¥
   ～
#define UDI_COMPOSITE_API ¥
```

```
   &udi_api_hid_mouse, ¥
   ～
```

作業の流れ

1. conf_usb.hが利用可能でUSB複合装置構成設定に必要とされる以下のﾊﾟﾗﾒｰﾀを含むことを確実にしてください。

```
// ｴﾝﾄﾞﾎﾟｲﾝﾄ制御容量、これは以下でなければなりません。
// - 低速(LS)装置に対して、8
// - 全速(FS)装置に対して、8, 16 ,32 または 64 (RAMを節約するために8が推奨されます。)
// - 高速(HS)装置に対して、64
#define USB_DEVICE_EP_CTRL_SIZE   64
// このUSB装置での総ｲﾝﾀｰﾌｪｰｽ数
// HIDﾏｳｽ用に1を加算
#define USB_DEVICE_NB_INTERFACE  (X+1)
// このUSB装置での総ｴﾝﾄﾞﾎﾟｲﾝﾄ数
// これには各ｲﾝﾀｰﾌｪｰｽに対する各々のｴﾝﾄﾞﾎﾟｲﾝﾄを含めなければなりません。
// HIDﾏｳｽ用に1を加算
#define USB_DEVICE_MAX_EP  (X+1)
```

2. conf_usb.hが複合装置の記述子を含むことを確実にしてください。

```
// あなたが選んだﾏｳｽ用のｴﾝﾄﾞﾎﾟｲﾝﾄ番号
// ｴﾝﾄﾞﾎﾟｲﾝﾄ番号は1から始まります。
#define UDI_HID_MOUSE_EP_IN   (X | USB_EP_DIR_IN)
// 0から始まるｲﾝﾀｰﾌｪｰｽのｲﾝﾀｰﾌｪｰｽ指標
#define UDI_HID_MOUSE_IFACE_NUMBER   X
```

3. conf_usb.hがUSB複合装置構成設定に必要とされる以下のパラメータを含むことを確実にしてください。

```
// USBｲﾝﾀｰﾌｪｰｽ記述子構造体
#define UDI_COMPOSITE_DESC_T ¥
   ～
   udi_hid_mouse_desc_t udi_hid_mouse; ¥
   ～
// 全速(FS)用USBｲﾝﾀｰﾌｪｰｽ記述子値
#define UDI_COMPOSITE_DESC_FS ¥
   ～
   .udi_hid_mouse = UDI_HID_MOUSE_DESC, ¥
   ～
// 高速(HS)用USBｲﾝﾀｰﾌｪｰｽ記述子値
#define UDI_COMPOSITE_DESC_HS ¥
   ～
   .udi_hid_mouse = UDI_HID_MOUSE_DESC, ¥
   ～
// USBｲﾝﾀｰﾌｪｰｽAPI
#define UDI_COMPOSITE_API ¥
   ～
   &udi_api_hid_mouse, ¥
   ～
```

注：上の4つの一覧で与えられた記述子の順番は全てのｲﾝﾀｰﾌｪｰｽ指標によって定義された順番と同じでなければなりません。ｲﾝﾀｰﾌｪｰｽ指標の順番はUDI_X_IFACE_NUMBER定義を通して定義されます。

5.3. 構成設定ﾌｧｲﾙ例

5.3.1. conf_usb.h

5.3.1.1. 単一UDI HID MOUSE

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_USB_H_
#define _CONF_USB_H_

#include "compiler.h"
```

```
#warning You must refill the following definitions with a correct values

#define  USB_DEVICE_VENDOR_ID             USB_VID_ATMEL
#define  USB_DEVICE_PRODUCT_ID            USB_PID_ATMEL_ASF_HIDMOUSE
#define  USB_DEVICE_MAJOR_VERSION         1
#define  USB_DEVICE_MINOR_VERSION         0
#define  USB_DEVICE_POWER                 100 // VBUS線での消費(mA)
#define  USB_DEVICE_ATTR                  ¥
    (USB_CONFIG_ATTR_SELF_POWERED)
// (USB_CONFIG_ATTR_BUS_POWERED)
//   (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_SELF_POWERED)
//   (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_BUS_POWERED)

// #define  USB_DEVICE_MANUFACTURE_NAME      "Manufacture name"
// #define  USB_DEVICE_PRODUCT_NAME          "Product name"
// #define  USB_DEVICE_SERIAL_NAME           "12...EF"

//#define  USB_DEVICE_LOW_SPEED

#if (UC3A3||UC3A4)
//#define  USB_DEVICE_HS_SUPPORT
#endif

// #define  UDC_VBUS_EVENT(b_vbus_high)      user_callback_vbus_action(b_vbus_high)
// extern void user_callback_vbus_action(bool b_vbus_high);
// #define  UDC_SOF_EVENT()                  user_callback_sof_action()
// extern void user_callback_sof_action(void);
// #define  UDC_SUSPEND_EVENT()              user_callback_suspend_action()
// extern void user_callback_suspend_action(void);
// #define  UDC_RESUME_EVENT()               user_callback_resume_action()
// extern void user_callback_resume_action(void);
// #define  UDC_REMOTEWAKEUP_ENABLE()        user_callback_remotewakeup_enable()
// extern void user_callback_remotewakeup_enable(void);
// #define  UDC_REMOTEWAKEUP_DISABLE()       user_callback_remotewakeup_disable()
// extern void user_callback_remotewakeup_disable(void);
// #define  UDC_GET_EXTRA_STRING()

#define  UDI_HID_MOUSE_ENABLE_EXT()       true
#define  UDI_HID_MOUSE_DISABLE_EXT()
// #define UDI_HID_MOUSE_ENABLE_EXT() my_callback_mouse_enable()
// extern bool my_callback_mouse_enable(void);
// #define UDI_HID_MOUSE_DISABLE_EXT() my_callback_mouse_disable()
// extern void my_callback_mouse_disable(void);

#include "udi_hid_mouse_conf.h"

#endif // _CONF_USB_H_
```

5.3.1.2. 複数UDI HID MOUSE（複合）

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_USB_H_
#define _CONF_USB_H_

#include "compiler.h"

#warning You must refill the following definitions with a correct values
```

```
#define  USB_DEVICE_VENDOR_ID             USB_VID_ATMEL
#define  USB_DEVICE_PRODUCT_ID            0xFFFF
#define  USB_DEVICE_MAJOR_VERSION         1
#define  USB_DEVICE_MINOR_VERSION         0
#define  USB_DEVICE_POWER                 100 // VBUS線での消費(mA)
#define  USB_DEVICE_ATTR                  ¥
         (USB_CONFIG_ATTR_SELF_POWERED)
// (USB_CONFIG_ATTR_BUS_POWERED)
// (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_SELF_POWERED)
// (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_BUS_POWERED)

// #define  USB_DEVICE_MANUFACTURE_NAME      "Manufacture name"
// #define  USB_DEVICE_PRODUCT_NAME          "Product name"
// #define  USB_DEVICE_SERIAL_NAME           "12...EF" // MSC用ﾃﾞｨｽｸ通番

//#define  USB_DEVICE_LOW_SPEED

#if (UC3A3||UC3A4)
//#define  USB_DEVICE_HS_SUPPORT
#endif

// #define  UDC_VBUS_EVENT(b_vbus_high)      user_callback_vbus_action(b_vbus_high)
// extern void user_callback_vbus_action(bool b_vbus_high);
// #define  UDC_SOF_EVENT()                  user_callback_sof_action()
// extern void user_callback_sof_action(void);
// #define  UDC_SUSPEND_EVENT()              user_callback_suspend_action()
// extern void user_callback_suspend_action(void);
// #define  UDC_RESUME_EVENT()               user_callback_resume_action()
// extern void user_callback_resume_action(void);
// #define  UDC_REMOTEWAKEUP_ENABLE()        user_callback_remotewakeup_enable()
// extern void user_callback_remotewakeup_enable(void);
// #define  UDC_REMOTEWAKEUP_DISABLE()       user_callback_remotewakeup_disable()
// extern void user_callback_remotewakeup_disable(void);
// #define  UDC_GET_EXTRA_STRING()

#define  USB_DEVICE_EP_CTRL_SIZE       64

#define  USB_DEVICE_NB_INTERFACE       1 // 1またはそれ以上

#define  USB_DEVICE_MAX_EP             1 // 0～ｲﾝﾀｰﾌｪｰｽによって要求された最大ｴﾝﾄﾞﾎﾟｲﾝﾄ

#define  UDI_CDC_PORT_NB 1

#define  UDI_CDC_ENABLE_EXT(port)              true
#define  UDI_CDC_DISABLE_EXT(port)
#define  UDI_CDC_RX_NOTIFY(port)
#define  UDI_CDC_TX_EMPTY_NOTIFY(port)
#define  UDI_CDC_SET_CODING_EXT(port,cfg)
#define  UDI_CDC_SET_DTR_EXT(port,set)
#define  UDI_CDC_SET_RTS_EXT(port,set)
/*
 * #define UDI_CDC_ENABLE_EXT(port) my_callback_cdc_enable()
 * extern bool my_callback_cdc_enable(void);
 * #define UDI_CDC_DISABLE_EXT(port) my_callback_cdc_disable()
 * extern void my_callback_cdc_disable(void);
 * #define  UDI_CDC_RX_NOTIFY(port) my_callback_rx_notify(port)
 * extern void my_callback_rx_notify(uint8_t port);
 * #define  UDI_CDC_TX_EMPTY_NOTIFY(port) my_callback_tx_empty_notify(port)
 * extern void my_callback_tx_empty_notify(uint8_t port);
 * #define  UDI_CDC_SET_CODING_EXT(port,cfg) my_callback_config(port,cfg)
 * extern void my_callback_config(uint8_t port, usb_cdc_line_coding_t * cfg);
```

```
 * #define  UDI_CDC_SET_DTR_EXT(port,set) my_callback_cdc_set_dtr(port,set)
 * extern void my_callback_cdc_set_dtr(uint8_t port, bool b_enable);
 * #define  UDI_CDC_SET_RTS_EXT(port,set) my_callback_cdc_set_rts(port,set)
 * extern void my_callback_cdc_set_rts(uint8_t port, bool b_enable);
 */

#define  UDI_CDC_LOW_RATE

#define  UDI_CDC_DEFAULT_RATE             115200
#define  UDI_CDC_DEFAULT_STOPBITS         CDC_STOP_BITS_1
#define  UDI_CDC_DEFAULT_PARITY           CDC_PAR_NONE
#define  UDI_CDC_DEFAULT_DATABITS         8

#define  UDI_CDC_DATA_EP_IN_0          (1 | USB_EP_DIR_IN)  // 送信
#define  UDI_CDC_DATA_EP_OUT_0         (2 | USB_EP_DIR_OUT) // 受信
#define  UDI_CDC_COMM_EP_0             (3 | USB_EP_DIR_IN)  // 通知ｴﾝﾄﾞﾎﾟｲﾝﾄ
#define  UDI_CDC_DATA_EP_IN_2          (4 | USB_EP_DIR_IN)  // 送信
#define  UDI_CDC_DATA_EP_OUT_2         (5 | USB_EP_DIR_OUT) // 受信
#define  UDI_CDC_COMM_EP_2             (6 | USB_EP_DIR_IN)  // 通知ｴﾝﾄﾞﾎﾟｲﾝﾄ
#define  UDI_CDC_DATA_EP_IN_3          (7 | USB_EP_DIR_IN)  // 送信
#define  UDI_CDC_DATA_EP_OUT_3         (8 | USB_EP_DIR_OUT) // 受信
#define  UDI_CDC_COMM_EP_3             (9 | USB_EP_DIR_IN)  // 通知ｴﾝﾄﾞﾎﾟｲﾝﾄ

#define  UDI_CDC_COMM_IFACE_NUMBER_0   0
#define  UDI_CDC_DATA_IFACE_NUMBER_0   1
#define  UDI_CDC_COMM_IFACE_NUMBER_2   2
#define  UDI_CDC_DATA_IFACE_NUMBER_2   3
#define  UDI_CDC_COMM_IFACE_NUMBER_3   4
#define  UDI_CDC_DATA_IFACE_NUMBER_3   5

#define UDI_MSC_GLOBAL_VENDOR_ID            ¥
   'A', 'T', 'M', 'E', 'L', ' ', ' ', ' '
#define UDI_MSC_GLOBAL_PRODUCT_VERSION           ¥
   '1', '.', '0', '0'

#define  UDI_MSC_ENABLE_EXT()          true
#define  UDI_MSC_DISABLE_EXT()
#define  UDI_MSC_NOTIFY_TRANS_EXT()
/*
 * #define UDI_MSC_ENABLE_EXT() my_callback_msc_enable()
 * extern bool my_callback_msc_enable(void);
 * #define UDI_MSC_DISABLE_EXT() my_callback_msc_disable()
 * extern void my_callback_msc_disable(void);
 * #define  UDI_MSC_NOTIFY_TRANS_EXT()    msc_notify_trans()
 * extern void msc_notify_trans(void) {
 */

#define  UDI_MSC_EP_IN                 (1 | USB_EP_DIR_IN)
#define  UDI_MSC_EP_OUT                (2 | USB_EP_DIR_OUT)

#define  UDI_MSC_IFACE_NUMBER          0

#define  UDI_HID_MOUSE_ENABLE_EXT()       true
#define  UDI_HID_MOUSE_DISABLE_EXT()
// #define UDI_HID_MOUSE_ENABLE_EXT() my_callback_mouse_enable()
// extern bool my_callback_mouse_enable(void);
// #define UDI_HID_MOUSE_DISABLE_EXT() my_callback_mouse_disable()
// extern void my_callback_mouse_disable(void);

#define  UDI_HID_MOUSE_EP_IN           (1 | USB_EP_DIR_IN)
```

```
#define  UDI_HID_MOUSE_IFACE_NUMBER    0

#define  UDI_HID_KBD_ENABLE_EXT()       true
#define  UDI_HID_KBD_DISABLE_EXT()
// #define UDI_HID_KBD_ENABLE_EXT() my_callback_keyboard_enable()
// extern bool my_callback_keyboard_enable(void);
// #define UDI_HID_KBD_DISABLE_EXT() my_callback_keyboard_disable()
// extern void my_callback_keyboard_disable(void);
#define  UDI_HID_KBD_CHANGE_LED(value)
// #define  UDI_HID_KBD_CHANGE_LED(value) my_callback_keyboard_led(value)
// extern void my_callback_keyboard_led(uint8_t value)

#define  UDI_HID_KBD_EP_IN           (1 | USB_EP_DIR_IN)

#define  UDI_HID_KBD_IFACE_NUMBER    0

#define  UDI_HID_GENERIC_ENABLE_EXT()       true
#define  UDI_HID_GENERIC_DISABLE_EXT()
#define  UDI_HID_GENERIC_REPORT_OUT(ptr)
#define  UDI_HID_GENERIC_SET_FEATURE(f)
/*
 * #define UDI_HID_GENERIC_ENABLE_EXT() my_callback_generic_enable()
 * extern bool my_callback_generic_enable(void);
 * #define UDI_HID_GENERIC_DISABLE_EXT() my_callback_generic_disable()
 * extern void my_callback_generic_disable(void);
 * #define  UDI_HID_GENERIC_REPORT_OUT(ptr) my_callback_generic_report_out(ptr)
 * extern void my_callback_generic_report_out(uint8_t *report);
 * #define  UDI_HID_GENERIC_SET_FEATURE(f) my_callback_generic_set_feature(f)
 * extern void my_callback_generic_set_feature(uint8_t *report_feature);
 */
#define  UDI_HID_REPORT_IN_SIZE             64
#define  UDI_HID_REPORT_OUT_SIZE            64
#define  UDI_HID_REPORT_FEATURE_SIZE        4
#define  UDI_HID_GENERIC_EP_SIZE            64

#define  UDI_HID_GENERIC_EP_OUT    (2 | USB_EP_DIR_OUT)
#define  UDI_HID_GENERIC_EP_IN     (1 | USB_EP_DIR_IN)

#define  UDI_HID_GENERIC_IFACE_NUMBER    0

#define  UDI_PHDC_ENABLE_EXT()          true
#define  UDI_PHDC_DISABLE_EXT()

#define  UDI_PHDC_DATAMSG_FORMAT       USB_PHDC_DATAMSG_FORMAT_11073_20601
#define  UDI_PHDC_SPECIALIZATION       {0x2345} // 11073_20601での定義

#define  UDI_PHDC_QOS_OUT              ¥
        (USB_PHDC_QOS_MEDIUM_BETTER|USB_PHDC_QOS_HIGH_BEST)
#define  UDI_PHDC_QOS_IN               ¥
        (USB_PHDC_QOS_LOW_GOOD|USB_PHDC_QOS_MEDIUM_BETTER|USB_PHDC_QOS_MEDIUM_BEST)

#define  UDI_PHDC_METADATA_DESC_BULK_IN    {0x01,0x02,0x03}
#define  UDI_PHDC_METADATA_DESC_BULK_OUT   {0x01,0x02,0x03}
#define  UDI_PHDC_METADATA_DESC_INT_IN     {0x01,0x02,0x03}

#define  UDI_PHDC_EP_BULK_OUT          (1 | USB_EP_DIR_OUT)
#define  UDI_PHDC_EP_BULK_IN           (2 | USB_EP_DIR_IN)
#if ((UDI_PHDC_QOS_IN&USB_PHDC_QOS_LOW_GOOD)==USB_PHDC_QOS_LOW_GOOD)
// UDI_PHDC_QOS_INの場合にだけUSB_PHDC_QOS_LOW_GOODをｲﾝｸﾙｰﾄﾞ
#  define  UDI_PHDC_EP_INTERRUPT_IN      (3 | USB_EP_DIR_IN)
#endif
```

```
#define  UDI_PHDC_EP_SIZE_BULK_OUT     32
#define  UDI_PHDC_EP_SIZE_BULK_IN      32
#define  UDI_PHDC_EP_SIZE_INT_IN       8

#define  UDI_PHDC_IFACE_NUMBER          0

#define  UDI_VENDOR_ENABLE_EXT()          true
#define  UDI_VENDOR_DISABLE_EXT()
#define  UDI_VENDOR_SETUP_OUT_RECEIVED()  false
#define  UDI_VENDOR_SETUP_IN_RECEIVED()   false
/*
 * #define UDI_VENDOR_ENABLE_EXT() my_callback_vendor_enable()
 * extern bool my_callback_vendor_enable(void);
 * #define UDI_VENDOR_DISABLE_EXT() my_callback_vendor_disable()
 * extern void my_callback_vendor_disable(void);
 *
 * #define  UDI_VENDOR_SETUP_OUT_RECEIVED()  my_vendor_setup_out_received()
 * extern bool my_vendor_setup_out_received(void);
 * #define  UDI_VENDOR_SETUP_IN_RECEIVED()   my_vendor_setup_in_received()
 * extern bool my_vendor_setup_in_received(void);
 */

#define  UDI_VENDOR_EPS_SIZE_INT_FS    64
#define  UDI_VENDOR_EPS_SIZE_BULK_FS   64
#define  UDI_VENDOR_EPS_SIZE_ISO_FS   256

#define  UDI_VENDOR_EPS_SIZE_INT_HS    64
#define  UDI_VENDOR_EPS_SIZE_BULK_HS  512
#define  UDI_VENDOR_EPS_SIZE_ISO_HS    64

#define  UDI_VENDOR_EP_INTERRUPT_IN  (1 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_INTERRUPT_OUT (2 | USB_EP_DIR_OUT)
#define  UDI_VENDOR_EP_BULK_IN       (3 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_BULK_OUT      (4 | USB_EP_DIR_OUT)
#define  UDI_VENDOR_EP_ISO_IN        (5 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_ISO_OUT       (6 | USB_EP_DIR_OUT)

#define  UDI_VENDOR_IFACE_NUMBER     0

//... 最終的に他のｲﾝﾀｰﾌｪｰｽ構成設定を追加してください。

#define UDI_COMPOSITE_DESC_T

#define UDI_COMPOSITE_DESC_FS

#define UDI_COMPOSITE_DESC_HS

#define UDI_COMPOSITE_API

/* CDC、MSC、HIDﾏｳｽ ｲﾝﾀｰﾌｪｰｽを持つ装置の例
#define UDI_COMPOSITE_DESC_T ¥
    usb_iad_desc_t udi_cdc_iad; ¥
    udi_cdc_comm_desc_t udi_cdc_comm; ¥
    udi_cdc_data_desc_t udi_cdc_data; ¥
    udi_msc_desc_t udi_msc; ¥
    udi_hid_mouse_desc_t udi_hid_mouse

#define UDI_COMPOSITE_DESC_FS ¥
    .udi_cdc_iad               = UDI_CDC_IAD_DESC_0, ¥
    .udi_cdc_comm              = UDI_CDC_COMM_DESC_0, ¥
```

```
    .udi_cdc_data               = UDI_CDC_DATA_DESC_0_FS, ¥
    .udi_msc                    = UDI_MSC_DESC_FS, ¥
    .udi_hid_mouse              = UDI_HID_MOUSE_DESC

#define UDI_COMPOSITE_DESC_HS ¥
    .udi_cdc_iad                = UDI_CDC_IAD_DESC_0, ¥
    .udi_cdc_comm               = UDI_CDC_COMM_DESC_0, ¥
    .udi_cdc_data               = UDI_CDC_DATA_DESC_0_HS, ¥
    .udi_msc                    = UDI_MSC_DESC_HS, ¥
    .udi_hid_mouse              = UDI_HID_MOUSE_DESC

#define UDI_COMPOSITE_API  ¥
    &udi_api_cdc_comm,        ¥
    &udi_api_cdc_data,        ¥
    &udi_api_msc,             ¥
    &udi_api_hid_mouse
*/

/* インターフェース用インクルードの例
#include "udi_msc.h"
#include "udi_hid_kbd.h"
#include "udi_hid_mouse.h"
#include "udi_cdc.h"
#include "udi_phdc.h"
#include "udi_vendor.h"
*/
/* USBによって使われる呼び戻しの宣言
#include "callback_def.h"*/

#endif // _CONF_USB_H_
```

5.3.2. conf_clock.h

5.3.2.1. AT32UC3A0, ATUC3A1, ATUC3Bデバイス (USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== システム クロック元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RCSYS
#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_OSC0
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLL0

// ===== PLL0任意選択
//#define CONFIG_PLL0_SOURCE          PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL0_MUL             4 /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL0_DIV             1 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== PLL1任意選択
#define CONFIG_PLL1_SOURCE            PLL_SRC_OSC0
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC1
#define CONFIG_PLL1_MUL               8 /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL1_DIV               2 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== システム クロック バス分周任意選択
//#define CONFIG_SYSCLK_CPU_DIV         0 /* Fcpu = Fsys/(2 ^ CPU_div) */
//#define CONFIG_SYSCLK_PBA_DIV         0 /* Fpba = Fsys/(2 ^ PBA_div) */
//#define CONFIG_SYSCLK_PBB_DIV         0 /* Fpbb = Fsys/(2 ^ PBB_div) */
```

```
// ===== 周辺機能ｸﾛｯｸ管理任意選択
//#define CONFIG_SYSCLK_INIT_CPUMASK  ((1 << SYSCLK_SYSTIMER) | (1 << SYSCLK_OCD))
//#define CONFIG_SYSCLK_INIT_PBAMASK  (1 << SYSCLK_USART0)
//#define CONFIG_SYSCLK_INIT_PBBMASK  (1 << SYSCLK_HMATRIX)
//#define CONFIG_SYSCLK_INIT_HSBMASK  (1 << SYSCLK_MDMA_HSB)

// ===== USBｸﾛｯｸ元任意選択
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_OSC0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL0
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV             1 /* Fusb = Fsys/(2 ^ USB_div) */

#endif /* CONF_CLOCK_H_INCLUDED */
```

5.3.2.2. AT32UC3A3, AT32UC3A4ﾃﾞﾊﾞｲｽ (高速(HS)支援付USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== ｼｽﾃﾑ ｸﾛｯｸ元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RCSYS
#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_OSC0
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLL0

// ===== PLL0任意選択
//#define CONFIG_PLL0_SOURCE          PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL0_MUL             11 /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL0_DIV             2  /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== PLL1任意選択
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC0
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL1_MUL             8 /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL1_DIV             2 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== ｼｽﾃﾑ ｸﾛｯｸ ﾊﾞｽ分周任意選択
#define CONFIG_SYSCLK_CPU_DIV         0 /* Fcpu = Fsys/(2 ^ CPU_div) */
#define CONFIG_SYSCLK_PBA_DIV         0 /* Fpba = Fsys/(2 ^ PBA_div) */
//#define CONFIG_SYSCLK_PBB_DIV       0 /* Fpbb = Fsys/(2 ^ PBB_div) */

// ===== 周辺機能ｸﾛｯｸ管理任意選択
//#define CONFIG_SYSCLK_INIT_CPUMASK  ((1 << SYSCLK_SYSTIMER) | (1 << SYSCLK_OCD))
//#define CONFIG_SYSCLK_INIT_PBAMASK  (1 << SYSCLK_USART0)
//#define CONFIG_SYSCLK_INIT_PBBMASK  (1 << SYSCLK_HMATRIX)
//#define CONFIG_SYSCLK_INIT_HSBMASK  (1 << SYSCLK_MDMA_HSB)

// ===== USBｸﾛｯｸ元任意選択
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_OSC0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV             1 /* Fusb = Fsys/(2 ^ USB_div) */

#endif /* CONF_CLOCK_H_INCLUDED */
```

5.3.2.3. AT32UC3C, ATUCXXD, ATUCXXL3U, ATUCXXL4Uﾃﾞﾊﾞｲｽ (USBC)

```
/*
```

```
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== システム クロック元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RCSYS
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_OSC0
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_OSC1
#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_PLL0
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLL1
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RC8M

// ===== PLL0任意選択
#define CONFIG_PLL0_SOURCE            PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL0_SOURCE          PLL_SRC_RC8M
#define CONFIG_PLL0_MUL               3 /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL0_DIV               1 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== PLL1任意選択
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC0
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL1_SOURCE          PLL_SRC_RC8M
//#define CONFIG_PLL1_MUL             3 /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL1_DIV             1 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== システム クロック バス分周任意選択
//#define CONFIG_SYSCLK_CPU_DIV       0 /* Fcpu = Fsys/(2 ^ CPU_div) */
//#define CONFIG_SYSCLK_PBA_DIV       0 /* Fpba = Fsys/(2 ^ PBA_div) */
//#define CONFIG_SYSCLK_PBB_DIV       0 /* Fpbb = Fsys/(2 ^ PBB_div) */
//#define CONFIG_SYSCLK_PBC_DIV       0 /* Fpbc = Fsys/(2 ^ PBC_div) */

// ===== 周辺機能ｸﾛｯｸ管理任意選択
//#define CONFIG_SYSCLK_INIT_CPUMASK  ((1 << SYSCLK_SYSTIMER) | (1 << SYSCLK_OCD))
//#define CONFIG_SYSCLK_INIT_PBAMASK  (1 << SYSCLK_USART0)
//#define CONFIG_SYSCLK_INIT_PBBMASK  (1 << SYSCLK_HMATRIX)
//#define CONFIG_SYSCLK_INIT_HSBMASK  (1 << SYSCLK_MDMA_HSB)

// ===== USBｸﾛｯｸ元任意選択
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_OSC0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_OSC1
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_PLL0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV             1 /* Fusb = Fsys/(2 ^ USB_div) */

#endif /* CONF_CLOCK_H_INCLUDED */
```

5.3.2.4. SAM3X, SAM3Aﾃﾞﾊﾞｲｽ (UOTGHS:USB OTG 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== システム クロック(MCK)元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_BYPASS
```

```
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_4M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_8M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_12M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_BYPASS
#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLLACK
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_UPLLCK

// ===== システム クロック(MCK)前置分周器任意選択    (Fmck = Fsys / (SYSCLK_PRES))
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_1
#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_2
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_4
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_8
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_16
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_32
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_64
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_3

// ===== PLL0 (A)任意選択    (Fpll = (Fclk * PLL_mul) / PLL_div)
// ここにmulとdivの実効値を使ってください。
#define CONFIG_PLL0_SOURCE          PLL_SRC_MAINCK_XTAL
#define CONFIG_PLL0_MUL             14
#define CONFIG_PLL0_DIV             1

// ===== 480MHzで固定化されたUPLL (UTMI)ハードウェア

// ===== USBクロック元任意選択    (Fusb = FpllX / USB_div)
// ここにdivの実効値を使ってください。
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL0
#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_UPLL
#define CONFIG_USBCLK_DIV           1

// ===== 目的対象周波数 (システム クロック)
// - XTAL周波数: 12MHz
// - システム クロック元: PLLA
// - システム クロック前置分周器: 2 (2分周)
// - PLLA供給元: XTAL
// - PLLA出力: XTAL * 14 / 1
// - システム クロックは: 12 * 14 / 1 /2 = 84MHz
// ===== 目的対象周波数 (USBクロック)
// - USBクロック元: UPLL
// - USBクロック分周器: 1 (分周なし)
// - UPLL周波数: 480MHz
// - USBクロック: 480 / 1 = 480MHz

#endif /* CONF_CLOCK_H_INCLUDED */
```

5.3.3. conf_clocks.h

5.3.3.1. SAM D21ﾃﾞﾊﾞｲｽ (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#include <clock.h>

#ifndef CONF_CLOCKS_H_INCLUDED
#  define CONF_CLOCKS_H_INCLUDED

/* システム クロック ﾊﾞｽ構成設定 */
#  define CONF_CLOCK_CPU_CLOCK_FAILURE_DETECT      false
#  define CONF_CLOCK_FLASH_WAIT_STATES             2
```

```
#  define CONF_CLOCK_CPU_DIVIDER                    SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBA_DIVIDER                   SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBB_DIVIDER                   SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBC_DIVIDER                   SYSTEM_MAIN_CLOCK_DIV_1

/* SYSTEM_CLOCK_SOURCE_OSC8M構成設定 - 内部8MHz発振器 */
#  define CONF_CLOCK_OSC8M_PRESCALER                SYSTEM_OSC8M_DIV_1
#  define CONF_CLOCK_OSC8M_ON_DEMAND                true
#  define CONF_CLOCK_OSC8M_RUN_IN_STANDBY           false

/* SYSTEM_CLOCK_SOURCE_XOSC構成設定 - 外部ｸﾛｯｸ/発振器 */
#  define CONF_CLOCK_XOSC_ENABLE                    false
#  define CONF_CLOCK_XOSC_EXTERNAL_CRYSTAL          SYSTEM_CLOCK_EXTERNAL_CRYSTAL
#  define CONF_CLOCK_XOSC_EXTERNAL_FREQUENCY        12000000UL
#  define CONF_CLOCK_XOSC_STARTUP_TIME              SYSTEM_XOSC_STARTUP_32768
#  define CONF_CLOCK_XOSC_AUTO_GAIN_CONTROL         true
#  define CONF_CLOCK_XOSC_ON_DEMAND                 true
#  define CONF_CLOCK_XOSC_RUN_IN_STANDBY            false

/* SYSTEM_CLOCK_SOURCE_XOSC32K構成設定 - 外部32kHzｸﾘｽﾀﾙ/ｸﾛｯｸ発振器 */
#  define CONF_CLOCK_XOSC32K_ENABLE                 false
#  define CONF_CLOCK_XOSC32K_EXTERNAL_CRYSTAL       SYSTEM_CLOCK_EXTERNAL_CRYSTAL
#  define CONF_CLOCK_XOSC32K_STARTUP_TIME           SYSTEM_XOSC32K_STARTUP_65536
#  define CONF_CLOCK_XOSC32K_AUTO_AMPLITUDE_CONTROL  false
#  define CONF_CLOCK_XOSC32K_ENABLE_1KHZ_OUPUT      false
#  define CONF_CLOCK_XOSC32K_ENABLE_32KHZ_OUTPUT    true
#  define CONF_CLOCK_XOSC32K_ON_DEMAND              true
#  define CONF_CLOCK_XOSC32K_RUN_IN_STANDBY         false

/* SYSTEM_CLOCK_SOURCE_OSC32K構成設定 - 内部32kHz発振器 */
#  define CONF_CLOCK_OSC32K_ENABLE                  false
#  define CONF_CLOCK_OSC32K_STARTUP_TIME            SYSTEM_OSC32K_STARTUP_130
#  define CONF_CLOCK_OSC32K_ENABLE_1KHZ_OUTPUT      true
#  define CONF_CLOCK_OSC32K_ENABLE_32KHZ_OUTPUT     true
#  define CONF_CLOCK_OSC32K_ON_DEMAND               true
#  define CONF_CLOCK_OSC32K_RUN_IN_STANDBY          false

/* SYSTEM_CLOCK_SOURCE_DFLL構成設定 - ﾃﾞｼﾞﾀﾙ周波数固定化閉路 */
#  define CONF_CLOCK_DFLL_ENABLE                    true
#  define CONF_CLOCK_DFLL_LOOP_MODE                 SYSTEM_CLOCK_DFLL_LOOP_MODE_USB_RECOVERY
#  define CONF_CLOCK_DFLL_ON_DEMAND                 true

/* DFLL開路動作構成設定 */
#  define CONF_CLOCK_DFLL_FINE_VALUE                (512)

/* DFLL閉路動作構成設定 */
#  define CONF_CLOCK_DFLL_SOURCE_GCLK_GENERATOR     GCLK_GENERATOR_1
#  define CONF_CLOCK_DFLL_MULTIPLY_FACTOR           (48000000 / 32768)
#  define CONF_CLOCK_DFLL_QUICK_LOCK                true
#  define CONF_CLOCK_DFLL_TRACK_AFTER_FINE_LOCK     true
#  define CONF_CLOCK_DFLL_KEEP_LOCK_ON_WAKEUP       true
#  define CONF_CLOCK_DFLL_ENABLE_CHILL_CYCLE        true
#  define CONF_CLOCK_DFLL_MAX_COARSE_STEP_SIZE      (0x1f / 4)
#  define CONF_CLOCK_DFLL_MAX_FINE_STEP_SIZE        (0xff / 4)

/* SYSTEM_CLOCK_SOURCE_DPLL構成設定 - ﾃﾞｼﾞﾀﾙ位相固定化閉路 */
#  define CONF_CLOCK_DPLL_ENABLE                    false
#  define CONF_CLOCK_DPLL_ON_DEMAND                 true
#  define CONF_CLOCK_DPLL_RUN_IN_STANDBY            false
#  define CONF_CLOCK_DPLL_LOCK_BYPASS               false
#  define CONF_CLOCK_DPLL_WAKE_UP_FAST              false
```

```
#  define CONF_CLOCK_DPLL_LOW_POWER_ENABLE         false

#  define CONF_CLOCK_DPLL_LOCK_TIME                SYSTEM_CLOCK_SOURCE_DPLL_LOCK_TIME_DEFAULT
#  define CONF_CLOCK_DPLL_REFERENCE_CLOCK          SYSTEM_CLOCK_SOURCE_DPLL_REFERENCE_CLOCK_XOSC32K
#  define CONF_CLOCK_DPLL_FILTER                   SYSTEM_CLOCK_SOURCE_DPLL_FILTER_DEFAULT

#  define CONF_CLOCK_DPLL_REFERENCE_FREQUENCY      32768
#  define CONF_CLOCK_DPLL_REFERENCE_DIVIDER        1
#  define CONF_CLOCK_DPLL_OUTPUT_FREQUENCY         48000000

/* DPLL GCLK基準構成設定 */
# define CONF_CLOCK_DPLL_REFERENCE_GCLK_GENERATOR GCLK_GENERATOR_1
/* DPLL GCLK固定化計時器構成設定 */
# define CONF_CLOCK_DPLL_LOCK_GCLK_GENERATOR GCLK_GENERATOR_1

/* clocks_init走行時にGCLKを構成設定するためにこれをtrueに設定してください。
 * falseに設定した場合、GCLK発振器はclocks_init()で何も構成設定されません。*/
#  define CONF_CLOCK_CONFIGURE_GCLK                true

/* GCLK発振器0構成設定（主ｸﾛｯｸ）*/
#  define CONF_CLOCK_GCLK_0_ENABLE                 true
#  define CONF_CLOCK_GCLK_0_RUN_IN_STANDBY         true
#  define CONF_CLOCK_GCLK_0_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_DFLL
#  define CONF_CLOCK_GCLK_0_PRESCALER              1
#  define CONF_CLOCK_GCLK_0_OUTPUT_ENABLE          false

/* GCLK発振器1構成設定 */
#  define CONF_CLOCK_GCLK_1_ENABLE                 false
#  define CONF_CLOCK_GCLK_1_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_1_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_XOSC32K
#  define CONF_CLOCK_GCLK_1_PRESCALER              1
#  define CONF_CLOCK_GCLK_1_OUTPUT_ENABLE          false

/* GCLK発振器2構成設定（RTC）*/
#  define CONF_CLOCK_GCLK_2_ENABLE                 false
#  define CONF_CLOCK_GCLK_2_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_2_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC32K
#  define CONF_CLOCK_GCLK_2_PRESCALER              32
#  define CONF_CLOCK_GCLK_2_OUTPUT_ENABLE          false

/* GCLK発振器3構成設定 */
#  define CONF_CLOCK_GCLK_3_ENABLE                 false
#  define CONF_CLOCK_GCLK_3_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_3_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_3_PRESCALER              1
#  define CONF_CLOCK_GCLK_3_OUTPUT_ENABLE          false

/* GCLK発振器4構成設定 */
#  define CONF_CLOCK_GCLK_4_ENABLE                 false
#  define CONF_CLOCK_GCLK_4_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_4_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_4_PRESCALER              1
#  define CONF_CLOCK_GCLK_4_OUTPUT_ENABLE          false

/* GCLK発振器5構成設定 */
#  define CONF_CLOCK_GCLK_5_ENABLE                 false
#  define CONF_CLOCK_GCLK_5_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_5_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_5_PRESCALER              1
#  define CONF_CLOCK_GCLK_5_OUTPUT_ENABLE          false
```

```
/* GCLK発振器6構成設定 */
#  define CONF_CLOCK_GCLK_6_ENABLE                 false
#  define CONF_CLOCK_GCLK_6_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_6_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_6_PRESCALER              1
#  define CONF_CLOCK_GCLK_6_OUTPUT_ENABLE          false

/* GCLK発振器7構成設定 */
#  define CONF_CLOCK_GCLK_7_ENABLE                 false
#  define CONF_CLOCK_GCLK_7_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_7_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_7_PRESCALER              1
#  define CONF_CLOCK_GCLK_7_OUTPUT_ENABLE          false

/* GCLK発振器8構成設定 */
#  define CONF_CLOCK_GCLK_8_ENABLE                 false
#  define CONF_CLOCK_GCLK_8_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_8_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_8_PRESCALER              1
#  define CONF_CLOCK_GCLK_8_OUTPUT_ENABLE          false

#endif /* CONF_CLOCKS_H_INCLUDED */
```

5.3.4. conf_board.h

5.3.4.1. AT32UC3A0, AT32UC3A1, AT32UC3Bﾃﾞﾊﾞｲｽ (USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// この例については既定基板初期化(ｽｲｯﾁ/LED)だけが必要です。

#endif /* CONF_BOARD_H_INCLUDED */
```

5.3.4.2. AT32UC3A3, AT32UC3A4ﾃﾞﾊﾞｲｽ (高速(HS)支援付USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// この例については既定基板初期化(ｽｲｯﾁ/LED)だけが必要です。

#endif /* CONF_BOARD_H_INCLUDED */
```

5.3.4.3. AT32UC3C, ATUCXXD, ATUCXXL3U, ATUCXXL4Uﾃﾞﾊﾞｲｽ (USBC)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// この例については既定基板初期化(ｽｲｯﾁ/LED)だけが必要です。

#endif /* CONF_BOARD_H_INCLUDED */
```

5.3.4.4. SAM3X, SAM3Aﾃﾞﾊﾞｲｽ (UOTGHS:USB OTG 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// USBﾋﾟﾝが使われます。
#define CONF_BOARD_USB_PORT

#endif /* CONF_BOARD_H_INCLUDED */
```

5.3.4.5. SAM D21ﾃﾞﾊﾞｲｽ (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

/* USB VBUS検出許可 */
#define CONF_BOARD_USB_VBUS_DETECT

#endif /* CONF_BOARD_H_INCLUDED */
```

6. 大容量記憶装置ｸﾗｽ(MSC)用USB装置ｲﾝﾀｰﾌｪｰｽ(UDI)

大容量記憶装置ｸﾗｽ(MSC)用USB装置ｲﾝﾀｰﾌｪｰｽ(UDI)はUSB MSC記憶装置の構成設定と管理に関するｲﾝﾀｰﾌｪｰｽを提供します。

この資料の概要は以下のとおりです。

- API概要
- USB装置大容量記憶装置単位部(UDI MSC)用の即時開始の手引き
- 構成設定ﾌｧｲﾙ例

Atmel®ｿﾌﾄｳｪｱ枠組み(ASF) USB装置階層に関するより多くの詳細については以下の応用記述を参照してください。

- AVR4900：ASF - USB装置階層
- AVR4920：ASF - USB装置階層 - 適合と性能係数
- AVR4921：ASF - USB装置階層 - ASF V1とV2間の違い

6.1. API概要

6.1.1. 変数と型定義

6.1.1.1. UDC用UDI MSCｲﾝﾀｰﾌｪｰｽ

udi_api_msc変数

```
UDC_DESC_STORAGE udi_api_t udi_api_msc
```

UDC用の標準UDI APIを含む全域構造体

6.1.2. 構造体定義

6.1.2.1. udi_msc_desc_t構造体

MSC用ｲﾝﾀｰﾌｪｰｽ記述子構造体

表6-1. ﾒﾝﾊﾞ

型	名前	説明
usb_ep_desc_t	ep_in	ﾃﾞｰﾀINｴﾝﾄﾞﾎﾟｲﾝﾄ記述子
usb_ep_desc_t	ep_out	ﾃﾞｰﾀOUTｴﾝﾄﾞﾎﾟｲﾝﾄ記述子
usb_iface_desc_t	iface	標準USBｲﾝﾀｰﾌｪｰｽ記述構造体

6.1.3. ﾏｸﾛ定義

6.1.3.1. USBｲﾝﾀｰﾌｪｰｽ記述子

以下の構造体は予め定義されたUSBｲﾝﾀｰﾌｪｰｽ記述子を提供します。これは最終的なUSB記述子を定義するのに使われなければなりません。

UDI_MSC_STRING_ID ﾏｸﾛ

```
#define UDI_MSC_STRING_ID
```

既定によってこのｲﾝﾀｰﾌｪｰｽに関連する文字列はなし

UDI_MSC_EPS_SIZE_FS ﾏｸﾛ

```
#define UDI_MSC_EPS_SIZE_FS
```

全速(FS)用のMSCｴﾝﾄﾞﾎﾟｲﾝﾄの大きさ

UDI_MSC_EPS_SIZE_HS ﾏｸﾛ

```
#define UDI_MSC_EPS_SIZE_HS
```

高速(HS)用のMSCｴﾝﾄﾞﾎﾟｲﾝﾄの大きさ

UDI_MSC_DESC ﾏｸﾛ

```
#define UDI_MSC_DESC
```

全ての速度に対するMSCｲﾝﾀｰﾌｪｰｽ記述子の内容

UDI_MSC_DESC_FS ﾏｸﾛ

```
#define UDI_MSC_DESC_FS
```

全速(FS)専用のMSCｲﾝﾀｰﾌｪｰｽ記述子の内容

UDI_MSC_DESC_HS マクロ

```
#define UDI_MSC_DESC_HS
```

高速(HS)専用のMSCインターフェース記述子の内容

6.1.4. 関数定義

6.1.4.1. udi_msc_process_trans()関数

背面読み書き命令処理

```
bool udi_msc_process_trans( void )
```

6.1.4.2. udi_msc_trans_block()関数

データとUSB MSCエンドポイント間で転送

```
bool udi_msc_trans_block( bool b_read, uint8_t * block, iram_size_t block_size,
  void(*)(udd_ep_status_t status, iram_size_t n, udd_ep_id_t ep) callback)
```

表6-2. パラメータ

パラメータ名	データ方向	説明
b_read	[入力]	真ならば、メモリからUSBへ
block	[入力,出力]	送るまたは満たすための内部RAM上の緩衝部
block_size	[入力]	送るまたは満たすための緩衝部の大きさ
callback	[入力]	転送の最後で呼ぶ関数。NULLならば、転送終了時にこのルーチンを抜けます。

戻り値： 関数が成功裏に終了した場合に1、さもなければ0

6.2. USB装置大容量記憶装置単位部(UDI MSC)用の即時開始の手引き

これは使用事例の選択に於いて単位部をどう構成設定して使うかを段階的に指示する、USB装置インターフェースMSC単位部(UDI MSC)用の即時開始の手引きです。

使用事例は様々なコードの断片を含みます。構成設定に関する段階でのコードの断片は独自初期化関数内に複写することができ、一方使用に関する段階は例えば主応用関数内に複写することができます。

6.2.1. 基本的な使用事例

この使用事例では"USB MSC (Single Interface Device)"単位部が使われます。"USB MSC (Composite Device)"単位部の使い方は高度な使用事例で記述されます。

6.2.1.1. 構成設定段階

USB装置のため、共通USB装置構成設定段階に従います。USB装置基本構成設定を参照してください。

USB MSCインターフェースはASF内の共通抽象層(ctrl_access)を通してメモリをアクセスします。メモリ インターフェース用共通抽象層をご覧ください。

メモリ インターフェース用共通抽象層

共通抽象層(ctrl_access)はメモリとUSB間のインターフェースを提供することができます。USB MSC UDIでは読み書きは以下のctrl_access関数を呼び出します。

```
extern Ctrl_status memory_2_usb(U8 lun, U32 addr, U16 nb_sector);
extern Ctrl_status usb_2_memory(U8 lun, U32 addr, U16 nb_sector);
```

その後にctrl_accessは各種LUNに対して読み書き操作を処理します。

ctrl_accessでのメモリ アクセスはconf_access.hを通して構成設定されます。例えば、仮想メモリ ディスクをアクセスするのにLUN0を使うには構成設定が以下を含むべきです。

```
#define LUN_0                   ENABLE // LUN0アクセス許可
//...

#define VIRTUAL_MEM             LUN_0
#define LUN_ID_VIRTUAL_MEM      LUN_ID_0
#define LUN_0_INCLUDE           "virtual_mem.h"          // (ctrl_accessに応じる)API
#define Lun_0_test_unit_ready   virtual_test_unit_ready  // ディスク準備可検査
#define Lun_0_read_capacity     virtual_read_capacity    // ディスク容量取得
#define Lun_0_wr_protect        virtual_wr_protect       // 保護検査
#define Lun_0_removal           virtual_removal          // ディスクが可搬型か検査
#define Lun_0_usb_read_10       virtual_usb_read_10      // ディスクからUSBへ転送
#define Lun_0_usb_write_10      virtual_usb_write_10     // USBからディスクへ転送
```

```
#define LUN_0_NAME            "¥"On-Chip Virtual Memory¥""
//...

#define ACCESS_USB                true     // USBインターフェース
//...

#define GLOBAL_WR_PROTECT         false
```

LUN_0が"仮想メモリ"として定義されるため、ディスクとしてアクセスするために内部または基板上のメモリをカプセル化するための単位部が含まれます。仮想メモリ ディスクのようなものの構成設定はconf_virtual_mem.h内です。例えば、メモリ ディスクのようなものを構築するのに内部RAMを使うには構成設定が以下を含むべきです。

```
#define VMEM_NB_SECTOR 48 // 内部RAM 24KB (20KB以上であるべき、さもなければPCはフォーマットできません。)
```

制御アクセスまたはディスク構成設定のより多くの説明についてはconf_access.hまたはconf_virtual_mem.hを参照してください。

メモリ制御アクセスについてのより多くの情報に関しては以下のオンライン資料を参照してください。

- Atmelソフトウェア枠組み – メモリ制御アクセス

6.2.1.2. 使用段階

コード例

conf_usb.hの内容

```
#define  USB_DEVICE_SERIAL_NAME  "12...EF" // MSC用ディスク通番
#define UDI_MSC_GLOBAL_VENDOR_ID ¥
   'A', 'T', 'M', 'E', 'L', ' ', ' ', ' '
#define UDI_MSC_GLOBAL_PRODUCT_VERSION ¥
   '1', '.', '0', '0'
#define UDI_MSC_ENABLE_EXT() my_callback_msc_enable()
extern bool my_callback_msc_enable(void);
#define UDI_MSC_DISABLE_EXT() my_callback_msc_disable()
extern void my_callback_msc_disable(void);
#include "udi_msc_conf.h" // conf_usb.hファイルの最後で
```

応用Cファイルに追加してください。

```
static bool my_flag_autorize_msc_transfert = false;
bool my_callback_msc_enable(void)
{
   my_flag_autorize_msc_transfert = true;
   return true;
}
void my_callback_msc_disable(void)
{
   my_flag_autorize_msc_transfert = false;
}

void task(void)
{
   udi_msc_process_trans();
}
```

作業の流れ

1. conf_usb.hが利用可能でUSB装置MSC構成設定である以下の構成設定を含むことを確実にしてください。

   ```
   #define USB_DEVICE_SERIAL_NAME  "12...EF" // MSCに対するディスク通番
   ```

   注：USB通番はMSC割り込みが使われる時に必須です。

   ```
   #define UDI_MSC_GLOBAL_VENDOR_ID ¥
      'A', 'T', 'M', 'E', 'L', ' ', ' ', ' '
   #define UDI_MSC_GLOBAL_PRODUCT_VERSION ¥
      '1', '.', '0', '0'
   ```

   注：USB MSCインターフェースは供給者ID(ASCII 8文字)と製品版番号(ASCII 4文字)が必要です。

   ```
   #define UDI_MSC_ENABLE_EXT() my_callback_msc_enable()
   extern bool my_callback_msc_enable(void);
   ```

注：装置列挙(USB装置の検出と識別)後、USBホストは装置の構成設定を始めます。装置からのUSB MSCインターフェースがホストによって受け入れられると、USBホストはこのインターフェースを許可してUDI_MSC_ENABLE_EXT()呼び戻し関数が呼ばれ、trueが返ります。故に、この事象を受け取った時にudi_msc_process_trans()を呼ぶ作業(タスク)が許可されなければなりません。

```
#define UDI_MSC_DISABLE_EXT() my_callback_msc_disable()
extern void my_callback_msc_disable(void);
```

注：USB装置が切断されるか、またはUSBホストによってリセットされると、USBインターフェースは禁止され、UDI_MSC_DISABLE_EXT()呼び戻し関数が呼ばれます。故に、udi_msc_process_trans()を呼ぶ作業(タスク)を禁止することが推奨されます。

2. MSCはメモリ インターフェースを提供するメモリ制御アクセス部と自動的に繋げられます。けれども、メモリ データ転送はUSB割り込みルーチンの外側で行われなければなりません。これは主繰り返しによって呼ばれるMSC処理(”udi_msc_process_trans()”)で行われます。

```
void task(void) {
  udi_msc_process_trans();
}
```

3. MSC速度は作業(タスク)周期に依存します。最速を得るため、(例えば、排他制御を通して)この作業(タスク)を起こすのに”UDI_MSC_NOTIFY_TRANS_EXT”通知呼び戻しを使うことができます。

```
#define  UDI_MSC_NOTIFY_TRANS_EXT()     msc_notify_trans()
void msc_notify_trans(void) {
  wakeup_my_task();
}
```

6.2.2. 高度な使用事例

UDI MSC単位部の複数インターフェースの使用については以下をご覧ください。

・複合装置でのMSC

UDI CDC単位部のもっと高度な使用については以下をご覧ください。

・USB装置の高度な使用事例

6.2.3. 複合装置でのMSC

USB複合装置は1つよりも多くのUSBクラスを使うUSB装置です。この使用事例ではUSB複合装置を作成するのに”USB MSC (Composite Device)”単位部が使われます。故に、このUSB単位部は”USB HID Mouse (Composite Device)”のような別な”複合装置(Composite Device)”単位部と連携することができます。

また、「AVR4902:ASF - USB複合装置」応用記述を参照することもできます。

6.2.3.1. 構成設定段階

動くためのこの事例の構成設定ｺｰﾄﾞについては基本的な使用事例に従わなければなりません。

6.2.3.2. 使用段階

ｺｰﾄﾞ例

conf_usb.hの内容

```
#define USB_DEVICE_EP_CTRL_SIZE  64
#define USB_DEVICE_NB_INTERFACE (X+1)
#define USB_DEVICE_MAX_EP (X+2)
#define UDI_MSC_EP_IN  (X | USB_EP_DIR_IN)
#define UDI_MSC_EP_OUT (Y | USB_EP_DIR_OUT)
#define UDI_MSC_IFACE_NUMBER  X
#define UDI_COMPOSITE_DESC_T ¥
   udi_msc_desc_t udi_msc; ¥
   ～
#define UDI_COMPOSITE_DESC_FS ¥
   .udi_msc = UDI_MSC_DESC, ¥
   ～
#define UDI_COMPOSITE_DESC_HS ¥
   .udi_msc = UDI_MSC_DESC, ¥
   ～
#define UDI_COMPOSITE_API ¥
   &udi_api_msc, ¥
   ～
```

作業の流れ

1. conf_usb.hが利用可能でUSB複合装置構成設定に必要とされる以下のパラメータを含むことを確実にしてください。

```
// エンドポイント制御容量、これは以下でなければなりません。
// - 全速(FS)装置に対して、8, 16 ,32 または 64 (RAMを節約するために8が推奨されます。)
// - 高速(HS)装置に対して、64
#define USB_DEVICE_EP_CTRL_SIZE   64
// このUSB装置での総インターフェース数
// MSC用に1を加算
#define USB_DEVICE_NB_INTERFACE  (X+1)
// このUSB装置での総エンドポイント数
// これには各インターフェースに対する各々のエンドポイントを含めなければなりません。
// MSC用に2を加算
#define USB_DEVICE_MAX_EP  (X+2)
```

2. conf_usb.hが複合装置の記述子を含むことを確実にしてください。

```
// あなたが選んだMSC用のエンドポイント番号
// エンドポイント番号は1から始まります。
#define UDI_MSC_EP_IN   (X | USB_EP_DIR_IN)
#define UDI_MSC_EP_OUT  (Y | USB_EP_DIR_OUT)
// 0から始まるインターフェースのインターフェース指標
#define UDI_MSC_IFACE_NUMBER   X
```

3. conf_usb.hがUSB複合装置構成設定に必要とされる以下のパラメータを含むことを確実にしてください。

```
// USBインターフェース記述子構造体
#define UDI_COMPOSITE_DESC_T ¥
    ～
    udi_msc_desc_t udi_msc; ¥
    ～
// 全速(FS)用USBインターフェース記述子値
#define UDI_COMPOSITE_DESC_FS ¥
    ～
    .udi_msc = UDI_MSC_DESC_FS, ¥
    ～
// 高速(HS)用USBインターフェース記述子値
#define UDI_COMPOSITE_DESC_HS ¥
    ～
    .udi_msc = UDI_MSC_DESC_HS, ¥
    ～
// USBインターフェースAPI
#define UDI_COMPOSITE_API ¥
    ～
    &udi_api_msc, ¥
    ～
```

注：上の4つの一覧で与えられた記述子の順番は全てのインターフェース指標によって定義された順番と同じでなければなりません。インターフェース指標の順番はUDI_X_IFACE_NUMBER定義を通して定義されます。

6.3. 構成設定ファイル例

6.3.1. conf_usb.h

6.3.1.1. 単一UDI MSC

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_USB_H_
#define _CONF_USB_H_

#include "compiler.h"

#warning You must refill the following definitions with a correct values
```

```
#define  USB_DEVICE_VENDOR_ID             USB_VID_ATMEL
#define  USB_DEVICE_PRODUCT_ID            USB_PID_ATMEL_ASF_MSC
#define  USB_DEVICE_MAJOR_VERSION         1
#define  USB_DEVICE_MINOR_VERSION         0
#define  USB_DEVICE_POWER                 100 // VBUS線での消費(mA)
#define  USB_DEVICE_ATTR                  ¥
    (USB_CONFIG_ATTR_SELF_POWERED)
// (USB_CONFIG_ATTR_BUS_POWERED)
//  (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_SELF_POWERED)
//  (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_BUS_POWERED)

// #define  USB_DEVICE_MANUFACTURE_NAME      "Manufacture name"
// #define  USB_DEVICE_PRODUCT_NAME          "Product name"
#define  USB_DEVICE_SERIAL_NAME           "12...EF" // MSC用ﾃﾞｨｽｸ通番

#if (UC3A3||UC3A4)
//#define  USB_DEVICE_HS_SUPPORT
#endif

// #define  UDC_VBUS_EVENT(b_vbus_high)      user_callback_vbus_action(b_vbus_high)
// extern void user_callback_vbus_action(bool b_vbus_high);
// #define  UDC_SOF_EVENT()                  user_callback_sof_action()
// extern void user_callback_sof_action(void);
// #define  UDC_SUSPEND_EVENT()              user_callback_suspend_action()
// extern void user_callback_suspend_action(void);
// #define  UDC_RESUME_EVENT()               user_callback_resume_action()
// extern void user_callback_resume_action(void);
// #define  UDC_REMOTEWAKEUP_ENABLE()        user_callback_remotewakeup_enable()
// extern void user_callback_remotewakeup_enable(void);
// #define  UDC_REMOTEWAKEUP_DISABLE()       user_callback_remotewakeup_disable()
// extern void user_callback_remotewakeup_disable(void);
// #define  UDC_GET_EXTRA_STRING()

#define UDI_MSC_GLOBAL_VENDOR_ID           ¥
   'A', 'T', 'M', 'E', 'L', ' ', ' ', ' '
#define UDI_MSC_GLOBAL_PRODUCT_VERSION           ¥
   '1', '.', '0', '0'

#define  UDI_MSC_ENABLE_EXT()          true
#define  UDI_MSC_DISABLE_EXT()
#define  UDI_MSC_NOTIFY_TRANS_EXT()
/*
 * #define UDI_MSC_ENABLE_EXT() my_callback_msc_enable()
 * extern bool my_callback_msc_enable(void);
 * #define UDI_MSC_DISABLE_EXT() my_callback_msc_disable()
 * extern void my_callback_msc_disable(void);
 * #define  UDI_MSC_NOTIFY_TRANS_EXT()    msc_notify_trans()
 * extern void msc_notify_trans(void) {
 */

#include "udi_msc_conf.h"

#endif // _CONF_USB_H_
```

6.3.1.2. 複数UDI MSC (複合)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_USB_H_
#define _CONF_USB_H_
```

```
#include "compiler.h"

#warning You must refill the following definitions with a correct values

#define  USB_DEVICE_VENDOR_ID             USB_VID_ATMEL
#define  USB_DEVICE_PRODUCT_ID            0xFFFF
#define  USB_DEVICE_MAJOR_VERSION         1
#define  USB_DEVICE_MINOR_VERSION         0
#define  USB_DEVICE_POWER                 100 // VBUS線での消費(mA)
#define  USB_DEVICE_ATTR                  ¥
        (USB_CONFIG_ATTR_SELF_POWERED)
// (USB_CONFIG_ATTR_BUS_POWERED)
// (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_SELF_POWERED)
// (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_BUS_POWERED)

// #define  USB_DEVICE_MANUFACTURE_NAME      "Manufacture name"
// #define  USB_DEVICE_PRODUCT_NAME          "Product name"
// #define  USB_DEVICE_SERIAL_NAME           "12...EF" // MSC用ﾃﾞｨｽｸ通番

//#define  USB_DEVICE_LOW_SPEED

#if (UC3A3||UC3A4)
//#define  USB_DEVICE_HS_SUPPORT
#endif

// #define  UDC_VBUS_EVENT(b_vbus_high)      user_callback_vbus_action(b_vbus_high
// extern void user_callback_vbus_action(bool b_vbus_high);
// #define  UDC_SOF_EVENT()                  user_callback_sof_action()
// extern void user_callback_sof_action(void);
// #define  UDC_SUSPEND_EVENT()              user_callback_suspend_action()
// extern void user_callback_suspend_action(void);
// #define  UDC_RESUME_EVENT()               user_callback_resume_action()
// extern void user_callback_resume_action(void);
// #define  UDC_REMOTEWAKEUP_ENABLE()        user_callback_remotewakeup_enable()
// extern void user_callback_remotewakeup_enable(void);
// #define  UDC_REMOTEWAKEUP_DISABLE()       user_callback_remotewakeup_disable()
// extern void user_callback_remotewakeup_disable(void);
// #define  UDC_GET_EXTRA_STRING()

#define  USB_DEVICE_EP_CTRL_SIZE       64

#define  USB_DEVICE_NB_INTERFACE       1 // 1またはそれ以上

#define  USB_DEVICE_MAX_EP             1 // 0～ｲﾝﾀｰﾌｪｰｽによって要求された最大ｴﾝﾄﾞﾎﾟｲﾝﾄ

#define  UDI_CDC_PORT_NB 1

#define  UDI_CDC_ENABLE_EXT(port)             true
#define  UDI_CDC_DISABLE_EXT(port)
#define  UDI_CDC_RX_NOTIFY(port)
#define  UDI_CDC_TX_EMPTY_NOTIFY(port)
#define  UDI_CDC_SET_CODING_EXT(port,cfg)
#define  UDI_CDC_SET_DTR_EXT(port,set)
#define  UDI_CDC_SET_RTS_EXT(port,set)
/*
 * #define UDI_CDC_ENABLE_EXT(port) my_callback_cdc_enable()
 * extern bool my_callback_cdc_enable(void);
 * #define UDI_CDC_DISABLE_EXT(port) my_callback_cdc_disable()
 * extern void my_callback_cdc_disable(void);
 * #define  UDI_CDC_RX_NOTIFY(port) my_callback_rx_notify(port)
```

```
 * extern void my_callback_rx_notify(uint8_t port);
 * #define  UDI_CDC_TX_EMPTY_NOTIFY(port) my_callback_tx_empty_notify(port)
 * extern void my_callback_tx_empty_notify(uint8_t port);
 * #define  UDI_CDC_SET_CODING_EXT(port,cfg) my_callback_config(port,cfg)
 * extern void my_callback_config(uint8_t port, usb_cdc_line_coding_t * cfg);
 * #define  UDI_CDC_SET_DTR_EXT(port,set) my_callback_cdc_set_dtr(port,set)
 * extern void my_callback_cdc_set_dtr(uint8_t port, bool b_enable);
 * #define  UDI_CDC_SET_RTS_EXT(port,set) my_callback_cdc_set_rts(port,set)
 * extern void my_callback_cdc_set_rts(uint8_t port, bool b_enable);
 */

#define  UDI_CDC_LOW_RATE

#define  UDI_CDC_DEFAULT_RATE             115200
#define  UDI_CDC_DEFAULT_STOPBITS         CDC_STOP_BITS_1
#define  UDI_CDC_DEFAULT_PARITY           CDC_PAR_NONE
#define  UDI_CDC_DEFAULT_DATABITS         8

#define  UDI_CDC_DATA_EP_IN_0          (1 | USB_EP_DIR_IN)  // 送信
#define  UDI_CDC_DATA_EP_OUT_0         (2 | USB_EP_DIR_OUT) // 受信
#define  UDI_CDC_COMM_EP_0             (3 | USB_EP_DIR_IN)  // 通知ｴﾝﾄﾞﾎﾟｲﾝﾄ
#define  UDI_CDC_DATA_EP_IN_2          (4 | USB_EP_DIR_IN)  // 送信
#define  UDI_CDC_DATA_EP_OUT_2         (5 | USB_EP_DIR_OUT) // 受信
#define  UDI_CDC_COMM_EP_2             (6 | USB_EP_DIR_IN)  // 通知ｴﾝﾄﾞﾎﾟｲﾝﾄ
#define  UDI_CDC_DATA_EP_IN_3          (7 | USB_EP_DIR_IN)  // 送信
#define  UDI_CDC_DATA_EP_OUT_3         (8 | USB_EP_DIR_OUT) // 受信
#define  UDI_CDC_COMM_EP_3             (9 | USB_EP_DIR_IN)  // 通知ｴﾝﾄﾞﾎﾟｲﾝﾄ

#define  UDI_CDC_COMM_IFACE_NUMBER_0   0
#define  UDI_CDC_DATA_IFACE_NUMBER_0   1
#define  UDI_CDC_COMM_IFACE_NUMBER_2   2
#define  UDI_CDC_DATA_IFACE_NUMBER_2   3
#define  UDI_CDC_COMM_IFACE_NUMBER_3   4
#define  UDI_CDC_DATA_IFACE_NUMBER_3   5

#define UDI_MSC_GLOBAL_VENDOR_ID           ¥
   'A', 'T', 'M', 'E', 'L', ' ', ' ', ' '
#define UDI_MSC_GLOBAL_PRODUCT_VERSION           ¥
   '1', '.', '0', '0'

#define  UDI_MSC_ENABLE_EXT()          true
#define  UDI_MSC_DISABLE_EXT()
#define  UDI_MSC_NOTIFY_TRANS_EXT()
/*
 * #define UDI_MSC_ENABLE_EXT() my_callback_msc_enable()
 * extern bool my_callback_msc_enable(void);
 * #define UDI_MSC_DISABLE_EXT() my_callback_msc_disable()
 * extern void my_callback_msc_disable(void);
 * #define  UDI_MSC_NOTIFY_TRANS_EXT()    msc_notify_trans()
 * extern void msc_notify_trans(void) {
 */

#define  UDI_MSC_EP_IN                 (1 | USB_EP_DIR_IN)
#define  UDI_MSC_EP_OUT                (2 | USB_EP_DIR_OUT)

#define  UDI_MSC_IFACE_NUMBER          0

#define  UDI_HID_MOUSE_ENABLE_EXT()       true
#define  UDI_HID_MOUSE_DISABLE_EXT()
// #define UDI_HID_MOUSE_ENABLE_EXT() my_callback_mouse_enable()
// extern bool my_callback_mouse_enable(void);
```

```
// #define UDI_HID_MOUSE_DISABLE_EXT() my_callback_mouse_disable()
// extern void my_callback_mouse_disable(void);

#define  UDI_HID_MOUSE_EP_IN           (1 | USB_EP_DIR_IN)

#define  UDI_HID_MOUSE_IFACE_NUMBER    0

#define  UDI_HID_KBD_ENABLE_EXT()       true
#define  UDI_HID_KBD_DISABLE_EXT()
// #define UDI_HID_KBD_ENABLE_EXT() my_callback_keyboard_enable()
// extern bool my_callback_keyboard_enable(void);
// #define UDI_HID_KBD_DISABLE_EXT() my_callback_keyboard_disable()
// extern void my_callback_keyboard_disable(void);
#define  UDI_HID_KBD_CHANGE_LED(value)
// #define  UDI_HID_KBD_CHANGE_LED(value) my_callback_keyboard_led(value)
// extern void my_callback_keyboard_led(uint8_t value)

#define  UDI_HID_KBD_EP_IN           (1 | USB_EP_DIR_IN)

#define  UDI_HID_KBD_IFACE_NUMBER    0

#define  UDI_HID_GENERIC_ENABLE_EXT()       true
#define  UDI_HID_GENERIC_DISABLE_EXT()
#define  UDI_HID_GENERIC_REPORT_OUT(ptr)
#define  UDI_HID_GENERIC_SET_FEATURE(f)
/*
 * #define UDI_HID_GENERIC_ENABLE_EXT() my_callback_generic_enable()
 * extern bool my_callback_generic_enable(void);
 * #define UDI_HID_GENERIC_DISABLE_EXT() my_callback_generic_disable()
 * extern void my_callback_generic_disable(void);
 * #define  UDI_HID_GENERIC_REPORT_OUT(ptr) my_callback_generic_report_out(ptr)
 * extern void my_callback_generic_report_out(uint8_t *report);
 * #define  UDI_HID_GENERIC_SET_FEATURE(f) my_callback_generic_set_feature(f)
 * extern void my_callback_generic_set_feature(uint8_t *report_feature);
 */
#define  UDI_HID_REPORT_IN_SIZE             64
#define  UDI_HID_REPORT_OUT_SIZE            64
#define  UDI_HID_REPORT_FEATURE_SIZE        4
#define  UDI_HID_GENERIC_EP_SIZE            64

#define  UDI_HID_GENERIC_EP_OUT    (2 | USB_EP_DIR_OUT)
#define  UDI_HID_GENERIC_EP_IN     (1 | USB_EP_DIR_IN)

#define  UDI_HID_GENERIC_IFACE_NUMBER    0

#define  UDI_PHDC_ENABLE_EXT()          true
#define  UDI_PHDC_DISABLE_EXT()

#define  UDI_PHDC_DATAMSG_FORMAT       USB_PHDC_DATAMSG_FORMAT_11073_20601
#define  UDI_PHDC_SPECIALIZATION       {0x2345} // 11073_20601での定義

#define  UDI_PHDC_QOS_OUT              ¥
        (USB_PHDC_QOS_MEDIUM_BETTER|USB_PHDC_QOS_HIGH_BEST)
#define  UDI_PHDC_QOS_IN               ¥
        (USB_PHDC_QOS_LOW_GOOD|USB_PHDC_QOS_MEDIUM_BETTER|USB_PHDC_QOS_MEDIUM_BEST)

#define  UDI_PHDC_METADATA_DESC_BULK_IN    {0x01,0x02,0x03}
#define  UDI_PHDC_METADATA_DESC_BULK_OUT   {0x01,0x02,0x03}
#define  UDI_PHDC_METADATA_DESC_INT_IN     {0x01,0x02,0x03}

#define  UDI_PHDC_EP_BULK_OUT          (1 | USB_EP_DIR_OUT)
```

```
#define  UDI_PHDC_EP_BULK_IN           (2 | USB_EP_DIR_IN)
#if ((UDI_PHDC_QOS_IN&USB_PHDC_QOS_LOW_GOOD)==USB_PHDC_QOS_LOW_GOOD)
// UDI_PHDC_QOS_INの場合にだけUSB_PHDC_QOS_LOW_GOODをｲﾝｸﾙｰﾄﾞ
#  define  UDI_PHDC_EP_INTERRUPT_IN      (3 | USB_EP_DIR_IN)
#endif

#define  UDI_PHDC_EP_SIZE_BULK_OUT     32
#define  UDI_PHDC_EP_SIZE_BULK_IN      32
#define  UDI_PHDC_EP_SIZE_INT_IN       8

#define  UDI_PHDC_IFACE_NUMBER          0

#define  UDI_VENDOR_ENABLE_EXT()           true
#define  UDI_VENDOR_DISABLE_EXT()
#define  UDI_VENDOR_SETUP_OUT_RECEIVED()  false
#define  UDI_VENDOR_SETUP_IN_RECEIVED()   false
/*
 * #define UDI_VENDOR_ENABLE_EXT() my_callback_vendor_enable()
 * extern bool my_callback_vendor_enable(void);
 * #define UDI_VENDOR_DISABLE_EXT() my_callback_vendor_disable()
 * extern void my_callback_vendor_disable(void);
 *
 * #define  UDI_VENDOR_SETUP_OUT_RECEIVED()  my_vendor_setup_out_received()
 * extern bool my_vendor_setup_out_received(void);
 * #define  UDI_VENDOR_SETUP_IN_RECEIVED()   my_vendor_setup_in_received()
 * extern bool my_vendor_setup_in_received(void);
 */

#define  UDI_VENDOR_EPS_SIZE_INT_FS    64
#define  UDI_VENDOR_EPS_SIZE_BULK_FS   64
#define  UDI_VENDOR_EPS_SIZE_ISO_FS   256

#define  UDI_VENDOR_EPS_SIZE_INT_HS    64
#define  UDI_VENDOR_EPS_SIZE_BULK_HS  512
#define  UDI_VENDOR_EPS_SIZE_ISO_HS    64

#define  UDI_VENDOR_EP_INTERRUPT_IN  (1 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_INTERRUPT_OUT (2 | USB_EP_DIR_OUT)
#define  UDI_VENDOR_EP_BULK_IN       (3 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_BULK_OUT      (4 | USB_EP_DIR_OUT)
#define  UDI_VENDOR_EP_ISO_IN        (5 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_ISO_OUT       (6 | USB_EP_DIR_OUT)

#define  UDI_VENDOR_IFACE_NUMBER     0

//... 最終的に他のｲﾝﾀｰﾌｪｰｽ構成設定を追加してください。

#define UDI_COMPOSITE_DESC_T

#define UDI_COMPOSITE_DESC_FS

#define UDI_COMPOSITE_DESC_HS

#define UDI_COMPOSITE_API

/* CDC、MSC、HIDﾏｳｽ ｲﾝﾀｰﾌｪｰｽを持つ装置の例
#define UDI_COMPOSITE_DESC_T ¥
    usb_iad_desc_t udi_cdc_iad; ¥
    udi_cdc_comm_desc_t udi_cdc_comm; ¥
    udi_cdc_data_desc_t udi_cdc_data; ¥
    udi_msc_desc_t udi_msc; ¥
```

```
    udi_hid_mouse_desc_t udi_hid_mouse

#define UDI_COMPOSITE_DESC_FS ¥
    .udi_cdc_iad               = UDI_CDC_IAD_DESC_0, ¥
    .udi_cdc_comm              = UDI_CDC_COMM_DESC_0, ¥
    .udi_cdc_data              = UDI_CDC_DATA_DESC_0_FS, ¥
    .udi_msc                   = UDI_MSC_DESC_FS, ¥
    .udi_hid_mouse             = UDI_HID_MOUSE_DESC

#define UDI_COMPOSITE_DESC_HS ¥
    .udi_cdc_iad               = UDI_CDC_IAD_DESC_0, ¥
    .udi_cdc_comm              = UDI_CDC_COMM_DESC_0, ¥
    .udi_cdc_data              = UDI_CDC_DATA_DESC_0_HS, ¥
    .udi_msc                   = UDI_MSC_DESC_HS, ¥
    .udi_hid_mouse             = UDI_HID_MOUSE_DESC

#define UDI_COMPOSITE_API  ¥
    &udi_api_cdc_comm,        ¥
    &udi_api_cdc_data,        ¥
    &udi_api_msc,             ¥
    &udi_api_hid_mouse
*/

/* インターフェース用インクルードの例
#include "udi_msc.h"
#include "udi_hid_kbd.h"
#include "udi_hid_mouse.h"
#include "udi_cdc.h"
#include "udi_phdc.h"
#include "udi_vendor.h"
*/
/* USBによって使われる呼び戻しの宣言
#include "callback_def.h"
*/

#endif // _CONF_USB_H_
```

6.3.2. conf_clock.h

6.3.2.1. XMEGA (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

#define CONFIG_USBCLK_SOURCE      USBCLK_SRC_RCOSC
#define CONFIG_OSC_RC32_CAL       48000000UL

#define CONFIG_OSC_AUTOCAL_RC32MHZ_REF_OSC  OSC_ID_USBSOF

#define CONFIG_SYSCLK_SOURCE      SYSCLK_SRC_RC32MHZ
#define CONFIG_SYSCLK_PSADIV      SYSCLK_PSADIV_2
#define CONFIG_SYSCLK_PSBCDIV     SYSCLK_PSBCDIV_1_1

/*

#define CONFIG_PLL0_SOURCE        PLL_SRC_XOSC
#define CONFIG_PLL0_MUL           6
#define CONFIG_PLL0_DIV           1
```

```
#define CONFIG_USBCLK_SOURCE      USBCLK_SRC_PLL

#define CONFIG_SYSCLK_SOURCE      SYSCLK_SRC_PLL
#define CONFIG_SYSCLK_PSADIV      SYSCLK_PSADIV_2
#define CONFIG_SYSCLK_PSBCDIV     SYSCLK_PSBCDIV_1_1
*/

#endif /* CONF_CLOCK_H_INCLUDED */
```

6.3.2.2. AT32UC3A0, ATUC3A1, ATUC3Bﾃﾞﾊﾞｲｽ (USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== ｼｽﾃﾑ ｸﾛｯｸ元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RCSYS
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_OSC0
#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_PLL0

// ===== PLL0任意選択
#define CONFIG_PLL0_SOURCE            PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE          PLL_SRC_OSC1
#define CONFIG_PLL0_MUL               8 /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL0_DIV               2 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== PLL1任意選択
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC0
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL1_MUL             8 /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL1_DIV             2 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== ｼｽﾃﾑ ｸﾛｯｸ ﾊﾞｽ分周任意選択
//#define CONFIG_SYSCLK_CPU_DIV         0 /* Fcpu = Fsys/(2 ^ CPU_div) */
//#define CONFIG_SYSCLK_PBA_DIV         0 /* Fpba = Fsys/(2 ^ PBA_div) */
//#define CONFIG_SYSCLK_PBB_DIV         0 /* Fpbb = Fsys/(2 ^ PBB_div) */

// ===== 周辺機能ｸﾛｯｸ管理任意選択
//#define CONFIG_SYSCLK_INIT_CPUMASK  ((1 << SYSCLK_SYSTIMER) | (1 << SYSCLK_OCD))
//#define CONFIG_SYSCLK_INIT_PBAMASK  (1 << SYSCLK_USART0)
//#define CONFIG_SYSCLK_INIT_PBBMASK  (1 << SYSCLK_HMATRIX)
//#define CONFIG_SYSCLK_INIT_HSBMASK  (1 << SYSCLK_MDMA_HSB)

// ===== USBｸﾛｯｸ元任意選択
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_OSC0
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_PLL0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV             1 /* Fusb = Fsys/(2 ^ USB_div) */

#endif /* CONF_CLOCK_H_INCLUDED */
```

6.3.2.3. AT32UC3A3, AT32UC3A4ﾃﾞﾊﾞｲｽ (高速(HS)支援付USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED
```

```
// ===== システム クロック元任意選択
//#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_RCSYS
//#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_OSC0
#define CONFIG_SYSCLK_SOURCE            SYSCLK_SRC_PLL0

// ===== PLL0任意選択
#define CONFIG_PLL0_SOURCE              PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE            PLL_SRC_OSC1
#define CONFIG_PLL0_MUL                 11 /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL0_DIV                 2  /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== PLL1任意選択
//#define CONFIG_PLL1_SOURCE            PLL_SRC_OSC0
//#define CONFIG_PLL1_SOURCE            PLL_SRC_OSC1
//#define CONFIG_PLL1_MUL               8 /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL1_DIV               2 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== システム クロック バス分周任意選択
#define CONFIG_SYSCLK_CPU_DIV           0 /* Fcpu = Fsys/(2 ^ CPU_div) */
#define CONFIG_SYSCLK_PBA_DIV           0 /* Fpba = Fsys/(2 ^ PBA_div) */
//#define CONFIG_SYSCLK_PBB_DIV         0 /* Fpbb = Fsys/(2 ^ PBB_div) */

// ===== 周辺機能クロック管理任意選択
//#define CONFIG_SYSCLK_INIT_CPUMASK    ((1 << SYSCLK_SYSTIMER) | (1 << SYSCLK_OCD))
//#define CONFIG_SYSCLK_INIT_PBAMASK    (1 << SYSCLK_USART0)
//#define CONFIG_SYSCLK_INIT_PBBMASK    (1 << SYSCLK_HMATRIX)
//#define CONFIG_SYSCLK_INIT_HSBMASK    (1 << SYSCLK_MDMA_HSB)

// ===== USBクロック元任意選択
#define CONFIG_USBCLK_SOURCE            USBCLK_SRC_OSC0
//#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_PLL0
//#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV               1 /* Fusb = Fsys/(2 ^ USB_div) */

#endif /* CONF_CLOCK_H_INCLUDED */
```

6.3.2.4. AT32UC3C, ATUCXXD, ATUCXXL3U, ATUCXXL4Uﾃﾞﾊﾞｲｽ (USBC)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== システム クロック元任意選択
//#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_RCSYS
//#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_OSC0
//#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_OSC1
#define CONFIG_SYSCLK_SOURCE            SYSCLK_SRC_PLL1
//#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_PLL1
//#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_RC8M

// ===== PLL0任意選択
#define CONFIG_PLL0_SOURCE              PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE            PLL_SRC_OSC1
//#define CONFIG_PLL0_SOURCE            PLL_SRC_RC8M
#define CONFIG_PLL0_MUL                 3 /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL0_DIV                 1 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== PLL1任意選択
#define CONFIG_PLL1_SOURCE              PLL_SRC_OSC0
```

```
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL1_SOURCE          PLL_SRC_RC8M
#define CONFIG_PLL1_MUL               3 /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL1_DIV               1 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== システム クロック バス分周任意選択
//#define CONFIG_SYSCLK_CPU_DIV       0 /* Fcpu = Fsys/(2 ^ CPU_div) */
//#define CONFIG_SYSCLK_PBA_DIV       0 /* Fpba = Fsys/(2 ^ PBA_div) */
//#define CONFIG_SYSCLK_PBB_DIV       0 /* Fpbb = Fsys/(2 ^ PBB_div) */
//#define CONFIG_SYSCLK_PBC_DIV       0 /* Fpbc = Fsys/(2 ^ PBC_div) */

// ===== 周辺機能クロック管理任意選択
//#define CONFIG_SYSCLK_INIT_CPUMASK  ((1 << SYSCLK_SYSTIMER) | (1 << SYSCLK_OCD))
//#define CONFIG_SYSCLK_INIT_PBAMASK  (1 << SYSCLK_USART0)
//#define CONFIG_SYSCLK_INIT_PBBMASK  (1 << SYSCLK_HMATRIX)
//#define CONFIG_SYSCLK_INIT_HSBMASK  (1 << SYSCLK_MDMA_HSB)

// ===== USBクロック元任意選択
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_OSC0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_OSC1
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_PLL0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV             1 /* Fusb = Fsys/(2 ^ USB_div) */

#endif /* CONF_CLOCK_H_INCLUDED */
```

6.3.2.5. SAM3S, SAM3SD, SAM4Sﾃﾞﾊﾞｲｽ (UPD:USB周辺機能装置)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== システム クロック(MCK)元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_BYPASS
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_4M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_8M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_12M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_BYPASS
#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLLACK
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLLBCK

// ===== システム クロック(MCK)前置分周器任意選択    (Fmck = Fsys / (SYSCLK_PRES))
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_1
#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_2
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_4
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_8
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_16
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_32
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_64
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_3

// ===== PLL0 (A)任意選択    (Fpll = (Fclk * PLL_mul) / PLL_div)
// ここにmulとdivの実効値を使ってください。
#define CONFIG_PLL0_SOURCE          PLL_SRC_MAINCK_XTAL
#define CONFIG_PLL0_MUL             32
#define CONFIG_PLL0_DIV             3
```

```
// ===== PLL1 (B)任意選択    (Fpll = (Fclk * PLL_mul) / PLL_div)
// ここにmulとdivの実効値を使ってください。
#define CONFIG_PLL1_SOURCE          PLL_SRC_MAINCK_XTAL
#define CONFIG_PLL1_MUL             16
#define CONFIG_PLL1_DIV             2

// ===== USBｸﾛｯｸ元任意選択    (Fusb = FpllX / USB_div)
// ここにdivの実効値を使ってください。
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL0
#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV           2

// ===== 目的対象周波数 (ｼｽﾃﾑ ｸﾛｯｸ)
// - XTAL周波数: 12MHz
// - ｼｽﾃﾑ ｸﾛｯｸ元: PLLA
// - ｼｽﾃﾑ ｸﾛｯｸ前置分周器: 2 (2分周)
// - PLLA供給元: XTAL
// - PLLA出力: XTAL * 32 / 3
// - ｼｽﾃﾑ ｸﾛｯｸは: 12 * 32 / 3 / 2 = 64MHz
// ===== 目的対象周波数 (USBｸﾛｯｸ)
// - USBｸﾛｯｸ元: PLLB
// - USBｸﾛｯｸ分周器: 2 (2分周)
// - PLLB出力: XTAL * 16 / 2
// - USBｸﾛｯｸ: 12 * 16 / 2 / 2 = 48MHz

#endif /* CONF_CLOCK_H_INCLUDED */
```

6.3.2.6. SAM3Uﾃﾞﾊﾞｲｽ (UPDHS:USB周辺機能装置 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== ｼｽﾃﾑ ｸﾛｯｸ(MCK)元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_BYPASS
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_4M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_8M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_12M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_BYPASS
#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLLACK
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_UPLLCK

// ===== ｼｽﾃﾑ ｸﾛｯｸ(MCK)前置分周器任意選択    (Fmck = Fsys / (SYSCLK_PRES))
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_1
#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_2
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_4
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_8
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_16
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_32/
/#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_64
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_3

// ===== PLL0 (A)任意選択    (Fpll = (Fclk * PLL_mul) / PLL_div)
// ここにmulとdivの実効値を使ってください。
#define CONFIG_PLL0_SOURCE          PLL_SRC_MAINCK_XTAL
```

```
#define CONFIG_PLL0_MUL              16
#define CONFIG_PLL0_DIV              1

// ===== 480MHzで固定化されたUPLL (UTMI)ﾊｰﾄﾞｳｪｱ

// ===== UPLLで固定化されたUSBｸﾛｯｸ元

// ===== 目的対象周波数 (ｼｽﾃﾑ ｸﾛｯｸ)
// - XTAL周波数: 12MHz
// - ｼｽﾃﾑ ｸﾛｯｸ元: PLLA
// - ｼｽﾃﾑ ｸﾛｯｸ前置分周器: 2 (2分周)
// - PLLA供給元: XTAL
// - PLLA出力: XTAL * 16 / 1
// - ｼｽﾃﾑ ｸﾛｯｸは: 12 * 16 / 1 / 2 = 96MHz
// ===== 目的対象周波数 (USBｸﾛｯｸ)
// - USBｸﾛｯｸ元: UPLL
// - UPLL周波数: 480MHz
// - USBｸﾛｯｸ: 480MHz

#endif /* CONF_CLOCK_H_INCLUDED */
```

6.3.2.7. SAM3X, SAM3Aﾃﾞﾊﾞｲｽ (UOTGHS:USB OTG 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== ｼｽﾃﾑ ｸﾛｯｸ(MCK)元任意選択
//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_SLCK_RC
//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_SLCK_XTAL
//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_SLCK_BYPASS
//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_MAINCK_4M_RC
//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_MAINCK_8M_RC
//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_MAINCK_12M_RC
//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_MAINCK_XTAL
//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_MAINCK_BYPASS
#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_PLLACK
//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_UPLLCK

// ===== ｼｽﾃﾑ ｸﾛｯｸ(MCK)前置分周器任意選択    (Fmck = Fsys / (SYSCLK_PRES))
//#define CONFIG_SYSCLK_PRES           SYSCLK_PRES_1
#define CONFIG_SYSCLK_PRES           SYSCLK_PRES_2
//#define CONFIG_SYSCLK_PRES           SYSCLK_PRES_4
//#define CONFIG_SYSCLK_PRES           SYSCLK_PRES_8
//#define CONFIG_SYSCLK_PRES           SYSCLK_PRES_16
//#define CONFIG_SYSCLK_PRES           SYSCLK_PRES_32
//#define CONFIG_SYSCLK_PRES           SYSCLK_PRES_64
//#define CONFIG_SYSCLK_PRES           SYSCLK_PRES_3

// ===== PLL0 (A)任意選択    (Fpll = (Fclk * PLL_mul) / PLL_div)
// ここにmulとdivの実効値を使ってください。
#define CONFIG_PLL0_SOURCE           PLL_SRC_MAINCK_XTAL
#define CONFIG_PLL0_MUL              14
#define CONFIG_PLL0_DIV              1

// ===== 480MHzで固定化されたUPLL (UTMI)ﾊｰﾄﾞｳｪｱ

// ===== USBｸﾛｯｸ元任意選択    (Fusb = FpllX / USB_div)
// ここにdivの実効値を使ってください。
```

```
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL0
#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_UPLL
#define CONFIG_USBCLK_DIV           1

// ===== 目的対象周波数 (システム クロック)
// - XTAL周波数: 12MHz
// - システム クロック元: PLLA
// - システム クロック前置分周器: 2 (2分周)
// - PLLA供給元: XTAL
// - PLLA出力: XTAL * 14 / 1
// - システム クロックは: 12 * 14 / 1 /2 = 84MHz
// ===== 目的対象周波数 (USBクロック)
// - USBクロック元: UPLL
// - USBクロック分周器: 1 (分周なし)
// - UPLL周波数: 480MHz
// - USBクロック: 480 / 1 = 480MHz

#endif /* CONF_CLOCK_H_INCLUDED */
```

6.3.3. conf_clocks.h

6.3.3.1. SAM D21ﾃﾞﾊﾞｲｽ (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#include <clock.h>

#ifndef CONF_CLOCKS_H_INCLUDED
#  define CONF_CLOCKS_H_INCLUDED

/* システム クロック ﾊﾞｽ構成設定 */
#  define CONF_CLOCK_CPU_CLOCK_FAILURE_DETECT     false
#  define CONF_CLOCK_FLASH_WAIT_STATES            2
#  define CONF_CLOCK_CPU_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBA_DIVIDER                 SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBB_DIVIDER                 SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBC_DIVIDER                 SYSTEM_MAIN_CLOCK_DIV_1

/* SYSTEM_CLOCK_SOURCE_OSC8M構成設定 - 内部8MHz発振器 */
#  define CONF_CLOCK_OSC8M_PRESCALER              SYSTEM_OSC8M_DIV_1
#  define CONF_CLOCK_OSC8M_ON_DEMAND              true
#  define CONF_CLOCK_OSC8M_RUN_IN_STANDBY         false

/* SYSTEM_CLOCK_SOURCE_XOSC構成設定 - 外部ｸﾛｯｸ/発振器 */
#  define CONF_CLOCK_XOSC_ENABLE                  false
#  define CONF_CLOCK_XOSC_EXTERNAL_CRYSTAL        SYSTEM_CLOCK_EXTERNAL_CRYSTAL
#  define CONF_CLOCK_XOSC_EXTERNAL_FREQUENCY      12000000UL
#  define CONF_CLOCK_XOSC_STARTUP_TIME            SYSTEM_XOSC_STARTUP_32768
#  define CONF_CLOCK_XOSC_AUTO_GAIN_CONTROL       true
#  define CONF_CLOCK_XOSC_ON_DEMAND               true
#  define CONF_CLOCK_XOSC_RUN_IN_STANDBY          false

/* SYSTEM_CLOCK_SOURCE_XOSC32K構成設定 - 外部32kHzｸﾘｽﾀﾙ/ｸﾛｯｸ発振器 */
#  define CONF_CLOCK_XOSC32K_ENABLE               false
#  define CONF_CLOCK_XOSC32K_EXTERNAL_CRYSTAL     SYSTEM_CLOCK_EXTERNAL_CRYSTAL
#  define CONF_CLOCK_XOSC32K_STARTUP_TIME         SYSTEM_XOSC32K_STARTUP_65536
#  define CONF_CLOCK_XOSC32K_AUTO_AMPLITUDE_CONTROL  false
#  define CONF_CLOCK_XOSC32K_ENABLE_1KHZ_OUPUT    false
#  define CONF_CLOCK_XOSC32K_ENABLE_32KHZ_OUTPUT  true
#  define CONF_CLOCK_XOSC32K_ON_DEMAND            true
#  define CONF_CLOCK_XOSC32K_RUN_IN_STANDBY       false
```

```
/* SYSTEM_CLOCK_SOURCE_OSC32K構成設定 - 内部32kHz発振器 */
#  define CONF_CLOCK_OSC32K_ENABLE                  false
#  define CONF_CLOCK_OSC32K_STARTUP_TIME            SYSTEM_OSC32K_STARTUP_130
#  define CONF_CLOCK_OSC32K_ENABLE_1KHZ_OUTPUT      true
#  define CONF_CLOCK_OSC32K_ENABLE_32KHZ_OUTPUT     true
#  define CONF_CLOCK_OSC32K_ON_DEMAND               true
#  define CONF_CLOCK_OSC32K_RUN_IN_STANDBY          false

/* SYSTEM_CLOCK_SOURCE_DFLL構成設定 - ﾃﾞｼﾞﾀﾙ周波数固定化閉路 */
#  define CONF_CLOCK_DFLL_ENABLE                    true
#  define CONF_CLOCK_DFLL_LOOP_MODE                 SYSTEM_CLOCK_DFLL_LOOP_MODE_USB_RECOVERY
#  define CONF_CLOCK_DFLL_ON_DEMAND                 true

/* DFLL開路動作構成設定 */
#  define CONF_CLOCK_DFLL_FINE_VALUE                (512)

/* DFLL閉路動作構成設定 */
#  define CONF_CLOCK_DFLL_SOURCE_GCLK_GENERATOR     GCLK_GENERATOR_1
#  define CONF_CLOCK_DFLL_MULTIPLY_FACTOR           (48000000 / 32768)
#  define CONF_CLOCK_DFLL_QUICK_LOCK                true
#  define CONF_CLOCK_DFLL_TRACK_AFTER_FINE_LOCK     true
#  define CONF_CLOCK_DFLL_KEEP_LOCK_ON_WAKEUP       true
#  define CONF_CLOCK_DFLL_ENABLE_CHILL_CYCLE        true
#  define CONF_CLOCK_DFLL_MAX_COARSE_STEP_SIZE      (0x1f / 4)
#  define CONF_CLOCK_DFLL_MAX_FINE_STEP_SIZE        (0xff / 4)

/* SYSTEM_CLOCK_SOURCE_DPLL構成設定 - ﾃﾞｼﾞﾀﾙ位相固定化閉路 */
#  define CONF_CLOCK_DPLL_ENABLE                    false
#  define CONF_CLOCK_DPLL_ON_DEMAND                 true
#  define CONF_CLOCK_DPLL_RUN_IN_STANDBY            false
#  define CONF_CLOCK_DPLL_LOCK_BYPASS               false
#  define CONF_CLOCK_DPLL_WAKE_UP_FAST              false
#  define CONF_CLOCK_DPLL_LOW_POWER_ENABLE          false

#  define CONF_CLOCK_DPLL_LOCK_TIME                 SYSTEM_CLOCK_SOURCE_DPLL_LOCK_TIME_DEFAULT
#  define CONF_CLOCK_DPLL_REFERENCE_CLOCK           SYSTEM_CLOCK_SOURCE_DPLL_REFERENCE_CLOCK_XOSC32K
#  define CONF_CLOCK_DPLL_FILTER                    SYSTEM_CLOCK_SOURCE_DPLL_FILTER_DEFAULT

#  define CONF_CLOCK_DPLL_REFERENCE_FREQUENCY       32768
#  define CONF_CLOCK_DPLL_REFERENCE_DIVIDER         1
#  define CONF_CLOCK_DPLL_OUTPUT_FREQUENCY          48000000

/* DPLL GCLK基準構成設定 */
# define CONF_CLOCK_DPLL_REFERENCE_GCLK_GENERATOR GCLK_GENERATOR_1
/* DPLL GCLK固定化計時器構成設定 */
# define CONF_CLOCK_DPLL_LOCK_GCLK_GENERATOR GCLK_GENERATOR_1

/* clocks_init走行時にGCLKを構成設定するためにこれをtrueに設定してください。
 * falseに設定した場合、GCLK発振器はclocks_init()で何も構成設定されません。*/
#  define CONF_CLOCK_CONFIGURE_GCLK                 true

/* GCLK発振器0構成設定 (主ｸﾛｯｸ) */
#  define CONF_CLOCK_GCLK_0_ENABLE                  true
#  define CONF_CLOCK_GCLK_0_RUN_IN_STANDBY          true
#  define CONF_CLOCK_GCLK_0_CLOCK_SOURCE            SYSTEM_CLOCK_SOURCE_DFLL
#  define CONF_CLOCK_GCLK_0_PRESCALER               1
#  define CONF_CLOCK_GCLK_0_OUTPUT_ENABLE           false

/* GCLK発振器1構成設定 */
#  define CONF_CLOCK_GCLK_1_ENABLE                  false
```

```
#  define CONF_CLOCK_GCLK_1_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_1_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_XOSC32K
#  define CONF_CLOCK_GCLK_1_PRESCALER              1
#  define CONF_CLOCK_GCLK_1_OUTPUT_ENABLE          false

/* GCLK発振器2構成設定 (RTC) */
#  define CONF_CLOCK_GCLK_2_ENABLE                 false
#  define CONF_CLOCK_GCLK_2_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_2_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC32K
#  define CONF_CLOCK_GCLK_2_PRESCALER              32
#  define CONF_CLOCK_GCLK_2_OUTPUT_ENABLE          false

/* GCLK発振器3構成設定 */
#  define CONF_CLOCK_GCLK_3_ENABLE                 false
#  define CONF_CLOCK_GCLK_3_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_3_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_3_PRESCALER              1
#  define CONF_CLOCK_GCLK_3_OUTPUT_ENABLE          false

/* GCLK発振器4構成設定 */
#  define CONF_CLOCK_GCLK_4_ENABLE                 false
#  define CONF_CLOCK_GCLK_4_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_4_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_4_PRESCALER              1
#  define CONF_CLOCK_GCLK_4_OUTPUT_ENABLE          false

/* GCLK発振器5構成設定 */
#  define CONF_CLOCK_GCLK_5_ENABLE                 false
#  define CONF_CLOCK_GCLK_5_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_5_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_5_PRESCALER              1
#  define CONF_CLOCK_GCLK_5_OUTPUT_ENABLE          false

/* GCLK発振器6構成設定 */
#  define CONF_CLOCK_GCLK_6_ENABLE                 false
#  define CONF_CLOCK_GCLK_6_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_6_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_6_PRESCALER              1
#  define CONF_CLOCK_GCLK_6_OUTPUT_ENABLE          false

/* GCLK発振器7構成設定 */
#  define CONF_CLOCK_GCLK_7_ENABLE                 false
#  define CONF_CLOCK_GCLK_7_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_7_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_7_PRESCALER              1
#  define CONF_CLOCK_GCLK_7_OUTPUT_ENABLE          false

/* GCLK発振器8構成設定 */
#  define CONF_CLOCK_GCLK_8_ENABLE                 false
#  define CONF_CLOCK_GCLK_8_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_8_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_8_PRESCALER              1
#  define CONF_CLOCK_GCLK_8_OUTPUT_ENABLE          false

#endif /* CONF_CLOCKS_H_INCLUDED */
```

6.3.4. conf_board.h

6.3.4.1. AT32UC3A0, AT32UC3A1, AT32UC3Bﾃﾞﾊﾞｲｽ (USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */
```

```
#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// AT45DBX部許可
#define CONF_BOARD_AT45DBX
// SDとMMCｶｰﾄﾞ部許可
#define CONF_BOARD_SD_MMC_SPI

#endif /* CONF_BOARD_H_INCLUDED */
```

6.3.4.2. AT32UC3A3, AT32UC3A4ﾃﾞﾊﾞｲｽ (高速(HS)支援付USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// AT45DBX部許可
#define CONF_BOARD_AT45DBX
// SDとMMCｶｰﾄﾞ部許可
#define CONF_BOARD_SD_MMC_MCI

#endif /* CONF_BOARD_H_INCLUDED */
```

6.3.4.3. AT32UC3C, ATUCXXD, ATUCXXL3U, ATUCXXL4Uﾃﾞﾊﾞｲｽ (USBC)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// AT45DBX部許可
#define CONF_BOARD_AT45DBX
// SDとMMCｶｰﾄﾞ部許可
#define CONF_BOARD_SD_MMC_SPI

#endif /* CONF_BOARD_H_INCLUDED */
```

6.3.4.4. SAM3X, SAM3Aﾃﾞﾊﾞｲｽ (UOTGHS:USB OTG 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

/* NANDﾌﾗｯｼｭを使用 */
#define CONF_BOARD_NAND

// HSMCIを通してSD/MMCｲﾝﾀｰﾌｪｰｽを許可
#define CONF_BOARD_SD_MMC_HSMCI

/* USBﾋﾟﾝを使用 */
#define CONF_BOARD_USB_PORT

#endif /* CONF_BOARD_H_INCLUDED */
```

6.3.4.5. SAM D21デバイス (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

/* USB VBUS検出許可 */
#define CONF_BOARD_USB_VBUS_DETECT

#endif /* CONF_BOARD_H_INCLUDED */
```

6.3.5. conf_access.h

6.3.5.1. AT32UC3A0, AT32UC3A1, AT32UC3Bデバイス (USBB)

EVK1100ではAT45DBxと1つのSD/MMCがMSC用です。

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_ACCESS_H_
#define _CONF_ACCESS_H_

#include "compiler.h"
#include "board.h"

#define LUN_0                   DISABLE
#define LUN_1                   ENABLE
#define LUN_2                   ENABLE
#define LUN_3                   DISABLE
#define LUN_4                   DISABLE
#define LUN_5                   DISABLE
#define LUN_6                   DISABLE
#define LUN_7                   DISABLE
#define LUN_USB                 DISABLE

#define VIRTUAL_MEM                             LUN_0
#define LUN_ID_VIRTUAL_MEM                      LUN_ID_0
#define LUN_0_INCLUDE                           "virtual_mem.h"
#define Lun_0_test_unit_ready                   virtual_test_unit_ready
#define Lun_0_read_capacity                     virtual_read_capacity
#define Lun_0_wr_protect                        virtual_wr_protect
#define Lun_0_removal                           virtual_removal
#define Lun_0_usb_read_10                       virtual_usb_read_10
#define Lun_0_usb_write_10                      virtual_usb_write_10
#define Lun_0_mem_2_ram                         virtual_mem_2_ram
#define Lun_0_ram_2_mem                         virtual_ram_2_mem
#define LUN_0_NAME                              "¥"On-Chip Virtual Memory¥""

#define AT45DBX_MEM                             LUN_1
#define LUN_ID_AT45DBX_MEM                      LUN_ID_1
#define LUN_1_INCLUDE                           "at45dbx_mem.h"
#define Lun_1_test_unit_ready                   at45dbx_test_unit_ready
#define Lun_1_read_capacity                     at45dbx_read_capacity
#define Lun_1_wr_protect                        at45dbx_wr_protect
#define Lun_1_removal                           at45dbx_removal
#define Lun_1_usb_read_10                       at45dbx_usb_read_10
#define Lun_1_usb_write_10                      at45dbx_usb_write_10
#define Lun_1_mem_2_ram                         at45dbx_df_2_ram
#define Lun_1_ram_2_mem                         at45dbx_ram_2_df
```

```
#define LUN_1_NAME                              "¥"AT45DBX Data Flash¥""

#define SD_MMC_0_MEM                            LUN_2
#define LUN_ID_SD_MMC_0_MEM                     LUN_ID_2
#define LUN_2_INCLUDE                           "sd_mmc_mem.h"
#define Lun_2_test_unit_ready                   sd_mmc_test_unit_ready_0
#define Lun_2_read_capacity                     sd_mmc_read_capacity_0
#define Lun_2_wr_protect                        sd_mmc_wr_protect_0
#define Lun_2_removal                           sd_mmc_removal_0
#define Lun_2_usb_read_10                       sd_mmc_usb_read_10_0
#define Lun_2_usb_write_10                      sd_mmc_usb_write_10_0
#define Lun_2_mem_2_ram                         sd_mmc_mem_2_ram_0
#define Lun_2_ram_2_mem                         sd_mmc_ram_2_mem_0
#define LUN_2_NAME                              "¥"SD/MMC Card Slot 0¥""

#define MEM_USB                                 LUN_USB
#define LUN_ID_MEM_USB                          LUN_ID_USB
#define LUN_USB_INCLUDE                         "host_mem.h"
#define Lun_usb_test_unit_ready(lun)            host_test_unit_ready(lun)
#define Lun_usb_read_capacity(lun, nb_sect)     host_read_capacity(lun, nb_sect)
#define Lun_usb_read_sector_size(lun)           host_read_sector_size(lun)
#define Lun_usb_wr_protect(lun)                 host_wr_protect(lun)
#define Lun_usb_removal()                       host_removal()
#define Lun_usb_mem_2_ram(addr, ram)            host_read_10_ram(addr, ram)
#define Lun_usb_ram_2_mem(addr, ram)            host_write_10_ram(addr, ram)
#define LUN_USB_NAME                            "¥"Host Mass-Storage Memory¥""

#define memory_start_read_action(nb_sectors)    ui_start_read()
#define memory_stop_read_action()               ui_stop_read()
#define memory_start_write_action(nb_sectors)   ui_start_write()
#define memory_stop_write_action()              ui_stop_write()
#include "ui.h"

#define ACCESS_USB              true
#define ACCESS_MEM_TO_RAM       false
#define ACCESS_STREAM           false
#define ACCESS_STREAM_RECORD    false
#define ACCESS_MEM_TO_MEM       false
#define ACCESS_CODEC            false

#define GLOBAL_WR_PROTECT       false

#endif // _CONF_ACCESS_H_
```

6.3.5.2. AT32UC3A3, AT32UC3A4ﾃﾞﾊﾞｲｽ (高速(HS)支援付USBB)

EVK1104ではAT45DBxと2つのSD/MMCがMSC用です。

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_ACCESS_H_
#define _CONF_ACCESS_H_

#include "compiler.h"
#include "board.h"

#define LUN_0                   DISABLE
#define LUN_1                   ENABLE
#define LUN_2                   ENABLE
#define LUN_3                   ENABLE
#define LUN_4                   DISABLE
```

```
#define LUN_5                 DISABLE
#define LUN_6                 DISABLE
#define LUN_7                 DISABLE
#define LUN_USB               DISABLE

#define VIRTUAL_MEM                           LUN_0
#define LUN_ID_VIRTUAL_MEM                    LUN_ID_0
#define LUN_0_INCLUDE                         "virtual_mem.h"
#define Lun_0_test_unit_ready                 virtual_test_unit_ready
#define Lun_0_read_capacity                   virtual_read_capacity
#define Lun_0_wr_protect                      virtual_wr_protect
#define Lun_0_removal                         virtual_removal
#define Lun_0_usb_read_10                     virtual_usb_read_10
#define Lun_0_usb_write_10                    virtual_usb_write_10
#define Lun_0_mem_2_ram                       virtual_mem_2_ram
#define Lun_0_ram_2_mem                       virtual_ram_2_mem
#define LUN_0_NAME                            "¥"On-Chip Virtual Memory¥""

#define AT45DBX_MEM                           LUN_1
#define LUN_ID_AT45DBX_MEM                    LUN_ID_1
#define LUN_1_INCLUDE                         "at45dbx_mem.h"
#define Lun_1_test_unit_ready                 at45dbx_test_unit_ready
#define Lun_1_read_capacity                   at45dbx_read_capacity
#define Lun_1_wr_protect                      at45dbx_wr_protect
#define Lun_1_removal                         at45dbx_removal
#define Lun_1_usb_read_10                     at45dbx_usb_read_10
#define Lun_1_usb_write_10                    at45dbx_usb_write_10
#define Lun_1_mem_2_ram                       at45dbx_df_2_ram
#define Lun_1_ram_2_mem                       at45dbx_ram_2_df
#define LUN_1_NAME                            "¥"AT45DBX Data Flash¥""

#define SD_MMC_0_MEM                          LUN_2
#define LUN_ID_SD_MMC_0_MEM                   LUN_ID_2
#define LUN_2_INCLUDE                         "sd_mmc_mem.h"
#define Lun_2_test_unit_ready                 sd_mmc_test_unit_ready_0
#define Lun_2_read_capacity                   sd_mmc_read_capacity_0
#define Lun_2_wr_protect                      sd_mmc_wr_protect_0
#define Lun_2_removal                         sd_mmc_removal_0
#define Lun_2_usb_read_10                     sd_mmc_usb_read_10_0
#define Lun_2_usb_write_10                    sd_mmc_usb_write_10_0
#define Lun_2_mem_2_ram                       sd_mmc_mem_2_ram_0
#define Lun_2_ram_2_mem                       sd_mmc_ram_2_mem_0
#define LUN_2_NAME                            "¥"SD/MMC Card Slot 0¥""

#define SD_MMC_1_MEM                          LUN_3
#define LUN_ID_SD_MMC_1_MEM                   LUN_ID_3
#define LUN_3_INCLUDE                         "sd_mmc_mem.h"
#define Lun_3_test_unit_ready                 sd_mmc_test_unit_ready_1
#define Lun_3_read_capacity                   sd_mmc_read_capacity_1
#define Lun_3_wr_protect                      sd_mmc_wr_protect_1
#define Lun_3_removal                         sd_mmc_removal_1
#define Lun_3_usb_read_10                     sd_mmc_usb_read_10_1
#define Lun_3_usb_write_10                    sd_mmc_usb_write_10_1
#define Lun_3_mem_2_ram                       sd_mmc_mem_2_ram_1
#define Lun_3_ram_2_mem                       sd_mmc_ram_2_mem_1
#define LUN_3_NAME                            "¥"SD/MMC Card Slot 1¥""

#define MEM_USB                               LUN_USB
#define LUN_ID_MEM_USB                        LUN_ID_USB
#define LUN_USB_INCLUDE                       "host_mem.h"
#define Lun_usb_test_unit_ready(lun)          host_test_unit_ready(lun)
```

```
#define Lun_usb_read_capacity(lun, nb_sect)     host_read_capacity(lun, nb_sect)
#define Lun_usb_read_sector_size(lun)           host_read_sector_size(lun)
#define Lun_usb_wr_protect(lun)                 host_wr_protect(lun)
#define Lun_usb_removal()                       host_removal()
#define Lun_usb_mem_2_ram(addr, ram)            host_read_10_ram(addr, ram)
#define Lun_usb_ram_2_mem(addr, ram)            host_write_10_ram(addr, ram)
#define LUN_USB_NAME                            "¥"Host Mass-Storage Memory¥""

#define memory_start_read_action(nb_sectors)    ui_start_read()
#define memory_stop_read_action()               ui_stop_read()
#define memory_start_write_action(nb_sectors)   ui_start_write()
#define memory_stop_write_action()              ui_stop_write()
#include "ui.h"

#define ACCESS_USB            true
#define ACCESS_MEM_TO_RAM     false
#define ACCESS_STREAM         false
#define ACCESS_STREAM_RECORD  false
#define ACCESS_MEM_TO_MEM     false
#define ACCESS_CODEC          false

#define GLOBAL_WR_PROTECT     false

#endif // _CONF_ACCESS_H_
```

6.3.5.3. AT32UC3C, ATUCXXD, ATUCXXL3U, ATUCXXL4Uﾃﾞﾊﾞｲｽ (USBC)

EVK1100ではAT45DBxと1つのSD/MMCがMSC用です。

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_ACCESS_H_
#define _CONF_ACCESS_H_

#include "compiler.h"
#include "board.h"

#define LUN_0                 DISABLE
#define LUN_1                 ENABLE
#define LUN_2                 ENABLE
#define LUN_3                 DISABLE
#define LUN_4                 DISABLE
#define LUN_5                 DISABLE
#define LUN_6                 DISABLE
#define LUN_7                 DISABLE
#define LUN_USB               DISABLE

#define VIRTUAL_MEM                             LUN_0
#define LUN_ID_VIRTUAL_MEM                      LUN_ID_0
#define LUN_0_INCLUDE                           "virtual_mem.h"
#define Lun_0_test_unit_ready                   virtual_test_unit_ready
#define Lun_0_read_capacity                     virtual_read_capacity
#define Lun_0_wr_protect                        virtual_wr_protect
#define Lun_0_removal                           virtual_removal
#define Lun_0_usb_read_10                       virtual_usb_read_10
#define Lun_0_usb_write_10                      virtual_usb_write_10
#define Lun_0_mem_2_ram                         virtual_mem_2_ram
#define Lun_0_ram_2_mem                         virtual_ram_2_mem
#define LUN_0_NAME                              "¥"On-Chip Virtual Memory¥""

#define AT45DBX_MEM                             LUN_1
```

```
#define LUN_ID_AT45DBX_MEM                  LUN_ID_1d_10_ram(addr, ram)
#define LUN_1_INCLUDE                       "at45dbx_mem.h"
#define Lun_1_test_unit_ready               at45dbx_test_unit_ready
#define Lun_1_read_capacity                 at45dbx_read_capacity
#define Lun_1_wr_protect                    at45dbx_wr_protect
#define Lun_1_removal                       at45dbx_removal
#define Lun_1_usb_read_10                   at45dbx_usb_read_10
#define Lun_1_usb_write_10                  at45dbx_usb_write_10
#define Lun_1_mem_2_ram                     at45dbx_df_2_ram
#define Lun_1_ram_2_mem                     at45dbx_ram_2_df
#define LUN_1_NAME                          "¥"AT45DBX Data Flash¥""

#define SD_MMC_0_MEM                          LUN_2
#define LUN_ID_SD_MMC_0_MEM                   LUN_ID_2
#define LUN_2_INCLUDE                         "sd_mmc_mem.h"
#define Lun_2_test_unit_ready                 sd_mmc_test_unit_ready_0
#define Lun_2_read_capacity                   sd_mmc_read_capacity_0
#define Lun_2_wr_protect                      sd_mmc_wr_protect_0
#define Lun_2_removal                         sd_mmc_removal_0
#define Lun_2_usb_read_10                     sd_mmc_usb_read_10_0
#define Lun_2_usb_write_10                    sd_mmc_usb_write_10_0
#define Lun_2_mem_2_ram                       sd_mmc_mem_2_ram_0
#define Lun_2_ram_2_mem                       sd_mmc_ram_2_mem_0
#define LUN_2_NAME                            "¥"SD/MMC Card Slot 0¥""

#define MEM_USB                             LUN_USB
#define LUN_ID_MEM_USB                      LUN_ID_USB
#define LUN_USB_INCLUDE                     "host_mem.h"
#define Lun_usb_test_unit_ready(lun)        host_test_unit_ready(lun)
#define Lun_usb_read_capacity(lun, nb_sect) host_read_capacity(lun, nb_sect)
#define Lun_usb_read_sector_size(lun)       host_read_sector_size(lun)
#define Lun_usb_wr_protect(lun)             host_wr_protect(lun)
#define Lun_usb_removal()                   host_removal()
#define Lun_usb_mem_2_ram(addr, ram)        host_read_10_ram(addr, ram)
#define Lun_usb_ram_2_mem(addr, ram)        host_write_10_ram(addr, ram)
#define LUN_USB_NAME                        "¥"Host Mass-Storage Memory¥""

#define memory_start_read_action(nb_sectors)    ui_start_read()
#define memory_stop_read_action()               ui_stop_read()
#define memory_start_write_action(nb_sectors)   ui_start_write()
#define memory_stop_write_action()              ui_stop_write()
#include "ui.h"

#define ACCESS_USB            true
#define ACCESS_MEM_TO_RAM     false
#define ACCESS_STREAM         false
#define ACCESS_STREAM_RECORD  false
#define ACCESS_MEM_TO_MEM     false
#define ACCESS_CODEC          false

#define GLOBAL_WR_PROTECT     false

#endif // _CONF_ACCESS_H_
```

6.3.5.4. SAM3X, SAM3Aﾃﾞﾊﾞｲｽ (UOTGHS:USB OTG 高速(HS))

SAM3X-EKではSD/MMCと基板上のNANDがMSC用です。

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_ACCESS_H_
```

```
#define _CONF_ACCESS_H_

#include "compiler.h"
#include "board.h"

#define LUN_0                   DISABLE
#define LUN_1                   DISABLE
#define LUN_2                   ENABLE
#define LUN_3                   DISABLE
#define LUN_4                   ENABLE
#define LUN_5                   DISABLE
#define LUN_6                   DISABLE
#define LUN_7                   DISABLE
#define LUN_USB                 DISABLE

#define VIRTUAL_MEM                             LUN_0
#define LUN_ID_VIRTUAL_MEM                      LUN_ID_0
#define LUN_0_INCLUDE                           "virtual_mem.h"
#define Lun_0_test_unit_ready                   virtual_test_unit_ready
#define Lun_0_read_capacity                     virtual_read_capacity
#define Lun_0_unload                            NULL /* ｱﾝﾛｰﾄﾞ不可 */
#define Lun_0_wr_protect                        virtual_wr_protect
#define Lun_0_removal                           virtual_removal
#define Lun_0_usb_read_10                       virtual_usb_read_10
#define Lun_0_usb_write_10                      virtual_usb_write_10
#define Lun_0_mem_2_ram                         virtual_mem_2_ram
#define Lun_0_ram_2_mem                         virtual_ram_2_mem
#define LUN_0_NAME                              "¥"On-Chip Virtual Memory¥""

#define AT45DBX_MEM                             LUN_1
#define LUN_ID_AT45DBX_MEM                      LUN_ID_1
#define LUN_1_INCLUDE                           "at45dbx_mem.h"
#define Lun_1_test_unit_ready                   at45dbx_test_unit_ready
#define Lun_1_read_capacity                     at45dbx_read_capacity
#define Lun_1_unload                            NULL /* ｱﾝﾛｰﾄﾞ不可 */
#define Lun_1_wr_protect                        at45dbx_wr_protect
#define Lun_1_removal                           at45dbx_removal
#define Lun_1_usb_read_10                       at45dbx_usb_read_10
#define Lun_1_usb_write_10                      at45dbx_usb_write_10
#define Lun_1_mem_2_ram                         at45dbx_df_2_ram
#define Lun_1_ram_2_mem                         at45dbx_ram_2_df
#define LUN_1_NAME                              "¥"AT45DBX Data Flash¥""

#define SD_MMC_0_MEM                            LUN_2
#define LUN_ID_SD_MMC_0_MEM                     LUN_ID_2
#define LUN_2_INCLUDE                           "sd_mmc_mem.h"
#define Lun_2_test_unit_ready                   sd_mmc_test_unit_ready_0
#define Lun_2_read_capacity                     sd_mmc_read_capacity_0
#define Lun_2_unload                            sd_mmc_unload_0
#define Lun_2_wr_protect                        sd_mmc_wr_protect_0
#define Lun_2_removal                           sd_mmc_removal_0
#define Lun_2_usb_read_10                       sd_mmc_usb_read_10_0
#define Lun_2_usb_write_10                      sd_mmc_usb_write_10_0
#define Lun_2_mem_2_ram                         sd_mmc_mem_2_ram_0
#define Lun_2_ram_2_mem                         sd_mmc_ram_2_mem_0
#define LUN_2_NAME                              "¥"SD/MMC Card Slot 0¥""

#define NAND_FLASH_MEM                          LUN_4
#define LUN_ID_NAND_FLASH_MEM                   LUN_ID_4
#define LUN_4_INCLUDE                           "nand_flash_mem.h"
#define Lun_4_test_unit_ready                   nand_flash_test_unit_ready
```

```
#define Lun_4_read_capacity                     nand_flash_read_capacity
#define Lun_4_unload                            NULL
#define Lun_4_wr_protect                        nand_flash_wr_protect
#define Lun_4_removal                           nand_flash_removal
#define Lun_4_usb_read_10                       nand_flash_usb_read_10
#define Lun_4_usb_write_10                      nand_flash_usb_write_10
#define Lun_4_mem_2_ram                         nand_flash_mem_2_ram
#define Lun_4_ram_2_mem                         nand_flash_ram_2_mem
#define LUN_4_NAME                              "¥"nand_flash on EBI¥""

#define MEM_USB                                 LUN_USB
#define LUN_ID_MEM_USB                          LUN_ID_USB
#define LUN_USB_INCLUDE                         "uhi_msc_mem.h"
#define Lun_usb_get_lun()                       uhi_msc_mem_get_lun()
#define Lun_usb_test_unit_ready(lun)            uhi_msc_mem_test_unit_ready(lun)
#define Lun_usb_read_capacity(lun, nb_sect)     uhi_msc_mem_read_capacity(lun, nb_sect)
#define Lun_usb_read_sector_size(lun)           uhi_msc_mem_read_sector_size(lun)
#define Lun_usb_wr_protect(lun)                 uhi_msc_mem_wr_protect(lun)
#define Lun_usb_removal()                       uhi_msc_mem_removal()
#define Lun_usb_mem_2_ram(addr, ram)            uhi_msc_mem_read_10_ram(addr, ram)
#define Lun_usb_ram_2_mem(addr, ram)            uhi_msc_mem_write_10_ram(addr, ram)
#define LUN_USB_NAME                            "¥"Host Mass-Storage Memory¥""

#define memory_start_read_action(nb_sectors)    ui_start_read()
#define memory_stop_read_action()               ui_stop_read()
#define memory_start_write_action(nb_sectors)   ui_start_write()
#define memory_stop_write_action()              ui_stop_write()
#include "ui.h"

#define ACCESS_USB              true
#define ACCESS_MEM_TO_RAM       false
#define ACCESS_STREAM           false
#define ACCESS_STREAM_RECORD    false
#define ACCESS_MEM_TO_MEM       false
#define ACCESS_CODEC            false

#define GLOBAL_WR_PROTECT       false

#endif // _CONF_ACCESS_H_
```

6.3.5.5. SAM D21ﾃﾞﾊﾞｲｽ (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_ACCESS_H_
#define _CONF_ACCESS_H_

#include "compiler.h"
#include "board.h"

#define LUN_0                   ENABLE
#define LUN_1                   DISABLE
#define LUN_2                   DISABLE
#define LUN_3                   DISABLE
#define LUN_4                   DISABLE
#define LUN_5                   DISABLE
#define LUN_6                   DISABLE
#define LUN_7                   DISABLE
#define LUN_USB                 DISABLE
```

```
#define VIRTUAL_MEM                             LUN_0
#define LUN_ID_VIRTUAL_MEM                      LUN_ID_0
#define LUN_0_INCLUDE                           "virtual_mem.h"
#define Lun_0_test_unit_ready                   virtual_test_unit_ready
#define Lun_0_read_capacity                     virtual_read_capacity
#define Lun_0_unload                            NULL /* ｱﾝﾛｰﾄﾞ不可 */
#define Lun_0_wr_protect                        virtual_wr_protect
#define Lun_0_removal                           virtual_removal
#define Lun_0_usb_read_10                       virtual_usb_read_10
#define Lun_0_usb_write_10                      virtual_usb_write_10
#define Lun_0_mem_2_ram                         virtual_mem_2_ram
#define Lun_0_ram_2_mem                         virtual_ram_2_mem
#define LUN_0_NAME                              "¥"On-Chip Virtual Memory¥""

#define MEM_USB                                 LUN_USB
#define LUN_ID_MEM_USB                          LUN_ID_USB
#define LUN_USB_INCLUDE                         "uhi_msc_mem.h"
#define Lun_usb_get_lun()                       uhi_msc_mem_get_lun()
#define Lun_usb_test_unit_ready(lun)            uhi_msc_mem_test_unit_ready(lun)
#define Lun_usb_read_capacity(lun, nb_sect)     uhi_msc_mem_read_capacity(lun, nb_sect)
#define Lun_usb_read_sector_size(lun)           uhi_msc_mem_read_sector_size(lun)
#define Lun_usb_wr_protect(lun)                 uhi_msc_mem_wr_protect(lun)
#define Lun_usb_removal()                       uhi_msc_mem_removal()
#define Lun_usb_mem_2_ram(addr, ram)            uhi_msc_mem_read_10_ram(addr, ram)
#define Lun_usb_ram_2_mem(addr, ram)            uhi_msc_mem_write_10_ram(addr, ram)
#define LUN_USB_NAME                            "¥"Host Mass-Storage Memory¥""

#define memory_start_read_action(nb_sectors)    ui_start_read()
#define memory_stop_read_action()               ui_stop_read()
#define memory_start_write_action(nb_sectors)   ui_start_write()
#define memory_stop_write_action()              ui_stop_write()
#include "ui.h"

#define ACCESS_USB              true
#define ACCESS_MEM_TO_RAM       false
#define ACCESS_STREAM           false
#define ACCESS_STREAM_RECORD    false
#define ACCESS_MEM_TO_MEM       false
#define ACCESS_CODEC            false

#define GLOBAL_WR_PROTECT       false

#endif // _CONF_ACCESS_H_
```

6.3.6. conf_virtual_mem.h

6.3.6.1. ﾁｯﾌﾟ上仮想ﾒﾓﾘ ﾃﾞｨｽｸ

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_VIRTUAL_MEM_H_
#define _CONF_VIRTUAL_MEM_H_

#define VMEM_NB_SECTOR 48 // 内部RAM 24KB (20KB以上であるべき、さもなければPCはﾌｫｰﾏｯﾄできません。)

#endif // _CONF_VIRTUAL_MEM_H_
```

6.3.6.2. 基板上仮想ﾒﾓﾘ ﾃﾞｨｽｸ

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
```

```
 */

#ifndef _CONF_VIRTUAL_MEM_H_
#define _CONF_VIRTUAL_MEM_H_

// 仮想ﾒﾓﾘが外部RAM(PSRAM)の場合の開始ｱﾄﾞﾚｽ
#define VMEM_ADDRESS    0x60000000
#define VMEM_NB_SECTOR 2048 // 外部疑似SRAM 1MB

#endif // _CONF_VIRTUAL_MEM_H_
```

7. 供給者ｸﾗｽ装置用USB装置ｲﾝﾀｰﾌｪｰｽ(UDI)

供給者ｸﾗｽ装置用USB装置ｲﾝﾀｰﾌｪｰｽ(UDI)はUSB供給者装置の構成設定と管理に関するｲﾝﾀｰﾌｪｰｽを提供します。

この資料の概要は以下のとおりです。

- API概要
- USB装置供給者単位部(UDI供給者)用の即時開始の手引き
- 構成設定ﾌｧｲﾙ例

Atmel®ｿﾌﾄｳｪｱ枠組み(ASF) USB装置階層とUSB装置供給者に関するより多くの詳細については以下の応用記述を参照してください。

- AVR4900：ASF - USB装置階層
- AVR4901：ASF - USB装置供給者ｸﾗｽ応用
- AVR4920：ASF - USB装置階層 - 適合と性能係数
- AVR4921：ASF - USB装置階層 - ASF V1とV2間の違い

7.1. API概要

7.1.1. 変数と型定義

7.1.1.1. USB装置ｺｱ(UDC)とのｲﾝﾀｰﾌｪｰｽ

UDCによって必要とされる変数

udi_api_vendor変数

```
UDC_DESC_STORAGE udi_api_t udi_api_vendor
```

UDC用の標準UDI APIを含む全域構造体

7.1.2. 構造体定義

7.1.2.1. udi_vendor_desc_t構造体

供給者ｸﾗｽ ｲﾝﾀｰﾌｪｰｽ用ｲﾝﾀｰﾌｪｰｽ記述子構造体

表7-1. ﾒﾝﾊﾞ

型	名前	説明
usb_iface_desc_t	iface0	標準USBｲﾝﾀｰﾌｪｰｽ記述子構造体
usb_iface_desc_t	iface1	標準USBｲﾝﾀｰﾌｪｰｽ記述子構造体

7.1.3. ﾏｸﾛ定義

7.1.3.1. USBｲﾝﾀｰﾌｪｰｽ記述子

以下の構造体は予め定義されたUSBｲﾝﾀｰﾌｪｰｽ記述子を提供します。これは最終的なUSB記述子を定義するのに使われなければなりません。

UDI_VENDOR_EPS_INT_DESC ﾏｸﾛ

```
#define UDI_VENDOR_EPS_INT_DESC
```

ｴﾝﾄﾞﾎﾟｲﾝﾄ記述子

UDI_VENDOR_EPS_INT_DESC_FS ﾏｸﾛ

```
#define UDI_VENDOR_EPS_INT_DESC_FS
```

UDI_VENDOR_EPS_INT_DESC_HS ﾏｸﾛ

```
#define UDI_VENDOR_EPS_INT_DESC_HS
```

UDI_VENDOR_EPS_BULK_DESC ﾏｸﾛ

```
#define UDI_VENDOR_EPS_BULK_DESC
```

UDI_VENDOR_EPS_BULK_DESC_FS ﾏｸﾛ

```
#define UDI_VENDOR_EPS_BULK_DESC_FS
```

UDI_VENDOR_EPS_BULK_DESC_HS ﾏｸﾛ

```
#define UDI_VENDOR_EPS_BULK_DESC_HS
```

UDI_VENDOR_EPS_ISO_DESC ﾏｸﾛ

```
#define UDI_VENDOR_EPS_ISO_DESC
```

UDI_VENDOR_EPS_ISO_DESC_FS マクロ

```
#define UDI_VENDOR_EPS_ISO_DESC_FS
```

UDI_VENDOR_EPS_ISO_DESC_HS マクロ

```
#define UDI_VENDOR_EPS_ISO_DESC_HS
```

UDI_VENDOR_STRING_ID マクロ

```
#define UDI_VENDOR_STRING_ID
```

既定によってこのインターフェースと提携する文字列はありません。

UDI_VENDOR_EP_NB_INT マクロ

```
#define UDI_VENDOR_EP_NB_INT
```

供給者インターフェースによって使用される最大6つのエンドポイント

UDI_VENDOR_EP_NB_BULK マクロ

```
#define UDI_VENDOR_EP_NB_BULK
```

UDI_VENDOR_EP_NB_ISO マクロ

```
#define UDI_VENDOR_EP_NB_ISO
```

UDI_VENDOR_EP_NB マクロ

```
#define UDI_VENDOR_EP_NB
```

UDI_VENDOR_DESC マクロ

```
#define UDI_VENDOR_DESC
```

全ての速度に対する供給者インターフェース記述子の内容

UDI_VENDOR_DESC_FS マクロ

```
#define UDI_VENDOR_DESC_FS
```

全速(FS)専用供給者インターフェース記述子の内容

UDI_VENDOR_DESC_HS マクロ

```
#define UDI_VENDOR_DESC_HS
```

高速(HS)専用供給者インターフェース記述子の内容

7.1.4. 関数定義

7.1.4.1. 供給者クラス用USB装置インターフェース(UDI)

このUSBクラスを使うため上位応用によって使われる共通API

これらのルーチンはデータとUSB供給者エンドポイントとの転送に使われます。

USB装置供給者単位部に対する即時開始の手引きをご覧ください。

udi_vendor_interrupt_in_run()関数

割り込みINで転送開始

```
bool udi_vendor_interrupt_in_run( uint8_t * buf, iram_size_t buf_size,
  udd_callback_trans_t callback )
```

転送が終了または失敗(中断停止、リセットなど)された時に呼び戻しが呼ばれます。この呼び戻しは転送状態と結果的に転送されたバイト数を返します。

表7-2. パラメータ

パラメータ名	データ方向	説明
buf	[入力]	送るまたは満たすための内部RAM上の緩衝部。これは整列でなければならず、COMPILER_WORD_ALIGNEDを使用してください。
buf_size	[入力]	送るまたは満たすための緩衝部の大きさ
callback	[入力]	NULLまたは転送の最後で呼ぶ関数

戻り値：関数が成功裏に終了した場合に1、さもなければ0

udi_vendor_interrupt_out_run()関数

割り込みOUTで転送開始

```
bool udi_vendor_interrupt_out_run( uint8_t * buf, iram_size_t buf_size,
  udd_callback_trans_t callback)
```

転送が終了または失敗(中断停止、ﾘｾｯﾄなど)された時に呼び戻しが呼ばれます。この呼び戻しは転送状態と結果的に転送されたﾊﾞｲﾄ数を返します。

表7-3. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
buf	[入力]	送るまたは満たすための内部RAM上の緩衝部。これは整列でなければならず、COMPILER_WORD_ALIGNEDを使ってください。
buf_size	[入力]	送るまたは満たすための緩衝部の大きさ
callback	[入力]	NULLまたは転送の最後で呼ぶ関数

戻り値：関数が成功裏に終了した場合に1、さもなければ0

udi_vendor_bulk_in_run()関数

大量(ﾊﾞﾙｸ)INで転送開始

```
bool udi_vendor_bulk_in_run( uint8_t * buf, iram_size_t buf_size, udd_callback_trans_t callback )
```

転送が終了または失敗(中断停止、ﾘｾｯﾄなど)された時に呼び戻しが呼ばれます。この呼び戻しは転送状態と結果的に転送されたﾊﾞｲﾄ数を返します。

表7-4. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
buf	[入力]	送るまたは満たすための内部RAM上の緩衝部。これは整列でなければならず、COMPILER_WORD_ALIGNEDを使ってください。
buf_size	[入力]	送るまたは満たすための緩衝部の大きさ
callback	[入力]	NULLまたは転送の最後で呼ぶ関数

戻り値：関数が成功裏に終了した場合に1、さもなければ0

udi_vendor_bulk_out_run()関数

大量(ﾊﾞﾙｸ)OUTで転送開始

```
bool udi_vendor_bulk_out_run( uint8_t * buf, iram_size_t buf_size,
  udd_callback_trans_t callback )
```

転送が終了または失敗(中断停止、ﾘｾｯﾄなど)された時に呼び戻しが呼ばれます。この呼び戻しは転送状態と結果的に転送されたﾊﾞｲﾄ数を返します。

表7-5. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
buf	[入力]	送るまたは満たすための内部RAM上の緩衝部。これは整列でなければならず、COMPILER_WORD_ALIGNEDを使ってください。
buf_size	[入力]	送るまたは満たすための緩衝部の大きさ
callback	[入力]	NULLまたは転送の最後で呼ぶ関数

戻り値：関数が成功裏に終了した場合に1、さもなければ0

udi_vendor_iso_in_run()関数

等時(ｱｲｿｸﾛﾅｽ)INで転送開始

```
bool udi_vendor_iso_in_run( uint8_t * buf, iram_size_t buf_size, udd_callback_trans_t callback )
```

転送が終了または失敗(中断停止、ﾘｾｯﾄなど)された時に呼び戻しが呼ばれます。この呼び戻しは転送状態と結果的に転送されたﾊﾞｲﾄ数を返します。

表7-6. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
buf	[入力]	送るまたは満たすための内部RAM上の緩衝部。これは整列でなければならず、COMPILER_WORD_ALIGNEDを使ってください。
buf_size	[入力]	送るまたは満たすための緩衝部の大きさ
callback	[入力]	NULLまたは転送の最後で呼ぶ関数

戻り値：関数が成功裏に終了した場合に1、さもなければ0

udi_vendor_iso_out_run()関数

等時(ｱｲｿｸﾛﾅｽ)OUTで転送開始

```
bool udi_vendor_iso_out_run( uint8_t * buf, iram_size_t buf_size, udd_callback_trans_t callback )
```

転送が終了または失敗(中断停止、ﾘｾｯﾄなど)された時に呼び戻しが呼ばれます。この呼び戻しは転送状態と結果的に転送されたﾊﾞｲﾄ数を返します。

表7-7. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
buf	[入力]	送るまたは満たすための内部RAM上の緩衝部。これは整列でなければならず、COMPILER_WORD_ALIGNEDを使ってください。
buf_size	[入力]	送るまたは満たすための緩衝部の大きさ
callback	[入力]	NULLまたは転送の最後で呼ぶ関数

戻り値：関数が成功裏に終了した場合に1、さもなければ0

7.2. USB装置供給者単位部(UDI供給者)用の即時開始の手引き

これは使用事例の選択に於いて単位部をどう構成設定して使うかを段階的に指示する、USB装置供給者単位部(UDI供給者)用の即時開始の手引きです。

使用事例は様々なｺｰﾄﾞの断片を含みます。構成設定に関する段階でのｺｰﾄﾞの断片は独自初期化関数内に複写することができ、一方使用に関する段階は例えば主応用関数内に複写することができます。

7.2.1. 基本的な使用事例

この使用事例では”USB Vendor (Single Class support)”単位部が使われます。”USB Vendor (Composite Device)”単位部の使い方は高度な使用事例で記述されます。

7.2.1.1. 構成設定段階

USB装置のため、共通USB装置構成設定段階に従います。USB装置基本構成設定を参照してください。

7.2.1.2. 使用段階

ｺｰﾄﾞ例

conf_usb.hの内容

```
#define UDI_VENDOR_ENABLE_EXT() my_callback_vendor_enable()
extern bool my_callback_vendor_enable(void);
#define UDI_VENDOR_DISABLE_EXT() my_callback_vendor_disable()
extern void my_callback_vendor_disable(void);

#define  UDI_VENDOR_SETUP_OUT_RECEIVED()  my_vendor_setup_out_received()
extern bool my_vendor_setup_out_received(void);
#define  UDI_VENDOR_SETUP_IN_RECEIVED()   my_vendor_setup_in_received()
extern bool my_vendor_setup_in_received(void);

#define  UDI_VENDOR_EPS_SIZE_INT_FS    64
#define  UDI_VENDOR_EPS_SIZE_BULK_FS   64
#define  UDI_VENDOR_EPS_SIZE_ISO_FS   256
#define  UDI_VENDOR_EPS_SIZE_INT_HS    64
#define  UDI_VENDOR_EPS_SIZE_BULK_HS  512
#define  UDI_VENDOR_EPS_SIZE_ISO_HS    64
#include "udi_vendor_conf.h" // conf_usb.hﾌｧｲﾙの最後で
```

応用Cﾌｧｲﾙに追加してください。

```
static bool my_flag_autorize_vendor_transfert = false;
bool my_callback_vendor_enable(void)
{
   my_flag_autorize_vendor_transfert = true;
   return true;}void my_callback_vendor_disable(void)
{
   my_flag_autorize_vendor_transfert = false;
}

uint8_t global_buffer[X];
```

```
{
   if (my_flag_autorize_vendor_transfert) {
     // OUT割り込みｴﾝﾄﾞﾎﾟｲﾝﾄでの転送許可
     udi_vendor_interrupt_out_run(
            global_buffer,
            sizeof(global_buffer),
            NULL);
     // IN割り込みｴﾝﾄﾞﾎﾟｲﾝﾄでの転送許可
     udi_vendor_interrupt_in_run(
            global_buffer,
            sizeof(global_buffer),
            NULL);
     ～
   }
}
```

作業の流れ

1. conf_usb.hが利用可能でUSB装置供給者構成設定である以下の構成設定を含むことを確実にしてください。

```
#define UDI_VENDOR_ENABLE_EXT() my_callback_vendor_enable()
extern bool my_callback_vendor_enable(void);
```

注：装置列挙(USB装置の検出と識別)後、USBﾎｽﾄは装置の構成設定を始めます。装置からのUSB供給者ｲﾝﾀｰﾌｪｰｽがﾎｽﾄによって受け入れられると、USBﾎｽﾄはこのｲﾝﾀｰﾌｪｰｽを許可してUDI_VENDOR_ENABLE_EXT()呼び戻し関数が呼ばれ、trueが返ります。故に、この事象を受け取った時に供給者転送を開始することができます。

```
#define UDI_VENDOR_DISABLE_EXT() my_callback_vendor_disable()
extern void my_callback_vendor_disable(void);
```

注：USB装置が切断されるか、またはUSBﾎｽﾄによってﾘｾｯﾄされると、USBｲﾝﾀｰﾌｪｰｽは禁止され、UDI_VENDOR_DISABLE_EXT()呼び戻し関数が呼ばれます。故に、供給者ﾃﾞｰﾀ転送を禁止することが推奨されます。

```
#define  UDI_VENDOR_SETUP_OUT_RECEIVED()  my_vendor_setup_out_received()
extern bool my_vendor_setup_out_received(void);
#define  UDI_VENDOR_SETUP_IN_RECEIVED()   my_vendor_setup_in_received()
extern bool my_vendor_setup_in_received(void);
```

注：供給者のｲﾝﾀｰﾌｪｰｽに対する制御要求はこれら両方の呼び戻しを通して処理されます。

```
#define  UDI_VENDOR_EPS_SIZE_INT_FS    64
#define  UDI_VENDOR_EPS_SIZE_BULK_FS   64
#define  UDI_VENDOR_EPS_SIZE_ISO_FS   256
#define  UDI_VENDOR_EPS_SIZE_INT_HS    64
#define  UDI_VENDOR_EPS_SIZE_BULK_HS  512
#define  UDI_VENDOR_EPS_SIZE_ISO_HS    64
```

注：ｴﾝﾄﾞﾎﾟｲﾝﾄの大きさは最終的な応用によって定義され、全速(FS)の大きさが0の場合に禁止することができます。

2. 割り込み、大量(ﾊﾞﾙｸ)、等時(ｱｲｿｸﾛﾅｽ)ｴﾝﾄﾞﾎﾟｲﾝﾄでの供給者転送はこれらの関数を通して行われます。

```
// 割り込みINでの転送開始
udi_vendor_interrupt_in_run();
// 割り込みOUTでの転送開始
udi_vendor_interrupt_out_run();
// 大量(ﾊﾞﾙｸ)INでの転送開始
udi_vendor_bulk_in_run();
// 大量(ﾊﾞﾙｸ)OUTでの転送開始
udi_vendor_bulk_out_run();
// 等時(ｱｲｿｸﾛﾅｽ)INでの転送開始
udi_vendor_iso_in_run();
// 等時(ｱｲｿｸﾛﾅｽ)OUTでの転送開始
udi_vendor_iso_out_run();
```

7.2.2. 高度な使用事例

UDI供給者単位部の複数ｲﾝﾀｰﾌｪｰｽの使用については以下をご覧ください。

・複合装置での供給者

UDI供給者単位部のもっと高度な使用については以下をご覧ください。

・USB装置の高度な使用事例

7.2.3. 複合装置での供給者

USB複合装置は1つよりも多くのUSBｸﾗｽを使うUSB装置です。この使用事例ではUSB複合装置を作成するのに”USB Vendor (Composite Device)”単位部が使われます。故に、このUSB単位部は”USB HID Mouse (Composite Device)”のような別な”複合装置(Composite Device)”単位部と連携することができます。

また、「AVR4902:ASF - USB複合装置」応用記述を参照することもできます。

7.2.3.1. 構成設定段階

動くためのこの事例の構成設定ｺｰﾄﾞについては基本的な使用事例に従わなければなりません。

7.2.3.2. 使用段階

ｺｰﾄﾞ例

conf_usb.hの内容

```
#define USB_DEVICE_EP_CTRL_SIZE  64
#define USB_DEVICE_NB_INTERFACE (X+1)
#define USB_DEVICE_MAX_EP (X) to (X+6)

#define  UDI_VENDOR_EP_INTERRUPT_IN  (1 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_INTERRUPT_OUT (2 | USB_EP_DIR_OUT)
#define  UDI_VENDOR_EP_BULK_IN       (3 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_BULK_OUT      (4 | USB_EP_DIR_OUT)
#define  UDI_VENDOR_EP_ISO_IN        (5 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_ISO_OUT       (6 | USB_EP_DIR_OUT)

#define UDI_VENDOR_IFACE_NUMBER  X
#define UDI_COMPOSITE_DESC_T ¥
   udi_vendor_desc_t udi_vendor; ¥
   ～
#define UDI_COMPOSITE_DESC_FS ¥
   .udi_vendor = UDI_VENDOR_DESC, ¥
   ～
#define UDI_COMPOSITE_DESC_HS ¥
   .udi_vendor = UDI_VENDOR_DESC, ¥
   ～
#define UDI_COMPOSITE_API ¥
   &udi_api_vendor, ¥
   ～
```

作業の流れ

1. conf_usb.hが利用可能でUSB複合装置構成設定に必要とされる以下のﾊﾟﾗﾒｰﾀを含むことを確実にしてください。

```
// ｴﾝﾄﾞﾎﾟｲﾝﾄ制御容量、これは以下でなければなりません。
// - 全速(FS)装置に対して、8, 16 ,32 または 64 (RAMを節約するために8が推奨されます。)
// - 高速(HS)装置に対して、64
#define USB_DEVICE_EP_CTRL_SIZE  64
// このUSB装置での総ｲﾝﾀｰﾌｪｰｽ数
// 供給者用に1を加算
#define USB_DEVICE_NB_INTERFACE (X+1)
// このUSB装置での総ｴﾝﾄﾞﾎﾟｲﾝﾄ数
// これには各ｲﾝﾀｰﾌｪｰｽに対する各々のｴﾝﾄﾞﾎﾟｲﾝﾄを含めなければなりません。
// 供給者ｲﾝﾀｰﾌｪｰｽ用に0～6を加算
// 数はUDI_VENDOR_EPS_SIZE_～_FS定義に依存します。
#define USB_DEVICE_MAX_EP (X)～(X+6)
```

2. conf_usb.hが複合装置の記述子を含むことを確実にしてください。

```
// あなたが選んだ供給者用のｴﾝﾄﾞﾎﾟｲﾝﾄ番号
// ｴﾝﾄﾞﾎﾟｲﾝﾄ番号は1から始まります。
#define  UDI_VENDOR_EP_INTERRUPT_IN  (1 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_INTERRUPT_OUT (2 | USB_EP_DIR_OUT)
#define  UDI_VENDOR_EP_BULK_IN       (3 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_BULK_OUT      (4 | USB_EP_DIR_OUT)
```

```
#define  UDI_VENDOR_EP_ISO_IN         (5 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_ISO_OUT        (6 | USB_EP_DIR_OUT)
// 0から始まるインターフェースのインターフェース指標
#define UDI_VENDOR_IFACE_NUMBER  X
```

3. conf_usb.hがUSB複合装置構成設定に必要とされる以下のパラメータを含むことを確実にしてください。

```
// USBインターフェース記述子構造体
#define UDI_COMPOSITE_DESC_T ¥
   〜
   udi_vendor_desc_t udi_vendor; ¥
   〜
// 全速(FS)用USBインターフェース記述子値
#define UDI_COMPOSITE_DESC_FS ¥
   〜
   .udi_vendor = UDI_VENDOR_DESC_FS, ¥
   〜
// 高速(HS)用USBインターフェース記述子値
#define UDI_COMPOSITE_DESC_HS ¥
   〜
   .udi_vendor = UDI_VENDOR_DESC_HS, ¥
   〜
// USBインターフェースAPI
#define UDI_COMPOSITE_API ¥
   〜
   &udi_api_vendor, ¥
   〜
```

注：上の4つの一覧で与えられた記述子の順番は全てのインターフェース指標によって定義された順番と同じでなければなりません。インターフェース指標の順番はUDI_X_IFACE_NUMBER定義を通して定義されます。

7.3. 構成設定ファイル例

7.3.1. conf_usb.h

7.3.1.1. 単一UDI供給者

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_USB_H_
#define _CONF_USB_H_

#include "compiler.h"

#warning You must refill the following definitions with a correct values

#define  USB_DEVICE_VENDOR_ID             USB_VID_ATMEL
#define  USB_DEVICE_PRODUCT_ID            USB_PID_ATMEL_ASF_VENDOR_CLASS
#define  USB_DEVICE_MAJOR_VERSION         1
#define  USB_DEVICE_MINOR_VERSION         0
#define  USB_DEVICE_POWER                 100 // VBUS線での消費(mA)
#define  USB_DEVICE_ATTR                  ¥
    (USB_CONFIG_ATTR_SELF_POWERED)
// (USB_CONFIG_ATTR_BUS_POWERED)
//   (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_SELF_POWERED)
//   (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_BUS_POWERED)

// #define  USB_DEVICE_MANUFACTURE_NAME      "Manufacture name"
// #define  USB_DEVICE_PRODUCT_NAME          "Product name"
// #define  USB_DEVICE_SERIAL_NAME           "12...EF"

#if (UC3A3||UC3A4)
# define  USB_DEVICE_HS_SUPPORT
```

```
#endif

// #define  UDC_VBUS_EVENT(b_vbus_high)      user_callback_vbus_action(b_vbus_high)
// extern void user_callback_vbus_action(bool b_vbus_high);
// #define  UDC_SOF_EVENT()                  user_callback_sof_action()
// extern void user_callback_sof_action(void);
// #define  UDC_SUSPEND_EVENT()              user_callback_suspend_action()
// extern void user_callback_suspend_action(void);
// #define  UDC_RESUME_EVENT()               user_callback_resume_action()
// extern void user_callback_resume_action(void);
// #define  UDC_REMOTEWAKEUP_ENABLE()        user_callback_remotewakeup_enable()
// extern void user_callback_remotewakeup_enable(void);
// #define  UDC_REMOTEWAKEUP_DISABLE()       user_callback_remotewakeup_disable()
// extern void user_callback_remotewakeup_disable(void);
// #define  UDC_GET_EXTRA_STRING()

#define UDI_VENDOR_ENABLE_EXT()          true
#define UDI_VENDOR_DISABLE_EXT()
#define UDI_VENDOR_SETUP_OUT_RECEIVED()  false
#define UDI_VENDOR_SETUP_IN_RECEIVED()   false
/*
 * #define UDI_VENDOR_ENABLE_EXT() my_callback_vendor_enable()
 * extern bool my_callback_vendor_enable(void);
 * #define UDI_VENDOR_DISABLE_EXT() my_callback_vendor_disable()
 * extern void my_callback_vendor_disable(void);
 *
 * #define  UDI_VENDOR_SETUP_OUT_RECEIVED()  my_vendor_setup_out_received()
 * extern bool my_vendor_setup_out_received(void);
 * #define  UDI_VENDOR_SETUP_IN_RECEIVED()   my_vendor_setup_in_received()
 * extern bool my_vendor_setup_in_received(void);
 */

#define UDI_VENDOR_EPS_SIZE_INT_FS    64
#define UDI_VENDOR_EPS_SIZE_BULK_FS   64
#if SAMG55
#define UDI_VENDOR_EPS_SIZE_ISO_FS   0
#else
#define UDI_VENDOR_EPS_SIZE_ISO_FS   256
#endif

#define UDI_VENDOR_EPS_SIZE_INT_HS    64
#define UDI_VENDOR_EPS_SIZE_BULK_HS  512
#define UDI_VENDOR_EPS_SIZE_ISO_HS    64

#include "udi_vendor_conf.h"

#endif // _CONF_USB_H_
```

7.3.1.2. 複数UDI供給者（複合）

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_USB_H_
#define _CONF_USB_H_

#include "compiler.h"

#warning You must refill the following definitions with a correct values

#define  USB_DEVICE_VENDOR_ID             USB_VID_ATMEL
```

```
#define  USB_DEVICE_PRODUCT_ID            0xFFFF
#define  USB_DEVICE_MAJOR_VERSION         1
#define  USB_DEVICE_MINOR_VERSION         0
#define  USB_DEVICE_POWER                 100 // VBUS線での消費(mA)
#define  USB_DEVICE_ATTR                  ¥
        (USB_CONFIG_ATTR_SELF_POWERED)
// (USB_CONFIG_ATTR_BUS_POWERED)
// (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_SELF_POWERED)
// (USB_CONFIG_ATTR_REMOTE_WAKEUP|USB_CONFIG_ATTR_BUS_POWERED)

// #define  USB_DEVICE_MANUFACTURE_NAME      "Manufacture name"
// #define  USB_DEVICE_PRODUCT_NAME          "Product name"
// #define  USB_DEVICE_SERIAL_NAME           "12...EF" // MSC用ﾃﾞｨｽｸ通番

//#define  USB_DEVICE_LOW_SPEED

#if (UC3A3||UC3A4)
//#define  USB_DEVICE_HS_SUPPORT
#endif

// #define  UDC_VBUS_EVENT(b_vbus_high)      user_callback_vbus_action(b_vbus_high)
// extern void user_callback_vbus_action(bool b_vbus_high);
// #define  UDC_SOF_EVENT()                  user_callback_sof_action()
// extern void user_callback_sof_action(void);
// #define  UDC_SUSPEND_EVENT()              user_callback_suspend_action()
// extern void user_callback_suspend_action(void);
// #define  UDC_RESUME_EVENT()               user_callback_resume_action()
// extern void user_callback_resume_action(void);
// #define  UDC_REMOTEWAKEUP_ENABLE()        user_callback_remotewakeup_enable()
// extern void user_callback_remotewakeup_enable(void);
// #define  UDC_REMOTEWAKEUP_DISABLE()       user_callback_remotewakeup_disable()
// extern void user_callback_remotewakeup_disable(void);
// #define  UDC_GET_EXTRA_STRING()

#define  USB_DEVICE_EP_CTRL_SIZE       64

#define  USB_DEVICE_NB_INTERFACE       1 // 1またはそれ以上

#define  USB_DEVICE_MAX_EP             1 // 0～ｲﾝﾀｰﾌｪｰｽによって要求された最大ｴﾝﾄﾞﾎﾟｲﾝﾄ

#define  UDI_CDC_PORT_NB 1

#define  UDI_CDC_ENABLE_EXT(port)             true
#define  UDI_CDC_DISABLE_EXT(port)
#define  UDI_CDC_RX_NOTIFY(port)
#define  UDI_CDC_TX_EMPTY_NOTIFY(port)
#define  UDI_CDC_SET_CODING_EXT(port,cfg)
#define  UDI_CDC_SET_DTR_EXT(port,set)
#define  UDI_CDC_SET_RTS_EXT(port,set)
/*
 * #define UDI_CDC_ENABLE_EXT(port) my_callback_cdc_enable()
 * extern bool my_callback_cdc_enable(void);
 * #define UDI_CDC_DISABLE_EXT(port) my_callback_cdc_disable()
 * extern void my_callback_cdc_disable(void);
 * #define  UDI_CDC_RX_NOTIFY(port) my_callback_rx_notify(port)
 * extern void my_callback_rx_notify(uint8_t port);
 * #define  UDI_CDC_TX_EMPTY_NOTIFY(port) my_callback_tx_empty_notify(port)
 * extern void my_callback_tx_empty_notify(uint8_t port);
 * #define  UDI_CDC_SET_CODING_EXT(port,cfg) my_callback_config(port,cfg)
 * extern void my_callback_config(uint8_t port, usb_cdc_line_coding_t * cfg);
 * #define  UDI_CDC_SET_DTR_EXT(port,set) my_callback_cdc_set_dtr(port,set)
```

```
 * extern void my_callback_cdc_set_dtr(uint8_t port, bool b_enable);
 * #define  UDI_CDC_SET_RTS_EXT(port,set) my_callback_cdc_set_rts(port,set)
 * extern void my_callback_cdc_set_rts(uint8_t port, bool b_enable);
 */

#define  UDI_CDC_LOW_RATE

#define  UDI_CDC_DEFAULT_RATE             115200
#define  UDI_CDC_DEFAULT_STOPBITS         CDC_STOP_BITS_1
#define  UDI_CDC_DEFAULT_PARITY           CDC_PAR_NONE
#define  UDI_CDC_DEFAULT_DATABITS         8

#define  UDI_CDC_DATA_EP_IN_0          (1 | USB_EP_DIR_IN)  // 送信
#define  UDI_CDC_DATA_EP_OUT_0         (2 | USB_EP_DIR_OUT) // 受信
#define  UDI_CDC_COMM_EP_0             (3 | USB_EP_DIR_IN)  // 通知ｴﾝﾄﾞﾎﾟｲﾝﾄ
#define  UDI_CDC_DATA_EP_IN_2          (4 | USB_EP_DIR_IN)  // 送信
#define  UDI_CDC_DATA_EP_OUT_2         (5 | USB_EP_DIR_OUT) // 受信
#define  UDI_CDC_COMM_EP_2             (6 | USB_EP_DIR_IN)  // 通知ｴﾝﾄﾞﾎﾟｲﾝﾄ
#define  UDI_CDC_DATA_EP_IN_3          (7 | USB_EP_DIR_IN)  // 送信
#define  UDI_CDC_DATA_EP_OUT_3         (8 | USB_EP_DIR_OUT) // 受信
#define  UDI_CDC_COMM_EP_3             (9 | USB_EP_DIR_IN)  // 通知ｴﾝﾄﾞﾎﾟｲﾝﾄ

#define  UDI_CDC_COMM_IFACE_NUMBER_0   0
#define  UDI_CDC_DATA_IFACE_NUMBER_0   1
#define  UDI_CDC_COMM_IFACE_NUMBER_2   2
#define  UDI_CDC_DATA_IFACE_NUMBER_2   3
#define  UDI_CDC_COMM_IFACE_NUMBER_3   4
#define  UDI_CDC_DATA_IFACE_NUMBER_3   5

#define UDI_MSC_GLOBAL_VENDOR_ID            ¥
   'A', 'T', 'M', 'E', 'L', ' ', ' ', ' '
#define UDI_MSC_GLOBAL_PRODUCT_VERSION           ¥
   '1', '.', '0', '0'

#define  UDI_MSC_ENABLE_EXT()          true
#define  UDI_MSC_DISABLE_EXT()
#define  UDI_MSC_NOTIFY_TRANS_EXT()
/*
 * #define UDI_MSC_ENABLE_EXT() my_callback_msc_enable()
 * extern bool my_callback_msc_enable(void);
 * #define UDI_MSC_DISABLE_EXT() my_callback_msc_disable()
 * extern void my_callback_msc_disable(void);
 * #define  UDI_MSC_NOTIFY_TRANS_EXT()    msc_notify_trans()
 * extern void msc_notify_trans(void) {
 */

#define  UDI_MSC_EP_IN                 (1 | USB_EP_DIR_IN)
#define  UDI_MSC_EP_OUT                (2 | USB_EP_DIR_OUT)

#define  UDI_MSC_IFACE_NUMBER          0

#define  UDI_HID_MOUSE_ENABLE_EXT()        true
#define  UDI_HID_MOUSE_DISABLE_EXT()
// #define UDI_HID_MOUSE_ENABLE_EXT() my_callback_mouse_enable()
// extern bool my_callback_mouse_enable(void);
// #define UDI_HID_MOUSE_DISABLE_EXT() my_callback_mouse_disable()
// extern void my_callback_mouse_disable(void);

#define  UDI_HID_MOUSE_EP_IN           (1 | USB_EP_DIR_IN)

#define  UDI_HID_MOUSE_IFACE_NUMBER    0
```

```
#define  UDI_HID_KBD_ENABLE_EXT()        true
#define  UDI_HID_KBD_DISABLE_EXT()
// #define UDI_HID_KBD_ENABLE_EXT() my_callback_keyboard_enable()
// extern bool my_callback_keyboard_enable(void);
// #define UDI_HID_KBD_DISABLE_EXT() my_callback_keyboard_disable()
// extern void my_callback_keyboard_disable(void);
#define  UDI_HID_KBD_CHANGE_LED(value)
// #define  UDI_HID_KBD_CHANGE_LED(value) my_callback_keyboard_led(value)
// extern void my_callback_keyboard_led(uint8_t value)
#define  UDI_HID_KBD_EP_IN           (1 | USB_EP_DIR_IN)

#define  UDI_HID_KBD_IFACE_NUMBER    0

#define  UDI_HID_GENERIC_ENABLE_EXT()        true
#define  UDI_HID_GENERIC_DISABLE_EXT()
#define  UDI_HID_GENERIC_REPORT_OUT(ptr)
#define  UDI_HID_GENERIC_SET_FEATURE(f)
/*
 * #define UDI_HID_GENERIC_ENABLE_EXT() my_callback_generic_enable()
 * extern bool my_callback_generic_enable(void);
 * #define UDI_HID_GENERIC_DISABLE_EXT() my_callback_generic_disable()
 * extern void my_callback_generic_disable(void);
 * #define  UDI_HID_GENERIC_REPORT_OUT(ptr) my_callback_generic_report_out(ptr)
 * extern void my_callback_generic_report_out(uint8_t *report);
 * #define  UDI_HID_GENERIC_SET_FEATURE(f) my_callback_generic_set_feature(f)
 * extern void my_callback_generic_set_feature(uint8_t *report_feature);
 */
#define  UDI_HID_REPORT_IN_SIZE             64
#define  UDI_HID_REPORT_OUT_SIZE            64
#define  UDI_HID_REPORT_FEATURE_SIZE        4
#define  UDI_HID_GENERIC_EP_SIZE            64

#define  UDI_HID_GENERIC_EP_OUT    (2 | USB_EP_DIR_OUT)
#define  UDI_HID_GENERIC_EP_IN     (1 | USB_EP_DIR_IN)

#define  UDI_HID_GENERIC_IFACE_NUMBER    0

#define  UDI_PHDC_ENABLE_EXT()           true
#define  UDI_PHDC_DISABLE_EXT()

#define  UDI_PHDC_DATAMSG_FORMAT       USB_PHDC_DATAMSG_FORMAT_11073_20601
#define  UDI_PHDC_SPECIALIZATION       {0x2345} // 11073_20601での定義

#define  UDI_PHDC_QOS_OUT              ¥
        (USB_PHDC_QOS_MEDIUM_BETTER|USB_PHDC_QOS_HIGH_BEST)
#define  UDI_PHDC_QOS_IN               ¥
        (USB_PHDC_QOS_LOW_GOOD|USB_PHDC_QOS_MEDIUM_BETTER|USB_PHDC_QOS_MEDIUM_BEST)

#define  UDI_PHDC_METADATA_DESC_BULK_IN    {0x01,0x02,0x03}
#define  UDI_PHDC_METADATA_DESC_BULK_OUT   {0x01,0x02,0x03}
#define  UDI_PHDC_METADATA_DESC_INT_IN     {0x01,0x02,0x03}

#define  UDI_PHDC_EP_BULK_OUT          (1 | USB_EP_DIR_OUT)
#define  UDI_PHDC_EP_BULK_IN           (2 | USB_EP_DIR_IN)
#if ((UDI_PHDC_QOS_IN&USB_PHDC_QOS_LOW_GOOD)==USB_PHDC_QOS_LOW_GOOD)
// UDI_PHDC_QOS_INの場合にだけUSB_PHDC_QOS_LOW_GOODをｲﾝｸﾙｰﾄﾞ
#  define  UDI_PHDC_EP_INTERRUPT_IN      (3 | USB_EP_DIR_IN)
#endif

#define  UDI_PHDC_EP_SIZE_BULK_OUT     32
```

```
#define  UDI_PHDC_EP_SIZE_BULK_IN      32
#define  UDI_PHDC_EP_SIZE_INT_IN       8

#define  UDI_PHDC_IFACE_NUMBER          0

#define  UDI_VENDOR_ENABLE_EXT()          true
#define  UDI_VENDOR_DISABLE_EXT()
#define  UDI_VENDOR_SETUP_OUT_RECEIVED()  false
#define  UDI_VENDOR_SETUP_IN_RECEIVED()   false
/*
 * #define UDI_VENDOR_ENABLE_EXT() my_callback_vendor_enable()
 * extern bool my_callback_vendor_enable(void);
 * #define UDI_VENDOR_DISABLE_EXT() my_callback_vendor_disable()
 * extern void my_callback_vendor_disable(void);
 *
 * #define  UDI_VENDOR_SETUP_OUT_RECEIVED()  my_vendor_setup_out_received()
 * extern bool my_vendor_setup_out_received(void);
 * #define  UDI_VENDOR_SETUP_IN_RECEIVED()   my_vendor_setup_in_received()
 * extern bool my_vendor_setup_in_received(void);
 */

#define  UDI_VENDOR_EPS_SIZE_INT_FS    64
#define  UDI_VENDOR_EPS_SIZE_BULK_FS   64
#define  UDI_VENDOR_EPS_SIZE_ISO_FS   256

#define  UDI_VENDOR_EPS_SIZE_INT_HS    64
#define  UDI_VENDOR_EPS_SIZE_BULK_HS  512
#define  UDI_VENDOR_EPS_SIZE_ISO_HS    64

#define  UDI_VENDOR_EP_INTERRUPT_IN  (1 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_INTERRUPT_OUT (2 | USB_EP_DIR_OUT)
#define  UDI_VENDOR_EP_BULK_IN       (3 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_BULK_OUT      (4 | USB_EP_DIR_OUT)
#define  UDI_VENDOR_EP_ISO_IN        (5 | USB_EP_DIR_IN)
#define  UDI_VENDOR_EP_ISO_OUT       (6 | USB_EP_DIR_OUT)

#define  UDI_VENDOR_IFACE_NUMBER     0

//... 最終的に他のｲﾝﾀｰﾌｪｰｽ構成設定を追加してください。

#define UDI_COMPOSITE_DESC_T

#define UDI_COMPOSITE_DESC_FS

#define UDI_COMPOSITE_DESC_HS

#define UDI_COMPOSITE_API

/* CDC、MSC、HIDﾏｳｽ ｲﾝﾀｰﾌｪｰｽを持つ装置の例
#define UDI_COMPOSITE_DESC_T ¥
    usb_iad_desc_t udi_cdc_iad; ¥
    udi_cdc_comm_desc_t udi_cdc_comm; ¥
    udi_cdc_data_desc_t udi_cdc_data; ¥
    udi_msc_desc_t udi_msc; ¥
    udi_hid_mouse_desc_t udi_hid_mouse

#define UDI_COMPOSITE_DESC_FS ¥
    .udi_cdc_iad               = UDI_CDC_IAD_DESC_0, ¥
    .udi_cdc_comm              = UDI_CDC_COMM_DESC_0, ¥
    .udi_cdc_data              = UDI_CDC_DATA_DESC_0_FS, ¥
    .udi_msc                   = UDI_MSC_DESC_FS, ¥
```

```
    .udi_hid_mouse              = UDI_HID_MOUSE_DESC

#define UDI_COMPOSITE_DESC_HS ¥
    .udi_cdc_iad                = UDI_CDC_IAD_DESC_0, ¥
    .udi_cdc_comm               = UDI_CDC_COMM_DESC_0, ¥
    .udi_cdc_data               = UDI_CDC_DATA_DESC_0_HS, ¥
    .udi_msc                    = UDI_MSC_DESC_HS, ¥
    .udi_hid_mouse              = UDI_HID_MOUSE_DESC

#define UDI_COMPOSITE_API  ¥
    &udi_api_cdc_comm,        ¥
    &udi_api_cdc_data,        ¥
    &udi_api_msc,             ¥
    &udi_api_hid_mouse
*/

/* インターフェース用インクルードの例
#include "udi_msc.h"
#include "udi_hid_kbd.h"
#include "udi_hid_mouse.h"
#include "udi_cdc.h"
#include "udi_phdc.h"
#include "udi_vendor.h"
*/
/* USBによって使われる呼び戻しの宣言
#include "callback_def.h"
*/

#endif // _CONF_USB_H_
```

7.3.2. conf_clock.h

7.3.2.1. SAM3X, SAM3Aﾃﾞﾊﾞｲｽ (UOTGHS:USB OTG 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// システム クロック(MCK)元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_BYPASS
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_4M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_8M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_12M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_BYPASS
#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLLACK
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_UPLLCK

// ===== システム クロック(MCK)前置分周器任意選択    (Fmck = Fsys / (SYSCLK_PRES))
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_1
#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_2
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_4
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_8
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_16
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_32
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_64
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_3
```

```
// ===== PLL0(A)任意選択    (Fpll = (Fclk * PLL_mul) / PLL_div)
// ここにmulとdivの実効値を使ってください。
#define CONFIG_PLL0_SOURCE          PLL_SRC_MAINCK_XTAL
#define CONFIG_PLL0_MUL             14
#define CONFIG_PLL0_DIV             1

// ===== 480MHzで固定化されたUPLL(UTMI)ﾊｰﾄﾞｳｪｱ

// ===== USBｸﾛｯｸ元任意選択    (Fusb = FpllX / USB_div)
// ここにdivの実効値を使ってください。
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL0
#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_UPLL
#define CONFIG_USBCLK_DIV           1

// ===== 目的対象周波数(ｼｽﾃﾑ ｸﾛｯｸ)
// - XTAL周波数: 12MHz
// - ｼｽﾃﾑ ｸﾛｯｸ元: PLLA
// - ｼｽﾃﾑ ｸﾛｯｸ前置分周器: 2(2分周)
// - PLLA供給元: XTAL
// - PLLA出力: XTAL * 14 / 1
// - ｼｽﾃﾑ ｸﾛｯｸは: 12 * 14 / 1 /2 = 84MHz
// ===== 目的対象周波数(USBｸﾛｯｸ)
// - USBｸﾛｯｸ元: UPLL
// - USBｸﾛｯｸ分周器: 1(分周なし)
// - UPLL周波数: 480MHz
// - USBｸﾛｯｸ: 480 / 1 = 480MHz

#endif /* CONF_CLOCK_H_INCLUDED */
```

7.3.2.2. SAM4Lﾃﾞﾊﾞｲｽ(USBC)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

//#define CONFIG_SYSCLK_INIT_CPUMASK   (1 << SYSCLK_OCD)
//#define CONFIG_SYSCLK_INIT_PBAMASK   (1 << SYSCLK_IISC)
//#define CONFIG_SYSCLK_INIT_PBBMASK   (1 << SYSCLK_USBC_REGS)
//#define CONFIG_SYSCLK_INIT_PBCMASK   (1 << SYSCLK_CHIPID)
//#define CONFIG_SYSCLK_INIT_PBDMASK   (1 << SYSCLK_AST)
//#define CONFIG_SYSCLK_INIT_HSBMASK   (1 << SYSCLK_PDCA_HSB)

//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_RCSYS
//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_OSC0
#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_PLL0
//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_DFLL
//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_RC80M
//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_RCFAST
//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_RC1M

/* RCFAST周波数選択: 4MHzに対して0、8MHzに対して1、12MHzに対して2 */
//#define CONFIG_RCFAST_FRANGE     0
//#define CONFIG_RCFAST_FRANGE     1
//#define CONFIG_RCFAST_FRANGE     2

/* Fbus = Fsys / (2 ^ BUS_div) */
#define CONFIG_SYSCLK_CPU_DIV          0
#define CONFIG_SYSCLK_PBA_DIV          0
#define CONFIG_SYSCLK_PBB_DIV          0
```

```
#define CONFIG_SYSCLK_PBC_DIV          0
#define CONFIG_SYSCLK_PBD_DIV          0

// ===== 全ての必須でない周辺機能のｸﾛｯｸを禁止
//#define CONFIG_SYSCLK_INIT_CPUMASK   0
//#define CONFIG_SYSCLK_INIT_PBAMASK   SYSCLK_USART1
//#define CONFIG_SYSCLK_INIT_PBBMASK   0
//#define CONFIG_SYSCLK_INIT_PBCMASK   0
//#define CONFIG_SYSCLK_INIT_PBDMASK   0
//#define CONFIG_SYSCLK_INIT_HSBMASK   0

// ===== PLL任意選択
#define CONFIG_PLL0_SOURCE           PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE           PLL_SRC_GCLK9

/* Fpll0 = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL0_MUL              (48000000UL / BOARD_OSC0_HZ)
//#define CONFIG_PLL0_DIV              1
#define CONFIG_PLL0_MUL                (192000000 / FOSC0) /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL0_DIV                4 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ==== DFLL任意選択
//#define CONFIG_DFLL0_SOURCE          GENCLK_SRC_OSC0
//#define CONFIG_DFLL0_SOURCE          GENCLK_SRC_RCSYS
//#define CONFIG_DFLL0_SOURCE          GENCLK_SRC_OSC32K
//#define CONFIG_DFLL0_SOURCE          GENCLK_SRC_RC120M
//#define CONFIG_DFLL0_SOURCE          GENCLK_SRC_RC32K

/* Fdfll = (Fclk * DFLL_mul) / DFLL_div */
//#define CONFIG_DFLL0_FREQ            48000000UL
//#define CONFIG_DFLL0_MUL             ((4 * CONFIG_DFLL0_FREQ) / BOARD_OSC32_HZ)
//#define CONFIG_DFLL0_DIV             4
//#define CONFIG_DFLL0_MUL             (CONFIG_DFLL0_FREQ / BOARD_OSC32_HZ)
//#define CONFIG_DFLL0_DIV             1

// ===== USBｸﾛｯｸ元任意選択
//#define CONFIG_USBCLK_SOURCE         USBCLK_SRC_OSC0
#define CONFIG_USBCLK_SOURCE         USBCLK_SRC_PLL0
//#define CONFIG_USBCLK_SOURCE           USBCLK_SRC_DFLL

/* Fusb = Fsys / USB_div */
#define CONFIG_USBCLK_DIV            1

// ===== GCLK9任意選択
//#define CONFIG_GCLK9_SOURCE            GENCLK_SRC_GCLKIN0
//#define CONFIG_GCLK9_DIV               1

#endif /* CONF_CLOCK_H_INCLUDED */
```

7.3.3. conf_clocks.h

7.3.3.1. SAM D21ﾃﾞﾊﾞｲｽ (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#include <clock.h>

#ifndef CONF_CLOCKS_H_INCLUDED
#  define CONF_CLOCKS_H_INCLUDED

/* ｼｽﾃﾑ ｸﾛｯｸ ﾊﾞｽ構成設定 */
```

```
#  define CONF_CLOCK_CPU_CLOCK_FAILURE_DETECT      false
#  define CONF_CLOCK_FLASH_WAIT_STATES             2
#  define CONF_CLOCK_CPU_DIVIDER                   SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBA_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBB_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBC_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1

/* SYSTEM_CLOCK_SOURCE_OSC8M構成設定 - 内部8MHz発振器 */
#  define CONF_CLOCK_OSC8M_PRESCALER               SYSTEM_OSC8M_DIV_1
#  define CONF_CLOCK_OSC8M_ON_DEMAND               true
#  define CONF_CLOCK_OSC8M_RUN_IN_STANDBY          false

/* SYSTEM_CLOCK_SOURCE_XOSC構成設定 - 外部ｸﾛｯｸ/発振器 */
#  define CONF_CLOCK_XOSC_ENABLE                   false
#  define CONF_CLOCK_XOSC_EXTERNAL_CRYSTAL         SYSTEM_CLOCK_EXTERNAL_CRYSTAL
#  define CONF_CLOCK_XOSC_EXTERNAL_FREQUENCY       12000000UL
#  define CONF_CLOCK_XOSC_STARTUP_TIME             SYSTEM_XOSC_STARTUP_32768
#  define CONF_CLOCK_XOSC_AUTO_GAIN_CONTROL        true
#  define CONF_CLOCK_XOSC_ON_DEMAND                true
#  define CONF_CLOCK_XOSC_RUN_IN_STANDBY           false

/* SYSTEM_CLOCK_SOURCE_XOSC32K構成設定 - 外部32kHzｸﾘｽﾀﾙ/ｸﾛｯｸ発振器 */
#  define CONF_CLOCK_XOSC32K_ENABLE                false
#  define CONF_CLOCK_XOSC32K_EXTERNAL_CRYSTAL      SYSTEM_CLOCK_EXTERNAL_CRYSTAL
#  define CONF_CLOCK_XOSC32K_STARTUP_TIME          SYSTEM_XOSC32K_STARTUP_65536
#  define CONF_CLOCK_XOSC32K_AUTO_AMPLITUDE_CONTROL  false
#  define CONF_CLOCK_XOSC32K_ENABLE_1KHZ_OUPUT     false
#  define CONF_CLOCK_XOSC32K_ENABLE_32KHZ_OUTPUT   true
#  define CONF_CLOCK_XOSC32K_ON_DEMAND             true
#  define CONF_CLOCK_XOSC32K_RUN_IN_STANDBY        false

/* SYSTEM_CLOCK_SOURCE_OSC32K構成設定 - 内部32kHz発振器 */
#  define CONF_CLOCK_OSC32K_ENABLE                 false
#  define CONF_CLOCK_OSC32K_STARTUP_TIME           SYSTEM_OSC32K_STARTUP_130
#  define CONF_CLOCK_OSC32K_ENABLE_1KHZ_OUTPUT     true
#  define CONF_CLOCK_OSC32K_ENABLE_32KHZ_OUTPUT    true
#  define CONF_CLOCK_OSC32K_ON_DEMAND              true
#  define CONF_CLOCK_OSC32K_RUN_IN_STANDBY         false

/* SYSTEM_CLOCK_SOURCE_DFLL構成設定 - ﾃﾞｼﾞﾀﾙ周波数固定化閉路 */
#  define CONF_CLOCK_DFLL_ENABLE                   true
#  define CONF_CLOCK_DFLL_LOOP_MODE                SYSTEM_CLOCK_DFLL_LOOP_MODE_USB_RECOVERY
#  define CONF_CLOCK_DFLL_ON_DEMAND                true

/* DFLL開路動作構成設定 */
#  define CONF_CLOCK_DFLL_FINE_VALUE               (512)

/* DFLL閉路動作構成設定 */
#  define CONF_CLOCK_DFLL_SOURCE_GCLK_GENERATOR    GCLK_GENERATOR_1
#  define CONF_CLOCK_DFLL_MULTIPLY_FACTOR          (48000000 / 32768)
#  define CONF_CLOCK_DFLL_QUICK_LOCK               true
#  define CONF_CLOCK_DFLL_TRACK_AFTER_FINE_LOCK    true
#  define CONF_CLOCK_DFLL_KEEP_LOCK_ON_WAKEUP      true
#  define CONF_CLOCK_DFLL_ENABLE_CHILL_CYCLE       true
#  define CONF_CLOCK_DFLL_MAX_COARSE_STEP_SIZE     (0x1f / 4)
#  define CONF_CLOCK_DFLL_MAX_FINE_STEP_SIZE       (0xff / 4)

/* SYSTEM_CLOCK_SOURCE_DPLL構成設定 - ﾃﾞｼﾞﾀﾙ位相固定化閉路 */
#  define CONF_CLOCK_DPLL_ENABLE                   false
#  define CONF_CLOCK_DPLL_ON_DEMAND                true
#  define CONF_CLOCK_DPLL_RUN_IN_STANDBY           false
```

```
#  define CONF_CLOCK_DPLL_LOCK_BYPASS              false
#  define CONF_CLOCK_DPLL_WAKE_UP_FAST             false
#  define CONF_CLOCK_DPLL_LOW_POWER_ENABLE         false

#  define CONF_CLOCK_DPLL_LOCK_TIME                SYSTEM_CLOCK_SOURCE_DPLL_LOCK_TIME_DEFAULT
#  define CONF_CLOCK_DPLL_REFERENCE_CLOCK          SYSTEM_CLOCK_SOURCE_DPLL_REFERENCE_CLOCK_XOSC32K
#  define CONF_CLOCK_DPLL_FILTER                   SYSTEM_CLOCK_SOURCE_DPLL_FILTER_DEFAULT

#  define CONF_CLOCK_DPLL_REFERENCE_FREQUENCY      32768
#  define CONF_CLOCK_DPLL_REFERENCE_DIVIDER        1
#  define CONF_CLOCK_DPLL_OUTPUT_FREQUENCY         48000000

/* DPLL GCLK基準構成設定 */
# define CONF_CLOCK_DPLL_REFERENCE_GCLK_GENERATOR GCLK_GENERATOR_1
/* DPLL GCLK固定化計時器構成設定 */
# define CONF_CLOCK_DPLL_LOCK_GCLK_GENERATOR GCLK_GENERATOR_1

/* clocks_init走行時にGCLKを構成設定するためにこれをtrueに設定してください。
 * falseに設定した場合、GCLK発振器はclocks_init()で何も構成設定されません。*/
#  define CONF_CLOCK_CONFIGURE_GCLK                true

/* GCLK発振器0構成設定（主ｸﾛｯｸ） */
#  define CONF_CLOCK_GCLK_0_ENABLE                 true
#  define CONF_CLOCK_GCLK_0_RUN_IN_STANDBY         true
#  define CONF_CLOCK_GCLK_0_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_DFLL
#  define CONF_CLOCK_GCLK_0_PRESCALER              1
#  define CONF_CLOCK_GCLK_0_OUTPUT_ENABLE          false

/* GCLK発振器1構成設定 */
#  define CONF_CLOCK_GCLK_1_ENABLE                 false
#  define CONF_CLOCK_GCLK_1_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_1_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_XOSC32K
#  define CONF_CLOCK_GCLK_1_PRESCALER              1
#  define CONF_CLOCK_GCLK_1_OUTPUT_ENABLE          false

/* GCLK発振器2構成設定（RTC） */
#  define CONF_CLOCK_GCLK_2_ENABLE                 false
#  define CONF_CLOCK_GCLK_2_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_2_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC32K
#  define CONF_CLOCK_GCLK_2_PRESCALER              32
#  define CONF_CLOCK_GCLK_2_OUTPUT_ENABLE          false

/* GCLK発振器3構成設定 */
#  define CONF_CLOCK_GCLK_3_ENABLE                 false
#  define CONF_CLOCK_GCLK_3_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_3_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_3_PRESCALER              1
#  define CONF_CLOCK_GCLK_3_OUTPUT_ENABLE          false

/* GCLK発振器4構成設定 */
#  define CONF_CLOCK_GCLK_4_ENABLE                 false
#  define CONF_CLOCK_GCLK_4_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_4_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_4_PRESCALER              1
#  define CONF_CLOCK_GCLK_4_OUTPUT_ENABLE          false

/* GCLK発振器5構成設定 */
#  define CONF_CLOCK_GCLK_5_ENABLE                 false
#  define CONF_CLOCK_GCLK_5_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_5_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_5_PRESCALER              1
```

```
#  define CONF_CLOCK_GCLK_5_OUTPUT_ENABLE         false

/* GCLK発振器6構成設定 */
#  define CONF_CLOCK_GCLK_6_ENABLE                false
#  define CONF_CLOCK_GCLK_6_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_6_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_6_PRESCALER             1
#  define CONF_CLOCK_GCLK_6_OUTPUT_ENABLE         false

/* GCLK発振器7構成設定 */
#  define CONF_CLOCK_GCLK_7_ENABLE                false
#  define CONF_CLOCK_GCLK_7_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_7_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_7_PRESCALER             1
#  define CONF_CLOCK_GCLK_7_OUTPUT_ENABLE         false

/* GCLK発振器8構成設定 */
#  define CONF_CLOCK_GCLK_8_ENABLE                false
#  define CONF_CLOCK_GCLK_8_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_8_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_8_PRESCALER             1
#  define CONF_CLOCK_GCLK_8_OUTPUT_ENABLE         false

#endif /* CONF_CLOCKS_H_INCLUDED */
```

7.3.4. conf_board.h

7.3.4.1. SAM3X, SAM3Aﾃﾞﾊﾞｲｽ (UOTGHS:USB OTG 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

/* USBﾋﾟﾝを使用 */
#define CONF_BOARD_USB_PORT

#endif /* CONF_BOARD_H_INCLUDED */
```

7.3.4.2. SAM4Lﾃﾞﾊﾞｲｽ (USBC)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// board_init()が呼ばれる時にUSART用汎用入出力を自動初期化
//#define CONF_BOARD_COM_PORT

// USBｲﾝﾀｰﾌｪｰｽ(USB)を許可
#define CONF_BOARD_USB_PORT

#endif /* CONF_BOARD_H_INCLUDED */
```

7.3.4.3. SAM D21ﾃﾞﾊﾞｲｽ (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */
```

```
#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

/* USB VBUS検出許可 */
#define CONF_BOARD_USB_VBUS_DETECT

#endif /* CONF_BOARD_H_INCLUDED */
```

8. USBホスト制御部(UHC)

UHCはUSBホストの上位抽象物を提供します。主なホスト状態(開始/休止/再開)を制御するのにこれらの関数を使うことができます。
USB装置階層内の全てのUSBホスト インターフェース(UHI)はUSB列挙(接続認証)を支援するためのUHCに基づきます。

- AVR4950:ASF – USBホスト階層

この資料は以下のようにUHCに基づく共通的なUSBホストの使い方を記述します。

- API概要
- USBホスト基本構成設定
- USBホストの高度な使用事例

8.1. API概要

8.1.1. 構造体定義

8.1.1.1. uhc_device_t構造体

注: この構造体の領域は使用者応用によって変えられるべきではなく、それらは単位部の内部使用に対してだけ予約されます。

表8-1. ﾒﾝﾊﾞ

型	名前	説明
uint8_t	address	USBｱﾄﾞﾚｽ
usb_conf_desc_t *	conf_desc	現在のUSB構成設定記述子
usb_dev_desc_t	dev_desc	USB装置記述子
uhd_speed_t	speed	USB速度

8.1.2. 関数定義

8.1.2.1. USBホスト階層を制御するための関数

uhc_start()関数

ホスト動作を開始

```
void uhc_start( void )
```

uhc_stop()関数

ホスト動作を停止

```
void uhc_stop( bool b_id_stop )
```

表8-2. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
b_id_stop	[入力]	真ならば、USB IDﾋﾟﾝの管理を停止

uhc_suspend()関数

USB線を休止(ｻｽﾍﾟﾝﾄﾞ)

```
void uhc_suspend( bool b_remotewakeup )
```

表8-3. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
b_remotewakeup	[入力]	真ならば、遠隔起動機能を認可

uhc_is_suspend()関数

USB線で休止(ｻｽﾍﾟﾝﾄﾞ)状態が許可されているか調査

```
bool uhc_is_suspend( void )
```

戻り値：真(true)ならば、USB線が休止(SUSPEND)状態か、または装置が接続されていません。

uhc_resume()関数

USB線を再開

```
void uhc_resume(void)
```

uhc_suspend_lpm()関数

LPM機能(SAM D21)を通してUSB線を休止(ｻｽﾍﾟﾝﾄﾞ)

```
bool uhc_suspend_lpm( bool b_remotewakeup, uint8_t besl )
```

表8-4. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
b_remotewakeup	[入力]	真ならば、遠隔起動機能を認可
besl	[入力]	ﾍﾞｽﾄｴﾌｫｰﾄ ｻｰﾋﾞｽ遅延値

戻り値：USB装置によってLPMが支援されなければ偽(false)

8.1.2.2. 装置を管理するための使用者関数

uhc_get_device_number()関数

接続されている装置数を返します。

```
uint8_t uhc_get_device_number( void )
```

戻り値：USB樹状網に接続されている装置数

uhc_dev_get_string_manufacturer()関数

USB装置からUSB製造業者文字列を取得

```
char * uhc_dev_get_string_manufacturer( uhc_device_t * dev )
```

この関数は構成設定要求の最後を待ち、そのﾀｲﾐﾝｸﾞは(3ms～15秒)の長さで有り得ます。故に、割り込みﾙｰﾁﾝで呼んではなりません。この関数は使用者応用によって解放されなければならない緩衝部を割り当てます。

表8-5. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
dev	[入力]	要求する装置

戻り値：ﾕﾆｺｰﾄﾞ文字列でのﾎﾟｲﾝﾀ、または関数が失敗の場合にNULL

uhc_dev_get_string_product()関数

USB装置からUSB製品文字列を取得

```
char * uhc_dev_get_string_product( uhc_device_t * dev )
```

この関数は構成設定要求の最後を待ち、そのﾀｲﾐﾝｸﾞは(3ms～15秒)の長さで有り得ます。故に、割り込みﾙｰﾁﾝで呼んではなりません。この関数は使用者応用によって解放されなければならない緩衝部を割り当てます。

表8-6. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
dev	[入力]	要求する装置

戻り値：ﾕﾆｺｰﾄﾞ文字列でのﾎﾟｲﾝﾀ、または関数が失敗の場合にNULL

uhc_dev_get_string_serial()関数

USB装置からUSB通番文字列を取得

```
char * uhc_dev_get_string_serial( uhc_device_t * dev )
```

この関数は構成設定要求の最後を待ち、そのﾀｲﾐﾝｸﾞは(3ms～15秒)の長さで有り得ます。故に、割り込みﾙｰﾁﾝで呼んではなりません。この関数は使用者応用によって解放されなければならない緩衝部を割り当てます。

表8-7. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
dev	[入力]	要求する装置

戻り値：ﾕﾆｺｰﾄﾞ文字列でのﾎﾟｲﾝﾀ、または関数が失敗の場合にNULL

uhc_dev_get_string()関数

USB装置からUSB文字列を取得

```
char * uhc_dev_get_string( uhc_device_t * dev, uint8_t str_id )
```

この関数は構成設定要求の最後を待ち、そのﾀｲﾐﾝｸﾞは(3ms～15秒)の長さで有り得ます。故に、割り込みﾙｰﾁﾝで呼んではなりません。この関数は使用者応用によって解放されなければならない緩衝部を割り当てます。

表8-8. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
dev	[入力]	要求する装置
str_id	[入力]	要求する文字列ID

戻り値：ﾕﾆｺｰﾄﾞ文字列でのﾎﾟｲﾝﾀ、または関数が失敗の場合にNULL

uhc_dev_get_power()関数

装置の最大消費電流(mA)を取得

```
uint16_t uhc_dev_get_power( uhc_device_t * dev )
```

表8-9. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
dev	[入力]	要求する装置

戻り値：装置の最大消費電流(mA)

uhc_dev_get_speed()関数

現在の装置速度を返します。

```
uhd_speed_t uhc_dev_get_speed( uhc_device_t * dev )
```

表8-10. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
dev	[入力]	要求する装置

戻り値：装置速度

uhc_dev_is_high_speed_support()関数

装置が高速(HS)を支援するか調査。この関数は構成設定要求の最後を待ち、そのﾀｲﾐﾝｸﾞは(1ms～5秒)の長さで有り得ます。故に、割り込みルーチンで呼んではなりません。

```
bool uhc_dev_is_high_speed_support( uhc_device_t * dev )
```

表8-11. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
dev	[入力]	要求する装置

戻り値：高速(HS)が支援される場合に真(true)

8.1.3. 列挙定義

8.1.3.1. uhc_enum_status_t列挙

表8-12. ﾒﾝﾊﾞ

列挙値	説明
UHC_ENUM_SUCCESS	装置は列挙(接続/認証)されました。支援するUSB装置ｲﾝﾀｰﾌｪｰｽが許可されました。
UHC_ENUM_UNSUPPORTED	UHIによって支援されるｲﾝﾀｰﾌｪｰｽが全くなりません。
UHC_ENUM_OVERCURRENT	装置電力が支援されません。
UHC_ENUM_FAIL	USB列挙(接続/認証)中に問題発生
UHC_ENUM_HARDWARE_LIMIT	USBﾊｰﾄﾞｳｪｱはそれを支援できません。空きﾊﾟｲﾌﾟ不充分
UHC_ENUM_SOFTWARE_LIMIT	USBｿﾌﾄｳｪｱはそれを支援できません。実行制限
UHC_ENUM_MEMORY_LIMIT	USBｿﾌﾄｳｪｱはそれを支援できません。ﾒﾓﾘ不足
UHC_ENUM_DISCONNECT	USB列挙(接続/認証)中に装置が切断されました。

8.2. USBホスト基本構成設定

8.2.1. USBホスト使用者構成設定

以下のUSBホスト構成設定は応用のconf_usb_host.hファイルでインクルードされなければなりません。

1. USB_HOST_UHI (UHI APIの一覧)
   USBホストによって支援されるUHIの一覧を定義 (例えば:UHI_MSC, UHI_HID_MOUSE)
2. USB_HOST_POWER_MAX (mA)
   VBUSで許される最大電流
3. USB_HOST_HS_SUPPORT (定義のみ)
   高速(HS)での走行をUSBホストに認可
4. USB_HOST_HUB_SUPPORT (定義のみ)
   USBハブ支援を認可

8.2.2. USBホスト使用者呼び戻し

以下のに任意選択USBホスト呼び戻しは応用のconf_usb_host.hファイルで定義されなければなりません。

1. void UHC_MODE_CHANGE(bool b_host_mode)
   USB動作形態が自動的に切り替わったことを通知。これはIDピンが利用可能な時にだけ可能です。
2. void UHC_VBUS_CHANGE(bool b_present)
   VBUSレベルが変化したことを通知 (VBUS監視を持つUSBハードウェアでだけ利用可能)
3. void UHC_VBUS_ERROR(void)
   VBUS異常が発生したことを通知 (VBUS監視を持つUSBハードウェアでだけ利用可能)
4. void UHC_CONNECTION_EVENT(uhc_device_t* dev, bool b_present)
   装置が接続または切断されたことを通知
5. void UHC_WAKEUP_EVENT(void)
   USB装置またはホストがUSB線で起動される時に呼ばれます。
6. void UHC_SOF_EVENT(void)
   1ms毎でSOFを受信する毎に呼ばれます。高速(HS)と全速(FS)の動作形態で利用可能
7. uint8_t UHC_DEVICE_CONF(uhc_device_t* dev)
   USB装置構成設定が選ばれなければならない時に呼ばれます。故に、応用はこの装置に対する構成設定番号か、またはそれを拒否するために構成設定番号0のどちらかを選ぶことができます。呼び戻しが定義されていなければ、構成設定1が選ばれます。
8. void UHC_ENUM_EVENT(uhc_device_t* dev, uint8_t b_status)
   USB装置列挙(接続認証)が完了されるか、または失敗した時に呼ばれます。

8.2.3. USBホスト構成設定段階

8.2.3.1. USBホスト制御部 (UHC) – 事前必要条件

全てのUSBホストに対する共通的な事前必要条件

この単位部は完全な割り込み駆動のUSBホスト階層に基づき、sleepmgrを支援します。AVR®とAtmel®のSMART ARM®に基づくSAM3/4デバイスについてはクロック サービスが支援されます。SAM D21デバイスについてはクロック駆動部が支援されます。

プロジェクトを正しく構成設定するために以下の手続きが実行されなければなりません。

- クロック構成設定を指定してください。
  - USB高速(HS)支援のないUC3とSAM3/4デバイスは48MHzクロック入力が必要です。
    PLLと外部発振器を使わなければなりません。
  - USB高速(HS)支援付きのUC3とSAM3/4デバイスは12MHzクロック入力が必要です。
    外部発振器を使わなければなりません。
  - USBCハードウェア付きのUC3デバイスは25MHzよりも高いCPU周波数が必要です。
  - USB高速(HS)支援のないSAM D21デバイスは48MHzクロック入力が必要です。
    DFLLと外部発振器を使わなければなりません。
- conf_board.hに於いて、USB線を許可するためにCONF_BOARD_USB_PORT定義が追加されなければなりません。(全ての基板に対して必須ではありません。)
- 割り込みを許可してください。
- クロック サービスを初期化してください。

sleepmgr(休止管理)ｻｰﾋﾞｽの使用は任意選択ですが、消費電力を減らすために推奨されます。

- 休止管理ｻｰﾋﾞｽを初期化してください。
- 応用がｱｲﾄﾞﾙ状態の時に休止動作形態を活性にしてください。

conf_clock.h例

AVRとSAM3/4ﾃﾞﾊﾞｲｽについては以下の初期化ｺｰﾄﾞを追加してください。

```
sysclk_init();
irq_initialize_vectors();
cpu_irq_enable();
board_init();
sleepmgr_init(); // 任意選択
```

SAM D21ﾃﾞﾊﾞｲｽについては初期化ｺｰﾄﾞに以下を追加してください。

```
system_init();
irq_initialize_vectors();
cpu_irq_enable();
sleepmgr_init(); // 任意選択
```

主ｱｲﾄﾞﾙ繰り返しに以下を追加してください。

```
sleepmgr_enter_sleep(); // 任意選択
```

8.2.3.2. USBﾎｽﾄ制御部 (UHC) – ｺｰﾄﾞ例

全てのUSBﾎｽﾄに対する共通的なｺｰﾄﾞ例

conf_usb_host.hの内容

```
#define USB_HOST_POWER_MAX   500
```

応用Cﾌｧｲﾙに以下を追加してください。

```
void usb_init(void)
{
  uhc_start();
}
```

8.2.3.3. USBﾎｽﾄ制御部 (UHC) – 作業の流れ

全てのUSBﾎｽﾄに対する共通的な作業の流れ

1. conf_usb_host.hが利用可能で主USB装置構成設定である以下の構成設定を含むことを確実にしてください。

```
// 5V生成部に依存してVBUSで許される最大電流(mA)
#define USB_HOST_POWER_MAX   500 // (500mA)
```

2. USBﾎｽﾄ階層を許可するためにUSBﾎｽﾄ階層開始関数を呼んでください。

```
uhc_start();
```

8.2.4. conf_clock.h例

AT32UC3A0,AT32UC3A1,AT32UC3Bﾃﾞﾊﾞｲｽ(USBB)用のconf_colck.hの内容

```
// 12MHz外部発振器に基づく構成設定
#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC0
#define CONFIG_PLL1_MUL             8
#define CONFIG_PLL1_DIV             2
#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV           1 // Fusb = Fsys/(2 ^ USB_div)
```

AT32UC3A3とAT32UC3A4ﾃﾞﾊﾞｲｽ(高速(HS)支援付きUSBB)用のconf_colck.hの内容

```
// 12MHz外部発振器に基づく構成設定
#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_OSC0
#define CONFIG_USBCLK_DIV           1 // Fusb = Fsys/(2 ^ USB_div)
```

AT32UC3Cﾃﾞﾊﾞｲｽ(USBC)用のconf_colck.hの内容

```
// 12MHz外部発振器に基づく構成設定
#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC0
#define CONFIG_PLL1_MUL             8
#define CONFIG_PLL1_DIV             2
#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL1
```

```
#define CONFIG_USBCLK_DIV           1 // Fusb = Fsys/(2 ^ USB_div)
// USBCで動くために25MHz以上のCPUｸﾛｯｸが必要
#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLL1
```

SAM3XとSAM3Aﾃﾞﾊﾞｲｽ(UOTGHS:USB OTG 高速(HS))用のconf_colck.hの内容

```
// UPLLで固定化されるUSBｸﾛｯｸ元
#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_UPLL
#define CONFIG_USBCLK_DIV           1
```

SAM D21ﾃﾞﾊﾞｲｽ(USB)用のconf_colck.hの内容

```
// DFLLで固定化されるUSBｸﾛｯｸ元
// SYSTEM_CLOCK_SOURCE_XOSC32K構成設定 - 外部32kHzｸﾘｽﾀﾙ/ｸﾛｯｸ発振器
#  define CONF_CLOCK_XOSC32K_ENABLE                true
#  define CONF_CLOCK_XOSC32K_EXTERNAL_CRYSTAL      SYSTEM_CLOCK_EXTERNAL_CRYSTAL
#  define CONF_CLOCK_XOSC32K_STARTUP_TIME          SYSTEM_XOSC32K_STARTUP_65536
#  define CONF_CLOCK_XOSC32K_AUTO_AMPLITUDE_CONTROL  false
#  define CONF_CLOCK_XOSC32K_ENABLE_1KHZ_OUPUT     false
#  define CONF_CLOCK_XOSC32K_ENABLE_32KHZ_OUTPUT   true
#  define CONF_CLOCK_XOSC32K_ON_DEMAND             false
#  define CONF_CLOCK_XOSC32K_RUN_IN_STANDBY        true
// SYSTEM_CLOCK_SOURCE_DFLL構成設定 - ﾃﾞｼﾞﾀﾙ周波数固定化閉路
#  define CONF_CLOCK_DFLL_ENABLE                   true
#  define CONF_CLOCK_DFLL_LOOP_MODE                SYSTEM_CLOCK_DFLL_LOOP_MODE_CLOSED
#  define CONF_CLOCK_DFLL_ON_DEMAND                true

// DFLL閉路動作構成設定
#  define CONF_CLOCK_DFLL_SOURCE_GCLK_GENERATOR    GCLK_GENERATOR_1
#  define CONF_CLOCK_DFLL_MULTIPLY_FACTOR          (48000000/32768)
#  define CONF_CLOCK_DFLL_QUICK_LOCK               true
#  define CONF_CLOCK_DFLL_TRACK_AFTER_FINE_LOCK    true
#  define CONF_CLOCK_DFLL_KEEP_LOCK_ON_WAKEUP      true
#  define CONF_CLOCK_DFLL_ENABLE_CHILL_CYCLE       true
#  define CONF_CLOCK_DFLL_MAX_COARSE_STEP_SIZE     (0x1f / 8)
#  define CONF_CLOCK_DFLL_MAX_FINE_STEP_SIZE       (0xff / 8)

#  define CONF_CLOCK_CONFIGURE_GCLK                true

// GCLK発振器0(主ｸﾛｯｸ)構成設定
#  define CONF_CLOCK_GCLK_0_ENABLE                 true
#  define CONF_CLOCK_GCLK_0_RUN_IN_STANDBY         true
#  define CONF_CLOCK_GCLK_0_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_DFLL
#  define CONF_CLOCK_GCLK_0_PRESCALER              1
#  define CONF_CLOCK_GCLK_0_OUTPUT_ENABLE          false

// GCLK発振器1構成設定
#  define CONF_CLOCK_GCLK_1_ENABLE                 true
#  define CONF_CLOCK_GCLK_1_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_1_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_XOSC32K
#  define CONF_CLOCK_GCLK_1_PRESCALER              1
#  define CONF_CLOCK_GCLK_1_OUTPUT_ENABLE          true
```

8.3. USBﾎｽﾄの高度な使用事例

- USB高速(HS)支援許可
- 複数ｸﾗｽ支援
- 二重役割支援

8.3.1. USB高速(HS)支援許可

この事例ではUSB高速(HS)を支援するのにUSBﾎｽﾄが使われます。

8.3.1.1. 構成設定段階

この事例の実装に先だって、既に”基本的な使用事例”のUHI単位部が適用されていることを確実にしてください。

8.3.1.2. 使用段階

ｺｰﾄﾞ例

conf_usb_host.hの内容

```
#define USB_HOST_HS_SUPPORT
```

作業の流れ

1. conf_usb_host.hが利用可能でUSB装置高速(HS,480Mbps)に必要とされる以下のﾊﾟﾗﾒｰﾀを含むことを確実にしてください。

```
#define  USB_HOST_HS_SUPPORT
```

8.3.2. 複数ｸﾗｽ支援

この事例では様々なUSBｸﾗｽを支援するのにUSBﾎｽﾄが使われます。

8.3.2.1. 構成設定段階

この事例の実装に先だって、既に”基本的な使用事例”のUHI単位部が適用されていることを確実にしてください。

8.3.2.2. 使用段階

ｺｰﾄﾞ例

conf_usb_host.hの内容

```
#define USB_HOST_UHI    UHI_HID_MOUSE, UHI_MSC, UHI_CDC
```

作業の流れ

1. conf_usb_host.hが利用可能で以下のﾊﾟﾗﾒｰﾀを含むことを確実にしてください。

```
#define USB_HOST_UHI    UHI_HID_MOUSE, UHI_MSC, UHI_CDC
```

注：USB_HOST_UHIはUSBﾎｽﾄによって支援されるUHIの一覧を定義します。ここで、支援を望む全てのｸﾗｽを追加しなければなりません。

8.3.3. 二重役割支援

この事例ではUSBﾎｽﾄとUSB装置が許可され、これは二重役割です。

注：Atmelの基板では、USBの役割がUSB On The Go(OTG)ｺﾈｸﾀとそれのUSB IDﾋﾟﾝにより、USB階層によって自動的に管理されます。更なる情報については以下の応用記述で「二重役割」項を参照してください。

- Atmel AVR4950:ASF - USBﾎｽﾄ階層

8.3.3.1. 構成設定段階

この事例の実装に先だって、既に”基本的な使用事例”のUHI単位部が適用されていることを確実にしてください。

8.3.3.2. 使用段階

ｺｰﾄﾞ例

conf_usb_host.hの内容

```
#define UHC_MODE_CHANGE(b_host_mode)    my_callback_mode_change(b_host_mode)
extern void my_callback_mode_change(bool b_host_mode);
```

応用Cﾌｧｲﾙに以下を追加してください。

```
void usb_init(void)
{
  //udc_start();
  uhc_start();
}

bool my_host_mode;
void my_callback_mode_change(bool b_host_mode)
{
  my_host_mode = b_host_mode;
}

void my_usb_task(void)
{
  if (my_host_mode) {
    // USBﾎｽﾄ作業呼び出し
  } else {
```

```
    // USB装置作業呼び出し
  }
}
```

作業の流れ

1. USB二重役割(ホストと装置)の事例では、uhc_start()によってUSB階層が許可されなければならず、udc_start()が呼ばれてはなりません。

```
//udc_start();
uhc_start();
```

2. 二重役割では、現在のUSB動作形態を知るために、動作形態変更を通知する呼び戻しを使うことができます。

・conf_usb_host.hが以下のパラメータを含むことを確実にしてください。

```
#define UHC_MODE_CHANGE(b_host_mode)    my_callback_mode_change(b_host_mode)
extern void my_callback_mode_change(bool b_host_mode);
```

・応用が以下のコードを含むことを確実にしてください。

```
bool my_host_mode;
void my_callback_mode_change(bool b_host_mode)
{
  my_host_mode = b_host_mode;
}

void my_usb_task(void)
{
  if (my_host_mode) {
    // USBホスト作業呼び出し
  } else {
    // USB装置作業呼び出し
  }
}
```

9. 通信ｸﾗｽ装置(CDC)用USBﾎｽﾄ ｲﾝﾀｰﾌｪｰｽ(UHI)

通信ｸﾗｽ装置(CDC)用USBﾎｽﾄ ｲﾝﾀｰﾌｪｰｽ(UHI)はUSB CDC直列ﾎｽﾄの構成設定と管理に関するｲﾝﾀｰﾌｪｰｽを提供します。
この資料の概要は以下のとおりです。

- API概要
- USBﾎｽﾄ通信ｸﾗｽ装置単位部(UHI CDC)用の即時開始の手引き
- 構成設定ﾌｧｲﾙ例

Atmel®ｿﾌﾄｳｪｱ枠組み(ASF) USBﾎｽﾄ階層に関するより多くの詳細については以下の応用記述を参照してください。

- AVR4950：ASF – USBﾎｽﾄ階層

9.1. API概要

9.1.1. ﾏｸﾛ定義

9.1.1.1. USBﾎｽﾄ ｺｱ(UHC)とのｲﾝﾀｰﾌｪｰｽ

UHCによって必要とされる定義と関数

UHI_CDC ﾏｸﾛ

```
#define UHI_CDC
```

UHC用の標準UHI APIを含む全域定義。conf_usb_host.hﾌｧｲﾙからUSB_HOST_UHI定義で追加されなければなりません。

9.1.2. 関数定義

9.1.2.1. UHCによって必要とされる関数

uhi_cdc_install()関数

支援されるなら、割り当てｲﾝﾀｰﾌｪｰｽ ｴﾝﾄﾞﾎﾟｲﾝﾄのｲﾝﾀｰﾌｪｰｽを取り付け

```
uhc_enum_status_t uhi_cdc_install( uhc_device_t * dev )
```

表9-1. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
dev	[入力]	要求する装置

戻り値：取り付けの状態

uhi_cdc_enable()関数

ｲﾝﾀｰﾌｪｰｽを許可

```
void uhi_cdc_enable( uhc_device_t * dev )
```

UHIに対応するUSBｲﾝﾀｰﾌｪｰｽを許可

表9-2. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
dev	[入力]	要求する装置

uhi_cdc_enable()関数

(取り付けられていれば)ｲﾝﾀｰﾌｪｰｽを取り外し

```
void uhi_cdc_uninstall( uhc_device_t * dev )
```

表9-3. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
dev	[入力]	要求する装置

uhi_cdc_sof()関数

SOFが発生した合図

```
void uhi_cdc_sof( bool b_micro )
```

9.1.2.2. 通信装置ｸﾗｽ用UHI

このUSBﾎｽﾄ ｸﾗｽを使うために高位応用によって使われる共通的なAPI。これらのﾙｰﾁﾝはUSB CDCｲﾝﾀｰﾌｪｰｽとそのﾃﾞｰﾀを転送するのにﾒﾓﾘによって使われます。

uhi_cdc_open()関数

UHI CDCインターフェースのポートを開きます。

```
bool uhi_cdc_open( uint8_t port, usb_cdc_line_coding_t * configuration )
```

表9-4. パラメータ

パラメータ名	データ方向	説明
port	[入力]	通信ポート番号
configuration	[入力]	ポート構成設定上のポインタ

戻り値：ポートが利用可能ならばtrue

uhi_cdc_close()関数

ポートを閉じます。

```
void uhi_cdc_close( uint8_t port )
```

表9-5. パラメータ

パラメータ名	データ方向	説明
port	[入力]	通信ポート番号

uhi_cdc_is_rx_ready()関数

この関数はCDC線で文字が受信されたかを調べます。

```
bool uhi_cdc_is_rx_ready( uint8_t port )
```

表9-6. パラメータ

パラメータ名	データ方向	説明
port	[入力]	通信ポート番号

戻り値：読むべきバイトが準備可ならばtrue

uhi_cdc_get_nb_received()関数

この関数はCDC線で利用可能な文字数を返します。

```
iram_size_t uhi_cdc_get_nb_received( uint8_t port )
```

表9-7. パラメータ

パラメータ名	データ方向	説明
port	[入力]	通信ポート番号

戻り値：受信したデータ数

uhi_cdc_getc()関数

待機してCDC線上の値を取得

```
int uhi_cdc_getc( uint8_t port )
```

表9-8. パラメータ

パラメータ名	データ方向	説明
port	[入力]	通信ポート番号

戻り値：CDC線で読んだ値

uhi_cdc_read_buf()関数

CDC線のRAM緩衝部を読みます。

```
iram_size_t uhi_cdc_read_buf( uint8_t port, void * buf, iram_size_t size )
```

表9-9. パラメータ

パラメータ名	データ方向	説明
port	[入力]	通信ポート番号
buf	[出力]	読んだ値
size	[入力]	読んだ値の数

戻り値：残りﾃﾞｰﾀ数

uhi_cdc_is_tx_ready()関数

この関数は新しい文字の送出が可能かを調べます。

```
bool uhi_cdc_is_tx_ready( uint8_t port )
```

ｺﾝﾊﾟｲﾗLIBからのscanf再指定を支援するためにint型が使われます。

表9-10. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
port	[入力]	通信ﾎﾟｰﾄ番号

戻り値：新しい文字を送れる場合にtrue

uhi_cdc_putc()関数

CDC線にﾊﾞｲﾄを出力。ｺﾝﾊﾟｲﾗLIBからのprintf再指定を支援するためにint型が使われます。

```
int uhi_cdc_putc( uint8_t port, int value )
```

表9-11. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
port	[入力]	管理する通信ﾎﾟｰﾄ番号
value	[入力]	送出する値

戻り値：関数が成功裏に終了した場合にtrue、さもなければfalse

uhi_cdc_write_buf()関数

CDC線のRAM緩衝部に書きます。

```
iram_size_t uhi_cdc_write_buf( uint8_t port, const void * buf, iram_size_t size )
```

表9-12. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
port	[入力]	通信ﾎﾟｰﾄ番号
buf	[入力]	書く値
size	[入力]	書く値の数

戻り値：残りﾃﾞｰﾀ数

9.2. USBﾎｽﾄ通信装置ｸﾗｽ単位部(UHI CDC)用の即時開始の手引き

これは使用事例の選択に於いて単位部をどう構成設定して使うかを段階的に指示する、USBﾎｽﾄ通信装置ｸﾗｽ単位部(UHI CDC)用の即時開始の手引きです。

使用事例は様々なｺｰﾄﾞの断片を強調します。構成設定に関する段階でのｺｰﾄﾞの断片は独自初期化関数内に複写することができ、一方使用に関する段階は例えば主応用関数内に複写することができます。

9.2.1. 基本的な使用事例

この基本的な使用事例では”USB Host CDC (Single Class support)”単位部が使われます。”USB Host CDC (Multiple Classes support)”単位部の使い方は高度な使用事例で記述されます。

9.2.1.1. 構成設定段階

USBﾎｽﾄのため、共通USBﾎｽﾄ構成設定段階に従います。USBﾎｽﾄ基本構成設定を参照してください。

9.2.1.2. 使用段階

ｺｰﾄﾞ例

conf_usb_host.hの内容

```
#define USB_HOST_UHI        UHI_CDC
#define UHI_CDC_CHANGE(dev, b_plug) my_callback_cdc_change(dev, b_plug)
extern bool my_callback_cdc_change(uhc_device_t* dev, bool b_plug);
#define UHI_CDC_RX_NOTIFY() my_callback_cdc_rx_notify()
extern void my_callback_cdc_rx_notify(void);
#include "uhi_cdc.h" // conf_usb_host.hﾌｧｲﾙの最後で
```

応用Cﾌｧｲﾙに追加してください。

```
static bool my_flag_cdc_available = false;
bool my_callback_cdc_change(uhc_device_t* dev, bool b_plug)
{
   if (b_plug) {

   // USB装置CDC接続
   my_flag_cdc_available = true;
   // USB CDCﾎﾟｰﾄを開いて構成設定
   usb_cdc_line_coding_t cfg = {
         .dwDTERate   = CPU_TO_LE32(115200),
         .bCharFormat = CDC_STOP_BITS_1,
         .bParityType = CDC_PAR_NONE,
         .bDataBits   = 8,
   };
   uhi_cdc_open(0, &cfg);

   } else {

   my_flag_cdc_available = false;

   }
}

void my_callback_cdc_rx_notify(void)
{
   // my_task_rx()ﾀｽｸ起動
}
#define MESSAGE "Hello"
void my_task(void)
{
   static bool startup = true;

   if (!my_flag_cdc_available) {
      startup = true;
      return;
   }

   if (startup) {
      startup = false;
      // CDC通信ﾎﾟｰﾄにﾃﾞｰﾀ送出
      uhi_cdc_write_buf(0, MESSAGE, sizeof(MESSAGE)-1);
      uhi_cdc_putc(0,'¥n');
      return;
   }
}

void my_task_rx(void)
{
   while (uhi_cdc_is_rx_ready(0)) {
      int value = uhi_cdc_getc(0);
   }
}
```

作業の流れ

1. conf_usb_host.hが利用可能でUSBﾎｽﾄCDC構成設定である以下の構成設定を含むことを確実にしてください。

```
#define USB_HOST_UHI   UHI_CDC
```

注：これはUSBﾎｽﾄによって支援されるUHIの一覧を定義します。

```
#define UHI_CDC_CHANGE(dev, b_plug) my_callback_cdc_change(dev, b_plug)
extern bool my_callback_cdc_change(uhc_device_t* dev, bool b_plug);
```

注：この呼び戻しはUSB装置CDCが接続または切断される時に呼ばれます。通信ﾎﾟｰﾄはここで開かれて構成設定されます。

```
#define UHI_CDC_RX_NOTIFY() my_callback_cdc_rx_notify()
extern void my_callback_cdc_rx_notify(void);
```

注：この呼び戻しは新しいデータが受信された時に呼ばれます。これは割り込みを通してデータ受信を管理して滞留を避けるのに使うことができます。

2. CDCデータ アクセス関数はUHI CDC API概要で記述されます。

9.2.2. 高度な使用事例

UHI CDC単位部のもっと高度な使用については以下をご覧ください。

・USBホストの高度な使用事例

9.3. 構成設定ファイル例

9.3.1. conf_usb_host.h

9.3.1.1. 単一UHI CDC

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_USB_HOST_H_
#define _CONF_USB_HOST_H_

#include "compiler.h"

#define USB_HOST_UHI         UHI_CDC

#define USB_HOST_POWER_MAX   500

// #define USB_HOST_HUB_SUPPORT

#if (UC3A3||UC3A4)
#  define USB_HOST_HS_SUPPORT
#endif

//#define UHC_MODE_CHANGE(b_host_mode)         usb_host_mode_change(b_host_mode)

//#define UHC_VBUS_CHANGE(b_present)           usb_host_vbus_change(b_present)

//#define UHC_VBUS_ERROR()                     usb_host_vbus_error()

//#define UHC_CONNECTION_EVENT(dev,b_present)  usb_host_connection_event(dev,b_present)

//#define UHC_WAKEUP_EVENT()                   usb_host_wakeup_event()

//#define UHC_SOF_EVENT()                      usb_host_sof_event()

//#define UHC_DEVICE_CONF(dev)                 uint8_t usb_host_device_conf(dev)

//#define UHC_ENUM_EVENT(dev,b_status)         usb_host_enum_event(dev,b_status)

#define UHI_CDC_CHANGE(dev,b_plug)
#define UHI_CDC_RX_NOTIFY()

#include "uhi_cdc.h"

#endif // _CONF_USB_HOST_H_
```

9.3.1.2. 複数UHI CDC (複合)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */
```

```
#ifndef _CONF_USB_HOST_H_
#define _CONF_USB_HOST_H_

#include "compiler.h"

#define USB_HOST_UHI         // UHI_MSC, UHI_HID_MOUSE, UHI_CDC, UHI_VENDOR

#define USB_HOST_POWER_MAX  500

// #define USB_HOST_HUB_SUPPORT

#if (UC3A3 || UC3A4)
#  define USB_HOST_HS_SUPPORT
#endif

//#define UHC_MODE_CHANGE(b_host_mode)         usb_host_mode_change(b_host_mode)

//#define UHC_VBUS_CHANGE(b_present)           usb_host_vbus_change(b_present)

//#define UHC_VBUS_ERROR()                     usb_host_vbus_error()

//#define UHC_CONNECTION_EVENT(dev,b_present) usb_host_connection_event(dev,b_present)

//#define UHC_WAKEUP_EVENT()                   usb_host_wakeup_event()

//#define UHC_SOF_EVENT()                      usb_host_sof_event()

//#define UHC_DEVICE_CONF(dev)                 uint8_t usb_host_device_conf(dev)

//#define UHC_ENUM_EVENT(dev,b_status)         usb_host_enum_event(dev,b_status)

#define UHI_HID_MOUSE_CHANGE(dev,b_plug)
#define UHI_HID_MOUSE_EVENT_BTN_LEFT(b_state)
#define UHI_HID_MOUSE_EVENT_BTN_RIGHT(b_state)
#define UHI_HID_MOUSE_EVENT_BTN_MIDDLE(b_state)
#define UHI_HID_MOUSE_EVENT_MOUVE(x,y,scroll)

#define UHI_MSC_CHANGE(dev,b_plug)

#define UHI_CDC_CHANGE(dev,b_plug)
#define UHI_CDC_RX_NOTIFY()

#define UHI_VENDOR_CHANGE(dev, b_plug)
#define UHI_VENDOR_VID_PID_LIST {USB_VID_ATMEL, USB_PID_ATMEL_ASF_VENDOR_CLASS}

//#include "uhi_msc.h"
//#include "uhi_hid_mouse.h"

#endif // _CONF_USB_HOST_H_
```

9.3.2. conf_clock.h

9.3.2.1. AT32UC3A0, AT32UC3A1, AT32UC3Bﾃﾞﾊﾞｲｽ (USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED
```

```
/* ===== システム クロック元任意選択 */
//#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_RCSYS
#define CONFIG_SYSCLK_SOURCE            SYSCLK_SRC_OSC0
//#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_PLL0

/* ===== PLL0任意選択 */
//#define CONFIG_PLL0_SOURCE            PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE            PLL_SRC_OSC1
//#define CONFIG_PLL0_MUL               4 /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL0_DIV               1 /* Fpll = (Fclk * PLL_mul) / PLL_div */

/* ===== PLL1任意選択 */
#define CONFIG_PLL1_SOURCE              PLL_SRC_OSC0
//#define CONFIG_PLL1_SOURCE            PLL_SRC_OSC1
#define CONFIG_PLL1_MUL                 8 /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL1_DIV                 2 /* Fpll = (Fclk * PLL_mul) / PLL_div */

/* ===== システム クロック バス分周任意選択 */
//#define CONFIG_SYSCLK_CPU_DIV         0 /* Fcpu = Fsys/(2 ^ CPU_div) */
//#define CONFIG_SYSCLK_PBA_DIV         0 /* Fpba = Fsys/(2 ^ PBA_div) */
//#define CONFIG_SYSCLK_PBB_DIV         0 /* Fpbb = Fsys/(2 ^ PBB_div) */

/* ===== 周辺機能クロック管理任意選択 */
//#define CONFIG_SYSCLK_INIT_CPUMASK    ((1 << SYSCLK_SYSTIMER) | (1 << SYSCLK_OCD))
//#define CONFIG_SYSCLK_INIT_PBAMASK    (1 << SYSCLK_USART0)
//#define CONFIG_SYSCLK_INIT_PBBMASK    (1 << SYSCLK_HMATRIX)
//#define CONFIG_SYSCLK_INIT_HSBMASK    (1 << SYSCLK_MDMA_HSB)

/* ===== USBクロック元任意選択 */
//#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_OSC0
//#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_PLL0
#define CONFIG_USBCLK_SOURCE            USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV               1 /* Fusb = Fsys/(2 ^ USB_div) */

#endif /* CONF_CLOCK_H_INCLUDED */
```

9.3.2.2. AT32UC3A3, AT32UC3A4ﾃﾞﾊﾞｲｽ (高速(HS)支援付USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

/* ===== システム クロック元任意選択 */
//#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_RCSYS
//#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_OSC0
#define CONFIG_SYSCLK_SOURCE            SYSCLK_SRC_PLL0

/* ===== PLL0任意選択 */
#define CONFIG_PLL0_SOURCE              PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE            PLL_SRC_OSC1
#define CONFIG_PLL0_MUL                 11 /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL0_DIV                 2  /* Fpll = (Fclk * PLL_mul) / PLL_div */

/* ===== PLL1任意選択 */
//#define CONFIG_PLL1_SOURCE            PLL_SRC_OSC0
//#define CONFIG_PLL1_SOURCE            PLL_SRC_OSC1
//#define CONFIG_PLL1_MUL               8 /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL1_DIV               2 /* Fpll = (Fclk * PLL_mul) / PLL_div */
```

```
/* ===== システム クロック バス分周任意選択 */
#define CONFIG_SYSCLK_CPU_DIV          0 /* Fcpu = Fsys/(2 ^ CPU_div) */
#define CONFIG_SYSCLK_PBA_DIV          0 /* Fpba = Fsys/(2 ^ PBA_div) */
//#define CONFIG_SYSCLK_PBB_DIV        0 /* Fpbb = Fsys/(2 ^ PBB_div) */

/* ===== 周辺機能クロック管理任意選択 */
//#define CONFIG_SYSCLK_INIT_CPUMASK  ((1 << SYSCLK_SYSTIMER) | (1 << SYSCLK_OCD))
//#define CONFIG_SYSCLK_INIT_PBAMASK  (1 << SYSCLK_USART0)
//#define CONFIG_SYSCLK_INIT_PBBMASK  (1 << SYSCLK_HMATRIX)
//#define CONFIG_SYSCLK_INIT_HSBMASK  (1 << SYSCLK_MDMA_HSB)

/* ===== USBクロック元任意選択 */
#define CONFIG_USBCLK_SOURCE           USBCLK_SRC_OSC0
//#define CONFIG_USBCLK_SOURCE         USBCLK_SRC_PLL0
//#define CONFIG_USBCLK_SOURCE         USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV              1 /* Fusb = Fsys/(2 ^ USB_div) */

#endif /* CONF_CLOCK_H_INCLUDED */
```

9.3.2.3. AT32UC3C, ATUCXXD, ATUCXXL3U, ATUCXXL4Uﾃﾞﾊﾞｲｽ (USBC)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== システム クロック元任意選択
//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_RCSYS
//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_OSC0
//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_OSC1
#define CONFIG_SYSCLK_SOURCE           SYSCLK_SRC_PLL0
//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_PLL1
//#define CONFIG_SYSCLK_SOURCE         SYSCLK_SRC_RC8M

// ===== PLL0任意選択
#define CONFIG_PLL0_SOURCE             PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE           PLL_SRC_OSC1
//#define CONFIG_PLL0_SOURCE           PLL_SRC_RC8M
#define CONFIG_PLL0_MUL                3 /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL0_DIV                1 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== PLL1任意選択
//#define CONFIG_PLL1_SOURCE           PLL_SRC_OSC0
//#define CONFIG_PLL1_SOURCE           PLL_SRC_OSC1
//#define CONFIG_PLL1_SOURCE           PLL_SRC_RC8M
//#define CONFIG_PLL1_MUL              3 /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL1_DIV              1 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== システム クロック バス分周任意選択
//#define CONFIG_SYSCLK_CPU_DIV        0 /* Fcpu = Fsys/(2 ^ CPU_div) */
//#define CONFIG_SYSCLK_PBA_DIV        0 /* Fpba = Fsys/(2 ^ PBA_div) */
//#define CONFIG_SYSCLK_PBB_DIV        0 /* Fpbb = Fsys/(2 ^ PBB_div) */
//#define CONFIG_SYSCLK_PBC_DIV        0 /* Fpbc = Fsys/(2 ^ PBC_div) */

// ===== 周辺機能クロック管理任意選択
//#define CONFIG_SYSCLK_INIT_CPUMASK  ((1 << SYSCLK_SYSTIMER) | (1 << SYSCLK_OCD))
//#define CONFIG_SYSCLK_INIT_PBAMASK  (1 << SYSCLK_USART0)
//#define CONFIG_SYSCLK_INIT_PBBMASK  (1 << SYSCLK_HMATRIX)
//#define CONFIG_SYSCLK_INIT_HSBMASK  (1 << SYSCLK_MDMA_HSB)
```

```
// ===== USBクロック元任意選択
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_OSC0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_OSC1
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_PLL0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL1
define CONFIG_USBCLK_DIV             1 /* Fusb = Fsys/(2 ^ USB_div) */

#endif /* CONF_CLOCK_H_INCLUDED */
```

9.3.2.4. SAM3X, SAM3Aﾃﾞﾊﾞｲｽ (UOTGHS:USB OTG 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

/* ===== システム クロック(MCK)元任意選択 */
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_BYPASS
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_4M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_8M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_12M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_BYPASS
#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLLACK
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_UPLLCK

/* ===== システム クロック(MCK)前置分周器任意選択    (Fmck = Fsys / (SYSCLK_PRES)) */
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_1
#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_2
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_4
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_8
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_16
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_32
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_64
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_3

/* ===== PLL0 (A)任意選択    (Fpll = (Fclk * PLL_mul) / PLL_div)
   ここにmulとdivの実効値を使ってください。 */
#define CONFIG_PLL0_SOURCE          PLL_SRC_MAINCK_XTAL
#define CONFIG_PLL0_MUL             14
#define CONFIG_PLL0_DIV             1

/* ===== 480MHzで固定化されたUPLL (UTMI)ﾊｰﾄﾞｳｪｱ */

/* ===== USBクロック元任意選択    (Fusb = FpllX / USB_div)
   ここにdivの実効値を使ってください。 */
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL0
#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_UPLL
#define CONFIG_USBCLK_DIV           1

/*
===== 目的対象周波数 (システム クロック)
- XTAL周波数: 12MHz
- システム クロック元: PLLA
- システム クロック前置分周器: 2 (2分周)
- PLLA供給元: XTAL
- PLLA出力: XTAL * 14 / 1
- システム クロックは: 12 * 14 / 1 /2 = 84MHz
```

```
===== 目的対象周波数 (USBｸﾛｯｸ)
- USBｸﾛｯｸ元: UPLL
- USBｸﾛｯｸ分周器: 1 (分周なし)
- UPLL周波数: 480MHz
*/

#endif /* CONF_CLOCK_H_INCLUDED */
```

9.3.3. conf_clocks.h

9.3.3.1. SAM D21ﾃﾞﾊﾞｲｽ (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#include <clock.h>

#ifndef CONF_CLOCKS_H_INCLUDED
#  define CONF_CLOCKS_H_INCLUDED

/* ｼｽﾃﾑ ｸﾛｯｸ ﾊﾞｽ構成設定 */
#  define CONF_CLOCK_CPU_CLOCK_FAILURE_DETECT      false
#  define CONF_CLOCK_FLASH_WAIT_STATES             2
#  define CONF_CLOCK_CPU_DIVIDER                   SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBA_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBB_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBC_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1

/* SYSTEM_CLOCK_SOURCE_OSC8M構成設定 - 内部8MHz発振器 */
#  define CONF_CLOCK_OSC8M_PRESCALER               SYSTEM_OSC8M_DIV_1
#  define CONF_CLOCK_OSC8M_ON_DEMAND               true
#  define CONF_CLOCK_OSC8M_RUN_IN_STANDBY          true

/* SYSTEM_CLOCK_SOURCE_XOSC構成設定 - 外部ｸﾛｯｸ/発振器 */
#  define CONF_CLOCK_XOSC_ENABLE                   false
#  define CONF_CLOCK_XOSC_EXTERNAL_CRYSTAL         SYSTEM_CLOCK_EXTERNAL_CRYSTAL
#  define CONF_CLOCK_XOSC_EXTERNAL_FREQUENCY       12000000UL
#  define CONF_CLOCK_XOSC_STARTUP_TIME             SYSTEM_XOSC_STARTUP_32768
#  define CONF_CLOCK_XOSC_AUTO_GAIN_CONTROL        true
#  define CONF_CLOCK_XOSC_ON_DEMAND                true
#  define CONF_CLOCK_XOSC_RUN_IN_STANDBY           false

/* SYSTEM_CLOCK_SOURCE_XOSC32K構成設定 - 外部32kHzｸﾘｽﾀﾙ/ｸﾛｯｸ発振器 */
#  define CONF_CLOCK_XOSC32K_ENABLE                true
#  define CONF_CLOCK_XOSC32K_EXTERNAL_CRYSTAL      SYSTEM_CLOCK_EXTERNAL_CRYSTAL
#  define CONF_CLOCK_XOSC32K_STARTUP_TIME          SYSTEM_XOSC32K_STARTUP_65536
#  define CONF_CLOCK_XOSC32K_AUTO_AMPLITUDE_CONTROL  false
#  define CONF_CLOCK_XOSC32K_ENABLE_1KHZ_OUPUT     false
#  define CONF_CLOCK_XOSC32K_ENABLE_32KHZ_OUTPUT   true
#  define CONF_CLOCK_XOSC32K_ON_DEMAND             false
#  define CONF_CLOCK_XOSC32K_RUN_IN_STANDBY        true

/* SYSTEM_CLOCK_SOURCE_OSC32K構成設定 - 内部32kHz発振器 */
#  define CONF_CLOCK_OSC32K_ENABLE                 false
#  define CONF_CLOCK_OSC32K_STARTUP_TIME           SYSTEM_OSC32K_STARTUP_130
#  define CONF_CLOCK_OSC32K_ENABLE_1KHZ_OUTPUT     false
#  define CONF_CLOCK_OSC32K_ENABLE_32KHZ_OUTPUT    true
#  define CONF_CLOCK_OSC32K_ON_DEMAND              true
#  define CONF_CLOCK_OSC32K_RUN_IN_STANDBY         false

/* SYSTEM_CLOCK_SOURCE_DFLL構成設定 - ﾃﾞｼﾞﾀﾙ周波数固定化閉路 */
#  define CONF_CLOCK_DFLL_ENABLE                   true
```

```
#  define CONF_CLOCK_DFLL_LOOP_MODE                SYSTEM_CLOCK_DFLL_LOOP_MODE_CLOSED
#  define CONF_CLOCK_DFLL_ON_DEMAND                true

/* DFLL開路動作構成設定 */
#  define CONF_CLOCK_DFLL_FINE_VALUE               (512)

/* DFLL閉路動作構成設定 */
#  define CONF_CLOCK_DFLL_SOURCE_GCLK_GENERATOR    GCLK_GENERATOR_1
#  define CONF_CLOCK_DFLL_MULTIPLY_FACTOR          (48000000/32768)
#  define CONF_CLOCK_DFLL_QUICK_LOCK               true
#  define CONF_CLOCK_DFLL_TRACK_AFTER_FINE_LOCK    true
#  define CONF_CLOCK_DFLL_KEEP_LOCK_ON_WAKEUP      true
#  define CONF_CLOCK_DFLL_ENABLE_CHILL_CYCLE       true
#  define CONF_CLOCK_DFLL_MAX_COARSE_STEP_SIZE     (0x1f / 8)
#  define CONF_CLOCK_DFLL_MAX_FINE_STEP_SIZE       (0xff / 8)

/* SYSTEM_CLOCK_SOURCE_DPLL構成設定 - ﾃﾞｼﾞﾀﾙ位相固定化閉路 */
#  define CONF_CLOCK_DPLL_ENABLE                   false
#  define CONF_CLOCK_DPLL_ON_DEMAND                false
#  define CONF_CLOCK_DPLL_RUN_IN_STANDBY           true
#  define CONF_CLOCK_DPLL_LOCK_BYPASS              false
#  define CONF_CLOCK_DPLL_WAKE_UP_FAST             false
#  define CONF_CLOCK_DPLL_LOW_POWER_ENABLE         true
#  define CONF_CLOCK_DPLL_LOCK_TIME                SYSTEM_CLOCK_SOURCE_DPLL_LOCK_TIME_DEFAULT
#  define CONF_CLOCK_DPLL_REFERENCE_CLOCK          SYSTEM_CLOCK_SOURCE_DPLL_REFERENCE_CLOCK_XOSC32K
#  define CONF_CLOCK_DPLL_FILTER                   SYSTEM_CLOCK_SOURCE_DPLL_FILTER_DEFAULT
#  define CONF_CLOCK_DPLL_REFERENCE_FREQUENCY      32768
#  define CONF_CLOCK_DPLL_REFEREMCE_DIVIDER        1
#  define CONF_CLOCK_DPLL_OUTPUT_FREQUENCY         48000000

/* DPLL GCLK基準構成設定 */
# define CONF_CLOCK_DPLL_REFERENCE_GCLK_GENERATOR GCLK_GENERATOR_1
/* DPLL GCLK固定化計時器構成設定 */
# define CONF_CLOCK_DPLL_LOCK_GCLK_GENERATOR GCLK_GENERATOR_1

/* clocks_init走行時にGCLKを構成設定するためにこれをtrueに設定してください。
 * falseに設定した場合、GCLK発振器はclocks_init()で何も構成設定されません。*/
#  define CONF_CLOCK_CONFIGURE_GCLK                true

/* GCLK発振器0構成設定 (主ｸﾛｯｸ) */
#  define CONF_CLOCK_GCLK_0_ENABLE                 true
#  define CONF_CLOCK_GCLK_0_RUN_IN_STANDBY         true
#  define CONF_CLOCK_GCLK_0_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_DFLL
#  define CONF_CLOCK_GCLK_0_PRESCALER              1
#  define CONF_CLOCK_GCLK_0_OUTPUT_ENABLE          false

/* GCLK発振器1構成設定 */
#  define CONF_CLOCK_GCLK_1_ENABLE                 true
#  define CONF_CLOCK_GCLK_1_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_1_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_XOSC32K
#  define CONF_CLOCK_GCLK_1_PRESCALER              1
#  define CONF_CLOCK_GCLK_1_OUTPUT_ENABLE          false

/* GCLK発振器2構成設定 (RTC) */
#  define CONF_CLOCK_GCLK_2_ENABLE                 false
#  define CONF_CLOCK_GCLK_2_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_2_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC32K
#  define CONF_CLOCK_GCLK_2_PRESCALER              32
#  define CONF_CLOCK_GCLK_2_OUTPUT_ENABLE          false

/* GCLK発振器3構成設定 */
```

```
#  define CONF_CLOCK_GCLK_3_ENABLE                 false
#  define CONF_CLOCK_GCLK_3_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_3_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_3_PRESCALER              1
#  define CONF_CLOCK_GCLK_3_OUTPUT_ENABLE          false

/* GCLK発振器4構成設定 */
#  define CONF_CLOCK_GCLK_4_ENABLE                 false
#  define CONF_CLOCK_GCLK_4_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_4_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_4_PRESCALER              1
#  define CONF_CLOCK_GCLK_4_OUTPUT_ENABLE          false

/* GCLK発振器5構成設定 */
#  define CONF_CLOCK_GCLK_5_ENABLE                 false
#  define CONF_CLOCK_GCLK_5_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_5_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_5_PRESCALER              1
#  define CONF_CLOCK_GCLK_5_OUTPUT_ENABLE          false

/* GCLK発振器6構成設定 */
#  define CONF_CLOCK_GCLK_6_ENABLE                 false
#  define CONF_CLOCK_GCLK_6_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_6_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_6_PRESCALER              1
#  define CONF_CLOCK_GCLK_6_OUTPUT_ENABLE          false

/* GCLK発振器7構成設定 */
#  define CONF_CLOCK_GCLK_7_ENABLE                 false
#  define CONF_CLOCK_GCLK_7_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_7_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_7_PRESCALER              1
#  define CONF_CLOCK_GCLK_7_OUTPUT_ENABLE          false

/* GCLK発振器8構成設定 */
#  define CONF_CLOCK_GCLK_8_ENABLE                 false
#  define CONF_CLOCK_GCLK_8_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_8_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_8_PRESCALER              1
#  define CONF_CLOCK_GCLK_8_OUTPUT_ENABLE          false

#endif /* CONF_CLOCKS_H_INCLUDED */
```

9.3.4. conf_board.h

9.3.4.1. AT32UC3A0, AT32UC3A1, AT32UC3Bﾃﾞﾊﾞｲｽ (USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

/* UARTﾎﾟｰﾄ許可 */
#define CONF_BOARD_COM_PORT

#endif /* CONF_BOARD_H_INCLUDED */
```

9.3.4.2. AT32UC3A3, AT32UC3A4ﾃﾞﾊﾞｲｽ (高速(HS)支援付USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
```

```
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

/* UARTﾎﾟｰﾄ許可 */
#define CONF_BOARD_COM_PORT

#endif /* CONF_BOARD_H_INCLUDED */
```

9.3.4.3. AT32UC3C, ATUCXXD, ATUCXXL3U, ATUCXXL4Uﾃﾞﾊﾞｲｽ (USBC)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// UARTﾎﾟｰﾄ許可
#define CONF_BOARD_COM_PORT

#endif /* CONF_BOARD_H_INCLUDED */
```

9.3.4.4. SAM3X, SAM3Aﾃﾞﾊﾞｲｽ (UOTGHS:USB OTG 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

/* RXD,TXD(UARTﾋﾟﾝ)に対応するﾋﾟﾝの記述 */
#define CONSOLE_PINS          {PINS_UART}

/* ｺﾝｿｰﾙ(UART0)によって使われるUSARTﾊｰﾄﾞｳｪｱID */
#define CONSOLE_UART_ID            ID_UART

/* UARTﾋﾟﾝ構成設定 */
#define CONF_BOARD_UART_CONSOLE

/* A/D変換器のﾋﾟﾝ構成設定 */
//#define CONF_BOARD_ADC

/* PWM LED0ﾋﾟﾝ構成設定 */
//#define CONF_BOARD_PWM_LED0

/* PWM LED1ﾋﾟﾝ構成設定 */
//#define CONF_BOARD_PWM_LED1

/* PWM LED2ﾋﾟﾝ構成設定 */
//#define CONF_BOARD_PWM_LED2

/* SPI0ﾋﾟﾝ構成設定 */
//#define CONF_BOARD_SPI0
//#define CONF_BOARD_SPI0_NPCS0
//#define CONF_BOARD_SPI0_NPCS1
//#define CONF_BOARD_SPI0_NPCS2
//#define CONF_BOARD_SPI0_NPCS3

/* SPI1ﾋﾟﾝ構成設定 */
//#define CONF_BOARD_SPI1
```

```
//#define CONF_BOARD_SPI1_NPCS0
//#define CONF_BOARD_SPI1_NPCS1
//#define CONF_BOARD_SPI1_NPCS2
//#define CONF_BOARD_SPI1_NPCS3

//#define CONF_BOARD_TWI0

//#define CONF_BOARD_TWI1

/* USART RXDﾋﾟﾝ構成設定 */
//#define CONF_BOARD_USART_RXD

/* USART TXDﾋﾟﾝ構成設定 */
//#define CONF_BOARD_USART_TXD

/* USART CTSﾋﾟﾝ構成設定 */
//#define CONF_BOARD_USART_CTS

/* USART RTSﾋﾟﾝ構成設定 */
//#define CONF_BOARD_USART_RTS

/* USART同期通信SCKﾋﾟﾝ構成設定 */
//#define CONF_BOARD_USART_SCK

/* ADM3312許可ﾋﾟﾝ構成設定 */
//#define CONF_BOARD_ADM3312_EN

/* IrDA送受信部停止ﾋﾟﾝ構成設定 */
//#define CONF_BOARD_TFDU4300_SD

/* RS485送受信部ADM3485 REﾋﾟﾝ構成設定 */
//#define CONF_BOARD_ADM3485_RE

//#define CONF_BOARD_SMC_PSRAM

/* LCD EBIﾋﾟﾝ構成設定 */
//#define CONF_BOARD_HX8347A

/* 背面灯制御ﾋﾟﾝ構成設定 */
//#define CONF_BOARD_AAT3194

/* USBﾋﾟﾝ構成設定 */
#define CONF_BOARD_USB_PORT

#endif /* CONF_BOARD_H_INCLUDED */
```

9.3.4.5. SAM D21ﾃﾞﾊﾞｲｽ (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

/* USB VBUS検出許可 */
#define CONF_BOARD_USB_VBUS_DETECT
/* ID検出許可 */
#define CONF_BOARD_USB_ID_DETECT

#endif /* CONF_BOARD_H_INCLUDED */
```

10. 人間インターフェース装置マウス(HIDマウス)用USBホスト インターフェース(UHI)

人間インターフェース装置マウス(HIDマウス)用USBホスト インターフェース(UHI)はUSB HIDマウス ホストの構成設定と管理に関するインターフェースを提供します。

この資料の概要は以下のとおりです。

- API概要
- USBホスト マウス単位部(UHIマウス)用の即時開始の手引き
- 構成設定ファイル例

Atmel®ソフトウェア枠組み(ASF) USB装置階層に関するより多くの詳細については以下の応用記述を参照してください。

- AVR4950：ASF – USBホスト階層

10.1. API概要

10.1.1. マクロ定義

10.1.1.1. USBホスト コア(UHC)とのインターフェース

UHCによって必要とされる定義と関数

UHI_HID_MOUSE マクロ

```
#define UHI_HID_MOUSE
```

UHC用の標準UHI APIを含む全域定義

これはconf_usb_host.hファイルのUSB_HOST_UHI定義で追加されなければなりません。

10.1.1.2. 人間インターフェース装置マウス クラス用UHI

このUSBホスト クラスを使う上位応用によって使われる共通的なAPI

これらのAPIは以下の定義を通したconf_usb_host.hファイルでの呼び戻し定義だけが必要です。

UHI_HID_MOUSE_CHANGE マクロ

```
#define UHI_HID_MOUSE_CHANGE(dev, b_plug)
```

UHI_HID_MOUSE_EVENT_BTN_LEFT マクロ

```
#define UHI_HID_MOUSE_EVENT_BTN_LEFT(b_state)
```

UHI_HID_MOUSE_EVENT_BTN_RIGHT マクロ

```
#define UHI_HID_MOUSE_EVENT_BTN_RIGHT(b_state)
```

UHI_HID_MOUSE_EVENT_BTN_MIDDLE マクロ

```
#define UHI_HID_MOUSE_EVENT_BTN_MIDDLE(b_state)
```

UHI_HID_MOUSE_EVENT_MOUVE マクロ

```
#define UHI_HID_MOUSE_EVENT_MOUVE(x, y, scroll)
```

10.1.2. 関数定義

10.1.2.1. UHCによって必要とされる関数

uhi_hid_mouse_install()関数

インターフェースを取り付けして支援されていればインターフェース エンドポイントを割り当て

```
uhc_enum_status_t uhi_hid_mouse_install( uhc_device_t * dev )
```

表10-1. パラメータ

パラメータ名	データ方向	説明
dev	[入力]	要求する装置

戻り値：取り付けの状態

uhi_hid_mouse_enable()関数

インターフェース許可

```
void uhi_hid_mouse_enable( uhc_device_t * dev )
```

UHIに対応するUSBインターフェースを許可

表10-2. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
dev	[入力]	要求する装置

uhi_hid_mouse_uninstall()関数

(取り付けられていれば)ｲﾝﾀｰﾌｪｰｽを取り外し

```
void uhi_hid_mouse_uninstall( uhc_device_t * dev )
```

表10-3. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
dev	[入力]	要求する装置

10.2. USBﾎｽﾄ ﾏｳｽ単位部(UHIﾏｳｽ)用の即時開始の手引き

これは使用事例の選択に於いて単位部をどう構成設定して使うかを段階的に指示する、USBﾎｽﾄ ﾏｳｽ単位部(UHIﾏｳｽ)用の即時開始の手引きです。

使用事例は様々なｺｰﾄﾞの断片を強調します。構成設定に関する段階でのｺｰﾄﾞの断片は独自初期化関数内に複写することができ、一方使用に関する段階は例えば主応用関数内に複写することができます。

10.2.1. 基本的な使用事例

この基本的な使用事例では"USB Host HID Mouse (Single Class support)"単位部が使われます。"USB Host HID Mouse (Multiple Classes support)"単位部の使い方は高度な使用事例で記述されます。

10.2.1.1. 構成設定段階

USBﾎｽﾄのため、共通USBﾎｽﾄ構成設定段階に従います。USBﾎｽﾄ基本構成設定を参照してください。

10.2.1.2. 使用段階

ｺｰﾄﾞ例

conf_usb_host.hの内容

```
#define USB_HOST_UHI         UHI_HID_MOUSE
#define UHI_HID_MOUSE_CHANGE(dev, b_plug) my_callback_mouse_change(dev, b_plug)
extern bool my_callback_mouse_change(uhc_device_t* dev, bool b_plug);
#define UHI_HID_MOUSE_EVENT_BTN_LEFT(b_state) my_callback_event_btn_left(b_state)
extern void my_callback_event_btn_left(bool b_state);
#define UHI_HID_MOUSE_EVENT_BTN_RIGHT(b_state) my_callback_event_btn_right(b_state)
extern void my_callback_event_btn_right(bool b_state);
#define UHI_HID_MOUSE_EVENT_BTN_MIDDLE(b_state) my_callback_event_btn_middle(b_state)
extern void my_callback_event_btn_middle(bool b_state);
#define UHI_HID_MOUSE_EVENT_MOUVE(x, y, scroll) my_callback_event_mouse(x, y, scroll)
extern void my_callback_event_mouse(int8_t x, int8_t y, int8_t scroll);
#include "uhi_hid_mouse.h" // conf_usb_host.hﾌｧｲﾙの最後で
```

応用Cﾌｧｲﾙに追加してください。

```
bool my_callback_mouse_change(uhc_device_t* dev, bool b_plug)
{
   if (b_plug) {
      my_display_on_mouse_icon();
   } else {
      my_display_off_mouse_icon();
   }
}

void my_callback_event_btn_left(bool b_state)
{
  if (b_state) {
      // ここはﾏｳｽ左釦押下
   } else {
      // ここはﾏｳｽ左釦解放
   }
}
```

```
void my_callback_event_mouse(int8_t x, int8_t y, int8_t scroll)
{
   if (!x) {
      // ここはﾏｳｽがX軸で移動
      cursor_x += x;
   }
   if (!y) {
      // ここはﾏｳｽがY軸で移動
      cursor_y += y;
   }
   if (!scroll) {
      // ここはﾏｳｽの回転輪が移動
      wheel += scroll;
   }
}
```

作業の流れ

1. conf_usb_host.hが利用可能でUSBﾎｽﾄ ﾏｳｽ構成設定である以下の構成設定を含むことを確実にしてください。

```
#define USB_HOST_UHI    UHI_HID_MOUSE
```

注：これはUSBﾎｽﾄによって支援されるUHIの一覧を定義します。

```
#define UHI_HID_MOUSE_CHANGE(dev, b_plug) my_callback_mouse_change(dev, b_plug)
extern bool my_callback_mouse_change(uhc_device_t* dev, bool b_plug);
```

注：この呼び戻しはUSB装置ﾏｳｽが接続または切断された時に呼ばれます。

```
##define UHI_HID_MOUSE_EVENT_BTN_LEFT(b_state) my_callback_event_btn_left(b_state)
extern void my_callback_event_btn_left(bool b_state);
#define UHI_HID_MOUSE_EVENT_BTN_RIGHT(b_state) my_callback_event_btn_right(b_state)
extern void my_callback_event_btn_right(bool b_state);
#define UHI_HID_MOUSE_EVENT_BTN_MIDDLE(b_state) my_callback_event_btn_middle(b_state)
extern void my_callback_event_btn_middle(bool b_state);
#define UHI_HID_MOUSE_EVENT_MOUVE(x, y, scroll) my_callback_event_mouse(x, y, scroll)
extern void my_callback_event_mouse(int8_t x, int8_t y, int8_t scroll)
```

注：これらの呼び戻しはUSB装置ﾏｳｽ事象を受け取った時に呼ばれます。

10.2.2. 高度な使用事例

UHI HIDﾏｳｽ単位部のもっと高度な使用については以下をご覧ください。

・USBﾎｽﾄの高度な使用事例

10.3. 構成設定ﾌｧｲﾙ例

10.3.1. conf_usb_host.h

10.3.1.1. 単一UHI HID MOUSE

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_USB_HOST_H_
#define _CONF_USB_HOST_H_

#include "compiler.h"

#define USB_HOST_UHI         UHI_HID_MOUSE

#define USB_HOST_POWER_MAX   500

// #define USB_HOST_HUB_SUPPORT

#if (UC3A3 || UC3A4)
#  define USB_HOST_HS_SUPPORT
```

```
#endif

//#define UHC_MODE_CHANGE(b_host_mode)         usb_host_mode_change(b_host_mode)

//#define UHC_VBUS_CHANGE(b_present)           usb_host_vbus_change(b_present)

//#define UHC_VBUS_ERROR()                     usb_host_vbus_error()

//#define UHC_CONNECTION_EVENT(dev,b_present) usb_host_connection_event(dev,b_present)

//#define UHC_WAKEUP_EVENT()                   usb_host_wakeup_event()

//#define UHC_SOF_EVENT()                      usb_host_sof_event()

//#define UHC_DEVICE_CONF(dev)                 uint8_t usb_host_device_conf(dev)

//#define UHC_ENUM_EVENT(dev,b_status)         usb_host_enum_event(dev,b_status)

#define UHI_HID_MOUSE_CHANGE(dev,b_plug)
#define UHI_HID_MOUSE_EVENT_BTN_LEFT(b_state)
#define UHI_HID_MOUSE_EVENT_BTN_RIGHT(b_state)
#define UHI_HID_MOUSE_EVENT_BTN_MIDDLE(b_state)
#define UHI_HID_MOUSE_EVENT_MOUVE(x,y,scroll)

#include "uhi_hid_mouse.h"

#endif // _CONF_USB_HOST_H_
```

10.3.1.2. 複数UHI HID MOUSE (複合)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_USB_HOST_H_
#define _CONF_USB_HOST_H_

#include "compiler.h"

#define USB_HOST_UHI         // UHI_MSC, UHI_HID_MOUSE, UHI_CDC, UHI_VENDOR

#define USB_HOST_POWER_MAX  500

// #define USB_HOST_HUB_SUPPORT

#if (UC3A3 || UC3A4)
#  define USB_HOST_HS_SUPPORT
#endif

//#define UHC_MODE_CHANGE(b_host_mode)         usb_host_mode_change(b_host_mode)

//#define UHC_VBUS_CHANGE(b_present)           usb_host_vbus_change(b_present)

//#define UHC_VBUS_ERROR()                     usb_host_vbus_error()

//#define UHC_CONNECTION_EVENT(dev,b_present) usb_host_connection_event(dev,b_present)

//#define UHC_WAKEUP_EVENT()                   usb_host_wakeup_event()

//#define UHC_SOF_EVENT()                      usb_host_sof_event()

//#define UHC_DEVICE_CONF(dev)                 uint8_t usb_host_device_conf(dev)
```

```
//#define UHC_ENUM_EVENT(dev,b_status)         usb_host_enum_event(dev,b_status)

#define UHI_HID_MOUSE_CHANGE(dev,b_plug)
#define UHI_HID_MOUSE_EVENT_BTN_LEFT(b_state)
#define UHI_HID_MOUSE_EVENT_BTN_RIGHT(b_state)
#define UHI_HID_MOUSE_EVENT_BTN_MIDDLE(b_state)
#define UHI_HID_MOUSE_EVENT_MOUVE(x,y,scroll)

#define UHI_MSC_CHANGE(dev,b_plug)

#define UHI_CDC_CHANGE(dev,b_plug)
#define UHI_CDC_RX_NOTIFY()

#define UHI_VENDOR_CHANGE(dev, b_plug)
#define UHI_VENDOR_VID_PID_LIST {USB_VID_ATMEL, USB_PID_ATMEL_ASF_VENDOR_CLASS}

//#include "uhi_msc.h"
//#include "uhi_hid_mouse.h"

#endif // _CONF_USB_HOST_H_
```

10.3.2. conf_clock.h

10.3.2.1. AT32UC3A0, ATUC3A1, ATUC3Bﾃﾞﾊﾞｲｽ (USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== ｼｽﾃﾑ ｸﾛｯｸ元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RCSYS
#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_OSC0
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLL0

// ===== PLL0任意選択
//#define CONFIG_PLL0_SOURCE          PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL0_MUL             4 /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL0_DIV             1 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== PLL1任意選択
#define CONFIG_PLL1_SOURCE            PLL_SRC_OSC0
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC1
#define CONFIG_PLL1_MUL               8 /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL1_DIV               2 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== ｼｽﾃﾑ ｸﾛｯｸ ﾊﾞｽ分周任意選択
//#define CONFIG_SYSCLK_CPU_DIV       0 /* Fcpu = Fsys/(2 ^ CPU_div) */
//#define CONFIG_SYSCLK_PBA_DIV       0 /* Fpba = Fsys/(2 ^ PBA_div) */
//#define CONFIG_SYSCLK_PBB_DIV       0 /* Fpbb = Fsys/(2 ^ PBB_div) */

// ===== 周辺機能ｸﾛｯｸ管理任意選択
//#define CONFIG_SYSCLK_INIT_CPUMASK  ((1 << SYSCLK_SYSTIMER) | (1 << SYSCLK_OCD))
//#define CONFIG_SYSCLK_INIT_PBAMASK  (1 << SYSCLK_USART0)
//#define CONFIG_SYSCLK_INIT_PBBMASK  (1 << SYSCLK_HMATRIX)
//#define CONFIG_SYSCLK_INIT_HSBMASK  (1 << SYSCLK_MDMA_HSB)

// ===== USBｸﾛｯｸ元任意選択
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_OSC0
```

```
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL0
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV             1 /* Fusb = Fsys/(2 ^ USB_div) */

#endif /* CONF_CLOCK_H_INCLUDED */
```

10.3.2.2. AT32UC3A3, AT32UC3A4ﾃﾞﾊﾞｲｽ (高速(HS)支援付USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== システム クロック元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RCSYS
#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_OSC0
//#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_PLL0

// ===== PLL0任意選択
#define CONFIG_PLL0_SOURCE            PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE          PLL_SRC_OSC1
#define CONFIG_PLL0_MUL               11 /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL0_DIV               2  /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== PLL1任意選択
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC0
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL1_MUL             8 /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL1_DIV             2 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== システム クロック バス分周任意選択
#define CONFIG_SYSCLK_CPU_DIV         0 /* Fcpu = Fsys/(2 ^ CPU_div) */
#define CONFIG_SYSCLK_PBA_DIV         0 /* Fpba = Fsys/(2 ^ PBA_div) */
//#define CONFIG_SYSCLK_PBB_DIV       0 /* Fpbb = Fsys/(2 ^ PBB_div) */

// ===== 周辺機能ｸﾛｯｸ管理任意選択
//#define CONFIG_SYSCLK_INIT_CPUMASK  ((1 << SYSCLK_SYSTIMER) | (1 << SYSCLK_OCD))
//#define CONFIG_SYSCLK_INIT_PBAMASK  (1 << SYSCLK_USART0)
//#define CONFIG_SYSCLK_INIT_PBBMASK  (1 << SYSCLK_HMATRIX)
//#define CONFIG_SYSCLK_INIT_HSBMASK  (1 << SYSCLK_MDMA_HSB)

// ===== USBｸﾛｯｸ元任意選択
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_OSC0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV             1 /* Fusb = Fsys/(2 ^ USB_div) */

#endif /* CONF_CLOCK_H_INCLUDED */
```

10.3.2.3. AT32UC3C, ATUCXXD, ATUCXXL3U, ATUCXXL4Uﾃﾞﾊﾞｲｽ (USBC)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== システム クロック元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RCSYS
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_OSC0
```

```
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_OSC1
#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_PLL0
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLL1
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RC8M

// ===== PLL0任意選択
#define CONFIG_PLL0_SOURCE            PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL0_SOURCE          PLL_SRC_RC8M
#define CONFIG_PLL0_MUL               3 /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL0_DIV               1 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== PLL1任意選択
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC0
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL1_SOURCE          PLL_SRC_RC8M
//#define CONFIG_PLL1_MUL             3 /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL1_DIV             1 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== システム クロック バス分周任意選択
//#define CONFIG_SYSCLK_CPU_DIV       0 /* Fcpu = Fsys/(2 ^ CPU_div) */
//#define CONFIG_SYSCLK_PBA_DIV       0 /* Fpba = Fsys/(2 ^ PBA_div) */
//#define CONFIG_SYSCLK_PBB_DIV       0 /* Fpbb = Fsys/(2 ^ PBB_div) */
//#define CONFIG_SYSCLK_PBC_DIV       0 /* Fpbc = Fsys/(2 ^ PBC_div) */

// ===== 周辺機能ｸﾛｯｸ管理任意選択
//#define CONFIG_SYSCLK_INIT_CPUMASK  ((1 << SYSCLK_SYSTIMER) | (1 << SYSCLK_OCD))
//#define CONFIG_SYSCLK_INIT_PBAMASK  (1 << SYSCLK_USART0)
//#define CONFIG_SYSCLK_INIT_PBBMASK  (1 << SYSCLK_HMATRIX)
//#define CONFIG_SYSCLK_INIT_HSBMASK  (1 << SYSCLK_MDMA_HSB)

// ===== USBｸﾛｯｸ元任意選択
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_OSC0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_OSC1
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_PLL0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV             1 /* Fusb = Fsys/(2 ^ USB_div) */

#endif /* CONF_CLOCK_H_INCLUDED */
```

10.3.2.4. SAM3X, SAM3Aﾃﾞﾊﾞｲｽ (UOTGHS:USB OTG 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

/* ===== システム クロック(MCK)元任意選択 */

//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_BYPASS
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_4M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_8M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_12M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_BYPASS
#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLLACK
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_UPLLCK
```

```
/* ===== システム クロック(MCK)前置分周器任意選択    (Fmck = Fsys / (SYSCLK_PRES)) */
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_1
#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_2
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_4
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_8
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_16
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_32
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_64
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_3

/* ===== PLL0 (A)任意選択    (Fpll = (Fclk * PLL_mul) / PLL_div)
   ここにmulとdivの実効値を使ってください。 */
#define CONFIG_PLL0_SOURCE          PLL_SRC_MAINCK_XTAL
#define CONFIG_PLL0_MUL             14
#define CONFIG_PLL0_DIV             1

/* ===== 480MHzで固定化されたUPLL (UTMI)ﾊｰﾄﾞｳｪｱ */

/* ===== USBクロック元任意選択    (Fusb = FpllX / USB_div)
   ここにdivの実効値を使ってください。 */
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL0
#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_UPLL
#define CONFIG_USBCLK_DIV           1

/*
===== 目的対象周波数 (システム クロック)
- XTAL周波数: 12MHz
- システム クロック元: PLLA
- システム クロック前置分周器: 2 (2分周)
- PLLA供給元: XTAL
- PLLA出力: XTAL * 14 / 1
- システム クロックは: 12 * 14 / 1 /2 = 84MHz
===== 目的対象周波数 (USBクロック)
- USBクロック元: UPLL
- USBクロック分周器: 1 (分周なし)
- UPLL周波数: 480MHz
- USBクロック: 480 / 1 = 480MHz
*/

#endif /* CONF_CLOCK_H_INCLUDED */
```

10.3.3. conf_clocks.h

10.3.3.1. SAM D21ﾃﾞﾊﾞｲｽ (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#include <clock.h>

#ifndef CONF_CLOCKS_H_INCLUDED
#  define CONF_CLOCKS_H_INCLUDED

/* システム クロック ﾊﾞｽ構成設定 */
#  define CONF_CLOCK_CPU_CLOCK_FAILURE_DETECT      false
#  define CONF_CLOCK_FLASH_WAIT_STATES             2
#  define CONF_CLOCK_CPU_DIVIDER                   SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBA_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBB_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBC_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1

/* SYSTEM_CLOCK_SOURCE_OSC8M構成設定 - 内部8MHz発振器 */
```

```
#  define CONF_CLOCK_OSC8M_PRESCALER               SYSTEM_OSC8M_DIV_1
#  define CONF_CLOCK_OSC8M_ON_DEMAND               true
#  define CONF_CLOCK_OSC8M_RUN_IN_STANDBY          true

/* SYSTEM_CLOCK_SOURCE_XOSC構成設定 - 外部ｸﾛｯｸ/発振器 */
#  define CONF_CLOCK_XOSC_ENABLE                   false
#  define CONF_CLOCK_XOSC_EXTERNAL_CRYSTAL         SYSTEM_CLOCK_EXTERNAL_CRYSTAL
#  define CONF_CLOCK_XOSC_EXTERNAL_FREQUENCY       12000000UL
#  define CONF_CLOCK_XOSC_STARTUP_TIME             SYSTEM_XOSC_STARTUP_32768
#  define CONF_CLOCK_XOSC_AUTO_GAIN_CONTROL        true
#  define CONF_CLOCK_XOSC_ON_DEMAND                true
#  define CONF_CLOCK_XOSC_RUN_IN_STANDBY           false

/* SYSTEM_CLOCK_SOURCE_XOSC32K構成設定 - 外部32kHzｸﾘｽﾀﾙ/ｸﾛｯｸ発振器 */
#  define CONF_CLOCK_XOSC32K_ENABLE                true
#  define CONF_CLOCK_XOSC32K_EXTERNAL_CRYSTAL      SYSTEM_CLOCK_EXTERNAL_CRYSTAL
#  define CONF_CLOCK_XOSC32K_STARTUP_TIME          SYSTEM_XOSC32K_STARTUP_65536
#  define CONF_CLOCK_XOSC32K_AUTO_AMPLITUDE_CONTROL  false
#  define CONF_CLOCK_XOSC32K_ENABLE_1KHZ_OUPUT     false
#  define CONF_CLOCK_XOSC32K_ENABLE_32KHZ_OUTPUT   true
#  define CONF_CLOCK_XOSC32K_ON_DEMAND             false
#  define CONF_CLOCK_XOSC32K_RUN_IN_STANDBY        true

/* SYSTEM_CLOCK_SOURCE_OSC32K構成設定 - 内部32kHz発振器 */
#  define CONF_CLOCK_OSC32K_ENABLE                 false
#  define CONF_CLOCK_OSC32K_STARTUP_TIME           SYSTEM_OSC32K_STARTUP_130
#  define CONF_CLOCK_OSC32K_ENABLE_1KHZ_OUTPUT     false
#  define CONF_CLOCK_OSC32K_ENABLE_32KHZ_OUTPUT    true
#  define CONF_CLOCK_OSC32K_ON_DEMAND              true
#  define CONF_CLOCK_OSC32K_RUN_IN_STANDBY         false

/* SYSTEM_CLOCK_SOURCE_DFLL構成設定 - ﾃﾞｼﾞﾀﾙ周波数固定化閉路 */
#  define CONF_CLOCK_DFLL_ENABLE                   true
#  define CONF_CLOCK_DFLL_LOOP_MODE                SYSTEM_CLOCK_DFLL_LOOP_MODE_CLOSED
#  define CONF_CLOCK_DFLL_ON_DEMAND                true

/* DFLL開路動作構成設定 */
#  define CONF_CLOCK_DFLL_COARSE_VALUE             (0x1f / 4)
#  define CONF_CLOCK_DFLL_FINE_VALUE               (0xff / 4)

/* DFLL閉路動作構成設定 */
#  define CONF_CLOCK_DFLL_SOURCE_GCLK_GENERATOR    GCLK_GENERATOR_1
#  define CONF_CLOCK_DFLL_MULTIPLY_FACTOR          (48000000/32768)
#  define CONF_CLOCK_DFLL_QUICK_LOCK               true
#  define CONF_CLOCK_DFLL_TRACK_AFTER_FINE_LOCK    true
#  define CONF_CLOCK_DFLL_KEEP_LOCK_ON_WAKEUP      true
#  define CONF_CLOCK_DFLL_ENABLE_CHILL_CYCLE       true
#  define CONF_CLOCK_DFLL_MAX_COARSE_STEP_SIZE     (0x1f / 8)
#  define CONF_CLOCK_DFLL_MAX_FINE_STEP_SIZE       (0xff / 8)

/* SYSTEM_CLOCK_SOURCE_DPLL構成設定 - ﾃﾞｼﾞﾀﾙ位相固定化閉路 */
#  define CONF_CLOCK_DPLL_ENABLE                   false
#  define CONF_CLOCK_DPLL_ON_DEMAND                false
#  define CONF_CLOCK_DPLL_RUN_IN_STANDBY           true
#  define CONF_CLOCK_DPLL_LOCK_BYPASS              false
#  define CONF_CLOCK_DPLL_WAKE_UP_FAST             false
#  define CONF_CLOCK_DPLL_LOW_POWER_ENABLE         true

#  define CONF_CLOCK_DPLL_LOCK_TIME                SYSTEM_CLOCK_SOURCE_DPLL_LOCK_TIME_DEFAULT
#  define CONF_CLOCK_DPLL_REFERENCE_CLOCK          SYSTEM_CLOCK_SOURCE_DPLL_REFERENCE_CLOCK_XOSC32K
#  define CONF_CLOCK_DPLL_FILTER                   SYSTEM_CLOCK_SOURCE_DPLL_FILTER_DEFAULT
```

```
#  define CONF_CLOCK_DPLL_REFERENCE_FREQUENCY     32768
#  define CONF_CLOCK_DPLL_REFERENCE_DIVIDER       1
#  define CONF_CLOCK_DPLL_OUTPUT_FREQUENCY        48000000

/* DPLL GCLK基準構成設定 */
# define CONF_CLOCK_DPLL_REFERENCE_GCLK_GENERATOR GCLK_GENERATOR_1
/* DPLL GCLK固定化計時器構成設定 */
# define CONF_CLOCK_DPLL_LOCK_GCLK_GENERATOR GCLK_GENERATOR_1

/* clocks_init走行時にGCLKを構成設定するためにこれをtrueに設定してください。
 * falseに設定した場合、GCLK発振器はclocks_init()で何も構成設定されません。*/
#  define CONF_CLOCK_CONFIGURE_GCLK               true

/* GCLK発振器0構成設定（主ｸﾛｯｸ） */
#  define CONF_CLOCK_GCLK_0_ENABLE                true
#  define CONF_CLOCK_GCLK_0_RUN_IN_STANDBY        true
#  define CONF_CLOCK_GCLK_0_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_DFLL
#  define CONF_CLOCK_GCLK_0_PRESCALER             1
#  define CONF_CLOCK_GCLK_0_OUTPUT_ENABLE         false

/* GCLK発振器1構成設定 */
#  define CONF_CLOCK_GCLK_1_ENABLE                true
#  define CONF_CLOCK_GCLK_1_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_1_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_XOSC32K
#  define CONF_CLOCK_GCLK_1_PRESCALER             1
#  define CONF_CLOCK_GCLK_1_OUTPUT_ENABLE         false

/* GCLK発振器2構成設定（RTC） */
#  define CONF_CLOCK_GCLK_2_ENABLE                false
#  define CONF_CLOCK_GCLK_2_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_2_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC32K
#  define CONF_CLOCK_GCLK_2_PRESCALER             32
#  define CONF_CLOCK_GCLK_2_OUTPUT_ENABLE         false

/* GCLK発振器3構成設定 */
#  define CONF_CLOCK_GCLK_3_ENABLE                false
#  define CONF_CLOCK_GCLK_3_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_3_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_3_PRESCALER             1
#  define CONF_CLOCK_GCLK_3_OUTPUT_ENABLE         false

/* GCLK発振器4構成設定 */
#  define CONF_CLOCK_GCLK_4_ENABLE                false
#  define CONF_CLOCK_GCLK_4_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_4_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_4_PRESCALER             1
#  define CONF_CLOCK_GCLK_4_OUTPUT_ENABLE         false

/* GCLK発振器5構成設定 */
#  define CONF_CLOCK_GCLK_5_ENABLE                false
#  define CONF_CLOCK_GCLK_5_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_5_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_5_PRESCALER             1
#  define CONF_CLOCK_GCLK_5_OUTPUT_ENABLE         false

/* GCLK発振器6構成設定 */
#  define CONF_CLOCK_GCLK_6_ENABLE                false
#  define CONF_CLOCK_GCLK_6_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_6_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_6_PRESCALER             1
```

```
#  define CONF_CLOCK_GCLK_6_OUTPUT_ENABLE         false

/* GCLK発振器7構成設定 */
#  define CONF_CLOCK_GCLK_7_ENABLE                false
#  define CONF_CLOCK_GCLK_7_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_7_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_7_PRESCALER             1
#  define CONF_CLOCK_GCLK_7_OUTPUT_ENABLE         false

/* GCLK発振器8構成設定 */
#  define CONF_CLOCK_GCLK_8_ENABLE                false
#  define CONF_CLOCK_GCLK_8_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_8_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_8_PRESCALER             1
#  define CONF_CLOCK_GCLK_8_OUTPUT_ENABLE         false

#endif /* CONF_CLOCKS_H_INCLUDED */
```

10.3.4. conf_board.h

10.3.4.1. AT32UC3A0, AT32UC3A1, AT32UC3Bﾃﾞﾊﾞｲｽ (USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// この例については既定基板初期化(ｽｲｯﾁ/LED)だけが必要です。

#endif /* CONF_BOARD_H_INCLUDED */
```

10.3.4.2. AT32UC3A3, AT32UC3A4ﾃﾞﾊﾞｲｽ (高速(HS)支援付USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// この例については既定基板初期化(ｽｲｯﾁ/LED)だけが必要です。

#endif /* CONF_BOARD_H_INCLUDED */
```

10.3.4.3. AT32UC3C, ATUCXXD, ATUCXXL3U, ATUCXXL4Uﾃﾞﾊﾞｲｽ (USBC)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// この例については既定基板初期化(ｽｲｯﾁ/LED)だけが必要です。

#endif /* CONF_BOARD_H_INCLUDED */
```

10.3.4.4. SAM3X, SAM3Aﾃﾞﾊﾞｲｽ (UOTGHS:USB OTG 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */
```

```
#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// USBﾋﾟﾝが使われます。
#define CONF_BOARD_USB_PORT

#endif /* CONF_BOARD_H_INCLUDED */
```

10.3.4.5. SAM D21ﾃﾞﾊﾞｲｽ (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

/* USB VBUS検出許可 */
#define CONF_BOARD_USB_VBUS_DETECT
/* ID検出許可 */
#define CONF_BOARD_USB_ID_DETECT

#endif /* CONF_BOARD_H_INCLUDED */
```

11. 大容量記憶装置ｸﾗｽ(MSC)用USBﾎｽﾄ ｲﾝﾀｰﾌｪｰｽ(UHI)

大容量記憶装置ｸﾗｽ(MSC)用USBﾎｽﾄ ｲﾝﾀｰﾌｪｰｽ(UHI)はUSB MSCﾎｽﾄの構成設定と管理に関するｲﾝﾀｰﾌｪｰｽを提供します。
この資料の概要は以下のとおりです。

- API概要
- USBﾎｽﾄ大容量記憶装置単位部(UHI MSC)用の即時開始の手引き
- 構成設定ﾌｧｲﾙ例

Atmel®ｿﾌﾄｳｪｱ枠組み(ASF) USB装置階層に関するより多くの詳細については以下の応用記述を参照してください。

- AVR4950：ASF – USBﾎｽﾄ階層

11.1. API概要

11.1.1. 変数と型定義

11.1.1.1. uhi_msc_scsi_callback_t型

```
typedef void(* uhi_msc_scsi_callback_t )(bool)
```

uhi_msc_scsi()関数によって使われる呼び戻し型

11.1.2. 構造体定義

11.1.2.1. uhi_msc_lun_t構造体

LUN構造体情報

表11-1. ﾒﾝﾊﾞ

型	名前	説明
bool	b_write_protected	書き込み保護許可
struct sbc_read_capacity10_data	capacity	SBC-2容量読み込み(10)ﾊﾟﾗﾒｰﾀ ﾃﾞｰﾀ
lun_status_t	status	LUNの状態

11.1.3. ﾏｸﾛ定義

11.1.3.1. USBﾎｽﾄ ｺｱ(UHC)とのｲﾝﾀｰﾌｪｰｽ

UHCによって必要とされる定義と関数

UHI_MSC ﾏｸﾛ

```
#define UHI_MSC
```

UHC用の標準UHI APIを含む全域定義。これはconf_usb_host.hﾌｧｲﾙからUSB_HOST_UHI定義で追加されなければなりません。

11.1.4. 関数定義

11.1.4.1. UHCによって必要とされる関数

uhi_msc_install()関数

ｲﾝﾀｰﾌｪｰｽ取り付け

```
uhc_enum_status_t uhi_msc_install( uhc_device_t * dev )
```

支援されるなら、ｲﾝﾀｰﾌｪｰｽ ｴﾝﾄﾞﾎﾟｲﾝﾄを割り当て

表11-2. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
dev	[入力]	要求する装置

戻り値：取り付けの状態

uhi_msc_enable()関数

ｲﾝﾀｰﾌｪｰｽ許可

```
void uhi_msc_enable( uhc_device_t * dev )
```

UHIに対応するUSBｲﾝﾀｰﾌｪｰｽを許可

表11-3. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
dev	[入力]	要求する装置

uhi_msc_uninstall()関数

(取り付けられていれば)インターフェースを取り外し

```
void uhi_msc_uninstall( uhc_device_t * dev )
```

表11-4. パラメータ

パラメータ名	データ方向	説明
dev	[入力]	要求する装置

11.1.4.2. 大容量記憶装置クラス用UHI

USBホスト クラスを使う上位応用によって使われる共通的なAPI

uhi_msc_is_available()関数

UHI大容量記憶装置インターフェースが利用可能か調査

```
bool uhi_msc_is_available( void )
```

USB装置MSCの列挙(接続認証)中、UHI大容量記憶装置は多忙になり得ます。

戻り値：UHI大容量記憶装置が利用可能ならば真(true)

uhi_msc_get_lun()関数

利用可能なLUNの番号を与えます。

```
uint8_t uhi_msc_get_lun( void )
```

注：LUNが利用可能なのにLUN_NOT_PRESET状態であることが有り得ます。

これはカードなしのカード読み取り器の場合です。

戻り値：利用可能なLUNの番号

uhi_msc_get_lun_desc()関数

LUNについての情報を与えます。

```
uhi_msc_lun_t * uhi_msc_get_lun_desc( uint8_t lun )
```

表11-5. パラメータ

パラメータ名	データ方向	説明
lun	[入力]	LUN番号

戻り値：LUN情報構造体上のポインタ

uhi_msc_scsi_test_unit_ready()関数

LUNの状態を調査して更新

```
bool uhi_msc_scsi_test_unit_ready( uint8_t lun, uhi_msc_scsi_callback_t callback )
```

表11-6. パラメータ

パラメータ名	データ方向	説明
lun	[入力]	LUN番号
callback	[入力]	SCSI命令の最後で呼ぶための呼び戻し

戻り値：小コンピュータ系インターフェース(SCSI:Small Computer System Interface)命令命令が受け入れられた場合に真(true)

uhi_msc_scsi_read_10()関数

LUNデータ部をRAM緩衝部に読みます。

```
bool uhi_msc_scsi_read_10( uint8_t lun, uint32_t addr, uint8_t * ram, uint8_t nb_sector,
  uhi_msc_scsi_callback_t callback)
```

注：データ部を定義するのに使われるセクタ容量は領域に於いてLUNによって返されるセクタ容量です。

表11-7. パラメータ

パラメータ名	データ方向	説明
lun	[入力]	LUN番号
addr	[入力]	読むセクタのアドレス
ram	[出力]	データを格納するのに使われるRAMアドレス
nb_sector	[入力]	読むセクタ番号
callback	[入力]	SCSI命令の最後で呼ぶための呼び戻し

戻り値：SCSI命令が受け入れられた場合に真(true)

uhi_msc_scsi_write_10()関数

RAM緩衝部をLUNﾃﾞｰﾀ部に書きます。

```
bool uhi_msc_scsi_write_10( uint8_t lun, uint32_t addr, const uint8_t * ram, uint8_t nb_sector,
  uhi_msc_scsi_callback_t callback)
```

注：ﾃﾞｰﾀ部を定義するのに使われるｾｸﾀ容量は領域に於いてLUNによって返されるｾｸﾀ容量です。

表11-8. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
lun	[入力]	LUN番号
addr	[入力]	書くｾｸﾀのｱﾄﾞﾚｽ
ram	[出力]	書くﾃﾞｰﾀのRAMｱﾄﾞﾚｽ
nb_sector	[入力]	書くｾｸﾀ番号
callback	[入力]	SCSI命令の最後で呼ぶための呼び戻し

戻り値：SCSI命令が受け入れられた場合に真(true)

11.1.4.3. 制御ｱｸｾｽ単位部用USBﾎｽﾄ大容量記憶装置ｲﾝﾀｰﾌｪｰｽ

制御ｱｸｾｽ単位部の使用を許すためにUHI MSCｲﾝﾀｰﾌｪｰｽに追加される階層

uhi_msc_mem_get_lun()関数

利用可能なLUN番号を与えます。

```
uint8_t uhi_msc_mem_get_lun( void )
```

注：LUNが利用可能なのに不在状態であることが有り得ます。
これはｶｰﾄﾞなしのｶｰﾄﾞ読み取り器の場合です。

戻り値：利用可能なLUNの番号

uhi_msc_mem_test_unit_ready()関数

LUNの状態を調査して更新

```
Ctrl_status uhi_msc_mem_test_unit_ready( uint8_t lun )
```

表11-9. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
lun	[入力]	LUN番号

戻り値：LUNの状態

uhi_msc_mem_read_capacity()関数

LUNの容量を返します。

```
Ctrl_status uhi_msc_mem_read_capacity( uint8_t lun, uint32_t * u32_nb_sector )
```

表11-10. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
lun	[入力]	LUN番号
u32_nb_sector	[入力]	このLUNで可能な最後のｾｸﾀ ｱﾄﾞﾚｽを格納するためのﾎﾟｲﾝﾀ

戻り値：LUNの状態

uhi_msc_mem_read_sector_size()関数

LUNのｾｸﾀ容量を返します。

```
uint8_t uhi_msc_mem_read_sector_size( uint8_t lun )
```

表11-11. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
lun	[入力]	LUN番号

戻り値：(512ﾊﾞｲﾄ単位の)ｾｸﾀ容量

uhi_msc_mem_wr_protect()関数

LUNが書き込み保護が調査

```
bool uhi_msc_mem_wr_protect( uint8_t lun )
```

表11-12. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
lun	[入力]	LUN番号

戻り値：書き込み保護の場合に真(true)

uhi_msc_mem_removal()関数

装置が取り外されているか調査

```
bool uhi_msc_mem_removal( void )
```

戻り値：利用可能なLUNの番号

uhi_msc_mem_read_10_ram()関数

現在のLUNから512ﾊﾞｲﾄを読みます。

```
Ctrl_status uhi_msc_mem_read_10_ram( uint32_t addr, void * ram )
```

LUNはuhi_msc_mem_test_unit_ready()関数またはuhi_msc_mem_read_capacity()関数によって選択されます。

表11-13. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
addr	[入力]	(512ﾊﾞｲﾄ単位の)ﾃﾞｨｽｸ ｱﾄﾞﾚｽ
ram	[出力]	ﾃﾞｰﾀを格納するためのﾎﾟｲﾝﾀ

戻り値：LUNの状態

uhi_msc_mem_write_10_ram()関数

現在のLUNに512ﾊﾞｲﾄを書きます。

```
Ctrl_status uhi_msc_mem_write_10_ram( uint32_t addr, const void * ram )
```

LUNはuhi_msc_mem_test_unit_ready()関数またはuhi_msc_mem_read_capacity()関数によって選択されます。

表11-14. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
addr	[入力]	(512ﾊﾞｲﾄ単位の)ﾃﾞｨｽｸ ｱﾄﾞﾚｽ
ram	[入力]	ﾃﾞｰﾀのﾎﾟｲﾝﾀ

戻り値：LUNの状態

11.1.5. 列挙定義

11.1.5.1. lun_status_t列挙

LUNの状態

表11-15. ﾒﾝﾊﾞ

列挙値	説明
LUN_GOOD	成功、ﾒﾓﾘ準備可
LUN_FAIL	異常発生
LUN_NOT_PRESENT	ﾒﾓﾘ切断
LUN_BUSY	ﾒﾓﾘ未初期化または無変化

11.2. USBホスト大容量記憶装置単位部(UHI MSC)用の即時開始の手引き

これは使用事例の選択に於いて単位部をどう構成設定して使うかを段階的に指示する、USBホスト大容量記憶装置単位部(UHI MSC)用の即時開始の手引きです。

使用事例は様々なｺｰﾄﾞの断片を強調します。構成設定に関する段階でのｺｰﾄﾞの断片は独自初期化関数内に複写することができ、一方使用に関する段階は例えば主応用関数内に複写することができます。

11.2.1. 基本的な使用事例

この基本的な使用事例では"USB Host MSC (Single Class support)"単位部が使われます。

"USB Host MSC (Multiple Classes support)"単位部の使い方は高度な使用事例で記述されます。

この例は簡単な物理ﾒﾓﾘ ｱｸｾｽを行いますが、USBﾒﾓﾘ ﾌｧｲﾙ ｼｽﾃﾑを復号するためにﾌｧｲﾙ ｼｽﾃﾑ単位部を追加することができます。FatFS例をご覧ください。

11.2.1.1. 構成設定段階

USBホストのため、共通USBホスト構成設定段階に従います。USBホスト基本構成設定を参照してください。

11.2.1.2. 使用段階

ｺｰﾄﾞ例

conf_usb_host.hの内容

```
#define USB_HOST_UHI         UHI_MSC
#define UHI_MSC_CHANGE(dev, b_plug) my_callback_msc_change(dev, b_plug)
extern bool my_callback_msc_change(uhc_device_t* dev, bool b_plug);
#include "uhi_msc_mem.h" // conf_usb_host.hﾌｧｲﾙの最後で
```

応用Cﾌｧｲﾙに追加してください。

```
static bool my_flag_autorize_msc_check = false;
bool my_callback_msc_change(uhc_device_t* dev, bool b_plug)
{
   if (b_plug) {
      my_flag_autorize_msc_check = true;
   } else {
      my_flag_autorize_msc_check = false;
   }
}

void my_task(void)
{
   if (!my_flag_autorize_msc_check) {
      return;
   }
   my_flag_autorize_msc_check = false;

   // 接続された新しいUSBﾃﾞｨｽｸ全てを調査
   for (uint8_t lun=0; lun < uhi_msc_mem_get_lun(); lun++) {
      // USBﾃﾞｨｽｸ取り付け終了待機
      while (CTRL_BUSY == uhi_msc_mem_test_unit_ready(lun));
      if (CTRL_GOOD != uhi_msc_mem_test_unit_ready(lun)) {
         // 可搬型ﾃﾞｨｽｸ不在または失敗
         continue;
      }
      // 容量読み込み
      uint32_t max_lba;
      uhi_msc_mem_read_capacity(lun, &max_lba);
   }
}
```

作業の流れ

1. conf_usb_host.hが利用可能でUSBホストMSC構成設定である以下の構成設定を含むことを確実にしてください。

   ```
   #define USB_HOST_UHI    UHI_MSC
   ```

   注：これはUSBホストによって支援されるUHIの一覧を定義します。

   ```
   #define UHI_MSC_CHANGE(dev, b_plug) my_callback_msc_change(dev, b_plug)
   ```

```
extern bool my_callback_msc_change(uhc_device_t* dev, bool b_plug);
```

注：この呼び戻しはUSB装置MSCが接続または切断された時に呼ばれます。

2. USBメモリのアクセスはAPI概要で記述された関数を通して許されます。

11.2.2. 高度な使用事例

UHI MSC単位部のもっと高度な使用については以下をご覧ください。

・USBホストの高度な使用事例

11.3. 構成設定ファイル例

11.3.1. conf_usb_host.h

11.3.1.1. 単一UHI MSC

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_USB_HOST_H_
#define _CONF_USB_HOST_H_

#include "compiler.h"

#define USB_HOST_UHI         UHI_MSC

#define USB_HOST_POWER_MAX   500

// #define USB_HOST_HUB_SUPPORT

#if (UC3A3 || UC3A4)
#  define USB_HOST_HS_SUPPORT
#endif

//#define UHC_MODE_CHANGE(b_host_mode)         usb_host_mode_change(b_host_mode)

//#define UHC_VBUS_CHANGE(b_present)           usb_host_vbus_change(b_present)

//#define UHC_VBUS_ERROR()                     usb_host_vbus_error()

//#define UHC_CONNECTION_EVENT(dev,b_present) usb_host_connection_event(dev,b_present)

//#define UHC_WAKEUP_EVENT()                   usb_host_wakeup_event()

//#define UHC_SOF_EVENT()                      usb_host_sof_event()

//#define UHC_DEVICE_CONF(dev)                 uint8_t usb_host_device_conf(dev)

//#define UHC_ENUM_EVENT(dev,b_status)         usb_host_enum_event(dev,b_status)

#define UHI_MSC_CHANGE(dev,b_plug)

#include "uhi_msc.h"

#endif // _CONF_USB_HOST_H_
```

11.3.1.2. 複数UHI MSC（複合）

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_USB_HOST_H_
#define _CONF_USB_HOST_H_
```

```
#include "compiler.h"

#define USB_HOST_UHI         // UHI_MSC, UHI_HID_MOUSE, UHI_CDC, UHI_VENDOR

#define USB_HOST_POWER_MAX  500

// #define USB_HOST_HUB_SUPPORT

#if (UC3A3 || UC3A4)
#  define USB_HOST_HS_SUPPORT
#endif

//#define UHC_MODE_CHANGE(b_host_mode)         usb_host_mode_change(b_host_mode)

//#define UHC_VBUS_CHANGE(b_present)           usb_host_vbus_change(b_present)

//#define UHC_VBUS_ERROR()                     usb_host_vbus_error()

//#define UHC_CONNECTION_EVENT(dev,b_present) usb_host_connection_event(dev,b_present)

//#define UHC_WAKEUP_EVENT()                   usb_host_wakeup_event()

//#define UHC_SOF_EVENT()                      usb_host_sof_event()

//#define UHC_DEVICE_CONF(dev)                 uint8_t usb_host_device_conf(dev)

//#define UHC_ENUM_EVENT(dev,b_status)         usb_host_enum_event(dev,b_status)

#define UHI_HID_MOUSE_CHANGE(dev,b_plug)
#define UHI_HID_MOUSE_EVENT_BTN_LEFT(b_state)
#define UHI_HID_MOUSE_EVENT_BTN_RIGHT(b_state)
#define UHI_HID_MOUSE_EVENT_BTN_MIDDLE(b_state)
#define UHI_HID_MOUSE_EVENT_MOUVE(x,y,scroll)

#define UHI_MSC_CHANGE(dev,b_plug)

#define UHI_CDC_CHANGE(dev,b_plug)
#define UHI_CDC_RX_NOTIFY()

#define UHI_VENDOR_CHANGE(dev, b_plug)
#define UHI_VENDOR_VID_PID_LIST {USB_VID_ATMEL, USB_PID_ATMEL_ASF_VENDOR_CLASS}

//#include "uhi_msc.h"
//#include "uhi_hid_mouse.h"

#endif // _CONF_USB_HOST_H_
```

11.3.2. conf_clock.h

11.3.2.1. AT32UC3A0, ATUC3A1, ATUC3Bﾃﾞﾊﾞｲｽ (USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== ｼｽﾃﾑ ｸﾛｯｸ元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RCSYS
#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_OSC0
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLL0
```

```
// ===== PLL0任意選択
//#define CONFIG_PLL0_SOURCE          PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL0_MUL             4 /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL0_DIV             1 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== PLL1任意選択
#define CONFIG_PLL1_SOURCE            PLL_SRC_OSC0
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC1
#define CONFIG_PLL1_MUL               8 /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL1_DIV               2 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== システム クロック バス分周任意選択
//#define CONFIG_SYSCLK_CPU_DIV       0 /* Fcpu = Fsys/(2 ^ CPU_div) */
//#define CONFIG_SYSCLK_PBA_DIV       0 /* Fpba = Fsys/(2 ^ PBA_div) */
//#define CONFIG_SYSCLK_PBB_DIV       0 /* Fpbb = Fsys/(2 ^ PBB_div) */

// ===== 周辺機能ｸﾛｯｸ管理任意選択
//#define CONFIG_SYSCLK_INIT_CPUMASK  ((1 << SYSCLK_SYSTIMER) | (1 << SYSCLK_OCD))
//#define CONFIG_SYSCLK_INIT_PBAMASK  (1 << SYSCLK_USART0)
//#define CONFIG_SYSCLK_INIT_PBBMASK  (1 << SYSCLK_HMATRIX)
//#define CONFIG_SYSCLK_INIT_HSBMASK  (1 << SYSCLK_MDMA_HSB)

// ===== USBｸﾛｯｸ元任意選択
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_OSC0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL0
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV             1 /* Fusb = Fsys/(2 ^ USB_div) */

#endif /* CONF_CLOCK_H_INCLUDED */
```

11.3.2.2. AT32UC3A3, AT32UC3A4ﾃﾞﾊﾞｲｽ (高速(HS)支援付USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== システム クロック元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RCSYS
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_OSC0
#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_PLL0

// ===== PLL0任意選択
#define CONFIG_PLL0_SOURCE            PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE          PLL_SRC_OSC1
#define CONFIG_PLL0_MUL               11 /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL0_DIV               2  /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== PLL1任意選択
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC0
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL1_MUL             8 /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL1_DIV             2 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== システム クロック バス分周任意選択
#define CONFIG_SYSCLK_CPU_DIV         0 /* Fcpu = Fsys/(2 ^ CPU_div) */
#define CONFIG_SYSCLK_PBA_DIV         0 /* Fpba = Fsys/(2 ^ PBA_div) */
//#define CONFIG_SYSCLK_PBB_DIV       0 /* Fpbb = Fsys/(2 ^ PBB_div) */
```

```
// ===== 周辺機能ｸﾛｯｸ管理任意選択
//#define CONFIG_SYSCLK_INIT_CPUMASK   ((1 << SYSCLK_SYSTIMER) | (1 << SYSCLK_OCD))
//#define CONFIG_SYSCLK_INIT_PBAMASK   (1 << SYSCLK_USART0)
//#define CONFIG_SYSCLK_INIT_PBBMASK   (1 << SYSCLK_HMATRIX)
//#define CONFIG_SYSCLK_INIT_HSBMASK   (1 << SYSCLK_MDMA_HSB)

// ===== USBｸﾛｯｸ元任意選択
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_OSC0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV             1 /* Fusb = Fsys/(2 ^ USB_div) */

#endif /* CONF_CLOCK_H_INCLUDED */
```

11.3.2.3. AT32UC3C, ATUCXXD, ATUCXXL3U, ATUCXXL4Uﾃﾞﾊﾞｲｽ (USBC)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

// ===== ｼｽﾃﾑ ｸﾛｯｸ元任意選択
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RCSYS
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_OSC0
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_OSC1
#define CONFIG_SYSCLK_SOURCE          SYSCLK_SRC_PLL0
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLL1
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RC8M

// ===== PLL0任意選択
#define CONFIG_PLL0_SOURCE            PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL0_SOURCE          PLL_SRC_RC8M
#define CONFIG_PLL0_MUL               3 /* Fpll = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL0_DIV               1 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== PLL1任意選択
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC0
//#define CONFIG_PLL1_SOURCE          PLL_SRC_OSC1
//#define CONFIG_PLL1_SOURCE          PLL_SRC_RC8M
//#define CONFIG_PLL1_MUL             3 /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL1_DIV             1 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ===== ｼｽﾃﾑ ｸﾛｯｸ ﾊﾞｽ分周任意選択
//#define CONFIG_SYSCLK_CPU_DIV       0 /* Fcpu = Fsys/(2 ^ CPU_div) */
//#define CONFIG_SYSCLK_PBA_DIV       0 /* Fpba = Fsys/(2 ^ PBA_div) */
//#define CONFIG_SYSCLK_PBB_DIV       0 /* Fpbb = Fsys/(2 ^ PBB_div) */
//#define CONFIG_SYSCLK_PBC_DIV       0 /* Fpbc = Fsys/(2 ^ PBC_div) */

// ===== 周辺機能ｸﾛｯｸ管理任意選択
//#define CONFIG_SYSCLK_INIT_CPUMASK   ((1 << SYSCLK_SYSTIMER) | (1 << SYSCLK_OCD))
//#define CONFIG_SYSCLK_INIT_PBAMASK   (1 << SYSCLK_USART0)
//#define CONFIG_SYSCLK_INIT_PBBMASK   (1 << SYSCLK_HMATRIX)
//#define CONFIG_SYSCLK_INIT_HSBMASK   (1 << SYSCLK_MDMA_HSB)

// ===== USBｸﾛｯｸ元任意選択
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_OSC0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_OSC1
#define CONFIG_USBCLK_SOURCE          USBCLK_SRC_PLL0
```

```
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL1
#define CONFIG_USBCLK_DIV             1 /* Fusb = Fsys/(2 ^ USB_div) */

#endif /* CONF_CLOCK_H_INCLUDED */
```

11.3.2.4. SAM3X, SAM3Aﾃﾞﾊﾞｲｽ (UOTGHS:USB OTG 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

/* ===== ｼｽﾃﾑ ｸﾛｯｸ(MCK)元任意選択 */
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_BYPASS
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_4M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_8M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_12M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_BYPASS
#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLLACK
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_UPLLCK

/* ===== ｼｽﾃﾑ ｸﾛｯｸ(MCK)前置分周器任意選択    (Fmck = Fsys / (SYSCLK_PRES)) */
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_1
#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_2
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_4
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_8
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_16
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_32
/#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_64
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_3

/* ===== PLL0 (A)任意選択    (Fpll = (Fclk * PLL_mul) / PLL_div)
   ここにmulとdivの実効値を使ってください。 */
#define CONFIG_PLL0_SOURCE          PLL_SRC_MAINCK_XTAL
#define CONFIG_PLL0_MUL             14
#define CONFIG_PLL0_DIV             1

/* ===== 480MHzで固定化されたUPLL (UTMI)ﾊｰﾄﾞｳｪｱ */

/* ===== USBｸﾛｯｸ元任意選択    (Fusb = FpllX / USB_div)
   ここにdivの実効値を使ってください。 */
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL0
#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_UPLL
#define CONFIG_USBCLK_DIV           1

/*
===== 目的対象周波数 (ｼｽﾃﾑ ｸﾛｯｸ)
- XTAL周波数: 12MHz
- ｼｽﾃﾑ ｸﾛｯｸ元: PLLA
- ｼｽﾃﾑ ｸﾛｯｸ前置分周器: 2 (2分周)
- PLLA供給元: XTAL
- PLLA出力: XTAL * 14 / 1
- ｼｽﾃﾑ ｸﾛｯｸは: 12 * 14 / 1 /2 = 84MHz
===== 目的対象周波数 (USBｸﾛｯｸ)
- USBｸﾛｯｸ元: UPLL
- USBｸﾛｯｸ分周器: 1 (分周なし)
- UPLL周波数: 480MHz
```

```
- USBクロック: 480 / 1 = 480MHz
*/

#endif /* CONF_CLOCK_H_INCLUDED */
```

11.3.3. conf_clocks.h

11.3.3.1. SAM D21ﾃﾞﾊﾞｲｽ (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#include <clock.h>

#ifndef CONF_CLOCKS_H_INCLUDED
#  define CONF_CLOCKS_H_INCLUDED

/* システム クロック ﾊﾞｽ構成設定 */
#  define CONF_CLOCK_CPU_CLOCK_FAILURE_DETECT      false
#  define CONF_CLOCK_FLASH_WAIT_STATES             2
#  define CONF_CLOCK_CPU_DIVIDER                   SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBA_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBB_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBC_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1

/* SYSTEM_CLOCK_SOURCE_OSC8M構成設定 - 内部8MHz発振器 */
#  define CONF_CLOCK_OSC8M_PRESCALER               SYSTEM_OSC8M_DIV_1
#  define CONF_CLOCK_OSC8M_ON_DEMAND               true
#  define CONF_CLOCK_OSC8M_RUN_IN_STANDBY          true

/* SYSTEM_CLOCK_SOURCE_XOSC構成設定 - 外部ｸﾛｯｸ/発振器 */
#  define CONF_CLOCK_XOSC_ENABLE                   false
#  define CONF_CLOCK_XOSC_EXTERNAL_CRYSTAL         SYSTEM_CLOCK_EXTERNAL_CRYSTAL
#  define CONF_CLOCK_XOSC_EXTERNAL_FREQUENCY       12000000UL
#  define CONF_CLOCK_XOSC_STARTUP_TIME             SYSTEM_XOSC_STARTUP_32768
#  define CONF_CLOCK_XOSC_AUTO_GAIN_CONTROL        true
#  define CONF_CLOCK_XOSC_ON_DEMAND                true
#  define CONF_CLOCK_XOSC_RUN_IN_STANDBY           false

/* SYSTEM_CLOCK_SOURCE_XOSC32K構成設定 - 外部32kHzｸﾘｽﾀﾙ/ｸﾛｯｸ発振器 */
#  define CONF_CLOCK_XOSC32K_ENABLE                true
#  define CONF_CLOCK_XOSC32K_EXTERNAL_CRYSTAL      SYSTEM_CLOCK_EXTERNAL_CRYSTAL
#  define CONF_CLOCK_XOSC32K_STARTUP_TIME          SYSTEM_XOSC32K_STARTUP_65536
#  define CONF_CLOCK_XOSC32K_AUTO_AMPLITUDE_CONTROL  false
#  define CONF_CLOCK_XOSC32K_ENABLE_1KHZ_OUPUT     false
#  define CONF_CLOCK_XOSC32K_ENABLE_32KHZ_OUTPUT   true
#  define CONF_CLOCK_XOSC32K_ON_DEMAND             false
#  define CONF_CLOCK_XOSC32K_RUN_IN_STANDBY        true

/* SYSTEM_CLOCK_SOURCE_OSC32K構成設定 - 内部32kHz発振器 */
#  define CONF_CLOCK_OSC32K_ENABLE                 false
#  define CONF_CLOCK_OSC32K_STARTUP_TIME           SYSTEM_OSC32K_STARTUP_130
#  define CONF_CLOCK_OSC32K_ENABLE_1KHZ_OUTPUT     false
#  define CONF_CLOCK_OSC32K_ENABLE_32KHZ_OUTPUT    true
#  define CONF_CLOCK_OSC32K_ON_DEMAND              true
#  define CONF_CLOCK_OSC32K_RUN_IN_STANDBY         false

/* SYSTEM_CLOCK_SOURCE_DFLL構成設定 - ﾃﾞｼﾞﾀﾙ周波数固定化閉路 */
#  define CONF_CLOCK_DFLL_ENABLE                   true
#  define CONF_CLOCK_DFLL_LOOP_MODE                SYSTEM_CLOCK_DFLL_LOOP_MODE_CLOSED
#  define CONF_CLOCK_DFLL_ON_DEMAND                true
```

```
/* DFLL開路動作構成設定 */
#  define CONF_CLOCK_DFLL_FINE_VALUE               (512)

/* DFLL閉路動作構成設定 */
#  define CONF_CLOCK_DFLL_SOURCE_GCLK_GENERATOR    GCLK_GENERATOR_1
#  define CONF_CLOCK_DFLL_MULTIPLY_FACTOR          (48000000/32768)
#  define CONF_CLOCK_DFLL_QUICK_LOCK               true
#  define CONF_CLOCK_DFLL_TRACK_AFTER_FINE_LOCK    true
#  define CONF_CLOCK_DFLL_KEEP_LOCK_ON_WAKEUP      true
#  define CONF_CLOCK_DFLL_ENABLE_CHILL_CYCLE       true
#  define CONF_CLOCK_DFLL_MAX_COARSE_STEP_SIZE     (0x1f / 8)
#  define CONF_CLOCK_DFLL_MAX_FINE_STEP_SIZE       (0xff / 8)

/* SYSTEM_CLOCK_SOURCE_DPLL構成設定 - ﾃﾞｼﾞﾀﾙ位相固定化閉路 */
#  define CONF_CLOCK_DPLL_ENABLE                   false
#  define CONF_CLOCK_DPLL_ON_DEMAND                false
#  define CONF_CLOCK_DPLL_RUN_IN_STANDBY           true
#  define CONF_CLOCK_DPLL_LOCK_BYPASS              false
#  define CONF_CLOCK_DPLL_WAKE_UP_FAST             false
#  define CONF_CLOCK_DPLL_LOW_POWER_ENABLE         true

#  define CONF_CLOCK_DPLL_LOCK_TIME                SYSTEM_CLOCK_SOURCE_DPLL_LOCK_TIME_DEFAULT
#  define CONF_CLOCK_DPLL_REFERENCE_CLOCK          SYSTEM_CLOCK_SOURCE_DPLL_REFERENCE_CLOCK_XOSC32K
#  define CONF_CLOCK_DPLL_FILTER                   SYSTEM_CLOCK_SOURCE_DPLL_FILTER_DEFAULT

#  define CONF_CLOCK_DPLL_REFERENCE_FREQUENCY      32768
#  define CONF_CLOCK_DPLL_REFERENCE_DIVIDER        1
#  define CONF_CLOCK_DPLL_OUTPUT_FREQUENCY         48000000

/* DPLL GCLK基準構成設定 */
# define CONF_CLOCK_DPLL_REFERENCE_GCLK_GENERATOR GCLK_GENERATOR_1
/* DPLL GCLK固定化計時器構成設定 */
# define CONF_CLOCK_DPLL_LOCK_GCLK_GENERATOR GCLK_GENERATOR_1

/* clocks_init走行時にGCLKを構成設定するためにこれをtrueに設定してください。
 * falseに設定した場合、GCLK発振器はclocks_init()で何も構成設定されません。*/
#  define CONF_CLOCK_CONFIGURE_GCLK                true

/* GCLK発振器0構成設定 (主ｸﾛｯｸ) */
#  define CONF_CLOCK_GCLK_0_ENABLE                 true
#  define CONF_CLOCK_GCLK_0_RUN_IN_STANDBY         true
#  define CONF_CLOCK_GCLK_0_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_DFLL
#  define CONF_CLOCK_GCLK_0_PRESCALER              1
#  define CONF_CLOCK_GCLK_0_OUTPUT_ENABLE          false

/* GCLK発振器1構成設定 */
#  define CONF_CLOCK_GCLK_1_ENABLE                 true
#  define CONF_CLOCK_GCLK_1_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_1_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_XOSC32K
#  define CONF_CLOCK_GCLK_1_PRESCALER              1
#  define CONF_CLOCK_GCLK_1_OUTPUT_ENABLE          false

/* GCLK発振器2構成設定 (RTC) */
#  define CONF_CLOCK_GCLK_2_ENABLE                 false
#  define CONF_CLOCK_GCLK_2_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_2_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC32K
#  define CONF_CLOCK_GCLK_2_PRESCALER              32
#  define CONF_CLOCK_GCLK_2_OUTPUT_ENABLE          false

/* GCLK発振器3構成設定 */
#  define CONF_CLOCK_GCLK_3_ENABLE                 false
```

```
#  define CONF_CLOCK_GCLK_3_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_3_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_3_PRESCALER             1
#  define CONF_CLOCK_GCLK_3_OUTPUT_ENABLE         false

/* GCLK発振器4構成設定 */
#  define CONF_CLOCK_GCLK_4_ENABLE                false
#  define CONF_CLOCK_GCLK_4_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_4_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_4_PRESCALER             1
#  define CONF_CLOCK_GCLK_4_OUTPUT_ENABLE         false

/* GCLK発振器5構成設定 */
#  define CONF_CLOCK_GCLK_5_ENABLE                false
#  define CONF_CLOCK_GCLK_5_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_5_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_5_PRESCALER             1
#  define CONF_CLOCK_GCLK_5_OUTPUT_ENABLE         false

/* GCLK発振器6構成設定 */
#  define CONF_CLOCK_GCLK_6_ENABLE                false
#  define CONF_CLOCK_GCLK_6_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_6_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_6_PRESCALER             1
#  define CONF_CLOCK_GCLK_6_OUTPUT_ENABLE         false

/* GCLK発振器7構成設定 */
#  define CONF_CLOCK_GCLK_7_ENABLE                false
#  define CONF_CLOCK_GCLK_7_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_7_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_7_PRESCALER             1
#  define CONF_CLOCK_GCLK_7_OUTPUT_ENABLE         false

/* GCLK発振器8構成設定 */
#  define CONF_CLOCK_GCLK_8_ENABLE                false
#  define CONF_CLOCK_GCLK_8_RUN_IN_STANDBY        false
#  define CONF_CLOCK_GCLK_8_CLOCK_SOURCE          SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_8_PRESCALER             1
#  define CONF_CLOCK_GCLK_8_OUTPUT_ENABLE         false

#endif /* CONF_CLOCKS_H_INCLUDED */
```

11.3.4. conf_board.h

11.3.4.1. AT32UC3A0, AT32UC3A1, AT32UC3Bﾃﾞﾊﾞｲｽ (USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// この例については既定基板初期化(ｽｲｯﾁ/LED)だけが必要です。

#endif /* CONF_BOARD_H_INCLUDED */
```

11.3.4.2. AT32UC3A3, AT32UC3A4ﾃﾞﾊﾞｲｽ (高速(HS)支援付USBB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */
```

```
#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// この例については既定基板初期化(ｽｲｯﾁ/LED)だけが必要です。

#endif /* CONF_BOARD_H_INCLUDED */
```

11.3.4.3. AT32UC3C, ATUCXXD, ATUCXXL3U, ATUCXXL4Uﾃﾞﾊﾞｲｽ (USBC)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// この例については既定基板初期化(ｽｲｯﾁ/LED)だけが必要です。

#endif /* CONF_BOARD_H_INCLUDED */
```

11.3.4.4. SAM3X, SAM3Aﾃﾞﾊﾞｲｽ (UOTGHS:USB OTG 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

/* USBﾋﾟﾝが使われます。 */
#define CONF_BOARD_USB_PORT

#endif /* CONF_BOARD_H_INCLUDED */
```

11.3.4.5. SAM D21ﾃﾞﾊﾞｲｽ (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

/* USB VBUS検出許可 */
#define CONF_BOARD_USB_VBUS_DETECT
/* ID検出許可 */
#define CONF_BOARD_USB_ID_DETECT

#endif /* CONF_BOARD_H_INCLUDED */
```

12. 供給者ｸﾗｽ装置用USBﾎｽﾄ ｲﾝﾀｰﾌｪｰｽ(UHI)

供給者ｸﾗｽ装置用USBﾎｽﾄ ｲﾝﾀｰﾌｪｰｽ(UHI)はUSB供給者ﾎｽﾄの構成設定と管理に関するｲﾝﾀｰﾌｪｰｽを提供します。

この資料の概要は以下のとおりです。

- API概要
- USBﾎｽﾄ供給者単位部(UHI供給者)用の即時開始の手引き
- 構成設定ﾌｧｲﾙ例

Atmel®ｿﾌﾄｳｪｱ枠組み(ASF) USB装置階層に関するより多くの詳細については以下の応用記述を参照してください。

- AVR4950：ASF – USBﾎｽﾄ階層

12.1. API概要

12.1.1. ﾏｸﾛ定義

12.1.1.1. USBﾎｽﾄ ｺｱ(UHC)とのｲﾝﾀｰﾌｪｰｽ

UHI_VENDOR ﾏｸﾛ

```
#define UHI_VENDOR
```

UHC用の標準UHI APIを含む全域定義。これはconf_usb_host.hﾌｧｲﾙからUSB_HOST_UHI定義で追加されなければなりません。

12.1.2. 関数定義

12.1.2.1. UHCによって必要とされる関数

uhi_vendor_install()関数

ｲﾝﾀｰﾌｪｰｽ取り付け

```
uhc_enum_status_t uhi_vendor_install( uhc_device_t * dev )
```

支援されるなら、ｲﾝﾀｰﾌｪｰｽ ｴﾝﾄﾞﾎﾟｲﾝﾄを割り当て

表12-1. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
dev	[入力]	要求する装置

戻り値：取り付けの状態

uhi_vendor_enable()関数

ｲﾝﾀｰﾌｪｰｽ許可

```
void uhi_vendor_enable( uhc_device_t * dev )
```

UHIに対応するUSBｲﾝﾀｰﾌｪｰｽを許可

表12-2. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
dev	[入力]	要求する装置

uhi_vendor_uninstall()関数

(取り付けられていれば)ｲﾝﾀｰﾌｪｰｽを取り外し

```
void uhi_vendor_uninstall( uhc_device_t * dev )
```

表12-3. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
dev	[入力]	要求する装置

12.1.2.2. 供給者ｸﾗｽ用UHI

USBﾎｽﾄ ｸﾗｽを使う上位応用によって使われる共通的なAPI。

この供給者ｸﾗｽ実装は制御IN、制御OUT、割り込みIN、割り込みOUT、大量(ﾊﾞﾙｸ)IN、大量OUT、等時(ｱｲｿｸﾛﾅｽ)IN、等時OUTの全方向の全ｴﾝﾄﾞﾎﾟｲﾝﾄ形式に対して1つのｴﾝﾄﾞﾎﾟｲﾝﾄを支援します。

この実装は一例で、2つの大量INｴﾝﾄﾞﾎﾟｲﾝﾄのようにより多くのｴﾝﾄﾞﾎﾟｲﾝﾄを支援する別の供給者ｸﾗｽを作成するための基にすることができます。

uhi_vendor_control_in_run()関数

制御INで転送開始

```
bool uhi_vendor_control_in_run( uint8_t * buf, iram_size_t buf_size,
  uhd_callback_setup_end_t callback)
```

転送が終了または失敗(中断停止、リセットなど)された時に呼び戻しが呼ばれます。この呼び戻しは転送状態と結果的に転送されたバイト数を返します。

表12-4. パラメータ

パラメータ名	データ方向	説明
buf	[出力]	送るまたは満たすための内部RAM上の緩衝部。これは整列でなければならず、COMPILER_WORD_ALIGNEDを使ってください。
buf_size	[入力]	送るまたは満たすための緩衝部の大きさ
callback	[入力]	NULLまたは転送の最後で呼ぶ関数

戻り値：関数が成功裏に終了した場合に1、さもなければ0

udi_vendor_control_out_run()関数

制御OUTで転送を開始

```
bool uhi_vendor_control_out_run( uint8_t * buf, iram_size_t buf_size,
  uhd_callback_setup_end_t callback )
```

転送が終了または失敗(中断停止、リセットなど)された時に呼び戻しが呼ばれます。この呼び戻しは転送状態と結果的に転送されたバイト数を返します。

表12-5. パラメータ

パラメータ名	データ方向	説明
buf	[入力]	送るまたは満たすための内部RAM上の緩衝部。これは整列でなければならず、COMPILER_WORD_ALIGNEDを使ってください。
buf_size	[入力]	送るまたは満たすための緩衝部の大きさ
callback	[入力]	NULLまたは転送の最後で呼ぶ関数

戻り値：関数が成功裏に終了した場合に1、さもなければ0

uhi_vendor_bulk_in_run()関数

大量(バルク)INで転送開始

```
bool uhi_vendor_bulk_in_run( uint8_t * buf, iram_size_t buf_size, uhd_callback_trans_t callback )
```

転送が終了または失敗(中断停止、リセットなど)された時に呼び戻しが呼ばれます。この呼び戻しは転送状態と結果的に転送されたバイト数を返します。

表12-6. パラメータ

パラメータ名	データ方向	説明
buf	[出力]	送るまたは満たすための内部RAM上の緩衝部。これは整列でなければならず、COMPILER_WORD_ALIGNEDを使ってください。
buf_size	[入力]	送るまたは満たすための緩衝部の大きさ
callback	[入力]	NULLまたは転送の最後で呼ぶ関数

戻り値：関数が成功裏に終了した場合に1、さもなければ0

udi_vendor_bulk_out_run()関数

大量(バルク)OUTで転送を開始

```
bool uhi_vendor_bulk_out_run( uint8_t * buf, iram_size_t buf_size, uhd_callback_trans_t callback )
```

転送が終了または失敗(中断停止、リセットなど)された時に呼び戻しが呼ばれます。この呼び戻しは転送状態と結果的に転送されたバイト数を返します。

表12-7. パラメータ

パラメータ名	データ方向	説明
buf	[入力]	送るまたは満たすための内部RAM上の緩衝部。これは整列でなければならず、COMPILER_WORD_ALIGNEDを使ってください。
buf_size	[入力]	送るまたは満たすための緩衝部の大きさ
callback	[入力]	NULLまたは転送の最後で呼ぶ関数

戻り値：関数が成功裏に終了した場合に1、さもなければ0

uhi_vendor_int_in_run()関数

割り込みINで転送開始

```
bool uhi_vendor_int_in_run( uint8_t * buf, iram_size_t buf_size, uhd_callback_trans_t callback )
```

転送が終了または失敗(中断停止、リセットなど)された時に呼び戻しが呼ばれます。この呼び戻しは転送状態と結果的に転送されたバイト数を返します。

表12-8. パラメータ

パラメータ名	データ方向	説明
buf	[出力]	送るまたは満たすための内部RAM上の緩衝部。これは整列でなければならず、COMPILER_WORD_ALIGNEDを使ってください。
buf_size	[入力]	送るまたは満たすための緩衝部の大きさ
callback	[入力]	NULLまたは転送の最後で呼ぶ関数

戻り値：関数が成功裏に終了した場合に1、さもなければ0

uhi_vendor_int_out_run()関数

割り込みOUTで転送を開始

```
bool uhi_vendor_int_out_run( uint8_t * buf, iram_size_t buf_size, uhd_callback_trans_t callback )
```

転送が終了または失敗(中断停止、リセットなど)された時に呼び戻しが呼ばれます。この呼び戻しは転送状態と結果的に転送されたバイト数を返します。

表12-9. パラメータ

パラメータ名	データ方向	説明
buf	[入力]	送るまたは満たすための内部RAM上の緩衝部。これは整列でなければならず、COMPILER_WORD_ALIGNEDを使ってください。
buf_size	[入力]	送るまたは満たすための緩衝部の大きさ
callback	[入力]	NULLまたは転送の最後で呼ぶ関数

戻り値：関数が成功裏に終了した場合に1、さもなければ0

uhi_vendor_iso_in_run()関数

等時(アイソクロナス)INで転送開始

```
bool uhi_vendor_iso_in_run( uint8_t * buf, iram_size_t buf_size, uhd_callback_trans_t callback )
```

転送が終了または失敗(中断停止、リセットなど)された時に呼び戻しが呼ばれます。この呼び戻しは転送状態と結果的に転送されたバイト数を返します。

表12-10. パラメータ

パラメータ名	データ方向	説明
buf	[出力]	送るまたは満たすための内部RAM上の緩衝部。これは整列でなければならず、COMPILER_WORD_ALIGNEDを使ってください。
buf_size	[入力]	送るまたは満たすための緩衝部の大きさ
callback	[入力]	NULLまたは転送の最後で呼ぶ関数

戻り値：関数が成功裏に終了した場合に1、さもなければ0

uhi_vendor_iso_out_run()関数

等時(アイソクロナス)OUTで転送を開始

```
bool uhi_vendor_iso_out_run( uint8_t * buf, iram_size_t buf_size, uhd_callback_trans_t callback )
```

転送が終了または失敗(中断停止、リセットなど)された時に呼び戻しが呼ばれます。この呼び戻しは転送状態と結果的に転送されたバイト数を返します。

表12-11. パラメータ

パラメータ名	データ方向	説明
buf	[入力]	送るまたは満たすための内部RAM上の緩衝部。これは整列でなければならず、COMPILER_WORD_ALIGNEDを使ってください。
buf_size	[入力]	送るまたは満たすための緩衝部の大きさ
callback	[入力]	NULLまたは転送の最後で呼ぶ関数

戻り値：関数が成功裏に終了した場合に1、さもなければ0

uhi_vendor_bulk_is_available()関数

大量(ﾊﾞﾙｸ)での転送が可能か調査

```
bool uhi_vendor_bulk_is_available( void )
```

戻り値：可能ならば1、さもなければ0

uhi_vendor_int_is_available()関数

割り込みでの転送が可能か調査

```
bool uhi_vendor_int_is_available( void )
```

戻り値：可能ならば1、さもなければ0

uhi_vendor_iso_is_available()関数

等時(ｱｲｿｸﾛﾅｽ)での転送が可能か調査

```
bool uhi_vendor_int_is_available( void )
```

戻り値：可能ならば1、さもなければ0

12.2 USBホスト供給者単位部(UHI供給者)用の即時開始の手引き

これは使用事例の選択に於いて単位部をどう構成設定して使うかを段階的に指示する、USBﾎｽﾄ供給者単位部(UHI供給者)用の即時開始の手引きです。

使用事例は様々なｺｰﾄﾞの断片を強調します。構成設定に関する段階でのｺｰﾄﾞの断片は独自初期化関数内に複写することができ、一方使用に関する段階は例えば主応用関数内に複写することができます。

12.2.1. 基本的な使用事例

この基本的な使用事例では"USB Host Vendor (Single Class support)"単位部が使われます。

"USB Host Vendor (Composite)"単位部の使い方は高度な使用事例で記述されます。

12.2.1.1. 構成設定段階

USBﾎｽﾄのため、共通USBﾎｽﾄ構成設定段階に従います。USBﾎｽﾄ基本構成設定を参照してください。

12.2.1.2. 使用段階

ｺｰﾄﾞ例

conf_usb_host.hの内容

```
#define USB_HOST_UHI         UHI_VENDOR
#define UHI_VENDOR_CHANGE(dev, b_plug) my_callback_vendor_change(dev, b_plug)
extern void my_callback_vendor_change(uhc_device_t* dev, bool b_plug);
#define UHI_VENDOR_VID_PID_LIST {USB_VID_ATMEL, USB_PID_ATMEL_ASF_VENDOR_CLASS}
#include "uhi_vendor.h" // conf_usb_host.hﾌｧｲﾙの最後で
```

応用Cﾌｧｲﾙに追加してください。

```
static bool my_flag_vendor_test_start = false;
void my_callback_vendor_change(uhc_device_t* dev, bool b_plug)
{
   // USB装置供給者接続
   my_flag_vendor_test_start = b_plug;
}

static void my_callback_bulk_in_done (usb_add_t add,
        usb_ep_t ep, uhd_trans_status_t status, iram_size_t nb_transfered)
{
  if (status != UHD_TRANS_NOERROR) {
    return; // 転送中に異常
  }
  // ﾃﾞｰﾀ受信後に検査再開
  my_flag_vendor_test_start = true;
}

#define MESSAGE "Hello bulk"
#define HELLO_SIZE 5
#define HELLO_BULK_SIZE 10
uint8_t my_out_buffer[MESSAGE_SIZE+1] = MESSAGE;
```

```
uint8_t my_in_buffer[MESSAGE_SIZE+1];
void my_task(void)
{
   if (!my_flag_vendor_test_start) {
     return;
   }
   my_flag_vendor_test_start = false;

   // 制御ｴﾝﾄﾞﾎﾟｲﾝﾄを通してﾃﾞｰﾀ送出
   uhi_vendor_control_out_run(my_out_buffer, HELLO_SIZE, NULL);

   // 大量(ﾊﾞﾙｸ)ｴﾝﾄﾞﾎﾟｲﾝﾄが利用可能か調査
   if (uhi_vendor_bulk_is_available()) {
     // 大量(ﾊﾞﾙｸ)INｴﾝﾄﾞﾎﾟｲﾝﾄを通してﾃﾞｰﾀ送出
     uhi_vendor_bulk_out_run(my_out_buffer, HELLO_BULK_SIZE, NULL);
     // 大量(ﾊﾞﾙｸ)INｴﾝﾄﾞﾎﾟｲﾝﾄを通してﾃﾞｰﾀ受信
     uhi_vendor_bulk_in_run(my_in_buffer, sizeof(my_in_buffer), my_callback_bulk_in_done);
   }
}
```

作業の流れ

1. conf_usb_host.hが利用可能でUSBホスト供給者構成設定である以下の構成設定を含むことを確実にしてください。

```
#define USB_HOST_UHI    UHI_HID_VENDOR
```

注：これはUSBホストによって支援されるUHIの一覧を定義します。

```
#define UHI_VENDOR_CHANGE(dev, b_plug) my_callback_vendor_change(dev, b_plug)
extern bool my_callback_vendor_change(uhc_device_t* dev, bool b_plug);
```

注：この呼び戻しはUSB装置供給者が接続または切断された時に呼ばれます。

```
#define UHI_VENDOR_VID_PID_LIST {USB_VID_ATMEL, USB_PID_ATMEL_ASF_VENDOR_CLASS}
```

注：これは(VIDとPIDによって定義され、)USBホストによって支援される装置の一覧を定義します。

2. 供給者ﾃﾞｰﾀ転送関数はuhi_vendor_groupで記述されます。

```
uhi_vendor_control_out_run(), uhi_vendor_bulk_out_run(), ～
```

12.2.2. 高度な使用事例

UHI供給者単位部のもっと高度な使用については以下をご覧ください。

・USBホストの高度な使用事例

12.3. 構成設定ファイル例

12.3.1. conf_usb_host.h

12.3.1.1. 単一UHI供給者

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_USB_HOST_H_
#define _CONF_USB_HOST_H_

#include "compiler.h"

#define USB_HOST_UHI         UHI_VENDOR

#define USB_HOST_POWER_MAX   500

// #define USB_HOST_HUB_SUPPORT

#if (UC3A3 || UC3A4)
#  define USB_HOST_HS_SUPPORT
#elif (SAM3XA)
```

```
#  define USB_HOST_HS_SUPPORT
#endif

//#define UHC_MODE_CHANGE(b_host_mode)         usb_host_mode_change(b_host_mode)

//#define UHC_VBUS_CHANGE(b_present)           usb_host_vbus_change(b_present)

//#define UHC_VBUS_ERROR()                     usb_host_vbus_error()

//#define UHC_CONNECTION_EVENT(dev,b_present) usb_host_connection_event(dev,b_present)

//#define UHC_WAKEUP_EVENT()                   usb_host_wakeup_event()

//#define UHC_SOF_EVENT()                      usb_host_sof_event()

//#define UHC_DEVICE_CONF(dev)                 uint8_t usb_host_device_conf(dev)

//#define UHC_ENUM_EVENT(dev,b_status)         usb_host_enum_event(dev,b_status)

#define UHI_VENDOR_CHANGE(dev, b_plug)
#define UHI_VENDOR_VID_PID_LIST {USB_VID_ATMEL, USB_PID_ATMEL_ASF_VENDOR_CLASS}

#include "uhi_vendor.h"

#endif // _CONF_USB_HOST_H_
```

12.3.1.2. 複数UHI供給者（複合）

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef _CONF_USB_HOST_H_
#define _CONF_USB_HOST_H_

#include "compiler.h"

#define USB_HOST_UHI         // UHI_MSC, UHI_HID_MOUSE, UHI_CDC, UHI_VENDOR

#define USB_HOST_POWER_MAX  500

// #define USB_HOST_HUB_SUPPORT

#if (UC3A3 || UC3A4)
#  define USB_HOST_HS_SUPPORT
#endif

//#define UHC_MODE_CHANGE(b_host_mode)         usb_host_mode_change(b_host_mode)

//#define UHC_VBUS_CHANGE(b_present)           usb_host_vbus_change(b_present)

//#define UHC_VBUS_ERROR()                     usb_host_vbus_error()

//#define UHC_CONNECTION_EVENT(dev,b_present) usb_host_connection_event(dev,b_present)

//#define UHC_WAKEUP_EVENT()                   usb_host_wakeup_event()

//#define UHC_SOF_EVENT()                      usb_host_sof_event()

//#define UHC_DEVICE_CONF(dev)                 uint8_t usb_host_device_conf(dev)

//#define UHC_ENUM_EVENT(dev,b_status)         usb_host_enum_event(dev,b_status)
```

```
#define UHI_HID_MOUSE_CHANGE(dev,b_plug)
#define UHI_HID_MOUSE_EVENT_BTN_LEFT(b_state)
#define UHI_HID_MOUSE_EVENT_BTN_RIGHT(b_state)
#define UHI_HID_MOUSE_EVENT_BTN_MIDDLE(b_state)
#define UHI_HID_MOUSE_EVENT_MOUVE(x,y,scroll)

#define UHI_MSC_CHANGE(dev,b_plug)

#define UHI_CDC_CHANGE(dev,b_plug)
#define UHI_CDC_RX_NOTIFY()

#define UHI_VENDOR_CHANGE(dev, b_plug)
#define UHI_VENDOR_VID_PID_LIST {USB_VID_ATMEL, USB_PID_ATMEL_ASF_VENDOR_CLASS}

//#include "uhi_msc.h"
//#include "uhi_hid_mouse.h"

#endif // _CONF_USB_HOST_H_
```

12.3.2. conf_clock.h

12.3.2.1. SAM3X, SAM3Aﾃﾞﾊﾞｲｽ (UOTGHS:USB OTG 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

/* ===== ｼｽﾃﾑ ｸﾛｯｸ(MCK)元任意選択 */
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_SLCK_BYPASS
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_4M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_8M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_12M_RC
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_XTAL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_MAINCK_BYPASS
#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLLACK
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_UPLLCK

/* ===== ｼｽﾃﾑ ｸﾛｯｸ(MCK)前置分周器任意選択    (Fmck = Fsys / (SYSCLK_PRES)) */
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_1
#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_2
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_4
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_8
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_16
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_32
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_64
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_3

/* ===== PLL0 (A)任意選択    (Fpll = (Fclk * PLL_mul) / PLL_div)
   ここにmulとdivの実効値を使ってください。 */
#define CONFIG_PLL0_SOURCE          PLL_SRC_MAINCK_XTAL
#define CONFIG_PLL0_MUL             14
#define CONFIG_PLL0_DIV             1

/* ===== 480MHzで固定化されたUPLL (UTMI)ﾊｰﾄﾞｳｪｱ */

/* ===== USBｸﾛｯｸ元任意選択    (Fusb = FpllX / USB_div)
   ここにdivの実効値を使ってください。 */
```

```
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL0
#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_UPLL
#define CONFIG_USBCLK_DIV           1

/*
===== 目的対象周波数 (システム クロック)
- XTAL周波数: 12MHz
- システム クロック元: PLLA
- システム クロック前置分周器: 2 (2分周)
- PLLA供給元: XTAL
- PLLA出力: XTAL * 14 / 1
- システム クロックは: 12 * 14 / 1 /2 = 84MHz
===== 目的対象周波数 (USBクロック)
- USBクロック元: UPLL
- USBクロック分周器: 1 (分周なし)
- UPLL周波数: 480MHz
- USBクロック: 480 / 1 = 480MHz
*/

#endif /* CONF_CLOCK_H_INCLUDED */
```

12.3.2.2. SAM4Lﾃﾞﾊﾞｲｽ (USBC)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_CLOCK_H_INCLUDED
#define CONF_CLOCK_H_INCLUDED

//#define CONFIG_SYSCLK_INIT_CPUMASK  (1 << SYSCLK_OCD)
//#define CONFIG_SYSCLK_INIT_PBAMASK  (1 << SYSCLK_IISC)
//#define CONFIG_SYSCLK_INIT_PBBMASK  (1 << SYSCLK_USBC_REGS)
//#define CONFIG_SYSCLK_INIT_PBCMASK  (1 << SYSCLK_CHIPID)
//#define CONFIG_SYSCLK_INIT_PBDMASK  (1 << SYSCLK_AST)
//#define CONFIG_SYSCLK_INIT_HSBMASK  (1 << SYSCLK_PDCA_HSB)

//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RCSYS
#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_OSC0
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_PLL0
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_DFLL
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RC80M
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RCFAST
//#define CONFIG_SYSCLK_SOURCE        SYSCLK_SRC_RC1M

/* RCFAST周波数選択: 4MHzに対して0、8MHzに対して1、12MHzに対して2 */
//#define CONFIG_RCFAST_FRANGE    0
//#define CONFIG_RCFAST_FRANGE    1
//#define CONFIG_RCFAST_FRANGE    2

/* Fbus = Fsys / (2 ^ BUS_div) */
#define CONFIG_SYSCLK_CPU_DIV       0
#define CONFIG_SYSCLK_PBA_DIV       0
#define CONFIG_SYSCLK_PBB_DIV       0
#define CONFIG_SYSCLK_PBC_DIV       0
#define CONFIG_SYSCLK_PBD_DIV       0

// ===== 全ての必須でない周辺機能のクロックを禁止
//#define CONFIG_SYSCLK_INIT_CPUMASK  0
//#define CONFIG_SYSCLK_INIT_PBAMASK  SYSCLK_USART1
//#define CONFIG_SYSCLK_INIT_PBBMASK  0
//#define CONFIG_SYSCLK_INIT_PBCMASK  0
```

```
//#define CONFIG_SYSCLK_INIT_PBDMASK  0
//#define CONFIG_SYSCLK_INIT_HSBMASK  0

// ===== PLL任意選択
#define CONFIG_PLL0_SOURCE          PLL_SRC_OSC0
//#define CONFIG_PLL0_SOURCE          PLL_SRC_GCLK9

/* Fpll0 = (Fclk * PLL_mul) / PLL_div */
#define CONFIG_PLL0_MUL             (48000000UL / BOARD_OSC0_HZ)
#define CONFIG_PLL0_DIV             1
//#define CONFIG_PLL0_MUL             (192000000 / FOSC0) /* Fpll = (Fclk * PLL_mul) / PLL_div */
//#define CONFIG_PLL0_DIV             4 /* Fpll = (Fclk * PLL_mul) / PLL_div */

// ==== DFLL任意選択
//#define CONFIG_DFLL0_SOURCE         GENCLK_SRC_OSC0
//#define CONFIG_DFLL0_SOURCE         GENCLK_SRC_RCSYS
//#define CONFIG_DFLL0_SOURCE         GENCLK_SRC_OSC32K
//#define CONFIG_DFLL0_SOURCE         GENCLK_SRC_RC120M
//#define CONFIG_DFLL0_SOURCE         GENCLK_SRC_RC32K

/* Fdfll = (Fclk * DFLL_mul) / DFLL_div */
//#define CONFIG_DFLL0_FREQ           48000000UL
//#define CONFIG_DFLL0_MUL            ((4 * CONFIG_DFLL0_FREQ) / BOARD_OSC32_HZ)
//#define CONFIG_DFLL0_DIV            4
//#define CONFIG_DFLL0_MUL            (CONFIG_DFLL0_FREQ / BOARD_OSC32_HZ)
//#define CONFIG_DFLL0_DIV            1

// ===== USBｸﾛｯｸ元任意選択
#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_PLL0
//#define CONFIG_USBCLK_SOURCE        USBCLK_SRC_DFLL

/* Fusb = Fsys / USB_div */
#define CONFIG_USBCLK_DIV           1

// ===== GCLK9任意選択
//#define CONFIG_GCLK9_SOURCE           GENCLK_SRC_GCLKIN0
//#define CONFIG_GCLK9_DIV              1

#endif /* CONF_CLOCK_H_INCLUDED */
```

12.3.3. conf_clocks.h

12.3.3.1. SAM D21ﾃﾞﾊﾞｲｽ (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#include <clock.h>

#ifndef CONF_CLOCKS_H_INCLUDED
#  define CONF_CLOCKS_H_INCLUDED

/* ｼｽﾃﾑ ｸﾛｯｸ ﾊﾞｽ構成設定 */
#  define CONF_CLOCK_CPU_CLOCK_FAILURE_DETECT     false
#  define CONF_CLOCK_FLASH_WAIT_STATES            2
#  define CONF_CLOCK_CPU_DIVIDER                  SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBA_DIVIDER                 SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBB_DIVIDER                 SYSTEM_MAIN_CLOCK_DIV_1
#  define CONF_CLOCK_APBC_DIVIDER                 SYSTEM_MAIN_CLOCK_DIV_1

/* SYSTEM_CLOCK_SOURCE_OSC8M構成設定 - 内部8MHz発振器 */
#  define CONF_CLOCK_OSC8M_PRESCALER              SYSTEM_OSC8M_DIV_1
```

```
#  define CONF_CLOCK_OSC8M_ON_DEMAND               true
#  define CONF_CLOCK_OSC8M_RUN_IN_STANDBY          true

/* SYSTEM_CLOCK_SOURCE_XOSC構成設定 - 外部ｸﾛｯｸ/発振器 */
#  define CONF_CLOCK_XOSC_ENABLE                   false
#  define CONF_CLOCK_XOSC_EXTERNAL_CRYSTAL         SYSTEM_CLOCK_EXTERNAL_CRYSTAL
#  define CONF_CLOCK_XOSC_EXTERNAL_FREQUENCY       12000000UL
#  define CONF_CLOCK_XOSC_STARTUP_TIME             SYSTEM_XOSC_STARTUP_32768
#  define CONF_CLOCK_XOSC_AUTO_GAIN_CONTROL        true
#  define CONF_CLOCK_XOSC_ON_DEMAND                true
#  define CONF_CLOCK_XOSC_RUN_IN_STANDBY           false

/* SYSTEM_CLOCK_SOURCE_XOSC32K構成設定 - 外部32kHzｸﾘｽﾀﾙ/ｸﾛｯｸ発振器 */
#  define CONF_CLOCK_XOSC32K_ENABLE                true

#  define CONF_CLOCK_XOSC32K_EXTERNAL_CRYSTAL      SYSTEM_CLOCK_EXTERNAL_CRYSTAL
#  define CONF_CLOCK_XOSC32K_STARTUP_TIME          SYSTEM_XOSC32K_STARTUP_65536
#  define CONF_CLOCK_XOSC32K_AUTO_AMPLITUDE_CONTROL  false
#  define CONF_CLOCK_XOSC32K_ENABLE_1KHZ_OUPUT     false
#  define CONF_CLOCK_XOSC32K_ENABLE_32KHZ_OUTPUT   true
#  define CONF_CLOCK_XOSC32K_ON_DEMAND             false
#  define CONF_CLOCK_XOSC32K_RUN_IN_STANDBY        true

/* SYSTEM_CLOCK_SOURCE_OSC32K構成設定 - 内部32kHz発振器 */
#  define CONF_CLOCK_OSC32K_ENABLE                 false
#  define CONF_CLOCK_OSC32K_STARTUP_TIME           SYSTEM_OSC32K_STARTUP_130
#  define CONF_CLOCK_OSC32K_ENABLE_1KHZ_OUTPUT     false
#  define CONF_CLOCK_OSC32K_ENABLE_32KHZ_OUTPUT    true
#  define CONF_CLOCK_OSC32K_ON_DEMAND              true
#  define CONF_CLOCK_OSC32K_RUN_IN_STANDBY         false

/* SYSTEM_CLOCK_SOURCE_DFLL構成設定 - ﾃﾞｼﾞﾀﾙ周波数固定化閉路 */
#  define CONF_CLOCK_DFLL_ENABLE                   true
#  define CONF_CLOCK_DFLL_LOOP_MODE                SYSTEM_CLOCK_DFLL_LOOP_MODE_CLOSED
#  define CONF_CLOCK_DFLL_ON_DEMAND                true

/* DFLL開路動作構成設定 */
#  define CONF_CLOCK_DFLL_FINE_VALUE               (512)

/* DFLL閉路動作構成設定 */
#  define CONF_CLOCK_DFLL_SOURCE_GCLK_GENERATOR    GCLK_GENERATOR_1
#  define CONF_CLOCK_DFLL_MULTIPLY_FACTOR          (48000000/32768)
#  define CONF_CLOCK_DFLL_QUICK_LOCK               true
#  define CONF_CLOCK_DFLL_TRACK_AFTER_FINE_LOCK    true
#  define CONF_CLOCK_DFLL_KEEP_LOCK_ON_WAKEUP      true
#  define CONF_CLOCK_DFLL_ENABLE_CHILL_CYCLE       true
#  define CONF_CLOCK_DFLL_MAX_COARSE_STEP_SIZE     (0x1f / 8)
#  define CONF_CLOCK_DFLL_MAX_FINE_STEP_SIZE       (0xff / 8)

/* SYSTEM_CLOCK_SOURCE_DPLL構成設定 - ﾃﾞｼﾞﾀﾙ位相固定化閉路 */
#  define CONF_CLOCK_DPLL_ENABLE                   false
#  define CONF_CLOCK_DPLL_ON_DEMAND                false
#  define CONF_CLOCK_DPLL_RUN_IN_STANDBY           true
#  define CONF_CLOCK_DPLL_LOCK_BYPASS              false
#  define CONF_CLOCK_DPLL_WAKE_UP_FAST             false
#  define CONF_CLOCK_DPLL_LOW_POWER_ENABLE         true

#  define CONF_CLOCK_DPLL_LOCK_TIME                SYSTEM_CLOCK_SOURCE_DPLL_LOCK_TIME_DEFAULT
#  define CONF_CLOCK_DPLL_REFERENCE_CLOCK          SYSTEM_CLOCK_SOURCE_DPLL_REFERENCE_CLOCK_XOSC32K
#  define CONF_CLOCK_DPLL_FILTER                   SYSTEM_CLOCK_SOURCE_DPLL_FILTER_DEFAULT
```

```
#  define CONF_CLOCK_DPLL_REFERENCE_FREQUENCY      32768
#  define CONF_CLOCK_DPLL_REFERENCE_DIVIDER        1
#  define CONF_CLOCK_DPLL_OUTPUT_FREQUENCY         48000000

/* DPLL GCLK基準構成設定 */
# define CONF_CLOCK_DPLL_REFERENCE_GCLK_GENERATOR GCLK_GENERATOR_1
/* DPLL GCLK固定化計時器構成設定 */
# define CONF_CLOCK_DPLL_LOCK_GCLK_GENERATOR GCLK_GENERATOR_1

/* clocks_init走行時にGCLKを構成設定するためにこれをtrueに設定してください。
 * falseに設定した場合、GCLK発振器はclocks_init()で何も構成設定されません。*/
#  define CONF_CLOCK_CONFIGURE_GCLK                true

/* GCLK発振器0構成設定（主ｸﾛｯｸ）*/
#  define CONF_CLOCK_GCLK_0_ENABLE                 true
#  define CONF_CLOCK_GCLK_0_RUN_IN_STANDBY         true
#  define CONF_CLOCK_GCLK_0_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_DFLL
#  define CONF_CLOCK_GCLK_0_PRESCALER              1
#  define CONF_CLOCK_GCLK_0_OUTPUT_ENABLE          false

/* GCLK発振器1構成設定 */
#  define CONF_CLOCK_GCLK_1_ENABLE                 true
#  define CONF_CLOCK_GCLK_1_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_1_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_XOSC32K
#  define CONF_CLOCK_GCLK_1_PRESCALER              1
#  define CONF_CLOCK_GCLK_1_OUTPUT_ENABLE          false

/* GCLK発振器2構成設定（RTC）*/
#  define CONF_CLOCK_GCLK_2_ENABLE                 false
#  define CONF_CLOCK_GCLK_2_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_2_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC32K
#  define CONF_CLOCK_GCLK_2_PRESCALER              32
#  define CONF_CLOCK_GCLK_2_OUTPUT_ENABLE          false

/* GCLK発振器3構成設定 */
#  define CONF_CLOCK_GCLK_3_ENABLE                 false
#  define CONF_CLOCK_GCLK_3_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_3_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_3_PRESCALER              1
#  define CONF_CLOCK_GCLK_3_OUTPUT_ENABLE          false

/* GCLK発振器4構成設定 */
#  define CONF_CLOCK_GCLK_4_ENABLE                 false
#  define CONF_CLOCK_GCLK_4_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_4_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_4_PRESCALER              1
#  define CONF_CLOCK_GCLK_4_OUTPUT_ENABLE          false

/* GCLK発振器5構成設定 */
#  define CONF_CLOCK_GCLK_5_ENABLE                 false
#  define CONF_CLOCK_GCLK_5_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_5_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_5_PRESCALER              1
#  define CONF_CLOCK_GCLK_5_OUTPUT_ENABLE          false

/* GCLK発振器6構成設定 */
#  define CONF_CLOCK_GCLK_6_ENABLE                 false
#  define CONF_CLOCK_GCLK_6_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_6_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_6_PRESCALER              1
#  define CONF_CLOCK_GCLK_6_OUTPUT_ENABLE          false
```

```
/* GCLK発振器7構成設定 */
#  define CONF_CLOCK_GCLK_7_ENABLE                 false
#  define CONF_CLOCK_GCLK_7_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_7_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_7_PRESCALER              1
#  define CONF_CLOCK_GCLK_7_OUTPUT_ENABLE          false

/* GCLK発振器8構成設定 */
#  define CONF_CLOCK_GCLK_8_ENABLE                 false
#  define CONF_CLOCK_GCLK_8_RUN_IN_STANDBY         false
#  define CONF_CLOCK_GCLK_8_CLOCK_SOURCE           SYSTEM_CLOCK_SOURCE_OSC8M
#  define CONF_CLOCK_GCLK_8_PRESCALER              1
#  define CONF_CLOCK_GCLK_8_OUTPUT_ENABLE          false

#endif /* CONF_CLOCKS_H_INCLUDED */
```

12.3.4. conf_board.h

12.3.4.1. SAM3X, SAM3Aﾃﾞﾊﾞｲｽ (UOTGHS:USB OTG 高速(HS))

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

// USBﾋﾟﾝが使われます。
#define CONF_BOARD_USB_PORT

#endif /* CONF_BOARD_H_INCLUDED */
```

12.3.4.2. SAM4Lﾃﾞﾊﾞｲｽ (USBC)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

/* board_init()が呼ばれる時にUSART用汎用入出力を自動初期化 */
//#define CONF_BOARD_COM_PORT

/* USBｲﾝﾀｰﾌｪｰｽ(USB)を許可) */
#define CONF_BOARD_USB_PORT
/* ID検出許可、PB05/USBｼﾞｬﾝﾊﾟ設定ならば注釈を外してください。 */
#define CONF_BOARD_USB_ID_DETECT
/* ﾎｽﾄVBUS制御許可、PC08/USBｼﾞｬﾝﾊﾟ設定ならば注釈を外してください。 */
#define CONF_BOARD_USB_VBUS_CONTROL
/* ﾎｽﾄVBUS制御許可、PC08/USBｼﾞｬﾝﾊﾟ設定ならば注釈を外してください。 */
#define CONF_BOARD_USB_VBUS_ERR_DETECT

/* 基板監視を制御するためにUSARTを許可 */
//#define CONF_BOARD_BM_USART

/* LCD背面灯を初期化 */
#define CONF_BOARD_BL

#endif /* CONF_BOARD_H_INCLUDED */
```

12.3.4.3. SAM D21デバイス (USB)

```
/*
 * 支援とFAQ: <a href="http://www.atmel.com/design-support/">Atmel Support</a>を尋ねてください。
 */

#ifndef CONF_BOARD_H_INCLUDED
#define CONF_BOARD_H_INCLUDED

/* USB VBUS検出許可 */
#define CONF_BOARD_USB_VBUS_DETECT
/* ID検出許可 */
#define CONF_BOARD_USB_ID_DETECT

#endif /* CONF_BOARD_H_INCLUDED */
```

文書改訂履歴

文書改訂	日付	注釈
42336A	2014年12月	初版公開

Atmel Corporation 1600 Technology Drive, San Jose, CA 95110 USA TEL:(+1)(408) 441-0311 FAX: (+1)(408) 436-4200 | www.atmel.com

---

本応用記述はAtmelのAT09331応用記述(Rev.42336A-12/2014)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。