Atmel マイクロ コントローラ

AT09332：通信クラス装置(CDC)用USB装置インターフェース(UDI)

応用記述

序説

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

目次

1. 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。

2. API概要

2.1. 構造体定義

2.1.1. 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

2.1.2. 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

2.2. ﾏｸﾛ定義

2.2.1. 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。

2.2.1.1. 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

2.2.1.2. 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

2.2.1.3. 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

2.2.1.4. UDI_CDC_IAD_DESC_0 ﾏｸﾛ

```
#define UDI_CDC_IAD_DESC_0
```

ﾎﾟｰﾄ0用IAD記述子

2.2.1.5. UDI_CDC_COMM_DESC_0 ﾏｸﾛ

```
#define UDI_CDC_COMM_DESC_0
```

ﾎﾟｰﾄ0用COMM記述子

2.2.1.6. UDI_CDC_DATA_DESC_0_FS ﾏｸﾛ

```
#define UDI_CDC_DATA_DESC_0_FS
```

全速(FS)装置のﾎﾟｰﾄ0用DATA記述子

2.2.1.7. UDI_CDC_DATA_DESC_0_HS ﾏｸﾛ

```
#define UDI_CDC_DATA_DESC_0_HS
```

高速(HS)装置のﾎﾟｰﾄ0用DATA記述子

2.2.1.8. 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

2.2.1.9. 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

2.2.1.10. UDI_CDC_DATA_STRING_ID_1 マクロ

```
#define UDI_CDC_DATA_STRING_ID_1
```

2.2.1.11. UDI_CDC_IAD_DESC_1 マクロ

```
#define UDI_CDC_IAD_DESC_1
```

2.2.1.12. UDI_CDC_COMM_DESC_1 マクロ

```
#define UDI_CDC_COMM_DESC_1
```

2.2.1.13. UDI_CDC_DATA_DESC_1_FS マクロ

```
#define UDI_CDC_DATA_DESC_1_FS
```

2.2.1.14. UDI_CDC_DATA_DESC_1_HS マクロ

```
#define UDI_CDC_DATA_DESC_1_HS
```

2.2.1.15. 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

2.2.1.16. 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

2.2.1.17. UDI_CDC_DATA_STRING_ID_2 マクロ

```
#define UDI_CDC_DATA_STRING_ID_2
```

2.2.1.18. UDI_CDC_IAD_DESC_2 マクロ

```
#define UDI_CDC_IAD_DESC_2
```

2.2.1.19. UDI_CDC_COMM_DESC_2 マクロ

```
#define UDI_CDC_COMM_DESC_2
```

2.2.1.20. UDI_CDC_DATA_DESC_2_FS マクロ

```
#define UDI_CDC_DATA_DESC_2_FS
```

2.2.1.21. UDI_CDC_DATA_DESC_2_HS マクロ

```
#define UDI_CDC_DATA_DESC_2_HS
```

2.2.1.22. 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

2.2.1.23. 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

2.2.1.24. UDI_CDC_DATA_STRING_ID_3 マクロ

```
#define UDI_CDC_DATA_STRING_ID_3
```

2.2.1.25. UDI_CDC_IAD_DESC_3 マクロ

```
#define UDI_CDC_IAD_DESC_3
```

2.2.1.26. UDI_CDC_COMM_DESC_3 マクロ

```
#define UDI_CDC_COMM_DESC_3
```

2.2.1.27. UDI_CDC_DATA_DESC_3_FS マクロ

```
#define UDI_CDC_DATA_DESC_3_FS
```

2.2.1.28. UDI_CDC_DATA_DESC_3_HS マクロ

```
#define UDI_CDC_DATA_DESC_3_HS
```

2.2.1.29. 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

2.2.1.30. 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

2.2.1.31. UDI_CDC_DATA_STRING_ID_4 マクロ

```
#define UDI_CDC_DATA_STRING_ID_4
```

2.2.1.32. UDI_CDC_IAD_DESC_4 マクロ

```
#define UDI_CDC_IAD_DESC_4
```

2.2.1.33. UDI_CDC_COMM_DESC_4 マクロ

```
#define UDI_CDC_COMM_DESC_4
```

2.2.1.34. UDI_CDC_DATA_DESC_4_FS マクロ

```
#define UDI_CDC_DATA_DESC_4_FS
```

2.2.1.35. UDI_CDC_DATA_DESC_4_HS マクロ

```
#define UDI_CDC_DATA_DESC_4_HS
```

2.2.1.36. 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

2.2.1.37. 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

2.2.1.38. UDI_CDC_DATA_STRING_ID_5 マクロ

```
#define UDI_CDC_DATA_STRING_ID_5
```

2.2.1.39. UDI_CDC_IAD_DESC_5 マクロ

```
#define UDI_CDC_IAD_DESC_5
```

2.2.1.40. UDI_CDC_COMM_DESC_5 マクロ

```
#define UDI_CDC_COMM_DESC_5
```

2.2.1.41. UDI_CDC_DATA_DESC_5_FS マクロ

```
#define UDI_CDC_DATA_DESC_5_FS
```

2.2.1.42. UDI_CDC_DATA_DESC_5_HS マクロ

```
#define UDI_CDC_DATA_DESC_5_HS
```

2.2.1.43. 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

2.2.1.44. 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

2.2.1.45. UDI_CDC_DATA_STRING_ID_6 マクロ

```
#define UDI_CDC_DATA_STRING_ID_6
```

2.2.1.46. UDI_CDC_IAD_DESC_6 マクロ

```
#define UDI_CDC_IAD_DESC_6
```

2.2.1.47. UDI_CDC_COMM_DESC_6 マクロ

```
#define UDI_CDC_COMM_DESC_6
```

2.2.1.48. UDI_CDC_DATA_DESC_6_FS マクロ

```
#define UDI_CDC_DATA_DESC_6_FS
```

2.2.1.49. UDI_CDC_DATA_DESC_6_HS マクロ

```
#define UDI_CDC_DATA_DESC_6_HS
```

2.2.2. 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

2.2.3. 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

2.2.4. 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

2.2.5. 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

2.2.6. 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

2.2.7. 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)

2.2.8. 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)

2.2.9. 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

2.3. 関数定義

2.3.1. 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

2.3.1.1. udi_cdc_ctrl_signal_dcd()関数

ﾃﾞｰﾀ搬送波検出(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. ﾊﾟﾗﾒｰﾀ

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

2.3.1.2. udi_cdc_ctrl_signal_dsr()関数

ﾃﾞｰﾀ設定準備可(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. ﾊﾟﾗﾒｰﾀ

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

2.3.1.3. udi_cdc_signal_framing_error()関数

ﾌﾚｰﾐﾝｸﾞ異常を通知

```
void udi_cdc_signal_framing_error( void )
```

2.3.1.4. udi_cdc_signal_parity_error()関数

ﾊﾟﾘﾃｨ誤りを通知

```
void udi_cdc_signal_parity_error( void )
```

2.3.1.5. udi_cdc_signal_overrun()関数

ｵｰﾊﾞｰﾗﾝを通知

```
void udi_cdc_signal_overrun( void )
```

2.3.1.6. udi_cdc_get_nb_received_data()関数

受信ﾊﾞｲﾄ数取得

```
iram_size_t udi_cdc_get_nb_received_data( void )
```

戻り値：利用可能なﾃﾞｰﾀ数

2.3.1.7. 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：読まれるべきﾊﾞｲﾄが準備可の場合に1

2.3.1.8. 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

2.3.1.9. 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. ﾊﾟﾗﾒｰﾀ

ﾊﾟﾗﾒｰﾀ名	ﾃﾞｰﾀ方向	説明
buf	[出力]	読んだ値
size	[入力]	読んだ値の数

戻り値：残りﾃﾞｰﾀ数

2.3.1.10. 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

2.3.1.11. 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

2.3.1.12. 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

2.3.1.13. 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

2.3.2. 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

2.3.2.1. 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

2.3.2.2. 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

2.3.2.3. 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. 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

2.3.2.4. udi_cdc_multi_signal_parity_error()関数

ﾊﾟﾘﾃｨ誤りを通知

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

表2-11. 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

2.3.2.5. 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

2.3.2.6. 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

2.3.2.7. 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

2.3.2.8. 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

2.3.2.9. 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. 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
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

2.3.2.10. udi_cdc_multi_get_free_tx_buffer()関数

送信緩衝部内の空きバイト数を取得

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

表2-17. 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：送信緩衝部内の空きバイト数

2.3.2.11. 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

2.3.2.12. 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

2.3.2.13. 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

3. 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。

3.1. 基本的な使用事例

この使用事例では1つの通信ﾎﾟｰﾄだけで"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。

3.1.1. 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。

3.1.2. 使用段階

3.1.2.1. 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();
 }
}
```

3.1.2.2. 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);
```

3.2. 高度な使用事例

UDI CDC単位部のもっと高度な使用については以下の使用事例をご覧ください。

- 複合装置でのCDC
- 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
- 独自構成設定
- VBUS監視

3.3. 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。

3.3.1. 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。

3.3.2. 使用段階

3.3.2.1. 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,        ¥
   ～
```

3.3.2.2. 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 ¥
  ～
  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。

3.4. 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。

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。

3.4.2. 使用段階

3.4.2.1. 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
```

3.4.2.2. 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
```

3.5. 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。

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。

3.5.2. 使用段階

3.5.2.1. ｺｰﾄﾞ例

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"
```

3.5.2.2. 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"
```

3.6. 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。

3.6.1. 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。

3.6.2. 使用段階

3.6.2.1. ｺｰﾄﾞ例

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();
```

```
}
```

3.6.2.2. 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パ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)

```
udc_remotewakeup();
```

3.7. バス給電応用推奨

この事例では、USB装置バ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。

3.7.1. 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。

3.7.2. 使用段階

3.7.2.1. コード゙例

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ドウェア部分を禁止
}
void user_callback_resume_action(void)
{
   // ハードウェア部分を再許可
}
```

3.7.2.2. 作業の流れ

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

```
// バス給電機能を認可
#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();
}
```

3.8. 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。

3.8.1. 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。

3.8.2. 使用段階

3.8.2.1. 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';
}
```

3.8.2.2. 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';
}
```

4. 構成設定ﾌｧｲﾙ例

4.1. conf_usb.h

4.1.1. 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

#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"

    #endif // _CONF_USB_H_
```

4.1.2. 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_
```

4.2. conf_clock.h

4.2.1. 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 */
```

4.2.2. 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 */
```

4.2.3. 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 */
```

4.2.4. 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 */
```

4.2.5. 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 */
```

4.2.6. 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_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 */
```

4.2.7. 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 */
```

4.3. conf_clocks.h

4.3.1. 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                true
#  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 */
```

4.4. conf_board.h

4.4.1. 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 */
```

4.4.2. 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 */
```

4.4.3. 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 */
```

4.4.4. 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 */
```

4.4.5. 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 */
```

5. USB装置基本構成設定

5.1. 独自構成設定

以下のUSB装置構成設定は応用のconf_usb.hﾌｧｲﾙでｲﾝｸﾙｰﾄﾞさ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)

5.2. 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();
}
```

5.3. USB装置基本構成設定

5.3.1. 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
```

5.3.2. 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();
}
```

5.3.3. 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。

5.4. 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
```

6. 文書改訂履歴

文書改訂	日付	注釈
42337A	2014年12月	初版文書公開
42337B	2015年12月	誤植修正

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AT09332応用記述(Rev.42337B-12/2015)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。