Smart-USB Sigma 製品対応

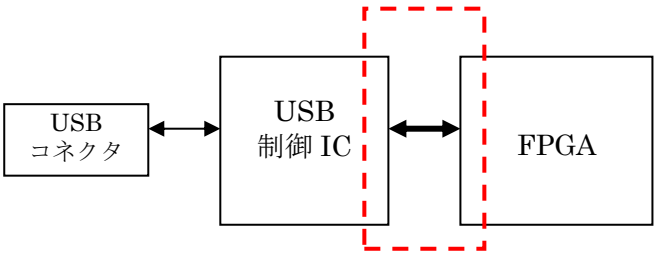
USB インタフェース・プロトコルの概要

1. 概要

Smart-USB Sigma 製品ファミリでは、USB 制御用 IC としてサイプレス社の CYUSB3014 デバイス（以下、USB 制御 IC）を採用しています。この USB 制御 IC と FPGA がボード上で接続しています。このアプリケーション・ノートでは、USB 制御 IC とインタフェースを実現するための FPGA 回路について解説します。図 1 に示す破線で囲んだ部分です。
また、製品に添付する DVD-ROM 内に収録する mnl_Smart-USB_Sigma.pdf を併せてご覧ください。

【適用ボード】

- Smart-USB Sigma 製品ファミリのボード



<図 1. Smart-USB Sigma 製品の基本構成>

2. インタフェース・プロトコルの概要

USB 制御 IC と FPGA 間のインタフェース・プロトコルは以下の信号線を使用して行います。　アクセス種別は 2 種類あり、FPGA 内に実装したハードウエア・レジスタへの書き込み、読み出しをする「レジスタ・アクセス」（USB コントロール転送）とメモリに対するデータの読み出し、書き込み動作をする「メモリ・アクセス」（USB バルク転送）があります。

信号名	FPGA からみた I/O 方向
PCLK	入力
RDn	入力
WRn	入力
CMDn	入力
RGDTn	入力
WAITn	出力
FD[31:0]	双方向

<表 1．IF 信号>

2.1　信号線の概要

信号名　　：PCLK
信号方向　：USB 制御 IC←{100MHz OSC}→FPGA
機能　　：USB 制御を行うための基本クロックです。外部のクロック源（水晶発振器）から USB 制御 IC と FPGA 間の両方に同位相の 100MHz クロックを供給します。

信号名　　：RDn　（Active Low）
信号方向　：USB 制御 IC→FPGA
機能　　：FPGA が USB 制御 IC にデータを送信する際、この信号に同期して USB データバス（FD[31:0]）上にデータを送出します。この信号は PCLK の立ち上がりエッジに同期しています。

信号名　　：WRn（Active Low）
信号方向　：USB 制御 IC→FPGA
機能　　：USB 制御 IC が FPGA にデータを送信する際、この信号に同期して USB データバス（FD[31:0]）上にデータを送出します。この信号は PCLK の立ち上がりエッジに同期しています。

信号名　　：CMDn　（Active Low）
信号方向　：USB 制御 IC→FPGA
機能　　：USB 制御 IC が FPGA に対してコマンドを発行していることを示します。この信号に同期して USB 制御 IC が USB データバス上にコマンド・データを送出します。受信する FPGA では、この信号により USB データバス上のデータが、コマンドであることを判断することができます。この信号は PCLK の立ち上がりエッジに同期しています。

信号名　　：RGDTn
信号方向　：USB 制御 IC→FPGA
機能　　：USB 制御 IC が FPGA に対してアクセス種別を表示します。High レベルの場合、レジスタ・アクセスであることを表示し、Low レベルの場合、メモリ・アクセスであることを示します。この信号は PCLK の立ち上がりエッジに同期しています。

信号名　：WAITn （Active Low）
信号方向　：USB 制御 IC←FPGA
機能　：USB 制御 IC とのデータ転送中（メモリ・アクセス）、FPGA 側でデータの転送を任意にウエイトすることができます。

信号名　：FD[31:0]
信号方向　：USB 制御 IC←（双方向）→FPGA
機能　：100MHz の PCLK の立ち上がりに同期した 32bit データバスです。バスレートは 100MHz x 4 バイト＝400MB/s になります。

3. データ転送プロトコル

レジスタ・アクセス（USB コントロール転送仕様）、メモリ・アクセス（USB バルク転送仕様）を効率よく、高速に処理するプロトコルです。

3.1 レジスタ・アクセス

レジスタ・アクセスでは、1～512 バイトまでバイト単位に転送データ量を指定できます。このアクセスは、USB プロトコルで定義する「コントロール転送」を利用しています。
※1ms 周期で 1 回以上のコントロール転送ができます。

【対応する API（USB プロトコルを実現する専用関数）】

＊ **SUSlv_Reg_Write**
（レジスタへの書き込み動作）
＊ **SUSlv_Reg_Read**
（レジスタからの読み出し動作）

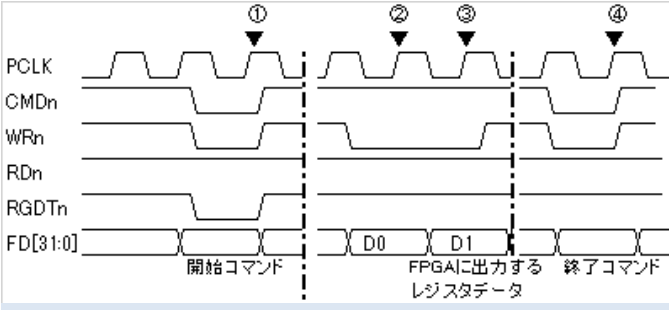
Windows で動作する USB 制御アプリケーションが発行するこれらの API により、以下のアクセスを行います。
USB 制御 IC は、FPGA に開始コマンドを送出し、レジスタ・アクセスであることを FPGA に表示します(RGDTn=High)。開始コマンドには、転送状態(開始/終了)、転送モード(READ/WRITE)、レジスタ No.（レジスタ・アドレスまたはレジスタ番号）と転送するバイト数を含みます。　次に、レジスタライトならば、開始コマンドで示したレジスタに対して、書き込みたいデータを WRn 信号とともに FPGA に出力します。レジスタリードなら、FPGA に RDn 信号を出力して、FPGA が RDn に同期したデータを USB 制御 IC に出力します。

3.2 レジスタ・アクセス・タイミング

① レジスタにデータを書き込む場合（レジスタライト）

具体例 1：FPGA に構成したハードウエア・レジスタに 6 バイトの任意の値を書き込む。

・ 開始コマンド内容：80C00002（bit31= 1 ）
・ （レジスタ No.2 に 6 バイトのデータを書き込む）
・ 終了コマンド内容：00C00002　（bit31=0）



<図 2.　基本レジスタ・アクセス・タイミング>

USB 制御アプリケーションが専用 API の SUSlv_Reg_Write を発行すると、USB 制御 IC は各制御信号をアクティブにし、USB データバス（FD[31:0]）上にコマンドと設定データを出力します。FPGA 側では図 2 に示す①のクロックエッジで FD[31:0]のデータを取り込みます。この値がレジスタ・アクセスの開始を FPGA に通知する開始コマンドになります。このコマンドには、転送開始ビット、ライト動作を表す転送モード表示、転送サイズ（6 バイト）、レジスタ No.情報（No.2）を含みます。　FPGA ではこの開始コマンドを適切にデコードすることで、所望のレジスタに正しくアクセスすることができます。

【レジスタ・アクセスのコマンド内容】

Bit	31	30	29-21(9bit)	20-0(21bit)
内容	状態	モード	転送サイズ (1～512 バイト)	レジスタ No.指定 (0～2,097,151)

[bit 31]　転送状態表示
1 ：レジスタ・アクセス(転送)の開始を表示
0 ：レジスタ・アクセス(転送)の終了を表示

[bit 30] 転送モード表示
1 ：レジスタ・アクセス（READ）を表示
0 ：レジスタ・アクセス（WRITE）を表示

[bit29-21] データ転送サイズ
000h: 512 バイト、
001h: 1 バイト
002h: 2 バイト
| |
1FFh: 511 バイト

[bit 20-0] レジスタ No.指定
0～1F FFFF までの指定が可能です

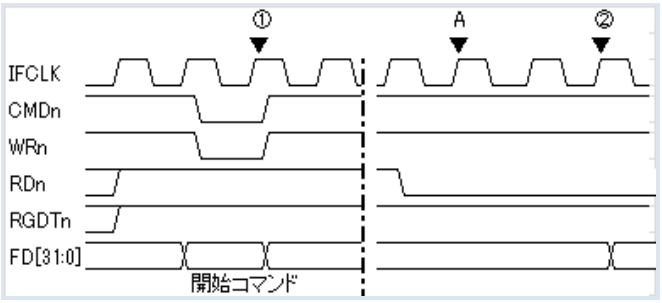
FPGA は開始コマンドを受信後、図 2 の②と③のタイミングで FPGA 内のレジスタに設定する任意のデータを受信します。WRn 信号を信号取り込みの条件として、FD[31:0]バス上のデータを FPGA 内に設定したレジスタに書き込みます。
この例では 6 バイト転送なので、7 バイト目と 8 バイト目のデータ内容は don't care です。これらのデータは FPGA で破棄します。
その他の転送サイズの場合も同様に、USB 制御 IC から開始コマンドに続き、レジスタに設定するデータが連続して FPGA に出力されます。開始コマンドとレジスタ設定データの時間間隔は規定していませんが、通常数 20～30us の時間間隔があります。また、最後に転送するレジスタ設定データと終了コマンドの時間間隔も規定していませんが、約 60～70us の時間間隔です。
終了コマンドは、FPGA で必ずデコードする必要はありません。無視することもできます。

32bit データ線 FD[31:0]のバイトオーダーは、リトルエンディアンです。詳細は mnl_Smart-USB_Sigma.pdf を参照して下さい。
6 バイト転送の例では、③のタイミングで受信したデータの上位 16bit(FD[31:16])は 7 バイト目と 8 バイト目になり、不要なデータになります。

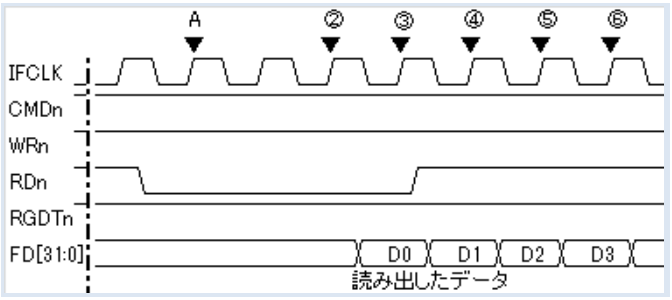
② レジスタからデータを読み出す場合（レジスタリード）

具体例 2：FPGA に構成したハードウエア・レジスタから 6 バイトの任意の値を読み出す。

- 開始コマンド内容：C0C00002（bit31= 1 ）
- （レジスタ No.2 から 6 バイトのデータを読み出す）
- 終了コマンド内容：40C00002（bit31=0）



<図 3. 6 バイト・Read アクセスタイミング>

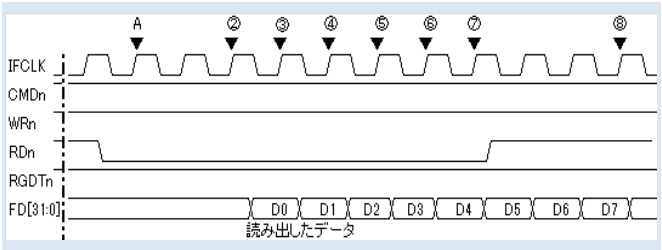


<図 3-1：6 バイト・Read アクセスの続き>

USB 制御アプリケーションが API の SUSlv_Reg_Read を発行すると、USB 制御 IC は各制御信号をアクティブにします。FPGA 側では①のタイミングで FD[31:0]バス上のデータを開始コマンドとして受信します。ここまでは、レジスタライトと同じ動作です。FPGA ではレジスタ No.やデータ転送数をデコードし、データを出力する準備を整えます。
コマンド解釈後、FPGA は"A"のタイミングで RDn 信号がアサートされたことを検出します。 ここから 2 クロック後の②のタイミングで、FPGA がレジスタに保持しているデータを FD[31:0]バス上に出力します。USB 制御 IC は③のタイミングでデータを読み込みます（リードレイテンシ=3 クロック）。以下、④、⑤のタイミングで FPGA が FD[31:0]バス上にデータを出力し、USB 制御 IC がデータを読み込みます。
ここで、実際に読み出すデータサイズは 6 バイトです。実際に読み出したいデータは、FPGA が③のタイミングで出力したデータの下位 16bit までです。しかし、仕様上、RDn 信号

は最低 4 クロック分出力されます。16 バイトのデータを読み出す分です（⑥のタイミングで最後の読み出しが完了します）。このため、FPGA が FD バス上に出力したデータ“D1”の上位 16bit から“D3”までのデータは、USB 制御 IC 側で読み出しても破棄されます。
⑥のタイミング後、レジスタライトの動作と同じように終了コマンドが発行されて、レジスタリード動作が完了します。
終了コマンドは、FPGA で必ずデコードする必要はありません。無視することもできます。

レジスタリード動作は、16 バイト（4 ワード）単位の制御になります。具体例 2 では 6 バイトでしたので、16 バイト分のデータを読み出す動作を行いました。 ここで、25 バイトのデータを読み出す場合を考えます。この場合、16 バイト 2 回の読み出し動作になり、図 4 に示す FD[31:0]のデータ“D0”から”D6”の下位 8bit までが有効なデータになります。
ここで、RDn アクティブ期間は、4 クロック単位 2 個分の 8 クロック分となります。
以下同様に、16 バイト（4 クロック）単位での読み出し動作になります。



<図 4. 24 バイトのレジスタリードアクセス>

【RefApp7 アプリケーションで運用する場合の注意点】
Smart-USB Sigma 製品と USB2.0 対応の Smart-USB Plus 製品では、共通の RefApp7 制御アプリケーションが利用できます。このため、RefApp7 で運用する場合、転送できるバイト数は 1／2／4／8 バイトのいずれかに制限されます。また、レジスタ No.指定は bit13～bit0 (16,384 個)まで有効になります。

【連続したレジスタ・アクセスの注意点】
USB バスプロトコルでは、USB ターゲット（Smart-USB Sigma 製品）から USB ホスト（PC）への割り込みができません。そのため、レジスタのポーリングにより FPGA 側の状態確認をしながら制御します。

連続したレジスタ・アクセス（ポーリング）の場合、通常は 1ms 以内に 1 回のアクセスができます。長時間ポーリングを続ける場合、Windows OS のスケジューリング等でアクセス間隔が予想外に広がる場合があります。
使用する PC 性能に依りますが、レジスタ・アクセスの間隔が広がっても対処できるシステム検討が必要です。
一般的には、数十秒以上、1ms 周期で連続したレジスタアクセスによるポーリングは行わないでシステム設計ができます。

3.3 メモリ・アクセス

大量のデータを高速にデータ転送することのできる“USB バルク転送”を利用したデータ転送方式を示します。
USB 制御 IC は、バルク転送（以下、メモリ転送）を実行することを開始コマンドで示し、FPGA がそのコマンドを解釈してデータ転送を開始します。

FD[31:0]バス上のデータの流れは以下の通りです。

開始コマンド ＋データ 0＋データ 1＋・・・
＋データ n-1＋終了コマンド

ホスト PC の制御アプリケーションでは、次の専用 API を発行することで、メモリ・アクセスを開始します。

【対応する API（USB バルク転送を実現する専用関数）】

＊ SUSlv_Data_Write
（メモリデータの書き込み動作）
＊ SUSlv_Data_Read
（メモリデータの読み出し動作）

【コマンドの内容】

Bit	31	30	29～0
内容	状態	モード	データ転送長指定

[bit 31] 転送状態表示
1 ：メモリ・アクセス(転送)の開始を表示
0 ：メモリ・アクセス(転送)の終了を表示
[bit 30] 転送モード表示
1 ：メモリ・アクセス（READ）を表示
0 ：メモリ・アクセス（WRITE）を表示

[bit [29:0]] データ転送長指定（ワード(4 バイト)単位）
開始コマンド発行後に行われるデータ転送の転送長（転送量）を FPGA に通知するためのデータです。Max.4G バイトまでの設定が可能です。
※SUSlv_Data_Read 、 SUSlv_Data_Write で指定する FileSize パラメータでは、バイト数を指定します。

3.4 メモリ・アクセス概要

CMDn 信号が Low レベルの時、RGDTn 信号が Low レベルになると、メモリ転送を行うことを示します。このとき、USB 制御 IC は FD[31:0]バス上に開始コマンドを出力します。FPGA ではこのコマンドをデコードし、メモリ転送であることと、Read 動作か Write 動作かを検出します。開始コマンド出力後、メモリリードなら RDn、メモリライトなら WRn 信号がアクティブになり、データ転送が始まります。すべてのデータを転送後、USB 制御 IC は終了コマンドを発行し、メモリ転送動作が完了します。
また、データ転送長指定ビットにより、FPGA 側にデータ転送量を通知します。FPGA ではこの値を元に適切な回路を構成すれば、実際に転送されるデータ量が正常かどうか確認することができます。

3.5 メモリライト・タイミング

（PC からボードへのデータ転送）
ウエイト制御のないメモリライト時の手順を示します。P.9 の図 6 はデータ転送全体を表しています。WRn/RGDTn が Low アクティブになっている箇所が、32K バイト単位でデータ転送を行っている部分です。その一部（WRn/RGDTn が Low アクティブになっている部分）を拡大したものが図 5 です。以下、図 5 のタイミングについて記載します。

1. **[メモリデータ転送の準備]**
USB 制御アプリケーションでデータ転送の開始アドレスを指し示すレジスタの設定を行います。これは転送ターゲットのメモリ空間を意識した設定です。メモリ空間のどのアドレスから書き込むか、レジスタ・アクセスにより、あらかじめデータ転送前に決めておく方法です。 メモリの開始アドレスが固定されていれば、アドレスの設定は不要です。
データ書き込みできる最小データ転送量は 32KB(32,768 バイト)です。
最小データ転送量は USB の接続スピードにより変化します。詳細は mnl_Smart-USB_Sigma.pdf または「3.8 最小データ転送量のまとめ」を参照して下さい。

2. **[開始コマンドの発行]**
USB 制御 IC が開始コマンドを発行します。コマンド内容は、[B:10xxx...x]です。29bit 以下にはデータ転送量の設定値を含みます。FPGA は①のタイミングでコマンドを解釈します。

3. **[WAIT 制御有無の確認]**
②のタイミングで FPGA がウェイト要求しているかどうか確認します（ウェイトの詳細は後述の 3.7 項を参照してください）。FPGA が出力する WAITn 信号をこのタイミングで High レベルにすれば、USB 制御 IC はウェイトしません。Low レベルにすると、WRn、RDn、RGDTn の制御線をアクティブにしません。
(注)①と②のタイミングは規定していません。
通常、数 ms 以下です。

4. **[データ転送開始]**
FPGA がウェイト要求していないので、USB 制御 IC は③のタイミングで WRn、RGDTn 信号を Low レベルに変化させ、FD[31:0]データバス上にメモリ転送データを出力します。FPGA は④のタイミングからデ

ータを取り込みます。

5．**[データ転送]**

④のタイミング以降、USB 制御 IC は 32K バイト単位でデータ転送するので、WRn、RGDTn 信号も PCLK が 8,192 クロックごとに Low レベルに変化します。

6．**[データ転送完了]**

32K バイト単位ですべてのデータ転送が完了すると、USB 制御 IC は 2 項と同様のタイミングで終了コマンドを発行し、データ転送処理を完了します。コマンド内容は、「B:00xx...xxxx」です。FPGA 回路では、このコマンドを無視することもできます。

3.6 メモリリード・タイミング

（ボードから PC へのデータ転送）

ウエイト制御のないメモリリード時の手順を示します。P.10 の図 7 はデータ転送全体を表しています。RDn/RGDTn が Low アクティブになっている箇所が、4K バイト単位でデータ転送を行っている部分です。その一部を拡大したものが図 8 です。以下、図 8 のタイミングについて記載します。

1．**[メモリデータ転送の準備]**

USB 制御アプリケーションでデータ転送の開始アドレスを指し示すレジスタの設定を行います。これは転送ターゲットのメモリ空間を意識した設定です。メモリ空間のどのアドレスから読み出すか、レジスタ・アクセスにより、あらかじめデータ転送前に決めておく方法です。 メモリの開始アドレスが固定されていれば、アドレスの設定は不要です。

データ読み出しできる最小データ転送量は 4KB(4,096 バイト)です。

最小データ転送量は USB の接続スピードにより変化します。詳細は mnl_Smart-USB_Sigma.pdf または「3.8 最小データ転送量のまとめ」を参照して下さい。

2．**[開始コマンドの発行]**

USB 制御 IC が開始コマンドを発行します。コマンド内容は、[B: 1100...0011]です。29bit 以下にはデータ転送量の設定値を含みます。FPGA は①のタイミングでコマンドを解釈します。

3．**[WAIT 制御有無の確認]**

②のタイミングで FPGA がウェイト要求しているかどうか確認します（ウェイトの詳細は 3.7 項を参照してください）。FPGA が出力する WAITn 信号をこのタイミングで High レベルにすれば、USB 制御 IC はウェイトしません。Low レベルにすると、WRn、RDn、RGDTn の制御線をアクティブにしません。

(注) ①と②のタイミングは規定していません。通常 1ms 以下です。

4．**[データ転送開始]**

FPGA がウェイト要求していないので、USB 制御 IC は③のタイミングで RDn、RGDTn 信号を Low レベルにします。その 2 クロック後、④のタイミングで FPGA は FD バス上に転送データを出力し、USB 制御 IC は⑤のタイミングでデータを取り込みます。

5．**[データ転送]**

メモリライト時と同様に、③のタイミング以降 USB 制御 IC は 4K バイト単位でデータを扱うので RDn、RGDTn 信号も PCLK が 1,024 クロックごとに Low レベルに変化します。

また、リードレイテンシが 3 なので、RDn/RGDTn が⑥のタイミングで High になっても、USB 制御 IC にとっては、⑦のタイミングで受信するデータが 4K バイトの最後のデータになります。

6．**[データ転送完了]**

4K バイト単位ですべてのデータ転送が完了すると、USB 制御 IC は 2 項と同様のタイミングで終了コマンドを発行し、データ転送処理を完了します。コマンド内容は、[B:01xx...xxxx]です。29bit 以下の数値は don't care です。

FPGA 回路では、このコマンドを無視することもできます。

3.6.1 データを読み出す場合の注意点

USB 制御 IC は、制御アプリケーション上で設定したデータ転送量設定値(転送バイト数)よりも 65,536 バイト(64KB)多くボードからデータを読み出します。

USB 制御 IC ではデータ転送量を管理していないので、設定した転送量の最後のデータを転送完了後、USB 制御 IC はデータ転送完了を認識せずに内蔵 FIFO にデータを書き込む動作をするためです。このため、終了コマンド発行前に FPGA から 64KB 分の余分なデータを読み出す動作を行います。この読み出しデータは、USB 制御 IC が終了コマンドを受信したときにクリアされるので、ホスト PC にはデータ転送されません。

以下、ホスト PC からメモリリード関数が実行されたときの USB 制御 IC の動作を説明します。

1. ホスト PC がメモリリードを実行。
2. USB 制御 IC がメモリリード開始コマンドをホスト PC から受け取る。
3. USB 制御 IC は 4,096 バイト単位でデータを FPGA から読み出す。
4. USB 制御 IC がホスト PC に設定量の最後のデータを送信すると、内蔵バッファに空きができるので、データ転送量を管理していない USB 制御 IC は、FPGA に対してデータ読み出しを継続します。これが 64KB です。
5. メモリリード関数で設定したバイト数を転送し終わると、通常 1ms 以下のタイミングでホスト PC が USB 制御 IC にデータ転送が終了したことを示す終了コマンドを発行します。このコマンドにより、USB 制御 IC のバッファ(64KB)がクリアされてデータ転送が終了します。

【なぜ、余分なデータを読み出すのか？】

設定バイト数のデータ転送が完了すると、USB 制御 IC のバッファは空になります。この時点で USB 制御 IC はデータ転送が完了したかどうかわかりません。ホスト PC から終了コマンドを受信していないためです。このため、USB 制御 IC は FPGA から 64K バイト分のデータを設定バイト数以外に読み出してしまいます。図 7-1 を参照してください。

64K バイト余計に読み出してしまったデータは、ホスト PC から発行された終了コマンドにより、クリアされます。

例えば、1M バイトのメモリリードを行う場合、実際に読み出されるデータは 1M+64K バイト分のデータになります。

問題点：

FPGA に実装した回路構成が、FIFO からデータを読み出して、USB 制御 IC に転送するような方式やメモリリード後にクリアされるメモリ構成の場合、意図するデータに加えて 64K バイトの余分なデータが読み出されるため、データが消失し、システムの誤動作を招きます。

対策：

FPGA 側でデータ転送量を管理することにより、この問題点を回避することができます。 メモリ転送開始コマンドは、転送設定量を 30 ビットで指定できます。 PCLK をカウンタクロックとし、RDn 信号をイネーブルとしてデータ数をカウントすることで、最後の 64K バイト分の読み出し要求を無視することができます。 3.9 章を参照して下さい。

3.7 メモリ・アクセス時の WAIT 制御

FPGA が USB 制御 IC に対して出力する WAITn 信号を使ってウエイト制御ができます。ホスト PC からボードへデータ転送する場合（メモリライト）、ボードがホスト PC からのデータを受け取れない時には、ウエイト制御することでデータ転送を一時的に停止し、データを失うことなくデータ転送することができます。ボードからホスト PC へのデータ転送（メモリリード）の場合も同様です。

USB 制御 IC は、P.11 の図 9（メモリリード）で示す②のタイミングで WAITn 信号の Low/High レベルの検出を行います。High レベルの時、その後の WAITn 信号レベルは don't care です。また、ウエイトするタイミングは、②で示すように、WRn/RDn/RGDTn の各信号レベルが High レベルの時だけです。すなわち、実際にデータ転送が行われている時にウエイト制御はできません。 4K バイト単位のデータ転送が終わった時点（図 6,7 の青矢印で示した区間）で、USB 制御 IC は FPGA からの WAIT 要求の有無を確認します。ウエイト

要求がなければ、引き続きデータ転送を行います。

図 9-1 で示すように、WAITn=Low レベルの間、ホスト PC からボードに対してデータの読み出しが行われません。
メモリライトの場合も同様です。

このように、WAITn 信号を利用したウエイト制御は、ボード外に低速なメモリを接続している場合や、外部システムからのデータ転送速度が USB の実効データ転送速度よりも低速な場合に便利な機能です。

3.8 最小転送データ量のまとめ

Smart-USB Sigma 製品の USB インタフェースでは、5Gbps の Super Speed（SS）、480Mbps の High Speed（HS）、12Mbps の Full Speed（FS）に対応しています。
これら接続スピードにより、転送できるデータ量の最小値が異なります。また、メモリリード時に余分に読み出すデータ量も変わります。
FPGA 設計時、USB 接続スピードを意識せず、メモリリード 4KB、メモリ WR32KB として設計することを推奨します。

接続速度	SS	HS	FS
最小データ転送量（WR）	32KB	16KB	2KB
最小データ転送量（RD）	4KB	2KB	256B
余分に読み出すデータ量	64KB	32KB	4KB

【 最小データ転送量未満のデータ転送を行いたい場合 】
接続スピードが SS で、実効データが 24KB バイトの場合でも、データ転送量を 32KB に設定すれば、転送することができます。 FPGA 回路では 32K バイト分のデータが送られてくる動作になりますが、先頭から 24K バイトまでの実効データだけ受信できるような回路にすることで対処できます。
原則として、メモリ・アクセスでは数 MB 以上の単位でデータ転送することを想定しています。1MB 未満の小さなデータの転送では、データ転送時のオーバーヘッド処理がデータ量に比べて相対的に大きくなり、非効率になります。

3.9 RefApp7 利用時のメモリコマンド

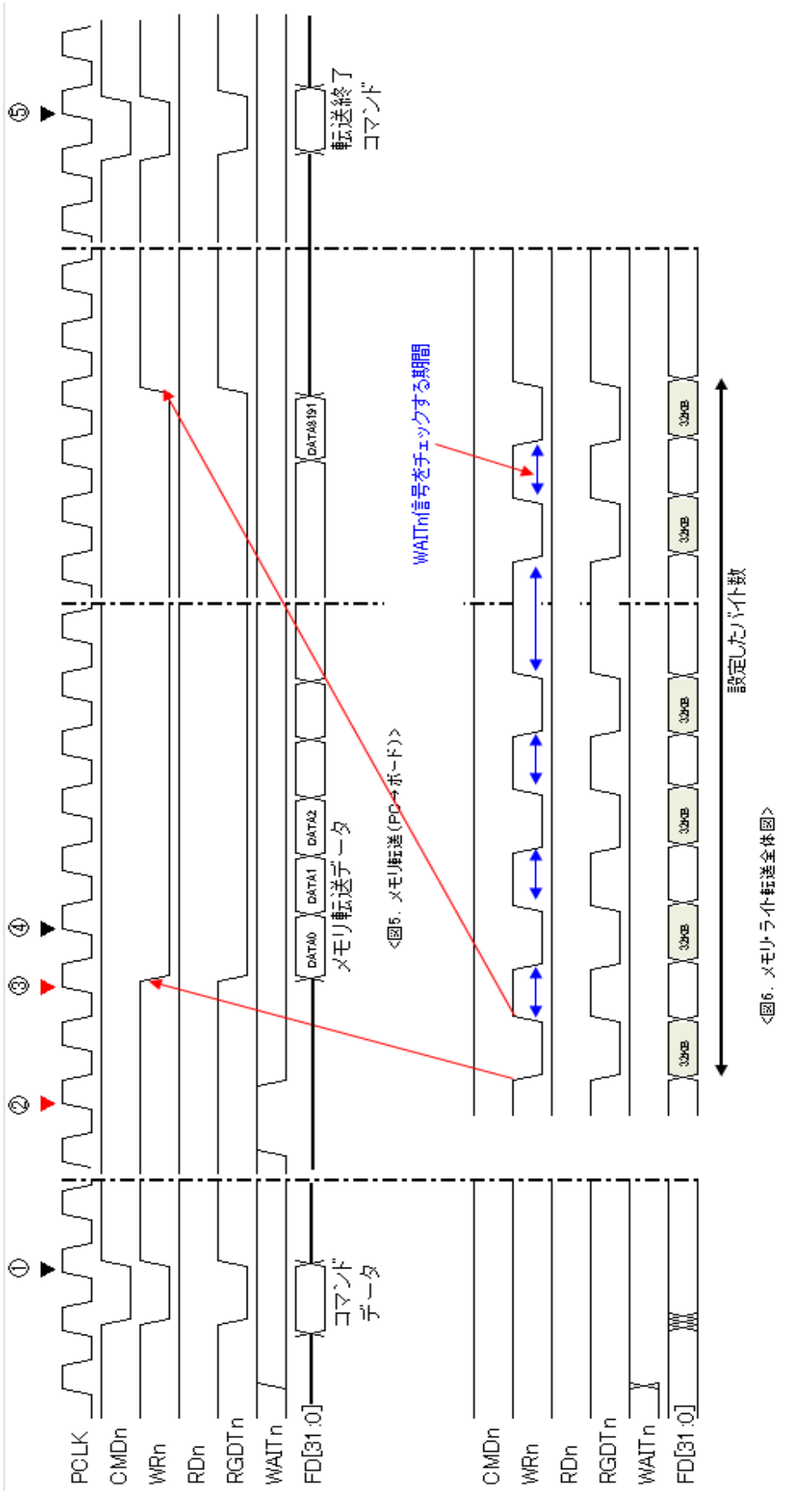
RefApp7 を利用してボード制御する場合、「メモリ操作」タブで、ツールバーのオプション→「Sigma モード専用のメモリコマンドを使用する」にチェックを入れると、転送開始コマンドの bit29～bit0 に転送レングス指定した値が入ります。
FPGA ではこの設定値を元に、転送データ量をチェックできるので、メモリリード動作を行う際に、余分なデータ出力をさせない回路設計が可能です。
また、設定したデータ量を転送した後に発生する余分なデータを読み出すシーケンスは、通常のデータリード動作と同じです。このときに waitn 信号が Low レベルの場合、データ転送動作が完了しなくなります。このため、設定したデータ量を転送した後は waitn 信号も強制的に High レベルにしてください。

3.10 小回りのきくデータ・アクセス

これまで記述したメモリ・アクセスは、API の SUSlv_Data_Read、SUSlv_Data_Write の内容です。これらの API を分解できる API を用意しています。

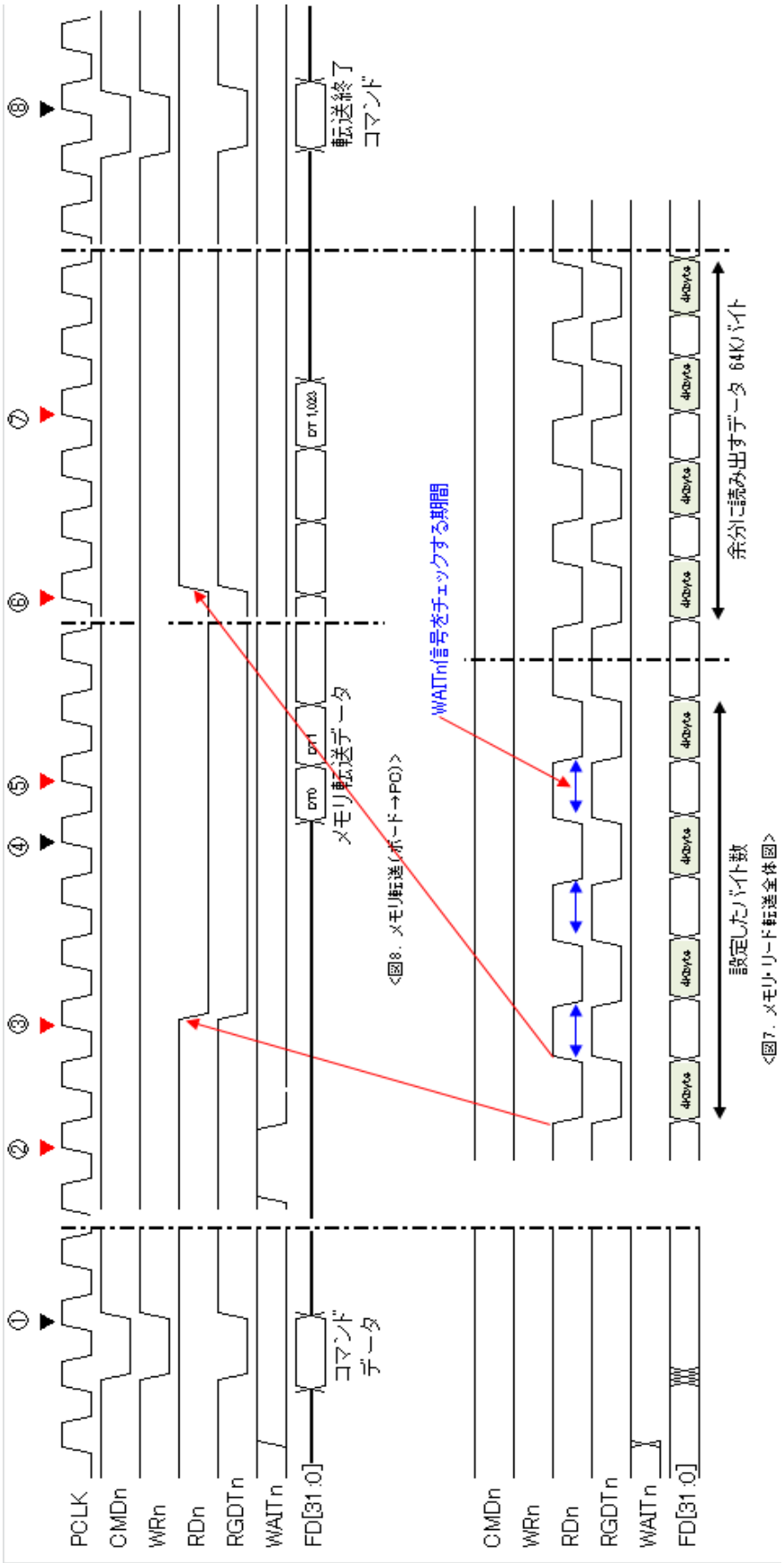
・SUSlv_Mem_Cmd：メモリコマンド
・SUSlv_Bulk_Read：データリード
・SUSlv_Bulk_Write：データライト

[SUSlv_Data_Read] の内容は、[SUSlv_Mem_Cmd] + [SUS_Bulk_Read]+[SUSlv_Mem_Cmd] と同じです。
[SUSlv_Data_Read] を何度も繰り返す場合、[SUSlv_Mem_Cmd]部がオーバーヘッドになります。
[SUSlv_Mem_Cmd]に挟まれた[SUS_Bulk_Read]は何度も繰り返すことができます。
周期的にデータ転送する様な場合に利用できます。
[SUSlv_Bulk_Write]も同様です。



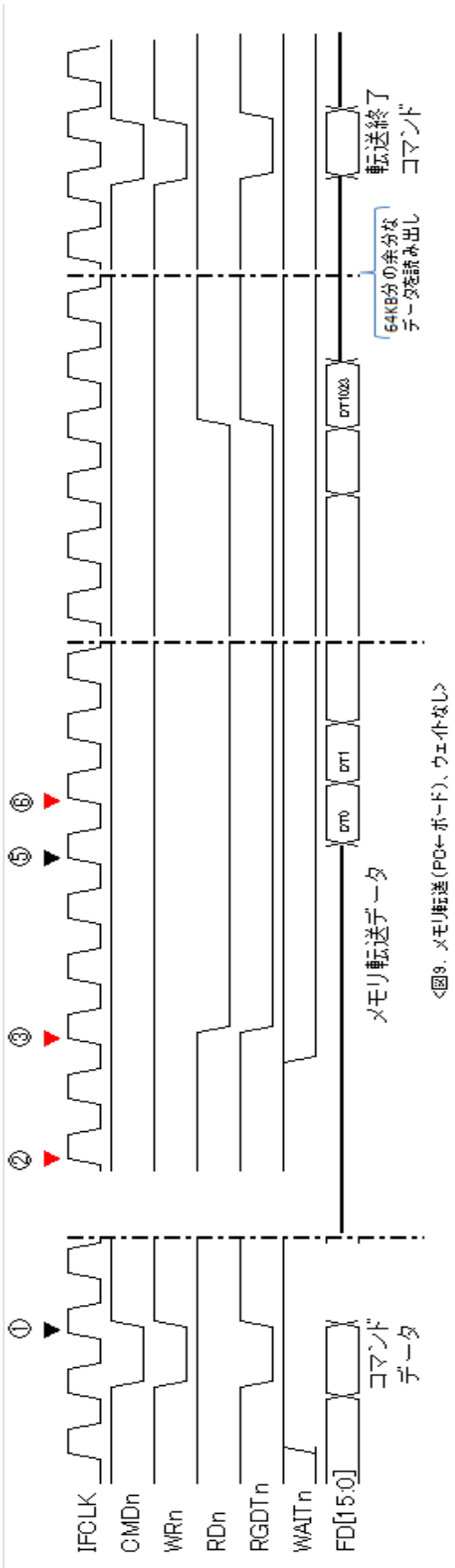
<図5. メモリ転送(PCイボード)>

<図6. メモリライト転送全体図>

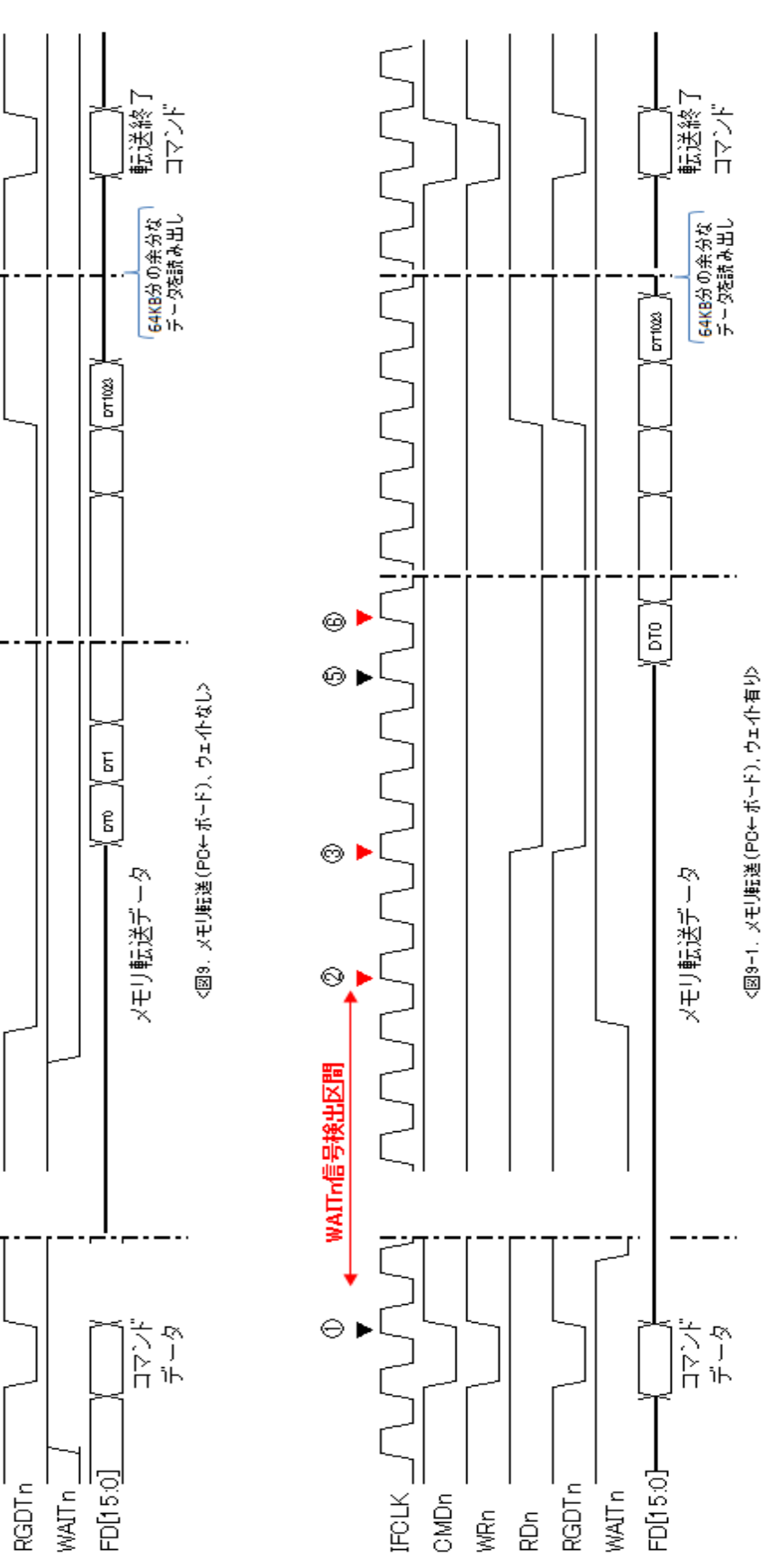


<図8. メモリ転送(ボード→PC)>

<図7. メモリ・リード転送全体図>



<図9. メモリ転送(PC←ボード)、ウェイトなし>



<図9-1. メモリ転送(PC←ボード)、ウェイト有り>