SiTCP 説明書

第 3.0 版

2021 年 05 月 25 日

内田智久

Electronics system group, IPNS, KEK

加筆修正：(株)Bee Beans Technologies

1. 履歴

修正日	版数	内容
2011/01/18	1.0	初版制定
2011/01/19	1.1	誤字修正
2011/04/14	1.2	内部レジスタの詳細説明を追加
2012/10/24	1.3	「10. SiTCP 内部レジスタ / 詳細手順」に補足を追加
2012/11/12	1.4	ユーザーI/F 信号 入出力 誤記修正 (TCP_CLOSE_REQ, TCP_CLOSE_ACK)
2012/11/21	1.5	TCP_CLOSE_REQ/ACK に関する記述を追加
2014/08/11	1.6	図 6 内の誤字訂正
2016/04/18	2.0	データ受信の FIFO 容量の条件追加
以後、メンテナンスとして(株)Bee Beans Technologies が加筆修正		
2021/05/25	3.0	「2. はじめに」 ● 新規追加、以降の章番号を繰り上げ 「10. SiTCP 内部レジスタ / 詳細手順」 ● 内部レジスタマップへの追記（+0x10 bit6-3、+0x3C-3F） 「12. 参考文献」 ● SiTCP ホームページ、BBT ホームページの URL を削除

2. はじめに

本文書は、内田智久博士が作成した「SiTCP 説明書」を(株)Bee Beans Technologies（以下 BBT）が加筆修正したものです。SiTCP は内田智久博士が開発し、BBT が管理・配布を行っているネットワーク・プロセッサです。

3. 内容

ネットワーク・プロセッサ(SiTCP)の紹介と使い方の概要を説明した文書です。SiTCP ライブラリを使った具体的な実装方法については本文書を読んだ後に別文書「SiTCP ライブラリ」を読んでください。

4. SiTCP とは

SiTCP は物理実験のフロントエンドをイーサネットで PC に接続する技術です。

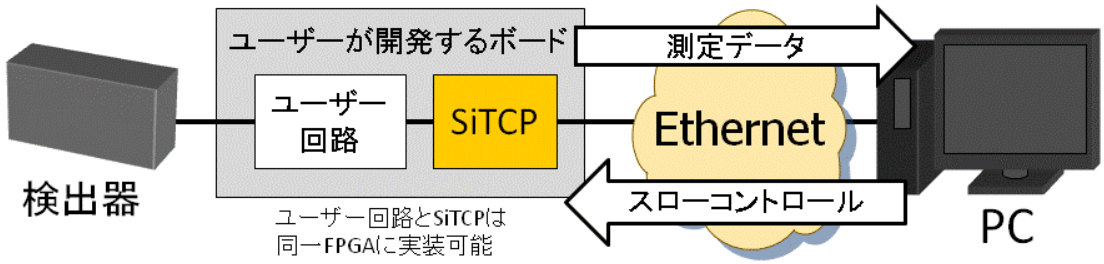


図 1　標準的な SiTCP 使用例

実験でエレクトロニクスを開発する場合、検出器に近い読み出し回路を開発する機会が多いです。それは、検出器に近い部分に使用できる汎用機器が無いからです。検出器のデータは何らかの方法で PC へ転送しなければいけません。SiTCP を用いる事でイーサネットをデータ転送に使用する事が出来るようになります。

SiTCP を用いて測定データを PC へ転送する場合、同期 FIFO に書き込む手順でデータを SiTCP に書き込めばデータが転送されます。SiTCP はユーザー回路と PC 間のデータパイプを提供する技術と言えます。SiTCP にデータを書き込めば、そのデータが PC に現れます。

PC からユーザー回路のスローコントロールは遠隔バス制御です。SiTCP はユーザー回路制御用に VME バスの様な単純なプロトコルで動作するバスを持っています。PC から予め決められたパケット・フォーマットを持つ UDP パケットを SiTCP に送る事でバス信号が発生しユーザー回路が制御されます。読み込み書き込みをサポートしているのでユーザー回路に簡単な回路を付け加える事でスローコントロールを行う事が出来ます。

5. イーサネット接続の利点

イーサネットを利用する利点として以下を挙げる事が出来ます。

- PC 側デバイスドライバを書かなくてよい
  - 多くの場合、標準で OS に実装されている

- 標準関数のみでプログラムが書ける
  - ソケットプログラミング
  - 多くの OS がソケット関数をサポートしている
- 柔軟なシステム構成をバーチャルに構築可能
  - HUB に接続しておけばソフトウエアのみで構成変更可能
- 長距離通信可能
- 特にイーサネットは以下の利点がある
  - 異なる技術世代間の高い接続性（互換性）
  - 技術発展に応じた技術進化（最新は 40G/100Gbps）
  - 安価で多種多様な市販製品
  - 複数メディア（光、UTP など）

6. 特徴

SiTCP は以下の特徴を持っています。

- ハードウエアによる TCP/IP/Ethernet 通信
  - 10Mbps～1Gbps Ethernet
  - TCP 上限値で安定した高速通信
  - Slow control 機能（UDP 使用）
- 容易な実装
  - 小さい回路規模
  - FPGA ライブラリ (Xilinx)として提供
  - 使いやすい容易なユーザーI/F

7. 標準的な実装例

SiTCP はXilinx社のライブラリとして提供されているので通常はFPGAに実装します。イーサネットポートとして標準的なツイストケーブル(UTP)を採用する場合と光ケーブルを採用する場合の例を下に示します。

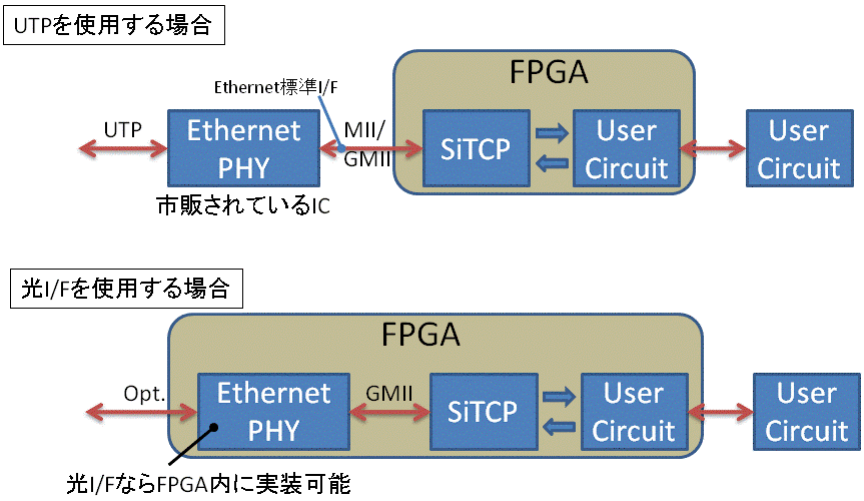


図 2　標準実装例

標準的なツイストケーブルを使用する例が上、光ケーブルを使用する例が下に示されています。

ツイストケーブルを使用する場合、イーサネットに接続する為の I/F チップが必要になります。SiTCP は標準 I/F である MII/GMII を採用しているので各社から販売されている標準チップを使用する事が出来ます。SiTCP ライブラリを使用する場合は実績のあるデバイスが奨励されていますので特に問題が無い場合は奨励デバイスを使用してください。奨励デバイス以外を使用する場合、PHY デバイスのレジスタ初期設定回路を新たに製作する必要があります。

SiTCP を動作させる為に必要な部品とユーザーI/F を下図に示します。

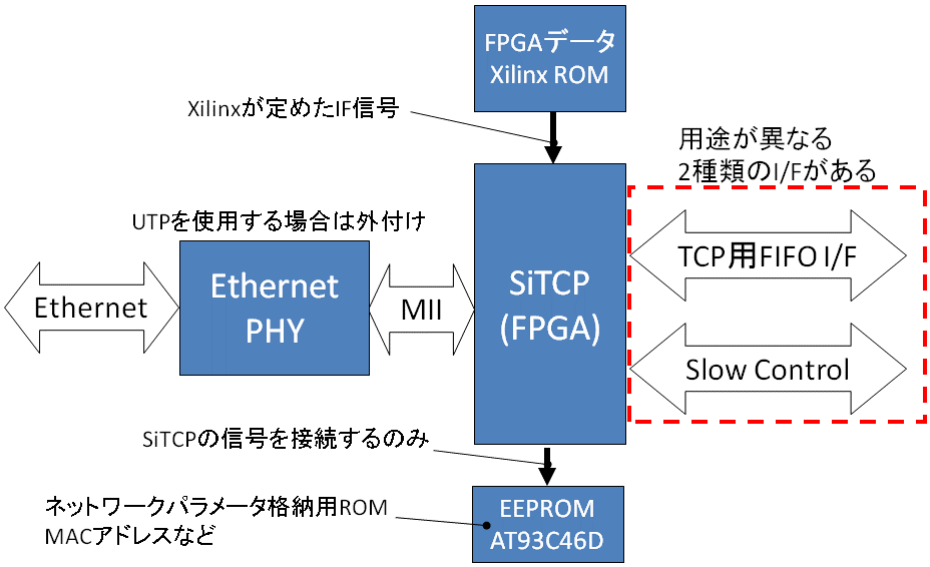


図 3　必要部品とユーザーI/F

SiTCP を動作させる為に必要な部品は下の 4 つです。

1. Xilinx 社 FPGA：SiTCP を実装する為の FPGA
2. FPGA 用 ROM：FPGA データを格納する為の ROM
3. EEPROM：MAC アドレスなどのネットワークパラメータを格納する為の ROM
4. Ethernet PHY デバイス：UTP を採用する場合に必要

1 と 2 について、SiTCP を実装する為に必要です。FPGA 容量は余裕を持たせて決めてください。SiTCP のサイズはライブラリを合成する事で分かりますので事前に合成して容量を見積もってから FPGA 容量を選択するようにしてください。

3 について、SiTCP のネットワークパラメータを格納する為の ROM です。IP アドレス、MAC アドレス、TCP 動作パラメータなど多くのパラメータが格納されます。SiTCP は起動後に一度だけこの ROM からパラメータを読み出し回路に設定します。製造直後や異常値が書き込まれてしまった時に対応する為に、標準的な値であるデフォルト値を強制的に読み込ませるモードも持っています。このモードは通常 EEPROM の内容を書き換える時やライブラリの評価をする時に使用します。

4 について、UTP を使用する時に必須です。SiTCP は標準 I/F である MII/GMII を採用しているので各社から販売されている標準チップを使用する事が出来ます。SiTCP ライブラリでは実績あるデバイスが奨励されています。問題が無い場合は奨励デバイスを使用してください。奨励デバイス以外を使用する場合、PHY デバイスのレジスタ初期設

定回路を新たに製作する必要があります。光 I/F 用 PHY デバイスが組み込まれた製品もありますので、光 I/F を使用する時は必ずしも外部 PHY デバイスは必要ではありません。

SiTCP とユーザー回路を接続するユーザーI/F は 2 種類あります。データ転送用に使用する TCP FIFO I/F とレジスタ制御などに使用するスローコントロール I/F です。ユーザーが回路開発の非専門家である事を想定して開発されていますので簡単なインタフェースを採用しています。データ転送用 TCP FIFO I/F は同期 FIFO メモリに似た I/F ですので送りたいデータを順番に SiTCP に書き込めば自動的に PC にデータ転送されます。スローコントロール I/F はアドレス信号、データ信号、リードライト信号などから構成される単純なバスアクセスプロトコルを採用した制御 I/F です。同期回路ですので VME バスなどの標準バスシステムよりも簡単に設計する事が出来ます。ユーザーI/F については後の章で詳細に解説していますので参考にしてください。

8. I/F 信号

以下の項目毎に SiTCP ライブラリの I/F 信号を説明します。

- システム I/F
- MII/GMII
- ユーザーI/F
- ネットワークパラメータ設定 I/F

システム I/F

以下の SiTCP 全体の動作に関わるシステム I/F があります。

信号名	I/O	説明
CLK	I	システムクロック 130MHz 以上を奨励 イーサネットを 100Mbps 以下で動作させる時は 15MHz 以上を奨励
RST	I	システムリセット
TIM_1US	I	1us 周期パルス(*1)
TIM_1MS	I	1ms 周期パルス(*1)
TIM_1S	I	1 秒周期パルス(*1)
TIM_1M	I	1 分周期パルス(*1)
EEPROM_CS	O	AT93C46D の CS 端子に接続
EEPROM_SK	O	AT93C46D の SK 端子に接続
EEPROM_DI	O	AT93C46D の DI 端子に接続
EEPROM_DO	I	AT93C46D の DO 端子に接続
SiTCP_RST	O	SiTCP 内部リセット出力(*2)

全ての信号はシステムクロック CLK の同期信号

(*1) システムクロック 1 パルス幅 H になる周期信号、全ての TIM*信号の位相を合わせる事。

(*2)SiTCP がリセット状態の時に出力されるリセット信号。外部回路のリセットとして使用可能。SiTCP 内部はシステムリセット RST を内部回路で延長しているのでシステムリセットが解除された後も一定時間リセットされ続けます。

MII/GMII

MII/GMII インタフェースを SiTCP に接続する時に幾つか注意点があります。

MII は 100Mbps までの Ethernet で使用される規格で GMII は Gigabit Ethernet で使用する規格です。この 2 つの規格は良く似ていますが幾つか異なる点があります。共通点が多い為に PHY 信号は兼用されています。Ethernet のスピードに合わせて MII として動作するのか、GMII として動作するのか切り替えなければいけません。

GigaSiTCP を MII 専用で使用する事も出来ます。

I/F 信号

以下に MII/GMII に関係する信号を示します。

信号名	I/O	説明
MODE_GMII	I	GMII 対応 PHY を使用する時は 1、 MII のみ対応 PHY を使用する時は 0 を設定
GMII_1000M	I	PHY デバイスの SPEED LED から生成（注 1）
GMII_RSTn	O	PHY デバイスのリセットに接続
GTX_CLK	O	SiTCP の信号ではないが GMII 接続時に必要 PHY デバイスの GTX_CLK に接続 FPGA 内で 125MHz を生成する（注 2）
GMII_TX_CLK	I	MII 動作の時は PHY デバイスの TX_CLK を入力、GMII 動作の時は FPGA で生成した GTX_CLK と同じ 125MHz を入力
GMII_TX_EN	O	PHY デバイスの TX_EN に接続
GMII_TXD	O	PHY デバイスの TXD に接続
GMII_TX_ER	O	PHY デバイスの TX_ER に接続
GMII_RX_CLK	I	PHY デバイスの RX_CLK に接続
GMII_RX_DV	I	PHY デバイスの RX_DV に接続
GMII_RXD	I	PHY デバイスの RXD に接続
GMII_RX_ER	I	PHY デバイスの RX_ER に接続
GMII_CRS	I	PHY デバイスの RX_CRS に接続
GMII_COL	I	PHY デバイスの RX_COL に接続
GMII_MDC	O	PHY デバイスの MDC に接続

GMII_MDIO_IN	I	PHY デバイスの MDIO に接続
GMII_MDIO_OUT	O	3 ステートバッファを介して PHY デバイスの MDIO に接続
GMII_MDIO_OE	O	GMII_MDIO_OUT の 3 ステートバッファ制御信号

（注 1）MII/GMII 動作を切り替える信号。PHY が出力しているスピードを表示する LED から生成する。1 で Gigabit Ethernet(GMII)、0 で 100Mbps Ethernet(MII)。LED の場合論理が逆になっている事があるので注意する。

（注 2）Gigabit Ethernet 時の送信クロック。125MHz を PHY に向けて FPGA が出力する。MII 使用時には送信クロックは PHY が出力するが GMII では FPGA が送信クロックを出力しなければいけない。

GIGABIT ETHERNET 対応 PHY デバイス使用時の接続

Gigabit Ethernet 対応 PHY デバイスを使用する時の SiTCP と MII/GMII の接続を下図に示します。

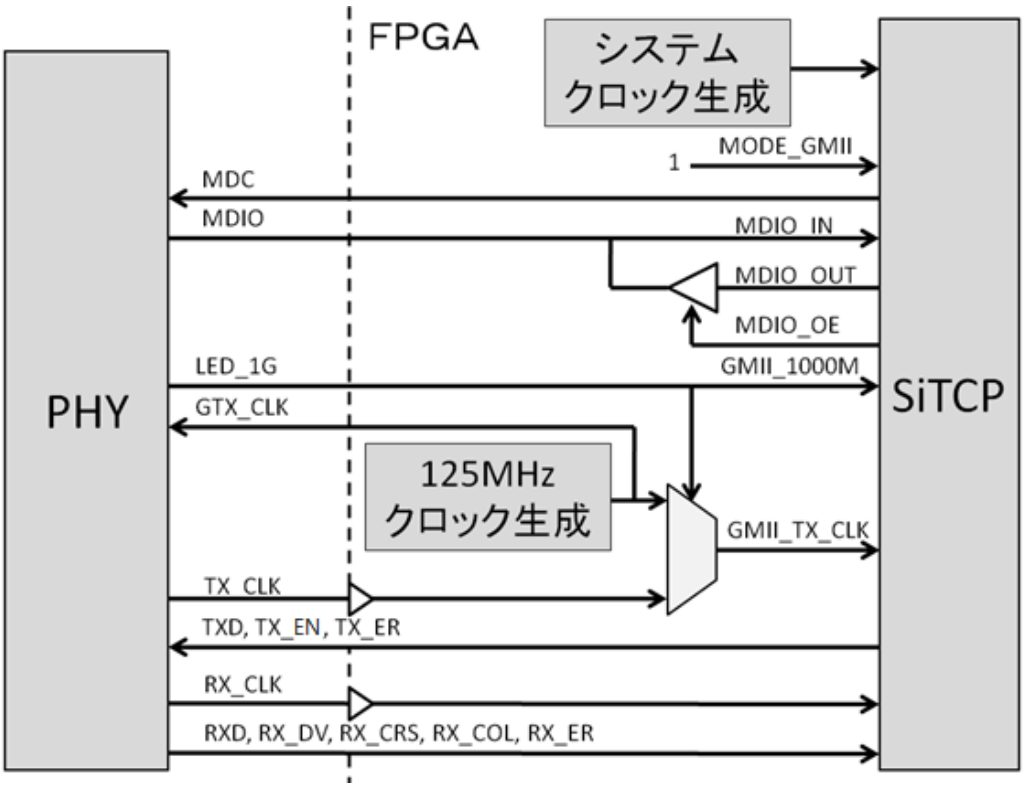


図 4　SiTCP と MII/GMII の接続(Gigabit Ethernet 対応)

以下に注意点を挙げます。

- TX_CLK, RX_CLK は FPGA のグローバル･クロック入力ピンに接続してください。
- MDIO は 3 ステートバッファを使用して接続してください。
- GMII_TX_CLK を切り替える為にプリミティブ BUFGMUX または同等のプリミティブを使用してください。

GIGABIT ETHERNET 未対応 PHY デバイス使用時の接続

Gigabit Ethernet 未対応 PHY デバイスを使用する時の SiTCP と MII の接続を下図に示します。

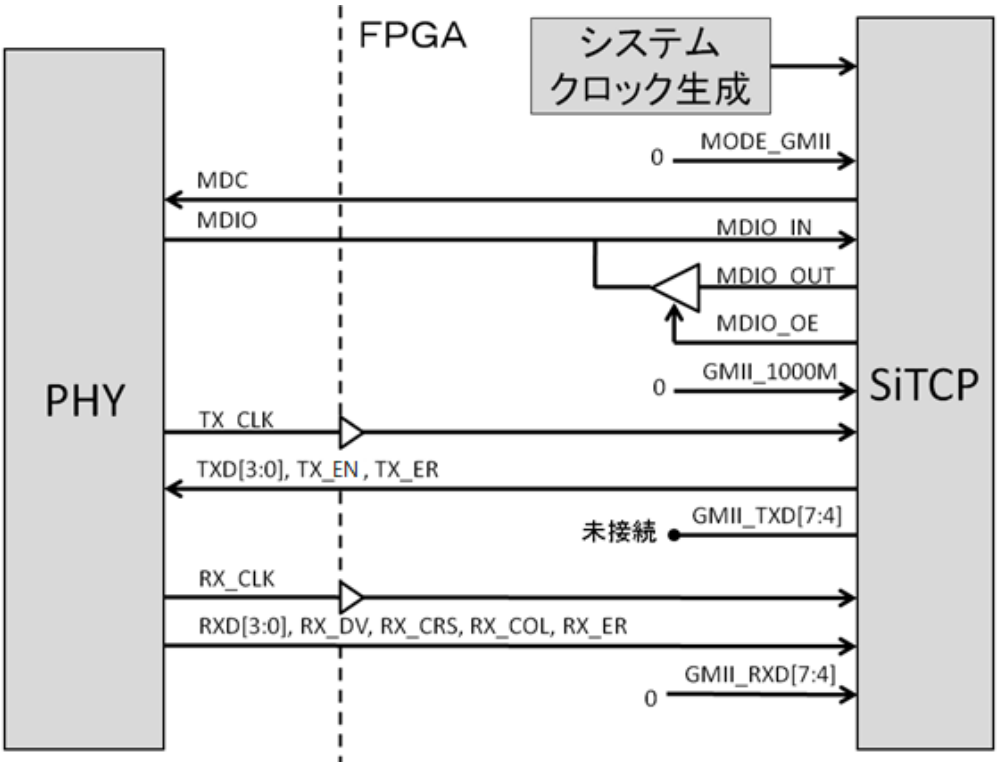


図 5　SiTCP と MII/GMII の接続(Gigabit Ethernet 未対応)

以下に注意点を挙げます。

- TX_CLK, RX_CLK は FPGA のグローバル・クロック入力ピンに接続してください
- MDIO は 3 ステートバッファを使用して接続してください
- GMII_1000M は 0 を入力してください
- GMII_TXD[7:4]は使用しませんので未接続のままとしてください
- GMII_RXD[7:4]は使用しませんので 0 を入力してください

ユーザーI/F

SiTCP は以下 2 つのユーザーI/F を持っています。

- TCP I/F（データ転送用）
- スローコントロール I/F (レジスタアクセスなど制御用)

TCP I/F

以下の特徴があります。

- PC とのデータ転送に使用
- 高速転送が可能
- 全ての信号はシステムクロックに同期
- 全ての信号が正論理
- 同期 FIFO に似た I/F
- 送受信独立
- TCP コネクションが確立すると使用可能状態になる

TCP I/F は独立送受信可能です。受信機能を使用する場合には SiTCP とユーザー回路の間に FIFO メモリを挿入する必要があります。PC からユーザー回路へのデータ転送が不要なら TCP 受信機能は不要ですが、必要であっても次章で説明するスローコントロール機能で十分なら TCP 受信機能ではなくスローコントロール機能を使用してください。出来るだけ受信機能を使用しない方が良い理由として、ソケットプログラミングを用いて高速処理可能な送受信同時制御は難しい事、FIFO メモリを追加する必要がある事を挙げる事が出来ます。

I/F 信号

I/F 信号を下に示します。I/O は SiTCP からみた入出力です。SiTCP へ与える信号が I、SiTCP が出力する信号が O です。

信号名	I/O	説明
TCP_OPEN_ACK	O	通信可能状態を示す
TCP_CLOSE_REQ	O	PC からの通信終了要求を受信した事を示す
TCP_CLOSE_ACK	I	通信終了応答
TCP_TX_DATA[7:0]	I	送信バイトデータ
TCP_TX_WR	I	送信データ書き込みイネーブル
TCP_TX_FULL	O	送信バッファ・フルフラグ
TCP_RX_DATA[7:0]	O	受信データ
TCP_RX_WR	O	受信データ有効
TCP_RX_WC[15:0]	I	受信バッファに格納されているデータバイト数

通信状態制御

TCP 通信は送受信する 2 者がお互いにデータの転送状況を通知し合いながら確実にデータを転送します。

最初に相手が通信できるかどうかを確認します。確認が出来たら（TCP コネクションの確立）データ転送を開始します。TCP コネクションの確立はソケットプログラミングでの connect 関数が正常に終了した事に相当します。

通信を終了したくなった場合、相手に通信停止要求を発行します。相手から通信終了応答を受信して通信が終了します。

以上の様に TCP 通信は TCP コネクションが確立している期間に通信を行わなければいけません。また、PC から通信終了要求を受信した場合には速やかにデータ転送を終了して通信終了応答を送信しなければいけません。SiTCP は以上の TCP 規約に従って通信する為に制御信号を持っています。下に TCP コネクションが確立してから終了するまでの制御信号のタイムチャートを示します。

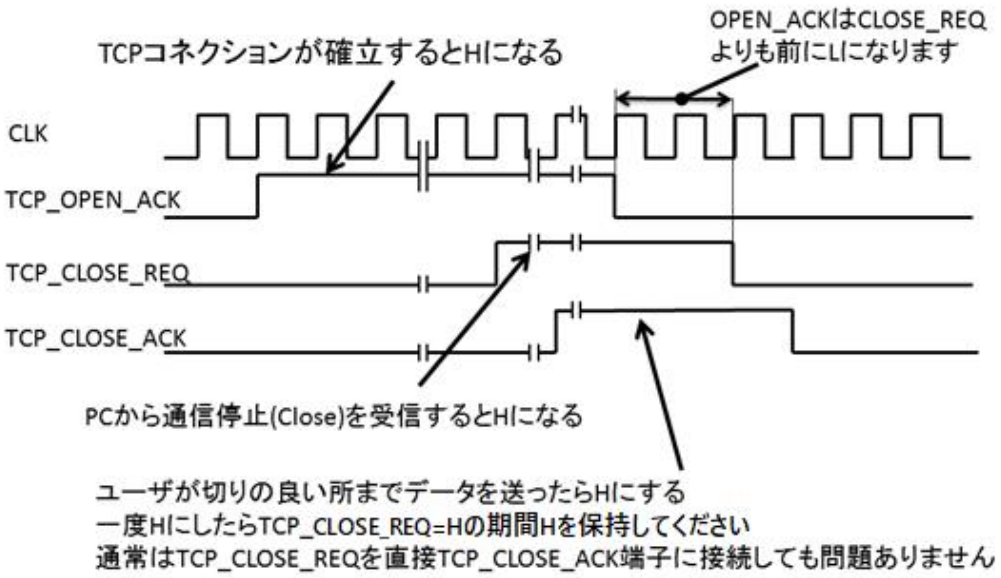


図 6　TCP 制御信号のタイムチャート

TCP 通信を行って良い期間は TCP_OPEN_ACK が H であり TCP_CLOSE_REQ が L の期間内です。TCP_CLOSE_REQ が H になったらユーザー回路は切りの良い所までデータ書き込みを行い、その後に TCP_CLOSE_ACK を H にして終了応答してください。SiTCP は送信データを全て送り終えた時点でコネクション終了確認を送ります。

実際使用する際は TCP_CLOSE_REQ を受信した時点で既に PC はデータ受信を終了していますので、その後に送ったデータを PC は破棄します。従って、多くの場合は TCP_CLOSE_REQ と TCP_CLOSE_ACK を直接接続し TCP_CLOSE_REQ=H になった後にユーザー回路ができるだけ早く送信を終了すれば問題なく動作します。仮に TCP_CLOSE_REQ=H になったのちに TCP_CLOSE_ACK=H として応答した状態で送信データを書き込み続けると SiTCP は PC へデータを送信し続けます。従って、この状態では TCP_CLOSE_ACK=H であってもコネクションは閉じません。
TCP_CLOSE_REQ = H になったら必ず送信を止めてください。

TCP_CLOSE_REQ が H になっている期間は TCP_CLOSE_ACK を H に保持してください。

以上の様に TCP_OPEN_ACK 信号は PC がデータ受信可能状態と同等な信号です。PC が受信可能で無い場合にユーザー回路を動作させたくない場合などはユーザー回路の負論理リセット信号として TCP_OPEN_ACK を使用する事が出来ます。

TCP_OPEN_ACK は TCP_CLOSE_REQ よりも早く L になりますので、
TCP_OPEN_ACK をリセットとして使用する場合には TCP_CLOSE_REQ が H になっている期間に TCP_CLOSE_ACK が L にならないように十分注意して設計してください。

データ送信

TCP コネクションが確立した通信可能状態の時のみデータを送信する事が出来ます。送信データを書き込む時に注意する事は送信バッファが一杯になった時に書き込み動作を停止しなければいけない事です。下に送信データ書き込み時のタイムチャートを示します。

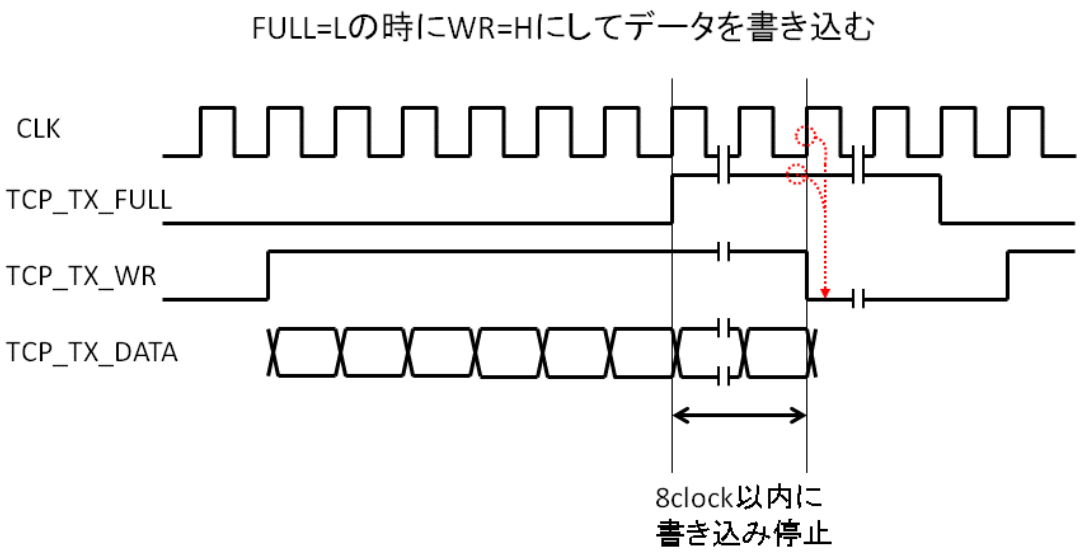


図 7　送信データ書き込みタイムチャート

同期 FIFO メモリと同じインタフェースです。

送信データの書き込みは TCP_TX_WR と TCP_TX_DATA[7:0]を使用します。システムクロックに同期させ送信データを TCP_TX_DATA に置き TCP_TX_WR=H に設定すると送信されます。同期回路ですので CLK の立ち上がりエッヂ毎に 1 バイトのデータと見なされます。

TCP コネクション確立状態であり TCP_TX_FULL=L の時にデータを書き込む事が出来ます。途中で TCP_TX_FULL=H になったら 8 クロック以内に送信動作を停止して TCP_TX_WR=L にしてください。TCP_TX_FULL=L になったら送信動作を再開する事が出来ます。TCP コネクションが確立されている間に以上を繰り返してください。

TCP 受信機能を使用する場合、受信バッファ用の FIFO をユーザー回路と SiTCP 間に置いてください。SiTCP の TCP データ受信信号は Xilinx 社 Core Generator で生成した FIFO に直接接続できるように設定されています。

Core Generator により FIFO メモリを生成する場合以下の様に設定してください。

- 書き込みデータビット幅=8bit
- Word count オプション有効、Word count bit 幅を最大
- FIFO 容量の条件を満たす事
  - 65536 ≥ FIFO 容量（バイト）> 2×(SiTCP の TCP MSS + 58)
  - SiTCP の MSS デフォルト値は 1460（10 章　内部レジスタマップ参照）

下にデータ受信時のタイムチャートを示します。

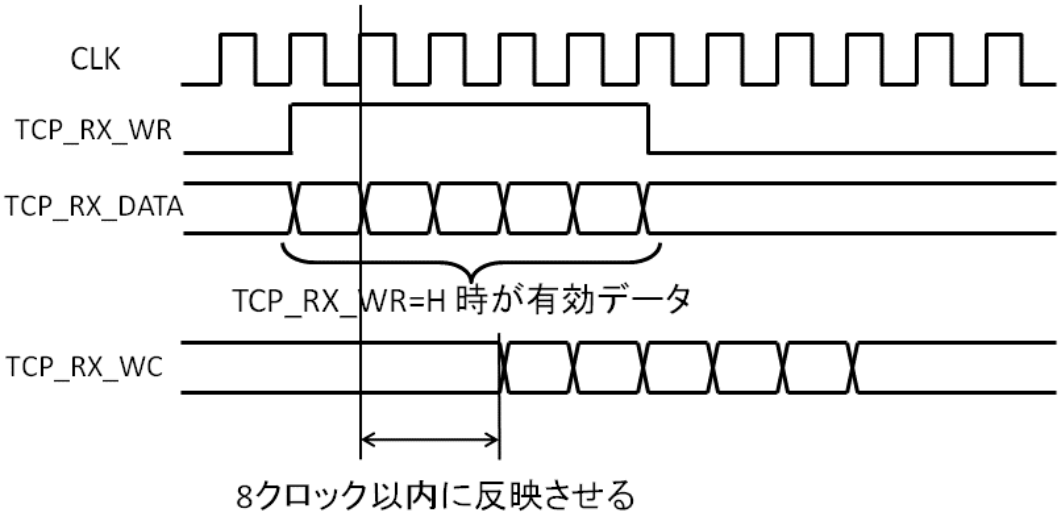


図 8　データ受信タイムチャート

SiTCP はデータを受信すると TCP_RX_DATA[7:0]に受信データを受信順に置き、TCP_RX_WR=H にして有効な受信データを受信した事を通知します。
TCP_RX_DATA[7:0]を同期 FIFO の書き込みデータ端子に、TCP_RX_WR を書き込みイネーブル端子にそれぞれ接続してください。

TCP では受信者が送信者に何バイト受信可能かを常に通知しなければいけませんので、TCP_RX_WC[15:0]により受信バッファに何バイトデータが格納されているのかデータを受け取ってから 8 クロック以内に通知してください。TCP_RX_WC は同期 FIFO の Word count 端子に下位ビットから接続してください。未使用ビットは全て 1 を設定

してください。例えば、受信 FIFO が 2Kbyte の容量の時、Word count 端子は 11bit になります。従って TCP_RX_WC[10:0]=Word count[10:0]を接続し、TCP_RX_WC[15:11]の全てのビットに 1 を設定してください。Verilog HDL は TCP_RX_WC[15:11] = 5'b11111;となります。

TCP 受信機能を使用しない場合は次の節を参照してください。

TCP 受信機能を使用しない場合

TCP 受信機能を使用しない場合は TCP_RX_WC[15:0] に 0 を設定してください。他の値を設定しても動作しますが 0 設定を奨励します。

スローコントロール I/F

UDP パケットを使用して制御します。ここではユーザー回路から見た動作と使い方を説明します。PC 側の制御については次章で説明します。

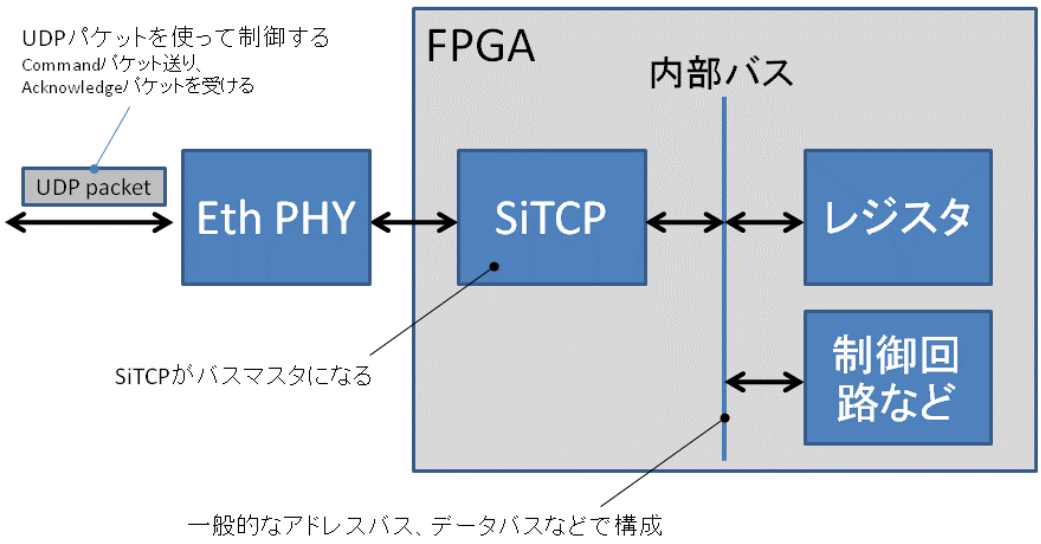


図 9　スローコントロール概念図

上図にスローコントロールの概念図を示します。PC から UDP パケットを SiTCP に送信して SiTCP がバスマスタになっている内部バスを制御します。内部バスはアドレス、データ、制御信号で構成されていて、単純なバスアクセスプロトコルを採用しています。PC がデータをユーザー回路に書き込む時は、UDP パケットにアドレスおよび書き込みデータを格納して SiTCP に転送すると、SiTCP 内のバスマスタが動作して内部バスを

経由後ユーザー回路にデータを書き込みます。PC がデータを読み込む場合はアドレスを指定する事で SiTCP がユーザー回路から読み込んだデータを PC へ送り返します。

アドレス空間に関する注意

RBCP アドレス 0xFFFF0000 から 0xFFFFFFFF は SiTCP 内部レジスタ用に予約されていますので使用できません。上記のアドレス以外を使用してください。

制御信号

信号名	I/O	説明
CLK	I	システムクロック TCP IF の CLK と同一信号
RBCT_ACT	O	バスが動作している事を示す
RBCT_ADDR[31:0]	O	アクセス中のアドレス
RBCT_WE	O	ライトイネーブル
RBCT_WD[7:0]	O	ライトデータ
RBCT_RE	O	リードイネーブル
RBCT_RD[7:0]	I	リードデータ
RBCT_ACK	I	アクセス応答

書き込み（PC からユーザー回路へ）

PC からユーザー回路へデータを書き込む場合は SiTCP から RBCP_ADDR にアクセスするアドレス、書き込みデータが設定されると同時に RBCP_WE=H として書き込み動作である事をユーザー回路に通知します。

ユーザー回路は該当するアドレスを受信した時には書き込み動作を行い、書き込み終了後に RBCP_ACK を H にする事で応答しなければいけません。該当回路が無いアドレスをアクセスした場合などは RBCP_ACK が L のままになりますが、待ち時間が約 100msec を超えた時点で SiTCP はエラーと判断してバスサイクルを終了します。PC へはエラーを通知するパケットを送ります。

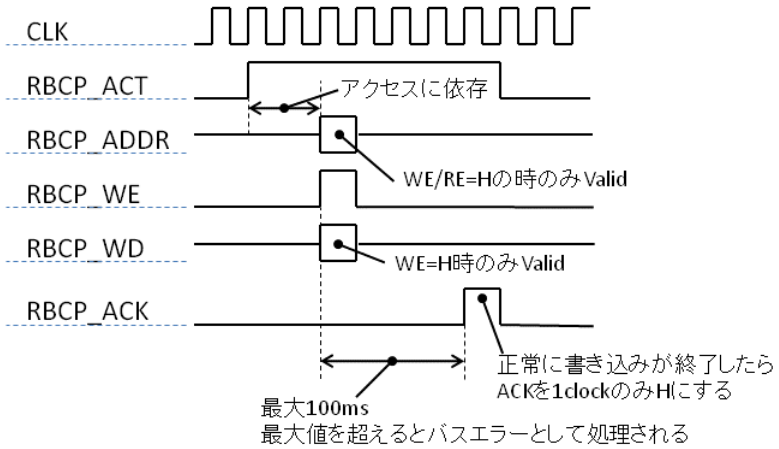


図 10　スローコントロールでのデータ書き込み

上図にデータ書き込みでのタイムチャートを示します。

読み込み（ユーザー回路から PC へ）

ユーザー回路から PC がデータを読み込む場合、SiTCP は RBCP_ADDR にアクセスするアドレスを設定すると同時に RBCP_RE=H として読み出し動作である事をユーザー回路に通知します。

ユーザー回路は該当するアドレスのデータを RBCP_RD[7:0]に設定すると同時に RBCP_ACK を 1 クロックのみ H にして応答しなければいけません。読み込みの場合も ACK 応答時間が約 100msec を超えると SiTCP はエラーと判断してバスサイクルを終了し PC へエラーを通知するパケットを送ります。

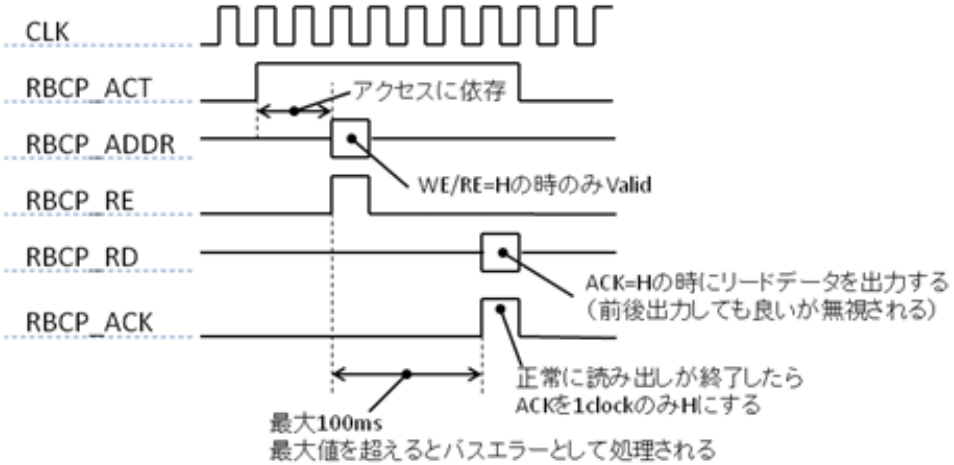


図 11　スローコントロールでのデータ読み込み

上図にデータ読み出しのタイムチャートを示します。

ネットワークパラメータ設定 I/F

IPアドレスなどのネットワークパラメータはEEPROMに格納された値を用いて動作しますが、DIP SW など外部入力信号でも使用する事が出来ます。また、EEPROM を書き換える時や異常時に強制的にデフォルト値を読み込ませて動作させる端子などもあります。

信号名	I/O	説明
FORCE_DEFAULTn (*1)	I	0: 強制的にデフォルト値を使用 1: EEPROM の値を使用
IP_ADDR_IN (*2)	I	IP アドレス入力端子
IP_ADDR_DEFAULT	O	レジスタに設定された IP アドレス値
TCP_MAIN_PORT_IN (*2)	I	TCP ポート番号入力端子（メインポート）
TCP_MAIN_PORT_DEFAULT	O	レジスタに設定された TCP ポート番号（メインポート）
TCP_SUB_PORT_IN (*2)	I	TCP ポート番号入力端子（サブポート）
TCP_SUB_PORT_DEFAULT	O	レジスタに設定された TCP ポート番号（サブポート）
RBCP_PORT_IN (*2)	I	RBCP ポート番号入力端子
RBCP_PORT_DEFAULT	O	レジスタに設定された RBCP ポート番号
PHY_ADDR (*3)	I	PHY デバイスの MIF アドレスを設定

(*1) 通常は DIP_SW やジャンパーピンなどを接続する。

(*2) 通常は_DEFAULT が末尾に付いた同信号名と接続してください。ネットワークパラメータを外部信号（例えば DIP SW など）で設定できるようにする時は次節「DIP SW など外部信号によりネットワークパラメータを設定する方法」を参照してください。

(*3) PHY デバイスのデータシートを読んで設定してください。多くの場合、電源投入時に PHY デバイス端子の状態により決定するようになっています。PHY デバイスのアドレスの設定方法はデバイスにより異なります。

DIP SW など外部信号によりネットワークパラメータを設定する方法

以下のネットワークパラメータは DIP SW などを用いて外部信号により設定する事も出来ます。

- IP アドレス
- TCP メインポート番号
- TCP サブポート番号
- RBCP ポート番号

以下で IP アドレスについて説明しますが、他の値も同様です。

SiTCP は IP_ADDR_IN に入力されている値を常に IP アドレスとして使用して動作します。FORCE_DEFAULTn の値に依存しません。IP アドレスを DIP SW で設定したい場合は DIP SW を IP_ADDR_IN に接続してください。

IP_ADDR_DEFAULT は IP アドレスに該当するレジスタ値が出力されています。FORCE_DEFAULTn =0 の時はデフォルト値が出力されています。FORCE_DEFAULTn =1 の時は EEPROM から読み出した値が設定されています。この IP_ADDR_DEFAULT の一部のビットと DIP SW と組み合わせて IP_ADDR_IN に入力する事も出来ます。例えば上位ビットはレジスタ値、下位ビットは DIP SW 値を設定可能です。

9. RBCP

スローコントロールの制御プロトコルである Remote Bus Control Protocol (RBCP)について説明します。

RBCP は SiTCP がバスマスタになるバスを遠隔で制御する方法を提供するプロトコルです。UDP パケットにカプセル化された RBCP パケットを使用します。

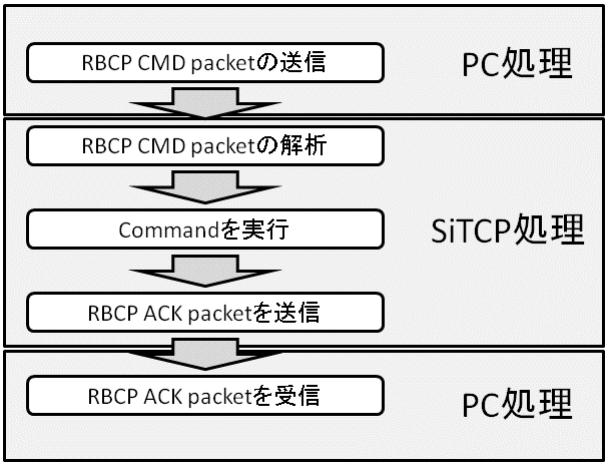


図 12　RBCP 処理の流れ

上図に処理の流れを示します。先ず、PC から処理内容に合わせて生成したコマンド・パケットを SiTCP へ送ります。この時の SiTCP 待ちポートは EEPROM に格納された値かデフォルト値である 4660 です。コマンド・パケットは SiTCP により処理され内部バスアクセスが実行されます。バスアクセスが終了すると応答として ACK パケットを PC に送り返します。PC は ACK パケットを受け取った時点でアクセスが終了したと認識する事が出来ます。

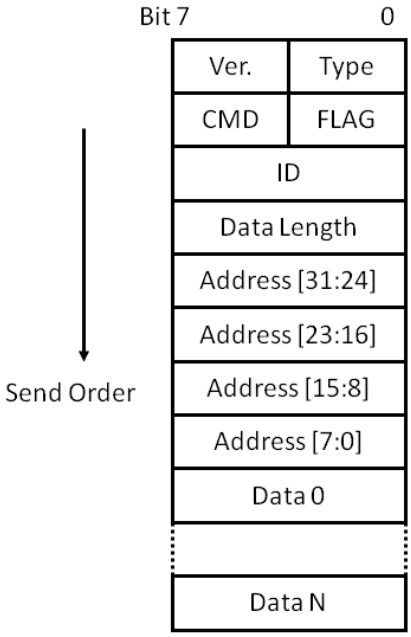
RBCP パケットは UDP を用いて転送していますのでネットワーク内で損失する可能性があります。PC は必ず ACK パケットを受け取りアクセスが終了した事を確認した後に次の処理を行ってください。また、PC は ACK パケットを待つ間にタイマを設定しタイムアウトしたら再度コマンド・パケットを送信してアクセスを確実に終了させてください。タイムアウト時間は SiTCP のバスタイムアウトが 100ms に設定されている事を考慮すると 200msc 以上の値を設定する方が良いです。PC 処理時間やネットワーク内の転送時間に依存しますので実際に動作させて確認してください。動作が遅くても良い場合はタイムアウトを 1sec 程度にしてください。実際の使用経験から、パケットが損失する事は珍しく、多くの場合は正常に ACK パケットが戻ってきます。

RBCP の具体的な実装は付録のサンプルプログラム「RBCP_sample.zip」を参照してください。Cygwin および Linux で動作確認されています。

RBCP パケット・フォーマット

RBCP パケットは UDP でカプセル化します。

ソケットプログラミングでは下に示す RBCP パケットを send 関数により送信してください。



CMD bit	Name	Description
3	Access	Bus Access
2	R/W	0: Write, 1:Read
1	Reserve	Always 0
0	Reserve	Always 0

FLAG bit	Name	Description
3	REQ/ACK	0: Request, 1: Acknowledge
2	Reserve	Always 0
1	Reserve	Always 0
0	Error	0: Normal, 1: Bus Error

図 13　RBCP パケット・フォーマット

上図にパケット・フォーマットを示します。以下で各フィールドの意味を説明します。

VER

RBCP パケットのバージョンです。現在は 15(0xF)のみ受け付けます。必ず 15 に設定してください。他の値の場合パケットは破棄されます。

TYPE

Packet type です。現在は 15(0xF)のみ受け付けます。必ず 15 に設定してください。他の値の場合パケットは破棄されます。

CMD

アクセスの種類を示すコードです。現在使用しているのは次の 2 つの値です

- 0xc: Read
- 0x8: Write

図の右側に掲載されている CMD bit も参照してください。

FLAG

パケットの種類と実行結果を示します。ACK packet のみに有効なフィールドです。コマンド・パケットでは必ず bit3=0 を設定してください。バスエラーが発生すると bit0 が 1 に設定された ACK パケットが戻ってきます。

図の右側に掲載されている Flag bit も参照してください。

ID

パケット ID です。実行確認用に使用します。

ACK パケットは対応するコマンド・パケットと同じ ID を持っています。SiTCP はこのフィールドを確認・操作しませんので任意の値を設定してください。

この ID を毎回変化させればどのコマンド・パケットに対応した ACK パケットかを判別できるようになります。

DATA LENGTH

コマンド・パケットの時：Read / Write するデータ長をバイト数で設定します。0 を設定すると動作しませんので 0 以外の長さを設定してください。

ACK パケットの時：実際に正常実行したデータ長です。エラーにより途中で中断した時には正常に実行されたデータ数が格納されています。エラー発生時はこの値から何処のアドレスでエラーが発生したかを知る事が出来ます。

8bit 長フィールドなので一度にアクセスできるバイト長は 255 バイトに制限されています。255 バイトを超えるアクセスを行う時は複数回のアクセスに分割して 1 アクセスのデータ長が 255 以下になるようにしてください。

ADDRESS

アクセスするバスのアドレスです。このフィールドに格納されているアドレスを開始アドレスとして Data Length フィールドに格納されているデータ長をアクセスします。

DATA

コマンド・パケットと ACK パケットでは意味が異なります。

コマンド・パケット

- ライト時：Data Length に相当するバイト長のデータを格納してください。もし、Data Length フィールドの値より少ないデータを送信した場合、内部メモリに残っているデータが書き込まれますので書き込まれる値が不定になります。
- リード時：このフィールドは使用しないでください。

ACK パケット

- ライト時：実際にライトしたデータが格納されています。ただし、相当するアドレスをリードしたものではありません。あくまでライト時に使用したデータです。例えば読み出し専用レジスタに書き込んだ場合、実際は書き込みできませんが書き込んだデータがこのフィールドに格納されます。
- リード時：読み出したデータが格納されています。

RBCP サンプルプログラム

RBCP プログラムの実装例としてデバック用に使用しているサンプルプログラムを付録「RBCP_sample.zip」として配布します。

rbcp.c を gcc などでコンパイルして使用してください。Linux で動作確認してあります。他の OS の場合は必要個所を修正してください。

Windows 用としてコンパイル済み実行ファイル“rbcpWin.exe”が格納されています。Window のコマンドプロンプトから実行してください。

サンプルプログラムの起動画面を下に示します。

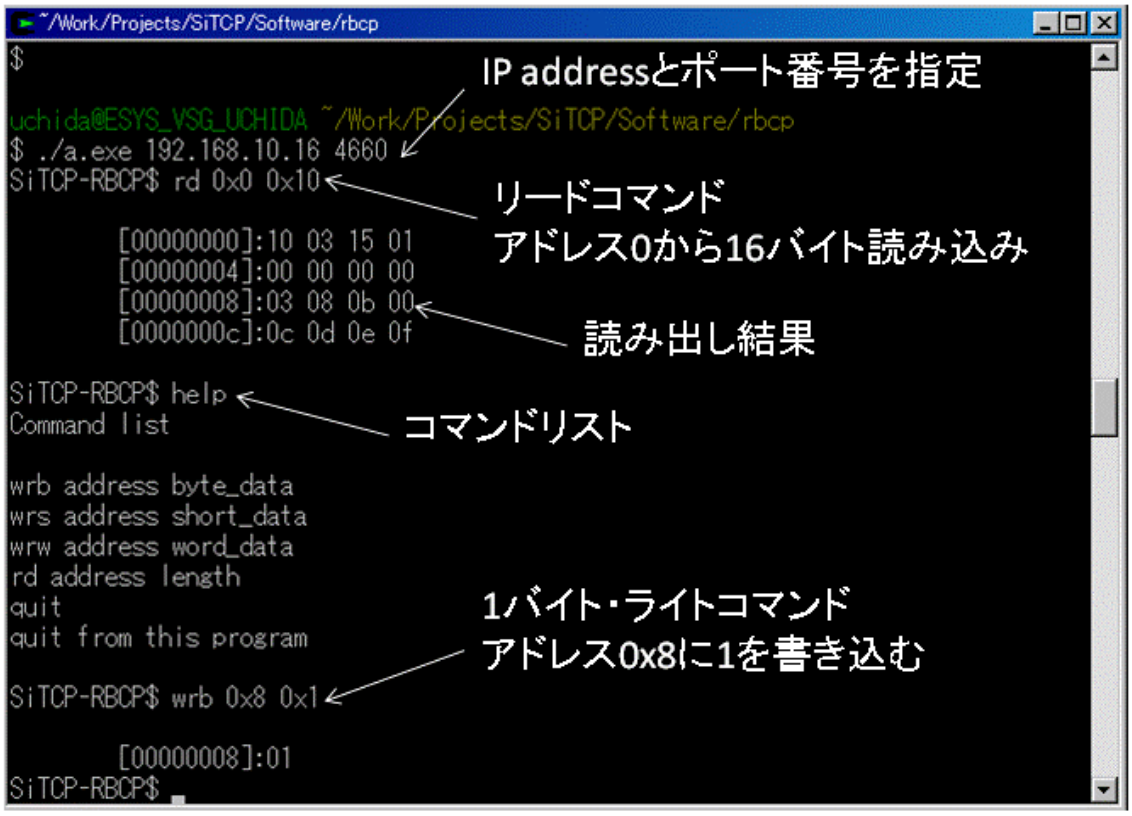


図 14　サンプルプログラム起動画面

実行方法は、

実行ファイル名 [IP アドレス]　[ポート番号]

図では IP アドレス 192.168.10.16、ポート番号 4660 の場合を示しています。起動するとプロンプトが SiTCP-RBCP＄に変わります。入力する数字表現は 10 進または 16 進表現です。16 進表現を使用する時は先頭に 0x を付けてください。以下に各コマンドを説明します。

- wrb [address] [write-data]
  - 1 バイト書き込み
  - 指定したアドレスに write-data を書き込む
- wrs [address] [write-data]
  - 2 バイト書き込み、ビックエンディアン
- wrw [address] [write-data]
  - 4 バイト書き込み、ビックエンディアン

- rd [address] [length]
  - 先頭アドレス[address]から[length] バイト長の読み込みおよび表示
- help
  - 使用できるコマンドリスト
- quit
  - プログラムを終了

テスト例

- DIP SW 値の読み込み
  - アドレス 0 から 16 バイト読んでください
  - DIP SW を動かして値を読み直してください
    - アドレス 0x8 の値が DIP SW の値に対応して変わります
- LED 点灯消灯
  - アドレス 0x7 に 0xFF を書き込んでください
    - 7 つの LED が点灯します
  - アドレス 0x7 に 0x0 を書き込んでください
    - 7 つの LED が消灯します

10. SITCP 内部レジスタ

SiTCP はネットワークパラメータ格納や PHY 制御の為に内部レジスタを持っています。RBCP を使用してアクセスする事が出来ます。

アドレス空間マップ

RBCP アドレス	説明
0xFFFF_0000 – 0xFFFF_FBFF	Reserved
0xFFFF_FC00 – 0xFFFF_FCFF	EEPROM
0xFFFF_FD00 – 0xFFFF_FDFF	Reserved
0xFFFF_FE00 – 0xFFFF_FEFF	Ethernet PHY MIF I/F
0xFFFF_FF00 – 0xFFFF_FFFF	内部レジスタ、SiTCP 制御レジスタ

Reserved 空間は絶対にアクセスしないで下さい。

内部レジスタマップ

ビックエンディアンでアクセスしてください。表中のアドレスはベースアドレス 0xFFFFFF00 からの差分で表示しています。

アドレス	R/W		標準値
+0x00-03	R	SiTCP 合成日時：YY MM DD NN※（2 進化 10 進表示） ※Y=西暦下二桁、M=月、D=日、N=同一日の合成回数	
+0x04-0F	R	ID：特定の値が格納されている	
+0x10	R/W	SiTCP and TCP control [7] SiTCP reset (1: Reset / 0: Normal) [6] Client ARP (1: ON / 0: OFF) [5] Dup Ack (1: ON / 0: OFF) [4] MIF Initializer (1: ON / 0: OFF) [3] MAC flow control (1: ON / 0: OFF) [2] Keep alive packet (1: ON / 0: OFF) [1] Fast retrains. (1: ON / 0: OFF) [0] Nagle buffering (1: ON / 0: OFF)	0x01

+0x11	R/W	Reserved	
+0x12-17	R/W	MAC address (*1)	
+0x18-1B	R/W	IP address	
+0x1C-1D	R/W	TCP ポート番号（main port）	0x18
+0x1E-1F	R/W	TCP ポート番号（alternative port）	0x17
+0x20-21	R/W	TCP MSS (バイト数)	0x05B4
+0x22-23	R/W	RBCP ポート番号	0x1234
+0x24-25	R/W	TCP キープアライブパケット送信タイマ（送信バッファにデータがある時）（msec 単位）	0x03E8
+0x26-27	R/W	TCP キープアライブパケット送信タイマ（送信バッファが空の時）（msec 単位）	0xEA60
+0x28-29	R/W	TCP オープン時のタイムアウト時間（msec 単位）	0x1388
+0x2A-2B	R/W	TCP クローズ時のタイムアウト時間（256msec 単位）	0x2BF2
+0x2C-2D	R/W	TCP コネクション-コネクション間の待機時間（msec 単位）	0x01F4
+0x2E-2F	R/W	TCP 再送時間タイムアウト時間（msec 単位）	0x01F4
+0x30-3B	R/W	Reserved	0x0
+0x3C-3F	R/W	ユーザー領域 (*2)	0x00000000
+0x40-FF		アクセス禁止領域 (*1)	

（*1）MAC アドレス（+0x12-17）およびアクセス禁止領域（+0x40-FF）に値を書き込むと正常に動作しなくなりますので十分に注意して作業してください

（*2）任意の値を書き込むことが可能です。

EEPROM の書き換え

以下に IP アドレスの変更を例にネットワークパラメータの変更方法を説明します。

IP アドレス変更は EEPROM に格納されている SiTCP パラメータを変更する事で行います。この作業により不適切なパラメータを EEPROM に書き込んでしまうと、SiTCP が正常に動作しなくなり通信できなくなってしまいますので十分に注意して作業してください。

EEPROM に格納されているパラメータのメモリマップは前節「内部レジスタマップ」でベースアドレスを 0xFFFFFC00 と置き換えたものです（+0x00 から+0x0F を除く）。例えば IP アドレスは 0xFFFFFC18-1B に格納されています。

作業手順概要

SiTCP をデフォルト値強制設定モード（EEPROM の内容に関わらず常に同じパラメータで動作するモード）で立ち上げ、下の固定アドレスで通信できるようにする。

- IP アドレス：192.168.10.16
- RBCP ポート番号：4660

その後、RBCP により EEPROM 内に格納されている IP アドレスを書き換える。

モードを通常に戻して電源を再投入する。

詳細手順

1. デフォルトモードでの立ち上げ
   A) 電源を切る
   B) デフォルト値強制設定モードにする（FORCE_DEFAULTn=0）。通常はジャンパ、DIPSW により設定する場合が多い。
   C) 電源を投入
   D) Ping コマンドによりで動作している事を確認する(*1)
2. EEPROM の書き換え
   A) RBCP により 0xFFFFFCFF に 0x00 を 1 バイト書き込む
   B) 0xFFFFFC18-1B に IP address を書き込む
      ① 0xFFFFFC18 が上位バイト、IP address の bit31-24 に相当

C) RBCP により 0xFFFFFC18-1B を読み込み設定した値が設定されている事を確認する
D) 電源を切る

3. 動作確認
A) 動作モードを通常モードに設定する（FORCE_DEFAULTn=1）
B) 電源投入
C) FPGA のダウンロード終了後に Ping コマンドで希望のアドレスに変更されている事を確認
4. 終了

(*1) IP アドレスはライブラリの IP_ADDR_IN に入力されている値で決定します。FORCE_DEFAULTn の値に依存しません。ラッパーを利用して SiTCP ライブラリを使用している時はラッパー内の信号接続に依存します。ラッパーの信号接続から値を知るか、SiTCP ライブラリ配布元に問い合わせてください。

（注意）IP アドレスを変更する場合、PC 側のネットワーク設定の変更が必要な場合があります。

11. 付録

100Mbps Ethernet 参考回路図(mii.pdf)

Gigabit Ethernet 参考回路図(gmii.pdf)

デバック用 RBCP 制御プログラム(RBCP_sample.zip)

12. 参考文献

1. SiTCP の論文：T.Uchida, “Hardware-Based TCP Processor for Gigabit Ethernet,” http://hdl.handle.net/2261/15490