SiTCP VME-Master module Mode2
BBT-002-2

取扱説明書

Rev 2.7 (July7, 2016)

変更履歴

Rev	変更日	変更ページ	変更内容
0.4	2008/02/13	P12	Address Fix モード時の制限事項を追加
0.5	2008/02/14	P3, 11	非整列転送の非サポートを明記
1.0	2008/04/04	P6	IP Address, TCP port 番号についての説明を追加
1.1	2009/04/01	P2	インタフェイスの表記を変更
1.2	2009/10/29	P4,	誤記修正
2.0	2014/05/28	—	割り込み・ポーリングサポートに伴う全面修正
2.1	2014/07/25	P9	Address Mode の値を修正
2.2	2016/05/02	P17	誤記修正
2.3	2016/06/07	P6	図 6 の誤記修正
2.4	2016/06/14	P8	誤記修正 INTR エラー時の STATUS/ID を明記
2.5	2016/06/14	P8,P12	STATUS/ID（ベクタ）と表記を変更
2.6	2016/06/21	—	Type2 を Mode2 に変更
2.7	2016/07/07	P12	URL 変更

特徴

- Ethernet を経由した VME slave module 制御
- VME32 マスタ機能サポート
- 7 レベルの割り込み時のアクセスを定義可能
- 定期ポーリングのアクセスを定義可能

注意

現在のバージョンでは複数マスタシステムに対応していません。従って、本モジュールを必ずシステムモジュール(Slot 0)に挿入して使用し、他のマスタ・モジュールを挿入しないで下さい。

Rev2.0 から Length フィールドが 32bit から 8bit に変更となりました。上位 24bit は無視されますので、Length を 255byte 以下で使用していれば、互換に使用できます。Rev2.0 から CRC エラーやデータ不足によるアンダーランエラーなどの通信上のエラーを検出した場合を除き、TCP セッションを切断しません。

インタフェイス

以下のインタフェイスを持っています。

1. VME インタフェイス
2. Ethernet

VME インタフェイス

以下の VME マスタ機能をサポートしています。

- D32, D16, D8
- A32, A24, A16
- BLT

※ データバスの非整列転送はサポートしていません。

ETHERNET

以下の IEEE802.3 規格をサポートしています。

- 10BASE-T
- 100BASE-TX
- 1000BASE-T

概要

本モジュールは Ethernet 経由で VME バスを制御する為の VME master module です。

Ethernet を持つ PC などを使用して Ethernet を経由して本モジュールと同一クレートに挿入されている VME slave module をアクセスする事ができます。

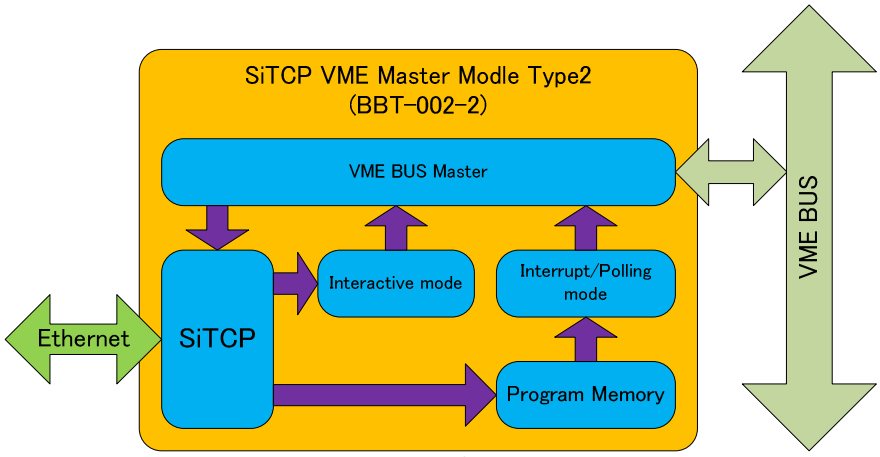


図 1　全体ブロック図

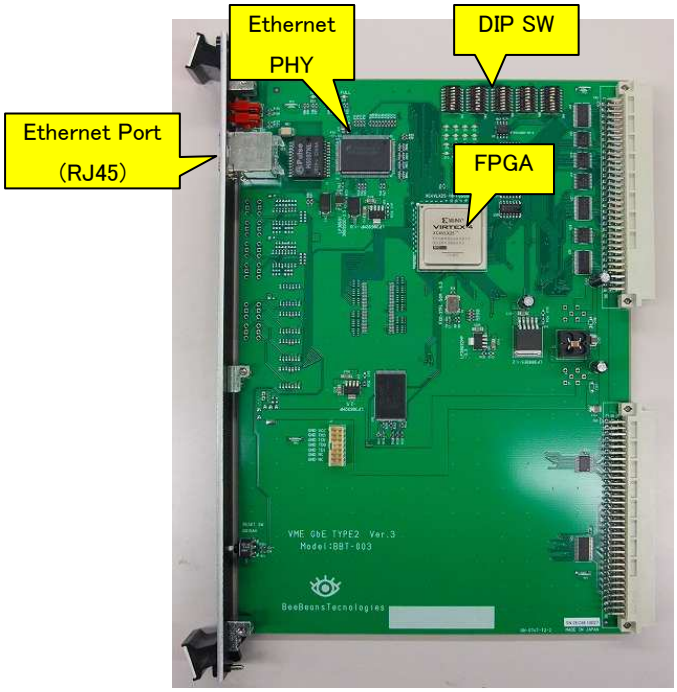


図 2　全体写真

図 1 に全体ブロック図、図 2 に全体写真を示します。本モジュールの主要部品は一つの FPGA と Ethernet PHY device です。FPGA と VME バックプレーン間の信号はバスバッファを介して接続されています。

モジュールの設定

電源を入れる前に以下を設定する必要があります。

- IPv4 address
  モジュールの IP アドレスを設定してください。
  すべて 0(OFF)に設定すると EEPROM に登録したアドレスになります。
  すべて 1(ON)に設定するとデフォルト（192.168.10.16　Port24）設定となります。

- TCP Port Number
  データ通信に使う TCP 待ちうけポートの値を設定してください。
  すべて 0(OFF)に設定すると EEPROM に登録したポート番号になります。

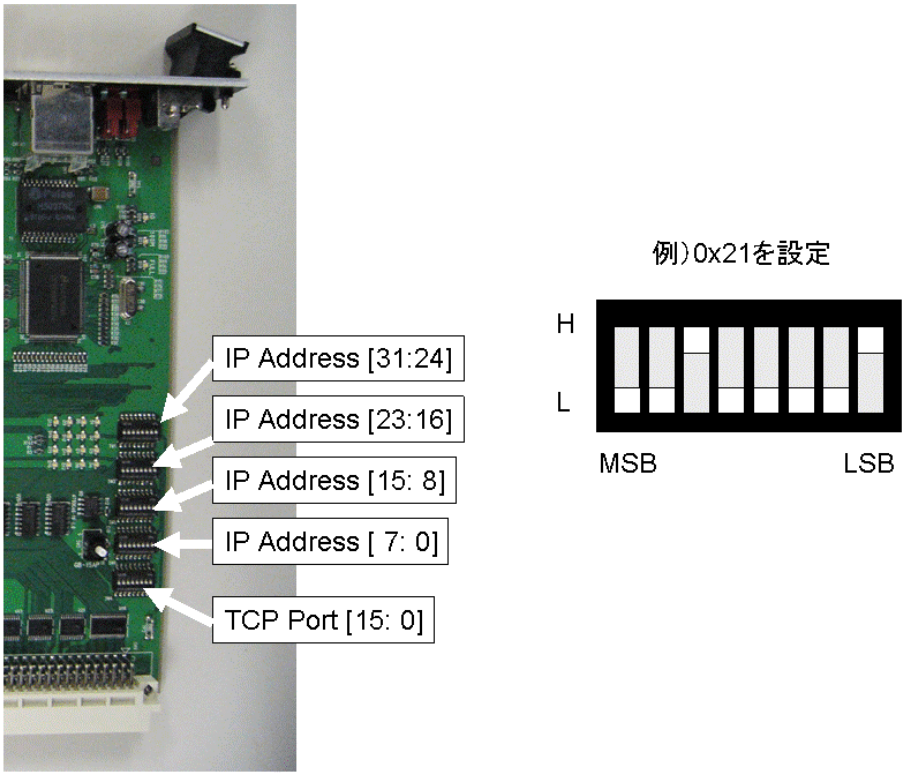


図 3　DIPSW の位置と設定例

図 3 に DIP SW の場所と設定例を示します。DIP に 16 進で設定してください。
拡張機能を設定するための UDP ポートは、出荷時には 4660 に設定されています。
UDP ポートは EEPROM に保存されています。ポート番号の変更は SiTCP ユティリティーで変更できます。

VME インタラクティブ・アクセスの概要

イーサネット経由で TCP パケットを交換する事で VME アクセスを行います。

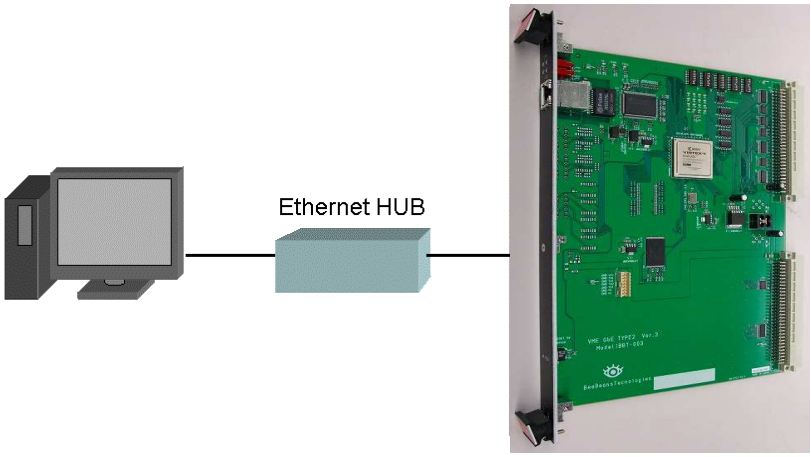


図 4　標準システム構成

図 4 に標準的なシステムを示します。基本的なシステム構成での構成装置は、PC、イーサネット・ハブ、そして本モジュールです。PC は VME バスを制御する為、イーサネット・ハブは PC と本モジュールを接続する為に使用します。

TCP パケットを交換する事で VME アクセスを行うので VME をアクセスするスピードはモジュール性能だけでなくネットワーク性能にも依存します。ネットワークに依存しない安定した性能を求める場合は PC に専用ポートを設け、インターネットなどの公共ネットワークと分離してください。専用ネットワークを使用する場合、性能は PC 性能とアクセスするモジュール数により決まります。

モジュールは複数バスマスタに対応していません。必ず本モジュールはシステム・スロット（Slot 0、通常一番左側のスロット）に挿入してください。また、他のマスタ・モジュールを同一クレート内に挿入しないで下さい。

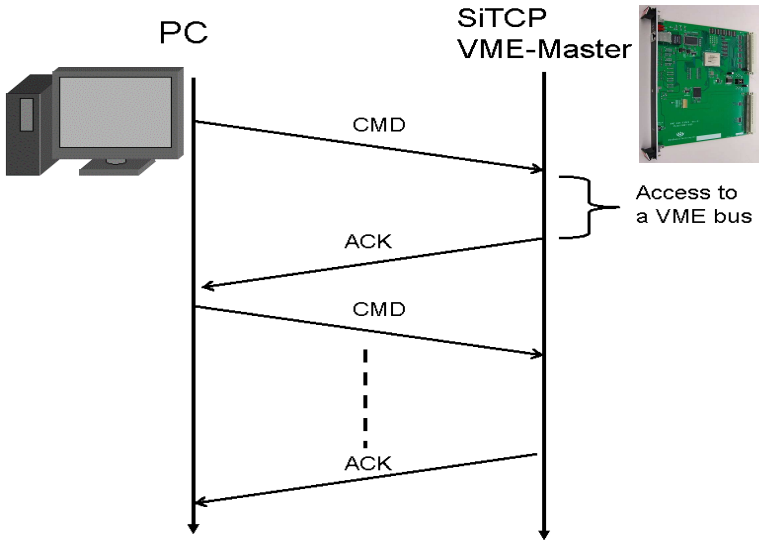


図 5 VME アクセス方法

VME アクセスは TCP 上の命令パケットと応答パケットを使用したハンドシェイク方式で行います。図 5 は PC が SiTCP VME-Master を使用して VME をアクセスした時のパケットを交換の様子を表した図です。時間は図の上から下に流れています。

この図はTCPコネクション確立後のパケットが書かれています。実際に使用するときは、最初にＴＣＰコネクションを確立させることが必要です。通常、ＴＣＰコネクションの確立は SOCKET 関数での connect()関数を使用することで行います。

VMEをアクセスする場合は、初めに PC が希望する VME アクセス情報を CMD パケット（詳細は後述）に設定し TCP データとしてモジュールへ通知します。本モジュールは受信したパケットの構文解析を行い指定された VME サイクルを発生させバス・アクセスを行います。アクセスが終了すると応答パケット（ACK パケット）が送り返されてきます。この時、長いデータをリードした場合は適当なサイズに分割され複数パケットが返送されてきます。

ACK パケットは、アクセスした内容（CMD で指定した内容）とアクセス結果が格納されています。アクセス結果は正常終了か異状終了かがわかるフラグを持っています。フラグの内容は正常終了、パラメータ・エラー、VME バス・タイムアウトなどがあります。パラメータ・エラーは受信した CMD パケットの構文解析中にエラーを検出した時に発生します。

CRC8 はパケットの先頭位置が何らかの理由によりずれた事を検出する為に使用しています。TCP はストリーム型通信ですから、一度データがずれると先頭を見つけることが出来ません。先頭がずれたままバス・アクセスするとシステムに致命的な問題を引き起こす可能性があります。その問題を回避する為に使用しています。CRC8 エラーを検出した場合は TCP コネクションをクローズします。

VME バス・タイムアウトが発生した場合、どのアドレスをアクセスした時にエラーが発生したかが分かるようにアクセスに成功したバイト数がパケットヘッダに格納されます。

コマンドの途中でデータが途切れて、VME アクセスが続けられなくなった場合は、TCP セッションを切断します。

パケット・フォーマット

VME アクセスは TCP 上の命令パケットと応答パケットを使用したハンドシェイク方式で行います。

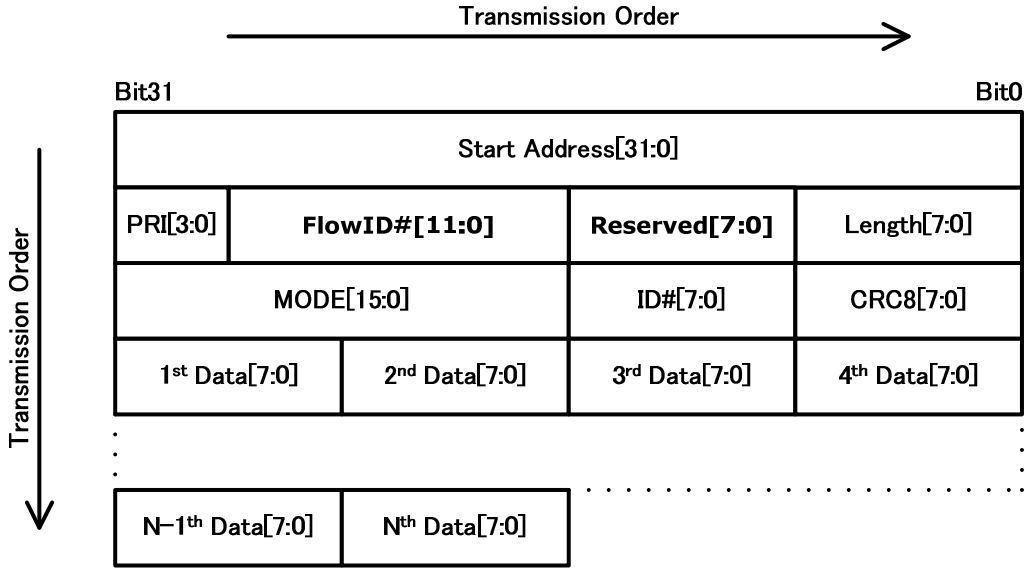


図 6 パケット・フォーマット

図 6 にパケット・フォーマットを、表 1 に MODE フィールドの詳細を示します。

Start Address [31:0]

アクセスする先頭アドレスを設定します。アドレスは VME 仕様に基づいています（本モジュールでアドレス変換等は行っておりません）。MODE フィールド内のアクセス・モードにより使用するアドレス幅が決まります。例えば A24 を使用すれば、このフィールドの A[23:0]が使用されます。現在のバージョンでは、非整列転送はサポートしていません。したがって、32bit アクセスの場合は、下位 2bit が 0 である必要があります。同様に 6bit アクセスの場合は、下位 1bit が 0 である必要があります。ただし、割り込み応答サイクルの場合は bit3～bit1 で応答割り込みレベルを指定します。この時、それ以外の bit は 0 として下さい。

PRI3:0]

このフィールドは、プログラマブル・アクセス時に echo パケットに設定される領域です。インタラクティブ・アクセスでは、CMD パケットで設定された値がそのまま返送されます。

プログラマブル・アクセスでは、コマンドを実行したプログラムのプライオリティが設定されます。PRI[3:1]がプライオリティで PRI[0]がポーリングの時が 1 で割り込みの時が 0 です。

Flow ID[11:0]

このフィールドは、プログラマブル・アクセス時に echo パケットに設定される領域です。インタラクティブ・アクセスでは、CMD パケットで設定された値がそのまま返送されます。

プログラマブル・アクセスでは、コマンドを実行したプログラムの実行回数を表示します。初期値は指定できないので、実行回数の相対カウンタとして使用してください。

Reserved [7:0]

このフィールドは、未使用領域です。インタラクティブ・アクセスでは、CMD パケットで設定された値がそのまま返送されます。拡張時にも互換性を保つため 0x00 を設定してください。

Access Length [7:0]

アクセス長をバイト数で指定します。1 バイトをアクセスする時は 0x1 を設定してください。MODE フィールド内のアクセス・モード（アクセス・データ幅）により指定できるバイト数が異なりますので注意してください。本モジュールは CMD パケットで設定された内容を VME プロトコルに直接変換する為に制限があります。例えば、データ幅を 16bit アクセス指定した場合には奇数バイト数は許されません。奇数バイト数アクセスする場合には 8 ビットアクセスを使用するか、端数部分以外は 16 bit アクセス、端数部分は 8 bit アクセスと CMD パケットを分割して使用してください。同様に 32bit アクセスを指定した場合は、4 の倍数を指定してください。

Mode [15:0]

アクセス・モードを指定します。以下に各ビットの詳細を示します。

表 1 MODE フィールド詳細

Bit	Name	Description
15	Write	Write access=1 Read access=0
14	Echo Write data	Disable echo back of write data = 0, Enable echo back of write data = 1
13	No Echo	Enable echo Packet =0(Normal Setting) Disable echo Packet = 1(*3)
12	Reserved	Should be 0
11-10	Data Mode	D8 = 0, D16=1, D32=2 (*1) Value 3 is reserved. Do not use it.
9-8	Address Mode	A16 = 0, A24=1, A32=2 Value 3 is reserved. Do not use it.
7-4	Access Mode	User Data=0, User Prog. =1, User BLT =2, INTR =3(*4), Fixed Address and User Data =8(*2), Fixed Address and User Prog. =9(*2), Supervisor Data=4, Supervisor Prog. =5, Supervisor BLT =6, Supervisor data & Fixed Address =0xC(*2), Supervisor prog. & Fixed Address =0xD(*2), Other values are reserved. Do not use those.
3	Acknowledge Flag	Command Packet =0, ACK packet = 1
2	Flag	Active in ACK packets only Normal =0; VME error = 1
1	Flag	Active in ACK packets only Reserved Always 0
0	Flag	Active in ACK packets only Normal = 0, Detected parameter error = 1

(*1) 非整列転送はサポートしていません。

(*2) Fixed Address とは Start Address フィールドに指定したアドレスを Access Length フィールドで指定されたデータ長に相当するだけ、同じアドレスをアクセスします。

(*3) Disable echo Packet に設定するとアクセスがリード・ライトにかかわらず Echo パケットを返送しません。ただし、エラーの場合は Echo パケットを返します。

(*4) エラーの場合 Status/ ID(ベクタ)は 0xFF になります。

ID

CMD パケットと対応する ACK パケットを認識する為に使用します。

ACK が戻ってくる前に次の CMD パケットを発行できます（コマンド・パイプライン機能）。この機能を使用する場合 CMD パケットと対応する ACK パケットの対応が分からないと制御プログラムが複雑になるため、この複雑さを回避する為に使用します。コマンド・パイプライン機能を使用することにより高速アクセスする事が可能です。

CRC8

パケットヘッダが壊れていないか検査するために使用する CRC コードです。本モジュールはこのコードを使用してパケットの先頭がずれていないかを検査します。エラーを検出した場合はセッションを切断してインタラクティブ・アクセスを終了します。

使用している多項式は x8+x2+x+1 です。また、計算する際に CRC 初期値を 0xFF にしています。計算範囲はパケット先頭から 11 バイトです。計算はデータ送信順序で計算します（ビットのスワップ等は行っていません各バイトの bit7 から計算してください）。計算結果をこのフィールドに格納します。格納する際にビット反転させません、計算値をそのまま格納してください。

ACK パケットにも再計算された CRC8 値がこのフィールドに格納されています。再計算する理由は少なくとも MODE フィールドの値が変更される為です。この再計算された CRC8 フィールドをエラー検査に使用することが出来ます。高信頼システムを構築する為に、このフィールドの検査を行ってください。エラーが発生した場合は、セッションを切断してインタラクティブ・アクセスをいったん終了してください。

Data

ライトコマンドを持つ CMD パケットはライトデータを持つ必要があります。このフィールドはそのデータを格納するフィールドです。格納するバイト数は Access Length フィールドに格納されている値と一致していなければいけません。矛盾がある場合は次のパケット先頭を正しく認識できなくなり、CRC エラーが発生します。リードコマンドを持つ CMD パケットにこのフィールドを付けてはいけません。

ライトコマンドの ACK パケットは基本的にはこのフィールドを持ちませんが、MODE 内の Echo Write Data を有効にした場合は正常に書き込まれたデータが格納されます。

リードコマンドの ACK パケットのこのフィールドには正常に読み込まれたデータが格納されます。格納されているデータ長は Access Length フィールドに格納されています。1 が 1 バイトを意味します。例えばエラーが発生し、成功したアクセスがない時には 0 が格納されます。途中で失敗した場合には、正しく読み込むことが出来たデータが格納されます。

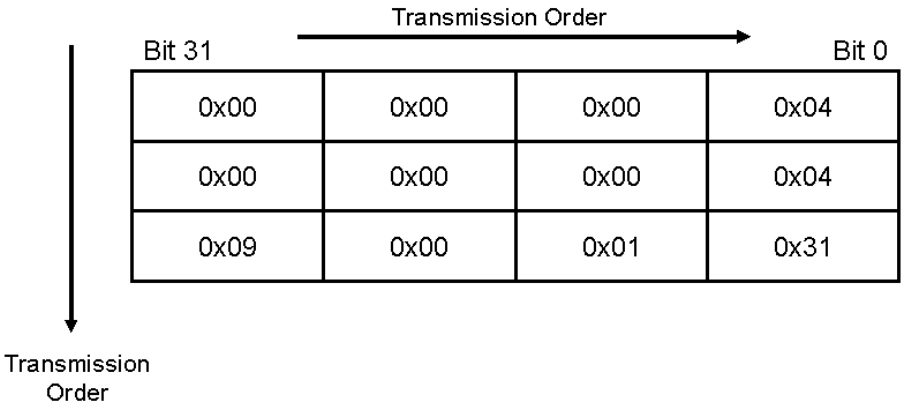
パケット例

以下に幾つかパケットの具体例を挙げます。

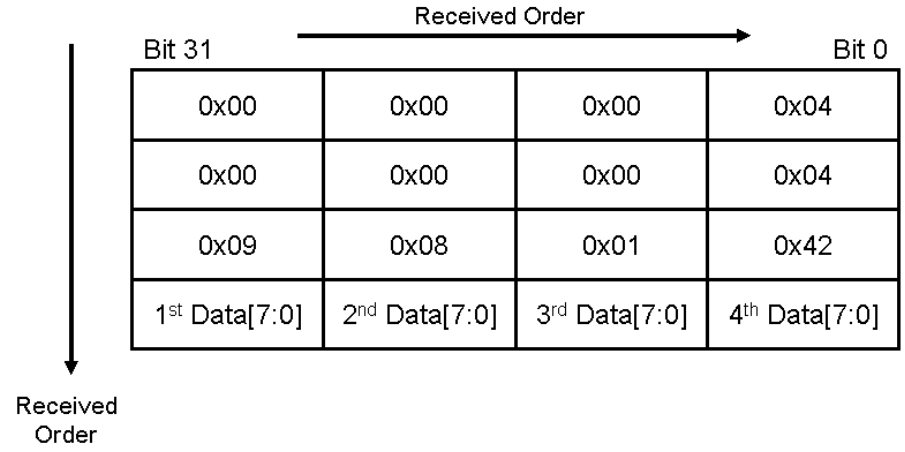
例 1

- VME read
- A24
- D32
- User data access

ＰＣから送信する命令パケット



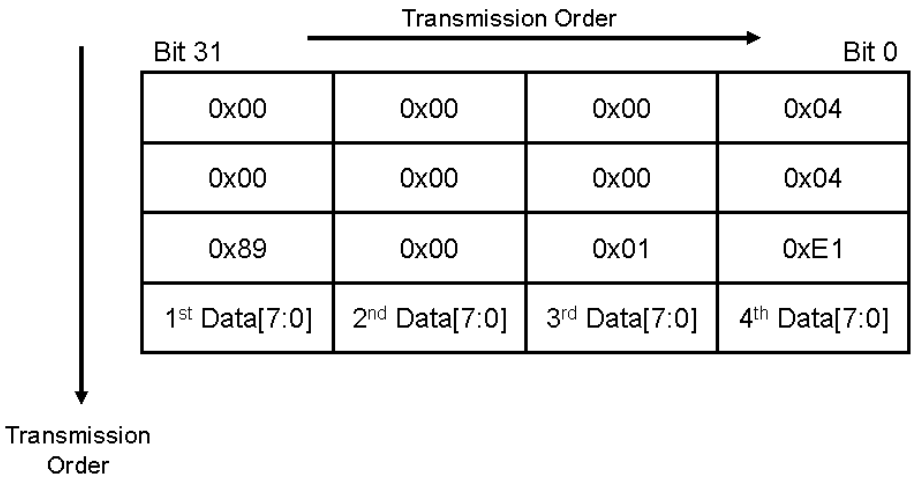
VME-Master module から返送されてくる応答パケット



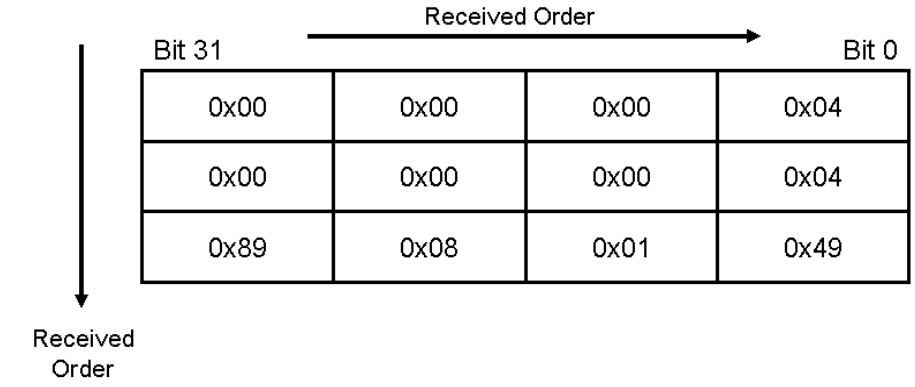
例 2

- VME write
- A24
- D32
- User data access

ＰＣから送信する命令パケット



VME-Master module から返送されてくる応答パケット



VME プログラマブル・アクセスの概要

モジュール内のメモリにイベントが発生した時に実行する VME アクセスを事前にプログラムしておく事によって、イベントに対してリアルタイムで固定値の書き込みや読み出し処理を行えます。イベントは 7 レベルの割り込みと 1 つのポーリングタイマです。VME アクセスを実施すると、インタラクティブ・アクセスと同じ echo パケットを返送します。

割り込み時に割り込み応答サイクルで獲得したベクタをデコードする為の Vector Decode memory も搭載しています。なお、STATUS/ID（ベクタ）のサイズは 8bit です。

プログラマブル・アクセスの各種設定は RBCP(Remote Bus Control Protocol)によって行います。RBCP に関しては、以下のサイトから詳細を入手できます。

https://sitcp.bbtech.co.jp/

RBCP のメモリマップ

Address	Byte	Explanation	Note
00000000 - 00000003	4	Version Register	Read only
00000004	1	Event Enable Register	
00000005	1	Not Use	Do not write
00000006	1	Polling Priority Register	
00000007	1	Execute Priority Register	Read only
00000008 - 0000000B	4	Not Use	Do not write
0000000C - 0000000F	4	Polling Interval Time Register	
00000010 - 00000017	8	Program Top Position Register	
00000018 - 0000001F	8	Number of Instruction Register	
00000020 – 000003FF	992	Not Use	Do not write
00000400 – 000007FF	1024	Vector Decode memory	
00000800 – 00000FFF	2048	Program memory	

Version Register （00000000-00000003）

8 桁の BCD データです。初めの 6 桁は、マイナチェンジの変更を示します。最後の 2 桁が機能変更を伴う Rev コードで、現在は 02 です。

Event Enable Register （00000004）

割り込みやポーリングのイベントを許可します。各ビットが対応するイベントの許可を設定します。1が許可で 0 がマスクです。bit0 がポーリング、bit1 が割り込みレベル 1、bit7 が割り込みレベル 7 です。

Polling Priority Register （00000006）

ポーリングのイベントで実行するプライオリティ・レベルを設定します。0～7の範囲で設定できます。

Execute Priority Register (00000007)

現在実行中のプライオリティを表示します。0～7の値が表示されます。

Polling Interval Time Register (0000000C-0000000F)

ポーリングイベントの周期をμsec 単位で設定します。

Program Top Position Register (00000010-00000017)

各イベント発生時のプログラム開始位置を 1Byte で指定します。設定値を X とするとプログラムの開始位置は(0x00000800+8×X)となります。

0x00000010 がポーリング用、0x00000011 が割り込みレベル 1 用、0x00000017 が割り込みレベル 7 用です。

Number of Instruction Register (00000018-0000001F)

各イベント用のプログラムの(CMD 数-1)を 1Byte で指定します。一つのコマンドしか実行しない場合は 0x00 を設定してください。割り込み応答サイクルを実施した場合は、その時に読み出したベクタの下位 1Byte を Vector Decode memory でデコードした結果の開始位置と CMD 数で実行が引き継がれます。

0x00000018 がポーリング用、0x00000019 が割り込みレベル 1 用、0x0000001F が割り込みレベル 7 用です。

Vector Decode memory (00000400-000007FF)

割り込み応答サイクルで獲得したベクタの下位 1byte をデコードする為のメモリです。ベクタ番号を V とすると(0x00000400+4×V)からの 3byte がデコード結果です。

初めの 1byte が(CMD 数-1)で、次の 2Byte が(プログラムの開始位置-0x800)です。開始位置は 0x000 から 0x7FF の範囲で設定してください。

Program memory (00000800-00000FFF)

実行するプログラムを格納します。このメモリに格納する CMD のフォーマットを図 7 に示します。インタラクティブ・アクセスのフォーマットと異なるので注意してください。

各フィールドの内容はインタラクティブ・アクセスと同じです。インタラクティブ・アクセスのフィールドの一部を削除して詰めた構造になっています。

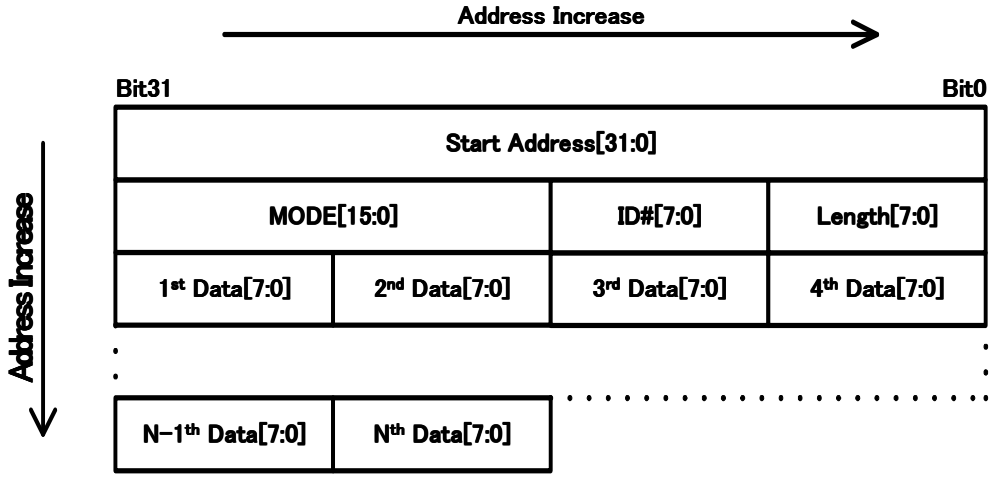


図 7 Program Memory 内・フォーマット

プログラム例

以下に幾つかプログラムの具体例を挙げます。ここで「write ([address],[data])」は、RBCP で[address]に[data]を書き込む処理を表します。

例 1

- Read, Length=4,Address=0x11223344
- Write, Length=4,Address=0x55667788,Write Data= 0x12345678

この 2 回のアクセス(A24, D32,User data access)を 100ms 周期で行う

書き込みに関しては、エラーがない場合は echo を返さない

```
// Program memory
  Write (0x00000810,0x11)  ※開始アドレスは 8 の倍数であれば任意
  Write (0x00000811,0x22)
  Write (0x00000812,0x33)
  Write (0x00000813,0x44)   // Address = 0x11223344
  write  (0x00000814,0x0A)
  write  (0x00000815,0x00)   // MODE   read, D32, A32, User Data
  write  (0x00000816,0x01)   // ID = 1 ※値は任意
  write  (0x00000817,0x04)   // Length = 4
  write  (0x00000818,0x55)
  write  (0x00000819,0x66)
  write  (0x0000081A,0x77)
  write  (0x0000081B,0x88)   // Address = 0x55667788
  write  (0x0000081C,0xCA)
  write  (0x0000081D,0x00)   // MODE   write, No Echo, D32, A32, User Data
  write  (0x0000081E,0x02)   // ID = 2  ※値は任意
  write  (0x0000081F,0x04)   // Length = 4
  write  (0x00000820,0x12)
  write  (0x00000821,0x34)
  write  (0x00000822,0x56)
  write  (0x00000823,0x78)   // write Data = 0x12345678
// Program Top Position Register
  write  (0x00000010,0x02)  // (Top Address – 0x800) / 8= 0x02
// Number of Instruction Register
  write  (0x00000018,0x01)  // (Number of Instruction – 1) = 0x01
// Polling Interval Time Register
  write  (0x0000000C,0x00)
  write  (0x0000000D,0x01)
  write  (0x0000000E,0x86)
  write  (0x0000000F,0xA0)  // Polling Interval = 0x000186a0 = 100,000 us
// Event Enable Register
  write  (0x00000004,0x01)
```

例 2

- 割り込みレベル 3 応答　ベクタ 0x20
- Read, Length=4,Address=0x11223344
- Write, Length=4,Address=0x55667788,Write Data= 0x12345678

この 2 回のアクセス（A24, D32,User data access）をベクタ 0x20 の処理として行う。

書き込みに関しては、エラーがない場合は echo を返さない

```
// Vector Decode memory  ※アドレスは 0x400+ベクタ(0x20)×4=0x480
  Write (0x00000480,0x01)  // (Number of Instruction – 1) = 0x01
  Write (0x00000481,0x00)
  Write (0x00000482,0x10)  // (Top Address – 0x800) = 0x0010
// Program memory
  Write (0x00000810,0x11)  ※開始アドレスは任意
  Write (0x00000811,0x22)
  Write (0x00000812,0x33)
  Write (0x00000813,0x44)   // Address = 0x11223344
  write  (0x00000814,0x0A)
  write  (0x00000815,0x00)   // MODE   read, D32, A32, User Data
  write  (0x00000816,0x01)   // ID = 1 ※値は任意
  write  (0x00000817,0x04)   // Length = 4
  write  (0x00000818,0x55)
  write  (0x00000819,0x66)
  write  (0x0000081A,0x77)
  write  (0x0000081B,0x88)   // Address = 0x55667788
  write  (0x0000081C,0xCA)
  write  (0x0000081D,0x00)   // MODE   write, No Echo, D32, A32, User Data
  write  (0x0000081E,0x02)   // ID = 2  ※値は任意
  write  (0x0000081F,0x04)   // Length = 4
  write  (0x00000820,0x12)
  write  (0x00000821,0x34)
  write  (0x00000822,0x56)
  write  (0x00000823,0x78)   // write Data = 0x12345678

  Write (0x00000840,0x00)  ※開始アドレスは 8 の倍数であれば任意
  Write (0x00000841,0x00)
  Write (0x00000842,0x00)
  Write (0x00000843,0x06)   // Address = (Interrupt Level)*2 = 0x00000006
  write  (0x00000844,0x28)
  write  (0x00000845,0x30)   // MODE   read, No Echo, D32, INTR
  write  (0x00000846,0x00)   // ID = 0 ※値は任意
  write  (0x00000847,0x04)   // Length = 4
```

```
// Program Top Position Register
  write  (0x00000013,0x08)  // (Top Address – 0x800) / 8= 0x08
// Number of Instruction Register
  write  (0x0000001B,0x00)  // (Number of Instruction – 1) = 0x00
// Event Enable Register
  write  (0x00000004,0x08)
```

本マニュアルおよび製品に関するお問い合わせ

株式会社　BeeBeans Technologies
〒300-3256　茨城県つくば市大穂 109
TEL:029-875-3642　FAX:029-875-3564
E-MAIL　:tech-support@bbtech.co.jp
URL: http://www.bbtech.co.jp