JUMP-1

PATCH仕様書

Revision History

1.0	’99 03/30	M. Goshima	第1稿
1.1	’99 07/12	M. Goshima	クラスタ・ボードの修正点を追加
2.0	’99 11/19	M. Goshima	CCのバグに対する対応を追加

目 次

1　概要

本資料は，JUMP-1 の MMC および CC に存在する（'99 11/19 現在）バグを回避するための方法について述べる．

バグの回避は，クラスタ・ボードに搭載されるパッチ IC **PATCH** と MBP Core プログラムとの協調動作によって実現される．

2　MMC/CC のバグと回避方法

'99 03/31 現在，MMC には以下の 4 つのバグがあることが分かっている：

BGRTO　自身がパケット送信中に X_BGRT を受け取ると，パケット送信中にも関わらず，次のパケットの送信を開始しようとする．

RETRY　X_BRETRY のサンプリング・タイミングが間違っており，CC の出した X_BRETRY を無視することがある．

SER　パケット送信時に次のパケットを送信バッファに書込もうとすると，バッファを取り違うことがある．これは，パケットが MMC からのものであるか Core からのものであるかに関わらず起こり得る．

PEND　PEND Counter を increment したパケットが，MBP Core に割り込みを発生すると，この Counter を decrement してしまう．

また，'99/11/19 現在，CC には以下の 1 つのバグがあることが分かっている：

CRPLY　短い期間の間に連続してパケットを受信すると，先行するパケットが後続のパケットによって上書きされてしまうことがある．

以下，各バグの回避方法について詳述する．

2.1　BGRTO

MMC がパケット送信中には，MMC への X_BGRT をカットする．

Bus Chip の出力する X_BGRT は一旦 PATCH に入力し，MMC へは代わりに PATCH が出力する X_BGRT を渡す．PATCH は，通常は Bus Chip の出力する X_BGRT をそのまま MMC に渡すが，MMC が X_ASTB をアサートしてから 4 サイクルの間 X_BGRT を強制的にディアサートする．X_BGRT をディアサートする期間 4 サイクルは，最大パケット長（5 フリット）から決まる[1]．

---

[1] MMC からの PLEN を参照することによりディアサートする期間を短縮することも考えられるが，それによる大きな性能向上は期待できない．

2.2 RETRY

通常のX_BRETRYシグナルより1サイクル早い版と，通常のX_BRETRYシグナルを，MMCに入力する．1サイクル早い版は，各デバイスが出力するX_BRETRYシグナルをORしたものである．

すなわちMMCには，2サイクル連続でX_BRETRYが入力される．CBusには最高で2サイクルに1度しかパケットが送出されないので，対応するパケットを間違えることは無い．

2.3 SER

MMCが同時に2つ以上のパケットを送信しようとすることが無いようにする．

- PATCHは，MMCが1つのパケットを処理している間，後続のMMCが処理する可能性のあるパケットにリトライを要求する．
- その要求がCoreへの割り込みなった場合には，Coreプログラムはその旨をPATCHに通知し，次のパケットを受け付けるようにする．
- Coreプログラムはまた，MMCに対してパケットの送信を依頼する前に，MMCが同時にパケットの処理を行わないようにMMCのバッファのドレイン処理を行う．

このことは，性能に深刻な影響を与える可能性が高い．リトライを要求すべきパケットを選択するフィルタリングの精度が重要である．

2.4 PEND

MMC内のPEND Counterはまるで用をなさないので，PEND Counterの本体はCoreのプログラムでSW的に用意する．PATCHは3bのカウンタを持ち，パケットの送受信とCoreのSWカウンタのincrementとの時間差を埋める役割を担う．

2.5 CRPLY

一定期間X_BGRTをアサートするタイミングを送らせることによって，各CCに対してX_ASTB_Iがアサートされてから次にアサートされるまで3 cycle空ける．

なお，前述したBGRTOでは，MMCに対するX_BGRTをカットするので，それとの関係に注意する必要がある．

3 PATCH の仕様

PATCH は，バグ BGRTO，RETRY，SER，PEND，CRPLY に対応するブロックからなる．このうち主に SER，PEND と，CRPLY について述べる．

3.1 インターフェイス・シグナル

x_astb_d2b4	MMC の出力する X_ASTB．
x_bgrt_org	Bus Chip からの X_BGRT．
x_bgrt_new	PATCH から CC/MMC への X_BGRT．
scmd, pcmd, ack, did[2], sid, clsh	アドレス・フリットのフィールド．Bus Chip の#5 ポートから採る．
x_astb_b2d	Bus Chip が出力する X_ASTB．Bus Chip の#5 ポートから採る．
x_bretry_d2b	各 CC/MMC から Bus Chip への X_BRETRY．
x_bretry_b2d	Bus Chip から各 CC/MMC の X_BRETRY．
x_crply_d2b	CC から Bus Chip への X_CRPLY．
x_crply_b2d4	Bus Chip から MMC への X_CRPLY．
x_pend	各 CC への X_PEND．
lio	LBus I/O 空間選択シグナル．1 で I/O を選択．
ladr(15:12)	LBus アドレスの上位 4b．
ldata(3:0)	LBus データの下位 4b．
loe	LBus Output Enable．Active-Low．
lwe	LBus Write Enable．Active-Low．非同期ライト・パルス．
clk, x_rst	クロックとリセット．

3.2 SER

3.2.1 レジスタ

SER には，以下のレジスタがある：

MPND MMC が処理中の読み出し要求がある間セットされる．

3.2.2 動作

MPND は，以下の事象に対してセット/リセットされる．

セット

- MMC が処理する読み出し要求パケットに対してセットされる．

リセット

要求パケット	応答パケット
read_req	read_reply
n_c_read_req	n_c_read_reply
allread_req	allread_reply
swap_req	swap_reply
rdinv_req	rdinv_reply

表 1: 読み出しパケット

- MMC が処理した読み出し要求に対する応答パケットに対してリセットされる．
- Core からの LBus を通じたリセット指示によってリセットされる．

3.2.3 パケットの識別

MMC が処理する読み出し要求は，以下の条件を満たす：

- 表 1 に示す読み出し要求である．
- DID(2) = 1，すなわち，MMC 宛か，ブロード・キャストである．
- X_BRETRY も X_CRPLY もアサートされなかった．

MMC が処理した読み出し要求に対する応答は，以下の条件を満たす：

- 表 1 に示す読み出し応答である．
- SID(2) = 1，すなわち，MMC からのものである．

3.2.4 タイム・チャート

図 1 に，SER のタイム・チャートを示す．

図 1 中，rq，rp は，表 1 に示す要求，応答をデコードしたシグナルである．

3.3 PEND

3.3.1 レジスタ

PEND には，4 つの CC に対応する，以下のレジスタがある：

3.3.2 動作

X_PEND シグナルは，対応する PEND レジスタの，以下のいずれかが成立しているときアサートされる：

- SF = 1
- HC /= 0

HC をオーバーフローさせるようなパケットに対しては，リトライを要求する．

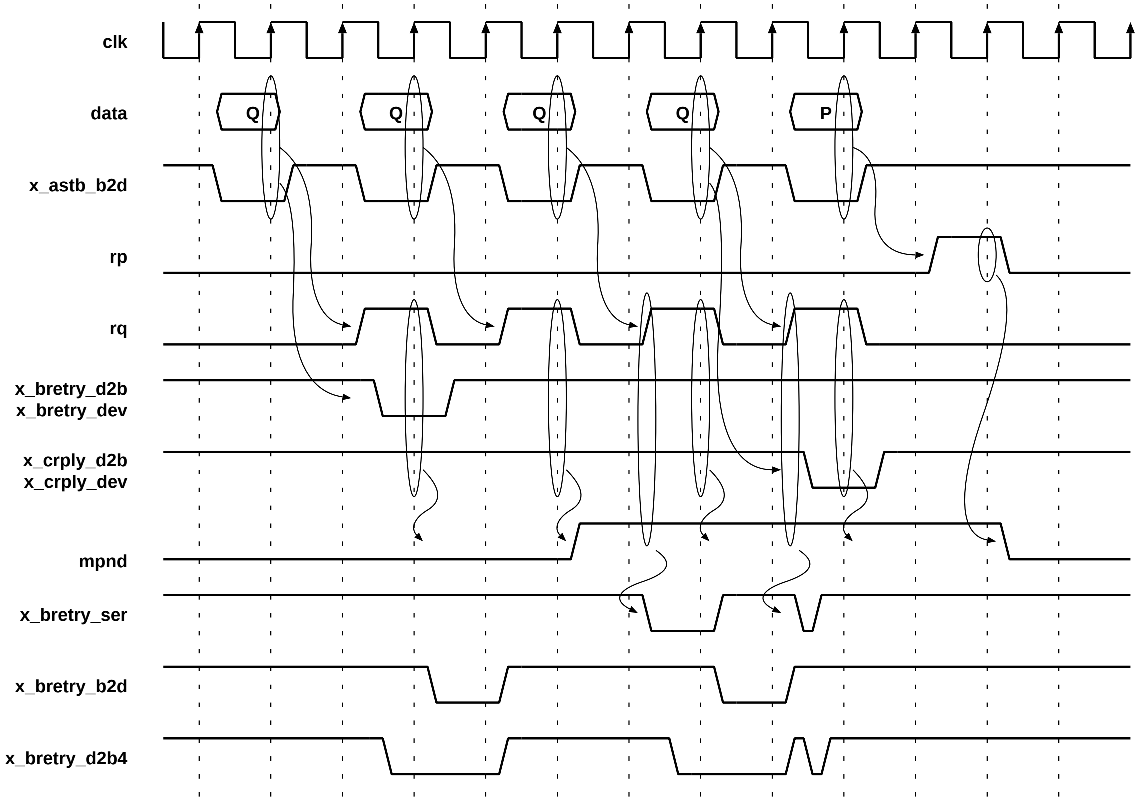
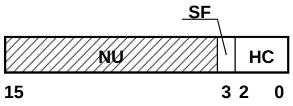


図 1: SERのタイム・チャート



NU 読み出しデータは不定，書き込みデータは無視される．

SF 1bのレジスタ．Coreプログラムが管理するSWカウンタが0でないときセットされる．

HC 3bのカウンタ．MMCは受信したが，Coreプログラムは認識していないパケットの数を数える．

図 2: PEND

3.3.3 パケットの識別

HCのdecrementは，Coreの指示によって行われる．したがってincrementするパケットは，常にCoreに割り込まれるものでなければならない．また，それらのうち，実際にincrementを行うものは$\mathrm{ack} = 1$，すなわち，PENDによる順序保証を行うものだけでよい．

表2に，PEND(0:3).HCをincrementするパケットとincrementする条件を示す．

表2中の条件は，以下のとおり：

パケット	条件
update_via_home_req	clsh
invalidate_via_home_req	clsh
update_direct_req	clsh & ack
update_direct_s_req	clsh & ack
write_xa_req	ack
write_xb_req	ack
write_xc_req	ack
write_xd_req	ack
update_via_home_q_req	ack
n_c_write_m_req	true
control_write_req	did(2)

表 2: HC を increment するパケットと条件

clsh	これらのパケットでは，clsh フィールドが有効である． ack フィールドは，常に 1 である．
clsh & ack	これらのパケットでは，clsh フィールドが有効である． ack フィールドは，L2C の設定によって変化する．
ack	これらのパケットでは，clsh フィールドが有効ではないが，無条件に Core への割り込みになる． ack フィールドは，L2C の設定によって変化する．
true	n_c_write_m_req は，常に MMC/MBP 宛であり，無条件に Core への割り込みになる． ack フィールドは，常に 1 である．
did(2)	control_write_req は，MMC/MBP 宛の場合，常に Core への割り込みになる． ack フィールドは，常に 1 である．

non-cacheable write に起因する non-cacheable な write_req は，MMC が直接処理するかどうか，PATCH では判断できない．increment しておいて Core に割り込みがかからないとおかしくなるので，write_req に対して increment を行うわけにはいかない．結果，non-cacheable write に対する書込み順序は，PEND 機構では保証できない．

3.3.4 タイム・チャート

図 3 に，PEND のタイム・チャートを示す．

図 3 中，wq は，表 2 に示す要求パケットをデコードしたシグナルである．

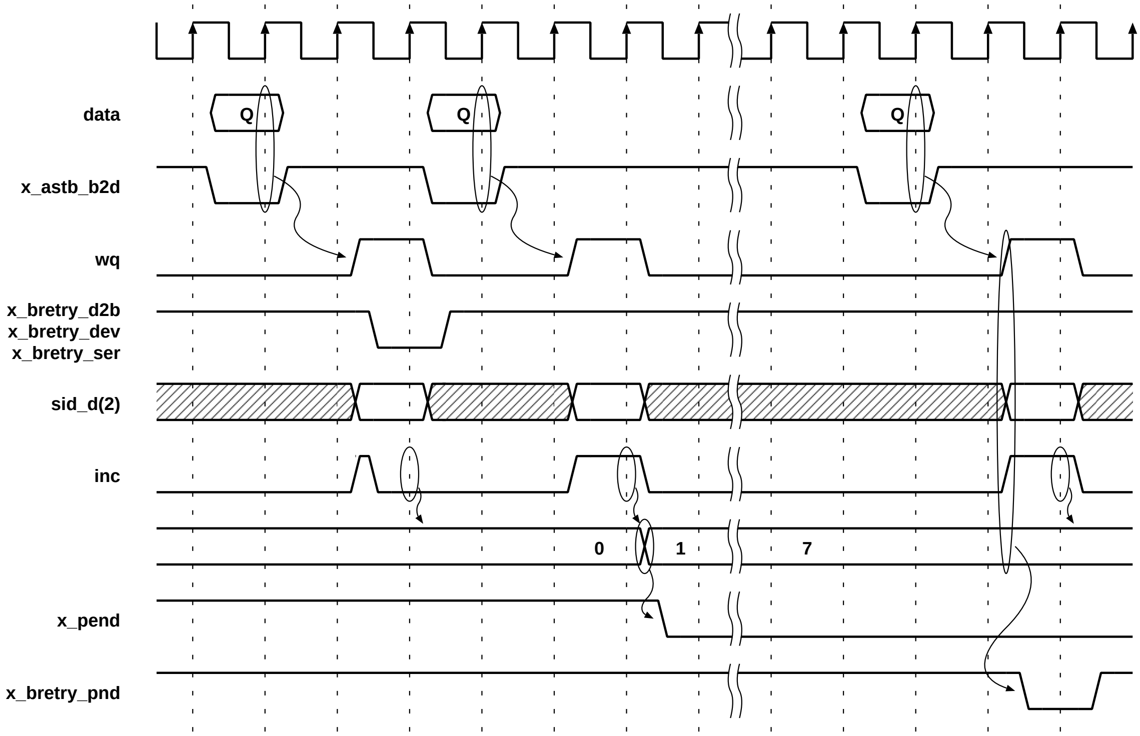


図3: PENDのタイム・チャート

3.4 CRPLY

図4に，同一のデバイスが連続してパケットを送出する場合の ;図5に，異なるデバイスが連続してパケットを送出する場合のCRPLYのタイム・チャートを示す．

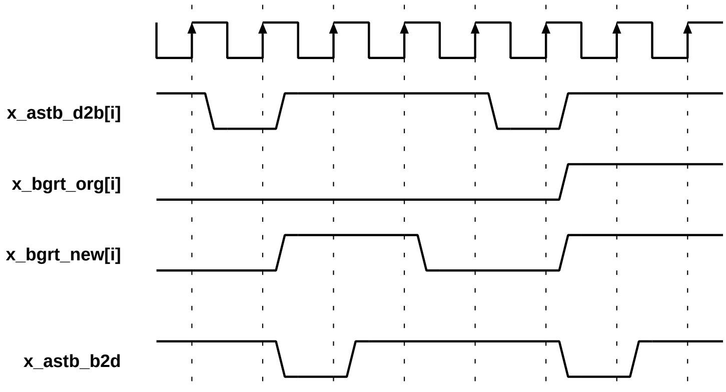


図4: CRPLYのタイム・チャート

チャートから，結局どのデバイスがアサートしたかに関わらず，x_astb_d2bがアサートされてから2 cycleの間，x_bgrtをカットすればよいことが分かる．

なお，MMCに対するx_bgrt，x_bgrt[4]は，バグBGRTOを回避するため，x_astb_d2b[4]がアサートされてから4 cycleの間 カットされることに注意．

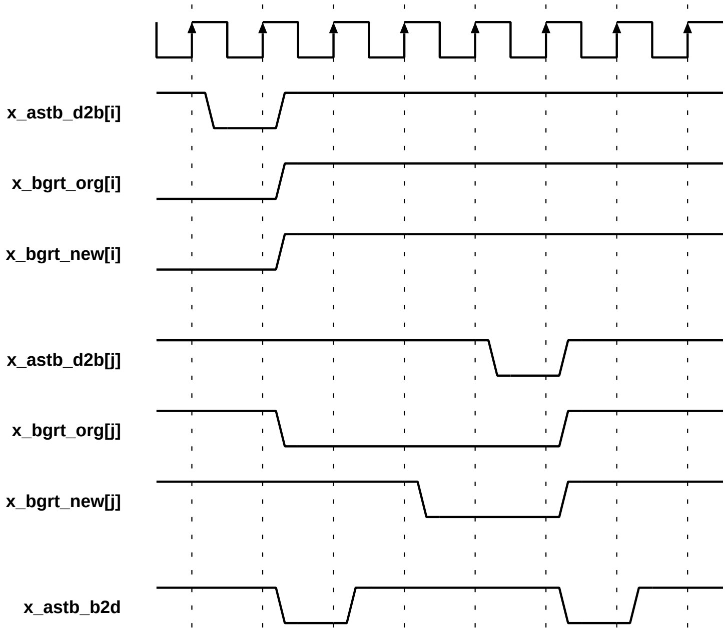


図 5: CRPLY のタイム・チャート

3.5 PATCH 内部資源と LBus インターフェイス

本章では，PATCH の内部資源と LBus インターフェイスについてまとめる．LBus I/O 空間の 0xB000 から 0xFFFF が PATCH に割り当てられている．表 3 に，コマンドを示す．

ladr(15:12)	ldata(3:0)	R/W	操作
0b1011	0b?1??	W	MPND をリセットする．
0b1011	0b?0AB	W	PEND(0bAB).SF をリセットする．
0b11AB	------	R	PEND(0bAB)を読み出し， PEND(0bAB).SF をセットし， PEND(0bAB).HC を decrement する．
0b11AB	0bXYZW	W	PEND(0bAB)に 0bXYZW を書き込む．

表 3: コマンド

なお，underflow を起こした場合，操作後の HC は不定となる．

4 クラスタ・ボードの変更点

本章では，PATCH の搭載にあたってのクラスタ・ボードの変更点について述べる．

4.1 変更点リスト

クラスタ・ボードの配線の変更点は以下のとおり：

1. **X_BGRT** Bus Chip から MBP-light，CC への X_BGRT を PATCH 経由に変更
2. **X_ASTB_D2B4** X_BGRT 絡みで，MMC からの X_ASTB を PATCH に入力
3. **X_PEND** MBP-light の出力をカットし，PATCH の出力を使用
4. **X_CRPLY** Bus Chip 経由から PATCH 経由に変更
5. **X_BRETRY** Bus Chip 経由から PATCH 経由に変更
6. **X_ASTB_B2D** Bus Chip の#5 ポートの出力を PATCH に入力
7. **PDATA** Bus Chip の#5 ポートから取ったデータのうち表 4 に示すものを PATCH に入力
8. **X_P_MSH** Bus Chip の#5 ポートの出力を PATCH に入力
9. **PKTLEN（PWWE）** Bus Chip の#5 ポートの出力を PATCH に入力
10. **LADR** MBP-light ローカルバスの LADR[15..12] を PATCH に入力
11. **LDATA** MBP-light ローカルバスの LDATA[3..0] を PATCH に入力
12. **LIO_IO** MBP-light からの LIO_IO 信号を PATCH に入力
13. **LCTRL** MBP-light からの LCTRL (X_LOE，X_LWE) 信号を PATCH に入力
14. **X_MBP_RST，MBP_CLK** X_MBP_RST，MBP_CLK を PATCH に入力

ビット位置	アドレス・フリットのフィールド
PDATA59	CCBL
PDATA[58..56]	SCMD[2..0]
PDATA[55..52]	PCMD[3..0]
PDATA51	SIZE2
PDATA48	ACK
PDATA[46..44]	DID[2..0]
PDATA[43..41]	SID[2..0]
PDATA38	CLSH

表 4: PATCH に入力される DATA

これらのうち，3.1 節で挙げられていない以下のシグナルは，将来に備えて PATCH に接続しておくものである：

1. **X_P_MSH**
2. **PKTLEN**
3. **CCBL**
4. **SIZE2**
5. **DID[1..0]**
6. **CLSH**

5 ソフトウェアの変更点

ここでは，SPARC と MBP Core のプログラムの変更点についてまとめる．

5.1 SPARC プログラムの変更点

non-cacheable write に対する書き込み順序保証は，基本的には，行うことができない．ただし，対象が当該クラスタに閉じている場合には，以下の場合プログラム・オーダでの書き込みが保証される．

- SIZE が doubleword である write はそれらの間で，プログラム・オーダでの書き込みが保証される．
- SIZE が doubleword でない write は（Core プログラムが到着順に処理すれば）プログラム・オーダでの書き込みが保証される．

SIZE が doubleword である write と doubleword でない write の間での書き込み順序は非決定的である．

5.2 MBP Core プログラムの変更点

MBP Core プログラムの変更点は，SER に関することと，PEND に関することの 2 点である．

5.2.1 SER

受信 表 1 に示す読み出し要求が Core への割り込みになった場合には，速やかに MPND をリセットする．

送信 MMC に対してパケットの送信を依頼する前に，MMC が同時に別のパケットの処理を行わないように，MMC のバッファのドレイン処理を行う．

5.2.2 PEND

Core プログラムは，各 CC の pending write transaction を数えるカウンタ SC(0:3)を用意する．

1. CC#Sから表 2 に示すパケットが送出されると，PEND(S).HC が increment され，X_PEND(S)がアサートされる．
2. Core に割り込みがかかると，Core は，それが PEND(S).HC を increment したパケットであると認識し，SC(S)を increment し，PEND(S).HC を decrement する．同時に PEND(S).SF がセットされるので，X_PEND(S)はアサートされ続ける．
3. SC(S')が 0 になったら，PEND(S').SF をリセットする．