1D−6

フォールトトレラントシステムのメモリ制御

福田　洋之 , 石田　仁志 , 徳永　雄一 , 峯崎　春洋

三菱電機 (株) 情報電子研究所

1　はじめに

密結合マルチプロセッサ方式を採用したフォールトトレラントシステムでは、各プロセッサはキャッシュメモリを持ち、システムバスを通して、メインメモリをアクセスする形式をとる。メモリを二重化している場合、キャッシュからメモリへのデータ転送が二回になり、バス負荷が高くなる。本報告では、このデータ転送を二回から一回にするハードウェア構成・を紹介する。

2　システム構成

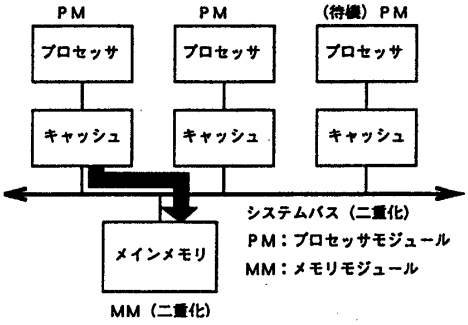

図 1: システム構成

図 1がマルチプロセッサシステムの構成図である。各プロセッサは独立にキャッシュを持っており、これをプロセッサモジュールと呼ぶ。各プロセッサモジュールはメインメモリにデータを共有している。また、メインメモリ、システムバスは信頼性向上のため、二重化されている。プロセッサモジュールの故障発生時においては、故障したモジュールがシステムから切り離され、予備系のモジュールが処理を続ける。

3　リカバリポイントの設定

プロセッサモジュールに故障が起こると、システムは過去のある時点まで戻って、そこから処理を再開する。その点を「リカバリポイント」と呼び、再開のための処理を「リカバリ処理」という。図 2において、点A、Bでリカバリポイントの設定が行なわれる。ここでは各プロセッサモジュール内のキャッシュメモリ内の更新されたブロックと、プロセッサのレジスタ内のデータを、メインメモリに書き戻す処理が行なわれる [1]。

プロセッサモジュールの故障が検出された時は、リカバリポイント設定時のレジスタのデータを待機系のプロセッサにロードして、リカバリポイントから再度実行させる。

A Memory control method in a fault-tolerant system
Hiroyuki FUKUDA, Hitoshi ISHIDA, Yuuichi TOKUNAGA,
Shunyou MINESAKI
Mitsubishi Electric Corp.

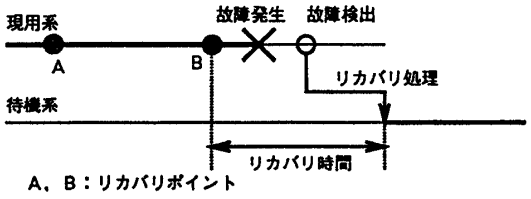

図 2: リカバリポイント設定

4　従来方式

従来のシステムでは、二重化メインメモリ（一方をプライマリメモリ、もう一方をシャドウメモリと呼ぶ）へのライト動作は二回に分けて行なわれている [2]。

一度に二つのメモリにデータ転送を行なうと、転送中のエラーにより転送が中断した場合、両メモリ内のデータは新旧リカバリポイントどちらかの状態であるとは保証できなくなるため、データの復帰が困難となる。そのため、シャドウメモリにデータを転送し正常終了した後、プライマリメモリにデータを転送するという方式をとっている。この方式ならば一回目のシャドウメモリへの転送失敗に対しては、プライマリメモリからシャドウメモリへデータを転送することで旧リカバリポイントの状態で復帰することができるし、二回目のプライマリメモリへのデータ転送失敗に対しては、シャドウメモリのデータをプライマリメモリへ転送することで新リカバリポイントの状態へ復帰することができる。

しかし、データ転送を二回行なわなければならず、そのことが、システム性能低下原因の一つになる。

5　提案方式

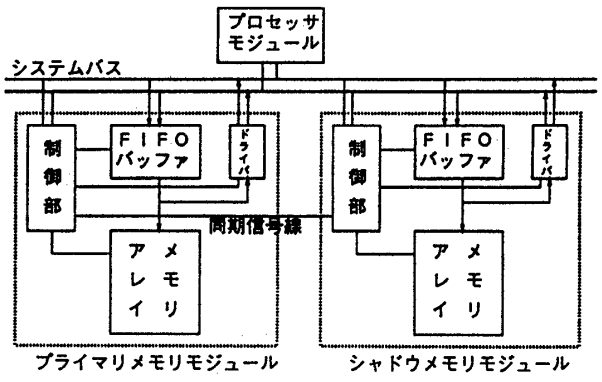

図 3: ブロック構成

メインメモリへのライト動作が指定された時には、2つのメモリモジュールは、それぞれの転送データ受け取り用のFIFOバッファで（図3）データを同時に受け取る。制御部はシステムバスとの信号のやり取りを行なう。制御入力信号は2つのメモリモジュールの制御部がどちらも受け取るが、メモリからの制御信号はプライマリメモリが代表して出力する。

なお、リード動作が指定された時は、データはドライバを通して、システムバスの要求者にデータを転送するが、これも、プライマリメモリだけが応答して、その処理を行なう。

プライマリメモリの故障が検出されてシステムから切り離された時は、同期信号線を介して、シャドウメモリに故障が通報され、シャドウメモリがプライマリメモリに代わってシステムバスとのやり取りを行ない、処理を単独で行なう。また、シャドウメモリの故障が検出された時には、同期信号線を介して、プライマリメモリに故障が通報され、プライマリメモリがその後単独で処理を行なう。

5.1 正常転送時

ライト動作において、アドレスフェーズで該当アドレスが来ると、プライマリメモリが代表して応答し、どちらのFIFOバッファもデータ受け取り状態に入り、一度の転送で両バッファにデータが書き込まれる。

2つのメモリモジュールは同期信号線でお互いのFIFOバッファへのデータ書き込みを確認した後、それぞれがメモリアレイにデータを書き込む。2つのメモリモジュールのメモリアレイへのライト動作終了後、プライマリメモリがプロセッサモジュールにACK信号を返し、それを受け取って初めてプロセッサモジュールは次の処理に入る。

5.2 転送エラー発生時

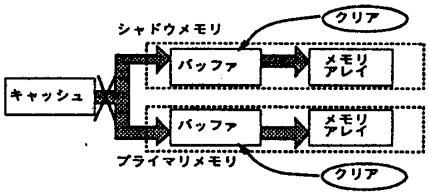

図 4: 転送エラー発生

ライト動作において、パリティエラーが検出されて、データの転送が失敗することがある。この時は、プライマリメモリの制御部がプロセッサモジュールへエラーを通達し、両方のFIFOバッファの内容をクリアさせ、データ転送の再実行に備える。

6 考察

図5のように、従来方式はデータ転送を二回行なうのに対して、提案方式では、データをアクセス速度の速い高速バッファに一回転送するだけである。

従来方式では、メモリに書き込む時間は、バス転送時間より長いため、プロセッサモジュール側に待ちが生じて、バス転送速度が低下する。よって、データ転送時間はメモリアクセス時間に依存する。

提案方式では、高速バッファのためバスの転送速度低下が起こらない。よって、バス使用時間はバス転送時間に依存する。また、FIFOバッファからメモリアレイへの書き込み時間は、メモリのアクセス時間に依存する。

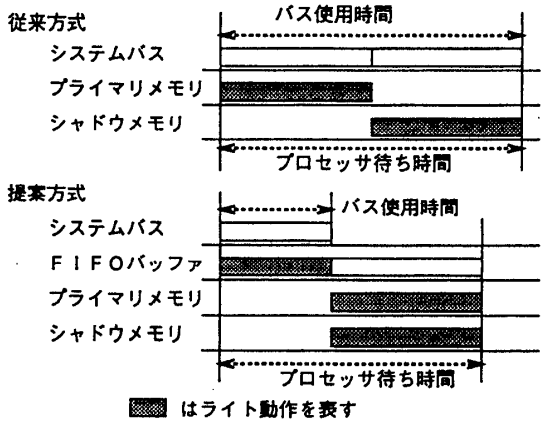

図 5: データ転送タイムチャート

システムバスのデータ転送時間を $0.25\mu s/block$、メモリのアクセス時間を $0.34\mu s/block$、転送長250ブロックとして、計算を行なうと、計算結果は下表のようになり、本方式により、バスの使用時間は4割弱に、また、プロセッサの待ち時間は9割弱に改善される。特に、FIFOバッファにデータ転送してから、プライマリメモリがACKを返すまでの間にバスが空くために生じるバスの使用時間の減少は、他のプロセッサモジュールのI／Oアクセスなどに当てることができるため、システム性能向上に大きく寄与する。

	バス使用時間	プロセッサ待ち時間
従来方式	170 μs(100%)	170 μs(100%)
提案方式	62.5 μs(37%)	147.5 μs(87%)

表1．転送時間比較

7 まとめ

提案方式により、プロセッサ待ち時間の減少、バス使用時間の大幅な減少が図られ、リカバリポイント設定に伴うオーバーヘッドを小さくできる。今後の課題として、バス負荷減少のメリットを生かして、二重化メモリを複数対使用するシステムについて検討を加えたいと考える。

■ 参考文献

[1]. 峯崎、福田、徳永、石田、"リアルタイムマルチプロセッサシステムにおけるリカバリポイントの設定条件"、情報処理学会第 *44* 回全国大会 *(1992)*.

[2]. *Philip A. Bernstein, "Sequoia: a fault-tolerant tightly coupled multiprocessor for transaction processing", IEEE Computer, Feb, 37-45, (1988).*