

大塚 義浩 平田 明
三菱電機（株）情報技術総合研究所

1. はじめに

一般に、計算機システムにおける割込処理は、オペレーティングシステム等に組み込まれ、プログラム実行中に、何らかの割込要因によってプログラムの実行を一時中断し、この要因に応じた処理プログラムを実行する。通常、プロセッサが処理する割込の種類とその数は限られているため、割込制御装置や周辺デバイスに組み込まれたレジスタを使うことで、多数の周辺デバイスからの割込を処理できるようにした方式がある。また、割込はその発生から処理されるまでの範囲がハードウェアからソフトウェアにまたいでいるため、割込に関連した障害の原因を検出するのは容易ではない。そこで、割込処理の動作検証が容易にできる方式について検討した。本稿では、割込制御装置とその検証方法について述べる。

2. 割込とマスク

通常の割込制御装置は、周辺デバイスからの割込要求信号(IRQ)が入力されると、その中からプライオリティの高い IRQ が選択され、IRQ をエンコードした信号(IRE)をプロセッサへ通知する。プロセッサは IRE を受信すると、割込ハンドラによってその IRE に対応する割込処理を選択して起動する。通常、IRE を割込通知に使用するプロセッサは、その IRE 入力ピンの数で割込種類の数が決まる。例えば、IRE 入力ピンが 4 本であれば、すなわち 4 ビットであり、この場合は 15 レベルまでの割込種類しか区別することができない(残り 1 つは割込のない状態を示す)。そこで、より多くの周辺デバイスによる割込を可能にするために、割込制御装置にレジスタを組み込んだ方式をとっている例は数多くある。このような方式をとった割込制御装置では、割込状態表示レジスタ、割込マスクレジスタ、エンコード状態表示レジスタが組み込まれているケースが多い。その動作は次のようなものである。

- ① 周辺デバイスからの IRQ が割込制御装置に

通知されると、割込状態表示レジスタのうち、接続先の周辺デバイスを示すビットが ON になる。複数の IRQ が同時に通知されていればそれらのビットも全て ON になる。

- ② プライオリティエンコードによって複数の IRQ の中からプライオリティの高い IRQ が選択され、その IRQ をエンコードした IRE がプロセッサに対して出力される。このとき、出力中の IRE がエンコード状態表示レジスタに示される。
- ③ 割込マスクレジスタのいずれかのビットを ON にすると、該当する周辺デバイスからの IRQ はエンコードされず、プロセッサに対しても出力されなくなる。IRE が出力中のときにその IRQ 発生元の周辺デバイスに対するビットを ON にすると、その IRE 出力は停止される。例えば、プロセッサが IRE を受信すると、割込ハンドラがこのマスクを ON にしてから該当する割込処理プログラムを起動することによって、システムを IRQ からすばやく解放するといった例が一般的である。

しかし、このような割込制御装置であっても、システム開発のときに割込制御処理の動作検証をするという目的には、十分ではない。なぜなら、ハードウェアの割込発生からソフトウェアの割込処理に至るまでの広い範囲において、割込に関連した障害が発生した場合の原因追求は複雑かつ困難である。また、周辺デバイスが多くなると、その分、割込の数が増え、割込処理の動作検証がより複雑になってしまう。

3. 割込制御装置の実現

3.1 疑似割込生成機能

前述の割込制御装置に、疑似割込生成機能を追加する。これは、疑似割込生成レジスタを新たに追加することによって実現できる。このレジスタは、いずれかのビットを ON にすると、そのビットが示す

周辺デバイスからあたかも IRQ が通知されたかのように、疑似 IRQ を生成してプライオリティエンコーダへ通知する仕組みである。また、生成した疑似 IRQ を反映して、割込状態表示レジスタの該当するビットが ON になるようにする(図 1)。

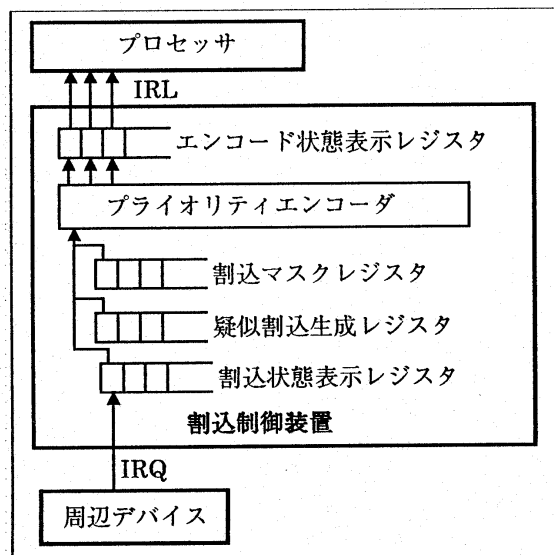


図 1. 疑似割込生成機能

3.2 割込履歴とその管理

さらに、前述の割込制御装置に割込履歴機能を追加する。この割込履歴機能は、割込履歴レジスタによって実現する。この割込履歴レジスタは、割込状態表示レジスタと割込マスクレジスタによって変化する特徴を持ち、IRQ が入力されている間、ソフトウェアによってマスクが ON になると、割込履歴レジスタの該当するビットが連動して ON になる。そしてマスクが ON の状態で IRQ が停止すると、割込履歴レジスタの該当するビットが OFF になる。もし IRQ が一度も停止しないうちからソフトウェアがマスクを OFF にしようとする、割込制御装置はプロセッサへ異常を通知するようにする(図 2)。

4. 割込制御検証方法

4.1 疑似割込生成機能による検証

疑似割込生成機能を使うと、各々の IRQ に対するプロセッサの割込ハンドラによるソフトウェアの割込処理起動までの動作を検証することができる。例えば、割込ハンドラが異なる IRQ に対するマスクを操作していないかどうか、異なる割込処理プログラムを起動していないかどうか容易に検証できる。

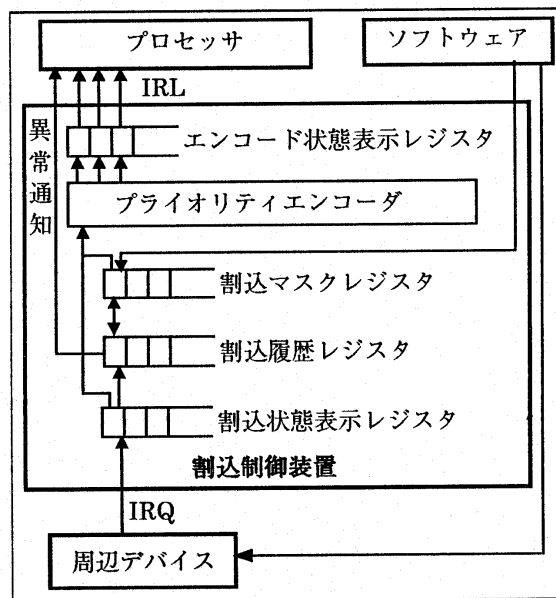


図 2. 割込履歴機能

また、起動された割込処理プログラムがマスクの解除処理を実行しているかどうか、異なるマスクを操作していないかどうか容易に検証することも可能である。

4.2 割込履歴機能による検証

割込履歴機能を使うと、IRQ に対する割込処理プログラムの動作を検証することができる。例えば、周辺デバイスからの IRQ が通知され、割込ハンドラがマスクを ON にして、該当する割込処理プログラムを起動したあと、割込処理プログラムが周辺デバイスの IRQ を解除せずにマスクを OFF にしていないかどうか容易に検証できる。もし割込処理プログラムで正常に処理がなされているにも関わらず、IRQ が停止していないのであれば、当然、周辺デバイス側に問題がある可能性も考えられる。

5. おわりに

これらの機能を使用することで、割込関連の障害原因がハードウェアかソフトウェアかといった最初の切り分けが容易になる。疑似割込生成機能では、ソフトウェアにおける割込ハンドラおよび割込処理起動の基本的な動作検証が容易にできるようになる。さらに、割込履歴機能には、ソフトウェアの割込処理の検証だけでなく、周辺デバイスの割込動作の正当性を検証することも可能である。システム開発においては、開発時間の短縮といった効果があり、非常に有効な検証方法である。