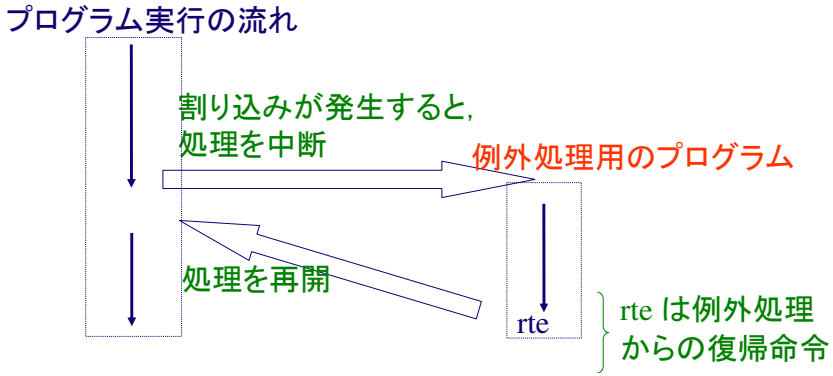
スーパバイザモード，特権命令，割り込み

到達目標

- CPUの割り込みメカニズム
  - 割り込みの種類ごとに，所定の例外処理が呼び出される
  - スーパーバイザモードに，自動的に切り替わる
  - 割り込み終了後に「元のモード」に戻る
- ハードウエア割り込みについて
  - 割り込み禁止

割り込み発生時のCPUの挙動

- 現在の処理を中断
- 例外処理用のプログラム（ハンドラともいう）が起動される



割り込みの発生

- プログラム実行を中断せねばならないような，何かの要因が起きたことを「割り込みの発生」という

  例）
  - プロセッサ外部からのハードウエアからの要求
  - 現在実行中のプログラム内の何らかのトラブル

  など

割り込みの種類

- 外部割り込み （外部要因）
  - ハードウエアリセット
  - 入出力： 入出力装置の処理終了や何かの要求
  - タイマ： ある一定時間が経過したことの通知
  - マシンチェック： ハードウエアの何らかの異常通知

  など
- 内部割り込み （内部要因）
  - 演算例外： オーバーフロー，0による除算
  - 特権命令違反： システム管理命令をユーザモードで実行してしまった場合
  - トラップ命令： オペレーティングシステムの機能呼び出し
  - アドレスエラー： ワードデータが奇数アドレスからあった

  など

外部割り込みの例

- 入出力
  - 入出力用のサブプロセッサ（DMAコントローラなど）からの信号
  - 入力処理や，出力処理の終了などを信号としてCPUに伝える
- マシンチェック
  - メモリのエラーチェック回路（ECC）からの信号
  - メモリに何らかの異常があることを伝える
- タイマ
  - タイマから一定時間ごとに信号が来て，時計機能を実現

内部割り込みの例

- 浮動小数点演算のオーバーフロー
- 除算命令で，0による除算
- システムのデバッグのために，一命令の実行ごとに強制的に掛かる割り込み
- 特権命令を，ユーザモードで実行した場合

→ プログラム内に何らかの特別な条件が成立することによる割り込み （外部とは関係ない）

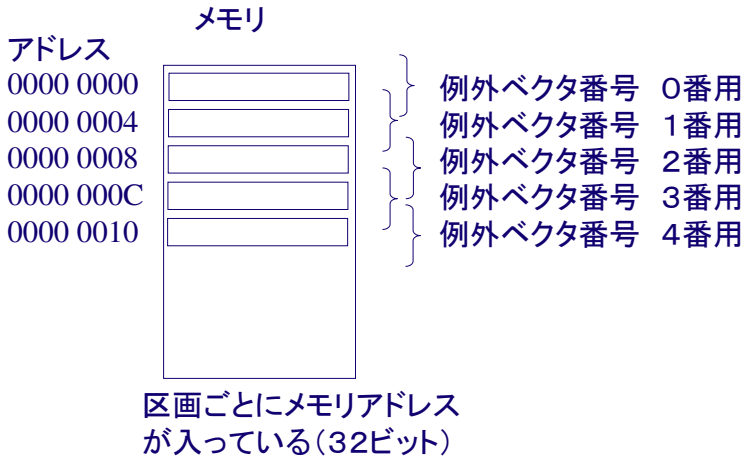
割り込みメカニズムの必要性

- 実時間処理
  - 入力があったら，現在の処理を中断して，特別な処理を直ちに行いたい．
- システムの利用効率の向上
  - 「入力が来たか」などを常に見張っているのはCPUの無駄．割り込み機能を利用すれば，見張る必要なし．
- オペレーティングシステムの実現
  - プログラムは普通のモード（ユーザモード）で動くが，オペレーティングシステムの機能を呼び出すときは，自動的にスーパバイザモード（特権命令が実行可能になるような特別なモードのこと）に切り替える

例外ベクタ方式

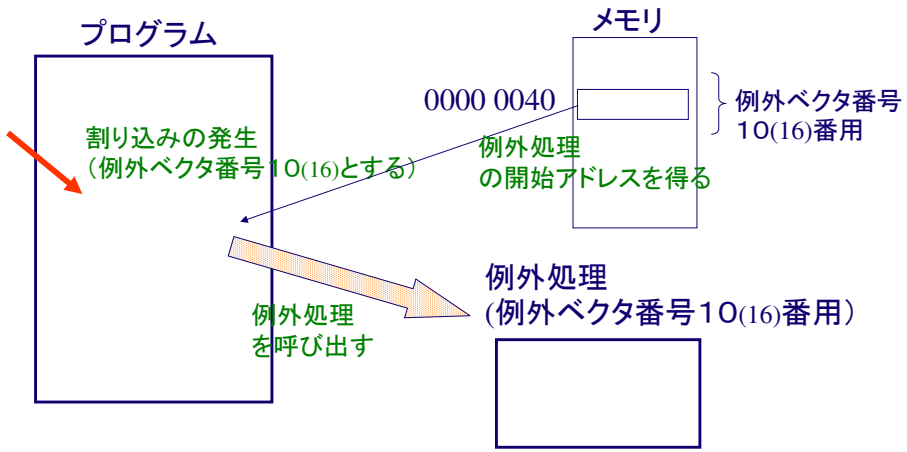
- 例外ベクタ番号
  - 割り込みの種類ごとに番号（例外ベクタ番号）が決まっている
- 例外処理プログラムの開始アドレスの決定法
  - CPUは，例外ベクタ番号を使って，メモリ内の所定のアドレスを読み，例外処理プログラムの開始アドレスを得る
  - あらかじめ，メモリの所定のアドレスに「例外処理の開始アドレス」を書き込んでおく

例外ベクタ方式

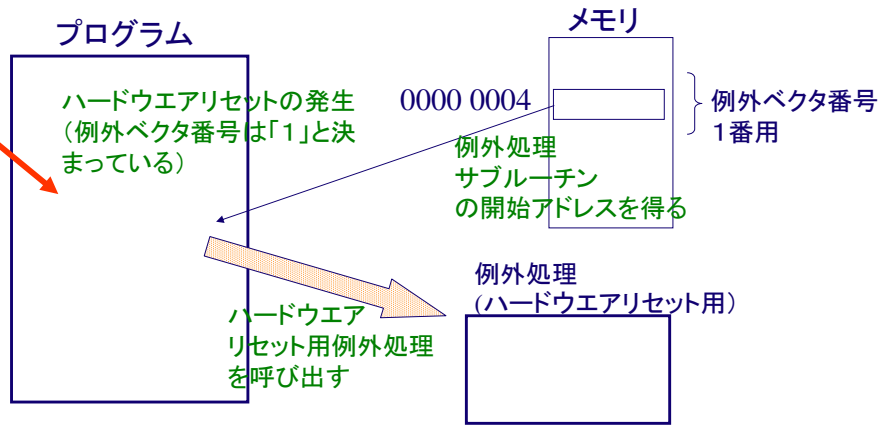


区画ごとにメモリアドレスが入っている（32ビット）

例外ベクタ方式



ハードウエアリセットの場合



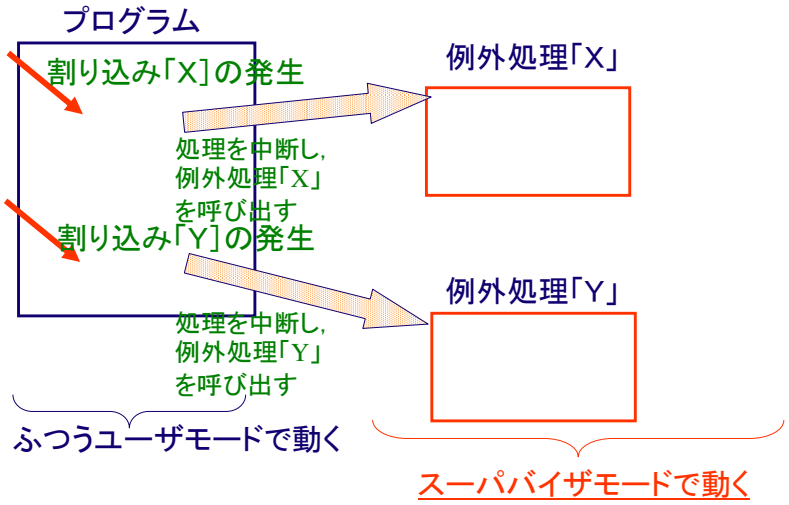
ここまでのまとめ

- 割り込みの種類ごとに，所定の例外処理が呼び出される仕組み
  - 種類に応じて「ベクタ」が割り当てられている
    → 割り込みの識別番号のこと
  - ベクタ番号と，例外処理プログラムの開始アドレスの対応表がメモリ上にある

    例外ベクタ番号： 0からFF(16)

    使うべきメモリ領域： 0から3FF(16)

例外処理とスーパバイザモード



スーパバイザモードとは

- 特権命令を実行可能なモード
- ステータスレジスタ(68000CPU内のレジスタ)のSビットの値が1

特権命令

- システムの信頼性を高めるために，いくつかの命令は特権化されている

  68000では，

  RESET　　リセット

  move ・・・, %sr　　ステータスレジスタの書き換え

  など

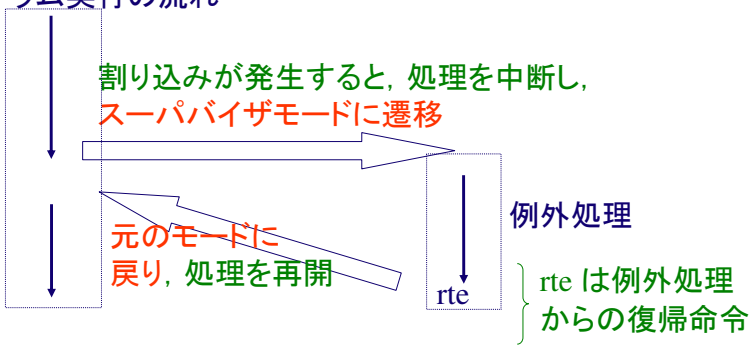
ユーザモードと スーパバイザモードの違い

- ユーザモード
  - ステータスレジスタのSビットが0
  - 特権命令を実行できない
  - スタックポインタとして，ユーザスタックポインタが使われる
- スーパバイザモード
  - ステータスレジスタのSビットが1
  - 特権命令を実行できる
  - スタックポインタとして，スーパバイザスタックポインタが使われる

割り込み時の「モード」の変化

- 「割り込み」が発生すると:
  - スーパバイザモード（特権命令が実行可能になるような特別なモードのこと）に遷移
  - 所定の例外処理プログラムが呼び出される

プログラム実行の流れ



割り込みの発生によって 何が行われるのか

① 通常のプログラム実行を中断
  - プログラムカウンタの値をシステムスタックに保存
  - ステータスレジスタの値もシステムスタックに保存

② スーパバイザモードに遷移
  - ステータスレジスタ内の「S（スーパバイザ）」ビットを1に書き換え

③ 所定の例外処理を呼び出す
  - 例外処理の開始アドレスの取得
  - プログラムカウンタ値の書き換え

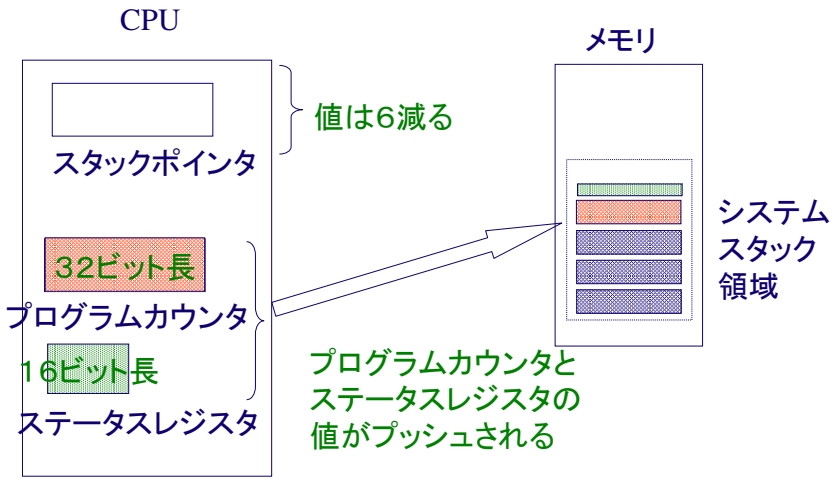
例外処理からの復帰で何が 行われるのか

例外処理からの復帰命令： rte

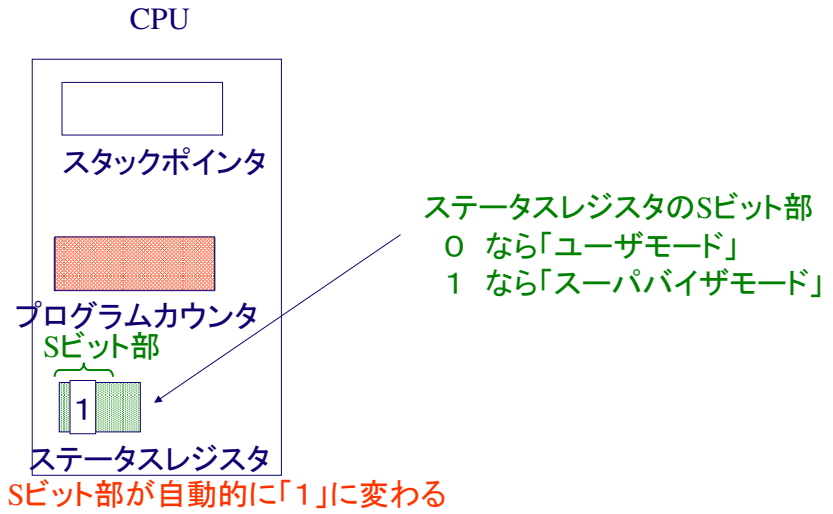
④ 元のモードに遷移
  - システムスタックに保存されていた値を使って，ステータスレジスタを元に戻す

⑤ 通常のプログラム実行を再開
  - システムスタックに保存されていた値を使って，プログラムカウンタを元に戻す

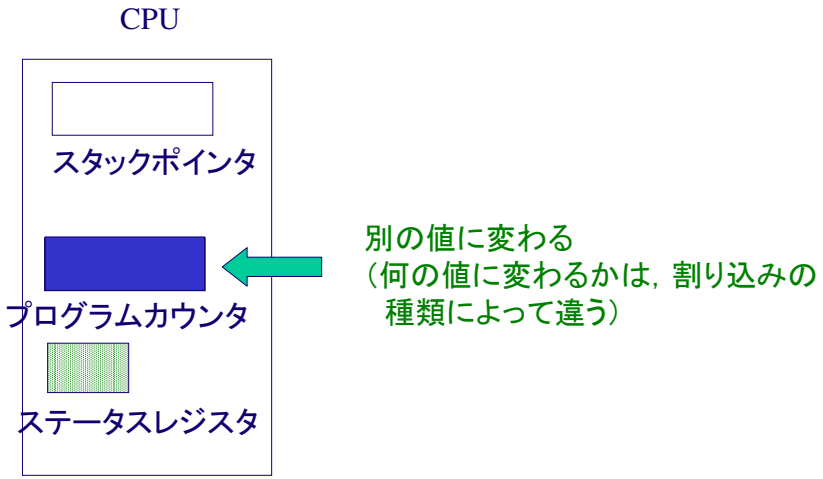
① 通常のプログラム実行を中断



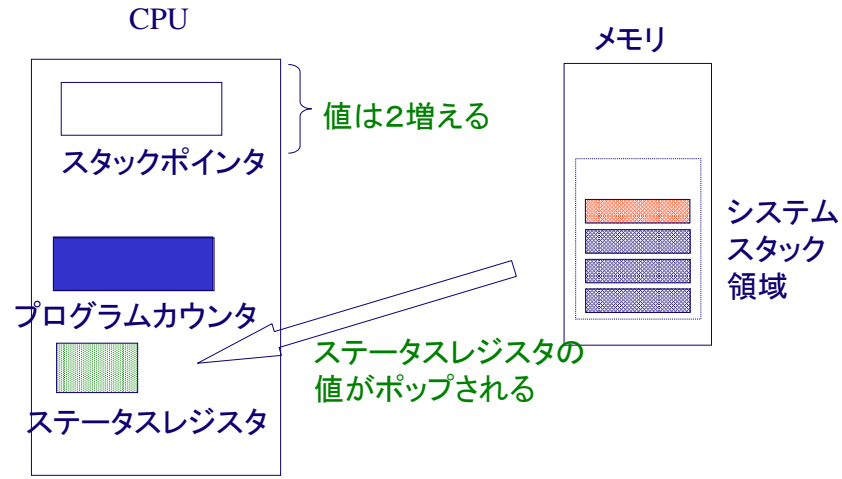
② スーパバイザモードに遷移



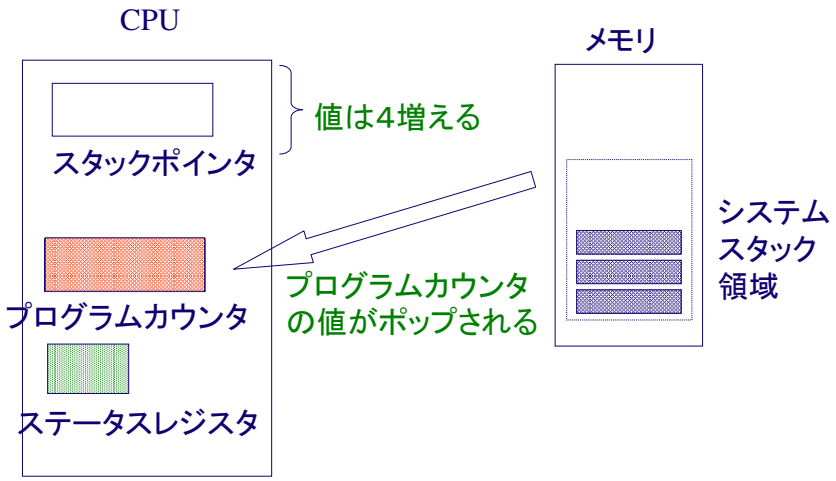
③ 所定の例外処理を呼び出す



④ 元のモードに遷移

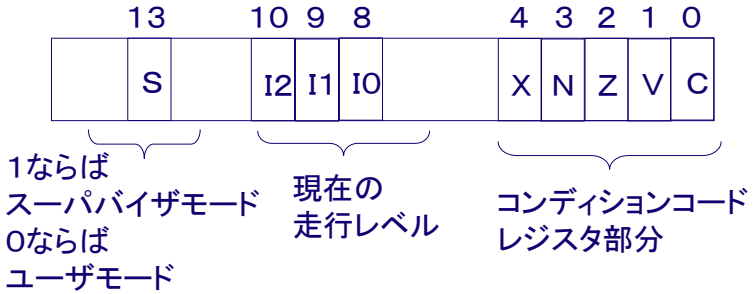


⑤ 通常のプログラム実行を再開



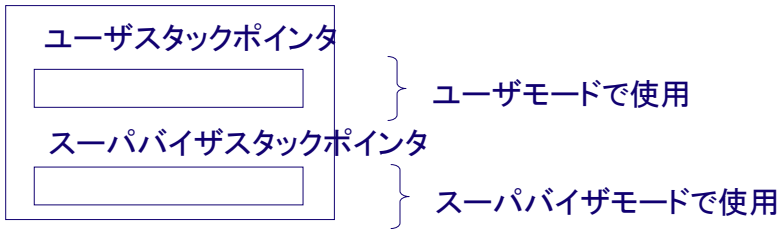
ステータスレジスタ

- 16ビットの長さ
- 下半分にコンディションコードレジスタを含む



ユーザスタックポインタと スーパバイザスタックポインタ

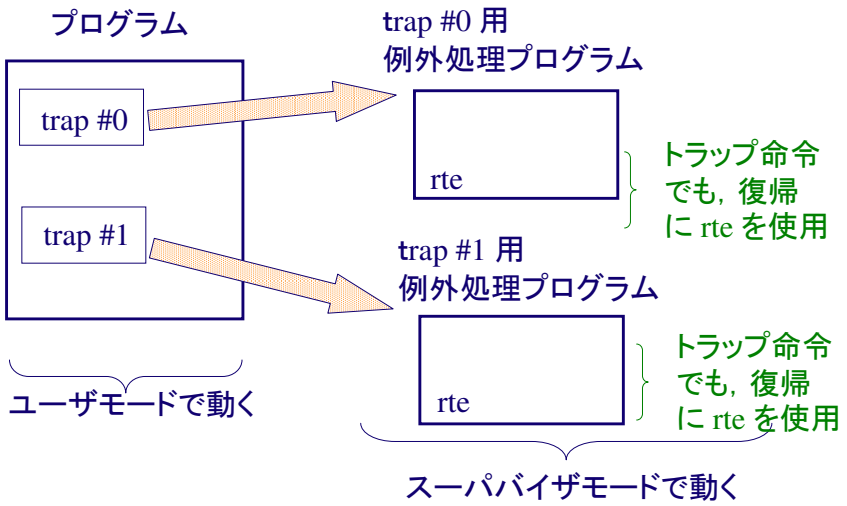
- 68000 には，2個のスタックポインタがあり，モードに応じて，使われるスタックポインタが決まる
- プログラムでは，ふつう，単に　%sp と書くだけ



トラップ命令

- ユーザモード，スーパバイザモードという2つのモードがあることが，オペレーティングシステムの実現に必須
- トラップ命令の使用法
  - 「番号」を付ける
    例） trap #0，trap #1
- トラップ命令の機能
  - ユーザモードから，スーパバイザモードへの切り替え（遷移）
  - 「番号」に応じて，定められたプログラム（サブルーチン）が自動的に呼び出される

トラップ命令



トラップ命令の用途

- オペレーティングシステムの機能を呼び出すとき トラップ命令を使う
- 通常，プログラムはユーザモードで動く
  - ユーザモードでは，「特権命令」が実行できないなどの「保護」が働く
- システムの資源のアクセスなどは，オペレーティングシステムに依頼するという形を取る
  - オペレーティングシステムの機能は，トラップ命令で呼び出され，スーパバイザモードで実行される
  - スーパバイザモードなので，ステータスレジスタの操作など特権命令が実行可能

トラップ命令の使い方

- 「番号」は0から15まで

  trap #0, trap #1 ..., trap #15
- それぞれ，例外ベクタ番号は，20(16)から2F(16)まで

  20, 21, ..., 2F(16)

ハードウエア割り込み

- 外部要因の割り込み
- CPUの端子(IPL0, IPL1, IPL2)に信号が来ることで発生

  普通： 000 (3つとも0)

  割り込み： 001, 010, 011, 100, 101, 110, 111

  3ビットのビットパターン
- 「例外ベクタ番号」(例外の種類を表す番号)も，外部からの信号で受け取る

割り込みレベル

- 割り込みの優先度のこと
- ハードウエア割り込み時の，CPUの端子（IPL0，IPL1，IPL2）の信号ビットパターンで，1から7に決まる

IPL0, IPL1, IPL2	割り込みレベル
0 0 1	1
0 1 0	2
0 1 1	3
1 0 0	4
1 0 1	5
1 1 0	6
1 1 1	7

割り込みマスク

- ステータスレジスタの3つのビット（I2，I1，I0）に，現在の例外処理のレベルが入っている
- より高レベルの割り込みのみを受け付ける

  例） いま，ステータスレジスタの I2, I1, I0 に「5」が入っているとしよう

  ハードウエア割り込みが来ると

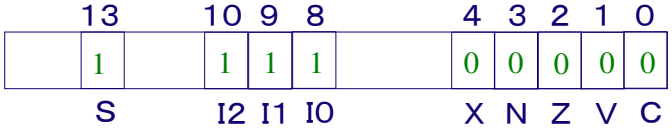
  優先度が5以下： 無視される

  優先度が6以上： 受け付ける
- 例外として，優先度7のものだけは，無視されることはない

  → I2, I1, I0 に「7」が入っていても，「6」が入っているのと同じ

割り込み禁止

例） move #0x2700, %sr



- ステータスレジスタの I2, I1, I0 を強制的に「7」に設定
- すると，優先度6以下のハードウエア割り込みは無視されることになる
- move #0x2400, %sr など，他の走行レベルに設定することももちろん可能
- 意図的に I2, I1, I0 を書き換えることを「割り込み禁止」という

割り込み禁止の手順

割り込みを禁止しておきたい」ときは

- ステータスレジスタを，システムスタックに保存

  move.w %sr, -(%sp)
- 割り込み禁止

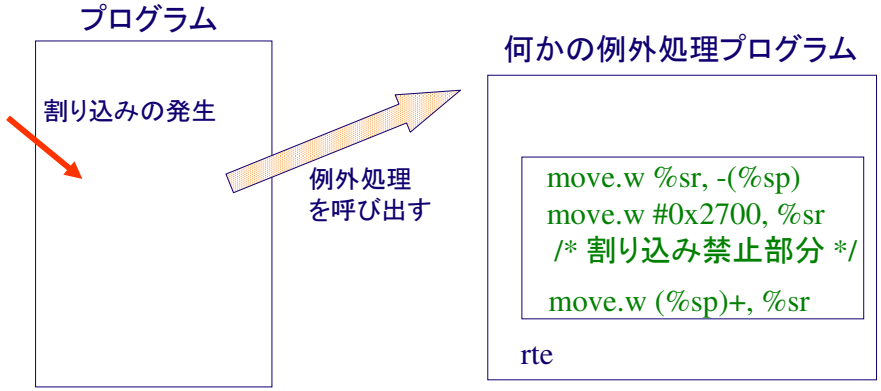
  move.w #0x2700, %sr

割り込み禁止をやめてもよくなったら，

- ステータスレジスタの復帰

  move.w (%sp)+, %sr

割り込み禁止の手順



例外処理プログラム中では，しばしば，
割り込み禁止が必要になる