第3回の課題について

ある割込み処理中にほかの割込みを許す場合、その処理には何を注意するべきか。

以下のような項目が考えられそうです。

- 割込みの禁止と受付けについて
  割込み事象には優先度が設定されており、割込み処理実行中に優先度のより高いものが発生した場合に、それに対する処理を行うには割込みを許可しておくことになります。
  しかし、割込み処理の初期段階には割込み情報の収集やその後の処理の準備などがあるので、この処理の間に割込みが発生すると処理に矛盾が生じてしまいます。そのため、このような初期段階では割込み禁止としますが、割込み禁止状態はできるだけ早く解除するようにします。

- 処理プログラムの構造について
  割込み処理の実行中に他の割込みを受付けて処理可能とするには、他の割込み処理を呼出す際にレジスタなど現在の実行状態やデータをスタックに保存し、呼出し元へ処理が戻ったときにスタックからそれらのデータを復帰するような処理が必要になります。

以下の点がまとめになります

- 割込みハンドラでの処理は必要最小限にし、それ以外は通常処理する
- 割込みハンドラでの割込み禁止の状態は、必要最小限にする
- 緊急性や重要性を考慮して割込みレベルを適切に設計する

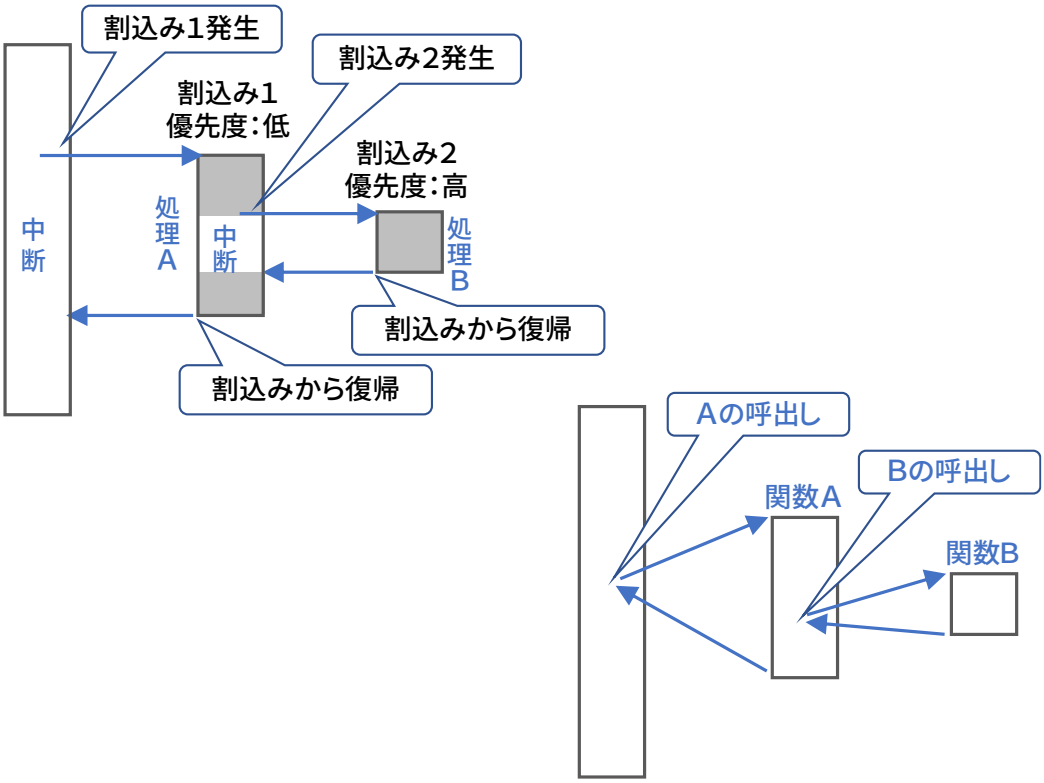
【提出いただいた解答へのコメント】

「割込みレベル (優先度) を確認する」や「PC や PSW を保存する」、「上書きやデータの破壊に注意する」、「入れ子で繰り返して実行できるように」、「同時に割込みが発生した場合の処理」などといった点を解答いただきました。処理のオーバーヘッドに関する注意点もありますが、正しく動作するという観点から、これらが割込みの処理で重要なことです。「共有資源を排他制御する」との解答もありました。割込み禁止区間が排他の区間に対応すると考えられます。

どのようにし「優先度を確認する」か、などについて検討することは必要ありませんが、通常、現在発生している割込みのレベルを保持するレジスタがあって、それを参照して新たに発生した割込みのレベルと比較し、その割り込みを許すかどうかがハードウェアの機構として決定されるようになっています。

割込み処理の実行も関数の処理 (呼出し) と同じように考えられますので、入れ子の実行 (呼出し) に対して、スタックを用いた実装をソフトウェア主体で行います。割込み処理中に同じ割込みが発生した場合に関する注意点を挙げた方もいらっしゃいましたが、スタッ

クの利用によって再帰的な処理にも対応できます。前回スライドの p.11 の図で、割込み 1 に対する処理 A を実行中に優先度の高い 割込み 2 が発生すると、その処理 B を実行しますが、これは、関数 A の中で関数 B を呼び出した場合と似た動作であり、この時点で A の処理は中断されていると考えられます。B の処理が終了すると、「復帰」の処理を行うことで A の処理が中断点から再開されます。



実は、PC や PSW は割込み発生時にハードウェアで自動的に保存されます。それに加えて、割込みハンドラのプログラムでは汎用レジスタなどを保存 (退避) することになります。これらのデータが破壊されないように、スタックを使用します。前回や今回の講義でプロセスのメモリ空間にスタックがあるとしていますが、それとは別にカーネルの中に割込み処理などカーネルで使用するためのスタックがあります (教科書では 4.4 で「メモリの特定場所」となっています)。

割込みから復帰するときに退避した汎用レジスタなどを回復し、割込みから復帰する命令を実行すると、退避されていた PC と PSW が回復されて、この割込みを受け付けた場所の次から実行 (多重割込みの場合は中断されていた割込み処理の再開) となります。