平成 24 年度 オペレーティングシステム 中間試験問題 平成 24 年 5 月 30 日

問題 1.
プロセスの飢餓 (Starvation) 状態とはどういう状態か、デッドロック状態との違いを明確にしながら説明し、どのような状況の時に生じるか例を 2 つ上げて説明せよ。

問題 2.
実行中のプロセスから他のプロセスに実行を移すことをプロセスコンテキストスイッチあるいはコンテクストスイッチと呼ぶ。コンテキストスイッチに関して以下の問いに答えよ。

1. コンテキストスイッチが生じるきっかけを 2 つあげて説明せよ。
2. プロセス毎に独立した仮想メモリ空間を持たせるために CPU アーキテクチャが提供しているメモリ管理機構の概要を図を用いて説明せよ。
3. コンテキストスイッチ処理のためには、CPU レジスタ群の退避・復帰、仮想メモリ空間の切り替えが必要となる。上記メモリ管理機構を用いて、コンテキストスイッチ処理の概要を説明せよ。

問題 3.
以下の表に示すプロセス群が、実行可能待ち行列に格納されているとする。CPU スケジューリングに関する以下の問いに答えよ。

Process	Burst Time
P1	220 ミリ秒
P2	60 ミリ秒
P3	120 ミリ秒

1. ラウンドロビン CPU スケジューリングによってスケジューリングした時の様子を図示せよ。また、平均ターンアラウンド時間および平均待ち時間を求めよ。小数点以下は切り捨ててよい。なお、上記表の順番に実行され、プロセスは一回実行権を得ると 50 ミリ秒 CPU 時間を消費することが可能とする。プロセス切り替えにかかる時間は無視する。
2. Shortest Job First CPU スケジューリングによってスケジューリングした時の様子を図示せよ。また、平均ターンアラウンド時間および平均待ち時間を求めよ。小数点以下は切り捨ててよい。

問題 4.
以下のプログラムでは、producer と consumer1, consumer2 は、それぞれプロセスとして起動されると仮定する。3 つのプロセスは、push、pop を用いてデータ授受を行っている。以下の問いに答えよ。

```
#define MAX  8
int data[MAX];
int count = 0;
void push(int d) {
 while (count == MAX);
 data[count] = d;
 count = count + 1;
}
int pop(void) {
 int val;
 while (count == 0);
 val = data[count - 1];
 count = count - 1;
 return val;
}
```

```
void producer() {
 int  i;
 for (i = 0; i < 32; i++) {
  /* ... */
  push(i);
 }
}
void consumer1() {
 int  i, val;
 for (i = 0; i < 16; i++) {
  val = pop();
  /* ... */
 }
}
void consumer2() {
 int  i, val;
 for (i = 0; i < 16; i++) {
  val = pop();
  /* ... */
 }
}
```

1. 本プログラムはプロセス間の同期をとっていないために正しく動かない。どのような状況のときに問題が生じるか、実行順序を示して問題点を示せ。

2. 以下に示す test_and_set 関数が用意されていると仮定する。本関数は他のプロセスの実行と不可分に実行される。

   ```
   int test_and_set(int *p) {  int  tmp = *p;  *p = 1;  return tmp; }
   ```

   本関数を用いて、上記プログラムを以下のように修正したが、デッドロックする。デッドロックする条件を述べよ。

```
int   lock = 0;
void push(int d) {
 while (test_and_set(&lock) == 1);
 while (count == MAX);
 data[count] = d;
 count = count + 1;
 lock = 0;
}
```

```
int pop(void) {
 int val;
 while (test_and_set(&lock) == 1);
 while (count == 0);
 val = data[count - 1];
 count = count - 1;
 lock = 0;
 return val;
}
```

3. 上記プログラムを以下のように変えた。デッドロックはしないが、正しく動作しない。理由を述べよ。ヒント：2つのconsumerプロセスがいることに着目せよ。

```
int   lock = 0;
void push(int d) {
 while (count == MAX);
 while (test_and_set(&lock) == 1);
 data[count] = d;
 count = count + 1;
 lock = 0;
}
```

```
int pop(void) {
 int val;
 while (count == 0);
 while (test_and_set(&lock) == 1);
 val = data[count - 1];
 count = count - 1;
 lock = 0;
 return val;
}
```

4. デッドロックせず、かつ、正しく動作するpush、popプログラムを書け。