

Premiere tentative  
-----

```
sem sleepEntree = 1,
   attente = 0;
int nbThreadsEnAttente = 0;

procedure sleep()
# Bloque l'execution du processus appelant,
# sauf s'il s'agit du dernier processus encore actif.
{
  P(sleepEntree);
  nbThreadsEnAttente += 1;

  if (nbThreadsEnAttente < nbThreads) {
    V(sleepEntree);
    P(attente);
  } else {
    nbThreadsEnAttente -= 1;
    V(sleepEntree);
  }
}

procedure wakeup()
# Reactive tous les processus bloques via sleep.
{
  P(sleepEntree);
  while (nbThreadsEnAttente > 0) {
    nbThreadsEnAttente -= 1;
    V(attente);
  }
  V(sleepEntree);
}
```

Deuxieme tentative  
-----

```
sem sleepEntree = 1,
   attente = 0;
int nbThreadsEnAttente = 0;

procedure sleep()
# Bloque l'execution du processus appelant,
# sauf s'il s'agit du dernier processus encore actif.
{
  P(sleepEntree);
  nbThreadsEnAttente += 1;
  if (nbThreadsEnAttente < nbThreads) {
    V(sleepEntree);
    P(attente);
    # Si reactivation, alors sleepEntree est deja verrouille.
    nbThreadsEnAttente -= 1;
    if (nbThreadsEnAttente > 0) {
      # Il reste d'autres processus a reactiver.
      V(attente);
    } else {
      # Dernier processus a etre reactive => liberer le verrou.
      V(sleepEntree)
    }
  } else {
    # nbThreadsEnAttente == nbThreads
    # Le dernier processus ne doit pas attendre.
    nbThreadsEnAttente -= 1;
    V(sleepEntree);
  }
}

procedure wakeup()
# Reactive tous les processus bloques via sleep.
{
  P(sleepEntree);
  if (nbThreadsEnAttente > 0) {
    V(attente);
  } else {
    V(sleepEntree);
  }
}
```