**L6Z1.** Podaj w pseudokodzie semantykę **instrukcji atomowej** compare-and-swap i z jej użyciem zaimplementuj **blokadę wirującą** (ang. spin-lock). W jakich systemach komputerowych stosuje się ten typ blokad? Wymień zalety i wady blokad wirujących w porównaniu do **blokad usypiających**. Opisz rozwiązanie pośrednie, czyli **blokady adaptacyjne**.

**operacja atomowa** – operacja niepodzielna na określonym poziomie abstrakcji. W przypadku instrukcji niepodzielność polega na tym, że żaden proces nie może przerwać działania funkcji i zarejestrować stan, który istnieje podczas jej wykonywania, mimo że na poziomie sprzętowym instrukcje te mogą być wykonywane w kilku krokach.

**compare and swap** – jej działanie polega na porównaniu zawartości pewnej lokacji w pamięci z zadaną wartością, a następnie jeżeli obie wartości są równe zmodyfikowaniu jej do nowej wartości. Wynik operacji wskazuje, czy działanie zapisu zostało dokonane czy nie. Dwie wersje: zwracanie wartości logicznej lub wartości odczytanej w podanej lokalizacji przed zapisem

ver1. (ta chyba bardziej intuicyjna)
```
function cas(p : pointer to int, old : int, new : int) {
      if *p ≠ old {
            return false
      }
      *p ← new
      return true
}
```

```
ver2.
function cas(p : pointer to int, old : int, new : int) {
      old_p_val = *p
      if *p = old {
            *p ← new
      }
      return old_p_val
}
```

**blokada wirująca** (**spinlock)** - sposób realizacji synchronizacji międzyprocesowej, w którym oczekiwanie na zwolnienie blokady polega na ciągłym badaniu jej stanu. Spinlock działa więc na zasadzie aktywnego oczekiwania (za co na programistów czeka osobny krąg w piekle).

```
int owner = 0;  //wskazuje id wątku, który jest w sekcji krytycznej (0 - żaden)

void spinlock( int *lock, int thread_id ) {
      while( cas( lock, 0, thread_id) ) {
            /* do nothing */
      }
}

void unlock(int *lock) {
      *lock = 0;
}

/* sposób użycia */
{
      spinlock( &owner, my_id );
      /* kod sekcji krytycznej */
      unlock( &owner );
}
```

Spinlock nie jest efektywny w systemach z podziałem czasu – tam lepiej sprawdzają się mutexy. Natomiast w systemach wieloprocesorowych blokada wirująca może być bardziej efektywna, ponieważ pętla wykonuje się w kontekście procesu - unika się kosztownego czasowo wstrzymania procesu i przełączenia kontekstu na kod systemowy.
Podsumowując spinlocka używamy gdy:

- mamy do dyspozycji więcej niż jeden procesor
- wykonujemy krótką pracę w sekcji krytycznej
- jesteśmy w interrupt context – nie jest wskazane oczekiwanie, gdy znajdujemy się w trybie obsługi przerwania (wykorzystywane w Linux Kernel programming)

**blokada usypiająca (sleeping lock)** – blokada nie korzystająca z aktywnego czekania. Zamiast tego proces korzysta z wywołania systemowego w celu zablokowania samego siebie i zmiany stanu na oczekujący. Zostaje wtedy dodany do kolejki oczekującej związanej z danym semaforem, a kontrola przekazana jest planiście CPU.

| | SPIN LOCK | SLEEPING LOCK |
|---|---|---|
| ZADY | Nie ma sensu gdy do dyspozycji mamy jeden procesor | Nieefektywne gdy do dyspozycji wiele procesorów i mamy wiele procesów które musimy często budzić i usypiać na krótki czas. |
| | Marnuje czas procesora | Wysyłanie wątku aby szedł spać a później budzenie kosztowne czasowo |
| WALETY | Dobry do krótkiego oczekiwania | Dobry do długiego oczekiwania |

**blokada adaptacyjna** - jeśli w trakcie zakładania blokady obecna jest rywalizacja, to chwilę aktywnie czekamy optymistycznie zakładając, że blokada zostanie zwolniona za kilkadziesiąt cykli zegarowych. Jeśli nadal zajęta, to prosimy jądro o uśpienie zadania w oczekiwaniu na zwolnienie blokady.
Podsumowując: blokada adaptacyjna początkowo przez określony czas zachowuje się jak spin lock a następnie jak sleeping lock.

Źródła:
https://en.wikipedia.org/wiki/Compare-and-swap
https://www.quora.com/Is-a-spinlock-faster-than-a-mutex-lock
https://wiki.eecs.yorku.ca/course_archive/2015-16/W/3221/_media/eecs3221-w15-w7-p6.pdf
https://stackoverflow.com/questions/5869825/when-should-one-use-a-spinlock-instead-of-mutex?answertab=votes#tab-top
wykład 6b slajdy: 6,7,8,12