L6.Z7. Opisz semantykę operacji «FUTEX_WAIT» i «FUTEX_WAKE» mechanizmu `futex(2)` wykorzystywanego w systemie Linux do implementacji środków synchronizacji w przestrzeni użytkownika. Podaj w pseudokodzie implementację funkcji «`lock`» i «`unlock`» semafora binarnego korzystając wyłącznie z futeksów i atomowej instrukcji compare-and-swap. Odczyty i zapisy komórek pamięci są atomowe.
Podpowiedź: Wartość futeksa wyraża stany: (0) unlocked, (1) locked ∧ |waiters| = 0, (2) locked ∧ |waiters| ≥ 0.

futex (ang. *fast user space mutex*) – implementuje podstawowe blokowanie (podobnie jak muteks), ale unika odwoływania się do jądra, jeśli nie jest to bezwzględnie konieczne.

Składa się z dwóch części:

- usługa jądra: kolejka oczekiwania, która umożliwia oczekiwanie na blokadę wielu procesom. Procesy nie będą działać, jeśli jądro wyraźnie ich nie odblokuje. Umieszczenie procesu w kolejce oczekiwania wymaga kosztownego wywsysa, więc należy go unikać. Z tego powodu, w przypadku braku rywalizacji, futex działa w całości w przestrzeni użytkownika, nie wykonuje wywsysów.

- biblioteka użytkownika: futex to blokada o początkowej wartości 1 (wolna). Gdy wątek chce przechwycić blokadę, która jest wolna – ok, jeśli nie jest wolna – biblioteka obsługi futeksu nie wykonuje pętli, ale używa wywsysa w celu umieszczenia wątku w kolejce oczekiwania w jądrze.

Operacje na futeksie
Każda operacja futeksa startuje z user space, ale może być konieczna komunikacja z kernelem, używając wywsysa futex(2). Futex ma licznik domyślnie ustawiony na 1.

`futex_wait` zmniejsza o 1 licznik

- jeśli zdekrementowano do 0, to oznacza, ze nie ma rywalizacji i można przejść do sekcji krytycznej.
- jeśli jest ujemny, to oznacza, ze jest rywalizacja – trzeba ustawić licznik na -1 i za pośrednictwem wywsysa futex() wysłać proces do kolejki zablokowanych, czekających, aż inny proces zrobi `up` na futeksie.

`futex_wake` zwiększa o 1 licznik.

- jeśli zwiększono licznik z 0 do 1, to oznacza, ze nie było rywalizacji, więc nic nie trzeba robić.
- jeśli licznik nie jest dodatni, to trzeba ustawić go na 1 i wybudzić dowolną liczbę uśpionych procesów. Używa się do tego system wywsysa futex(2).

compare-and-swap – porównaj i zamień. Jej działanie polega na porównaniu zawartości pewnej lokacji w pamięci z zadaną wartością a następnie, jeśli obie wartości są równe, jej zmodyfikowaniu do nowej wartości. Niepodzielność operacji zapewnia, że nowa wartość jest wyznaczona na podstawie aktualnych danych. Jeśli wartość zostałaby w międzyczasie zmodyfikowana przez inny wątek to zapis by się nie powiódł. Wynik operacji musi wskazywać, czy działanie zapisu zostało wykonane czy nie.

```
futex:
0 - unlocked
1 - locked & waiters = 0
2 - locked & waiters >= 0

lock():
  while true:
    if CAS(state, 0, 1) == 0:
      break
    CAS(state, 1, 2)
      futex_wait(state, 2)


unlock():
  while true:
    if CAS(state, 1, 0) == 1:
      break
    if CAS(state, 2, 0) == 2:
      futex_wake(state, 1)
      break
```