SYSTEMY OPERACYNE – LABORATORIUM

Synchronizacja procesów z wykorzystaniem semaforów

Przygotował: Tomasz Bocheński

1. TREŚĆ ZADANIA:

Wyobraźmy sobie fabrykę, w której produkowany jest przedmiot P składający się z trzech elementów: X,Y,Z. W fabryce pracują N + M + O + 1 robotów (procesów):

- N robotów specjalizuje się w produkcji elementów X i po wyprodukowaniu elementu kładzie go na taśmie produkcyjnej (buforze) BP,
- M robotów specjalizuje się w produkcji elementów Y i po wyprodukowaniu elementu kładzie go na taśmie produkcyjnej (buforze) BY,
- robotów specjalizuje się w produkcji elementów Z i po wyprodukowaniu elementu kładzie go na taśmie produkcyjnej (buforze) BZ,
- 1 robot pobiera i składa elementy z taśm produkcyjnych BX, BY i BZ – tak powstały przedmiot kładzie na taśmie produkcyjnej (buforze) BP.

Przyjmijmy, że elementy X i Z to liczby naturalne mniejsze od 10, a element Y to znak z przedziału ‘a’-‘j’. Złożenie elementu P to operacja polegająca na stworzeniu ciągu znaków postaci XYZ1.
Należy przyjąć że taśmy produkcyjne mogą jednocześnie pomieścić najwyżej 5 elementów.
Produkcja powinna zostać zakończona po zbudowaniu 10 elementów P.
Dla uproszczenia można przyjąć, że robot składający pobiera najpierw element z kolejki X, następnie z kolejki Y a na końcu z Z. Jeśli któraś z tych kolejek jest pusta, robot oczekuje na pojawienie się elementu.
W ramach zadania należy stworzyć program symulujący pracę fabryki przy pomocy N + M + O + 1 procesów (roboty) i 3 buforów (taśmy produkcyjne). Aby zapobiec przepełnieniu buforów i jednoczesnemu wykonywaniu operacji na buforze należy wykorzystać mechanizm semaforów.
Należy zwrócić szczególną uwagę, aby żaden z elementów nie został „zdublowany” albo zagubiony.
Dodatkowo należy zasymulować czas pracy robotów przez uśpienie procesów na losową wartość z zakresu:

- <1, P> dla procesów produkujących element X,
- <1, R> dla procesów produkujących element Y,
- <1, S> dla procesów produkujących elementy Z.

Podczas implementacji, należy skorzystać z semaforów oferowanych przez system Linux. Należy pamiętać o usunięciu semaforów po zakończeniu działania programu.
Bufor można zaimplementować jako tablicę bądź listę. Pobieranie elementów ma być zgodne z zasadą FIFO.
Wartości M, N, P, R i S powinny być konfigurowalne przy uruchomieniu.

2. PRZYJĘTE ZAŁOŻENIA:

- elementy X,Y,Z to wartości typu char, gdzie X i Z to znaki z przedziału ‘0’ – ‘9’, a Y to znak z przedziału ‘a’ – ‘j’;
- taśmy produkcyjne mogą pomieścić najwyżej 5 elementów;
- produkcja zostaje zakończona po wyprodukowaniu 10 elementów typu P;
- robot składający pobiera najpierw element z kolejki X, potem z Y, potem z Z. Jeśli któraś kolejka jest pusta, robot oczekuje na pojawienie się elementów;
- czas pracy robotów symulowany jest przez uśpienie procesów na losową wartość z zakresu
    - <1, P> dla procesów produkujących element X,
    - <1, R> dla procesów produkujących element Y,
    - <1, S> dla procesów produkujących elementy Z.
- Czas pracy robota pobierającego elementy z taśm produkcyjnych i składającego je w jedną całość jest równy 0 (gotowy element P powstaje od razu po zebraniu wszystkich trzech typów elementów);
- w zadaniu wykorzystany zostanie mechanizm semaforów (zostaną wykorzystane semafory oferowane przez system Linux);
- bufor zaimplementowany zostanie jako tablica typu char z dwoma pomocniczymi zmiennymi, określającymi indeksy w tej tablicy: pierwszy wolny, pierwszy zajęty. Pobieranie elementów jest zgodne z FIFO;
- wartości M, N, O, P, R, S będą wczytywane na początku, następnie nastąpi przejście do symulacji programu.

3. PROPONOWANE ROZWIAZANIE (SPOSÓB DZIAŁANIA POSZCZEGOLNYCH PROCESÓW):

- powstaną 3 pliki wykonywalne, każdy plik wykonywalny będzie dotyczył innego typu robotów, produkujących innego typu elementy (X, Y, Z):

Roboty produkujące elementy X, Y, Z działać będą w skończonej pętli. Wyjście z pętli nastąpi gdy wartość pewnej zmiennej pomocniczej będzie równa 0, a zdarzy się to tylko w przypadku gdy ilość wyprodukowanych elementów będzie równa wartości maksymalnej (10)

Po zadanym przez użytkownika czasie dany element zostanie wyprodukowany (czyli zwrócona zostanie losowa wartość z odpowiedniego dla każdego elementu zakresu). Następnie wyprodukowany element zostanie włożony na pierwsze wolne miejsce. W przypadku braku wolnych miejsc produkcja zostanie zatrzymana (wyprodukowany produkt będzie oczekiwał na zwolnienie się miejsca na taśmie produkcyjnej). Gdy element zostanie odłożony do bufora nastąpi powrót do początku pętli. Przy operacjach wkładania do bufora zostaną wykorzystane semafory ogólny oraz binarny.

- powstanie plik wykonywalny dotyczący robota tworzącego ciągi znaków z elementów X, Y, Z:

Robot składający elementy X, Y, Z działać będzie w pętli skończonej. Wyprodukowane elementy będą odkładane na taśmie produkcyjnej. Po wyprodukowaniu elementu inkrementowany będzie licznik określający ilość wyprodukowanych elementów. Gdy licznik osiągnie wartość maksymalną (10), nastąpi wyjście z pętli.

Robot będzie kolejno pobierał elementy X, Y, Z. W razie braku jakiegokolwiek z elementów będzie on oczekiwał na pojawienie się brakującego elementu. Również w przypadku gdy wszystkie miejsca na taśmie będą już zajęte, robot będzie oczekiwał aż jakiś z produktów zostanie zdjęty z BP. Przy operacjach wkładania do bufora zostaną wykorzystane semafory ogólny oraz binarny.

- powstanie plik wykonywalny dotyczący „robota" zdejmującego elementy z BP i wyświetlającego je na ekran / zapisujący do pliku:

Program będzie działał w pętli. Po każdym wyświetlonym elemencie inkrementowany będzie licznik dotyczący liczby wyświetlonych elementów. Gdy osiągnie on wartość maksymalną (10), zostanie wyświetlony odpowiedni komunikat o zakończeniu symulacji.

- powstanie plik wykonywalny dotyczący symulacji:

Będzie on alokował pamięć, tworzył bufory oraz semafory (oraz wykonywał na nich odpowiednie operacje), uruchamiał proces dotyczący tworzenia napisów z elementów X, Y, Z, uruchamiał za pomocą funkcji fork() odpowiednią ilość procesów zajmujących się produkcją określonych elementów. Po zakończeniu symulacji będzie on odpowiedzialny za usunięcie semaforów.

4. WYKORZYSTANE STRUKTURY ORAZ STAŁE:

```
#define ROZMIAR_BUFORA 5

struct Bufor
{
  char elementy[ROZMIAR_BUFORA];
  unsigned pierwszy_wolny;
  unsigned pierwszy_zajety;
};        (struct Bufor *BX, *BY, *BZ)


struct PBufor
{
  char product[4];
  unsigned pierwszy_wolny;
  unsigned pierwszy_zajety;
};        (struct Bufor *BP)
```

5. SPOSOBY WYKORZYSTANIA SEMAFORÓW DO REALIZACJI ZADANEGO PROBLEMU:

Rodzaje wykorzystanych semaforów (z wartościami początkowymi):

Semafory ogólne (zliczające ilość elementów w buforze):

ZAJETE_X = 0, PUSTE_X = ROZMIAR_BUFORA;

ZAJETE_Y = 0; PUSTE_Y = ROZMIAR_BUFORA;

ZAJETE_Z = 0; PUSTE_Z = ROZMIAR_BUFORA;

ZAJETE_P = 0; PUSTE_P = ROZMIAR_BUFORA;

Semafory binarne (określające czy można dostać się do bufora czy nie):

MUTEX_X = 1, MUTEX_Y = 1, MUTEX_Z = 1, MUTEX_P = 1;

Pseudokod prezentujący wykorzystanie semaforów:

plik wykonywalny dotyczący robota tworzącego ciągi znaków z elementów X, Y, Z:

```
int produkt;
while( nie_wyprodukowano_10_napisow)
{
  Uśpij robota na czas produkcji;
  Produkt = generuj_produkt;
  P(PUSTE_XYZ)      // jeśli ilość pustych miejsc jest różna od 0 to wchodzi, w przeciwnym
wypadku czeka
  P(MUTEX_XYZ)
  BXYZ->elementy[BXYZ->pierwszy_wolny] = produkt;
  BXYZ->pierwszy_wolny = (BXYZ->pierwszy_wolny + 1) % ROZMIAR_BUFORA;
  V(MUTEX_XYZ)
  V(ZAJETE_X)
}
```

plik wykonywalny dotyczący robota tworzącego ciągi znaków z elementów X, Y, Z:

```
char stworzony[4];
while(licznik_wyprodukowanych_mniejszy_od_10)
{
  P(ZAJETE_X)
  stworzony[0] = BX->elementy[BX->pierwszy_zajety];
  BX->pierwszy_zajety = (BX->pierwszy_zajety + 1) % ROZMIAR_BUFORA;
  V(PUSTE_X)
  P(ZAJETE_Y)
  stworzony[1] = BY->elementy[BY->pierwszy_zajety];
  BY->pierwszy_zajety = (BY->pierwszy_zajety + 1) % ROZMIAR_BUFORA;
  V(PUSTE_Y)
  P(ZAJETE_Z)
  stworzony[2] = BZ->elementy[BZ->pierwszy_zajety];
```

```
  BZ->pierwszy_zajety = (BZ->pierwszy_zajety + 1) % ROZMIAR_BUFORA;
  V(PUSTE_Z)
    stworzony[3] = '\0';
  P(PUSTE_P)
  /// odkładam stworzony produkt do BP
  V(ZAJETE_P)
}
```

Zdejmowanie z bufora BP analogiczne, czyli P(ZAJETY_P), następuje zdejmowanie, V(PUSTE_P).

6. WYWOŁANIA SYSTEMOWE DO REALIZACJI ZADANIA:

Wykorzystane zostaną wywołania systemowe dla semaforów oraz pamięci współdzielonej:

- Dla semaforów: semget, semctl, semop;
- Dla pamięci współdzielonej: shmget, shmat, shmdt.

7. SPOSÓB PRZETESTOWANIA POPRAWNOŚCI ZAIMPLEMENTOWANEGO ZADANIA:

Po wyprodukowaniu każdego napisu będzie on wyświetlany na ekran. Ponadto przewiduję możliwość wyświetlania „komentarzy" odnośnie postępu produkcji. Po każdorazowym wyprodukowaniu elementu X, Y, lub Z pojawiałby się komunikat o wyprodukowaniu przez robota o konkretnym ID konkretnego elementu. Ponadto przewiduję możliwość wyświetlania stanu kolejek, buforów, czyli informowania o ilości miejsc wolnych i zajętych dla danego bufora (albo cyklicznie co jakiś czas, albo przy zmianie stanu). W przypadku dużej ilości wyświetlanych informacji, zostałyby one podzielone na kilka plików. Po zakończonej symulacji użytkownik miałby możliwość otworzyć interesujący go plik (informujący o zajętości kolejek, o kolejności kończenia produkcji przez konkretne roboty, o ilości wykonanych przez nich produktów), oraz powoli prześledzić dokładny obraz symulacji.