



# BÀI 5 : CÁC GIẢI PHÁP ĐỒNG BỘ HOÁ

---

- Nhóm giải pháp **Busy Waiting**
  - Sử dụng các biến cờ hiệu
  - Sử dụng việc kiểm tra luân phiên
  - Giải pháp của Peterson
  - Cấm ngắt
  - Chỉ thị TSL
- Nhóm giải pháp **Sleep & Wakeup**
  - Semaphore
  - Monitor
  - Message



## Các giải pháp “Busy waiting”

---

**While (chưa có quyền) donothing() ;**

CS;

**Từ bỏ quyền sử dụng CS**

- Tiếp tục tiêu thụ CPU trong khi chờ đợi vào miền gá
- Không đòi hỏi sự trợ giúp của Hệ điều hành



## Các giải pháp “Sleep & Wake up”

if (chưa có quyền) Sleep() ;

CS;

Wakeup( somebody);

- **Từ bỏ CPU khi chưa được vào miền găng**
- **Cần được Hệ điều hành hỗ trợ**



# Semaphore

```
Semaphore s; // s >= 0  
Down (s) & Up(s)
```


- Được hỗ trợ bởi HĐH

- Tổ chức độc quyền truy xuất

```
Down (s)  
CS;  
Up(s)
```

```
P1 :  
Job1;  
Up(s)
```

```
P2:  
Down (s);  
Job2;
```



- Tổ chức “hò hện”



# Monitor

Monitor m

int x;

Condition c;

Function F1()

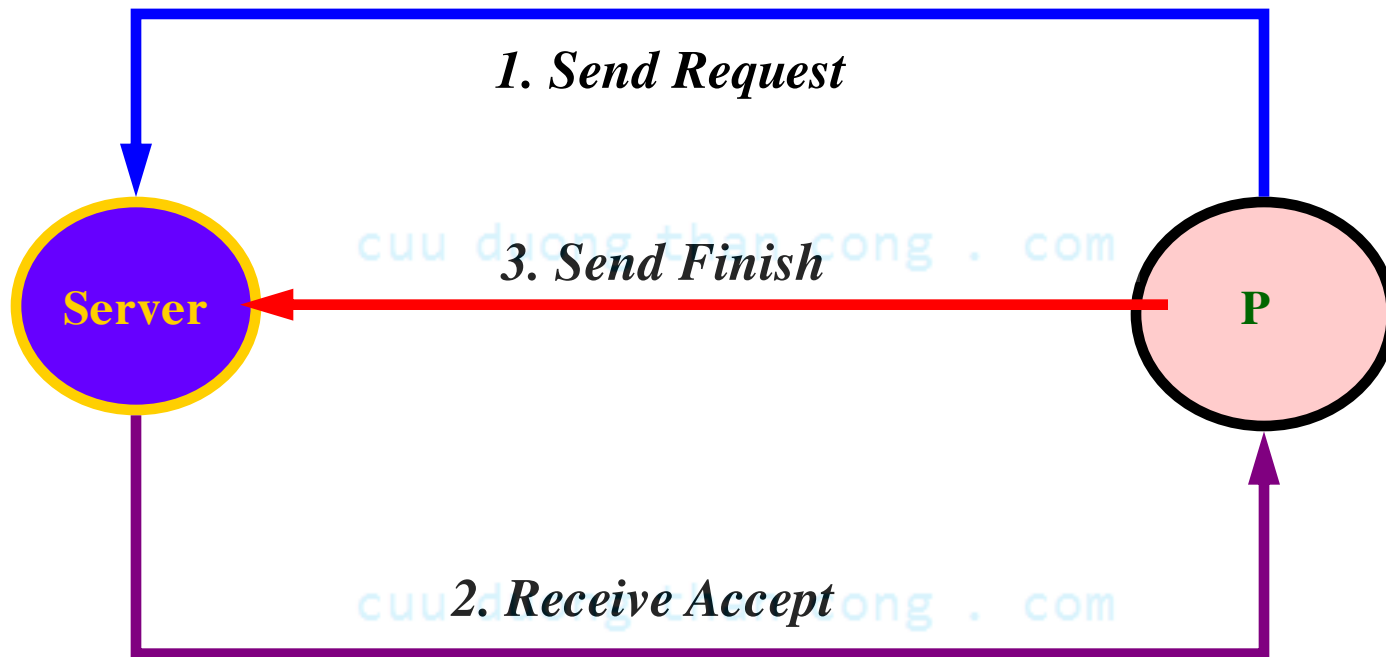
{ ....wait(c); ...}

Function F2()

{ ....signal(c); ...}

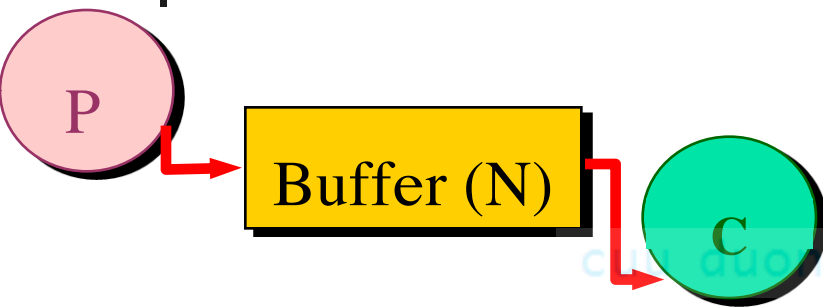
- Được hỗ trợ bởi NNLT
- Bảo đảm độc quyền truy xuất tự động
- Sử dụng biến điều kiện để thực hiện “Hò hẹn”

# Message



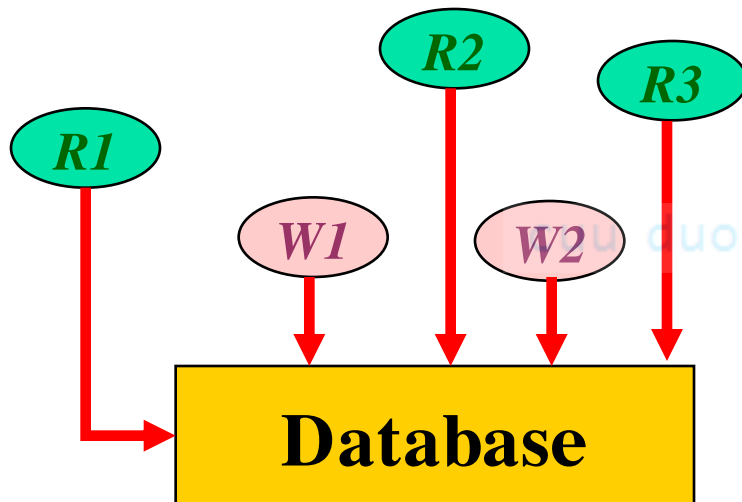
- Được hỗ trợ bởi HĐH
- Đồng bộ hóa trên môi trường phân tán

# Các bài toán đồng bộ hoá kinh điển



## Producer-Consumer

- P không được ghi dữ liệu vào buffer đã đầy
- C không được đọc dữ liệu từ buffer đang trống
- P và C không được thao tác trên buffer cùng lúc



## Readers - Writers

- W không được cập nhật dữ liệu khi có một R đang truy xuất CSDL.
- Tại một thời điểm, chỉ cho phép một W được sửa đổi nội dung CSDL.