**ĐẠI HỌC QUỐC GIA TP HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

🙞∙∙∙☼∙∙∙🙜



**THIẾT KẾ VÀ PHÁT TRIỂN ỨNG DỤNG IOT**

**BÀI TẬP VỀ NHÀ: MULTITHREAD VÀ PROTOTHREAD**

**LỚP L01 --- HK232**

NGÀY NỘP: 09/02/2024

Giảng viên hướng dẫn: Thầy Võ Quế Sơn

|  |  |
| --- | --- |
| **Sinh viên thực hiện** | **Mã số sinh viên** |
| Châu Ngọc Tình | 2110588 |

Thành phố Hồ Chí Minh – 2024

# Nội dung

## Protothread

Protothread là một cách để cấu trúc code cho phép hệ thống chạy các hoạt động khác khi code đang chờ điều gì đó xảy ra.

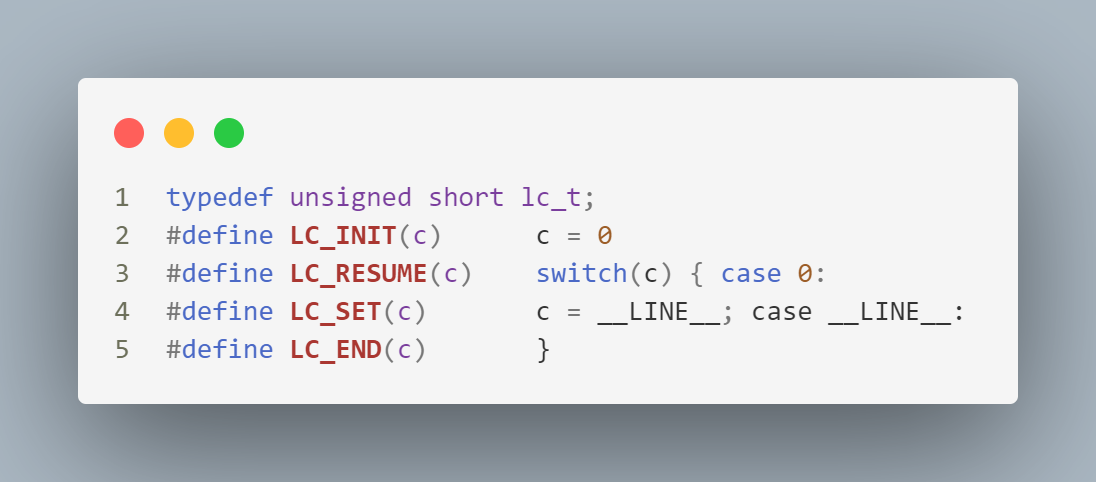
Protothread là một hàm C thông thường, được bắt đầu và kết thúc bằng hai macro `PT\_BEGIN()` và `PT\_END()`. Giữa hai macro có thể sử dụng các protothread khác đã được khai báo.

Preprocessor C của các hoạt động chính của protothread:

A screenshot of a computer program

Description automatically generated

Khai báo local continuations với cấu trúc switch trong C:



struct pt { lc\_t lc}: bao gồm một biến `lc` kiểu `unsigned short`, viết tắt của `local continuation`. Vậy mỗi protothread chiếm 2 byte dung lượng.

Protothread sử dụng macro thay vì dùng hàm là vì phù hợp với mục đích làm thay đổi luồng điều khiển. Nếu dùng hàm thì khi biên dịch cần thêm các đoạn hợp ngữ khi gọi hàm, vì vậy dùng macro để thay đổi luồng điều khiển bằng các cấu trúc C tiêu chuẩn sẽ tối ưu hơn.

Ví dụ: (giả sử `\_\_LINE\_\_` đang ở dòng thứ 12 của code):

|  |  |
| --- | --- |
| A screenshot of a computer program  Description automatically generated |  |

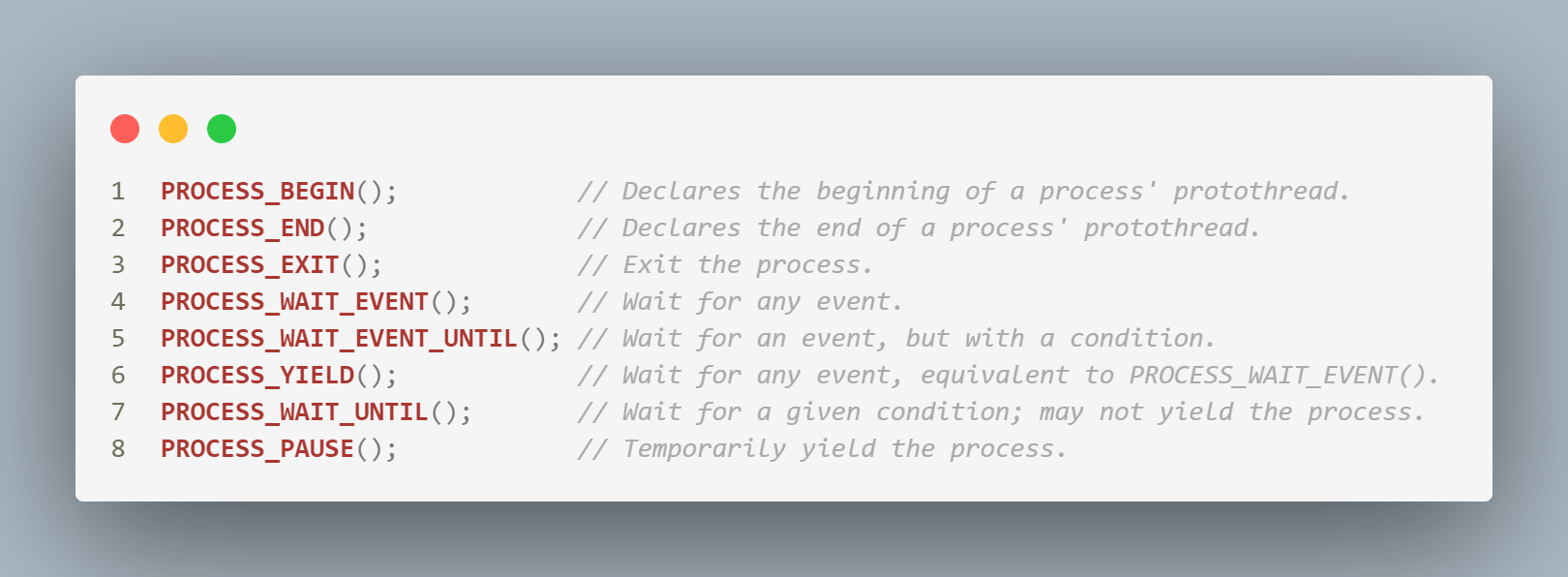
## Protothread in Process

Process trong Contiki OS được viết dựa trên protothread của Dunkel (http://dunkels.com/adam/pt/). Vì vậy Contiki có phiên bản protothread riêng của nó, cho phép chờ đợi các sự kiện đến.

Process trong Contiki OS có những đặc điểm sau:

- Các process hoạt động theo dạng `cooperative multithreading`, các source code cho biết chính xác điểm nào sẽ được thực thi. Không phải ở hình thức `preemptive multithreading` nên các process sẽ không chuyển đổi tự động giữa các thread, mà là thực hiện xoay vòng tuần tự.

- Các process cùng sử dụng chung stack, vì vậy không có biến thread-local. Thay vào đó dùng các static variable để bảo toàn các giá trị qua nhiều lệnh gọi process.



Ví dụ:



# Bài làm

## Multithread

### Đoạn code sử dụng Multithread

A screenshot of a computer program

Description automatically generated



### Mô phỏng bằng cooja

A screenshot of a computer

Description automatically generated

## Protothread

### Đoạn code sử dụng Protothread của Contiki OS

A screenshot of a computer program

Description automatically generated

*A screenshot of a computer program

Description automatically generated*

### Mô phỏng trên cooja

A screenshot of a computer

Description automatically generated