Bài 5: Storage Classes

Nguyễn Hoàng Anh

Extern

Đặt vấn đề:

Giả sử ở file test.c có 1 biến a bất kỳ, nếu như file khác muốn sử dụng biến này, ngoài việc sử dụng #include thì liệu có cách nào khác không ?

Tại sao không sử dụng #include ?

Extern

Khái niệm Extern trong ngôn ngữ lập trình C được sử dụng để thông báo rằng một biến hoặc hàm đã được khai báo ở một nơi khác trong chương trình hoặc trong một file nguồn khác. Điều này giúp chương trình hiểu rằng biến hoặc hàm đã được định nghĩa và sẽ được sử dụng từ một vị trí khác, giúp quản lý sự liên kết giữa các phần khác nhau của chương trình hoặc giữa các file nguồn.

Extern

File test.c

```c
#include <string.h>

char var[10];


void display(void)
{
     strcpy(var, (char*)"Hello");

}
```

File test1.c

```c
#include <stdint.h>

/*
Description: delay x giay
*/
void delay(void)
{
     uint32_t i;
     for (i = 0; i < 5000000; ++i)
     {

     }
}
```

Static local variables

Khi **static** được sử dụng với **local variables** (biến cục bộ - khai báo biến trong một hàm), nó giữ giá trị của biến qua các lần gọi hàm và giữ phạm vi của biến chỉ trong hàm đó.

Static local variables

```c
#include <stdio.h>

void exampleFunction() {
    static int count = 0;  // Biến static giữ giá trị qua các lần gọi hàm
    count++;
    printf("Count: %d\n", count);
}

int main() {
    exampleFunction();  // In ra "Count: 1"
    exampleFunction();  // In ra "Count: 2"
    exampleFunction();  // In ra "Count: 3"
    return 0;
}
```

Static global variables

Khi static được sử dụng với global variables (biến toàn cục - khai báo biến bên ngoài hàm), nó hạn chế phạm vi của biến đó chỉ trong file nguồn hiện tại.

Ứng dụng: dùng để thiết kế các file thư viện.

Static global variables

File calculation.h

```c
#include <math.h>


typedef struct {
    float x1;
    float x2;
} Equation;

static int A,B,C;

void inputCoefficients(int a, int b, int c) {
      A = a;
      B = b;
      C = c;
}
```

Static global variables

File motor.c

```c
#include <stdio.h>
#include "motor.h"


// General function
void startMotor(PIN pin) {
   printf("Start motor at PIN %d\n",
pin);
}

void stopMotor(PIN pin) {
   printf("Stop motor at PIN %d\n",
pin);
}

void changeSpeedMotor(PIN pin, int
```

File motor.h

```c
#ifndef __MOTOR_H
#define __MOTOR_H

typedef struct {
       void (*start)(int gpio);
   void (*stop)(int gpio);
   void (*changeSpeed)(int gpio, int
speed);
} MotorController;

typedef int PIN;

static void startMotor(PIN pin);
static void stopMotor(PIN pin);
static void changeSpeedMotor(PIN
pin, int speed);
```

Static trong class

Khi một thành viên của lớp được khai báo là static, nó thuộc về lớp chứ không thuộc về các đối tượng cụ thể của lớp đó. Các đối tượng của lớp sẽ chia sẻ cùng một bản sao của thành viên static, và nó có thể được truy cập mà không cần tạo đối tượng. Nó thường được sử dụng để lưu trữ dữ liệu chung của tất cả đối tượng.

Static trong class

```
#include <iostream>

typedef enum
{
    red = 0,
    blue,
    green,
    purple,
    black,
    yellow
} Pen_Color;

void print_color_pen(Pen_Color color)
{
    switch (color)
    {
    case red:
        std::cout << "Red\n";
        break;
```

Volatile

Từ khóa volatile trong ngôn ngữ lập trình C/C++ được sử dụng để báo hiệu cho trình biên dịch rằng một biến có thể thay đổi ngẫu nhiên, ngoài sự kiểm soát của chương trình. Việc này ngăn chặn trình biên dịch tối ưu hóa hoặc xóa bỏ các thao tác trên biến đó, giữ cho các thao tác trên biến được thực hiện như đã được định nghĩa.

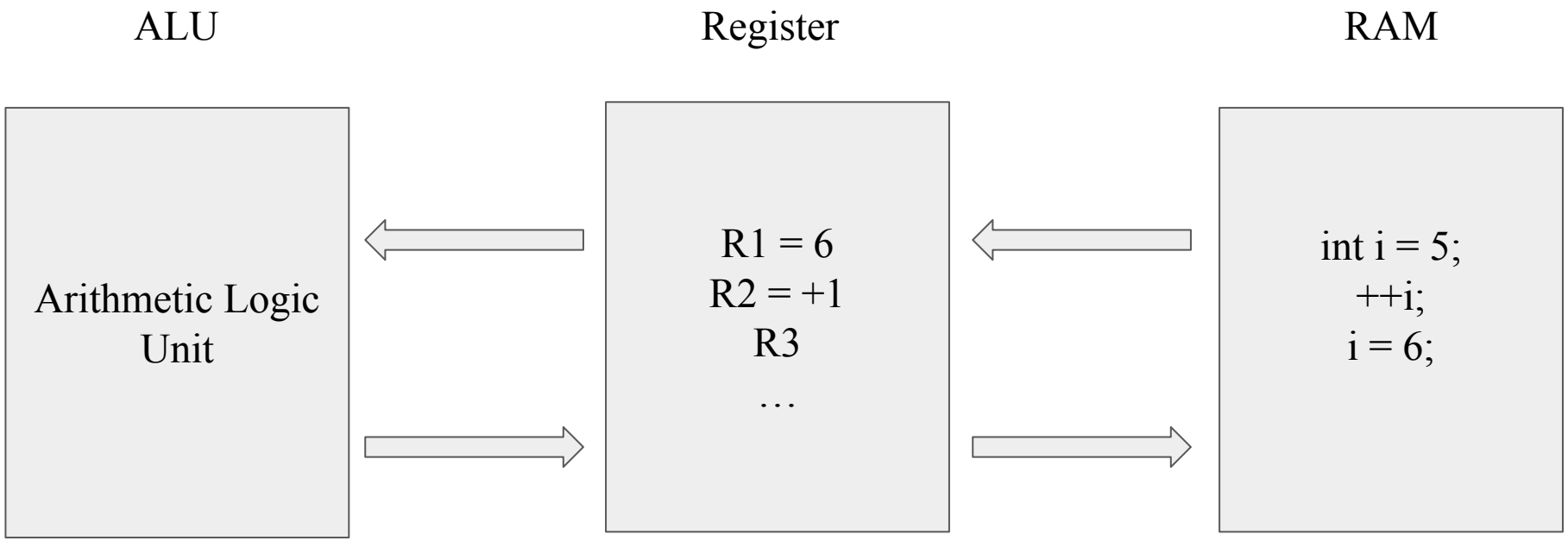
Volatile

```
#include "stm32f10x.h"

volatile int i = 0;
int a = 100;

int main()
{

	while(1)
	{
		i = *((int*) 0x20000000);
		if (i > 0)
		{
			break;
		}

	}
	a = 200;
}
```

Register

Trong ngôn ngữ lập trình C, từ khóa register được sử dụng để chỉ ra ý muốn của lập trình viên rằng một biến được sử dụng thường xuyên và có thể được lưu trữ trong một thanh ghi máy tính, chứ không phải trong bộ nhớ RAM. Việc này nhằm tăng tốc độ truy cập. Tuy nhiên, lưu ý rằng việc sử dụng register chỉ là một đề xuất cho trình biên dịch và không đảm bảo rằng biến sẽ được lưu trữ trong thanh ghi. Trong thực tế, trình biên dịch có thể quyết định không tuân thủ lời đề xuất này.

Register



Register

```
#include <stdio.h>
#include <time.h>

int a;
int b;

int main() {
    // Lưu thời điểm bắt đầu
    clock_t start_time = clock();
    reigster int i;

    // Đoạn mã của chương trình
    for (i = 0; i < 2000000; ++i) {
        // Thực hiện một số công việc bất kỳ
    }

    // Lưu thời điểm kết thúc
    clock_t end_time = clock();

    // Tính thời gian chạy bằng miligiây
    double time_taken = ((double)(end_time - start_time)) / CLOCKS_PER_SEC;
```