BÁO CÁO TUẦN 2

1/ Sự giống và khác nhau giữa C thông thường với Embedded C

* Giống nhau
* không có nhiều khác biệt giữa C và Embedded C ngoài một vài phần mở rộng và môi trường hoạt động
* có cú pháp, kiểu dữ liệu, hàm gần như giống nhau
* khác nhau
* Embedded C là một phần mở rộng cho Standard C Programming với các tính năng bổ sung như Addressing I/O, multiple memory addressing và fixed-point arithmetic.. v.v
* C Programming thường được sử dụng để phát triển các ứng dụng máy tính
* Embedded C được sử dụng để phát triển các ứng dụng dựa trên Vi điều khiển.

2/ lưu ý khi lập trình nhúng

* ROM và RAM hạn chế.
* Lập trình phụ thuộc phần cứng.
* Cần đáp ứng chính xác về thời gian (hàm xử lý ngắt, tác vụ…)
* Nhiều kiểu pointer (far/rom/ui/paged/…)
* Một số keywords và token đặc biệt (@, interrupt, tiny,..)

3/ C memory management

Memory management là quá trình các chương trình máy tính được gán cho không gian bộ nhớ vật lý hoặc ảo.

Việc cấp phát bộ nhớ có thể được thực hiện trước hoặc tại thời điểm thực hiện chương trình. Có hai kỹ thuật cấp phát bộ nhớ: cấp phát bộ nhớ tĩnh và cấp phát bộ nhớ động.

+ bộ nhớ tĩnh :sử dụng các hàm thông thường

+ bộ nhớ động :  sử dụng các hàm malloc(), calloc(), realloc()và free()

[calloc()](https://www.codecademy.com/resources/docs/c/memory-management/calloc)

Phân bổ động một mảng các khối bộ nhớ thuộc loại được chỉ định.

[free()](https://www.codecademy.com/resources/docs/c/memory-management/free)

Tự động phân bổ lại bộ nhớ khi chạy.

[malloc()](https://www.codecademy.com/resources/docs/c/memory-management/malloc)

Cấp phát một khối bộ nhớ trong heap nhưng không khởi tạo.

[realloc()](https://www.codecademy.com/resources/docs/c/memory-management/realloc)

Phân bổ lại khối bộ nhớ đã được phân bổ trước đó.

4/ Compilation process, toolchain, compiler, linker.

* Compliation process

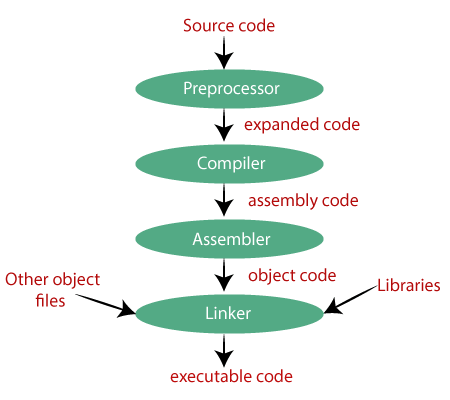
Compliation process chuyển đổi mã nguồn được lấy làm đầu vào thành mã đối tượng hoặc mã máy. Quá trình biên dịch có thể được chia thành bốn bước:

+ Pre-processing

+ Compiling

+Assembling

+ Linking.



* Compiler và assembler

Code được preprocessor mở rộng sẽ được chuyển tới compiler. compiler biên dịch chuyển mã này thành mã hợp ngữ. Hoặc chúng ta có thể nói rằng compiler chuyển đổi code được xử lý trước thành mã hợp ngữ.

Mã hợp ngữ chuyển thành mã đối tượng bằng assembler

* Linker( trình liên kết )

liên kết mã đối tượng của chương trình của chúng ta với mã đối tượng của các tệp thư viện và các tệp khác. Đầu ra của trình liên kết là tệp thực thi. Tên của tệp thực thi giống với tên tệp nguồn nhưng chỉ khác nhau ở phần mở rộng của chúng

* Toolchain

Toolchain là tập hợp của các tool dùng để biên dịch source code thành các file executable có thể chạy được trên target device (thiết bị thật). Toolchain bao gồm compiler, linker, các thư viện runtime và một số tool linh tinh khác

+***Native toolchain***: Loại toolchain chạy trên hệ thống giống với target device (thiết bị dùng để chạy chương trình build ra bởi toolchain). Thường là toolchain cho desktop dùng để build ra các chương trình chạy trên destop luôn.

***+Cross toolchain***: Loại toolchain này chạy trên hệ thống khác với target device, ví dụ dùng desktop để build ra chương trình chạy trên một thiết bị nhúng khác. Toolchain này giúp cho quá trình phát triển phần mềm trở nên nhanh hơn vì chúng ta có thể develop phần mềm trên desktop PC và load chúng lên chạy trên target device.