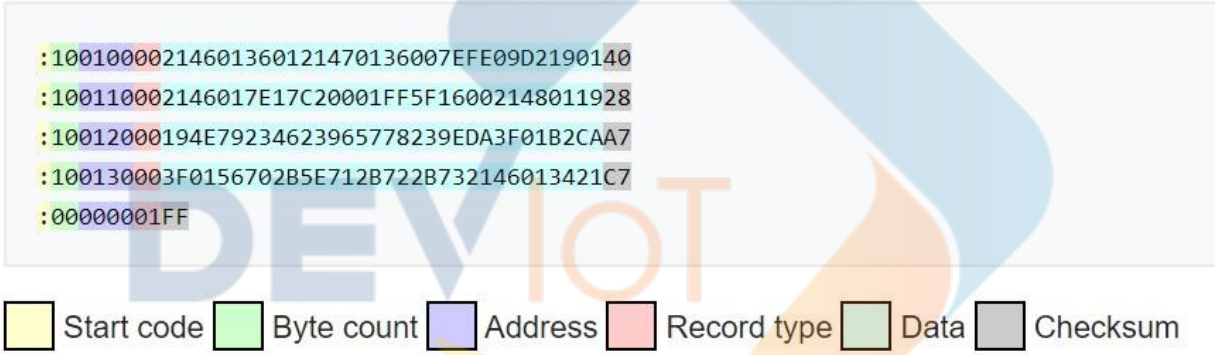
Bài 25 : Binary File và Hex File

1 . File *.bin và *.hex là gì ?

File *.hex, *.bin là 2 định dạng đầu ra được sử dụng rất nhiều khi biên dịch chương trình dùng cho vi điều khiển. Mỗi định dạng có ưu điểm khác nhau và được sử dụng cho các mục đích tương ứng.

a. Hex File

Nó là một tập tin chứa nội dung chương trình cùng các thông tin chỉ dẫn . Mỗi dòng tương ứng với một record . Mỗi record có cấu trúc 6 trường HEX khác nhau theo định dạng Intel HEX. Mỗi trường biểu diễn các giá trị theo mã hexadecimal. Trong quá trình update Firmware, từ File Intel HEX chúng ta phải tách nội dung chương trình ra và lưu vào bộ nhớ flash của vi điều khiển theo các chỉ dẫn địa chỉ đi kèm trong file.



- **Start code**
  Số lượng kí tự: 1
  Dấu hai chấm (:) trong bảng ASCII.
- **Byte count**
  Số lượng kí tự: 2
  Đại diện cho số lượng các byte sẽ xuất hiện trong Data field. Giá trị byte count có thể nằm trong khoảng từ 0 – 255 (0x00 – 0xFF). Thông thường byte count có giá trị 16 (0x10).
- **Address**
  Số lượng kí tự: 4
  Với Data Record thì trường này cung cấp 16 bit địa chỉ bộ nhớ chương trình có trọng số thấp (0-15). Được tính bằng địa chỉ gốc (base address + địa chỉ offset)

- **Record type**

Record type	Ý nghĩa
00	Data nạp vào flash
01	Kết thúc file
02	Extended Segment Address(CPU 80x86)
04	Base address
05	Start Address(CPU 80x86)

Số lượng kí tự: 2 từ 00 đến 05

Dùng để xác định chức năng của record đó. Sẽ có 6 loại record tương ứng với 6 giá trị:

º *Record type = 00 Data Record*

Record này sẽ chứa dữ liệu, chính là chương trình của chúng ta. Bên cạnh đó nó cũng sẽ cung cấp giá trị 16 bit địa chỉ bộ nhớ có trọng số thấp (bit 0-15) nơi lưu dữ liệu trong bộ nhớ chương trình của vi điều khiển.

º *Record type = 04 Extended Linear Address Record*

Record này chứa thông tin bit16-31 địa chỉ của bộ nhớ lưu chương trình.

º *Record type* = 01 *End of File Record*

Record này xuất hiện cuối cùng file chương trình nhằm thông báo kết thúc file.

º *Record type = 02 Extended Segment Address Record*

Record này để thay đổi bit 4-19 trong địa chỉ bộ nhớ lưu chương trình so với record đã cung cấp trước đó. Với file được build từ KeilC thì Extended Linear Address Record thường xuyên được sử dụng hơn.

º *Record type = 03 Start Segment Address Record*

Record này không xuất hiện trong trong file chương trình của STM32 nên có thể được bỏ qua.

º *Record type = 05 Start Linear Address Record*

Record này có thể được bỏ qua vì nó không chứa thông tin nào cần thiết để lưu vào bộ nhớ chương trình.

- **Data**

Số lượng kí tự: 2 x Byte count

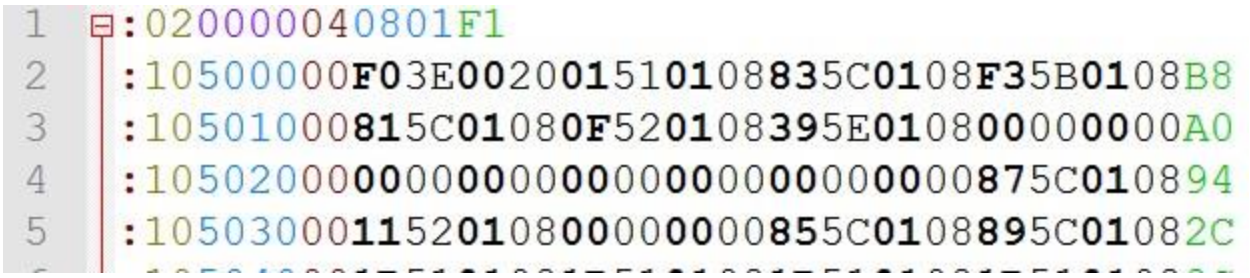
Thường sử dụng để biểu thi thông tin hoặc dữ liệu tùy thuộc vào từng loại record. Có một số record sẽ không có trường này.

Checksum

Số lượng kí tự: 2

Một giá trị được tính toán được sử dụng để xác nhận record không có lỗi.

Ví dụ :



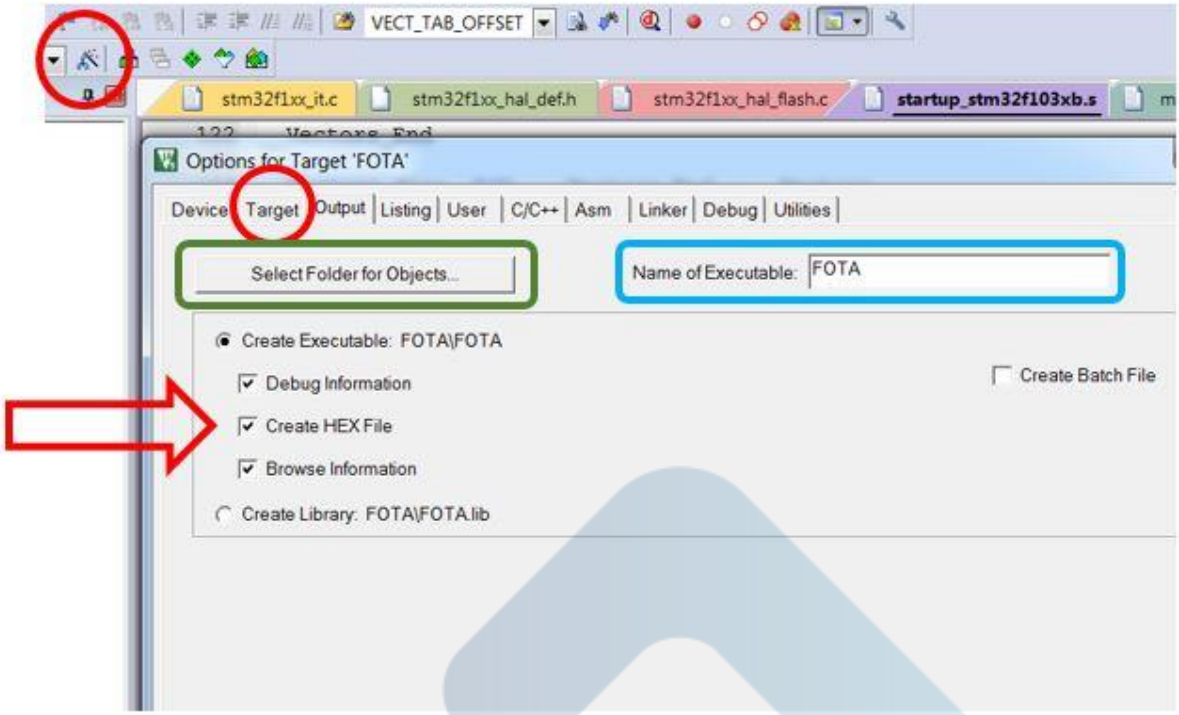
Start code	Byte count	Address	Record type	Data	Checksum
:	02	0x08000000	04	0801	F1

0x02 là số lượng data byte có trong record
0x0000 là address field. Trong extended linear address record thì giá trị trường này luôn là 0x0000 với STM32 thì Address là 0x08000000
0x04 là record type, chỉ ra rằng đây là một extended linear address records
0x0801 là giá trị 16 bit cao (từ 16-31) trong địa chỉ bộ nhớ chương trình

- Tạo file HEX bằng Kiel C

Để tạo file chương trình (đuôi .hex) bằng KeilC IDE thì làm các bước như sau:

Bước 1: Vào **Option** -> Tab **Output** -> Kích chọn **Create Executable** -> **Create HEX File**
Bước 2: Thư mục mặc định lưu file chương trình là MDK-ARM/YourProjectName. Thay đổi thư mục lưu tại **Select Folder for Objects**
Bước 3: Thay đổi tên file chương trình tại **Name of Executable**
Bước 4: Chọn OK rồi **Build**



b. Binary File

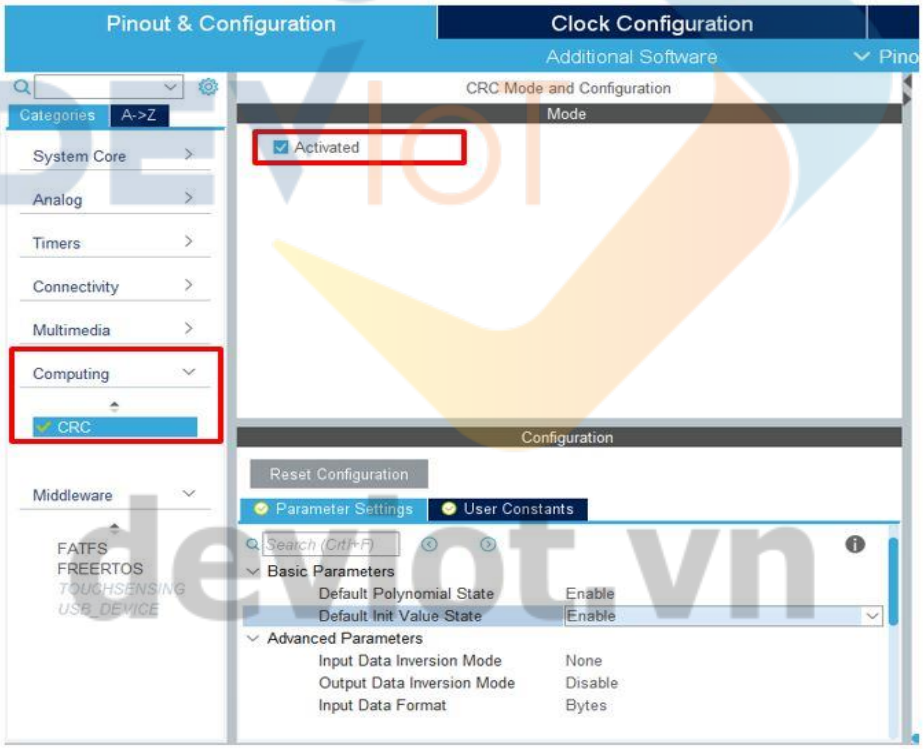
Binary file có đuôi *.bin. Đây là một dạng tệp tin phi văn bản, được lưu trữ dưới định dạng mã nhị phân. Máy tính chúng ta hoạt động dựa theo mã nhị phân nên chúng ta thường dùng loại tệp tin này để lưu tất cả thông tin dữ liệu của bộ nhớ chương trình trên vi điều khiển.

Cấu trúc của một file định dạng BIN là:

- Các giá trị trong file là mã nhị phân 1 và 0.
- Dữ liệu được lưu một cách liên tục, không chia ra thành từ dòng.
- Không chứa địa chỉ bắt đầu nạp.
- Chỉ chứa dữ liệu, thông tin của chương trình chứ không chứa các trường không liên quan như Start code, Byte count, Address, Record type, Checksum
- Không chứa các kí tự xuống dòng (\n)

c. Kết luận

- **Về kích thước:**
  Hiện tại mình có một Project, từ đó tạo ra 2 file có kích thước là (11.6 kByte) đối với định dạng Intel HEX và (4 kByte) với định dạng BIN.
  => kích thước của file định dạng BIN nhỏ hơn nhiều so với file định dạng Intel HEX. Kích thước của file ảnh hưởng rất lớn đến quá trình update Firmware, file nhẹ thì thời gian tải xuống vi điều khiển nhanh hơn. Cùng với đó, khi truyền tải dữ liệu ít hơn thì nguy cơ xảy ra lỗi cũng ít hơn.
- **Về quá trình xử lý file**: cấu trúc file định dạng BIN đơn giản hơn, không cần phải xử lý các thông tin của các trường khác như file định dạng Intel HEX.
  => Tối giản được quá trình xử lý file, từ đó có thể tiết kiệm được thời gian xử lý trong quá trình cập nhật chương trình.
- **Kiểm tra dữ liệu:** File định dạng BIN không có các byte kiểm tra như trong file định dạng Intel HEX nên khả năng kiểm tra lỗi trong quá trình truyền nhận dữ liệu là không có.
  Tuy nhiên, ta có thể khắc phục được nhược điểm này bằng cách tự thêm các byte kiểm tra vào. Một phương pháp kiểm tra khá hiệu quả là CRC-32, tuy phức tạp nhưng khả năng phát hiện lỗi cao. Đồng thời, STM32 có hỗ trợ hàm tính toán CRC nên có thể áp dụng dễ dàng.



Có thể kết luận rằng thực hiện quá trình với Binary File sẽ tối ưu hơn cho chương trình Bootloader.

DEVIOT - CÙNG NHAU HỌC LẬP TRÌNH IOT

📌 Website: deviot.vn

📌 FanPage: Deviot - Thời sự kỹ thuật & IoT

📌 Group: Deviot - Cùng nhau học lập trình IOT

📌 Hotline: 0969.666.522

📌 Address: Số 101C, Xã Đàn 2

📌 Đào tạo thật, học thật, làm thật