Hướng dẫn tìm hiểu code xe MCR

Lời nói đầu:

Đa số học sinh, sinh viên khi mới bắt đầu tham gia MCR sẽ thường mua tất cả cơ khí, điện tử, code. Tuy nhiên, việc sở hữu xe cũng như việc các bạn đặt xe trên đường đua và về đích không phải là việc chơi xe. Mình dùng từ chơi ở đây với nghĩa ẩn dụ (chơi xe MCR đồng nghĩa với việc học rất nhiều thứ từ cuộc đua đến kết cấu của xe).

Để giúp các bạn mới biết về xe MCR cũng như các bạn đã chơi lâu nhưng vẫn còn khá mập mờ về code cũng như các cách thức hoạt động, mình xin ra mắt một số tài liệu do mình tự soạn thảo trên cơ sở kinh nghiệm và các kiến thức đúc kết trong quá trình tham gia MCR từ năm 2013 đến nay.

Chương 1: chương trình kiểm tra xe - test.

Chương đầu tiên mình sẽ đề cập đến hệ thống code một các tổng quát.
Mời các bạn xem đoạn code bên dưới rồi mình sẽ đi vào giải thích:

```
int main()
{
	init();
	asm("sei");
	speed(0,0);
	testing_flag = 1;
	beep_long(300);
	_delay_ms(500);
	LCD_Puts("eROBOT");
	LCD_Gotoxy(3,1);
	LCD_Puts("HI THERE!");
	_delay_ms(1000);
	test();
	run();
	return 1;
}
```

Vị trí đoạn code nằm trong file MCR2016.c

Đây là đoạn code chính trong chương trình xe vì đây là hàm main. Hàm main là gì, mình nói luôn cho các bạn mới tiếp cận với vi điều khiển cũng như mới học lập trình, đây là hàm chính trong một chương trình. Trong chương trình có rất nhiều hàm con nhưng khi vi điều khiển được cấp nguồn nó sẽ nhảy thẳng vào chạy cái hàm này. Sau đó từ hàm main nó sẽ gọi các hàm con khác.

Dòng đầu tiên: `int main();` dòng này là dòng khai báo một hàm, chữ int là định nghĩa kiểu trả về của hàm (nghĩa là khi chạy xong hàm này nó sẽ trả về một số thuộc kiểu mình đã khai báo, ở đây là kiểu int). Chữ main() là tên của hàm, mình nên đặt tên để phân biệt các hàm, phân biệt chức năng.

Dòng code tiếp theo là: `init();` đây là dòng lệnh gọi một hàm con có tên là init(), hàm con này có chức năng khai báo ban đầu cũng như cấu hình cho vi điều khiển hoạt động đúng chức năng mong muốn. Dòng này luôn phải đặt ở đầu hàm main vì nếu đặt sau thì các hàm con gọi trước có thể hoạt động sai mục đích.

`Dòng asm("sei");` cho phép ngắt toàn cục trong vi điều khiển (liên quan đến phần cứng).

Dòng tiếp theo là: `speed(0,0);` đây cũng là một hàm con, mình sẽ giải thích sau.

Dòng tiếp theo: `testing_flag = 1;` gán cho biến testing_flag giá trị là 1, biến này có chức năng xác định chương trình đang chạy ở đoạn nào, nếu biến này bằng 1 nó sẽ nhảy vào kiểm tra bằng tay các chức năng trên xe.

`Dòng tiếp: beep_long(300);` đây là một hàm con được gọi, dùng để phát một tiếng “beep” ở loa của xe, số 300 có nghĩa thời gian kéo dài tiếng “beep” là 300ms (300 mili giây).

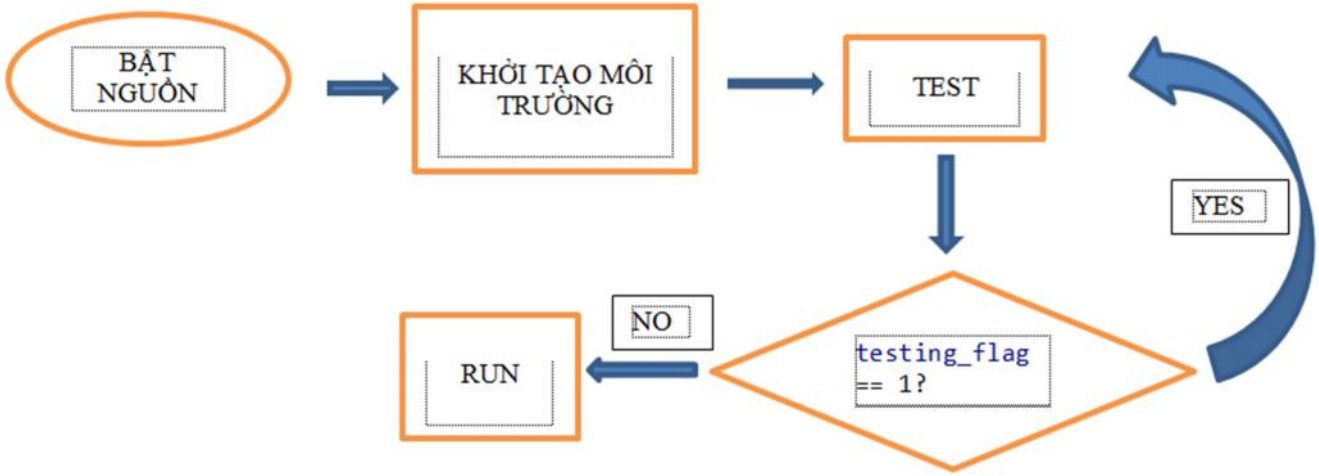
Tiếp theo là:
```
LCD_Puts("eROBOT");
LCD_Gotoxy(3,1);
LCD_Puts("HI THERE!");
```
đoạn này không quan trọng, nó chỉ hiển thị lên màn hình một dòng chữ chào hỏi cho vui mà mình chèn vào để chào các bạn ^^.

Tiếp là: `_delay_ms(1000);` đây lại gọi thêm hàm con, hàm con này có chức năng là ngừng chương trình trong 1s để các thiết bị trên xe ổn định và cũng để cho các bạn thấy được dòng chữ chào hỏi trên LCD ở 3 dòng trên mà mình chèn vào :D đặt biệt, hàm này thuộc vào hàm quan trọng vì nó có mặt rất nhiều trong các đoạn code. Các bạn chỉ cần nhớ tên của nó `_delay_ms(xxx);` xxx là thời gian ngừng chương trình tính theo đơn vị mili giây nhé.

Dòng tiếp theo là quan trọng `test`(); hàm này là một hàm con quan trọng luôn luôn được thực hiện mỗi khi đem ra chơi, đua, khoe, chạy thử,.... Nó được khai báo và hiện thực ở file test.c, nó bao gồm một tập hợp test các chức năng của xe xem cái nào còn dùng được, cái nào lỗi hoặc chưa chuẩn về mặt phần cứng và cơ khí để ta tiến hành bảo trì.

Dòng kế `run();` đây là hàm con quan trọng nhất trong xe, khi xe đang chạy có nghĩa là nó đang thực hiện hàm này, mọi chi tiết liên quan đến việc chạy bao gồm tốc độ, giải thuật chạy đều nằm trong hàm này. Nó được khai báo trong file run.c (các bạn mới chơi xe thì chỉ nên chỉnh sửa trong run.c thôi).

Dòng cuối cùng `return 1;` là giá trị trả về của hàm main(), các bạn không cần quan tâm nó vì nó chẳng bao giờ được thực hiện nhưng nếu bỏ nó thì sẽ bị lỗi cấu trúc :3.



Hình trên là sơ đồ giải thuật tổng quát nhất của xe. Mình giải thích chi tiết như sau

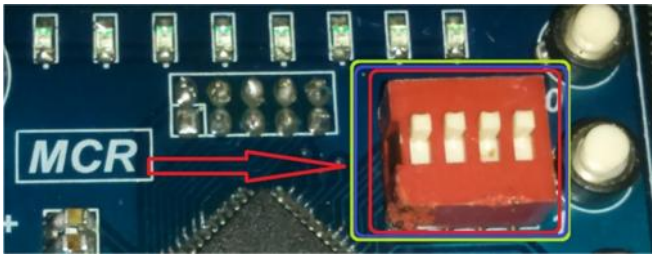
Bước 1: các bạn bật nguồn lên.
Bước 2: xe sẽ tự nhảy vào hàm main và tạo môi trường trên vi điều khiển cho các chức năng trên xe.
Bước 3: Gán cho biến testing_flag =1 và nhảy vào hàm con test() và test các kiểu
Bước 4: kiểm tra xem biến testing_flag có bị hàm con test() thay đổi thành 0 hay không, nếu có thì qua bước 5 không thì quay lại bước 3.
Bước 5: xe nhảy vào chương trình chạy và vù vù về đích nếu bạn có thể :v

Vậy là xong phần tổng quát, tiếp theo mình sẽ giải thích chi tiết các hàm, các biến quan trọng trong chương trình. Mình sẽ đi theo trình tự trong main() cho dễ hiểu: hàm quan trọng đầu tiên là test() nên mình sẽ tìm hiểu test.c có gì và tại sao nó quan trọng. Mời các bạn mở code lên và vào phần test.c nhé.

Đầu tiên khi vào chúng ta thấy một loạt các dòng có cấu trúc sau: `#include "..xxx.h"` đấy là các dòng khai báo cho trình biên dịch biết la ta thêm thư viện vào. Vì trong test.c ta test tất cả mọi thứ nên phải dùng rất nhiều hàm con được khai báo trong các thư viện khác nhau như động cơ, bẻ lái, đọc cảm biến, loa, lcd,...
Chẳng chờ đợi gì thêm, sau nhiều dòng # ở trên ta bắt gặp `void test() đấy là chỗ hiện thực hàm test đấy các bạn.`

Ngay sau đó ta bắt gặp `LCD_Clr();`
`LCD_Gotoxy(0,0);` 2 dòng này có ý nghĩa xóa sạch màn hình LCD (sau khi chào hỏi rồi thì xóa để hiển thị cái khác) và đưa con trỏ LCD về đầu hàng 0 (tọa độ 0,0). (ai chưa biết LCD là gì thì có thể tìm hiểu trên mạng nhé ở đây mình dùng LCD1602, hoặc để từ từ tìm hiểu cũng được vì nó cũng không mấy quan trọng trong code)
Nhảy ngay xuống dưới đó `while(testing_flag == 1)` đây là dòng đầu tiên của một đoạn code lặp có điều kiện là testing_flag == 1, nếu nó bằng 0 thì không lặp nữa, ban đầu ta đã cho nó bằng 1 rồi, phía dưới có chỗ đổi cho nó bằng 0 để mình nhảy vô RUN.

Dòng dưới nó là `switch(get_dipsw())` đây là dòng đầu của cấu trúc `switch-case` cấu trúc này chẳng qua là chia trường hợp để thực hiện thôi, nó thay thế cho if-else khi lồng vào nhau nhiều quá. Bên trong cặp () là `get_dipsw()` (một hàm con trả về giá trị int) hàm này lấy giá trị của cái dipsw màu đỏ



đỏ trên mạch vi điều khiển `xem hình dưới nhé: cái màu đỏ chính là cái dipsw đấy. Từ chữ MCR` nhìn qua ta đánh dấu các nút theo thứ tự là 4321 nhé. Quy ước theo hệ cơ số 16 (thập lục phân hay hexa) ta có 16 giá trị tương ứng 4 cái nút gạt. Khi tất cả gạt lên là 0000 tức là mode 0, nhớ gạt lên là 0 gạt xuống mới là 1 nhé. Ta có dãy như sau:

0000 mode 0
0001 mode 1
0010 mode 2
.
.
1110 mode 14
1111 mode 15

Các bạn học thuộc các số này, mình biết có một cách nhớ như sau (học theo các bạn bên Cao Thắng) "tám bốn hai một" nghĩa là sao: đây là tám 1000, đây là bốn 0100, đây là hai 0010, đây là một 0001. Như vậy cứ thấy nút tương ứng gạt xuống thì cứ cộng vào, ví dụ: 1100 ta thấy ở vị trí 8 và vị trí 4 gạt xuống 1 thì ta lấy 8+4=12 vậy đây là số 12 hay mode 12 ☺ "easy" phải không.

Trong phần này chúng ta chỉ quan tâm đến các case từ 0 đến 6, chúng có chức năng như sau:
Mode 0: học màu, nhấn nút 0 học màu đen, nhấn 1 học màu trắng, nhấn 2 để lưu lại các giá trị đã học vào bộ nhớ.

```
update_black(); học màu đen
update_white(); học màu trắng
update_vcompare(); lưu lại, nếu không lưu lại thì việc học màu ko có ý nghĩa.
```

Mode 1: test động cơ, nhấn nút 0 động cơ trái quay tới, nhấn nút 1 động cơ trái quay lùi, nhấn nút 2 động cơ phải quay tới, nhấn nút 3 động cơ phải quay lùi. `speed`(vận tốc bánh trái , vân tốc bánh phải); vận tốc bánh trái (phải là một giá trị nguyên nhỏ hơn 255 và có thể âm (dương là quay tới, âm quay lùi).

Mode 2: chỉnh giữa đầu servo nhấn nút 2 để qua phải, nhấn nút 1 để qua trái, khi đã thấy đầu servo giữa rồi thì nhấn 0 hoặc 3 để lưu lại góc giữa. Hàm liên quan là `handle(giá trị góc)`; giá trị góc là số nguyên nếu bằng 0 là góc giữa tức xe chạy thẳng, giá trị âm xe quẹo trái, giá trị dương xe quẹo phải, đơn vị tính bằng độ.

Mode 3: test encoder tuy nhiên mình chưa viết code trong mode này, bạn nào có dùng encoder thì phải tự hiện thực code.

Mode 4: hiển thị các giá trị adc đo từ cảm biến ra lcd.

Mode 5: bẻ lái 1 góc 45 độ khi nhấn nút 1 hoặc 2, dùng kiểm tra góc bẻ có đủ để qua cua 90 hay không.

Mode 6: mode này mình viết thêm cho ai thích dùng smartphone để điều khiển từ xa.

Mode 7 đến mode 15 là các mode chạy tương ứng với các tốc độ tăng dần. Khi các bạn để ở mode 7 -15 khi nhấn bất kì nút nào chương trình sẽ nhảy sang chế độ chạy bằng cách ghi giá trị 0 vào biến testing_flag

Chương 2: chương trình chạy – RUN

Đầu tiên mình sẽ liệt kê một số hàm trong run.c và giải thích trước chức năng của nó. Các bạn mở run.h để biết các hàm có sẵn trong run.c nhé.
Những hàm khai báo trong run.h:

```
void run();
unsigned char bit_change( char unsigned in );
unsigned char sensor_inp( unsigned char mask );
char check_crossline( void );
char check_crosshalfline(void);
void handleAndSpeed (int angle,int speed);
void runforwardline (int tocdo);
int gocturn90(int tocdo);
int left_lance (int tocdo);
int right_lance(int tocdo);
```

Đầu tiên: `void run();` đây là hàm chính trong run.c và nó như hàm main vậy, nó mặc định được gọi ở main() trong file MCR2016.c. Chức năng: tất cả các hoạt động chạy của xe nằm trong hàm này.

`unsigned char bit_change( char unsigned in );` các bạn không cần quan tâm đến hàm này lắm, nó dùng để quay các bit trong sensor cho đúng trình tự số hex cho các bạn dễ thao tác. Ví dụ như giá trị sensor là 11000000 thì khi `bit_change( sensor );` thì nó sẽ trả về 00000011. Do phần cứng trả về các giá trị bị ngược so với thứ tự của số hex cũng như trình tự từ trái qua phải nên mới có hàm này. Tuy hàm này hầu như được gọi rất nhiều nhưng chúng ta không cần quan tâm đến nó làm chi.

`unsigned char sensor_inp( unsigned char mask );` đây là hàm che mặt nạ cho sensor, nghĩa là chỉ xét các sensor cần thiết chứ không xét hết các sensor. Ví dụ khi gọi `sensor_inp(0x08);` số 0x08 chuyển sang nhị phân là 0b00001000 như vậy nếu 8 sensor thì qua hàm này nó sẽ trả về 2 giá trị là 0x00 hoặc 0x08 vì ngoài vị trí thứ 4 từ phải đếm qua các vị trí khác ta không xét nên tự động nó sẽ trả về 0. Để hiểu hơn các bạn theo dõi kĩ các hàm có gọi đến `sensor_inp này.`

`char check_crossline( void );` hàm này có nhiệm vụ kiểm tra vạch trắng trên đường (trước cua 90 sẽ có vạch này), trả về 1 nếu phát hiện và 0 nếu không phát hiện.

`char check_crosshalfline(void);` tương tự, hàm này có nhiệm vụ kiểm tra nửa vạch ngang (trước chuyển làn sẽ có vạch này), trả về 1 nếu phát hiện nửa vạch bên phải, 2 nếu phát hiện nửa vạch bên trái và trả về 0 khi không phát hiện gì cả.

`void handleAndSpeed (int angle,int speed);` đọc tên cũng có thể xác định chức năng của hàm này rồi, nó dùng để bẻ lái và kèm theo chạy luôn và đặc biệt là hàm này nó sẽ tự gọi luôn vi sai cho chúng ta. Bên trong hàm này nó sẽ gọi 2 hàm con như sau: `handle ()` có nhiệm vụ điều khiển bẻ lái một góc cố định; `và speed ();` có nhiệm vụ điều khiển tốc độ 2 động cơ. Bên trong hàm này có tính toán để gọi vi sai theo góc bẻ lái, vấn đề vi sai là gì và mức độ quan trọng của nó mình sẽ trình bày ở các chương sau. Tham số: có 2 tham số, đầu tiên là góc và sau là tốc độ.

`void runforwardline (int tocdo);` đây là hàm được chạy xuyên suốt trong chương trình, nhiệm vụ của nó là giúp xe bám line và chạy mượt theo line. Nếu hàm này được lập trình tốt thì xe sẽ chạy rất

mượt, không tốt thì xe sẽ lắc thậm chí chạy ra ngoài. Bên trong hàm này nó sẽ gọi rất nhiều hàm con khác như: `handleAndSpeed(); speed (); handle ();sensor_inp().` Và trên đoạn đường gần 60m thì hàm này giúp xe chạy hơn 50m trên các đường cong và thẳng trừ chuyển làn, cua vuông. Tham số là tốc độ chạy thấp nhất là 0 và cao nhất là 255.

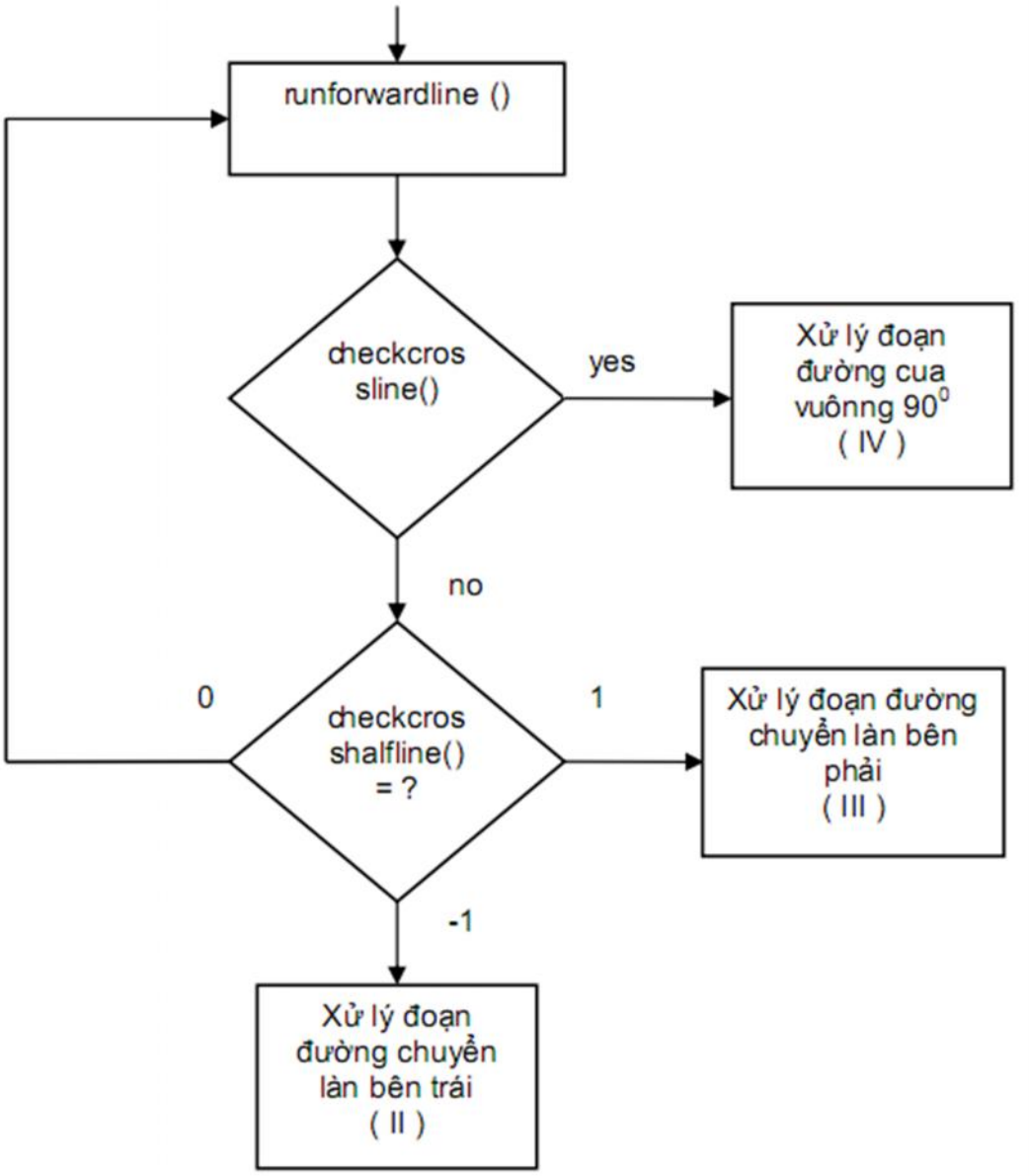
`int gocturn90(int tocdo);` hàm xử lý qua cua vuông, hàm này chưa được hiện thực. Việc hiện thực nó là của các bạn. Tham số là tốc độ.

`int left_lance (int tocdo);` hàm xử lý chuyển làn trái, hàm này chưa được hiện thực. Việc hiện thực nó là của các bạn. Tham số là tốc độ.

`int right_lance(int tocdo);` hàm xử lý chuyển làn phải, hàm này chưa được hiện thực. Việc hiện thực nó là của các bạn. Tham số là tốc độ.

Các hàm xử lý qua cua 90 và chuyển làn chưa được thực hiện nhưng các bạn cũng có thể tham khảo trong các chương trình mẫu có sẵn trên mạng như BCR2010 chẳng hạn.

Tiếp theo chúng ta đi sâu và tìm hiểu rõ hơn về các phối hợp của các hàm thành một chương trình như thế nào nhé.



Trên là sơ đồ giải thuật chính của chương trình, bây giờ chúng ta sẽ đi diễn giải trực tiếp các đoạn code.

Khai báo thư viện:

```
#include <util/delay.h>
#include    "../LED/led.h"
#include    "../INPUT/input.h"
#include    "../HANDLE/handle.h"
#include    "../TIMER/timer.h"
#include    "../ADC/adc.h"
#include    "../SPEED/speed.h"
#include    "../BEEP/beep.h"
#include    "../ENCODER/encoder.h"
#include    "../TEST/test.h"
#include    "run.h"
```

Khai báo các biến cần thiết trong chương trình:

```
signed char vitri, straightLine;
unsigned int Run_time;
signed int timerbrakeexchangelane;
```

```
signed int timerbraketurn90;
signed int line;
unsigned char speed_run_forward;

bảng vi sai tính theo công thức cụ thể:
// HE SO VI SAI TUONG DOI
const char hesoR1[51] ={ 100, 99,   98,   97,   97,   96,   95,   94,   93,   92,
      91,   90,   89,   88,   88,   87,   86,   85,   84,   83,   82,   81,   80,
      79,   78,   77,   76,   75,   73,   72,   71,   70,   69,   68,   66,   65,
      64,   62,   61,   60,   58,   57,   55,   53,   52,   50,   48,   46,   44,
      42,   40
}; //BANH XE BEN TRONG
const char hesoR3[51] ={100, 101,   102,  103,  103,  104,  105,  106,  107,  108,
      109,  110,  111,  112,  112,  113,  114,  115,  116,  117,  118,  119,  120,
      121,  122,  123,  124,  125,  127,  128,  129,  130,  131,  132,  134,  135,
      136,  138,  139,  140,  142,  143,  145,  147,  148,  150,  152,  154,  156,
      158,  160
};// BANH XE BEN NGOAI

Hàm run:
void run(void)
{
      // khởi tạo các giá trị ban đầu khi khởi động
      unsigned char pattern;
      handle( 0 );      //bẻ lái thẳng
      speed( 0, 0 ); // tốc độ 2 bánh bằng 0
      pattern = 0; // cài trạng thái chạy về 0 (chuẩn bị xuất phát)
      cnt1 = 0;   // đồng hồ ảo 1 bằng 0, giá trị này tự tăng mỗi ms, nó dùng vào
những giải thuật cần có thời gian xử lý.
      cnt4 = 0;   // đồng hồ ảo 2 dùng để xác định thời gian chạy, cũng tự tăng lên
mỗi ms
      RYGB(0,0,0,0);    // tắt hết các đèn màu trên mạch MCU
      Run_time = 50000; //cài tự động sau 50s xe sẽ tự dừng (đơn vị thời gian là
mili giây)
      vitri = 0;        //cài vị trí góc lệch ban đầu là 0, trong quá trình chạy nó
sẽ tự thay đổi
      straightLine = 1; // xác định xe đang trên đường thẳng, khi vào đường cong
biến này sẽ được xóa
      switch (get_dipsw()) // đọc mode từ dipsw tương ứng tốc độ
      {
            case 7: // 25 %
            timerbraketurn90 = 200; // thời gian thắn xe khi nhận diện được cua
90(đv ms)
            timerbrakeexchangelane = 120;// thời gian thắn xe khi nhận diện được
chuyển làn
            speed_run_forward = 70; // tốc độ chạy của xe (tăng dần theo mode)

            break;
            case 8: // 30 %
            timerbraketurn90 = 300;
            timerbrakeexchangelane = 120;
            speed_run_forward = 80;

            break;
```

```
        case 9: // 40 %
        timerbraketurn90 = 300;
        timerbrakeexchangelane = 120;
        speed_run_forward = 90;

        break;
        case 10: // 50 %
        timerbraketurn90 = 350;
        timerbrakeexchangelane = 120;
        speed_run_forward = 100;

        break;
        case 11: // 60 %
        timerbraketurn90 = 350;
        timerbrakeexchangelane = 120;
        speed_run_forward = 110;


        break;
        case 12: // 70 %
        timerbraketurn90 = 350;
        timerbrakeexchangelane = 130;
        speed_run_forward = 120;

        break;
        case 13: // 80 %
        timerbraketurn90 = 200;
        timerbrakeexchangelane = 130;
        speed_run_forward = 140;

        break;
        case 14: // 90 %
        timerbraketurn90 = 200;
        timerbrakeexchangelane = 130;
        speed_run_forward = 160;

        break;
        case 15: // 100%
        timerbraketurn90 = 300;
        timerbrakeexchangelane = 140;
        speed_run_forward = 200;

        break;
        default: // trường hợp này không có, chỉ thêm vào cho đầy đủ
        timerbraketurn90 = 0;
        timerbrakeexchangelane = 0;
        speed_run_forward = 0;
        break;
}
while( 1 ) { // đoạn code lặp mãi mãi
      switch( pattern ) { //kiểm tra biến trạng thái pattern
      case 0:
            if((get_gatesensor() == 1)){ // nếu nhận cổng
                  pattern = 1; // cài trạng thái bằng 1
```

```
                        beep_long(100); // kêu bíp 100ms
                        _delay_ms(500); // chờ 500ms
                        break;
                  }
                  if (K3) nếu không có cổng hoặc không có cảm biến cổng thì bấm nút
3
                  {
                        pattern = 1;
                        break;
                  }
                  break;

            case 1: // bootspeed
                  if ((get_gatesensor() == 0) || K3) // nếu cổng mở ra hoặc bấm nút
3
                  {
                        pattern = 2; // cài trạng thái sang 2 chuẩn bị lăn bánh
                        beep_long(1000); // beep một tiếng thật dài báo hiệu chạy
                        cnt1 = 0; // cài các biến thời gian về 0
                        cnt4 = 0;
                        break;
                  }
                  else if ((cnt1%200) == 0)// nếu cổng chưa mở hoặc chưa bấm nút 3
thì nó sẽ beep beep xác nhận chuẩn bị chạy
                  {
                        beep_long(100);
                  }
                  break;
                  case 2: phát động
                  if( cnt1 < 500 ) // chạy chậm trong 0.5s đầu để phòng xe sụt áp
                  runforwardline (70);
                  else {
                        pattern = 11;
                        cnt1 = 0;

                  }
                  break;

            case 11:
                  /* time run = cnt4 */
                        if(cnt4 > Run_time){ // thoi gian chay laf 50 giay
                              pattern = 100; // sau khi hết thời gian chạy xe sẽ
dừng hẳn
                              break;
                        }

                  if(check_crossline()) {       // Kiem tra line ngang cua vuong
                        pattern = 21; // nếu có phát hiện cua 90 thì sẽ chuyển sang
trạng thái 21 để thực hiện xử lý cua 90
                        cnt1 = 0; // cài biến thời gian lại bằng 0

                        break;
                  }
```

```
                else {                                  // kiem tra nua line ngang
qua chuyen lan
                      switch (check_crosshalfline())
                      {
                            case HALF_LEFT_LINE: // nếu phát hiện chuyển làn trái
                            cnt1 = 0;

                            pattern = 12; // thì chuyển sang trạng thái 12
                            break;
                            case HALF_RIGHT_LINE: // nếu phát hiện chuyển làn
phải
                            cnt1 = 0;

                            pattern = 13;// thì chuyển sang trạng thái 13
                            break;
                            default :
                            break;
                      }
                }
                runforwardline(speed_run_forward); // nếu không phát hiện vạch
ngang thì cứ chạy bám line với tốc độ speed_run_forward.
                if (straightLine) // nếu xe đang chạy trên đường thẳng thì
                {
                      RYGB(0,0,0,200);// đèn xanh trên vi điều khiển sẽ sáng lên
để chúng ta có giải thuật thắn xe khi vào đường cong nếu bạn nghĩ ra ☺
                }
                break;

          case 12: //trạng thái thực hiện chuyển làn trái
                /* Check of large turning to the right completion */
                if( check_crossline()) {   /* Crossline check even during turning
*/
                      pattern = 21;// kiểm tra nếu bắt được một line ngang sau
khi bắt nữa line thì chuyển sang thực hiện cua 90 nhằm phòng cảm biến bị xéo so với
line ngang.
                      cnt1 = 0;
                      break;
                }
                beep_long(200);// beep một tiếng báo hiệu xử lý chuyển làn trái
                pattern = left_lance(50);// xử lý chuyển làn trái với tốc độ 50
và sau khi xong thì chuyển về trạng thái 11. Tốc độ càng cao thì qua càng nhanh,
nếu quá nhanh sẽ bị văng ra khỏi đường đua.
                cnt1 = 0;
                straightLine = 0;// xóa biến đường thẳng vì xe mới thực hiện xong
chuyển làn chứ không phải chạy bám line
                break;
          case 13: // tương tự như trên cho bên phải

                /* Check of large turning completion to the left */
                if( check_crossline() ) {   /* Crossline check even during large
turn */
                      pattern = 21;
                      cnt1 = 0;
```

```
                        break;
                    }
                    beep_long(200);
                    pattern = right_lance(50);
                    cnt1 = 0;
                    straightLine = 0;
                    break;

                    case 21: // xử lý cua 90
                    /* Process when first crossline is detected */
                    //    beep_long(500);
                    pattern = gocturn90(50); tốc độ là 50, tốc độ càng lớn qua cua
càng nhanh tuy nhiên cần có giải thuật tốt nếu không sẽ bị văng ra khỏi đường đua.
                    cnt1 = 0;
                    straightLine = 0;
                    break;

              case 100:
                    speed(0,0); // trạng thái hết thời gian chạy
                    break;
              default:
                    pattern = 11;
                    cnt1 = 0;

                    break;
              }
        }
}
```

Các hàm phụ trợ:

```
char check_crossline()// kiểm tra line ngang với cảm biến 7led
{
      unsigned char b;

      b = sensor_inp(0xe7);
      if(b == 0xe7)
      return 1;
      else
      return 0;

}
char check_crosshalfline()
{
      if ((check_crossline() == 0) && (straightLine == 1)){
            if
((sensor_inp(0xf0)==0xf0)||(sensor_inp(0xf0)==0xd0)||(sensor_inp(0xf0)==0xb0))
            return HALF_LEFT_LINE;
            else if
((sensor_inp(0x0f)==0x0f)||(sensor_inp(0x0f)==0x0b)||(sensor_inp(0x0f)==0x0d))
            return HALF_RIGHT_LINE;
            else
            return 0;
      }
      else return 0;
```

```
}
void brake(int time)// thắn xe với tốc độ 0 trong thời gian time
{
	cnt2 = 0;
	while (cnt2<time)
	runforwardline(0);
}
void brake_timer (int time,int tocdo)// thắn xe trong thời gian time với tốc độ
tocdo
{
	cnt2 = 0;
	while (cnt2<time)
	runforwardline(tocdo);
}
unsigned char sensor_inp (unsigned char MASK)// hàm mặt nạ (dùng khá nhiều trong xử
lý cua)
{
	return ( bit_change(sensor) & MASK);
}
unsigned char bit_change( unsigned char in )//hàm quay bit (không cần quan tâm)
{
	unsigned char ret = 0;
	unsigned char i;
	for( i=0; i<8; i++ ) {
		ret >>= 1;			/* Right shift of return value  */
		ret |= in & 0x80;		/* Ret bit7 = in bit7           */
		in  <<= 1;			/* Left shift of argument       */
	}
	return ret;
}
void handleAndSpeed (int angle,int speed1)// hàm bẻ lái và điều khiển 2 bánh có vi
sai
{
	int speed2;
	int speed3;
	handle (angle); // bẻ góc
	if (angle<0){
		angle = -angle; // nếu góc âm thì chuyển thành dương
		speed2 = speed1*hesoR1[angle]/100; // tính tốc độ theo bảng vi sai
		speed3 = speed1*hesoR3[angle]/100;
		speed (speed2,speed3);
	}
	else if (angle == 0 )// nếu góc bẻ bằng 0 thì không cần tính vi sai
	{
		speed (speed1,speed1);
	}
	else { // nếu góc dương thì khỏi đổi và tính luôn vi sai qua góc
		speed2 = speed1*hesoR1[angle]/100;
		speed3 = speed1*hesoR3[angle]/100;
		speed (speed3,speed2);
	}
}
```

Hàm bám line và giúp xe chạy nhanh, mượt trên đường thẳng và cong.

```
void runforwardline (int tocdo)
```

```
{
      unsigned char temp;// khai báo biến tạm
      temp = sensor_inp(0xfe);// biến tạm này sẽ chứa giá trị sensor được chuyển
đổi phù hợp với số hex thông qua hàm sensor_inp(0xfe); vì cảm biến 7 led nên ta bỏ
đi giá trị cuối, tức cho bit cuối cùng của biến tạm là 0 vô điều kiện.

      switch(temp) { kiểm tra trạng thái sensor
            case 0x10:// 00010000  GOC THANG xxxoxxx
            case 0x38:// 00111000 xxoooxx
            handleAndSpeed (0,tocdo);
            vitri = 0;// góc thẳng nên bị trí bằng 0
            straightLine = 1; // hiển nhiên biến này cũng bằng 1 và đèn xanh sáng
            break;
//////////////////////////////////////////////////////////////////////////////
/////////////

            case 0x18:// 00011000 LECH TRAI 1
            if ( vitri < -3) // tránh nhận nhầm 2 đường biên là line giữa
            {
                  handleAndSpeed (-30,0);// nếu nhận line biên thì bẻ mạnh vào
trong đồng thời thắn xe lại.
                  break;
            }
            else handleAndSpeed (1,tocdo);// các góc từ 3 độ trở xuống vẫn nhỏ nên
cứ xem như đang chạy thẳng
            vitri = 1;// vị trí tăng lên 1 tức đã lệch rồi
            straightLine = 1; // vẫn xem như thẳng
            RYGB(0,0,0,0);
            break;
//////////////////////////////////////////////////////////////////////////////
/////////

            case 0x01:// 00001000 LECH TRAI 2
            case 0x1c:
            if ( vitri < -3) // tránh nhận nhầm 2 đường biên là line giữa
            {
                  handleAndSpeed (-30,0); );// nếu nhận line biên thì bẻ mạnh vào
trong đồng thời thắn xe lại.

                  break;
            }
            else handleAndSpeed (3,tocdo);// các góc từ 3 độ trở xuống vẫn nhỏ nên
cứ xem như đang chạy thẳng
            vitri = 2;
            straightLine = 1;
            RYGB(0,0,0,0);
            break;
//////////////////////////////////////////////////////////////////////////////
/////////////

            case 0x0c:// 00000100 LECH TRAI 3
            if ( vitri < -3) // tránh nhận nhầm 2 đường biên là line giữa
            {
```

```
                handleAndSpeed (-30,0); );// nếu nhận line biên thì bẻ mạnh vào
trong đồng thời thắn xe lại.

                break;
            }
            else handleAndSpeed (7,tocdo); //góc lệch lớn nếu xe lắc thì cần cân
chỉnh góc 7 độ này cho phù hợp
            vitri = 3;
            straightLine = 0;RYGB(0,0,0,0);// biến thẳng bị xóa và đèn xanh tắt
            break;
////////////////////////////////////////////////////////////////////////////////
///////

            case 0x04:// 00000100 LECH TRAI 4
            if ( vitri < -3) // tránh nhận nhầm 2 đường biên là line giữa
            {
                handleAndSpeed (-30,0); );// nếu nhận line biên thì bẻ mạnh vào
trong đồng thời thắn xe lại.

                break;
            }
            else handleAndSpeed (16,tocdo*85/100);  // góc bẻ 16 độ và giảm tốc độ
xuống còn 85%. Các bạn chú ý chỉnh góc và tốc độ cho phù hợp xe mới chạy mượt được
            RYGB(0,0,0,0);
            vitri = 4;
            straightLine = 0;
            break;
////////////////////////////////////////////////////////////////////////////////
//////////////////////

            case 0x06:// 00000110 LECH TRAI 5
            if ( vitri < -3)
            {
                handleAndSpeed (-30,0); );// nếu nhận line biên thì bẻ mạnh vào
trong đồng thời thắn xe lại.

                break;
            }
            else handleAndSpeed (16,tocdo*85/100);
            RYGB(0,0,0,0);
            vitri = 5;
            straightLine = 0;
            break;
////////////////////////////////////////////////////////////////////////////////
//////////////////////////

            case 0x02:// 00000010 LECH TRAI 6
            if ( vitri < -3)
            {
                handleAndSpeed (-30,0);
                break;
            }
            else handleAndSpeed (21,tocdo*85/100);
            RYGB(0,0,0,0);
```

```
            vitri = 6;
            straightLine = 0;
            break;
////////////////////////////////////////////////////////////////////////////////
//////////////////////////

            case 0x82:// 10000010 VI TRI MAT LINE
            case 0x00:// 00000000
            if ( vitri < -4)
            {
                  handleAndSpeed (-27,tocdo*50/100);
                  break;
            }
            else if ( vitri > 4)
            {
                  handleAndSpeed (27,tocdo*50/100);
                  break;
            }
            break;
////////////////////////////////////////////////////////////////////////////////
////////////////////////////

            case 0x30:// 00110000 LECH PHAI 1
            if ( vitri > 3)
            {
                  handleAndSpeed (30,0);
                  break;
            }
            else handleAndSpeed (-1,tocdo);
            vitri = -1;
            straightLine = 1;
            break;
////////////////////////////////////////////////////////////////////////////////
//////////////////

            case 0x20:// 00100000 LECH PHAI 2
            case 0x70:
            if ( vitri > 3)
            {
                  handleAndSpeed (30,0);
                  break;
            }
            else handleAndSpeed (-3,tocdo);
            vitri = -2;
            straightLine = 1;
            break;
////////////////////////////////////////////////////////////////////////////////
//////////////////

            case 0x60:// 01100000 LECH PHAI 3
            if ( vitri > 3)
            {
                  handleAndSpeed (30,0);
                  break;
```

```
            }
            else handleAndSpeed (-7,tocdo);
            vitri = -3;
            straightLine = 0;
            RYGB(0,0,0,0);
            break;
////////////////////////////////////////////////////////////////////////////////
///////////////

            case 0x40:// 01100000 LECH PHAI 4

            if ( vitri > 3)
            {
                  handleAndSpeed (30,0);
                  break;
            }
            else handleAndSpeed (-16,tocdo*85/100);
            RYGB(0,0,0,0);
            vitri = -4;
            straightLine = 0;
            break;
////////////////////////////////////////////////////////////////////////////////
////////////

            case 0xc0:// 11000000 LECH PHAI 5
            if ( vitri > 3)
            {
                  handleAndSpeed (30,0);
                  break;
            }
            else handleAndSpeed (-16,tocdo*85/100);
            RYGB(0,0,0,0);
            vitri = -5;
            straightLine = 0;
            break;
////////////////////////////////////////////////////////////////////////////////
//////////////////////////

            case 0x80:// 10000000 LECH PHAI 6
            if ( vitri > 3)
            {
                  handleAndSpeed (30,0);
                  break;
            }
            else handleAndSpeed (-21,tocdo*85/100);
            RYGB(0,0,0,0);
            vitri = -6;
            straightLine = 0;
            break;
////////////////////////////////////////////////////////////////////////////////
//////////////////////////


            default:
```

```
            straightLine = 1;
            speed(0,0);
            break;
      }
}
```

Các hàm chưa hiện thực và các bạn phải thực hiện các giải thuật cho việc xử lý cua 90 và chuyển làn.

```
int gocturn90(int tocdo)
{
      <<Code của bạn viết ở đây>>
      return 11;
}
int left_lance (int tocdo)
{
      <<Code của bạn viết ở đây>>
      return 11;
}
int right_lance(int tocdo)
{
      <<Code của bạn viết ở đây>>
      return 11;
}
```

Tobe continued...