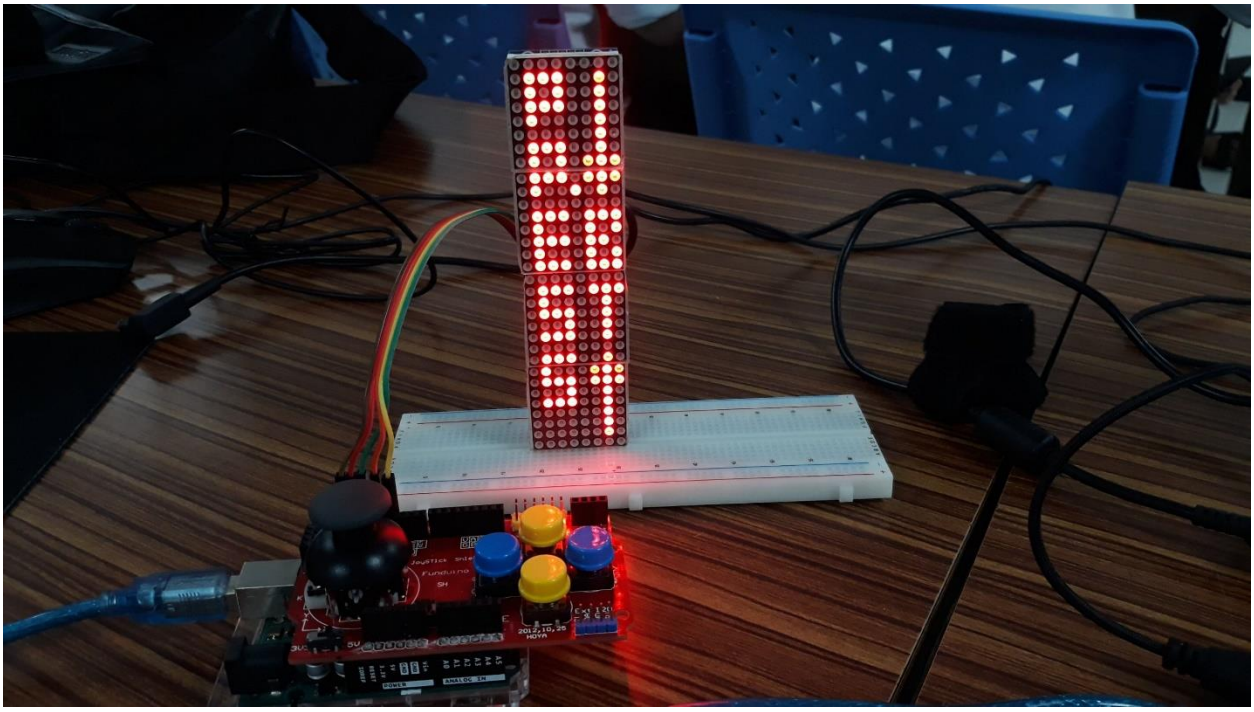
LED MATRIX (8X32) PROJECT

GAME : STAR LASER

กลุ่ม : ขายตรงแบบ 300 %



LED DOT MATRIX PROJECT'S DETAILED

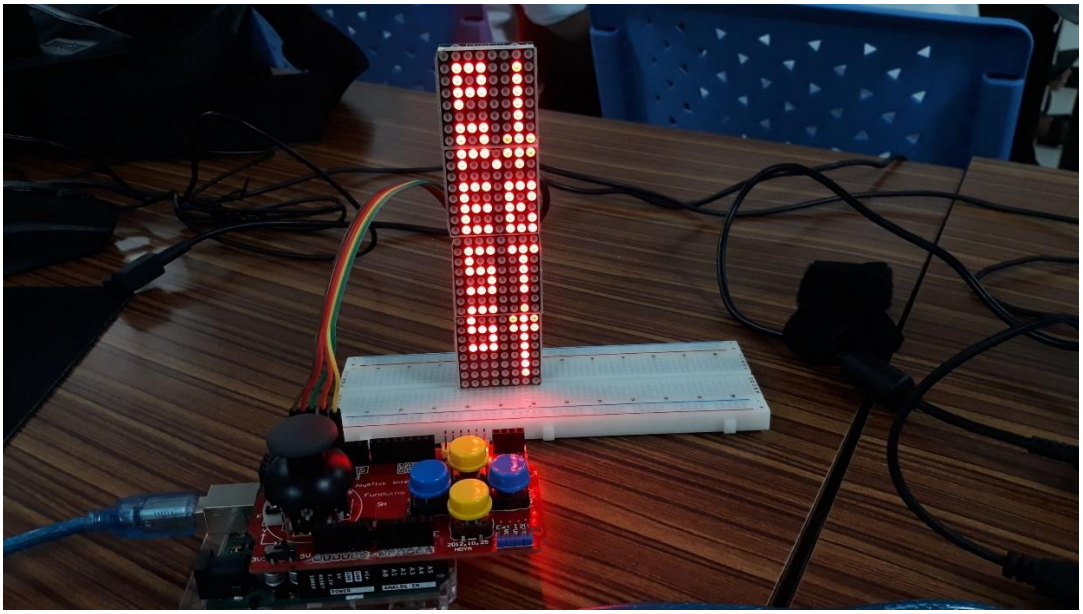
1. เกมส์นี้จะเป็นการแข่งขันระหว่างผู้เล่นกับ BOT

2. ผู้เล่นสามารถ เคลื่อนย้ายยาน โดยการโยกปุ่มอนาล๊อค

3. ผู้เล่นสามารถ ยิงกระสุน โดยการกดปุ่ม B(ปุ่มขวา)

4. เกมส์นี้จะมีผลลัพธ์ออกมา 2 รูปแบบ คือ ชนะ หรือ แพ้

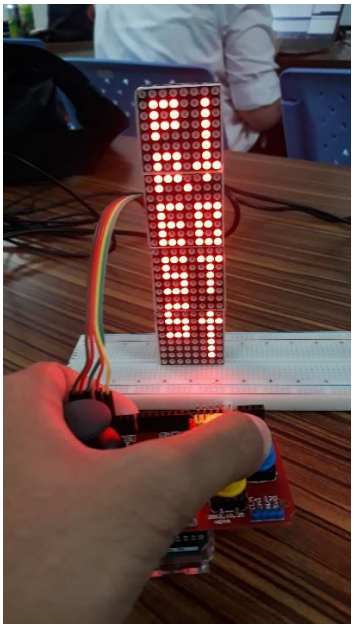
PICTURE AND DISCUSSION

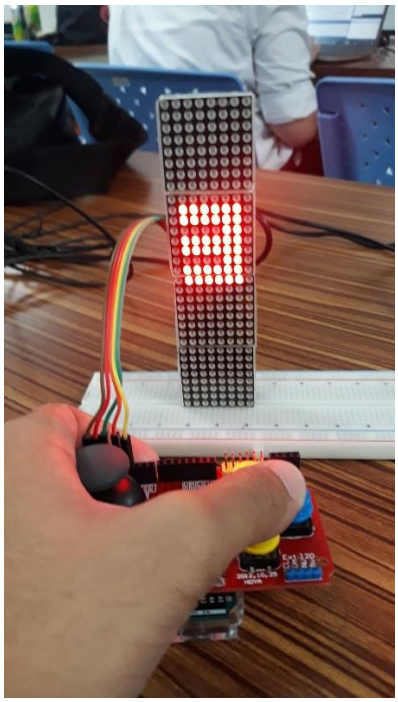
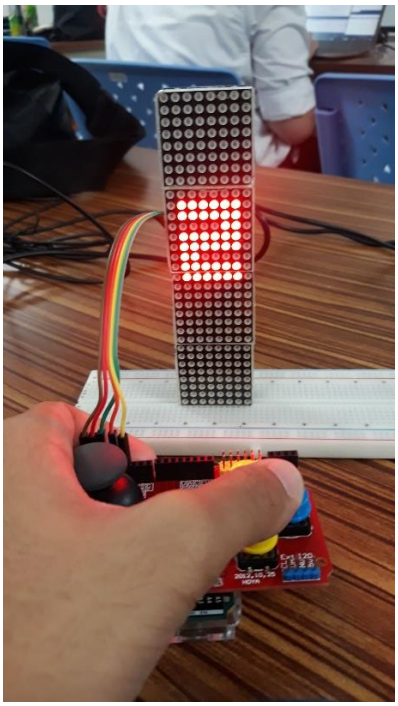
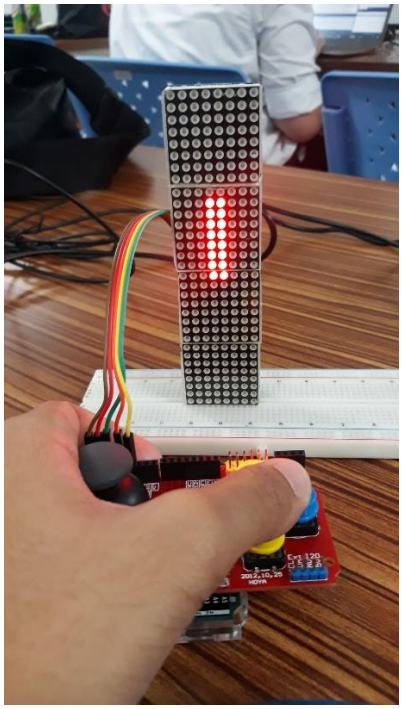


เริ่มต้น ก่อนที่จะเข้าไปสู่ตัวเกม เราจะเจอกับ Interface

“Press BT” ซึ่งหมายความว่าให้เรากดปุ่มใดปุ่มหนึ่ง

(A , B , C หรือ D) เพื่อเล่นเกม



หลังจากที่เรากดปุ่มเพื่อเข้าเกมแล้ว หน้าจอก็จะแสดง Interface ที่เป็นตัวเลข 3 , 2 และ 1 เพื่อให้ทราบว่า

เกมที่เราจะเล่น กำลังจะเริ่มใน 3 วินาที

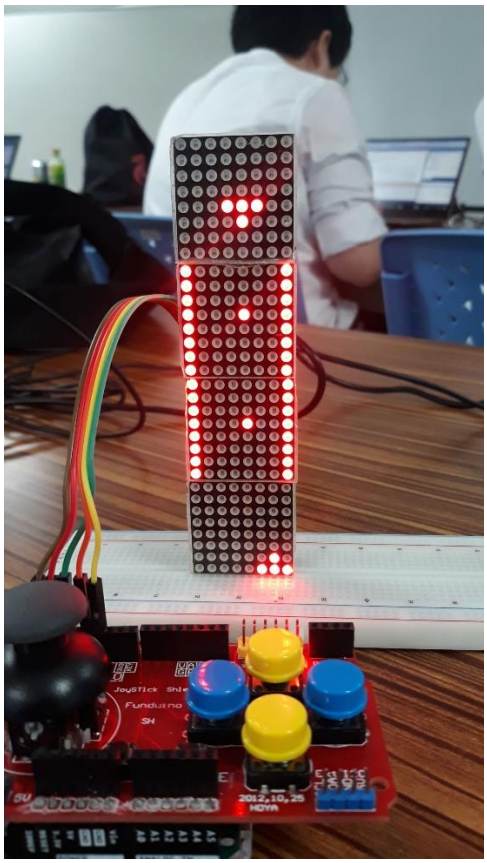
3

2

1

...

GAME START!



PART OF GAME

1.ยานผู้เล่น - ยานผู้เล่นจะอยู่ข้างล่างสุดของตัวจอ

ผู้เล่นสามารถขยับยานได้ 4 ทิศ (ขึ้น,ลง,ซ้าย,ขวา)

โดยกดโยกปุ่มอนาล๊อค (Analog)

2.ยานศัตรู - ยานศัตรูจะอยู่ด้านบนสุดของตัวจอ

ยานของศัตรูจะเคลื่อนที่อัตโนมัติแบบสุ่ม

3.พลังชีวิต – พลังชีวิตจะอยู่ด้านซ้ายและขวาของตัวจอ

ด้านซ้ายของจอ – พลังชีวิตของผู้เล่นจำนวน 16 ชีวิต

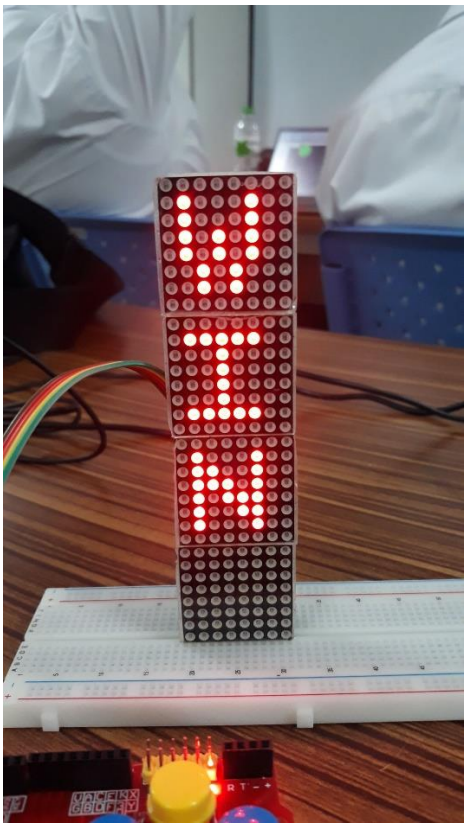
ด้านขวาของจอ – พลังชีวิตของศัตรูจำนวน 16 ชีวิต

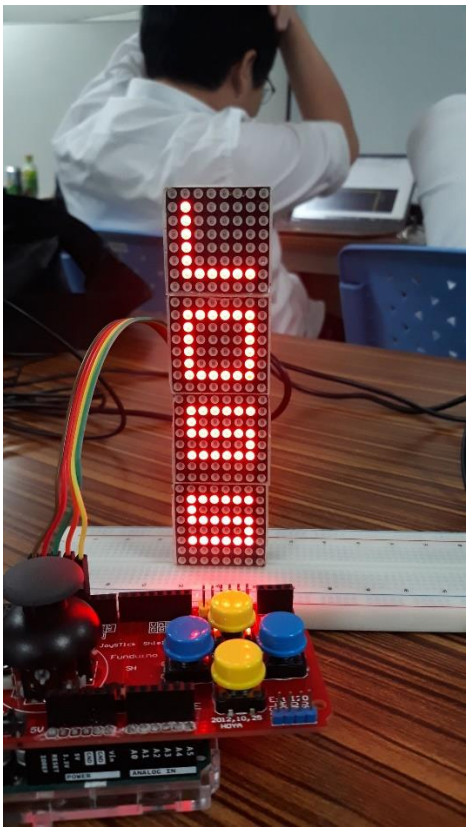
หมายเหตุ : พลังชีวิตจะลดลงทีละ 1 ต่อกระสุน 1 นัด

4.ผลลัพธ์ของเกม – ชนะ หรือ แพ้

ถ้าพลังชีวิตของศัตรูหมดก่อน ผู้เล่นก็จะ ชนะ

ถ้าพลังชีวิตของผู้เล่นหมดก่อน ผู้เล่นก็จะ แพ้





MAIN PROCESS OF PROJECT

ก่อนที่จะเริ่ม coding นั้น เราจะใช้ lib LedControl.h เข้ามาช่วย เพื่อให้ประหยัดพื้นที่ในการเขียน code และเรียกใช้ฟังก์ชั่นอื่นๆได้อีกด้วย เช่น

lc.setLed(); lc.setRow(); lc.setCol(); เป็นต้น

```
#include "LedControl.h"

/////////////////////////////////// DEFINE VARIABLE ////////////////////////////////////////////
LedControl lc=LedControl(11,13,10,4);  // DIN,CLK,CS,Number of LED Module
```

ในรูปเราจะกำหนดให้ Din = 11 / CLK = 13 / CS = 10

และเราจำควบคุมโมดูลทั้งหมด 4 **ตัวด้วยกัน**

หมายเหตุ :

8x8 LED MATRIX = 1 module

8x32 LED MATRIX = 4 modules

```
//////////////////////////////////// MAIN LOOP /////////////////////////////////////////////////
void loop() {
  Input_system();

  switch (main_state){
    case 0: Setup_interface();    //Main menu
            if(UP == 0 || DOWN == 0 || LEFT == 0 || RIGHT == 0){  // If something was pressed...
              clear_display();
              Pre_interface();
              clear_display();

              Set_newgame();
              main_state = 1;
            }
            break;

    case 1: Control_ship();       //MY SHIP
            Check_bullet();

            Control_enemy();      //ENEMY
            Check_enemybullet();

            Check_collision();    //CHECK if have collision
            if(HP == 24 || enemyHP == 24){
              clear_display();
              main_state = 2;
            }
            break;

    case 2: End_interface();      //Tell Win or Lose
            main_state = 0;
            break;
  }

  delay(50);
}
```

ฟังก์ชั่นการทำงานหลักของเราก็จะเป็นดังรูปนี้

รายละเอียดการทำงานของแต่ละฟังก์ชั่นจะอยู่ด้านล่าง

```
//////////////////////////////// GAME SYSTEM ////////////////////////////////////////////
void Input_system(){
  now = millis();
  joy_x = map(analogRead(joystick_axis_x),0,1000,-1,1);
  joy_y = map(analogRead(joystick_axis_y),0,1000,-1,1);
  UP    = digitalRead(up_button);
  DOWN  = digitalRead(down_button);
  LEFT  = digitalRead(left_button);
  RIGHT = digitalRead(right_button);
```

Input_system();

คือฟังก์ชั่นที่บ่งบอกว่าเราจะกำหนดค่าปุ่มไว้ที่เท่าไหร่ ใช้งานอย่างไร ให้ไปทิศทางไหน และฟังก์ชั่นนี้ก็จะใช้ map(); กับค่า Analog หรือปุ่มโยกเพื่อให้การเคลื่อนที่ง่ายและตรงกับการเคลื่อนย้ายพิกัดใน LED MATRIX 8x32

เช่น เราต้องการให้ยานเราเคลื่อนที่ไปทางซ้าย เราก็จะเปลี่ยนค่าจากช่วง 0-1023 ให้เป็น -1(ซ้าย) และ 1(ขวา) เพื่อที่จะได้เข้าใจง่ายมากขึ้น พอเรากำหนดค่าให้ปุ่มต่างๆแล้ว เราก็จะมาเข้าสู่การทำงานของเกมกันเลยครับ // ส่วนของ input

---

ต่อมาเราก็จะใช้ Switch case เนื่องจากตัวเกมของเราแบ่งได้เป็น 3 ช่วงหลักๆ คือ

1.ช่วงแรก

```
switch (main_state){
  case 0: Setup_interface();    //Main menu
          if(UP == 0 || DOWN == 0 || LEFT == 0 || RIGHT == 0){  // If something was pressed...
            clear_display();
            Pre_interface();
            clear_display();

            Set_newgame();
            main_state = 1;
          }
          break;
```

เราจะเข้า case แรกของตัวโปรแกรมนั่นก็คือ main state ซึ่งเรากำหนดไว้ตั้งแต่แรกแล้วว่า main state = 0 `int main_state = 0;`

ซึ่งมาถึงเราก็จะเจอฟังก์ชั่น 1.1 Setup_interface(); ก่อน

```
void Setup_interface(){
   // P
  plot(2,2);  plot(2,3);  plot(2,4);
  plot(2,5);  plot(3,2);  plot(3,4);
  plot(4,2);  plot(4,3);  plot(4,4);

  // R
  plot(2,7);  plot(2,8);  plot(2,9);
  plot(2,10); plot(3,7);  plot(3,9);
  plot(4,7);  plot(4,8);  plot(4,9);
  plot(5,10);

  // E
  plot(2,12); plot(2,13); plot(2,14);
  plot(2,15); plot(2,16); plot(3,12);
  plot(3,14); plot(3,16); plot(4,12);
  plot(4,14); plot(4,16);

  // S
  plot(2,18); plot(2,19); plot(2,20);
  plot(2,22); plot(3,18); plot(3,20);
  plot(3,22); plot(4,18); plot(4,20);
  plot(4,21); plot(4,22);

  // S
  plot(2,24); plot(2,25); plot(2,26);
  plot(2,28); plot(3,24); plot(3,26);
  plot(3,28); plot(4,24); plot(4,26);
  plot(4,27); plot(4,28);

  // B
  plot(6,12); plot(6,13); plot(6,14);
  plot(6,15); plot(6,16); plot(7,12);
  plot(7,14); plot(7,16); plot(8,12);
  plot(8,13); plot(8,15); plot(8,16);

  // T
  plot(6,18); plot(7,18); plot(7,19);
  plot(7,20); plot(7,21); plot(7,22);
  plot(8,18);

  if(now - last >= 500 ){
```

ซึ่งฟังก์ชั่นนี้จะแสดง Interface PRESS BT. ตอนแรกของตัวโปรแกรมนั่นเอง ซึ่งจะมีการใช้ฟังก์ชั่น plot(,); มาเกี่ยวข้องกับฟังกชั่นนี้อีกด้วย

```
void clear_display() { //clear All 4 screen
  for (int address = 0; address < 4; address++) {
    lc.clearDisplay(address);
  }
}
void plot_display(){
  for(int i = 1; i<=8; i++){
    for(int j = 1; j<=32; j++){
      plot(i,j);
      delay(1);
    }
  }
  for(int i = 1; i<=8; i++){
    for(int j = 1; j<=32; j++){
      delete_plot(i,j);
      delay(1);
    }
  }
}
```

1.2 clear_display(); คือฟังก์ชั่นที่สั่งให้เคลีย address ใน module ตัวนั้นๆ ซึ่งเราจะสั่งให้เคลีย 4 module ไปก่อน

```
void plot(int x,int y){ //8,32
  int address;
  int row = 8-x; //0-7
  int col = (y-1)%8; //0-7
  if(y <= 8){address = 3;}
  else if(y <= 16){address = 2;}
  else if(y <= 24){address = 1;}
  else if(y <= 32){address = 0;}
  lc.setLed(address,row,col,true);
}
void delete_plot(int x,int y){ //8,32
  int address;
  int row = 8-x; //0-7
  int col = (y-1)%8; //0-7
  if(y <= 8){address = 3;}
  else if(y <= 16){address = 2;}
  else if(y <= 24){address = 1;}
  else if(y <= 32){address = 0;}
  lc.setLed(address,row,col,false);
}
```

1.3.1 plot(int x,int y); คือฟังก์ชั่นที่กำหนดพิกัดในการแสดงตำแหน่งของไฟในตัว LED MATRIX

1.3.2 delete_plot(int x,int y); คือฟังก์ชั่นที่กำหนดพิกัดในการลบตำแหน่งของไฟในตัว LED MATRIX

เหตุผลที่สร้างฟังก์ชั่นนี้ขึ้นมา

- เนื่องจากการจ่ายไฟของ LED MATRIX 8x32 นั้น จะจ่ายจากช่อง 0-7 ในแกน x และ ช่อง 0-7 ในแกน y ของแต่ละโมดูล

เพื่อไม่ให้สับสนในการกำหนดพิกัด จึงสร้างฟังก์ชั่นที่

สามารถป้อนค่า 1-8 ได้ในแกน x และ plot จุดไฟ

สามารถป้อนค่า 1-32 ได้ในแกน y และ plot จุดไฟ

- และปกติ LED MATRIX จะนับตั้งแต่ล่างขึ้นข้างบนและนับไม่ติดกัน ในแกน y คือ 0-7 (4รอบ) เราจึงเลยสร้างฟังก์ชั่นนี้ขึ้นมาพร้อมกับ

ให้เริ่มนับจากข้างบนซ้ายเป็นพิกัด 1,1 จนถึง ด้านล่างขวา 8,32

เพื่อง่ายต่อการเขียนโปรแกรม

// ฟังก์ชั่นที่กล่าวมาจากข้างต้นเป็นฟังก์ชั่นที่ถูกเรียกใช้ภายในโปรแกรมอยู่ตลอดทั้ง 3 ส่วน

เนื่องจากวงจรเราเป็นการต่อแบบ INTERNAL_PULLUP

เราจึงกำหนดให้ว่า ถ้ายังไม่กดปุ่ม (1) ให้แสดง INTERFACE (PRESS BT) ไปเรื่อยๆ จนกว่าจะมีการกดปุ่ม (0)

หลังจากกดปุ่มแล้ว ก็จะมี INTERFACE ตัวเลขขึ้นมาให้แสดงว่าอีก 3 วินาทีเกมจะเริ่มขึ้น 1.4 Pre_interface();

```
void Pre_interface(){
  // 3
  plot(2,10); plot(2,11); plot(2,13); plot(2,14);
  plot(2,16); plot(2,17); plot(3,10); plot(3,11);
  plot(3,13); plot(3,14); plot(3,16); plot(3,17);
  plot(4,10); plot(4,11); plot(4,13); plot(4,14);
  plot(4,16); plot(4,17); plot(5,10); plot(5,11);
  plot(5,13); plot(5,14); plot(5,16); plot(5,17);
  plot(6,10); plot(6,11); plot(6,12); plot(6,13);
  plot(6,14); plot(6,15); plot(6,16); plot(6,17);
  plot(7,10); plot(7,11); plot(7,12); plot(7,13);
  plot(7,14); plot(7,15); plot(7,16); plot(7,17);
  delay(1000);
  clear_display();
  delay(100);

  // 2
  plot(2,10); plot(2,11); plot(2,13); plot(2,14);
  plot(2,16); plot(2,17); plot(3,10); plot(3,11);
  plot(3,13); plot(3,14); plot(3,16); plot(3,17);
  plot(4,10); plot(4,11); plot(4,13); plot(4,14);
  plot(4,16); plot(4,17); plot(5,10); plot(5,11);
  plot(5,13); plot(5,14); plot(5,16); plot(5,17);
  plot(6,10); plot(6,11); plot(6,12); plot(6,13);
  plot(6,14); plot(6,16); plot(6,17); plot(7,10);
  plot(7,11); plot(7,12); plot(7,13); plot(7,14);
  plot(7,16); plot(7,17); plot(2,15); plot(3,15);
  delay(1000);
  clear_display();
  delay(100);

  // 1
  plot(4,10); plot(4,11); plot(4,12); plot(4,13);
  plot(4,14); plot(4,15); plot(4,16); plot(4,17);
  plot(5,10); plot(5,11); plot(5,12); plot(5,13);
  plot(5,14); plot(5,15); plot(5,16); plot(5,17);
  delay(1000);
  clear_display();
  delay(100);
}
```

เป็นการใส่ INTERFACE เพิ่มเติมก่อนเริ่มเกม เพื่อนที่จะแสดงว่า เกมกำลังจะเริ่มใน 3...2...1→ GAME START!

หลังจากนี้จะเข้าสู่ตัวเกมของโปรแกรมทันที

1.5 Set_newgame(); คือ ฟังก์ชั่นที่เป็นการเริ่ม Set เกมส์ใหม่ของโปรแกรม ซึ่งจะทำงานหลังจาก Pre_interface();

```
void Set_newgame(){
  setup_healthbar();
  set_bulletstate();
  set_enemybulletstate();
  draw_ship(x_ship,y_ship);  //set draw_ship
  draw_enemy(x_enemy,y_enemy); //set draw_enemy
}
```

กำหนดฟังก์ชั่นนี้ขึ้นมาเพื่อแสดงยานผู้เล่น,ยานศัตรู,พลังชีวิตของผู้เล่นและศัตรู/ฟังก์ชั่นสถานะกระสุน

```
//////////////////////////////// MY SHIP ////////////////////////////////////////////
void draw_ship(int x, int y){
  plot(x,y);
  plot(x,y-1);
  plot(x-1,y);
  plot(x+1,y);
}
void delete_ship(int x, int y){
  delete_plot(x,y);
  delete_plot(x,y-1);
  delete_plot(x-1,y);
  delete_plot(x+1,y);
}

//////////////////////////////// ENEMY SHIP ////////////////////////////////////////////
void draw_enemy(int x, int y){
  plot(x,y);
  plot(x+1,y);
  plot(x-1,y);
  plot(x,y+1);
}
void delete_enemy(int x, int y){
  delete_plot(x,y);
  delete_plot(x-1,y);
  delete_plot(x+1,y);
  delete_plot(x,y+1);
}
```

กำหนดฟังก์ชั่นให้วาดและลบตัวยานทั้งสองฝั่งเพื่อเคลื่อนที่แบบไม่ทิ้งรอยเก่า

```
void setup_healthbar(){
  HP = 8;
  enemyHP = 8;
  for(int y = HP+1; y <= 24 ; y++){        //LEFT SIDE for Me
    plot(1,y);
  }
  for(int y = enemyHP+1; y <= 24 ; y++){    //RIGHT SIDE for Enemy
    plot(8,y);
  }
}
void Health_bar(){
    delete_plot(1,HP);
    delete_plot(8,enemyHP);
}
```

กำหนดฟังก์ชั่นพลังชีวิตทั้งสองฝั่ง (มีค่า16ชีวิต) แล้วฝั่งซ้ายเป็นของผู้เล่น ฝั่งขวาเป็นของศัตรู

หลังจากนั้นเรากำหนดให้ main_sate = 1 เพื่อที่จะทำให้เราเข้าสู่ case 1 ได้ (ช่วงที่สอง) // จบช่วงแรก

2.ช่วงที่สอง คือ ช่วงของตัวเกมหลัก โดยเราจะเจอฟังก์ชั่น Control_ship(); ก่อน

```
    case 1: Control_ship();       //MY SHIP
            Check_bullet();

            Control_enemy();      //ENEMY
            Check_enemybullet();

            Check_collision();    //CHECK if have collision
            if(HP == 24 || enemyHP == 24){
              clear_display();
              main_state = 2;
            }
            break;
```

2.1 Control_ship();

```
void Control_ship(){
  draw_ship(x_ship,y_ship);
  if(joy_x == -1) {if(x_ship!=2)    {delete_ship(x_ship,y_ship);  draw_ship(--x_ship,y_ship);}  }     //LEFT
  if(joy_x == 1)  {if(x_ship!=7)    {delete_ship(x_ship,y_ship);  draw_ship(++x_ship,y_ship);}  }     //RIGHT
  if(joy_y == -1) {if(y_ship!=32)   {delete_ship(x_ship,y_ship);  draw_ship(x_ship,++y_ship);}  }     //DOWN
  if(joy_y == 1)  {if(y_ship!=26)   {delete_ship(x_ship,y_ship);  draw_ship(x_ship,--y_ship);}  }     //UP
```

เราจะกำหนดให้ปุ่มของเราเคลื่อนที่โดยการโยกปุ่ม และโยกได้ 4 ทิศ และจำกัดขอบเขตให้กับยานด้วย

2.2 Control_enemy();

```
void Control_enemy(){
  int randNum = random(0,10);
  draw_enemy(x_enemy,y_enemy);
  if(randNum == 3)  {if(x_enemy!=2)  {delete_enemy(x_enemy,y_enemy);  draw_enemy(--x_enemy,y_enemy);}  } //RANDOM LEFT
  if(randNum == 7)  {if(x_enemy!=7)  {delete_enemy(x_enemy,y_enemy);  draw_enemy(++x_enemy,y_enemy);}  } //RANDOM RIGHT
  if(randNum == 4)  {if(y_enemy!=7)  {delete_enemy(x_enemy,y_enemy);  draw_enemy(x_enemy,++y_enemy);}  } //RANDOM DOWN
  if(randNum == 8)  {if(y_enemy!=1)  {delete_enemy(x_enemy,y_enemy);  draw_enemy(x_enemy,--y_enemy);}  } //RANDOM UP
}
```

เราจะกำหนดให้ยานของศัตรูเคลื่อนที่อัตโนมัติแบบสุ่ม และจำกัดขอบเขตให้กับยานด้วย

2.3 Check_bullet();

```
//////////////////////////////// MY BULLETS ////////////////////////////////////////////
void Check_bullet(){
  if(RIGHT == 0){
    for(int i = 0; i < Num_bullet ; i++ ){ // Check Bullet
      if(bullet_state[i] == 0){
        bullet_state[i] = 1;
        set_bullet(i);
        break;
      }
    }
  }
  for(int i = 0 ; i < Num_bullet ; i++){
    if(bullet_state[i] == 1){
      delete_bullet(x_bullet[i],y_bullet[i]);
      if(x_bullet[i] == x_enemy && y_bullet[i] == y_enemy+1 ||
         x_bullet[i] == x_enemy-1 && y_bullet[i] == y_enemy ||
         x_bullet[i] == x_enemy+1 && y_bullet[i] == y_enemy){
        enemyHP++;
        Health_bar();
        bullet_state[i] = 0;
        x_bullet[i] = 0; y_bullet[i] = 0;
        Tempstate = 1;
      }
      if(y_bullet[i]!=0 && Tempstate == 0){
        draw_bullet(x_bullet[i],--y_bullet[i]);
      }
      else{
        Tempstate = 0;
        bullet_state[i] = 0;
        x_bullet[i] = 0; y_bullet[i] = 0;
      }
    }
  }
```

เราจะทำการเช็คกระสุนว่ากระสุนจะเคลื่อนที่ในรูปแบบไหน แล้วเราจะกำหนดขนาดกระสุนด้วย (ถ้าปุ่มขวากด ปืนถึงจะยิงกระสุน)

ถ้ากระสุนเรายิงไม่โดนอะไรเลย ศัตรูก็จะเลือดไม่ลดนั่นเอง

2.4 Check_enemybullet();

```
void Check_enemybullet(){
  int randNum = random(0,7);
  if(randNum == 4){
    for(int i = 0; i < Num_bullet ; i++ ){ // Check Bullet
      if(bullet_enemystate[i] == 0){
        bullet_enemystate[i] = 1;
        set_enemybullet(i);
        break;
      }
    }
  }
  for(int i = 0 ; i < Num_bullet ; i++){
    if(bullet_enemystate[i] == 1){
      delete_enemybullet(x_enemybullet[i],y_enemybullet[i]);
      if(x_enemybullet[i] == x_ship && y_enemybullet[i] == y_ship-1 ||
         x_enemybullet[i] == x_ship-1 && y_enemybullet[i] == y_ship ||
         x_enemybullet[i] == x_ship+1 && y_enemybullet[i] == y_ship){
        HP++;
        Health_bar();
        bullet_enemystate[i] = 0;
        x_enemybullet[i] = 0; y_enemybullet[i] = 0;
        enemyTempstate = 1;
      }
      if(y_enemybullet[i]!=32 && enemyTempstate == 0){
        draw_enemybullet(x_enemybullet[i],++y_enemybullet[i]);
      }
      else{
        enemyTempstate = 0;
        bullet_enemystate[i] = 0;
        x_enemybullet[i] = 0; y_enemybullet[i] = 0;
      }
    }
  }
}
```

กระสุนฝ่ายศัตรูจะถูกยิงโดยอัตโนมัติ เพราะเราใช้ฟังก์ชั่น random(); ค่ากระสุนนั่นเองและกำจัดจำนวนกระสุนให้กับยานศัตรูด้วย

ถ้าศัตรูยิงเราไม่โดน เลือดของยานเราก็จะไม่ลดนั่นเอง

2.5 Check_collision();

```
void Check_collision(){
  for(int i = 0; i < Num_bullet; i++){
    if(y_bullet[i] < 30){
      for(int j = 0; j < Num_bullet; j++){
        if(y_enemybullet[j] >2){
            if(y_bullet[i] == y_enemybullet[j] && x_bullet[i] == x_enemybullet[j]){
              bullet_state[i] = 0;
              bullet_enemystate[j] = 0;
              delete_plot(x_bullet[i],y_bullet[i]);
              delete_plot(x_enemybullet[j],y_enemybullet[j]);
              x_bullet[i] = 0; y_bullet[i] = 0;
              x_enemybullet[j] = 0,y_enemybullet[j] = 0;
            }
            else if(y_bullet[i] == 1+y_enemybullet[j] &&  x_bullet[i] == x_enemybullet[j]){
              bullet_state[i] = 0;
              bullet_enemystate[j] = 0;
              delete_plot(x_bullet[i],y_bullet[i]);
              delete_plot(x_enemybullet[j],y_enemybullet[j]);
              x_bullet[i] = 0; y_bullet[i] = 0;
              x_enemybullet[j] = 0; y_enemybullet[j] = 0;
            }
        }
      }
    }
  }
}
```

ฟังก์ชั่นนี้จะเป็นการเช็คตำแหน่งของกระสุนที่ชนกัน ก่อนอื่นเราต้องกำหนดให้ระยะกระสุนในแต่ละฝั่งตันที่ y=30 ไม่ฉะนั้นกระสุนจะเลยออกนอกเจอไปเรื่อยๆ

ถ้ากระสุนของผู้เล่นและกระสุนของศัตรูมาชนกัน จะทำให้กระสุน ณ ตำแหน่งที่ชนกันไฟดับ กล่าวคือเป็นการชนแล้วหายนั่นเอง

ทั้งสองอย่างนี้ เมื่อเป็นจริงอันใดอันหนึ่งให้ set สถานะของกระสุนกลับไปพร้อมยิงอีกครั้ง

หลังจากฟังก์ชั่นที่กล่าวไปข้างต้น จะเป็นเงื่อนไขพลังชีวิตของผู้เล่นและศัตรูนั่นเอง

```
if(HP == 24 || enemyHP == 24){
  clear_display();
  main_state = 2;
}
break;
```

เงื่อนไขนี้กล่าวคือ ถ้า HP หรือเลือดฝั่งใดฝั่งหนึ่งเป็น 24 (ก็คือเลือดหมดนั่นเอง) จะให้ล้างจอแล้วไปเข้า main_state = 2 เลย เพื่อเข้า case 2 หรือ ช่วงที่3 ของตัวโปรแกรมครับ

// จบช่วงที่สอง

3.ช่วงที่สาม(สุดท้าย) คือ ช่วงที่แสดง Interface ว่าเราชนะหรือแพ้(Win or Loss)

```
    case 2: End_interface();      //Tell Win or Lose
            main_state = 0;
            break;
  }

  delay(50);
}
```

```
void End_interface(){
  plot_display();
  delay(200);
  if(HP == 24){
    for(int i = 0; i < 4; i++){ //Health reach 0
      Lose();
      delay(700);
      clear_display();
      delay(300);
    }
  }
  else if(enemyHP == 24){  //enemyHealth reach 0
    for(int i = 0; i < 4; i++){ //Health reach 0
      Win();
      delay(700);
      clear_display();
      delay(300);
    }
  }
}
```

3.1 End_interface(); - ฟังก์ชั่นนี้จะแสดง

3.1.1 win();

3.1.2 lose():

```
void Win(){
  // W
  plot(2,2);  plot(2,3);  plot(2,4);  plot(2,5);
  plot(3,6);  plot(3,7);  plot(4,4);  plot(4,5);
  plot(5,6);  plot(5,7);  plot(6,2);  plot(6,3);
  plot(6,4);  plot(6,5);

  // I
  plot(2,10); plot(2,15); plot(3,10); plot(3,15);
  plot(4,10); plot(4,11); plot(4,12); plot(4,13);
  plot(4,14); plot(4,15); plot(5,10); plot(5,15);
  plot(6,10); plot(6,15);

  //N
  plot(2,18); plot(2,19); plot(2,20); plot(2,21);
  plot(2,22); plot(2,23); plot(3,19); plot(3,20);
  plot(4,20); plot(4,21); plot(5,21); plot(5,22);
  plot(6,18); plot(6,19); plot(6,20); plot(6,21);
  plot(6,22); plot(6,23);
}
void Lose(){
  //L
  plot(2,2);  plot(2,3);  plot(2,4);  plot(2,5);
  plot(2,6);  plot(2,7);  plot(3,7);  plot(4,7);
  plot(5,7);  plot(6,7);  plot(7,7);

  //O
  plot(2,10); plot(2,11); plot(2,12); plot(2,13);
  plot(2,14); plot(2,15); plot(3,10); plot(3,15);
  plot(4,10); plot(4,15); plot(5,10); plot(5,15);
  plot(6,10); plot(6,15); plot(7,10); plot(7,11);
  plot(7,12); plot(7,13); plot(7,14); plot(7,15);

  //S
  plot(2,18); plot(2,19); plot(2,20); plot(2,22);
  plot(3,18); plot(3,20); plot(3,22); plot(4,18);
  plot(4,20); plot(4,22); plot(5,18); plot(5,20);
  plot(5,22); plot(6,18); plot(6,20); plot(6,22);
  plot(7,18); plot(7,20); plot(7,21); plot(7,22);

  //S
  plot(2,26); plot(2,27); plot(2,28); plot(2,30);
  plot(3,26); plot(3,28); plot(3,30); plot(4,26);
  plot(4,28); plot(4,30); plot(5,26); plot(5,28);
  plot(5,30); plot(6,26); plot(6,28); plot(6,30);
  plot(7,26); plot(7,28); plot(7,29); plot(7,30);
}
```

```
void End_interface(){
  plot_display();
  delay(200);
  if(HP == 24){
    for(int i = 0; i < 4; i++){ //Health reach 0
      Lose();
      delay(700);
      clear_display();
      delay(300);
    }
  }
  else if(enemyHP == 24){  //enemyHealth reach 0
    for(int i = 0; i < 4; i++){ //Health reach 0
      Win();
      delay(700);
      clear_display();
      delay(300);
    }
  }
}
```

ฟังก์ชั่นช่วงสุดท้ายจะเป็นการแสดง interface ว่าเราชนะหรือแพ้ โดย

ถ้าเลือดผู้เล่นหมด (HP = 24 หมายถึง ถ้าลบไฟจนถึง y = 24) จอภาพก็จะแสดงคำว่า LOSS

ถ้าเลือดศัตรูหมด (enemyHP = 24 หมายถึง ถ้าลบไฟจนถึง y = 24) จอภาพก้จะแสดงคำว่า WIN

เมื่อแสดงค่า WIN || LOSS เสร็จแล้ว ก็จะส่งค่า main_state = 0 เพื่อกลับไปหน้าหลักของเกม

END OF PROJECT