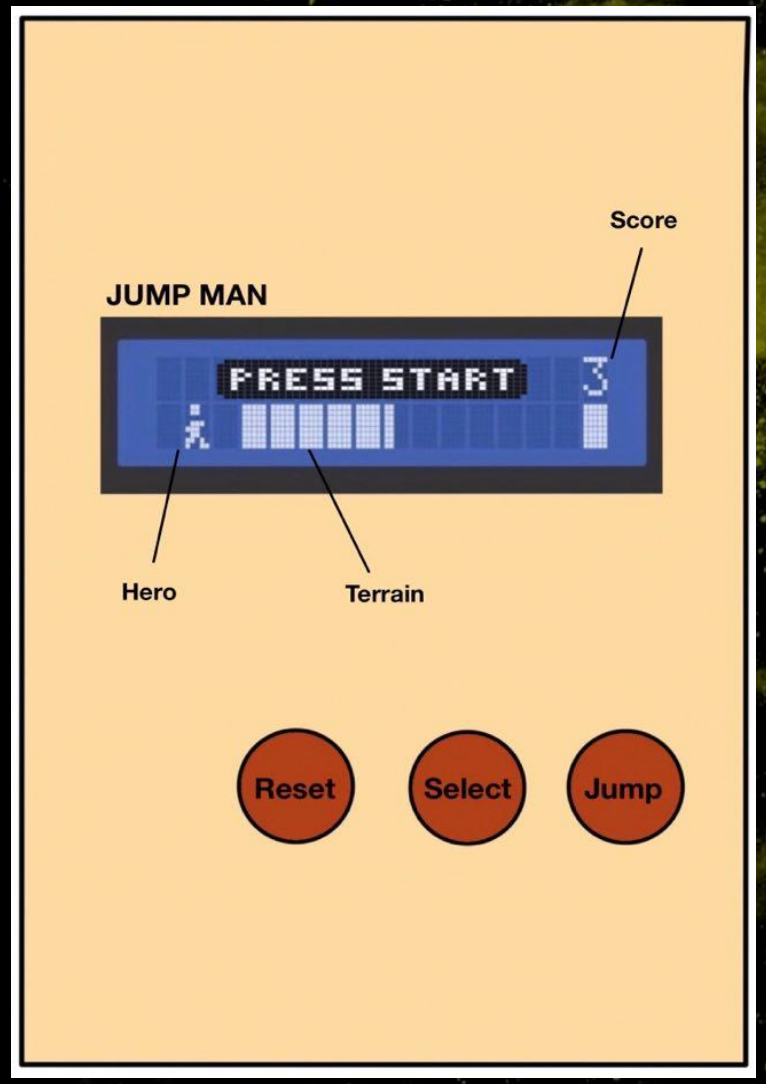

L
R
Jump man
PRESS START
3
A
B
START
SELECT

ที่มาและความสำคัญ

เนื่องจากความชื่นชอบในการเล่นวีดีโอเกม และอยากเรียนรู้วิธีการสร้างเกมแบบง่าย จึงได้เลือกจัดทำเครื่องเกมพกพา ที่เขียนด้วยพื้นฐานภาษา C/C++ รันบนฮาร์ดแวร์ และ ซอฟแวร์ แบบเปิด ที่สามารถเข้าถึงได้ง่าย อย่างบอร์ด Arduino และโปรแกรม Arduino IDE ในการจัดทำเกม Jump man ขึ้น

วิธีการเล่น



ปุ่มที่ใช้ในการเล่นเกมมี 3 ปุ่ม คือ

Jump = ใช้ในการบังคับ Hero เพื่อหลบหลีกสิ่งกีดขวาง

Select = ใช้เลือกโหมดการเล่น มีทั้งหมด 4 โหมด Easy , Normal , Hard , Impossible

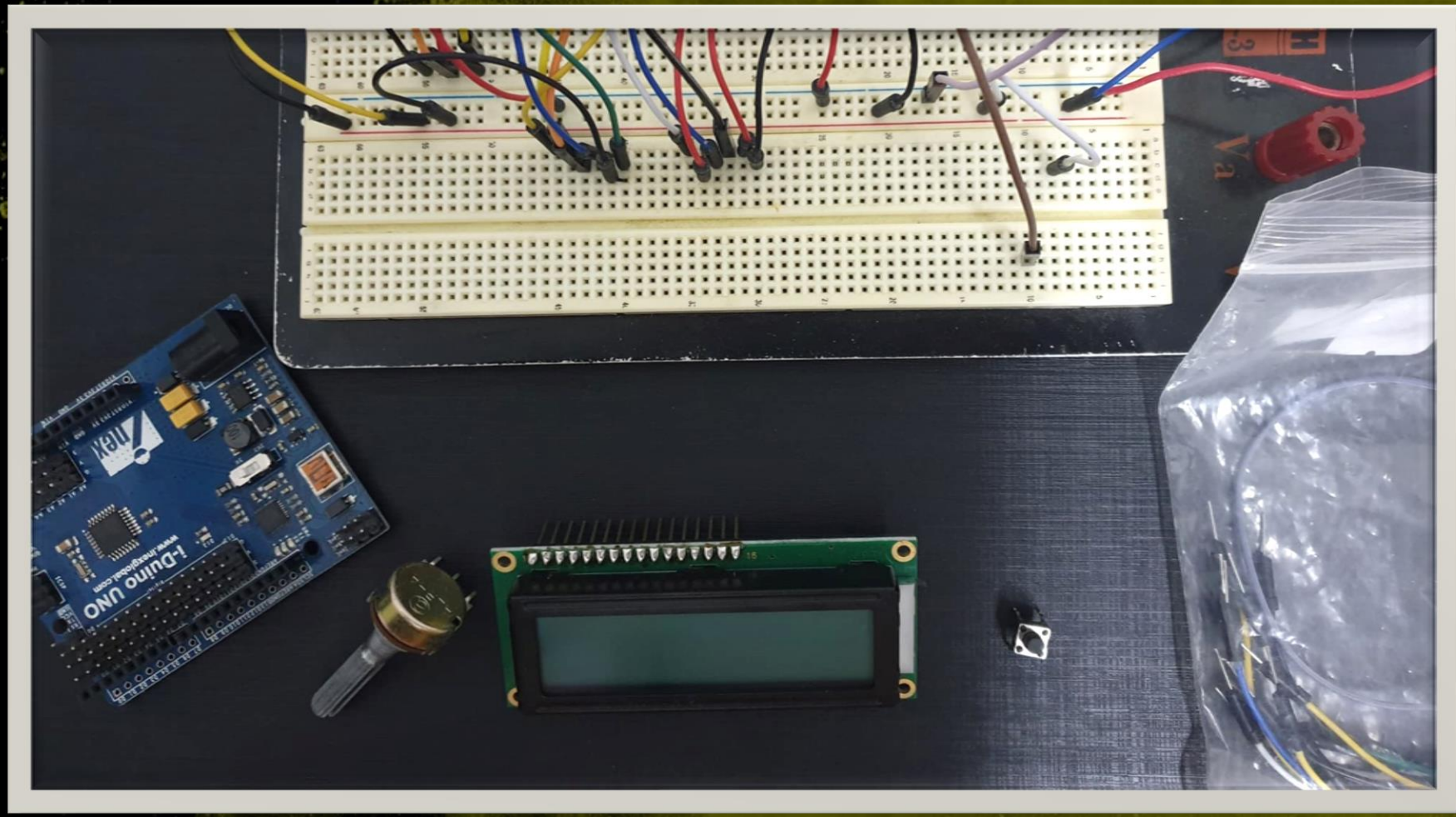
Reset = ใช้เริ่มเกมใหม่

เป็นเกมแนว Side Scrolling ที่วิ่งแบบไร้จุดสิ้นสุด เก็บคะแนนด้วยการนับเวลา ซึ่งเป็นเวลาที่ใช้ในการผ่านสิ่งกีดขวาง โดยการควบคุม Hero กระโดดหลบสิ่งกีดขวางที่เคลื่อนที่เข้ามาแบบสุ่ม สิ่งกีดขวางนั้นจะมีความเร็วตามโหมดที่เลือกไว้ก่อนหน้า หาก Hero ชนเข้ากับสิ่งกีดขวาง Score จะหยุดนับ แล้วเกมจะ Over ทันที

หมายเหตุ ผู้เล่นสามารถเลือกที่จะเล่นใหม่อีกครั้ง โดยการกดปุ่ม Jump หรือเลือกโหมดการเล่นใหม่โดยการกดปุ่ม Reset นั่นเอง

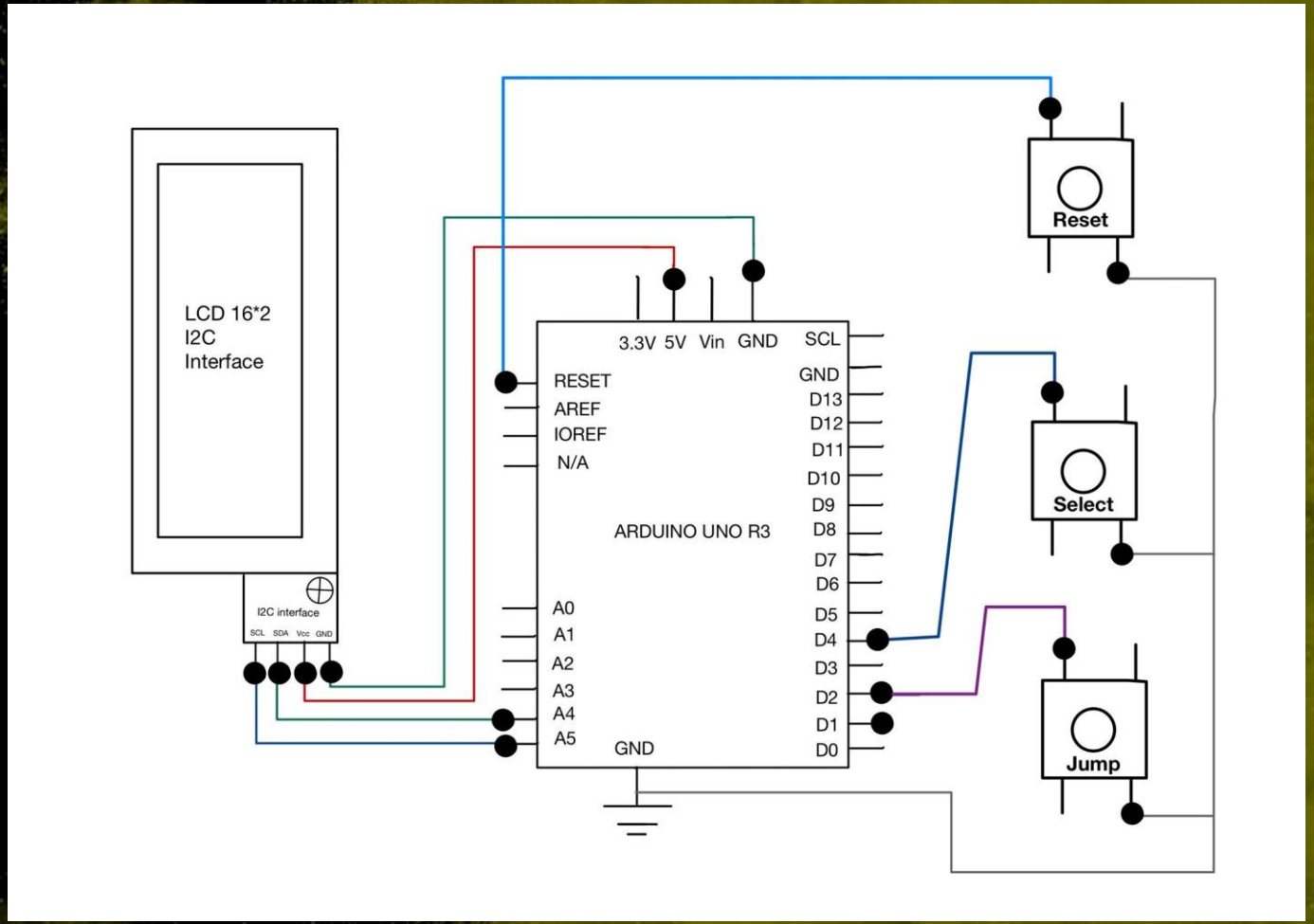
ข้อจำกัด ตัวเกมยังมี bug บางส่วน ของความเร็วของโหมด Impossible ที่ไม่สามารถปรับให้เร็วขึ้นได้อีก ผู้จัดทำคาดว่าปัญหาดังกล่าวมาจากความไวในการตอบสนองของหน้าจอที่ไม่สามารถเพิ่มได้ หรืออาจเป็นที่ Code ที่ไม่มามารถเรียกใช้ฟังก์ชันดังกล่าวได้

อุปกรณ์



(บอร์ด Arduino UNO) (Potentiometer) (LCD 16X2) (Push button) (สายไฟ) (แผงวงจร)

การต่อวงจร


LCD 16*2
I2C
Interface
I2C interface
SCL SDA Vcc GND
3.3V 5V Vin GND
RESET
AREF
IOREF
N/A
ARDUINO UNO R3
A0
A1
A2
A3
A4
A5
GND
SCL
GND
D13
D12
D11
D10
D9
D8
D7
D6
D5
D4
D3
D2
D1
D0
Reset
Select
Jump

Code

```
#include <LiquidCrystal_I2C.h>     // library ของ LCD
volatile int grasw; //ลำดับข้อความ
volatile int napis; //ข้อความ
volatile int podmenu; //เมนูย่อย
#define PIN_BUTTON 2
#define PIN_AUTOPLAY 1
#define PIN_READWRITE 10
#define PIN_CONTRAST 7
#define SPRITE_RUN1 1
#define SPRITE_RUN2 2
#define SPRITE_JUMP 3
#define SPRITE_JUMP_UPPER '.'
#define SPRITE_JUMP_LOWER 4
#define SPRITE_TERRAIN_EMPTY ' '
#define SPRITE_TERRAIN_SOLID 5
#define SPRITE_TERRAIN_SOLID_RIGHT 6
#define SPRITE_TERRAIN_SOLID_LEFT 7
#define HERO_HORIZONTAL_POSITION 1
```

```
#define TERRAIN_WIDTH 16
#define TERRAIN_EMPTY 0
#define TERRAIN_LOWER_BLOCK 1
#define TERRAIN_UPPER_BLOCK 2
#define HERO_POSITION_OFF 0
#define HERO_POSITION_RUN_LOWER_1 1
#define HERO_POSITION_RUN_LOWER_2 2
#define HERO_POSITION_JUMP_1 3
#define HERO_POSITION_JUMP_2 4
#define HERO_POSITION_JUMP_3 5
#define HERO_POSITION_JUMP_4 6
#define HERO_POSITION_JUMP_5 7
#define HERO_POSITION_JUMP_6 8
#define HERO_POSITION_JUMP_7 9
#define HERO_POSITION_JUMP_8 10
#define HERO_POSITION_RUN_UPPER_1 11
#define HERO_POSITION_RUN_UPPER_2 12
```

Code

```
 LiquidCrystal_I2C lcd(0x27, 16, 2);
static char terrainUpper[TERRAIN_WIDTH + 1];
static char terrainLower[TERRAIN_WIDTH + 1];
static bool buttonPushed = false;
char wybor;
char wyborpoz;
 //สร้างชุดเลขฐานสองแสดงกราฟฟิคต่างๆของเกม
void initializeGraphics(){
  static byte graphics[] = {

// Run position 1
   B01100,       //////
   B01100,      //////
   B00000,
   B01110,      //////////
   B11100,     //////////
   B01100,     /////
   B11010,   //////  /////
   B10011,  /////    //////////
```

```
  // Run position 2
   B01100,    ///////
   B01100,    ///////
   B00000,
   B01100,    ///////
   B01100,    ///////
   B01100,    ///////
   B01100,    ///////
   B01110,    //////////////
// Jump
   B01100,    ///////
   B01100,    ///////
   B00000,
   B11110,  ///////////////
   B01101,    //////////      /////
   B11111,  ////////////////////
   B10000,  ////
   B00000,
```

```
// Jump lower
   B11110,  //////////////
   B01101,   /////////      ////
   B11111,  ////////////////////
   B10000,  ////
   B00000,
   B00000,
   B00000,
   B00000,
   B00000,
   // Ground
   B11111,  ////////////////////
   B11111,  ////////////////////
   B11111,  ////////////////////
   B11111,  ////////////////////
   B11111,  ////////////////////
   B11111,  ////////////////////
   B11111,  ////////////////////
   B11111,  ////////////////////
```

```
// Ground right
   B00011,          //
   B00011,         //
   B00011,         //
   B00011,         //
   B00011,         //
   B00011,         //
   B00011,         //
   B00011,         //
   // Ground left
   B11000,  //
   B11000,  //
   B11000,  //
   B11000,  //
   B11000,  //
   B11000,  //
   B11000,  //
   B11000,  //
   B11000,  //
  };
```

Code

```
 int i;  //ฟังก์ชันรันกราฟฟิค
  for (i = 0; i < 7; ++i) {
    lcd.createChar(i + 1, &graphics[i * 8]);
  }
  for (i = 0; i < TERRAIN_WIDTH; ++i) {
    terrainUpper[i] = SPRITE_TERRAIN_EMPTY;
    terrainLower[i] = SPRITE_TERRAIN_EMPTY;
  }
}
void advanceTerrain(char* terrain, byte newTerrain){
  for (int i = 0; i < TERRAIN_WIDTH; ++i) {
    char current = terrain[i];
    char next = (i == TERRAIN_WIDTH-1) ? newTerrain : terrain[i+1];
switch (current){
      case SPRITE_TERRAIN_EMPTY:
        terrain[i] = (next == SPRITE_TERRAIN_SOLID) ?
SPRITE_TERRAIN_SOLID_RIGHT : SPRITE_TERRAIN_EMPTY;
break;
```

```
case SPRITE_TERRAIN_SOLID:
        terrain[i] = (next == SPRITE_TERRAIN_EMPTY) ? SPRITE_TERRAIN_SOLID_LEFT :
SPRITE_TERRAIN_SOLID;
        break;
      case SPRITE_TERRAIN_SOLID_RIGHT:
        terrain[i] = SPRITE_TERRAIN_SOLID;
        break;
      case SPRITE_TERRAIN_SOLID_LEFT:
        terrain[i] = SPRITE_TERRAIN_EMPTY;
        break;
    }
  }
}

void gl (int t){  //กำหนดเวลา delay
delay(t);
}
```

Code

```
bool drawHero(byte position, char* terrainUpper, char* terrainLower, unsigned int score) {
//ระบุตำแหน่งสิ่งกีดขวาง score และ Hero เมื่อเกิดการผ่านสิ่งกีดขวางหรือชน
 bool collide = false;
 char upperSave = terrainUpper[HERO_HORIZONTAL_POSITION];
 char lowerSave = terrainLower[HERO_HORIZONTAL_POSITION];
 byte upper, lower;
 switch (position) {
  case HERO_POSITION_OFF:
   upper = lower = SPRITE_TERRAIN_EMPTY;
   break;
  case HERO_POSITION_RUN_LOWER_1:
   upper = SPRITE_TERRAIN_EMPTY;
   lower = SPRITE_RUN1;
   break;
  case HERO_POSITION_RUN_LOWER_2:
   upper = SPRITE_TERRAIN_EMPTY;
   lower = SPRITE_RUN2;
   break;
  case HERO_POSITION_JUMP_1:
  case HERO_POSITION_JUMP_8:
  upper = SPRITE_TERRAIN_EMPTY;
   lower = SPRITE_JUMP;
   break;
  case HERO_POSITION_JUMP_2:
  case HERO_POSITION_JUMP_7:
  upper = SPRITE_JUMP_UPPER;
  lower = SPRITE_JUMP_LOWER;
  break;
 case HERO_POSITION_JUMP_3:
 case HERO_POSITION_JUMP_4:
 case HERO_POSITION_JUMP_5:
 case HERO_POSITION_JUMP_6:
  upper = SPRITE_JUMP;
  lower = SPRITE_TERRAIN_EMPTY;
  break;
  case HERO_POSITION_RUN_UPPER_1:
  upper = SPRITE_RUN1;
  lower = SPRITE_TERRAIN_EMPTY;
  break;
 case HERO_POSITION_RUN_UPPER_2:
  upper = SPRITE_RUN2;
  lower = SPRITE_TERRAIN_EMPTY;
  break;
}
```

Code

```
//เงื่อนไขการแสดงสิ่งกีดขวาง
  if (upper != ' ') {
    terrainUpper[HERO_HORIZONTAL_POSITION] = upper;
    collide = (upperSave == SPRITE_TERRAIN_EMPTY) ? false : true;
  }
  if (lower != ' ') {
    terrainLower[HERO_HORIZONTAL_POSITION] = lower;
    collide |= (lowerSave == SPRITE_TERRAIN_EMPTY) ? false : true;
  }
  byte digits = (score > 9999) ? 5 : (score > 999) ? 4 : (score > 99) ? 3 : (score > 9) ? 2 : 1;
// Draw the scene
  terrainUpper[TERRAIN_WIDTH] = '\0';
  terrainLower[TERRAIN_WIDTH] = '\0';
  char temp = terrainUpper[16-digits];
  terrainUpper[16-digits] = '\0';
  lcd.setCursor(0,0);
  lcd.print(terrainUpper);
  terrainUpper[16-digits] = temp;
  lcd.setCursor(0,1);
  lcd.print(terrainLower);
  lcd.setCursor(16 - digits,0);
  lcd.print(score);
  terrainUpper[HERO_HORIZONTAL_POSITION] = upperSave;
  terrainLower[HERO_HORIZONTAL_POSITION] = lowerSave;
  return collide;
}
// Handle the button push as an interrupt
 void buttonPush() {
  buttonPushed = true;
}
```

Code

```
void origin(){  //ฟังก์ชั่นแสดงหน้าต่างก่อนเข้าเกม
  lcd.setCursor(0,0);
  lcd.print("Game");
  lcd.setCursor(2,4);
  lcd.print("Jump Man");
  gl(100);
  delay(1000);
  lcd.clear();
}
void game(int level){  //ฟังก์ชั่นกำหนดระดับความยากของเกม และการแสดง
  static byte heroPos = HERO_POSITION_RUN_LOWER_1;
  static byte newTerrainType = TERRAIN_EMPTY;
  static byte newTerrainDuration = 1;
  static bool playing = false;
  static bool blink = false;
  static unsigned int distance = 0;
 if (!playing) {
    drawHero((blink) ? HERO_POSITION_OFF : heroPos, terrainUpper,
terrainLower, distance >> 3);
```

```
  if (blink) {
      lcd.clear();
      lcd.setCursor(0,0);
      lcd.print("PRESS START");
    }
delay(250);
    blink = !blink;
    if (buttonPushed) {
initializeGraphics();
      heroPos = HERO_POSITION_RUN_LOWER_1;
      playing = true;
      buttonPushed = false;
      distance = 0;
    }
    return;
  }
```

Code

```
// random สิ่งกีดขวาง
  // Shift the terrain to the left
  advanceTerrain(terrainLower, newTerrainType == TERRAIN_LOWER_BLOCK ?
SPRITE_TERRAIN_SOLID : SPRITE_TERRAIN_EMPTY);
  advanceTerrain(terrainUpper, newTerrainType == TERRAIN_UPPER_BLOCK ?
SPRITE_TERRAIN_SOLID : SPRITE_TERRAIN_EMPTY);
// Make new terrain to enter on the right
  if (--newTerrainDuration == 0) {
    if (newTerrainType == TERRAIN_EMPTY) {
      newTerrainType = (random(3) == 0) ? TERRAIN_UPPER_BLOCK :
TERRAIN_LOWER_BLOCK;
      newTerrainDuration = 2 + random(10);
    } else {
      newTerrainType = TERRAIN_EMPTY;
      newTerrainDuration = 10 + random(10);
    }
  }
 if (buttonPushed) {
 if (heroPos <= HERO_POSITION_RUN_LOWER_2)
// HERO speed
```

```
heroPos = HERO_POSITION_JUMP_1;
    buttonPushed = false;
  }
 if (drawHero(heroPos, terrainUpper, terrainLower, distance >> 3)) {
    playing = false; // The hero collided with something. Too bad.
    gl(500);
    lcd.clear();
  } else {
    if (heroPos == HERO_POSITION_RUN_LOWER_2 || heroPos ==
HERO_POSITION_JUMP_8) {
      heroPos = HERO_POSITION_RUN_LOWER_1;
    } else if ((heroPos >= HERO_POSITION_JUMP_3 && heroPos <=
HERO_POSITION_JUMP_5) && terrainLower[HERO_HORIZONTAL_POSITION] !=
SPRITE_TERRAIN_EMPTY) {
      heroPos = HERO_POSITION_RUN_UPPER_1;
    } else if (heroPos >= HERO_POSITION_RUN_UPPER_1 &&
terrainLower[HERO_HORIZONTAL_POSITION] == SPRITE_TERRAIN_EMPTY) {
      heroPos = HERO_POSITION_JUMP_5;
 } else if (heroPos == HERO_POSITION_RUN_UPPER_2) {
      heroPos = HERO_POSITION_RUN_UPPER_1;
```

Code

```
} else {
    ++heroPos;
  }
  ++distance;
  digitalWrite(PIN_AUTOPLAY, terrainLower[HERO_HORIZONTAL_POSITION + 2] ==
SPRITE_TERRAIN_EMPTY ? HIGH : LOW);
 }
 delay(level);
 }
void setup(){
 pinMode(4, INPUT_PULLUP);
 pinMode(PIN_READWRITE, OUTPUT);
 digitalWrite(PIN_READWRITE, LOW);
 pinMode(PIN_CONTRAST, OUTPUT);
 digitalWrite(PIN_CONTRAST, LOW);
 pinMode(PIN_BUTTON, INPUT);
 digitalWrite(PIN_BUTTON, HIGH);
 pinMode(PIN_AUTOPLAY, OUTPUT);
 digitalWrite(PIN_AUTOPLAY, HIGH);
 napis = 1;
attachInterrupt(0/*PIN_BUTTON*/, buttonPush, FALLING);
```

```
initializeGraphics();
lcd.begin();
  origin();
  do{menu();}while(digitalRead(2) == HIGH);
}
void loop() {
switch (grasw) {
case 1:
game(30); //speed Easy
break;
case 2:
game(10); //speed medium
break;
case 3:
game(5); //speed Hard
break;
case 4:
game(1); //impossible
break;
   }
 }
```

```
 void menu() {
if (napis > 1 || napis < 0) {napis == 0;}
if (digitalRead(4) == LOW) { napis++;}
 switch (napis) {
  case 1:
 lcd.setCursor(0,0);
 lcd.print("Select mode:");
 lcd.setCursor(0,1);
 lcd.print("<    Easy    >");
 delay(500);
 lcd.clear();
 lcd.setCursor(0,0);
 lcd.print("Select mode:");
 lcd.setCursor(0,1);
 lcd.print("<            >");
 delay(500);
 if(digitalRead(2) == LOW) {
  grasw = 1;
  grasw;}
break;
```

Code

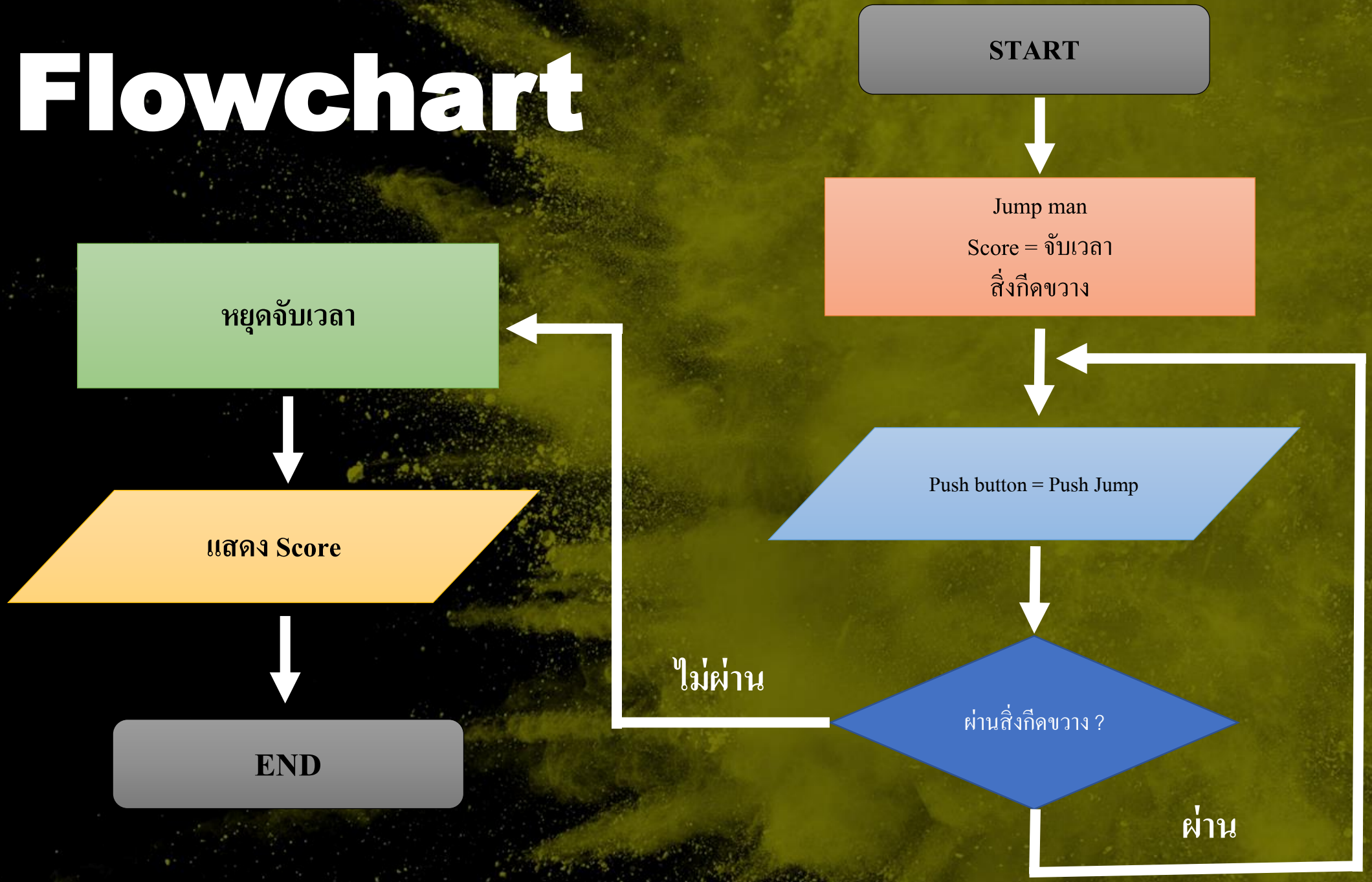
```
case 2:
  lcd.setCursor(0,0);
  lcd.print("Select mode:");
  lcd.setCursor(0,1);
  lcd.print("<    Medium    >");
  delay(500);
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Select mode:");
  lcd.setCursor(0,1);
  lcd.print("<              >");
  delay(500);
if(digitalRead(2) == LOW) {
    grasw = 2;
    grasw;}
break;
```

```
 case 3:
lcd.setCursor(0,0);
  lcd.print("Select mode:");
  lcd.setCursor(0,1);
  lcd.print("<     Hard     >");
  delay(500);
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Select mode:");
  lcd.setCursor(0,1);
  lcd.print("<              >");
  delay(500);
  if(digitalRead(2) == LOW) {
    grasw = 3;
    grasw;}
break;
```

```
case 4:
lcd.setCursor(0,0);
  lcd.print("Select mode:");
  lcd.setCursor(0,1);
  lcd.print("< impossible >");
  delay(500);
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Select mode:");
  lcd.setCursor(0,1);
  lcd.print("<              >");
  delay(500);
  if(digitalRead(2) == LOW) {
    grasw = 4;
    grasw;}
 break;
```

```
default:
napis = 0;
  break;
    }
  }
```

Flowchart


START
Jump man
Score = จับเวลา
สิ่งกีดขวาง
Push button = Push Jump
ผ่านสิ่งกีดขวาง ?
ไม่ผ่าน
ผ่าน
หยุดจับเวลา
แสดง Score
END