

Part 4

Lesson

2

**Project Course:
Greedy Snake**

Overview

In this lesson, we will use the MAX7219 display module and MPU6050 module to build a gravity-sensing control greedy snake game. By tilting the circuit board from side to side, we can control the movement of the snake and find the food.

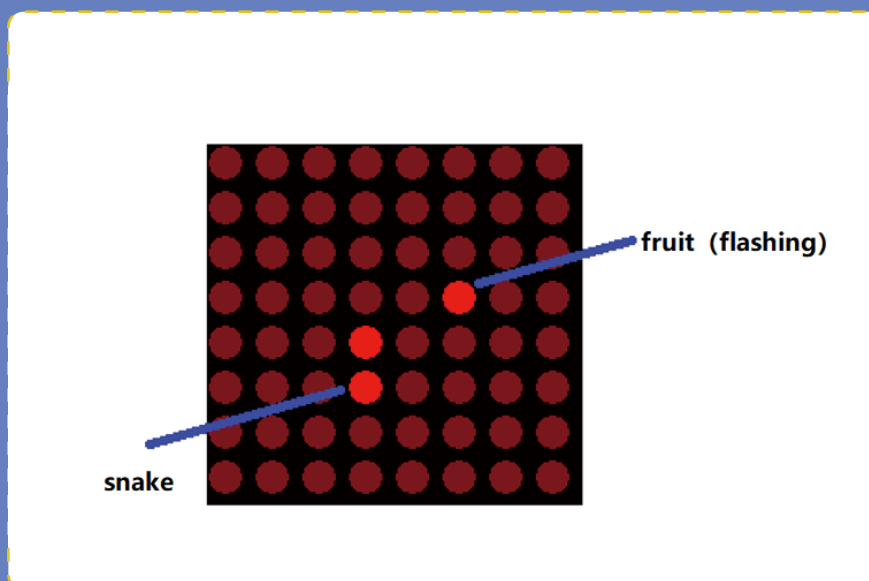
Component Required:

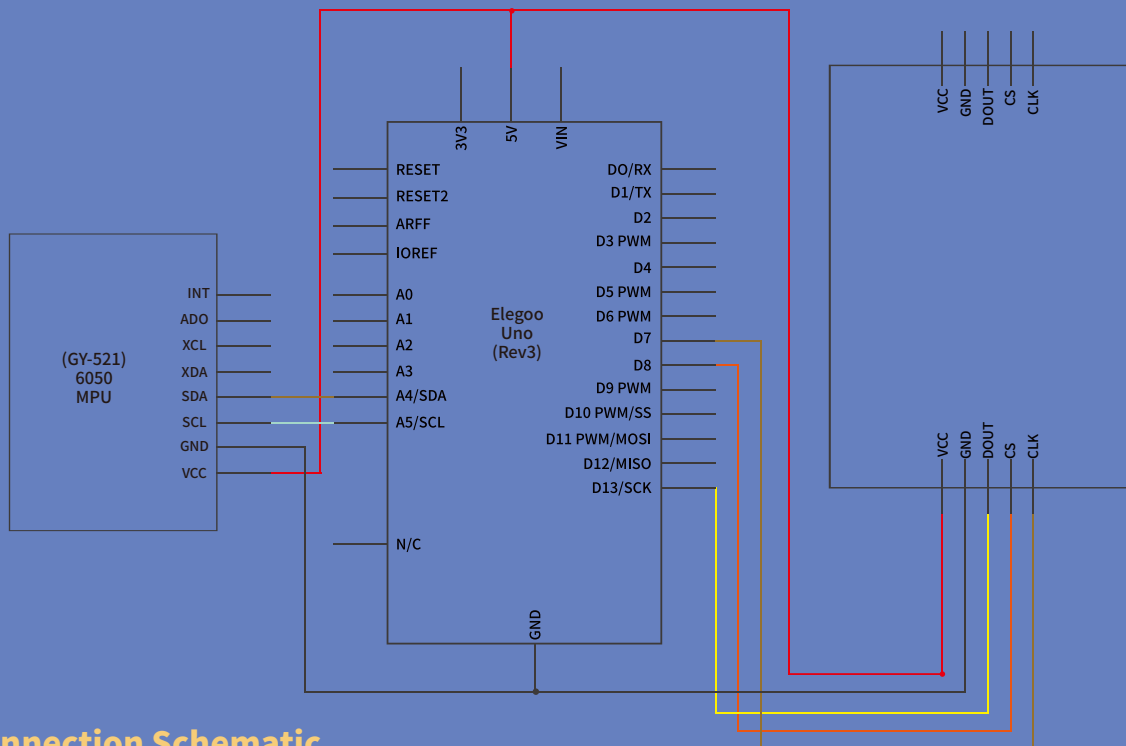
- (1) X Elegoo UNO R3
- (1) X GY521
- (1) X MAX7219
- (1) X ALL IN ONE Sensor Shield



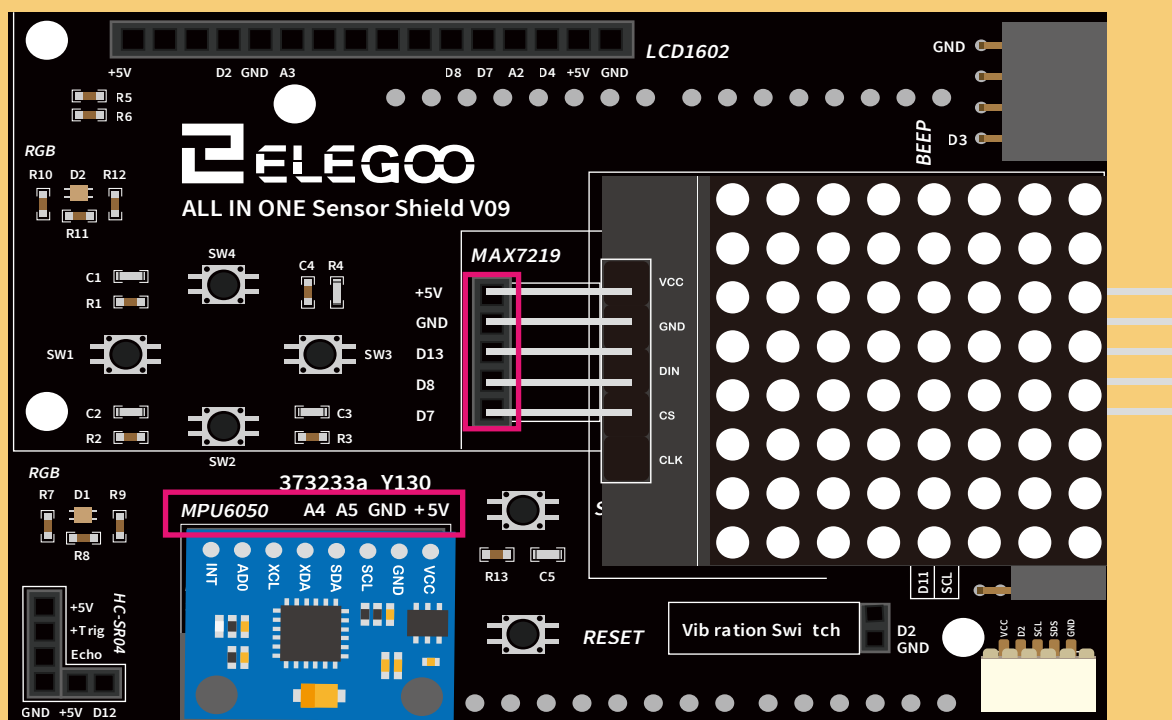
Game Introduction:

The snake keeps moving, gravity sensing to change the direction of the snake's progress. When the snake eats the fruit, the snake body grows and the snake moves faster. The game ends when the snake touches itself.







Tips: Please insert the extended board into UNO.



Code:

Make sure you add libraries before downloading the program

 LedControl.zip	2019/8/14 16:06	9 KB
 Wire.zip	2019/8/14 16:10	17 KB

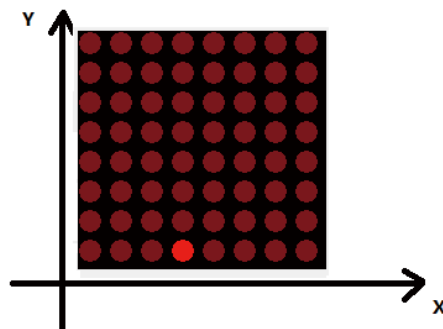
Part1: Control snake movement

1.Snake display

```
int snakeX[64];      // X-coordinates of snake
int snakeY[64];      // Y-coordinates of snake
snakeX[0] = 0;
snakeY[0] = 4;
int snakeLength = 2;
void drawSnake(){
  for(int i=0; i<snakeLength; i++){
    /* Set the status of a single Led.
    * Params :
    * addr  address of the display
    * row   the row of the Led (0..7)
    * col   the column of the Led (0..7)
    * state If true the led is switched on, if false it is switched off

    setLed(int addr, int row, int col, boolean state);
    */
    lc.setLed(0,snakeX[i], snakeY[i], true);
  }
}
```

Create two arrays to represent the position of the snake in the 8*8 dot matrix, snakeX [0], snakeY [0] indicates the position of the first section of the snake. snakeX[1], snakeY[1] indicates the position of the second section of the snake. snakeX[2], snakeY[2] indicates the position of the third section of the snake.



2.The movement of snakes.

First, the snake's nodes are moved backward by one through the for loop.

E.g:snakeLength=3: The third part of the snake is replaced by the second part of the snake,The second part of the snake is replaced by the first part of the snake, The first part is empty after the for loop.

Then use the switch function to select the direction of the first part.

3.Get the snake next direction command

Change the direction command when gy521 is in different directions. Then pass to the switch in step 2 to complete the snake direction change.

```
void nextstep(){
  for(int i=snakeLength-1; i>0; i--){
    snakeX[i] = snakeX[i-1];
    snakeY[i] = snakeY[i-1];
  }

  switch(direction){
    case UP:
      snakeY[0] = snakeY[0]-1;
      if(snakeY[0]==-1)
        snakeY[0]=7;
      break;
    case RIGHT:
      snakeX[0] = snakeX[0]+1;
      if(snakeX[0]==8)
        snakeX[0]=0;
      break;
    case DOWN:
      snakeY[0] = snakeY[0]+1;
      if(snakeY[0]==8)
        snakeY[0]=0;
      break;
    case LEFT:
      snakeX[0]=snakeX[0]-1;
      if(snakeX[0]==-1)
        snakeX[0]=7;
      break;
  }
}
```

```
void checkGy521(){
static unsigned long delaytime = millis();
if(millis() - delaytime > 20){
  delaytime = millis();
  Wire.beginTransmission(MPU_addr);
  Wire.write(0x3B); // starting with register 0x3B (ACCEL_XOUT_H)
  Wire.endTransmission(false);
  Wire.requestFrom(MPU_addr,14,true); // request a total of 14 registers
  AcX=Wire.read()<<8|Wire.read(); // 0x3B (ACCEL_XOUT_H) & 0x3C (ACCEL_XOUT_L)
  AcY=Wire.read()<<8|Wire.read(); // 0x3D (ACCEL_YOUT_H) & 0x3E (ACCEL_YOUT_L)
  delay(33);
  Serial.print(AcX);
  Serial.print('\t');
  Serial.println(AcY);
  if(AcX>-1000&&AcX<1000&&AcY>8000&&direction!= DOWN){
    direction=UP;
    Serial.println("DOWN");
  }
  else if(AcX>-1000&&AcX<1000&&AcY<-8000&& direction!= UP){
    direction=DOWN;
    Serial.println("UP");
  }
  else if(AcX<-8000&& AcY>-1000&&AcY<1000&& direction!= LEFT){
    direction=RIGHT;
    Serial.println("LEFT");
  }
  else if(AcX>8000&&AcY>-1000&&AcY<1000&& direction!= RIGHT){
    direction=LEFT;
    Serial.println("RIGHT");
  }
}
}
```

Part2

Random Fruit

1.Fruit display

Fruit display is the same as snake display. However, adding a judgment on the time in the function causes the fruit to blink. Distinguish from snakes.

```
void drawFruit(){
  if(inPlayField(fruitX, fruitY)){
    unsigned long currenttime = millis();
    if(currenttime - fruitPrevTime >= fruitBlinkTime){
      /*
       *Fruit keeps flashing
       */
      fruitLed = (fruitLed == true) ? false : true;
      fruitPrevTime = currenttime;
    }
    lc.setLed(0,fruitX, fruitY, fruitLed);
  }
}
```

2.Random fruit

Use the random function to randomly obtain the fruit position, and the fruit cannot appear on the snake.

```
void makeFruit(){
  int x, y;
  x = random(0, 8);
  y = random(0, 8);
  while(isPartOfSnake(x, y)){
    x = random(0, 8);
    y = random(0, 8);
  }
  fruitX = x;
  fruitY = y;
}
```

Part3

Eat Fruit

When the snake moves and eats the fruit, its length will increase, and as the length increases, the speed is gradually increasing.

```
if((snakeX[0] == fruitX) && (snakeY[0] == fruitY)){
  snakeLength++;
  score++;
  if(snakeLength < MAX_SNAKE_LENGTH){
    makeFruit();
  }
  else {
    fruitX = fruitY = -1;
  }
  if(score%8==0)
  {
    delayTime = delayTime - 50;
  }
}
```

Part4

Touches Itself

Judging Condition: Check whether the first node of the snake coincides with the other node of the snake. If the snake hits itself, restart the game.

```
void snakeItSelf(){ // check if snake touches itself
    for(int i=1;i<snakeLength;i++){
        if((snakeX[0] == snakeX[i] && (snakeY[0] == snakeY[i]))){
            gameOver();
        }
    }
}

void gameOver(){
    lc.clearDisplay(0);
    for(int r = 0; r < 8; r++){
        for(int c = 0; c < 8; c++){
            lc.setLed(0, r, c, HIGH);
            delay(50);
        }delay(50);
    }
    delay(300);
    score = 3;
    snakeLength = 3;
    direction = RIGHT;
    snakeX[0]=3;
    snakeY[0]=4;
    delayTime = 500;
    loop();
}
```