Digital Electronics - Project

Topic: Arduino based Snake Game

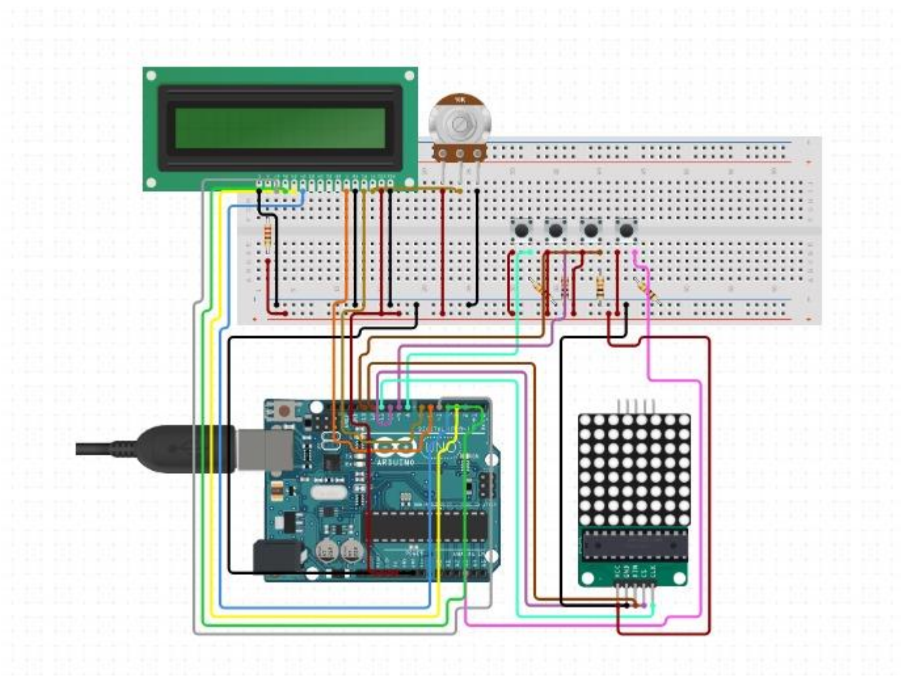
Introduction:

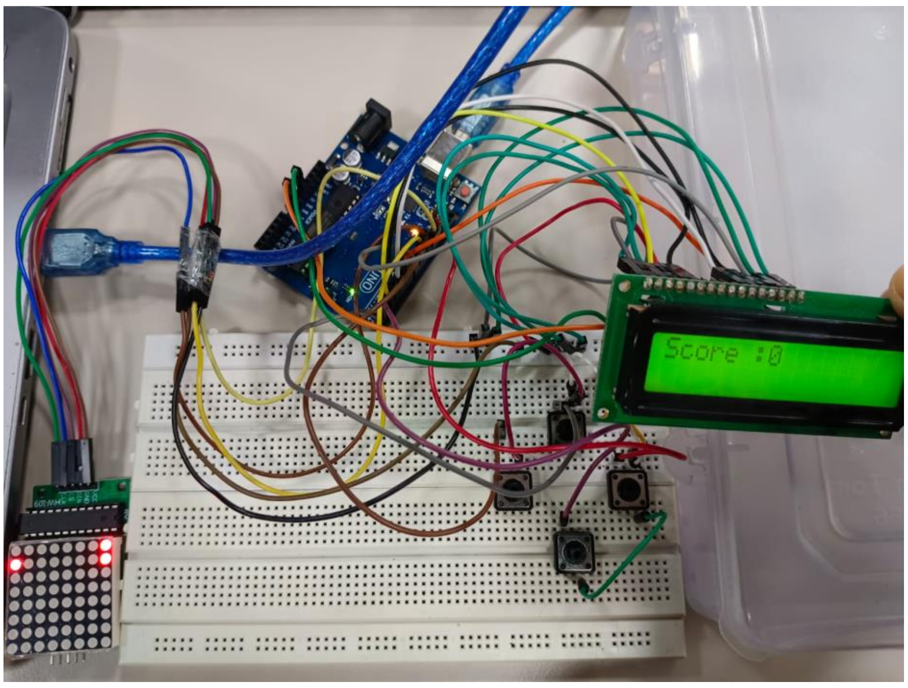
- ➢ Discuss the purpose of implementing the Snake game in digital electronics as a hardware project.

Hardware Components:

- Breadboard
- Arduino
- Display
- Buttons
- Jumper wires
- Printer wire
- Power Supply

Circuit Design Photo:





Code :

```
#include <LedControl.h>
#include <LiquidCrystal.h>

class RedDots {
public:
  int x;
  int y;
};

class Node {
public:
  int x, y;
  Node *next;
  Node *prev;
  Node() {
    x = y = 0;
    next = NULL;
    prev = NULL;
  }
  Node(int x, int y) {
    this->x = x;
    this->y = y;
    this->next = NULL;
    this->prev = NULL;
  }
};

class Linkedlist {
public:
  Node *head;
  Node *tail;
  Linkedlist() {
    head = NULL;
    tail = NULL;
  }
  void add(int, int);
  void addlast(int, int);
  void remove();
};

void Linkedlist::add(int x, int y) {
  Node *new_node = new Node(x, y);
  new_node->next = head;
  new_node->prev = NULL;

  if (head != NULL) {
    head->prev = new_node;
    head = new_node;
  } else {
    head = new_node;
    tail = new_node;
  }
}

void Linkedlist::addlast(int x, int y) {
  Node *new_node = new Node(x, y);
  new_node->next = NULL;
  new_node->prev = tail;
  tail->next = new_node;
  tail = tail->next;
}

void Linkedlist::remove() {
  tail = tail->prev;
  Node *tmp = tail->next;
  tail->next = tail->next->x = tail->next->y = tail->next->next = NULL;
  free(tmp);
}


int DIN = 12;
int CS = 11;
int CLK = 10;
int upButton = 0;
int downButton = 1;
int leftButton = 9;
int rightButton = 13;

int start = 0;
int direction = 2;
int score = 0;

LedControl lc = LedControl(DIN, CLK, CS, 1);

LiquidCrystal lcd(3, 4, 5, 6, 7, 8);

Linkedlist *Snake = new Linkedlist();
RedDots *reddot = new RedDots();

void setup() {
  lcd.begin(16, 2);
  lcd.print("TAP TO START");

  reddot->x = (int)random(0, 8);
  reddot->y = (int)random(0, 8);
  lc.setLed(0, reddot->x, reddot->y, true);

  Snake->add(0, 0);
  Snake->add(0, 1);
  lc.setLed(0, 0, 0, true);
  lc.setLed(0, 0, 1, true);

  pinMode(upButton, INPUT_PULLUP);
  pinMode(downButton, INPUT_PULLUP);
  pinMode(leftButton, INPUT_PULLUP);
  pinMode(rightButton, INPUT_PULLUP);

  lc.shutdown(0, false);
  lc.setIntensity(0, 8);
}

void loop() {
restart:
  start = max(start, !digitalRead(upButton));
  if (start) {
    if (collision()) {
      lcd.print("GAME OVER..!!");
      lcd.setCursor(0, 1);
      lcd.print("YOUR SCORE:");
      lcd.setCursor(13, 1);
      lcd.print(score);
      byte GameOver[8] = { B11111111, B11111111, B11111111, B11111111,
B11111111,B11111111, B11111111, B11111111 };
      for (int i = 0; i < 8; i++) {
        lc.setRow(0, i, GameOver[i]);
      }
      delay(2500);
      lc.clearDisplay(0);
      lcd.clear();
      lcd.print("TAP TO START");
      reset();
      score = 0;
      start = 0;
      goto restart;
    }

    redDotCheck();

    lcd.print("Score : ");
    lcd.setCursor(7, 0);
    lcd.print(score);
    lc.setLed(0, reddot->x, reddot->y, true);

    Node *body = Snake->head;
    while (body != NULL) {
      lc.setLed(0, body->x, body->y, true);
      body = body->next;
    }

    Buttons();

    delay(250);
    lc.clearDisplay(0);
    lcd.clear();
  }
}

void redDotCheck() {
  if ((Snake->head->x == reddot->x) && (Snake->head->y == reddot->y)) {
    lc.setLed(0, reddot->x, reddot->y, false);
    reddot->x = (int)random(0, 8);
    reddot->y = (int)random(0, 8);
    Snake->addlast(Snake->tail->x, Snake->tail->y);
    score++;
  }
}

bool collision() {
  Node *tmp = Snake->head->next;
  int chk_x = Snake->head->x, chk_y = Snake->head->y;
  while (tmp != NULL) {
    if (tmp->x == chk_x && tmp->y == chk_y) {
      return 1;
    }
    tmp = tmp->next;
  }
  return 0;
}
void reset() {
    while (Snake->head != NULL) {
    Node *tmp = Snake->head->next;
    Node *clr = Snake->head;
    Snake->head->x = Snake->head->y = Snake->head->prev = NULL;
    Snake->head = tmp;
    free(clr);
  }
  Snake->tail = NULL;
  direction = 2;

  reddot->x = (int)rand() % 8;
  reddot->y = (int)rand() % 8;
  lc.setLed(0, reddot->x, reddot->y, true);

  Snake->add(0, 0);
  Snake->add(0, 1);
  lc.setLed(0, 0, 0, true);
  lc.setLed(0, 0, 1, true);

  lc.shutdown(0, false);
  lc.setIntensity(0, 8);
}

void Buttons() {
  if (!digitalRead(downButton)) {
    if (direction != 1) {
      direction = 2; // down
    }
  } else if (!digitalRead(upButton)) {
    if (direction != 2) {
      direction = 1; // up
    }
  } else if (!digitalRead(leftButton)) {
```

```
    if (direction != 4) {
      direction = 3; // left
    }
  } else if (!digitalRead(rightButton)) {
    if (direction != 3) {
      direction = 4; // right
    }
  }
  chooseDirection(direction);
}

void chooseDirection(int data) {
  if (data == 1) {
    moveUp();
  } else if (data == 2) {
    moveDown();
  } else if (data == 3) {
    moveLeft();
  } else {
    moveRight();
  }
}

void moveDown() {
  int x = Snake->head->x;
  int y = Snake->head->y + 1;
  y %= 8;
  Snake->add(x, y);
  Snake->remove();
}

void moveLeft() {
  int x = Snake->head->x + 1;
  int y = Snake->head->y;
  x %= 8;
  Snake->add(x, y);
  Snake->remove();
}

void moveRight() {
  int x = Snake->head->x + 7;
  int y = Snake->head->y;
  x %= 8;
  Snake->add(x, y);
  Snake->remove();
}
s
void moveUp() {
  int x = Snake->head->x;
  int y = Snake->head->y + 7;
  y %= 8;
  Snake->add(x, y);
  Snake->remove();
}
```