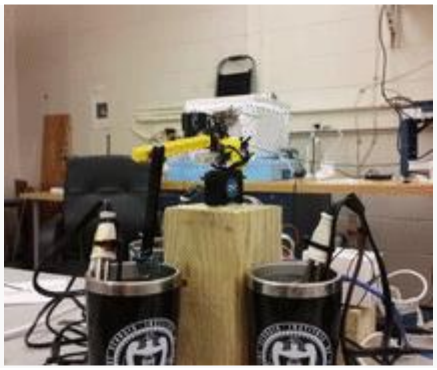
Module 3: Biomolecules

Description: By employing techniques from statistical mechanics and nonlinear dynamics, we are able to study the underlying mechanisms of how biomolecules move and interact to perform cellular processes. Processes such as transcription, diffusion, and other molecular interactions are explored by studying energy landscapes of given molecules as well as running molecular dynamics (MD) simulations of biomolecules in specific landscapes.

DIY PCR machine

Goal: Construct a simple PCR machine with three water baths, amplify DNA with your PCR machine, and perform gel electrophoresis to confirm DNA amplification.



Materials

- Arduino uno (2)
- jumper cables
- solid state relay (SSR) (3)
- immersion heater (3)
- extension cord (3)
- waterproof temperature sensor, DS18B20 (3)
- high temperature thermometer for calibration
- thermos (3)
- breadboard (3)

- servo motors (2)
- LED *(optional)*
- LEGOs
- scrap wood (for stand)
- microcentrifuge tubes
- 5-50uL micropipette
- 5X loading dye (Lonza)
- 100 bp DNA marker (Lonza)
- DNA cocktail (dependent on DNA being amplified)
- wire

Procedure

1. Stack servos such that the bottom one rotates horizontally and the top one rotates vertically.
2. Build a stand for the servo motors.
3. Place water baths around the stand and determine angular positions.
4. Use the servo circuit below to connect the servos to the Arduino.
5. With the servo code, calibrate the water bath locations and servo control.
   a. The time constants can be shortened during testing.
6. Once the servos are working, use the heater circuit below to connect the water baths, temperature sensors and temperature sensors to the second Arduino.
7. Then with the heater code, test that all three water baths can maintain a relatively constant temperature using the high temperature thermometer.
8. Finally, reset the time constants (if necessary) and run both codes (one on each Arduino) to amplify DNA.

Servo code:

```
#include <Servo.h>

Servo blue; // left / right servo
Servo yellow; // up / down servo

int LED = 13; // LED on pin 13

// time constants
long int firstT = 1200; // 2 mins in bath (at beginning)
long int downT = 30000;      // 30s in each bath
long int lastT = 3000; // 5 mins in bath (at end)
long int endT = 6000; // long hold time (after amplifying)

// water bath angular positions
int bath_92 = 10;
int bath_58 = 90;
int bath_72 = 180;
```

```
// arm angular position
int up = 0;
int down = 90;

void setup() {

  Serial.begin(9600);
  pinMode(LED, OUTPUT);

  blue.attach(9);
  yellow.attach(11);

}

void loop() {
   int first = 1;
   int count;
   int flash = 0;

   yellow.write(0);
   delay(500);
   blue.write(0);
   Serial.println("ready");
   delay(1000);

   for (int i = 0; i = 35; i++) {
       if (first == 1) {
       first = 0;
       count = 0;
       blue.write(bath_92);  // Turn Servo Left to 45 degrees (92 bath)
       delay(500);
       yellow.write(down);
       Serial.println("wait 2 mins; 92");
       delay(firstT*100);  // wait 2 mins
    }
    else {
      blue.write(bath_92);  // Turn Servo to (92 bath)
      delay(500);
      yellow.write(down);
      count++;
      digitalWrite(LED, HIGH);
      Serial.println("\n");
      Serial.println(count);
      Serial.println("wait 30s; 92");
      delay(downT);
      digitalWrite(LED, LOW);
      yellow.write(up);
      delay(500);

      blue.write(bath_58);  // Turn Servo to  (58 bath)
      delay(500);
      yellow.write(down);
      digitalWrite(LED, HIGH);
      Serial.println("wait 30s; 58");
      delay(downT);
      digitalWrite(LED, LOW);
      yellow.write(up);
      delay(500);

      blue.write(bath_72); // turn servo to 72 bath
      delay(500);
      yellow.write(down);
      digitalWrite(LED, HIGH);
      Serial.println("wait 30s; 72");
      delay(downT);
      digitalWrite(LED, LOW);
      yellow.write(up);
      delay(500);

      if (count == 35) {
        yellow.write(down);
        digitalWrite(LED, HIGH);
        Serial.println("");
        Serial.println("wait 5 mins; 72");
        delay(lastT*100);
        yellow.write(up);
        delay(500);
        blue.write(0);
        digitalWrite(LED, LOW);
        Serial.println("done!");
```

```
          for (int j = 0; j = 4; j++) {
          flash++;
          digitalWrite(LED, HIGH);
          delay(1000);
          digitalWrite(LED, LOW);
          delay(1000);
          Serial.println("turn me off!");
          if (flash == 4) {
            break;
            //delay(endT);
          }
          }
          digitalWrite(LED, HIGH);
          delay(endT);
          break;
        }
      }
    }

    digitalWrite(LED, HIGH);
    delay(endT*100);

    Serial.println("turn me off!");
    digitalWrite(LED, LOW);
    delay(1000);
    digitalWrite(LED, HIGH);
    delay(1000);
    Serial.println("turn me off!");
    delay(endT);

    digitalWrite(LED, LOW);
    delay(1000);
    digitalWrite(LED, HIGH);
    delay(1000);
    Serial.println("turn me off!");
    delay(endT);
}
```

Heater code:

```
#include <OneWire.h>

int DS18S20_58 = 2; //DS18S20 Signal pin on digital 12
```

```
int DS18S20_72 = 13;
int DS18S20_92 = 12;
int relay_58 = 4;
int relay_72 = 8;
int relay_92 = 7;

float gain_high=80;
float gain_low=0.5;

long t58_old = 0;
long t72_old = 0;
long t92_old = 0;

//Temperature chip i/o
OneWire ds(DS18S20_58);  // on digital pin 2
OneWire ds72(DS18S20_72); // on pin 12
OneWire ds92(DS18S20_92); // on pin 13

void setup() {
  Serial.begin(9600);
  pinMode(relay_58, OUTPUT);
  pinMode(relay_72, OUTPUT);
  pinMode(relay_92, OUTPUT);

}

void loop() {

  // control for 58
  long t58_new = millis();
  float temp58 = getTemp58();

  //Serial.println(temp58);

  float diff_58=58-temp58;
  long wait58 = gain_high*diff_58;
```

```
if (t58_new - t58_old >= wait58){
  t58_old = t58_new;

if (diff_58 < 0){
  digitalWrite(relay_58, LOW);
  //digitalWrite(LED, LOW);

  Serial.println(diff_58);
  Serial.println(temp58);
  Serial.println("58 Off");
  Serial.println("\n");
}
else{
  while(diff_58 > 0){
    float temp58 = getTemp58();

    float diff_58=58-temp58;
    long wait58 = gain_low*abs(diff_58);

    digitalWrite(relay_58, HIGH);
    Serial.println(diff_58);
    Serial.println(temp58);
    Serial.println("58 on");
    Serial.println("\n");

    delay(wait58);
    if(diff_58 < 0){
      digitalWrite(relay_58, LOW);
      Serial.println("58 off / break");
      Serial.println("\n");
      break;
    }
    //digitalWrite(relay_58, LOW);
    //Serial.println("58 off");
    //Serial.println("\n");
  }
}
}
```

```
// control for 72
long t72_new = millis();
float temp72 = getTemp72();

float diff_72=72-temp72;
long wait72 = gain_high*diff_72;

if (t72_new - t72_old >= wait72){
  t72_old = t72_new;

if (diff_72 < 0){
  digitalWrite(relay_72, LOW);
  //digitalWrite(LED, LOW);

  Serial.println(diff_72);
  Serial.println(temp72);
  Serial.println("72 Off");
  Serial.println("\n");
}
else{
  while(diff_72 > 0){
    float temp72 = getTemp72();

    float diff_72=72-temp72;
    long wait72 = gain_low*abs(diff_72);

    digitalWrite(relay_72, HIGH);
    Serial.println(diff_72);
    Serial.println(temp72);
    Serial.println("72 on");
    Serial.println("\n");

    delay(wait72);
    if(diff_72 < 0){
      digitalWrite(relay_72, LOW);
      Serial.println("72 off / break");
      Serial.println("\n");
      break;
    }
  }
```

```
}
}

  // control for 92
long t92_new = millis();
float temp92 = getTemp92();

float diff_92=92-temp92;
long wait92 = gain_high*diff_92;

if (t92_new - t92_old >= wait92){
  t92_old = t92_new;

if (diff_92 < 0){
  digitalWrite(relay_92, LOW);
  //digitalWrite(LED, LOW);

  Serial.println(diff_92);
  Serial.println(temp92);
  Serial.println("92 Off");
  Serial.println("\n");
}
else{
  while(diff_92 > 0){
    float temp92 = getTemp92();

    float diff_92=92-temp92;
    long wait92 = gain_low*abs(diff_92);

    digitalWrite(relay_92, HIGH);
    Serial.println(diff_92);
    Serial.println(temp92);
    Serial.println("92 on");
    Serial.println("\n");

    delay(wait92);
    if(diff_92 < 0){
      digitalWrite(relay_92, LOW);
      Serial.println("92 off / break");
      Serial.println("\n");
```

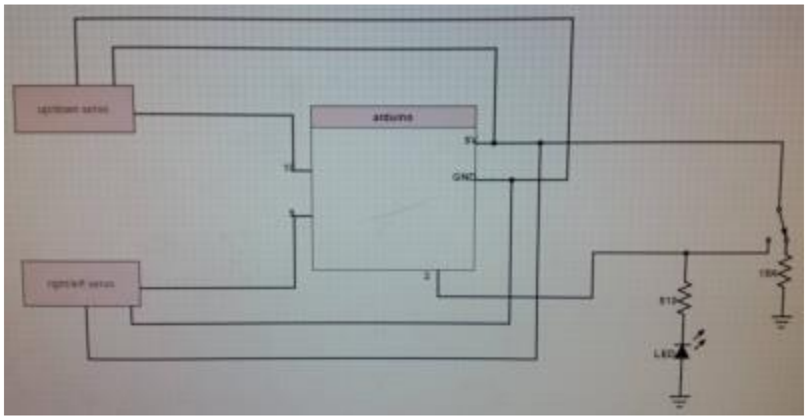
```
        break;
       }

     }
   }
   }
 }

 float getTemp58(){
   //returns the temperature from one DS18S20 in DEG Celsius

   byte data[12];
   byte addr[8];

   if ( !ds.search(addr)) {
       //no more sensors on chain, reset search
       ds.reset_search();
       return -1000;
   }

   if ( OneWire::crc8( addr, 7) != addr[7]) {
       Serial.println("CRC is not valid!");
       return -1000;
   }

   if ( addr[0] != 0x10 && addr[0] != 0x28) {
       Serial.print("Device is not recognized");
       return -1000;
   }

   ds.reset();
   ds.select(addr);
   ds.write(0x44,1); // start conversion, with parasite power on at the end

   byte present = ds.reset();
   ds.select(addr);
   ds.write(0xBE); // Read Scratchpad


   for (int i = 0; i < 9; i++) { // we need 9 bytes
```

```
    data[i] = ds.read();
  }

  ds.reset_search();

  byte MSB = data[1];
  byte LSB = data[0];

  float tempRead = ((MSB << 8) | LSB); //using two's compliment
  float TemperatureSum = tempRead / 16;

  return TemperatureSum;

}

float getTemp72(){
    // control for 72
    byte data[12];
    byte addr[8];

  if ( !ds72.search(addr)) {
      //no more sensors on chain, reset search
      ds72.reset_search();
      return -1000;
  }

  if ( OneWire::crc8( addr, 7) != addr[7]) {
      Serial.println("CRC is not valid!");
      return -1000;
  }

  if ( addr[0] != 0x10 && addr[0] != 0x28) {
      Serial.print("Device is not recognized");
      return -1000;
  }

  ds72.reset();
  ds72.select(addr);
  ds72.write(0x44,1); // start conversion, with parasite power on at the end
  byte present72 = ds72.reset();
  ds72.select(addr);
  ds72.write(0xBE); // Read Scratchpad


  for (int i = 0; i < 9; i++) { // we need 9 bytes
    data[i] = ds72.read();
  }

  ds72.reset_search();

  byte MSB = data[1];
  byte LSB = data[0];

  float tempRead = ((MSB << 8) | LSB); //using two's compliment
  float TemperatureSum = tempRead / 16;

  return TemperatureSum;
}

float getTemp92(){
    // control for 92
 byte data[12];
    byte addr[8];

  if ( !ds92.search(addr)) {
      //no more sensors on chain, reset search
      ds92.reset_search();
      return -1000;
  }

  if ( OneWire::crc8( addr, 7) != addr[7]) {
      Serial.println("CRC is not valid!");
      return -1000;
  }

  if ( addr[0] != 0x10 && addr[0] != 0x28) {
      Serial.print("Device is not recognized");
      return -1000;
  }
```

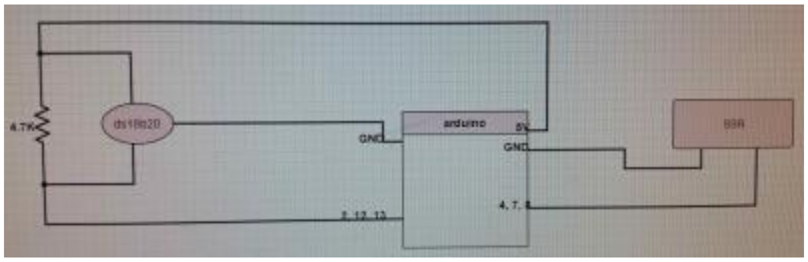
```
ds92.reset();
ds92.select(addr);
ds92.write(0x44,1); // start conversion, with parasite power on at the end

byte present92 = ds92.reset();
ds92.select(addr);
ds92.write(0xBE); // Read Scratchpad


for (int i = 0; i < 9; i++) { // we need 9 bytes
  data[i] = ds92.read();
}

ds92.reset_search();

byte MSB = data[1];
byte LSB = data[0];

float tempRead = ((MSB << 8) | LSB); //using two's compliment
float TemperatureSum = tempRead / 16;

return TemperatureSum;
}
```

Experimental set-up

Servo circuit



Heater circuit



For amplification of yellow jacket DNA

Water bath temperatures:

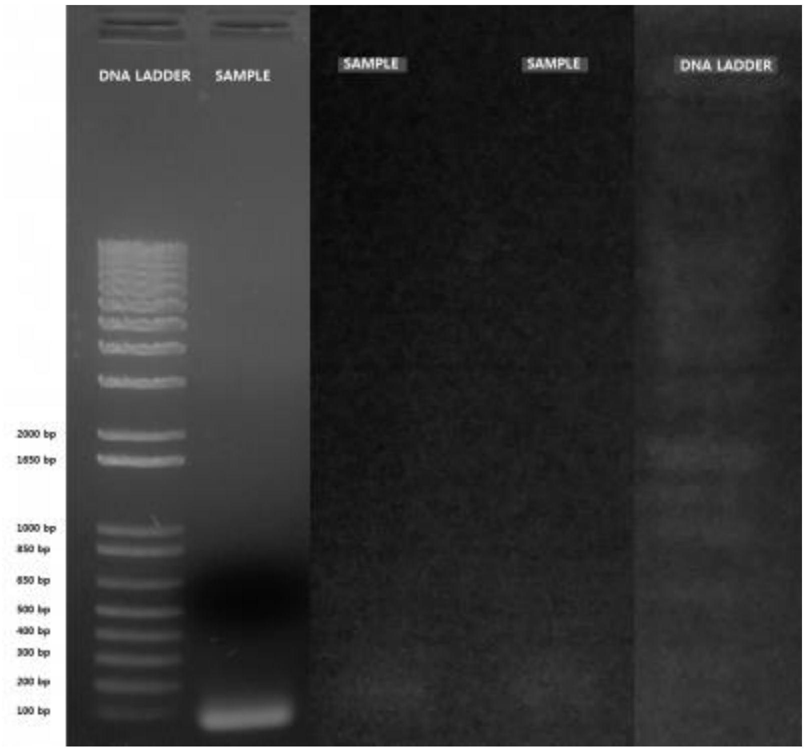
- 92 deg C
- 72 deg C
- 58 deg C

PCR cycle:

1. 92 deg C for 2 minutes
2. Repeat below 35 times:
   1. 92 deg C for 30s
   2. 58 deg C for 30s
   3. 72 deg C for 30s
3. 72 deg C for 5 minutes

Results

Lonza FlashGels were used and run for 5-7 minutes at 275 V.



Resources

- Rapid and Low-cost PCR Thermal Cycler
- Lonza FlashGel DNA system