

题目：基于 Arduino 的小型六足机器人

班级：10011303 学号：2013302539 姓名：缪宇飏

邮箱：myyerrol@126.com 手机：18706873458

一、背景

在自然界和人类社会中存在一些人类无法到达的地方和可能危及人类生命的特殊场合。如行星表面、灾难发生矿井、防灾救援和反恐斗争等，对这些危险环境进行不断地探索和研究，寻求一条解决问题的可行途径成为科学技术发展和人类社会进步的需要。地形不规则和崎岖不平是这些环境的共同特点。从而使轮式机器人和履带式机器人的应用受到限制。以往的研究表明轮式移动方式在相对平坦的地形上行驶时，具有相当的优势运动速度迅速、平稳，结构和控制也较简单，但在不平地面上行驶时，能耗将大大增加，而在松软地面或严重崎岖不平的地形上，车轮的作用也将严重丧失移动效率大大降低。为了改善轮子对松软地面和不平地面的适应能力，履带式移动方式应运而生但履带式机器人在不平地面上的机动性仍然很差行驶时机身晃动严重。与轮式、履带式移动机器人相比在崎岖不平的路面步行机器人具有独特优越性能在这种背景下多足步行机器人的研究蓬勃发出步行机器人的优展起来。而仿生步行机器人的出现更加显示势。

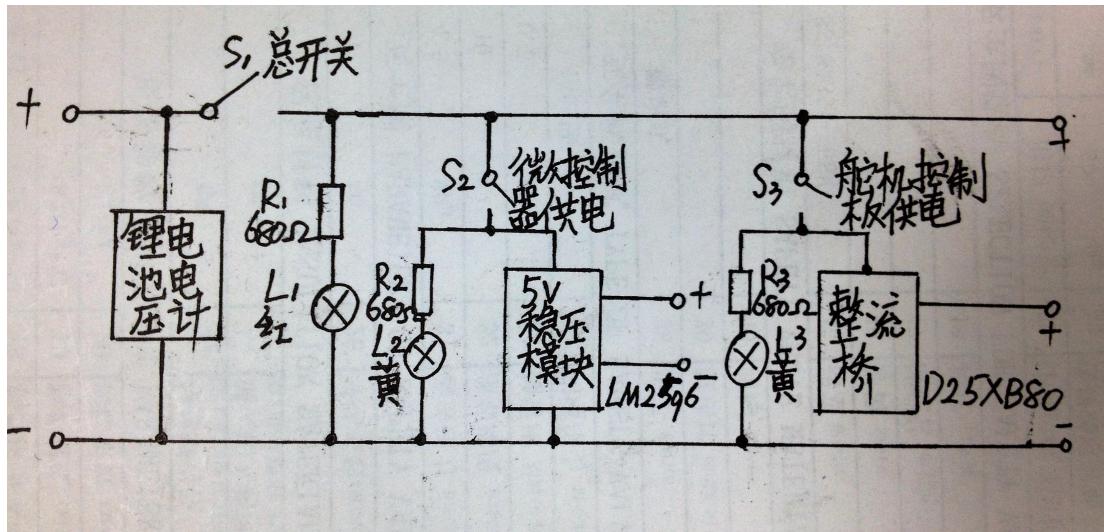
多足步行机器人的运动轨迹是一系列离散的足印运动时只需要离散的点接触地面对环境的破坏程度也较小可以在可能到达的地面上选择最优的支撑点对崎岖地形的适应性强。正因为如此多足步行机器人对环境的破坏程度也较小。轮式和履带式机器人的则是一条条连续的辙迹。崎岖地形中往往含有岩石、泥土、沙子甚至峭壁和陡坡等障碍物可以稳定支撑机器人的连续路径十分有限，这意味着轮式和履带式机器人在这种地形中已经不适用。多足步行机器人的腿部具有多个自由度使运动的灵活性大大增强。它可以通过调节腿的长度保持身体水平也可以通过调节腿的伸展程度调整重心的位置因此不易翻倒稳定性更高。当然多足步行机器人也存在一些不足之处。比如为使腿部协调稳定运动从机械结构设计到控制系统算法都比较复杂相比自然界的节肢动物仿生多足步行机器人的机动性还有很大差距。

二、原理

1、电路

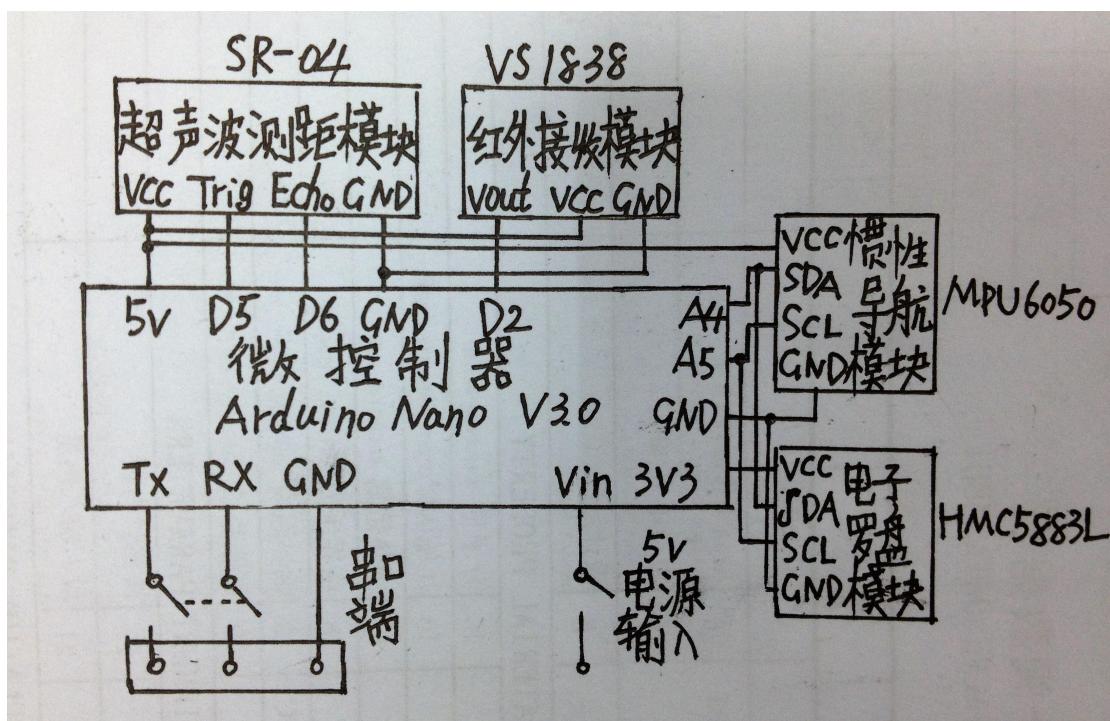
电源降压模块：

电源降压模块可以说是整个六足机器人的核心。它通过降压电路分别给 18 路舵机控制板和 Arduino Nano 微控制器供电。我设计了简单的开关 LED 电路来控制和显示当前降压电路的状态。大体的原理是：闭合 S1，L1 灯亮，7.4V 总电源接入降压电路。在 S1 闭合的情况下，闭合 S2 或 S3 可以分别点亮 L2 或 L3 灯，并开启微控制器或舵机控制板的供电。



微控制器外围模块：

微控制器外围电路包含了 SR-04 超声波测距模块、VS1838 红外接收模块、MPU6050 惯性导航模块和 HMC5883L 电子罗盘模块。通过对微控制器 Arduino Nano 进行嵌入式编程，可以使六足机器人完成红外遥控、超声波避障等基本功能。



2、仿生

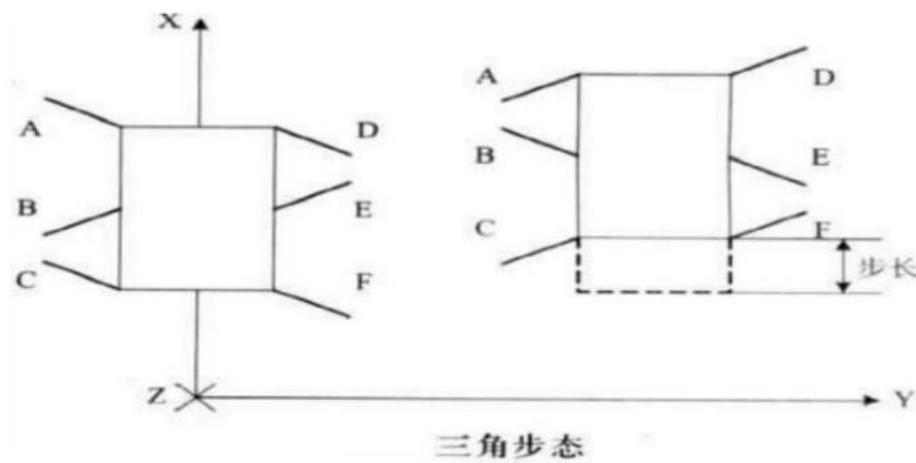
六足机器人又叫蜘蛛机器人，是多足机器人的一种。仿生式六足机器人，顾名思义，六足机器人在我们理想架构中，我们借鉴了自然界。

昆虫的运动原理。足是昆虫的运动器官。昆虫有 3 对足，在前胸、中胸和后胸各有一对，我们相应地称为前足、中足和后足。每个足由基节、转节、腿节、胫节、跗节和前跗节几部分组成。基节是足最基部的一节，多粗短。转节常与腿

节紧密相连而不活动。腿节是最长最粗的一节。第四节叫胫节，一般比较细长，长着成排的刺。第五节叫跗节，一般由2-5个亚节组成；为的是便于行走。在最末节的端部还长着两个又硬又尖的爪，可以用它们来抓住物体。行走是以三条腿为一组进行的，即一侧的前、后足与另一侧的中足为一组。这样就形成了一个三角形支架结构，当这三条腿放在地面并向后蹬时，另外三条腿即抬起向前准备替换。前足用爪固定物体后拉动虫体向前，中足用来支持并举起所属一侧的身体，后足则推动虫体前进，同时使虫体转向。这种行走方式使昆虫可以随时随地停息下来，因为重心总是落在三角支架之内。并不是所有成虫都用六条腿来行走，有些昆虫由于前足发生了特化，有了其他功用或退化，行走就主要靠中、后足来完成了。大家最为熟悉的要算螳螂了，我们常可看到螳螂一对钳子般的前足高举在胸前，而由后面四条足支撑地面行走。

3、步态

六足步行机器人的步态是多样的，其中三角步态是六足步行机器人实现步行的典型步态。“六足纲”昆虫步行时，一般不是六足同时直线前进，而是将三对足分成两组，以三角形支架结构交替前行。目前，大部分六足机器人采用了仿昆虫的结构，6条腿分布在身体的两侧，身体左侧的前、后足及右侧的中足为一组，右侧的前、后足和左侧的中足为另一组，分别组成两个三角形支架，依靠大腿前后划动实现支撑和摆动过程，这就是典型的三角步态行走法，如图所示。图中机器人的髋关节在水平和垂直方向上运动。此时，B、D、F脚为摆动脚，A、C、E脚原地不动，只是支撑身体向前。由于身体重心低，不用协调Z向运动，容易稳定，所以这种行走方案能得到广泛运用。



三、完成

- 1、完成六足机器人的电源降压驱动模块的焊接与调试。
- 2、完成六足机器人的主控和外围传感器电路的焊接与调试。
- 3、实现经典的六足机器人三角步态算法。
- 4、实现简单的六足机器人避障算法。

四、代码

1、构造函数

定义:

```
HexapodBionicRobot(IRrecv *ir_receiver, decode_results *ir_results);
```

实现:

```
HexapodBionicRobot::HexapodBionicRobot(IRrecv *ir_recviver,
                                         decode_results *ir_results)
    : ir_receiver_(ir_recviver),
      ir_results_(ir_results)
{
    pinMode(PIN_LED, OUTPUT);
    pinMode(PIN_TRIGGER, OUTPUT);
    pinMode(PIN_ECHO, INPUT);

    mode_flag_ = MODE_REMOTE;

    duration_ = 0.0;
    distance_ = 0.0;
}
```

功能:

接受红外库类的对象指针，将对红外信息的操作进行封装。初始化 Arduino 的一些引脚和变量。

2、避障函数

定义:

```
void avoidFrontObstacle(void);
```

实现:

```
void HexapodBionicRobot::avoidFrontObstacle(void)
{
    float distance = getUltrasonicDistance();
    Serial.println(distance);

    if (distance == false) {
        return ;
    }
    else if (distance <= 2.5) {
        moveRobotBody(DIR_STOP, 2);
    }
    else if (distance <= 5.0) {
        moveRobotBody(DIR_BACK, 2);
    }
}
```

```
    }  
}
```

功能:

当检测到前方的障碍物时，机器人后退或停止。

3、处理红外信息函数

定义:

```
void handleInfraredInformation(void);
```

实现:

```
void HexapodBionicRobot::handleInfraredInformation(void)  
{  
    float distance = 0.0;  
    uint32_t ir_results = getInfraredInformation();  
  
    if (ir_results == false) {  
        return ;  
    }  
    else {  
#if DEBUG  
        Serial.print("Infrared code: ");  
        Serial.println(ir_results, HEX);  
#endif  
        if (ir_results == 0xFF629D) {  
            mode_flag_ = MODE_REMOTE;  
        }  
        else if (ir_results == 0xFFE21D) {  
            mode_flag_ = MODE_AUTO;  
        }  
  
        if (mode_flag_ == MODE_REMOTE) {  
            digitalWrite(PIN_LED, LOW);  
            if (ir_results == 0xFF02FD) {  
                moveRobotBody(DIR_FRONT, 2);  
                delay(RUNTIME);  
            }  
            else if (ir_results == 0xFF9867) {  
                moveRobotBody(DIR_BACK, 2);  
                delay(RUNTIME);  
            }  
            else if (ir_results == 0FFE01F) {  
                moveRobotBody(DIR_LEFT, 2);  
                delay(RUNTIME);  
            }  
        }  
    }  
}
```

```

        }
        else if (ir_results == 0xFF906f) {
            moveRobotBody(DIR_RIGHT, 2);
            delay(RUNTIME);
        }
        else if (ir_results == 0xFFA857) {
            moveRobotBody(DIR_STOP, 2);
            delay(RUNTIME);
        }
        avoidFrontObstacle();
    }
    else if (mode_flag_ == MODE_AUTO) {
        digitalWrite(PIN_LED, HIGH);
        while (ir_results != 0xFF629D) {
            ir_results = getInfraredInformation();
            moveRobotBody(DIR_FRONT, 2);
            delay(RUNTIME);
            avoidFrontObstacle();
        }
        mode_flag_ = MODE_REMOTE;
    }
}
}

```

功能：

接收红外遥控器的数据并进行相应的处理，可以完成机器人的简单红外遥控和自主移动。

4、获取红外信息函数

定义：

```
uint32_t getInfraredInformation(void);
```

实现：

```

uint32_t HexapodBionicRobot::getInfraredInformation(void)
{
    if (ir_receiver_->decode(ir_results_)) {
        ir_receiver_->resume();
        return ir_results_->value;
    }
    else {
        return false;
    }
}

```

功能:

调用红外库来获取红外编码的信息。

5、处理超声波距离函数

定义:

```
void handleUltrasonicDistance(void);
```

实现:

```
void HexapodBionicRobot::handleUltrasonicDistance(void)
{
    float distance = getUltrasonicDistance();

    if (distance == false) {
        return ;
    }
    else if (distance <= 5.0) {
        digitalWrite(PIN_LED, HIGH);
#if DEBUG
        Serial.println("Warning! Distance is too close!!!");
#endif
    }
    else {
        digitalWrite(PIN_LED, LOW);
    }
#if DEBUG
    Serial.print("Distance: ");
    Serial.print(distance);
    Serial.println("cm");
#endif
    delay(100);
}
```

功能:

获取超声波模块的数据，并做相应的处理操作。

6、获取超声波距离函数

定义:

```
float getUltrasonicDistance(void);
```

实现:

```
float HexapodBionicRobot::getUltrasonicDistance(void)
```

```

{
    duration_ = 0.0;
    distance_ = 0.0;

    digitalWrite(PIN_TRIGGER, LOW);
    delayMicroseconds(10);
    digitalWrite(PIN_TRIGGER, HIGH);
    delayMicroseconds(10);
    digitalWrite(PIN_TRIGGER, LOW);

    duration_ = pulseIn(PIN_ECHO, HIGH, TIMEOUT);

    if (duration_ == 0.0) {
        return false;
    }
    else {
        distance_ = duration_ * 0.017;
        return distance_;
    }
}

```

功能：

根据超声波模块的使用方法，计算得到障碍物与超声波模块的之间的距离。

7、移动机器人躯体函数

定义：

```
void moveRobotBody(uint8_t direction, uint8_t times);
```

实现：

```

void HexapodBionicRobot::moveRobotBody(uint8_t direction, uint8_t times)
{
    char string_direction[5];
    char string_times[5];

    itoa(direction, string_direction, RADIX);
    itoa(times, string_times, RADIX);

    Serial.print("#");
    Serial.print(string_direction);
    Serial.print("G");
    Serial.print(string_times);
    Serial.println("C");
}

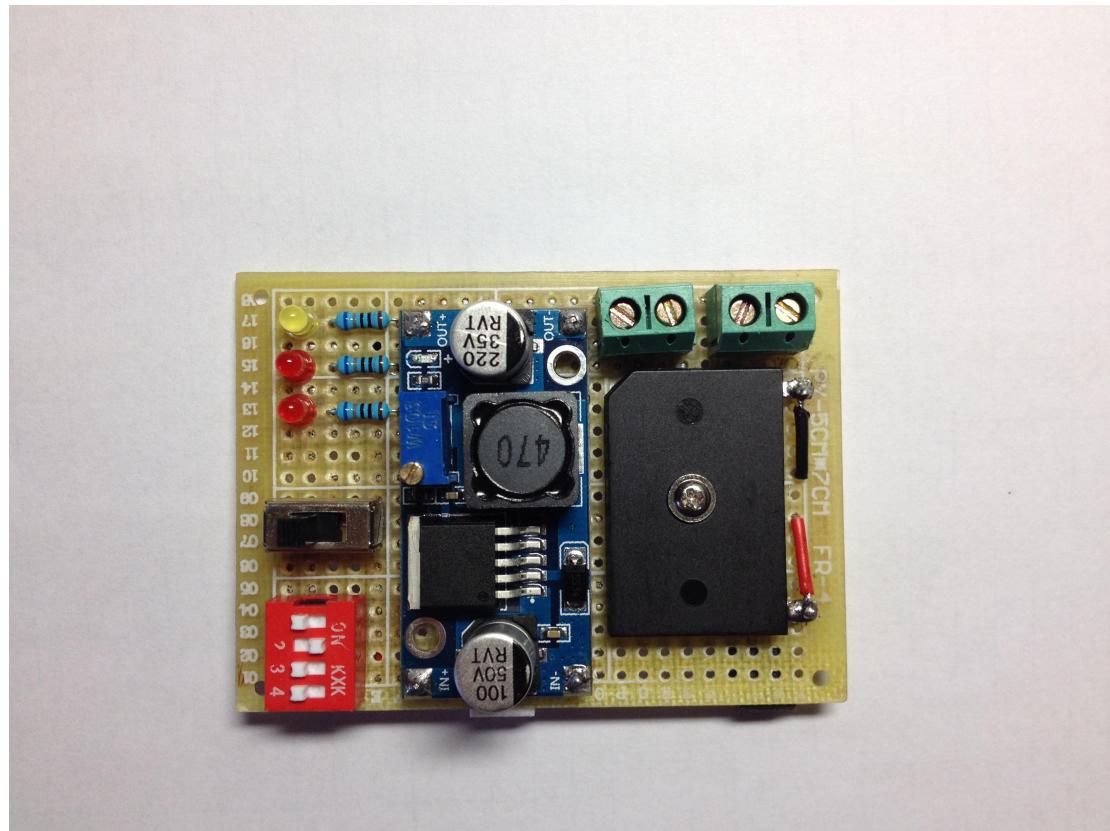
```

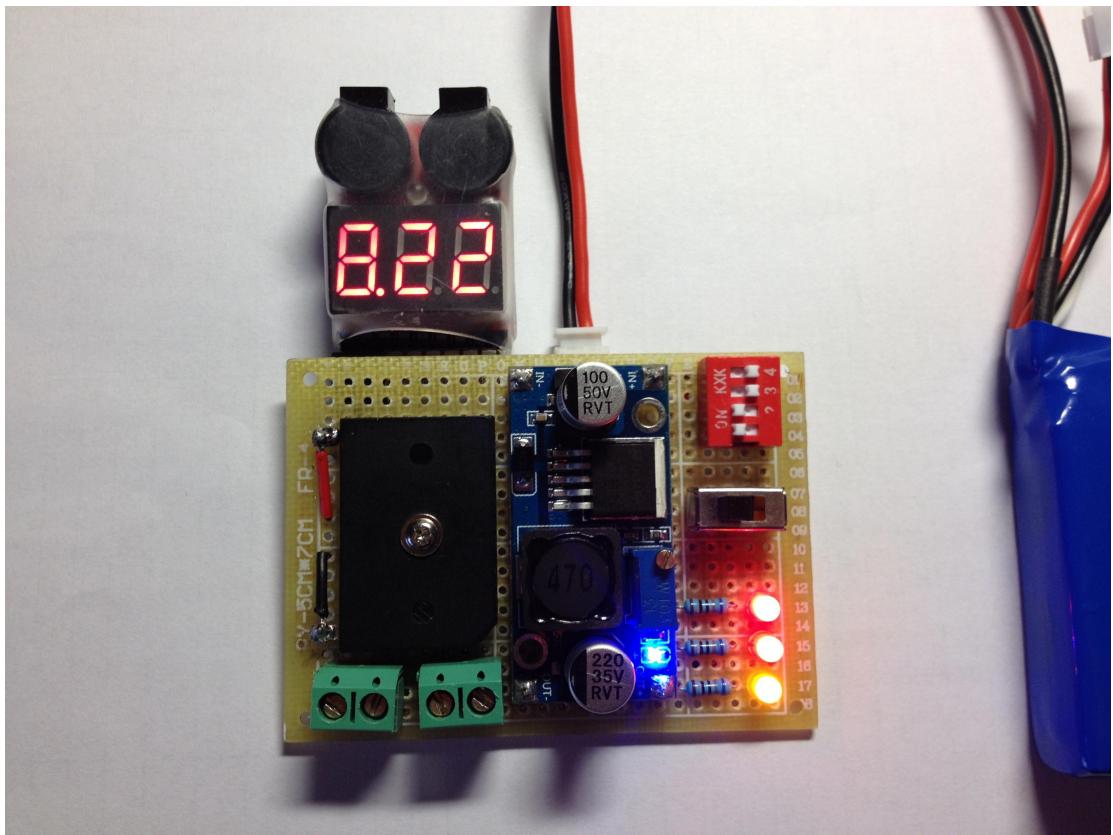
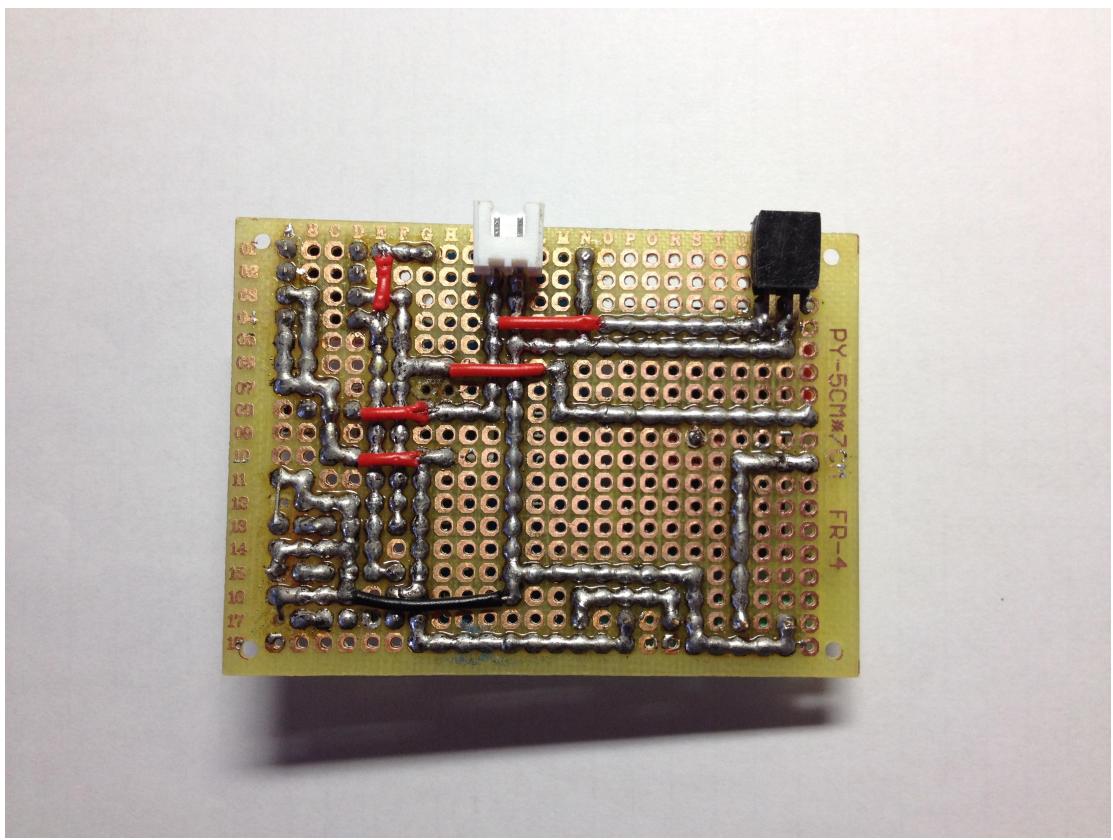
功能:

根据移动的方向和次数来给串口发送相应数据，使得舵机控制板完成相应的动作。

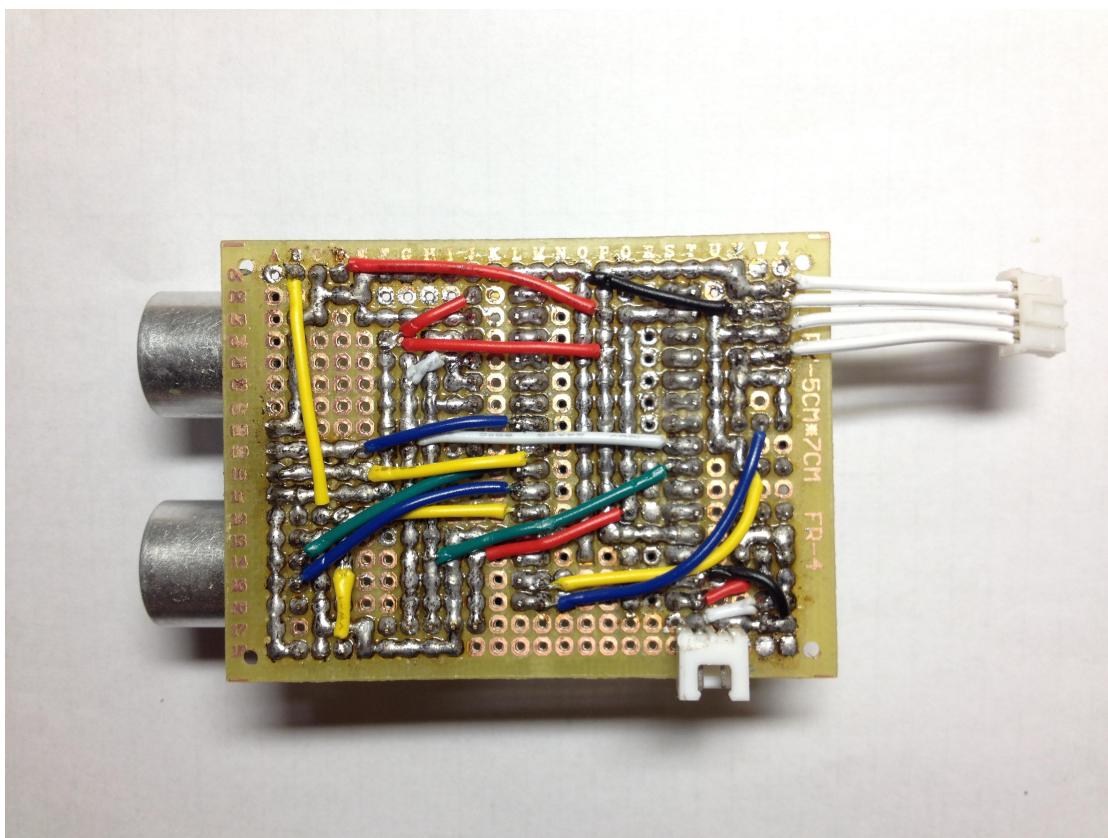
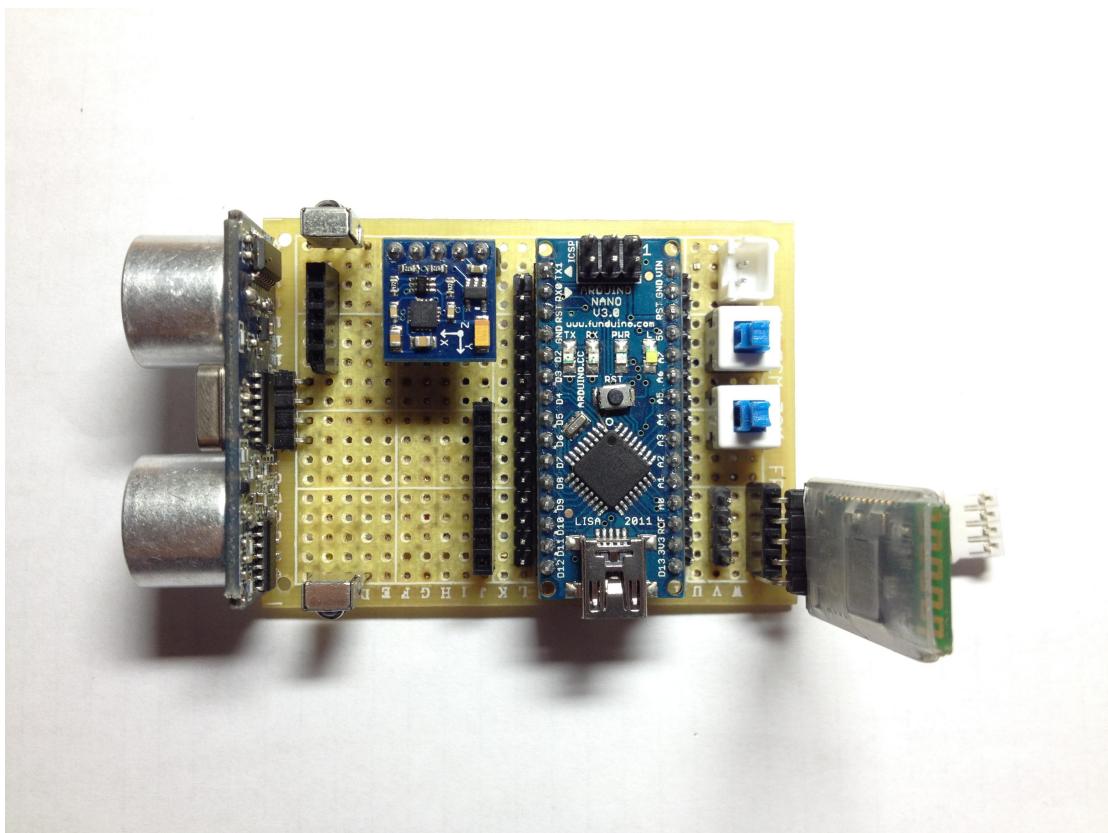
五、结果

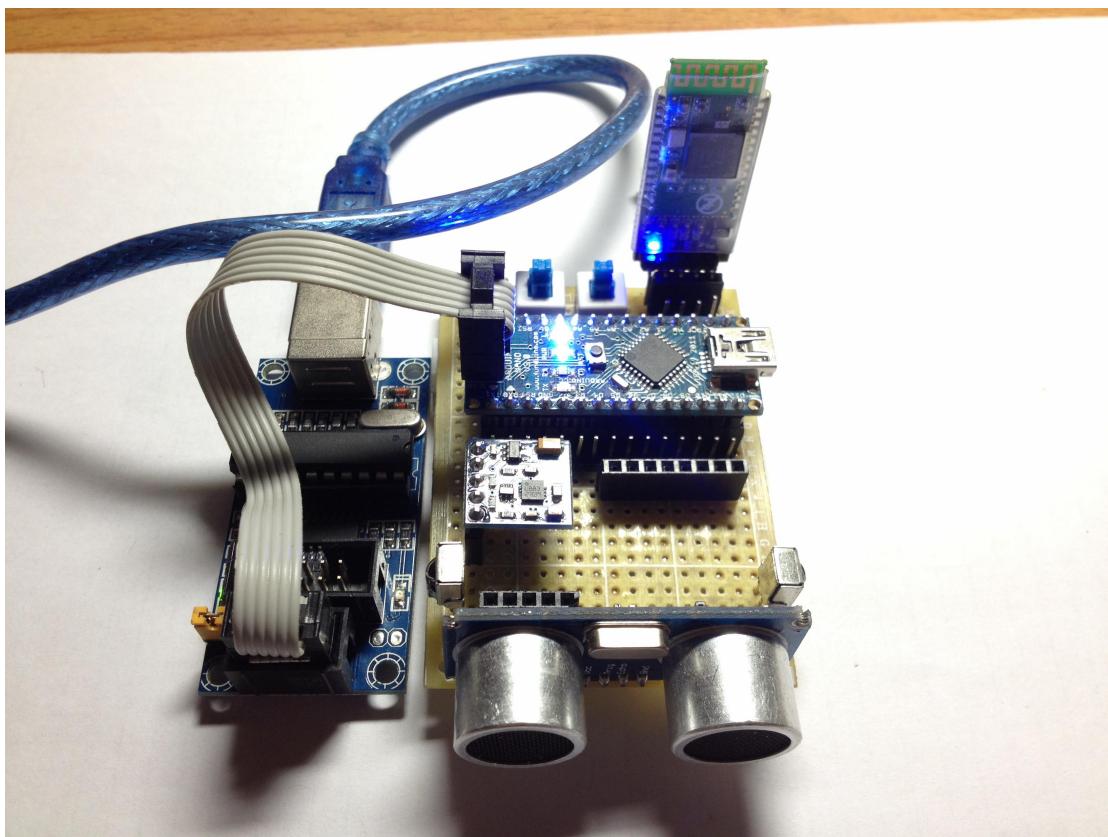
1、电源降压模块:



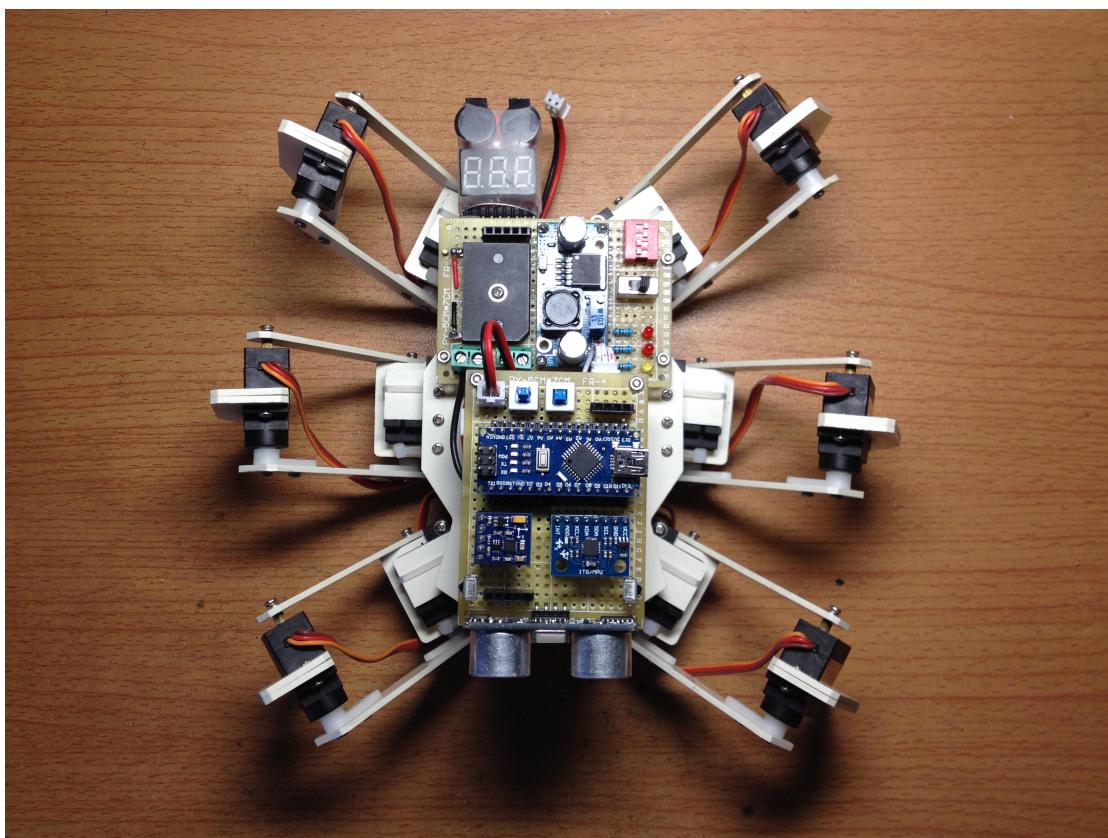


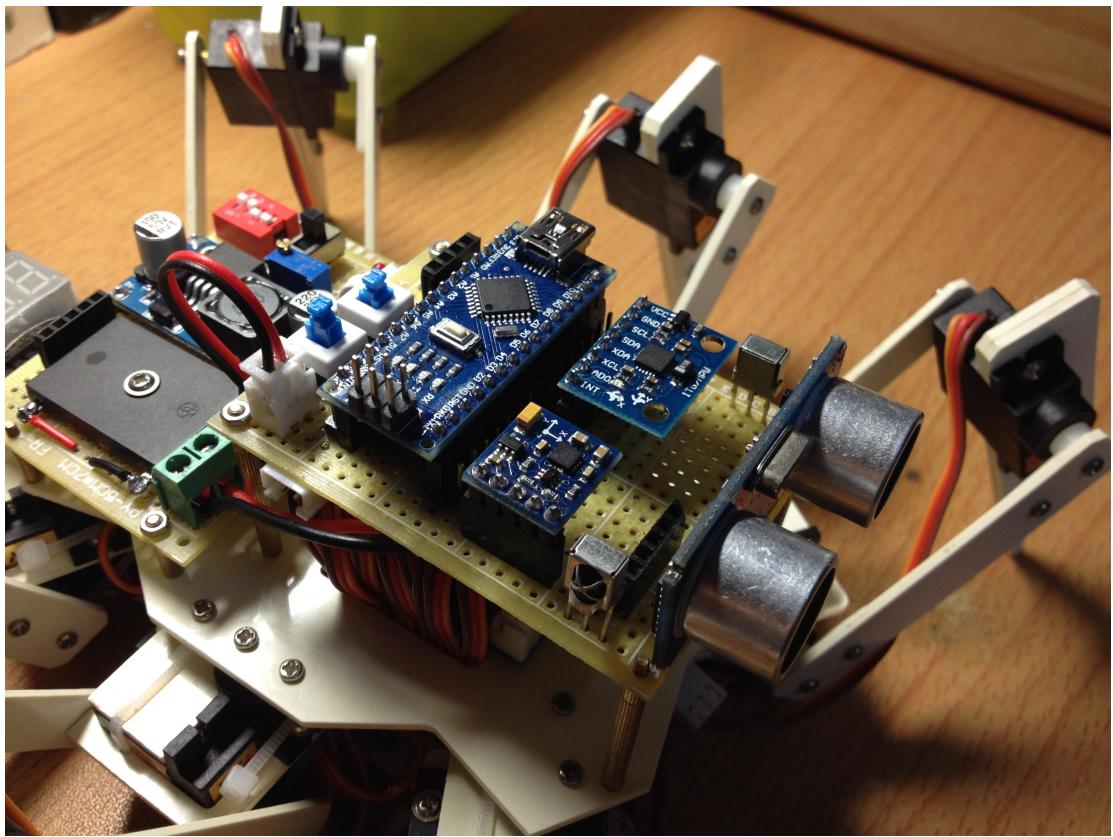
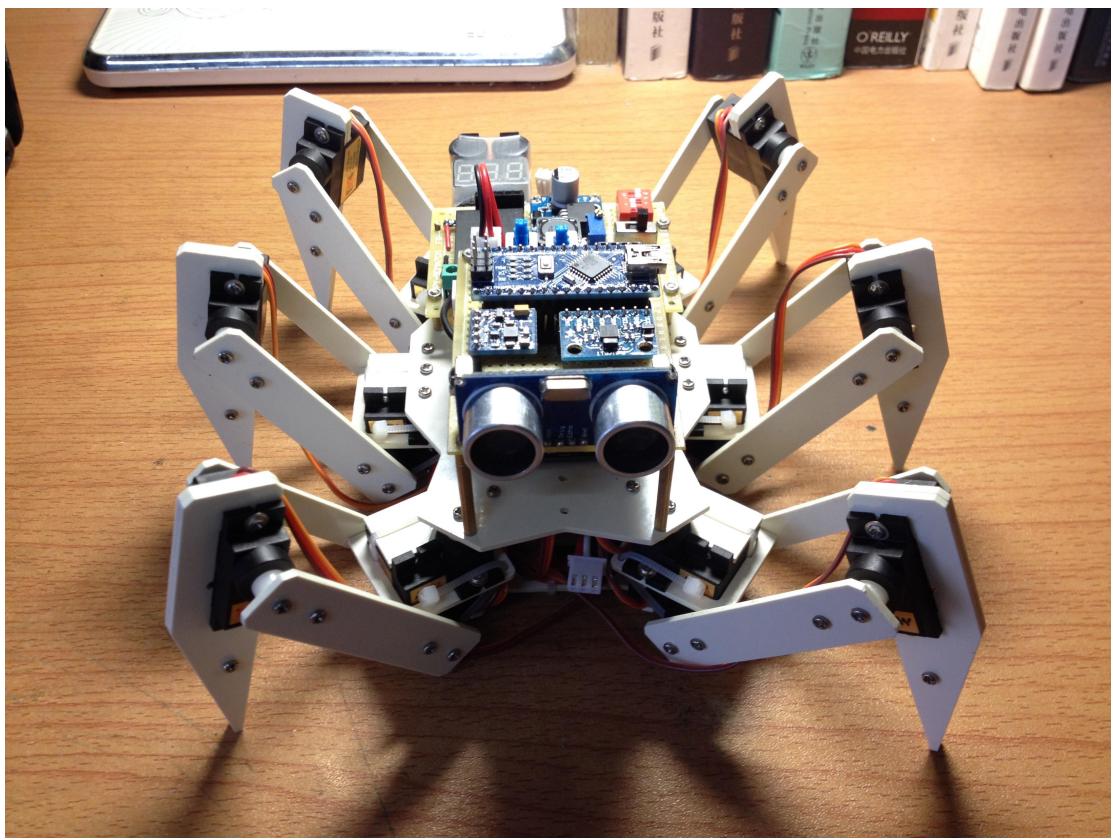
2、微控制器外围模块：

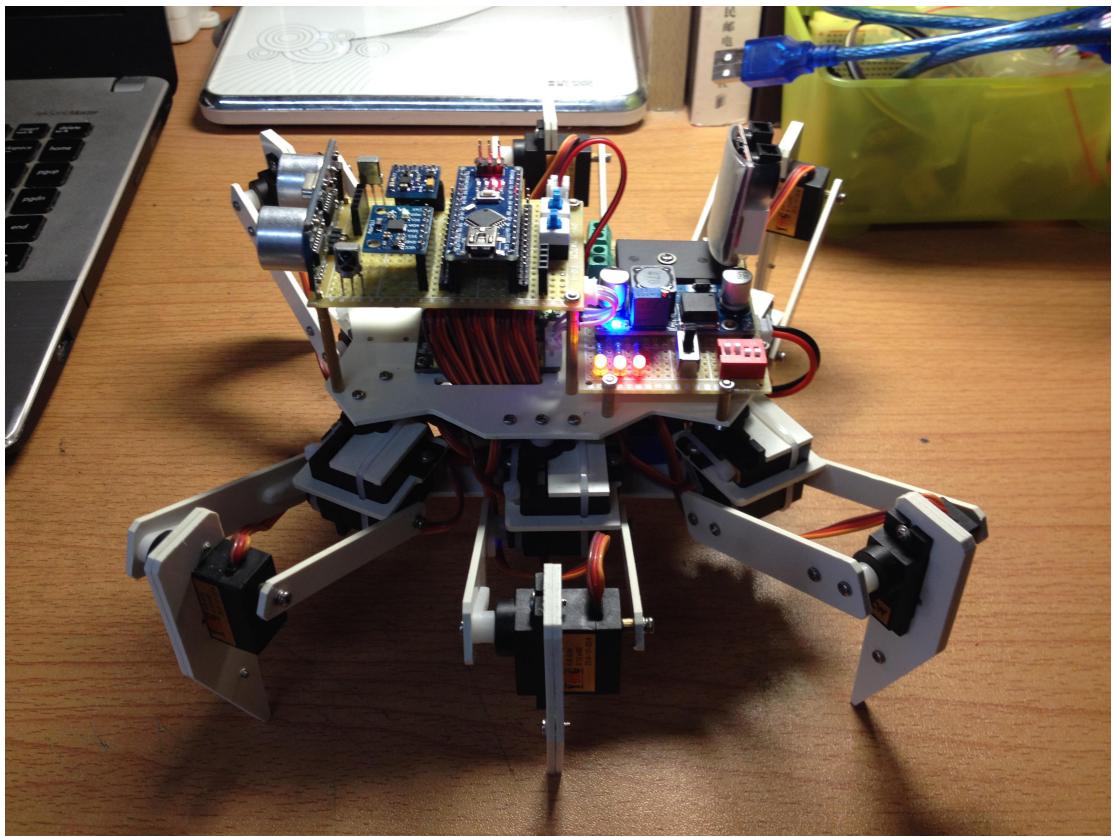
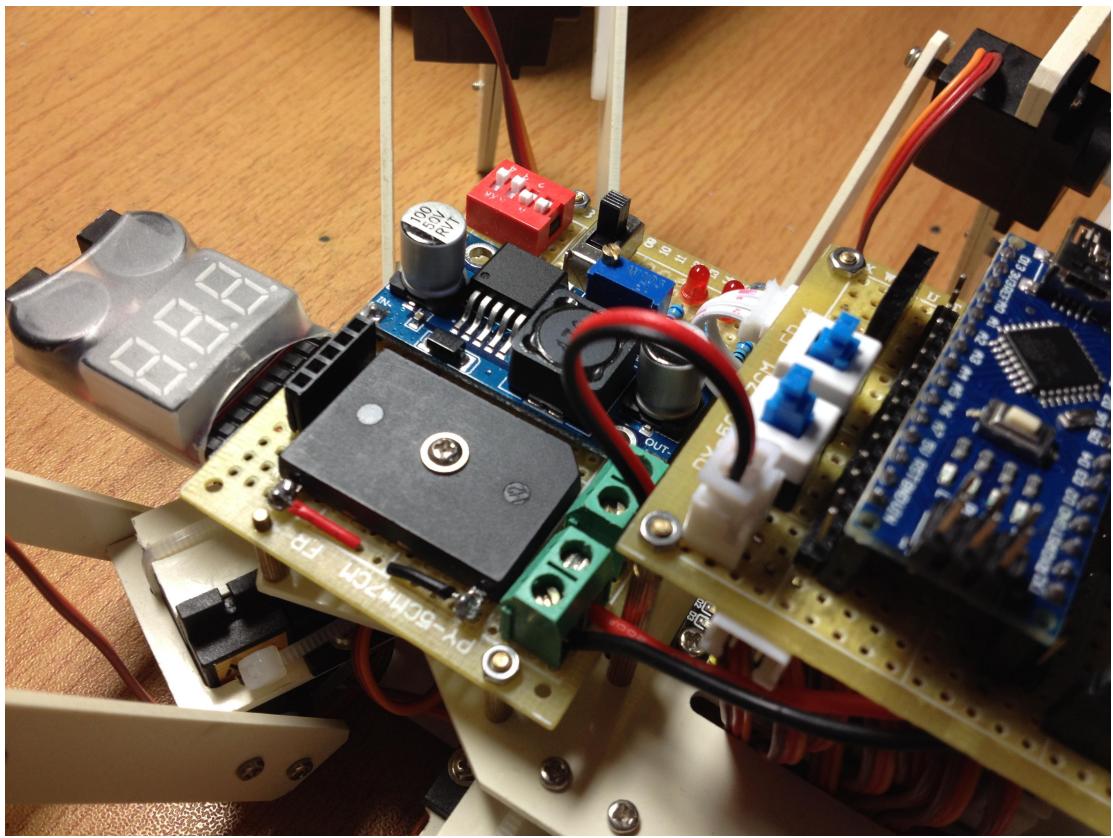


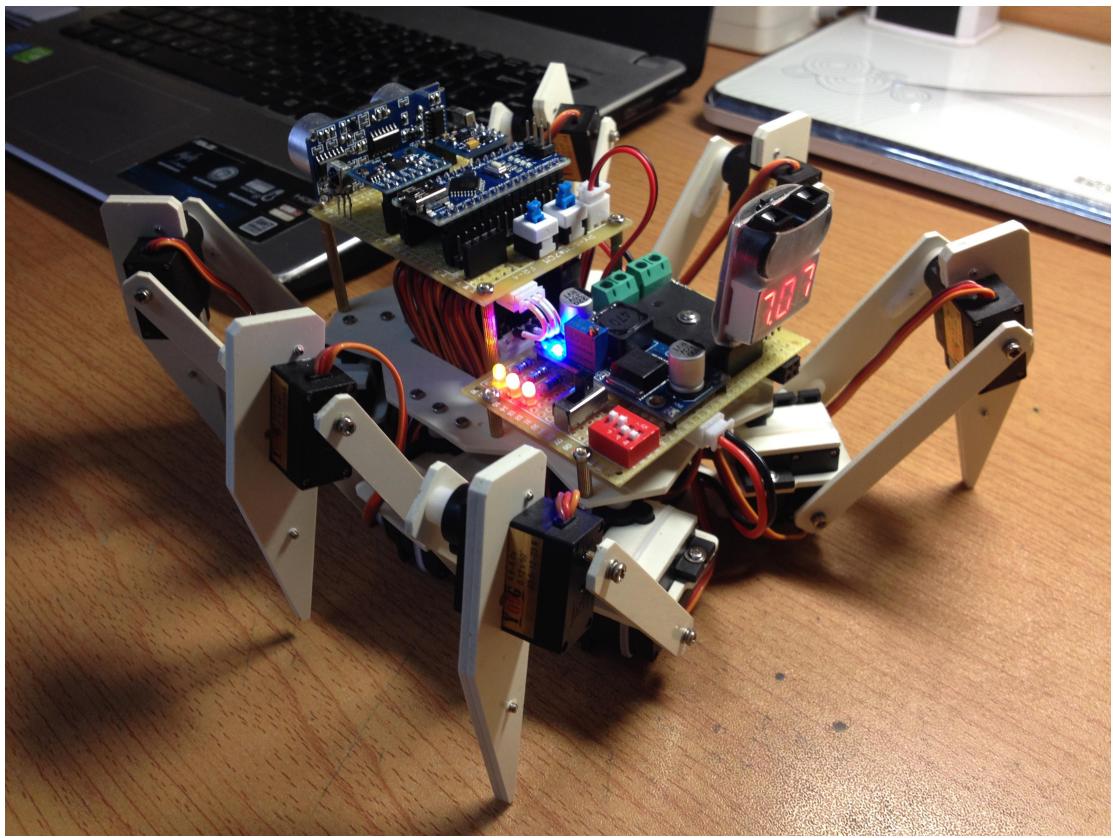


3、六足机器人整体：









六、问题

- 1、在通电测试电源减压驱动模块的时候，发现板子放到机器人背部的某些地方会出现短路冒黑烟的现象。
- 2、在测试通用串口的电路功能时，发现 Arduino 发送的串口数据，舵机控制板能接收到但并没有任何反应。
- 3、在从六足机器人上拆下主控外围电路板的时候，其与舵机控制板的连接线发生了断裂。
- 4、在使用 SR-04 超声波模块的时候，发现模块总是返回超时，计算得到的距离也是错误的。

七、解决

- 1、这个问题的原因终于被我找到了。原来六足机器人的背部有一些固定肢体用的螺丝，而恰好板子上的正负极与螺丝接触，导致短路的发生。我的解决办法是使用四个铜柱把板子架高，这样短路就不会发生了。
- 2、仔细排查之后，我发现原来自己在焊接 TX 和 RX 的时候，不小心把 Arduino 的 TX 和 RX 分别与舵机控制板的 TX 和 RX 连到了一起。解决办法是直接将 TX 和 RX 交叉焊接，即 Arduino 的 TX 和 RX 分别连舵机控制板的 RX 和 TX 端。
- 3、这个问题的原因主要是我在设计接口的时候，是直接将电线组的一端焊在了板子的背部。多次的弯折以及用力拉扯电线导致了金属疲劳，从而发生了电线组的断裂。我的解决办法是在两个板子上都焊上标准的母接插件。同样地，电线组的两端也都焊接上公接插件。这样在调试的时候，插拔电线组就很轻松方便，不

会出现电线断裂的情况了。

4、问题出现的原因是给模块供电的电压不够。要使用 Arduino 的 5V 电压而不是之前的 3.3V。

八、计划

- 1、完善之前实现的六足机器人避障算法，使其可以在遇到障碍物的时候，可以更及时地做出相应的规避动作。
- 2、增强串口通信的协议，让 Arduino 可以直接控制 18 个舵机的角度位置，为之后的步态解算和规划做准备。
- 3、使用 MPU6050（三轴陀螺仪+三轴加速度计）和 HMC5883L（三轴电子罗盘）来做六足机器人的运动闭环控制。
- 4、使用 ROS（机器人操作系统）来做整个软硬件系统的集成。

九、总结

这次做的六足机器人算是圆了我大一时候的梦想。我大一时曾经尝试做过六足机器人，但由于六足机器人本身制作与控制的复杂性，以及当时我个人对嵌入式知识一窍不通，第一次的六足制作失败。但随着这次大四开的自主开放实验课，我终于有了充分的时间来重新挑战六足机器人的制作。在此次制作的过程中，我学会了从软硬件协同设计的角度来完成对工程的设计，这包括对电路元器件的合理布局、焊接的合理走线以及模块之间摆放位置的规划等。总之，虽然现在的六足机器人只能做到简单的移动和避障，但我相信在不久之后的开发中，它可以变得越来越智能！