Lampiran

Lampiran 1 listing program

```
include <SPI.h>
#include <LoRa.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);

#define SW1 7
#define SW2 6
#define SW3 5
#define SW4 4
#define SW5 3

//Lora setPin
#define ss 10
#define rst 9
#define dio0 2

String data = "";
String Power;
long lastSendTime = 0; // last send time
int interval = 50;     // interval between sends
byte msgCount = 0;           // count of outgoing messages
byte localAddress = 0xBB;     // address of this device
byte destination = 0xFF;      // destination to send to

void setup() {
  Serial.begin(9600);
  lcd.init(); lcd.backlight();
  lcd.setCursor(0,0); lcd.print("Boat Controller ");
  pinMode(SW1,INPUT_PULLUP);
  pinMode(SW2,INPUT_PULLUP);
  pinMode(SW3,INPUT_PULLUP);
  pinMode(SW4,INPUT_PULLUP);
  pinMode(SW5,INPUT_PULLUP);

  while (!Serial);
  //Serial.println("LoRa Sender");
  LoRa.setPins(ss, rst, dio0);
  if (!LoRa.begin(433E6)) { // or 915E6, the MHz speed of yout module
    //Serial.println("Starting LoRa failed!");
    lcd.setCursor(0,0); lcd.print("   Lora Error    ");
    delay(100);
    while (1);
```

```
bool Flag1 = false;
bool Flag2 = false;
bool Flag3 = false;
bool Flag4 = false;
bool Flag5 = false;

void loop() {
  if(!digitalRead(SW1) && !Flag1){
      data = "11";  Flag(1,0,0,0,0);
      //Serial.println("DataTX" + data);
      lcd.setCursor(0,0); lcd.print("  Belok Kiri    ");
    }
    if(!digitalRead(SW2) && !Flag2){
      data = "22";
      //Serial.println("DataTX " + data);
      lcd.setCursor(0,0); lcd.print("  Belok Kanan   ");
      Flag(0,1,0,0,0);
    }
    if(!digitalRead(SW3) && !Flag3){
      data = "33"; Flag(0,0,1,0,0);
      //Serial.println("DataTX " + data);
      lcd.setCursor(0,0); lcd.print("      Maju      ");
    }
    if(!digitalRead(SW4) && !Flag4){
      data = "44"; Flag(0,0,0,1,0);
      //Serial.println("DataTX " + data);
      lcd.setCursor(0,0); lcd.print("      Stop      ");
    }
    if(!digitalRead(SW5) && !Flag5){
      data = "55";  Flag(0,0,0,0,1);
      //Serial.println("DataTX " + data);
      lcd.setCursor(0,0); lcd.print("     Mundur     ");
    }

  if(millis()- lastSendTime > interval) {
    sendMessage(data); data = "";
    lastSendTime = millis();   // timestamp the message
    interval = random(50) + 100;
  }
  // parse for a packet, and call onReceive with the result:
  onReceive(LoRa.parsePacket());
}

void Flag(bool F1,bool F2,bool F3,bool F4,bool F5) {
     Flag1=F1; Flag2=F2; Flag3=F3; Flag4=F4; Flag5=F5;
}
```

```
    //Serial.print("Flag: "); Serial.print(Flag1);
    //Serial.print(Flag2); Serial.print(Flag3);
    //Serial.print(Flag4); Serial.println(Flag5);
}

void sendMessage(String dataTX) {
  LoRa.beginPacket();          // start packet
  LoRa.write(destination);     // add destination address
  LoRa.write(localAddress);    // add sender address
  LoRa.write(msgCount);        // add message ID
  LoRa.write(dataTX.length()); // add payload length
  LoRa.print(dataTX);          // add payload
  LoRa.endPacket();            // finish packet and send it
  msgCount++;                  // increment message ID
}

void onReceive(int packetSize) {
   if(!packetSize) return;  // if there's no packet, return
   // read packet header bytes:
   int recipient = LoRa.read();          // recipient address
   byte sender = LoRa.read();            // sender address
   byte incomingMsgId = LoRa.read();     // incoming msg ID
   byte incomingLength = LoRa.read();

   // ====================================
   //received a packet, print RSSI of packet
   Serial.print("Received packet:  ");
   String incoming = LoRa.readString();
   //Serial.print(incoming);
   // read packet
   while (LoRa.available()) {
     //Serial.print((char)LoRa.read());
     (char)LoRa.read();
   }
   //Serial.print("' with RSSI ");
   //Serial.println(LoRa.packetRssi());
   //====================================

   Power = incoming.substring(0, incoming.length());

   //Serial.println ("Power: "+Power+" %");
   lcd.setCursor(3,1); lcd.print("           ");
   lcd.setCursor(3,1);
   lcd.print("Power: "); lcd.print(Power); lcd.print('%');
}
```

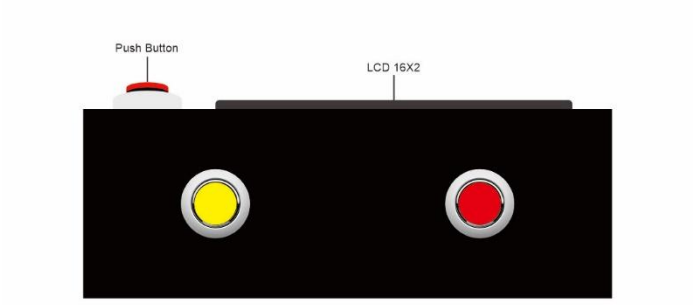
```
#include <SPI.h>
#include <LoRa.h>

#define  MKiri_1  A3
#define  MKiri_2  A2
#define  MKanan_1 A1
#define  MKanan_2 A0
#define  PWMKr 5
#define  PWMKn 6

//Lora setPin
#define ss 10
#define rst 9
#define dio0 2

int priviousValue = 0;
int liveValue = 0;
int val = 0, Adc, Power;
String sPower, dataRX;
long lastSendTime = 0;        // last send time
int interval = 50;            // interval between sends
byte msgCount = 0;            // count of outgoing messages
byte localAddress = 0xFF;     // address of this device
byte destination = 0xBB;      // destination to send to
byte count=0;

void setup() {
  Serial.begin(9600);
  pinMode(MKiri_1,OUTPUT);
  pinMode(MKiri_2,OUTPUT);
  pinMode(MKanan_1,OUTPUT);
  pinMode(MKanan_2,OUTPUT);
  pinMode(PWMKr,OUTPUT);
  pinMode(PWMKn,OUTPUT);
  analogWrite(PWMKr, 120);
  analogWrite(PWMKn, 120);

   while (!Serial);
  LoRa.setPins(ss, rst, dio0);
  Serial.println("LoRa Receiver");
  if (!LoRa.begin(433E6)) { // or 915E6, the MHz speed of yout module
    Serial.println("Starting LoRa failed!");
    delay(100);
    while (1);
}
```

```
void loop() {
  if (millis() - lastSendTime > interval) {
     Adc = analogRead(A5);
     Power = map(Adc,378,14, 0,100);

     /*count++;
     if(count>100) { count=0;
      Serial.println("Adc: "+String(Adc)+"  Power: "+String(Power));
     }
     */

     sPower = (String)Power;
     sendMessage(sPower);
     lastSendTime = millis();
     interval = random(50) + 100;    // 2-3 seconds
  }
  // parse for a packet, and call onReceive with the result:
  onReceive(LoRa.parsePacket());
}

void onReceive(int packetSize) {
  if(!packetSize) return;  // if there's no packet, return
  // read packet header bytes:
  int recipient = LoRa.read();       // recipient address
  byte sender = LoRa.read();         // sender address
  byte incomingMsgId = LoRa.read();  // incoming msg ID
  byte incomingLength = LoRa.read();

  // =====================================
  //received a packet, print RSSI of packet
  //Serial.print("Received packet:  ");
  String incoming = LoRa.readString();
  //Serial.print(incoming);
  // read packet
  while (LoRa.available()) {
    //Serial.print((char)LoRa.read());
    (char)LoRa.read();
  }
  //Serial.print("' with RSSI ");
  //Serial.println(LoRa.packetRssi());
  //=======================================
  dataRX = incoming.substring(0, incoming.length());
  val = dataRX.toInt();
```

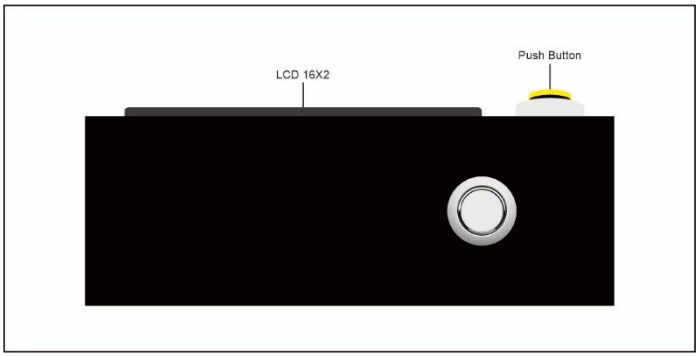
```
  liveValue = val;

  if(priviousValue != liveValue){
    /*Serial.print(val);
    if(val == 11){ Kiri();   Serial.println(" (Kiri)"); }
    if(val == 22){ Kanan();  Serial.println(" (Kanan)");}
    if(val == 33){ Maju();   Serial.println(" (Maju)"); }
    if(val == 44){ Stop();   Serial.println(" (Stop)"); }
    if(val == 55){ Mundur(); Serial.println(" (Mundur)");}
    */
    if(val == 11){ Kiri();  }
    if(val == 22){ Kanan(); }
    if(val == 33){ Maju();  }
    if(val == 44){ Stop();  }
    if(val == 55){ Mundur();}
    priviousValue = liveValue;
  }
}

void sendMessage(String outgoing) {
  LoRa.beginPacket();             // start packet
  LoRa.write(destination);        // add destination address
  LoRa.write(localAddress);       // add sender address
  LoRa.write(msgCount);           // add message ID
  LoRa.write(outgoing.length());  // add payload length
  LoRa.print(outgoing);           // add payload
  LoRa.endPacket();               // finish packet and send it
  msgCount++;                     // increment message ID
}

void Maju(){ Gerak(1,0,1,0); }
void Mundur(){ Gerak(0,1,0,1); }
void Kiri(){ Gerak(0,0,1,0); }
void Kanan(){ Gerak(1,0,0,0); }
void Stop(){ Gerak(0,0,0,0); }

void Gerak(bool MKr1, bool MKr2,
           bool MKn1, bool MKn2){
   digitalWrite(MKiri_1,MKr1);
   digitalWrite(MKiri_2,MKr2);
   digitalWrite(MKanan_1,MKn1);
   digitalWrite(MKanan_2,MKn2);
   //Serial.print("Data: "); Serial.print(MKr1); Serial.print(MKr2);
   //Serial.print(MKn1); Serial.println(MKn2);
}
```

Lampiran 2 Dokumentasi pengujian alat

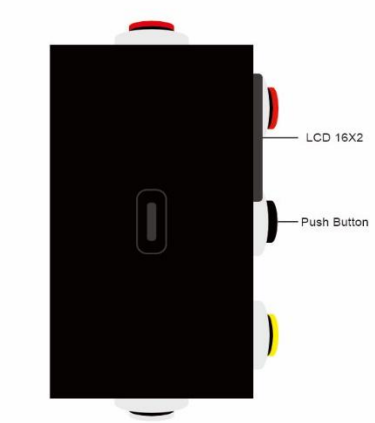
Lampiran 3 Desain konstruksi remot kontrol



Desain konstruksi tampak atas remot kontrol



Desain konstruksi Tampak bawah remot kontrol



Desain konstruksi Tampak samping remot kontrol