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

#define STARTMARKER 0xFF
#define ADRESS 0xF5

const int kierSilnikLewy=4;
const int kierSilnikPrawy=7;
const int SilnikLewy=5;
const int SilnikPrawy=6;

//Instrukcje
//0x01  Read  Instruction to read data from the Device
//0x02  Write Instruction to write data on the Device
//Rejestr
//0x00 Parametr 1
//0x01 Parametr 2
//0x02 Parametr 3
//0x03 Parametr 4

uint8_t  COMMAND[16];
uint8_t EEP[16];
uint8_t  REJESTR[4];
boolean newData = false;
byte DataLength = 0x00;

void setup() {
   Serial.begin(9600);
   pinMode(LED_BUILTIN, OUTPUT);
    pinMode(kierSilnikLewy,OUTPUT);
    pinMode(kierSilnikPrawy,OUTPUT);
    pinMode(SilnikLewy,OUTPUT);
    pinMode(SilnikPrawy,OUTPUT);
   digitalWrite(13, HIGH);
  for(int i = 0 ; i < 16 ; i++) {
    EEP[i] = 0x30;
    COMMAND[i] = 0x30;
  }
  REJESTR[0] = 0x30;
  REJESTR[1] = 0x30;
  REJESTR[2] = 0x30;
  REJESTR[3] = 0x30;
}

void loop() {
      receiveChar(EEP);
      ReadBufor(EEP, COMMAND);
       ExecuteCommand(COMMAND, REJESTR);
      CheckReg(REJESTR);
}

//Funkcja do odczytywania z portu szeregowego
```

```
//zapisuje znak ASCII do bufora
//bufor jest zmienna globalna
void receiveChar(uint8_t* bufor) {
    static int i = 0;
    char rc;
     while (Serial.available() > 0) {
       rc = Serial.read();
       bufor[i] = rc;
       i = (i + 1) % 16;
    }
}

//Funkcja do sprawdzania poprawnosci odebranej wiadomosci
//w tablicy COMMAND znajduje sie Komenda wraz z CHECKSUMa
//Dodatkowo biore pod uwage stalosc Markera i adresu
bool CheckCHECKSUM(uint8_t* comm, byte l) {
  long sum = ADRESS + l;
  byte temp = l;
  while(l >= 0x02) {
    sum = sum + comm[l - 0x02];
    l--;
  }
  sum = 0xFF ^ sum;
  return (sum == comm[temp - 0x01]);
}

long GenerateCHECKSUM(byte DataL, byte value1, byte value2) {
  long sum = ADRESS + DataL + value1 + value2;
  sum = 0xFF ^ sum;
  return sum;
}

void CheckReg(uint8_t* rej) {
  if(rej[0] == 0x30) {
    return;
  }
  else if(rej[0] == 0x01) {
       digitalWrite(kierSilnikLewy,LOW);
       digitalWrite(kierSilnikPrawy,LOW);
        analogWrite(SilnikLewy,(rej[1]+rej[2]/2));
        analogWrite(SilnikPrawy,(rej[1]-rej[2]/2));
      delay(1000);
       analogWrite(SilnikLewy,0);
       analogWrite(SilnikPrawy,0);
      delay(1000);
  }

}

void ExecuteCommand(uint8_t* comm, uint8_t* rej) {
  if(newData == true) {
    if(comm[0] == 0x01) {
        SendMsg(comm[1], rej[comm[1]]);
    }
    if(comm[0] == 0x02) {
       rej[comm[1]] = comm[2];
    }
       newData = false;
       clearBufer(comm);
       clearBufer(EEP);
  }
  return;
}

void clearBufer(uint8_t* bufor) {
  for( int j = 0 ; j < 16 ; j++)
  {
    bufor[j] = 0x30;
  }
}

void ReadBufor(uint8_t* bufor, uint8_t* comm) {
  static boolean recvInProgress = false;
  static int parser = 0;
  static byte MsgLength = 0;
  static int Cparser = 0;
  byte temp = 0x00;

  while(recvInProgress == true) {
    if(bufor[parser] == ADRESS) {
      parser = (parser + 1) % 16;
       MsgLength = bufor[parser]; //z CHECKSUM/ENDMAKER
      temp = MsgLength;
      DataLength = MsgLength;
      parser = (parser + 1) % 16;
      while(temp > 0x00) {
         comm[Cparser] = bufor[parser];
        Cparser = (Cparser + 1) % 16;
        parser = (parser + 1) % 16;
        temp--;
      }
     //if(CheckCHECKSUM(comm, MsgLength)) {
    //jezeli na koncu paczki jest ENDMAKER jest sygnal o nowej komendzie
     if(CheckCHECKSUM(COMMAND, DataLength)) {
      newData = true;
       recvInProgress = false;
      Cparser = 0;
      return;
    }
    else {
       recvInProgress = false;
      Cparser = 0;
       clearBufer(comm);
      return;
    }
    }
    else {
       recvInProgress = false;
      parser = (parser + 1) % 16;
      return;
    }
  }

  if(bufor[parser] == STARTMARKER && newData == false) {
    recvInProgress = true;
    parser = (parser + 1) % 16;
    return;
  }
  else {
  parser = (parser + 1) % 16;
  return;
  }
}

void SendMsg(byte RegPos, byte value) {
   Serial.write(STARTMARKER);
   Serial.write(ADRESS);
   Serial.write(0x03);
   Serial.write(RegPos);
   Serial.write(value);
   Serial.write(GenerateCHECKSUM(0x03,  RegPos,  value));
}
```