```
#define dataPin 3
#define clockPin 2
byte hamrx[7]={0};
byte bitCount = 0;
int data[3] = {0};
int error = 0;
int p[3] = {0};

void invhamming() {
  p[0] = hamrx[2]^hamrx[4]^hamrx[6];
  p[1] = hamrx[2]^hamrx[5]^hamrx[6];
  p[2] = hamrx[4]^hamrx[5]^hamrx[6];
  data[0] = hamrx[2];
  data[1] = hamrx[4];
  data[2] = hamrx[5];
  data[3] = hamrx[6];

  //Serial.println((String)
p[0]+p[1]+data[0]+p[2]+data[1]+data[2]+data[
3]);
  //Serial.println((String)
p[0]+p[1]+data[0]+p[2]+data[1]+data[2]+data[
3]);
   if((p[0]!=hamrx[0])&&(p[1]!
=hamrx[1])&&(p[2]==hamrx[3])){
```

```
    data[0] = !hamrx[2];
    Serial.println((String)"Error en P1 &
P2 "+ !hamrx[2]);
    error++;
  }

if((p[0]!=hamrx[0])&&(p[1]==hamrx[1])&&(p[2]
!=hamrx[3])){
    data[1] = !hamrx[4];
    Serial.println("Error en P1 & P4");
    error++;
  }

if((p[0]==hamrx[0])&&(p[1]!=hamrx[1])&&(p[2]
!=hamrx[3])){
    data[2] = !hamrx[5];
    Serial.println("Error en P2 & P4");
    error++;
  }

if((p[0]!=hamrx[0])&&(p[1]!=hamrx[1])&&(p[2]
!=hamrx[3])){
    data[3] = !hamrx[6];
    Serial.println("Error en P1, P2 & P4");
    error++;
```

```
  }
  digitalWrite(23,data[0]);
  digitalWrite(25,data[1]);
  digitalWrite(27,data[2]);
  digitalWrite(29,data[3]);
 // Serial.println((String)
p[0]+p[1]+data[0]+p[2]+data[1]+data[2]+data[
3]);
}

void print_vector() {
  Serial.println((String)"ERRORES
DETECTADOS: "+error);
  error = 0;
  for (byte i = 0; i < 4; i++) {
    Serial.print(data[i]);
  }
  Serial.
println("\n_____________________________\n"
);
}

void clock_reciver() {

  byte stateClock = digitalRead(clockPin);
```

```
  if (stateClock == HIGH) {
    hamrx[bitCount] = digitalRead(dataPin);
    Serial.print(hamrx[bitCount]);
    bitCount++;

  }
  if (bitCount == 7) {
    Serial.println();
    invhamming();
    print_vector();
    bitCount = 0;

  }
}
void setup() {
  pinMode(dataPin, INPUT);
  pinMode(clockPin, INPUT);
  pinMode(23,OUTPUT);
  pinMode(25,OUTPUT);
  pinMode(27,OUTPUT);
  pinMode(29,OUTPUT);
  Serial.begin(9600);

attachInterrupt(digitalPinToInterrupt(clockP
in), clock_reciver, HIGH);
```

```
}
void loop() {


}
```