```
#include <Wire.h>
// include the Wire library so you can use its functions:

int t = 0;
int bit = 0;
int datashift = 0;
int i = 0;
int j = 0;
int k = 0;
int found = 1;
int bitValue[] = {
  0, 0, 0, 0, 0 };
int twoBitValue = 0;
int prevBitValue[] = {
  0, 0, 0, 0, 0 };
int startTime[] = {
  0, 0, 0, 0, 0};
int currentTime[] = {
  0, 0, 0, 0, 0};
int upperBitNummer = 0;
int lowerBitNummer = 0;
int valueUpperBit = 0;
int valueLowerBit = 0;
int deltaTime = 0;
int sliderString []= {
  128, 129, 130, 131, 132, 133, 134, 135, 128};
byte value[] = {
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
byte sensor[] = {
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};

// sliderdrawvariables
int upslivalue = 0;
int lowslivalue = 0;
int position = 0;

void setup()  {
  Wire.begin(); // join i2c bus (address optional for master)
  Serial.begin (115200);
}

void loop() {
  Wire.beginTransmission(0x63);
  Wire.send(" ");
  t = 12;
  Wire.send(t);
  Wire.send(4);
  t = 19;
  Wire.send(t);
  Wire.endTransmission();
```

```
// 128 boven_onderlijn vol
Wire.beginTransmission(0x63);
Wire.send(" ");
Wire.send(27);
Wire.send(128);
Wire.send(128);
Wire.send(159);
Wire.send(159);
Wire.send(128);
Wire.send(128);
Wire.send(159);
Wire.send(159);
Wire.send(128);
Wire.endTransmission();

// 129 bovenlijn vol onderlijn half
Wire.beginTransmission(0x63);
Wire.send(" ");
Wire.send(27);
Wire.send(129);
Wire.send(128);
Wire.send(159);
Wire.send(159);
Wire.send(128);
Wire.send(128);
Wire.send(156);
Wire.send(156);
Wire.send(128);
Wire.endTransmission();

// 130 bovenlijn vol onderlijn leeg
Wire.beginTransmission(0x63);
Wire.send(" ");
Wire.send(27);
Wire.send(130);
Wire.send(128);
Wire.send(159);
Wire.send(159);
Wire.send(128);
Wire.send(128);
Wire.send(128);
Wire.send(128);
Wire.send(128);
Wire.endTransmission();

// 131 bovenlijn half onderlijn vol
Wire.beginTransmission(0x63);
Wire.send(" ");
Wire.send(27);
```

```
Wire.send(131);
Wire.send(128);
Wire.send(156);
Wire.send(156);
Wire.send(128);
Wire.send(128);
Wire.send(159);
Wire.send(159);
Wire.send(128);
Wire.endTransmission();

// 132 bovenlijn half onderlijn half
Wire.beginTransmission(0x63);
Wire.send(" ");
Wire.send(27);
Wire.send(132);
Wire.send(128);
Wire.send(156);
Wire.send(156);
Wire.send(128);
Wire.send(128);
Wire.send(156);
Wire.send(156);
Wire.send(128);
Wire.endTransmission();

// 133 bovenlijn half onderlijn leeg
Wire.beginTransmission(0x63);
Wire.send(" ");
Wire.send(27);
Wire.send(133);
Wire.send(128);
Wire.send(156);
Wire.send(156);
Wire.send(128);
Wire.send(128);
Wire.send(128);
Wire.send(128);
Wire.send(128);
Wire.endTransmission();

// 134 bovenlijn leeg onderlijn vol
Wire.beginTransmission(0x63);
Wire.send(" ");
Wire.send(27);
Wire.send(134);
Wire.send(128);
Wire.send(128);
Wire.send(128);
Wire.send(128);
```

```
  Wire.send(128);
  Wire.send(159);
  Wire.send(159);
  Wire.send(128);
  Wire.endTransmission();

  // 135 bovenlijn leeg onderlijn half
  Wire.beginTransmission(0x63);
  Wire.send(" ");
  Wire.send(27);
  Wire.send(135);
  Wire.send(128);
  Wire.send(128);
  Wire.send(128);
  Wire.send(128);
  Wire.send(128);
  Wire.send(156);
  Wire.send(156);
  Wire.send(128);
  Wire.endTransmission();

  while (t == 19) {
   if (Serial.available() > 0) {
     while (found == 0) {
     for (int i = 0; i <= 16; i++)
     {
       sensor[i] = Serial.read();
     }
     Serial.flush();

   if(sensor[15] == 126) {
     if(sensor[16] == 127) {
       if(sensor[0] == 253) {
       datashift = 16;
       found = 1;
     }
   }
   }

   if(sensor[16] == 126) {
     if(sensor[0] == 127) {
       if(sensor[1] == 253) {
       datashift = 15;
       found = 1;
     }
   }
   }

    for (int i = 0; i <= 16; i++)      // Look for the combination 126, 127, 253 (closing
bytes of the measurement string)
```

```
{
 if(sensor[i] == 126) {
  if(sensor[i+1] == 127) {
   if(sensor[i+2] == 253) {
   datashift = 14 - i;
   found = 1;
  }
  }
 }
}
delay(4);
}
 found = 0;


// pijl die aangeeft of het upper of lower register actief is
j = (30 - datashift)%17;
bitValue[0] = sensor[j] & 1;
if ( bitValue[0] == 1 ) {
 Wire.beginTransmission(0x63);
 Wire.send(" ");
 Wire.send(2);
 Wire.send(9);
 Wire.send(126);
 Wire.endTransmission();
}
else
{
 Wire.beginTransmission(0x63);
 Wire.send(" ");
 Wire.send(2);
 Wire.send(9);
 Wire.send(127);
 Wire.endTransmission();
}

// Are the toggles on or off?
for (int i = 0; i <= 3; i++) {
 bit = 1 << i;
 j = (26 - datashift)%17;
 bitValue[i] = sensor[j] & bit;
 if ( bitValue[i] == bit ) {
  Wire.beginTransmission(0x63);
  Wire.send(" ");
  Wire.send(2);
  Wire.send((-i)*2 + 7);
  Wire.send("X");
  Wire.endTransmission();
 }
 else
```

```
    {
      Wire.beginTransmission(0x63);
      Wire.send(" ");
      Wire.send(2);
      Wire.send((-i)*2 + 7);
      Wire.send("O");
      Wire.endTransmission();
    }
   }

   for (int i = 0; i <= 3; i++) {
    bit = 3 << (i*2);
    j = (25 - datashift)%17;
    twoBitValue = (sensor[j] & bit) >> (i*2);
      Wire.beginTransmission(0x63);
      Wire.send(" ");
      Wire.send(2);
      Wire.send((-i)*2 + 17);
      Wire.send(48+twoBitValue);
      Wire.endTransmission();
    }

   // first two potmeters
   j = (18 - datashift)%17;
   k = (19 - datashift)%17;
   upperBitNummer = 3;
   lowerBitNummer = 2;
   valueUpperBit = 8;
   valueLowerBit = 4;
   position = 21;

   if (bitValue[0] == 0) {              // 0 = record mode & 1 = play mode
     if (bitValue[upperBitNummer] == valueUpperBit &
prevBitValue[upperBitNummer] == valueUpperBit ) {
      deltaTime = millis() - startTime [upperBitNummer];
      currentTime[upperBitNummer] = constrain(deltaTime, 0, 6000);
      upslivalue = currentTime[upperBitNummer]/375 - (sensor[j] >> 4);
     }

     if (bitValue[upperBitNummer] == valueUpperBit &
prevBitValue[upperBitNummer] == 0 ) {
      startTime [upperBitNummer] = millis();
      upslivalue = 1;
     }

     if (bitValue[upperBitNummer] == 0 ) {
      upslivalue = currentTime[upperBitNummer]/375 - (sensor[j] >> 4);
     }
```

```
    if (bitValue[lowerBitNummer] == valueLowerBit &
prevBitValue[lowerBitNummer] == valueLowerBit ) {
      deltaTime = millis() - startTime [lowerBitNummer];
      currentTime[lowerBitNummer] = constrain(deltaTime, 0, 6000);
      lowslivalue = currentTime[lowerBitNummer]/375 - (sensor[k] >> 4);
    }

    if (bitValue[lowerBitNummer] == valueLowerBit &
prevBitValue[lowerBitNummer] == 0 ) {
      startTime [lowerBitNummer] = millis();
      lowslivalue = 1;
    }

    if (bitValue[lowerBitNummer] == 0 ) {
      lowslivalue = currentTime[lowerBitNummer]/375 - (sensor[k] >> 4);
    }

    if (upslivalue == 0 & lowslivalue == 0) {
      Wire.beginTransmission(0x63);
      Wire.send(" ");
      Wire.send(2);
      Wire.send(position);
      Wire.send("buffer12");
      Wire.endTransmission();
    }
    else
    {
      twofaderdraw(upslivalue, lowslivalue, position);
    }
  }
  else
  {
    lowslivalue = currentTime[lowerBitNummer]/375 - (sensor[k] >> 4);
    upslivalue = currentTime[upperBitNummer]/375 - (sensor[j] >> 4);
    twofaderdraw(upslivalue, lowslivalue, position);
  }



  prevBitValue[upperBitNummer] = bitValue[upperBitNummer];
  prevBitValue[lowerBitNummer] = bitValue[lowerBitNummer];

  // second two potmeters
  j = (20 - datashift)%17;
  k = (21 - datashift)%17;
  upperBitNummer = 1;
  lowerBitNummer = 0;
  valueUpperBit = 2;
  valueLowerBit = 1;
  position = 41;
```

```
   if (bitValue[0] == 0) {              // 0 = record mode & 1 = play mode
     if (bitValue[upperBitNummer] == valueUpperBit &
prevBitValue[upperBitNummer] == valueUpperBit ) {
       deltaTime = millis() - startTime [upperBitNummer];
       currentTime[upperBitNummer] = constrain(deltaTime, 0, 6000);
       upslivalue = currentTime[upperBitNummer]/375 - (sensor[j] >> 4);
     }

     if (bitValue[upperBitNummer] == valueUpperBit &
prevBitValue[upperBitNummer] == 0 ) {
       startTime [upperBitNummer] = millis();
       upslivalue = 1;
     }

     if (bitValue[upperBitNummer] == 0 ) {
       upslivalue = currentTime[upperBitNummer]/375 - (sensor[j] >> 4);
     }

     if (bitValue[lowerBitNummer] == valueLowerBit &
prevBitValue[lowerBitNummer] == valueLowerBit ) {
       deltaTime = millis() - startTime [lowerBitNummer];
       currentTime[lowerBitNummer] = constrain(deltaTime, 0, 6000);
       lowslivalue = currentTime[lowerBitNummer]/375 - (sensor[k] >> 4);
     }

     if (bitValue[lowerBitNummer] == valueLowerBit &
prevBitValue[lowerBitNummer] == 0 ) {
       startTime [lowerBitNummer] = millis();
       lowslivalue = 1;
     }

     if (bitValue[lowerBitNummer] == 0 ) {
       lowslivalue = currentTime[lowerBitNummer]/375 - (sensor[k] >> 4);
     }

     if (upslivalue == 0 & lowslivalue == 0) {
       Wire.beginTransmission(0x63);
       Wire.send(" ");
       Wire.send(2);
       Wire.send(position);
       Wire.send("buffer34");
       Wire.endTransmission();
     }
     else
     {
       twofaderdraw(upslivalue, lowslivalue, position);
     }
   }
   else
```

```
{
  lowslivalue = currentTime[lowerBitNummer]/375 - (sensor[k] >> 4);
  upslivalue = currentTime[upperBitNummer]/375 - (sensor[j] >> 4);
  twofaderdraw(upslivalue, lowslivalue, position);
}

prevBitValue[upperBitNummer] = bitValue[upperBitNummer];
prevBitValue[lowerBitNummer] = bitValue[lowerBitNummer];

// Ribboncontroller
j = (24 - datashift)%17;
position = 61;
if (sensor[j] == 0) {
  Wire.beginTransmission(0x63);
  Wire.send(" ");
  Wire.send(2);
  Wire.send(position);
  Wire.send("Ribonctr");
  Wire.endTransmission();
}
else
{
  upslivalue = sensor[j] >> 4;
  lowslivalue = upslivalue;
  twofaderdraw(upslivalue, lowslivalue, position);
}

// pressure sensors
j = (23 - datashift)%17;
k = (22 - datashift)%17;
position = 31;
if (sensor[j] == 0 & sensor[k] == 0) {
  Wire.beginTransmission(0x63);
  Wire.send(" ");
  Wire.send(2);
  Wire.send(position);
  Wire.send("up&lowpr");
  Wire.endTransmission();
}
else
{
  upslivalue = sensor[j] >> 4;
  lowslivalue = sensor[k] >> 4;
  twofaderdraw(upslivalue, lowslivalue, position);
}

// upperslide
k = (27 - datashift)%17;
position = 51;
if (sensor[k] == 0) {
```

```
      Wire.beginTransmission(0x63);
      Wire.send(" ");
      Wire.send(2);
      Wire.send(position);
      Wire.send("uppersli");
      Wire.endTransmission();
    }
    else
    {
      upslivalue = 0;
      lowslivalue = sensor[k] >> 4;
      twofaderdraw(upslivalue, lowslivalue, position);
    }
    // middleslide & underslide
    j = (28 - datashift)%17;
    k = (29 - datashift)%17;
    position = 71;
    if (sensor[j] == 0 & sensor[k] == 0) {
      Wire.beginTransmission(0x63);
      Wire.send(" ");
      Wire.send(2);
      Wire.send(position);
      Wire.send("mi&unsli");
      Wire.endTransmission();
    }
    else
    {
      upslivalue = sensor[j] >> 4;
      lowslivalue = sensor[k] >> 4;
      twofaderdraw(upslivalue, lowslivalue, position);
    }
    delay (50);
  }
}
}

void twofaderdraw(int upslivalue, int lowslivalue, int position) {

  int bottomvalue = 0;
  //Twee druksensoren weergeven. Delen door 16 geeft acht vakjes.

  for ( i = 0; i <= 7; i++) {
    bottomvalue = i * 2;
    if ((upslivalue - bottomvalue) <= 0 & (lowslivalue - bottomvalue) <= 0 ) {
      sliderString[i] = 32;
    }
    if ((upslivalue - bottomvalue) <= 0 & (lowslivalue - bottomvalue) == 1 ) {
      sliderString[i] = 135;
    }
    if ((upslivalue - bottomvalue) <= 0 & (lowslivalue - bottomvalue) >= 2 ) {
```

```
    sliderString[i] = 134;
   }
   if ((upslivalue - bottomvalue) == 1 & (lowslivalue - bottomvalue) <= 0 ) {
    sliderString[i] = 133;
   }
   if ((upslivalue - bottomvalue) == 1 & (lowslivalue - bottomvalue) == 1 ) {
    sliderString[i] = 132;
   }
   if ((upslivalue - bottomvalue) == 1 & (lowslivalue - bottomvalue) >= 2 ) {
    sliderString[i] = 131;
   }
   if ((upslivalue - bottomvalue) >= 2 & (lowslivalue - bottomvalue) <= 0 ) {
    sliderString[i] = 130;
   }
   if ((upslivalue - bottomvalue) >= 2 & (lowslivalue - bottomvalue) == 1 ) {
    sliderString[i] = 129;
   }
   if ((upslivalue - bottomvalue) >= 2 & (lowslivalue - bottomvalue) >= 2 ) {
    sliderString[i] = 128;
   }
  }

  Wire.beginTransmission(0x63);
  Wire.send(" ");
  Wire.send(2);
  Wire.send(position);
  Wire.send(sliderString[0]);
  Wire.send(sliderString[1]);
  Wire.send(sliderString[2]);
  Wire.send(sliderString[3]);
  Wire.send(sliderString[4]);
  Wire.send(sliderString[5]);
  Wire.send(sliderString[6]);
  Wire.send(sliderString[7]);
  Wire.endTransmission();

}
```