```
/*
 *  Fast Serialsender
 *  Send pin values from Arduino to MAX/MSP
 *  This sketch is only suitable for nonpolling data sending from the arduino to the computer.
 *  It was designed for use in (almost) zero latency music applications
 *
 *  fastarduinoserialsender.pde
 *  ------------
 *  This version: .1, December 2007
 *  ------------
 *  Copyleft: use as you like
 *  by Hans Leeuw
 *  Based on Arduino2Max from Daniel Jolliffe
 *  Which was based on a sketch and patch by Thomas Ouellet Fredericks  tof.danslchamp.org
 *
 */

byte t = 0;                             // constant that gets #0 when the arduino has to sent data
int measurementnr = 0;
byte switchButton = 0;
byte button[9] = {
  0, 0, 0, 0, 0, 0, 0, 0, 0};     // remember last digital read-out
byte d[9] = {
  0, 0, 0, 0, 0, 0, 0, 0, 0};             // holding digital pin values
char x = 0;                             // a place to hold digital pin values
char x1 = 0;
char x2 = 0;
int y = 0;                              // a place to hold analog pin values
int Rib = 0;                            // a place to hold Ribbon control_value
int Rib1 = 0;                           // a place to hold former analog pin values
int Rib11 = 0;                          // a place to hold former former analog pin values
int Rib111 = 0;                         // a place to hold former former former analog pin values
int hoogMeer = 0;
int hoogMinder = 0;
int laagMeer = 0;
int laagMinder = 0;
int upperSlide = 0;
int middleSlide = 0;
int underSlide = 0;
char mask = 15;
long lastTime = 0;                      // a place to hold former former former analog pin values
int hoog = 4;                           // Voor het aangeven van het hoge register
int laag = 0;                           // Voor het aangeven van het lage register
int regist = 0;                         // Voor het vasthouden van het register
byte z = 0;                             // loopconstant
char q = 0;                             // byteconstant
byte countdown [9] = {
  0, 0, 0, 0, 0, 0, 0, 0, 0};  // made high to prevent doubble clicks
char sensorValues[18] = {
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 126, 127, 253 };
```

```
void setup()
{
 Serial.begin(115200);              // 115200 is the default Arduino Bluetooth speed
}

void loop() {
 if (Serial.available() > 0)         // Check serial buffer for characters
 {
  if (Serial.read() == 'r') {
   t++;
  }
 }
 if (t > 0) {
  for (measurementnr = 0; measurementnr <= 255; measurementnr++)
  {
   sensorValues[0] = measurementnr;
   for (z = 0; z <= 3; z++)
   {
    y = analogRead(z);              // read an analog pin
    sensorValues[z+1] = y >> 2 ;        // print y after deviding it by 4 so it will fit in one byte

   }

   y = analogRead(4);              // read an analog pin
   y = 360 -y;
   y  = constrain(y,  0, 127);
   sensorValues[5] = y << 1 ;       // print y after multiplying it by 2 so it will fill one byte

   y = analogRead(5);               // read an analog pin
   y = 360 -y;
   y  = constrain(y,  0, 127);
   sensorValues[6] = y << 1 ;       // print y after deviding it by 2 so it will fit in one byte


   Rib = analogRead(6);             // read an analog pin
   if (Rib <= 400) {
    Rib11 = Rib111;
    Rib1 = Rib11;
    Rib = Rib1;
    x = 0;                      // first digital pin is 0 (ribbon_controller wordt niet aangeraakt)
    sensorValues[7] = Rib >> 1;

   }
   else
   {
    Rib = constrain(Rib, 512, 1023);
    Rib = Rib - 512;
    Rib111 = Rib11;
    Rib11 = Rib1;
    Rib1 = Rib;
    x = 1;                        // first digital pin is 1 (ribbon_controller wordt aangeraakt)
    sensorValues[7] = Rib >> 1;        // print y after deviding it by 2 so it will fit in one byte

   }
```

```
if (digitalRead(6) == 1)
    {
     if (switchButton == 0)
     {
       switchButton = 1;
       d[0] = 1 - d[0];
     }
    }
    if (digitalRead(6) == 0) {
     if (switchButton == 1) {
       switchButton = 0;
     };
    }

    if (d[0] == 1)
    {
     regist = hoog;
     x2 = 0;
    }
    if (d[0] == 0)
    {
     regist = laag;
     x1 = 0;
    }


    for (int z = 1; z <= 4; z++)
    {
     if (digitalRead(z+1) == 1)
     {
       if (button[z+regist] == 0)
       {
         button[z+regist] = 1;
         if (regist == laag){
         d[z+regist] = 1 - d[z+regist];
         }
         if (regist == hoog){
         d[z+regist] = (d[z+regist] + 1)%4;
         }
       }
     }

     if (digitalRead(z+1) == 0) {
       if (button[z+regist] == 1) {
         button[z+regist] = 0;
       };
     }
     if (regist == laag){
     x1 = x1 << 1;                  // bitshift to the left (same as multiply by 2 but faster)
     x1 = x1 + d[z+regist];                 // read the digital pin and add to x
     }
     if (regist == hoog){
     x2 = x2 << 2;                  // bitshift to the left (same as multiply by 4 but faster)
     x2 = x2 + d[z+regist];                 // read the digital pin and add to x
     }
    }

    sensorValues[9] = x1;  // print x as a single byte that contains the values of 4 digital pins
    sensorValues[8] = x2; // print y as a single byte that contains the values of 4 2bits knobs
```

```
      //slider buttons
      hoogMeer = digitalRead(9);
      hoogMinder   = digitalRead(10);
      laagMeer  = digitalRead(11);
      laagMinder  = digitalRead(12);

      if (hoogMeer == 1 & laagMeer == 1) {
        middleSlide = ++middleSlide;
      }

      if (hoogMeer == 1 & laagMeer == 0) {
        upperSlide = ++upperSlide;
      }

      if (hoogMeer == 0 & laagMeer == 1) {
        underSlide = ++underSlide;
      }

      if (hoogMinder == 1 & laagMinder == 1) {
        middleSlide = --middleSlide;
      }

      if (hoogMinder == 1 & laagMinder == 0) {
        upperSlide = --upperSlide;
      }

      if (hoogMinder == 0 & laagMinder == 1) {
        underSlide = --underSlide;
      }

      upperSlide  = constrain(upperSlide,  0, 511);
      middleSlide = constrain(middleSlide, 0, 511);
      underSlide  = constrain(underSlide,  0, 511);

      sensorValues[10] = upperSlide >> 1;
      sensorValues[11] = middleSlide >> 1;
      sensorValues[12] = underSlide >> 1;
      sensorValues[13] = d[0];

      if (lastTime == 0) {
        lastTime = millis();
      }

      while (millis() <= lastTime) {
        delayMicroseconds(100);
      }
      lastTime = lastTime + 2.5;

      for (int i = 0; i <= 16; i++) {
        Serial.print(sensorValues[i]);
      }
    }
  }
}
```