#include <Adafruit\_Fingerprint.h>

#if (defined(\_\_AVR\_\_) || defined(ESP8266)) && !defined(\_\_AVR\_ATmega2560\_\_)

// For UNO and others without hardware serial, we must use software serial...

// pin #2 is IN from sensor (GREEN wire)

// pin #3 is OUT from arduino  (WHITE wire)

// Set up the serial port to use softwareserial..

SoftwareSerial mySerial(2, 3);

#else

// On Leonardo/M0/etc, others with hardware serial, use hardware serial!

// #0 is green wire, #1 is white

#define mySerial Serial1

#endif

Adafruit\_Fingerprint finger = Adafruit\_Fingerprint(&mySerial);

int getFingerprintIDez();

void setup()

{

  while (!Serial);

  Serial.begin(9600);

  Serial.println("Fingerprint template extractor");

  // set the data rate for the sensor serial port

  finger.begin(57600);

  if (finger.verifyPassword()) {

    Serial.println("Found fingerprint sensor!");

  } else {

    Serial.println("Did not find fingerprint sensor :(");

    while (1);

  }

  // Try to get the templates for fingers 1 through 10

  for (int finger = 1; finger < 10; finger++) {

    downloadFingerprintTemplate(finger);

  }

}

uint8\_t downloadFingerprintTemplate(uint16\_t id)

{

  Serial.println("------------------------------------");

  Serial.print("Attempting to load #"); Serial.println(id);

  uint8\_t p = finger.loadModel(id);

  switch (p) {

    case FINGERPRINT\_OK:

      Serial.print("Template "); Serial.print(id); Serial.println(" loaded");

      break;

    case FINGERPRINT\_PACKETRECIEVEERR:

      Serial.println("Communication error");

      return p;

    default:

      Serial.print("Unknown error "); Serial.println(p);

      return p;

  }

  // OK success!

  Serial.print("Attempting to get #"); Serial.println(id);

  p = finger.getModel();

  switch (p) {

    case FINGERPRINT\_OK:

      Serial.print("Template "); Serial.print(id); Serial.println(" transferring:");

      break;

    default:

      Serial.print("Unknown error "); Serial.println(p);

      return p;

  }

  // one data packet is 267 bytes. in one data packet, 11 bytes are 'usesless' :D

  uint8\_t bytesReceived[534]; // 2 data packets

  memset(bytesReceived, 0xff, 534);

  uint32\_t starttime = millis();

  int i = 0;

  while (i < 534 && (millis() - starttime) < 20000) {

    if (mySerial.available()) {

      bytesReceived[i++] = mySerial.read();

    }

  }

  Serial.print(i); Serial.println(" bytes read.");

  Serial.println("Decoding packet...");

  uint8\_t fingerTemplate[512]; // the real template

  memset(fingerTemplate, 0xff, 512);

  // filtering only the data packets

  int uindx = 9, index = 0;

  memcpy(fingerTemplate + index, bytesReceived + uindx, 256);   // first 256 bytes

  uindx += 256;       // skip data

  uindx += 2;         // skip checksum

  uindx += 9;         // skip next header

  index += 256;       // advance pointer

  memcpy(fingerTemplate + index, bytesReceived + uindx, 256);   // second 256 bytes

  for (int i = 0; i < 512; ++i) {

    //Serial.print("0x");

    printHex(fingerTemplate[i], 2);

    //Serial.print(", ");

  }

  Serial.println("\ndone.");

  return p;

  /\*

    uint8\_t templateBuffer[256];

    memset(templateBuffer, 0xff, 256);  //zero out template buffer

    int index=0;

    uint32\_t starttime = millis();

    while ((index < 256) && ((millis() - starttime) < 1000))

    {

    if (mySerial.available())

    {

      templateBuffer[index] = mySerial.read();

      index++;

    }

    }

    Serial.print(index); Serial.println(" bytes read");

    //dump entire templateBuffer.  This prints out 16 lines of 16 bytes

    for (int count= 0; count < 16; count++)

    {

    for (int i = 0; i < 16; i++)

    {

      Serial.print("0x");

      Serial.print(templateBuffer[count\*16+i], HEX);

      Serial.print(", ");

    }

    Serial.println();

    }\*/

}

void printHex(int num, int precision) {

  char tmp[16];

  char format[128];

  sprintf(format, "%%.%dX", precision);

  sprintf(tmp, format, num);

  Serial.print(tmp);

}

void loop()

{}