#ifndef ADAFRUIT\_FINGERPRINT\_H

#define ADAFRUIT\_FINGERPRINT\_H

/\*!

\* @file Adafruit\_Fingerprint.h

\*/

#include "Arduino.h"

#if defined(\_\_AVR\_\_) || defined(ESP8266)

#include <SoftwareSerial.h>

#elif defined(FREEDOM\_E300\_HIFIVE1)

#include <SoftwareSerial32.h>

#define SoftwareSerial SoftwareSerial32

#endif

#define FINGERPRINT\_OK 0x00 //!< Command execution is complete

#define FINGERPRINT\_PACKETRECIEVEERR 0x01 //!< Error when receiving data package

#define FINGERPRINT\_NOFINGER 0x02 //!< No finger on the sensor

#define FINGERPRINT\_IMAGEFAIL 0x03 //!< Failed to enroll the finger

#define FINGERPRINT\_IMAGEMESS \

0x06 //!< Failed to generate character file due to overly disorderly

//!< fingerprint image

#define FINGERPRINT\_FEATUREFAIL \

0x07 //!< Failed to generate character file due to the lack of character point

//!< or small fingerprint image

#define FINGERPRINT\_NOMATCH 0x08 //!< Finger doesn't match

#define FINGERPRINT\_NOTFOUND 0x09 //!< Failed to find matching finger

#define FINGERPRINT\_ENROLLMISMATCH \

0x0A //!< Failed to combine the character files

#define FINGERPRINT\_BADLOCATION \

0x0B //!< Addressed PageID is beyond the finger library

#define FINGERPRINT\_DBRANGEFAIL \

0x0C //!< Error when reading template from library or invalid template

#define FINGERPRINT\_UPLOADFEATUREFAIL 0x0D //!< Error when uploading template

#define FINGERPRINT\_PACKETRESPONSEFAIL \

0x0E //!< Module failed to receive the following data packages

#define FINGERPRINT\_UPLOADFAIL 0x0F //!< Error when uploading image

#define FINGERPRINT\_DELETEFAIL 0x10 //!< Failed to delete the template

#define FINGERPRINT\_DBCLEARFAIL 0x11 //!< Failed to clear finger library

#define FINGERPRINT\_PASSFAIL \

0x13 //!< Find whether the fingerprint passed or failed

#define FINGERPRINT\_INVALIDIMAGE \

0x15 //!< Failed to generate image because of lac of valid primary image

#define FINGERPRINT\_FLASHERR 0x18 //!< Error when writing flash

#define FINGERPRINT\_INVALIDREG 0x1A //!< Invalid register number

#define FINGERPRINT\_ADDRCODE 0x20 //!< Address code

#define FINGERPRINT\_PASSVERIFY 0x21 //!< Verify the fingerprint passed

#define FINGERPRINT\_STARTCODE \

0xEF01 //!< Fixed falue of EF01H; High byte transferred first

#define FINGERPRINT\_COMMANDPACKET 0x1 //!< Command packet

#define FINGERPRINT\_DATAPACKET \

0x2 //!< Data packet, must follow command packet or acknowledge packet

#define FINGERPRINT\_ACKPACKET 0x7 //!< Acknowledge packet

#define FINGERPRINT\_ENDDATAPACKET 0x8 //!< End of data packet

#define FINGERPRINT\_TIMEOUT 0xFF //!< Timeout was reached

#define FINGERPRINT\_BADPACKET 0xFE //!< Bad packet was sent

#define FINGERPRINT\_GETIMAGE 0x01 //!< Collect finger image

#define FINGERPRINT\_IMAGE2TZ 0x02 //!< Generate character file from image

#define FINGERPRINT\_SEARCH 0x04 //!< Search for fingerprint in slot

#define FINGERPRINT\_REGMODEL \

0x05 //!< Combine character files and generate template

#define FINGERPRINT\_STORE 0x06 //!< Store template

#define FINGERPRINT\_LOAD 0x07 //!< Read/load template

#define FINGERPRINT\_UPLOAD 0x08 //!< Upload template

#define FINGERPRINT\_DELETE 0x0C //!< Delete templates

#define FINGERPRINT\_EMPTY 0x0D //!< Empty library

#define FINGERPRINT\_READSYSPARAM 0x0F //!< Read system parameters

#define FINGERPRINT\_SETPASSWORD 0x12 //!< Sets passwords

#define FINGERPRINT\_VERIFYPASSWORD 0x13 //!< Verifies the password

#define FINGERPRINT\_HISPEEDSEARCH \

0x1B //!< Asks the sensor to search for a matching fingerprint template to the

//!< last model generated

#define FINGERPRINT\_TEMPLATECOUNT 0x1D //!< Read finger template numbers

#define FINGERPRINT\_AURALEDCONFIG 0x35 //!< Aura LED control

#define FINGERPRINT\_LEDON 0x50 //!< Turn on the onboard LED

#define FINGERPRINT\_LEDOFF 0x51 //!< Turn off the onboard LED

#define FINGERPRINT\_LED\_BREATHING 0x01 //!< Breathing light

#define FINGERPRINT\_LED\_FLASHING 0x02 //!< Flashing light

#define FINGERPRINT\_LED\_ON 0x03 //!< Always on

#define FINGERPRINT\_LED\_OFF 0x04 //!< Always off

#define FINGERPRINT\_LED\_GRADUAL\_ON 0x05 //!< Gradually on

#define FINGERPRINT\_LED\_GRADUAL\_OFF 0x06 //!< Gradually off

#define FINGERPRINT\_LED\_RED 0x01 //!< Red LED

#define FINGERPRINT\_LED\_BLUE 0x02 //!< Blue LED

#define FINGERPRINT\_LED\_PURPLE 0x03 //!< Purple LED

//#define FINGERPRINT\_DEBUG

#define DEFAULTTIMEOUT 1000 //!< UART reading timeout in milliseconds

///! Helper class to craft UART packets

struct Adafruit\_Fingerprint\_Packet {

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*!

@brief Create a new UART-borne packet

@param type Command, data, ack type packet

@param length Size of payload

@param data Pointer to bytes of size length we will memcopy into the

internal buffer

\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Adafruit\_Fingerprint\_Packet(uint8\_t type, uint16\_t length, uint8\_t \*data) {

this->start\_code = FINGERPRINT\_STARTCODE;

this->type = type;

this->length = length;

address[0] = 0xFF;

address[1] = 0xFF;

address[2] = 0xFF;

address[3] = 0xFF;

if (length < 64)

memcpy(this->data, data, length);

else

memcpy(this->data, data, 64);

}

uint16\_t start\_code; ///< "Wakeup" code for packet detection

uint8\_t address[4]; ///< 32-bit Fingerprint sensor address

uint8\_t type; ///< Type of packet

uint16\_t length; ///< Length of packet

uint8\_t data[64]; ///< The raw buffer for packet payload

};

///! Helper class to communicate with and keep state for fingerprint sensors

class Adafruit\_Fingerprint {

public:

#if defined(\_\_AVR\_\_) || defined(ESP8266) || defined(FREEDOM\_E300\_HIFIVE1)

Adafruit\_Fingerprint(SoftwareSerial \*ss, uint32\_t password = 0x0);

#endif

Adafruit\_Fingerprint(HardwareSerial \*hs, uint32\_t password = 0x0);

Adafruit\_Fingerprint(Stream \*serial, uint32\_t password = 0x0);

void begin(uint32\_t baud);

boolean verifyPassword(void);

uint8\_t getParameters(void);

uint8\_t getImage(void);

uint8\_t image2Tz(uint8\_t slot = 1);

uint8\_t createModel(void);

uint8\_t emptyDatabase(void);

uint8\_t storeModel(uint16\_t id);

uint8\_t loadModel(uint16\_t id);

uint8\_t getModel(void);

uint8\_t deleteModel(uint16\_t id);

uint8\_t fingerFastSearch(void);

uint8\_t fingerSearch(uint8\_t slot = 1);

uint8\_t getTemplateCount(void);

uint8\_t setPassword(uint32\_t password);

uint8\_t LEDcontrol(bool on);

uint8\_t LEDcontrol(uint8\_t control, uint8\_t speed, uint8\_t coloridx,

uint8\_t count = 0);

void writeStructuredPacket(const Adafruit\_Fingerprint\_Packet &p);

uint8\_t getStructuredPacket(Adafruit\_Fingerprint\_Packet \*p,

uint16\_t timeout = DEFAULTTIMEOUT);

/// The matching location that is set by fingerFastSearch()

uint16\_t fingerID;

/// The confidence of the fingerFastSearch() match, higher numbers are more

/// confidents

uint16\_t confidence;

/// The number of stored templates in the sensor, set by getTemplateCount()

uint16\_t templateCount;

uint16\_t status\_reg = 0x0; ///< The status register (set by getParameters)

uint16\_t system\_id = 0x0; ///< The system identifier (set by getParameters)

uint16\_t capacity = 64; ///< The fingerprint capacity (set by getParameters)

uint16\_t security\_level = 0; ///< The security level (set by getParameters)

uint32\_t device\_addr =

0xFFFFFFFF; ///< The device address (set by getParameters)

uint16\_t packet\_len = 64; ///< The max packet length (set by getParameters)

uint16\_t baud\_rate = 57600; ///< The UART baud rate (set by getParameters)

private:

uint8\_t checkPassword(void);

uint32\_t thePassword;

uint32\_t theAddress;

uint8\_t recvPacket[20];

Stream \*mySerial;

#if defined(\_\_AVR\_\_) || defined(ESP8266) || defined(FREEDOM\_E300\_HIFIVE1)

SoftwareSerial \*swSerial;

#endif

HardwareSerial \*hwSerial;

};

#endif