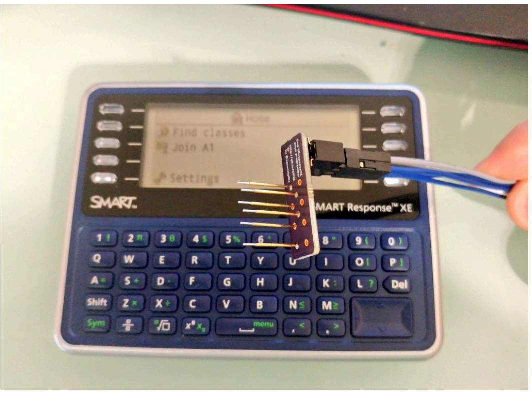
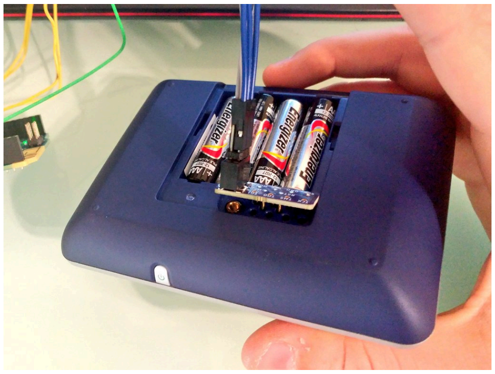
SmartResponse XE to Arduboy Project

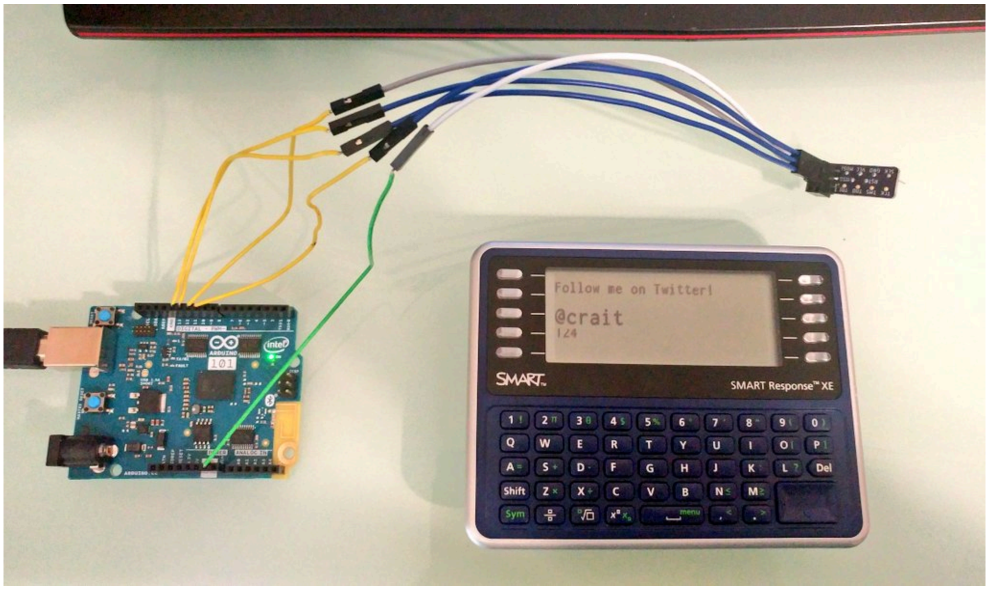
Parts List for Pogo-to-ICSP header adapter (https://github.com/securelyfitz/smartresponse):

- PCB from Osh Park: https://oshpark.com/shared_projects/1FKeJ3AI
- P50 pogo pins: https://www.ebay.ca/itm/100x-P50-B1-Dia-0-68mm-Length-16mm-75g-Spring-Pressure-Test-Probe-Pogo-Pin-F-Ch/392189237143?ssPageName=STRK%3AMEBIDX%3AIT&_trksid=p2057872.m2749.l2649
- ICSP 2x3 smt pins: Amphenol 54202-G08-03 or similar

  or: ICSP 2x3 smt socket: Harwin M20-7870346 or similar https://www.ebay.ca/itm/10pcs-1-27mm-Pitch-2x3-Pin-6-Pin-Female-Double-Row-SMT-SMD-Pin-Header-Strip-PCB/182933532983?ssPageName=STRK%3AMEBIDX%3AIT&_trksid=p2057872.m2749.l2649







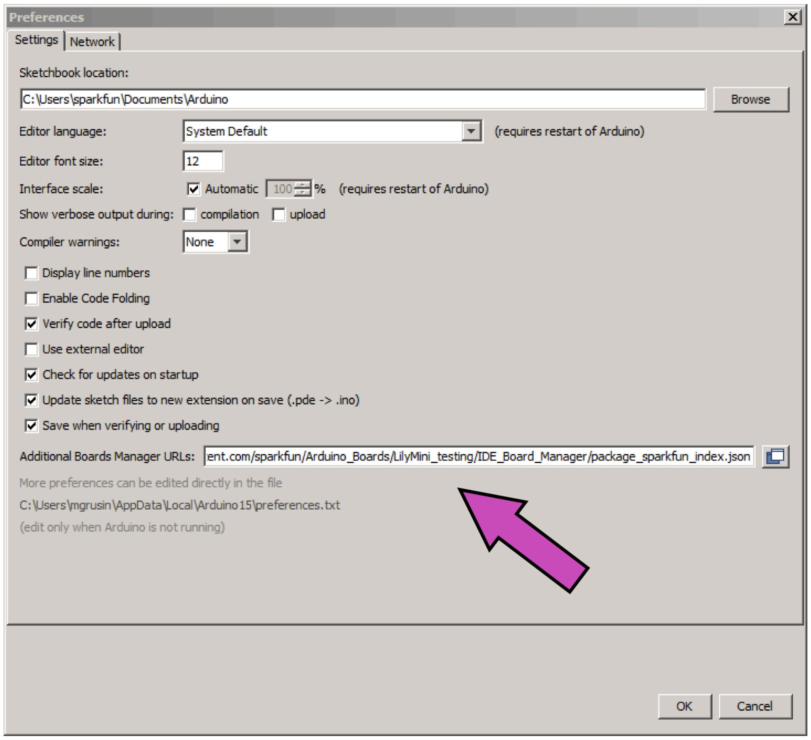
Important Links

https://www.hackster.io/news/classroom-tool-converted-into-an-arduboy-df7908d9ee8c
https://community.arduboy.com/t/smart-response-xe-re-purposed-into-arduboy/6094/411

Links from Joe Fitz
https://github.com/pdxbadgers/5ohBEE-2019
https://github.com/pdxbadgers/ATmega128RFA1_Dev

Setting up the Board

1. If using Linux you need to adjust the Udev rules to give access to the programmer and tell who the owner & permissions of the device will be (skip if using Windows or Arduino-as-isp).
   a. udev rules: copy helpers/99-USBasp.rules to /etc/udev/rules.d/ (this assumes you are using the USBasp programmer, update your IDE accordingly)
2. We need to Add the board to Arduino IDE (https://github.com/sparkfun/arduino_boards)
   a. Open Arduino IDE, go to File -> Preferences
   b. Add this link to the Additional Board Manager URLs:
   c. https://raw.githubusercontent.com/sparkfun/Arduino_Boards/master/IDE_Board_Manager/package_sparkfun_index.json



Sketchbook location: C:\Users\sparkfun\Documents\Arduino

Editor language: System Default (requires restart of Arduino)

Editor font size: 12

Interface scale: ☑ Automatic 100 % (requires restart of Arduino)

Show verbose output during: □ compilation □ upload

Compiler warnings: None

□ Display line numbers

□ Enable Code Folding

☑ Verify code after upload

□ Use external editor

☑ Check for updates on startup

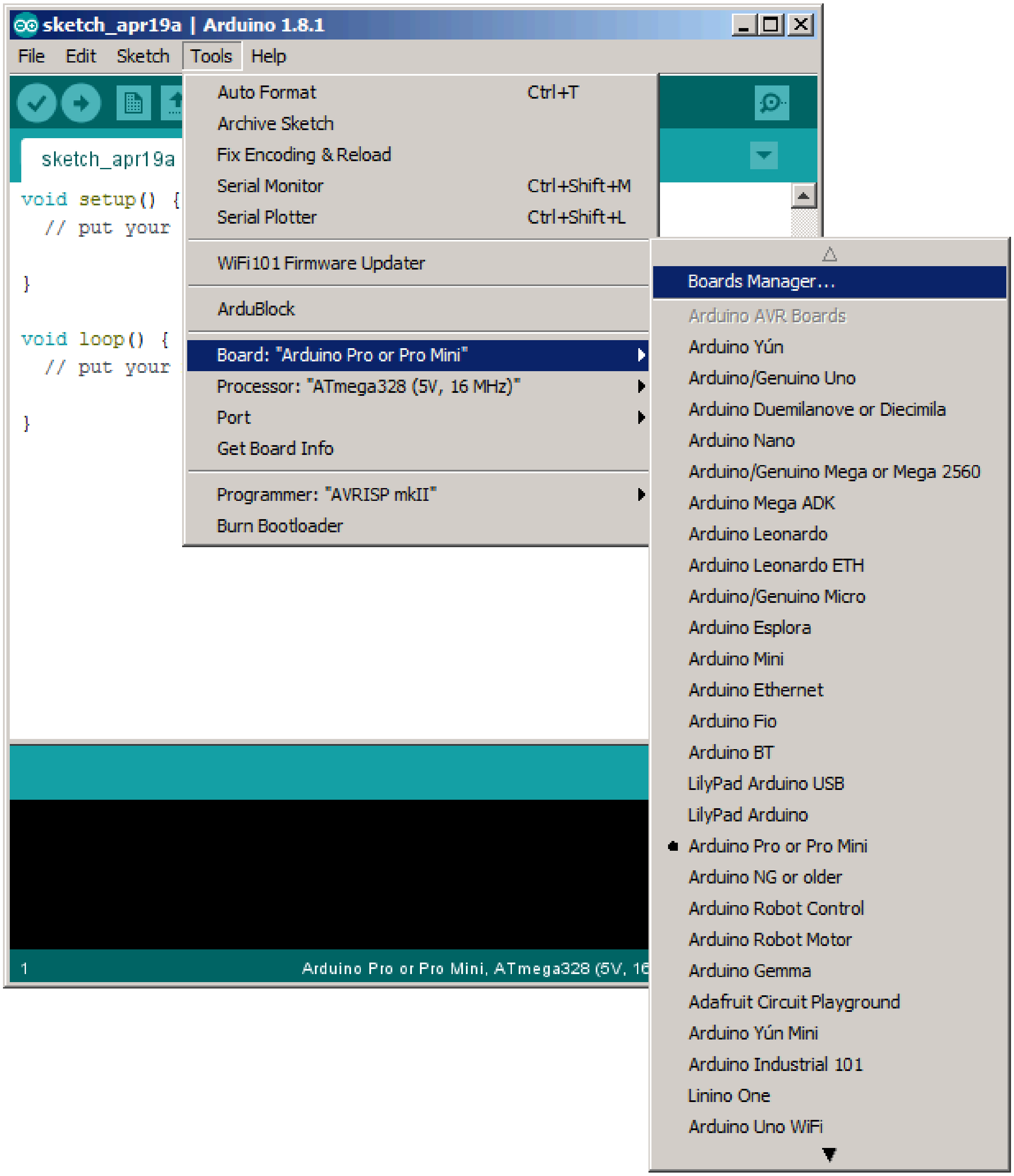
☑ Update sketch files to new extension on save (.pde -> .ino)

☑ Save when verifying or uploading

Additional Boards Manager URLs: ent.com/sparkfun/Arduino_Boards/LilyMini_testing/IDE_Board_Manager/package_sparkfun_index.json

More preferences can be edited directly in the file
C:\Users\mgrusin\AppData\Local\Arduino15\preferences.txt
(edit only when Arduino is not running)

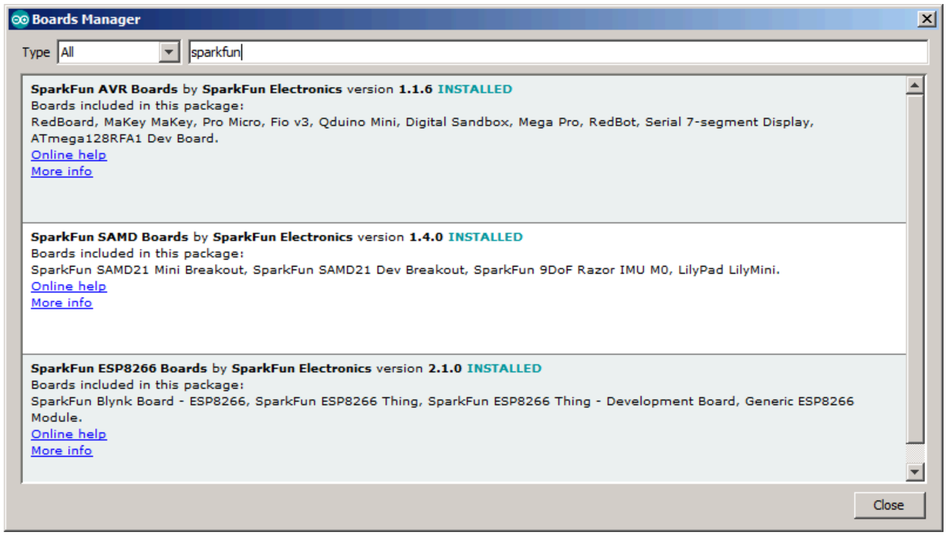
d. Go to Tools -> Board -> Boards Manager



```
void setup() {
  // put your

}

void loop() {
  // put your

}
```

1 Arduino Pro or Pro Mini, ATmega328 (5V, 16

e. If you type "sparkfun" (without quotes) into the "filter your search" field, you will see options to install SparkFun's Apollo3, AVR, and ESP board files. Click in the Sparkfun AVR box, and click the "Install" button that appears. Once installed, the boards will appear at the bottom of the board list.

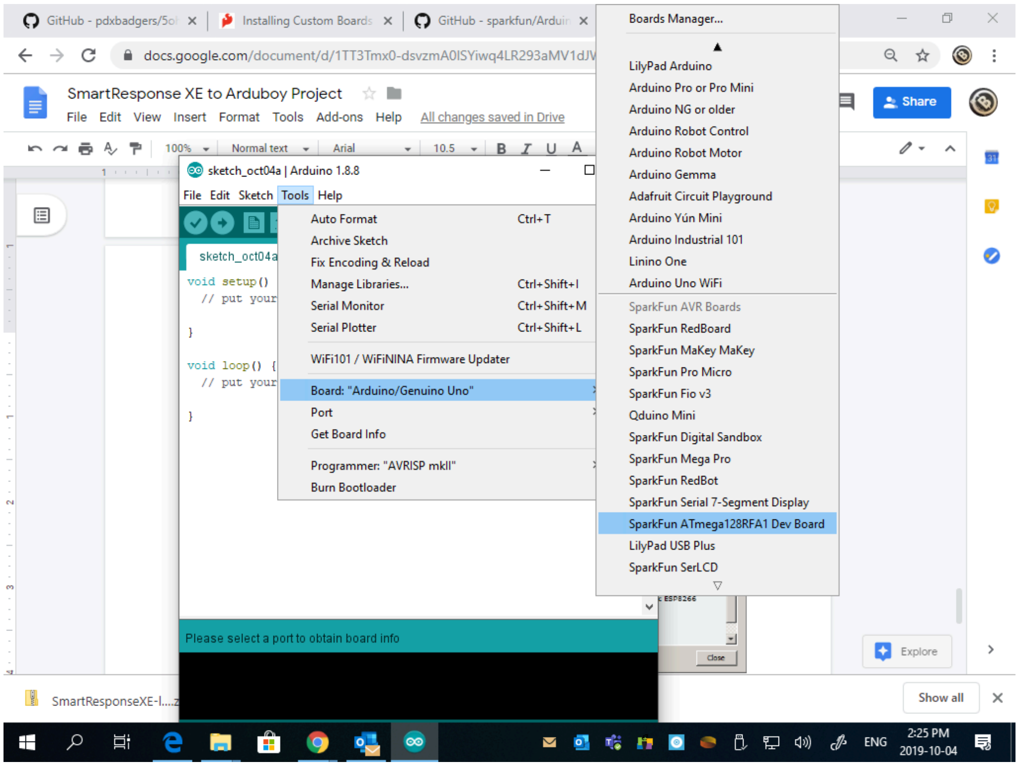


SparkFun AVR Boards by **SparkFun Electronics** version **1.1.6** **INSTALLED**
Boards included in this package:
RedBoard, MaKey MaKey, Pro Micro, Fio v3, Qduino Mini, Digital Sandbox, Mega Pro, RedBot, Serial 7-segment Display, ATmega128RFA1 Dev Board.
Online help
More info

SparkFun SAMD Boards by **SparkFun Electronics** version **1.4.0** **INSTALLED**
Boards included in this package:
SparkFun SAMD21 Mini Breakout, SparkFun SAMD21 Dev Breakout, SparkFun 9DoF Razor IMU M0, LilyPad LilyMini.
Online help
More info

SparkFun ESP8266 Boards by **SparkFun Electronics** version **2.1.0** **INSTALLED**
Boards included in this package:
SparkFun Blynk Board - ESP8266, SparkFun ESP8266 Thing, SparkFun ESP8266 Thing - Development Board, Generic ESP8266 Module.
Online help
More info

f. Go to Tools -> Boards Manager -> select "ATmega128RFA1 Dev Board" (under "Sparkfun AVR Boards")



```
void setup() {
  // put your
}

void loop() {
  // put your
}
```

Please select a port to obtain board info

3. Install dependencies: In Arduino IDE
    a. Go to https://github.com/pdxbadgers/5ohBEE-2019 and download the SmartResponseXE-lib.zip
    b. Select "Sketch" -> "Include Library" -> "Add .ZIP Library..." to add helpers/SmartResponseXE-lib.zip to the project

4.

5.

Downloaded 5ohBee.tar.gz from https://github.com/pdxbadgers/5ohBEE-2019
Install WinZip to open .gz format.

From the readme in the 5ohBee.tar.gz file:

# Arduino Board definitions for the ATmega128RFA1 Development Board

The _sparkfun_ folder should be added to a *hardware* directory within your Arduino sketchbook. Normally the sketchbook will install to your documents/Arduino folder. If a *hardware* folder isn't already there, make it.

This directory contains a *boards.txt* file which defines the ATmega128RFA1 Dev Board, and adds a selectable option under the *Tools > Board* menu. When you open the Arduino ID (close and reopen if it's already open), this option should be added to the menu.

Find out which COM port to use

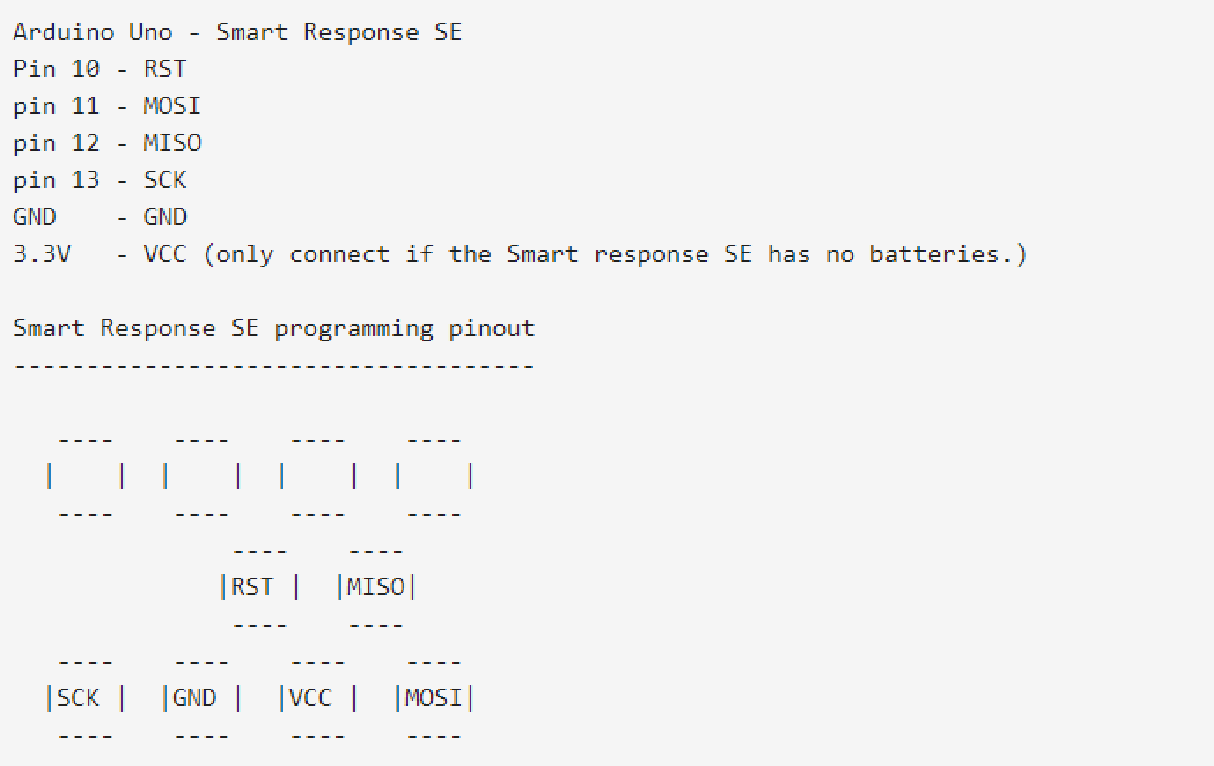
- go to device manager (WINKey + X), select Device manager and unfold ports (COM & LPT)
- connect Arduino Uno to PC and watch which COM port is added. This COM port must be selected in the Arduino IDE (and spacified on command line)
  Note: If there is no port added under ports but a device with an exclamation mark is added. Drivers need to be installed.
  Tip: if you want to use a fixed COM port number like COM4, you can change the port number in device manager by right clicking the COM port > properties > port settings tab > advanced > at the bottom choose the port from the drop down menu and click OK twice.

Make an ISP programmer from the Arduino Uno

- start Arduino IDE from the file menu, select examples > ArduinoISP > ArduinoISP and move to the new window.
- From tools menu select Arduino Uno/Genuino board (if not already selected)
- From the tools menu, Select the COM port and click the upload button (or press CTRL + U)
- Connect Arduino Uno to Smart Response XE using the following wiring:

```
Arduino Uno - Smart Response SE
Pin 10 - RST
pin 11 - MOSI
pin 12 - MISO
pin 13 - SCK
GND    - GND
3.3V   - VCC (only connect if the Smart response SE has no
batteries.)
```

```
Smart Response SE programming pinout
------------------------------------

   ----    ----    ----    ----
  |    |  |    |  |    |  |    |
   ----    ----    ----    ----
               ----    ----
              |RST |  |MISO|
               ----    ----
   ----    ----    ----    ----
  |SCK |  |GND |  |VCC |  |MOSI|
   ----    ----    ----    ----
```



```
Arduino Uno - Smart Response SE
Pin 10 - RST
pin 11 - MOSI
pin 12 - MISO
pin 13 - SCK
GND    - GND
3.3V   - VCC (only connect if the Smart response SE has no batteries.)
```

```
Smart Response SE programming pinout
------------------------------------

   ----    ----    ----    ----
  |    |  |    |  |    |  |    |
   ----    ----    ----    ----
               ----    ----
              |RST |  |MISO|
               ----    ----
   ----    ----    ----    ----
  |SCK |  |GND |  |VCC |  |MOSI|
   ----    ----    ----    ----
```

clearing lockbits and setting default fuses

The commandline tool avrdude is included with Arduino IDE and will be used to clear the lockbits and set the fuses.

- browse to the Arduino(installation)folder and browse into subfolder **hardware\tools\avr**
- rightclick the **bin** subfolder with shift pressed and click open command prompt here or open powershell here (Win10 default)
- for powershell: type cmd and press enter
- copy and paste the following line and press enter (Note: change COM4 to the correct port if it’s not COM4)

```
avrdude -C..\etc\avrdude.conf -pm128rfa1 -cstk500v1 -PCOM4 -b19200 -e
-Ulfuse:w:0xEE:m -Uhfuse:w:0x93:m -Uefuse:w:0xFC:m
```

This will erase the chip, reset the lock bits and sets the correct fuses for bootloader-less usage.

Uploading a hexfile using AVRdude

here’s the command to flash a hex file. Just repace <hexfile> with the fullpath to the hex file to flash:

```
avrdude -C..\etc\avrdude.conf -pm128rfa1 -cstk500v1 -PCOM4 -b19200 -e
-Ulfuse:w:0xEE:m -Uhfuse:w:0x93:m -Uefuse:w:0xFC:m
```

Here's a sample hO access with simplified pin numbering (e.g.exfile for testing. It’s Jetpac using an experimental interleaved display