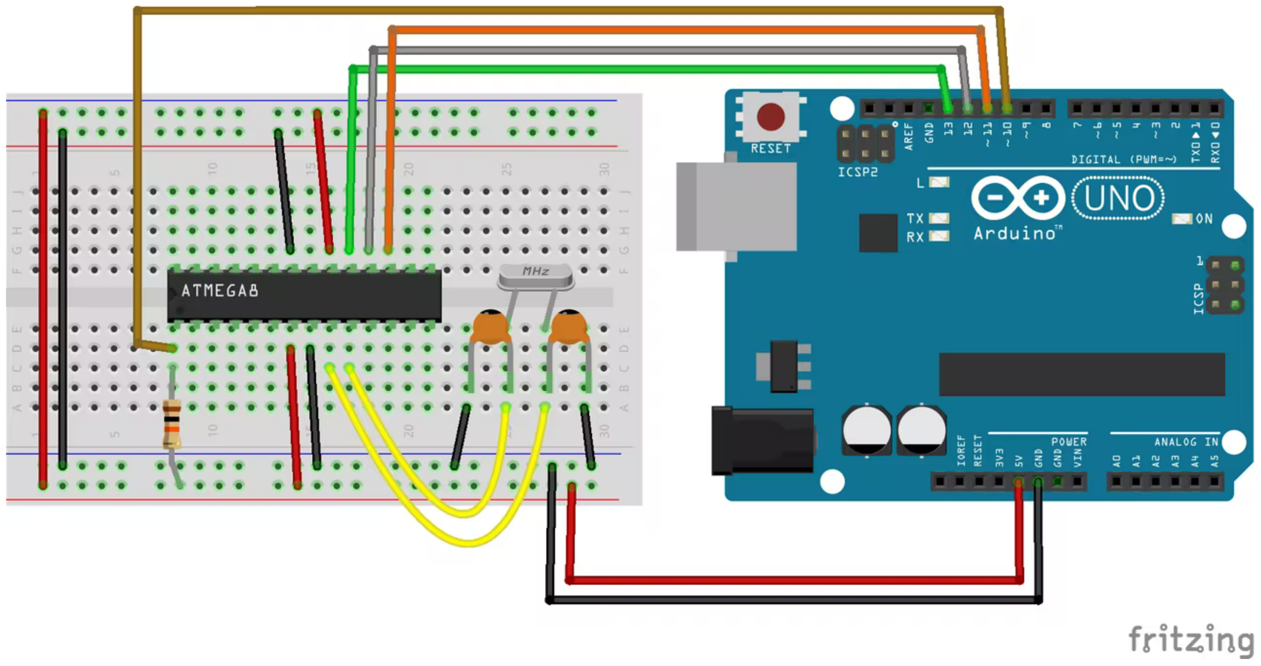
1. If the arduino you are going to use as a programmer has not yet been programmed to act as an In-system programmer, program it using the ArduinoISP sketch, which can be found under Examples → ArduinoISP

2. Wire the arduino and the ATMEGA8 as shown below



3. Programming the ATMEGA8 is done through Microchip studio. To start, go into Tools → External Tools, click "add", then paste the following into the "Arguments" field

-e -v -U lfuse:w:0xFF:m -patmega8 -carduino -PCOM5 -b19200 -D -Uflash:w:"$(ProjectDir)Debug\$(ItemFileName).hex":i -C"C:\Program Files (x86)\Arduino\hardware\tools\avr\etc\avrdude.conf"

PCOM5 is the COM port of the arduino, which may not be 5.
b19200 is the baud rate

-U is used to set the fuses for the ATMEGA8. Setting fuses improperly may cause issues that prevent programming, please be careful when changing fuses. Additionally, some fuses are only able to be set a single time. Avoid setting these if not needed.

For the title, write **&deploy**
In the command field, point to AVRDude.exe. My path was here:
C:\Program Files (x86)\Arduino\hardware\tools\avr\bin\avrdude.exe

Once the above steps are completed, they do not need to be done again, as they are saved

4. To upload code to the ATMEGA8,
   a. Clean and build the project
   b. Click the green play button to generate the hex file
   c. Build→Compile
   d. Tools→Deploy

5. If AVRDude complains about not finding the hex file, you may need to make sure the hex file is named properly. It can be found in the project folder, under “Debug”. If the device ID is incorrect (All Fs or all 0s), do not force upload. Something is not correct, and force uploading will not work and could damage the hardware.