

Witam. Dzisiaj kilka słów na temat programowania arduino, dokładnie to atmegi 328 P-PU, którą możemy znaleźć w Arduino Uno. Uważam, że zaczęcie od zaprogramowania własnej atmegi jest dobrym przygotowaniem do tworzenia własnych projektów. Oczywiście możemy kupić arduino i na nim projektować, jednak po czasie będziemy musieli i tak przenieść się na własną atmegę, chyba że kogoś stać na wstawianie w każdy projekt arduino, poza tym projektując własną płytkę PCB łatwiej jest wstawić samą atmegę niż całe arduino.

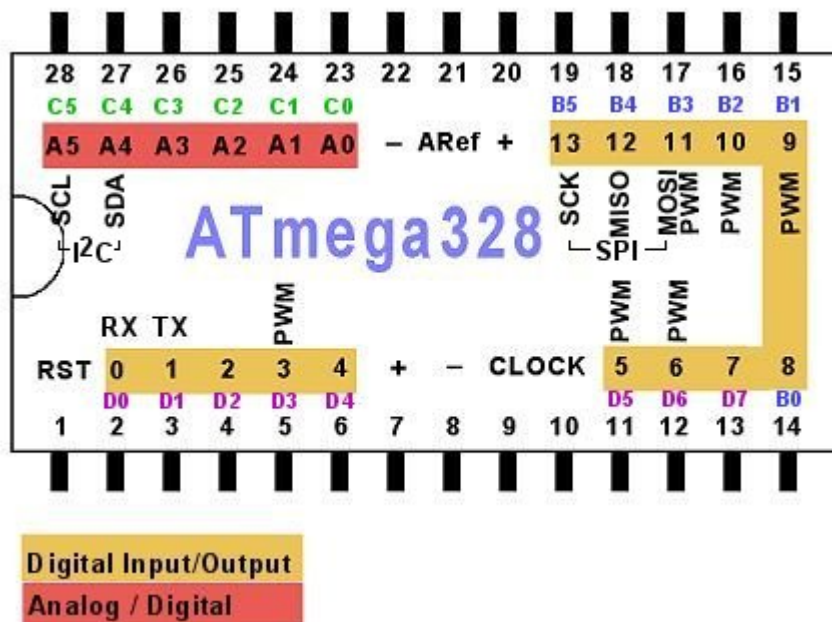
Ale zacznijmy od początku. Najpierw czym jest arduino. Wikipedia mówi nam następująco:

Arduino – [platforma programistyczna](#) dla [systemów wbudowanych](#) oparta na prostym projekcie [Open Hardware](#) przeznaczonym dla [mikrokontrolerów](#) montowanych w pojedynczym [obwodzie drukowanym](#), z wbudowaną obsługą wejścia/wyjścia oraz standaryzowanym językiem programowania. Język programowania Arduino jest oparty na środowisku [Wiring](#) i zasadniczo na języku [C/C++](#) (kilka prostych przekształceń kodu wykonywane przed przejściem do avr-gcc). Celem projektu Arduino jest przygotowanie narzędzi – ogólnodostępnych, tanich, nie wymagających dużych nakładów finansowych, elastycznych i łatwych w użyciu przez hobbystów. Częściowo Arduino stanowi również alternatywę dla osób, które nie mają dostępu do bardziej zaawansowanych kontrolerów, wymagających bardziej skomplikowanych narzędzi.

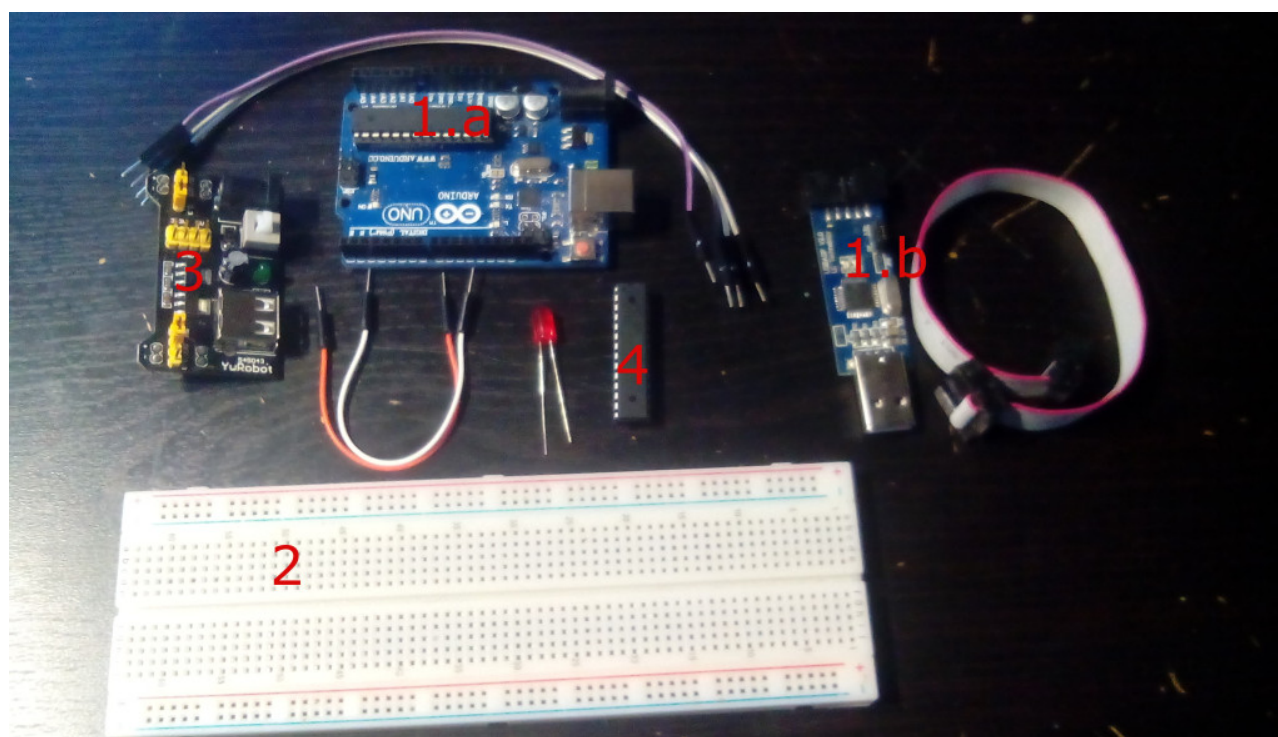
Prostymi słowami arduino to płytkę PCB z mikrokontrolerem który możemy dowolnie zaprogramować, oczywiście w miarę jego możliwości. Do dyspozycji mamy piny analogowe i cyfrowe. Komunikacje i2c, SPI i kilka innych.

Dziś zajmiemy się wgraniem bootloadera zgodnego z arduino, po to by móc wgrywać programy używając ArduinoIDE. Oczywiście możemy robić to za pomocą terminala czy innego programu do programowania avr'ów.

Zacniemy od poznania pinoutu (rozłożenia pinów) naszej atmegi.



Powyżej mamy uproszczony układ pinów. Do programowania będzie nam potrzebny poniższy zestaw:



1.a Arduino Uno, którego użyjemy jako programatora. Opcjonalnie możemy użyć



programatora USBASP (1.b) dostępnego na [allegro](#).

2. Płytki stykowa.

3. Moduł zasilania do płytek stykowych z opcją ustawienia zasilania na 5V i 3,3V.

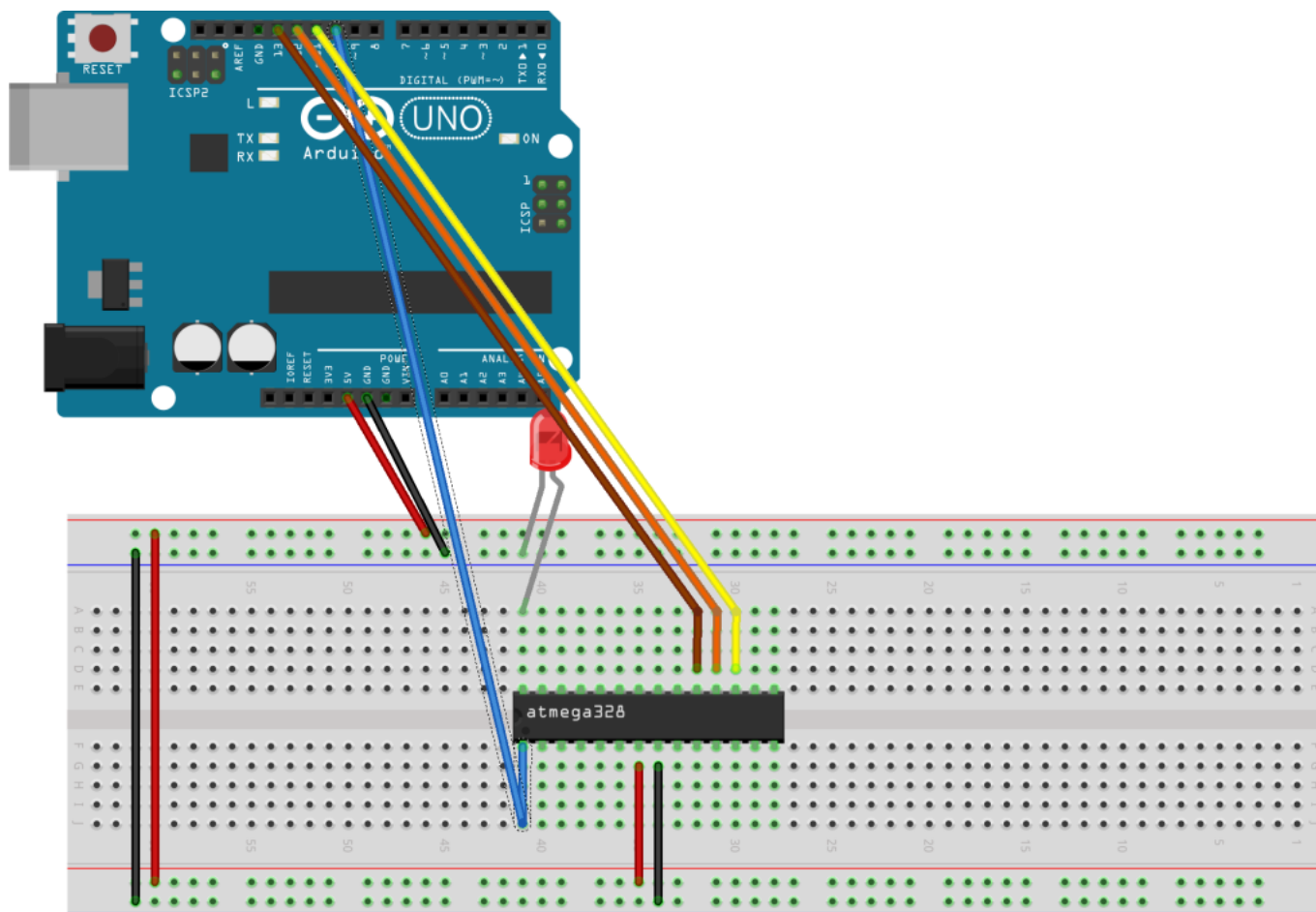
4. Atmega 328P-Pu, którą kupiłem jakiś czas temu na warszawskim wolumenie.

Oczywiście będziemy też potrzebować kabli do połączeń oraz jednej diody led do testu czy wszystko dobrze się wgrało.

Pozostaje nam połączenie elementów.

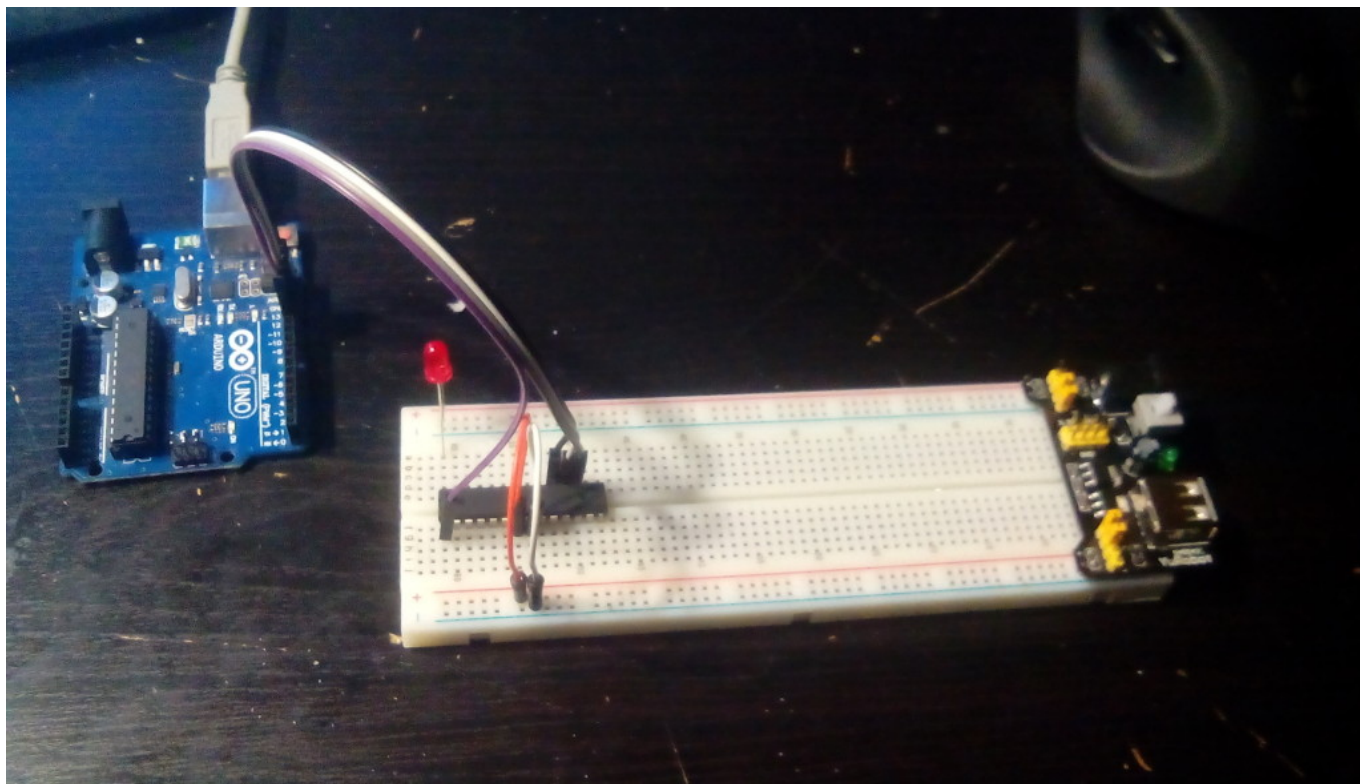
- pin 13 (arduino) - pin 19 atmega
- pin 12 (arduino) - pin 18 atmega
- pin 11 (arduino) - pin 17 atmega
- pin 10 (arduino) - pin 1 atmega

Schematycznie wygląda to tak:



fritzing

Na powyższym schemacie ujęta jest sytuacja gdy nie dysponujemy modułem zasilającym. Możemy wtedy spokojnie podłączyć zasilanie z arduino. U mnie w rzeczywistości wygląda to tak:



Teraz przygotowujemy nasze IDE. Zaczniemy od podmiany pliku [boards.txt](#). Umieszczamy go w `Arduino\hardware\arduino`.

Następnie musimy dodać [bootloader](#). Po rozpakowaniu umieszczamy go w `Arduino\hardware\arduino\bootloaders\atmega`.

Następnym krokiem będzie wgranie na nasze arduino szkicu ArduinoISP.



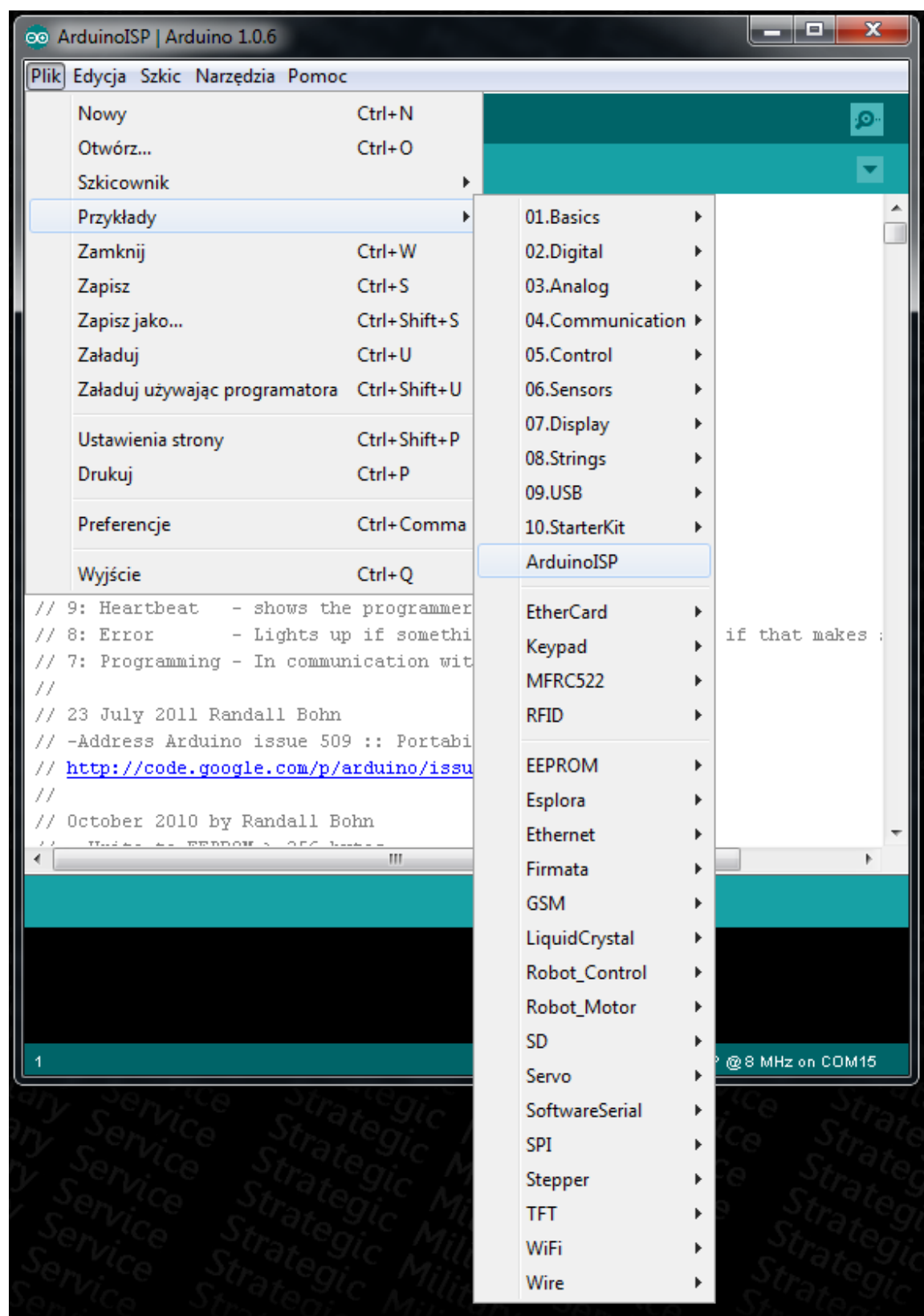
Arduino część pierwsza.



Arduino część pierwsza.

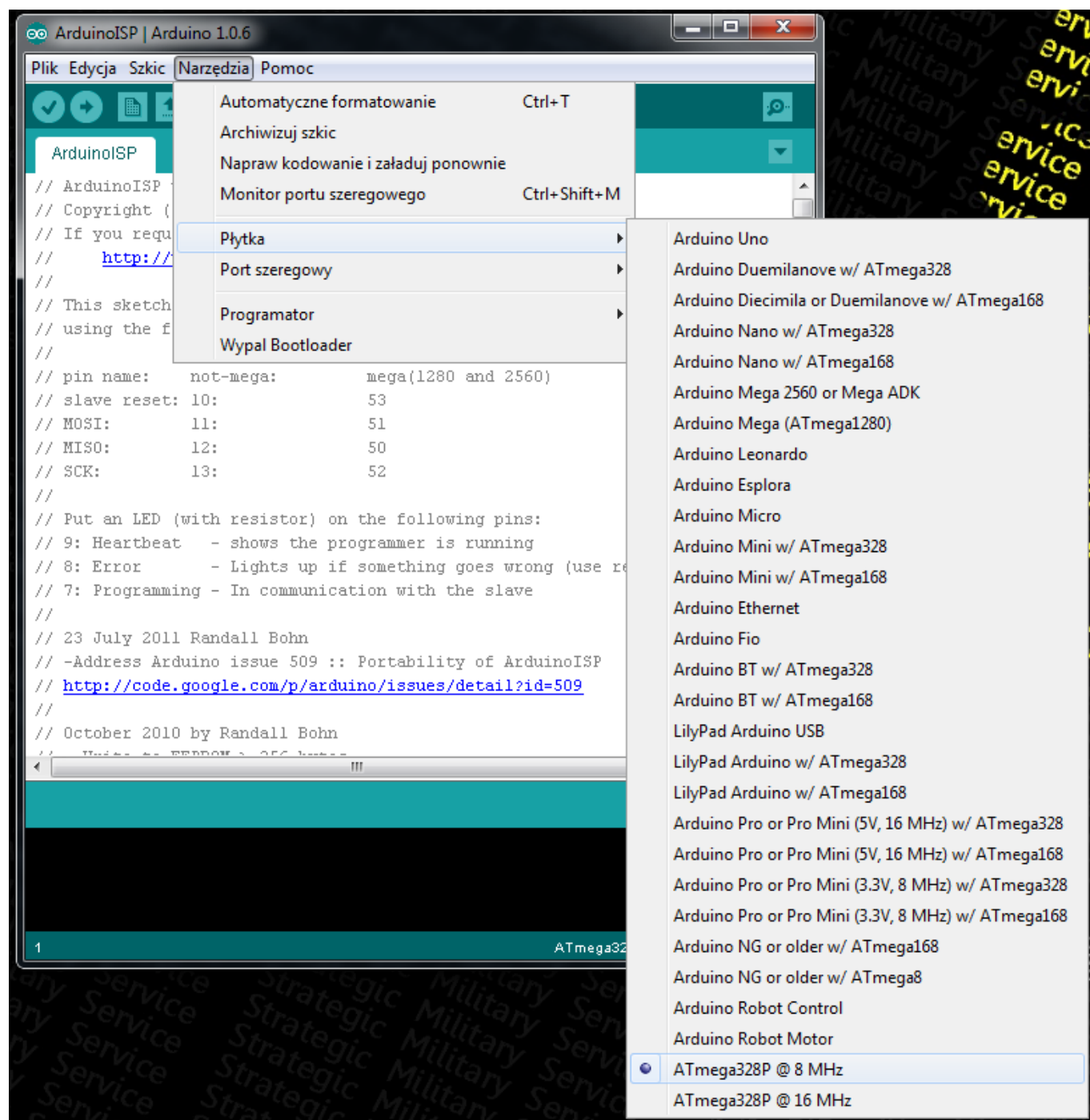


Arduino część pierwsza.



Wgrywamy

na nasze arduino. Następnym krokiem jest wybranie z listy naszej atmegi.



Z przykładowych szkiców wybieramy Basic -> Blink i modyfikujemy. Zamieniamy 13 pin na A5 – pin analogowy nr 5 który jest pinem nr 28 w naszej atmedze. Zabieg jest zalecany ponieważ do pierwotnego pinu przypisanego pod arduino podpięliśmy przewód



Arduino część pierwsza.

programatora.

Potem już tylko zostaje nam załadowanie naszego szkicu na atmege i sprawdzenie czy wszystko działa. Wiedza dotycząca obsługi ArduinoISP na pewno przyda się nam gdy będziemy tworzyć własne projekty oparte o pcb czy gdy utracimy oryginalny bootloader na naszym arduino.