

User manual for 4-axes controller designed for RepRap type 3d printers



## Table of Contents

1 Introduction.....	3
2 Assembly diagram .....	3
2.1 Description of jumpers responsible for choosing power source .....	5
2.2 Description of jumpers responsible for step mode selection.....	6
3 Windows drivers and software.....	7
3.1 Arduino installation.....	7
3.2 Teensyduino installation.....	10
3.3 FLIP installation.....	14
4 Linux Ubuntu drivers and software.....	19
5 Marlin compilation and installation on the KOS board.....	23
5.1 Marlin editing.....	23
5.2 Marlin compilation.....	25
5.3 Marlin.cpp.hex installation on Windows.....	29
5.4 Marlin.cpp.hex installation on Linux.....	36
6 Sources and additional info about KOS.....	38
6.1 LCD, Encoder, SD-card assembly diagram.....	38
7 Bibliography.....	39

# 1 Introduction

KOS 1.2 is electronics created with experience from building 3D Prusa printers in mind. It's equipped with screw terminals, widely used in industrial electronics, that eliminate popular problems like wrongly crimped plugs or low joints quality. KOS 1.2 driver has also possesses easily distinguishable connectors for LCD screen, SD-card reader and encoder. Thanks to this the control panel can be placed anywhere with ease. Additionally a potentiometer has been included that serves for screen's contrast regulation and a resistor for brightness regulation.

One undeniable quality of KOS electronics is the possibility of using 3 different power sources at once. It means that we can power motors from Vmot with 12-30V, heaters from Vhot 12-30V and logics from P1 = 12V or directly from the USB port. Not everyone needs such a diverse range of possibilities thus jumpers have been added to make the installation process easier.

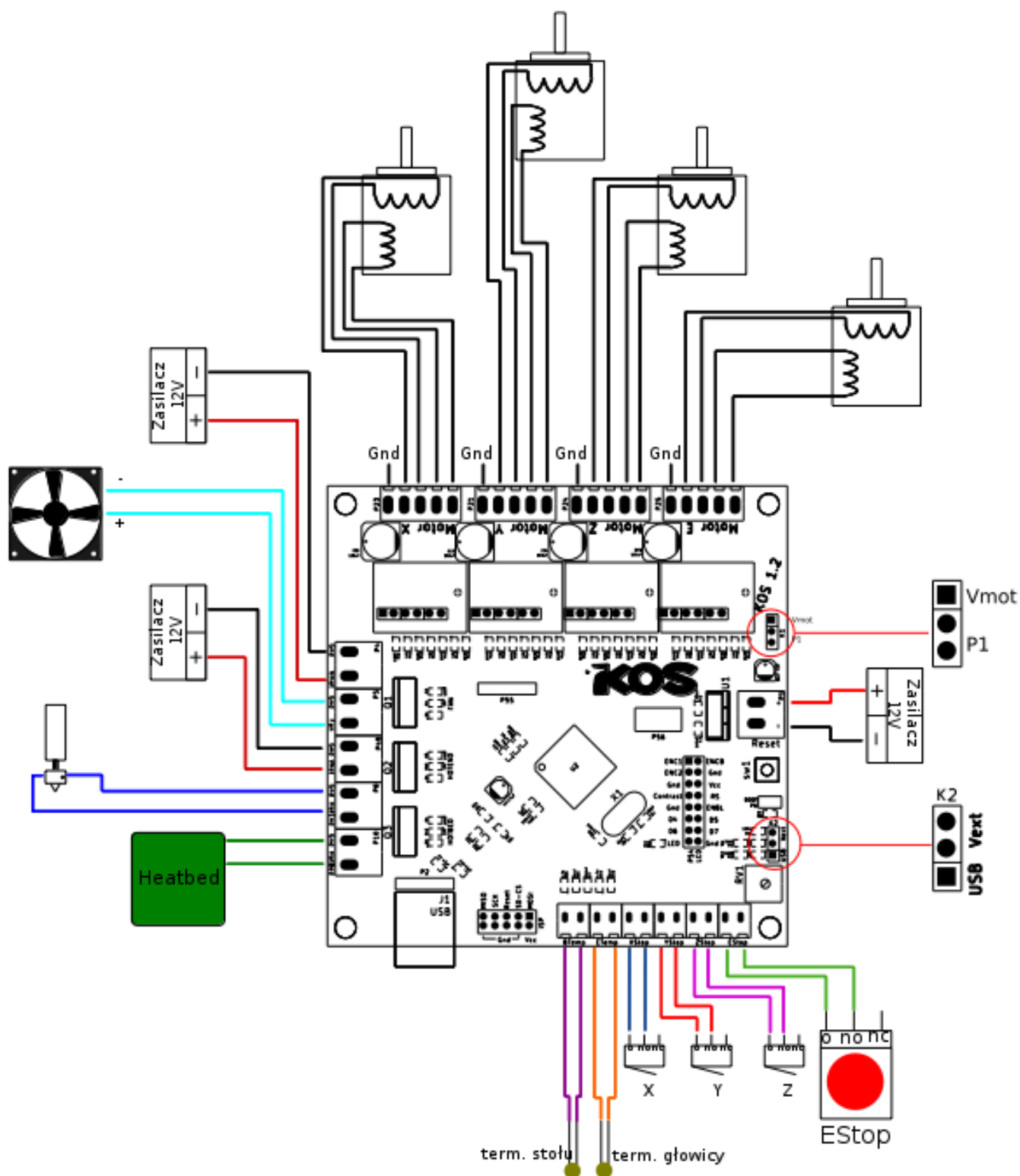
KOS 1.2 electronics features an AT90USB1286/7 processor - an 8-bit controller widely used in RepRap-type 3D printers. One of it's qualities is compatibility with USB port, allowing for quick data exchange between your computer and the printer. This processor is used in electronics like *Teensylu*, *Printboard*, *SAV MKI*, *Sunbeam*, *Uniqueone* and many others. It's the most popular processor from AVR family in the world of 3D printing.

## 2 Assembly diagram

A diagram below depicts assembly of basic elements of the printer such as:

a heater, a heatbed, engines etc. It's vital that cables of right diameter are used.

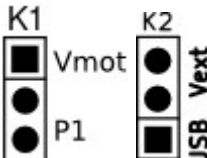
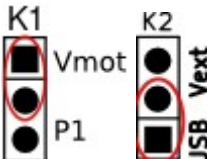
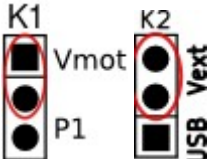
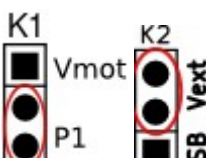
- Engines: 0.35mm<sup>2</sup>
- Heaters: 1.5mm<sup>2</sup>
- Limits, thermistors, safety switch E-Stop: 0.35mm<sup>2</sup>
- Power supplies: 1.5mm<sup>2</sup>



Ilustracja 1: Assembly diagram for KOS electronics






## 2.1 Description of jumpers responsible for choosing power source

KOS electronics features multiple power sources usage depending on application. To implement that function two goldpin arrays have been installed to allow switching between them.

nr	Jumper	Description
1		No power source has been chosen.
2		USB port chosen as power source. K1 array is being disregarded here.
3		Vmot chosen as power source.
4		P1 chosen as power source.

## 2.2 Description of jumpers responsible for step mode selection

Each engine has jumpers assigned to it responsible for step control on step engine drivers based on Allegro A4988 chipset. Possible configurations are listed below.

nr	Jumpers	Description
1		Full-step mode (1)
2		Half-step mode ( $\frac{1}{2}$ )
3		Quarter-step mode ( $\frac{1}{4}$ )
4		One-eighth-step mode ( $\frac{1}{8}$ )
5		One-sixteenth-step mode ( $\frac{1}{16}$ )

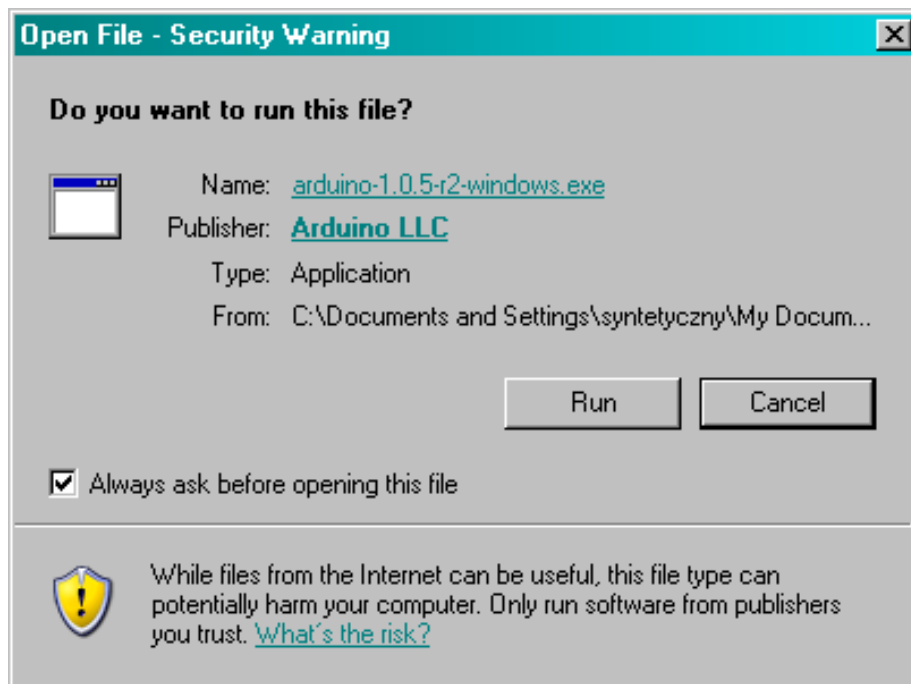
### 3 Windows drivers and software

KOS electronics is designed to be reversely compatible with *teensylu* electronics so for it to work properly drivers' installation is required. If the electronics has Marlin pre-installed then only teensy project's drivers are to be installed 1.

If that's not the case we have to create a new compilation of Marlin. It's how-to is described in „Marlin Compilation” chapter 5. Before that, however, our electronics requires libraries from Arduino, Teensyduino and FLIP that we have to install:

- Arduino's main page 2
- Teensyduino's main page 3
- FLIP for Windows 4

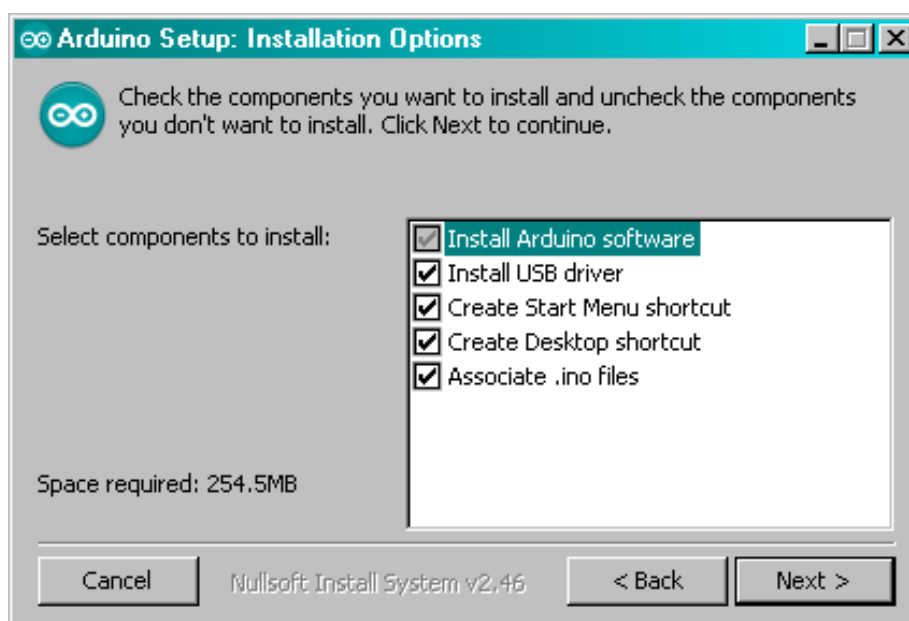
#### 3.1 Arduino installation



*Ilustracja 2: Allow the system to run the file*

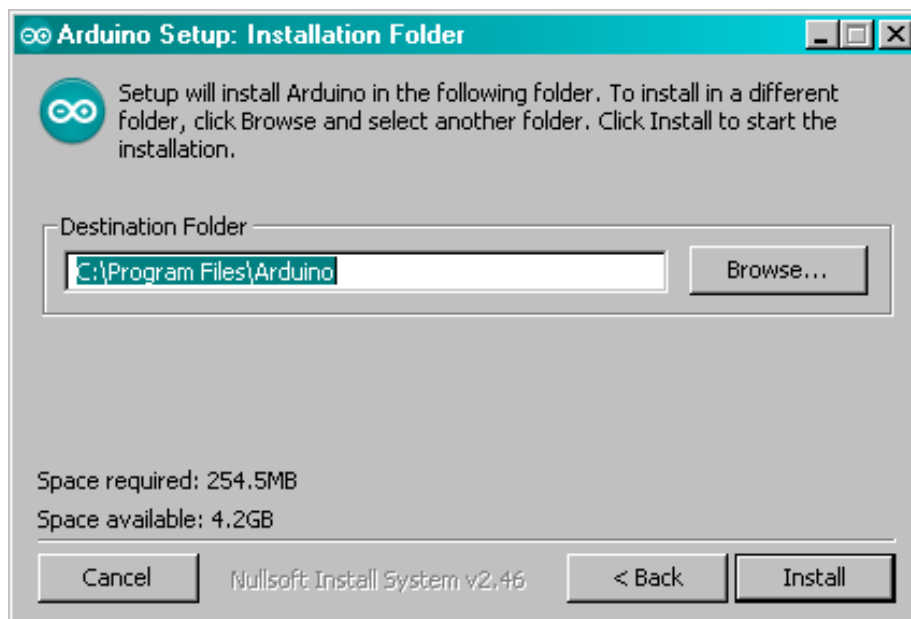


*Ilustracja 3: Agree to the license agreement*

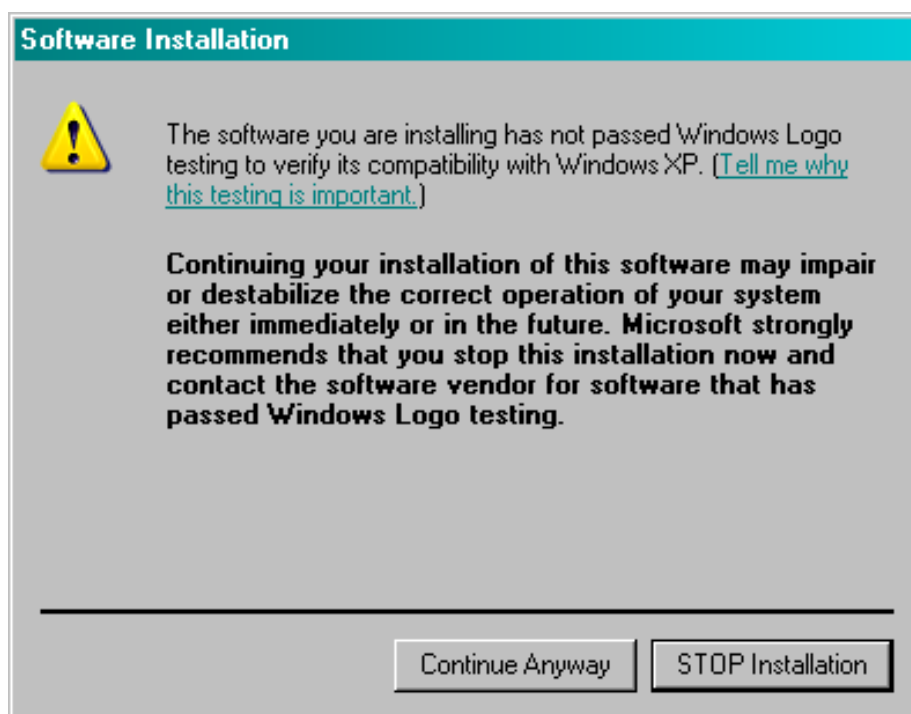


*Ilustracja 4: Pick components to be installed*

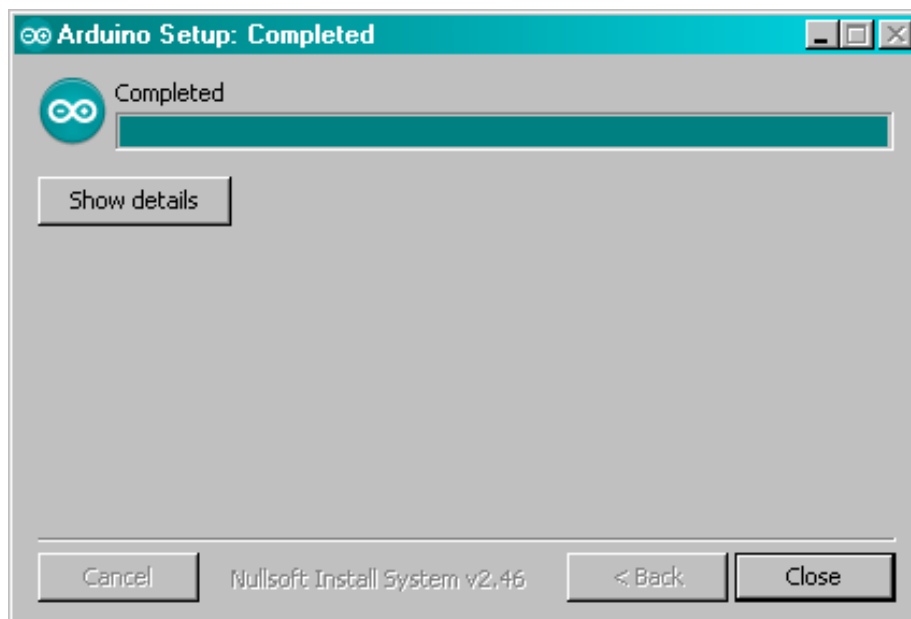




*Ilustracja 5: Pick a folder to install Arduino into*



*Ilustracja 6: Windows's warning against installing unchecked software*

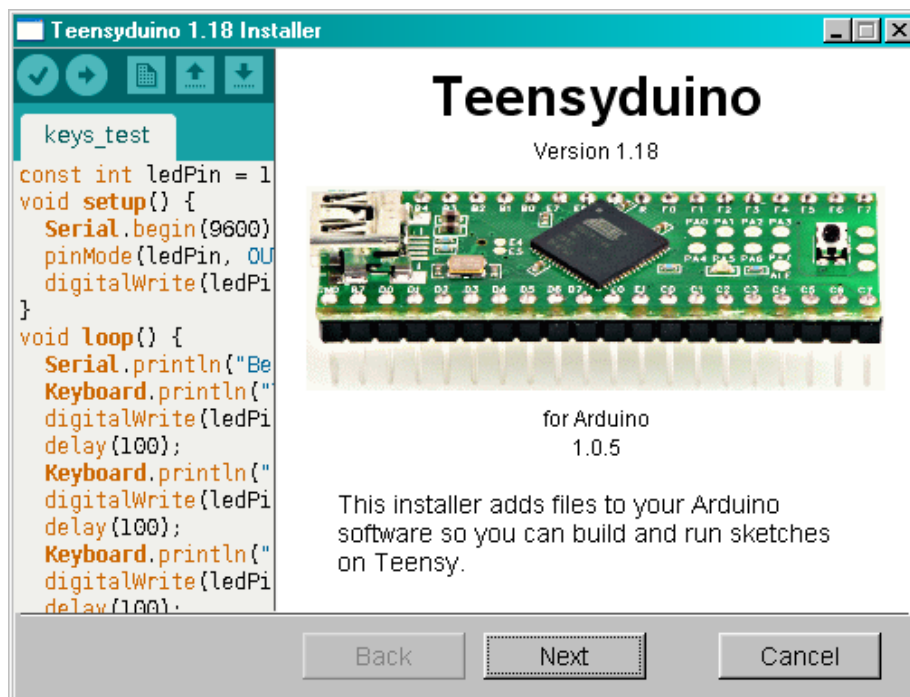


*Ilustracja 7: Arduino installation completed!*

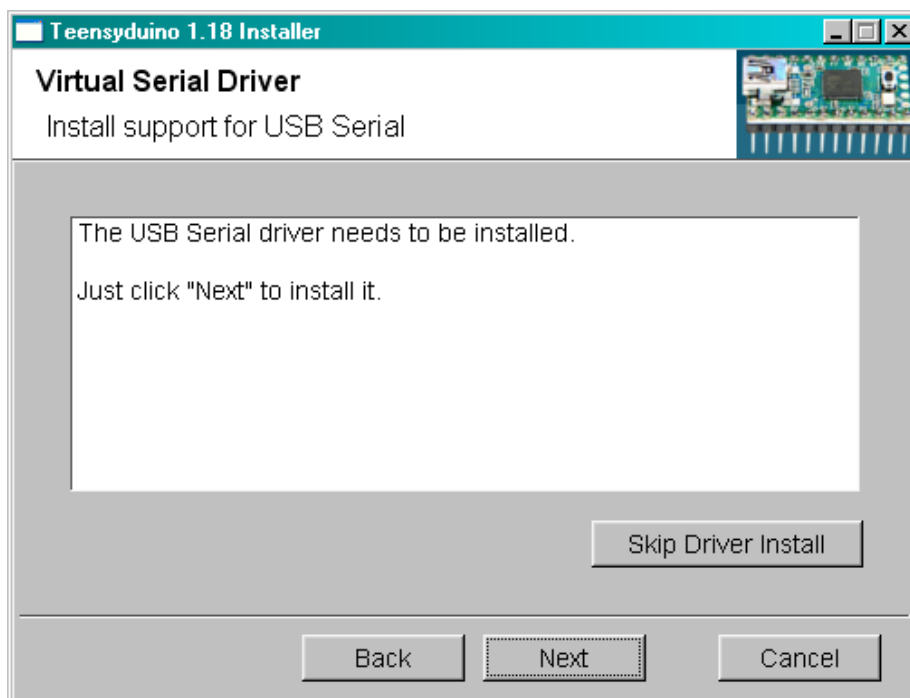
### 3.2 Teensyduino installation



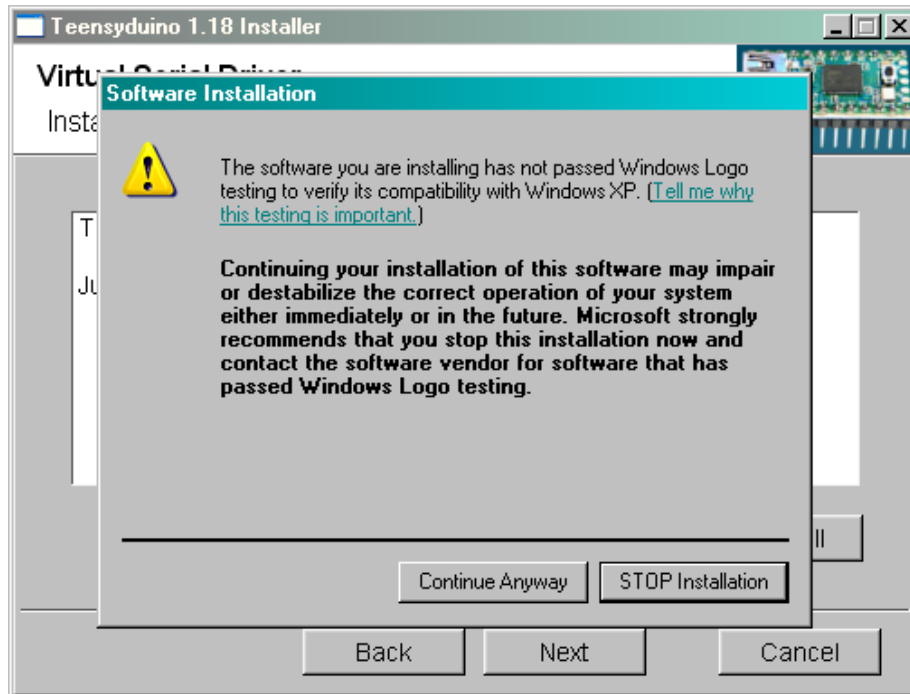
*Ilustracja 8: Run the file*



*Ilustracja 9: Choose Next*



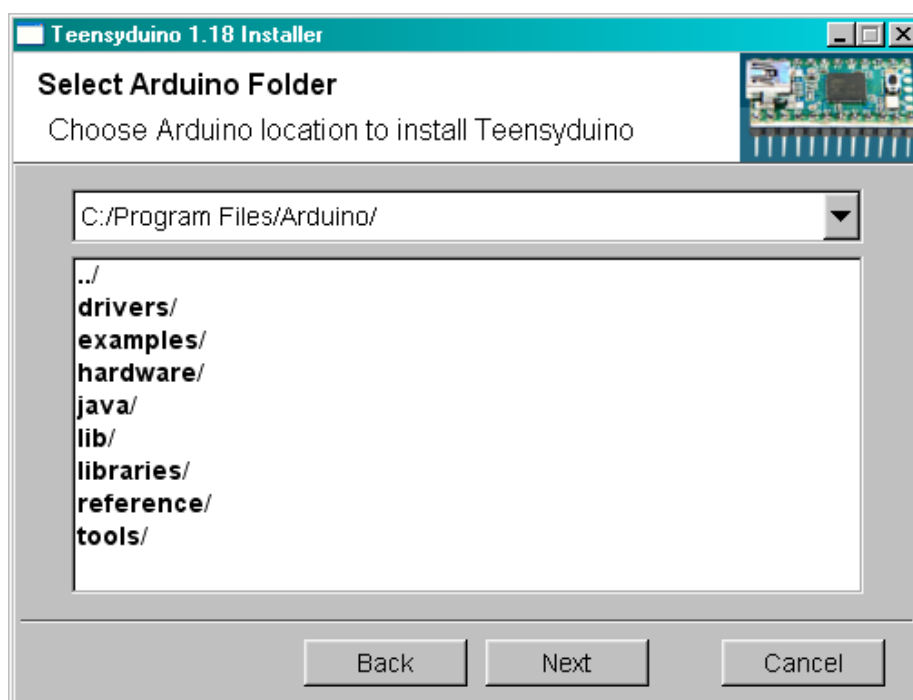
*Ilustracja 10: Pick Next again. If you haven't installed serial\_installer.exe before the program will install those drivers.*



*Ilustracja 11: Choose Continue Anyway like in the case of Arduino*

ATTENTION! An important step!

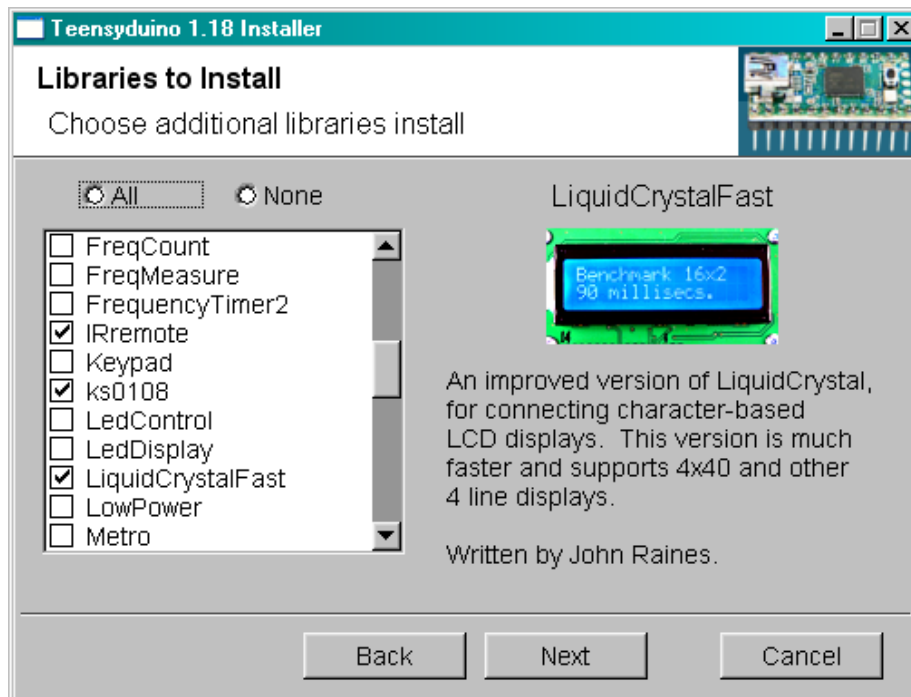
For installation to complete successfully we need to choose a folder with our earlier ARDUINO installation.



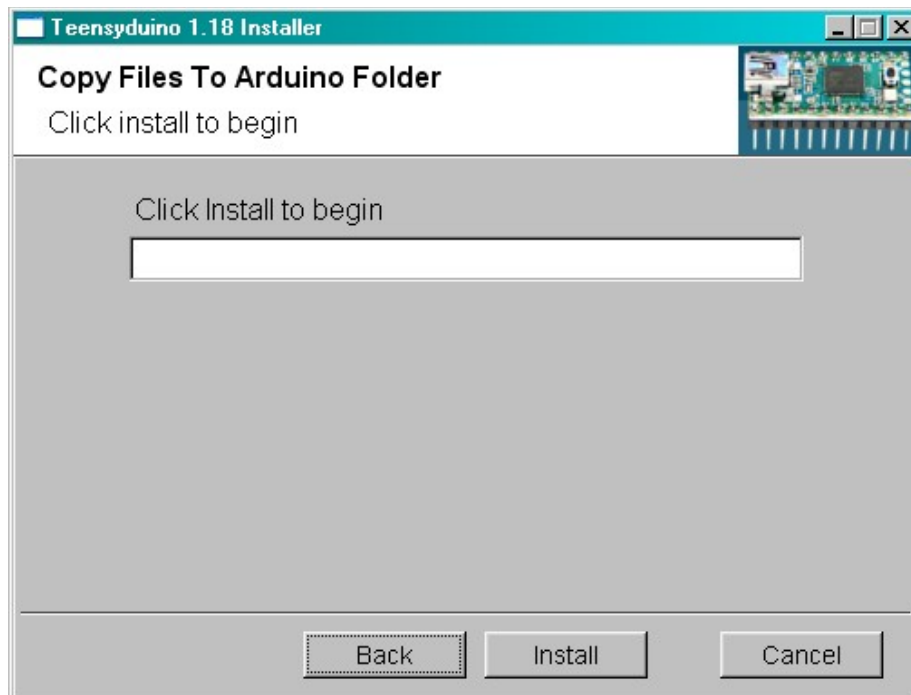
*Ilustracja 12: Pick the folder you installed Arduino to.*

REMEMBER!

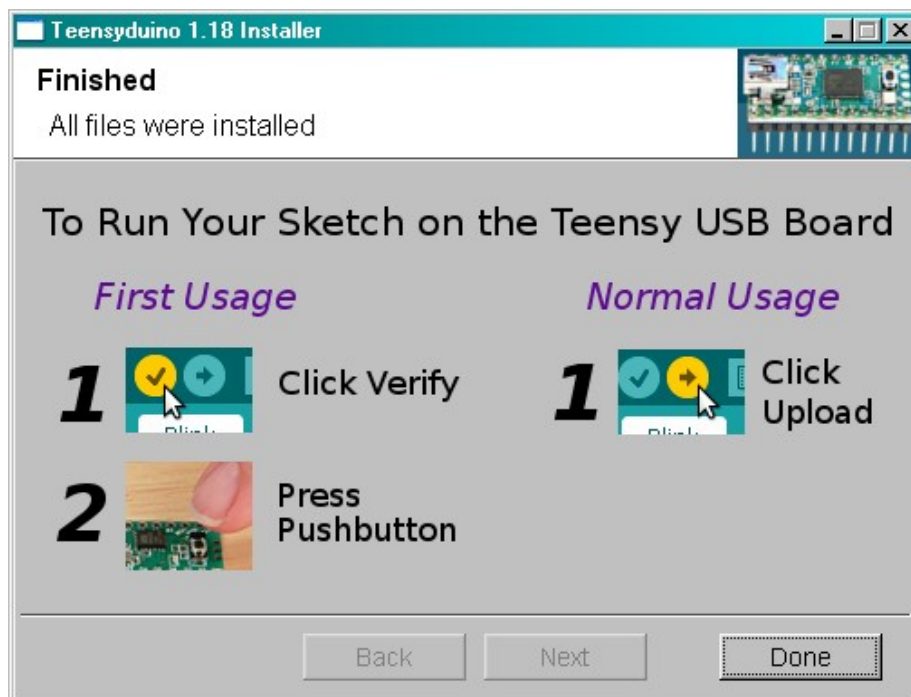
Pick only 3 libraries from the list: Iremote, ks0108 and LiquidCrystalFast.



*Ilustracja 13: Pick only Iremote, ks0108 and LiquidCrystalFast*

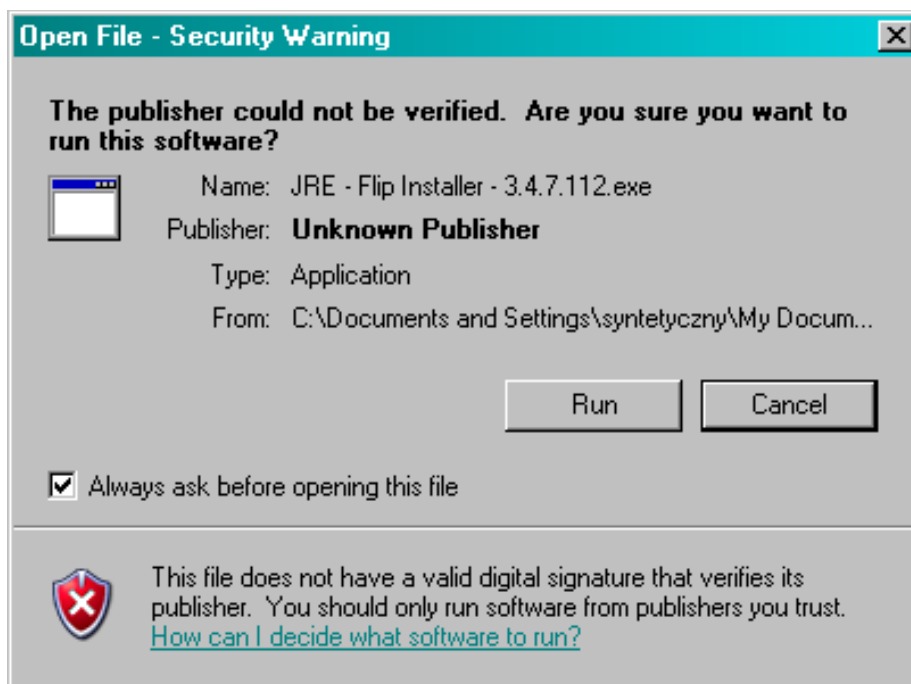


*Ilustracja 14: Begin the installation.*

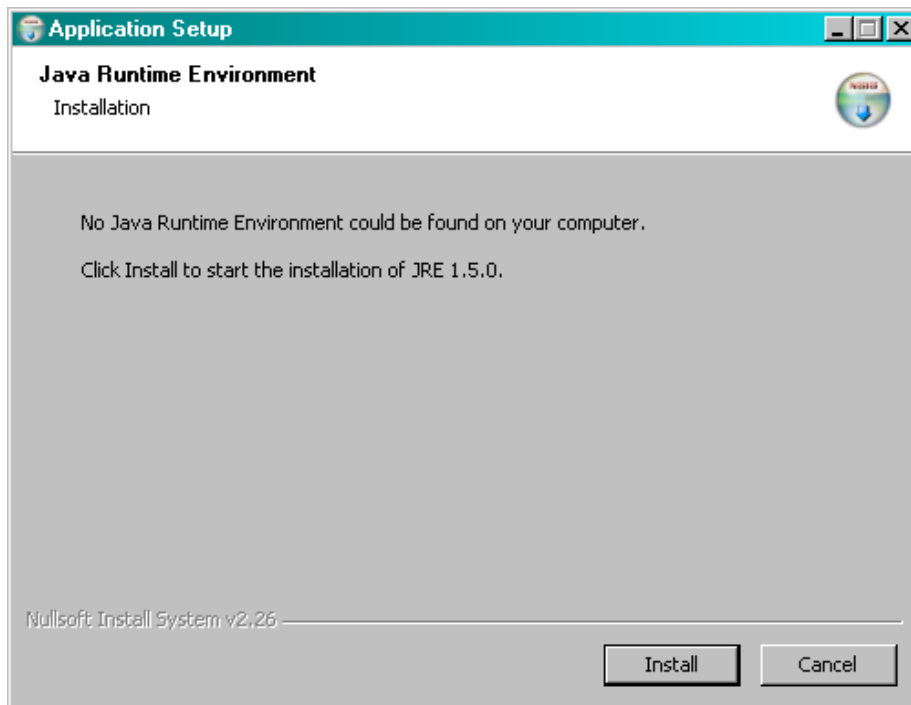


*Ilustracja 15: Teensyduino installation has been completed successfully!*

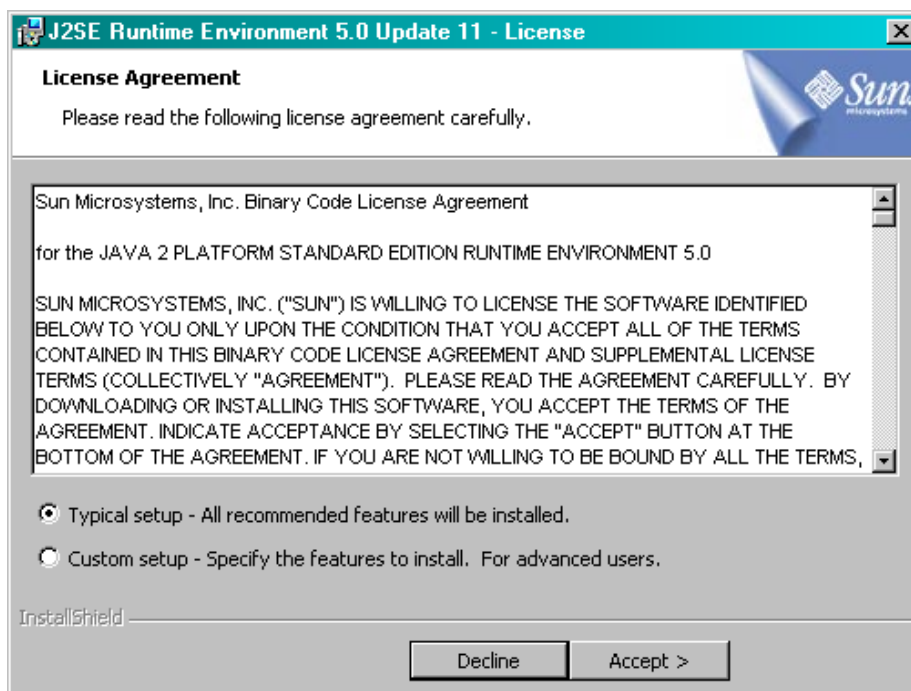
### 3.3 FLIP installation



*Ilustracja 16: Run the installer*



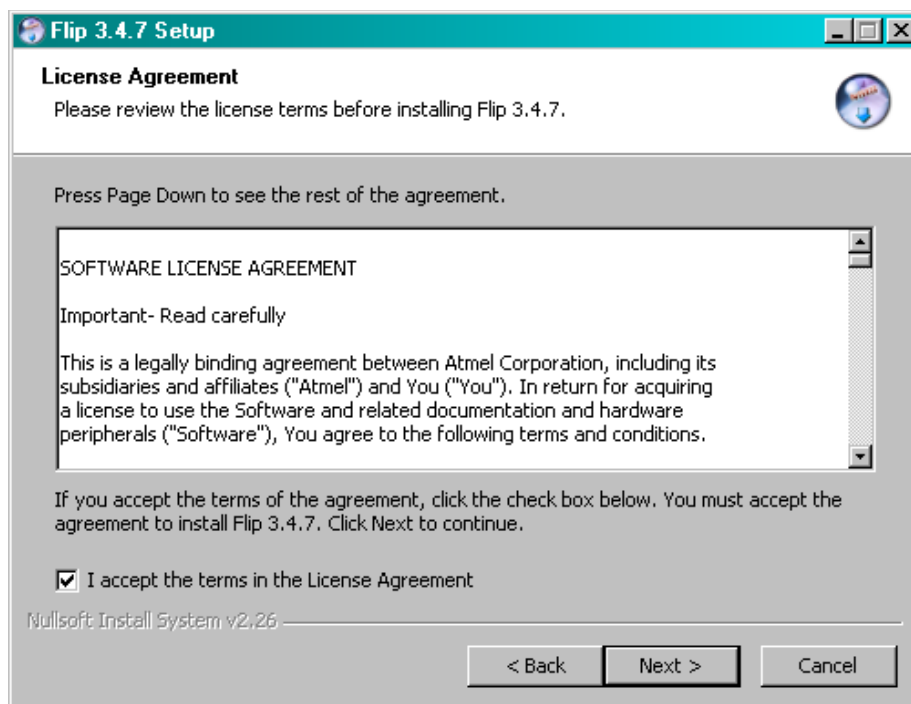
*Ilustracja 17: In case you don't have Java the program will install it for you*



*Ilustracja 18: Choose standard installation and Accept.*

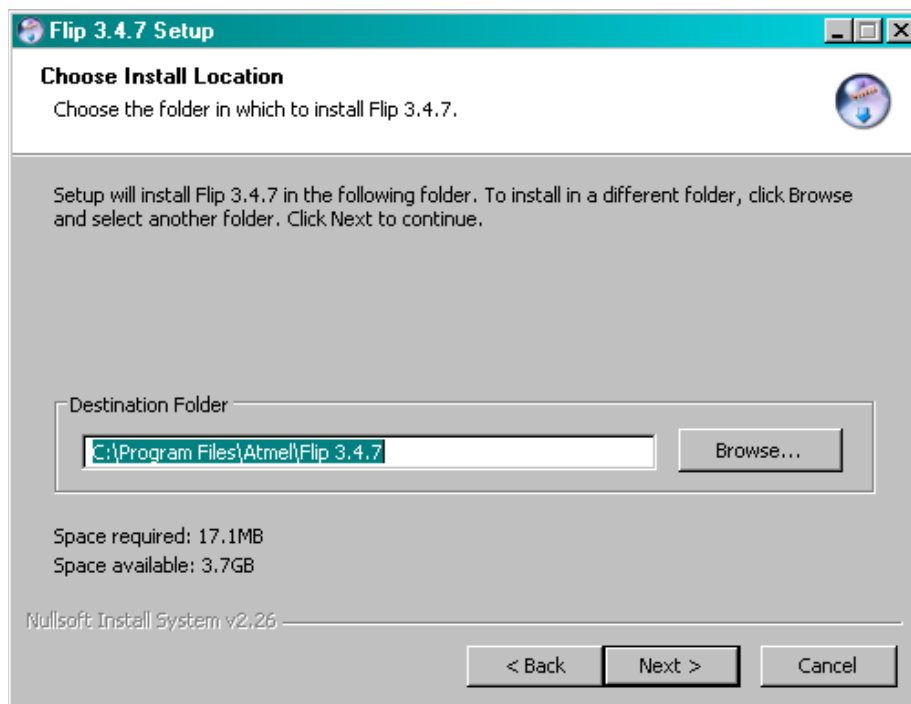


*Ilustracja 19: Press Next*

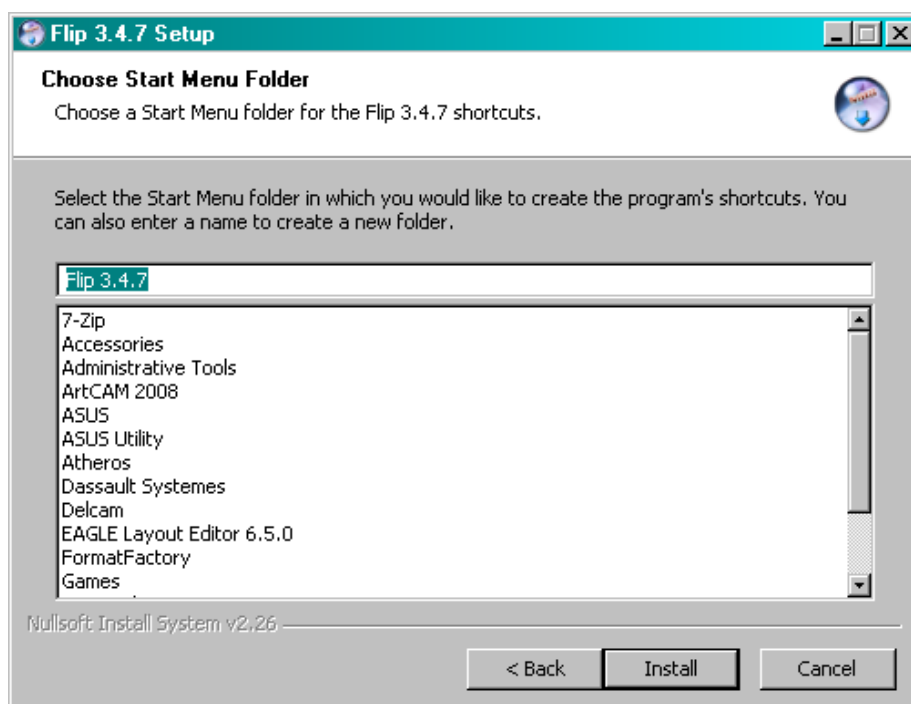


*Ilustracja 20: Accept the terms of the license agreement*

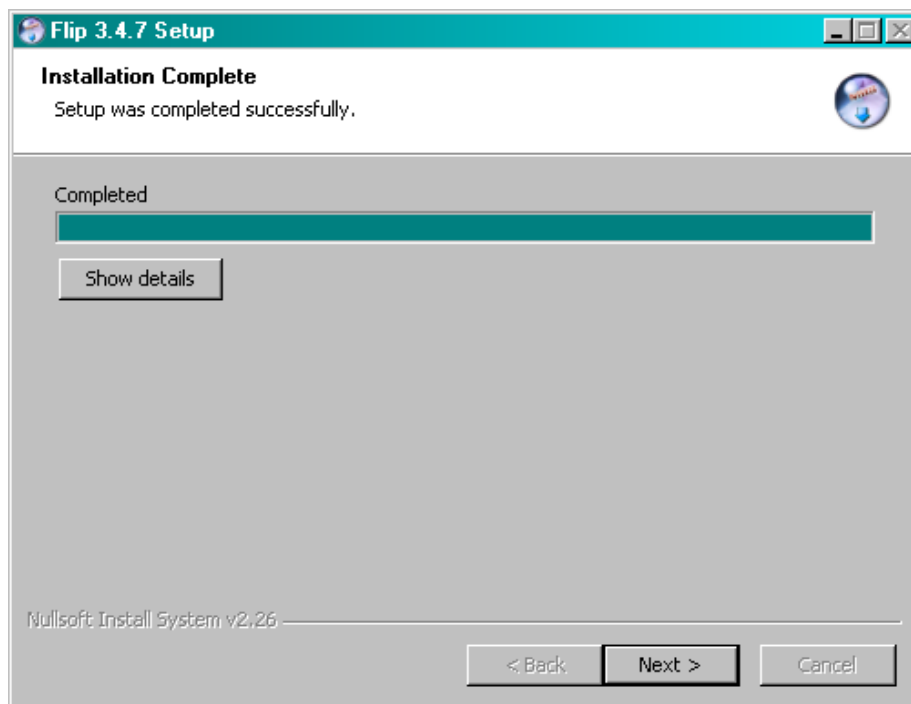




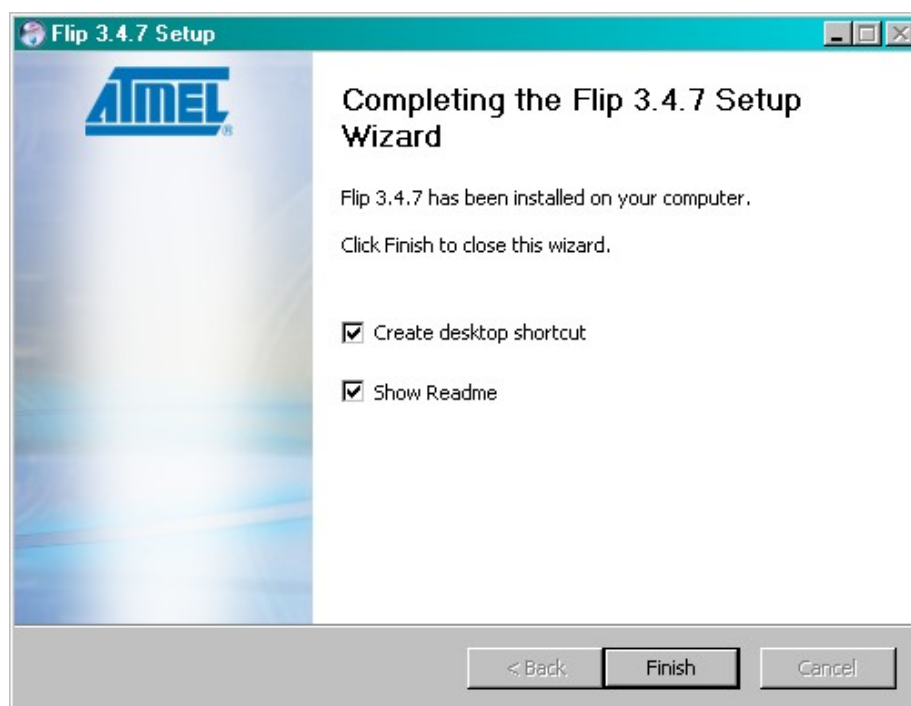
*Ilustracja 21: Choose install location for FLIP*



*Ilustracja 22: Choose a name for the Start Menu folder where FLIP's shortcut will be located.*



*Ilustracja 23: Installation completed successfully!*



*Ilustracja 24: Choose if you want to have a desktop shortcut created and if you want to open the Readme file.*

## 4 Linux Ubuntu drivers and software

Begin with downloading Arduino drivers from their main page 2 . After you finish unpacking drivers open a terminal (Ctrl + Alt + T) and install necessary files :

```
sudo apt-get install gcc-avr avr-libc
```

```
sudo apt-get install openjdk-6-jre
```

Next, we add execution rights for Arduino. To do that we have to open the Arduino installation folder and click the file with right mouse button opening *Properties* and in *Rights* tab choose *Allow to execute file*.

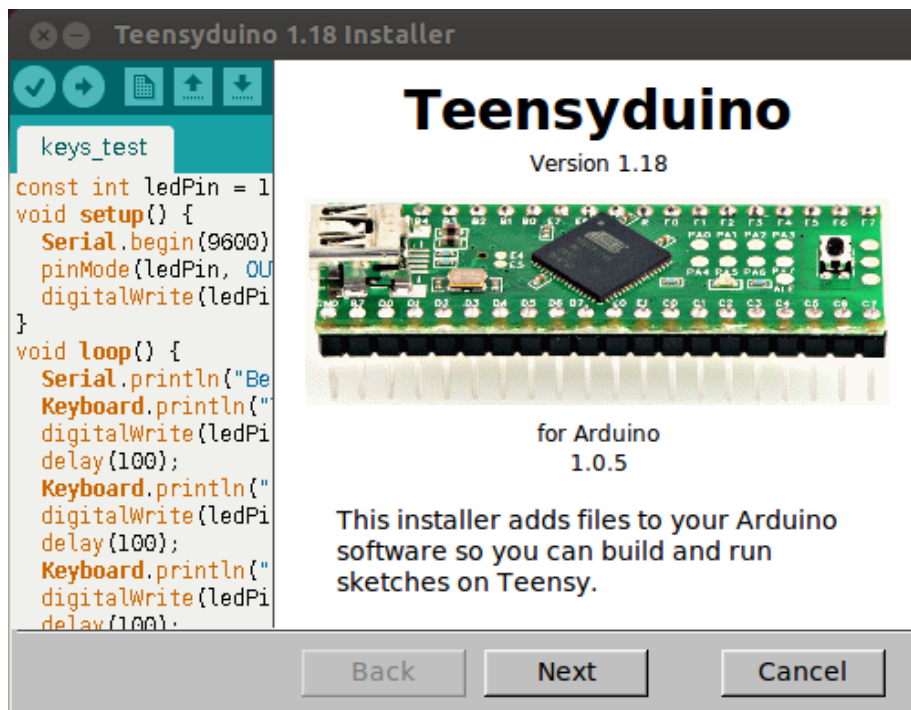
Next step is downloading teensyduino from it's homepage in a version for our operating system (32/64 bits) and saving it in a location of our choice. To execute the program we need to open the terminal (Ctrl + Alt + t) and select a location of the file:

```
cd /home/user_name/Location/File
```

Next by using the chmod command we add execution rights to *teensyduino.32bit*

```
chmod 755 teensyduino.32bit
```

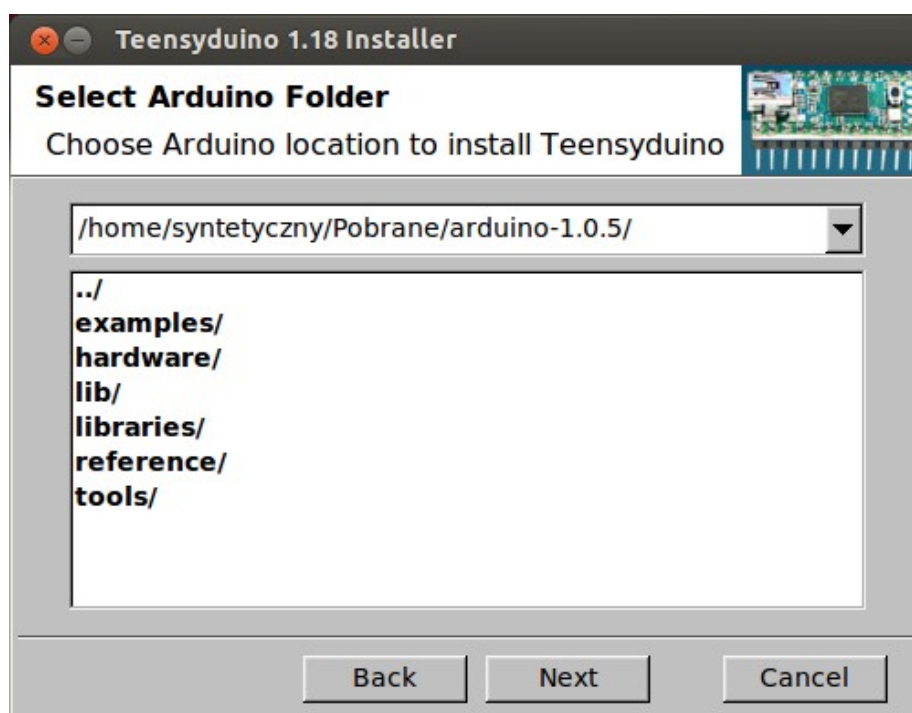
Then open the installer with *./teensyduino.32bit* command and follow the instructions:



Ilustracja 25: Press Next

ATTENTION!

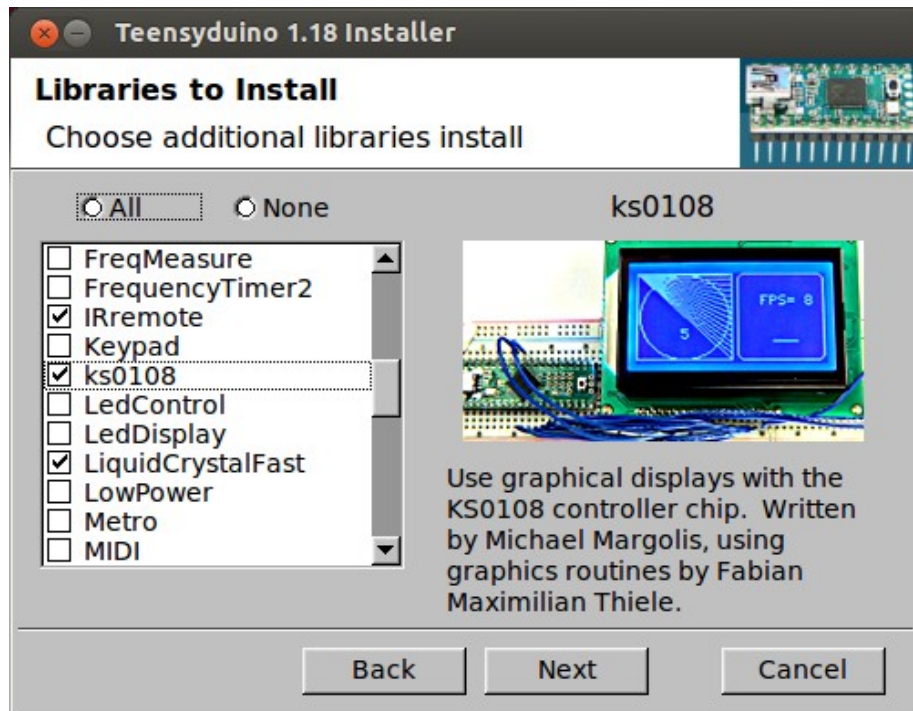
Select installation folder where you installed Arduino.



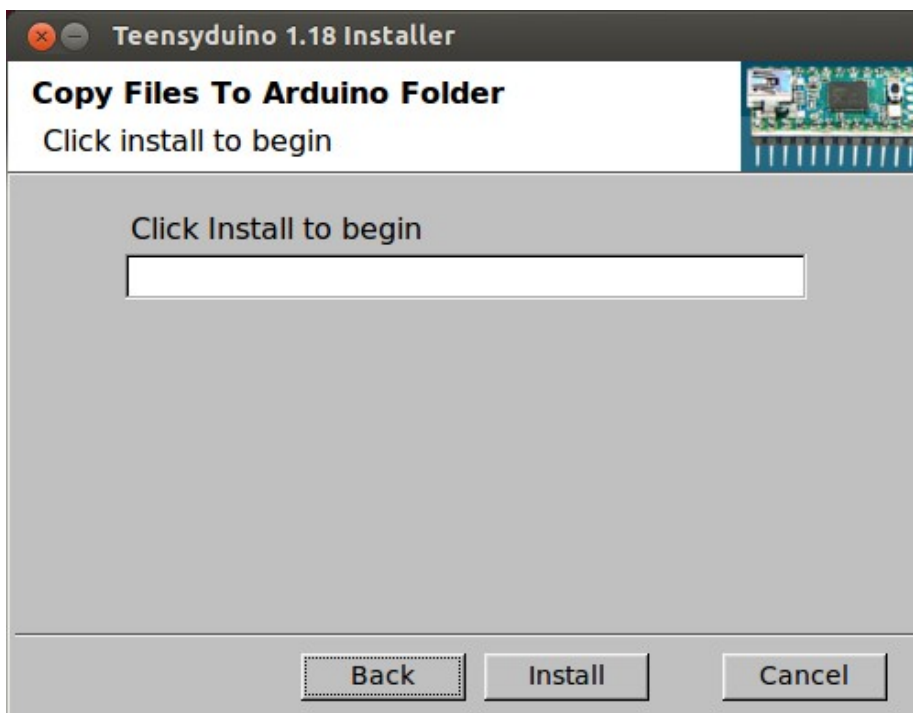
Ilustracja 26: Selection of location for Teensyduino

Important!

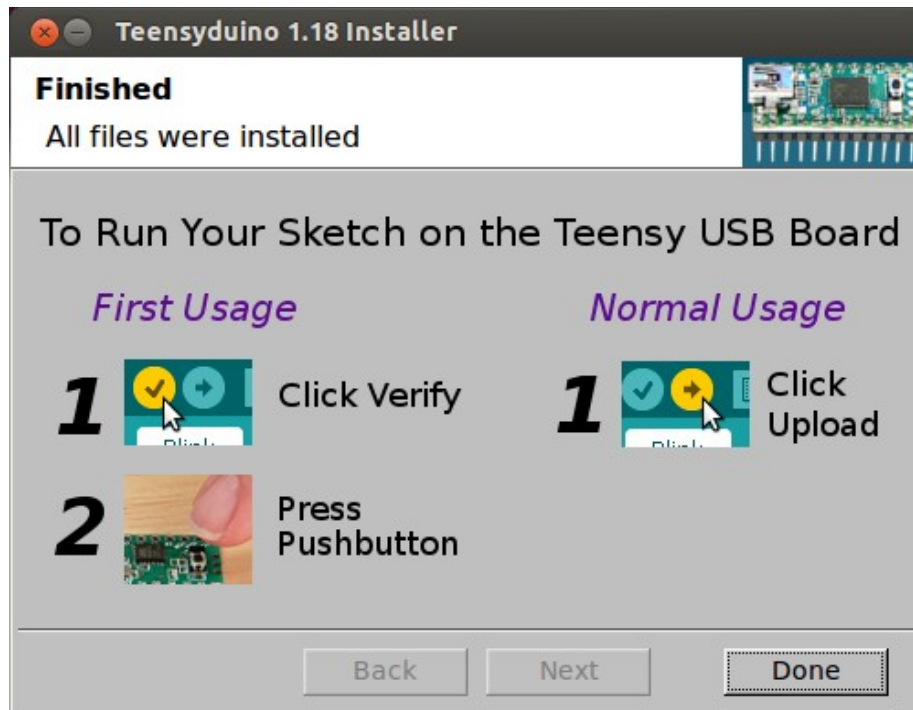
Select only 3 following libraries: IRremote, js0108, LiquidCrystalFast.



*Ilustracja 27: Select necessary libraries and press Next*



*Ilustracja 28: Press Install*



*Ilustracja 29: Press Done thus finishing the installation*

Next step we need to take is installation of equivalent of FLIP for AT90USB serial processors programming. Open the terminal and copy-paste following command:

```
sudo apt-get install dfu-programmer
```

The installer will download necessary application by itself and install the program on our computer. Thus finishing the installation of necessary drivers/libraries for Marlin.

## 5 Marlin compilation and installation on the KOS board

This chapter will focus on Marlin's compilation. That's because the installation differs slightly for different hardware configurations. Also this chapter will be divided into Ubuntu and Windows sections for the same reasons.

### 5.1 Marlin editing

Unpack files downloaded from Marlin's main page 5 to the folder you want Marlin installed to. Next by using any text-editing tool (e.x. Notepad++) open *Configuration.h* file and find the line : *#define BAUDRATE 250000* and switch 250000 for 115000.

Now find the *#define MOTHERBOARD 7* line and switch 7 for 8 thus changing the default configuration from ULTIMAKER to teensylu – one our driver is compatible with (because of the same processor). Next step is choosing termistors we use. By default most heaters use EPCOS high-temperature termistors. Let's find a line that starts like this then:

```
#define TEMP_SENSOR_0 -1  
  
#define TEMP_SENSOR_1 -1  
  
#define TEMP_SENSOR_2 0  
  
#define TEMP_SENSOR_BED 0
```

Each line defines one termistor and the last one defines the hotbed termistor. Values we have to set are 1 for TEMP\_SENSOR\_0 and TEMP\_SENSOR\_BED if we use a hotbed.

```
#define TEMP_SENSOR_0 1  
  
#define TEMP_SENSOR_1 0
```

```
#define TEMP_SENSOR_2 0  
  
#define TEMP_SENSOR_BED 1
```

This way we defined thermistors used in our printer. Next we'll configure engines by setting custom number of steps per mm and the rotation's direction. The latter is defined by following lines:

```
#define INVERT_X_DIR true  
  
#define INVERT_Y_DIR false  
  
#define INVERT_Z_DIR true  
  
#define INVERT_E0_DIR false  
  
#define INVERT_E1_DIR false  
  
#define INVERT_E2_DIR false
```

Each line defines spin in X,Y and Z axis and extruder's spin E0, E1 and E2. In most cases, however, only one extruder E0 is used.

*true* and *false* values describe the rotation's direction. Should the engines spin in the wrong direction just switch *true* for *false* and vice versa where necessary. To check if the extruder spins in the right direction just type in M302 command and press *extrude* to see if the extruder spins in the right direction.

Following line allows us to set a desired number of steps per mm.

```
#define DEFAULT_AXIS_STEPS_PER_UNIT {78.7402,78.7402,200.0*8/3,760*1.1}
```

The order of axes is as follows: X, Y, Z, E. The number of steps can be easily calculated using *Josef Prusa's calculator 6* .

When the file is read we can open *Arduino*.



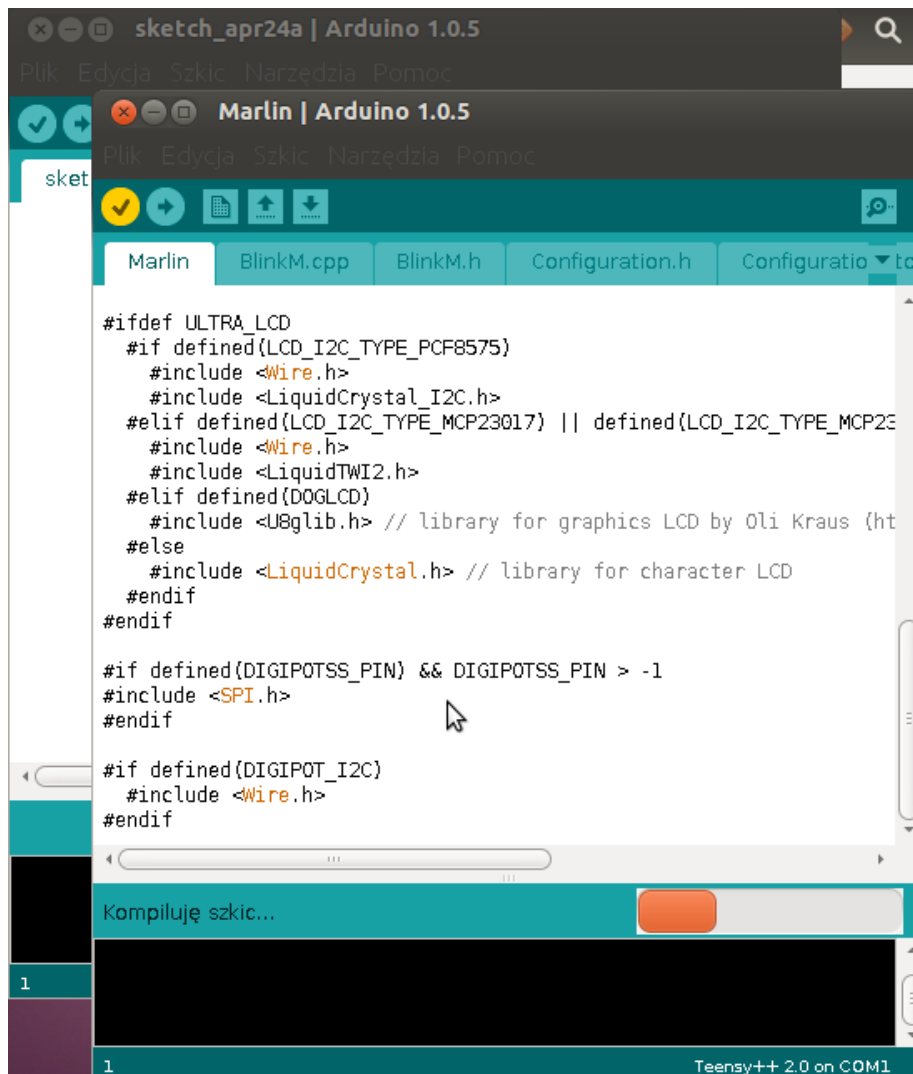
## 5.2 Marlin compilation

To open a Marlin project we need to click *File* → *Open* and choose the Marlin.ino file. The opened file is presented below. 30.



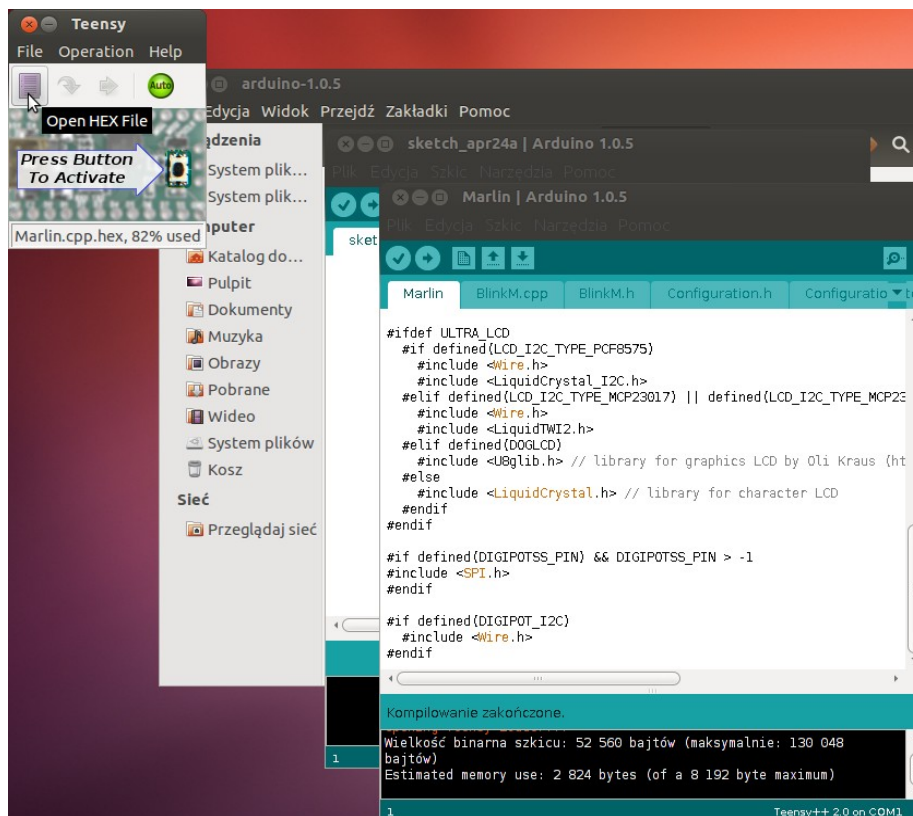
Ilustracja 30: Arduino window

All configuration settings are stored in *Configuration.h* file. In *Tools* → *Board* menu we have to pick Teensy2.0++ and compile the project by clicking the tick button in the top-left of the window (here marked yellow). 31.



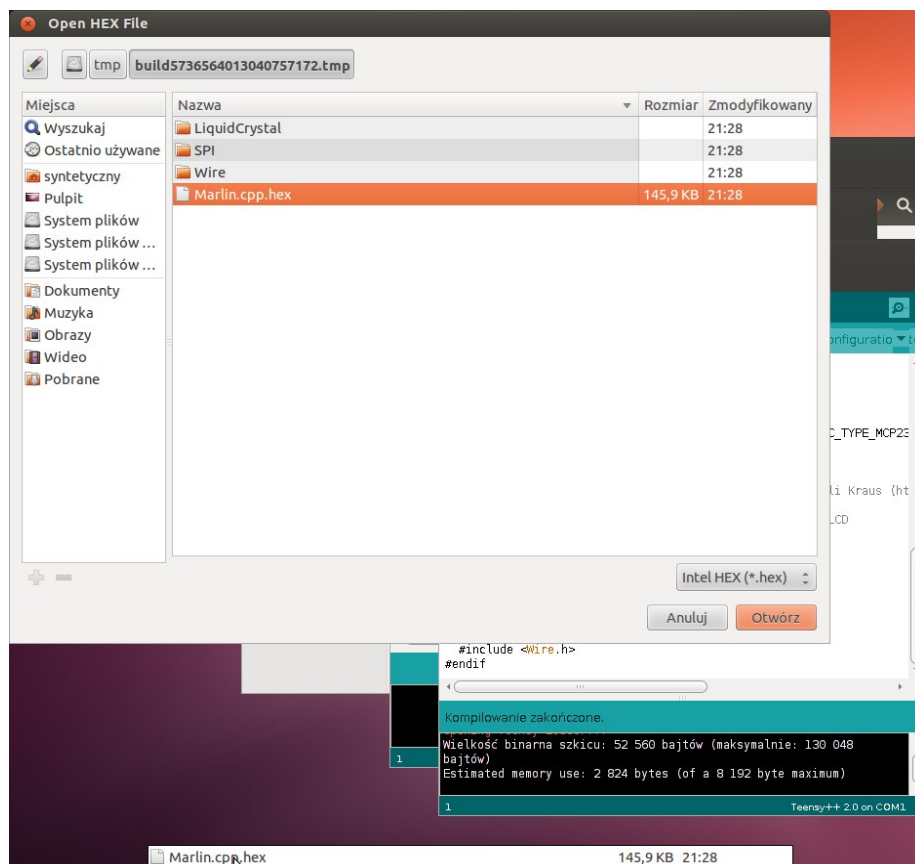
Ilustracja 31: Compiling Marlin

After the compilation has finished a window should open with a *Marlin.cpp.hex* file 32.



*Ilustracja 32: Choosing binary file*

Using the icon below the *File* menu or by going through *File* → *Open Hex File* we can choose *Marlin.cpp.hex* and move it to location of our choice 33.

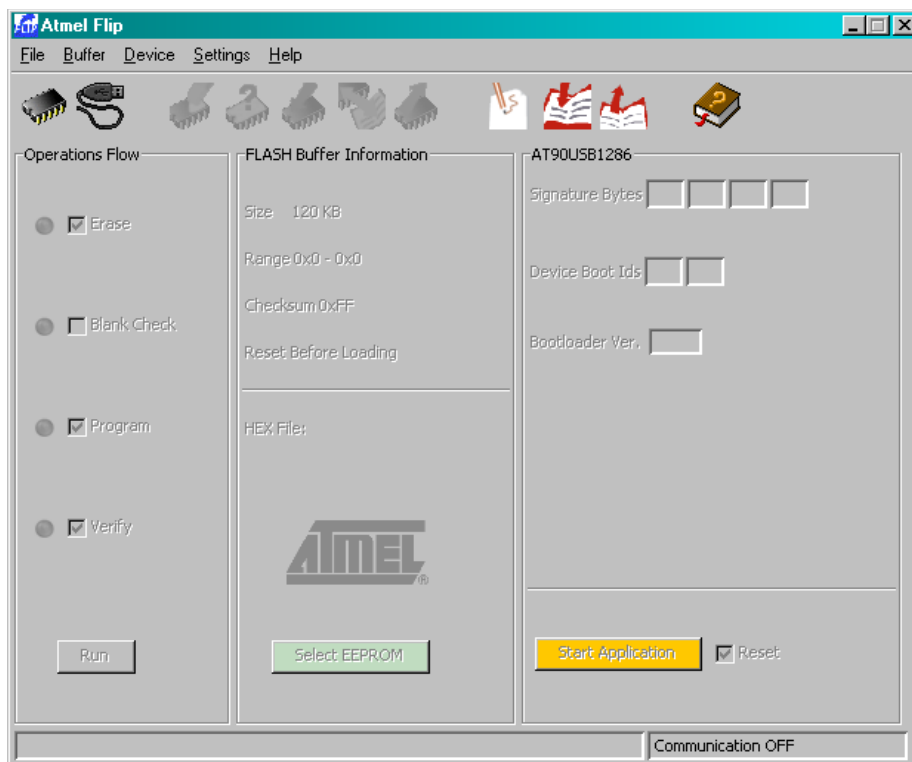


*Ilustracja 33: Store the Marlin.cpp.hex file in location of your preference*

### 5.3 Marlin.cpp.hex installation on Windows

Installing the program on the driver requires *Flip* software. Before we open it, however, KOS electronics has to be plugged into your computer, *BOOT* jumper taken out and *RESET* button held for 5 seconds thus triggering programming mode. If it's the first time you do that then the system should notify you of new hardware. Driver's installation is similar to any other USB hardware. The only difference is that we have to manually

After the installation process completes open Flip.

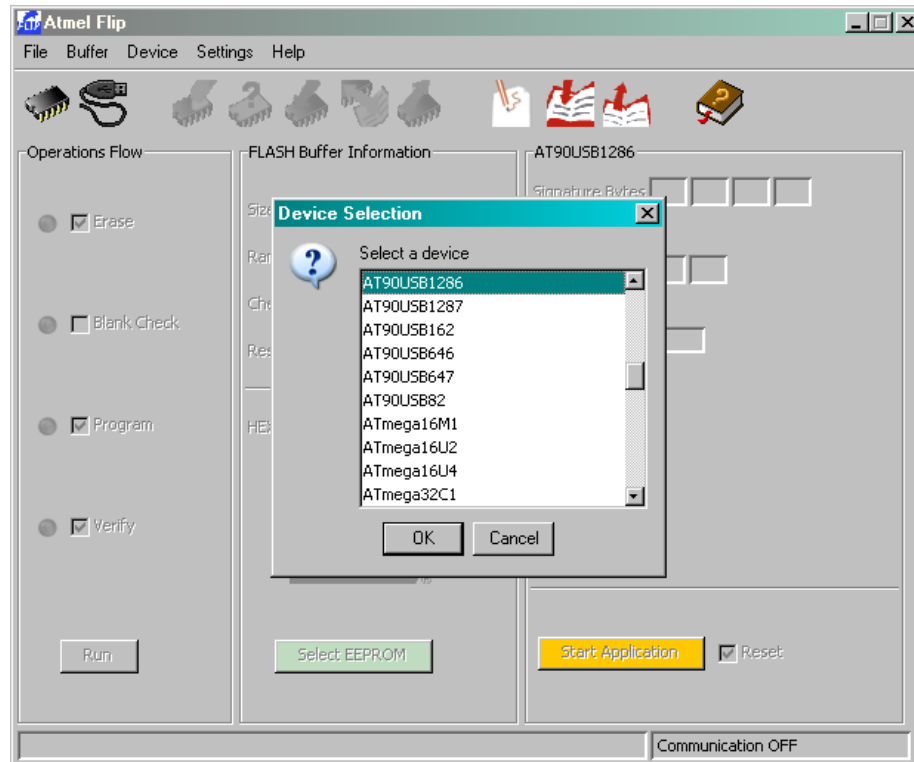


*Ilustracja 34: Flip's main window*

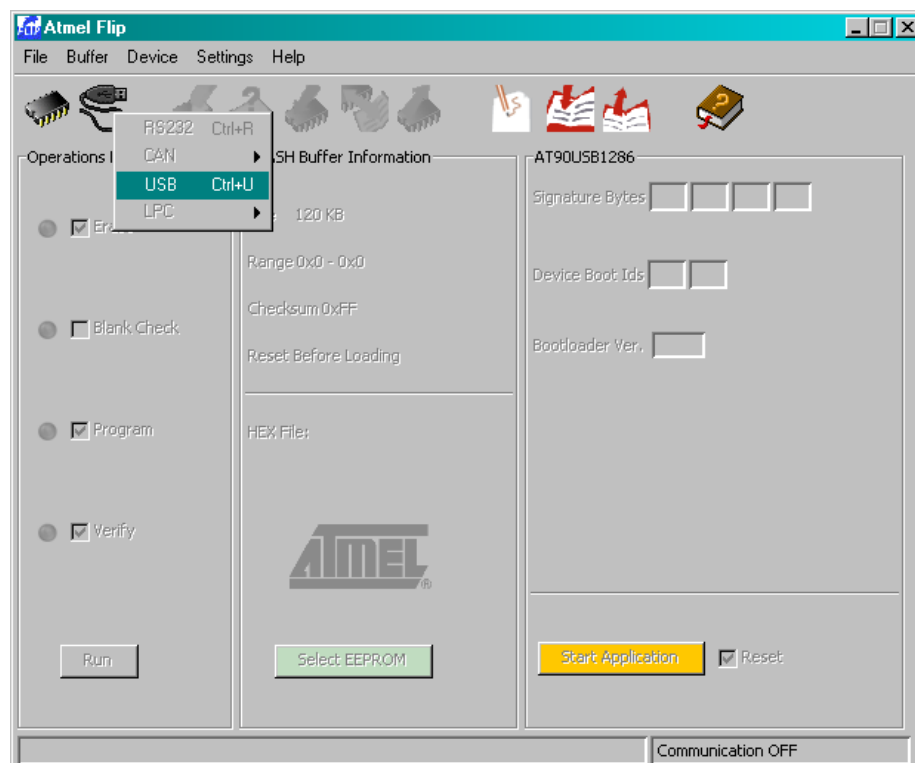
The software requires to be shown the processor to program. We can do that by pressing the first icon on the left.

### ATTENTION!

To install the software correctly make sure which of the two possible processors you have installed on your board. It can be either *AT90USB1286* or *AT90USB1287*.

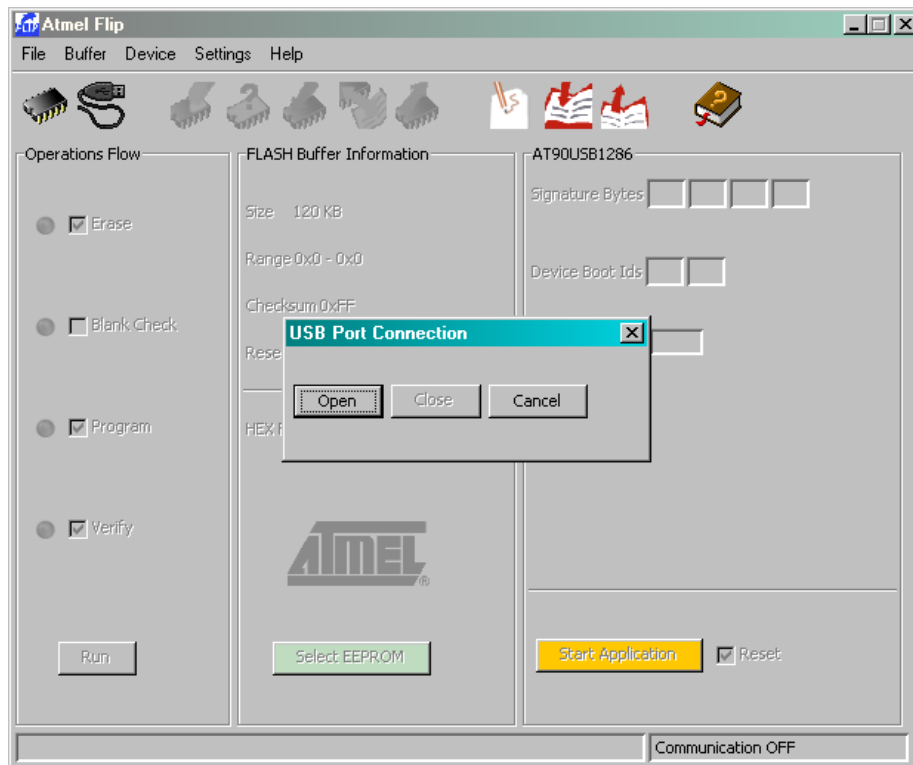


*Ilustracja 35: Processor selection window*



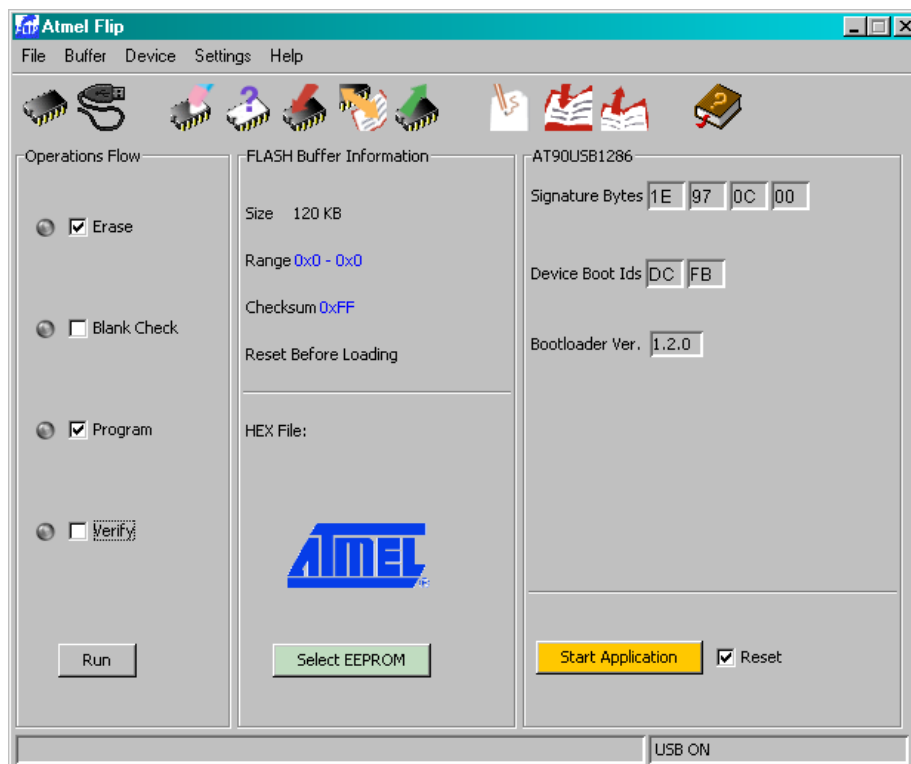
*Ilustracja 36: Selecting USB connection with predefined hardware*

Next use the second icon from the left to establish connection.



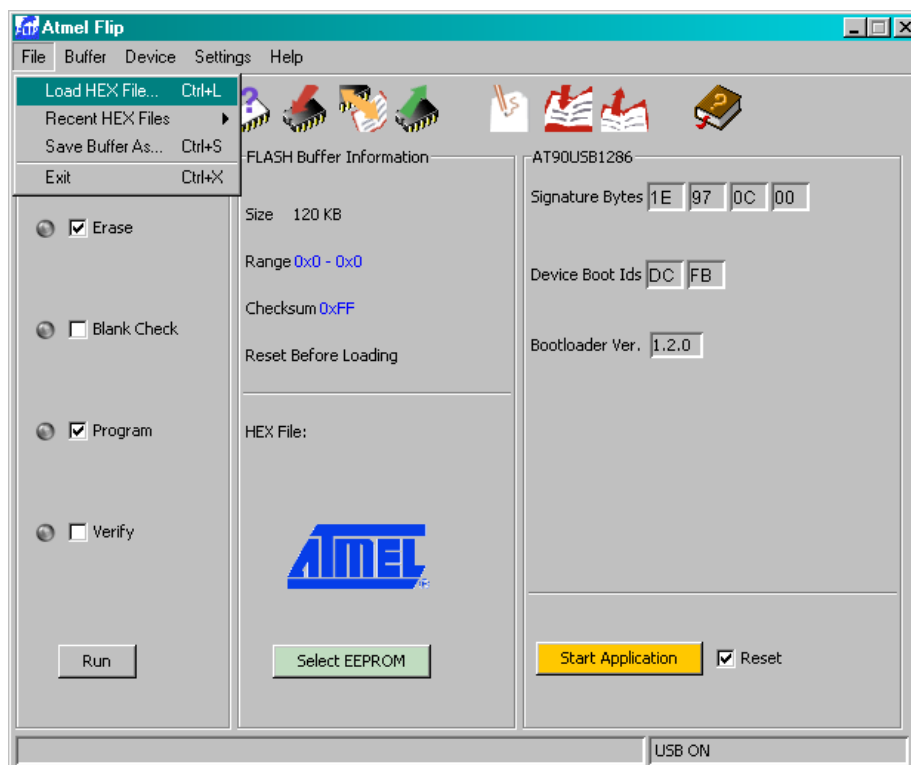
*Ilustracja 37: Open the port for connection*

If the hardware has been installed correctly program should establish connection without a problem. Next select the order of operations you want to execute. In this case pick *Erase* and *Program* in the *Operations Flow* window.



*Ilustracja 38: Setting the order of operations*

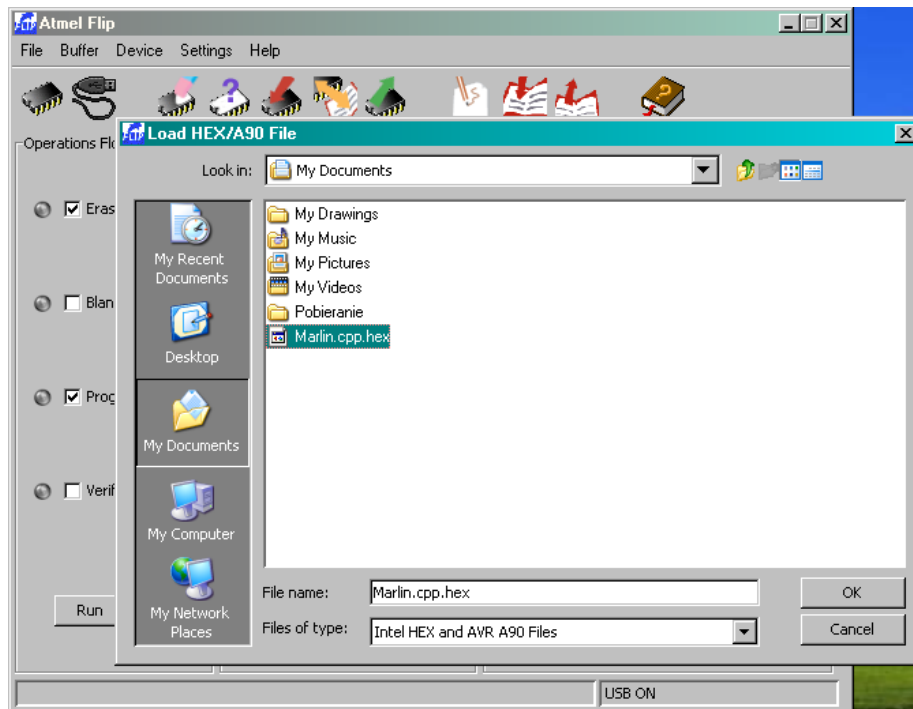
Then select a program you want to have installed into the processor.



*Ilustracja 39: Selecting a program to install*

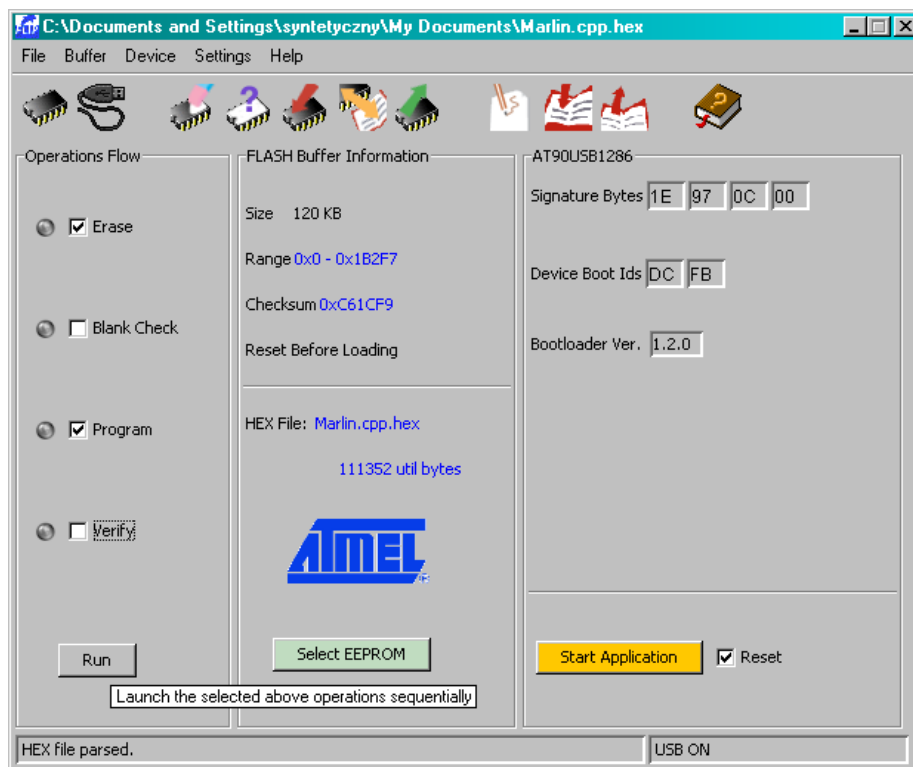


In this case *Marlin.cpp.hex* has been moved to My Documents.



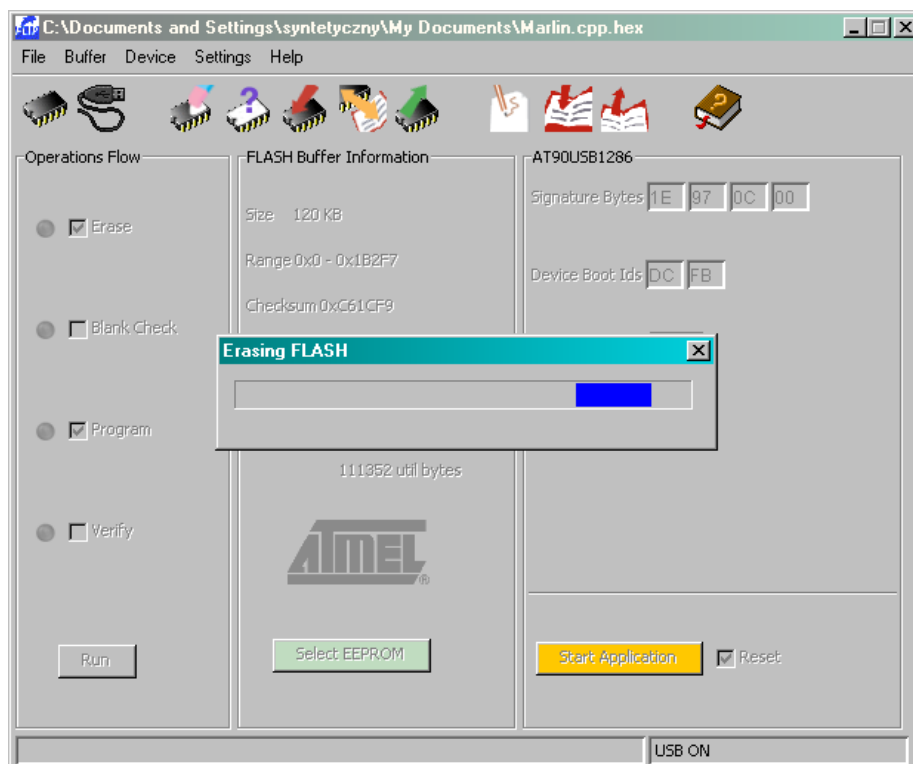
*Ilustracja 40: Selecting your Marlin.cpp.hex file*

After selecting right *Marlin.cpp.hex* file Flip window should look like this:

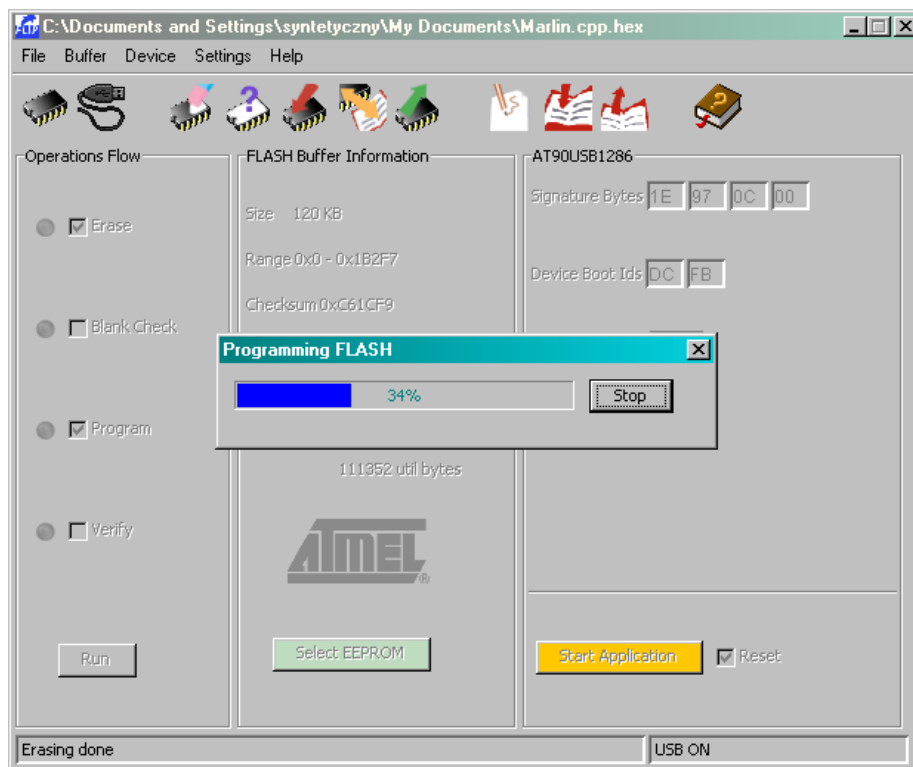


*Ilustracja 41: Flip is ready to install the software*

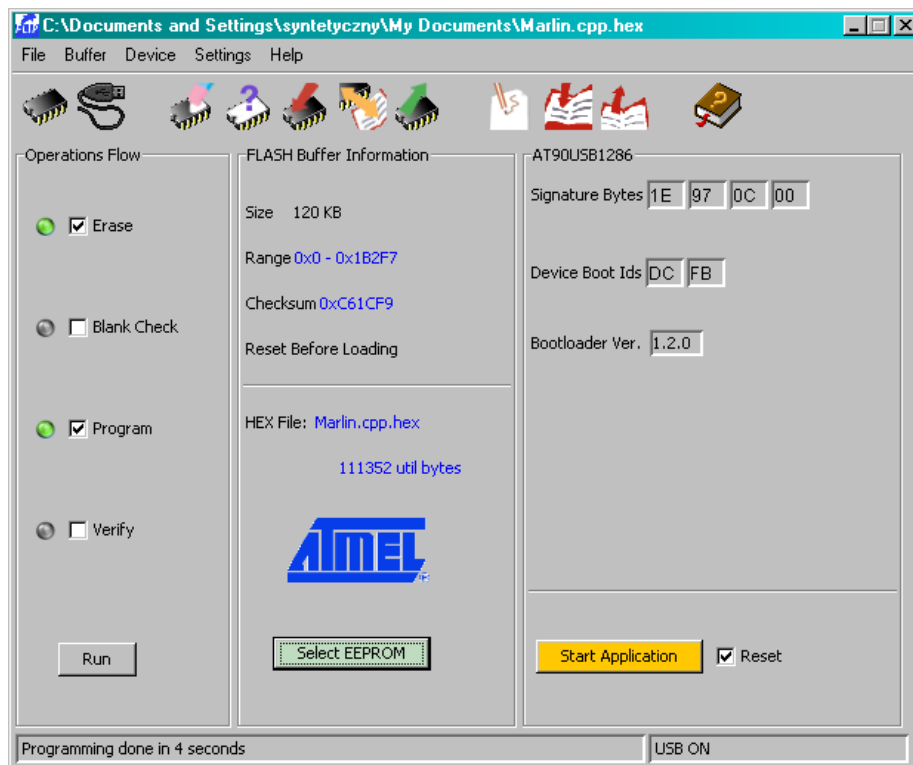
Click *Run* to start erasing and then programming the processor.



*Ilustracja 42: Erasing processor's stored data.*



Ilustracja 43: Programming AT90USB1286/7 processor



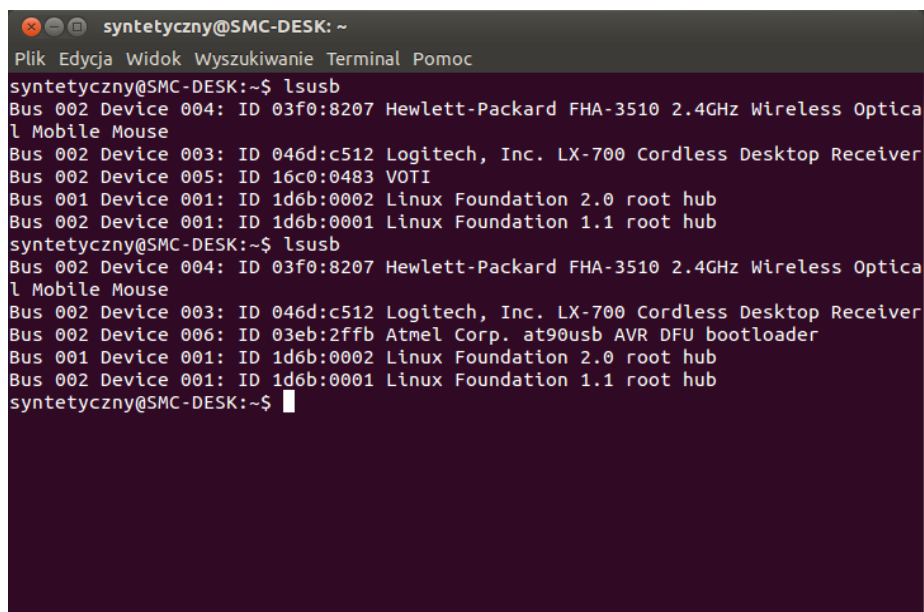
Ilustracja 44: Success!

After putting BOOT jumper back in its place and pressing RESET button once the driver should appear as a virtual port. This means the driver is ready to go.

## 5.4 Marlin.cpp.hex installation on Linux

To program KOS electronics driver first move *Marlin.cpp.hex* to main folder in your system `/home/user_name/` 33. Next summon a terminal using `ctrl + alt + t` and make sure the file is in the main folder using the `dir -l Marlin.cpp.hex` command.

Being sure you're in the right directory plug in the KOS controller using USB cable. Pull out the *BOOT* jumper and hold *RESET* button for 5 seconds to trigger programming mode. Using *lsusb* command check if the programming mode has been triggered.



```
syntetyczny@SMC-DESK: ~  
Plik Edycja Widok Wyszukiwanie Terminal Pomoc  
syntetyczny@SMC-DESK:~$ lsusb  
Bus 002 Device 004: ID 03f0:8207 Hewlett-Packard FHA-3510 2.4GHz Wireless Optical Mobile Mouse  
Bus 002 Device 003: ID 046d:c512 Logitech, Inc. LX-700 Cordless Desktop Receiver  
Bus 002 Device 005: ID 16c0:0483 VOTI  
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub  
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub  
syntetyczny@SMC-DESK:~$ lsusb  
Bus 002 Device 004: ID 03f0:8207 Hewlett-Packard FHA-3510 2.4GHz Wireless Optical Mobile Mouse  
Bus 002 Device 003: ID 046d:c512 Logitech, Inc. LX-700 Cordless Desktop Receiver  
Bus 002 Device 006: ID 03eb:2ffb Atmel Corp. at90usb AVR DFU bootloader  
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub  
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub  
syntetyczny@SMC-DESK:~$
```

*Ilustracja 45: Checking if the driver is in programming mode – it'll switch from VOTI response to Atmel Corp. at90usb*

Next by using *dfu-programmer* installed earlier clean the processor's memory and install the program into the processor.

### ATTENTION!

To successfully install the software make check which of two possible processors you have installed

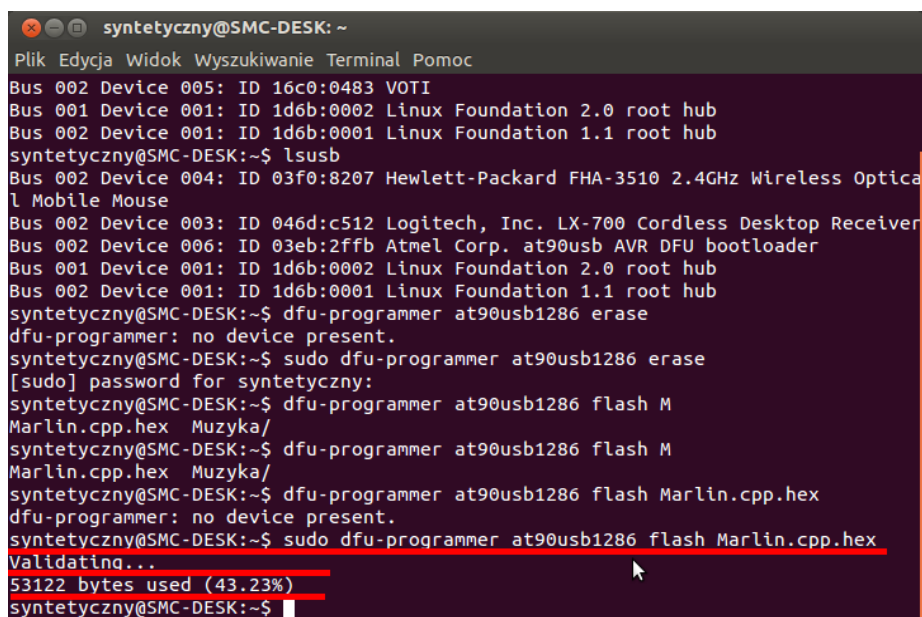
on the board. It can be either *AT90USB1286* or *AT90USB1287*. When you make sure use the following commands:

```
sudo dfu-programmer at90usb1286 erase
```

```
sudo dfu-programmer at90usb1286 flash Marlin.cpp.hex
```

(if your processor is *AT90USB1287* switch 6 for 7 in both commands)

If everything goes well the outcome should look like this:



```
syntetyczny@SMC-DESK: ~  
Plik Edycja Widok Wyszukiwanie Terminal Pomoc  
Bus 002 Device 005: ID 16c0:0483 VOTI  
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub  
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub  
syntetyczny@SMC-DESK:~$ lsusb  
Bus 002 Device 004: ID 03f0:8207 Hewlett-Packard FHA-3510 2.4GHz Wireless Optical Mobile Mouse  
Bus 002 Device 003: ID 046d:c512 Logitech, Inc. LX-700 Cordless Desktop Receiver  
Bus 002 Device 006: ID 03eb:2ffb Atmel Corp. at90usb AVR DFU bootloader  
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub  
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub  
syntetyczny@SMC-DESK:~$ dfu-programmer at90usb1286 erase  
dfu-programmer: no device present.  
syntetyczny@SMC-DESK:~$ sudo dfu-programmer at90usb1286 erase  
[sudo] password for syntetyczny:  
syntetyczny@SMC-DESK:~$ dfu-programmer at90usb1286 flash M  
Marlin.cpp.hex Muzyka/  
syntetyczny@SMC-DESK:~$ dfu-programmer at90usb1286 flash M  
Marlin.cpp.hex Muzyka/  
syntetyczny@SMC-DESK:~$ dfu-programmer at90usb1286 flash Marlin.cpp.hex  
dfu-programmer: no device present.  
syntetyczny@SMC-DESK:~$ sudo dfu-programmer at90usb1286 flash Marlin.cpp.hex  
Validating...  
53122 bytes used (43.23%)  
syntetyczny@SMC-DESK:~$
```

*Ilustracja 46: Programming finished successfully.*

After placing *BOOT* jumper back in it's place and pressing the RESET button once the driver should respond as VOTI after using the *lsusb* command. If it does, then the installation has been successful and the driver is ready to go.

## 6 Sources and additional info about KOS

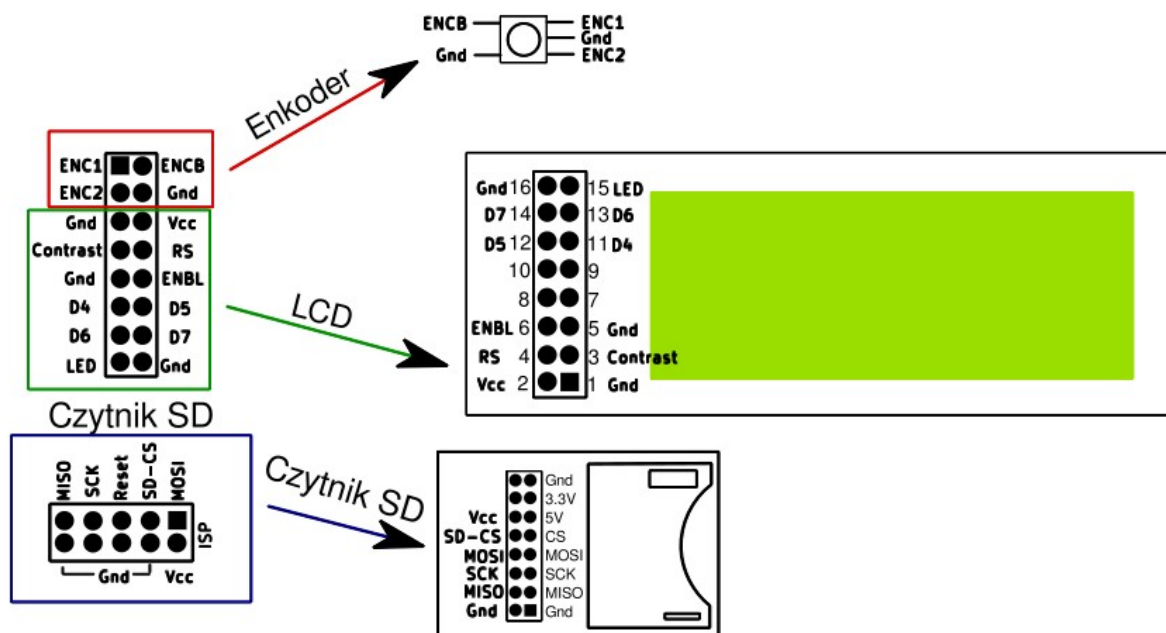
Besides Marlin Repetier software can be used too. For this reason github repositories have been created with forks for this particular firmware which require only compilation for mechanical configuration that we desire. To download those just use the links provided below:

[https://github.com/syntetyczny/Marlin/tree/Marlin\\_KOS](https://github.com/syntetyczny/Marlin/tree/Marlin_KOS)

<https://github.com/syntetyczny/Repetier-Firmware>

There you can find preconfigured software for KOS electronics. Those configurations were created with SD cards, encoders and LCD screen in mind. Diagram below shows where those devices are to be plugged in.

### 6.1 LCD, Encoder, SD-card assembly diagram



Ilustracja 47: Diagram depicting where LCD screen, Encoder and SD card reader are to be connected

## 7 Bibliography

1. serial\_install.exe from 10.02.2014, [www.pjrc.com/teensy/serial\\_install.exe](http://www.pjrc.com/teensy/serial_install.exe)
2. Arduino installer from 23.04.2014, <http://arduino.cc/en/Main/Software>
3. Teensyduino from 23.04.2014, [http://www.pjrc.com/teensy/td\\_download.html](http://www.pjrc.com/teensy/td_download.html)
4. Flip for Windows from 23.04.2014, <http://www.atmel.com/tools/flip.aspx>
5. Marlin from 30.04.2014, <https://github.com/ErikZalm/Marlin>
6. Josef Prusa's calculator from 30.04.2014, <http://calculator.josefprusa.cz>