**C64 Keyboard Controlled Kernal Switch Rev. 1**

**Programming Guide**

# Introduction

The microcontroller on the Kernal switch has to be programmed to function properly. There are several ways of transferring the program into IC1.

Since the program is based on the Arduino IDE, the program can be considered to consist of three parts:

* the Arduino Bootloader (optiboot\_atmega328.hex)
* the fuses
* the compiled sketch (c64kbksw.ino)

The bootloader will enable the Arduino IDE to transfer (compiled) sketches via the serial cable into the ATmega328P microcontroller on the Kernal switch.

The fuses are the configuration of the ATmega328P. If those are not set properly, the Kernal switch will not work like desired.

The sketch is the source code of the program. It needs to be compiled and uploaded by the Arduino IDE (=*Integrated Development Environment)*.

# One Pass Programming

It is possible to transfer the bootloader, the compiled sketch and the fuses in one pass. That requires a suitable programmer (like the Atmel ICE, an SDK500 based programmer or another supported programming hardware). Further on, the Atmel Studio software is required. It is the IDE for writing software for Atmel controllers and programming the chips.

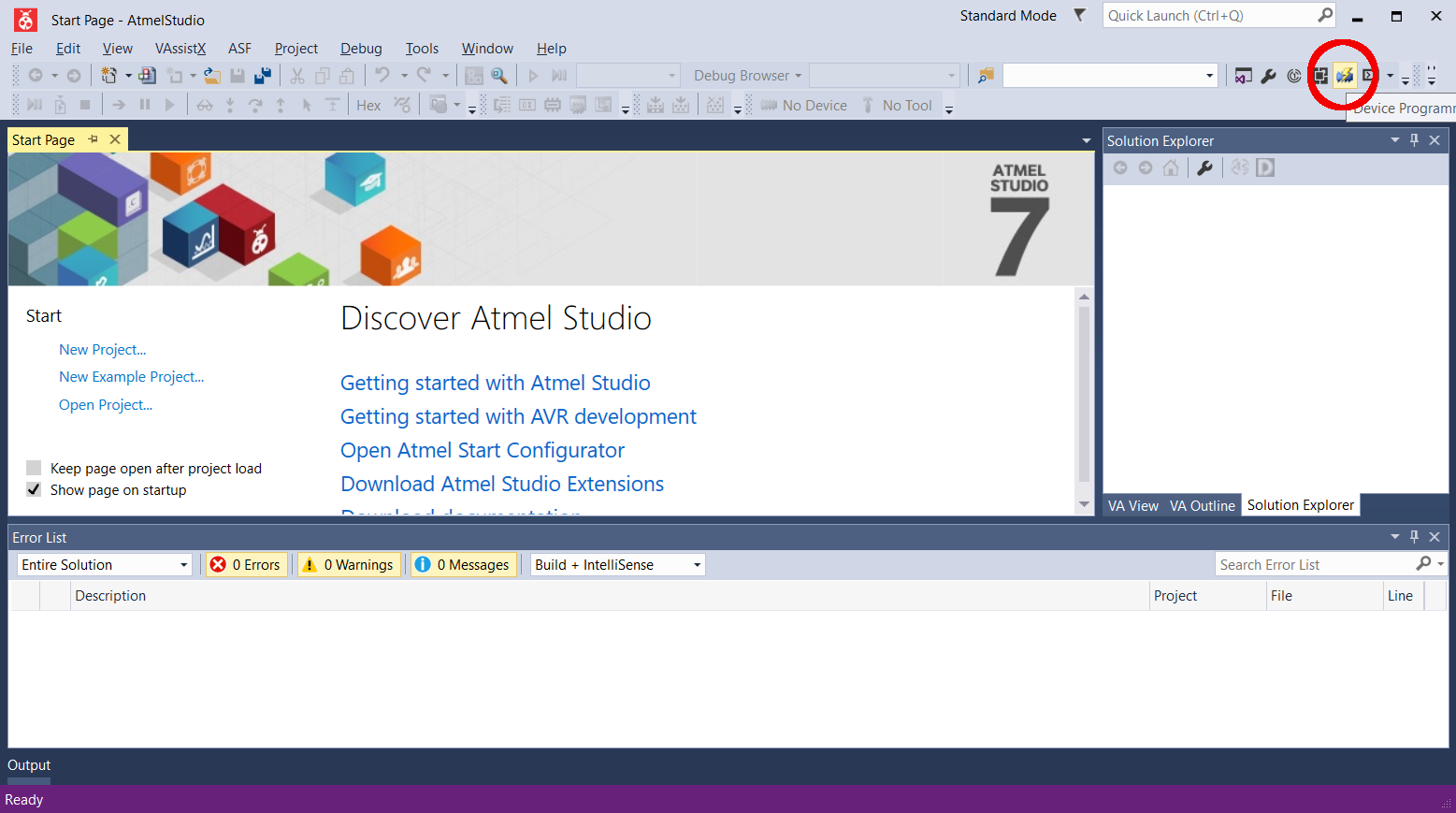


Figure 1: Atmel Studio IDE

The programming data is c64kbksw\_v0\_0.elf (or a later version). It was read out from an already programmed Kernal switch with a programmer and stored.

## Step 1

Start the Atmel Studio IDE and select the Device Programming Tool as shown in Figure 1.

## Step 2

Configure the tool (=the programmer), the device (ATmega328P), the interface (ISP = in-system programming) and click “Apply”, like in Figure 2.

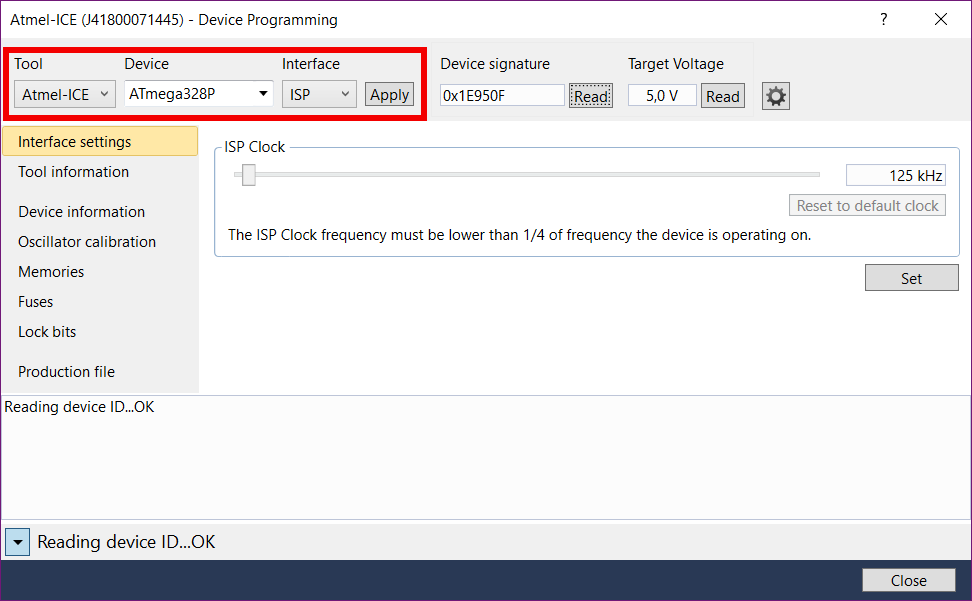


Figure 2: Device Programming

When you click “Read”, the Device signature should be displayed and the Target Voltage should be shown (about 5V). In case, it does not, the module needs to be powered externally with 5V (connector J3). It can be done with the USB/Serial adaptor and the serial cable, or just a bench power supply.

## Step 3

Click “Production File”, then select the ELF production file (*c64kbksw\_v0\_0.elf* or later version) from the location, where you have stored it. Check “Flash” and “Fuses”, also “Erase memory before programming” and “verify programmed content”, like shown in Figure 3.

Now, click the “Program” button. The programming will start.

While the programming, the status is reported in the lower part of the window. This is marked yellow in Figure 4. Erasing, programming Flash and Verifying, Programming Fuses and Verifying should all be reported “OK”.

Now, the program (bootloader and fuses included) is completely transferred to the ATmega328P. The module should be fully functional now. Since it contains the bootloader, the Kernal switch can be modified with the Arduino IDE and a serial cable.

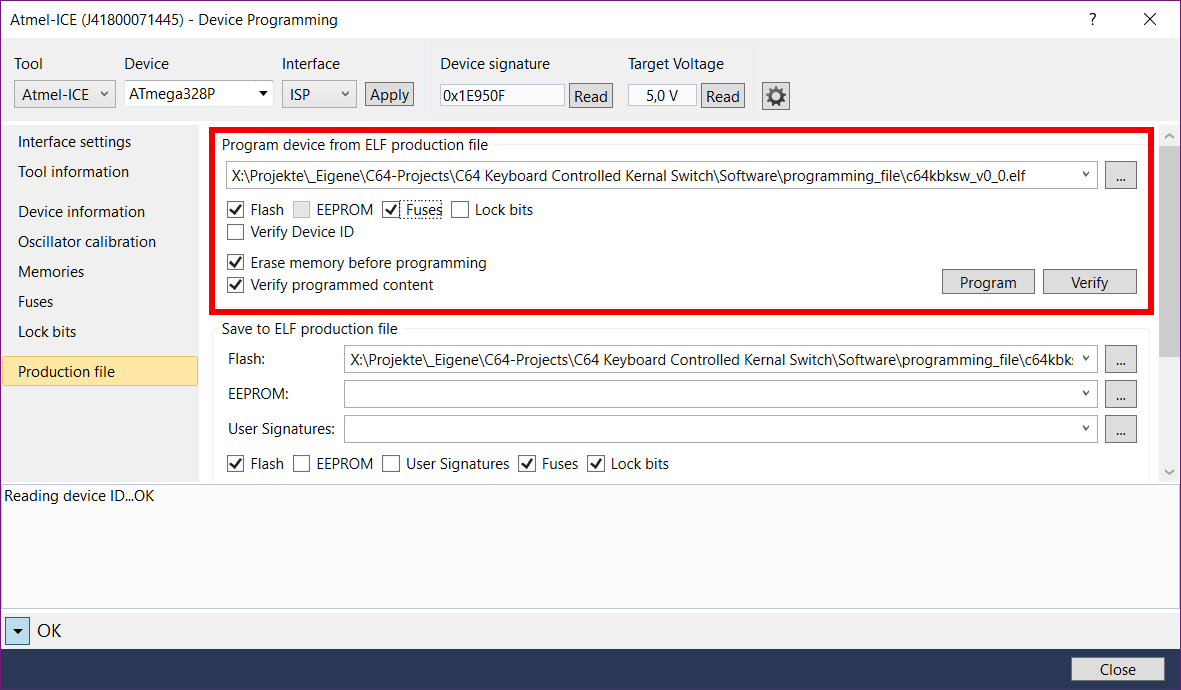


Figure 3: Production File settings

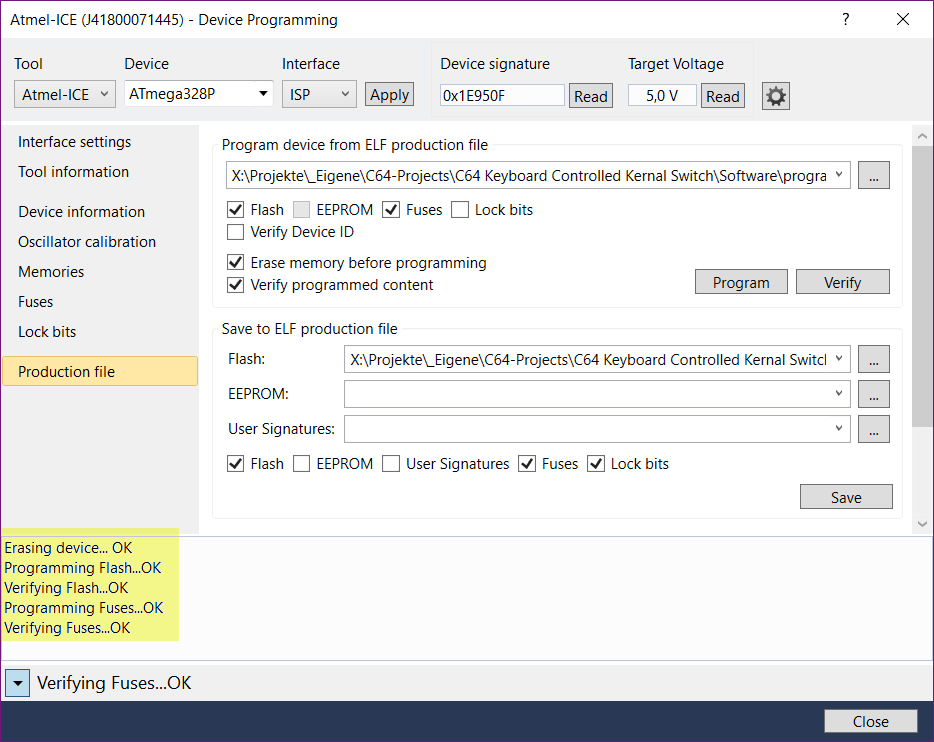


Figure 4: Programming the device

The fuses of the ATmega328P should be set like this:

* EXTENDED: 0xFD
* HIGH: 0xDE
* LOW: 0xFF

This can be checked with the Device Programming tool like in Figure 5.

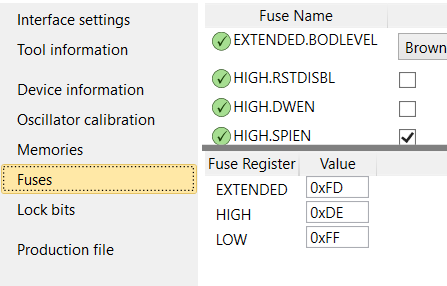


Figure 5: Fuses

# One Pass Programming TL866

The ATmega328P can be programmed either with a TQFP32 adaptor or via the ICSP port. In both cases, the program (c64kbksw\_v0\_0.hex or a later version) has to loaded to the programming buffer (the format is Intel HEX) and the fuses have to be set manually (see Figure 6). The file contains the bootloader and the compiled sketch. No further programming is required.

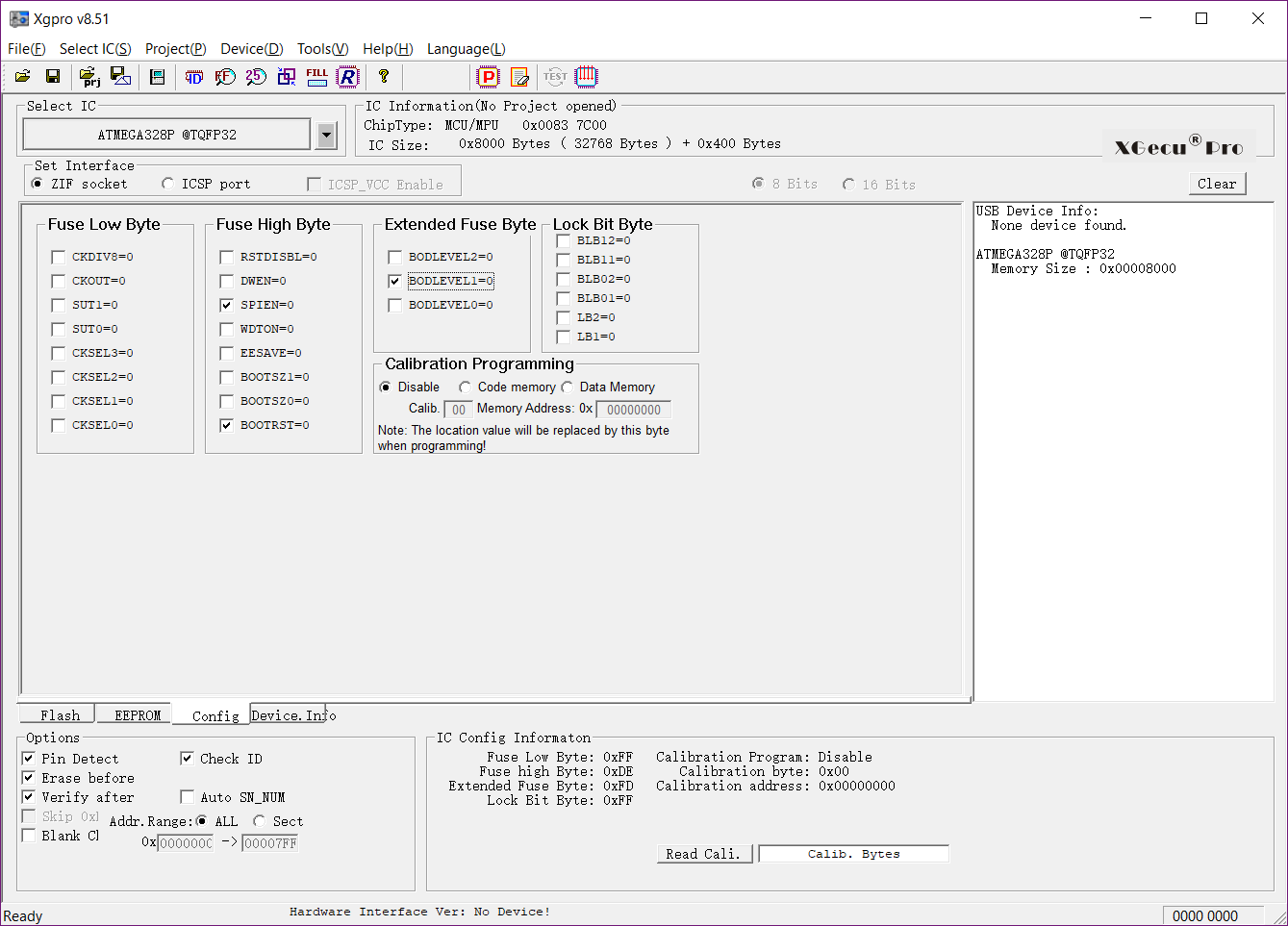


Figure 6: Correct setting of the fuses

This setting can be found in the “Config” tab.

Without a TQFP32 adaptor, the processor can be programmed in-system. The TL866II+ and the TL866A provide such an ICSP port. As a cable, a 6-wire Dupont cable will work (see Figure 9).

The connections should be made according to Figure 8.

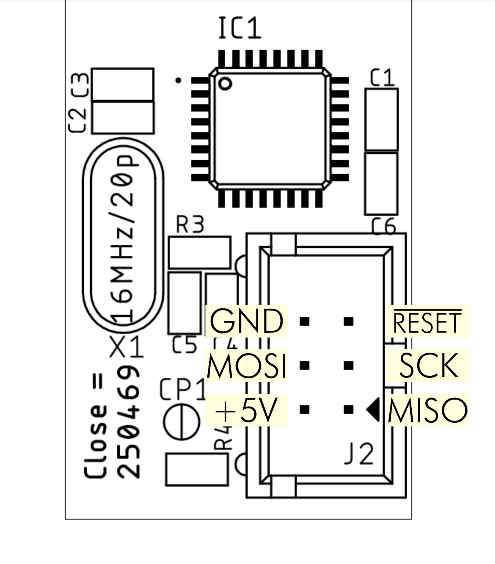


Figure 7: Pinout of the ICSP connector J2

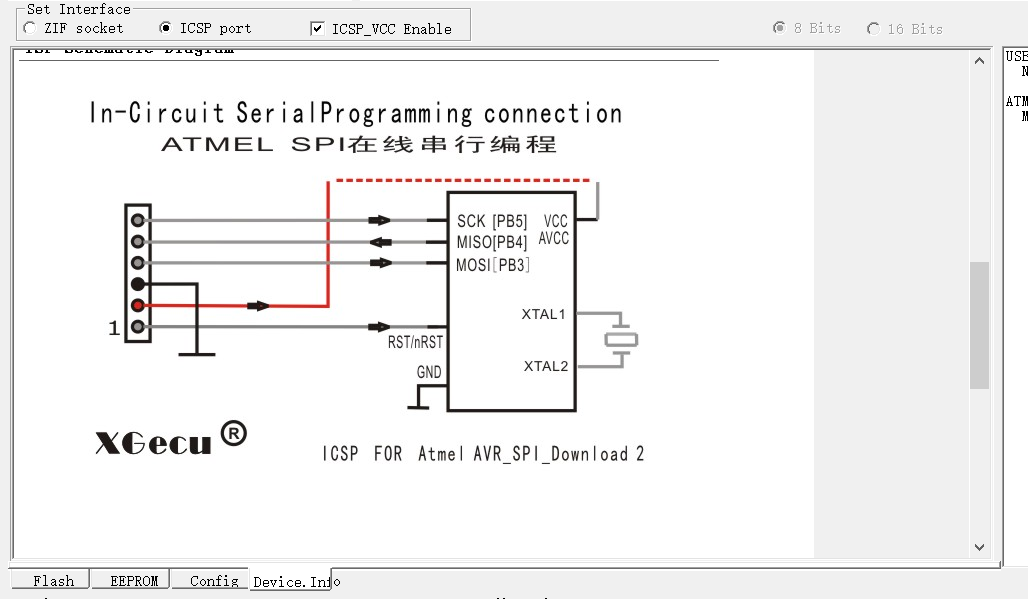


Figure 8: ICSP connection from TL866II+ to ATmega328P

The Dupont cable needs to be connected correctly, the connection has to be double checked to prevent harming the module or the programmer. In case, the repeated programming is intended, it might make sense to build a cable with a real 6x1 and 3x2 pin Dupont crimp housing (the 1pin crimp housings of the cable can be uninstalled).

When programming the controller, make sure, the code memory and the fuses are checked (Figure 10).

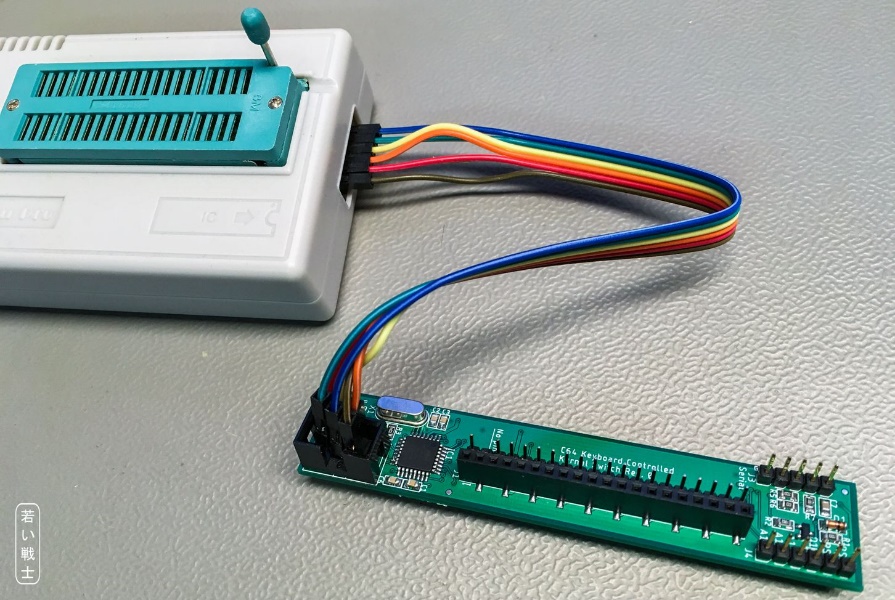


Figure 9: ICSP with a Dupont cable

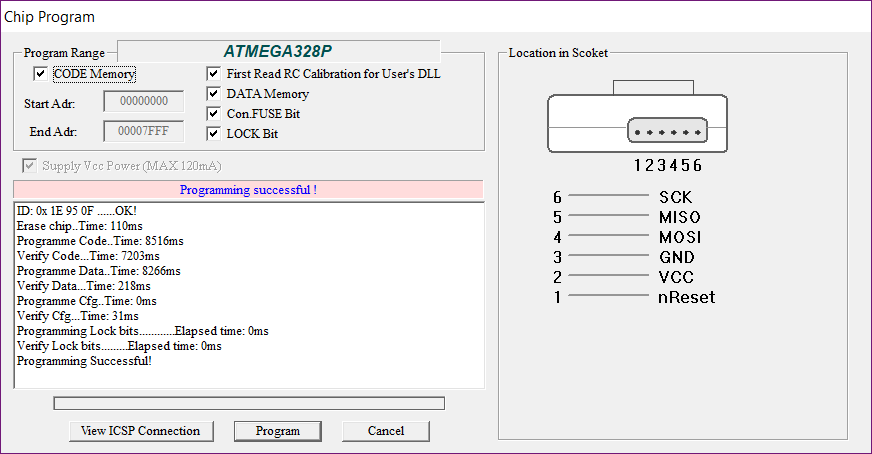


Figure 10: Programming the ATmega328P with the TL866II+

# Programming the bootloader from the Arduino IDE

With a supported programmer or a programmer made from an Arduino Uno, the Arduino bootloader can be programmed into IC1. The fuses will be set correctly by the software.

The cheaper In-Circuit programmers for the Atmel processors usually mimic the STK500 development board. The functions described can be found in the Tools menu. Select the right programmer type, for STK500 you also have to set the right port (this is not the COM-Port provided by the USB/Serial adaptor). Once the Programmer and the port are set, connect the module to this programmer and click “Burn Bootloader” in the Tools menu of the Arduino software.

Instructions how to set up an Arduino Uno for programming the bootloader can be found by Google.

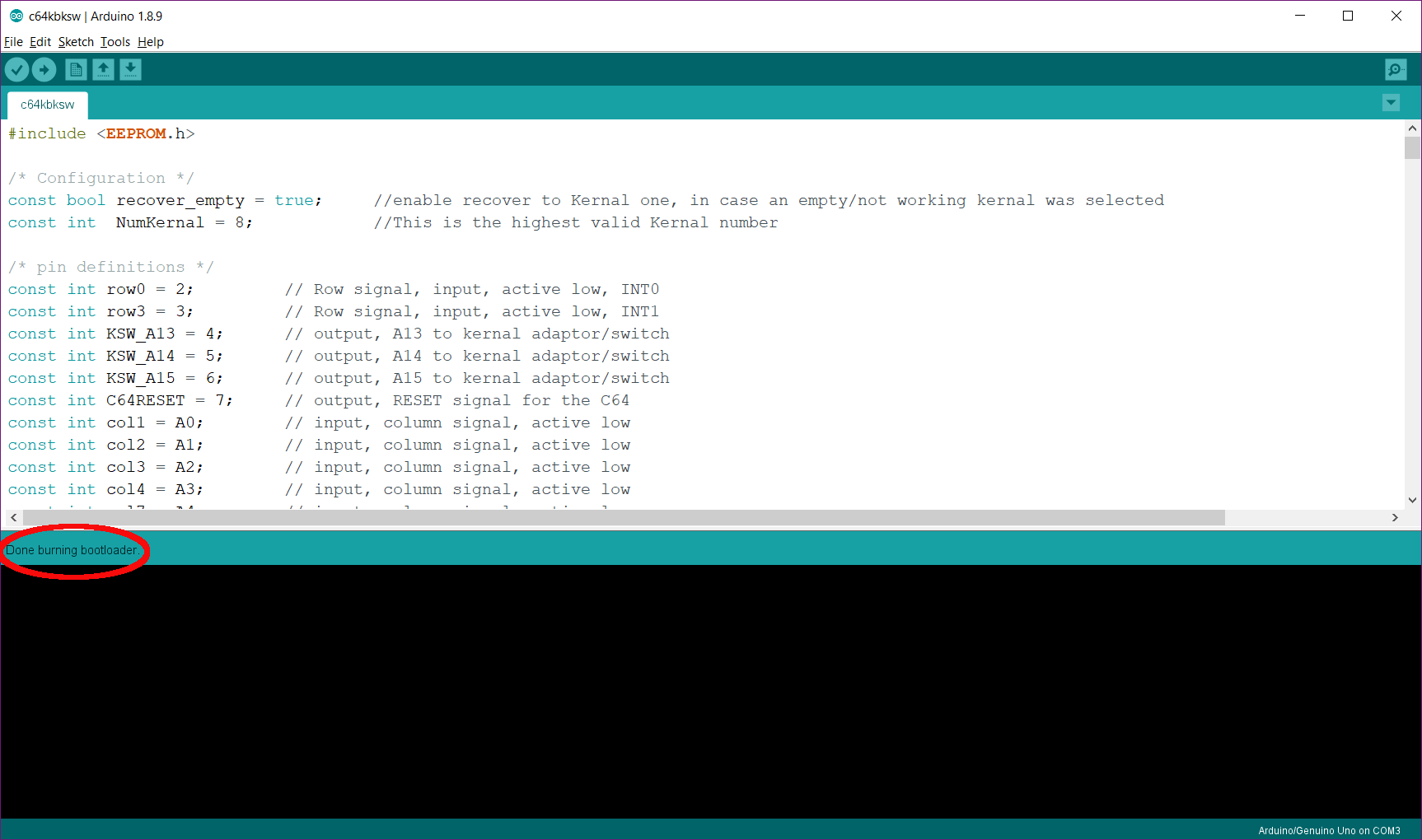


Figure 11: Burning the bootloader with the Arduino software

Note: the Atmel-ICE Programmer did not work properly for burning the bootloader from the Arduino software. If you own this programmer, you can program the bootloader with the Atmel Studio software.

# Programming the bootloader from Atmel Studio

The Arduino IDE software contains the programming (\*.hex) file for the optiboot bootloader, which is used for Arduino Uno etc.

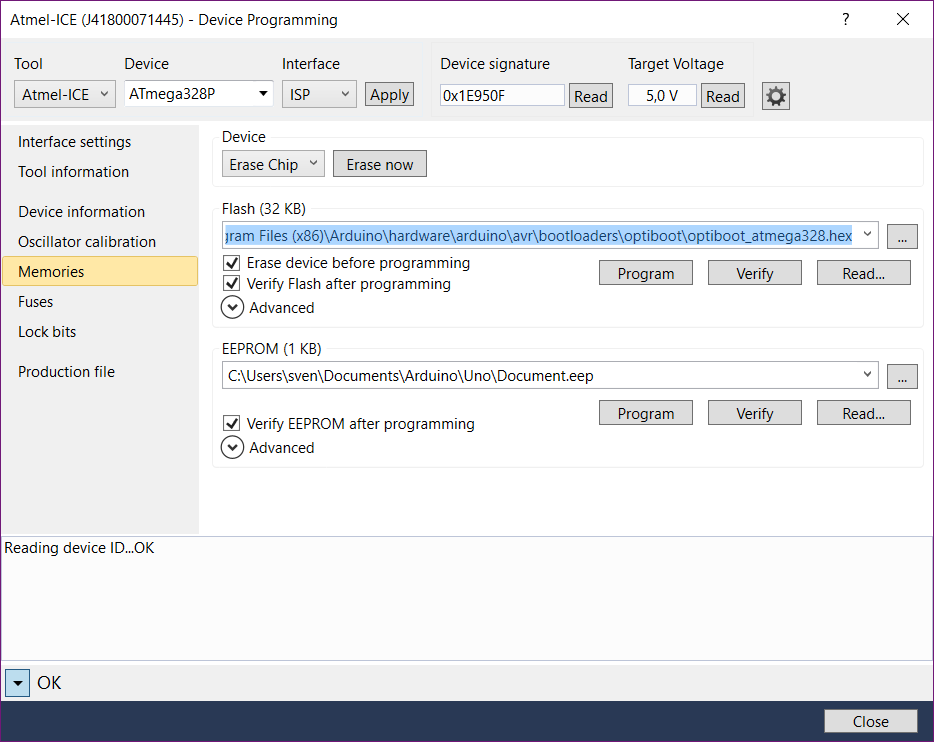


Figure 12: Programming the optiboot bootloader

It can be found here:

C:\Program Files (x86)\Arduino\hardware\arduino\avr\bootloaders\optiboot

The file name is:

optiboot\_atmega328.hex

The hex-file can be loaded into the program buffer. Follow the steps like shown in Figure 1 and Figure 2, then select memories and for the Flash, select the file optiboot\_atmega328.hex from the location described before.

The fuse settings are not contained in the \*.hex file, they have to be programmed separately (see Figure 5).

# Programming the bootloader with TL866II+

The bootloader that can be found in the Arduino software installation (see previous chapter) can also be programmed with the TL866II+ or TL866A. Please refer to Figure 6 to Figure 10. Remember, it is the file optiboot\_atmega328.hex. The fuses are set exactly the same and need to be programmed.

# Compiling and Uploading an Arduino sketch

The complete software for the Kernal Switch consists of two parts: the optiboot bootloader and the complied (and actually linked) Arduino sketch (c64kbksw.ino).

Once the bootloader is burnt, the sketch can be transferred via a serial cable as described in document number 128-6-04-00 (Wiring) and a required USB/Serial adaptor.

First the sketch has to be copied to the Arduino sketch folder (usually …\Documents\Arduino\c64kbksw). The sketch has to be opened (File → Open). After the Kernal switch is connected to the USB/serial converter (and this to the PC) via the serial cable and the power is on, the Board (in the Tools menu) has to be set to Arduino/Genuino Uno. Then the Port has to be selected. The USB/Serial adapter will set up as a COM-Port. If there are no other COM-Ports on the PC, it is the only one that will be listed in the (Tools → Port) list. In case you are not sure, press (windows key X) select the Device Manager and look it up in the COM & LPT section (top of the list). It should be something like “USB Serial Port (COMxx)” (xx is the number, you have to remember). In case there is more than one: disconnect the adapter and watch which port disappears and then connect it again and watch what appears.

Once you have found out and set the correct COM Port, the Kernal Switch is connected and powered, you can hit the “right arrow” (= upload icon, 2nd from the left). The sketch will be compiled and the uploaded. Alternatively, select “Upload” from the “Sketch” menu.

If it does not upload, first check the +5V from the USB-Serial adaptor. It can be measured between the +5V and the GND pin of the ICSP connector J2 (see Figure 7). Then, check the cable and finally check the soldering on the PCB (use a magnifying glass or microscope).

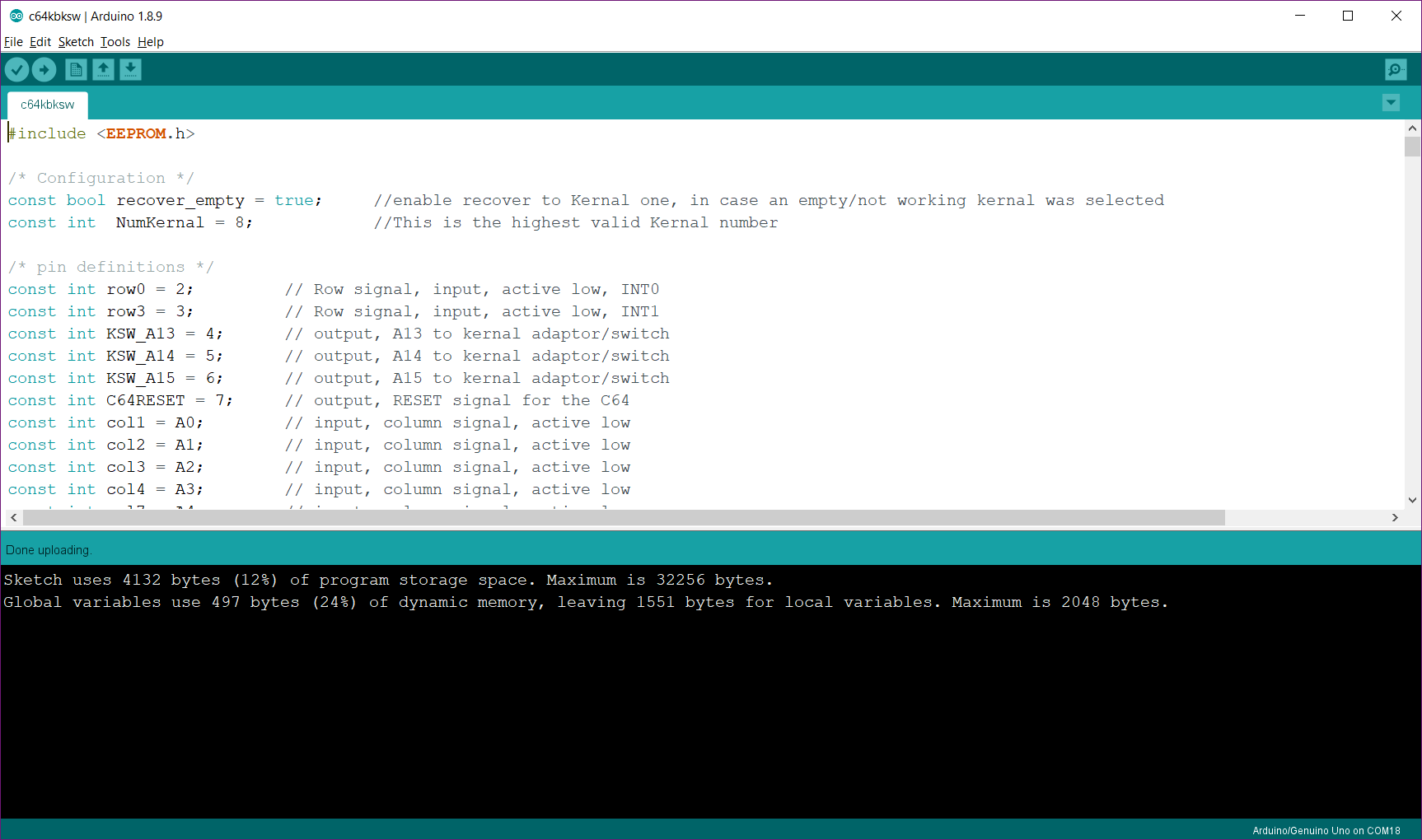


Figure 13: Compiling and uploading the sketch