



How to program your own Vicol-Audio volume controller

We use and recommend [usbtinyisp](#) as a very good programmer.

It is a very popular AVR programmer, very well documented, easy to make, works great with avrdude, is AVRStudio-compatible and tested under Windows, Linux and MacOS X.

Unless you want to build your own usbtinyisp from scratch, a kit with all components is available at <http://www.adafruit.com/products/46>

Detailed instructions on how to install this device under Windows, Linux and MacOS X can be found here <http://www.ladyada.net/make/usbtinyisp/avrdude.html>

Just a small parenthesis

```
{  
In case you are are compiling avrdude from source, as we did, you may get an error at first run  
(libtinfo.so.5 not found). This may be solved by creating a symbolic link inside lib directory:  
cd /lib  
ln -s libncurses.so.5.9 libtinfo.so.5  
}
```

Next step is to get an Atmel AVR fuse calculator. Use Google to search for one, or use this one

<http://www.engbedded.com/fusecalc/>

In our particular case (ATMega16) AVRDUDE arguments are **-U lfuse:w:0xd4:m -U hfuse:w:0xd9:m**

Connect your usbtinyisp with ISP connector to Vicol-Audio volume-controller board and power up the board.

Connect your usbtinyisp to PC – please note that if you run Windows you need

<http://www.ladyada.net/make/usbtinyisp/download.html> , no drivers are needed for Linux/Unix/MacOS X.

Test your communication with microcontroller (command you need to type is in red, output in blue):

```
root@tibix:/# avrdude -c usbtiny -p m16
```

```
avrdude: AVR device initialized and ready to accept instructions
```

```
Reading | ##### | 100% 0.01s
```

```
avrdude: Device signature = 0x1e9403
```

```
avrdude: safemode: Fuses OK
```

```
avrdude done. Thank you.
```



Next step is to write fusebit (command you need to type is in red, output in blue):

```
root@tibix:/# avrdude -c usbtiny -p m16 -U lfuse:w:0xd4:m -U hfuse:w:0xd9:m
```

```
avrdude: AVR device initialized and ready to accept instructions
```

```
Reading | ##### | 100% 0.02s
```

```
avrdude: Device signature = 0x1e9403  
avrdude: reading input file "0xd4"  
avrdude: writing lfuse (1 bytes):
```

```
Writing | ##### | 100% 0.02s
```

```
avrdude: 1 bytes of lfuse written  
avrdude: verifying lfuse memory against 0xd4:  
avrdude: load data lfuse data from input file 0xd4:  
avrdude: input file 0xd4 contains 1 bytes  
avrdude: reading on-chip lfuse data:
```

```
Reading | ##### | 100% 0.00s
```

```
avrdude: verifying ...  
avrdude: 1 bytes of lfuse verified  
avrdude: reading input file "0xd9"  
avrdude: writing hfuse (1 bytes):
```

```
Writing | ##### | 100% 0.02s
```

```
avrdude: 1 bytes of hfuse written  
avrdude: verifying hfuse memory against 0xd9:  
avrdude: load data hfuse data from input file 0xd9:  
avrdude: input file 0xd9 contains 1 bytes  
avrdude: reading on-chip hfuse data:
```

```
Reading | ##### | 100% 0.00s
```

```
avrdude: verifying ...  
avrdude: 1 bytes of hfuse verified
```

```
avrdude: safemode: Fuses OK
```

```
avrdude done. Thank you.
```

Now change directory to where your firmware is located and proceed with last step.



Write firmware to chip (command you need to type is in red, output in blue):

```
root@tibix:/home/tibi//firmware# avrdude -c usbtiny -p m16 -U  
flash:w:ATM16_SHT_R2R_LCD_Vicol.hex
```

```
avrdude: AVR device initialized and ready to accept instructions
```

```
Reading | ##### | 100% 0.02s
```

```
avrdude: Device signature = 0x1e9403  
avrdude: NOTE: FLASH memory has been specified, an erase cycle will be performed  
To disable this feature, specify the -D option.  
avrdude: erasing chip  
avrdude: reading input file "ATM16_SHT_R2R_LCD_Vicol.hex"  
avrdude: input file ATM16_SHT_R2R_LCD_Vicol.hex auto detected as Intel Hex  
avrdude: writing flash (14830 bytes):
```

```
Writing | ##### | 100% 8.88s
```

```
avrdude: 14830 bytes of flash written  
avrdude: verifying flash memory against ATM16_SHT_R2R_LCD_Vicol.hex:  
avrdude: load data flash data from input file ATM16_SHT_R2R_LCD_Vicol.hex:  
avrdude: input file ATM16_SHT_R2R_LCD_Vicol.hex auto detected as Intel Hex  
avrdude: input file ATM16_SHT_R2R_LCD_Vicol.hex contains 14830 bytes  
avrdude: reading on-chip flash data:
```

```
Reading | ##### | 100% 7.77s
```

```
avrdude: verifying ...  
avrdude: 14830 bytes of flash verified
```

```
avrdude: safemode: Fuses OK
```

```
avrdude done. Thank you.
```

Disconnect usbtinyisp and power off/on Vicol-Audio volume-controller.
Congratulations, you volume-controller is ready for use with a new firmware !