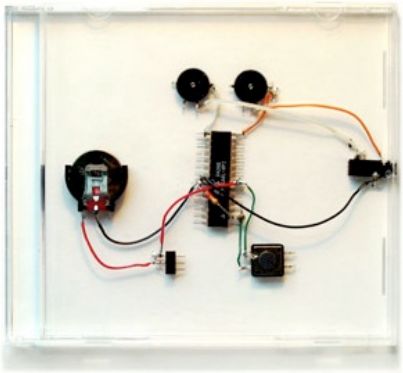
clubtransmediale08: xxxxx-workshops: one bit music /f0
-presentation-

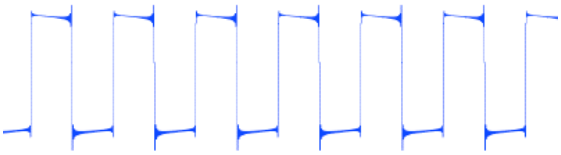
1bit?

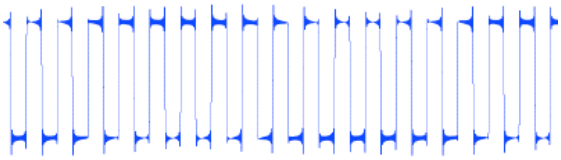
inspiration from tristan perich and others



(picture from tristan's www.onebitmusic.com)

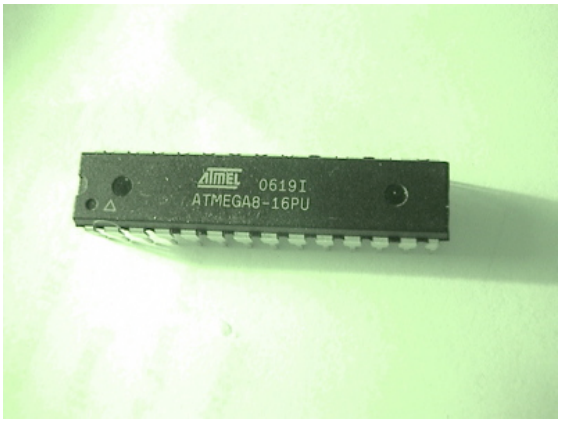
dogma: minimal circuit (battery, chip, speaker), only squarewaves (pwm, bit-bang)
+ volume control and switches
can we at all write music with this?





ATmega8

8bit microcontroller with 8kb program memory (8192 bytes)
one of the most used in the avr family (www.atmel.com)
allround and fairly cheap (~€2)
clock speed up to 16mhz



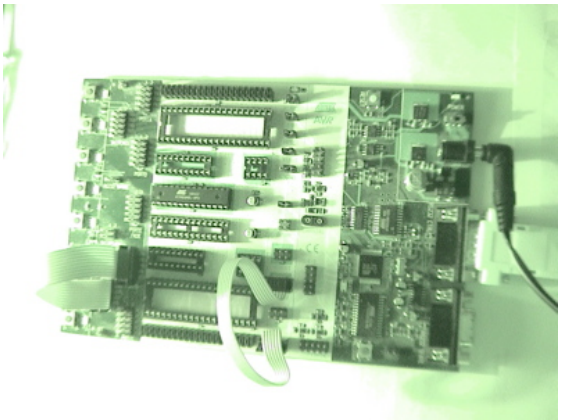
microcontroller ≈ tiny computer
usually you run them on 5 volts
ATmega8L - the low-power version that can run directly on 2 aaa batteries (3 volts)
many digital/analog inputs/outputs, usart/spi communication, internal timers/counters/oscillator, eeprom/sram memory, registers, interrupts, stack etc.

programming

```c
#include <avr/io.h>
int main(void) {
    DDRB= 0xff;
    int i;
    for(;;) {
        for(i= 0; i<0xffff; i++)
            PORTB= 4;
        for(i= 0; i<0xffff; i++)
            PORTB= 0;
    }
    return 0;
}
```

write programs for them in c, c++ or assembly
upload with a programmer (serial/parallel/usb)
'burns' a binary file (.hex) onto the chip's flash memory (aka firmware)
erases what was there before. write/erase cycle - thousands of times

buy or build a programmer yourself. here stk500
building is simple if you have a parallel port, a little bit harder with serial or usb



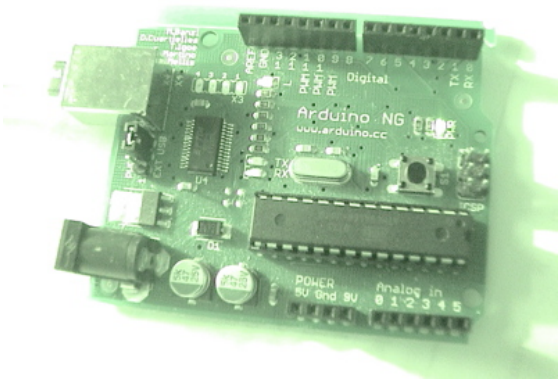
open source avr-gcc and avrdude package:
gcc is used to compile and link your source code into a binary (.hex)

avrdude is used to upload the binary to the chip (burn it to flash memory)
avrdude also lets you set fuses on the chip (e.g. set the internal oscillator's speed)

Arduino

we'll use it here for experimenting. thanks to our sponsors (www.tinker.it)
builds upon the ATmega168 chip
very similar to ATmega8 - pin compatible, more memory, more pwm outputs

arduino is basically chip + usb-to-rs232, voltage regulator, crystal, leds, button, connectors + bootloader - firmware acting as programmer, talks to the computer
arduino-0010 application: simplified coding in c



very good way to start with microcontrollers
also used in many serious projects
open source
loads of tutorials and intro material online (www.arduino.cc)
different boards available: ng, bt, lilypad, mini, diecimila, barebone, freeduino etc
buy finished, as kit or build from scratch

drawbacks for us in this project: size, cost, not the low-power version chip, bootloader steals memory

2 techniques

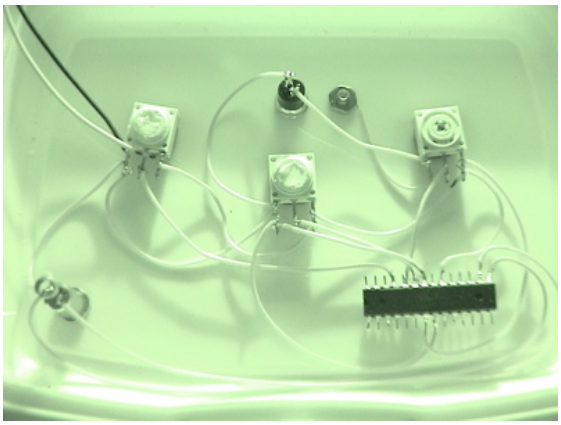
bit-banging: flip pin high/low manually
very simple method - same as the blinking led example
any digital output pin can be used
difficult to code anything more interesting than simple melodies
only one thread and delaying blocks other tasks (e.g. sensor input)

pulse width modulation: constant frequency but varying pulsewidth
used as analog output - average voltage
interrupt driven (sort of multitasking)
internal counter counts up to a max value, when reached it interrupts the main code and calls a function (isr)
more interesting sounds but still very hard to program music

demo

onebit_bitbang09

monijonsyn



'music for battery, chip and speaker'