Chiptunes on an Attiny4
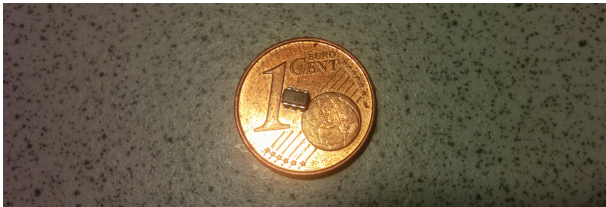
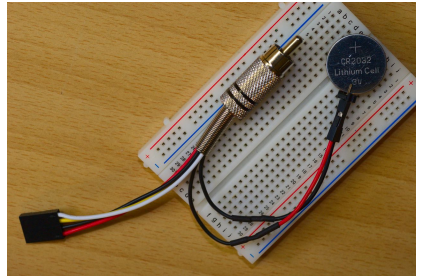or: *Bitshift variations in ASMinor*

Tobias Girstmair, //gir.st/

Inspiration

Rob Miles' *Bitshift variations in C minor*
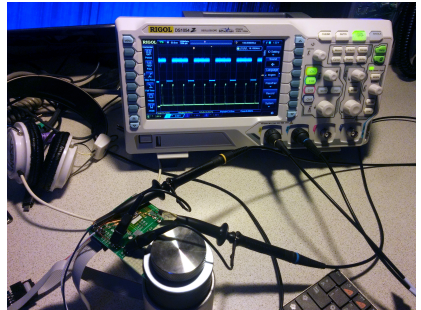

DoJoe's *The Noiseplug*

Development Environment

- 512 Bytes ROM
- 32 Bytes RAM
- 16 Registers
- 1 Timer/Counter

- GNU `avr-gcc` toolchain
- USBasp programmer
- Breakout Board
- Patience.

Strategy

Process

- hand-transcribed Rob's C code into AVR assembly
- incremental translation into "fake ASM" (C macros)
- `./a.out | head -c31457280 | diff 1.5h-orig.pcm && git commit -am "new version"`
- finally, removing *fakeasm* idiosyncrasies and implementing init

Objectives

- fit Attiny4 (512 bytes progmem)
- 4Mhz operation (< 500 cycles; for coin cell)
- play a recognisable rendition of the tune

Strategy

Process

- hand-transcribed Rob's C code into AVR assembly
- incremental translation into "fake ASM" (C macros)
- `./a.out | head -c31457280 | diff 1.5h-orig.pcm && git commit -am "new version"`
- finally, removing *fakeasm* idiosyncrasies and implementing init

Objectives

- fit Attiny4 (512 bytes progmem)   ✓ (446 bytes)
- 4Mhz operation (< 500 cycles; for coin cell)
- play a recognisable rendition of the tune

Strategy

Process

- hand-transcribed Rob's C code into AVR assembly
- incremental translation into "fake ASM" (C macros)
- `./a.out | head -c31457280 | diff 1.5h-orig.pcm &&`
  `git commit -am "new version"`
- finally, removing *fakeasm* idiosyncrasies and implementing init

Objectives

- fit Attiny4 (512 bytes progmem)   ✓ (446 bytes)
- 4Mhz operation ($<$ 500 cycles; for coin cell)   ✓ (466-ish)
- play a recognisable rendition of the tune

Strategy

Process

- hand-transcribed Rob's C code into AVR assembly
- incremental translation into "fake ASM" (C macros)
- `./a.out | head -c31457280 | diff 1.5h-orig.pcm && git commit -am "new version"`
- finally, removing *fakeasm* idiosyncrasies and implementing init

Objectives

- fit Attiny4 (512 bytes progmem)   ✓ (446 bytes)
- 4Mhz operation (< 500 cycles; for coin cell)   ✓ (466-ish)
- play a recognisable rendition of the tune   ∼ (not perfect)

Challenges

MUL, DIV, MOD is hard

- ▸ no hardware; standard routines are slow and large
- ▸ only need small number of multipliers/divisors
- ▸ unrolled functions with hard-coded parameters
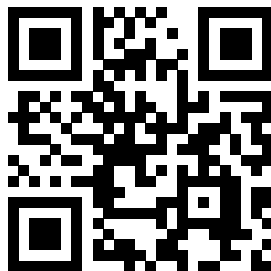- ▸ takes a 3rd of the time

Debugging is hard

- ▸ Never done something comparable before
- ▸ Atmel Studio is proprietary / Windows only
- ▸ No attiny10 core in `simavr` or `simulavr`
- ▸ Oscilloscope and trial-and-error

Thanks!

Get in touch: `tobi@isticktoit.net` or GSM 6699
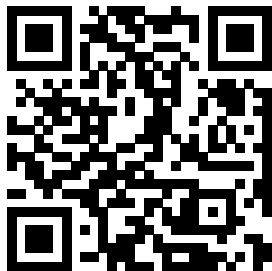
gir.st/chiptunes.htm

xkcd.wtf

- ▶ `http://txti.es/bitshiftvariationsincminor`
- ▶ `https://github.com/dop3j0e/noiseplug`