
Riding The Wave
Using radare2 — by pancake
CON 2018

Use Cases

R2 is known to be a very versatile tool.

It was born as a forensics tool and evolved into debugging, static analysis, emulation, data and code visualization, scripting, decompilation..

But there's still an area to explore...

DISCLAIMER

I'm not a Musician or an expert on Sound processing.

But I wanted to learn a bit on those topics and used r2 as an excuse.

As always (R)

The Waves

- We all love messing with bytes and hexadecimal values
- Chiptune music trackers and editors user interfaces look like hex editors
- Tweaking r2 a bit we can use it to make music

But why stick to just make music?

- isn't r2 a re framework? Provide tools to find and identify waves
- Parse music binary files, do and reverse stego, etc
- Improve visualization commands for data analysis

Sounds Chips

- Some of them are coprocessors that create different types of waves like sid or a basic pc speaker.
- Others provide a dsp interface, which basically plays what we feed in using a specific binary format to define the waves data.

- Everybody have tried this already
  - $ cat /dev/urandom > /dev/dsp
  - $ cat /dev/sda > /dev/dsp

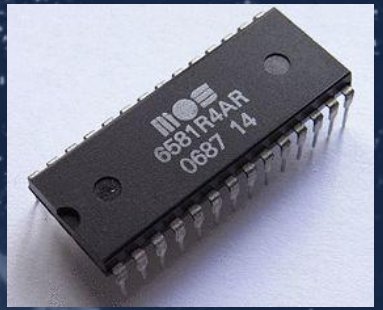
Uses

- Creating music with code
- Demoscene
- Electronic music / Chiptune
- Mostly for artistic purposes

Playing music in DOS

- Use rasm2 to assemble a .COM and use DOSBOX to run the PoC
- http://www.intel-assembler.it/portale/5/make-sound-from-the-speaker-in-assembly/8255-8255-8284-asm-program-example.asp

The SID



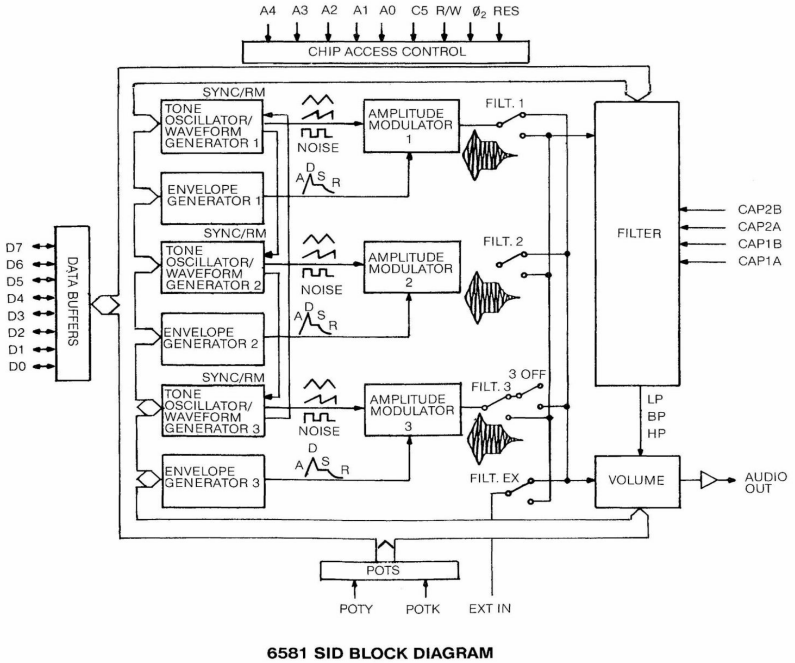
Probably the most famous chip in the chiptune scene. C64=

- SID 6581/8580

(3× osc, 4× wave, filter, ADSR, ring)

http://archive.6502.org/datasheets/mos_6581_sid.pdf

https://chipmusic.org/forums/topic/8189/basic-guide-to-programming-the-sid-with-assembly-language/



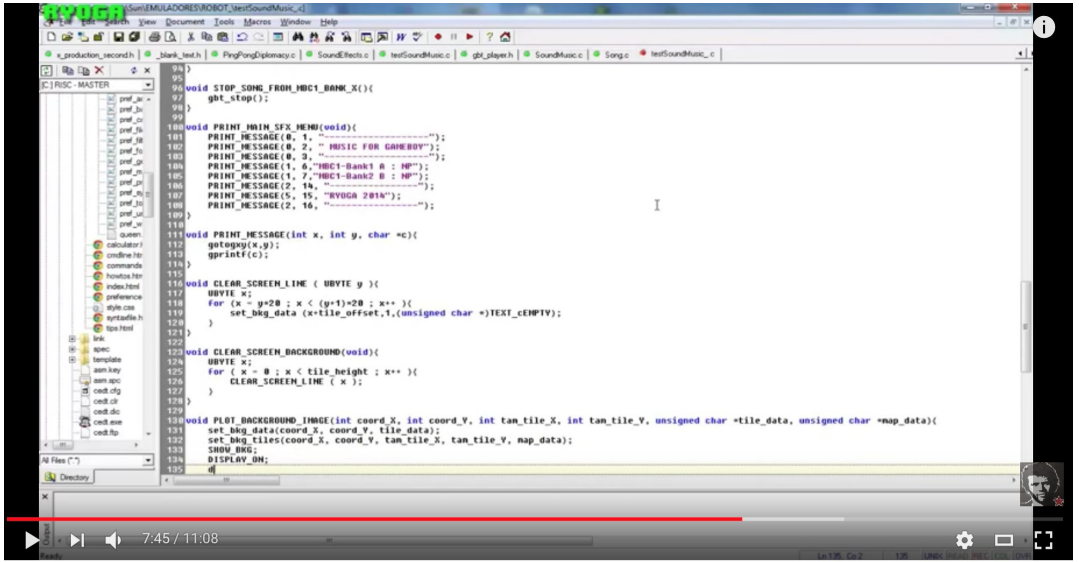
6581 SID BLOCK DIAGRAM

The Gameboy

Music was created programatically or implementing a tracker, but everything was done internally.

http://www.chrisantonellis.com/files/gameboy/gb-programming-manual.pdf

https://www.youtube.com/watch?v=n3Jqv8t7JU0



SOUND PROGRAMMING FOR GAMEBOY (BANK SWITCHING) - GBDK: Game Boy C Programming [GAMEBOY HOMEBREW]

1,443 views

LIKE DISLIKE SHARE

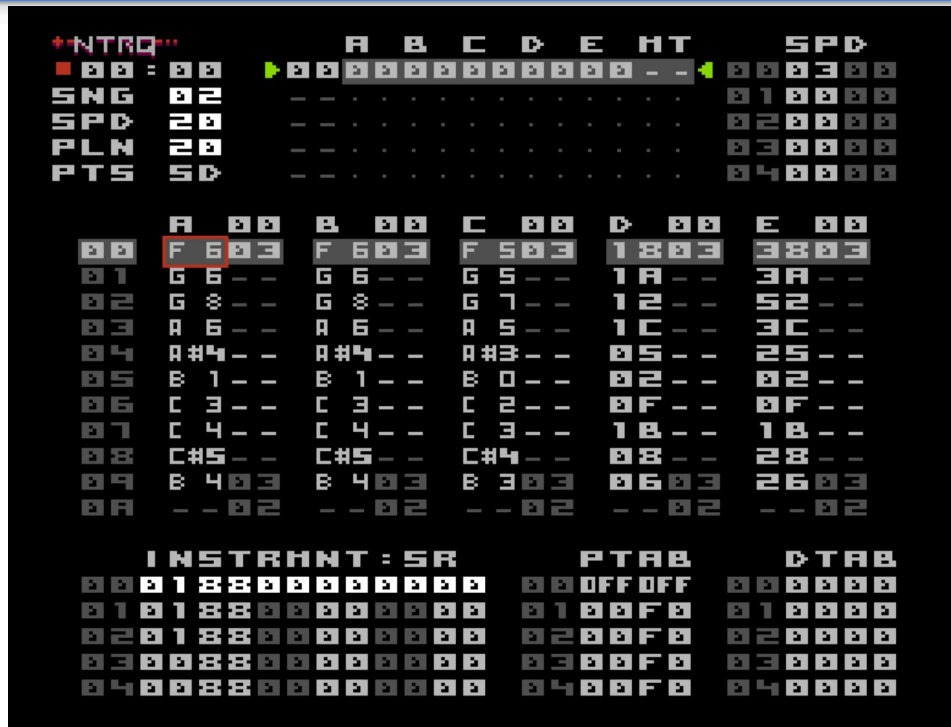
LSDJ / NanoLoop

There are ROM cartridges for gameboy (as well as other console) to use them as music instruments. Having a sample tracker interface and interacting with the gamepad controls.

NES: NTRQ

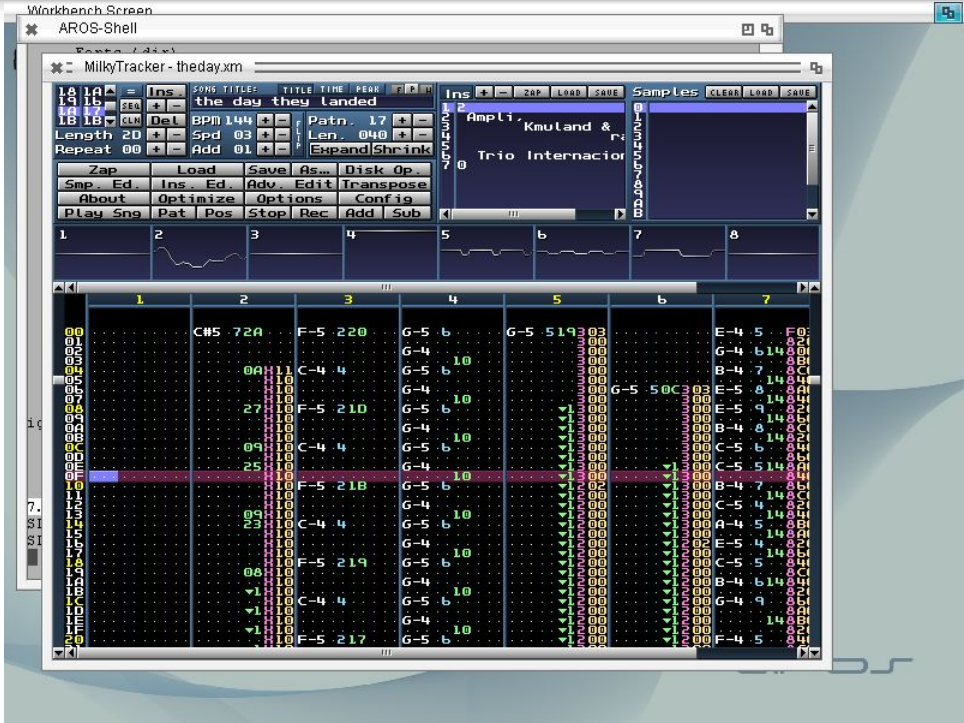
There are also chiptune trackers for other console platforms.

- This is for NES/Famicom



MilkyTracker

That's a nice multi-platform, open-source music tracker.




Encoding

Encoding it in data

- Configure the audio device
  - Signed 16bit integers
  - Mono / Stereo
  - 4000hz resolution XXX

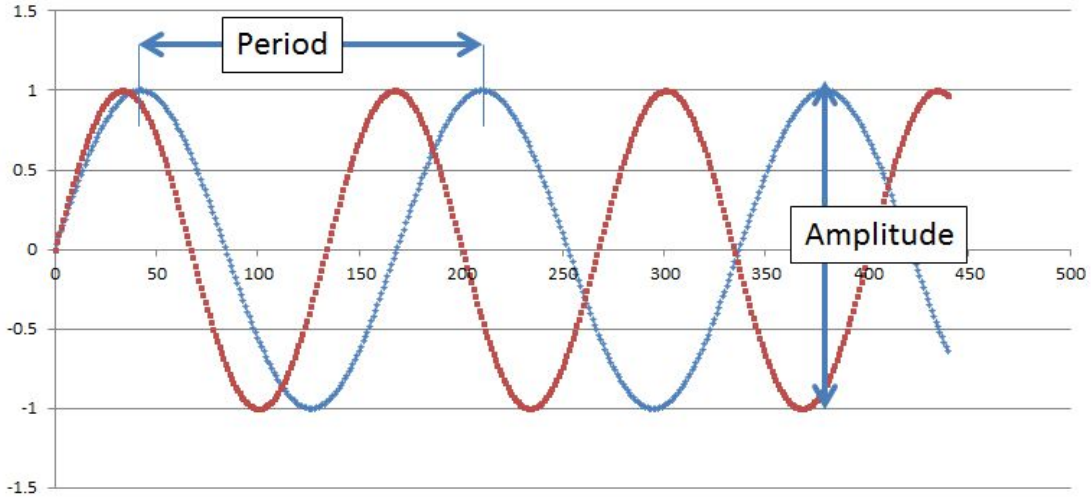
Audio wave is encoded as an array of shorts (16bit mono). In the case of being stereo, it is composed by a pair of 16bit signed values.

Easy!


Waves

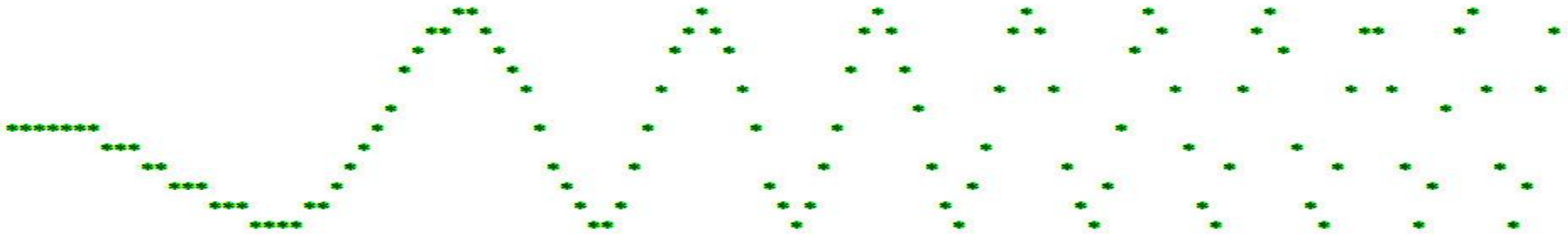
Amplitude

Aka volume. We can change the amplitude by dividing or multiplying each short value provided



Frequency

- Bass, Treble
- Defines the pitch
- Measured in Hz
- How many cycles the sinus wave will perform in



Arpeggio

The action of playing multiple relative tones starting at a base one in a short period of time.

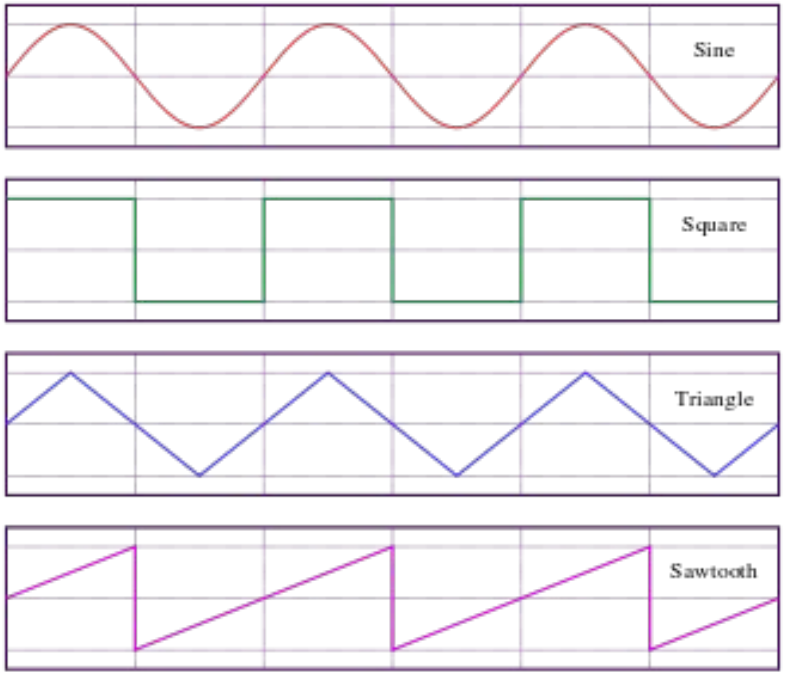
- See the auE command

Some musicians use arpeggio to simulate voices and chirps.

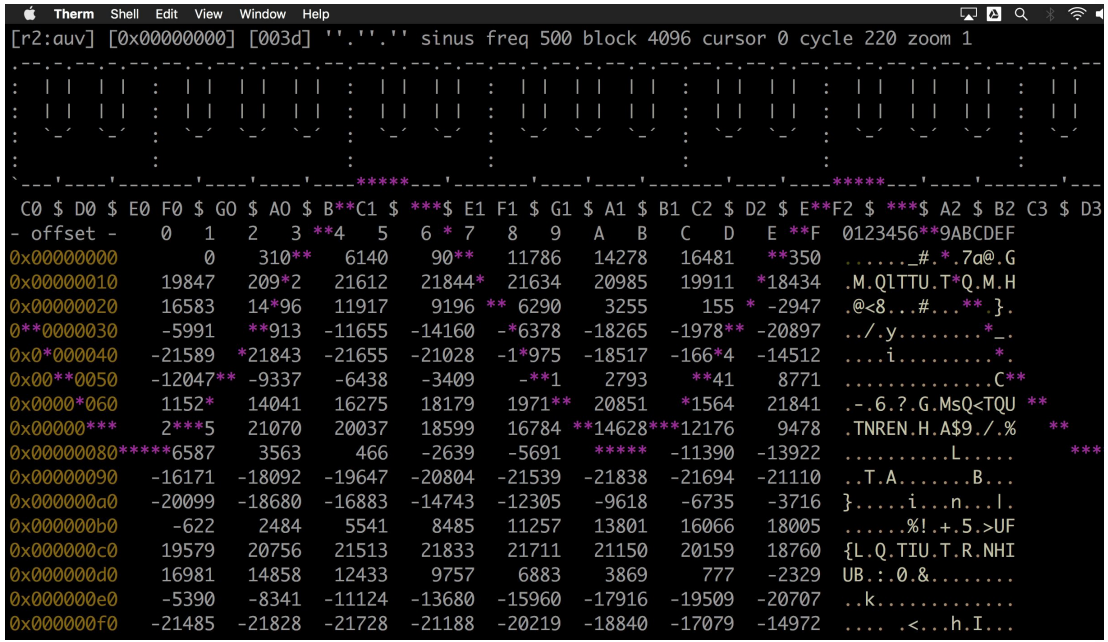
- Well known blirp sound from chiptune music.

Shape

- Triangle
- Square (Pulse)
- Sinus (Same as cosinus)
- Saw
- Noise
- Silence
- ...



Tones

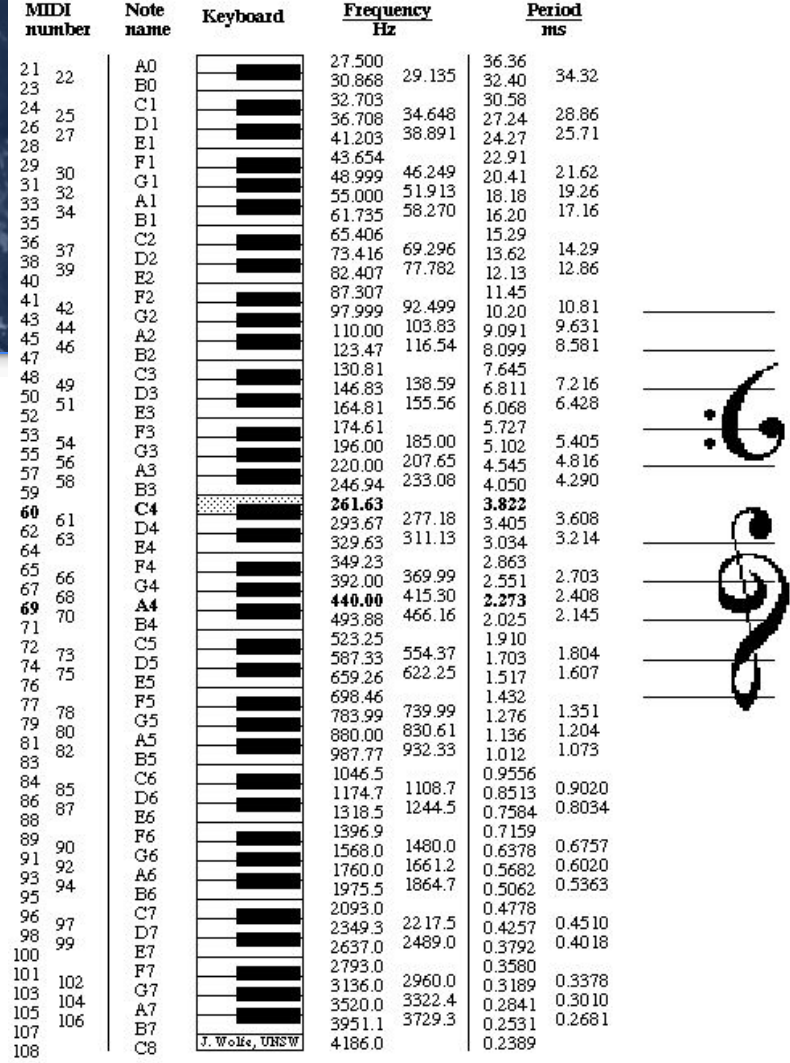


Musical Note	Freq (Hz)	Osc Fn (Decimal)	Osc Fn (Hex)	Musical Note	Freq (Hz)	Osc Fn (Decimal)	Osc Fn (Hex)
0 C0	16.35	274	0112	48 C4	261.63	4389	1125
1 C0$	17.32	291	0123	49 C4$	277.18	4650	122A
2 D0	18.35	308	0134	50 D4	293.66	4927	133F
3 D0$	19.44	326	0146	51 D4$	311.13	5220	1464
4 E0	20.60	346	015A	52 E4	329.63	5530	159A
5 F0	21.83	366	016E	53 F4	349.23	5859	16E3
6 F0$	23.12	388	0184	54 F4$	370.00	6207	183F
7 G0	24.50	411	018B	55 G4	392.00	6577	1981
8 G0$	25.96	435	01B3	56 G4$	415.30	6968	1B38
9 A0	27.50	461	01CD	57 A4	440.00	7382	1CD6
10 A0$	29.14	489	01E9	58 A4$	466.16	7821	1E80
11 B0	30.87	518	0206	59 B4	493.88	8286	205E
12 C1	32.70	549	0225	60 C5	523.25	8779	224B
13 C1$	34.65	581	0245	61 C5$	554.37	9301	2455
14 D1	36.71	616	0268	62 D5	587.33	9854	267E
15 D1$	38.89	652	028C	63 D5$	622.25	10440	28C8
16 E1	41.20	691	02B3	64 E5	659.25	11060	2B34
17 F1	43.65	732	02DC	65 F5	698.46	11718	2DC6
18 F1$	46.25	776	0308	66 F5$	740.00	12415	307F
19 G1	49.00	822	0336	67 G5	783.99	13153	3361
20 G1$	51.91	871	0367	68 G5$	830.61	13935	366F
21 A1	55.00	923	039B	69 A5	880.00	14764	39AC
22 A1$	58.27	978	03D2	70 A5$	932.33	15642	3D1A
23 B1	61.74	1036	040C	71 B5	987.77	16572	40BC
24 C2	65.41	1097	0449	72 C6	1046.50	17557	4495
25 C2$	69.30	1163	048B	73 C6$	1108.73	18601	48A9
26 D2	73.42	1232	04D0	74 D6	1174.66	19709	4CFC
27 D2$	77.78	1305	0519	75 D6$	1244.51	20897	518F
28 E2	82.41	1383	0567	76 E6	1318.51	22121	5669
29 F2	87.31	1465	05B9	77 F6	1396.91	23436	5B8C
30 F2$	92.50	1552	0610	78 F6$	1479.98	24830	60FE
31 G2	98.00	1644	066C	79 G6	1567.98	26306	6602
32 G2$	103.83	1742	06CE	80 G6$	1661.22	27871	6CDF
33 A2	110.00	1845	0735	81 A6	1760.00	29528	7358
34 A2$	116.54	1955	07A3	82 A6$	1864.65	31234	7A34
35 B2	123.47	2071	0817	83 B6	1975.53	33144	8178
36 C3	130.81	2195	0893	84 C7	2093.00	35115	892B
37 C3$	138.59	2325	0915	85 C7$	2217.46	37203	9153
38 D3	146.83	2463	099F	86 D7	2349.32	39415	99F7
39 D3$	155.56	2610	0A32	87 D7$	2489.01	41759	A31F
40 E3	164.81	2765	0ACD	88 E7	2637.02	44242	ACD2
41 F3	174.61	2930	0B72	89 F7	2793.83	46873	B719
42 F3$	185.00	3104	0C20	90 F7$	2959.95	49660	C1FC
43 G3	196.00	3288	0C08	91 G7	3135.96	52613	C085
44 G3$	207.65	3484	0D9C	92 G7$	3322.44	55741	D98D
45 A3	220.00	3691	0E6B	93 A7	3520.00	59056	E6B0
46 A3$	233.08	3910	0F46	94 A7$	3729.31	62567	F467
47 B3	246.94	4143	102F	95 B7	3951.06	*66288	*1F2F0

Notes

The notes defined in the Piano keys are always the same frequency.

Those can be exported flags.



Noise

- Brown
- White
- Pink

But there are more! Choose it with the `aun` command.

- Red, Green, Blue, Violet, Grey, Black, ...

https://en.wikipedia.org/wiki/Colors_of_noise

Binaurals!

- Mixing different frequencies and patterns of noises it is possible to stimulate parts of the brain to help on concentration, sleeping, meditation...

Now r2 can help you make your baby sleep!

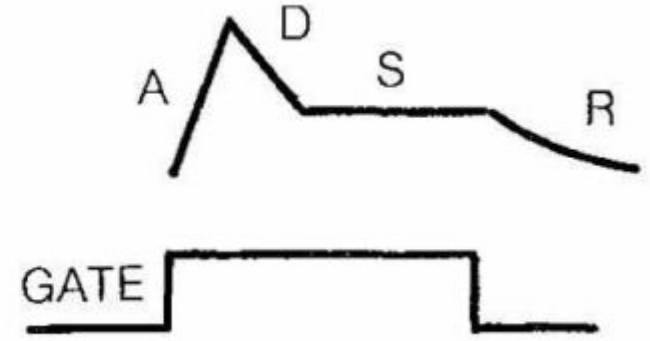
Effects

- Fade in/Fade out
- Pitch Glitching
- Frequency change over time
- Mirroring with delay (echo)
- Volume level

Attack, Decay, Sustain, Release

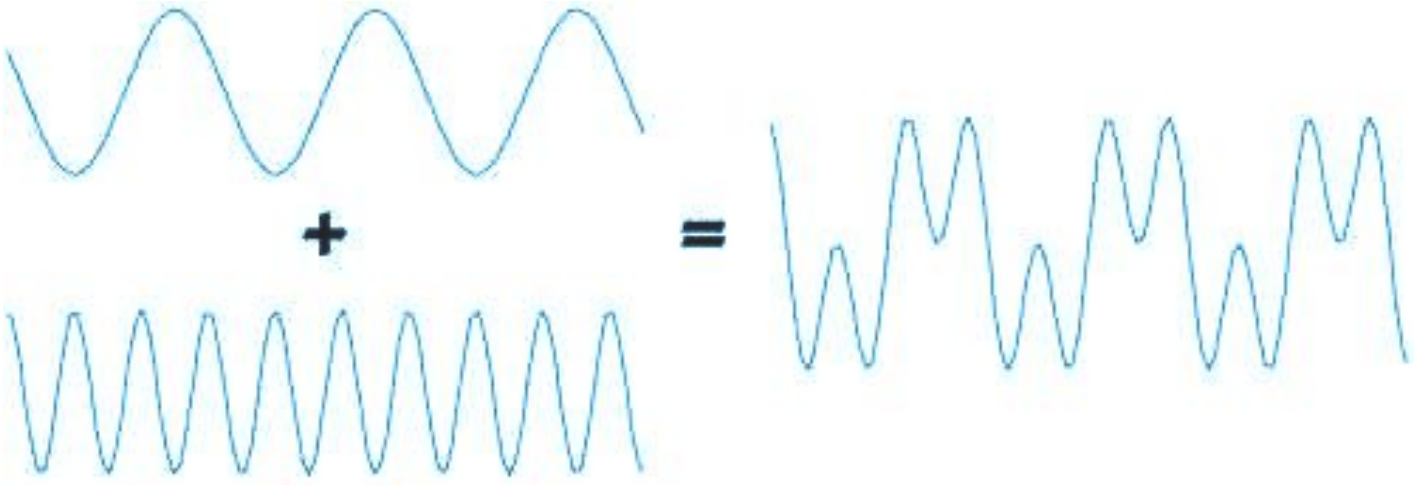
- Must be applied as post-processing filters

ATTACK:	10 ($A)	500 mS
DECAY:	8	300 mS
SUSTAIN:	10 ($A)	
RELEASE:	9	750 mS



Mixing waves

Use the aum command to sub the short values.




Designing our Audio CPU

Extending Existing Arch

We can easily extend r2 to handle syscall interruptions when emulating any architecture.

- Write assembly for mips, arm, x86, z80, ...
- Interact with the audio processor module using syscalls
- Hook e esil.hook.intr=
- We could emulate gameboy audio chip

But this is not that fun...

The Outside World

- Mapping hardware registers
- Making them available right from the register set
- Using IO ports?
- Using syscalls?
- Background mainloop thread?

Designing a new Audio CPU

- Register size
- Number of registers
- Instruction encodings
- Endianness
- How to make it sound?

ESIL

One of the best features of ESIL is the flexibility and extensibility of the language itself. Allows us to have our own custom instructions, hook memory operations, implement syscalls... and now, you can even run r2 commands!

- ahe !?E Hello world
- aes


The Plugin

r2au

- r2pm -i r2au

The dependencies to build this plugin are the following:

- C compiler
- pkg-config
- r2 installed
- Libao installed. (Linux, Mac), should be easy to build on Windows.

libao: How it works?

- ao_initialize()
  - Driver = ao_default_driver_id()
  - Dev = ao_open_live (Driver, rate, bts, channels, endian)
    - ao_play (buffer, length)
  - ao_close(Dev)
- ao_shutdown()

Commands in r2au

```
[0x00000000]> au
Usage: au[.abfeEimopvw] [args]
 au. - play current block (au.& in bg)
 aua - analyze wave in current block
 aub - audio blocksize
 auf - flags per freqs associated with keys
 aue - audio effects (arpeggio, percent, ..)
 auE [arpeggio] - arpeggio chords to use
 aui - init audio
 ...
```

Commands in r2au (2)

```
...
 aum - mix from given address into current with bsize
 aun [noisetype] - select [white, pink, brown]
 auN [amplitudes] - space separated volume changes 100 = 1
 auo - apply operation with immediate
 aup - print wave (aupi print piano)
 auv - visual wave mode
 auw - write wave (see auw?)
```

Commands in r2au (auw?)

Write waves following the specific shape, frequency, amplitude and arpeggio settings.

- See aun, auE and auN

```
[0x00000000]> auw?
Usage: auw[type] [args]
 fill current block with wave
 default wave shape type can be changed like this:
 > 'auws;auw 400' is the same as > 'auws 400'
 args: frequence
 types:
 (s)in    .''.''.
 (c)os    '..'..'
 (z)aw    /|/|/|/
 (Z)aw    \|\|\|\
 (p)ulse  |_|'|_|
 (v)pulse '__'__'
 (n)oise  /:./|.:
 (k)ross..  ><><><
 (t)ri..  /\/\/\/
 (-)silen _______
 (i)nc    _..--''
 (d)ec    ''--.._
[0x00000000]>
```

Background Loop

TheStranger implemented a pretty nice tasking model inside r2 that can be used to run code in background threads by sharing time and blocking when running a command. This allows us to run a play command repeated N times in background while we modify the buffer in real-time using the visual audio mode.


Visual

auv: visual audio mode

Scroll around the memory viewing the:

- Waveform
- HexDump
- Write keybindings
- Jam mode

Interactions

- Similar to r2 using hjkl to move around
- +/- to increase/decrease
- p/P to change the print mode
- Press ‘?’ for help

GamePad Support

If we want r2 embedded in consoles we need a gamepad designed interface.

- make -C gamepad
- make -C gamepad install
- r2 -c “god mode on” /bin/ls

DEMO

$ r2 -cauv malloc://4M

Show the following actions

- Make a sin wave
- Play it
- Create a wav file
- Enter audio visual mode

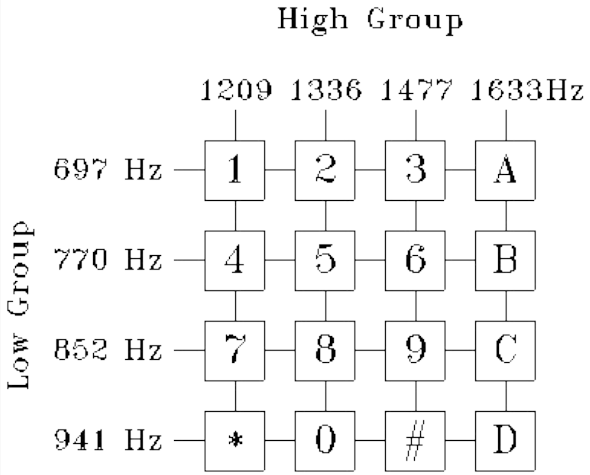
Bonus Demos

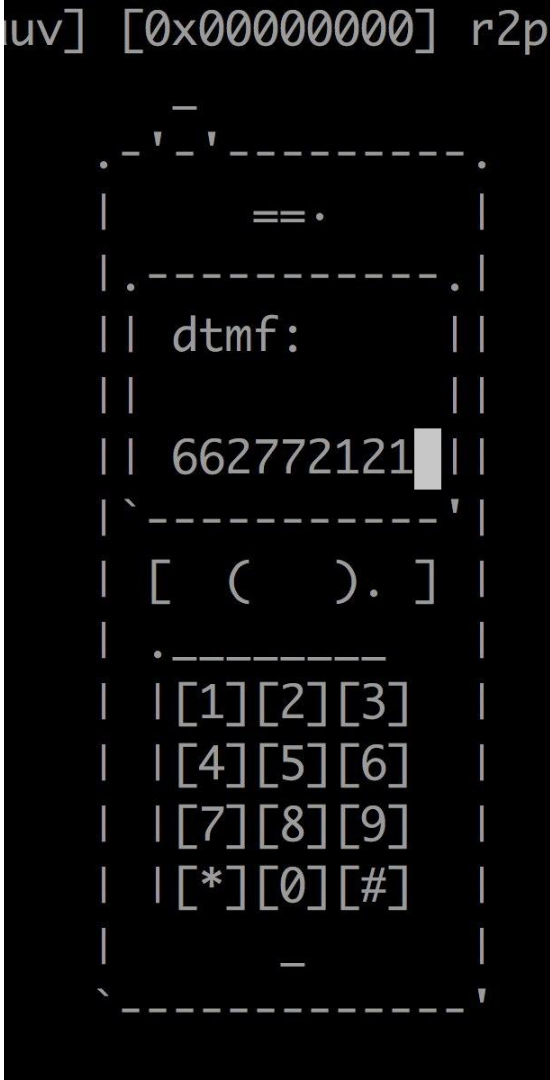
Telephone

Taking two sounds and mixing them together.

To make the each number and * # sound like a key.

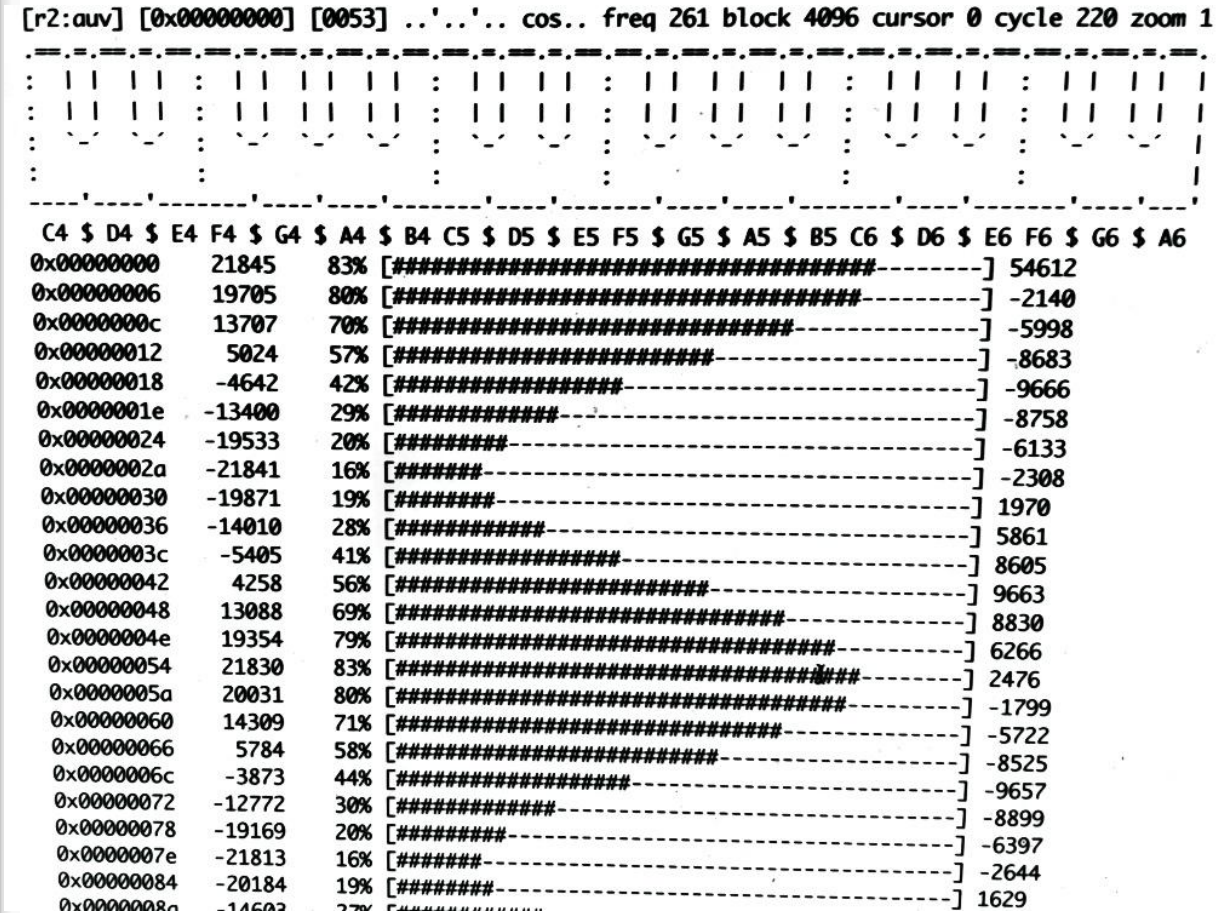
- auv!





JAM

Map your keyboard to different notes and play in real time



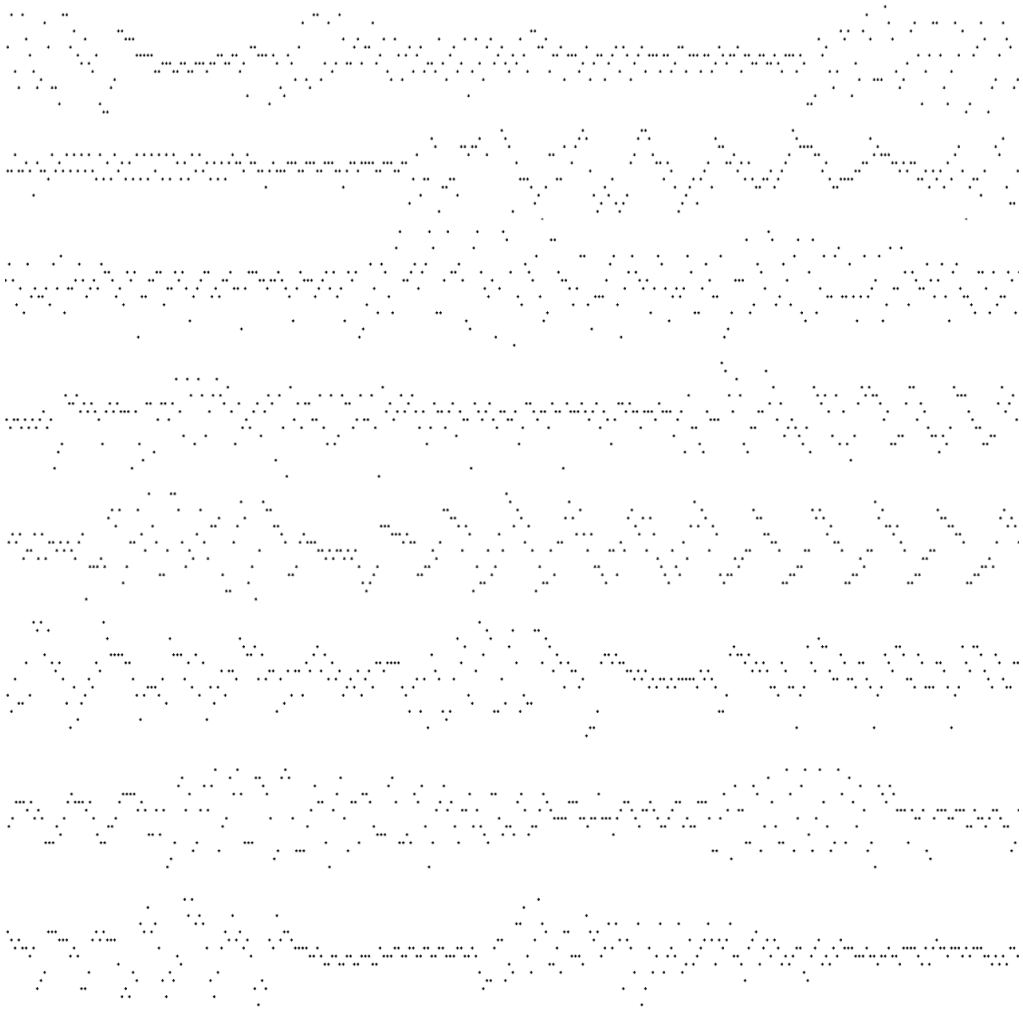
Samples

Importing a sample from an mp3/wav

```
[0x00000000 0% 230 malloc://1M]> pd $r
            0x00000000      0000fc17    nop
            0x00000004      0a2e        .short 11786
            0x00000006      6140        .short 16481
            0x00000008      874d        .short 19847
            0x0000000a      6c54        .short 21612
            0x0000000c      8254        .short 21634
            0x0000000e      c74d        .short 19911
            0x00000010      c740        .short 16583
            0x00000012      8d2e        .short 11917
            0x00000014      9218        .short 6290
            0x00000016      9b00        .short 155
            0x00000018      99e8        .short -5991
            0x0000001a      79d2        .short -11655
            0x0000001c      06c0bbb2    wait 48050
            0x00000020      abab        .short -21589
            0x00000022      69ab        .short -21655
            0x00000024      f9b1        .short -19975
            0x00000026      d4be        .short -16684
            0x00000028      f1d0        .short -12047
            0x0000002a      dae6        .short -6438
            0x0000002c      c9fe        .short -311
            0x0000002e      d116        .short 5841
            0x00000030      032d933f    mov r13, r32
            0x00000034      034d3c54    mov r13, r64
            0x00000038      ab54        .short 21675
            0x0000003a      454e        .short 20037
            0x0000003c      9041        .short 16784
            0x0000003e      902f        .short 12176
            0x00000040      bb19        .short 6587
            0x00000042      d201        .short 466
            0x00000044      c5e9        .short -5691
            0x00000046      82d3        .short -11390
            0x00000048      d5c0        .short -16171
            0x0000004a      41b3        .short -19647
            0x0000004c      ddab        .short -21539
            0x0000004e      42ab        .short -21694
            0x00000050      7db1        .short -20099
            0x00000052      0dbe        .short -16883
            0x00000054      efcf        .short -12305
            0x00000056      b1e5        .short -6735
            0x00000058      92fd        .short -622
            0x0000005a      a515        .short 5541
```

Play Song

Make a song like in the 90’s with Nokia



Sleep Fix

Try brown noise with your children!

$ r2 -c’aui;auN brown;’

-c ‘aub 1M;auwn 400;au.’

malloc://1M

AntiMosquito

We can finally be safe from mosquitos with radare2! (or not)


Questions?