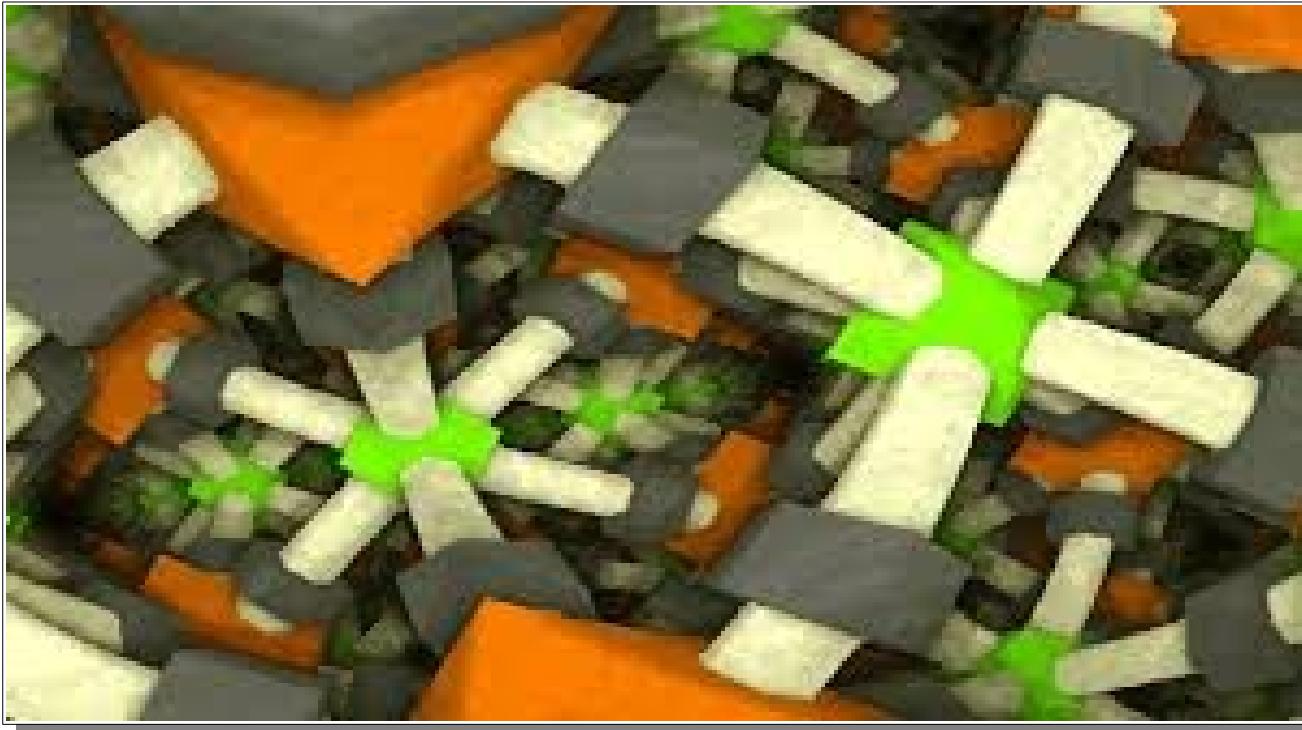


This page is intentionally left blank.

# 256-byte demoscene: extremly strong competition

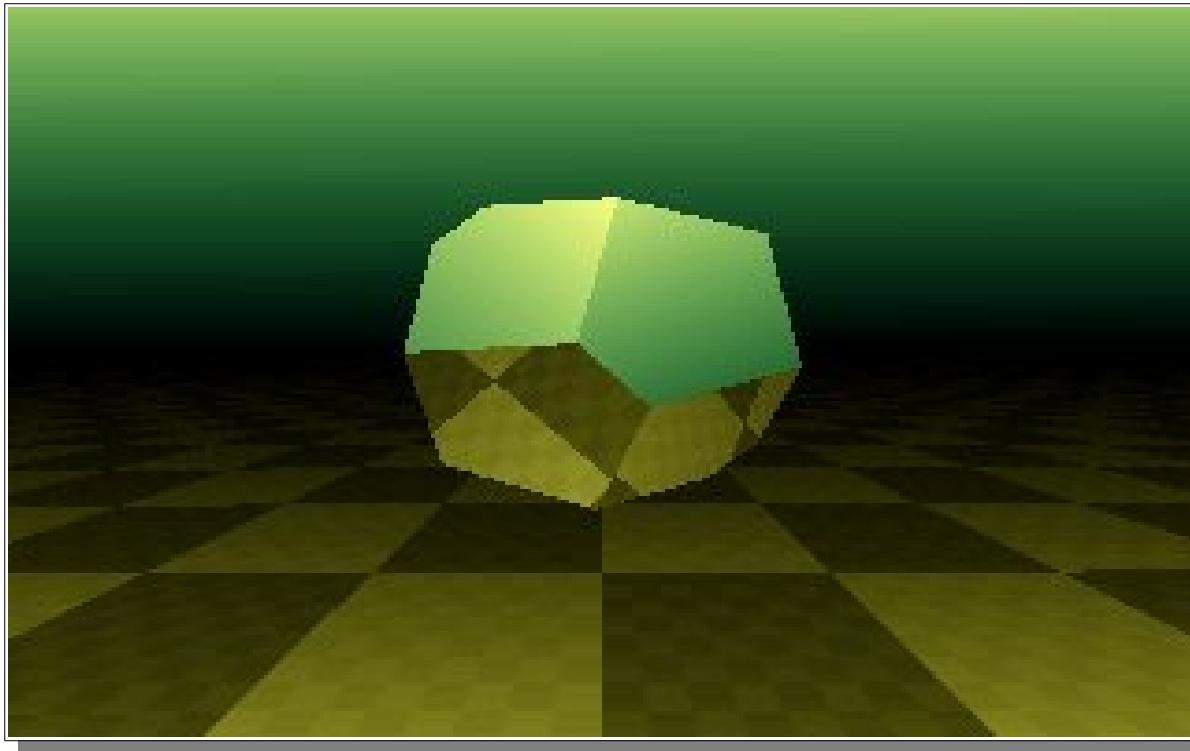


256-byte demoscene: extremly strong competition



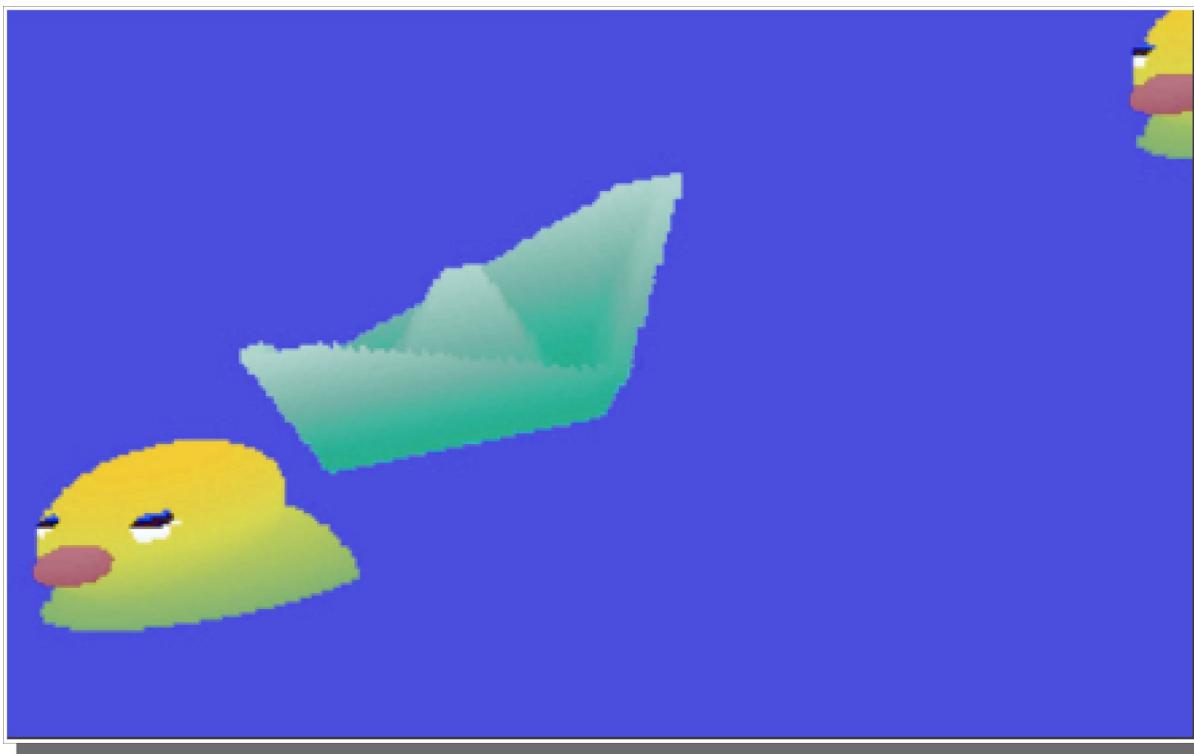
*Řrřola: Puls*

256-byte demoscene: extremly strong competition



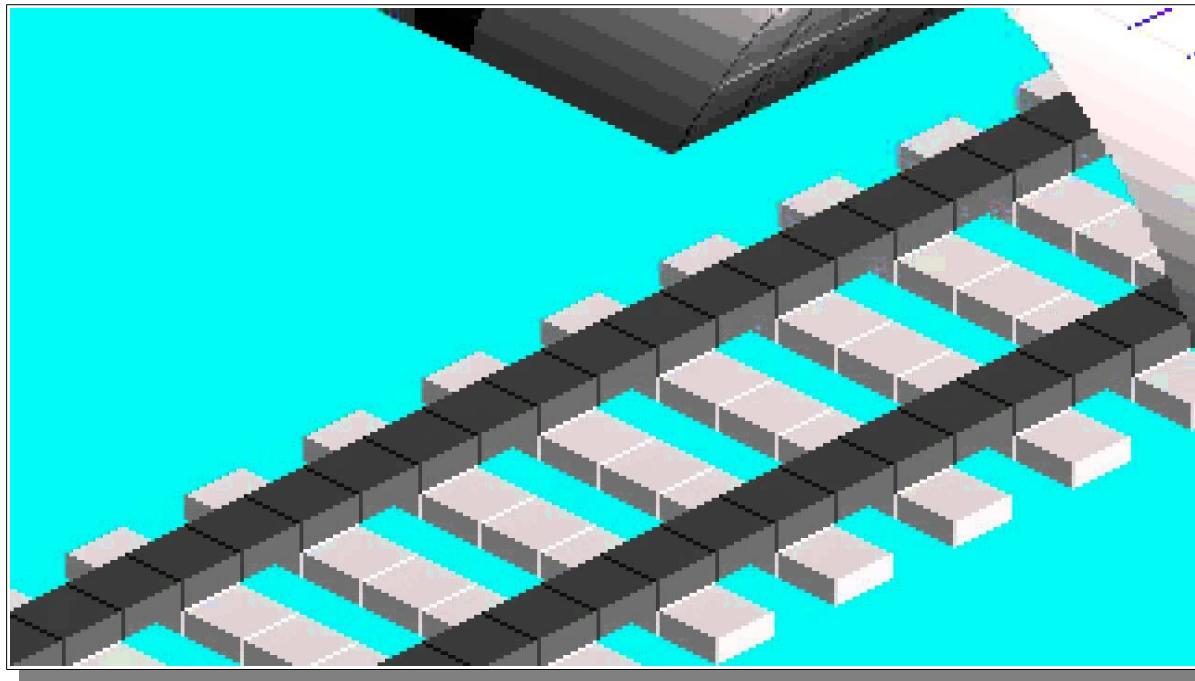
*Řrřola: Pyrit*

256-byte demoscene: extremly strong competition



*Digimind: Pool Patrol*

256-byte demoscene: extremly strong competition



*Digimind: Immediate Railways*

256-byte demoscene: extremly strong competition

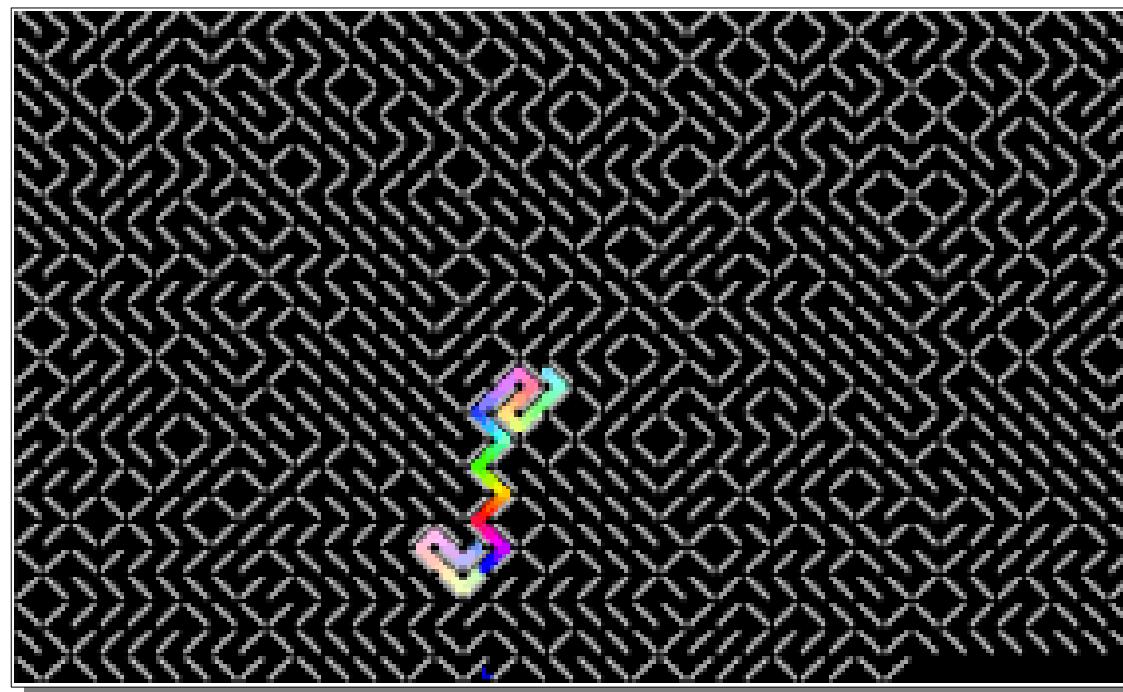


How to shine out of crowd?

# 256-byte demoscene: how to beat competition?

## Fun

(if you are not a hardcore sizecoder)



*ern0: Maze Solver*

# 256-byte demoscene: how to beat competition?

## Image processing



*TomCat: She – Weak Signal*

# 256-byte demoscene: how to beat competition?

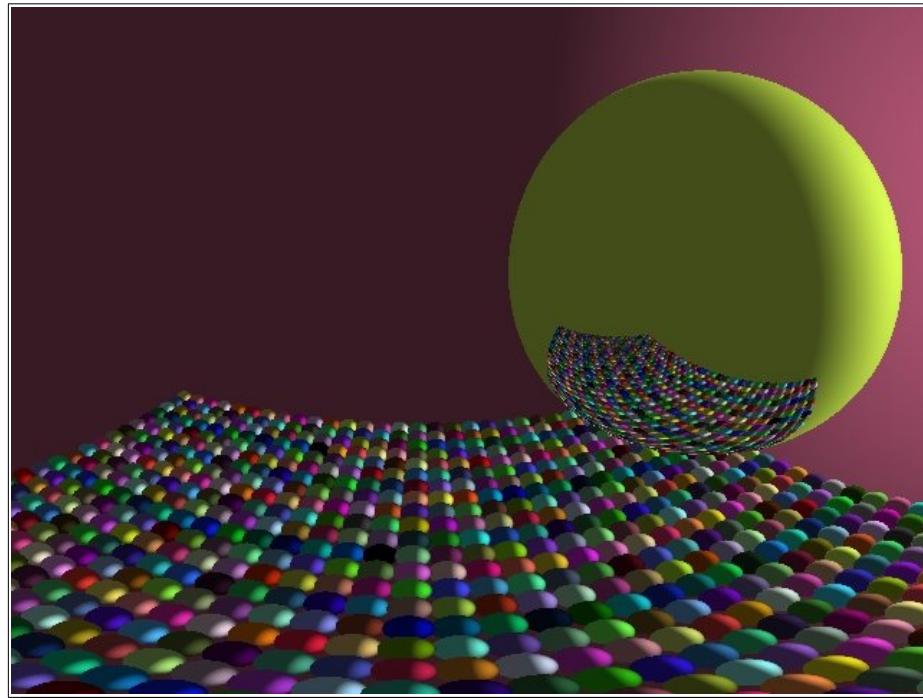
Image processing + fun



*TomCat: Be Happy!*

# 256-byte demoscene: how to beat competition?

## Raytracing



*TomCat: Colorful*

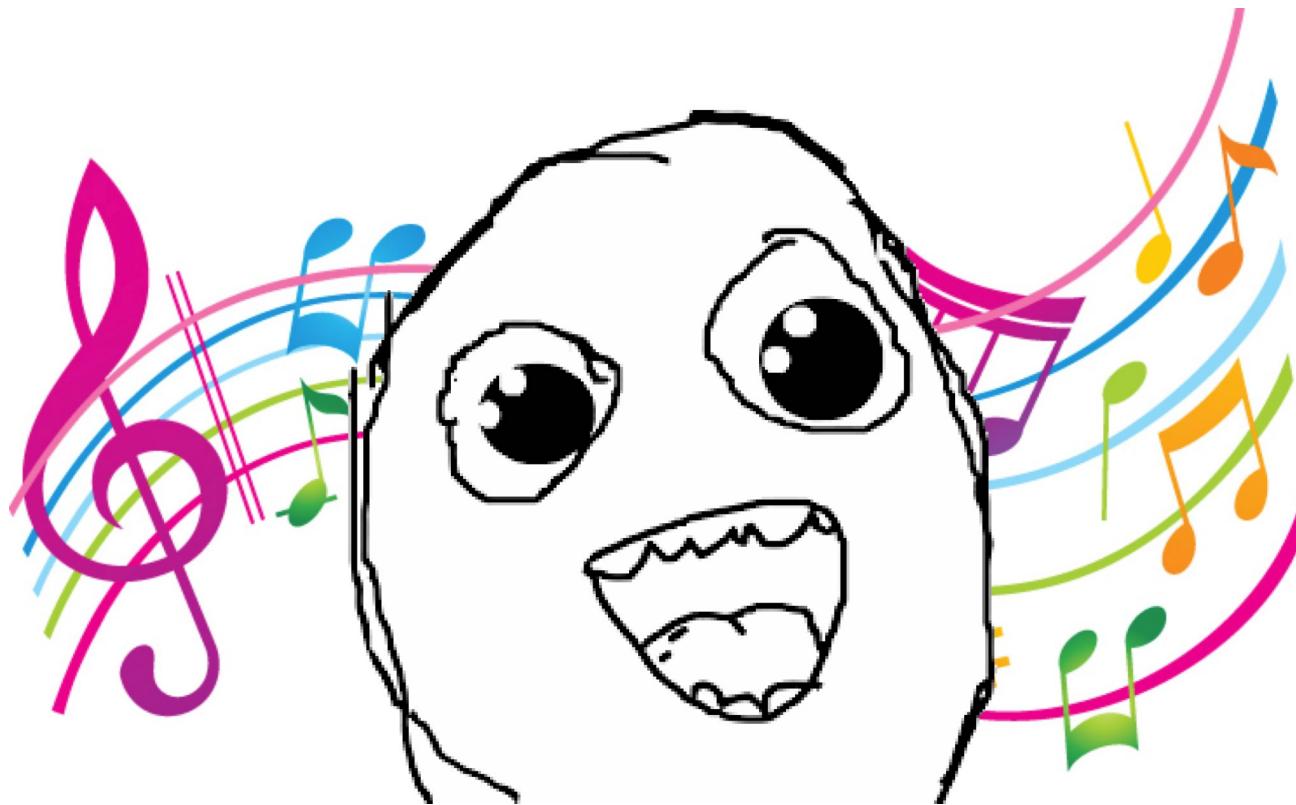
# 256-byte demoscene: how to beat competition?

Raytracing + fun



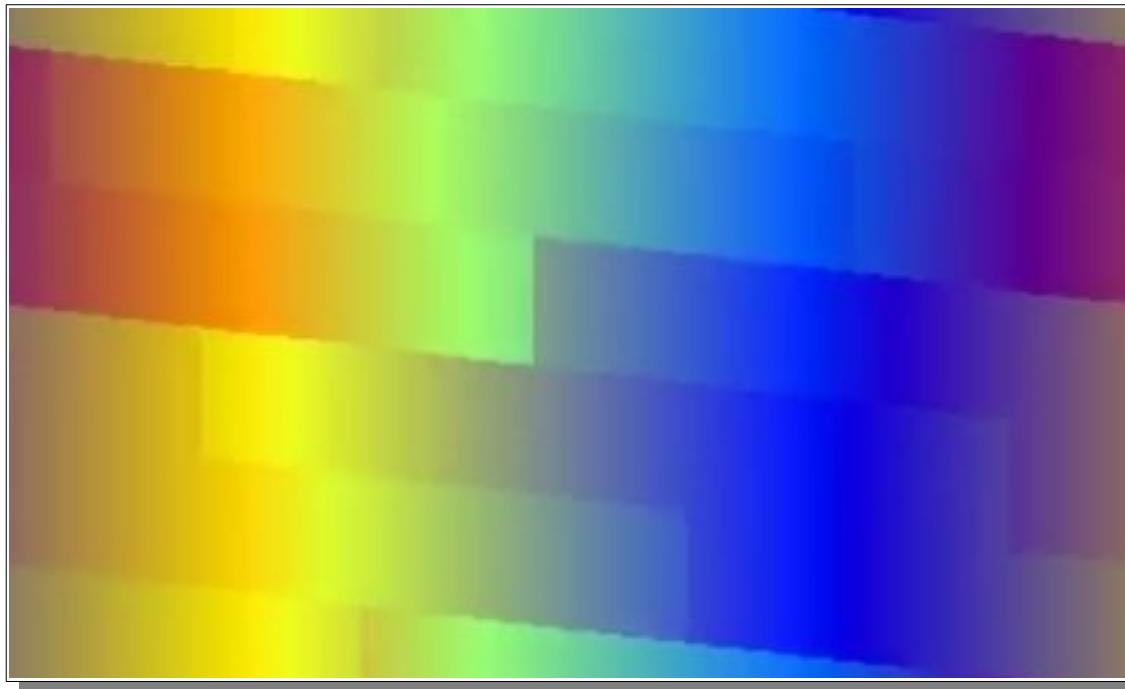
*TomCat: Pokeball*

# 256-byte demoscene: how to beat competition?



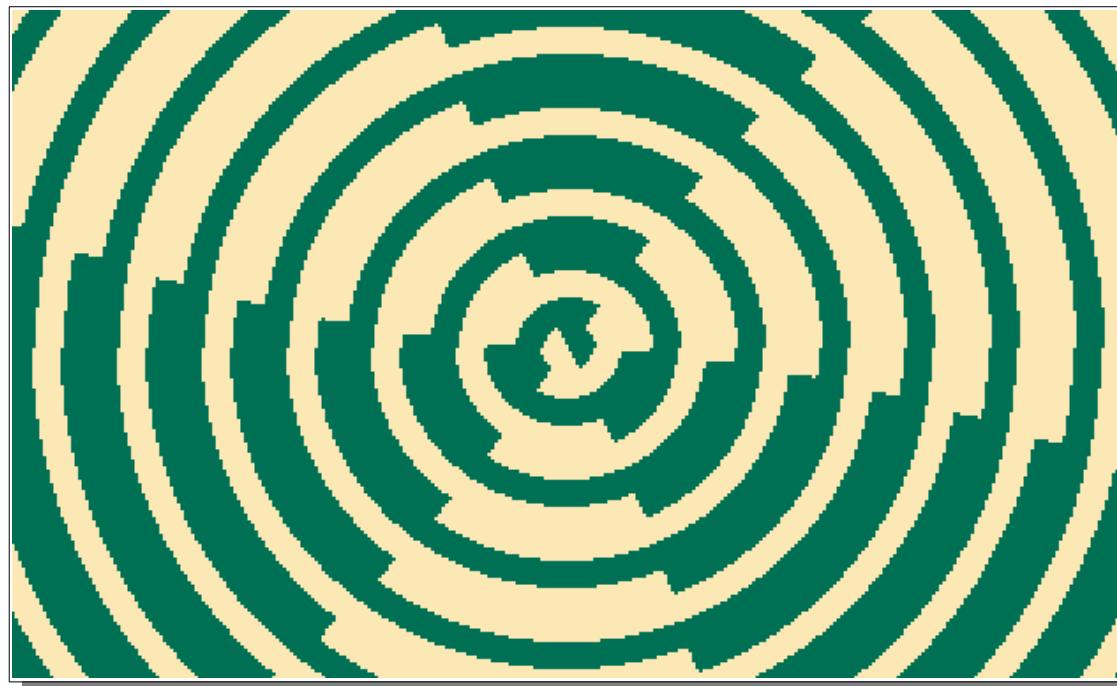
Music! Add music in 256-byte intros!

# 256 byte intro with music



*TomCat: 2(56)unlimited  
(bytebeat music by ern0)*

# 256 byte intro with music



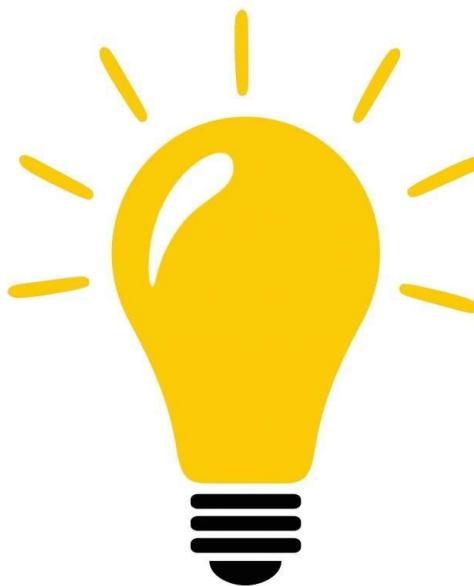
*TomCat: No Sleep!*  
*(buzzer music by ern0)*

256 byte intro with music

Everyone loves it!



# Create universal bytebeat tool



# Create universal bytebeat tool

- Bytebeat player & editor

*TomCat*



# Create universal bytebeat tool

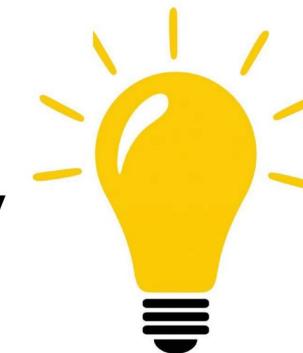
- Bytebeat player & editor

*TomCat*



- Formula compiler for assembly

*ern0*



# Create universal bytebeat tool

- Bytebeat player & editor  
*TomCat*
- Formula compiler for assembly  
*ern0*



# Create universal bytebeat tool

- Bytebeat player & editor

*TomCat*



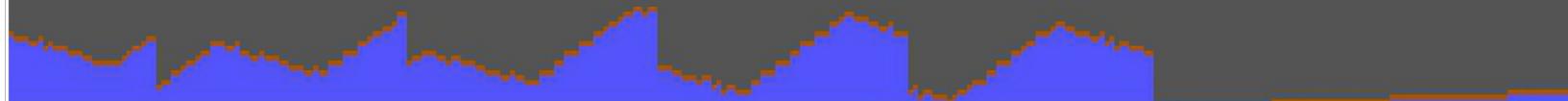
- Formula compiler for assembly

*ern0*



# Bytebeat Editor (TomCat)

```
BYTEBEAT by TomCat/Abaddon 7 24632
freq:18939 zoom:1 out:7 vol:11173
Kick drum no:CMP1Z skip:TSTONZ
rate:16383 len:24576 vol:64
Hihat no:CMP3B skip:TST96NZ
rate:63 ien:22 rnd:99 vol:64 fade:1
Instrument1 no:CMP1B wave:sawtooth
idx:0 mask:15 len:8 tune:4 fade:1
Instrument2 no:CMP5B wave:triangle
idx:16 mask:31 len:4 tune:16 fade:1
Arpeggio no:CMP4B
idx:48 mask:7 rate:4 len:4 vol:31
```



```
9405450574B445B425B4744525059474
4 405A405743474 4 405A405743474
0525352535052505 4 405A405743474
042474 4043474 4 3 3 3 3 3 3 3 3 3
```

# Bytebeat Editor (TomCat)

## Features:

- realtime feedback

The screenshot shows a window titled "BYTEBEAT by TomCat/Abaddon 2 12981". Inside, there's a text-based configuration section and a visual representation of the bytebeat pattern.

**Configuration (Text):**

```
BYTEBEAT by TomCat/Abaddon 2 12981
freq:18939 zoom:1 out:7 vol:35000
Kick drum no:CMP1Z skip:TSTONZ
rate:16383 len:24576 vol:64
Hihat no:CMP3B skip:TST96NZ
rate:63 len:22 rnd:99 vol:64 fade:1
Instrument1 no:CMP1B wave:sawtooth
idx:0 mask:15 len:8 tune:4 fade:1
Instrument2 no:CMP5B wave:triangle
idx:16 mask:31 len:4 tune:16 fade:1
Arpeggio no:CMP4B
idx:48 mask:7 rate:4 len:4 vol:31
```

**Visual Representation:** Below the config, there's a dark gray area with blue triangular shapes representing the waveform of the bytebeat pattern.

**Hex Dump (Text):**

```
9405450574B445B425B4744525059474
4 405A405743474 4 405A405743474
0525352535052505 4 405A405743474
042474 4043474 4 3 3 3 3 3 3 3 3
```

# Bytebeat Editor (TomCat)

## Features:

- realtime feedback
- graphical sound wave

The screenshot shows a window titled "BYTEBEAT by TomCat/Abaddon 5 44539". The top half contains configuration parameters for various instruments and effects. The bottom half features a graphical representation of a sound wave, consisting of a series of blue triangles on a black background, with a grid of numbers below it.

Configuration parameters (top half):

```
BYTEBEAT by TomCat/Abaddon 5 44539
freq:18939 zoom:1 out:7 vol:35000
Kick drum no:CMP1Z skip:TSTONZ
rate:16383 len:24576 vol:64
Hihat no:CMP3B skip:TST96NZ
rate:63 len:22 rnd:99 vol:64 fade:1
Instrument1 no:CMP1B wave:sawtooth
idx:0 mask:15 len:8 tune:4 fade:1
Instrument2 no:CMP5B wave:triangle
idx:16 mask:31 len:4 tune:16 fade:1
Arpeggio no:CMP4B
idx:48 mask:7 rate:4 len:4 vol:31
```

Graphical sound wave (middle half):

Below the graphical wave is a grid of numbers:

```
9405450574B445B425B4744525059474
4 405A405743474 4 405A405743474
0525352535052505 4 405A405743474
042474 4043474 4 3 3 3 3 3 3 3 3
```

## Bytebeat Editor (TomCat)

### Features:

- realtime feedback
- graphical sound wave
- save/restore modified code

BYTEBEAT by TomCat/Abaddon 2 12981  
freq:18939 zoom:1 out:7 vol:35000  
Kick drum no:CMP1Z skip:TSTONZ  
rate:16383 len:24576 vol:64  
Hihat no:CMP3B skip:TST96NZ  
rate:63 len:22 rnd:99 vol:64 fade:1  
Instrument1 no:CMP1B wave:sawtooth  
idx:0 mask:15 len:8 tune:4 fade:1  
Instrument2 no:CMP5B wave:triangle  
idx:16 mask:31 len:4 tune:16 fade:1  
Arpeggio no:CMP4B  
idx:48 mask:7 rate:4 len:4 vol:31

9405450574B445B425B4744525059474  
4 405A405743474 4 405A405743474  
0525352535052505 4 405A405743474  
042474 4043474 4 3 3 3 3 3 3 3 3

## Bytebeat Editor (TomCat)

### Features:

- realtime feedback
- graphical sound wave
- save/restore modified code

### Issues:

- more than 70 hotkeys

The screenshot shows the Bytebeat Editor interface. At the top, there is a text area containing assembly-like code for a 'BYTEBEAT' track. Below the code is a waveform visualization showing blue peaks on a black background. At the bottom, there is a hex dump of the raw byte data.

```
BYTEBEAT by TomCat/Abaddon 5 44539
freq:18939 zoom:1 out:7 vol:35000
Kick drum no:CMP1Z skip:TSTONZ
rate:16383 len:24576 vol:64
Hihat no:CMP3B skip:TST96NZ
rate:63 len:22 rnd:99 vol:64 fade:1
Instrument1 no:CMP1B wave:sawtooth
idx:0 mask:15 len:8 tune:4 fade:1
Instrument2 no:CMP5B wave:triangle
idx:16 mask:31 len:4 tune:16 fade:1
Arpeggio no:CMP4B
idx:48 mask:7 rate:4 len:4 vol:31
```

```
9405450574B445B425B4744525059474
4 405A405743474 4 405A405743474
0525352535052505 4 405A405743474
042474 4043474 4 3 3 3 3 3 3 3 3
```

# Bytebeat Editor (TomCat)

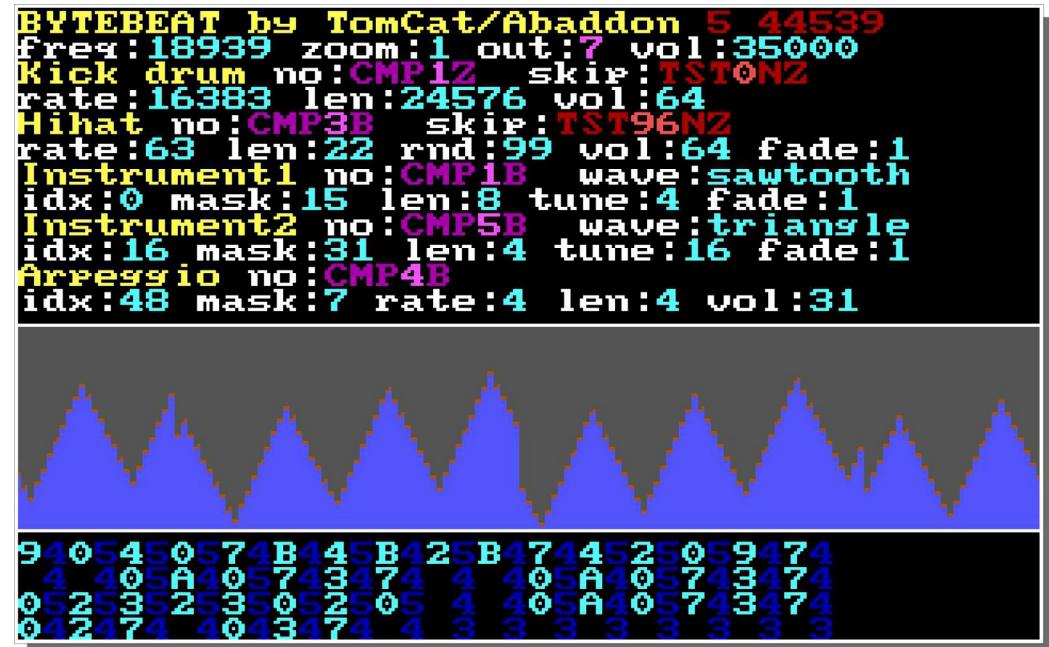
## Features:

- realtime feedback
- graphical sound wave
- save/restore modified code



## Issues:

- more than 70 hotkeys



# Bytebeat Editor (TomCat)

## Features:

- realtime feedback
- graphical sound wave
- save/restore modified code

## Issues:

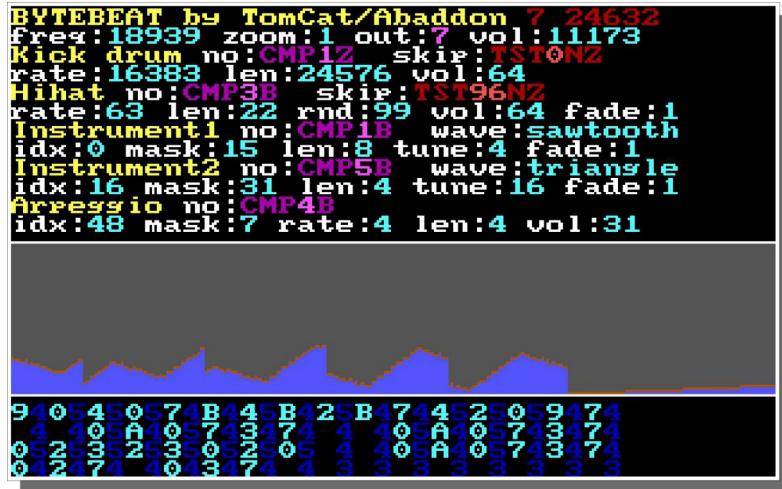
- more than 70 hotkeys
- needs some x86 coder knowledge  
e.g. you can set any flag for a conditional jump

The screenshot shows the Bytebeat Editor interface. At the top, there is assembly-like code for a song titled 'BYTEBEAT by TomCat/Abaddon 2 12981'. The code includes parameters for various instruments like Kick, Drum, Hihat, and Arpeggio. Below the code is a waveform visualization consisting of several blue triangles on a grey background. At the bottom, there is a memory dump showing a sequence of bytes.

```
BYTEBEAT by TomCat/Abaddon 2 12981
freq:18939 zoom:1 out:7 vol:35000
Kick drum no:CMP1Z skip:TSTONZ
rate:16383 len:24576 vol:64
Hihat no:CMP3B skip:TST96NZ
rate:63 len:22 rnd:99 vol:64 fade:1
Instrument1 no:CMP1B wave:sawtooth
idx:0 mask:15 len:8 tune:4 fade:1
Instrument2 no:CMP5B wave:triangle
idx:16 mask:31 len:4 tune:16 fade:1
Arpeggio no:CMP4B
idx:48 mask:7 rate:4 len:4 vol:31
```

```
9405450574B445B425B4744525059474
4405A4057434744405A405743474
05253525350525054405A405743474
042474404347443333333333333333
```

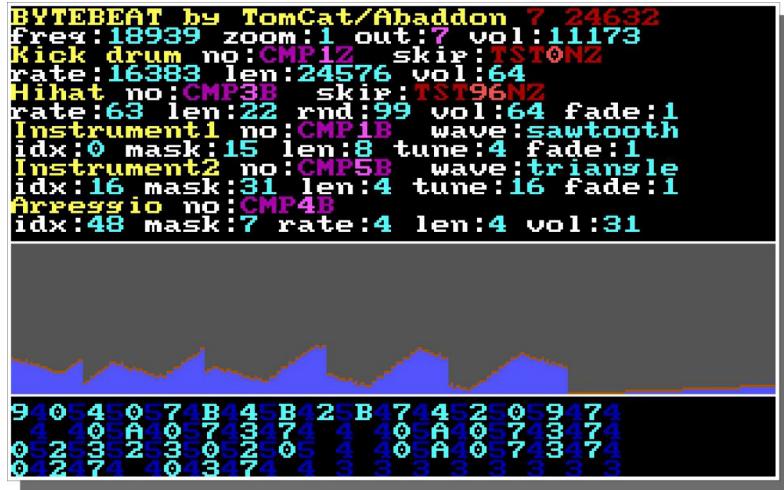
# *Bytebeat Editor (TomCat)*



Verdict:

- too complex, especially for musicians #UX

# Bytebeat Editor (TomCat)



## Verdict:

- too complex, especially for musicians #UX
- does not provide enough freedom

# *Bytebeat Editor (TomCat)*

**BYTEBEAT** by TomCat/Abaddon 7 24632  
freq:18939 zoom:1 out:7 vol:11173  
Kick drum no:**CMP12Z** skip:**TSTONZ**  
rate:16383 len:24576 vol:64  
Hihat no:**CMP3B** skip:**TST96NZ**  
rate:63 len:22 rnd:99 vol:64 fade:1  
**Instrument1** no:**CMP1B** wave:sawtooth  
idx:0 mask:15 len:8 tune:4 fade:1  
**Instrument2** no:**CMP5B** wave:triangle  
idx:16 mask:31 len:4 tune:16 fade:1  
**Arpeggio** no:**CMP4B**  
idx:48 mask:7 rate:4 len:4 vol:31

# Verdict

- Too complex, especially for musicians. UX does not provide enough freedom.

# *Assemblyzator (ern0)*

# *Assemblyzator (ern0)*

Transform bytebeat formula to assembly code...

## *Assemblyzator (ern0)*

Transform bytebeat formula to assembly code  
using a modern C compiler!

# Assemblyzator (ern0)

Transform bytebeat formula to assembly code using a modern C compiler!

```
int main() {
    int result = 0;

    for (int i = 0; i < 100; i++) {
        for (int j = 0; j < 100; j++) {
            result += i * j;
        }
    }

    return result;
}
```

```
b8 e4 e0 75 01
5c3
```

```
main:
    mov     eax,0x175e0e4
    ret
```

# Assemblyzator (ern0)

Transform bytebeat formula to assembly code using a modern C compiler!

```
int main() {  
    int result = 0;  
  
    for (int i = 0; i < 100; i++) {  
        for (int j = 0; j < 100; j++) {  
            result += i * j;  
        }  
    }  
  
    return result;  
}
```

```
b8 e4 e0 75 01  
5c3
```

```
main:  
    mov     eax, 0x175e0e4  
    ret
```

Very optimized!  
Such compiler!



# Assemblyzator (ern0)

# Transform bytebeat formula to assembly code using a modern C compiler!

~~VS, optimized!  
such compiler!~~



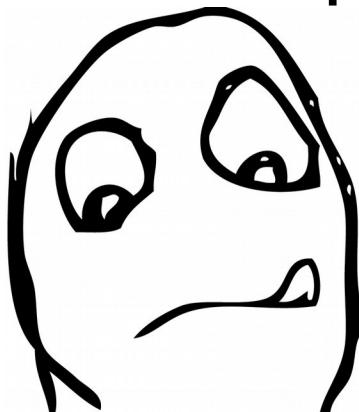
No modern compiler exists for **16-bit** target.

# *Assemblyzator (ern0)*

Let's write a compiler thing!

# *Assemblylyzator (ern0)*

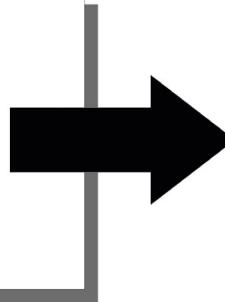
Let's write a compiler thing!



Split complex bytebeat formula  
to series of simple formulas,  
which is close to assembly

## Assemblyzator (ern0)

```
((t<<1) ^ ((t<<1) +  
(t>>7) &t>>12)) |  
t>>(4- (1^7&(t>>19)))  
| t>>7
```



```
var3 = t << 1  
var7 = t >> 7  
var5 = var3 + var7  
var6 = t >> 12  
var4 = var5 & var6  
var1 = var3 ^ var4  
var12 = t >> 19  
var11 = 7 & var12  
var10 = 1 ^ var11  
var9 = -var10  
var9 = var9 + 4  
var8 = t >> var9  
var2 = var8 | var7  
result = var1 | var2
```

# *Assemblyzator (ern0)*

*Features:*

- split formula

# *Assemblyzator (ern0)*

## *Features:*

- split formula
- handle num arrays

# *Assemblyzator (ern0)*

## *Features:*

- split formula
- handle num arrays
- handle string arrays

# *Assemblyzator (ern0)*

## *Features:*

- split formula
- handle num arrays
- handle string arrays
- remove duplications

# *Assemblyzator (ern0)*

## *Features:*

- split formula
- handle num arrays
- handle string arrays
- remove duplications

## *Design Flaws:*

- **3-op ( $A = B \text{ op } C$ )**  
8086 assembly instructions are 2-operand

# *Assemblyzator (ern0)*

## *Features:*

- split formula
- handle num arrays
- handle string arrays
- remove duplications

## *Design Flaws:*

- 3-op ( $A = B \text{ op } C$ )  
8086 assembly instructions are 2-operand
- can't handle cond. op.  
 $A = ( B \text{ op } C ? D : E )$   
improperly designed Abstract Syntax Tree

# *Assemblyzator (ern0)*

## *Features:*

- split formula
- handle num arrays
- handle string arrays
- remove duplications

## *Design Flaws:*

- 3-op ( $A = B \text{ op } C$ )  
8086 assembly instructions are 2-operand
- can't handle cond. op.  
 $A = ( B \text{ op } C ? D : E )$   
improperly designed Abstract Syntax Tree

## *Verdict:*

- nice try, but does not help much

# *Assemblyzator (ern0)*

## *Features:*

- split formula
- handle num arrays
- handle string arrays
- remove duplications

## *Design Flaws:*

- 3-op ( $A = B \text{ op } C$ )  
8086 assembly instructions are 2-operand
- can't handle cond. op.  
 $A = ( B \text{ op } C ? D : E )$   
improperly designed Abstract Syntax Tree

## *Verdict:*

- nice try, but does not help much
- writing a compiler is not as easy as it looks first

# Assemblyzator (ern0)

## Features:

- split formula
- handle num arrays
- handle string arrays
- remove duplications

## Verdict:

- nice try, but it does not help much
- writing a compiler is not as easy as it looks first

## Design Flaws:

- 3-op ( $A = B \text{ op } C$ )  
8086 assembly instructions are 2-operand  
can't handle cond. op.  
(A op, B ? C : D )  
improperly designed Abstract Syntax Tree



[TomCat] *Instead of creating universal tools,  
we should choose one song and  
optimize for it*

[TomCat] *Instead of creating universal tools,  
we should choose one song and  
optimize for it*

[ern0] *Right, I'll pick a song*

[TomCat] *Instead of creating universal tools,  
we should choose one song and  
optimize for it*

[ern0] *Right, I'll pick a song*

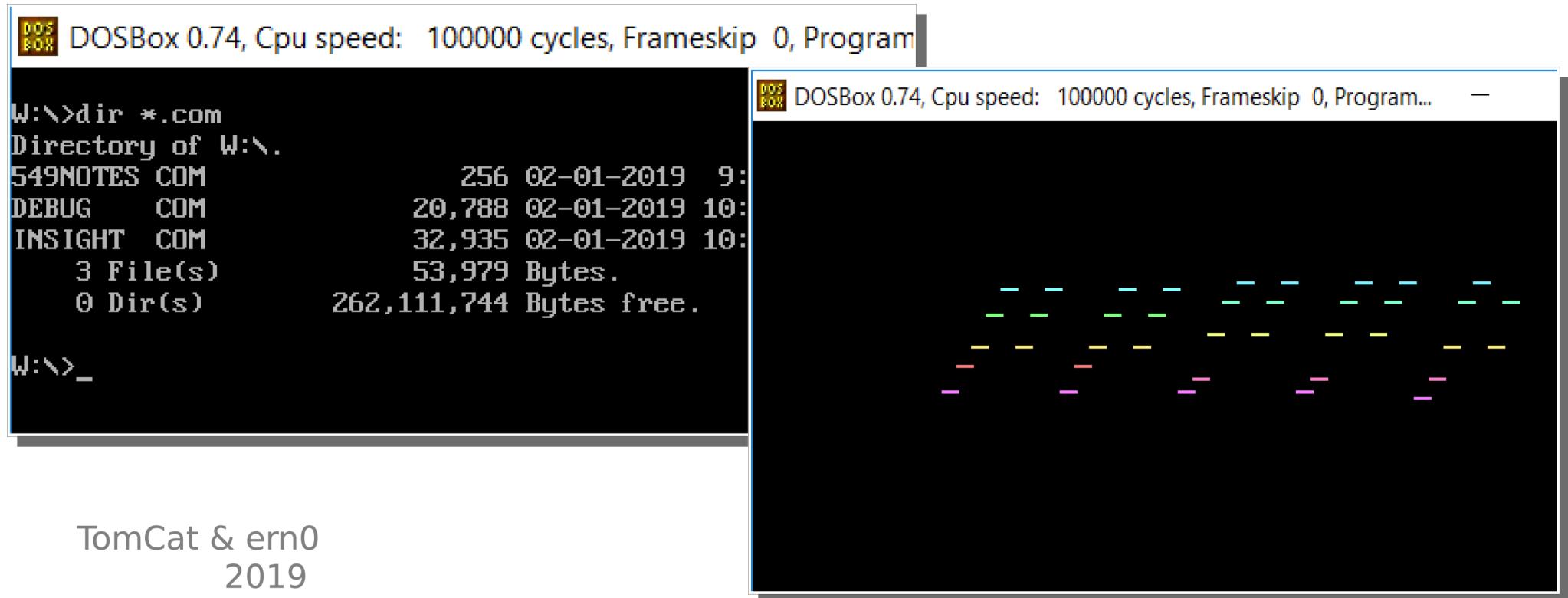


*Some hours later...*

[ern0] *I got the perfect one.*

# Making of **549NOTES.COM**

## the 256-byte intro for PC-DOS which plays 549 notes



## *Table Of Contents*

I. Song ★★★★☆

II. Data ★★★★★☆

III. Code ★★★★★★

# I. Song

# Prelude I

In C major

BWV 846

Johann Sebastian Bach (1685 - 1750)

Piano

This system shows the beginning of the prelude. The treble staff consists of eighth-note pairs followed by a rest. The bass staff consists of quarter notes with fermatas. The key signature is C major.

This system continues the pattern established in the first system. The treble staff has eighth-note pairs followed by rests. The bass staff has quarter notes with fermatas. A sharp sign appears in the key signature, indicating a temporary change to G major.

# *J. S. Bach: Prelude I. in C Major (BWV 846)*

1. Popular, well-known piece

## *J. S. Bach: Prelude I. in C Major (BWV 846)*

1. Popular, well-known piece
2. Written for piano: optimal for MIDI...

## *J. S. Bach: Prelude I. in C Major (BWV 846)*

- Piano (patch 1) is the default instrument on all channels for all General MIDI instruments

## *J. S. Bach: Prelude I. in C Major (BWV 846)*

- Piano (patch 1) is the default instrument on all channels for all General MIDI instruments

*Switch sound card  
to MIDI mode:*

```
mov    al,3fH
mov    dx,331H
out    dx,al
```

## *J. S. Bach: Prelude I. in C Major (BWV 846)*

- Piano (patch 1) is the default instrument on all channels for all General MIDI instruments
- Chord breaks: no „key up” message needed

*Switch sound card  
to MIDI mode:*

```
mov    al,3fH
mov    dx,331H
out    dx,al
```

# J. S. Bach: Prelude I. in C Major (BWV 846)

- Piano (patch 1) is the default instrument on all channels for all General MIDI instruments
- Chord breaks: no „key up” message needed

*Switch sound card  
to MIDI mode:*

```
mov    al,3fH
mov    dx,331H
out    dx,al
```

*Play a note:*

```
dec    dx
mov    al,90H ; key down, ch=1
out    dx,al
lodsb          ; pitch
out    dx,al
mov    al,7fH ; velocity=127
out    dx,al
```

## J. S. Bach: Prelude I. in C Major (BWV 846)

- Piano (patch 1) is the default instrument on all channels for all General MIDI instruments
- Chord breaks: no “key up” message needed

*Switch sound card to MIDI mode:*

```
mov al,3fH  
mov dx,331H  
out dx,al
```



```
; key down, ch=1  
; pitch  
; velocity=127
```

## *J. S. Bach: Prelude I. in C Major (BWV 846)*

1. Popular, well-known piece
2. Written for piano: optimal for MIDI
3. Simple rhythm, only a few tempo changes...

# *J. S. Bach: Prelude I. in C Major (BWV 846)*

Tempo changes:

- slow down around the end

## *J. S. Bach: Prelude I. in C Major (BWV 846)*

Tempo changes:

- slow down around the end
- set minimal pause for the last 5-note chord

## *J. S. Bach: Prelude I. in C Major (BWV 846)*

1. Popular, well-known piece
2. Written for piano: optimal for MIDI
3. Simple rhythm, only a few tempo changes
4. Contains repeating patterns...

J. S. Bach: *Prelude I. in C Major (BWV 846)*

Repeating patterns 1/2:

Piano

The musical score displays two staves for a piano. The top staff uses a treble clef and a common time signature (indicated by a '4'). The bottom staff uses a bass clef and a common time signature. Both staves feature a key signature of one sharp (F#). The music consists of repeating eighth-note patterns. In the treble clef staff, the pattern is a rest followed by a sixteenth note, then a sixteenth note followed by a sixteenth note. In the bass clef staff, the pattern is a sixteenth note followed by a sixteenth note, then a sixteenth note followed by a sixteenth note. The patterns are identical in both staves.

# J. S. Bach: Prelude I. in C Major (BWV 846)

Repeating patterns 1/2:

Piano

The image shows two staves of a piano musical score. The top staff is in treble clef and common time (indicated by '4'). The bottom staff is in bass clef and common time (indicated by '4'). The music consists of eighth-note patterns. In the first measure, there is a green circle around the first two notes of the treble staff. In the second measure, there is a blue circle around the first two notes of the treble staff. In the third measure, there is a pink circle around the first two notes of the treble staff. This pattern repeats in the bottom staff as well, with a green circle on the first two notes of the bass staff in the first measure, a blue circle on the second two notes in the second measure, and a pink circle on the second two notes in the third measure.

J. S. Bach: Prelude I. in C Major (BWV 846)

Repeating patterns 1/2:

Piano

A musical score for the piano part of J.S. Bach's Prelude I. in C Major (BWV 846). The score consists of two staves: a treble staff and a bass staff. The treble staff starts with a green circle highlighting a sixteenth-note pattern, followed by a blue circle, a pink circle, and another pink circle. The bass staff starts with a green circle, followed by a red circle, a purple circle, and a blue circle. The music is in common time (indicated by '4' at the beginning of each staff).

**16 → 8 notes**

A continuation of the musical score for the second half of the repeating pattern. The treble staff has a green circle, a red circle, a purple circle, and a blue circle. The bass staff has a green circle, a red circle, a purple circle, and a blue circle.

J. S. Bach: *Prelude I. in C Major (BWV 846)*

Repeating patterns 2/2:

Piano

The musical score consists of two staves for piano. The top staff is in common time (4/4) and the bottom staff is also in common time (4/4). Both staves feature a repeating pattern of eighth-note pairs followed by a sixteenth-note pair, with a bass note on the first beat of each measure. The pattern repeats four times in the top staff and three times in the bottom staff.

J. S. Bach: Prelude I. in C Major (BWV 846)

Repeating patterns 2/2:

Piano

The image shows two staves of musical notation for a piano. The top staff is in common time (indicated by '4') and the bottom staff is in common time (indicated by '4'). The notation consists of eighth and sixteenth notes. In the first measure, the right hand has a sixteenth-note pattern highlighted in purple, and the left hand has a eighth-note pattern highlighted in purple. In the second measure, the right hand has a sixteenth-note pattern highlighted in purple, and the left hand has a eighth-note pattern highlighted in purple. In the third measure, the right hand has a sixteenth-note pattern highlighted in green, and the left hand has a eighth-note pattern highlighted in green. In the fourth measure, the right hand has a sixteenth-note pattern highlighted in green, and the left hand has a eighth-note pattern highlighted in green. In the fifth measure, the right hand has a sixteenth-note pattern highlighted in blue, and the left hand has a eighth-note pattern highlighted in blue. In the sixth measure, the right hand has a sixteenth-note pattern highlighted in blue, and the left hand has a eighth-note pattern highlighted in blue. In the seventh measure, the right hand has a sixteenth-note pattern highlighted in yellow, and the left hand has a eighth-note pattern highlighted in yellow. In the eighth measure, the right hand has a sixteenth-note pattern highlighted in yellow, and the left hand has a eighth-note pattern highlighted in yellow. In the ninth measure, the right hand has a sixteenth-note pattern highlighted in red, and the left hand has a eighth-note pattern highlighted in red. In the tenth measure, the right hand has a sixteenth-note pattern highlighted in red, and the left hand has a eighth-note pattern highlighted in red.

J. S. Bach: Prelude I. in C Major (BWV 846)

Repeating patterns 2/2:

Piano

4

4

**8 → 5 notes**

4

4

*Raw Data*

<i>part</i>	<i>effective notes</i>	<i>raw data</i>
<i>repeating</i>	512	160
<i>non-repeating</i>	32	32
<i>final chord</i>	5	5
<b><i>total</i></b>	<b>549</b>	<b>197</b>

## II. Data

# *Data overview*

## Data overview

```
"c-3","e-3","g-3","c-4","e-4",
"c-3","d-3","a-3","d-4","f-4",
"h-2","d-3","g-3","d-4","f-4",
"c-3","e-3","g-3","c-4","e-4",
"c-3","e-3","a-3","e-4","a-4",
"c-3","d-3","f#3","a-3","d-4",
"h-2","d-3","g-3","d-4","g-4",
"h-2","c-3","e-3","g-3","c-4",
"a-2","c-3","e-3","g-3","c-4",
"d-2","a-2","d-3","f#3","c-4",
"g-2","h-2","d-3","g-3","h-3",
"g-2","a#2","e-3","g-3","c#4",
"f-2","a-2","d-3","a-3","d-4",
"f-2","g#2","d-3","f-3","h-3",
"e-2","g-2","c-3","g-3","c-4",
"e-2","f-2","a-2","c-3","f-3",
"d-2","f-2","a-2","c-3","f-3",
"g-1","d-2","g-2","h-2","f-3",
"c-2","e-2","g-2","c-3","e-3",
"c-2","g-2","a#2","c-3","e-3",
"f-1","f-2","a-2","c-3","e-3",
"f#1","c-2","a-2","c-3","e-3",
"g#1","f-2","h-2","c-3","d-3",
"g-1","f-2","g-2","h-2","d-3",
"g-1","e-2","g-2","c-3","e-3",
"g-1","d-2","g-2","c-3","f-3",
"g-1","d-2","g-2","h-2","f-3",
"g-1","d#2","a-2","c-3","f#3",
"g-1","e-2","g-2","c-3","g-3",
"g-1","d-2","g-2","c-3","f-3",
"g-1","d-2","g-2","h-2","f-3",
"c-1","c-2","g-2","a#2","e-3"
```

## Part 1:

# Data overview

```
"c-3","e-3","g-3","c-4","e-4",
"c-3","d-3","a-3","d-4","f-4",
"h-2","d-3","g-3","d-4","f-4",
"c-3","e-3","g-3","c-4","e-4",
"c-3","e-3","a-3","e-4","a-4",
"c-3","d-3","f#3","a-3","d-4",
"h-2","d-3","g-3","d-4","g-4",
"h-2","c-3","e-3","g-3","c-4",
"a-2","c-3","e-3","g-3","c-4",
"d-2","a-2","d-3","f#3","c-4",
"g-2","h-2","d-3","g-3","h-3",
"g-2","a#2","e-3","g-3","c#4",
"f-2","a-2","d-3","a-3","d-4",
"f-2","g#2","d-3","f-3","h-3",
"e-2","g-2","c-3","g-3","c-4",
"e-2","f-2","a-2","c-3","f-3",
"d-2","f-2","a-2","c-3","f-3",
"g-1","d-2","g-2","h-2","f-3",
"c-2","e-2","g-2","c-3","e-3",
"c-2","g-2","a#2","c-3","e-3",
"f-1","f-2","a-2","c-3","e-3",
"f#1","c-2","a-2","c-3","e-3",
"g#1","f-2","h-2","c-3","d-3",
"g-1","f-2","g-2","h-2","d-3",
"g-1","e-2","g-2","c-3","e-3",
"g-1","d-2","g-2","c-3","f-3",
"g-1","d-2","g-2","h-2","f-3",
"g-1","d#2","a-2","c-3","f#3",
"g-1","e-2","g-2","c-3","g-3",
"g-1","d-2","g-2","c-3","f-3",
"g-1","d-2","g-2","h-2","f-3",
"c-1","c-2","g-2","a#2","e-3"
```

## Part 1:

- 32 lines x 5 notes

# Data overview

"c-3", "e-3", "g-3", "c-4", "e-4",  
"c-3", "d-3", "a-3", "d-4", "f-4",  
"h-2", "d-3", "g-3", "d-4", "f-4",  
"c-3", "e-3", "g-3", "c-4", "e-4",  
  
"c-3", "e-3", "a-3", "e-4", "a-4",  
"c-3", "d-3", "f#3", "a-3", "d-4",  
"h-2", "d-3", "g-3", "d-4", "g-4",  
"h-2", "c-3", "e-3", "g-3", "c-4",  
  
"a-2", "c-3", "e-3", "g-3", "c-4",  
"d-2", "a-2", "d-3", "f#3", "c-4",  
"g-2", "h-2", "d-3", "g-3", "h-3",  
"g-2", "a#2", "e-3", "g-3", "c#4",  
  
"f-2", "a-2", "d-3", "a-3", "d-4",  
"f-2", "g#2", "d-3", "f-3", "h-3",  
"e-2", "g-2", "c-3", "g-3", "c-4",  
"e-2", "f-2", "a-2", "c-3", "f-3",  
  
"d-2", "f-2", "a-2", "c-3", "f-3",  
"g-1", "d-2", "g-2", "h-2", "f-3",  
"c-2", "e-2", "g-2", "c-3", "e-3",  
"c-2", "g-2", "a#2", "c-3", "e-3",  
  
"f-1", "f-2", "a-2", "c-3", "e-3",  
"f#1", "c-2", "a-2", "c-3", "e-3",  
"g#1", "f-2", "h-2", "c-3", "d-3",  
"g-1", "f-2", "g-2", "h-2", "d-3",  
  
"g-1", "e-2", "g-2", "c-3", "e-3",  
"g-1", "d-2", "g-2", "c-3", "f-3",  
"g-1", "d-2", "g-2", "h-2", "f-3",  
"g-1", "d#2", "a-2", "c-3", "f#3",  
  
"g-1", "e-2", "g-2", "c-3", "g-3",  
"g-1", "d-2", "g-2", "c-3", "f-3",  
"g-1", "d-2", "g-2", "h-2", "f-3",  
"c-1", "c-2", "g-2", "a#2", "e-3",

## Part 1:

- 32 lines x 5 notes
  - last 3 notes are repeated

## Data overview

```
"c-3","e-3","g-3","c-4","e-4",
"c-3","d-3","a-3","d-4","f-4",
"h-2","d-3","g-3","d-4","f-4",
"c-3","e-3","g-3","c-4","e-4",
"c-3","e-3","a-3","e-4","a-4",
"c-3","d-3","f#3","a-3","d-4",
"h-2","d-3","g-3","d-4","g-4",
"h-2","c-3","e-3","g-3","c-4",
"a-2","c-3","e-3","g-3","c-4",
"d-2","a-2","d-3","f#3","c-4",
"g-2","h-2","d-3","g-3","h-3",
"g-2","a#2","e-3","g-3","c#4",
"f-2","a-2","d-3","a-3","d-4",
"f-2","g#2","d-3","f-3","h-3",
"e-2","g-2","c-3","g-3","c-4",
"e-2","f-2","a-2","c-3","f-3",
"d-2","f-2","a-2","c-3","f-3",
"g-1","d-2","g-2","h-2","f-3",
"c-2","e-2","g-2","c-3","e-3",
"c-2","g-2","a#2","c-3","e-3",
"f-1","f-2","a-2","c-3","e-3",
"f#1","c-2","a-2","c-3","e-3",
"g#1","f-2","h-2","c-3","d-3",
"g-1","f-2","g-2","h-2","d-3",
"g-1","e-2","g-2","c-3","e-3",
"g-1","d-2","g-2","c-3","f-3",
"g-1","d-2","g-2","h-2","f-3",
"g-1","d#2","a-2","c-3","f#3",
"g-1","e-2","g-2","c-3","g-3",
"g-1","d-2","g-2","c-3","f-3",
"g-1","d-2","g-2","h-2","f-3",
"c-1","c-2","g-2","a#2","e-3"
```

### Part 1:

- 32 lines x 5 notes
- last 3 notes are repeated
- (8-note) lines are repeated

## Data overview

```
"c-3","e-3","g-3","c-4","e-4",
"c-3","d-3","a-3","d-4","f-4",
"b-2","d-3","g-3","d-4","f-4",
"c-3","e-3","g-3","c-4","e-4",
"c-3","e-3","a-3","e-4","a-4",
"c-3","d-3","f#3","a-3","d-4",
"b-2","d-3","g-3","d-4","g-4",
"b-2","c-3","e-3","g-3","c-4",
"b-2","c-3","e-3","g-3","c-4",
"b-2","d-3","f#3","c-4",
"b-2","d-3","g-3","h-3",
"b-2","a#2","e-3","g-3","c#4",
"b-2","a-2","d-3","a-3","d-4",
"b-2","g#2","d-3","f-3","h-3",
"e-2","g-2","c-3","g-3","c-4",
"e-2","f-2","a-2","c-3","f-3",
"b-2","f-2","a-2","c-3","f-3",
"b-1","d-2","g-2","h-2","f-3",
"c-2","e-2","g-2","c-3","e-3",
"c-2","g-2","a#2","c-3","e-3",
"b-1","f-2","a-2","c-3","e-3",
"b#1","c-2","a-2","c-3","e-3",
"b#1","f-2","h-2","c-3","d-3",
"b-1","f-2","g-2","h-2","d-3",
"b-1","e-2","g-2","c-3","e-3",
"b-1","d-2","g-2","c-3","f-3",
"b-1","d-2","g-2","h-2","f-3",
"b-1","d#2","a-2","c-3","f#3",
"b-1","e-2","g-2","c-3","g-3",
"b-1","d-2","g-2","c-3","f-3",
"b-1","d-2","g-2","h-2","f-3",
"b-1","c-2","g-2","a#2","e-3"
```

### Part 1:

- 32 lines x 5 notes
- last 3 notes are repeated
- (8-note) lines are repeated



## Data overview

```
"c-3","e-3","g-3","c-4","e-4",
"c-3","d-3","a-3","d-4","f-4",
"h-2","d-3","g-3","d-4","f-4",
"c-3","e-3","g-3","c-4","e-4",
"c-3","e-3","a-3","e-4","a-4",
"c-3","d-3","f#3","a-3","d-4",
"h-2","d-3","g-3","d-4","g-4",
"h-2","c-3","e-3","g-3","c-4",
"a-2","c-3","e-3","g-3","c-4",
"d-2","a-2","d-3","f#3","c-4",
"g-2","h-2","d-3","g-3","h-3",
"g-2","a#2","e-3","g-3","c#4",
"f-2","a-2","d-3","a-3","d-4",
"f-2","g#2","d-3","f-3","h-3",
"e-2","g-2","c-3","g-3","c-4",
"e-2","f-2","a-2","c-3","f-3",
"d-2","f-2","a-2","c-3","f-3",
"g-1","d-2","g-2","h-2","f-3",
"c-2","e-2","g-2","c-3","e-3",
"c-2","g-2","a#2","c-3","e-3",
"f-1","f-2","a-2","c-3","e-3",
"f#1","c-2","a-2","c-3","e-3",
"g#1","f-2","h-2","c-3","d-3",
"g-1","f-2","g-2","h-2","d-3",
"g-1","e-2","g-2","c-3","e-3",
"g-1","d-2","g-2","c-3","f-3",
"g-1","d-2","g-2","h-2","f-3",
"g-1","d#2","a-2","c-3","f#3",
"g-1","e-2","g-2","c-3","g-3",
"g-1","d-2","g-2","c-3","f-3",
"g-1","d-2","g-2","h-2","f-3",
"c-1","c-2","g-2","a#2","e-3"
```

### Part 1:

- 32 lines x 5 notes
- last 3 notes are repeated
- (8-note) lines are repeated

```
"c-1","c-2","f-2","a-2","c-3","f-3","c-3","a-2",
"c-3","a-2","f-2","a-2","f-2","d-2","f-2","d-2",
"c-1","h-1","g-3","h-3","d-4","f-4","d-4","h-3",
"d-4","h-3","g-3","h-3","d-3","f-3","e-3","d-3"
```

### Part 2:

- 32 notes
- no tricks

## Data overview

```
"c-3","e-3","g-3","c-4","e-4",
"c-3","d-3","a-3","d-4","f-4",
"h-2","d-3","g-3","d-4","f-4",
"c-3","e-3","g-3","c-4","e-4",
"c-3","e-3","a-3","e-4","a-4",
"c-3","d-3","f#3","a-3","d-4",
"h-2","d-3","g-3","d-4","g-4",
"h-2","c-3","e-3","g-3","c-4",
"a-2","c-3","e-3","g-3","c-4",
"d-2","a-2","d-3","f#3","c-4",
"g-2","h-2","d-3","g-3","h-3",
"g-2","a#2","e-3","g-3","c#4",
"f-2","a-2","d-3","a-3","d-4",
"f-2","g#2","d-3","f-3","h-3",
"e-2","g-2","c-3","g-3","c-4",
"e-2","f-2","a-2","c-3","f-3",
"d-2","f-2","a-2","c-3","f-3",
"g-1","d-2","g-2","h-2","f-3",
"c-2","e-2","g-2","c-3","e-3",
"c-2","g-2","a#2","c-3","e-3",
"f-1","f-2","a-2","c-3","e-3",
"f#1","c-2","a-2","c-3","e-3",
"g#1","f-2","h-2","c-3","d-3",
"g-1","f-2","g-2","h-2","d-3",
"g-1","e-2","g-2","c-3","e-3",
"g-1","d-2","g-2","c-3","f-3",
"g-1","d-2","g-2","h-2","f-3",
"g-1","d#2","a-2","c-3","f#3",
"g-1","e-2","g-2","c-3","g-3",
"g-1","d-2","g-2","c-3","f-3",
"g-1","d-2","g-2","h-2","f-3",
"c-1","c-2","g-2","a#2","e-3"
```

### Part 1:

- 32 lines x 5 notes
- last 3 notes are repeated
- (8-note) lines are repeated

```
"c-1","c-2","f-2","a-2","c-3","f-3","c-3","a-2",
"c-3","a-2","f-2","a-2","f-2","d-2","f-2","d-2",
"c-1","h-1","g-3","h-3","d-4","f-4","d-4","h-3",
"d-4","h-3","g-3","h-3","d-3","f-3","e-3","d-3"
```

### Part 2:

- 32 notes
- no tricks

```
"c-1","c-2","e-3","g-3","c-4"
```

### Part3:

- 5 notes
- no tricks

# Histogram of raw (31 values, 197 notes)

; c-1:36	4	####		;	1.	c-3:60	23	#####
; f-1:41	1	#		;	2.	g-2:55	14	#####
; f#1:42	1	#		;	3.	e-3:64	14	#####
; g-1:43	9	#####		;	4.	g-3:67	13	#####
; g#1:44	1	#		;	5.	d-3:62	12	#####
; h-1:47	1	#		;	6.	a-2:57	12	#####
; c-2:48	6	#####		;	7.	f-2:53	11	#####
; d-2:50	9	#####		;	8.	f-3:65	10	#####
; d#2:51	1	#		;	9.	h-2:59	9	#####
; e-2:52	5	####		;	10.	g-1:43	9	#####
; f-2:53	11	#####		;	11.	d-2:50	9	#####
; g-2:55	14	#####		;	12.	d-4:74	8	#####
; g#2:56	1	#		;	13.	c-4:72	7	#####
; a-2:57	12	#####		;	14.	h-3:71	6	#####
; a#2:58	3	##		;	15.	c-2:48	6	#####
; h-2:59	9	#####		;	16.	e-2:52	5	#####
; c-3:60	23	#####		;	17.	c-1:36	4	#####
; d-3:62	12	#####		;	18.	a-3:69	4	#####
; e-3:64	14	#####		;	19.	f-4:77	3	###
; f-3:65	10	#####		;	20.	f#3:66	3	###
; f#3:66	3	##		;	21.	e-4:76	3	###
; g-3:67	13	#####		;	22.	a#2:58	3	###
; a-3:69	4	##		;	23.	h-1:47	1	#
; h-3:71	6	##		;	24.	g-4:79	1	#
; c-4:72	7	##		;	25.	g#2:56	1	#
; c#4:73	1	#		;	26.	g#1:44	1	#
; d-4:74	8	##		;	27.	f-1:41	1	#
; e-4:76	3	##		;	28.	f#1:42	1	#
; f-4:77	3	##		;	29.	d#2:51	1	#
; g-4:79	1	#		;	30.	c#4:73	1	#
; a-4:81	1	#		;	31.	a-4:81	1	#

# Histogram of raw (31 values, 197 notes)

```

; c-1:36    4 #####
; f-1:41    1 #
; f#1:42   1 #
; g-1:43   1 #####
; g#1:44   1 #####
; h-1:47   1 #####
; c-2:48   1 #####
; d-2:50   1 #####
; d#2:51   1 #####
; e-2:52   1 #####
; f-2:53   1 #####
; g-2:55   1 #####
; g#2:56   1 #
; a-2:57  12 #####
; a#2:58   3 #####
; h-2:59   9 #####
; c-3:60  23 #####
; d-3:62  12 #####
; e-3:64  14 #####
; f-3:65  10 #####
; f#3:66   3 #####
; g-3:67  13 #####
; a-3:69   4 #####
; h-3:71   6 #####
; c-4:72   7 #####
; c#4:73   1 #
; d-4:74   8 #####
; e-4:76   3 #####
; f-4:77   3 #####
; g-4:79   1 #
; a-4:81   1 #

; 1. c-3:60  23 #####
; 2. g-2:55  14 #####
; 3. e-3:64  14 #####
; 4. g-3:67  13 #####
; 5. d-3:62  12 #####
; 6. f-2:53  11 #####
; 7. f-3:65  10 #####
; 9. h-2:59  9 #####
; 10. g-1:43 8 #####
; 11. d-2:51 7 #####
; 12. d-4:74 8 #####
; 13. c-4:72 7 #####
; 14. h-3:71 6 #####
; 15. c-2:48 6 #####
; 16. e-2:52 5 #####
; 17. c-1:36 4 #####
; 18. a-3:69 4 #####
; 19. f-4:77 3 #####
; 20. f#3:66 3 #####
; 21. e-4:76 3 #####
; 22. a#2:58 3 #####
; 23. h-1:47 1 #
; 24. g-4:79 1 #
; 25. g#2:56 1 #
; 26. g#1:44 1 #
; 27. f-1:41 1 #
; 28. f#1:42 1 #
; 29. d#2:51 1 #
; 30. c#4:73 1 #
; 31. a-4:81 1 #

```

**notes: 5 bit x 197 = 124 byte**

**table: 31 byte**

**total: 155 byte**

# Histogram of raw (31 values, 197 notes)

```

; c-1:36    4 #####
; f-1:41    1 #
; f#1:42    1 #
; g-1:43    0 #####
; g#1:44    1 #
; h-1:47    1 #
; c-2:48    6 #####
; d-2:50    9 #####
; d#2:51    1 #
; e-2:52    5 #####
; f-2:53    11 #####
; g-2:55    14 #####
; g#2:56    1 #
; a-2:57   12 #####
; a#2:58    3 #####
; h-2:59    9 #####
; c-3:60   23 #####
; d-3:62   12 #####
; e-3:64   14 #####
; f-3:65   10 #####
; f#3:66    3 #####
; g-3:67   13 #####
; a-3:69    4 #####
; h-3:71    6 #####
; c-4:72    7 #####
; c#4:73    1 #
; d-4:74    8 #####
; e-4:76    3 #####
; f-4:77    3 #####
; g-4:79    1 #
; a-4:81    1 #

; 1. c-3:60  23 #####
; 2. g-2:55  14 #####
; 3. e-3:64  14 #####
; 4. g-3:67  13 #####
; 5. d-3:62  12 #####
; 6. a-2:58  11 #####
; 7. f-2:53  10 #####
; 8. f-3:65  9 #####
; 9. h-2:59  8 #####
; 10. g-1:43 7 #####
; 11. d-2:50 6 #####
; 12. d#2:51 5 #####
; 13. c-4:72  7 #####
; 14. h-3:71  6 #####
; 15. c-2:48  6 #####
; 16. e-2:52  5 #####
; 17. c-1:36  4 #####
; 18. a-3:69  4 #####
; 19. f-4:77  3 #####
; 20. f#3:66  3 #####
; 21. e-4:76  3 #####
; 22. a#2:58  3 #####
; 23. h-1:47  1 #
; 24. g-4:79  1 #
; 25. g#2:56  1 #
; 26. g#1:44  1 #
; 27. f-1:41  1 #
; 28. f#1:42  1 #
; 29. d#2:51  1 #
; 30. c#4:73  1 #
; 31. a-4:81  1 #

```

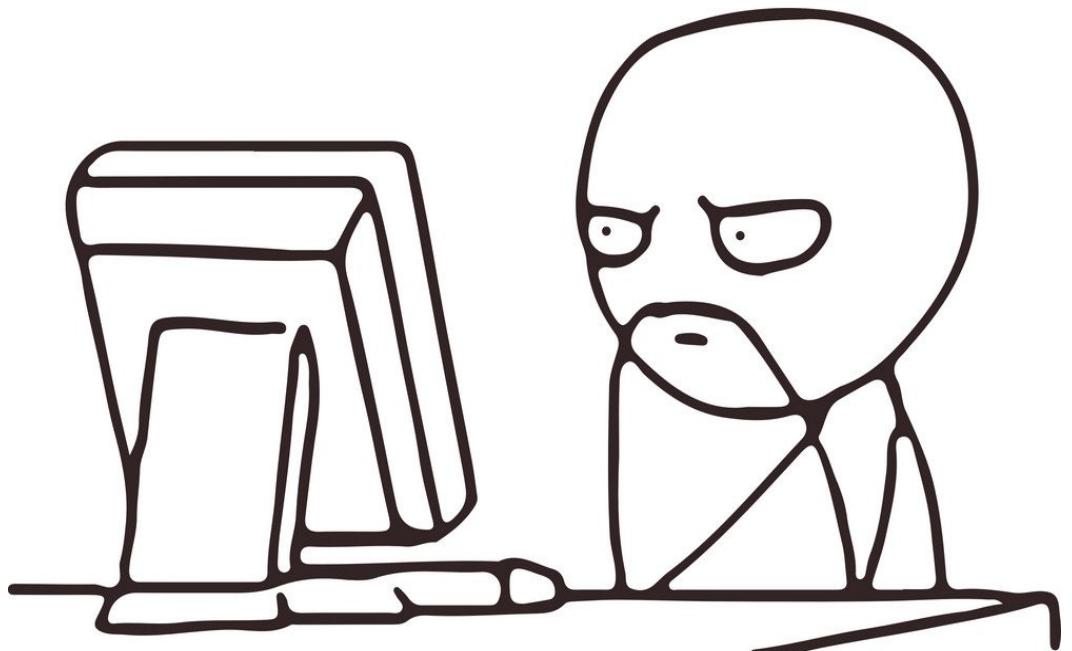
**note range: 36..81: 45 values**

**values: 6 bit x 197 = 148 byte**

*Think Diff*

# Think Diff

```
; c-3:60  e-3:64  g-3:67  c-4:72  e-4:76  (...)  
; c-3:60  d-3:62  a-3:69  d-4:74  f-4:77  (...)  
; h-2:59  d-3:62  g-3:67  d-4:74  f-4:77  (...)  
; c-3:60  e-3:64  g-3:67  c-4:72  e-4:76  (...)  
; c-3:60  e-3:64  a-3:69  e-4:76  a-4:81  (...)  
; c-3:60  d-3:62  f#3:66  a-3:69  d-4:74  (...)  
; h-2:59  d-3:62  g-3:67  d-4:74  g-4:79  (...)  
; h-2:59  c-3:60  e-3:64  g-3:67  c-4:72  (...)  
; a-2:57  c-3:60  e-3:64  g-3:67  c-4:72  (...)  
; d-2:50  a-2:57  d-3:62  f#3:66  c-4:72  (...)  
; g-2:55  h-2:59  d-3:62  g-3:67  h-3:71  (...)  
; g-2:55  a#2:58  e-3:64  g-3:67  c#4:73  (...)  
; f-2:53  a-2:57  d-3:62  a-3:69  d-4:74  (...)  
; f-2:53  g#2:56  d-3:62  f-3:65  h-3:71  (...)  
; e-2:52  g-2:55  c-3:60  g-3:67  c-4:72  (...)  
; e-2:52  f-2:53  a-2:57  c-3:60  f-3:65  (...)  
; d-2:50  f-2:53  a-2:57  c-3:60  f-3:65  (...)  
; g-1:43  d-2:50  g-2:55  h-2:59  f-3:65  (...)  
; c-2:48  e-2:52  g-2:55  c-3:60  e-3:64  (...)  
; c-2:48  g-2:55  a#2:58  c-3:60  e-3:64  (...)  
; f-1:41  f-2:53  a-2:57  c-3:60  e-3:64  (...)  
; f#1:42  c-2:48  a-2:57  c-3:60  e-3:64  (...)  
; g#1:44  f-2:53  h-2:59  c-3:60  d-3:62  (...)  
; g-1:43  f-2:53  g-2:55  h-2:59  d-3:62  (...)  
; g-1:43  e-2:52  g-2:55  c-3:60  e-3:64  (...)  
; g-1:43  d-2:50  g-2:55  c-3:60  f-3:65  (...)  
; g-1:43  d-2:50  g-2:55  h-2:59  f-3:65  (...)  
; g-1:43  d#2:51  a-2:57  c-3:60  f#3:66  (...)  
; g-1:43  e-2:52  g-2:55  c-3:60  g-3:67  (...)  
; g-1:43  d-2:50  g-2:55  c-3:60  f-3:65  (...)  
; g-1:43  d-2:50  g-2:55  h-2:59  f-3:65  (...)  
; c-1:36  c-2:48  g-2:55  a#2:58  e-3:64  (...)
```



```
; c-1:36  c-2:48  f-2:53  a-2:57  c-3:60  f-3:65  c-3:60  a-2:57  
; c-3:60  a-2:57  f-2:53  a-2:57  f-2:53  d-2:50  f-2:53  d-2:50  
; c-1:36  h-1:47  g-3:67  h-3:71  d-4:74  f-4:77  d-4:74  h-3:71  
; d-4:74  h-3:71  g-3:67  h-3:71  d-3:62  f-3:65  e-3:64  d-3:62  
; c-1:36  c-2:48  e-3:64  g-3:67  c-4:72
```

# Think Diff

```
; c-3:60 e-3:64 g-3:67 c-4:72 e-4:76 (...)  
; c-3:60 d-3:62 a-3:69 d-4:74 f-4:77 (...)  
; h-2:59 d-3:62 g-3:67 d-4:74 f-4:77 (...)  
; c-3:60 e-3:64 g-3:67 c-4:72 e-4:76 (...)  
; a-3:60 e-3:64 a-3:69 e-4:76 a-4:81 (...)  
; c-3:60 d-3:62 f#3:66 a-3:69 d-4:74 (...)  
h-2:59 d-3:62 g-3:67 d-4:74 g-4:79 (...)  
h-2:59 c-3:60 e-3:64 g-3:67 c-4:72 (...)  
a-2:57 c-3:60 e-3:64 g-3:67 c-4:72 (...)  
d-2:50 a-2:57 d-3:62 f#3:66 c-4:72 (...)  
g-2:55 h-2:59 d-3:62 g-3:67 h-3:71 (...)  
g-2:55 a#2:58 e-3:64 g-3:67 c#4:73 (...)  
f-2:53 a-2:57 d-3:62 a-3:69 d-4:74 (...)  
e-2:53 g#2:56 d-3:62 a-3:69 d-4:71 (...)  
e-2:52 g-2:55 c-3:60 g-3:67 (...)  
e-2:52 f-2:53 a-2:57 c-3:60 (...)  
d-2:50 f-2:53 a-2:57 c-3:60 f-3:65 (...)  
g-1:43 d-2:50 g-2:55 h-2:59 f-3:65 (...)  
c-2:48 e-2:52 g-2:55 c-3:60 e-3:64 (...)  
c-2:48 g-2:55 a#2:58 c-3:60 e-3:64 (...)  
f-1:41 f-2:53 a-2:57 c-3:60 e-3:64 (...)  
f#1:42 c-2:48 a-2:57 c-3:60 e-3:64 (...)  
g#1:44 f-2:53 h-2:59 c-3:60 d-3:62 (...)  
g-1:43 f-2:53 g-2:55 h-2:59 d-3:62 (...)  
g-1:43 e-2:52 g-2:55 c-3:60 e-3:64 (...)  
g-1:43 d-2:50 g-2:55 c-3:60 f-3:65 (...)  
g-1:43 d-2:50 g-2:55 h-2:59 f-3:65 (...)  
g-1:43 d#2:51 a-2:57 c-3:60 f#3:66 (...)  
g-1:43 e-2:52 g-2:55 c-3:60 g-3:67 (...)  
g-1:43 d-2:50 g-2:55 c-3:60 f-3:65 (...)  
g-1:43 d-2:50 g-2:55 h-2:59 f-3:65 (...)  
c-1:36 c-2:48 g-2:55 a#2:58 e-3:64 (...)
```

```
; c-1:36 c-2:48 f-2:53 a-2:57 c-3:60 f-3:65 c-3:60 a-2:57  
; c-3:60 a-2:57 f-2:53 a-2:57 f-2:53 d-2:50 f-2:53 d-2:50  
; c-1:36 h-1:47 g-3:67 h-3:71 d-4:74 f-4:77 d-4:74 h-3:71  
; d-4:74 h-3:71 g-3:67 h-3:71 d-3:62 f-3:65 e-3:64 d-3:62  
; c-1:36 c-2:48 e-3:64 g-3:67 c-4:72
```

## Focus on values of Part 1



# Think Diff

c-3:60	e-3:64	g-3:67	c-4:72	e-4:76	d-2:50	f-2:53	a-2:57	c-3:60	f-3:65
c-3:60	d-3:62	a-3:69	d-4:74	f-4:77	g-1:43	d-2:50	g-2:55	h-2:59	f-3:65
h-2:59	d-3:62	g-3:67	d-4:74	f-4:77	c-2:48	e-2:52	g-2:55	c-3:60	e-3:64
c-3:60	e-3:64	g-3:67	c-4:72	e-4:76	c-2:48	g-2:55	a#2:58	c-3:60	e-3:64
c-3:60	e-3:64	a-3:69	e-4:76	a-4:81	f-1:41	f-2:53	a-2:57	c-3:60	e-3:64
c-3:60	d-3:62	f#3:66	a-3:69	d-4:74	f#1:42	c-2:48	a-2:57	c-3:60	e-3:64
h-2:59	d-3:62	g-3:67	d-4:74	g-4:79	g#1:44	f-2:53	h-2:59	c-3:60	e-3:64
h-2:59	c-3:60	e-3:64	g-3:67	c-4:72	g-1:43	f-2:53	g-2:55	h-2:59	d-3:62
a-2:57	c-3:60	e-3:64	g-3:67	c-4:72	g-1:43	e-2:52	g-2:55	c-3:60	d-3:62
d-2:50	a-2:57	d-3:62	f#3:66	c-4:72	g-1:43	d-2:50	g-2:55	c-3:60	e-3:64
g-2:55	h-2:59	d-3:62	g-3:67	h-3:71	g-1:43	d-2:50	g-2:55	c-3:60	f-3:65
g-2:55	a#2:58	e-3:64	g-3:67	c#4:73	g-1:43	d-2:50	g-2:55	h-2:59	f-3:65
f-2:53	a-2:57	d-3:62	a-3:69	d-4:74	g-1:43	d#2:51	a-2:57	c-3:60	f#3:66
f-2:53	g#2:56	d-3:62	f-3:65	h-3:71	g-1:43	e-2:52	g-2:55	c-3:60	g-3:67
e-2:52	g-2:55	c-3:60	g-3:67	c-4:72	g-1:43	d-2:50	g-2:55	c-3:60	f-3:65
e-2:52	f-2:53	a-2:57	c-3:60	f-3:65	g-1:43	d-2:50	g-2:55	h-2:59	f-3:65
					c-1:36	c-2:48	g-2:55	a#2:58	e-3:64

# Think Diff

c-3:60	e-3:64	g-3:67	c-4:72	e-4:76	d-2:50	f-2:53	a-2:57	c-3:60	f-3:65
c-3:60	d-3:62	a-3:69	d-4:74	f-4:77	g-1:43	d-2:50	g-2:55	h-2:59	f-3:65
h-2:59	d-3:62	g-3:67	d-4:74	f-4:77	c-2:48	e-2:52	g-2:55	c-3:60	e-3:64
c-3:60	e-3:64	g-3:67	c-4:72	e-4:76	c-2:48	g-2:55	a#2:58	c-3:60	e-3:64
c-3:60	e-3:64	a-3:69	e-4:76	a-4:81	f-1:41	f-2:53	a-2:57	c-3:60	e-3:64
c-3:60	d-3:62	f#3:66	a-3:69	d-4:74	f#1:42	c-2:48	a-2:57	c-3:60	e-3:64
h-2:59	d-3:62	g-3:67	d-4:74	g-4:79	g#1:44	f-2:53	h-2:59	c-3:60	d-3:62
h-2:59	c-3:60	e-3:64	g-3:67	c-4:72	g-1:43	f-2:53	g-2:55	h-2:59	d-3:62
a-2:57	c-3:60	e-3:64	g-3:67	c-4:72	g-1:43	e-2:52	g-2:55	c-3:60	e-3:64
d-2:50	a-2:57	d-3:62	f#3:66	c-4:72	g-1:43	d-2:50	g-2:55	c-3:60	f-3:65
g-2:55	h-2:59	d-3:62	g-3:67	h-3:71	g-1:43	d-2:50	g-2:55	h-2:59	f-3:65
g-2:55	a#2:58	e-3:64	g-3:67	c#4:73	g-1:43	d#2:51	a-2:57	c-3:60	f#3:66
f-2:53	a-2:57	d-3:62	a-3:69	d-4:74	g-1:43	e-2:52	g-2:55	c-3:60	g-3:67
f-2:53	g#2:56	d-3:62	f-3:65	h-3:71	g-1:43	d-2:50	g-2:55	c-3:60	f-3:65
e-2:52	g-2:55	c-3:60	g-3:67	c-4:72	g-1:43	d-2:50	g-2:55	h-2:59	f-3:65
e-2:52	f-2:53	a-2:57	c-3:60	f-3:65	c-1:36	c-2:48	g-2:55	a#2:58	e-3:64

# Think Diff

c-3:60	e-3:64	g-3:67	c-4:72	e-4:76
c-3:60	d-3:62	a-3:69	d-4:74	f-4:77
h-2:59	d-3:62	g-3:67	d-4:74	f-4:77
c-3:60	e-3:64	g-3:67	c-4:72	e-4:76
c-3:60	e-3:64	a-3:69	e-4:76	a-4:81
c-3:60	d-3:62	f#3:66	a-3:69	d-4:74
h-2:59	d-3:62	g-3:67	d-4:74	g-4:79
h-2:59	c-3:60	e-3:64	g-3:67	c-4:72
a-2:57	c-3:60	e-3:64	g-3:67	c-4:72
d-2:50	a-2:57	d-3:62	f#3:66	c-4:72
g-2:55	h-2:59	d-3:62	g-3:67	h-3:71
g-2:55	a#2:58	e-3:64	g-3:67	c#4:73
f-2:53	a-2:57	d-3:62	a-3:69	d-4:74
f-2:53	g#2:56	d-3:62	f-3:65	h-3:71
e-2:52	g-2:55	c-3:60	g-3:67	c-4:72
e-2:52	f-2:53	a-2:57	c-3:60	f-3:65

d-2:50	f-2:53	a-2:57	c-3:60	f-3:65
g-1:43	d-2:50	g-2:55	h-2:59	f-3:65
c-2:48	e-2:52	g-2:55	c-3:60	e-3:64
c-2:48	g-2:55	a#2:58	c-3:60	e-3:64
f-1:41	f-2:53	a-2:57	c-3:60	e-3:64
f#1:42	c-2:48	a-2:57	c-3:60	e-3:64
g#1:44	f-2:53	h-2:59	c-3:60	d-3:62
g-1:43	f-2:53	g-2:55	h-2:59	d-3:62
g-1:43	e-2:52	g-2:55	c-3:60	e-3:64
g-1:43	d-2:50	g-2:55	c-3:60	f-3:65
g-1:43	d-2:50	g-2:55	h-2:59	f-3:65
g-1:43	d#2:51	a-2:57	c-3:60	f#3:66
g-1:43	e-2:52	g-2:55	c-3:60	g-3:67
g-1:43	d-2:50	g-2:55	c-3:60	f-3:65
g-1:43	d-2:50	g-2:55	h-2:59	f-3:65
c-1:36	c-2:48	g-2:55	a#2:58	e-3:64



The raw-diff-5 theory

*Think Diff: why raw-diff-5?*

Why raw?

## *Think Diff: why raw-diff-5?*

### Why raw?

- Two options:
  - raw MIDI notes or
  - mapped data

## *Think Diff: why raw-diff-5?*

### Why raw?

- Two options:
  - raw MIDI notes or
  - mapped data
- Dispersion: 31 values in range of 45

## *Think Diff: why raw-diff-5?*

### Why raw?

- Two options:
  - raw MIDI notes or
  - mapped data
- Dispersion: 31 values in range of 45
- Indexed requires extra 31-byte table

## *Think Diff: why raw-diff-5?*

### Why raw?

- Two options:
  - raw MIDI notes or
  - mapped data
- Dispersion: 31 values in range of 45
- Indexed requires extra 31-byte table
- Can't compress table, only data

*Think Diff: why raw-diff-5?*

Why diff-5?

*Think Diff: why raw-diff-5?*

Why diff-5?

- Part 1 contains chord breaks

*Think Diff: why raw-diff-5?*

## Why diff-5?

- Part 1 contains chord breaks
- Chords are evolving slowly to next one

*Think Diff: why raw-diff-5?*

## Why diff-5?

- Part 1 contains chord breaks
- Chords are evolving slowly to next one
- Slowness is emphasized by repeating all chord breaks twice (not stored)

*Think Diff: why raw-diff-5?*

## Why diff-5?

- Part 1 contains chord breaks
- Chords are evolving slowly to next one
- Slowness is emphasized by repeating all chord breaks twice (not stored)
- **Slow change means small diffs**

*Think Diff: why raw-diff-5?*

## Why diff-5?

- Part 1 contains chord breaks
- Chords are evolving slowly to next one
- Slowness is emphasized by repeating all chord breaks twice (not stored)
- Slow change means small diffs
- Chord breaks are 8 notes long

## *Think Diff: why raw-diff-5?*

### Why diff-5?

- Part 1 contains chord breaks
- Chords are evolving slowly to next one
- Slowness is emphasized by repeating all chord breaks twice (not stored)
- Slow change means small diffs
- Chord breaks are 8 notes long
- Which are stored in 5 bytes

## *Think Diff: why raw-diff-5?*

### Why diff-5?

- Part 1 contains chord breaks
- Chords are evolving slowly to next one
- Slowness is emphasized by repeating all chord breaks twice (not stored)
- Slow change means small diffs
- Chord breaks are 8 notes long
- Which are stored in 5 bytes
- Diff-5 is diff to previous line

*Intermission: raw-diff-mixed/1/5*

What is diff-mixed/1/5?

## *Intermission: raw-diff-mixed/1/5*

What is diff-mixed/1/5?

- First note of the line: diff-5 (prev line)

## *Intermission: raw-diff-mixed/1/5*

### What is diff-mixed/1/5?

- First note of the line: diff-5 (prev line)
- Other notes: diff-1 (prev note)

## *Intermission: raw-diff-mixed/1/5*

### What is diff-mixed/1/5?

- First note of the line: diff-5 (prev line)
- Other notes: diff-1 (prev note)
- **No negative diff-1 values**

## *Intermission: raw-diff-mixed/1/5*

### What is diff-mixed/1/5?

- First note of the line: diff-5 (prev line)
- Other notes: diff-1 (prev note)
- No negative diff-1 values
- Requires extra code

*Think Diff: raw-diff-5 data overview*

Added diff values in dump:

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01
c-3:60:=00	e-3:64:=00	a-3:69:+02	e-4:76:+04	a-4:81:+05
c-3:60:=00	d-3:62:-02	f#3:66:-03	a-3:69:-07	d-4:74:-07
h-2:59:-01	d-3:62:=00	g-3:67:+01	d-4:74:+05	g-4:79:+05
( . . . )				

# Think Diff: raw-diff-5 data overview

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00							
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01							
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00	d-2:50:-02	f-2:53:=00	a-2:57:=00	c-3:60:=00	f-3:65:=00		
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01	g-1:43:-07	d-2:50:-03	g-2:55:-02	h-2:59:-01	f-3:65:=00		
c-3:60:=00	e-3:64:=00	a-3:69:+02	e-4:76:+04	a-4:81:+05	c-2:48:+05	e-2:52:+02	g-2:55:=00	c-3:60:+01	e-3:64:-01		
c-3:60:=00	d-3:62:-02	f#3:66:-03	a-3:69:-07	d-4:74:-07	c-2:48:=00	g-2:55:+03	a#2:58:+03	c-3:60:=00	e-3:64:=00		
h-2:59:-01	d-3:62:=00	g-3:67:+01	d-4:74:+05	g-4:79:+05	f-1:41:-07	f-2:53:-02	a-2:57:-01	c-3:60:=00	e-3:64:=00		
h-2:59:=00	c-3:60:-02	e-3:64:-03	g-3:67:-07	c-4:72:-07	f#1:42:+01	c-2:48:-05	a-2:57:=00	c-3:60:=00	e-3:64:=00		
a-2:57:-02	c-3:60:=00	e-3:64:=00	g-3:67:=00	c-4:72:=00	g#1:44:+02	f-2:53:+05	h-2:59:+02	c-3:60:=00	d-3:62:-02		
d-2:50:-07	a-2:57:-03	d-3:62:-02	f#3:66:-01	c-4:72:=00	g-1:43:-01	f-2:53:=00	g-2:55:-04	h-2:59:-01	d-3:62:=00		
g-2:55:+05	h-2:59:+02	d-3:62:=00	g-3:67:+01	h-3:71:-01	g-1:43:=00	e-2:52:-01	g-2:55:=00	c-3:60:+01	e-3:64:+02		
g-2:55:=00	a#2:58:-01	e-3:64:+02	g-3:67:=00	c#4:73:+02	g-1:43:=00	d-2:50:-02	g-2:55:=00	c-3:60:=00	f-3:65:+01		
f-2:53:-02	a-2:57:-01	d-3:62:-02	a-3:69:+02	d-4:74:+01	g-1:43:=00	d-2:50:=00	g-2:55:=00	h-2:59:-01	f-3:65:=00		
f-2:53:=00	g#2:56:-01	d-3:62:=00	f-3:65:-04	h-3:71:-03	g-1:43:=00	d#2:51:+01	a-2:57:+02	c-3:60:+01	f#3:66:+01		
e-2:52:-01	g-2:55:-01	c-3:60:-02	g-3:67:+02	c-4:72:+01	g-1:43:=00	e-2:52:+01	g-2:55:-02	c-3:60:=00	g-3:67:+01		
e-2:52:=00	f-2:53:-02	a-2:57:-03	c-3:60:-07	f-3:65:-07	g-1:43:=00	d-2:50:-02	g-2:55:=00	c-3:60:=00	f-3:65:-02		
					g-1:43:=00	d-2:50:=00	g-2:55:=00	h-2:59:-01	f-3:65:=00		
					c-1:36:-07	c-2:48:-02	g-2:55:=00	a#2:58:-01	e-3:64:-01		

c-1:36:=00	c-2:48:=00	f-2:53:-02	a-2:57:-01	c-3:60:-04	f-3:65:+29	c-3:60:+12	a-2:57:+04				
c-3:60:+03	a-2:57:-03	f-2:53:-12	a-2:57:-03	f-2:53:-04	d-2:50:-10	f-2:53:-04	d-2:50:-03				
c-1:36:-21	h-1:47:-06	g-3:67:+17	h-3:71:+18	d-4:74:+24	f-4:77:+41	d-4:74:+27	h-3:71:+04				
d-4:74:+03	h-3:71:-03	g-3:67:-10	h-3:71:-03	d-3:62:-09	f-3:65:-09	e-3:64:-07	d-3:62:-05				

c-1:36:-35    c-2:48:-14    e-3:64:-01    g-3:67:+03    c-4:72:+10

# Think Diff: raw-diff-5 data overview

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01
c-3:60:=00	e-3:64:=00	a-3:69:+02	e-4:76:+04	a-4:81:+05
c-3:60:=00	d-3:62:-02	f#3:66:-03	a-3:69:-07	d-4:74:-07
h-2:59:-01	d-3:62:=00	g-3:67:+01	d-4:74:+05	g-4:79:+05
h-2:59:=00	c-3:60:-02	e-3:64:-03	g-3:67:-07	c-4:72:-07
a-2:57:-02	c-3:60:=00	e-3:64:=00	g-3:67:=00	c-4:72:=00
d-2:50:-07	a-2:57:-03	d-3:62:-02	f#3:66:-01	c-4:72:=00
g-2:55:+05	h-2:59:+02	d-3:62:=00	g-3:67:+01	h-3:71:-01
g-2:55:=00	a#2:58:-01	e-3:64:+02	g-3:67:=00	c#4:73:+02
f-2:53:-02	a-2:57:-01	d-3:62:-02	a-3:69:+02	d-4:74:+01
f-2:53:=00	g#2:56:-01	d-3:62:=00	f-3:65:-04	h-3:71:-03
e-2:52:-01	g-2:55:-01	c-3:60:-02	g-3:67:+02	c-4:72:+01
e-2:52:=00	f-2:53:-02	a-2:57:-03	c-3:60:-07	f-3:65:-07

d-2:50:-02	f-2:53:=00	a-2:57:=00	c-3:60:=00	f-3:65:=00
g-1:43:-07	d-2:50:-03	g-2:55:-02	h-2:59:-01	f-3:65:=00
c-2:48:+05	e-2:52:+02	g-2:55:=00	c-3:60:+01	e-3:64:-01
c-2:48:=00	g-2:55:+03	a#2:58:+03	c-3:60:=00	e-3:64:=00
f-1:41:-07	f-2:53:-02	a-2:57:-01	c-3:60:=00	e-3:64:=00
f#1:42:+01	c-2:48:-05	a-2:57:=00	c-3:60:=00	e-3:64:=00
g#1:44:+02	f-2:53:+05	h-2:59:+02	c-3:60:=00	d-3:62:-02
g-1:43:-01	f-2:53:=00	g-2:55:-04	h-2:59:-01	d-3:62:=00
g-1:43:=00	e-2:52:-01	g-2:55:=00	c-3:60:+01	e-3:64:+02
g-1:43:=00	d-2:50:-02	g-2:55:=00	c-3:60:=00	f-3:65:+01
g-1:43:=00	d-2:50:=00	g-2:55:=00	h-2:59:-01	f-3:65:=00
g-1:43:=00	d#2:51:+01	a-2:57:+02	c-3:60:+01	f#3:66:+01
g-1:43:=00	e-2:52:+01	g-2:55:-02	c-3:60:=00	g-3:67:+01
g-1:43:=00	d-2:50:-02	g-2:55:=00	c-3:60:=00	f-3:65:-02
g-1:43:=00	d-2:50:=00	g-2:55:=00	h-2:59:-01	f-3:65:=00
c-1:36:-07	c-2:48:-02	g-2:55:=00	a#2:58:-01	e-3:64:-01

c-1:36:=00	c-2:48:=00	f-2:53:-02	a-2:57:-01	c-3:60:-04	f-3:65:+29	c-3:60:+12	a-2:57:+04
c-3:60:+03	a-2:57:-03	f-2:53:+12	a-2:57:-03	f-2:53:-04	d-2:50:-10	f-2:53:-04	d-2:50:-03
c-1:36:-21	h-1:47:-06	g-3:67:+17	h-3:71:+18	d-4:74:+24	f-4:77:+41	d-4:74:+27	h-3:71:+04
d-4:74:+03	h-3:71:-03	g-3:67:-10	h-3:71:-03	d-3:62:-09	f-3:65:-09	e-3:64:-07	d-3:62:-05

c-1:36:**-35** c-2:48:**-14** e-3:64:-01 g-3:67:+03 c-4:72:+10

# Think Diff: raw-diff-5 data overview

```
c-3:60:/00 e-3:64:/00 g-3:67:/00 c-4:72:/00 e-4:76:/00
c-3:60:=00 d-3:62:-02 a-3:69:+02 d-4:74:+02 f-4:77:+01
h-2:59:-01 d-3:62:=00 g-3:67:-02 d-4:74:=00 f-4:77:=00
c-3:60:+01 e-3:64:+02 g-3:67:+00 c-4:72:-02 e-4:76:-01
c-3:60:=00 e-3:64:=00 a-3:69:+02 e-4:76:+04 a-4:81:+05
c-3:60:=00 d-3:62:-02 f#3:66:-03 a-3:69:-07 d-4:74:-07
h-2:59:-01 d-3:62:=00 g-3:67:+01 d-4:74:+05 g-4:79:+05
h-2:59:=00 c-3:60:-02 e-3:64:-03 g-3:67:-07 c-4:72:-07
a-2:57:-02 c-3:60:=00 e-3:64:=00 g-3:67:=00 c-4:72:=00
d-2:50:-07 a-2:57:-03 d-3:62:-02 f#3:66:-01 c-4:72:=00
g-2:55:+05 h-2:59:+02 d-3:62:=00 g-3:67:+01 h-3:71:-01
g-2:55:+00 a#2:58:-01 e-3:64:+02 g-3:67:=00 c#4:73:+02
f-2:53:-02 a-2:57:-01 d-3:62:-02 a-3:69:+02 d-4:74:+01
f-2:53:=00 g#2:56:-01 d-3:62:=00 f-3:65:-04 h-3:71:-03
e-2:52:-01 g-2:55:-01 c-3:60:-02 g-3:67:+02 c-4:72:+01
e-2:52:=00 f-2:53:-02 a-2:57:-03 c-3:60:-07 f-3:65:-07
```

d-2:50:-02	f-2:53:=00	a-2:57:=00	c-3:60:=00	f-3:65:=00
g-1:43:-07	d-2:50:-03	g-2:55:-02	h-2:59:-01	f-3:65:=00
c-2:48:+05	e-2:52:+02	g-2:55:=00	c-3:60:+01	e-3:64:-01
c-2:48:=00	g-2:55:+03	a#2:58:+03	c-3:60:=00	e-3:64:=00
f-1:41:-07	f-2:53:-02	a-2:57:-01	c-3:60:=00	e-3:64:=00
f#1:42:+01	c-2:48:-05	a-2:57:=00	c-3:60:=00	e-3:64:=00
g#1:44:+02	f-2:53:+05	h-2:59:+02	c-3:60:=00	d-3:62:-02
g-1:43:-01	f-2:53:=00	g-2:55:-04	h-2:59:-01	d-3:62:=00
g-1:43:=00	e-2:52:-01	g-2:55:=00	c-3:60:+01	e-3:64:+02
g-1:43:=00	d-2:50:-02	g-2:55:=00	c-3:60:=00	f-3:65:+01
g-1:43:=00	d-2:50:=00	g-2:55:=00	h-2:59:-01	f-3:65:=00
g-1:43:=00	d#2:51:+01	a-2:57:+02	c-3:60:+01	f#3:66:+01
g-1:43:=00	e-2:52:+01	g-2:55:-02	c-3:60:=00	g-3:67:+01
g-1:43:=00	d-2:50:-02	g-2:55:=00	c-3:60:=00	f-3:65:-02
g-1:43:=00	d-2:50:=00	g-2:55:=00	h-2:59:-01	f-3:65:=00
c-1:36:-07	c-2:48:-02	g-2:55:=00	a#2:58:-01	e-3:64:-01

```
c-1:36:=00 c-2:48:=00 f-2:53:-02 a-2:57:-01 c-3:60:-04 f-3:65:+29 c-3:60:+12 a-2:57:+04
c-3:60:+03 a-2:57:-03 f-2:53:+12 a-2:57:-03 f-2:53:-04 d-2:50:-10 f-2:53:-04 d-2:50:-03
c-1:36:-21 h-1:47:-06 g-3:67:+17 h-3:71:+18 d-4:74:+24 f-4:77:+41 d-4:74:+27 h-3:71:+04
d-4:74:+03 h-3:71:-03 g-3:67:-10 h-3:71:-03 d-3:62:-09 f-3:65:-09 e-3:64:-07 d-3:62:-05
```

```
c-1:36:+35 c-2:48:-14 e-3:64:-01 g-3:67:+03 c-4:72:+10
```

# Think Diff: raw-diff-5 data overview

```

c-3:60:/00 e-3:64:/00 g-3:67:/00 c-4:72:/00 e-4:76:/00
c-3:60:=00 d-3:62:-02 a-3:69:+02 d-4:74:+02 f-4:77:+01
h-2:59:-01 d-3:62:=00 g-3:67:-02 d-4:74:+00 f-4:77:+00
c-3:60:+01 e-3:64:+02 g-3:67:=00 c-4:72:-02 e-4:76:-01
c-3:60:=00 e-3:64:=00 a-3:69:+02 e-4:76:+04 a-4:81:+05
c-3:60:=00 d-3:62:-02 f#3:66:-03 a-3:69:-07 d-4:74:+07
h-2:59:-01 d-3:62:=00 g-3:67:+01 d-4:74:+05 g-4:79:+05
h-2:59:=00 c-3:60:-02 e-3:64:-03 g-3:67:-07 c-4:72:-07
a-2:57:-02 c-3:60:=00 e-3:64:=00 g-3:67:+00 c-4:72:+00
d-2:50:-07 a-2:57:-03 d-3:62:-02 f#3:66:-01 c-4:72:+00
g-2:55:+05 h-2:59:+02 d-3:62:=00 g-3:67:+01 h-3:71:-01
g-2:55:=00 a#2:58:-01 e-3:64:+02 g-3:67:+00 c#4:73:+02
f-2:53:-02 a-2:57:-01 d-3:62:-02 a-3:69:+02 d-4:74:+01
f-2:53:=00 g#2:56:-01 d-3:62:=00 f-3:65:-04 h-3:71:-03
e-2:52:-01 g-2:55:-01 c-3:60:-02 g-3:67:+02 c-4:72:+01
e-2:52:=00 f-2:53:-02 a-2:57:-03 c-3:60:-07 f-3:65:-07
d-2:50:-02 f-2:53:=00 a-2:57:+00 c-3:60:=00 f-3:65:=00
g-1:43:-07 d-2:50:-03 g-2:55:-02 h-2:59:-01 f-3:65:=00
c-2:48:+05 e-2:52:+02 g-2:55:+00 c-3:60:+01 e-3:64:-01
c-2:48:+00 g-2:55:+03 a#2:58:+03 c-3:60:=00 e-3:64:=00
f-1:41:-07 f-2:53:-02 a-2:57:-01 c-3:60:=00 e-3:64:=00
f#1:42:+01 c-2:48:-05 a-2:57:+00 c-3:60:=00 e-3:64:=00
g#1:44:+02 f-2:53:+03 h-2:59:+02 c-3:60:=00 d-3:62:-02
g-1:43:-01 f-2:53:=00 g-2:55:-04 h-2:59:-01 d-3:62:=00
g-1:43:-00 e-2:52:-01 g-2:55:+00 c-3:60:+01 e-3:64:+02
g-1:43:-00 d-2:52:-01 g-2:55:+00 c-3:60:+01 f-3:65:+01
g-1:43:-00 d-2:50:-02 g-2:55:+00 c-3:60:=00 f-3:65:+01
g-1:43:-00 d-2:50:-01 g-2:55:+00 h-2:59:-01 f-3:65:=00
g-1:43:-00 d#2:51:+01 a-2:57:+02 c-3:60:+01 f#3:66:+01
g-1:43:-00 e-2:52:+01 g-2:55:-02 c-3:60:=00 g-3:67:+01
g-1:43:-00 d-2:50:-02 g-2:55:+00 c-3:60:=00 f-3:65:-02
g-1:43:-00 d-2:50:+00 g-2:55:+00 h-2:59:-01 f-3:65:=00
c-1:36:-07 c-2:48:-02 g-2:55:+00 a#2:58:-01 e-3:64:-01

```

c-1:36:=00	c-2:48:=00	f-2:53:-02	a-2:57:-01	c-3:60:-04	f-3:65:+29	c-3:60:+12	a-2:57:+04
c-3:60:+03	a-2:57:-03	f-2:53:-12	a-2:57:-03	f-2:53:-04	d-2:50:-10	f-2:53:-04	d-2:50:-03
c-1:36:-21	h-1:47:-06	g-3:67:+17	h-3:71:+18	d-4:74:+24	f-4:77:+41	d-4:74:+27	h-3:71:+04
d-4:74:+03	h-3:71:-03	g-3:67:-10	h-3:71:-03	d-3:62:-09	f-3:65:-09	e-3:64:-07	d-3:62:-05

c-1:36:-35    c-2:48:-14    e-3:64:-01    g-3:67:+03    c-4:72:+10

*Histogram of raw-diff-5 (27 values, 197 notes)*

```
; -35 1 1 #
; -21 1 2 #
; -14 1 3 #
; -12 1 4 #
; -10 2 6 ##
; -09 2 8 ##
; -07 11 19 #####
; -06 1 20 #
; -05 2 22 ##
; -04 5 27 #####
; -03 11 38 #####
; -02 21 59 #####
; -01 22 81 #####
; =00 65 146 #####
; +01 15 161 #####
; +02 14 175 #####
; +03 5 180 #####
; +04 3 183 ###
; +05 6 189 #####
; +10 1 190 #
; +12 1 191 #
; +17 1 192 #
; +18 1 193 #
; +24 1 194 #
; +27 1 195 #
; +29 1 196 #
; +41 1 197 #
```

*Histogram of raw-diff-5 (27 values, 197 notes)*

```
; 1. =00 65 65 #####
; 2. -01 22 87 #####
; 3. -02 21 108 #####
; 4. +01 15 123 #####
; 5. +02 14 137 #####
; 6. -07 11 148 #####
; 7. -03 11 159 #####
; 8. +05 6 165 #####
; 9. -04 5 170 #####
; 10. +03 5 175 #####
; 11. +04 3 178 #####
; 12. -10 2 180 ##
; 13. -09 2 182 ##
; 14. -05 2 184 ##
; 15. -35 1 185 #
; 16. -21 1 186 #
; 17. -14 1 187 #
; 18. -12 1 188 #
; 19. -06 1 189 #
; 20. +41 1 190 #
; 21. +29 1 191 #
; 22. +27 1 192 #
; 23. +24 1 193 #
; 24. +18 1 194 #
; 25. +17 1 195 #
; 26. +12 1 196 #
; 27. +10 1 197 #
```

# *Histogram of raw-diff-5 (27 values, 197 notes)*

```

; 1. =00 65 65 #####
; 2. -01 22 87 #####
; 3. -02 21 108 #####
; 4. +01 15 123 #####
; 5. +02 14 137 #####
; 6. -07 11 148 #####
; 7. -03 11 159 #####
; 8. +05 6 165 #####
; 9. -04 5 170 #####
; 10. +03 5 175 #####
; 11. +04 3 178 #####
; 12. -10 2 180 ##
; 13. -09 2 182 ##
; 14. -05 2 184 ##
; 15. -35 1 185 #
; 16. -21 1 186 #
; 17. -14 1 187 #
; 18. -12 1 188 #
; 19. -06 1 189 #
; 20. +41 1 190 #
; 21. +29 1 191 #
; 22. +27 1 192 #
; 23. +24 1 193 #
; 24. +18 1 194 #
; 25. +17 1 195 #
; 26. +12 1 196 #
; 27. +10 1 197 #

```

Top-heavy:  
33% weight for  
top value (4%)

# Histogram of raw-diff-5 (27 values, 197 notes)

```

; 1. =00 65 65 #####
; 2. -01 22 87 #####
; 3. -02 21 108 #####
; 4. +01 15 123 #####
; 5. +02 14 137 #####
; 6. -07 11 148 #####
; 7. -03 11 159 #####
; 8. +05 6 165 #####
; 9. -04 5 170 #####
; 10. +03 5 175 #####


---


; 11. +04 3 178 #####
; 12. -10 2 180 #####
; 13. -09 2 182 #####
; 14. -05 2 184 #####
; 15. -35 1 185 #
; 16. -21 1 186 #
; 17. -14 1 187 #
; 18. -12 1 188 #
; 19. -06 1 189 #
; 20. +41 1 190 #
; 21. +29 1 191 #
; 22. +27 1 192 #
; 23. +24 1 193 #
; 24. +18 1 194 #
; 25. +17 1 195 #
; 26. +12 1 196 #
; 27. +10 1 197 #

```

Long tail:  
11% weight for  
63% of values

# *Compression*

## *Compression*

- Split data to top frequent and less frequent set of values around 80%..20% weight

## *Compression*

- Split data to top frequent and less frequent set of values around 80%..20% weight
- Store most frequent values in short-words

## *Compression*

- Split data to top frequent and less frequent set of values around 80%..20% weight
- **Store most frequent values in short-words**

```
ccc  ccc  ccc  ccc  ccc  ccc  ccc  ccc  ccc
```

## *Compression*

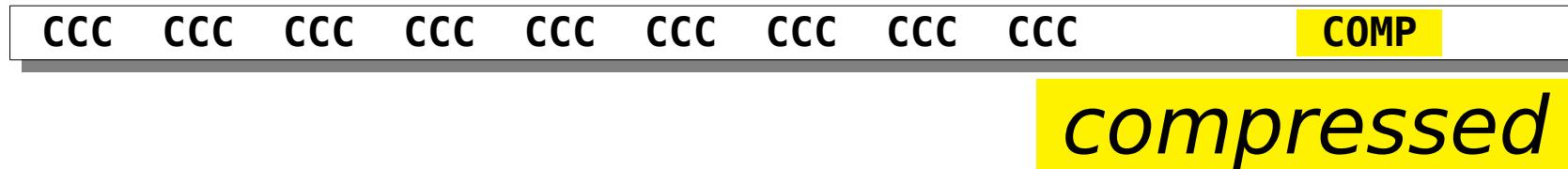
- Split data to top frequent and less frequent set of values around 80%..20% weight
- **Store most frequent values in short-words**

cccccccccc COMP

The diagram shows a sequence of ten 'ccc' abbreviations followed by a yellow box containing the word 'COMP'. This illustrates how frequent values are stored using short words, with a header indicating the compressed nature of the data.

# *Compression*

- Split data to top frequent and less frequent set of values around 80%..20% weight
- Store most frequent values in short-words



## Compression

- Split data to top frequent and less frequent set of values around 80%..20% weight
  - Store most frequent values in short-words
- 
- The diagram shows a sequence of ten short words, each consisting of three lowercase letters 'c', 'c', 'c'. These are followed by a single long word 'COMP' enclosed in a yellow rectangular box. The entire sequence is contained within a horizontal grey bar.
- Store special escape short-word followed by a long-word for less frequent values

## Compression

- Split data to top frequent and less frequent set of values around 80%..20% weight
- Store most frequent values in short-words

cccccccccc COMP

- Store special escape short-word followed by a long-word for less frequent values

sss+uuuuu sss+uuuuu sss+uuuuu ccc ccc sss+uuuu

# Compression

- Split data to top frequent and less frequent set of values around 80%..20% weight
- Store most frequent values in short-words

cccccccccc COMP

- Store special escape short-word followed by a long-word for less frequent values

sss+uuuuuu sss+uuuuuu sss+uuuuuu ccc ccc sss+uuuuuu UCOMP

# Compression

- Split data to top frequent and less frequent set of values around 80%..20% weight
- Store most frequent values in short-words

ccc ccc ccc ccc ccc ccc ccc ccc COMP

- Store special escape short-word followed by a long-word for less frequent values

sss+uuuuu sss+uuuuu sss+uuuuu ccc ccc sss+uuuu UCOMP

uncompressed

# Compression

- Split data to top frequent and less frequent set of values around 80%..20% weight
- Store most frequent values in short-words



- Store special escape short-word followed by a long-word for less frequent values



- Needs index tables

# Compression

- Split data to top frequent and less frequent set of values around 80%..20% weight
- Store most frequent values in short-words



- Store special escape short-word followed by a long-word for less frequent values



- Needs index tables
- First notes must be stored (have no diff)

# Compression

- Split data to top frequent and less frequent set of values around 80%..20% weight
- Store most frequent values in short-words



- Store special escape short-word followed by a long-word for less frequent values



- Needs index tables
- First notes must be stored (have no diff)

*Compression: raw-diff-5 @ 2*

<pre>; 1. =00 65 65 ########## ; 2. -01 22 87 ##### ; 3. -02 21 108 ##### ; 4. +01 15 123 ##### ; 5. +02 14 137 ##### ; 6. -07 11 148 ##### ; 7. -03 11 159 ##### ; 8. +05 6 165 ##### ; 9. -04 5 170 ##### ; 10. +03 5 175 ##### ; 11. +04 3 178 ##### ; 12. -10 2 180 ## ; 13. -09 2 182 ## ; 14. -05 2 184 ## ; 15. -35 1 185 # ; 16. -21 1 186 # ; 17. -14 1 187 # ; 18. -12 1 188 # ; 19. -06 1 189 # ; 20. +41 1 190 # ; 21. +29 1 191 # ; 22. +27 1 192 # ; 23. +24 1 193 # ; 24. +18 1 194 # ; 25. +17 1 195 # ; 26. +12 1 196 # ; 27. +10 1 197 #</pre>	<hr/> <p><b>55%: 2-bit</b></p> <hr/> <p><b>45%: 7-bit</b></p>
--	---

# *Compression: raw-diff-5 @ 3*

```

; 1. =00 65 65 ##########
; 2. -01 22 87 #####
; 3. -02 21 108 #####
; 4. +01 15 123 #####
; 5. +02 14 137 #####
; 6. -07 11 148 #####
; 7. -03 11 159 #####
; 8. +05 6 165 #####
; 9. -04 5 170 #####
; 10. +03 5 175 #####
; 11. +04 3 178 ##
; 12. -10 2 180 ##
; 13. -09 2 182 ##
; 14. -05 2 184 ##
; 15. -35 1 185 #
; 16. -21 1 186 #
; 17. -14 1 187 #
; 18. -12 1 188 #
; 19. -06 1 189 #
; 20. +41 1 190 #
; 21. +29 1 191 #
; 22. +27 1 192 #
; 23. +24 1 193 #
; 24. +18 1 194 #
; 25. +17 1 195 #
; 26. +12 1 196 #
; 27. +10 1 197 #

```

---

81%: 3-bit

---

19%: 7-bit

# *Compression: raw-diff-5 @ 4*

```

; 1. =00 65 65 ##########
; 2. -01 22 87 #####
; 3. -02 21 108 #####
; 4. +01 15 123 #####
; 5. +02 14 137 #####
; 6. -07 11 148 #####
; 7. -03 11 159 #####
; 8. +05 6 165 #####
; 9. -04 5 170 #####
; 10. +03 5 175 #####
; 11. +04 3 178 ###
; 12. -10 2 180 ##
; 13. -09 2 182 ##
; 14. -05 2 184 ##
; 15. -35 1 185 #
94%: 4-bit


---


; 16. -21 1 186 #
; 17. -14 1 187 #
; 18. -12 1 188 #
; 19. -06 1 189 #
; 20. +41 1 190 #
; 21. +29 1 191 #
; 22. +27 1 192 #
; 23. +24 1 193 #
; 24. +18 1 194 #
; 25. +17 1 195 #
; 26. +12 1 196 #
; 27. +10 1 197 #
6%: 8-bit

```

## *Compression: nutab*

No Uncompressed Table

## *Compression: nutab*

No Uncompressed Table:

- High number of values – large table

## *Compression: nutab*

### No Uncompressed Table:

- High number of values – large table
- Low utilization of the table (usually 1 note)

## *Compression: nutab*

### No Uncompressed Table:

- High number of values – large table
- Low utilization of the table (usually 1 note)
- Range of minimum and maximum value is not much bigger than table size

## *Compression: nutab*

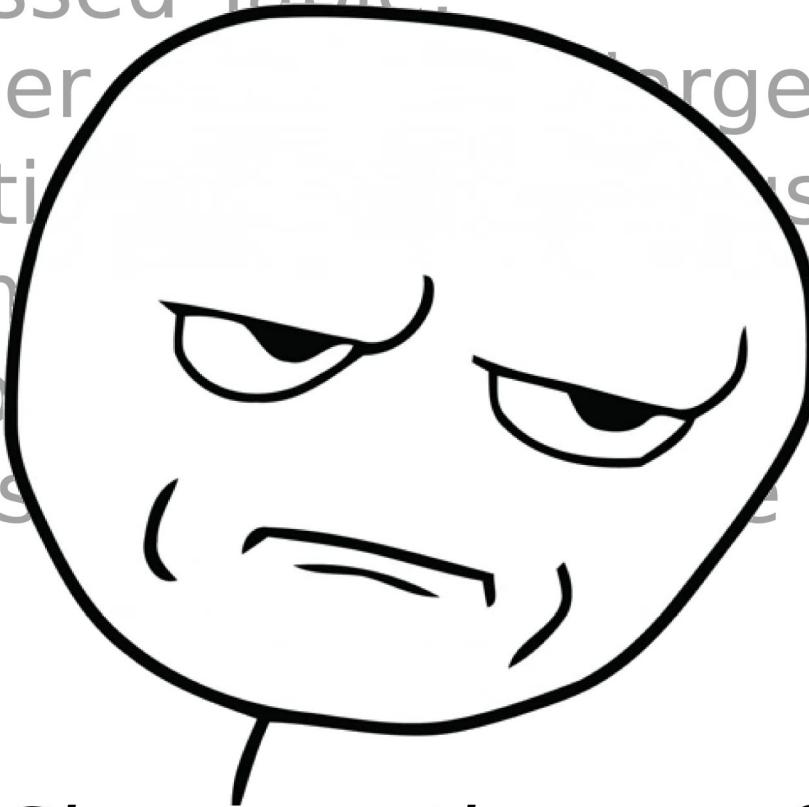
### No Uncompressed Table:

- High number of values – large table
- Low utilization of the table (usually 1 note)
- Range of minimum and maximum value is not much bigger than table size
- **Table needs some extra code**

## *Compression: nutab*

### No Uncompressed Table:

- High number of entries (large table)
- Low utilization (usually 1 note)
- Range of memory addresses (num
- value is not contiguous)
- Table needs to be indexed



*Show me the proof!  
I need evidence*

# *Compression: nutab*

<i>split</i>	<i>table</i>	<i>value range</i>	<i>bits per item</i>	<i>storage + table</i>	<i>ucomp total</i>
raw-diff-5 @ 2	yes	24	5	77 + 24	101
	nutab	76	7	100	100
raw-diff-5 @ 3	yes	20	5	38 + 20	58
	nutab	76	7	47	47
raw-diff-5 @ 4	yes	12	4	12 + 12	24
	nutab	62	6	15	15

# Compression: nutab

<i>split</i>	<i>table</i>	<i>value range</i>	<i>bits per item</i>	<i>storage + table</i>	<i>ucomp total</i>
raw-diff-5 @ 2	yes	24	5	77 + 24	101
	nutab	76	7	100	100
raw-diff-5 @ 3	yes	20	5	38 + 20	58
	nutab	76	7	47	47
raw-diff-5 @ 4	yes	12	4	12 + 12	24
	nutab	62	6	15	

Okay.



## *Compression: nctab*

No Compressed Table

## *Compression: nctab*

No Compressed Table:

- Top values are almost continuous

## *Compression: nctab*

### No Compressed Table:

- Top values are almost continuous
- Swap values for continuous range

## *Compression: nctab*

### No Compressed Table:

- Top values are almost continuous
- Swap values for continuous range
- Small compromise for eliminating  
Compressed Table

*Compression: nctab*

```

; 1. =00 65 65 ##########
; 2. -01 22 87 #####
; 3. -02 21 108 #####
; 4. +01 15 123 #####
; 5. +02 14 137 #####
; 6. -07 11 148 #####
; 7. -03 11 159 #####


---


; 8. +05 6 165 #####
; 9. -04 5 170 #####
; 10. +03 5 175 #####
; 11. +04 3 178 ###
; 12. -10 2 180 ##
; 13. -09 2 182 ##
; 14. -05 2 184 ##
; 15. -35 1 185 #

```

Swap -07: 11 notes  
With +03: 5 notes

*Compression: nctab*

```

; 1. =00 65 65 ##########
; 2. -01 22 87 #####
; 3. -02 21 108 #####
; 4. +01 15 123 #####
; 5. +02 14 137 #####
; 10. +03 5 175 #####
; 7. -03 11 159 #####
; 8. +05 6 165 #####
; 9. -04 5 170 #####
; 6. -07 11 148 #####
; 11. +04 3 178 ###
; 12. -10 2 180 ##
; 13. -09 2 182 ##
; 14. -05 2 184 ##
; 15. -35 1 185 #

```

Swap      -07: 11 notes  
 With      +03: 5 notes

---

*Compression: nctab*

```

; 1. =00 65 65 ##########
; 2. -01 22 87 #####
; 3. -02 21 108 #####
; 4. +01 15 123 #####
; 5. +02 14 137 #####
; 10. +03 5 175 #####
; 7. -03 11 159 #####
; 8. +05 6 165 #####
; 9. -04 5 170 #####
; 6. -07 11 148 #####
; 11. +04 3 178 ###
; 12. -10 2 180 ##
; 13. -09 2 182 ##
; 14. -05 2 184 ##
; 15. -35 1 185 #

```

Top notes: -03..+03

---

# *Compression: select method*

# *Compression: select method*

<i>note</i>	<i>diff</i>	<i>compressed word size</i>	<i>compressed table</i>	<i>uncompressed table</i>
raw-mapped-	diff-1 diff-2 diff-3 diff-4 diff-5 diff-6 diff-7 diff-8 diff-mixed/1/5	@ 2 @ 3 @ 4 @ 6	yes nctab	yes nutab

$$2 * 9 * 4 * 2 * 2 = \mathbf{288} \text{ variations}$$

## *Compression: select method*

Select compression method:

- Create estimation for all the 288 variations

## *Compression: select method*

### Select compression method:

- Create estimation for all the 288 variations
- Accurate estimation of data and table sizes

## *Compression: select method*

### Select compression method:

- Create estimation for all the 288 variations
- Accurate estimation of data and table sizes
- Can't calculate required code size

## *Compression: select method*

Select compression method:

- Create estimation for all the 288 variations
- Accurate estimation of data and table sizes
- Can't calculate required code size



*Challenge accepted.*

# *Compression: compare methods*

## *Compression: compare methods*

**; ----- estimation for raw-diff-5 @ 3 nutab -----**

## *Compression: compare methods*

**; ----- estimation for raw-diff-5 @ 3 nutab -----**

This is the estimation for:

- diff from 5 notes behind

## *Compression: compare methods*

**; ----- estimation for raw-diff-5 @ 3 nutab -----**

This is the estimation for:

- diff from 5 notes behind
- compressed word size is 3-bit

## *Compression: compare methods*

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
```

This is the estimation for:

- diff from 5 notes behind
- compressed word size is 3-bit
- use table for compressed values  
(not mentioned)

## *Compression: compare methods*

**; ----- estimation for raw-diff-5 @ 3 nutab -----**

This is the estimation for:

- diff from 5 notes behind
- compressed word size is 3-bit
- use table for compressed values
- no table for uncompressed values

## *Compression: compare methods*

```
; ----- estimation for raw-diff-5 @ 3 nutab -----  
;
```

## *Compression: compare methods*

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:      7.c    20.u    27.t
```

## *Compression: compare methods*

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:    7.c    20.u    27.t
```

Number of different note values:

## *Compression: compare methods*

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:    7.c    20.u    27.t
```

Number of different note values:

- 7 compressed (table index)

## *Compression: compare methods*

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:      7.c    20.u    27.t
```

Number of different note values:

- 7 compressed (table index)
- 20 uncompressed (nutab: raw pitch range)

## *Compression: compare methods*

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:    7.c    20.u    27.t
```

Number of different note values:

- 7 compressed (table index)
- 20 uncompressed (nutab: raw pitch range)
- 27 total

## *Compression: compare methods*

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:      7.c    20.u   27.t
; note count: 159.c    38.u   197.t
```

## *Compression: compare methods*

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:      7.c    20.u   27.t
; note count: 159.c    38.u   197.t
```

Note count (no. of occurrences):

## *Compression: compare methods*

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:      7.c    20.u   27.t
; note count: 159.c    38.u   197.t
```

Note count (no. of occurrences):

- 159 compressed

## *Compression: compare methods*

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:      7.c    20.u   27.t
; note count: 159.c  38.u  197.t
```

Note count (no. of occurrences):

- 159 compressed
- 38 uncompressed

## *Compression: compare methods*

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:      7.c    20.u   27.t
; note count: 159.c    38.u  197.t
```

Note count (no. of occurrences):

- 159 compressed
- 38 uncompressed
- 197 total

## *Compression: compare methods*

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:      7.c    20.u    27.t
; note count:   159.c   38.u   197.t
; note bits:     3.c    10.u
```

## *Compression: compare methods*

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:      7.c    20.u   27.t
; note count: 159.c    38.u   197.t
; note bits:     3.c    10.u
```

Storage needed by one note:

## *Compression: compare methods*

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:      7.c    20.u   27.t
; note count: 159.c    38.u   197.t
; note bits:     3.c    10.u
```

Storage needed by one note:

- 3 bits for compressed

## *Compression: compare methods*

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:      7.c    20.u    27.t
; note count:   159.c   38.u   197.t
; note bits:     3.c    10.u
```

Storage needed by one note:

- 3 bits for compressed
- 10 bits for uncompressed (spec: 3 + data: 7)

## *Compression: compare methods*

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:      7.c    20.u    27.t
; note count:   159.c   38.u   197.t
; note bits:     3.c    10.u
; storage:       59.c   47.u   107.t
```

## *Compression: compare methods*

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:      7.c    20.u   27.t
; note count: 159.c    38.u   197.t
; note bits:     3.c    10.u
; storage:       59.c   47.u  107.t
```

Storage needed for song data:

## Compression: compare methods

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:      7.c    20.u   27.t
; note count: 159.c    38.u   197.t
; note bits:     3.c    10.u
; storage:       59.c    47.u   107.t
```

Storage needed for song data:

- 59 bytes for compressed

## *Compression: compare methods*

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:      7.c    20.u   27.t
; note count: 159.c    38.u   197.t
; note bits:     3.c    10.u
; storage:       59.c   47.u   107.t
```

Storage needed for song data:

- 59 bytes for compressed
- 47 bytes for uncompressed

## Compression: compare methods

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:      7.c    20.u   27.t
; note count: 159.c    38.u   197.t
; note bits:     3.c    10.u
; storage:       59.c   47.u  107.t
```

Storage needed for song data:

- 59 bytes for compressed
- 47 bytes for uncompressed
- 107 bytes total

# *Compression: compare methods*

<b>; note num:</b>	<b>7.c</b>	<b>20.u</b>	<b>27.t</b>
<b>; note count:</b>	<b>159.c</b>	<b>38.u</b>	<b>197.t</b>
<b>; note bits:</b>	<b>3.c</b>	<b>10.u</b>	
<b>; storage:</b>	<b>59.c</b>	<b>47.u</b>	<b>107.t</b>
<b>; table:</b>	<b>7.c</b>	<b>0.u</b>	<b>7.t</b>

## *Compression: compare methods*

<b>; note num:</b>	<b>7.c</b>	<b>20.u</b>	<b>27.t</b>
<b>; note count:</b>	<b>159.c</b>	<b>38.u</b>	<b>197.t</b>
<b>; note bits:</b>	<b>3.c</b>	<b>10.u</b>	
<b>; storage:</b>	<b>59.c</b>	<b>47.u</b>	<b>107.t</b>
<b>; table:</b>	<b>7.c</b>	<b>0.u</b>	<b>7.t</b>

Storage needed for tables:

## *Compression: compare methods*

<b>; note num:</b>	<b>7.c</b>	<b>20.u</b>	<b>27.t</b>
<b>; note count:</b>	<b>159.c</b>	<b>38.u</b>	<b>197.t</b>
<b>; note bits:</b>	<b>3.c</b>	<b>10.u</b>	
<b>; storage:</b>	<b>59.c</b>	<b>47.u</b>	<b>107.t</b>
<b>; table:</b>	<b>7.c</b>	<b>0.u</b>	<b>7.t</b>

Storage needed for tables:

- 7 bytes for compressed

## Compression: compare methods

; note num:	7.c	20.u	27.t
; note count:	159.c	38.u	197.t
; note bits:	3.c	10.u	
<b>; storage:</b>	<b>59.c</b>	<b>47.u</b>	<b>107.t</b>
<b>; table:</b>	<b>7.c</b>	<b>0.u</b>	<b>7.t</b>

Storage needed for tables:

- 7 bytes for compressed
- none for uncompressed (nutab)

## Compression: compare methods

<b>; note num:</b>	<b>7.c</b>	<b>20.u</b>	<b>27.t</b>
<b>; note count:</b>	<b>159.c</b>	<b>38.u</b>	<b>197.t</b>
<b>; note bits:</b>	<b>3.c</b>	<b>10.u</b>	
<b>; storage:</b>	<b>59.c</b>	<b>47.u</b>	<b>107.t</b>
<b>; table:</b>	<b>7.c</b>	<b>0.u</b>	<b>7.t</b>

Storage needed for tables:

- 7 bytes for compressed
- none for uncompressed (nutab)
- 7 bytes total

## Compression: compare methods

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:      7.c    20.u    27.t
; note count:   159.c   38.u   197.t
; note bits:     3.c    10.u
; storage:       59.c   47.u   107.t
; table:         7.c    0.u    7.t
; total bytes (storage + leading + table): 120
```

Total storage required:

*Compression: compare methods*

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:      7.c    20.u    27.t
; note count:   159.c   38.u   197.t
; note bits:     3.c    10.u
; storage:       59.c   47.u   107.t
; table:         7.c    0.u    7.t
; total bytes (storage + leading + table): 120
```

Total storage required: 107

*Compression: compare methods*

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:      7.c    20.u    27.t
; note count:   159.c   38.u   197.t
; note bits:     3.c    10.u
; storage:       59.c   47.u   107.t
; table:         7.c     0.u     7.t
; total bytes (storage + leading + table): 120
```

Total storage required: 107 + 5

## Compression: compare methods

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:      7.c    20.u    27.t
; note count:   159.c   38.u   197.t
; note bits:     3.c    10.u
; storage:       59.c   47.u   107.t
; table:         7.c    0.u    7.t
; total bytes (storage + leading + table): 120
```

Total storage required: 107 + 5 + 7

## Compression: compare methods

```
; ----- estimation for raw-diff-5 @ 3 nutab -----
;
; note num:      7.c    20.u    27.t
; note count:   159.c   38.u   197.t
; note bits:     3.c    10.u
; storage:       59.c   47.u   107.t
; table:         7.c     0.u     7.t
; total bytes (storage + leading + table): 120
```

Total storage required:  $107 + 5 + 7 = \text{120 bytes}$

## *Compression: compare methods*

We have histogram,  
estimation and  
data generator  
for all the  
**288**  
variations

# *Compression: compare methods*

The winner is...



# Compression: compare methods

;	<b>raw-diff-mixed/1/5</b> @ 4 nctab nutab	=	114
;	<b>raw-diff-mixed/1/5</b> @ 3 nutab	=	117
;	<b>raw-diff-5</b> @ 3 nctab nutab	=	118
;	<b>raw-diff-5</b> @ 4 nctab nutab	=	118
;	<b>raw-diff-5</b> @ 3 nutab	=	120
;	<b>raw-diff-mixed/1/5</b> @ 4 nctab	=	122
;	<b>raw-diff-mixed/1/5</b> @ 4 nutab	=	124
;	<b>raw-diff-1</b> @ 3 nutab	=	125
;	<b>raw-diff-5</b> @ 4 nctab	=	126
;	<b>raw-diff-1</b> @ 2 nutab	=	127
;	<b>raw-diff-5</b> @ 3 nctab	=	127
;	<b>raw-diff-mixed/1/5</b> @ 2 nutab	=	127
;	<b>raw-diff-5</b> @ 4 nutab	=	128

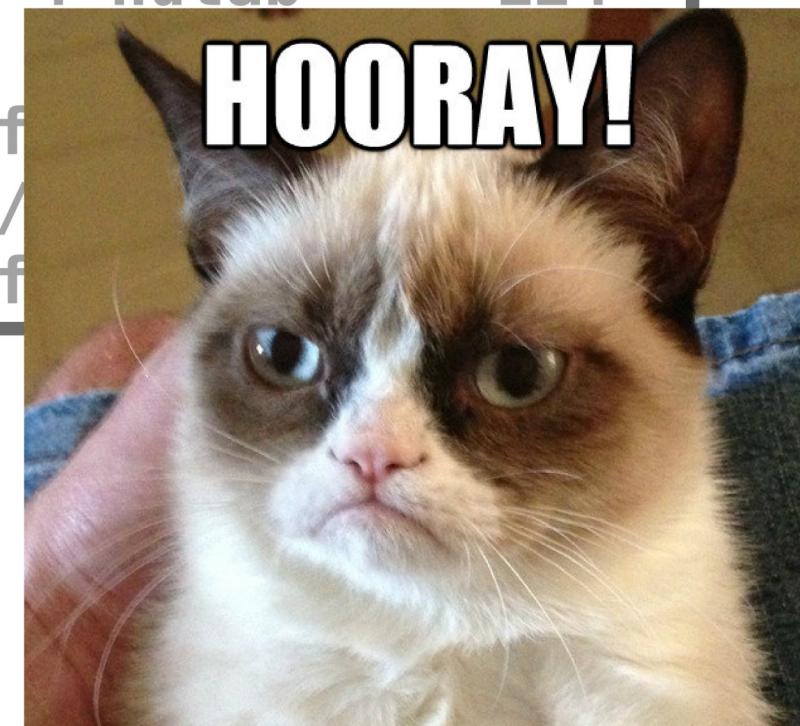


# Compression: compare methods

```
;raw-diff-mixed/1/5 @ 4 nctab nutab = 114
;raw-diff-mixed/1/5 @ 3 nutab = 117
;raw-diff-5 @ 3 nctab nutab = 118
;raw-diff-5 @ 4 nctab nutab = 118
;raw-diff-5 @ 3 nutab = 120
;raw-diff-mixed/1/5 @ 4 nctab = 122
;raw-diff-mixed/1/5 @ 4 nutab = 124
( . . . )
;mapped-diff-5 @ 3 = 160
;mapped-diff-mixed/1/5 @ 4 = 160
;raw-diff-1 @ 5 = 161
```

# Compression: compare methods

```
;  
; raw-diff-mixed/1/5 @ 4 nctab nutab = 114  
;  
; raw-diff-mixed/1/5 @ 3 nutab = 117  
;  
raw-diff-5 @ 3 nctab nutab = 118  
;  
raw-diff-5 @ 4 nctab nutab = 118  
;  
;  
; raw-diff-5 @ 3 nutab = 120  
;  
raw-diff-mixed/1/5 @ 4 nctab = 122  
raw-diff-mixed/1/5 @ 4 nutab = 124  
(...)  
mapped-dif  
mapped-diff-mixed/  
raw-dif
```



*Creating data (raw-diff-5 @ 3 nctab nutab)*

*Creating data (raw-diff-5 @ 3 nctab nutab)*

```
; 1. =00 65 65 #####  
; 2. -01 22 87 #####  
; 3. -02 21 108 #####  
; 4. +01 15 123 #####  
; 5. +02 14 137 #####  
; 10. +03 5 175 #####  
; 7. -03 11 159 #####  
; 8. +05 6 165 #####  
; 9. -04 5 170 #####  
; 6. -07 11 148 #####  
; 11. +04 3 178 ###  
; 12. -10 2 180 ##  
; 13. -09 2 182 ##  
; 14. -05 2 184 ##  
; 15. -35 1 185 #
```

Top notes: -03..+03



*remember...*

## *Creating data (raw-diff-5 @ 3 nctab nutab)*

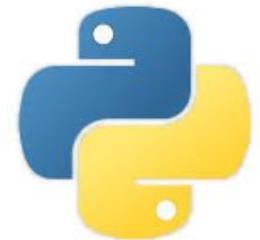
```
cSub = 4
uSub = 42
spec = 0      # -4 + cSub

for i in range(0,len(self.notes)):
    note = self.notes[i]
    diff = note.get("raw-diff-5")

    if diff in (-3,-2,-1,0,1,2,3):
        self.renderDataBits(3,diff + cSub)
    else:
        self.renderDataBits(3,spec)
        self.renderDataBits(7,diff + uSub)
```

## *Creating data (raw-diff-5 @ 3 nctab nutab)*

```
cSub = 4  
uSub = 42  
spec = 0      # -4 + cSub
```



```
for i in range(0,len(self.notes)):  
    note = self.notes[i]  
    diff = note.get("raw-diff-5")  
  
    if diff in (-3,-2,-1,0,1,2,3):  
        self.renderDataBits(3,diff + cSub)  
    else:  
        self.renderDataBits(3,spec)  
        self.renderDataBits(7,diff + uSub)
```

## *Creating data (raw-diff-5 @ 3 nctab nutab)*

compressed

```
cSub = 4
uSub = 42
spec = 0      # -4 + cSub

for i in range(0,len(self.notes)):
    note = self.notes[i]
    diff = note.get("raw-diff-5")

    if diff in (-3,-2,-1,0,1,2,3):
        self.renderDataBits(3,diff + cSub)
    else:
        self.renderDataBits(3,spec)
        self.renderDataBits(7,diff + uSub)
```

## *Creating data (raw-diff-5 @ 3 nctab nutab)*

uncompressed

```
cSub = 4
uSub = 42
spec = 0      # -4 + cSub

for i in range(0,len(self.notes)):
    note = self.notes[i]
    diff = note.get("raw-diff-5")

    if diff in (-3,-2,-1,0,1,2,3):
        self.renderDataBits(3,diff + cSub)
    else:
        self.renderDataBits(3,spec)
        self.renderDataBits(7,diff + uSub)
```

# *Creating data (raw-diff-5 @ 3 nctab nutab)*

## The final data

```
; value to subtract from compressed data  
DATA_CSUB = 4  
  
; value to subtract from uncompressed data  
DATA_USUB = 42  
  
data_notes: ; bit packed note data  
  
db $92,$49,$16,$d5,$c5,$25,$d1,$39  
db $30,$5c,$17,$c4,$42,$30,$8d,$ca  
db $17,$85,$f1,$10,$8c,$23,$52,$48  
db $11,$94,$e0,$5f,$a5,$71,$e9,$93  
db $5a,$c7,$02,$62,$da,$d6,$22,$11  
db $84,$6a,$49,$02,$32,$9c,$0b,$f4  
db $ae,$7f,$20,$46,$9c,$94,$25,$92  
db $60,$bf,$44,$e0,$4c,$e4,$72,$e8  
db $a4,$b2,$47,$25,$d6,$ca,$a5,$8a  
db $45,$24,$70,$23,$51,$b9,$13,$09  
db $84,$70,$d8,$2e,$e4,$1e,$21,$30  
db $40,$13,$10,$54,$24,$0e,$c3,$c1  
db $08,$53,$11,$42,$ee,$42,$02,$10  
db $84,$21,$18,$4a,$03,$83,$8f,$86  
db $95
```

# *Creating data (raw-diff-5 @ 3 nctab nutab)*

The final\* data

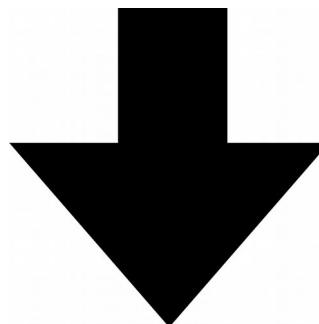
```
; value to subtract from compressed data  
DATA_CSUB = 4  
  
; value to subtract from uncompressed data  
DATA_USUB = 42  
  
data_notes: ; bit packed note data  
  
db $92,$49,$16,$d5,$c5,$25,$d1,$39  
db $30,$5c,$17,$c4,$42,$30,$8d,$ca  
db $17,$85,$f1,$10,$8c,$23,$52,$48  
db $11,$94,$e0,$5f,$a5,$71,$e9,$93  
db $5a,$c7,$02,$62,$da,$d6,$22,$11  
db $84,$6a,$49,$02,$32,$9c,$0b,$f4  
db $ae,$7f,$20,$46,$9c,$94,$25,$92  
db $60,$bf,$44,$e0,$4c,$e4,$72,$e8  
db $a4,$b2,$47,$25,$d6,$ca,$a5,$8a  
db $45,$24,$70,$23,$51,$b9,$13,$09  
db $84,$70,$d8,$2e,$e4,$1e,$21,$30  
db $40,$13,$10,$54,$24,$0e,$c3,$c1  
db $08,$53,$11,$42,$ee,$42,$02,$10  
db $84,$21,$18,$4a,$03,$83,$8f,$86  
db $95
```

\* not

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

```
db $92,$49,$16,$d5,$c5,$25,$d1,$39  
db $30,$5c,$17,$c4,$42,$30,$8d,$ca
```



c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72: -02	e-4:76: -01

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

<b>c-3:60:/00</b>	<b>e-3:64:/00</b>	<b>g-3:67:/00</b>	<b>c-4:72:/00</b>	<b>e-4:76:/00</b>
<b>c-3:60:=00</b>	<b>d-3:62:-02</b>	<b>a-3:69:+02</b>	<b>d-4:74:+02</b>	<b>f-4:77:+01</b>
<b>h-2:59:-01</b>	<b>d-3:62:=00</b>	<b>g-3:67:-02</b>	<b>d-4:74:=00</b>	<b>f-4:77:=00</b>
<b>c-3:60:+01</b>	<b>e-3:64:+02</b>	<b>g-3:67:=00</b>	<b>c-4:72:-02</b>	<b>e-4:76:-01</b>

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72: -02	e-4:76: -01

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

#####

4

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

##### #####

4 4

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

##### ##### #####

4 4 4

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

##### ##### ##### #####

4 4 4 4

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

##### ##### ##### ##### #####

4 4 4 4 4

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001

%00010110 %11010101

##### ##### ##### ##### #####

4 4 4 4 4

CSUB=4

USUB=42

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

# *Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001

%00010110 %11010101

# ##### # ##### # ##### # ##### # #####

4	4	4	4	4
0	0	0	0	0

CSUB=4

USUB=42

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

# *Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001

%00010110 %11010101

# ##### # ##### # ##### # ##### # #####

4	4	4	4	4
0	0	0	0	0

CSUB=4

USUB=42

c-3:60:/00

e-3:64:/00

g-3:67:/00

c-4:72:/00

e-4:76:/00

c-3:60:=00

d-3:62:-02

a-3:69:+02

d-4:74:+02

f-4:77:+01

h-2:59:-01

d-3:62:=00

g-3:67:-02

d-4:74:=00

f-4:77:=00

c-3:60:+01

e-3:64:+02

g-3:67:=00

c-4:72:-02

e-4:76:-01

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

##### ##### ##### ##### ##### ##### ##### #####

4	4	4	4	4
0	0	0	0	0

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

##### ##### ##### ##### ##### #####

4 4 4 4 4 4  
0 0 0 0 0

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

# *Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

##### ##### ##### ##### ##### ##### ##### #####

4	4	4	4	4	4
0	0	0	0	0	0

CSUB=4  
USUB=42

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

# *Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

# ##### # ##### # ##### # ##### # #####

4	4	4	4	4	4
0	0	0	0	0	0

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72: -02	e-4:76: -01

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

##### ##### ##### ##### ##### ##### ##### #####

4 4 4 4 4 4

0 0 0 0 0 0

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

##### ##### ##### ##### ##### ##### ##### #####

4	4	4	4	4	4	2
0	0	0	0	0	0	

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

# *Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

# ##### # ##### # ##### # ##### # ##### # #####

4	4	4	4	4	4	2
0	0	0	0	0	0	-2

CSUB=4

USUB=42

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

# *Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

# ##### # ##### # ##### # ##### # ##### # #####

4	4	4	4	4	4	2
0	0	0	0	0	0	-2

CSUB=4

USUB=42

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72: -02	e-4:76: -01

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

##### ##### ##### ##### ##### ##### ##### #####

4	4	4	4	4	4	2
0	0	0	0	0	0	-2

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

##### ##### ##### ##### ##### ##### ##### ##### #####

4	4	4	4	4	4	2
0	0	0	0	0	0	-2

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

##### ##### ##### ##### ##### ##### ##### ##### #####

4 4 4 4 4 4 2 6

0 0 0 0 0 0 -2

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

# *Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110

%11010101

# ##### # ##### # ##### # ##### # ##### # ##### # #####

4	4	4	4	4	4	2	6
0	0	0	0	0	0	-2	2

CSUB=4  
USUB=42

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

# *Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

# ##### # ##### # ##### # ##### # ##### # ##### # #####

4	4	4	4	4	4	2	6
0	0	0	0	0	0	-2	2

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

# ##### # ##### # ##### # ##### # ##### # ##### # ##### # #####

4	4	4	4	4	4	2	6	6
0	0	0	0	0	0	-2	2	2

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

# *Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

# ##### # ##### # ##### # ##### # ##### # ##### # #####

# #####

4

4

4

4

4

4

2

6

0

0

0

0

0

0

-2

2

6  
2

c-3:60:/00

e-3:64:/00

g-3:67:/00

c-4:72:/00

e-4:76:/00

c-3:60:=00

d-3:62:-02

a-3:69:+02

d-4:74:+02

f-4:77:+01

h-2:59:-01

d-3:62:=00

g-3:67:-02

d-4:74:=00

f-4:77:=00

c-3:60:+01

e-3:64:+02

g-3:67:=00

c-4:72:-02

e-4:76:-01

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

# ##### # ##### # ##### # ##### # ##### # ##### # ##### # #####

4	4	4	4	4	4	2	6	6	5
0	0	0	0	0	0	-2	2	2	1

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

*Decoding data (raw-diff-5 @ 3 nctab nutab)*

\$92

\$49

\$16

\$d5

84218421

84218421

84218421

84218421

%10010010 %01001001 %00010110 %11010101

# ##### # ##### # ##### # ##### # ##### # ##### # ##### # ##### # #####

4	4	4	4	4	4	4	2	6	6	5
0	0	0	0	0	0	0	-2	2	2	1

c-3:60:/00	e-3:64:/00	g-3:67:/00	c-4:72:/00	e-4:76:/00
c-3:60:=00	d-3:62:-02	a-3:69:+02	d-4:74:+02	f-4:77:+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

## Decoding data (raw-diff-5 @ 3 nctab nutab)

\$92

\$49

\$16

\$d5

8421				421
%1001				101
#####				###
4				
0				
c-3:0				/00
c-3:0				+01
h-2:59:-01	d-3:62:=00	g-3:67:-02	d-4:74:=00	f-4:77:=00
c-3:60:+01	e-3:64:+02	g-3:67:=00	c-4:72:-02	e-4:76:-01

Sorry, no uncompressed example...

## Decoding data (raw-diff-5 @ 3 nctab nutab)

\$92

\$49

\$16

\$d5

8421  
%1001  
#####  
421  
101  
###  
Sorry, no uncompressed example...  
Because it's so effective!

c-3:0

c-3:0

h-2:59:-01

d-3:62:=00

g-3:67:-02

d-4:74:=00

f-4:77:=00

c-3:60:+01

e-3:64:+02

g-3:67:=00

c-4:72:-02

e-4:76:-01

/00

+01

## Decoding data (raw-diff-5 @ 3 nctab nutab)

\$92

\$49

\$16

\$d5

8421

%1001

#####

4

0

Sorry, no uncompressed example...  
Because it's so effective!

421

101

###

c-3:0

c-3:0

h-2:59:-01

d-3:62:=00

g-3:67:-02

d-4:74

c-3:60:+01

e-3:64:+02

g-3:67:=00

c-4:72



# III. Code

### III. Code



Assembly code ahead!

*Clean Code Decoder (raw-diff-5 @ 3 nctab nutab)*

*Clean Code Decoder (raw-diff-5 @ 3 nctab nutab)*

How to decompress *diff* value:

## *Clean Code Decoder (raw-diff-5 @ 3 nctab nutab)*

How to decompress *diff* value:

- not optimized yet, no sizecoding tricks

## *Clean Code Decoder (*raw-diff-5 @ 3 nctab nutab*)*

How to decompress *diff* value:

- not optimized yet, no sizecoding tricks
- special code for *raw-diff-5 @ 3 nctab nutab*

# Clean Code Decoder (raw-diff-5 @ 3 nctab nutab)

```

load_play_note:
    mov bl,DATA_CSUB ; DATA_CSUB = 4
    mov ax,$2000       ; AL:=0, AH:=%xx10'0000: 3 SHL from zero
@next_bit:
    or ah,ah
    jnz @read_bit
;word_read:
    or al,al          ; check for %000 special value
    jnz @adjust_word
;load_uncompressed:
    mov bl,DATA_USUB ; 42, also a good value for bit counter
    mov ah,bl          ; %xxxx'xx10: 7 SHL from zero
    jmp @next_bit
@read_bit:
    or cl,cl
    jnz @shift_latch
;load_latch:
    inc cx             ; INC CX for CL:=1, %xxxx'xxx1: 8 SHL from zero
    mov ch,[bp]
    inc bp
@shift_latch:
    sal ax,1
    sal cx,1
    adc al,0
    jmp @next_bit
@adjust_word:
    sub al,bl

```

# Clean Code Decoder (raw-diff-5 @ 3 nctab nutab)

```

load_play_note:
    mov bl,DATA_CSUB ; DATA_CSUB = 4
    mov ax,$2000        ; AI = 0 AH = %xx10'0000 - 3 SHL from zero
@next_bit:
    or ah,ah
    jnz @read
;word_read:
    or al,al
    jnz @adjust
;load_uncompre
    mov bl,DA
    mov ah,bl
    jmp @next_bit
@read_bit:
    or cl,cl
    jnz @shift_latch
;load_latch:
    inc cx             ; INC CX for CL:=1, %xxxx'xxx1: 8 SHL from zero
    mov ch,[bp]
    inc bp
@shift_latch:
    sal ax,1
    sal cx,1
    adc al,0
    jmp @next_bit
@adjust_word:
    sub al,bl

```

word:	0	1	2	3	4	5	6	7
diff:	SPEC	-03	-02	-01	=00	+01	+02	+03

word - DATA\_CSUB = diff  
BL

# Clean Code Decoder (raw-diff-5 @ 3 nctab nutab)

```
; 1. =00 65 65 #####  
; 2. -01 22 87 #####  
; 3. -02 21 108 #####  
; 4. +01 15 123 #####  
; 5. +02 14 137 #####  
; 10. +03 5 175 #####  
; 7. -03 11 159 #####
```

Top notes: -03..+03

```
; 8. +05 6 165 #####  
; 9. -04 5 170 #####  
; 6. -07 11 148 #####  
; 11. +04 3 178 ###  
; 12. -10 2 180 ##  
; 13. -09 2 182 ##  
; 14. -05 2 184 ##  
; 15. -35 1 185 #
```



still remember...

# Clean Code Decoder (raw-diff-5 @ 3 nctab nutab)

```

load_play_note:
    mov bl,DATA_CSUB ; DATA_CSUB = 4
    mov ax,$2000          ; AI = 0 AH = %xy10'0000 - 3 SHL from zero
@next_bit:
    or ah,ah
    jnz @read
;word_read:
    or al,al
    jnz @adjust
;load_uncompre
    mov bl,DA
    mov ah,bl
    jmp @next_bit
@read_bit:
    or cl,cl
    jnz @shift_latch
;load_latch:
    inc cx              ; INC CX for CL:=1, %xxxx'xxx1: 8 SHL from zero
    mov ch,[bp]
    inc bp
@shift_latch:
    sal ax,1
    sal cx,1
    adc al,0
    jmp @next_bit
@adjust_word:
    sub al,bl

```

word:	0	1	2	3	4	5	6	7
diff:	SPEC	-03	-02	-01	=00	+01	+02	+03

word - DATA\_CSUB = diff  
BL

# Clean Code Decoder (raw-diff-5 @ 3 nctab nutab)

```

load_play_note:
    mov bl,DATA_CSUB ; DATA_CSUB = 4
    mov ax,$2000 ; AL:=0, AH:=%xx10'0000: 3 SHL from zero
@next_bit:
    or ah,ah
    jnz @read_bit
;word_read:
    or al,al
    jnz @adjust_word
;load_uncompressed:
    mov bl,DATA_USUB ; 42, also a good value for bit counter
    mov ah,bl          ; %xxxx'xx10: 7 SHL from zero
    jmp @next_bit
@read_bit:
    or cl,cl
    jnz @shift_latch
;load_latch:
    inc cx             ; INC CX for CL:=1, %xxxx'xxx1: 8 SHL from zero
    mov ch,[bp]
    inc bp
@shift_latch:
    sal ax,1
    sal cx,1
    adc al,0
    jmp @next_bit
@adjust_word:
    sub al,bl

```

**AL:** result diff value, initialize  
**AH:** shift counter, shift until zero  
**SHL AX:** shift value and counter

# Clean Code Decoder (raw-diff-5 @ 3 nctab nutab)

```

load_play_note:
    mov bl,DATA_CSUB ; DATA_CSUB = 4
    mov ax,$2000       ; AL:=0, AH:=%xx10'0000: 3 SHL from zero

@next_bit:
    or ah,ah
    jnz @read_bit

;word_read:
    or al,al           ;
    jnz @adjust_word

;load_uncompressed:
    mov bl,DATA_USUB   ; 42, also a good value for bit counter
    mov ah,bl           ; %xxxx'xx10: 7 SHL from zero
    jmp @next_bit

@read_bit:
    or cl,cl
    jnz @shift_latch

;load_latch:
    inc cx              ; INC CX for CL:=1, %xxxx'xxx1: 8 SHL from zero
    mov ch,[bp]
    inc bp

@shift_latch:
    sal ax,1
    sal cx,1
    adc al,0
    jmp @next_bit

@adjust_word:
    sub al,bl

```

AH: shift counter  
 If it's not zero, read next bit  
 If it's zero, word is read in AL

# Clean Code Decoder (raw-diff-5 @ 3 nctab nutab)

```

load_play_note:
    mov bl,DATA_CSUB ; DATA_CSUB = 4
    mov ax,$2000       ; AL:=0, AH:=%xx10'0000: 3 SHL from zero
@next_bit:
    or ah,ah
    jnz @read_bit
;word_read:
    or al,al ; check for %000 special value
    jnz @adjust_word
;load_uncompressed:
    mov bl,DATA_USUB
    mov ah,bl
    jmp @next_bit
@read_bit:
    or cl,cl
    jnz @shift_latch
;load_latch:
    inc cx           ; INC CX for CL:=1, %xxxx'xxx1: 8 SHL from zero
    mov ch,[bp]
    inc bp
@shift_latch:
    sal ax,1
    sal cx,1
    adc al,0
    jmp @next_bit
@adjust_word:
    sub al,bl

```

If it's not SPEC escape (%000),  
the word is almost ok (later)  
If it's a SPEC escape...

# Clean Code Decoder (raw-diff-5 @ 3 nctab nutab)

```

load_play_note:
    mov bl, DATA_CSUB ; DATA_CSUB = 4
@next_bit:
;word
    jnz @adjust_word
;load_uncompressed:
    mov bl,DATA_USUB ; 42, also a good value for bit counter
    mov ah,bl          ; %xxxx'xx10: 7 SHL from zero
    jmp @next_bit
@read_bit:
    or cl,cl
    jnz @shift_latch
;load_latch:
    inc cx             ; INC CX for CL:=1, %xxxx'xxx1: 8 SHL from zero
    mov ch,[bp]
    inc bp
@shift_latch:
    sal ax,1
    sal cx,1
    adc al,0
    jmp @next_bit
@adjust_word:
    sub al,bl

```

After SPEC: load uncompressed word  
 DATA\_USUB transforms 1..127 data to -35..+41  
 diff, there's space for some optimization...

...it works as shift counter as well.

# Clean Code Decoder (raw-diff-5 @ 3 nctab nutab)

```

load_play_note:
    mov bl,DATA_CSUB ; DATA_CSUB = 4
    mov ax,$2000       ; AL:=0, AH:=%xx10'0000: 3 SHL from zero
@next_bit:
    or ah,ah
    jnz @read_bit
;word_read:
    or al,al
    jnz @adjust_word
;load_uncompressed:
    mov bl,DATA_USUB
    mov ah,bl          ; %xxxx'xx10: 7 SHL from zero
    jmp @next_bit
@read_bit:
    or cl,cl
    jnz @shift_latch
;load_latch:
    inc cx             ; INC CX for CL:=1, %xxxx'xxx1: 8 SHL from zero
    mov ch,[bp]
    inc bp
@shift_latch:
    sal ax,1
    sal cx,1
    adc al,0
    jmp @next_bit
@adjust_word:
    sub al,bl

```

Read bits again with uncompressed counter (AH) and USUB (BL) value

# Clean Code Decoder (raw-diff-5 @ 3 nctab nutab)

```

load_play_note:
    mov bl,DATA_CSUB ; DATA_CSUB = 4
    mov ax,$2000       ; AL:=0, AH:=%xx10'0000: 3 SHL from zero
@next_bit:
    or ah,ah
    jnz @read_bit
;word_read:
    or al,al           ; check for %000 special value
    jnz @adjust_word
;load_uncompressed:
    mov bl,DATA_USUB   ; 42. also a good value for bit counter
    mov ah,bl
    jmp @next_bit
@read_bit:
    or cl,cl
    jnz @shift_latch
;load_latch:
    inc cx             ; INC CX for CL:=1, %xxxx'xxx1: 8 SHL from zero
    mov ch,[bp]
    inc bp
@shift_latch:
    sal ax,1
    sal cx,1
    adc al,0
    jmp @next_bit
@adjust_word:
    sub al,bl

```

CL is the latch counter. If zero,  
new data byte must be read.

# Clean Code Decoder (raw-diff-5 @ 3 nctab nutab)

```

load_play_note:
    mov bl,DATA_CSUB ; DATA_CSUB = 4
    mov ax,$2000       ; AL:=0, AH:=%xx10'0000: 3 SHL from zero
@next_bit:
    or ah,ah
    jnz @read_bit
;word_read:
    or al,al          ; check for %000 special value
    jnz @adjust_word
;load_uncompressed:
    mov bl,DATA_USUB ; 42, also a good value for bit counter
    mov ah,bl          ; %xxxx'xx10: 7 SHL from zero
    jmp @next_bit
@read_bit:
    or cl,cl
    jnz @shift_lat
;load_latch:
    inc cx ; INC CX for CL:=1, %xxxx'xxx1: 8 SHL from zero
    mov ch,[bp]
    inc bp
@shift_latch:
    sal ax,1
    sal cx,1
    adc al,0
    jmp @next_bit
@adjust_word:
    sub al,bl

```

**Initialize latch counter (CL)**  
**Read next data byte to latch (CH)**

# Clean Code Decoder (raw-diff-5 @ 3 nctab nutab)

```

load_play_note:
    mov bl,DATA_CSUB ; DATA_CSUB = 4
    mov ax,$2000       ; AL:=0, AH:=%xx10'0000: 3 SHL from zero
@next_bit:
    or ah,ah
    jnz @read_bit
;word_read:
    or al,al          ; check for %000 special value
    jnz @adjust_word
;load_uncompressed:
    mov bl,DATA_USUB ; 42, also a good value for bit counter
    mov ah,bl          ; %xxxx'xx10: 7 SHL from zero
    jmp @next_bit
@read_bit:
    or cl,cl
    jnz @shift_latch
;load_latch:
    inc cx             ; INC CX for CL:=1, %xxxx'xxx1: 8 SHL from zero
    mov ch,[bp]
    inc bp
@shift_latch:
    sal ax,1
    sal cx,1
    adc al,0
    jmp @next_bit
@adjust_word:
    sub al,bl

```

Shift result, low bit is  
to be read from latch...

# Clean Code Decoder (raw-diff-5 @ 3 nctab nutab)

```

load_play_note:
    mov bl,DATA_CSUB ; DATA_CSUB = 4
    mov ax,$2000       ; AL:=0, AH:=%xx10'0000: 3 SHL from zero
@next_bit:
    or ah,ah
    jnz @read_bit
;word_read:
    or al,al          ; check for %000 special value
    jnz @adjust_word
;load_uncompressed:
    mov bl,DATA_USUB ; 42, also a good value for bit counter
    mov ah,bl          ; %xxxx'xx10: 7 SHL from zero
    jmp @next_bit
@read_bit:
    or cl,cl
    jz @shift_latchn
;load:
    in al,1
    mov bl,al
    in al,1
    Shift latch counter (CL) and value (CH)
    Latch value is shifted to CF, copied to AL
@shift_latchn:
    sal ax,1
    sal cx,1
    adc al,0
    jmp @next_bit
@adjust_word:
    sub al,bl

```

Shift latch counter (CL) and value (CH)  
 Latch value is shifted to CF, copied to AL

# Clean Code Decoder (raw-diff-5 @ 3 nctab nutab)

```

load_play_note:
    mov bl,DATA_CSUB ; DATA_CSUB = 4
    mov ax,$2000       ; AL:=0, AH:=%xx10'0000: 3 SHL from zero
@next_bit:
    or ah,ah
    jnz @read_bit
;word_read:
    or al,al          ; check for %000 special value
    jnz @adjust_word
;load_uncompressed:
    mov bl,DATA_USUB ; 42, also a good value for bit counter
    mov ah,bl          ; %xxxx'xx10: 7 SHL from zero
    jmp @next_bit
@read_bit:
    or cl,cl
    jnz @shift_latch
;load_latch:
    inc cx             ; INC CX for CL:=1, %xxxx'xxx1: 8 SHL from zero
    mov ch,[bp]
    inc bp
@shift_latch:
    sal ax,1
    sal cx,1
    adc al,0
    jmp @next_bit
@adjust_word:
    sub al,bl

```

When a word is read to AL, then CSUB or USUB (BL) must be subtracted from it

# Clean Code Decoder (raw-diff-5 @ 3 nctab nutab)

```
load play_note:
```

```
@nex
```

```
;wor
```

```
;load
```

Player prototype with data: **228 bytes** (no repeat)

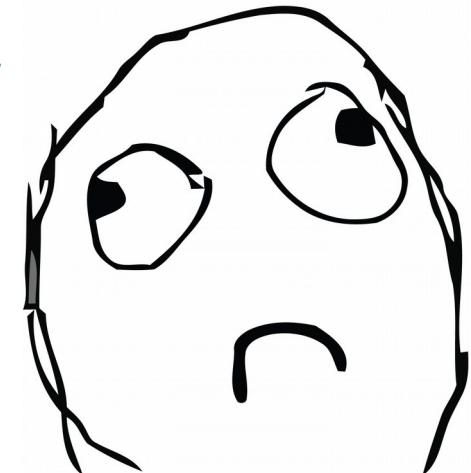
- data: 118 bytes
- playing all notes only once, not repeating
- no visual yet
- after draft implementation of repeating: 377 bytes

```
    mov ah,bl          ; %xxxx'xx10: 7 SHL from zero
    jmp @next_bit
@read_bit:
    or cl,cl
    jnz @shift_latch
;load_latch:
    inc cx            ; INC CX for CL:=1, %xxxx'xxx1: 8 SHL from zero
    mov ch,[bp]
    inc bp
@shift_latch:
    sal ax,1
    sal cx,1
    adc al,0
    jmp @next_bit
@adjust_word:
    sub al,bl
```

[ern0] *We are at 228 byte with repeats  
not implemented, no visual yet...*

[ern0] *We are at 228 byte with repeats  
not implemented, no visual yet...  
My concept does not work,  
it's a slap in the dead end.*

[ern0] *We are at 228 byte with repeats  
not implemented, no visual yet...  
My concept does not work,  
it's a slap in the dead end.*



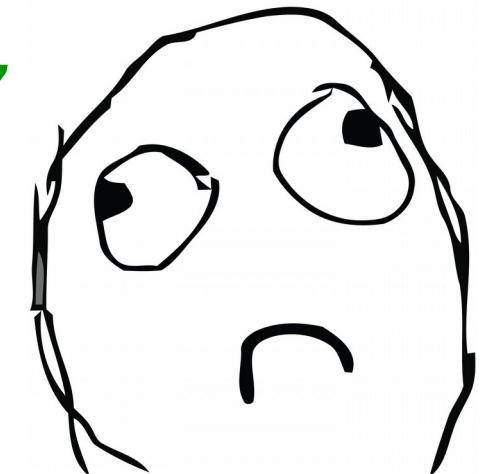
[ern0] *We are at 228 byte with repeats  
not implemented, no visual yet...  
My concept does not work,  
it's a slap in the dead end.*

[TomCat] *Well, let's see...*



[ern0] *We are at 228 byte with repeats  
not implemented, no visual yet...  
My concept does not work,  
it's a slap in the dead end.*

[TomCat] *Well, let's see...*



*Fit into 256 bytes*

# *Fit into 256 bytes*



*Fit into 256 bytes*

	<i>before</i>	<i>after</i>
date	118	114
play		
vis		
to	577	256



*Fit into 256 bytes*

	<i>before</i>	<i>after</i>
<i>data</i>	118	114
<i>player</i>	259	121
<i>visual</i>	0	21
<i>total</i>	377	256

*Fit into 256 bytes*

	<i>before</i>	<i>after</i>	
<i>data</i>	118	114	
<i>player</i>	259	121	-53%
<i>visual</i>	0	21	
<i>total</i>	377	256	

*Fit into 256 bytes*

Major steps of optimization:

*Fit into 256 bytes*

Major steps of optimization:

1. Shorter instructions

*Fit into 256 bytes*

## 1. Shorter instructions (example 1)

<i>before (15 bytes)</i>	<i>after (9 bytes)</i>
<b>push cx</b> <b>lea si, [data_start]</b> <b>lea di, [snapshot_start]</b> <b>mov cx, 5</b> <b>rep movsb</b> <b>pop cx</b>	<b>mov si, data_start</b> <b>mov di, snapshot_start</b> <b>movsw</b> <b>movsw</b> <b>movsb</b>

*Fit into 256 bytes*

1. Shorter instructions – result:

**cca. -60 bytes**

*Fit into 256 bytes*

Major steps of optimization:

1. Shorter instructions
2. Reorganizing the code

*Fit into 256 bytes*

## 2. Reorganizing the code

*Fit into 256 bytes*

## 2. Reorganizing the code

- eliminating subroutines: remove "play\_byte" and others

*Fit into 256 bytes*

## 2. Reorganizing the code

- eliminating subroutines: remove "play\_byte" and others
- less CALL and RET instructions

*Fit into 256 bytes*

## 2. Reorganizing the code

- eliminating subroutines: remove "play\_byte" and others
- less CALL and RET instructions
- usually only one subroutine is enough and optimal, we can reuse it's RET instruction to exit

## *Fit into 256 bytes*

### 2. Reorganizing the code

- eliminating subroutines: remove "play\_byte" and others
- less CALL and RET instructions
- usually only one subroutine is enough and optimal, we can reuse it's RET instruction to exit
- less jumps and conditional jumps

*Fit into 256 bytes*

2. Reorganizing the code - result:

**cca. -40 bytes**

## *Fit into 256 bytes*

Major steps of optimization:

1. Shorter instructions
2. Reorganizing the code
3. Bitfields

*Fit into 256 bytes*

### 3. Bitfields

<i>before</i>	<i>after</i>
<pre>@read_bit:     (...)     inc  cx     mov  ch,[bp]     inc  bp</pre>	<pre>@read_bit:     bt   [si-start+notes],bp     inc  bp     rcl  al,1     jnc  @read_bit</pre>
<pre>@shift_latch:     sal  ax,1     sal  cx,1     adc  al,0</pre>	

*Fit into 256 bytes*

3. Bitfields – result:

**cca. -20 bytes**

*Fit into 256 bytes*

3. Bitfields – result:

**cca. -20 bytes**

Requires flipping bit order of entire data

## *Fit into 256 bytes*

Major steps of optimization:

1. Shorter instructions
2. Reorganizing the code
3. Bitfields
4. Combine play and copy

*Fit into 256 bytes*

## 4. Combine play and copy

<i>before</i>	<i>after</i>
<pre>pusha call eight_of_eight popa xchg si,di call eight_of_eight</pre>	<pre>sub si,3 mov cl,3+8 @three_of_eight: call play_note loop @three_of_eight</pre>
<pre>eight_of_eight: movsw movsw movsb</pre>	<pre>movsb</pre>
<pre>mov cl,3 sub si,cx @three_of_eight: lodsb call play_note loop @three_of_eight</pre>	

*Fit into 256 bytes*

4. Combine play and copy – result:

**cca. -20 bytes**

*Fit into 256 bytes*

Every byte has its own story:

*Fit into 256 bytes*

Every byte has its own story:

- Learn tricks from others - [sizecoding.org](http://sizecoding.org)

*Fit into 256 bytes*

## The MIDI setup

<i>before (6 byte)</i>	<i>after (5 byte)</i>
<pre>org 100H mov al,3fH mov dx,331H out dx,al</pre>	<pre>org 100H db 3fH mov dx,331H outsb ; assume SI=100H</pre> <div data-bbox="1639 587 2201 889"><p>3F: AAS instruction, doesn't hurt</p></div>

## *Fit into 256 bytes*

Every byte has its own story:

- Learn tricks from others - [sizecoding.org](http://sizecoding.org)
- Utilize initial register values

## *Fit into 256 bytes*

When things go crazy - use initial register values

<i>before (5 byte)</i>	<i>after (4 byte)</i>
<pre>; at startup AH=0 add ah,4 jns @next_line</pre>	<pre>; at startup AH=0 adc ah,dh jns @next_line</pre>

*Fit into 256 bytes*

When things go crazy - use initial register values

<i>before (5 byte)</i>	<i>after (4 byte)</i>	
<pre>; at startup AH=0 add ah,4 jns @next_line</pre>	<pre>; at startup AH=0 adc ah,dh jns @next_line</pre>	DX=0331H (MIDI port) DH=3

## *Fit into 256 bytes*

When things go crazy - use initial register values

<i>before (5 byte)</i>	<i>after (4 byte)</i>
<pre>; at startup AH=0 add ah,4 jns @next_line</pre>	<pre>; at startup AH=0 adc ah,dh ins @next_line</pre>

with a compare instruction we can sure the carry flag is always set

## *Fit into 256 bytes*

Every byte has its own story:

- Learn tricks from others - [sizecoding.org](http://sizecoding.org)
- Utilize initial register values
- Optimize data for decoder

*Fit into 256 bytes*

## Optimal data for decoder (CSUB, USUB, SPEC)

<i>before (15 byte)</i>	<i>after (14 byte)</i>
<pre> @load_uncompressed:     mov  ax,256*DATA_USUB+2 @read_bit:     bt   [si-start+notes],bp     inc  bp     rcl  al,1     jnc  @read_bit ;word_read:     test al,al; check for SPEC (0)     jz   @load_uncompressed </pre>	<pre> @load_uncompressed:     mov  ah,DATA_USUB @read_bit:     bt   [si-start+notes],bp     inc  bp     rcl  al,1     jnc  @read_bit ;word_read:     cmp  al,2 ; check for SPEC (2)     je   @load_uncompressed </pre>

# *Fit into 256 bytes*

## Optimal data for decoder (CSUB, USUB, SPEC)

<i>before (15 byte)</i>	<i>after (14 byte)</i>
<pre> @load_uncompressed:     mov  ax,256*DATA_USUB+2 @read_bit:     bt   [si-start+notes],bp     inc  bp     rcl  al,1     jnc  @read_bit ;word_read:     test al,al; check for SPEC (0)     jz   @load_uncompressed </pre>	<pre> @load_uncompressed:     mov  ah,DATA_USUB @read_bit:     bt   [si-start+notes],bp     inc  bp     rcl  al,1     jnc  @read_bit ;word_read:     cmp  al,2 ; check for SPEC (2)     je   @load_uncompressed </pre> <div data-bbox="1572 809 2185 1000" style="border: 1px solid black; padding: 5px;"> <p>AL=%xxxx'xx10: 7 SHL to carry</p> </div>

*Fit into 256 bytes*

## Optimal data for decoder (CSUB, USUB, SPEC)

before (15 byte)	after (14 byte)
<pre> @load_uncompressed:     mov  ax,256*DATA_USUB+2 @read_bit:     bt   [si-start+notes],bp     inc  bp     rcl  al,1     jnc  @read_bit ;word_read:     test al,al; check for SPEC (0)     jz   @load_uncompressed </pre>	<pre> @load_uncompressed:     mov  ah,DATA_USUB @read_bit:     bt   [si-start+notes],bp     inc  bp     rcl  al,1     jnc  @read_bit ;word_read:     cmp  al,2 ; check for SPEC (2)     je   @load_uncompressed </pre> <div data-bbox="1572 809 2194 1000" style="border: 1px solid black; padding: 5px;">       AL=%xxxx'xx10:        7 SHL to carry     </div>

SPEC=0 CSUB=8 USUB=42

SPEC=2 CSUB=10 USUB=58

# *Bugs*

# *Bugs*



# *Bugs*

Bugs:

## *Bugs*

Bugs:

- Dummy instruction trick to skip a branch

# Bugs

## Dummy instruction to skip a branch

<i>wrong</i>	<i>correct</i>
<pre>mov  ax,256*DATA_CSUB+16 db   38H ; CMP ?,BH @load_uncompressed:     mov  ah,DATA_USUB @read_bit:</pre>	<pre>db   66H ; MOV EAX prefix mov  ax,256*DATA_CSUB+16 @load_uncompressed:     mov  ah,DATA_USUB @read_bit:</pre>

# Bugs

## Dummy instruction to skip a branch

<i>wrong</i>	<i>correct</i>
<pre>mov  ax,256*DATA_CSUB+16 db   38H ; CMP ?,BH @load_uncompressed: mov  ah,DATA_USUB @read_bit:</pre> <p>skip the next instruction</p>	<pre>db   66H ; MOV EAX prefix mov  ax,256*DATA_CSUB+16 @load_uncompressed: mov  ah,DATA_USUB @read_bit:</pre>

# Bugs

## Dummy instruction to skip a branch

<i>wrong</i>	<i>correct</i>
<pre>mov ax,256*DATA_CSUB+16 db 38H ; CMP ?,BH @load_uncompressed:     mov ah,DATA_USUB @read_bit:</pre> <p>skip the next instruction</p>	<pre>db 66H ; MOV EAX prefix mov ax,256*DATA_CSUB+16 @load_uncompressed:     mov ah,DATA_USUB @read_bit:</pre> <p>skip the MOV AH instruction</p>

## *Bugs*

### Bugs:

- Dummy instruction trick to skip a branch
- DOSBox timer - Windows vs MacOS

# Bugs

## Half speed on MacOS using BIOS timer

DOS	BIOS
<pre>; AH=2CH @wait_tick: int 21H cmp bl,dl je @wait_tick</pre>	<pre>@wait_tick: int 1aH cmp bp,dx je @wait_tick mov bp,dx</pre>

# Bugs

Half speed on MacOS using BIOS timer

DOS	BIOS
<pre>; AH=2CH @wait_tick: int 21H cmp bl,dL je @wait_tick</pre>	<pre>@wait_tick: int 1aH cmp bp,dx je @wait_tick mov bp,dx</pre>

Windows

MacOS

# Bugs

Half speed on MacOS using BIOS timer

DOS	BIOS
<pre>; AH=2CH @wait_tick: int 21H cmp bl,dL je @wait_tick</pre>	<pre>@wait_tick: int 1aH cmp bp,dx je @wait_tick mov bp,dx</pre>



Windows

MacOS

# Bugs

Half speed on MacOS using BIOS timer

DOS	BIOS
<pre>; AH=2CH @wait_tick: int 21H cmp bl,dL je @wait_tick</pre>	<pre>@wait_tick: int 1aH cmp bp,dx je @wait_tick mov bp,dx</pre>



Windows



MacOS

# Bugs

Half speed on MacOS using BIOS timer

DOS	BIOS
<pre>; AH=2CH @wait_tick: int 21H cmp bl,dL je @wait_tick</pre>	<pre>@wait_tick: int 1aH cmp bp,dx je @wait_tick mov bp,dx</pre>



Windows



MacOS

# Bugs

Half speed on MacOS using BIOS timer

DOS	BIOS
<pre>; AH=2CH @wait_tick: int 21H cmp bl,dL je @wait_tick</pre>	<pre>@wait_tick: int 1aH cmp bp,dx je @wait_tick mov bp,dx</pre>

-  Windows 
-  MacOS 

# Bugs

Half speed on MacOS using BIOS timer

DOS	BIOS
<pre>; AH=2CH @wait_tick: int 21H cmp bl,dl je @wait_tick</pre>	 <pre>t_tick: it 1aH bp, dx je @wait_tick mov bp, dx</pre>

- ✓ Windows ✓
- ✓ MacOS ✗

# *Integration test*

# *Integration test*



**Guillermo Rauch** 

@rauchg

Follow



**Write tests. Not too many. Mostly integration.**

5:43 PM - 10 Dec 2016 from San Francisco, CA

# Integration test



Guillermo Rauch

@rauchg

Follow



Write tests. Not too many. Mostly integration.

5:43 PM - 10 Dec 2016 from San Francisco, CA



# *Integration test*

How it works:

## *Integration test*

How it works:

- test data: 197 diff and 549 note values

## *Integration test*

### How it works:

- test data: 197 diff and 549 note values

```
test_note_data:  
db 60,64,67,72,76,67,72,76  
db 60,64,67,72,76,67,72,76  
db 60,62,69,74,77,69,74,77  
db 60,62,69,74,77,69,74,77  
db 59,62,67,74,77,67,74,77  
db 59,62,67,74,77,67,74,77  
(...)
```

```
test_diff_data:  
db 0,0,0,0,0  
db 0,0,0,0,0  
db 0,-2,2,2,1  
db 0,-2,2,2,1  
db -1,0,-2,0,0  
db -1,0,-2,0,0  
(...)
```

## *Integration test*

### How it works:

- test data: 197 diff and 549 note values
- conditional compilation, on-board execution

## *Integration test*

### How it works:

- test data: 197 diff and 549 note values
- conditional compilation, on-board execution
- calls framework with diff and note values

## *Integration test*

### How it works:

- test data: 197 diff and 549 note values
- conditional compilation, on-board execution
- calls framework with diff and note values

```
@adjust_word:  
    sub al,bl  
  
;rotate_notes:  
    if TEST_MODE > 0  
        call test_diff  
    end if
```

```
play_note:  
    if TEST_MODE > 0  
        jmp test_note  
    end if  
  
    skip note playing  
    and delay
```

## *Integration test*

### How it works:

- test data: 197 diff and 549 note values
- conditional compilation, on-board execution
- calls framework with diff and note values
- log reference and calculated values to file

## *Integration test*

### How it works:

- test data: 197 diff and 549 note
- conditional compilation, on-boar
- calls framework with diff and no
- log reference and calculated va

```
diff #000: =00
note #000: 060
diff #001: =00
note #001: 064
diff #002: =00
note #002: 067
diff #003: =00
note #003: 072
diff #004: =00
note #004: 076
note #005: 067
note #006: 072
note #007: 076
note #008: 060
note #009: 064
note #010: 067
note #011: 072
note #012: 076
note #013: 067
note #014: 072
note #015: 076
diff #005: =00
note #016: 060
diff #006: -02
note #017: 062
diff #007: +02
note #018: 069
```

## Integration test

### How it works:

- test data: 197 diff and 549 note
- conditional compilation, on-boar
- calls framework with diff and no
- log reference and calculated va



```
diff #000: =00
  note #000: 060
diff #001: =00
  note #001: 064
diff #002: =00
  note #002: 067
diff #003: =00
  note #003: 072
diff #004: =00
  note #004: 076
  note #005: 067
  note #006: 072
  note #007: 076
  note #008: 060
  note #009: 064
  note #010: 067
  note #011: 072
  note #012: 076
  note #013: 067
  note #014: 072
  note #015: 076
diff #005: =00
  note #016: 060
diff #006: -02
  note #017: 062
diff #007: +02
  note #018: 069
```

## *Integration test*

### How it works:

- test data: 197 diff and 549 note
- conditional compilation on hea
- calls framework with
- log reference and c

```
diff #000: =00  
note #000: 060  
diff #001: =00  
note #001: 064  
diff #002: =00  
note #002: 067  
diff #003: =00  
note #003: 072  
diff #004: =00  
note #004: 076  
note #005: 067  
note #006: 072  
note #007: 076  
note #008: 060  
note #009: 064  
note #010: 067  
note #011: 072  
note #012: 076  
note #013: 067  
note #014: 072  
note #015: 076  
diff #005: =00  
note #016: 060  
diff #006: -02  
note #017: 062  
diff #007: +02  
note #018: 069
```

## Integration test

### How it works:

- test data: 197 diff and 549 note
- conditional compilation
- calls framework with
- log reference and c



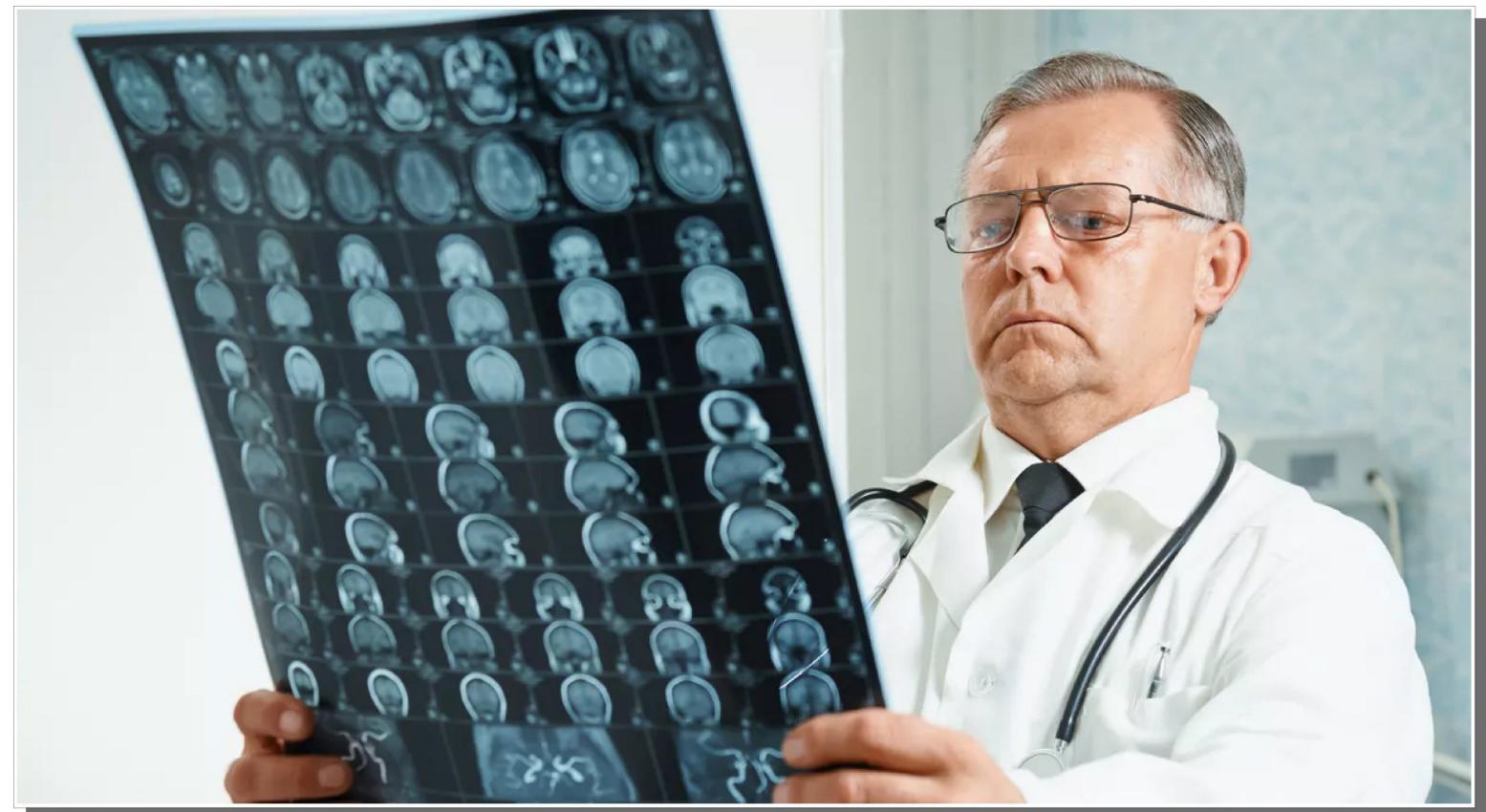
```
diff #000: -05 <---- =00  
note #000: 055 <-- 060  
diff #001: -08 <---- =00  
note #001: 056 <-- 064  
diff #002: -05 <---- =00  
note #002: 062 <-- 067  
diff #003: -08 <---- =00  
note #003: 064 <-- 072  
diff #004: -05 <---- =00  
note #004: 071 <-- 076  
note #005: 064 <-- 067
```

```
diff #000: =00  
note #000: 060  
diff #001: =00  
note #001: 064  
diff #002: =00  
note #002: 067  
diff #003: =00  
note #003: 072  
diff #004: =00  
note #004: 076  
note #005: 067  
note #006: 072  
note #007: 076  
note #008: 060  
note #009: 064  
note #010: 067  
note #011: 072  
note #012: 076  
note #013: 067  
note #014: 072  
note #015: 076  
diff #005: =00  
note #016: 060  
diff #006: -02  
note #017: 062  
diff #007: +02  
note #018: 069
```

# *Integration test - save bug*

# *Integration test - save bug*

Symptoms:



## *Integration test - save bug*

Symptoms:

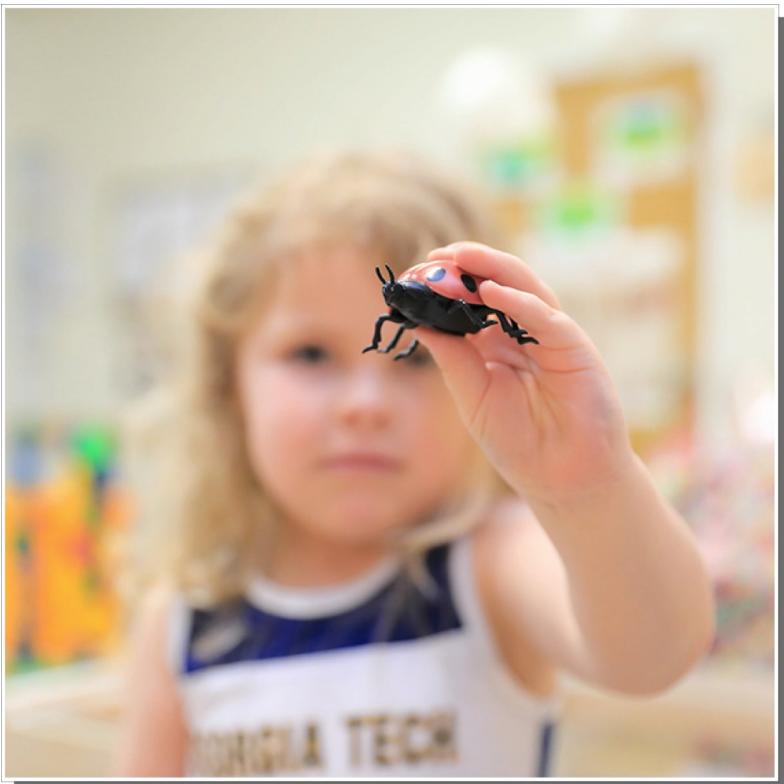
- Test result shows errors, but the song plays OK



## *Integration test - save bug*

### Symptoms:

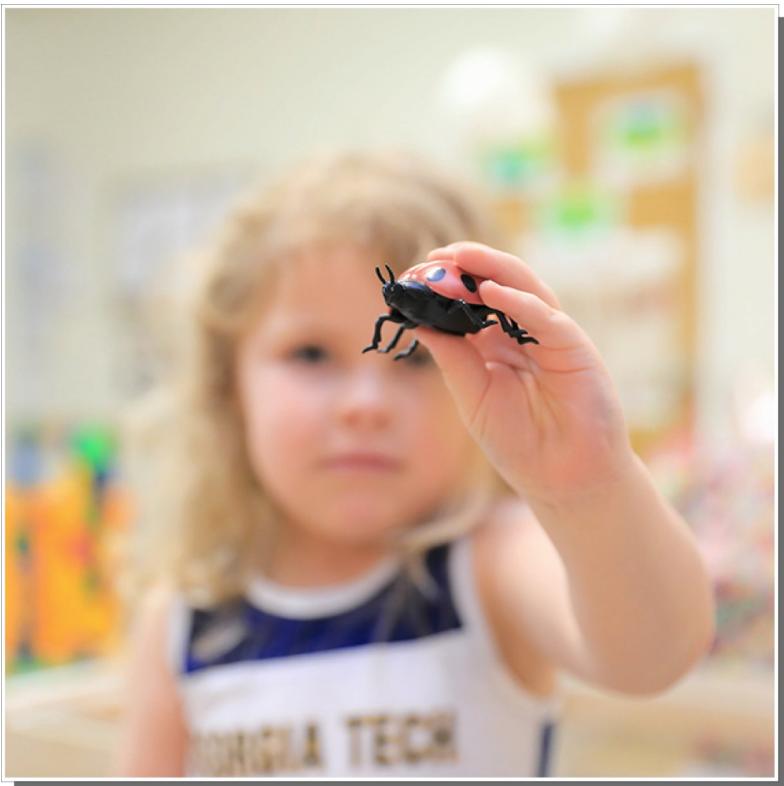
- Test result shows errors, but the song plays OK
- Closer look: it contains bad reference values



## *Integration test - save bug*

### Symptoms:

- Test result shows errors, but the song plays OK
- Closer look: it contains bad reference values

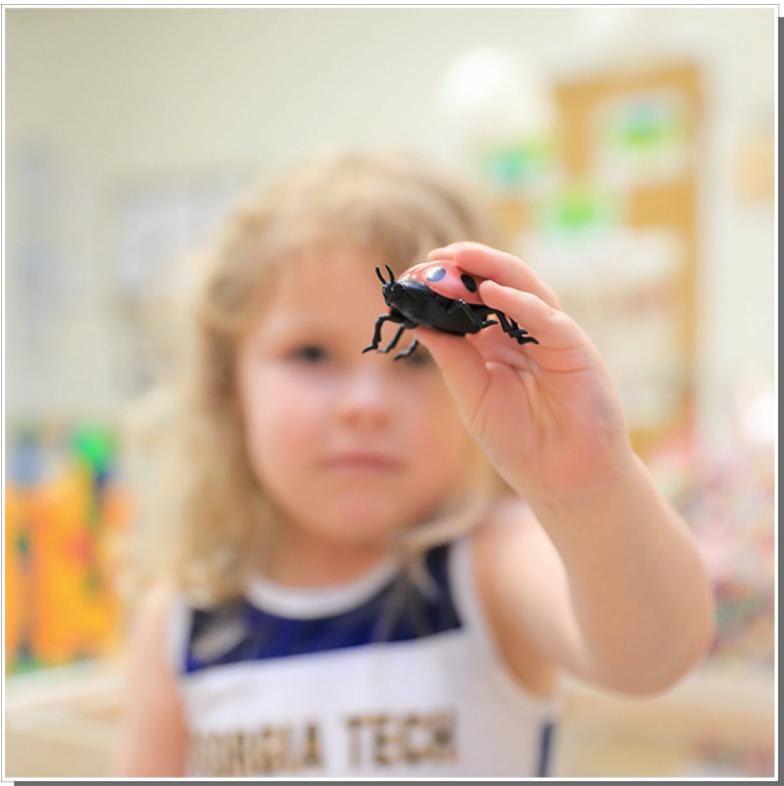


```
diff #000: -05 <---- =00
    note #000: 055 <-- 060
diff #001: -08 <---- =00
    note #001: 056 <-- 064
diff #002: -05 <---- =00
    note #002: 062 <-- 067
diff #003: -08 <---- =00
    note #003: 064 <-- 072
diff #004: -05 <---- =00
    note #004: 071 <-- 076
    note #005: 064 <-- 067
```

## *Integration test - save bug*

### Symptoms:

- Test result shows errors, but the song plays OK
- Closer look: it contains bad reference values



**X**

```
diff #000: -05 <---- =00
note #000: 055 <-- 060
diff #001: -08 <---- =00
note #001: 056 <-- 064
diff #002: -05 <---- =00
note #002: 062 <-- 067
diff #003: -08 <---- =00
note #003: 064 <-- 072
diff #004: -05 <---- =00
note #004: 071 <-- 076
note #005: 064 <-- 067
```

# Integration test - save bug

## Symptoms:

- Test result shows errors, but the song plays OK
- Closer look: it contains bad reference values
- Debugging: program works OK, good values

DOSBox 0.74-2, Cpu speed: 100000 cycles, Frameskip 0, Program: INSIGHT

80486		Insight 1.24																																																																																																																						
0100	→B03F	mov	al,3F	AX=0000	SI=0100	CS=0D21																																																																																																																		
0102	B03103	mov	dx,0331	BX=0000	DI=FFFE	DS=0D21																																																																																																																		
0105	EE	out	dx,al	CX=00FF	BP=091C	ES=0D21																																																																																																																		
0106	8D2E0C02	lea	bp,[020C]	DX=0D21	SP=FFFE	SS=0D21																																																																																																																		
010A	30C9	xor	cl,cl																																																																																																																					
010C	60	pusha		IP=0100	Flags=7202																																																																																																																			
010D	51	push	cx	OF DF IF SF ZF AF PF CF																																																																																																																				
010E	8D360702	lea	si,[0207]	0 0 1 0 0 0 0 0																																																																																																																				
0112	8D3E8301	lea	di,[0183]																																																																																																																					
0116	B90500	mov	cx,0005	SS:000C 0192 +000A 0000																																																																																																																				
0119	F3A4	repe	movsb	SS:000A 08A1 +0008 0000																																																																																																																				
011B	59	pop	cx	SS:0008 DEAD +0006 0000																																																																																																																				
011C	E86900	call	0188	SS:0006 FFFF +0004 0000																																																																																																																				
011F	8D368301	lea	si,[0183]	SS:0004 EA00 +0002 0000																																																																																																																				
0123	8D3E0702	lea	di,[0207]	SS:0002 9FFF +0000 0000																																																																																																																				
0127	B90500	mov	cx,0005	SS:0000 20CD -0002 0000																																																																																																																				
012A	F3A4	repe	movsb	SS:FFFE 0000 -0004 0000																																																																																																																				
0D21:0100																																																																																																																								
<table border="1"> <thead> <tr> <th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>A</th><th>B</th><th>C</th><th>D</th><th>E</th><th>F</th> <th>0123456789ABCDEF</th> </tr> </thead> <tbody> <tr> <td>0D21:0000</td><td>CD</td><td>20</td><td>FF</td><td>9F</td><td>00</td><td>EA</td><td>FF</td><td>FF</td><td>AD</td><td>DE</td><td>A1</td><td>08</td><td>92</td><td>01</td><td>00</td><td>00</td><td>= f.Ω i 1@1@..</td> </tr> <tr> <td>0D21:0010</td><td>18</td><td>01</td><td>10</td><td>01</td><td>18</td><td>01</td><td>92</td><td>01</td><td>01</td><td>01</td><td>00</td><td>02</td><td>FF</td><td>FF</td><td>FF</td><td>FF</td><td>↑@0@0@0@0@.0.</td> </tr> <tr> <td>0D21:0020</td><td>FF</td><td>FF</td><td>FF</td><td>FF</td><td>FF</td><td>FF</td><td>FF</td><td>FF</td><td>FF</td><td>FF</td><td>FF</td><td>FF</td><td>17</td><td>0D</td><td>00</td><td>00</td><td>↑@..</td> </tr> <tr> <td>0D21:0030</td><td>00</td><td>00</td><td>14</td><td>00</td><td>18</td><td>00</td><td>21</td><td>0D</td><td>FF</td><td>FF</td><td>FF</td><td>FF</td><td>00</td><td>00</td><td>00</td><td>00</td><td>..@.↑.!F .....</td> </tr> <tr> <td>0D21:0040</td><td>05</td><td>00</td><td>00</td><td>00</td><td>00</td><td>00</td><td>00</td><td>00</td><td>00</td><td>00</td><td>00</td><td>00</td><td>00</td><td>00</td><td>00</td><td>00</td><td>Φ.....</td> </tr> </tbody> </table>														0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	0123456789ABCDEF	0D21:0000	CD	20	FF	9F	00	EA	FF	FF	AD	DE	A1	08	92	01	00	00	= f.Ω i 1@1@..	0D21:0010	18	01	10	01	18	01	92	01	01	01	00	02	FF	FF	FF	FF	↑@0@0@0@0@.0.	0D21:0020	FF	17	0D	00	00	↑@..	0D21:0030	00	00	14	00	18	00	21	0D	FF	FF	FF	FF	00	00	00	00	..@.↑.!F .....	0D21:0040	05	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	Φ.....											
0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	0123456789ABCDEF																																																																																																								
0D21:0000	CD	20	FF	9F	00	EA	FF	FF	AD	DE	A1	08	92	01	00	00	= f.Ω i 1@1@..																																																																																																							
0D21:0010	18	01	10	01	18	01	92	01	01	01	00	02	FF	FF	FF	FF	↑@0@0@0@0@.0.																																																																																																							
0D21:0020	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	17	0D	00	00	↑@..																																																																																																							
0D21:0030	00	00	14	00	18	00	21	0D	FF	FF	FF	FF	00	00	00	00	..@.↑.!F .....																																																																																																							
0D21:0040	05	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	Φ.....																																																																																																							

## *Integration test - save bug*

### Symptoms:

- Test result shows errors, but the song plays OK
- Closer look: it contains bad reference values
- Debugging: program works OK, good values
- When exiting after few notes: everything is OK

## *Integration test - save bug*

### Symptoms:

- Test result shows errors, but the song plays OK
- Closer look: it contains bad reference values
- Debugging: program works OK, good values
- When exiting after few notes: everything is OK
- If the song is shorter than 128 notes: OK

## *Integration test - save bug*

### Symptoms:

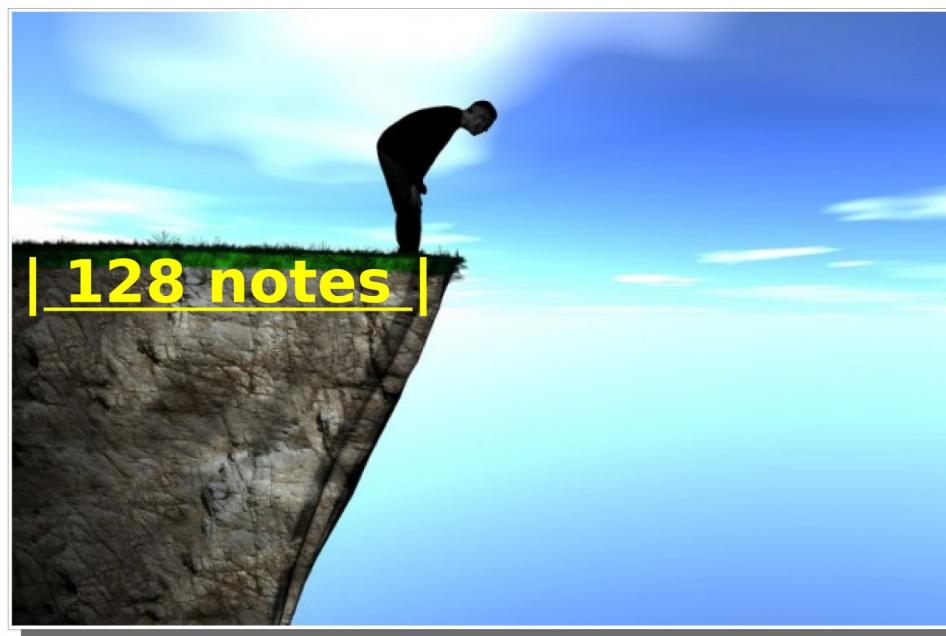
- Test result shows errors, but the song plays OK
- Closer look: it contains bad reference values
- Debugging: program works OK, good values
- When exiting after few notes: everything is OK
- If the song is shorter than 128 notes: OK



## *Integration test - save bug*

### Symptoms:

- Test result shows errors, but the song plays OK
- Closer look: it contains bad reference values
- Debugging: program works OK, good values
- When exiting after few notes: everything is OK
- If the song is shorter than 128 notes: OK



## *Integration test - save bug*

### Symptoms:

- Test result shows errors, but the song plays OK
- Closer look: it contains bad reference values
- Debugging: program works OK, good values
- When exiting after few notes: everything is OK
- If the song is shorter than 128 notes: OK
- Check test framework: no 8-bit counters used

## *Integration test - save bug*

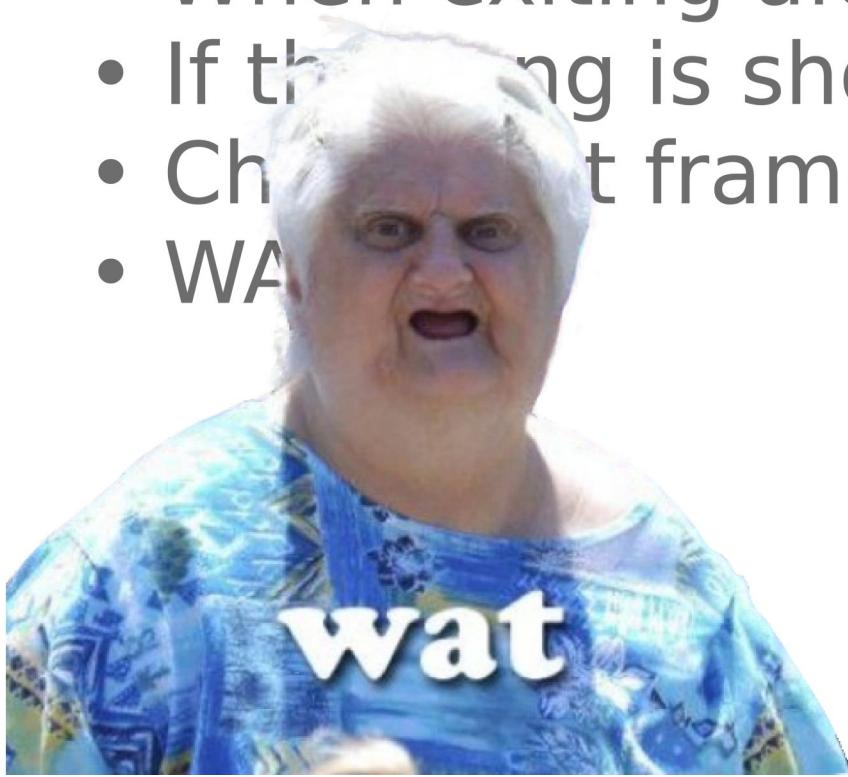
### Symptoms:

- Test result shows errors, but the song plays OK
- Closer look: it contains bad reference values
- Debugging: program works OK, good values
- When exiting after few notes: everything is OK
- If the song is shorter than 128 notes: OK
- Check test framework: no 8-bit counters used
- WAT?

## *Integration test - save bug*

### Symptoms:

- Test result shows errors, but the song plays OK
- Closer look: it contains bad reference values
- Debugging: program works OK, good values
- When exiting after few notes: everything is OK
- If the song is shorter than 128 notes: OK
- Change framework: no 8-bit counters used
- WA



# *Integration test - save bug*

Root cause:

## *Integration test - save bug*

Root cause:

- DOS write() ruins the file, if the data is long

## *Integration test - save bug*

Root cause:

- DOS write() ruins the file, if the data is long



## *Integration test - save bug*

Root cause:

- DOS write() ruins the file, if the data is long



## *Integration test - save bug*

Root cause:

- DOS write() ruins the file, if the data is long

Solution / workaround:

## *Integration test - save bug*

Root cause:

- DOS write() ruins the file, if the data is long

Solution / workaround:

- Close and re-open file frequently

# *Integration test - save bug*

Root cause:

- DOS write() ruins the f

Solution / workaround:

- Close and re-open file

`test_reopen:`

```
    mov  bx,[test_file_handle]
    mov  ah,3eH          ; close
    int  21H
    lea   dx,[test_close_failed_text]
    jc   .fail

    lea   dx,[test_file_name]
    mov  ax,3d02H        ; open for write
    int  21H
    lea   dx,[test_reopen_failed_text]
    jc   .fail

    mov  [test_file_handle],ax

    xor  cx,cx
    xor  dx,dx
    mov  bx,ax
    mov  ax,4202H        ; seek from end
    int  21H
    lea   dx,[test_lseek_failed_text]
    jc   .fail

    ret
```

# *Integration test - save bug*

Root cause:

- DOS write() ruins the f

Solution / workaround:

- Close and re-open file
- A bit slower

```
test_reopen:
```

```

    mov  bx,[test_file_handle]
    mov  ah,3eH          ; close
    int  21H
    lea   dx,[test_close_failed_text]
    jc   .fail

    lea   dx,[test_file_name]
    mov  ax,3d02H        ; open for write
    int  21H
    lea   dx,[test_reopen_failed_text]
    jc   .fail

    mov  [test_file_handle],ax

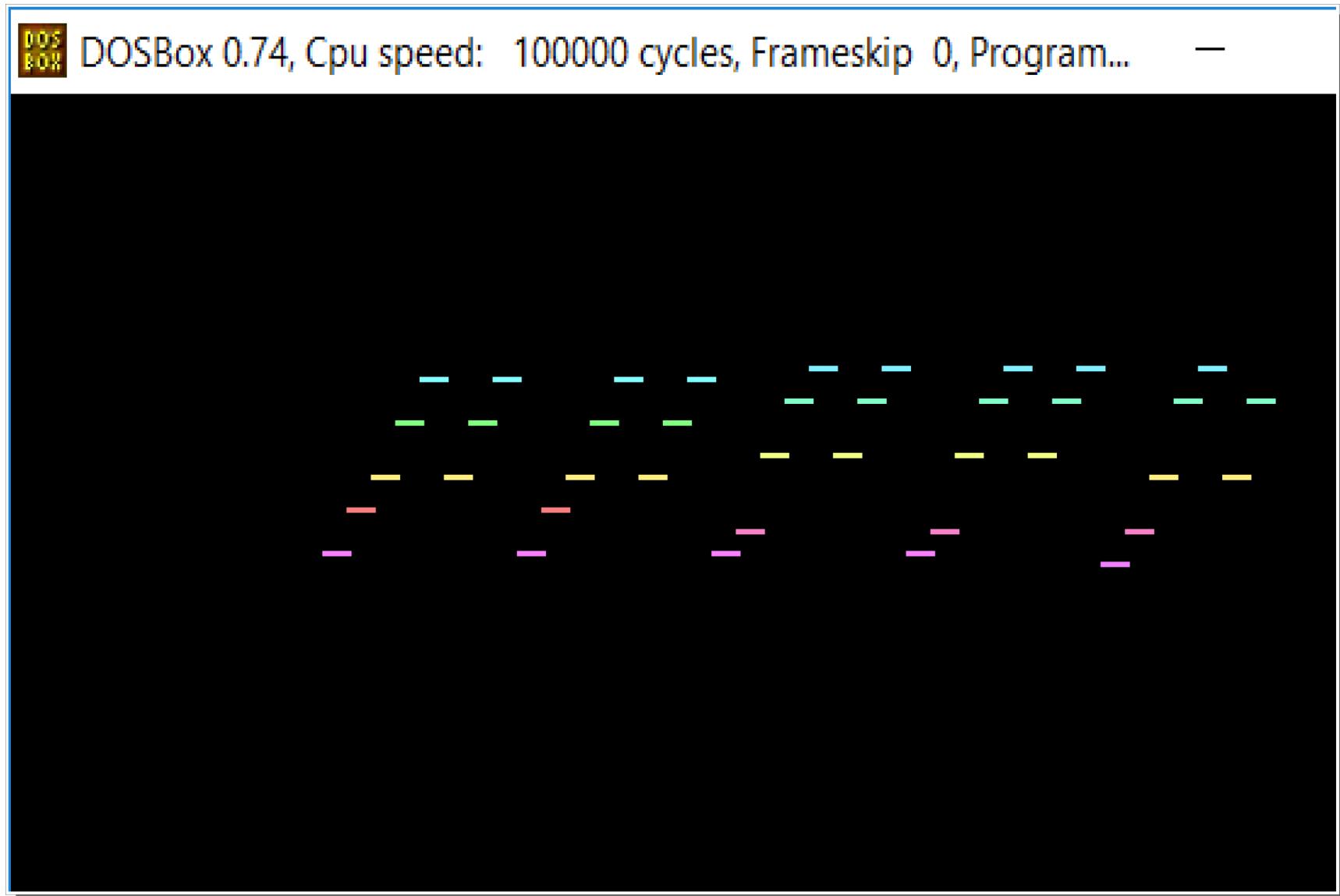
    xor  cx,cx
    xor  dx,dx
    mov  bx,ax
    mov  ax,4202H        ; seek from end
    int  21H
    lea   dx,[test_lseek_failed_text]
    jc   .fail

    ret

```

# *Visual*

# Visual



# Visual

Plot routine:

```
push 0a000H  
pop es  
pusha  
mov ax,90H  
out dx,al  
lodsb  
out dx,al  
imul di,ax,-320*2  
mov cl,bl  
rep stosb  
inc cx  
mov ax,2c7fH  
out dx,al
```

## Visual

Plot routine:

- Interleaved with MIDI player

```
push 0a000H  
pop es  
pusha  
mov ax,90H  
out dx,al  
lodsb  
out dx,al  
imul di,ax,-320*2  
mov cl,bl  
rep stosb  
inc cx  
mov ax,2c7fH  
out dx,al
```

# Visual

## Plot routine:

- Interleaved with MIDI player
- AL: MIDI note + visual input

```
push 0a000H  
pop es  
pusha  
mov ax,90H  
out dx,al  
lodsb  
out dx,al  
imul di,ax,-320*2  
mov cl,bl  
rep stosb  
inc cx  
mov ax,2c7fH  
out dx,al
```

## Visual

### Plot routine:

- Interleaved with MIDI player
- AL: MIDI note + visual input
- complex instruction, but we don't need other regs for calc

```
push 0a000H  
pop es  
pusha  
mov ax,90H  
out dx,al  
lodsb  
out dx,al  
imul di,ax,-320*2  
mov cl,bl  
rep stosb  
inc cx  
mov ax,2c7fH  
out dx,al
```

# Visual

## Plot routine:

- Interleaved with MIDI player
- AL: MIDI note + visual input
- complex instruction, but we don't need other regs for calc
- AX \* -1: mirror, AX \* 2: scale

(draw lower notes on lower part of the screen, higher address)

```
push 0a000H
pop es
pusha
mov ax,90H
out dx,al
lodsb
out dx,al
imul di,ax,-320*2
mov cl,bl
rep stosb
inc cx
mov ax,2c7fH
out dx,al
```

# Visual

## Plot routine:

- Interleaved with MIDI player
- AL: MIDI note + visual input
- complex instruction, but we don't need other regs for calc
- AX \* -1: mirror, AX \* 2: scale  
(draw lower notes on lower part of the screen, higher address)
- bar width = delay length

```
push 0a000H
pop es
pusha
mov ax,90H
out dx,al
lodsb
out dx,al
imul di,ax,-320*2
mov cl,bl
rep stosb
inc cx
mov ax,2c7fH
out dx,al
```

# Visual

## Plot routine:

- Interleaved with MIDI player
- AL: MIDI note + visual input
- complex instruction, but we don't need other regs for calc
- AX \* -1: mirror, AX \* 2: scale  
(draw lower notes on lower part of the screen, higher address)
- bar width = delay length
- bar color = note pitch

```
push 0a000H
pop es
pusha
mov ax,90H
out dx,al
lodsb
out dx,al
imul di,ax,-320*2
mov cl,bl
rep stosb
inc cx
mov ax,2c7fH
out dx,al
```

# Visual

Scroll routine:

```
push 0a000H  
pop es  
( . . . )  
sub di,di  
mov si,1  
mov ch,54H  
es:  
rep movsw
```

## Visual

Scroll routine:

- ES=A000: video segment

```
push 0a000H  
pop es  
( . . . )  
sub di,di  
mov si,1  
mov ch,54H  
es:  
rep movsw
```

## Visual

### Scroll routine:

- ES=A000: video segment
- Cheap destination address: 0

```
push 0a000H  
pop es  
(...)  
sub di,di  
mov si,1  
mov ch,54H  
es:  
rep movsw
```

## Visual

### Scroll routine:

- ES=A000: video segment
- Cheap destination address: 0
- Source, scrolling left by 1 pixel

```
push 0a000H  
pop es  
(...)  
sub di,di  
mov si,1  
mov ch,54H  
es:  
rep movsw
```

## Visual

### Scroll routine:

- ES=A000: video segment
- Cheap destination address: 0
- Source, scrolling left by 1 pixel
- Only 54H \* 256 \* 2 pixels

```
push 0a000H  
pop es  
(...)  
sub di,di  
mov si,1  
mov ch,54H  
es:  
rep movsw
```

## Visual

### Scroll routine:

- ES=A000: video segment
- Cheap destination address: 0
- Source, scrolling left by 1 pixel
- Only 54H \* 256 \* 2 pixels
- ES prefix: cheaper than set DS

```
push 0a000H  
pop es  
(....)  
sub di,di  
mov si,1  
mov ch,54H  
es:  
rep movsw
```

## Visual

### Scroll routine:

- ES=A000: video segment
- Cheap destination address: 0
- Source, scrolling left by 1 pixel
- Only 54H \* 256 \* 2 pixels
- ES prefix: cheaper than set DS
- Copying by words

```
push 0a000H  
pop es  
(....)  
sub di,di  
mov si,1  
mov ch,54H  
es:  
rep movsw
```

**THE END**



THE END

YouTube: <https://www.youtube.com/watch?v=ns7islpMe1U>  
GitHub: <https://github.com/ern0/549notes>



THE END

YouTube: <https://www.youtube.com/watch?v=ns7islpMe1U>  
GitHub: <https://github.com/ern0/549notes>