Nageru: Taking free software video mixing into 2016

Steinar H. Gunderson

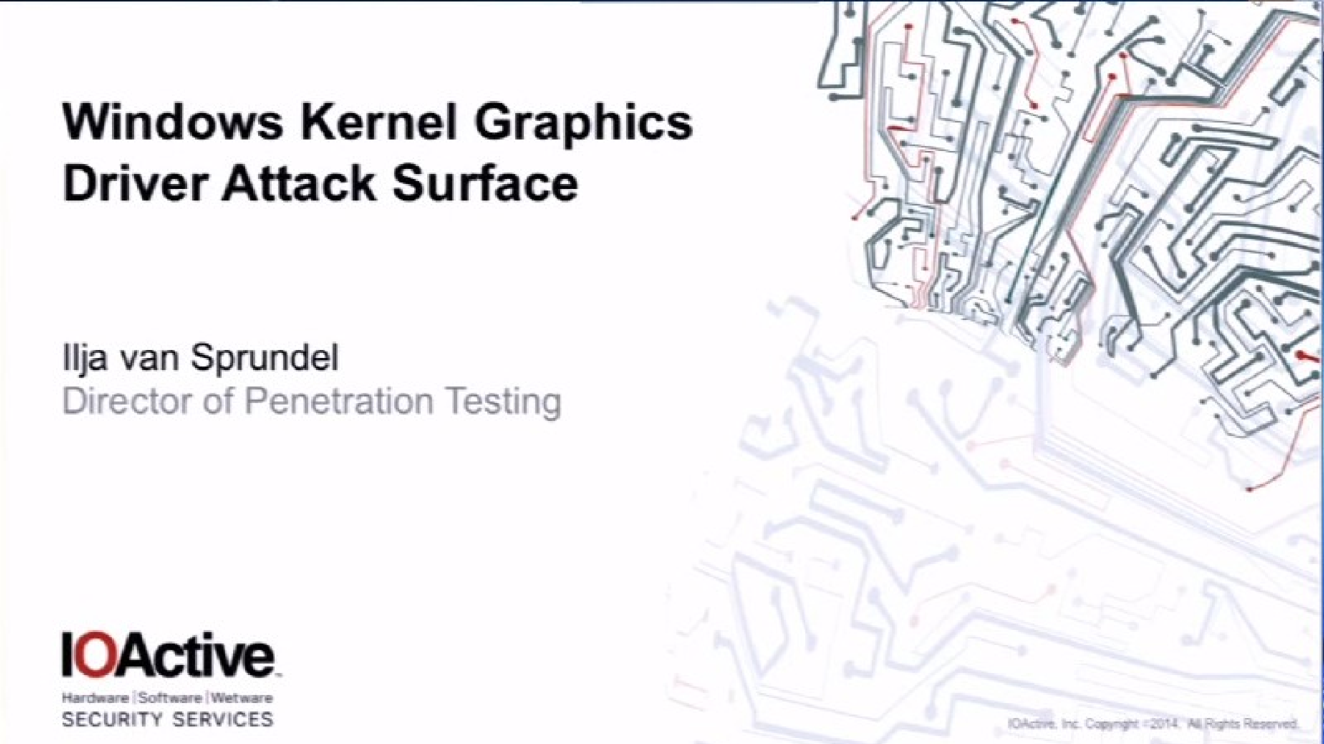
FOSDEM, January 30th 2016

投げる【なげる】 (v1,vt) (1) to throw; to cast away

投げる【なげる】 (v1,vt) (1) to throw; to cast away

(2) to face defeat; to give up;

- Primary goals:
  - High-quality
  - High-performance
  - Usable audio tools
- Nice to have:
  - Suitable for Debian main
  - HTML5 overlay graphics (and/or integration with CasparCG)
  - A pony

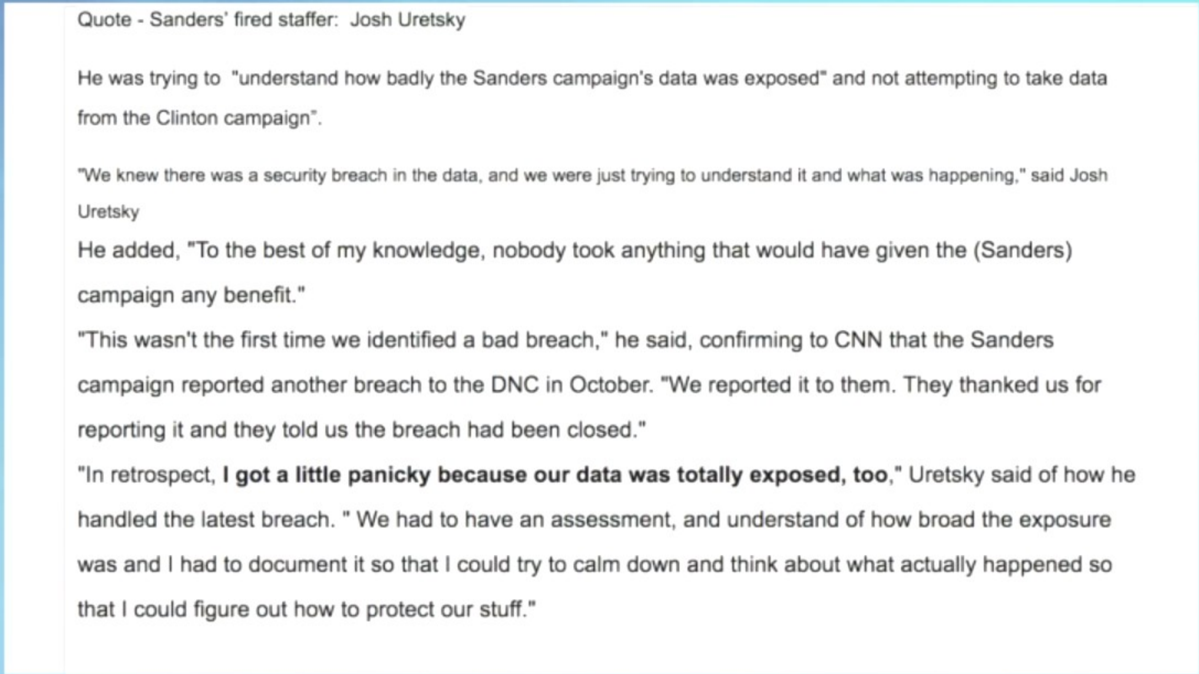


Windows Kernel Graphics Driver Attack Surface

Ilja van Sprundel
Director of Penetration Testing

IOActive
Hardware | Software | Wetware
SECURITY SERVICES

black hat USA 2014



Quote - Sanders' fired staffer: Josh Uretsky

He was trying to "understand how badly the Sanders campaign's data was exposed" and not attempting to take data from the Clinton campaign".

"We knew there was a security breach in the data, and we were just trying to understand it and what was happening," said Josh Uretsky

He added, "To the best of my knowledge, nobody took anything that would have given the (Sanders) campaign any benefit."

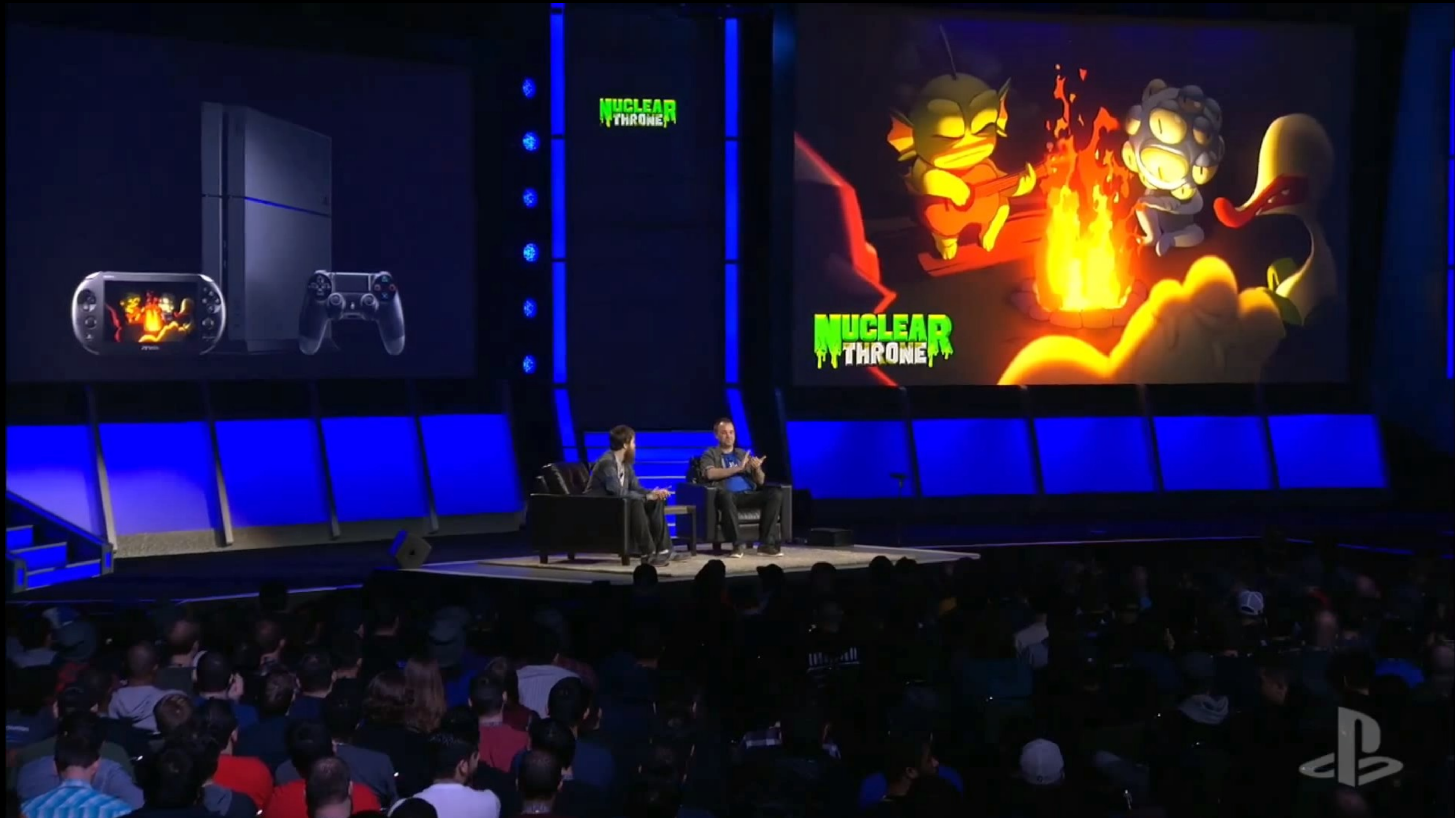
"This wasn't the first time we identified a bad breach," he said, confirming to CNN that the Sanders campaign reported another breach to the DNC in October. "We reported it to them. They thanked us for reporting it and they told us the breach had been closed."

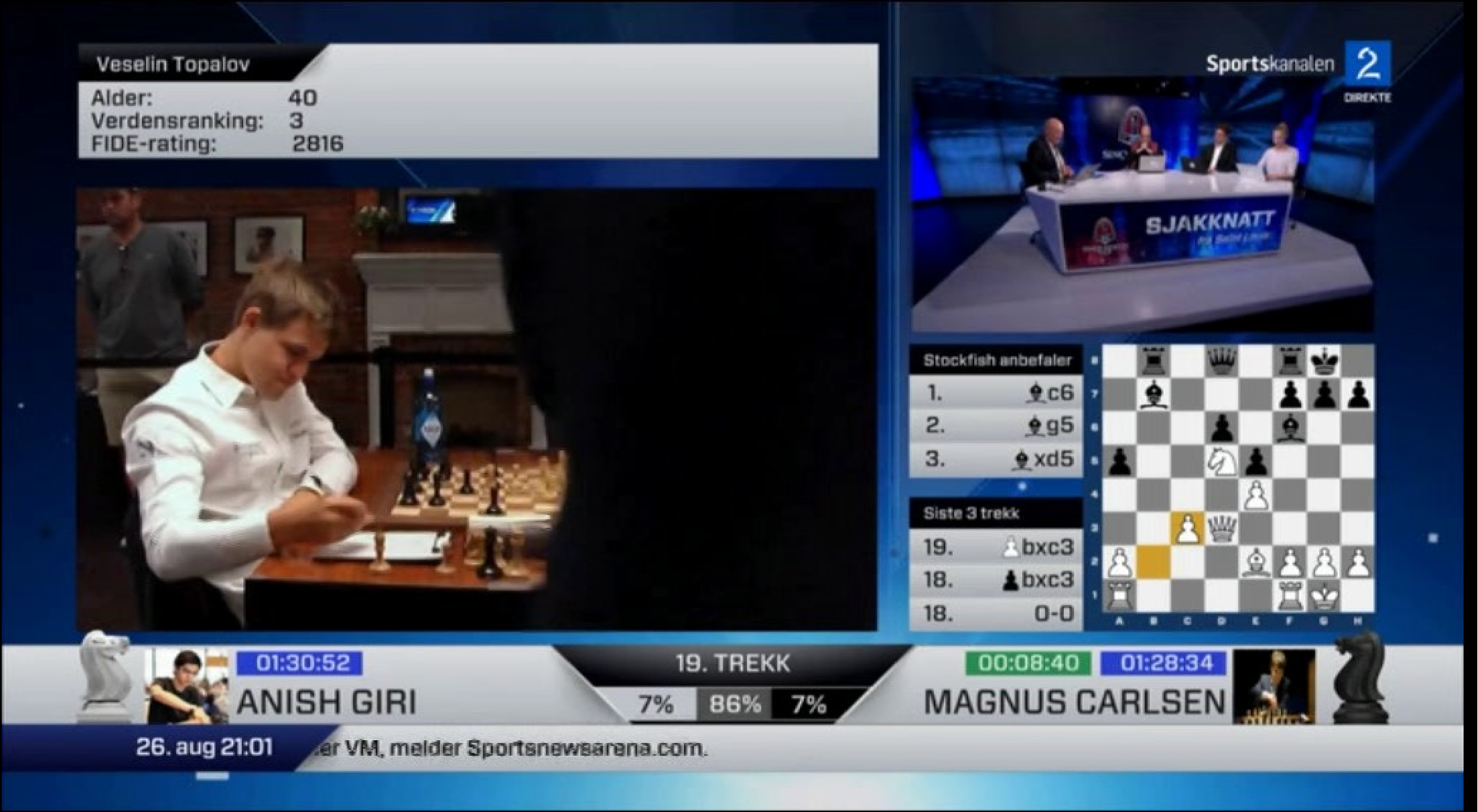
"In retrospect, **I got a little panicky because our data was totally exposed, too**," Uretsky said of how he handled the latest breach. " We had to have an assessment, and understand of how broad the exposure was and I had to document it so that I could try to calm down and think about what actually happened so that I could figure out how to protect our stuff."

From YouTube:

RM S: How dare they to question Sanders ethics!

#Bernie2016


NUCLEAR THRONE
NUCLEAR THRONE



Veselin Topalov

Alder:	40
Verdensranking:	3
FIDE-rating:	2816

Sportskanalen 2 DIREKTE

SJAKKNATT

Stockfish anbefaler	
1.	♗c6
2.	♗g5
3.	♗xd5

Siste 3 trekk	
19.	♙bxc3
18.	♟bxc3
18.	0-0

01:30:52

ANISH GIRI

19. TREKK

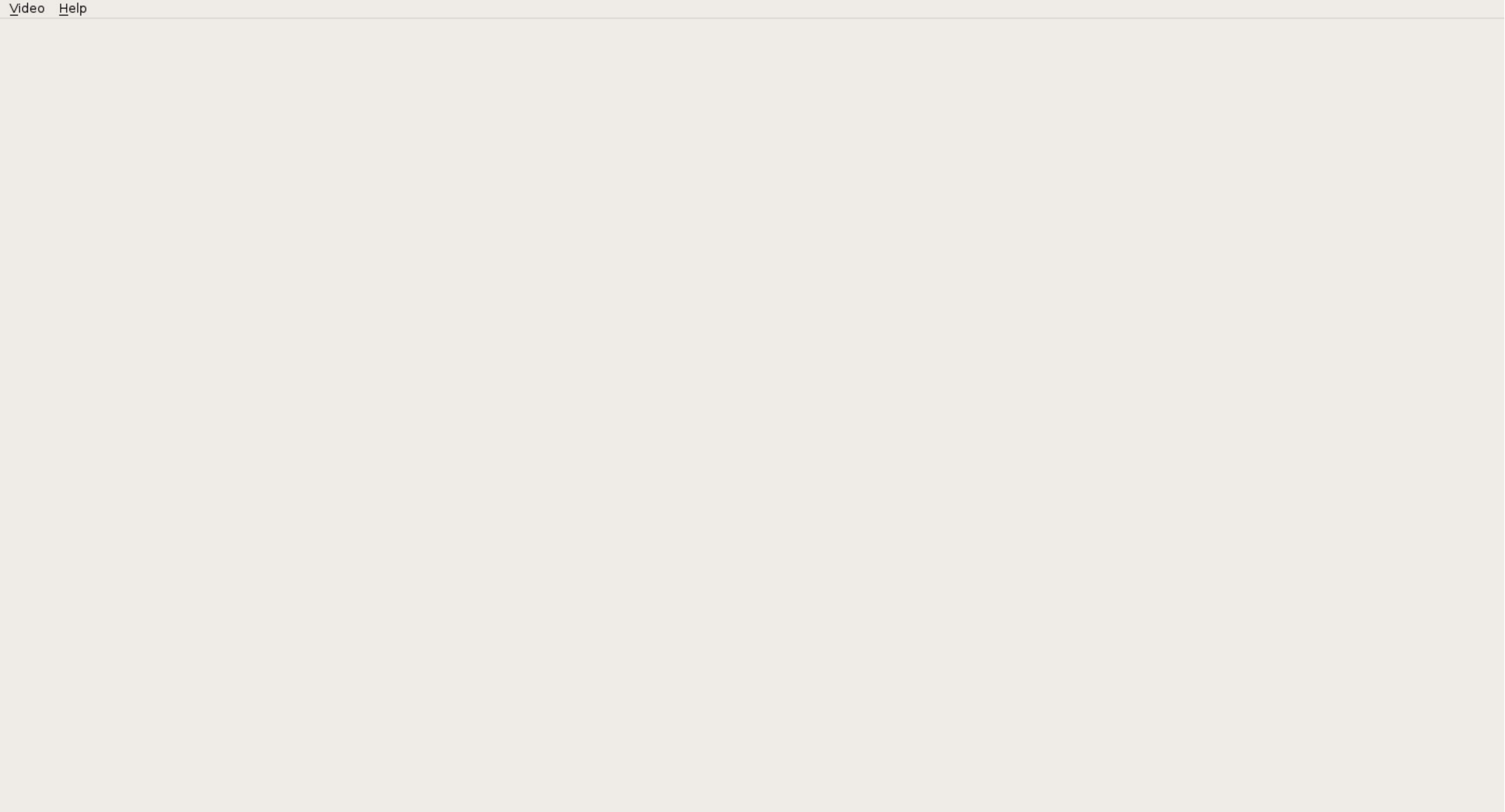
7% 86% 7%

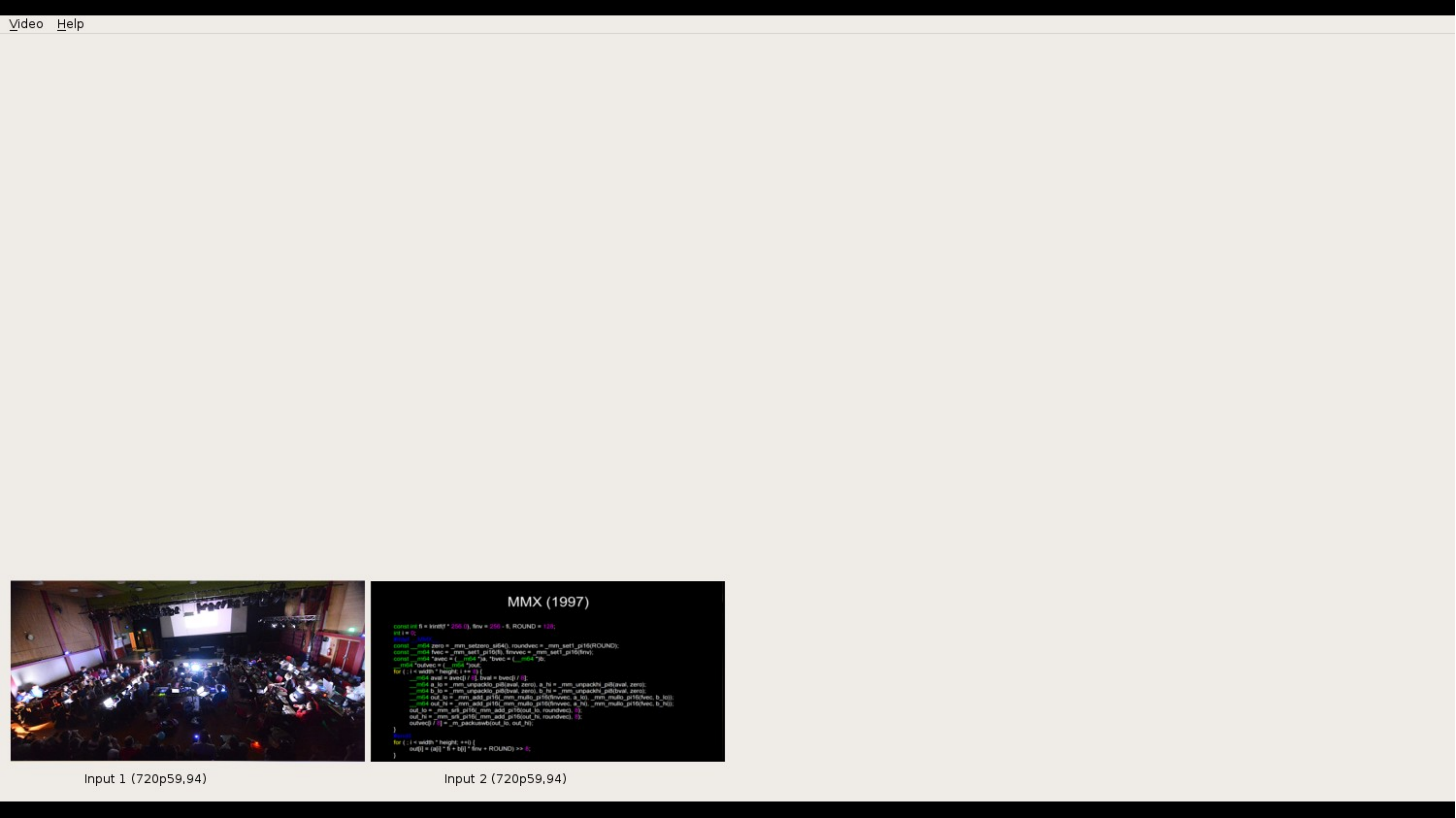
00:08:40 01:28:34

MAGNUS CARLSEN

26. aug 21:01

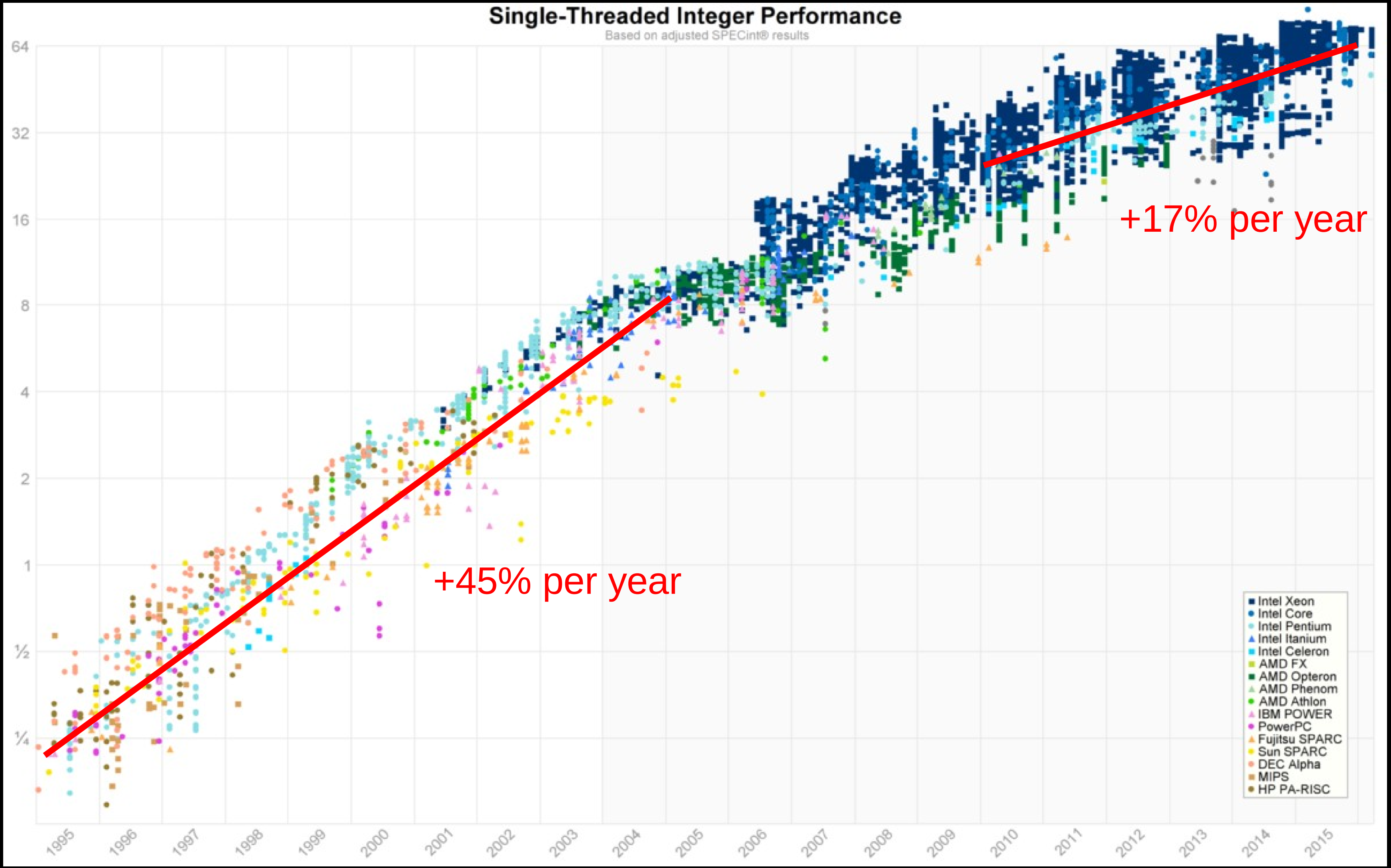
er VM, melder Sportsnewsarena.com.


Video
Help



MMX (1997)

```
const int fi = lrintf(f * 256.0), finv = 256 - fi, ROUND = 128;
int i = 0;
#ifdef __MMX__
const __m64 zero = _mm_setzero_si64(), roundvec = _mm_set1_pi16(ROUND);
const __m64 fvec = _mm_set1_pi16(fi), finvvec = _mm_set1_pi16(finv);
const __m64 *avec = (__m64 *)a, *bvec = (__m64 *)b;
__m64 *outvec = (__m64 *)out;
for ( ; i < width * height; i += 8) {
        __m64 aval = avec[i / 8], bval = bvec[i / 8];
        __m64 a_lo = _mm_unpacklo_pi8(aval, zero), a_hi = _mm_unpackhi_pi8(aval, zero);
        __m64 b_lo = _mm_unpacklo_pi8(bval, zero), b_hi = _mm_unpackhi_pi8(bval, zero);
        __m64 out_lo = _mm_add_pi16(_mm_mullo_pi16(finvvec, a_lo), _mm_mullo_pi16(fvec, b_lo));
        __m64 out_hi = _mm_add_pi16(_mm_mullo_pi16(finvvec, a_hi), _mm_mullo_pi16(fvec, b_hi));
        out_lo = _mm_srli_pi16(_mm_add_pi16(out_lo, roundvec), 8);
        out_hi = _mm_srli_pi16(_mm_add_pi16(out_hi, roundvec), 8);
        outvec[i / 8] = _m_packuswb(out_lo, out_hi);
}
#endif
for ( ; i < width * height; ++i) {
        out[i] = (a[i] * fi + b[i] * finv + ROUND) >> 8;
}
```

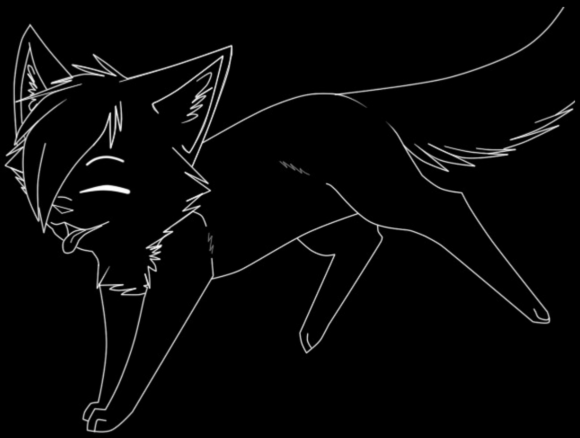


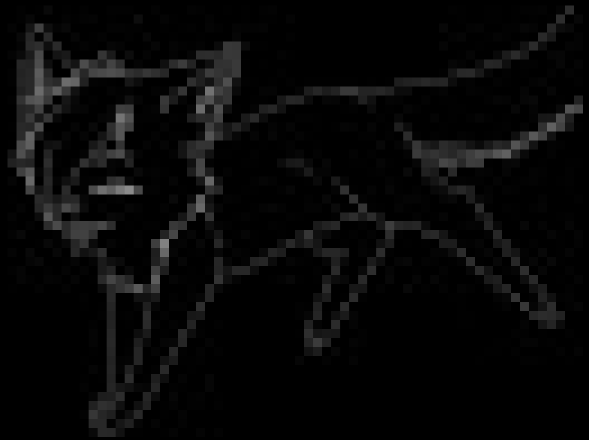
Source: https://github.com/preshing/analyze-spec-benchmarks

From: Steinar H. Gunderson
Cc: Markus Rechberger
Subject: [PATCH] Add support for usbfs zerocopy.

From: Steinar H. Gunderson
Subject: [PATCH] [libusb] Add support for persistent device memory

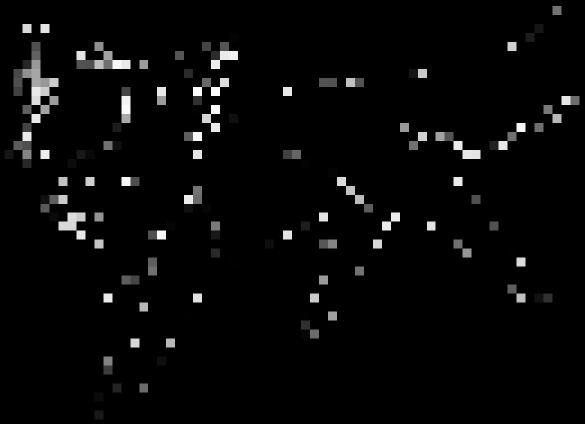
Original

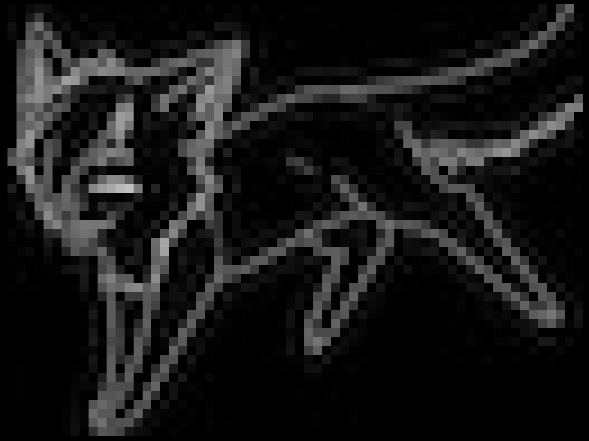




Bilinear

Nearest



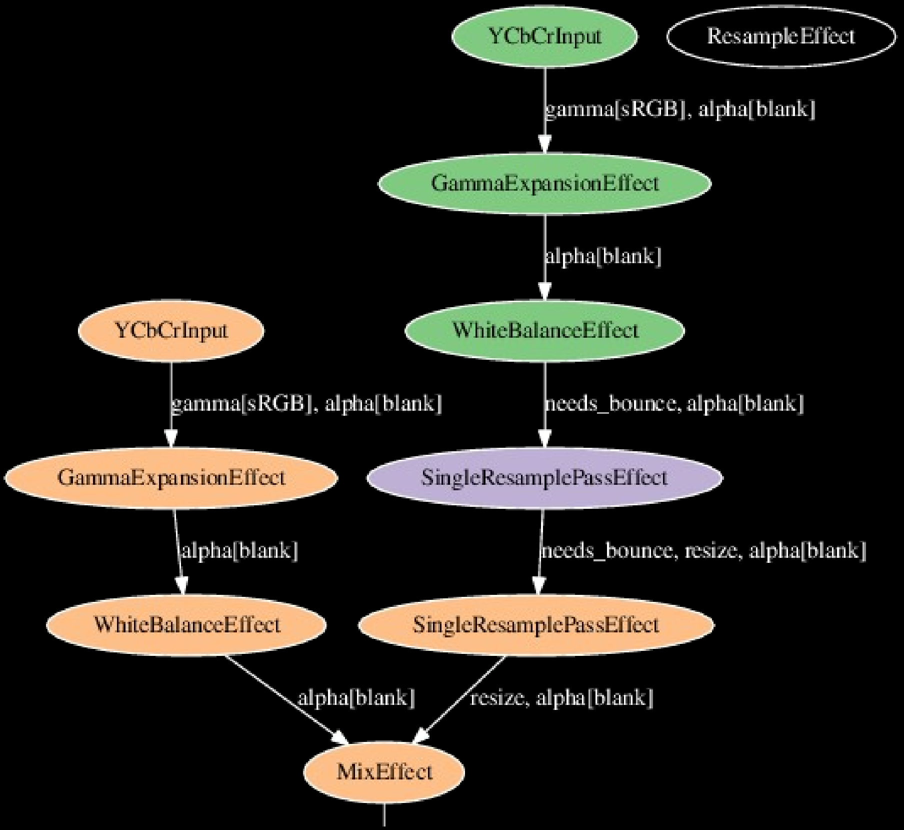


Lanczos, gamma-correct

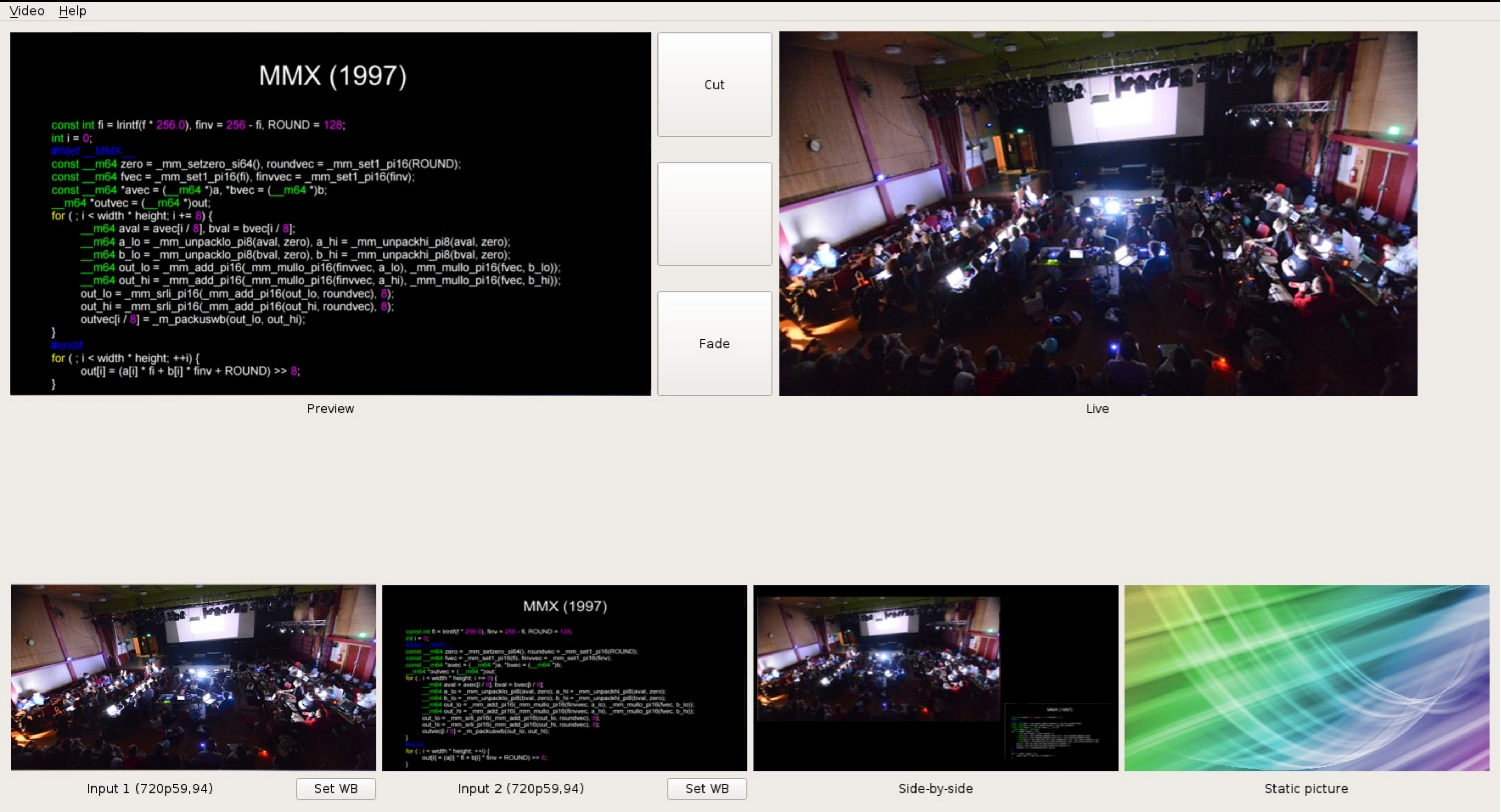
Drawing by sillycandy, CC-BY 3.0

```lua
function make_fade_input(chain, signal, live, deint, scale)
	local input, wb_effect, resample_effect, last
	if live then
		input = chain:add_live_input(false, deint)
		wb_effect = chain:add_effect(WhiteBalanceEffect.new())
```

theme.lua

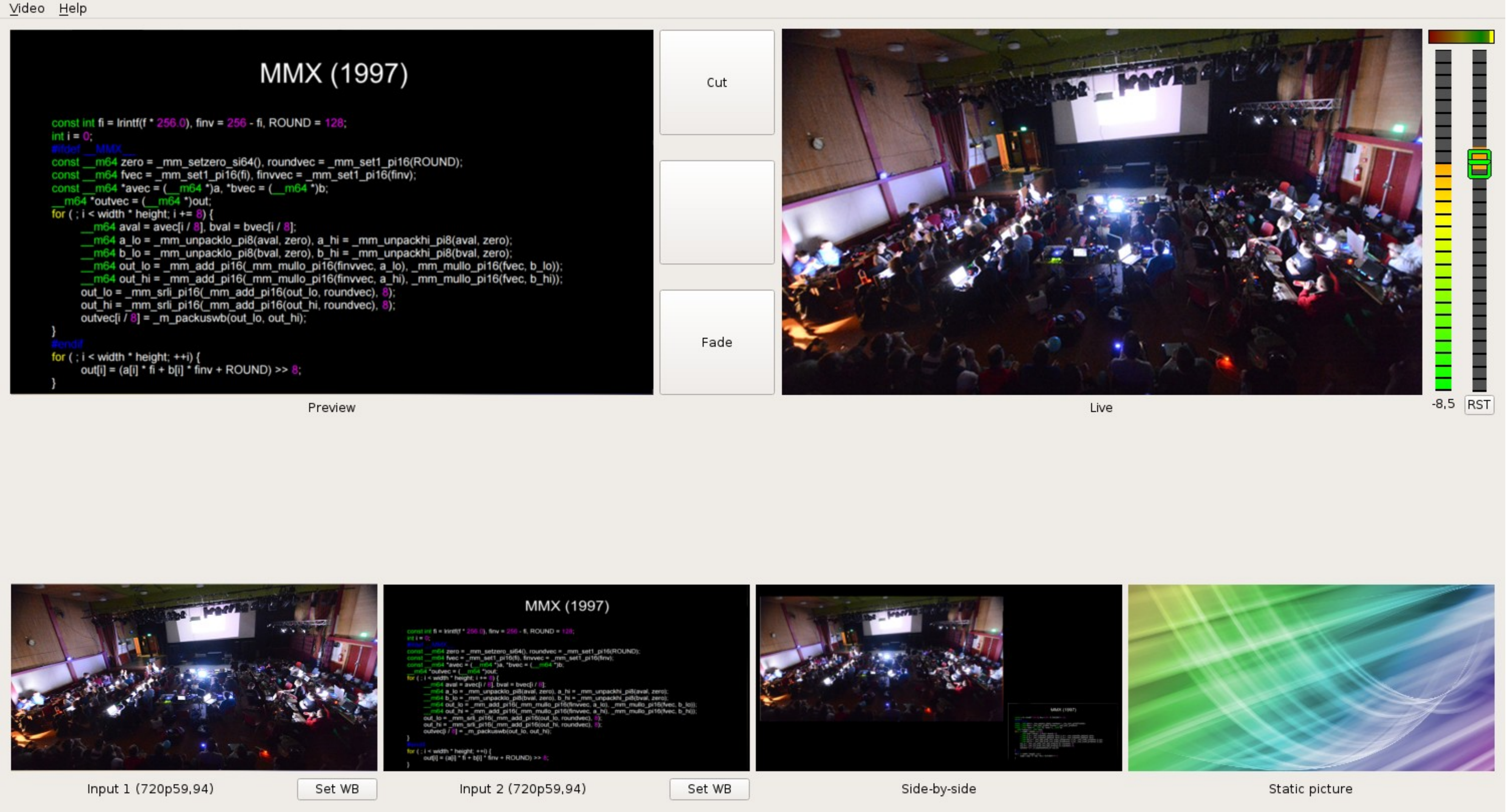


Movit filter graph



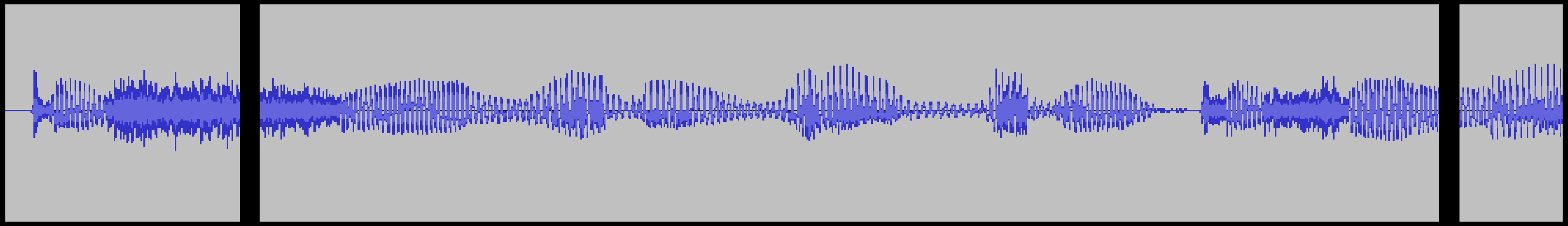
MMX (1997)

```
const int fi = lrintf(f * 256.0), finv = 256 - fi, ROUND = 128;
int i = 0;
#ifdef __MMX__
const __m64 zero = _mm_setzero_si64(), roundvec = _mm_set1_pi16(ROUND);
const __m64 fvec = _mm_set1_pi16(fi), finvvec = _mm_set1_pi16(finv);
const __m64 *avec = (__m64 *)a, *bvec = (__m64 *)b;
__m64 *outvec = (__m64 *)out;
for ( ; i < width * height; i += 8) {
        __m64 aval = avec[i / 8], bval = bvec[i / 8];
        __m64 a_lo = _mm_unpacklo_pi8(aval, zero), a_hi = _mm_unpackhi_pi8(aval, zero);
        __m64 b_lo = _mm_unpacklo_pi8(bval, zero), b_hi = _mm_unpackhi_pi8(bval, zero);
        __m64 out_lo = _mm_add_pi16(_mm_mullo_pi16(finvvec, a_lo), _mm_mullo_pi16(fvec, b_lo));
        __m64 out_hi = _mm_add_pi16(_mm_mullo_pi16(finvvec, a_hi), _mm_mullo_pi16(fvec, b_hi));
        out_lo = _mm_srli_pi16(_mm_add_pi16(out_lo, roundvec), 8);
        out_hi = _mm_srli_pi16(_mm_add_pi16(out_hi, roundvec), 8);
        outvec[i / 8] = _m_packuswb(out_lo, out_hi);
}
#endif
for ( ; i < width * height; ++i) {
        out[i] = (a[i] * fi + b[i] * finv + ROUND) >> 8;
}
```



MMX (1997)

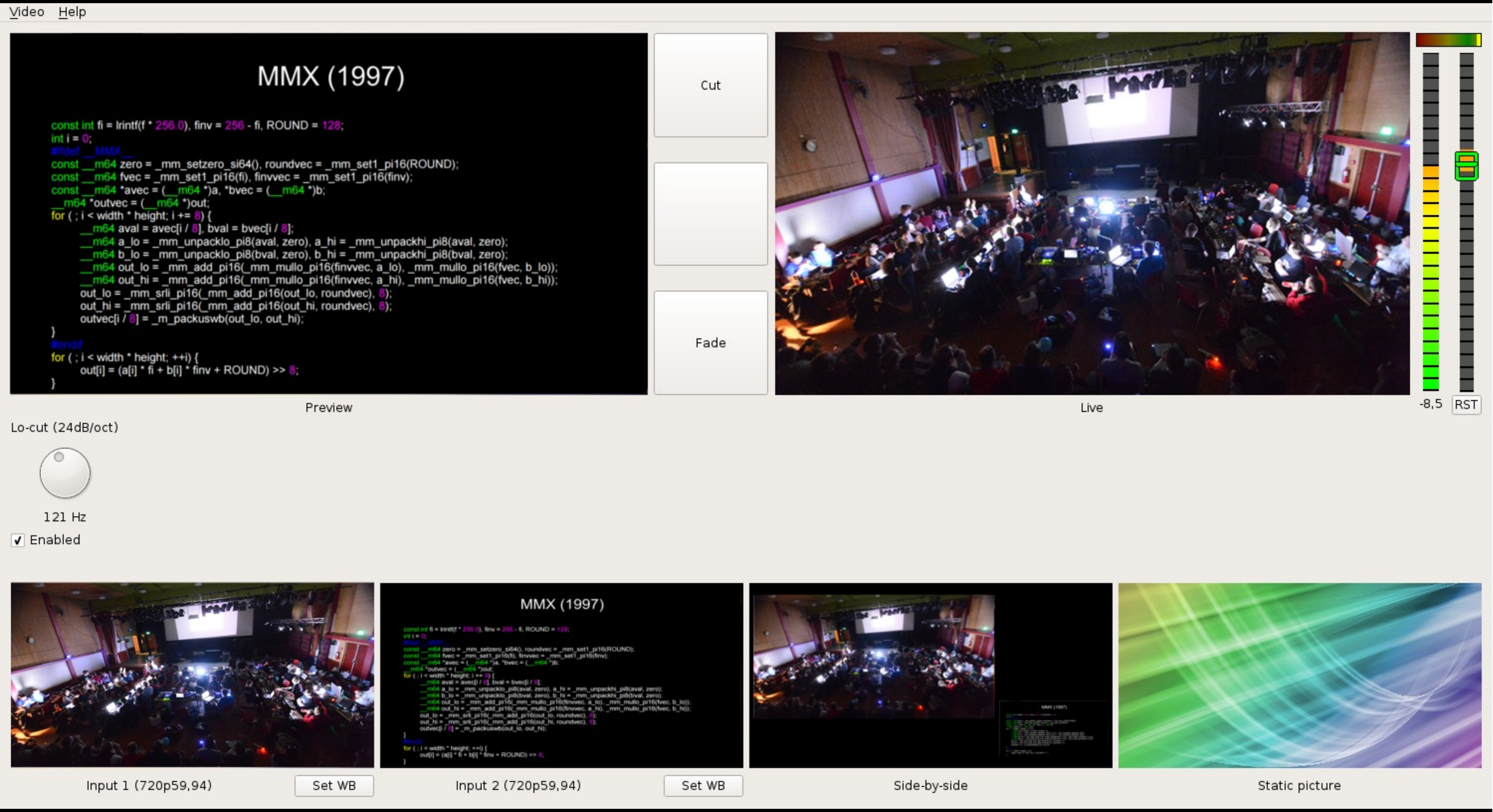
```
const int fi = lrintf(f * 256.0), finv = 256 - fi, ROUND = 128;
int i = 0;
#ifdef __MMX__
const __m64 zero = _mm_setzero_si64(), roundvec = _mm_set1_pi16(ROUND);
const __m64 fvec = _mm_set1_pi16(fi), finvvec = _mm_set1_pi16(finv);
const __m64 *avec = (__m64 *)a, *bvec = (__m64 *)b;
__m64 *outvec = (__m64 *)out;
for ( ; i < width * height; i += 8) {
        __m64 aval = avec[i / 8], bval = bvec[i / 8];
        __m64 a_lo = _mm_unpacklo_pi8(aval, zero), a_hi = _mm_unpackhi_pi8(aval, zero);
        __m64 b_lo = _mm_unpacklo_pi8(bval, zero), b_hi = _mm_unpackhi_pi8(bval, zero);
        __m64 out_lo = _mm_add_pi16(_mm_mullo_pi16(finvvec, a_lo), _mm_mullo_pi16(fvec, b_lo));
        __m64 out_hi = _mm_add_pi16(_mm_mullo_pi16(finvvec, a_hi), _mm_mullo_pi16(fvec, b_hi));
        out_lo = _mm_srli_pi16(_mm_add_pi16(out_lo, roundvec), 8);
        out_hi = _mm_srli_pi16(_mm_add_pi16(out_hi, roundvec), 8);
        outvec[i / 8] = _m_packuswb(out_lo, out_hi);
}
#endif
for ( ; i < width * height; ++i) {
        out[i] = (a[i] * fi + b[i] * finv + ROUND) >> 8;
}
```



Incoming

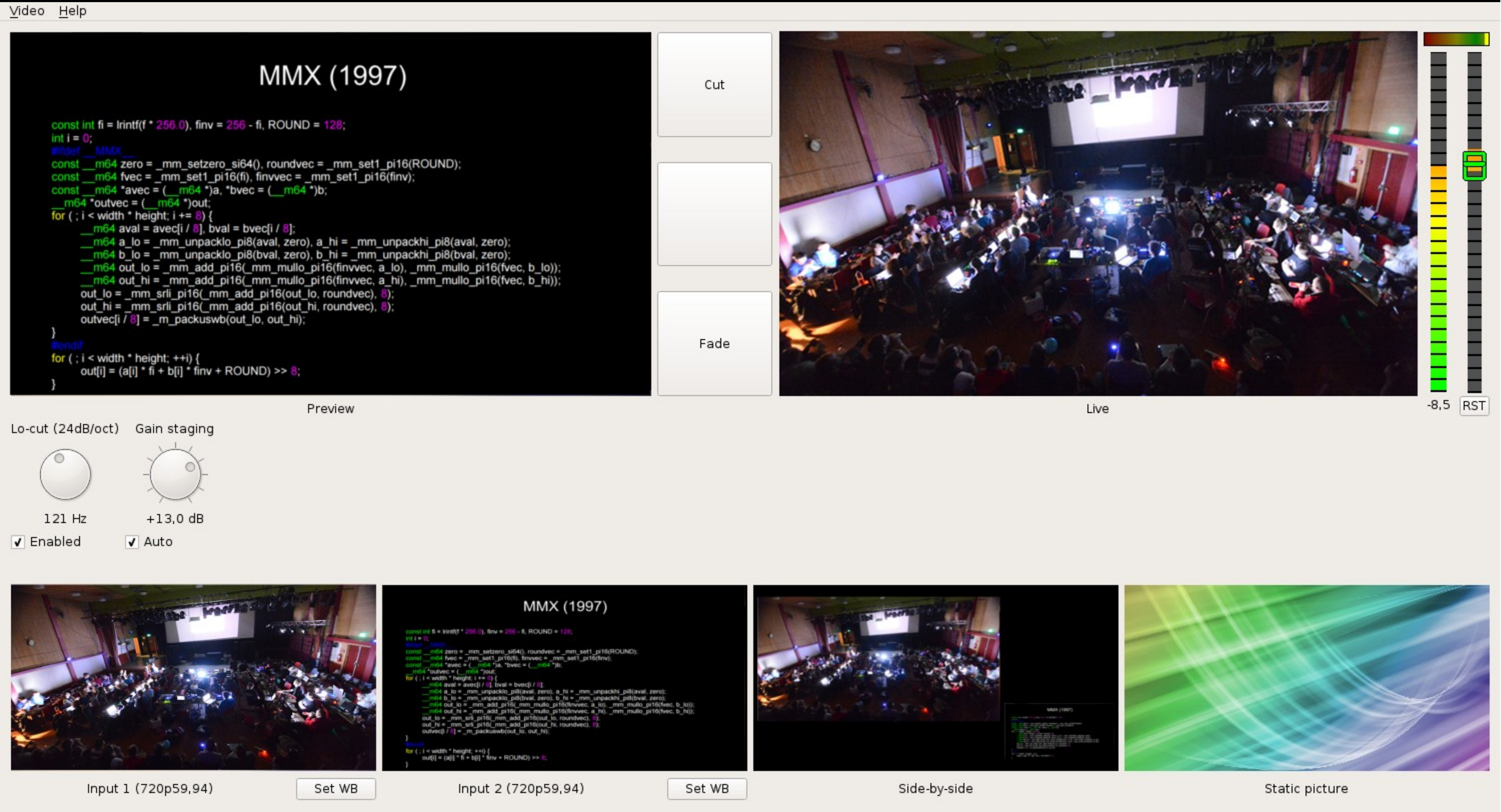
Queue (~100 ms)

Consumed



MMX (1997)

```
const int fi = lrintf(f * 256.0), finv = 256 - fi, ROUND = 128;
int i = 0;
#ifdef __MMX__
const __m64 zero = _mm_setzero_si64(), roundvec = _mm_set1_pi16(ROUND);
const __m64 fvec = _mm_set1_pi16(fi), finvvec = _mm_set1_pi16(finv);
const __m64 *avec = (__m64 *)a, *bvec = (__m64 *)b;
__m64 *outvec = (__m64 *)out;
for ( ; i < width * height; i += 8) {
        __m64 aval = avec[i / 8], bval = bvec[i / 8];
        __m64 a_lo = _mm_unpacklo_pi8(aval, zero), a_hi = _mm_unpackhi_pi8(aval, zero);
        __m64 b_lo = _mm_unpacklo_pi8(bval, zero), b_hi = _mm_unpackhi_pi8(bval, zero);
        __m64 out_lo = _mm_add_pi16(_mm_mullo_pi16(finvvec, a_lo), _mm_mullo_pi16(fvec, b_lo));
        __m64 out_hi = _mm_add_pi16(_mm_mullo_pi16(finvvec, a_hi), _mm_mullo_pi16(fvec, b_hi));
        out_lo = _mm_srli_pi16(_mm_add_pi16(out_lo, roundvec), 8);
        out_hi = _mm_srli_pi16(_mm_add_pi16(out_hi, roundvec), 8);
        outvec[i / 8] = _m_packuswb(out_lo, out_hi);
}
#endif
for ( ; i < width * height; ++i) {
        out[i] = (a[i] * fi + b[i] * finv + ROUND) >> 8;
}
```

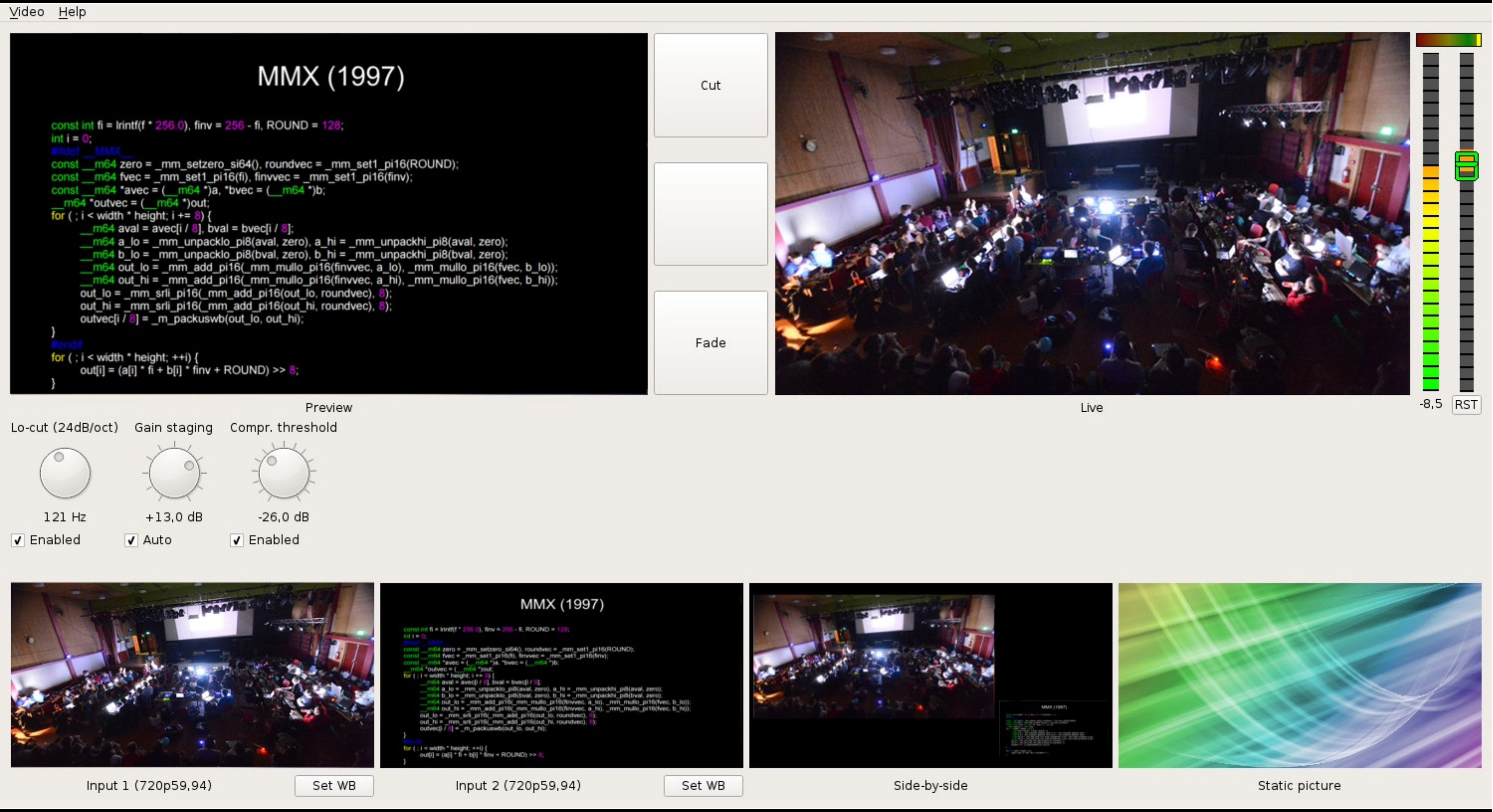


MMX (1997)

```
const int fi = lrintf(f * 256.0), finv = 256 - fi, ROUND = 128;
int i = 0;
#ifdef __MMX__
const __m64 zero = _mm_setzero_si64(), roundvec = _mm_set1_pi16(ROUND);
const __m64 fvec = _mm_set1_pi16(fi), finvvec = _mm_set1_pi16(finv);
const __m64 *avec = (__m64 *)a, *bvec = (__m64 *)b;
__m64 *outvec = (__m64 *)out;
for ( ; i < width * height; i += 8) {
        __m64 aval = avec[i / 8], bval = bvec[i / 8];
        __m64 a_lo = _mm_unpacklo_pi8(aval, zero), a_hi = _mm_unpackhi_pi8(aval, zero);
        __m64 b_lo = _mm_unpacklo_pi8(bval, zero), b_hi = _mm_unpackhi_pi8(bval, zero);
        __m64 out_lo = _mm_add_pi16(_mm_mullo_pi16(finvvec, a_lo), _mm_mullo_pi16(fvec, b_lo));
        __m64 out_hi = _mm_add_pi16(_mm_mullo_pi16(finvvec, a_hi), _mm_mullo_pi16(fvec, b_hi));
        out_lo = _mm_srli_pi16(_mm_add_pi16(out_lo, roundvec), 8);
        out_hi = _mm_srli_pi16(_mm_add_pi16(out_hi, roundvec), 8);
        outvec[i / 8] = _m_packuswb(out_lo, out_hi);
}
#endif
for ( ; i < width * height; ++i) {
        out[i] = (a[i] * fi + b[i] * finv + ROUND) >> 8;
}
```

Lo-cut (24dB/oct)
121 Hz
☑ Enabled

Gain staging
+13,0 dB
☑ Auto



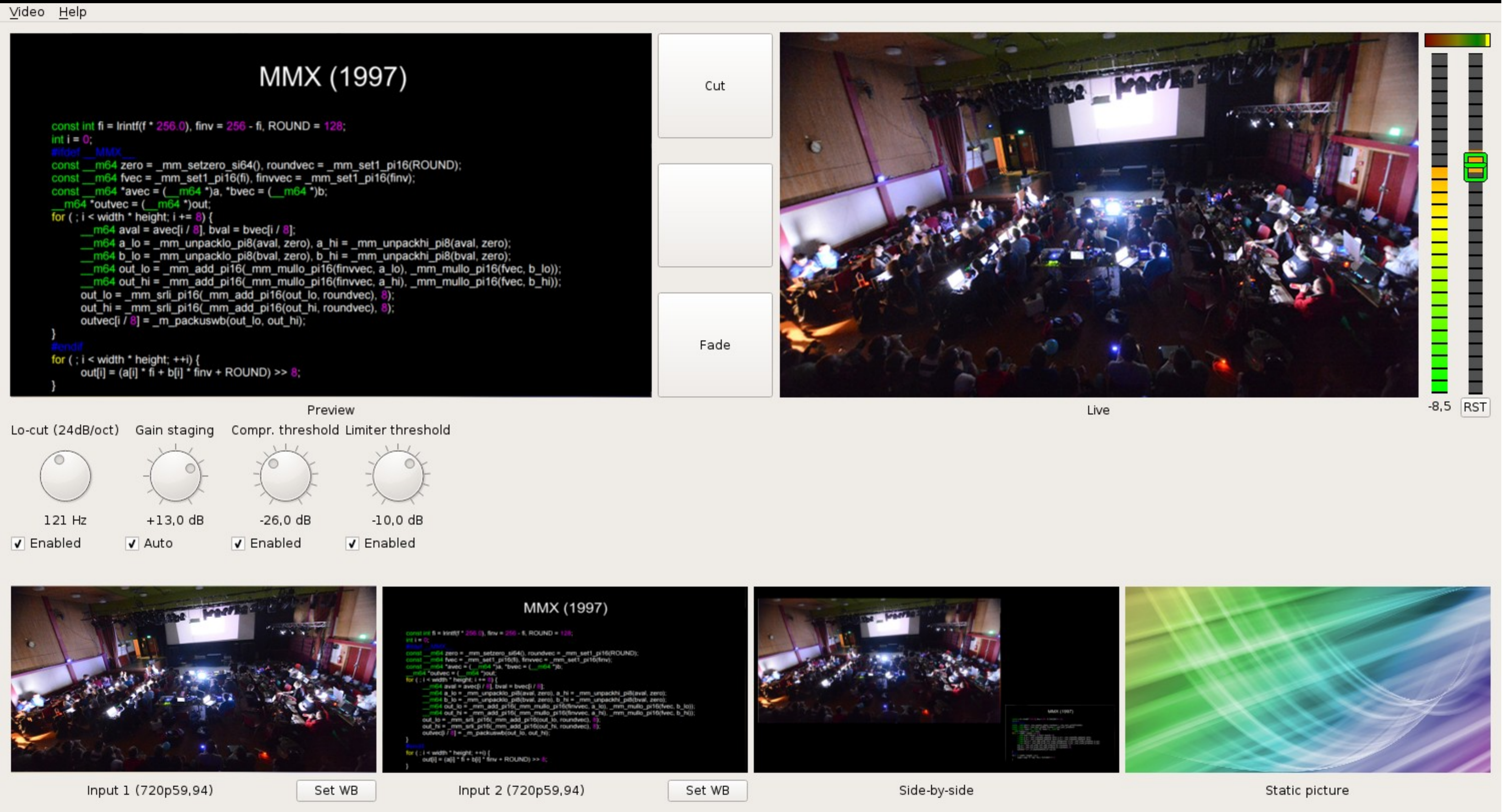
MMX (1997)

```
const int fi = lrintf(f * 256.0), finv = 256 - fi, ROUND = 128;
int i = 0;
#ifdef __MMX__
const __m64 zero = _mm_setzero_si64(), roundvec = _mm_set1_pi16(ROUND);
const __m64 fvec = _mm_set1_pi16(fi), finvvec = _mm_set1_pi16(finv);
const __m64 *avec = (__m64 *)a, *bvec = (__m64 *)b;
__m64 *outvec = (__m64 *)out;
for ( ; i < width * height; i += 8) {
        __m64 aval = avec[i / 8], bval = bvec[i / 8];
        __m64 a_lo = _mm_unpacklo_pi8(aval, zero), a_hi = _mm_unpackhi_pi8(aval, zero);
        __m64 b_lo = _mm_unpacklo_pi8(bval, zero), b_hi = _mm_unpackhi_pi8(bval, zero);
        __m64 out_lo = _mm_add_pi16(_mm_mullo_pi16(finvvec, a_lo), _mm_mullo_pi16(fvec, b_lo));
        __m64 out_hi = _mm_add_pi16(_mm_mullo_pi16(finvvec, a_hi), _mm_mullo_pi16(fvec, b_hi));
        out_lo = _mm_srli_pi16(_mm_add_pi16(out_lo, roundvec), 8);
        out_hi = _mm_srli_pi16(_mm_add_pi16(out_hi, roundvec), 8);
        outvec[i / 8] = _m_packuswb(out_lo, out_hi);
}
#endif
for ( ; i < width * height; ++i) {
        out[i] = (a[i] * fi + b[i] * finv + ROUND) >> 8;
}
```

Lo-cut (24dB/oct): 121 Hz ☑ Enabled

Gain staging: +13,0 dB ☑ Auto

Compr. threshold: -26,0 dB ☑ Enabled



MMX (1997)

```
const int fi = lrintf(f * 256.0), finv = 256 - fi, ROUND = 128;
int i = 0;
#ifdef __MMX__
const __m64 zero = _mm_setzero_si64(), roundvec = _mm_set1_pi16(ROUND);
const __m64 fvec = _mm_set1_pi16(fi), finvvec = _mm_set1_pi16(finv);
const __m64 *avec = (__m64 *)a, *bvec = (__m64 *)b;
__m64 *outvec = (__m64 *)out;
for ( ; i < width * height; i += 8) {
        __m64 aval = avec[i / 8], bval = bvec[i / 8];
        __m64 a_lo = _mm_unpacklo_pi8(aval, zero), a_hi = _mm_unpackhi_pi8(aval, zero);
        __m64 b_lo = _mm_unpacklo_pi8(bval, zero), b_hi = _mm_unpackhi_pi8(bval, zero);
        __m64 out_lo = _mm_add_pi16(_mm_mullo_pi16(finvvec, a_lo), _mm_mullo_pi16(fvec, b_lo));
        __m64 out_hi = _mm_add_pi16(_mm_mullo_pi16(finvvec, a_hi), _mm_mullo_pi16(fvec, b_hi));
        out_lo = _mm_srli_pi16(_mm_add_pi16(out_lo, roundvec), 8);
        out_hi = _mm_srli_pi16(_mm_add_pi16(out_hi, roundvec), 8);
        outvec[i / 8] = _m_packuswb(out_lo, out_hi);
}
#endif
for ( ; i < width * height; ++i) {
        out[i] = (a[i] * fi + b[i] * finv + ROUND) >> 8;
}
```

Lo-cut (24dB/oct): 121 Hz ☑ Enabled

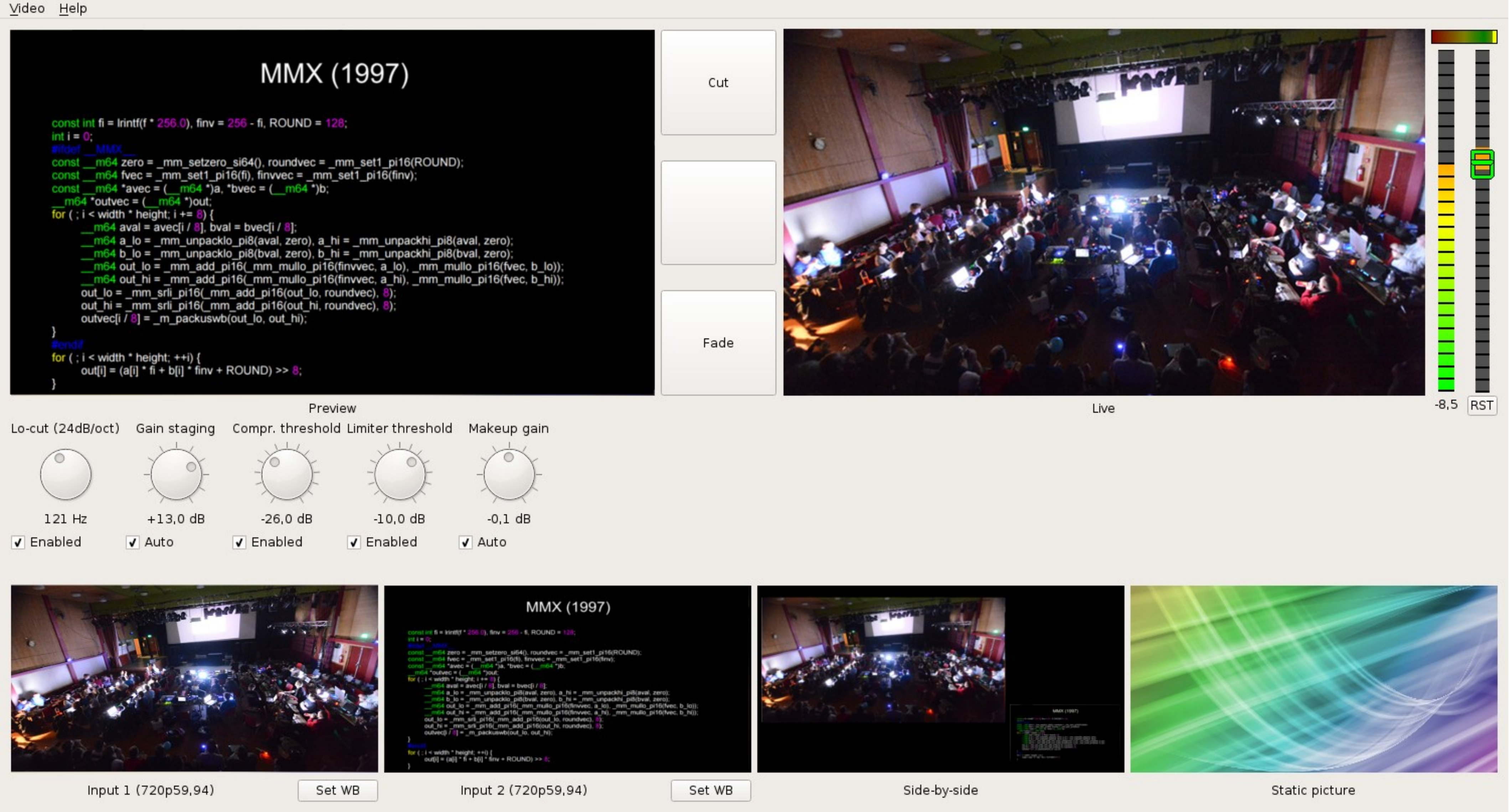
Gain staging: +13,0 dB ☑ Auto

Compr. threshold: -26,0 dB ☑ Enabled

Limiter threshold: -10,0 dB ☑ Enabled

Input 1 (720p59,94)

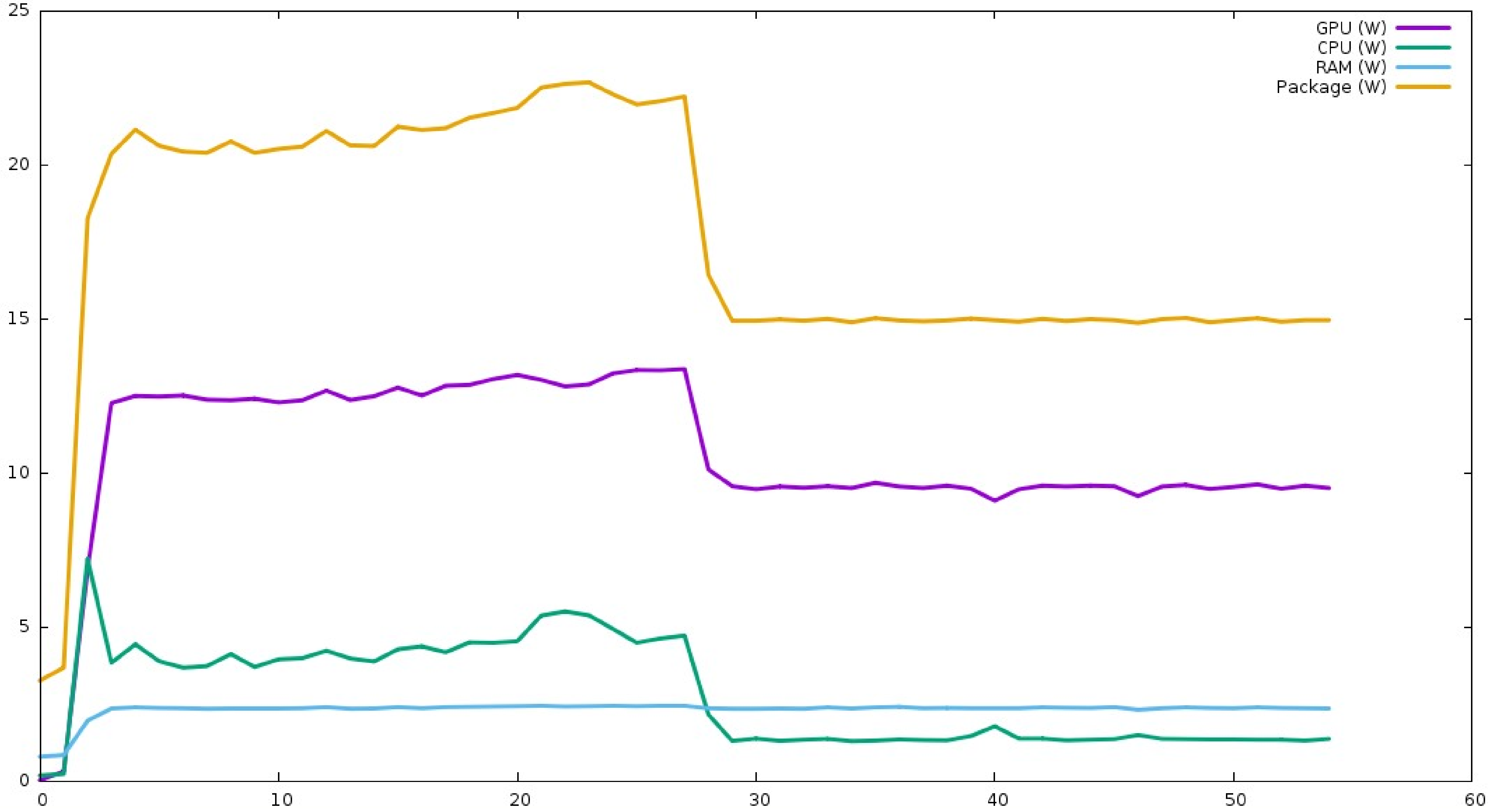
Input 2 (720p59,94)



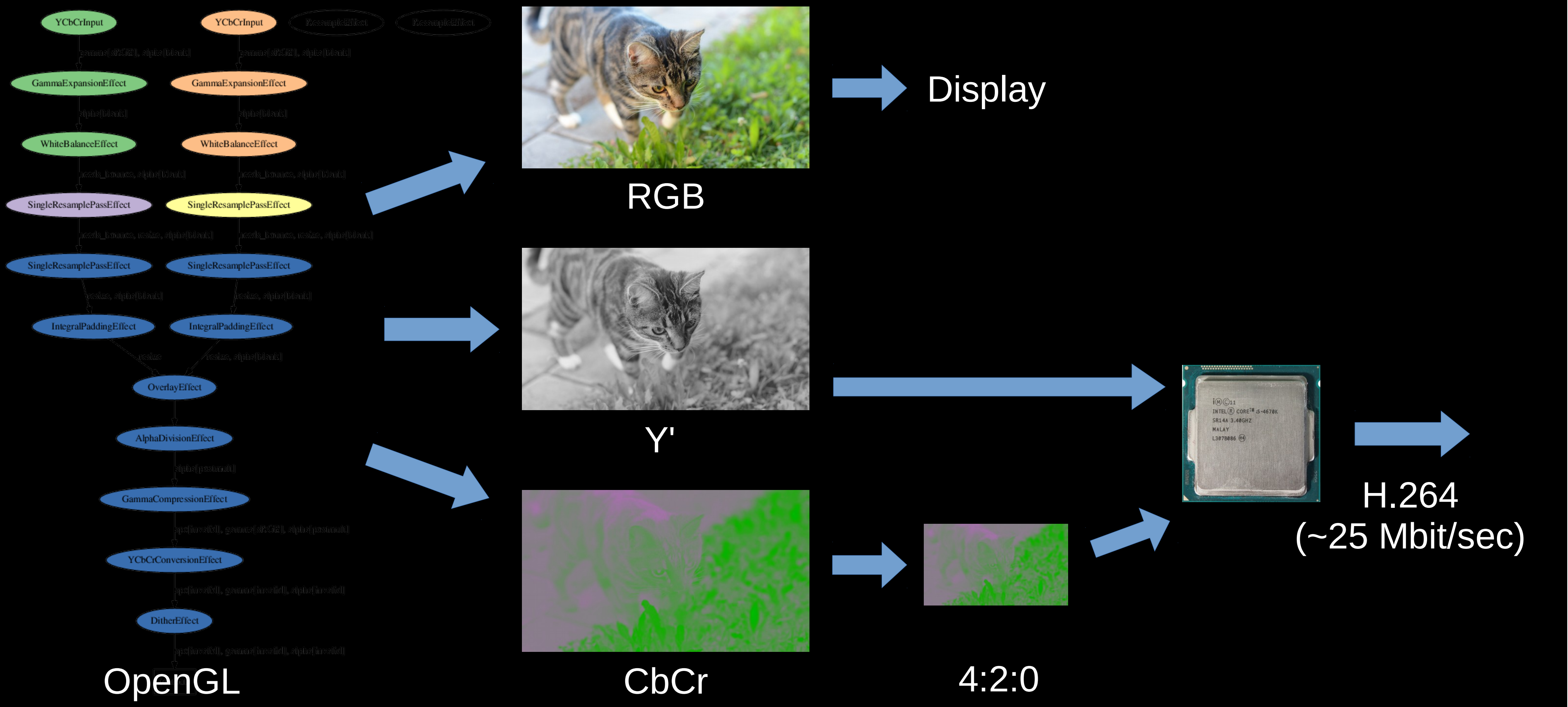
MMX (1997)

```
const int fi = lrintf(f * 256.0), finv = 256 - fi, ROUND = 128;
int i = 0;
#ifdef __MMX__
const __m64 zero = _mm_setzero_si64(), roundvec = _mm_set1_pi16(ROUND);
const __m64 fvec = _mm_set1_pi16(fi), finvvec = _mm_set1_pi16(finv);
const __m64 *avec = (__m64 *)a, *bvec = (__m64 *)b;
__m64 *outvec = (__m64 *)out;
for ( ; i < width * height; i += 8) {
        __m64 aval = avec[i / 8], bval = bvec[i / 8];
        __m64 a_lo = _mm_unpacklo_pi8(aval, zero), a_hi = _mm_unpackhi_pi8(aval, zero);
        __m64 b_lo = _mm_unpacklo_pi8(bval, zero), b_hi = _mm_unpackhi_pi8(bval, zero);
        __m64 out_lo = _mm_add_pi16(_mm_mullo_pi16(finvvec, a_lo), _mm_mullo_pi16(fvec, b_lo));
        __m64 out_hi = _mm_add_pi16(_mm_mullo_pi16(finvvec, a_hi), _mm_mullo_pi16(fvec, b_hi));
        out_lo = _mm_srli_pi16(_mm_add_pi16(out_lo, roundvec), 8);
        out_hi = _mm_srli_pi16(_mm_add_pi16(out_hi, roundvec), 8);
        outvec[i / 8] = _m_packuswb(out_lo, out_hi);
}
#endif
for ( ; i < width * height; ++i) {
        out[i] = (a[i] * fi + b[i] * finv + ROUND) >> 8;
}
```

Lo-cut (24dB/oct)	Gain staging	Compr. threshold	Limiter threshold	Makeup gain
121 Hz	+13,0 dB	-26,0 dB	-10,0 dB	-0,1 dB
☑ Enabled	☑ Auto	☑ Enabled	☑ Enabled	☑ Auto


GPU (W)
CPU (W)
RAM (W)
Package (W)
0
5
10
15
20
25
0
10
20
30
40
50
60

From: Steinar H. Gunderson
Subject: [PATCH] Add SSE2 support to zita-resampler



OpenGL

RGB

Display

Y'

CbCr

4:2:0

H.264
(~25 Mbit/sec)

Chip picture by Kolpar1, CC-BY-SA 3.0

Live demo!

Thank you!

(Q&A if people have not run out of patience yet)

https://nageru.sesse.net/