A Replay Approach to Software Validation Using Regular Expressions in C++11 and Python

James Pascoe

james@jamespascoe.com

www.jamespascoe.com

This is a story in two halves …

1. A ‘replay’ approach to software validation
    - Replay in what sense?
    - Why is this useful?
    - Replay for The Android Hardware Composer

2. Why I love Regular Expressions and why you should too
    - When (and when not) to use them
    - Regular expressions in C++11 and Python
    - Building a Replay Tool

Replay

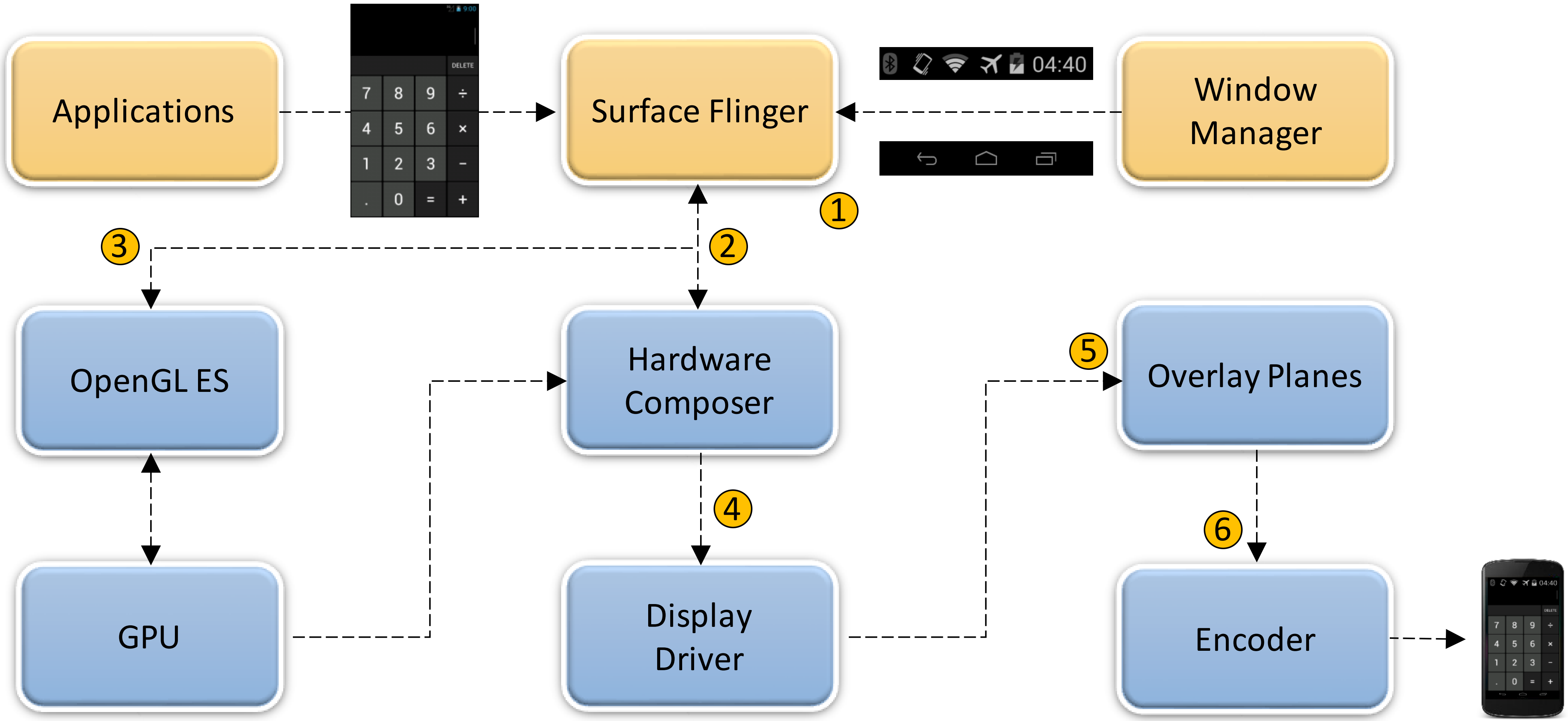
- Most software produces some form of textual logging:
  - Readable by humans (often for debug)
  - Already in use by customers, validation engineers, triage teams
  - Ideal for further processing …

- Instead of viewing logs purely as output, we can:
  1. Use C++11 Regular Expressions to parse them
  2. Interpret time-stamps to recreate event timing
  3. Generate the original input stimuli
  4. Call the underlying API at the 'correct' times

Replay Benefits

- Reduces the length of time for validation cycles
- Recreates bugs faster (and deterministically)
- Much easier to share experimental setups
- More effective bug triage
- Add automated test cases quickly
- Generate stimulus cases for benchmarking
- Create pathological test-cases
- Support TDD/BDD development activities

Android Surface Composition



Applications

Surface Flinger

Window Manager

OpenGL ES

Hardware Composer

Overlay Planes

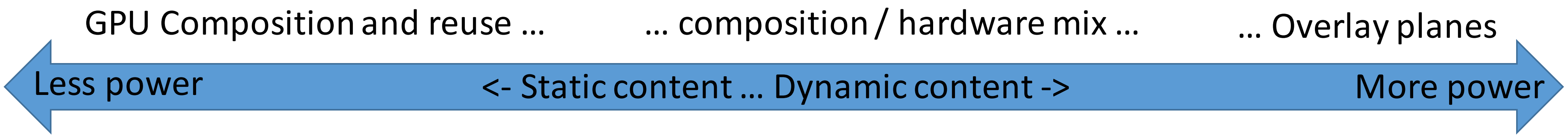
GPU

Display Driver

Encoder

The Hardware Composer

- Complex, highly dynamic, multi-threaded power optimisation engine
- Called by SurfaceFlinger. Determines which ‘layers’ should be:
  1. Composed in software (GPU / OpenGL)
  2. Mapped to hardware directly



GPU Composition and reuse … … composition / hardware mix … … Overlay planes

Less power <- Static content … Dynamic content -> More power

- Next buffer has to be ready in 16ms (i.e. 60hz refresh):
  - All HWC decision making, compositions, surface flinger, buffer writes …

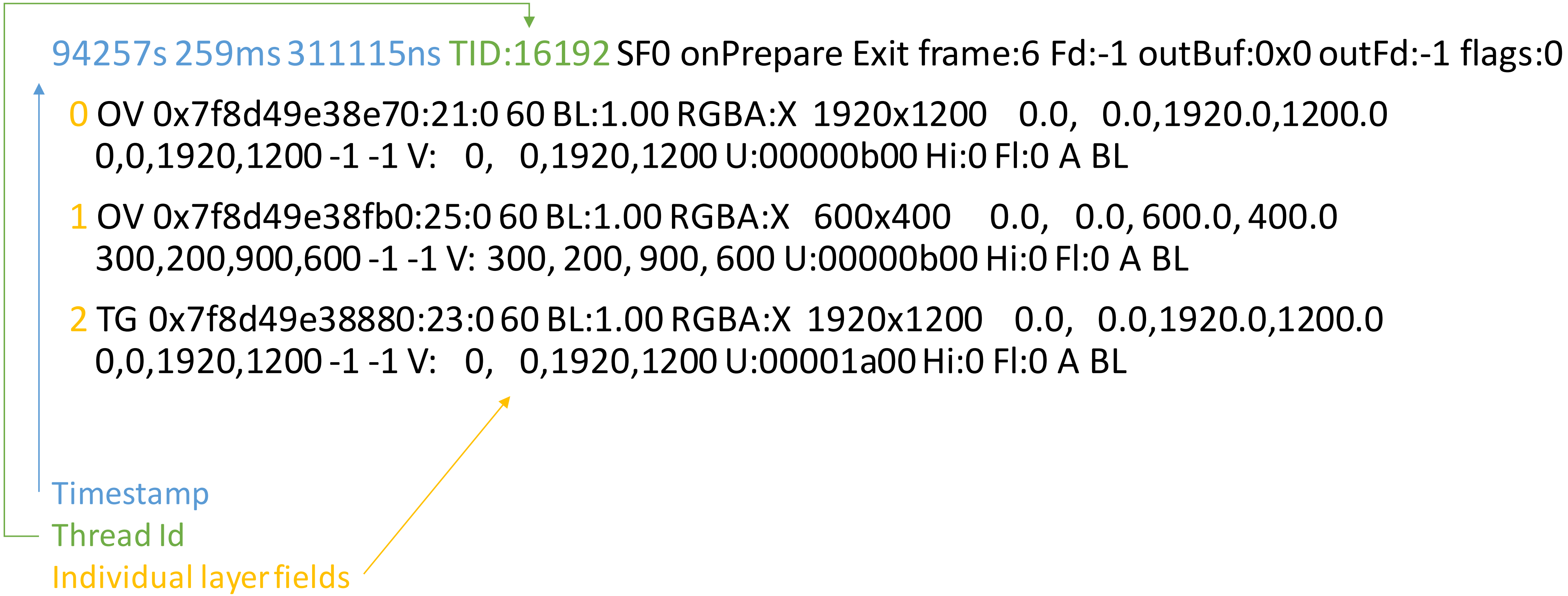
Hardware Composer API

```
/* (*prepare)() is called for each frame before composition
and is used by SurfaceFlinger to detect the compositions that
the HWC can handle. */
int (*prepare)(struct hwc_composer_device_1 *dev,
  size_t numDisplays, hwc_display_contents_1_t** displays);


/* (*set)() is called to update the displays with the content
of their work lists. */
int (*set)(struct hwc_composer_device_1 *dev,
  size_t numDisplays, hwc_display_contents_1_t** displays);
```

https://github.com/android/platform_hardware_libhardware/blob/master/include/hardware/hwcomposer.h

Hardware Composer Log File Extract



```
94257s 259ms 311115ns TID:16192 SF0 onPrepare Exit frame:6 Fd:-1 outBuf:0x0 outFd:-1 flags:0
 0 OV 0x7f8d49e38e70:21:0 60 BL:1.00 RGBA:X  1920x1200    0.0,   0.0,1920.0,1200.0
   0,0,1920,1200 -1 -1 V:   0,   0,1920,1200 U:00000b00 Hi:0 Fl:0 A BL
 1 OV 0x7f8d49e38fb0:25:0 60 BL:1.00 RGBA:X   600x400     0.0,   0.0, 600.0, 400.0
   300,200,900,600 -1 -1 V: 300, 200, 900, 600 U:00000b00 Hi:0 Fl:0 A BL
 2 TG 0x7f8d49e38880:23:0 60 BL:1.00 RGBA:X  1920x1200    0.0,   0.0,1920.0,1200.0
   0,0,1920,1200 -1 -1 V:   0,   0,1920,1200 U:00001a00 Hi:0 Fl:0 A BL
```

Timestamp

Thread Id

Individual layer fields

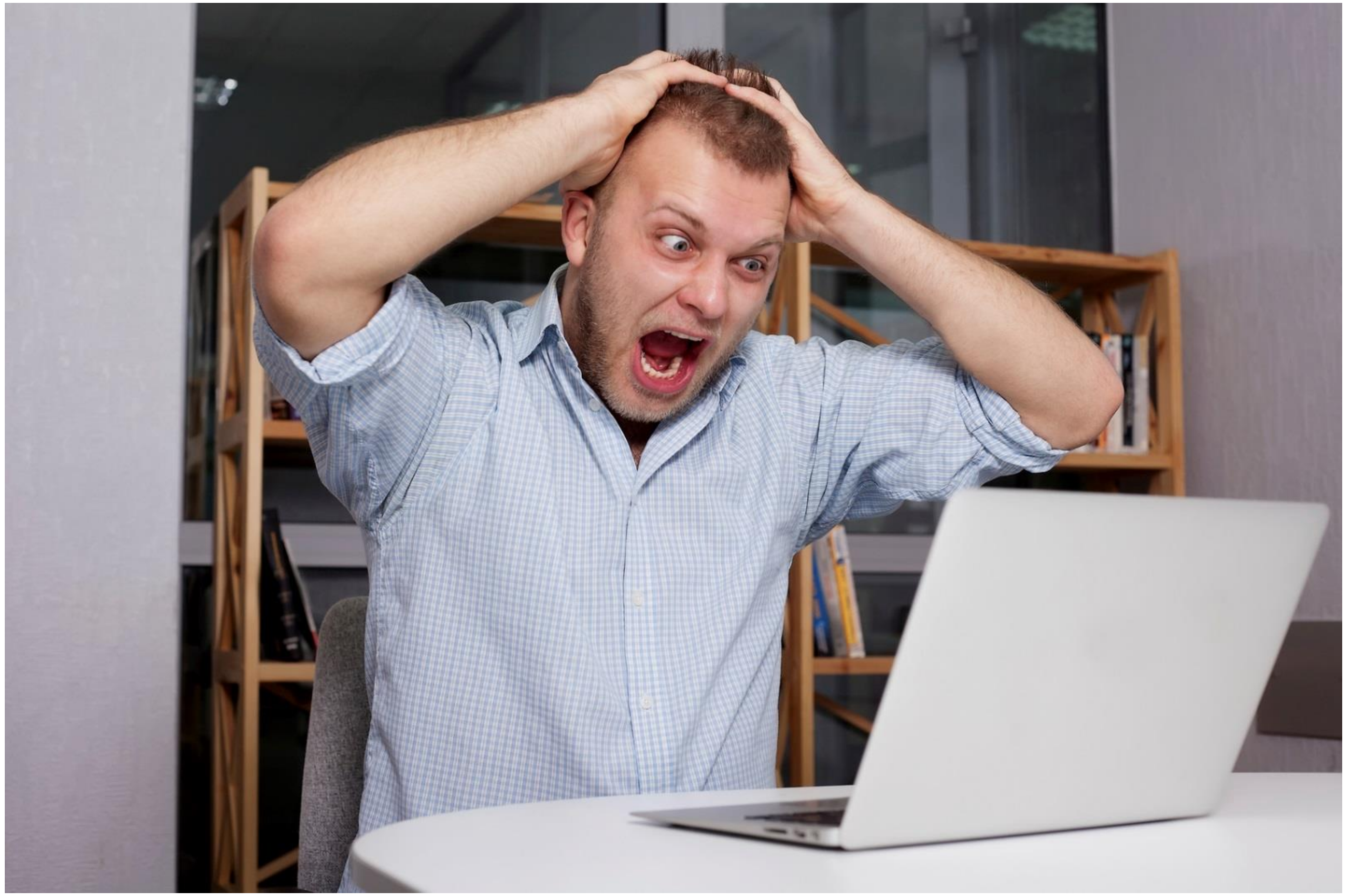
Parsing an HWC Log File – Incumbent Solution

```
bool Parse(std::string const& str)
{
  int layer_num;
  char layer_flag[3], layer_handle[13];

  int num_parsed = std::sscanf(str.c_str(), "  %d %s 0x%s:",
     &layer_num, layer_flag, layer_handle);
  int layer_handle_num = atoi(layer_handle);

  if (num_parsed == 3)
  { …
```

I know … I know …



Drawbacks

- No atomicity
  - The code breaks (subtly) when the log format changes (and it does often)
  - Code is always ‘slightly’ broken – no confidence in results
- Verbosity
  - `sscanf` format string is not flexible enough to cover all variations
  - Lots of duplication - required over 3 kloc just for parsing !
- Fragmentation
  - Number of leaf functions is huge !
  - All require unit-tests, causes code bloat, overall logic is obscured …

What we need is …

Std::regex!

Regular Expression History

- Started in the 1940s with two Neurophysiologists:[1]
  - Warren McCulloch and Walter Pitts
- Stephen Kleene described these models in algebra - ‘regular sets’
  - Kleene devised a notation to express these sets – ‘regular expressions’
- In 1968, Ken Thompson wrote a regular expression compiler that produced IBM 7094 object code:
  - This led to work on `qed` - the editor which became `ed` on Unix
  - `ed` had a command to display lines of a file that matched a regular expression
  - `g/Regular Expression/p` was read ‘Global Regular Expression Print’ and became `grep` (which was later extended into `egrep`)

[1] Mastering Regular Expressions, 3rd Edition, Jeffrey E.F. Friedl, O'Reilly, 2006.

Regular Expressions in C++11

- Full match, any match and replacement:
  - `regex_match(), regex_search(), regex_replace()`

- Iterate over matches / tokenize:
  - `regex_iterator(), regex_token_iterator()`

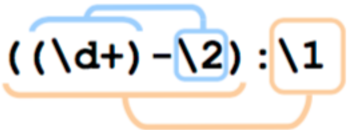
- Constants (defined in `std::regex_constants`):
  - Syntax options – case insensitivity, which regular expression grammar …
  - Match options – defines whether the first character matches ^ …
  - Error types – thrown when parsing badly formed regular expressions

C++11 ECMAScript Regex Refresher

.	Match any character
^	Match beginning of input
$	Match end of input
\b	Match word boundary
\B	Match anything other than a word boundary
|	Or operator

Capture groups

Denoted with parentheses
Referred to as \1, \2 etc.
Counted in order of left parentheses:



```
((\d+)-\2):\1
```

Repetition

Symbol	Repeats matched
?	<= 1
*	>= 0
+	>= 1
{n}	n
{n,}	>= n
{n, m}	>= n && <= m

Classes

alpha
digit or d
alnum or w

space or s
blank
cntrl

punct
lower
upper
graph
print
xdigit

Sets

Symbol	Matches
[abc]	Any of the characters included
[^abc]	Any of the characters NOT included
[a-z]	Any characters in the range
[a-zA-Z]	Any characters in the ranges
[=c=]	Equivalence class for the character
[.ae.]	Specified collating element

cpprocks.com

Regular Expressions for Replay

```
94257s 166ms 673083ns TID:16192 SF0 onSet Entry frame:0 Fd:-1 outBuf:0x0 outFd:-1 flags:1
```

```
const std::string onset_string =
    R"REGEX(^(\d+)s (\d+)ms (?:(\d+)ns )?(?:TID:(\d+) )?SF(\d) onSet Entry )REGEX"
    R"REGEX((?:frame:(\d+) )?Fd:(-?\d{1,2}) )REGEX"
    R"REGEX(outBuf:0x(.{1,8}) outFd:(-?\d{1,2}) [fF]lags:(\d+)(:?.*)$)REGEX";
```

Regular Expressions for Replay

```
 1 OV 0x7f8d49e38f60:22:0 60 BL:1.00 RGBA:X   600x400     0.0,   0.0, 600.0, 400.0
300, 200, 900, 600 -1 -1 V: 300, 200, 900, 600 U:00000b00 Hi:0 Fl:0 A BL
```

```
const std::string layer_string_hdr =
    "^\\s*(\\d+) (\\w{2}) *0x(.{1,12}): ?(?:--|(\\d{1,3})): ?(\\d) ?(\\d+) "
    "(\\w{2}): ?(.{1,4}) (\\w{1,5}) *:[XLY] *(\\d{1,4})x(\\d{1,4}) * "
    " *(-?\\d+\\.?\\d*), *(-?\\d+\\.?\\d*), *(-?\\d+\\.?\\d*), *(-?\\d+\\.?\\d*)"
    " *(-?\\d{1,4}), *(-?\\d{1,4}), *(-?\\d{1,4}), *(-?\\d{1,4}) "
    "(-?\\d+) (-?\\d+) V: *(\\d{1,4}), *(\\d{1,4}), *(\\d{1,4}), *(\\d{1,4}) ";

const std::string layer_string_trl =
    " *U:(.{1,8}) * Hi:(\\d+)(:?[[:alpha:]]*)? Fl:(\\d+)(:?[ [:alnum:]]*).*";
```

Matching and Searching

```
std::regex onset_regex(onset_string);

. . .

auto start = high_resolution_clock::now();

while (getline(infile, line))
{
  ++lines;

  if (std::regex_match(line, onset_regex) ||

     (std::regex_search(line, layer_hdr_regex) && std::regex_search(line, layer_trl_regex)))

        ++matches;

}

auto end = high_resolution_clock::now();


std::cout << "Took " << duration_cast<milliseconds>(end-start).count() << " ms for " << matches <<

  " matches (" << lines << " lines processed)" << std::endl;
```

- ➢ Took 2906 ms for 2605 matches over 7269 lines (Averaged over 10,000 runs) ~ 0.4 ms per line

Clang++-3.8 running on an I7 / 32 Gb (Ubuntu 14.04)

Capture Groups

Values are held in specialisations of `std::match_results`:

```
std::smatch onset_matches; // String specialization of std::match_results
if (std::regex_match(line, onset_matches, onset_regex))
{
  std::cout << "Saw onSet Entry with timestamp " <<
      onset_matches[1] << "s "  <<
      onset_matches[2] << "ms " <<
      std::stoi(onset_matches[3]) << "ns " << std::endl;

  // Note: onset_matches[0] is the whole match !
}
```

Advice

- Regular expressions are dense (with lots of different grammars)
  - POSIX, ECMAScript, Python, Perl, JavaScript, awk, grep, egrep
  - (C++11 regular expressions and Python `re` are compatible ☺ !)
- When writing Regular Expressions, start small and build
  - Start with: `^.*$` and not `^(?:(?:(?:0?[13578]|1[02])` ...
- Use visualisation websites:
  - regexpal.com and regexplained.co.uk
- Raw String Literals can be useful for Regexs
- Use counters to provide checks that are easy to verify with `grep`

How Do We Validate All of This?

- Question: how do you validate a Replay tool?
- Answer: compare the replayed log to the original log
- C++11 Regular Expressions and Python are compatible!
- We can develop validation / visualisation tools ☺ !

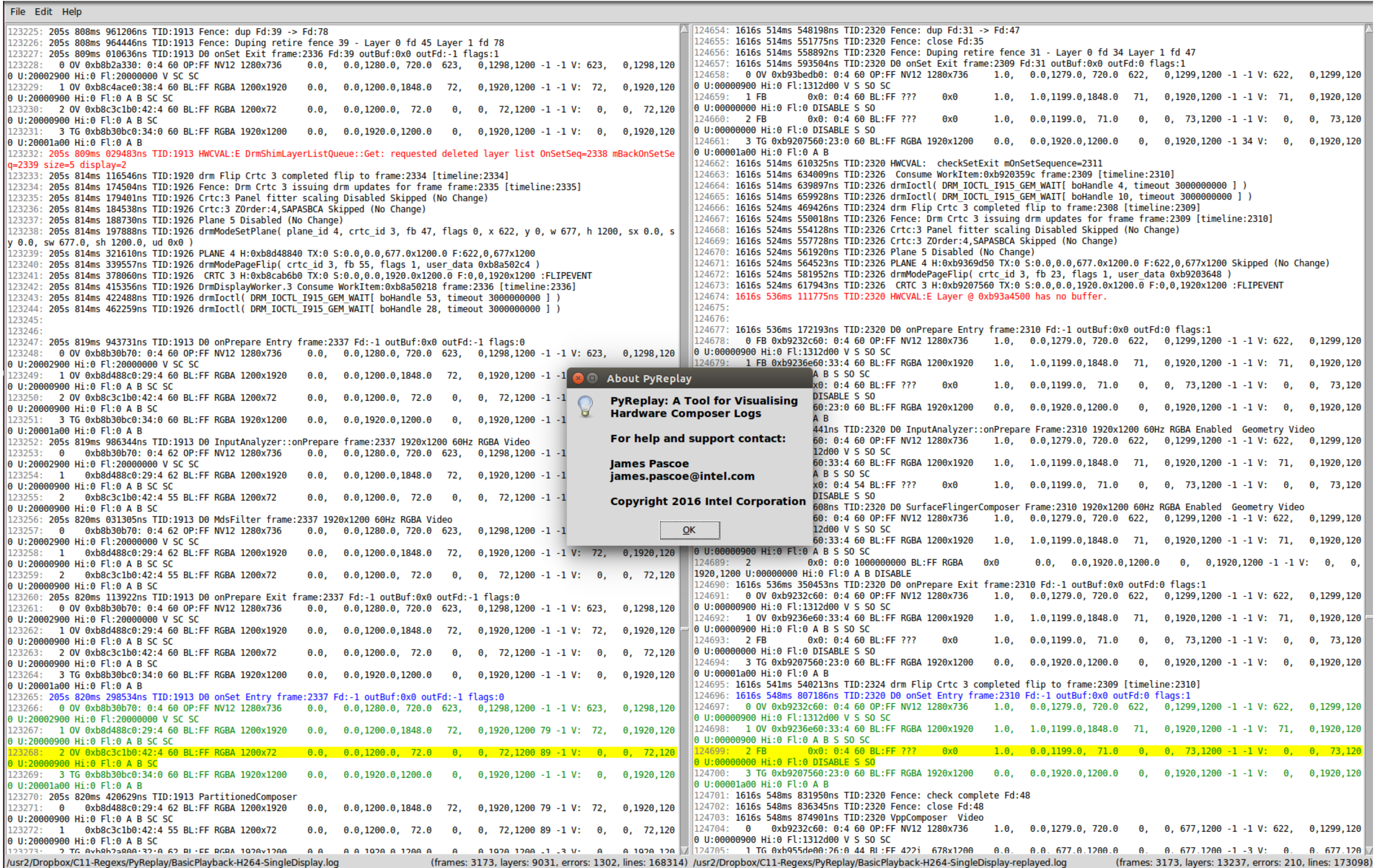
- PyReplay:
  - Compares two HWC log files. Identifies subtle mismatches
  - ~250 lines of Python (with comments). Based on TkInter
  - Python 3 regular expressions (`re`) are the same as C++11 !

Regular Expressions in Python

```
onset_string = \
     r'^(\d+)s (\d+)ms (?:(\d+)ns )?(?:TID:(\d+) )?SF(\d) onSet Entry ' \
     r'(?:frame:(\d+) )?Fd:(-?\d{1,2}) ' \
     r'outBuf:0x(.{1,8}) outFd:(-?\d{1,2}) [fF]lags:(\d+)(:?.*)$'

layer_string_hdr = \
     "^\\s*(\\d+) (\\w{2}) *0x(.{1,12}): ?(?:--|(\\d{1,3})): ?(\\d) ?(\\d+) " \
     "(\\w{2}): ?(.{1,4}) (\\w{1,5}) *:[XLY] *(\\d{1,4})x(\\d{1,4}) *" \
     " *(-?\\d+\\.?\\d*), *(-?\\d+\\.?\\d*), *(-?\\d+\\.?\\d*), *(-?\\d+\\.?\\d*)" \
     " *(-?\\d{1,4}), *(-?\\d{1,4}), *(-?\\d{1,4}), *(-?\\d{1,4}) (-?\\d+) (-?\\d+) " \
     "V: *(\\d{1,4}), *(\\d{1,4}), *(\\d{1,4}), *(\\d{1,4}) ";

layer_string_trl = \
     " *U:(.{1,8}) * Hi:(\\d+)(:?[[:alpha:]]*)? Fl:(\\d+)(:?[ [:alnum:]]*).*";
```



```
123225: 205s 808ms 961206ns TID:1913 Fence: dup Fd:39 -> Fd:78
123226: 205s 808ms 964446ns TID:1913 Fence: Duping retire fence 39 - Layer 0 fd 45 Layer 1 fd 78
123227: 205s 809ms 010636ns TID:1913 D0 onSet Exit frame:2336 Fd:39 outBuf:0x0 outFd:-1 flags:1
123228:   0 OV 0xb8b2a330: 0:4 60 OP:FF NV12 1280x736     0.0,   0.0,1280.0, 720.0  623,   0,1298,1200 -1 -1 V: 623,   0,1298,120
0 U:20002900 Hi:0 Fl:20000000 V SC SC
123229:   1 OV 0xb8c4ace0:38:4 60 BL:FF RGBA 1200x1920    0.0,   0.0,1200.0,1848.0   72,   0,1920,1200 -1 -1 V:  72,   0,1920,120
0 U:20000900 Hi:0 Fl:0 A B SC SC
123230:   2 OV 0xb8c3c1b0:42:4 60 BL:FF RGBA 1200x72      0.0,   0.0,1200.0,  72.0    0,   0,  72,1200 -1 -1 V:   0,   0,  72,120
0 U:20000900 Hi:0 Fl:0 A B SC
123231:   3 TG 0xb8b30bc0:34:0 60 BL:FF RGBA 1920x1200    0.0,   0.0,1920.0,1200.0    0,   0,1920,1200 -1 -1 V:   0,   0,1920,120
0 U:20001a00 Hi:0 Fl:0 A B
123232: 205s 809ms 029483ns TID:1913 HWCVAL:E DrmShimLayerListQueue::Get: requested deleted layer list OnSetSeq=2338 mBackOnSetSe
q=2339 size=5 display=2
123233: 205s 814ms 116546ns TID:1920 drm Flip Crtc 3 completed flip to frame:2334 [timeline:2334]
123234: 205s 814ms 174504ns TID:1926 Fence: Drm Crtc 3 issuing drm updates for frame frame:2335 [timeline:2335]
123235: 205s 814ms 179401ns TID:1926 Crtc:3 Panel fitter scaling Disabled Skipped (No Change)
123236: 205s 814ms 184538ns TID:1926 Crtc:3 ZOrder:4,SAPASBCA Skipped (No Change)
123237: 205s 814ms 188730ns TID:1926 Plane 5 Disabled (No Change)
123238: 205s 814ms 197888ns TID:1926 drmModeSetPlane( plane_id 4, crtc_id 3, fb 47, flags 0, x 622, y 0, w 677, h 1200, sx 0.0, s
y 0.0, sw 677.0, sh 1200.0, ud 0x0 )
123239: 205s 814ms 321610ns TID:1926 PLANE 4 H:0xb8d48840 TX:0 S:0.0,0.0,677.0x1200.0 F:622,0,677x1200
123240: 205s 814ms 339557ns TID:1926 drmModePageFlip( crtc_id 3, fb 55, flags 1, user_data 0xb8a502c4 )
123241: 205s 814ms 378060ns TID:1926  CRTC 3 H:0xb8cab6b0 TX:0 S:0.0,0.0,1920.0x1200.0 F:0,0,1920x1200 :FLIPEVENT
123242: 205s 814ms 415356ns TID:1926 DrmDisplayWorker.3 Consume WorkItem:0xb8a50218 frame:2336 [timeline:2336]
123243: 205s 814ms 422488ns TID:1926 drmIoctl( DRM_IOCTL_I915_GEM_WAIT[ boHandle 53, timeout 3000000000 ] )
123244: 205s 814ms 462259ns TID:1926 drmIoctl( DRM_IOCTL_I915_GEM_WAIT[ boHandle 28, timeout 3000000000 ] )
123245:
123246:
123247: 205s 819ms 943731ns TID:1913 D0 onPrepare Entry frame:2337 Fd:-1 outBuf:0x0 outFd:-1 flags:0
123248:   0 OV 0xb8b30b70: 0:4 60 OP:FF NV12 1280x736     0.0,   0.0,1280.0, 720.0  623,   0,1298,1200 -1 -1 V: 623,   0,1298,120
0 U:20002900 Hi:0 Fl:20000000 V SC SC
123249:   1 OV 0xb8d488c0:29:4 60 BL:FF RGBA 1200x1920    0.0,   0.0,1200.0,1848.0   72,   0,1920,1200 -1 -1
0 U:20000900 Hi:0 Fl:0 A B SC SC
123250:   2 OV 0xb8c3c1b0:42:4 60 BL:FF RGBA 1200x72      0.0,   0.0,1200.0,  72.0    0,   0,  72,1200 -1 -1
0 U:20000900 Hi:0 Fl:0 A B SC
123251:   3 TG 0xb8b30bc0:34:0 60 BL:FF RGBA 1920x1200    0.0,   0.0,1920.0,1200.0    0,   0,1920,1200 -1 -1
0 U:20001a00 Hi:0 Fl:0 A B
123252: 205s 819ms 986344ns TID:1913 D0 InputAnalyzer::onPrepare frame:2337 1920x1200 60Hz RGBA Video
123253:   0    0xb8b30b70: 0:4 62 OP:FF NV12 1280x736     0.0,   0.0,1280.0, 720.0  623,   0,1298,1200 -1 -1
0 U:20002900 Hi:0 Fl:20000000 V SC SC
123254:   1    0xb8d488c0:29:4 62 BL:FF RGBA 1200x1920    0.0,   0.0,1200.0,1848.0   72,   0,1920,1200 -1 -1
0 U:20000900 Hi:0 Fl:0 A B SC SC
123255:   2    0xb8c3c1b0:42:4 55 BL:FF RGBA 1200x72      0.0,   0.0,1200.0,  72.0    0,   0,  72,1200 -1 -1
0 U:20000900 Hi:0 Fl:0 A B SC
123256: 205s 820ms 031305ns TID:1913 D0 MdsFilter frame:2337 1920x1200 60Hz RGBA Video
123257:   0    0xb8b30b70: 0:4 62 OP:FF NV12 1280x736     0.0,   0.0,1280.0, 720.0  623,   0,1298,1200 -1 -1
0 U:20002900 Hi:0 Fl:20000000 V SC SC
123258:   1    0xb8d488c0:29:4 62 BL:FF RGBA 1200x1920    0.0,   0.0,1200.0,1848.0   72,   0,1920,1200 -1 -1 V:  72,   0,1920,120
0 U:20000900 Hi:0 Fl:0 A B SC SC
123259:   2    0xb8c3c1b0:42:4 55 BL:FF RGBA 1200x72      0.0,   0.0,1200.0,  72.0    0,   0,  72,1200 -1 -1 V:   0,   0,  72,120
0 U:20000900 Hi:0 Fl:0 A B SC
123260: 205s 820ms 113922ns TID:1913 D0 onPrepare Exit frame:2337 Fd:-1 outBuf:0x0 outFd:-1 flags:0
123261:   0 OV 0xb8b30b70: 0:4 60 OP:FF NV12 1280x736     0.0,   0.0,1280.0, 720.0  623,   0,1298,1200 -1 -1 V: 623,   0,1298,120
0 U:20002900 Hi:0 Fl:20000000 V SC SC
123262:   1 OV 0xb8d488c0:29:4 60 BL:FF RGBA 1200x1920    0.0,   0.0,1200.0,1848.0   72,   0,1920,1200 -1 -1 V:  72,   0,1920,120
0 U:20000900 Hi:0 Fl:0 A B SC SC
123263:   2 OV 0xb8c3c1b0:42:4 60 BL:FF RGBA 1200x72      0.0,   0.0,1200.0,  72.0    0,   0,  72,1200 -1 -1 V:   0,   0,  72,120
0 U:20000900 Hi:0 Fl:0 A B SC
123264:   3 TG 0xb8b30bc0:34:0 60 BL:FF RGBA 1920x1200    0.0,   0.0,1920.0,1200.0    0,   0,1920,1200 -1 -1 V:   0,   0,1920,120
0 U:20001a00 Hi:0 Fl:0 A B
123265: 205s 820ms 298534ns TID:1913 D0 onSet Entry frame:2337 Fd:-1 outBuf:0x0 outFd:-1 flags:0
123266:   0 OV 0xb8b30b70: 0:4 60 OP:FF NV12 1280x736     0.0,   0.0,1280.0, 720.0  623,   0,1298,1200 -1 -1 V: 623,   0,1298,120
0 U:20002900 Hi:0 Fl:20000000 V SC SC
123267:   1 OV 0xb8d488c0:29:4 60 BL:FF RGBA 1200x1920    0.0,   0.0,1200.0,1848.0   72,   0,1920,1200 79 -1 V:  72,   0,1920,120
0 U:20000900 Hi:0 Fl:0 A B SC SC
123268:   2 OV 0xb8c3c1b0:42:4 60 BL:FF RGBA 1200x72      0.0,   0.0,1200.0,  72.0    0,   0,  72,1200 89 -1 V:   0,   0,  72,120
0 U:20000900 Hi:0 Fl:0 A B SC
123269:   3 TG 0xb8b30bc0:34:0 60 BL:FF RGBA 1920x1200    0.0,   0.0,1920.0,1200.0    0,   0,1920,1200 -1 -1 V:   0,   0,1920,120
0 U:20001a00 Hi:0 Fl:0 A B
123270: 205s 820ms 420629ns TID:1913 PartitionedComposer
123271:   0    0xb8d488c0:29:4 62 BL:FF RGBA 1200x1920    0.0,   0.0,1200.0,1848.0   72,   0,1920,1200 79 -1 V:  72,   0,1920,120
0 U:20000900 Hi:0 Fl:0 A B SC SC
123272:   1    0xb8c3c1b0:42:4 55 BL:FF RGBA 1200x72      0.0,   0.0,1200.0,  72.0    0,   0,  72,1200 89 -1 V:   0,   0,  72,120
0 U:20000900 Hi:0 Fl:0 A B SC
123273:   2 TG 0xb8b2a800:32:0 62 BL:FF RGBA 1920x1200    0.0,   0.0,1920.0,1200.0    0,   0,1920,1200 -1 -3 V:   0,   0,1920,120
```

/usr2/Dropbox/C11-Regexs/PyReplay/BasicPlayback-H264-SingleDisplay.log (frames: 3173, layers: 9031, errors: 1302, lines: 168314)

```
124654: 1616s 514ms 548198ns TID:2320 Fence: dup Fd:31 -> Fd:47
124655: 1616s 514ms 551775ns TID:2320 Fence: close Fd:35
124656: 1616s 514ms 558892ns TID:2320 Fence: Duping retire fence 31 - Layer 0 fd 34 Layer 1 fd 47
124657: 1616s 514ms 593504ns TID:2320 D0 onSet Exit frame:2309 Fd:31 outBuf:0x0 outFd:0 flags:1
124658:   0 OV 0xb93bedb0: 0:4 60 OP:FF NV12 1280x736     1.0,   0.0,1279.0, 720.0  622,   0,1299,1200 -1 -1 V: 622,   0,1299,120
0 U:00000900 Hi:0 Fl:1312d00 V S SO SC
124659:   1 FB        0x0: 0:4 60 BL:FF ???     0x0       1.0,   1.0,1199.0,1848.0   71,   0,1920,1200 -1 -1 V:  71,   0,1920,120
0 U:00000000 Hi:0 Fl:0 DISABLE S SO
124660:   2 FB        0x0: 0:4 60 BL:FF ???     0x0       1.0,   0.0,1199.0,  71.0    0,   0,  73,1200 -1 -1 V:   0,   0,  73,120
0 U:00000000 Hi:0 Fl:0 DISABLE S SO
124661:   3 TG 0xb9207560:23:0 60 BL:FF RGBA 1920x1200    0.0,   0.0,1920.0,1200.0    0,   0,1920,1200 -1 34 V:   0,   0,1920,120
0 U:00001a00 Hi:0 Fl:0 A B
124662: 1616s 514ms 610325ns TID:2320 HWCVAL:  checkSetExit mOnSetSequence=2311
124663: 1616s 514ms 634009ns TID:2326  Consume WorkItem:0xb920359c frame:2309 [timeline:2310]
124664: 1616s 514ms 639897ns TID:2326 drmIoctl( DRM_IOCTL_I915_GEM_WAIT[ boHandle 4, timeout 3000000000 ] )
124665: 1616s 514ms 659928ns TID:2326 drmIoctl( DRM_IOCTL_I915_GEM_WAIT[ boHandle 10, timeout 3000000000 ] )
124666: 1616s 524ms 469426ns TID:2324 drm Flip Crtc 3 completed flip to frame:2308 [timeline:2309]
124667: 1616s 524ms 550018ns TID:2326 Fence: Drm Crtc 3 issuing drm updates for frame frame:2309 [timeline:2310]
124668: 1616s 524ms 554128ns TID:2326 Crtc:3 Panel fitter scaling Disabled Skipped (No Change)
124669: 1616s 524ms 557728ns TID:2326 Crtc:3 ZOrder:4,SAPASBCA Skipped (No Change)
124670: 1616s 524ms 561920ns TID:2326 Plane 5 Disabled (No Change)
124671: 1616s 524ms 564523ns TID:2326 PLANE 4 H:0xb9369d50 TX:0 S:0.0,0.0,677.0x1200.0 F:622,0,677x1200 Skipped (No Change)
124672: 1616s 524ms 581952ns TID:2326 drmModePageFlip( crtc_id 3, fb 23, flags 1, user_data 0xb9203648 )
124673: 1616s 524ms 617943ns TID:2326  CRTC 3 H:0xb9207560 TX:0 S:0.0,0.0,1920.0x1200.0 F:0,0,1920x1200 :FLIPEVENT
124674: 1616s 536ms 111775ns TID:2320 HWCVAL:E Layer @ 0xb93a4500 has no buffer.
124675:
124676:
124677: 1616s 536ms 172193ns TID:2320 D0 onPrepare Entry frame:2310 Fd:-1 outBuf:0x0 outFd:0 flags:1
124678:   0 FB 0xb9232c60: 0:4 60 OP:FF NV12 1280x736     1.0,   0.0,1279.0, 720.0  622,   0,1299,1200 -1 -1 V: 622,   0,1299,120
0 U:00000900 Hi:0 Fl:1312d00 V S SO SC
124679:   1 FB 0xb9236e60:33:4 60 BL:FF RGBA 1200x1920    1.0,   1.0,1199.0,1848.0   71,   0,1920,1200 -1 -1 V:  71,   0,1920,120
                  A B S SO SC
                  x0: 0:4 60 BL:FF ???     0x0       1.0,   0.0,1199.0,  71.0    0,   0,  73,1200 -1 -1 V:   0,   0,  73,120
                  DISABLE S SO
                  60:23:0 60 BL:FF RGBA 1920x1200    0.0,   0.0,1920.0,1200.0    0,   0,1920,1200 -1 -1 V:   0,   0,1920,120
                  A B
                  441ns TID:2320 D0 InputAnalyzer::onPrepare Frame:2310 1920x1200 60Hz RGBA Enabled  Geometry Video
                  60: 0:4 60 OP:FF NV12 1280x736     1.0,   0.0,1279.0, 720.0  622,   0,1299,1200 -1 -1 V: 622,   0,1299,120
                  12d00 V S SO SC
                  60:33:4 60 BL:FF RGBA 1200x1920    1.0,   1.0,1199.0,1848.0   71,   0,1920,1200 -1 -1 V:  71,   0,1920,120
                  A B S SO SC
                  x0: 0:4 54 BL:FF ???     0x0       1.0,   0.0,1199.0,  71.0    0,   0,  73,1200 -1 -1 V:   0,   0,  73,120
                  DISABLE S SO
                  608ns TID:2320 D0 SurfaceFlingerComposer Frame:2310 1920x1200 60Hz RGBA Enabled  Geometry Video
                  60: 0:4 60 OP:FF NV12 1280x736     1.0,   0.0,1279.0, 720.0  622,   0,1299,1200 -1 -1 V: 622,   0,1299,120
                  12d00 V S SO SC
                  60:33:4 60 BL:FF RGBA 1200x1920    1.0,   1.0,1199.0,1848.0   71,   0,1920,1200 -1 -1 V:  71,   0,1920,120
0 U:00000900 Hi:0 Fl:0 A B S SO SC
124689:   2          0x0: 0:0 1000000000 BL:FF RGBA    0x0       0.0,   0.0,1920.0,1200.0    0,   0,1920,1200 -1 -1 V:   0,   0,
1920,1200 U:00000000 Hi:0 Fl:0 A B DISABLE
124690: 1616s 536ms 350453ns TID:2320 D0 onPrepare Exit frame:2310 Fd:-1 outBuf:0x0 outFd:0 flags:1
124691:   0 OV 0xb9232c60: 0:4 60 OP:FF NV12 1280x736     1.0,   0.0,1279.0, 720.0  622,   0,1299,1200 -1 -1 V: 622,   0,1299,120
0 U:00000900 Hi:0 Fl:1312d00 V S SO SC
124692:   1 OV 0xb9236e60:33:4 60 BL:FF RGBA 1200x1920    1.0,   1.0,1199.0,1848.0   71,   0,1920,1200 -1 -1 V:  71,   0,1920,120
0 U:00000900 Hi:0 Fl:0 A B S SO SC
124693:   2 FB        0x0: 0:4 60 BL:FF ???     0x0       1.0,   0.0,1199.0,  71.0    0,   0,  73,1200 -1 -1 V:   0,   0,  73,120
0 U:00000000 Hi:0 Fl:0 DISABLE S SO
124694:   3 TG 0xb9207560:23:0 60 BL:FF RGBA 1920x1200    0.0,   0.0,1920.0,1200.0    0,   0,1920,1200 -1 -1 V:   0,   0,1920,120
0 U:00001a00 Hi:0 Fl:0 A B
124695: 1616s 541ms 540213ns TID:2324 drm Flip Crtc 3 completed flip to frame:2309 [timeline:2310]
124696: 1616s 548ms 807186ns TID:2320 D0 onSet Entry frame:2310 Fd:-1 outBuf:0x0 outFd:0 flags:1
124697:   0 OV 0xb9232c60: 0:4 60 OP:FF NV12 1280x736     1.0,   0.0,1279.0, 720.0  622,   0,1299,1200 -1 -1 V: 622,   0,1299,120
0 U:00000900 Hi:0 Fl:1312d00 V S SO SC
124698:   1 OV 0xb9236e60:33:4 60 BL:FF RGBA 1200x1920    1.0,   1.0,1199.0,1848.0   71,   0,1920,1200 -1 -1 V:  71,   0,1920,120
0 U:00000900 Hi:0 Fl:0 A B S SO SC
124699:   2 FB        0x0: 0:4 60 BL:FF ???     0x0       1.0,   0.0,1199.0,  71.0    0,   0,  73,1200 -1 -1 V:   0,   0,  73,120
0 U:00000000 Hi:0 Fl:0 DISABLE S SO
124700:   3 TG 0xb9207560:23:0 60 BL:FF RGBA 1920x1200    0.0,   0.0,1920.0,1200.0    0,   0,1920,1200 -1 -1 V:   0,   0,1920,120
0 U:00001a00 Hi:0 Fl:0 A B
124701: 1616s 548ms 831950ns TID:2320 Fence: check complete Fd:48
124702: 1616s 548ms 836345ns TID:2320 Fence: close Fd:48
124703: 1616s 548ms 874901ns TID:2320 VppComposer  Video
124704:   0    0xb9232c60: 0:4 60 OP:FF NV12 1280x736     1.0,   0.0,1279.0, 720.0    0,   0, 677,1200 -1 -1 V: 622,   0,1299,120
0 U:00000900 Hi:0 Fl:1312d00 V S SO SC
124705:   1 TG 0xb955de00:26:0 44 BL:FF 422i  678x1200    0.0,   0.0, 677.0,1200.0    0,   0, 677,1200 -1 -3 V:   0,   0, 677,120
```

/usr2/Dropbox/C11-Regexs/PyReplay/BasicPlayback-H264-SingleDisplay-replayed.log (frames: 3173, layers: 13237, errors: 210, lines: 173098)

About PyReplay

PyReplay: A Tool for Visualising Hardware Composer Logs

For help and support contact:

James Pascoe
james.pascoe@intel.com

Copyright 2016 Intel Corporation

Conclusions

- Replay:
  - Found numerous real bugs and often very subtle
  - Had some great cultural benefits and insights
    - Particularly when combined with Jenkins / CI
  - Low-cost to implement and maintain
  - Compatibility of `std::regex` and Python `re` is very useful

- Limitations:
  - Replay: multithreading can disrupt bug reproducibility
    - Better to use fuzz-testing + coverage + sanitisers for that
  - Regular expressions: not the right tool for sophisticated lexical analysis
    - Probably want to build something like an Abstract Syntax Tree for that …

```cpp
#include <iostream>

int main()
{
  std::string questions;
  while (1)
  {
    std::cout << "Questions?" << std::endl;

    if (std::cin >> questions && questions == "Y")
      std::cout << "Answers" << std::endl;
    else
      goto cornubia;
  }

cornubia:
  std::cout << "Thank you for coming !" << std::endl;
}
```