

Developing Music Systems on the JVM with Pink and Score

Steven Yi

Clojure/Conj 2014

2014-11-20

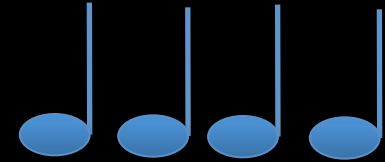
About Me

- Composer and Programmer
- PhD Candidate in Digital Arts and Humanities at Maynooth University, Ireland
- Author of Blue
- Developer of Csound

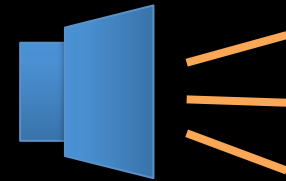


Pink and Score

Score



Pink



Clojure

Java Virtual Machine

Digital Audio

1. A sequence of numbers

||: 1 2 3 4 1 2 3 4 5 6 1 2 3 4 5 6 7 8 :||

2. Space and Time are linked

short[64] == ~0.00145 seconds at 44.1k sr

Digital Audio



External Buffer Size (i.e. 256 samples)



Internal Buffer Size (i.e. 64 samples)



Samples

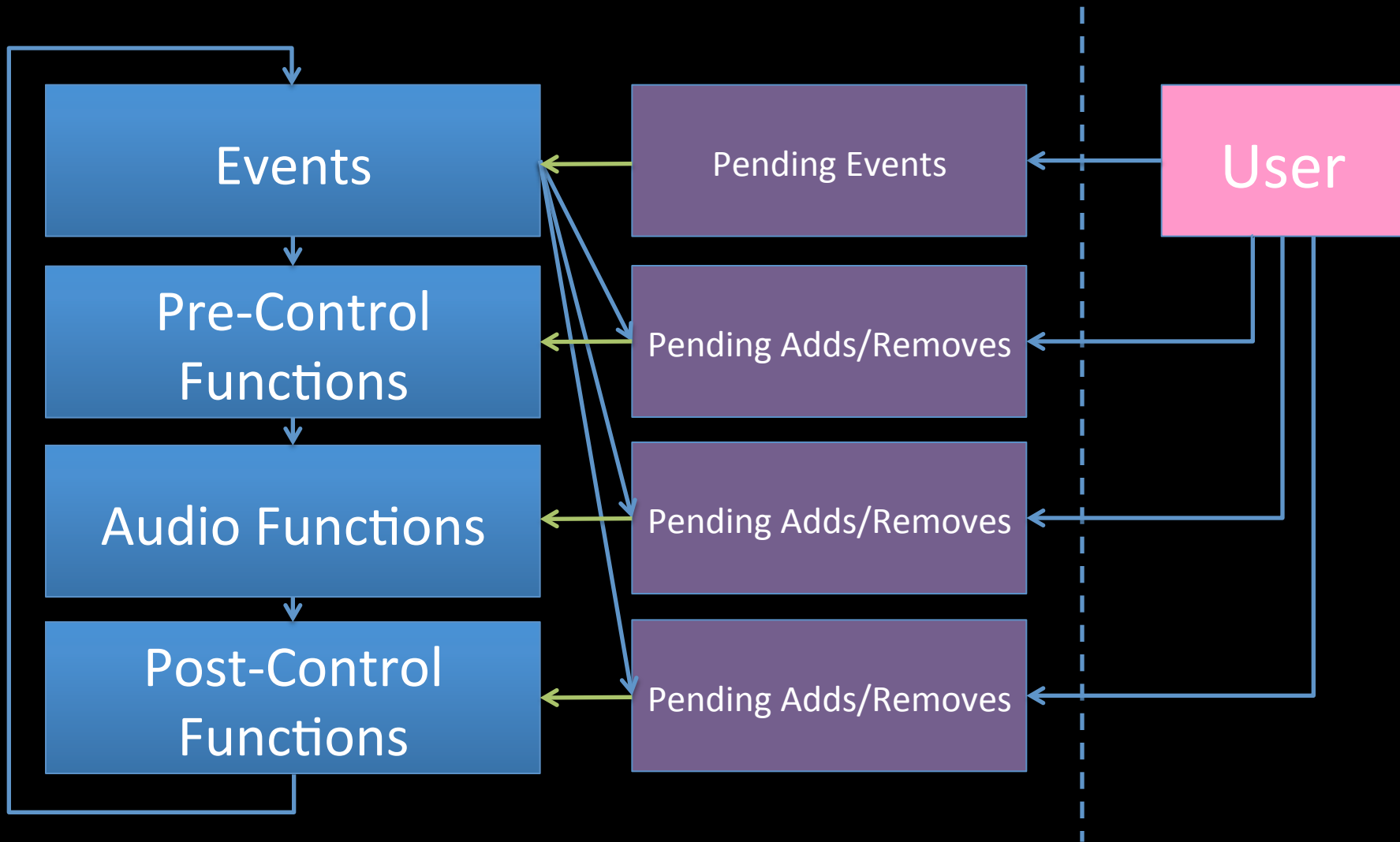
Demo – Minimal Engine

- `(start-engine) ; example1`
- `(def a (some-audio-func))`
- `(add-afunc a)`
- `(remove-afunc a)`
- `(stop-engine)`

Pink Breakdown

- Processing Fn: thread with fn with loop
- Events: Delayed Function Applications
 - Perform side effects, one-time
- Audio Functions: $fn \Rightarrow [double] \mid nil$
 - Produce Sound, Composed as DAG
- Control Functions: $fn \Rightarrow true \mid false$
 - Perform side effects, continuous

Audio Engine – Main Thread



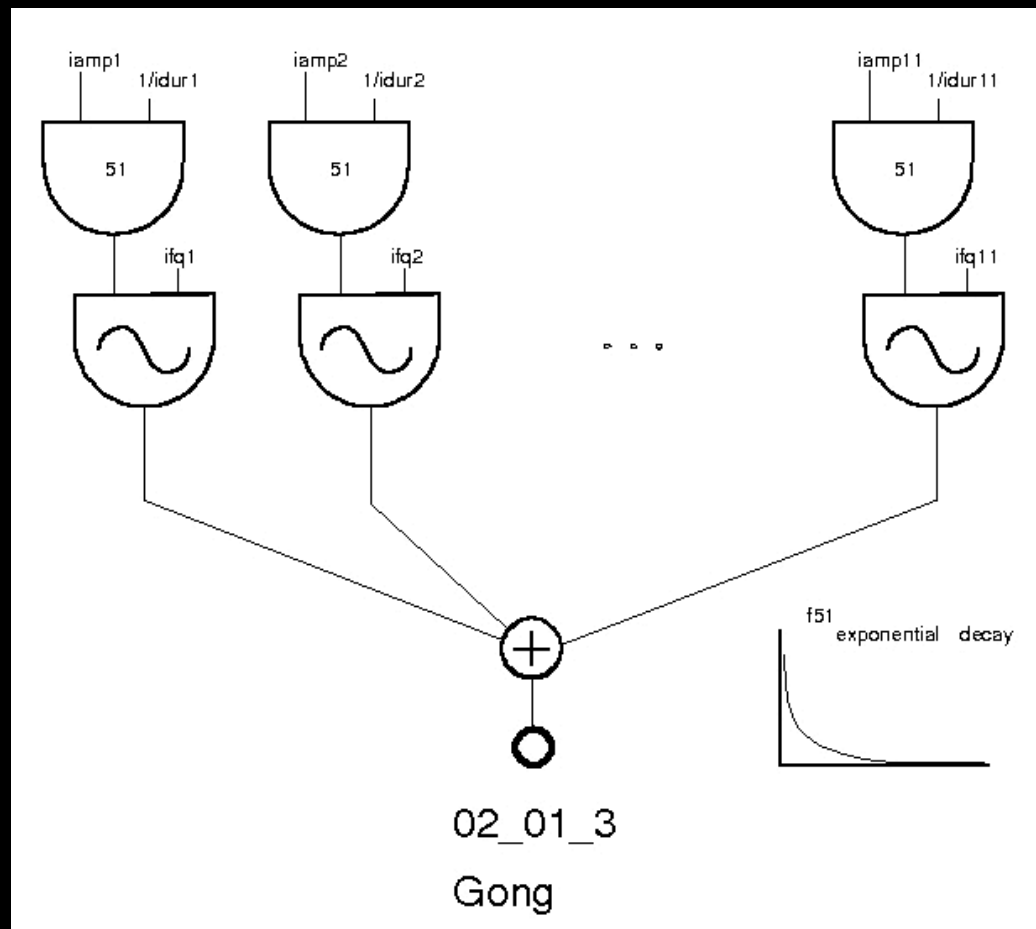
Unit Generators

“MUSIC 3 was my big breakthrough, because it was what was called a block diagram compiler, so that we could have little blocks of code that could do various things. One was a generalized oscillator ... other blocks were filters, and mixers, and noise generators.”

Max Mathews 2011 interview with Geeta Dayal, Frieze.

<http://120years.net/wordpress/music-n-max-mathews-usa-1957/>

Block Diagram



02_01_03 from Amsterdam Catalog version of Jean-Claude Risset's Bell (1969)
Accessed from: <http://www.codemist.co.uk/AmsterdamCatalog/02/index.html>

Unit Generator - Properties

- Directed Acyclic Graph
 - compose to create graph
 - Fan out, Fan in
- Require state
 - Current processing may depend on results from prior processing (i.e. filters, delay lines)
- Must short-circuit on nil from dependencies

Pink - Unit Generator

```
(defn something    <= This is the Setup function  
  [args]  
  (fn []))       <= This is the Audio function
```

Pink - Unit Generator

```
(defn some-audio-function
  [afn0 afn1 arg0 arg1]
  (let [...state data...
        ...pre-computed values...]
    (fn []
      (when-lets [buffers afns] ;; calculate deps
        (loop [...local state data...]
          (if (< indx *buffer-size*)
              (..calculate and recur..)
              (...save state and return out...)))))))
```

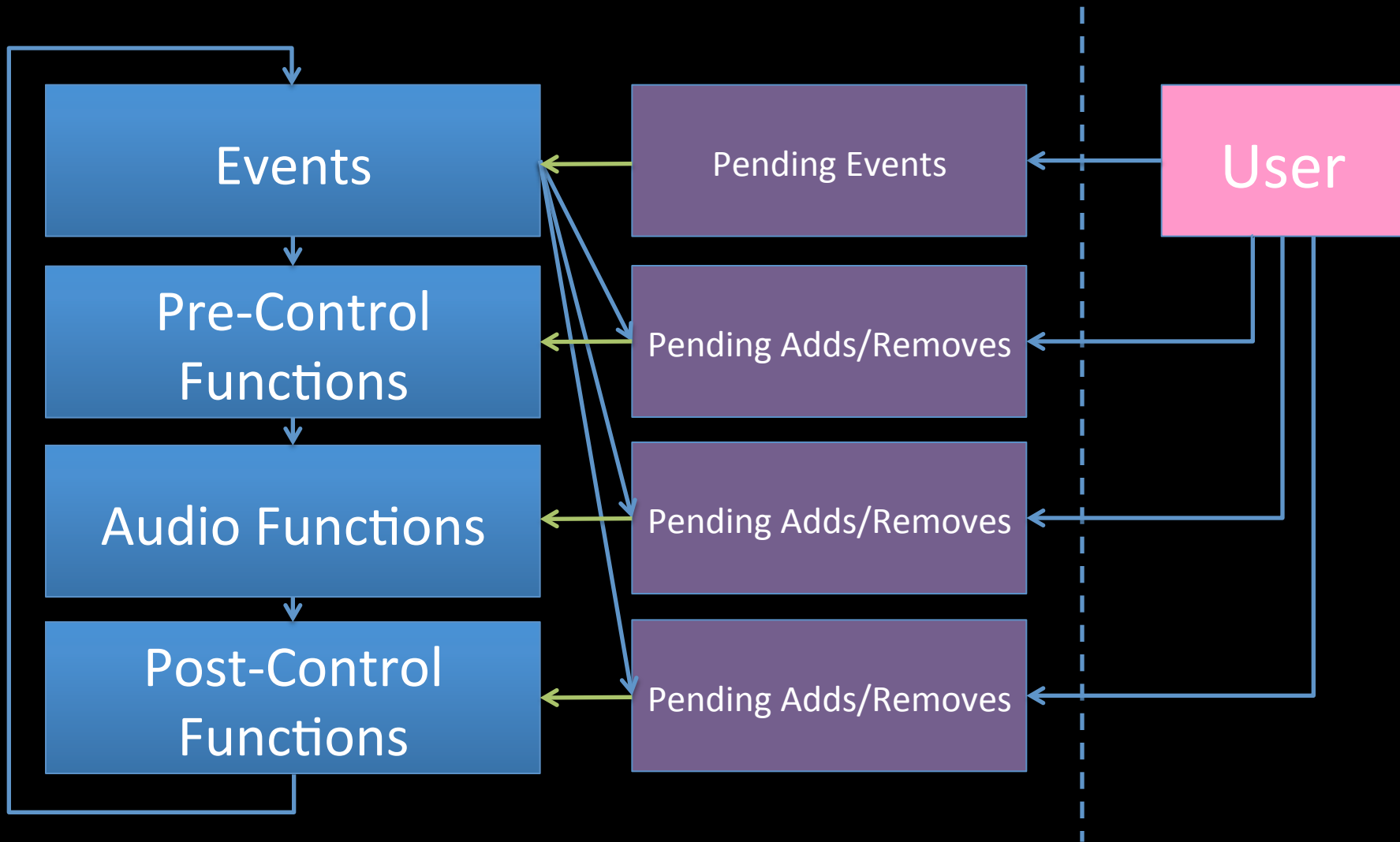
Pink - Unit Generator

```
(defn some-audio-function
  [afn0 afn1 arg0 arg1]
  (let [...pre-computed values...])
    (generator
      [loop-var1 init-val]
      [x afn0 y afn1]
      (..calculate and recur..)
      (yield out))))
```

Demos

- Audio Functions (MIDI Keyboard)
 - *duration*
 - *done*
- Events
 - Delayed Function Calls
 - Higher Order
 - Score
- Control Functions
 - Clock

Conclusions



References

- Mathews, Max V., Joan E. Miller, F. Richard Moore, John R. Pierce, and Jean-Claude Risset. The Technology of Computer Music. Cambridge: MIT Press, 1969.
- Dannenberg, Roger B. "Real-time scheduling and computer accompaniment." (1989).
- Dannenberg, Roger B., and Dean Rubine. "Toward modular, portable, real-time software." Computer Science Department (1995): 480.

Systems to Look At

- Aura
- Chuck
- Csound
- Common Lisp Music*
- Faust
- Max/MSP
- Nyquist*
- Pure Data
- RTCmix
- SuperCollider/Overtone*
- AC Toolbox*
- Common Music*
- Open Music*
- Opus Modus*
- Patchwork/PWGL*

Thank you!

- Github: [http://github.com/kunstmusik/pink score music-examples](http://github.com/kunstmusik/pink_score_music-examples)
- Email: stevenyi@gmail.com
- Web: <http://www.kunstmusik.com>