Peter Bell

SONGS OF THE SEAS

Ensemble & Code

Transposed Score

for the LEAP Ensemble

Song of Fives

Ensemble and Code

Peter Bell

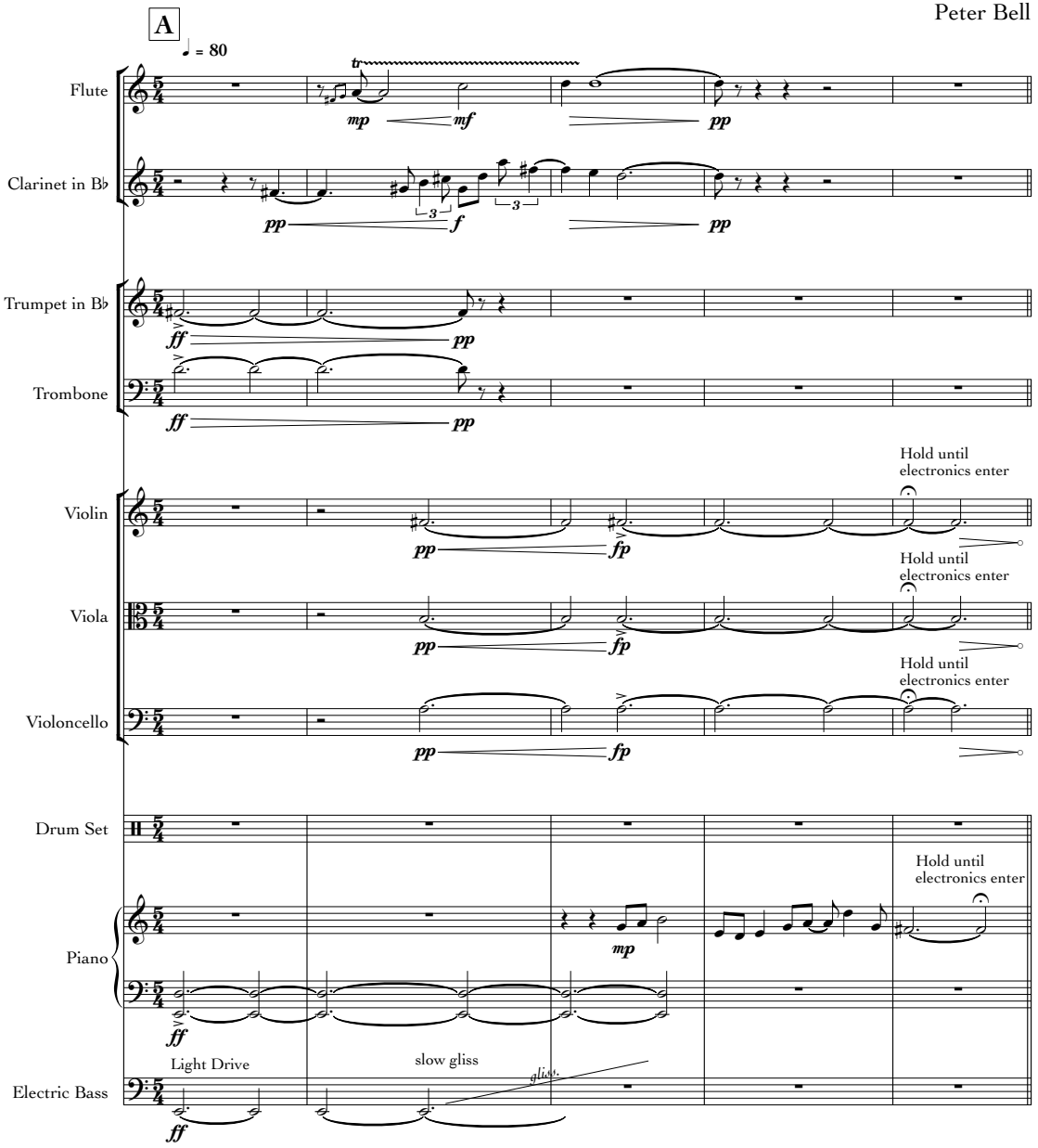
Performance Instructions

Electronics:
Laptop running Sonic Pi.
Ensemble amplified through PA.
Mono input to laptop from mixer: All parts except rhythm section.
Stereo output from audio interface to PA.
Each section is triggered by activating the next buffer in Sonic Pi.

Reading the score:
Section 1 (fig. A)
Figure A is played as written. The electronics are triggered at the start.
The electronics sample fig. A, then generate the rest of the section using the sample.

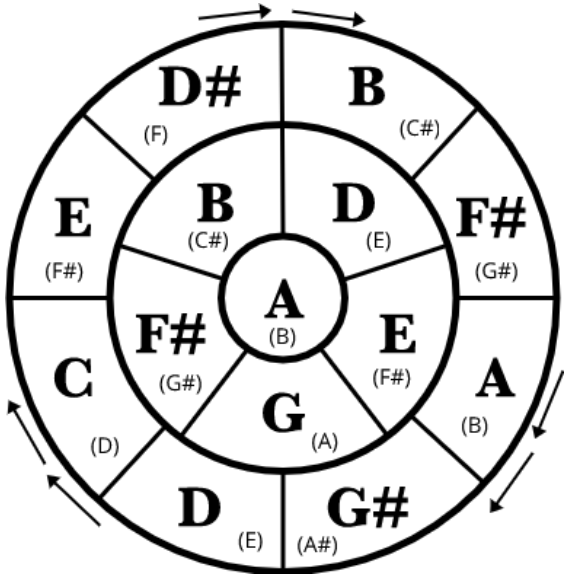
Section 2 (fig. B & C)
Once section 1 is complete section 2 is triggered.
The Drums repeat the first two bars of fig. C from the begining of the section until all the instruments have entered.
Wind, brass and strings enter gradually, as indicated, playing the circle from the centre.
Each segment of the circle is one quiet held note at the indicated pitch.
The small bracketed pitches are for instruments in Bflat.
The performers follow their own path around the circle moving clockwise and outwards.
Once they have reached the edge of the circle they can play the given musical cell (fig. B), before returning to the centre of the circle.
After all the wind, brass and strings have started playing, the rhythm section plays fig. C through 3 times.

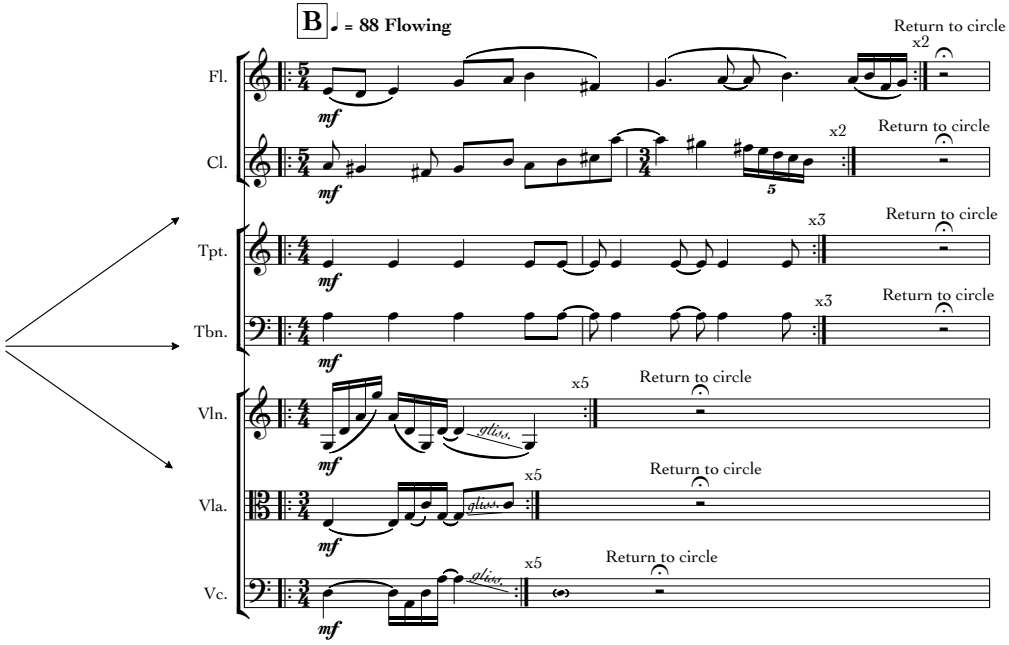
Section 3 (fig. D)
Once the rhythm section has completed fig. C the ensemble moves onto fig. D together.
Fig. D carries on directly from fig. C. The first bar of fig. D is a cue for the other performers to move on.
The rhythm section plays each cell together 5 times unless indicated until the end of the section.
The rest of the ensemble plays each cell through 3 - 5 times as they choose before repeating the last bar until the end of the piece.
It doesn't matter if bars become out of sync (1st beats of bar not lining up) but it is important to keep in tempo.

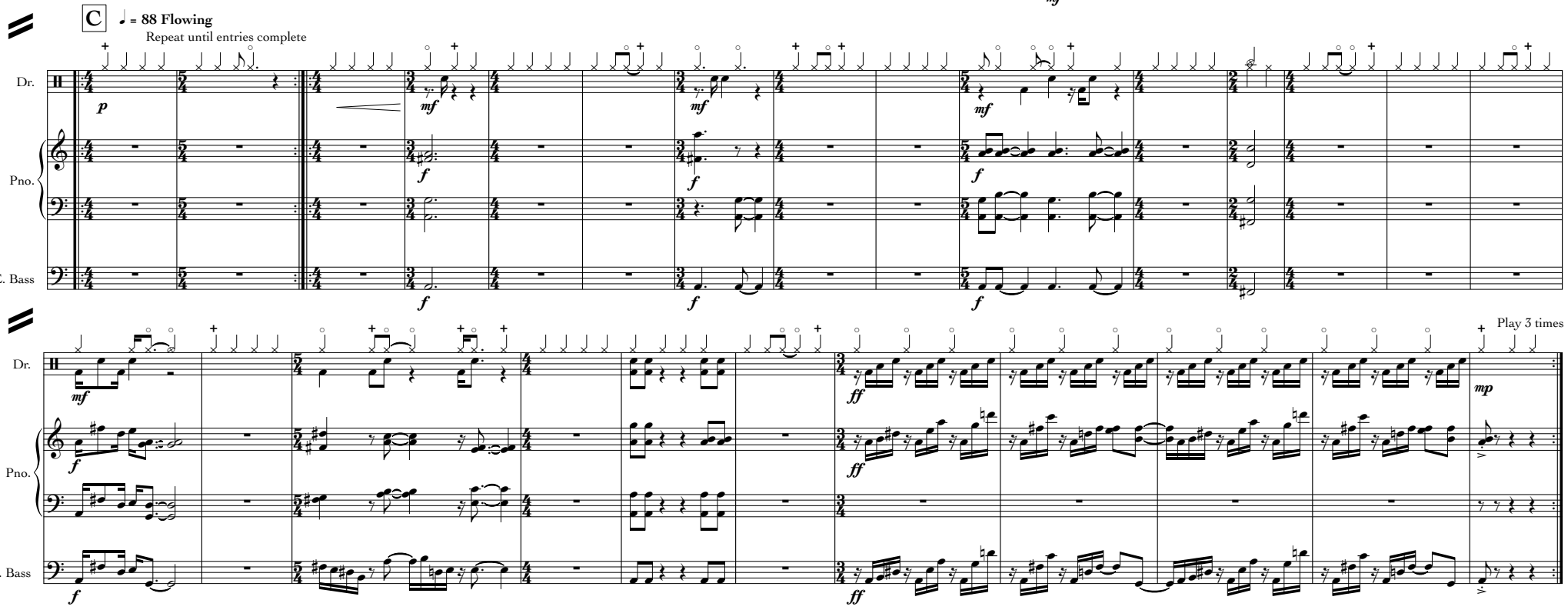


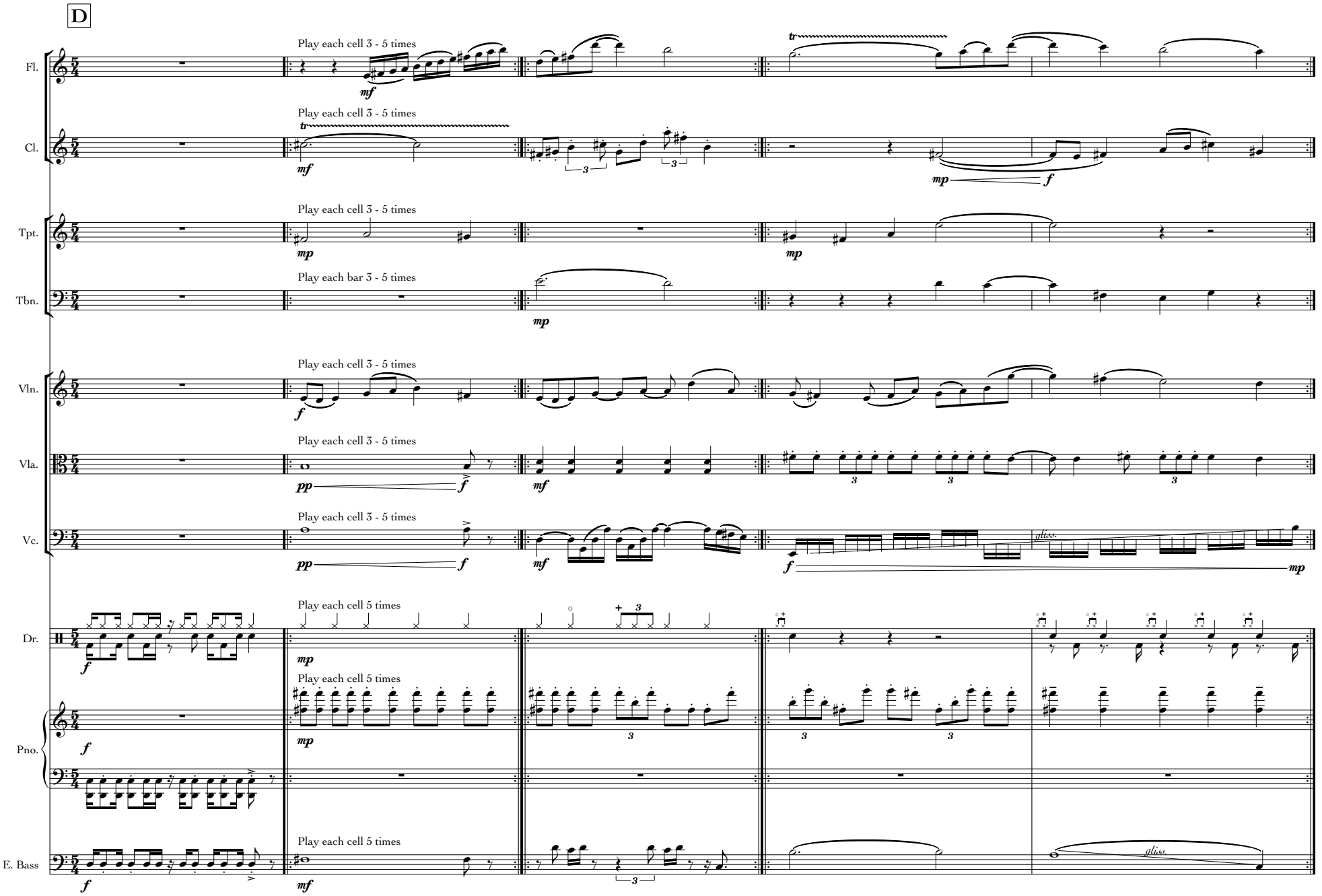
After section 2 of electronics triggered...

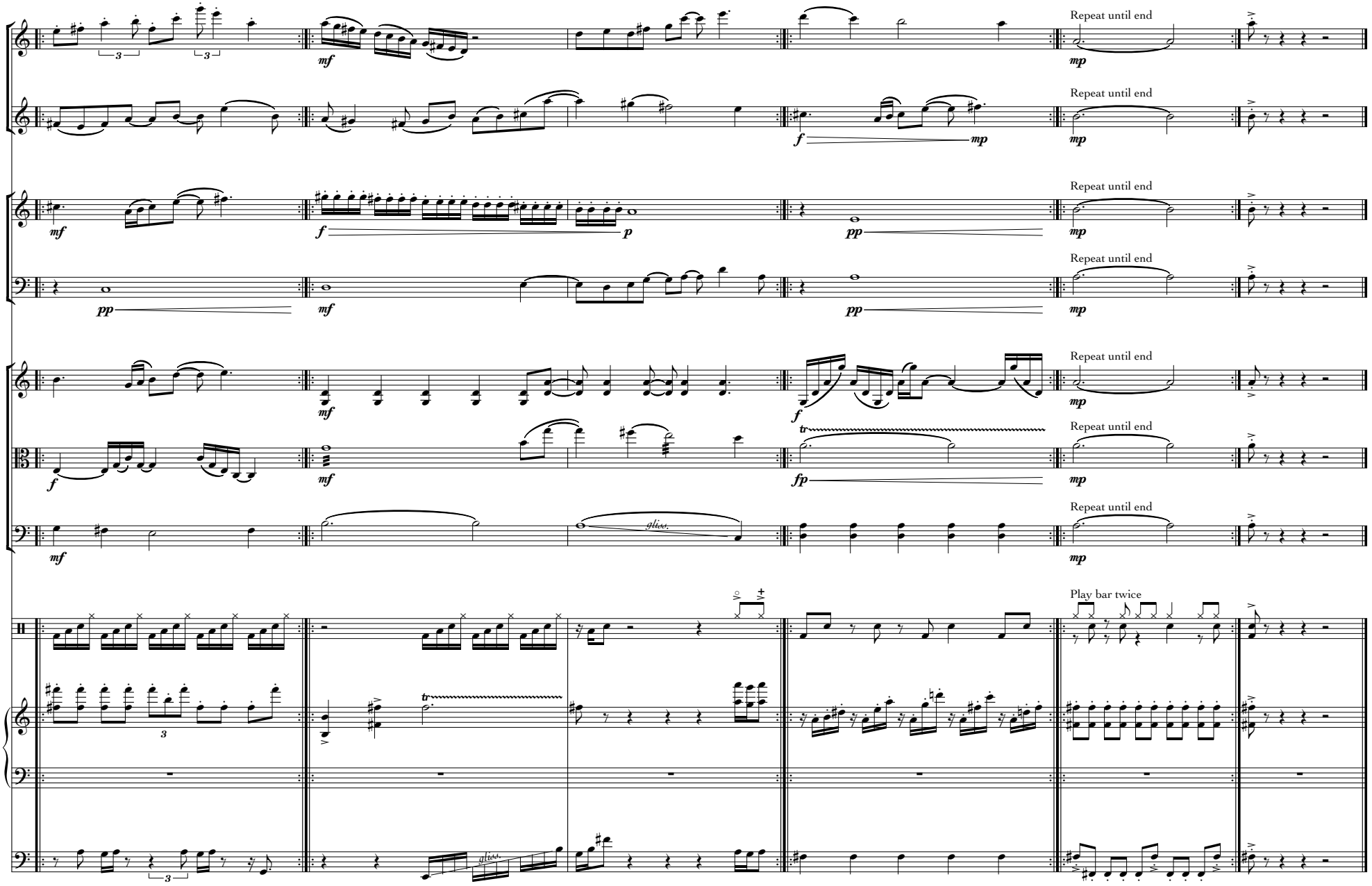
Approx. 20 secs. between each entry...
Violin, Viola (circle) & Kit (fig. C)
Flute
Cello
Clarinet
Brass
Rhythm section (fig. C) - play 3 times








D
Fl.
Play each cell 3 - 5 times
Cl.
Play each cell 3 - 5 times
Tpt.
Play each cell 3 - 5 times
Tbn.
Play each bar 3 - 5 times
Vln.
Play each cell 3 - 5 times
Vla.
Play each cell 3 - 5 times
Vc.
Play each cell 3 - 5 times
Dr.
Play each cell 5 times
Pno.
Play each cell 5 times
E. Bass
Play each cell 5 times


Fl.
Cl.
Tpt.
Tbn.
Vln.
Vla.
Vc.
Dr.
Pno.
E. Bass
Repeat until end
Play bar twice
mf
mp
f
p
pp
fp
tr
gliss.

Sonic Pi Code

Buffer 1

```
#Definitions

define :plays1 do |s, f, r, p|
  sample buffer[:sample1, 17], attack: 1,
release: 1, start: s, finish: f, rate: r, pitch: p
end

define :plays1b do |s, f, r, p|
  sample buffer[:sample1, 17], amp: 0.7, attack:
0, release: 0, start: s, finish: f, rate: r, pitch: p
end

define :plays1c do |s, f, r, p, a|
  sample buffer[:sample1, 17], amp: a, attack: 0,
release: 0, start: s, finish: f, rate: r, pitch: p
end

#Main Code
with_fx :sound_out, output: 3, amp: 0 do

  print "Start Recording Sample 1"
  with_fx :sound_out, output: 5, amp: 0 do
    with_fx :record, buffer: buffer[:sample1, 17]
do
      live_audio :audioin1, input: 1
    end
  end
  sleep 17
  print "Sample 1 Recorded"

in_thread do
  print "New Loop"
  20.times do
    plays1 0, 0.3, [0.8, 1, 0.6, 0.4].choose , 0
    sleep [3, 5, 7].choose
  end
end

in_thread do
  sleep 30
  print "New Loop"
  20.times do
    plays1 0, 0.3, 1 , [-2, 0, 0, 2].choose
    sleep [2, 6, 9].choose
  end
end

in_thread do
  sleep 50
  print "New Loop"
  20.times do
    plays1 0.4, 0.7, 1 , [-2, 0, 5, 0, 2].choose
    sleep [4, 6, 9].choose
  end
end

in_thread do
  sleep 100
  print "New Loop"
  with_fx :pitch_shift, pitch: 12 do
    20.times do
      plays1 0.4, 0.7, 1, [-2, 0, -4].choose
      sleep 2
    end
  end
end

sleep 130
print "Final S1 Loop"
with_fx :sound_out, output: 4, amp: 0 do
  set :s2trig, 0
  i = 1
  loop do
    tr1 = get[:s2trig]
    if tr1 == 0
      plays1 0.4, 1, 0.5, 0
      sleep 10
    else
      if i != 0
        plays1c 0.4, 0.6, 0.5, 0, 1
        sleep 5
        plays1c 0.4, 0.6, 0.5, 0, 0.65
        sleep 6
        plays1c 0.4, 0.6, 0.5, 0, 0.3
        sleep 8
        i = 0
      else
        print "Section 1 Stopped"
        sleep 500
      end
    end
  end
end
```

Buffer 2

```
#Definitions

define :plays2 do |s, f, r, p|
  sample buffer[:sample2, 10], attack: 1,
release: 1, start: s, finish: f, rate: r, pitch: p
end

define :plays2b do |s, f, r, p|
  sample buffer[:sample2, 10], attack: 0,
release: 0, start: s, finish: f, rate: r, pitch: p
end

define :plays3 do |s, f, r, p|
  sample buffer[:sample3, 10], attack: 1,
release: 1, start: s, finish: f, rate: r, pitch: p
end

define :plays3b do |s, f, r, p|
  sample buffer[:sample3, 10], attack: 0,
release: 0, start: s, finish: f, rate: r, pitch: p
end

#Main Code

set :s2trig, 1 #Sends command to fade out
section 1

with_fx :sound_out, output: 3, amp: 0 do
  with_fx :sound_out, output: 4, amp: 0.5 do
    print "Section 2 Start"

    in_thread do
      sleep 10
      print "Start Recording Sample 2"
      with_fx :sound_out, output: 5, amp: 0 do
        with_fx :record, buffer:
buffer[:sample2,10] do
          live_audio :audioin2, input: 1
        end
      end
      sleep 10
      print "Sample 2 Recorded"
    end

    10.times do
      plays1b 0.2, 0.2375, 1, 0
      sleep 0.375
      plays1b 0.4, 0.45, 1, 0
      sleep 0.375
      plays1b 0.7, 0.75, 1, 0
      sleep 0.75
      plays1b 0.3, 0.3375, 1, 0
      sleep 0.375
    end

    set :s3trig, 0
    ctl = 1

    loop do
      tr2 = get[:s3trig]
      if tr2 == 0
        v1 = [0.375, 0.75, 1.125].choose #sleep
length short/Timed
        v2 = [0.75, 1.125, 1.5].choose #sleep length
short/Timed2
        v3 = [7.5, 11.25, 15].choose #sleep length
long
        v4 = [22.5, 30, 37.5].choose #sleep length
long2
        v5 = [0.02, 0.0375, 0.05].choose #note
length1
        v6 = [0.0375, 0.05, 0.075].choose #note
length2
        v7 = [3, 5, 9].choose #note length3

        if ctl == 1
          in_thread do
            print "Start Recording Sample 3"
            with_fx :sound_out, output: 5, amp: 0 do
              with_fx :record, buffer:
buffer[:sample3, 10] do
                live_audio :audioin3, input: 1
              end
            end
            sleep 10
            print "Sample 3 Recorded"
          end

          in_thread do
            1.times do
              sleep v7
              plays2 0, 1, 1, 0
            end
          end

          print "New Loop"
```

```
      2.times do
        plays2b 0.2, 0.2 + v5, 1, 0
        sleep v1
        plays2b 0.4, 0.4 + v6, 1, 0
        sleep v1
        plays2b 0.7, 0.7 + v5, 1, 0
        sleep v1
        plays2b 0.3, 0.3 + v6, 1, 0
        sleep v2
        plays2b 0.2, 0.2 + v5, 1, 0
        sleep v2
        plays2b 0.4, 0.4 + v6, 1, 0
        sleep v2
        plays2b 0.7, 0.7 + v5, 1, 0
        sleep v1
        plays2b 0.3, 0.3 + v6, 1, 0
        sleep v2
      end
      print "End of Loop"
      ctl = 2
    else
      in_thread do
        sleep 10
        print "Start Recording Sample 2"
        with_fx :sound_out, output: 5, amp: 0 do
          with_fx :record, buffer:
buffer[:sample2,10] do
            live_audio :audioin4, input: 1
          end
        end
        sleep 10
        print "Sample 2 Recorded"
      end
```

```
      in_thread do
        1.times do
          sleep v3
          plays2 0, 1, 1, 0
        end
      end

      print "New Loop"
      2.times do
        plays3b 0.2, 0.2 + v6, 1, 0
        sleep v1
        plays3b 0.4, 0.4 + v6, 1, 0
        sleep v2
        plays3b 0.7, 0.7 + v5, 1, 0
        sleep v1
        plays3b 0.3, 0.3 + v5, 1, 0
        sleep v2
        plays3b 0.2, 0.2 + v5, 1, 0
        sleep v2
        plays3b 0.4, 0.4 + v6, 1, 0
        sleep v2
        plays3b 0.7, 0.7 + v6, 1, 0
        sleep v1
        plays3b 0.3, 0.3 + v5, 1, 0
        sleep v1
        ctl = 1
        print "End of Loop"
      end
    end
  else
    print "Section 2 Stopped"
    sleep 500
```

```
    end
  end
 end
end
```

Buffer 3

```
#Definitions

define :plays4 do |s, f, r, p|
 sample buffer[:sample4, 10], attack: 0.5,
release: 0.5, start: s, finish: f, rate: r, pitch: p
end

define :plays4c do |s, f, r, p, a|
 sample buffer[:sample4, 10], attack: 0.5,
release: 0.5, start: s, finish: f, rate: r, pitch: p,
amp: a
end

define :plays5 do |s, f, r, p|
 sample buffer[:sample5, 10], attack: 0.5,
release: 0.5, start: s, finish: f, rate: r, pitch: p
end

define :plays5c do |s, f, r, p, a|
 sample buffer[:sample5, 10], attack: 0.5,
release: 0.5, start: s, finish: f, rate: r, pitch: p,
amp: a
end

#Main Code
```

```
set :s3trig, 1 #Sends command to stop section 2

with_fx :sound_out, output: 3, amp: 0 do
 with_fx :sound_out, output: 4, amp: 1.2 do

  in_thread do
    print "Start Recording Sample 4"
    with_fx :sound_out, output: 5, amp: 0 do
      with_fx :record, buffer: buffer[:sample4,
10] do
        live_audio :audioin5, input: 1
      end
    end
    sleep 10
    print "Sample 4 Recorded"
  end

  7.times do #21 secs
    plays3 0, 0.075, 1, 0
    sleep 0.75
    plays3 0.1, 0.175, 1, 2
    sleep 0.75
    plays3 0.2, 0.275, 1, 4
    sleep 0.75
    plays3 0.3, 0.375, 1, 5
    sleep 0.75
  end

  in_thread do
    print "Start Recording Sample 5"
    with_fx :sound_out, output: 5, amp: 0 do
      with_fx :record, buffer: buffer[:sample5,
10] do
        live_audio :audioin6, input: 1
      end
    end
    sleep 10
    print "Sample 5 Recorded"
  end

  7.times do #21 secs
    plays4 0, 0.075, 1, 0
    sleep 0.75
    plays4 0.1, 0.175, 1, 0
    sleep 0.75
    plays4 0.2, 0.275, 1, 0
    sleep 0.75
    plays4 0.3, 0.375, 1, 0
    sleep 0.75
  end

  in_thread do
    print "Start Recording Sample 4"
    with_fx :sound_out, output: 5, amp: 0 do
      with_fx :record, buffer: buffer[:sample4,
10] do
        live_audio :audioin7, input: 1
      end
    end
    sleep 10
    print "Sample 4 Recorded"
  end

  5.times do #30 secs
    plays5 0, 0.15, 1, 0
    sleep 1.5
    plays5 0.1, 0.25, 1, 0
    sleep 1.5
    plays5 0.2, 0.35, 1, 0
    sleep 1.5
    plays5 0.3, 0.45, 1, 0
    sleep 1.5
  end

  in_thread do
    print "Start Recording Sample 5"
    with_fx :sound_out, output: 5, amp: 0 do
      with_fx :record, buffer: buffer[:sample5,
10] do
        live_audio :audioin8, input: 1
      end
    end
    sleep 10
    print "Sample 5 Recorded"
  end

  5.times do #45 secs
    plays4 0, 0.225, 1, 0
    sleep 2.25
    plays4 0.1, 0.325, 1, 0
    sleep 2.25
    plays4 0.2, 0.425, 1, 0
    sleep 2.25
    plays4 0.3, 0.525, 1, 0
    sleep 2.25
  end
```

```
  #51 to go
  in_thread do
    print "Start Recording Sample 4"
    with_fx :sound_out, output: 5, amp: 0 do
      with_fx :record, buffer: buffer[:sample4,
10] do
        live_audio :audioin9, input: 1
      end
    end
    sleep 10
    print "Sample 4 Recorded"
  end

  #15 seconds
  plays5 0, 0.3, 1, 0
  sleep 3
  plays5 0.1, 0.4, 1, 0
  sleep 3
  plays5 0.2, 0.5, 1, 0
  sleep 3
  plays5 0.3, 0.6, 1, 0
  sleep 3
  plays5 0.4, 0.7, 1, 0
  sleep 3

  #Just under 36 seconds
  plays4c 0, 0.3, 1, 0, 1
  sleep 3
  plays4c 0.1, 0.4, 1, 0, 0.9
  sleep 3
  plays4c 0.2, 0.5, 1, 0, 0.8
  sleep 3
  plays4c 0.3, 0.6, 1, 0, 0.7
  sleep 3
  plays4c 0.4, 0.7, 1, 0, 0.6
  sleep 3
  plays4c 0.5, 0.8, 1, 0, 0.5
  sleep 3
  plays4c 0.6, 0.9, 1, 0, 0.4
  sleep 3
  plays4c 0.7, 1, 1, 0, 0.3
  sleep 3
  plays4c 0.8, 1, 1, 0, 0.25
  sleep 3
  plays4c 0.9, 1, 1, 0, 0.25
  sleep 3

 end
end
```