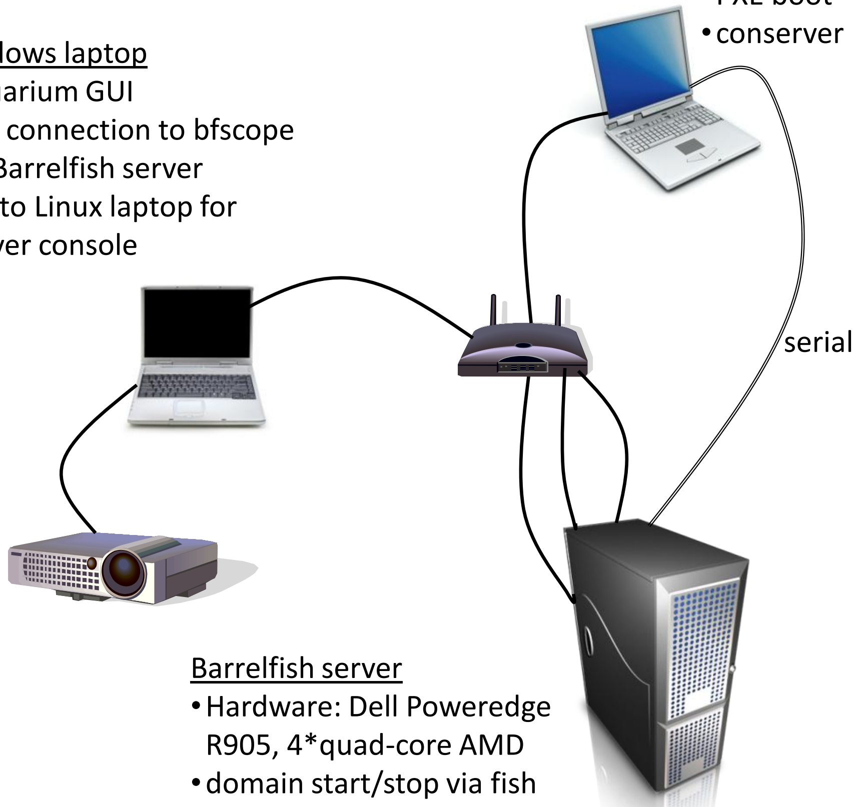
Guide to the Barrelfish Demo

December 2009

Setup

Live



Windows laptop

- Aquarium GUI
- TCP connection to bfscope on Barrelfish server
- ssh to Linux laptop for server console

Linux laptop

- DHCP/TFTP
- PXE boot
- conserver

Barrelfish server

- Hardware: Dell Poweredge R905, 4*quad-core AMD
- domain start/stop via fish
- trace control via bfscope
- bfscope uses e1000 to send traces to client

Canned

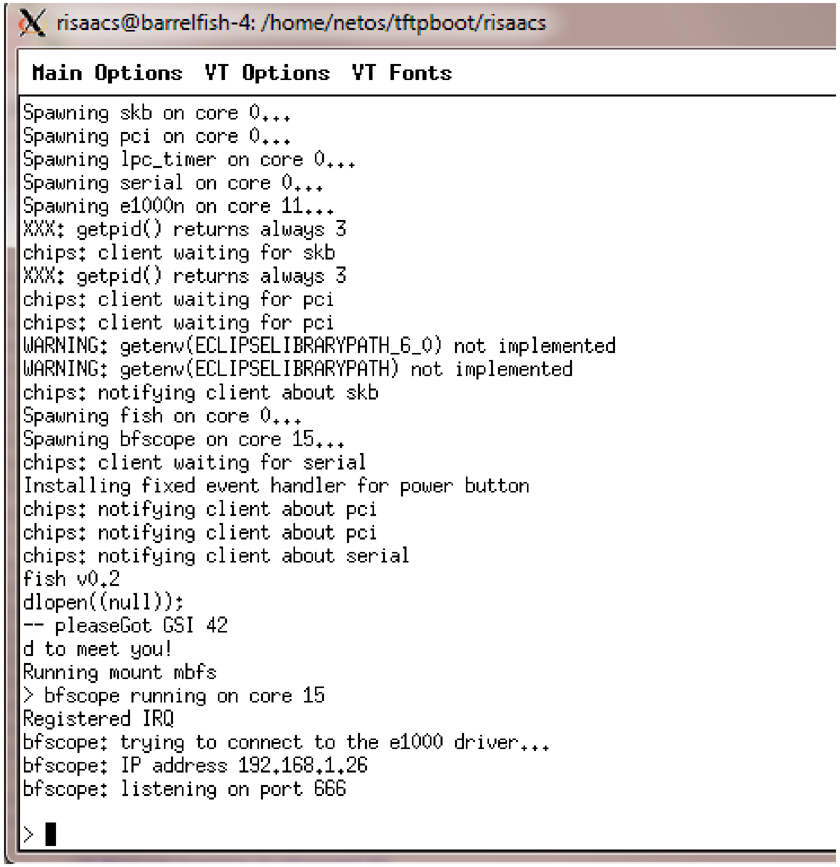
Windows laptop

- TestAquarium: serves Barrelfish trace files
- Aquarium GUI displays canned traces

1. Boot Barrelfish

menu.lst file for demo:

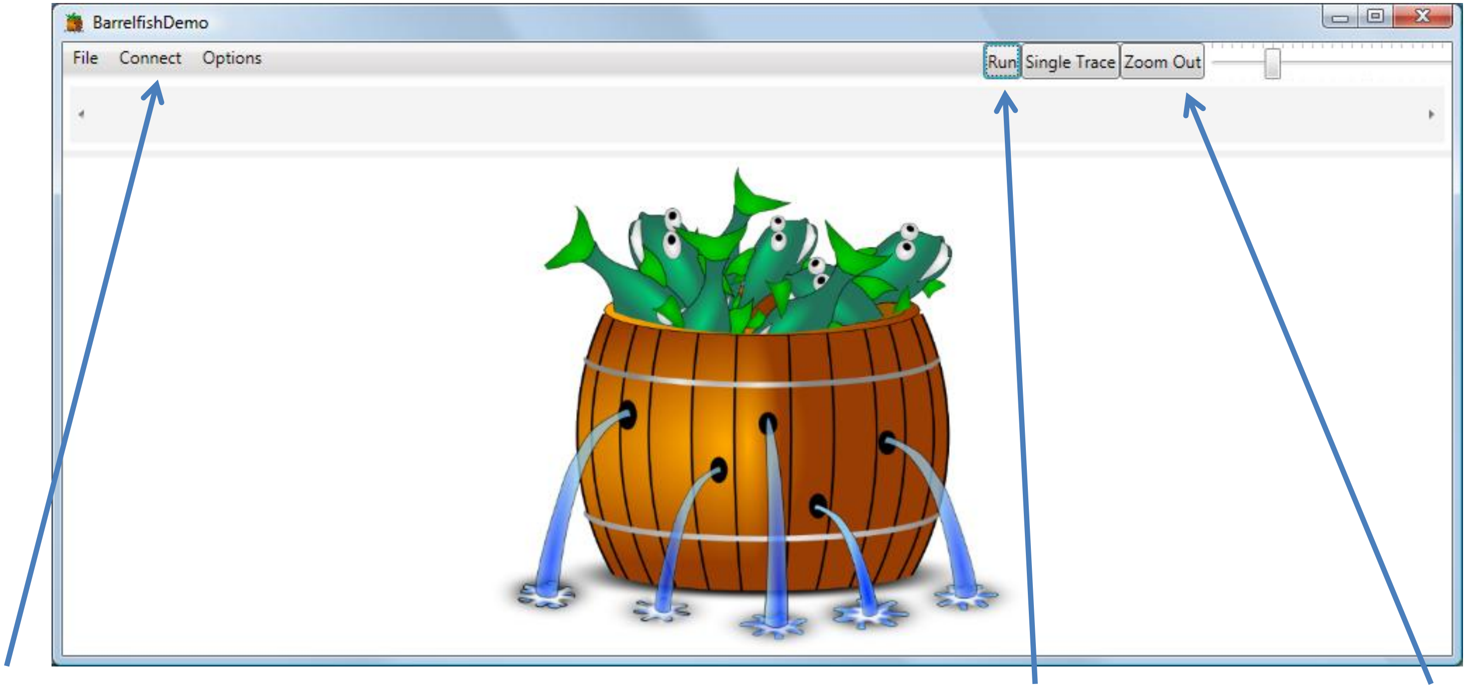
```
Title   Barrelfish-demo
Root    (nd)
Kernel  /barrelfish/x86_64/sbin/cpu loglevel=4 logmask=0
#
# Essentials
#
Module  /barrelfish/x86_64/sbin/init
Module  /barrelfish/x86_64/sbin/mem_serv
Module  /barrelfish/x86_64/sbin/monitor bootcpus=1-15
Module  /barrelfish/x86_64/sbin/chips
module  /barrelfish/x86_64/sbin/skb
module  /barrelfish/x86_64/sbin/pci
#
# Drivers
#
module /barrelfish/x86_64/sbin/lpc_timer
module /barrelfish/x86_64/sbin/serial
module /barrelfish/x86_64/sbin/e1000n core=11 irq
#
# Shell
#
module /barrelfish/x86_64/sbin/fish
#
# Demo
#
module /barrelfish/x86_64/sbin/bfscope core=15
module /barrelfish/x86_64/sbin/spantest nospawn
module /barrelfish/x86_64/sbin/pixels nospawn
```



```
Spawning skb on core 0...
Spawning pci on core 0...
Spawning lpc_timer on core 0...
Spawning serial on core 0...
Spawning e1000n on core 11...
XXX: getpid() returns always 3
chips: client waiting for skb
XXX: getpid() returns always 3
chips: client waiting for pci
chips: client waiting for pci
WARNING: getenv(ECLIPSELIBRARYPATH_6_0) not implemented
WARNING: getenv(ECLIPSELIBRARYPATH) not implemented
chips: notifying client about skb
Spawning fish on core 0...
Spawning bfscope on core 15...
chips: client waiting for serial
Installing fixed event handler for power button
chips: notifying client about pci
chips: notifying client about pci
chips: notifying client about serial
fish v0.2
dlopen((null));
-- pleaseGot GSI 42
d to meet you!
Running mount mbfs
> bfscope running on core 15
Registered IRQ
bfscope: trying to connect to the e1000 driver...
bfscope: IP address 192.168.1.26
bfscope: listening on port 666

> 
```

2. Connect Aquarium to Barrelfish



1. Connect to the Barrelfish computer. Either arrange port 666 on localhost to be forwarded, or add the IP address to this menu in the source code.

2. When connected, the Barrelfish console will show:

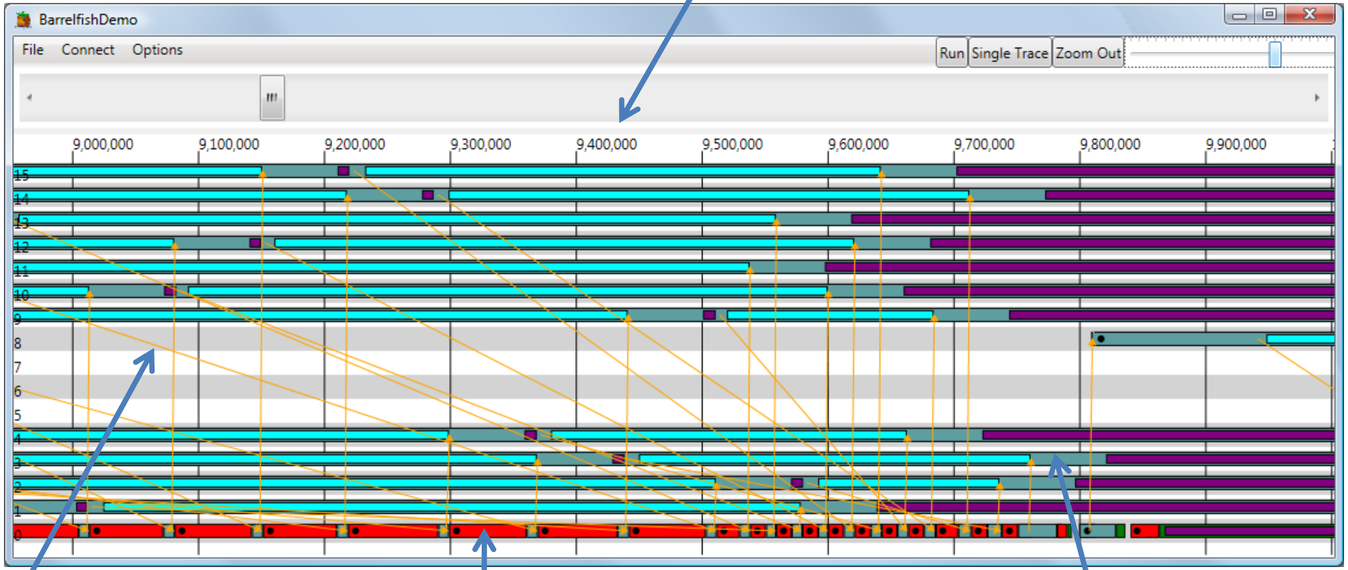
```
> bfscope: connected
T
>
```

3. Hit "Run". Button will toggle to "Stop".

4. Type a command into fish (the Barrelfish shell). See following slides for examples.

5. Wait for the trace. Once displayed, click on "Zoom Out" to automatically fit the trace to the window.

Trace visualisation in Aquarium



cycles since the start of the trace

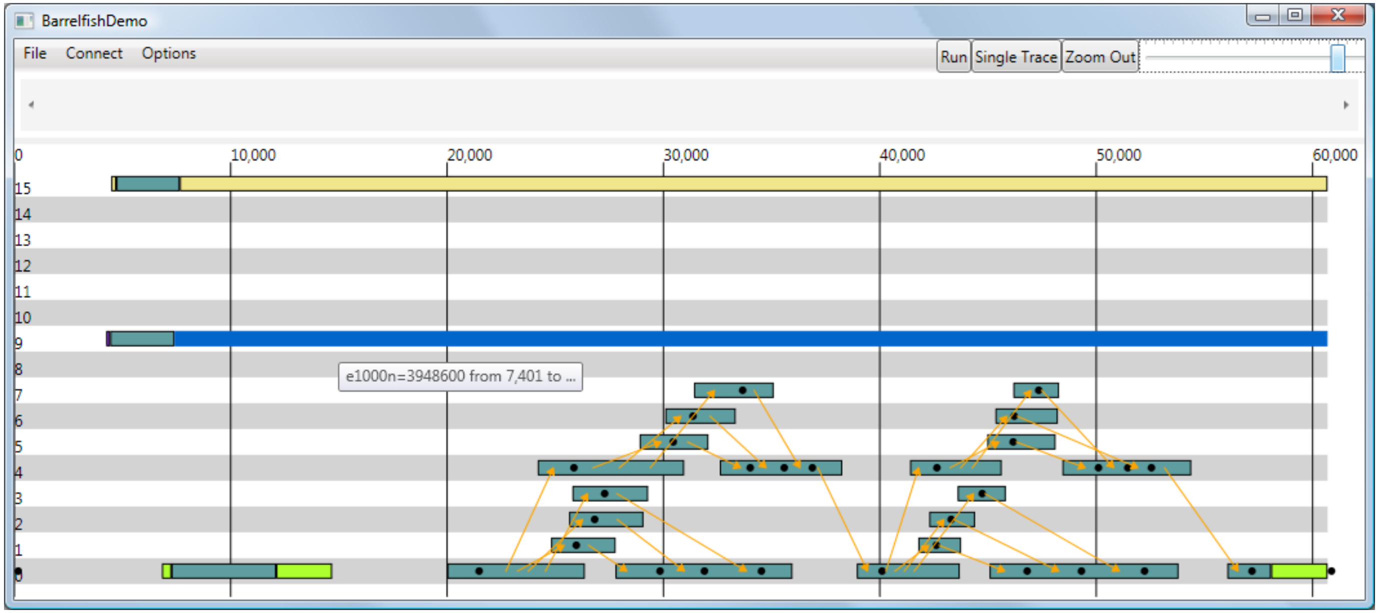
each horizontal line shows the activity on a core

messages between cores are represented by yellow arrows

different colours show different domains executing, eg red is the memory server, greeny-blue is the monitor – here we can see core 0 context switching between the memory server and the monitor

coloured overlay rectangles are used to show when the domain is blocked (dark gray), polling (cyan), or bzero-ing (purple)

Controlling Aquarium



Click on Run for continuous display of traces. Once the current trace is rendered, another is requested from bfscope (if running live) or from TestAquarium (if running offline).

Hover the mouse over any coloured rectangle for a tooltip explanation. If sluggish, try disabling tooltips in the Options menu.

Demo

1. Two-phase commit
2. Spantest
3. Pixels demo

Demo (i): Two-phase commit

- Shell built-in that invokes two-phase commit between the monitors on all the cores to demonstrate the performance difference between a unicast tree and a NUMA-aware multicast tree for routing the messages.
- Compare 16 cores unicast:

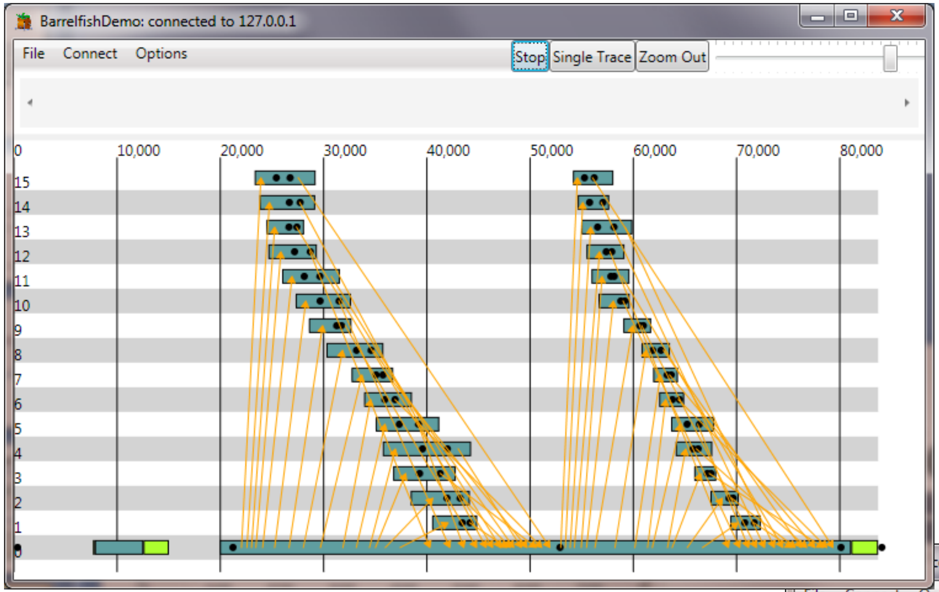
```
> 2pc 16 0
```

- With 16 cores multicast:

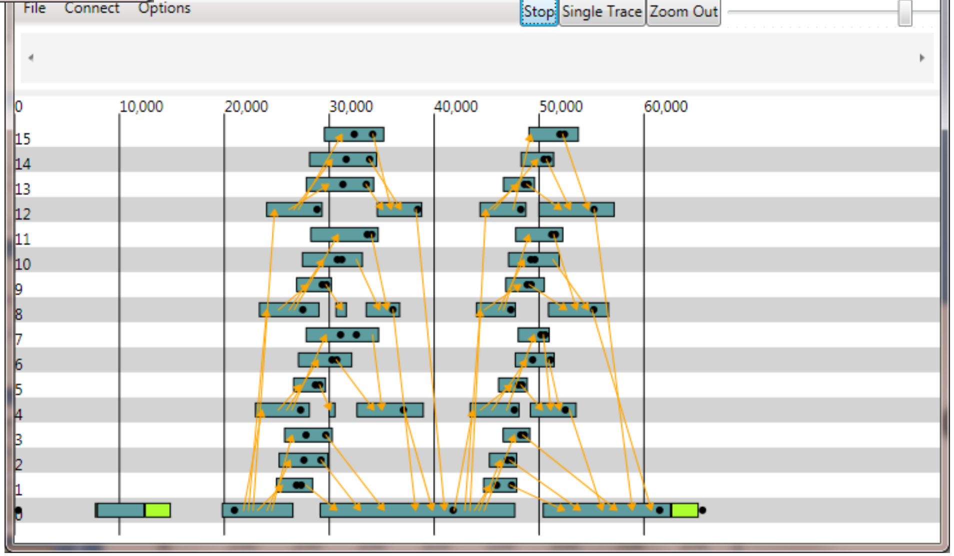
```
> 2pc 16 1
```

- Note that the multicast route is computed by the SKB

Two-phase commit visualisation



Unicast



NUMA-aware multicast

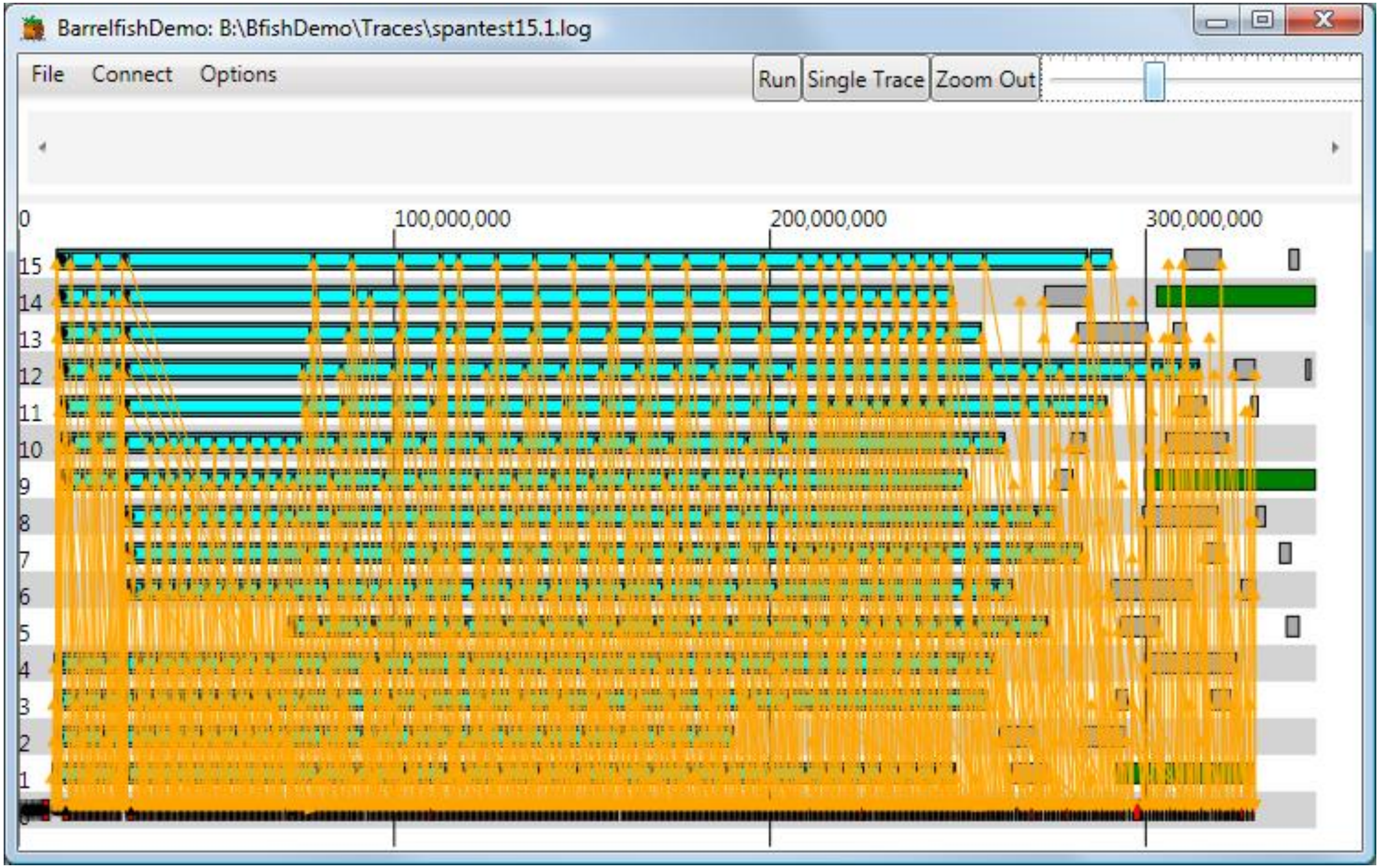
Demo (ii): Domain spanning

- This program creates a domain with n threads sharing a single address space, one thread running on each core, where n is the command-line argument

```
> spantest 15
```

Domain spanning : 15 cores

Zoom into the trace to see that the bottleneck is the memory server running on core 0

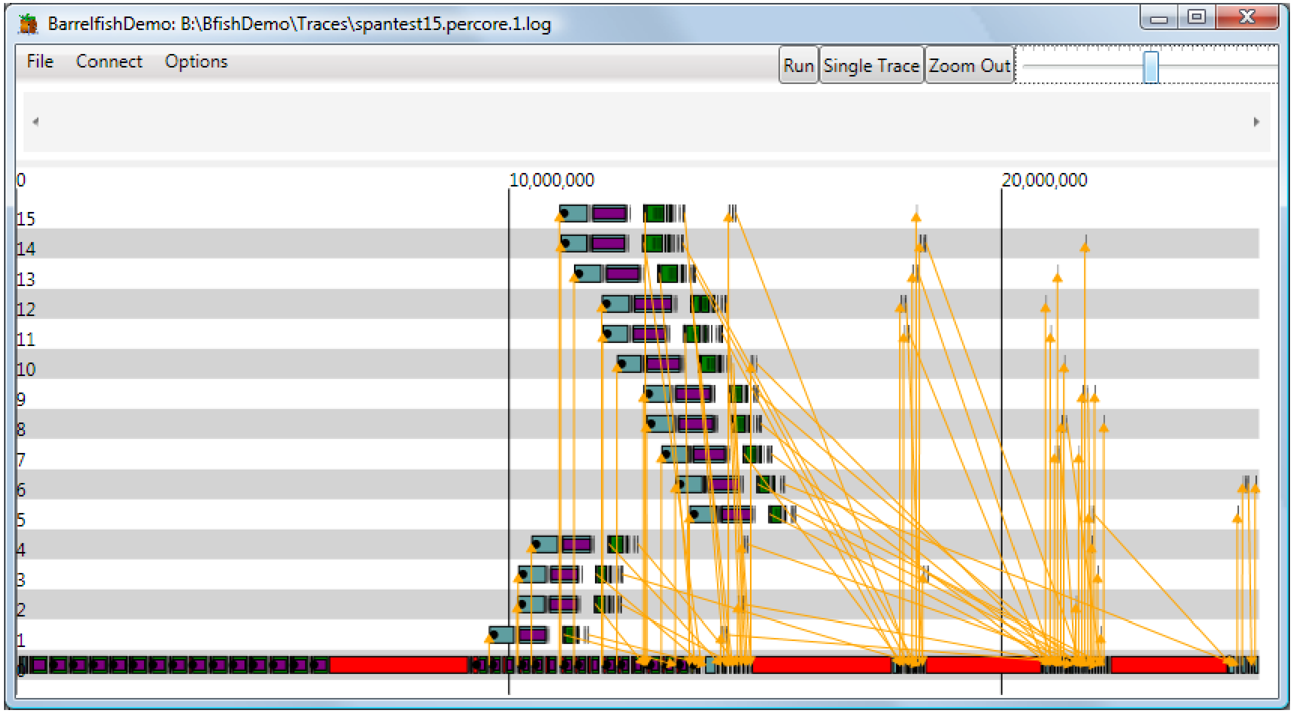


Domain spanning (cont)

- Now run a partitioned memory server

```
> percore
… [lots of debug spew]
> spantest 15
```

- Performance improvements are dramatic!

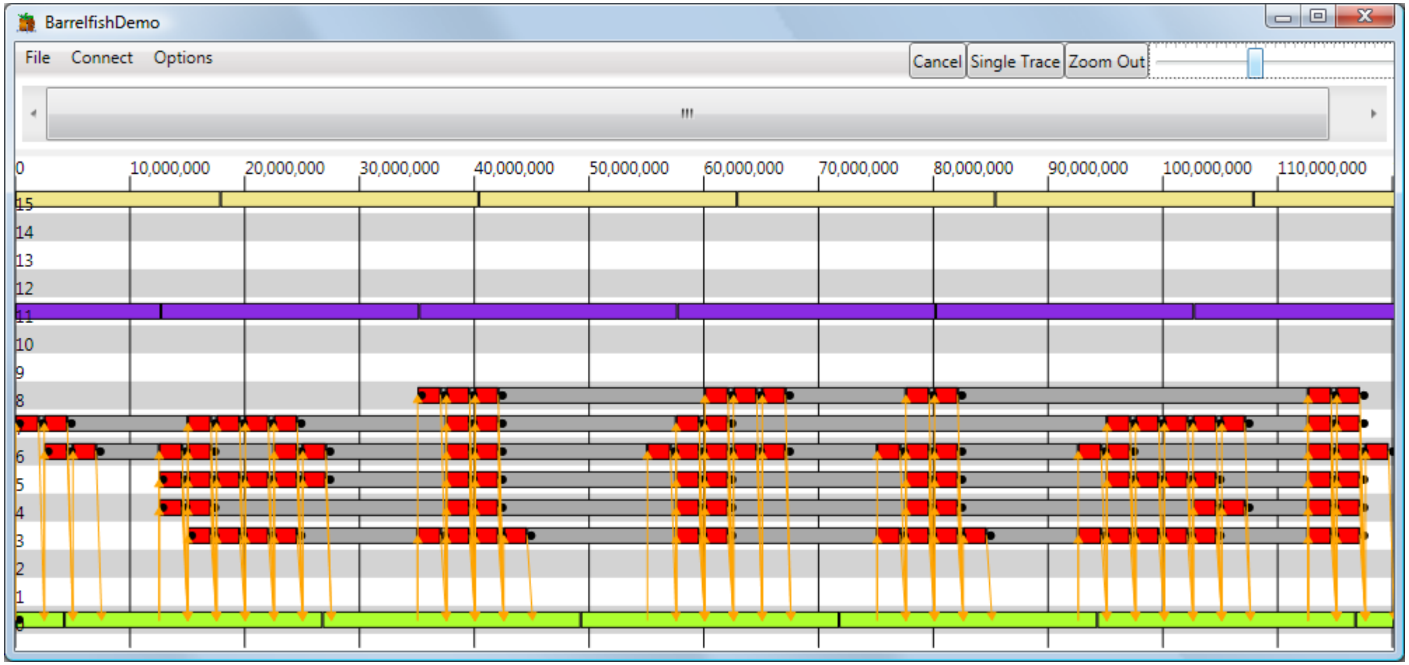


NB partitioned memory server is a demo app only (each partition has only 17MB to allocate for domains on its core, and Barrelfish doesn't reclaim memory yet)

Demo (iii): scrolling pixels

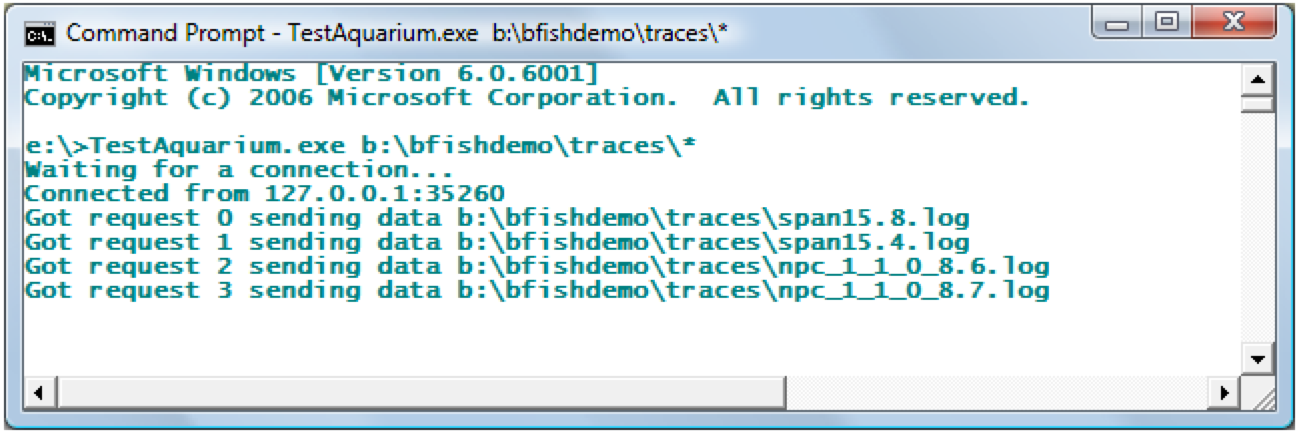
- This one is a bit of fun

```
> demo
```



Canned demo

- Run TestAquarium on Windows
  - Listens on port 666 of localhost
  - Serves randomly selected traces to Aquarium from the specified directory until stopped



```
Microsoft Windows [Version 6.0.6001]
Copyright (c) 2006 Microsoft Corporation.  All rights reserved.

e:\>TestAquarium.exe b:\bfishdemo\traces\*
Waiting for a connection...
Connected from 127.0.0.1:35260
Got request 0 sending data b:\bfishdemo\traces\span15.8.log
Got request 1 sending data b:\bfishdemo\traces\span15.4.log
Got request 2 sending data b:\bfishdemo\traces\npc_1_1_0_8.6.log
Got request 3 sending data b:\bfishdemo\traces\npc_1_1_0_8.7.log
```

Usage: TestAquarium.exe <filename or glob>