##0 in the same time slot as the active clock = no delay
##0 between clock events = ##1

NBA &

Mehdi, Cliff, Arturo, Gordon, Steven, Dave, Jonathan,

Here are some issues I think we need to discuss in the face-to-face:

1. Are continuous assigns in program blocks subject to reactive scheduling?
Yes.
No cb.port continuous assignments - cb.port requires clocking drives.
For bi-dir ports, the port must be a wire (not a logic) and then driven by a cb.port from an initial block.
Continuous assignments and cb.port clocking drives - legal and have normal resolution semantics.

2. Are program wire ports allowed to be collapsed using conventional port collapsing techniques?
Yes (please!)

3. Are program variable ports considered program variables?
Immaterial for the purposes of new assignment semantics. Assignments within the program execute in the reactive-regions and do not depend on the target.(!!)
Example:

```
// ref int a; causes problems!!
```

```
program p (.myport({mnet1, mnet2}), a, pwire, preg);
  ref     int    a;
  output         pwire, preg;
  output [1:0] myport;
  logic          mnet1, mnet2; // re-NBA scheduled
  logic          preg;
  wire           pwire;
```

4. Are program variable ports connected to parent objects in the same way that design variable ports are connected to parent objects?
Yes

5. Are we OK with the loophole of a program calling a blocking task in a design region? Such programs would wake up in the active region, and possibly return to program scope executing in the active region. I don't know how to plug this except by disallowing programs to call design tasks by hierarchical ref.
Yes - Some change to the LRM is necessary with regards to resynchronizing a reactive process after calling such a module task (the returned task output). Now all of the task code is going to execute in the reacitve and #0 re-inactive and NBA to the re-NBA region.

5a.Tasks declared in packages are considered design scopes (unless they are within anonymous program blocks). Calling such a task will have similar loopholes as 5.
Now Immaterial

6. Are we all OK with Arturo's proposal to create a continuous NBA? I think that is a good idea. - For now, this will not be added (but no strong objections - could be added later if stronger support exists).

7. Should we allow NBA's in program scope, and create a re-NBA region in which they are scheduled? Dave has illustrated the usefulness of this construct for testbench modeling. He's not convinced about the need for a re-NBA region. I think it is needed in order to avoid blurring between design code running through loopholes in tandem with program code running.
Yes (also covered in question #9)

8. If the answer to 7 is no, what are the semantics of a clocking drive to a program variable? Should that be allowed?
(The answer was yes!) And it will be updated in the re-NBA region

9 Should we allow NBAs to program variables? - Yes and make the updates occur to a new Re-NBA region.

Regards,
Doug

I reviewed all my emails and notes from the meeting and this mail is an attempt to consolidate everything.

If no one else wants to volunteer to crank the revision on Arturo's 890-4 to 890-5, I can go ahead and do it this week.

I found the following Mantis items which I think we should consider together with the overall 890 proposal:
http://www.verilog.org/svdb/bug_view_page.php?bug_id=0000608
http://www.verilog.org/svdb/bug_view_page.php?bug_id=0000609

http://www.verilog.org/svdb/bug_view_page.php?bug_id=0000597
http://www.verilog.org/svdb/bug_view_page.php?bug_id=0000239
http://www.verilog.org/svdb/bug_view_page.php?bug_id=0000236

I'm not a big fan of 609.

The others are pretty straightforward.

In a very interesting coincidence, Eugene Zhang apparently defined our most recent ##0 behavior way back in Oct 2004! See Mantis #236 above.

Also recall 594, which is slated to come up right after 890:
http://www.verilog.org/svdb/bug_view_page.php?bug_id=0000594
I'd like to hear your thoughts on all these items.

Most are quite short. And they could have a good deal of influence on 890's proposal.