File : c:\wzmail\mailbox.fld
Messages : .

126
To: bobmu davec loup paulma
Cc: bryanwi ericfo jbal jeffpar leifp moshed ralfha scottlu
Bcc:
Return-receipt-to:
Record-folder: paulma
Subject: Windows 16 apps on Win 32 (WOW)

Recommended Plan
================

A. P1 does not wait for WOW, instead, there is a P1.1.

B. First shot at WOW is aimed at maximally smooth, maximally fast system that runs as many apps as possible. Support for generic hardware, win16 "private" graphics drivers, and the like will be left out. (i.e. Pick speed, smoothness of function, and schedule in exchange for not supporting some functions of some apps.)

C. Buy as much code/rights as possible from insignia.

D. Development Staffing:

WOW - 3 people. AviN believed key.

Base - 2 people. JeffPar believed key.

+ a manager.

E. Schedule:

Best:

(All staff start 15 Jan 91)

Ready for Beta - End Apr 92

Worst:

Ready for Beta - End Sep 92

WOW Project Objectives
======================

To provide the ability to run applications written for Windows 3.0 to run as peers of native NT Win32 applications.

Note the user could run Win16 apps in a Win16Box without the need for this development. The ONLY benefits of WOW are the seamlessness of user interface and Win16 apps would be able to use Native printer drivers, which should work better. There is NO other functionality provided.

Most of the technology should be reusable to run Win16 apps on the x86

version of NT (without the need of the emulator). That project is not discussed here and would require VDM development on the x86 platform.

To there user there would be a DOS box for installation of Win16 apps. Once installed they would look to the user like NT Native apps.

Recommendations
===============

If the goal is to have Windows Everywhere, then WOW provides a good story - just take any Windows 3.0 application out of the box and run it on a JAZZ machine under NT.

Before we go ahead with this project it is important to understand that there are a number of risks which could cause the project to slip or the performance to suffer. Thus, I would NOT recommend holding the product #1 if this component were to become late. Also it should be noted that resource for this project would have to be pulled from other project which would effect their end dates.

What were MVDM end dates?

Risks
=====

There a quite a lot of unknowns here which could all effect our ability to produce WOW on time:-

- Resources
  This group needs a strong tech lead - someone who understands all the details, understands Windows very well, is really good technically, and can lead a team. AviN is definitely this person.

- Insignia
  We are reliant on them for producing a 286 protect mode emulator. Their current emulator V2.0 is 286 real mode only. They have not yet demo'd V3.0 in protect mode.

  They are reliant on MS to ship them enough Jazz h/w so that they can port their emulation to Jazz.

  Insignia could be late completing the v3.0 product

  Insignia could be late porting v3.0 to MIPS/JAZZ

  Insignia could be slow and hard to work with

  Insignia could have lots of bugs

  Slow communications with the UK - usually a 1 day turn around on questions/answers

  Its unlikely they'd be able to complete their DOS Box project until more of Win32 is available for them. If Win32 were to slip then so would their porting process.

  Supporting Jazz hardware in the UK could slow things down.

- Win32
  Win32 is late or behind schedule (especially GDI). They are still working on the Final decision.

92 undocumented GDI calls. GDI32 will not be supporting all of them this could lead to some compatibility problems.

- Performance
  Our current expectations of the Insignia performance are based on scaling of their V2.0 product and the scaling we should get based on the R4000 processor. Also that their 3.0 Product should be 1.5x to 2x faster than their 2.0 product.

  Performance of WOW DOSEM Win32 layers

- Generic H/W Support
  It might not be possible to get EISA plug in cards to function correctly in this environment. It is might be that the performance of the emulator might be sufficiently different for the add on cards to not function. This might well be a security hole.

- Security - all the Win16 apps are running in the same address space they will all share the same security level.

- Compatibility testing is really hard (as experience shows from Porthole)

- We don't get the right people on the team or the team won't start soon enough.

- Moving Target - new features to be compatible with in Win 3.1....

Minimizing Risks

Since there will be many risk involved it will be important to put the best people on the project so that they can cope with changes as they come along.

- Produce WOW layer for GDI in order of scheduled functions coming from GDI group.

- Flexibility in schedule, always try to have other items that can get moved around if something we are dependent on becomes late.

- Build emulation environment using second PC so as to not be reliant on Insignia's dates. Second PC runs DOS all calls to WOW come to NT using a serial link.

- Jazz - Insignia could most likely work on an R3000 emulator system until the R4000 has had all its bugs shaken out. We'd need a Jazz systems shipped to the UK (shouldn't be a problem)

- Could use real DOS and container files until DOSEmulation is available

- The Win16box could be completed by Insignia as a fall back plan for prod #1.

- Set up FAST link to Insignia for fast turnaround of email and new versions of the emulator.

- Insignia to stage their port in the following order:-

286 Emulation on Jazz + Interrupts (including MS Hooks)
Basic Virtual Machine Support (enough to boot DOS)
Rest of Virtual Machine - Keyboard, Mouse, Video, Com
Final SoftPC

That way we wouldn't have to wait for the complete port before we had a self hosted environment.

Impacts

Removing JeffPar SudeepB from MVDM for x86 will severely effect the end date for that project (removing half the team and the best players).

Removing AviN from Porthole would also have some effect to the current porthole plans.

Moving the Porthole test team to NT Test as soon as the Win Libraries for OS/2 testing is completed (May/June?). WOW is expected to require 0.5 of a performance person time upon code completion. This will have some effect on current performance plan.

Test Strategy

- Work-split with Insignia (NT Test reviews/approves Insignia's test plan and test results).

- Extensive Win16 apps compatibility testing utilizing a wide breadth of apps.

- Execution of Win API tests developed jointly by NT Test and Win/Dos test groups.

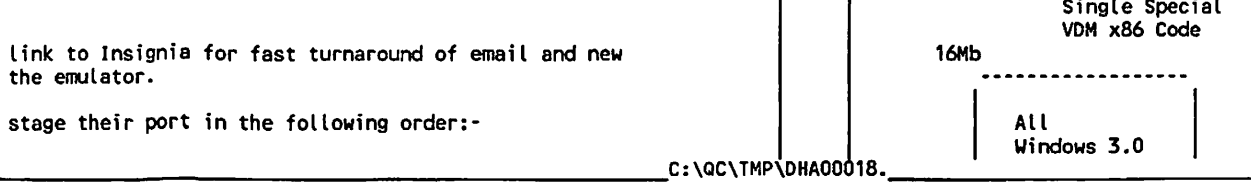
Functionality

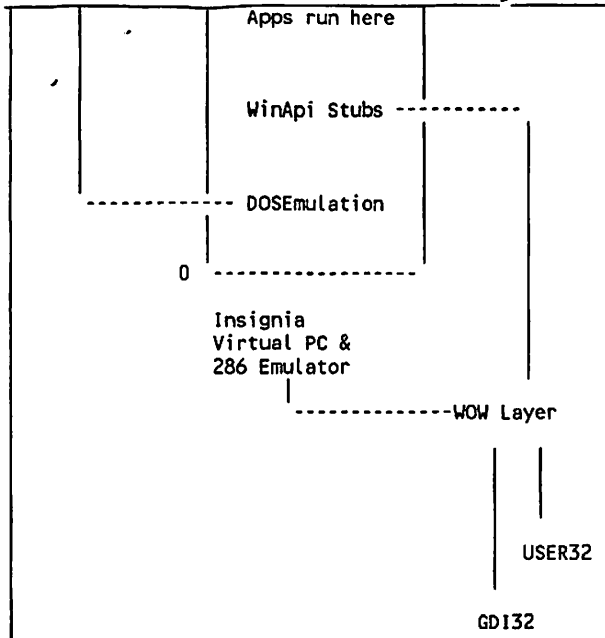
The table below shows the types of compatibility that could be supported, the Win16"Native" with 286 protect mode is the WOW project. The Win16BOX is the functionality which Insignia is working on for SUN.

	DOS BOX	Win16BOX	Win16"Native"
Real Mode	Y	Y	N
286 Protect Mode	Y	Y	Y#1
386 Protect Mode	N	N	N

Win16"Native"

DOS Emulation
NT File System





WOW provides the ability to run apps written for Windows 3.0 on NT OS/2 Jazz machine but look to the user as though they were native NT Windows apps. They will have:-

+ Transparent Access to the files system (only see 8.3 names)
+ DDE and Clipboard shared between Win16 and Win32
+ Look and feel like a native app
+ Use NT native printer drivers

Some functionality will could be cut for the first release in order to minimize development/test effort:-

- Generic H/W Support - ability for DOS/Win app to support an ISA plug in card, where there is no NT native support

- Win16 Drivers - eg app like PowerPoint have genographics driver

These items could be added for a future release or left out of the project. The generic h/w support has security impacts. The Win16 driver support would complicate the users view of applications; Win16 drivers could only be used by Win16 apps and not by Win32 apps.

Performance

Functionality provided by NT should look have a similar performance as other native applications - menus dialog boxes etc.

CPU bound apps will obviously run at the speed of the 286 emulator.

Windows Com apps might not be able to support high baud rates.

Win16BOX

An alternative to Windows 3.0 application support is to use the Insignia development without doing the WOW. This would be the same level of functionality that SUN will have:-

- File system access via container files or via DOS Redirector to local file system. This complicates the users view of his data. Drive D: maps to an NT director for example.

- All Windows 3.0 apps will run in a compatibility box. They will be a separate Windows 16 desktop with Windows program manager / print manager printer drivers etc.

- No clipboard DDE from win16 to win32

- look different to DOS Win32 product

We could use VDM DOS emulation to make the file system access transparent.

We could do some hacks to get Clipboard and DDE working.

Compatibility - should be excellent since we are really running Windows 3.0 without modification.

Performance

In this Model there is a stub Win16 screen driver which captures the drawing primitives from Windows and then turns those around to call the native Windowing system. Insignia have told us that a lot of their performance problems in the past have be due to the slowness of X Windows.

386 Protect Mode

Insignia's V3.0 technology will emulate a 286 only it will not support 386. So for this plan it will be ignored, but Insignia will be prototyping 386 in the future.

Open Issues

Security

NET Support

? Support for Win16 Net APIs
? Support for INT 5C NetBIOS
? Support for DOS NetAPIs - pipes, etc.

Size

The emulator is large and thus running win16 apps will require a lot of memory. Insignia design is based around execution speed rather than memory usage.

Time Line (Approx)

==========

GDI

Jazz Display Driver	Apr 91
GDI-Complete	Oct 91

Insignia

Demo 286 Prot mode	Feb 91
Jazz+NT shipped	Apr 91
Emulator Running	Aug 91
Final Release	Dec 91

WOW

Team Available	Jan 91
Design Complete	Mar 91
Func Complete	Nov 91
Integration Comp	Jan 92

Testing

Starts	Nov 91
Complete	Jul ~~91~~ 92

Beta Apr ~~91~~ 92

Assumptions

WOW completion date is tied to GDI complete and Final Insignia Complete
Best Case Team is available
Insignia would be able to provide staged functionality releases form MS.
Does not include any NLS support.

Insignia Source Code
====================

I recommend getting source rights to as much as we can - so for example we can use their VGA software emulation on x86 NT or x86 Windows. The contract should allow us maximum flexibility as to what we can do with the source code we receive (so we don't have to use it only with SoftPC). We should leave it open that we can write our own 286 emulator should we wish to at a later date.

WOW Compatibility
=================

It is believed that we could be more compatible than PortHole since we would be running all the Win16 apps in the same address space. The Win16 api should map better to Win32 support. Also we will have DOS emulation present so gaining more compatibility than PortHole.

- Journalling
  Win16 apps can set a windows hook to "journal" record or playback. This feature records / plays back hardware input. Journalling in this method will *only* work between other Win16 applications.

- Windows hooks
  Win16 applications can set windows hooks that get called when certain events happen within Windows. These hooks will *only* get called when a Win16 application triggers this event.

- Dos device drivers, hardware i/o, hardware interrupts.
  There must be dos some device drivers we can't support, some hardware i/o we won't recognize, some hardware interrupts we can't reproduce.

- Security. We need some discussion of how this affects a security rating. Must talk with JimK.

- Undocumented structures
  We are in the same boat as Porthole for undocumented structure accesses.

Appendix - Time Estimates
=========================

Base Estimates
==============

(mattfe)

- 286 Emulation - Insignia
  Virtual Machine - Insignia

NOTE: All time estimates for Insignia are based on our trip information rather than any quotations from Insignia directly.

- Insignia/MS Relationship
  design changes/UK trips (handling interface)

Best: 1 man month
Worst: 2 man month

- Integration With Insignia 286 Emulator
  1 Man Month

- Optimizations/Performance
  1-3 Man Months

- DOS Emulation - MS (sudeepb)
  DOS 5.0 Compatibility
  Transparent File System
  FCB Support

Best Case: 3 man months
Worst Case: 4 man months

- DOS 5 Utilities

Best Case: 1 man month
Worst Case: 4 man months

- Generic H/W Support - MS (Leave out of first release)
  Interrupt Management
  All unclaimed interrupts routed to VDMs
  DMA
  Ability for DOS app to performance DMA
  Memory Mapped IO

Ability to map EISA bus into memory map
Port IO
Ability to write to non trapped IO ports.

Best Case: 2 man months
Worst Case: 4 man months

WOW Estimates
=============
(scottlu's complete design note and estimates are on
\\hagar\scratch\scottlu\wow.txt
\wowest.txt)

This is an attempt to evaluate schedule and resource requirements specifically for Win16 emulation on RISC. Most of the code and effort is directly applicable to Win16 execution in a VDM.

What is in Win32:
---

* Task creation / termination
* Input compatibility with tasks
* Task wake / sleep primitives
* Task scheduler (non-preemptive scheduler)
* Some client/server logic to communicate the current task

Best case: 2 man months
Worst case: 3 man months

What is in WOW:
---

* Message and Api thunking, which includes:
  * Handle mapping
  * Pointer mapping
  * Structure mapping/aligning
  * Structure copying and subsequent freeing (in Intel space)
* Support for task based apis not present in Win32 (only a few)
* Support for callbacks (or 'return-backs') into emulator
* Client/server logic to switch stack to current task
* Support for outside Win16 load requests

Best case: 3 man months
Worst case: 6 man months

What is in emul/vdm space:

The Win16 memory allocator and Win16 loader live here.

Special win16/kernel:

* Layer logic to call kernel to switch tasks on demand
* Special api to convert emul 16 bit address into real linear address
* Layer logic to understand and perform callbacks (or 'return-backs')

* Special glue logic for WOW layer memory allocation (that does mem mgr function and automatically returns real linear address)

Special win16/gdi (optional):

* Special GDI which calls WOW for known devices, executes GDI16 for unknown devices

Other:

* Thunking setup for EVERY api and window message (setup real stack properly to point to emul stack arguments for WOW layer)

(Note: GDI32 will need to export a 16 bit metafile conversion service even for Win32 apps. WOW will take advantage of this support).

Best case: 3 man months
Worst case: 6 man months

Testing Environment DOS Machine-NT Link

* Macros to set/get emulator memory. This would be used for reading / writing to emulator memory. It would also be used by the WOW layer thunking code (both for callouts and callbacks)

* Macros for setting/getting emulator registers.

* Simple macros identifying what to do next - callback, continue executing, etc.

The rest of the code (special kernel, other emul/vdm stuff) is the same code that will exist in the eventual emulated environment.

Best case: 1 man month
Worst case: 2 man months

Misc:

Time for understanding, confusion, vacation, technical things I missed:

Best case: 2 man months
Worst case: 3 man months

Time for initial design.

Best case: 1 man month
Worst case 2 man months

Bug Fixing:

The DOS/Win32 group will be working on the same problem. Bugs found will either be in Win32 (some incompatibility that needs fixing) or in the Win16 mapping layer itself. Those bugs found in Win32 and fixed for Win16 apps will directly benefit the Win16/Emul effort. Since this is our greatest testing asset (running 16 bit apps) I believe this is where the USER32 group will spend much effort, which again should benefit the Win16/Emul effort.

The Win16/Emul layer will certainly have a bug crop of its own but it is a big help if half the bugs are being addressed by the similar effort in the DOS group.

Another interesting point is that application testing turned out to be one of the biggest tasks under Porthole. Most of the bugs are in Porthole, and this is because Porthole is totally new code and aims at reproducing Windows on top of PM. WOW isn't a rewrite of the api but a mapping on top of the 32 bit version of the api, and should require a smaller testing effort.

Best case: 5 people, 6 man months each
Worst case: 5 people, 10 man months each
(This would run in parallel with MosheD's test period)

Staging and time estimations:
---

5 people is optimum for this group. Everything can be done in parallel, but the final testing is a hit for each member of the group (as indicated). 3 - working on WOW, 2 - working on Base (DOS emulation etc.)

After considering the staging of implementation, looking at what can be done in parallel, what needs to be done in serial, etc., I have:

Best case: 8-10 months for a best case 5 person group.
Worst case: 15-18 months for a worst case 5 person group.

"Best case group" is AviN JeffPar SudeepB another D11/D12
"Worst case group" is a good lead (D12 at least but without prior knowledge of Windows) with 4 D11-D12 people.

* If WOW is done in 10 months it's done sooner than Win32 itself (this won't happen).

* Take it for granted that at least bug fixing will continue on the WOW layer (or in Win32 for Win16 apps) until NT ships.

Regardless, from these dates it implies WOW can be contained within NT product one, probably without affecting the schedule.

Group Players

This group needs a strong lead - someone who understands all the details, understands Windows very well, is really good technically, and can lead a team. AviN is definitely this person. (I'm not sure he's available though - he's on Porthole right now).

I know JeffPar is also very good technically, and he might be more available than Avi. He doesn't knows much about Windows but I am sure he would learn quickly

Avi and Jeff together would be the best you can get.

TESTING ESTIMATES

(MosheD)

Given that WOW testing will require the same breadth of Apps as Porthole, WOW apps testing team size needs to be the same as the porthole test team size. It is assumed that the entire Porthole test team will move to NT Test in June/july time frame. Their expertise will fully apply to WOW testing.

It is assumed that the number of bugs/compatibility problems will be significantly smaller than in Porthole (Porthole testing started in Jan/90, 2500 bugs/problems have been raised so far).

Testing/bug fixing duration from WOW development completion date to ship date:..... 6-10 months

File : c:\wzmail\mailbox.fld
Messages : .

127
From jeffpar Wed Dec 5 22:06:33 1990
To: mattfe
Cc: bryanwi
Subject: Times
Date: Wed Dec 05 22:06:28 1990

Based on the times shown below:

SoftPC v2.0 on Sparc is 1.2x a 6Mhz AT (Int SPECMARKS: 9.5)
SoftPC v2.0 on R2000 is 1.1x a 6Mhz AT (Int SPECMARKS: 11.3)
SoftPC v2.0 on R3000 is 1.5x a 6Mhz AT (Int SPECMARKS: 19.3)
SoftPC v2.0 on R4000 is ???? a 6Mhz AT (Int SPECMARKS: 37.0)

Extrapolating from this to predict performance of SoftPC v3.0 on Jazz, assuming a minimum 1.5x (max 2.0x) speed improvement of v3.0 over v2.0, and a minimum 1.5x (max 1.9x) speed improvement of the R4000 over the R3000, suggests that:

SoftPC v3.0 on R4000 is 3.4x a 6Mhz AT, at worst (ie, 20Mhz AT)
SoftPC v3.0 on R4000 is 5.7x a 6Mhz AT, at best (ie, 34Mhz AT)

Looking at compute-bound performance only (ie, spreadsheet recalc times), the ratios are:

SoftPC v2.0 on Sparc is 1.2x a 6Mhz AT (Int SPECMARKS: 9.5)
SoftPC v2.0 on R2000 is 1.4x a 6Mhz AT (Int SPECMARKS: 11.3)
SoftPC v2.0 on R3000 is 2.2x a 6Mhz AT (Int SPECMARKS: 19.3)

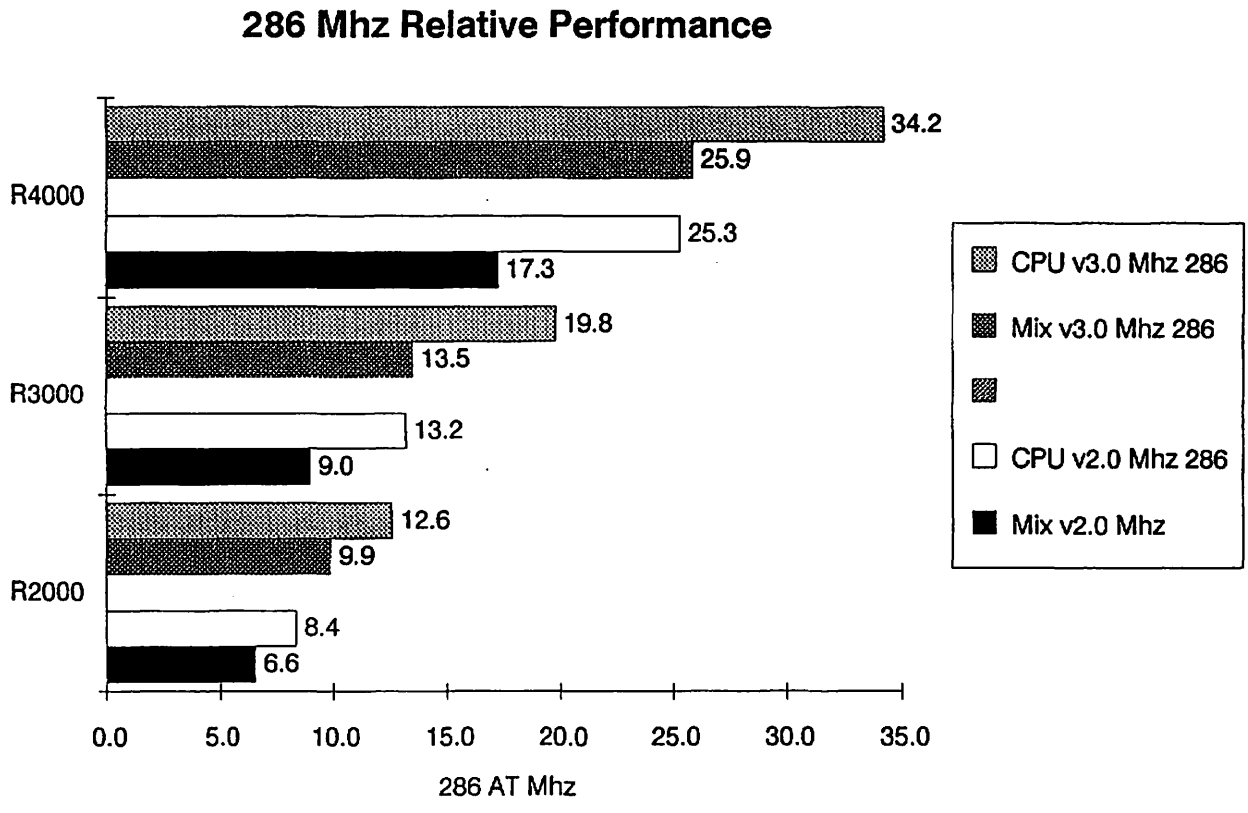
And the corresponding extrapolations become:

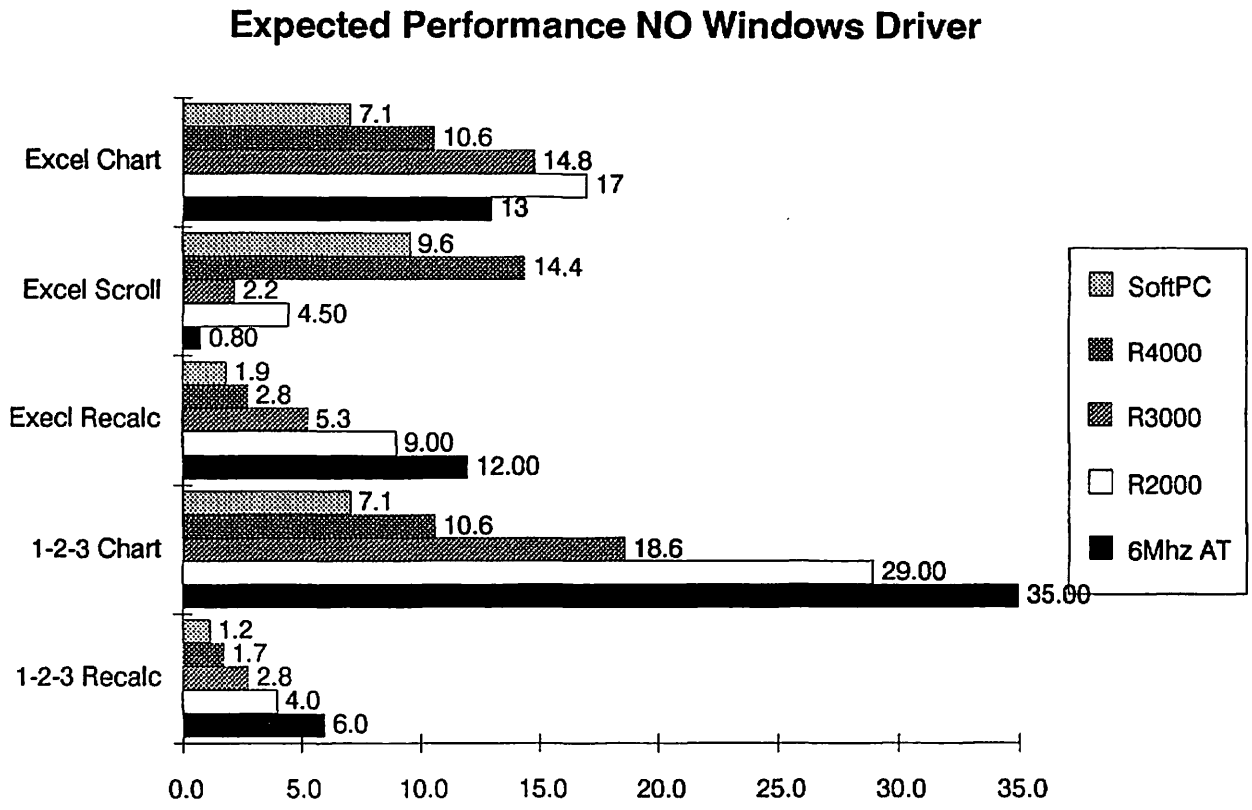
SoftPC v3.0 on R4000 is 5.0x a 6Mhz AT, at worst (ie, 30Mhz AT)
SoftPC v3.0 on R4000 is 6.3x a 6Mhz AT, at best (ie, 38Mhz AT)

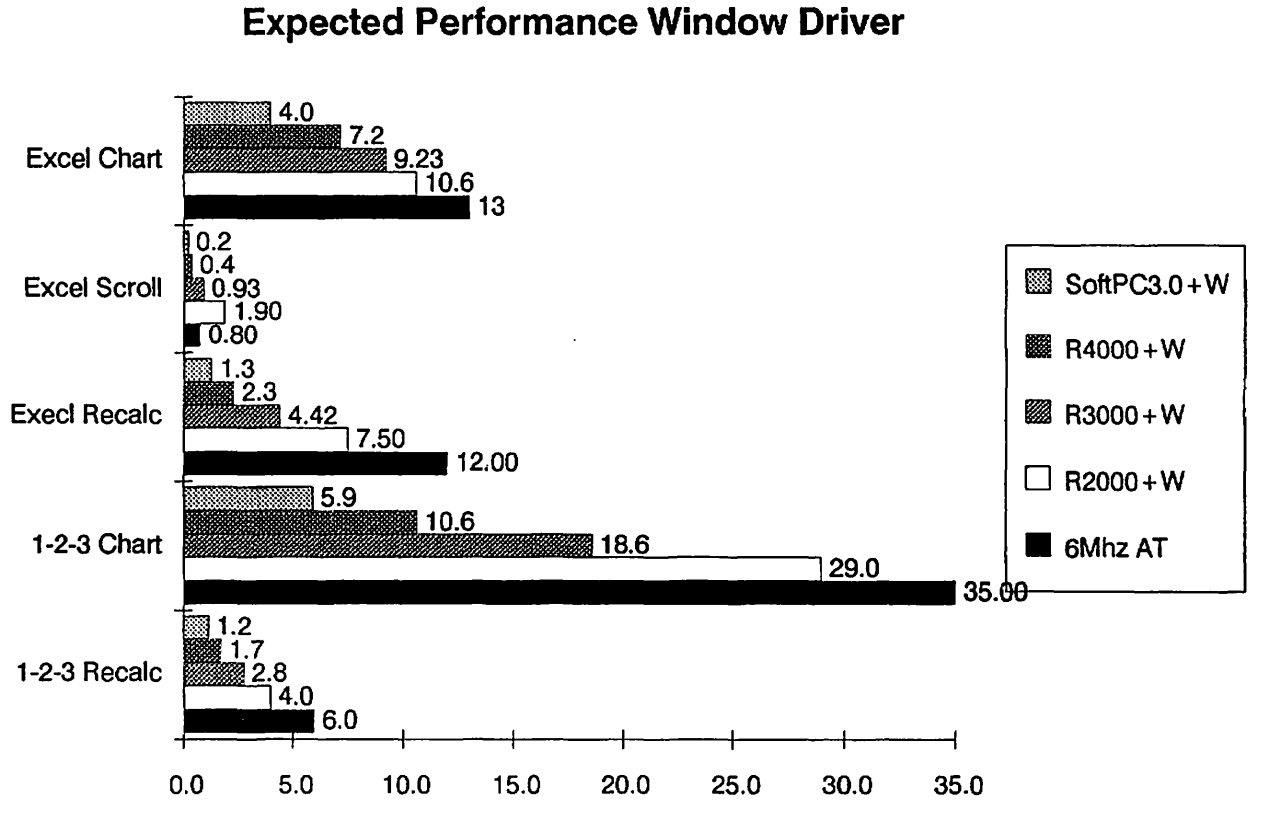
But extrapolations based on compute-bound times only should be regarded with pessimism. It's safe to say that, at a minimum, overall performance should be equivalent to a 20Mhz 386.

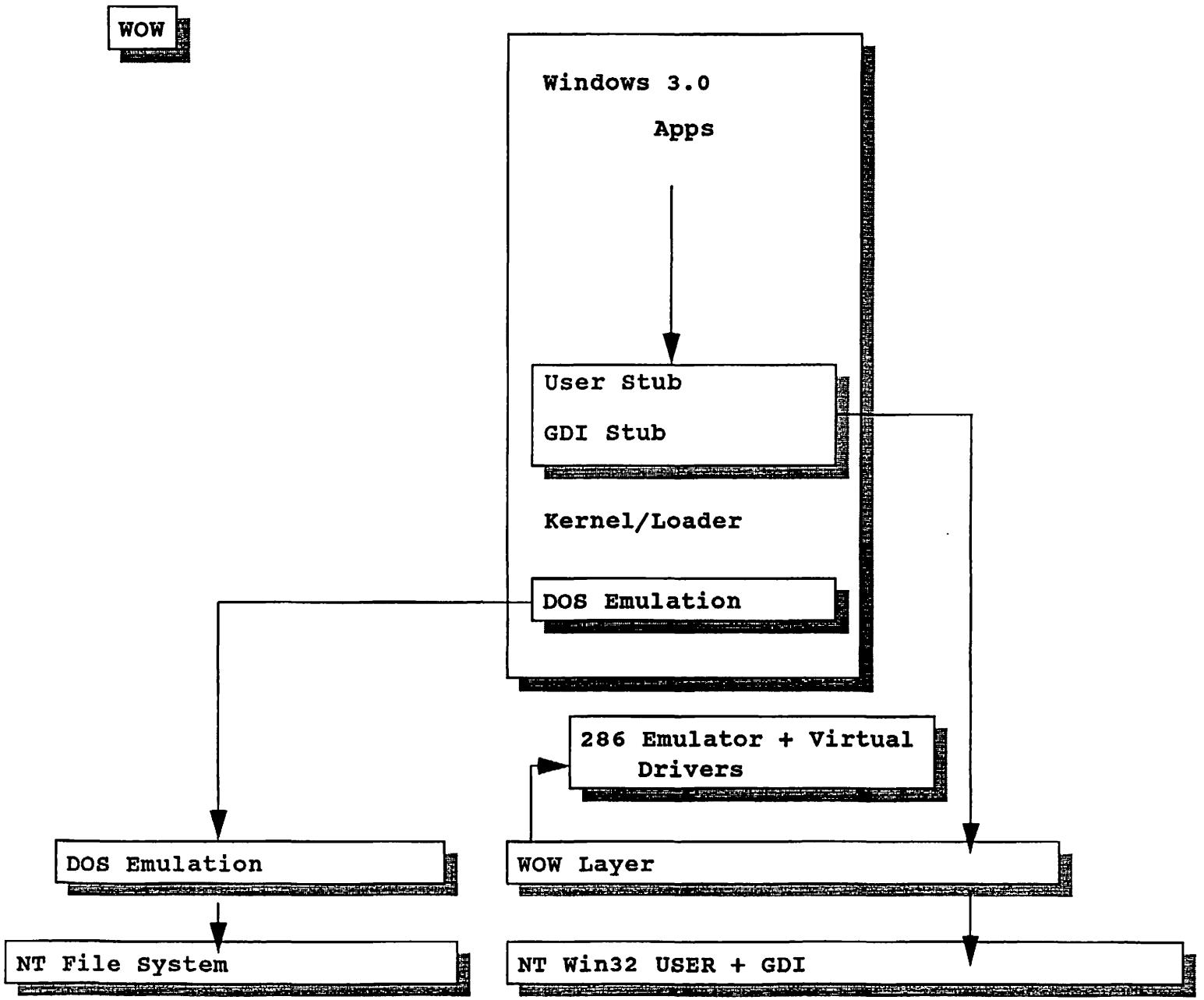
Task	6Mhz IBM-AT	Sparc	R2000	R2000 w/Windows Driver	R3000
1-2-3 recalc	6.0	3.5	4.0	(1)	2.8
1-2-3 chart	35.0	19.0	29.0	(1)	18.6
Excel recalc	12.0	12.0	9.0	7.5	5.3
Excel scroll	0.8	3.5	4.5	1.9	2.2
Excel chart	13.0	16.0	17.0	10.6	14.8

(1) Non-Windows application (not affected)


286 Mhz Relative Performance
R4000
R3000
R2000
34.2
25.9
25.3
17.3
19.8
13.5
13.2
9.0
12.6
9.9
8.4
6.6
0.0
5.0
10.0
15.0
20.0
25.0
30.0
35.0
286 AT Mhz
CPU v3.0 Mhz 286
Mix v3.0 Mhz 286
CPU v2.0 Mhz 286
Mix v2.0 Mhz


Expected Performance NO Windows Driver
Excel Chart
7.1
10.6
14.8
17
13
Excel Scroll
9.6
14.4
2.2
4.50
0.80
Execl Recalc
1.9
2.8
5.3
9.00
12.00
1-2-3 Chart
7.1
10.6
18.6
29.00
35.00
1-2-3 Recalc
1.2
1.7
2.8
4.0
6.0
0.0
5.0
10.0
15.0
20.0
25.0
30.0
35.0
SoftPC
R4000
R3000
R2000
6Mhz AT


Expected Performance Window Driver
Excel Chart
4.0
7.2
9.23
10.6
13
Excel Scroll
0.2
0.4
0.93
1.90
0.80
Execl Recalc
1.3
2.3
4.42
7.50
12.00
1-2-3 Chart
5.9
10.6
18.6
29.0
35.00
1-2-3 Recalc
1.2
1.7
2.8
4.0
6.0
0.0
5.0
10.0
15.0
20.0
25.0
30.0
35.0
SoftPC3.0+W
R4000+W
R3000+W
R2000+W
6Mhz AT



MattFe Dec 90