Microsoft Corporation
One Microsoft Way
Redmond, WA 98052-6399

Tel 206 882 8080
Telex 160520
Fax 206 883 8101

BillG

# Microsoft Memo

To: Charles Fitzgerald, Lloyd Frink, Sam Furukawa, Aaron Getz, Rob Glaser, Bruce Jacobsen, Brian McDonald, Nathan Myhrvold, Daniel Petre, Greg Riker, Darryl Rubin, Brad Silverberg, Pradeep Singh, Greg Slyngstad, Charles Stevens

From: Bill Gates

Cc: Steve Ballmer, Susan Boeschen, Adam Bosworth, Chris Graham, Joachim Kempin, Chris Larson, Tom Lennon, Mike Maples, Paul Maritz, Tandy Trower, Raleigh Roark, John Sabol, Steve Shaiman, Charles Simonyi

Date: February 25, 1992

Subject: Visual Basic+ Palmtop!

## Visual Basic+ Palmtop!

There are two bases of applications which are supported by Microsoft systems software: DOS applications and Windows applications. Unfortunately, neither of these sets of existing applications is well suited for palmtop computers. Users will not care to use most of these applications on their palmtops. The applications will not run very well on the palmtop hardware of the next three years. The standard system-software APIs make it too difficult to define a custom environment by creating new applications. The data format standards, even those promoted in Windows, are not high-level enough for many palmtop scenarios, such as multiple applications working off of the same schedule.

Based on the idea of the Model 100, which was mentioned recently in a great brainstorming memo from Lloyd, I suggest we use Visual Basic as the defining API for a palmtop PC.

Visual Basic is good at creating the types of applications need on a palmtop machine. The Pen group has already seen the majority of Pen applications created in Visual Basic. The small size of Visual Basic applications (without the run-time) allows the use of a slow serial or infrared links or low capacity JEIDA cards to transfer new programs from the PC to the small machine. However, Visual Basic by itself is not enough. With the right additions, mostly a rich high-level API to a set of key file formats, it could be made into an excellent palmtop environment.

In terms of a high-level API to key file formats, the Model 100 was truly innovative. For those who don't remember the Model 100, I will repeat the key features which are relevant. The model 100 had a Basic interpreter built in. The interpreter provided full access to all data files and was extensible for new device types. The 'Shell' was a simple listing of all data files and tools. The second version of the shell allowed you to organize files in one listing or grouped by types. All data files were simple text files which supported good search operations. All of the code for the model 100 resided in a 32k ROM - 24k for the Basic and 8k for the text capabilities.



MICROSOFT

The Model 100 was physically much larger than the machine I am discussing here, although the LCD resolution (320x200) was the same.

Because of the large size of the Model 100 compared to a palmtop, the markets for these products are different. The Model 100 lost out to DOS compatible machines. However, I believe the tradeoff of the Model 100 - focused functionality with a tailored interface - makes perfect sense for the palmtop market of the next 3 years.

The key addition required for Visual Basic is a high level API to documents and 'lists'. The schedule management and name/address management will be done using the 'list' API. This API would be both a subset and a superset of the Visual Basic extensions done for Cirrus. A simply query capability has to be built in. The ability to take a query result and display it for the user needs to be built in. List editing needs to be built in. Lists and Documents will be in memory files stored in a flat list. However, the shell will let you organize them in a few different ways. We would not support OLE capabilities.

We would align many of our applications to these new APIs and file formats. The calendar applet we bundle with Windows, our mail add-on scheduling package, and the pre-defined calendar package for the palmtop, would use the same APIs and share code and user-interface as much as possible. The note taking package for pen computers, the word processing applet in Windows, and the note taker in this palmtop PC would all be aligned. Where reasonable, this would be a subset of WinWord. The standards set by our file formats and the interchangability between PCs and the palmtop device would be crucial to our success.

A great deal of usability work is needed in order to define the user interface for the palmtop machine. I am assuming a gray scale LCD will be used, though a higher end color version should be supported. The applications for the machine would be constructed in Visual Basic and the source of these would be made freely available. The built in applications would be in ROM. The degree to which customization will be done on the palmtop versus on a normal PC will be up to the user, although I suspect it will mostly be done on a normal PC.

Someone might be asking - doesn't this imply all of Windows and why not just put it all in there? I don't think the cost/benefit of full windows is very good. For example, the ability to run DOS applications is unimportant. The shell makes no sense. The need to subset the API is unclear and depends on the cost of ROM. If costs are tight, we might have to limit the type of programming that can be done on the palmtop machine itself. Issues like printing support need to be investigated. I expect a machine with 1 megabyte of ROM and 1 megabyte of RAM will be appropriate. If we have 2 megabytes of ROM, we will have to spend less effort on the project, since we won't have to strip Windows as much and we can use VB without too many changes.

## Hardware features:

LCD (320x200) - touch sensitive Screen. I expect most user interaction would be touch based. In order to make this work, we will have to provide a lot of user feedback. Allowing a pen to be used for text, ink, and gestures should be investigated; the small physical size may make this difficult. I don't know if the gestures will be made using a finger only, something held in the hand, or something placed on the finger. I believe that the future desktop mouse will be a tablet device, somewhat bigger than the size of this machine, used in conjunction with

something on the finger, in order to increase precision. The same solution may be able to work on this device. Solving the user input problem is CRITICAL to making this machine a success. A keyboard would ruin the device (an optional keyboard should still be available).

Processor. Two years ago, I would have said an 8086. A year ago, I would have said a 286. Today, it is possible Intel would provide the right price ($4) and power consumption for this device to use a 386. We discussed this with them at an executive meeting in early February, and they agreed they need to do something in this segment.

Sound. I think sound will be important in order to give feedback. I don't think voice input can be used for at least a few years. The SPAG efforts on our sound card and the Dragon voice recognition code will give us some experience here. It is an interesting question what we should use sound feedback for, other than alarm clock beeps.

Storage. The device should have no floppy, hard disk or CD-Rom. It will not compete with the DiscMan/Bookman. A JEIDA card connector almost certainly makes sense, both for adding specific functionality and for data transport.

Communication. Unfortunately, desktop machines don't have JEIDA cards, so they can not be used to move data between the palmtop and the normal PC. This does bring up the question of whether Microsoft should promote a JEIDA connector on desktop machines. A high speed parallel port (SPAG has done some interesting work on defining the next generation parallel port) or a serial port might make sense. While it is fairly cheap to have both fax and normal modem capability, I am not sure of how well Fax would work on such a small machine. When a modem is included, or even when it isn't, a phone auto dialer might make sense. I'm not sure you can really hold a speaker near to a phone receiver and dial. It is unfortunate that bar code reading never caught on. I did code for the model 100 which read the bar code format, and we had BYTE magazine publish software this way (it is amazingly dense). Infrared communication is catching on for some things, but I don't believe it will be widespread enough to include in this machine. The biggest problem with this machine has to do with getting relevant real-world data in and out of the machine.

Scanner? One very interesting idea, that Greg Riker brought up recently, was including a hand scanner capability in a small machine. Even though you couldn't recognize most of the scanned text, you could recognize indentation and thus provide rich outline-style interaction with the information. For example, when you go to a conference, they would publish their schedule in this outline format and you would just scan it in. Information is not worth putting in a computer if you have to retype it! OCR type recognition software has advanced a lot. While I don't see this technology in the first generation machine, it is a critical technology that we need to track.

## Software:

Lloyd's memo talks about most of the possibilities. Software for this machine should certainly include a calculator, name/address list, calendar, note taker, PIM, forms package, personal finance package, alarm clock, PC connection, and everything you can get with a Sharp wizard. It might also include a small spreadsheet, charting package, and maps. Hopefully, additional

programs will end up being only about 20k of code/data. This would allow us to have dozens built in.

The hard parts of creating the software is defining the high level APIs for the file formats. Making it really easy for a Visual Basic program to search a schedule, add something to a schedule, search a document or use the built in hardware is crucial - only by building all the applications in Visual Basic will we get the APIs right - any backsliding to C code for the built in stuff is unacceptable!

## Cost/Competition:

Unless it can be sold for under $500 it won't fly. It shouldn't cost more than 2x the price of the Wizard or an advanced calculator. This price will compete with the HP95 and many clones. There are many more palmtop projects going on that I had an idea of, 4 in Taiwan alone. We need to learn more about these.

Obviously, this will compete with the myriad of small devices Apple seems to be involved in. Unfortunately, we don't really understand the details of their plans. Another key competitor is GO. We have heard many times that they are doing "sub-notebook" machines. Many of GO's ISVs are doing scheduling, notetaking, forms, PIM, and communication applications, which are the only applications which are appropriate for both notebooks and palmtops. This means that GO's ISV assets will be about as strong as Microsoft's, even though they have much fewer applications.

Someone can say we aren't following our "Windows everywhere" philosophy, but our answer will simply be that we are defining an appropriate version of Windows for the current state of the art of palmtop machines. The API will be called Windows, even though it will be a subset. Whenever a part of the palmtop API threatens to become a superset of the normal Windows API, we will enhance normal Windows.

## Next steps:

I have come to the realization that waiting around for Windows to fit into one of these machines isn't enough. If we don't do something soon, the window of opportunity for this type of device will pass us by. We have an incredible opportunity to take advantage and strengthen the asset we have in Visual Basic. We also want to leverage the good work that has been done in the pen group and the consumer group and the advanced thinking going on concerning "Wallet PC".

The project is pragmatic enough that it can be done in either Nathan's advanced development group or in the systems software division. I don't want to do the project unless we have program management and development that understands the VB+ software concept and is willing to do very creative UI work.

After we ship Pen Windows, I would like to organize a concrete effort to pursue this. I will be working with Mike Maples, Nathan Myhrvold and Paul Maritz on how we staff this effort. I

think we will need 2 program managers and 5 developers to start. I want a specification that will allow this product to ship in the second half of 1993.

We could do some hardware prototyping for this device internally, but we aren't going to build it. We want to own all of the software. But we need one or two key partners for the hardware. Hewlett Packard and Sharp are good possibilities, although they both have problems. Hewlett Packard might charge too much and might ask for exclusivity. While they didn't ask for exclusivity with Lotus on the 95 project, they may not know of all of the 95 clone projects going on. Also, HP moves fairly slowly. Sharp would be the best marketing and technology partner, but they might wonder why they want to help open up this market to competition. We have been discussing Wallet PC with them, and preliminary indications are they want to work with us. Tandy is another possibility, their views in the 2/13 meeting agree with this memo. Another possibility is Dell. They have a strong image and strong engineering, and they would probably jump at the chance and do a good job. I doubt that Compaq, Zenith, or NEC make sense. I would like to brainstorm with Joachim and Sam to determine which partners would be best.

WHG/ag