E Ink, Haptic Touch and MEMS

Jake Schwartz

Consider yourself on a typical day with a collection of HP calculators, whereby you can reach for and use any one, depending upon the particular need at the moment. For balancing the check book, you might pick up an HP32SII or 11C; for computing the interest payments on your next car loan, you might grab the HP17bII+; and for converting kilometres per liter to miles per gallon or perhaps plotting a function curve the 50g. Now how about if by magic, all these machines could share the same stack and memory registers, so a problem started on one could be continued and completed on another without skipping a beat. What would be simulated here would be a “universal” HP calculator with multiple personalities, each with a real tactile keyboard (and keys in all the familiar places) and actual authentic display. A not-quite-fulfilling attempt at making this come true might be to install multiple HP calculator emulators onto a touchscreen device such as one of the recently-popular “smartphones”. In fact, now that Hewlett-Packard has made it own Voyager-series (12C, 15C) emulators available on the iPhone, the entire concept has been further legitimized. However, for those of us who prefer to feel the tactile feedback of real keys, this solution is far from optimal.

Touchscreen HP-Calculator Proposals Through The Years

The idea of an HP calculator operating on a touchscreen device is by no means new; in fact an early proposal of such a device appeared in the PPC Journal back in 1984 when suggestions were being solicited for the never-implemented “PPC44” club project. This one shown in figure 1a suggested a configuration of real keys in a numeric keypad below a touchscreen which occupied the remainder of the top surface. Not until around fifteen years later did we see prototypes of the ill-fated HP Xpander, with virtually the identical layout.

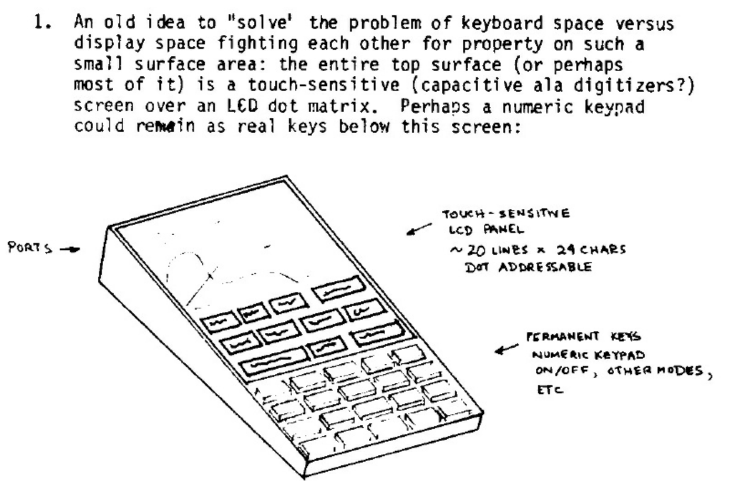

Figure 1a (above): From PPC Journal V11N1P42 - Proposed hardware layout for PPC44 machine in early 1984.

Figure 1b (right): Circa-2000 Xpander from the HP ACO group, never released.

In 1989, at the annual HP conference that year at Triton College in Melrose Park, Illinois (where we celebrated the HP41s tenth anniversary, by the way), a proposal was made, suggesting that the HP28 could be enhanced by locating all the “hard” keys on the right side and having a full touchscreen occupy the entire left side of the clamshell. This could allow multiple-row softkey menus, large graphics, a QWERTY keyboard and more. It was also the time of the release of the first Sharp Wizard, with its keys on one side and touch-sensitive clear surface on the other, under which a plug-in card with function markings denoted to the user the location of soft keys. The following year, the HP48 debuted with all the HP28 functionality on its single keyboard, but alas without the touchscreen.

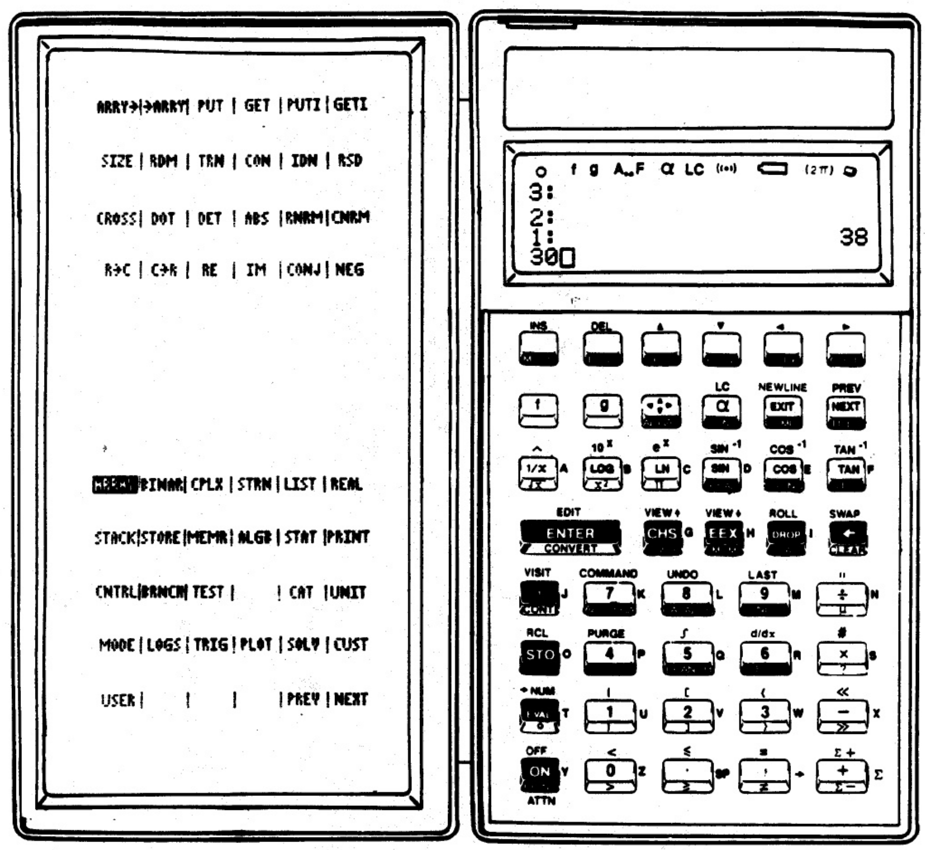

Figure 2. A proposed clamshell calc with HP28 functionality on the right and full touchscreen on the left.

Starting in the mid 1990s Sebastien Carlier’s emu48/49 program for the PC appeared and was later adapted to the Pocket PC PDAs running the Windows CE operating system. With its keyboard markup language (KML), users had the freedom to choose any calculator keyboard layout or image which could range from an actual photo of the machine to a line drawing. Keys would be mapped to x- and y-pixel positions within the KML file. After the HP49 version was developed, my favorite keyboard image was a modified key layout which restored the ENTER key’s size and position back to the HP48 arrangement, as shown in figure 3.

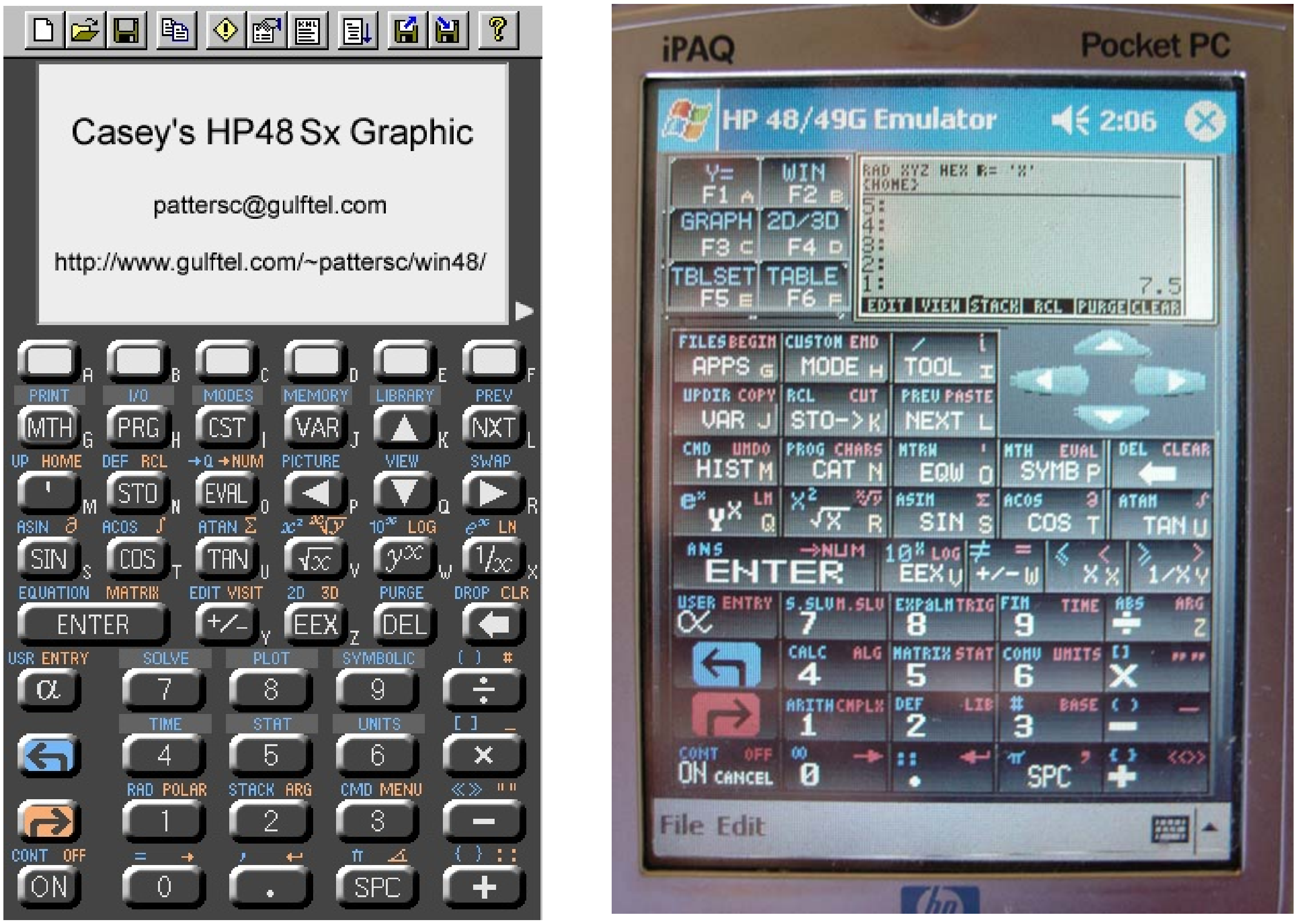

Figure 3. An example emu48 image file (left) and modified emu49 image file (right), both for the Pocket PC PDAs.

No Tactile Feedback

One of the problems with all these touchscreen solutions, from Xpander to iPhone has been the lack of tactile feedback like a real HP calculator keyboard. (In fact, it could be said that even some of the more recent actual HP calc keyboards have offered inconsistent key feel as compared to their predecessors from Corvallis.) Recently, a few companies have started offering “haptic” touchscreens on their products. These usually cause the device to vibrate momentarily when touched, giving the user a feeling of having positively pressed something on the flat surface. However, all the ones I have experienced have suffered with the same shortcoming; which is that feedback is always provided, no matter where the screen is touched. If a key has actually been pressed, I want feedback, but by the same token, if I press between the keys or anywhere else, I need that keyboard to do nothing. This represents the realistic way to know that the touch did not activate any key(s). It is felt that this difference would represent a significant advance toward simulating a real keyboard.

Electronic Ink and Keyboards

Another upcoming technology in consumer electronics has been the use of “electronic ink” (or E-ink) based displays. Two major advantages of these are extremely high contrast along with persistence when powered off. Both these attributes make E-ink ideal for digital signage, and examples of small digital signs using this technology were shown at a recent Consumer Electronics Show by the LG Company. Another currently-popular use for E-ink is in handheld digital book readers, such as the Amazon Kindle and Sony E-book Reader. However, it would be great to use in handhelds as well. One such idea, proposed in 2005 by Tim Wessman, won HP’s annual Design-the-Calculator contest. He suggested physical keys surrounded by an E-Ink surface where the key labels would appear and remain there (even if powered off), until changed. More recently, Samsung has developed a mobile phone called the “Alias 2”, with E-Ink on each key so the labels may change depending upon the currently-active application. (See figure 4.)

Figure 4.Left, Tim Wessman’s 2005 HP Design-A-Calculator contest winning entry with E-Ink “land” around blank real keys; Right, the Samsung Alias 2 mobile phone with E-Ink keyboard which changes with applications.

The Case for MEMS

To me, the ideal handheld display surface would be a touchscreen with tactile feedback exactly where the application needed it – and nowhere else. One idea I wondered about was whether this was a problem which might be solved by some sort of MEMS (MicroElectroMechanical Systems) solution. Recalling how Texas Instruments created a totally-original approach to projection television with their DLP (digital light processing) MEMS chip which consists of a large matrix of microscopic addressable, movable mirrors, perhaps a MEMS-based display/tactile screen could be devised for this use as well. Unless someone can confirm that something like this is on the drawing board or has already been done, consider the concept of a display consisting of an array of square “Surface Elements” (or perhaps “SurfEls”, for short?), whose dimensions are 75-by-75 to the inch. Each would be part of a MEMS device which could be individually addressed electronically. Like the DLP chip, these SurfEls would be in two possible states, “engaged” or “not engaged”, whereby when engaged, they would provide slight but definite mechanical feedback when pressed with a finger. Each one would also contain a 4-by-4 array of addressable e-ink pixels on its top surface, so as to comprise a 300 DPI display. (See figure 5.)

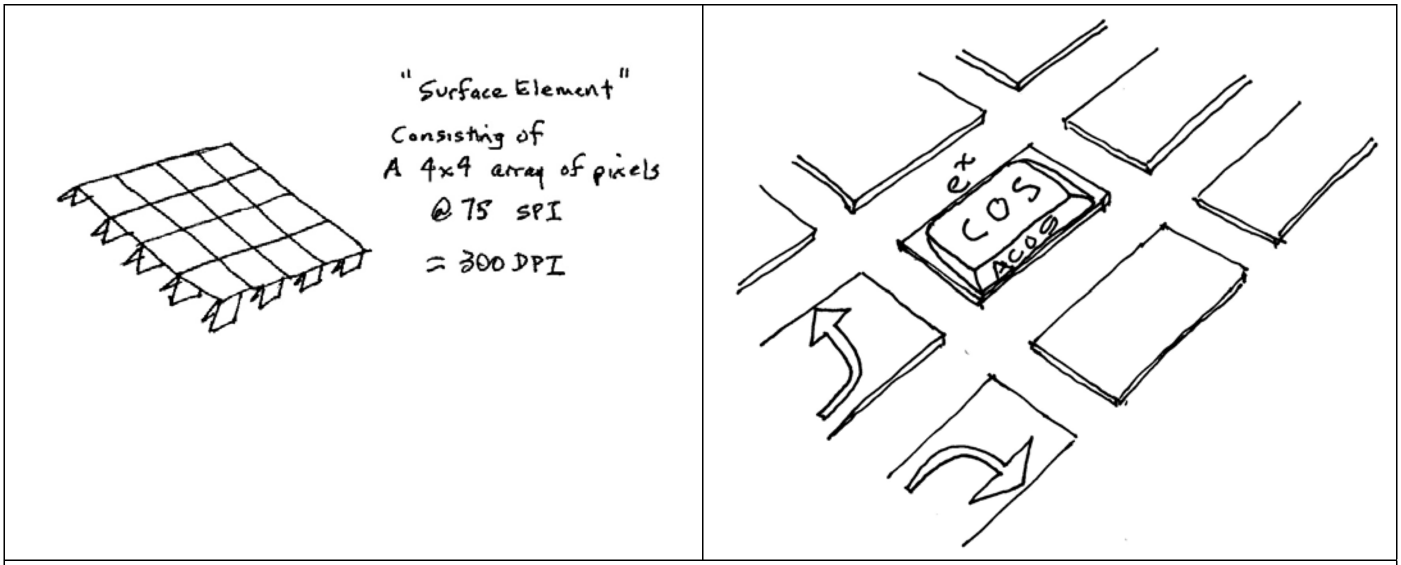

Figure 5. Left, a proposed “SurfaceElement”, shown in the “engaged” position; Right, a view of an E-ink./MEMS-based surface with calculator key labels drawn in the designated key positions.

In software, the upper portion of the surface could be designated as the calculator “display”, with the lower portion for the virtual keyboard. A KML-like application could not only specify the position and shape of the virtual keys, but could also provide image data for the portions of the E-ink display over each key. When activated, the Surface Elements in the designated key positions would be engaged and the images drawn. Pressing the surface between the designated keys would provide no feedback, but pressing in the area of a key would result in somewhat of a “click-like” feel. Should key presses cause a mode change, the drawn keys could be changed accordingly. In fact, if necessary, changing modes could completely change the virtual keyboard with respect to the position and number of keys as well. As a result, a machine could hold multiple calculator emulators in its firmware and could change between them via user commands, with virtual tactile keyboards changing at the same time. Imagine embedding Eric Smith’s Nonpareil emulator on a box like this, with an opening mode screen where the functions on the keyboard are labeled “Classic”, “Woodstock”, Spice”, “Voyager”, “Pioneer”, etc. and after making a selection, a subsequent keyboard would be labeled with calculator model numbers. That might be the HP fanatic’s ultimate dream machine.