Creating Icons for Leopard Buttons

Introduction

Among the new features that C-Max 2.0 brings to the Ocelot and Leopard controllers, one of the more sophisticated ones allows the user to create icons that can then be superimposed on Leopard touch button objects. This feature can be used in various ways to create highly customized screens that are more visually appealing and intuitive for the sometimes less technical end user. Professional installers can also provide a much more customer-oriented, value added service by allowing their clients to make specific choices and special requests as to what they would like to see on their screens.

The method used to implement the icons allows a given icon to be used on more then one button. This is done by linking an icon number to the button or text object. Doing it this way results in a considerable memory savings and eases maintenance and updating because only one icon drawing might need to be modified and then the new version automatically applies to all objects using it. To properly use this new feature, it is important that the screen object sizing be well understood beforehand. Let's start by reviewing some screen basics.

The Leopard screen has a graphic resolution of 320 pixels (individual dots) wide by 240 high. If you look at the Touchscreen Setup screen in C-Max you will notice that the grid divides the screen area in 10 horizontal touch squares by 6 vertical squares. Each square is thus 32 pixels wide by 40 pixels high (so they are not really "square", but we'll use that term for simplicity). A square is the minimum area that the Leopard can use to determine a button press (ie: pressing anywhere particular within a square makes no difference). The various screen button objects that can be created will always be composed of one or more of these touch squares. Contrasting with that, icons can be defined in almost any size, as long as their width is a multiple of 8 pixels. That means that icons can be created to cover only part of a button or its entire surface area. Furthermore, you can choose to make the original button frame invisible and make the icon itself become the button's image. This offers interesting possibilities that we will see later. Whatever the icon size is, it will always be centered in relation to the button it is linked to. The only minor constraint to the automatic centering is that the icon will always begin on an 8 pixel multiple in relation to the button's left edge. This means that if you create an icon that is 24 pixels or any other odd multiple of 8 wide, then that icon will display slightly to the left (4 pixels in fact) of the actual horizontal center position. If available memory constraints make you still want to do this, you can always compensate at icon design time by shifting your icon's image by four pixels to the right…

The other question you may have is: how many icons can you create ? The answer is: up to 200, or until the 48k of memory assigned to icon storage is used up, whichever occurs first. The 48k memory (49,152 bytes) area sets aside 1200 bytes for the icon "directory"; the rest is available for your icon designs. Figuring out the memory usage for an icon is simple: multiply the width of the icon by the height (in pixels), and then divide by 8. For example, the 48 x 32 icon designed in the example below equals 192 bytes. To give you an idea of how much memory is available, the 48k total memory size corresponds to about 5 full Leopard screens.

The icon display feature also incorporates the formerly separate "Screen bitmap" function to display on screen 0 when the Leopard is first powered up. As before, you first create a bitmap image that you want to use for this purpose, but you now assign it as icon# 0 in the System Map. It will then get loaded along with your other icons when you perform the download procedure explained in this application note. One advantage to this is that if you elect not to use a bitmap on screen 0, the memory it would have used becomes available for your other icons instead. The memory in question is part of the 48k total as explained above.

The best way to learn how to design and use icons is with real examples. The steps involved are the following: 1- Decide on the size (in pixels) of the icon you want to create based on what you want to use it for as discussed above. 2- Use a graphics editing program to create a black and white, bitmap format file of your icon. 3- Assign the bitmap file to an icon object number by editing the C-Max System Map and then download into the Leopard. 4- Design or update your touchscreen file to assign the icon object number to the touch button(s) you want to use it with and load the screen file (.tch) into the Leopard.

In this application note, we will present three examples of icon design and usage. The first example shows the detailed steps required to create a simple icon that will be displayed on top of a button object.

The second example uses a very large icon to create a whole screen image on which "hot spots" are shown that correspond to invisible buttons defined around the button displaying the large image. The third and last example shows a sample application using the new icon switching instruction for changing icons on buttons under program control.

Designing an icon

For this example, we will design a pair of icons, one showing a light bulb as being lit and the other as dark. We will make these 48 pixels wide and 32 pixels high. These will then be used as substitutes for the "On" and "Off" captions on buttons of various sizes. The 48 x 32 size was chosen here because in this example we will use them on "two square wide by one square wide " (2 x 1) buttons, one of the most commonly used sizes in Leopard screen design. A button this size is 64 pixels wide and 40 pixels high. If you look carefully, you will notice that the "raised border" look of the Leopard buttons uses 4 pixels along each edge. This means that the usable display area of a 2 x 1 button is reduced to 56 x 32. A width of 56 pixels corresponds to 7 bytes wide, not an even multiple of 8, so we'll reduce it by another 8 pixels, hence the 48 x 32 size.

The examples shown use the "Paint" program (Mspaint.exe) that comes standard with Microsoft Windows. Feel free to use any another graphics editing program of your choice.

Open the paint program (this is under Start → Programs → Accessories). This will give you a blank screen for a large image. Click on Image → Attributes as shown below (Fig. 1):

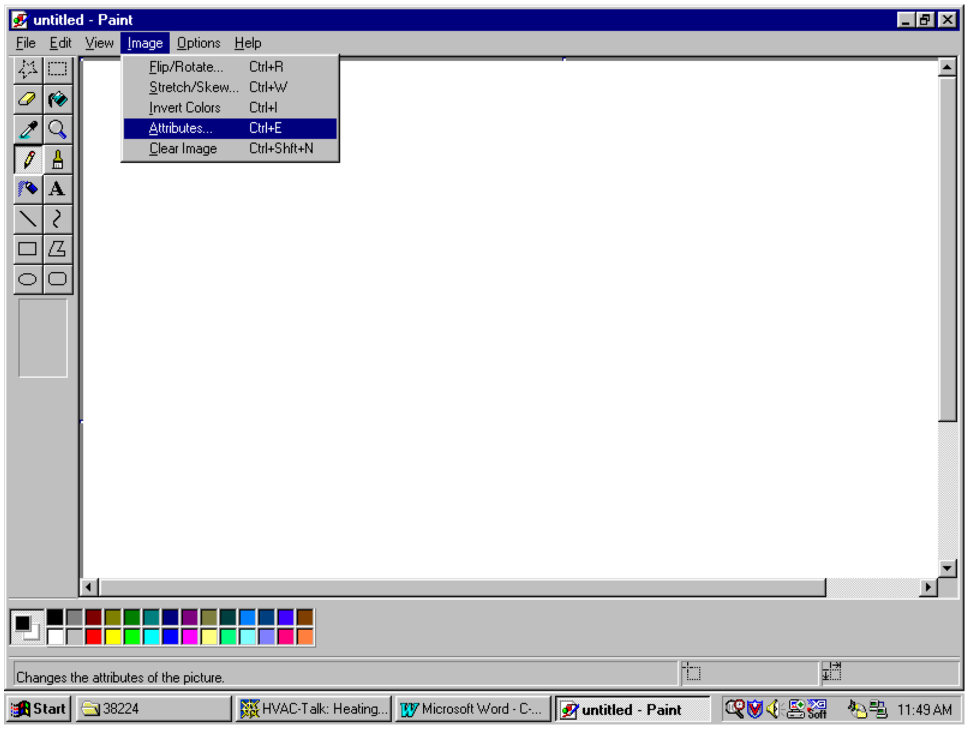
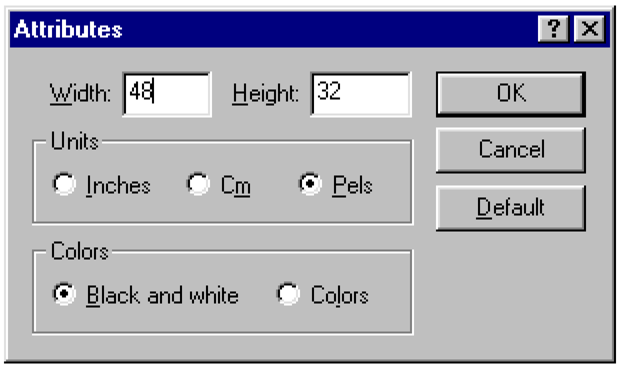


Fig. 1

This will open a smaller window allowing you to choose the image attributes of the bitmap file you will produce. Note that the width and height in pixels has been entered and that it will be a black and white image (Fig. 2)


Attributes

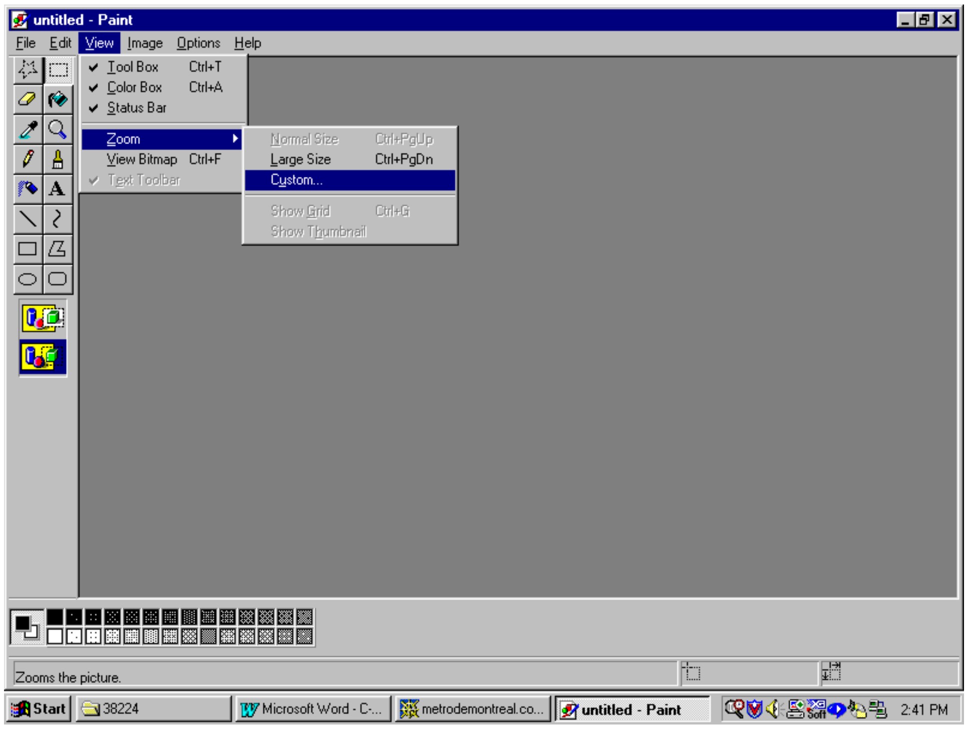
Width: 48 Height: 32

Units: Inches Cm [✓ Pels]

Colors: [✓ Black and white] Colors

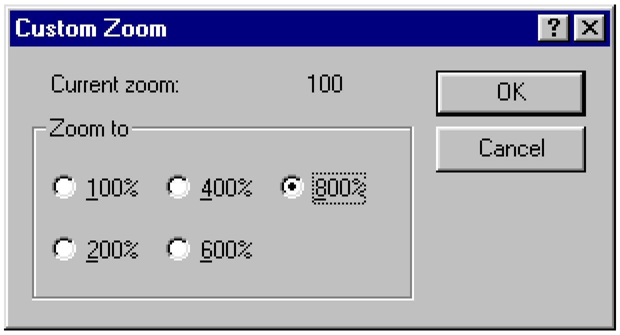
Fig. 2

When you click on “OK” you will probably get a warning message stating that converting to black and white cannot be undone. Just click on “Yes” to continue. You will now notice a rather small white rectangle where you are expected to create your icon… Unless you have lots of patience or terrific eyesight, you’ll want to zoom in on your image to make it easier to work with. Just click <u>View</u> → <u>Zoom</u> → <u>Custom…</u> to open up a custom zoom box and select 600% or 800% to bring it up to a much more workable size (Fig. 3 and Fig. 4)



Zooms the picture.

Fig. 3


Custom Zoom

Current zoom: 100

Zoom to: 100% 400% [✓ 800%] 200% 600%

Fig. 4

You may now use the various tools available in the program to create you caption design (Fig. 5)

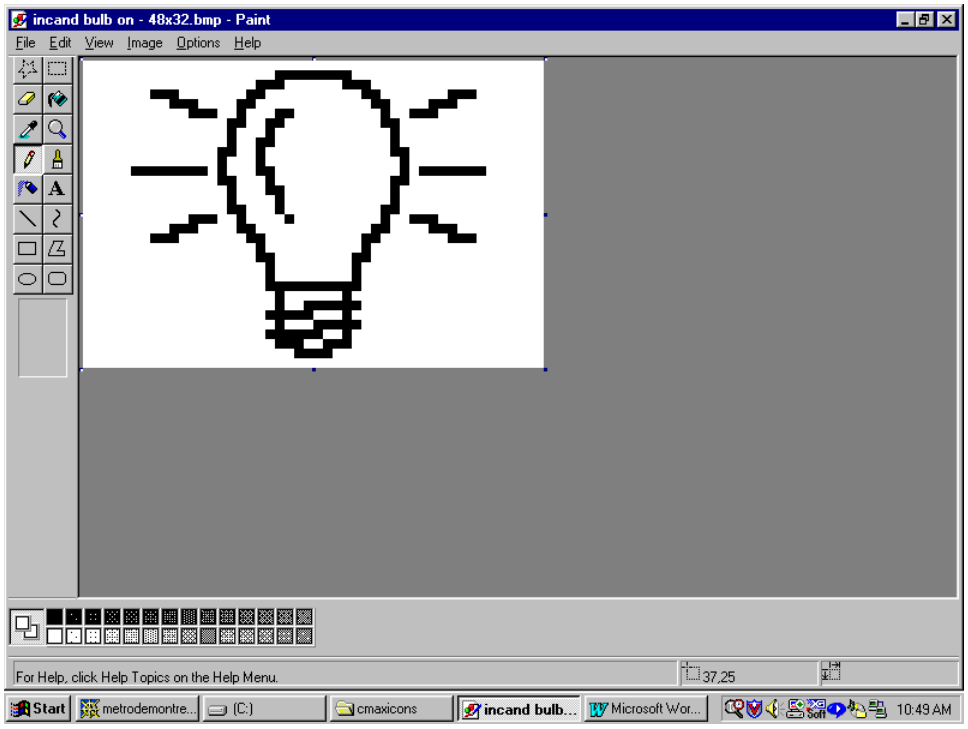

incand bulb on - 48x32.bmp - Paint
File Edit View Image Options Help
For Help, click Help Topics on the Help Menu.
37,25
Start
metrodemontre...
(C:)
cmaxicons
incand bulb...
Microsoft Wor...
10:49 AM

Fig. 5

The design will certainly have a "rough" look when viewed at such a large size but will look much better when viewed normally. You may do this at any time during your design work by simply turning off the zoom feature under View to take a look. Once you have finished, save the image using File → Save As… and give it a meaningful name. You may want to create a directory specifically for all your icons, or sort them by size or any other criteria. After your initial icon is designed, it becomes easy to design similar or related images by using the first one you did as a master. Let's create the "dark" bulb now by removing the surrounding rays and inverting the color inside the bulb (Fig. 6)

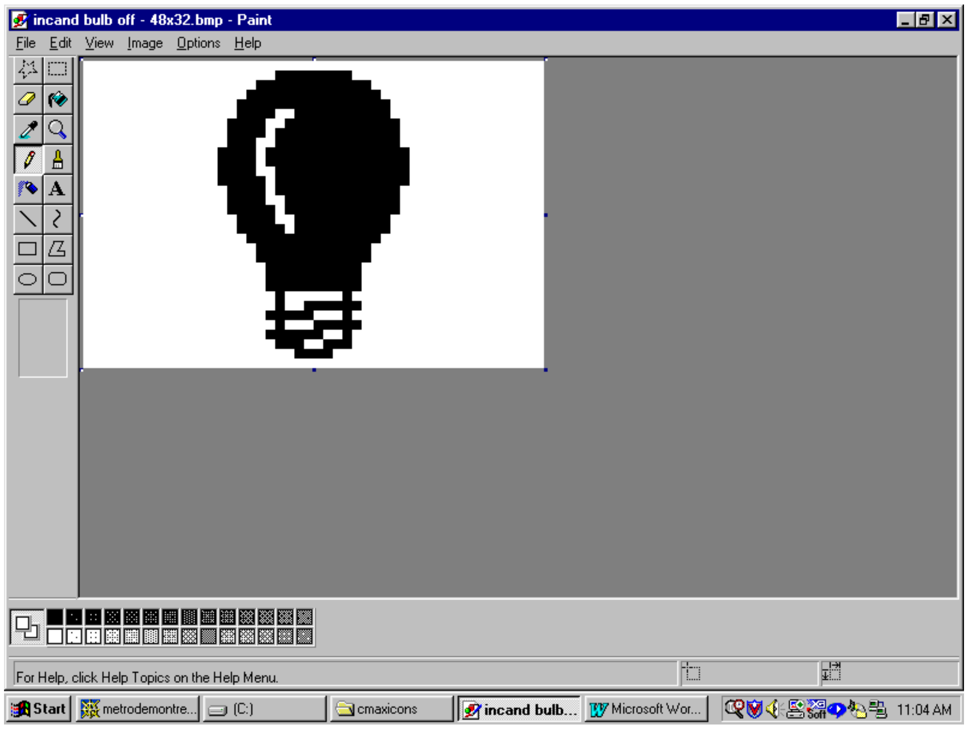
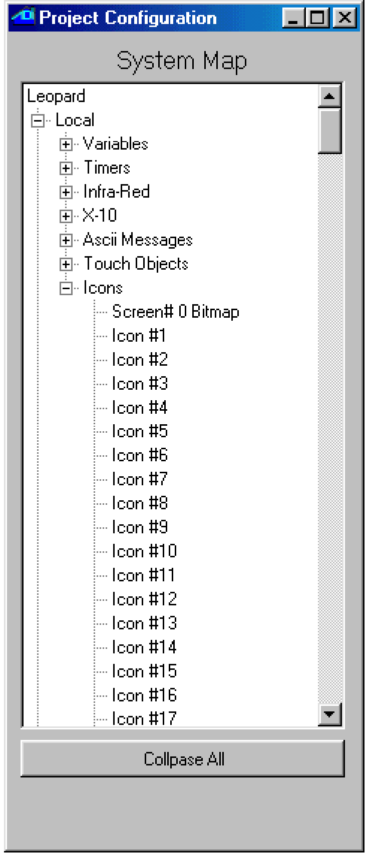

incand bulb off - 48x32.bmp - Paint
File Edit View Image Options Help
For Help, click Help Topics on the Help Menu.
Start
metrodemontre...
(C:)
cmaxicons
incand bulb...
Microsoft Wor...
11:04 AM

Fig. 6

One thing to consider when designing icons is the final desired dark/light "look" of the icon once displayed on a Leopard screen. In that respect, there is a difference between the original Leopard I and the newer Leopard II that you might want to take into consideration. The Leopard I displays buttons in "reverse video" (in relation to the icon design as seen above) while the Leopard II will show them as displayed above. The Leopard I reverses the colors so that the icon will blend in correctly against the already dark background of a button. In the case of a light bulb icon as we are doing here, we want the lit bulb to have the "light" color and the dark bulb to really look "dark". For a Leopard I we would thus make the lit bulb; the one with the rays, filled-in with black and the dark bulb with just the bulb outline instead so that the color inversion on the screen will give us the desired effect. Note that in either case, we'll leave the surrounding background white because that will allow a seamless display with the button's background over which the icon will display. If you need to invert the colors of your icon after designing it, you can very easily do this in Paint by clicking on Image → Invert Colors.

Once your icons are created, you must now load them into the Leopard's memory. This is done in C-Max 2.0 by opening the System Map configuration screen (Fig. 7):

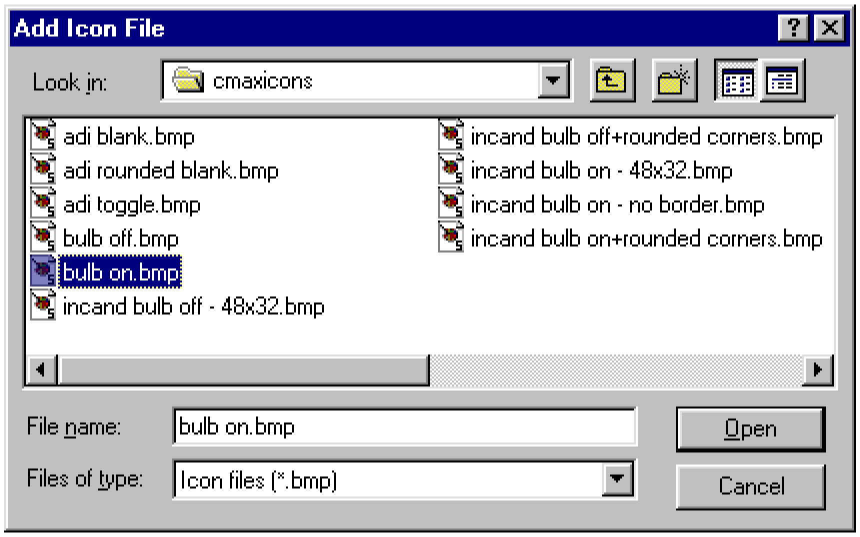

Project Configuration

System Map

```
Leopard
⊟ Local
   ⊞ Variables
   ⊞ Timers
   ⊞ Infra-Red
   ⊞ X-10
   ⊞ Ascii Messages
   ⊞ Touch Objects
   ⊟ Icons
      Screen# 0 Bitmap
      Icon #1
      Icon #2
      Icon #3
      Icon #4
      Icon #5
      Icon #6
      Icon #7
      Icon #8
      Icon #9
      Icon #10
      Icon #11
      Icon #12
      Icon #13
      Icon #14
      Icon #15
      Icon #16
      Icon #17
```

Fig. 7

Select “Icon #1” by clicking on it once and then right click on it to open the file browsing window (Fig. 8).


Add Icon File

Look in: cmaxicons

```
adi blank.bmp                   incand bulb off+rounded corners.bmp
adi rounded blank.bmp           incand bulb on - 48x32.bmp
adi toggle.bmp                  incand bulb on - no border.bmp
bulb off.bmp                    incand bulb on+rounded corners.bmp
bulb on.bmp
incand bulb off - 48x32.bmp
```

File name: bulb on.bmp

Files of type: Icon files (*.bmp)

Fig. 8

Select the icon file you want to assign to that icon number and click on “Open”. This will copy the icon file to your project directory (remember this fact; because if you create and keep your icons in a directory other then the project directory, like if you want to use them for several projects, you will need to re-copy any updated versions of the icon to the project directory). Once all your icon files have been assigned to icon numbers, you will need to load them into the Leopard’s memory. This is done using the “Download Project” selection under the “Project” pull down menu on the main C-Max screen (you can also perform any of the other downloads at this time too) and checking the “Download Icons” box (Fig. 9):

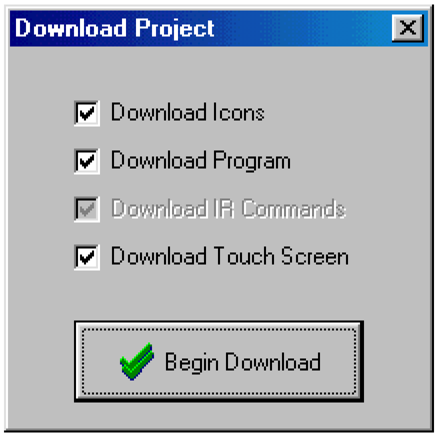


Fig. 9

Click on “Begin Download” and you will see the controller access screen showing the progress of the download operation. Once this is finished, you will now have the icons loaded into your Leopard. The only thing left to do now is to assign the icons to the screen buttons you want to display them on. Open the touchscreen editor and load your screen or create a new one and choose the button you want to put an icon on. Right click the button and click on “Enter Caption”. You will see that the caption editing box now has two new options: the icon number and a check box labeled “hidden border”. Leave that box unchecked for this example, and enter the icon number in the appropriate box. Note that we will not enter a caption since it is being replaced by the icon instead (Fig. 10):

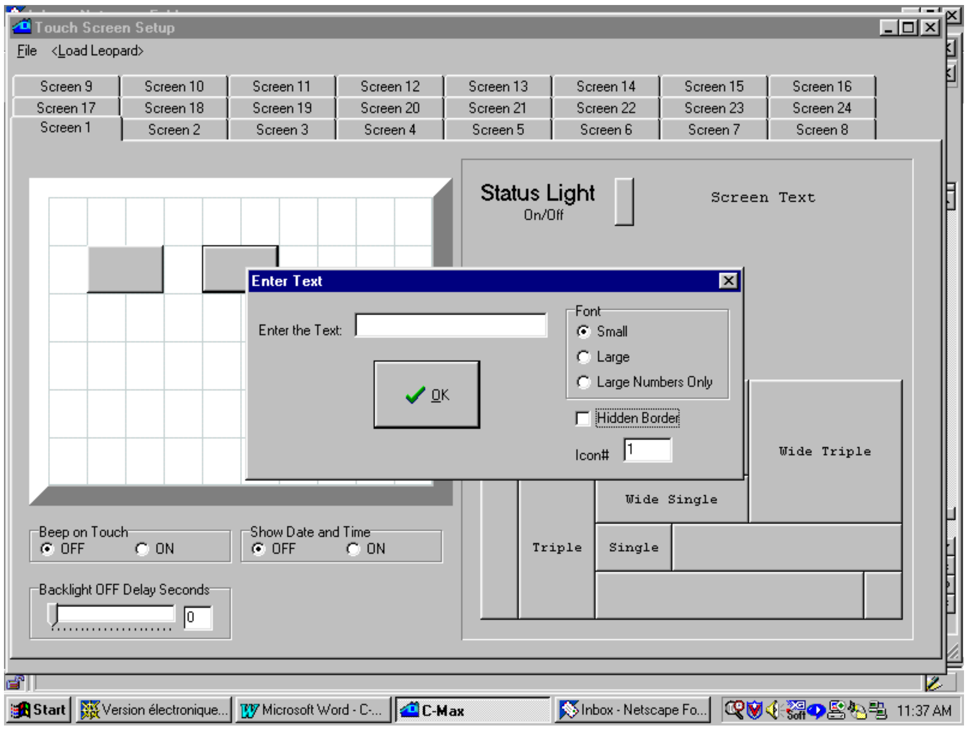


Fig 10

Do this for all the buttons you wish to use to display icons and download the touchscreen file by clicking on <Load Leopard> or with the new “Download Project” option on the main screen and see the results of your efforts! Here is an actual picture of various icons that were created, being displayed on a Leopard I screen (Fig 11):

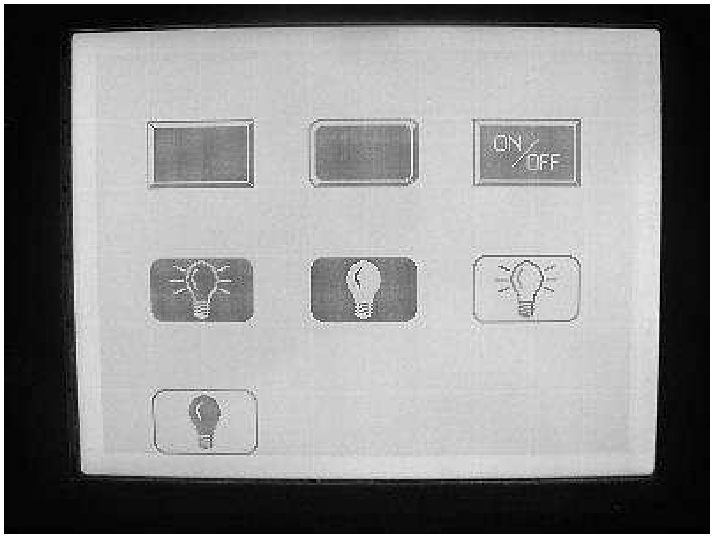


Fig. 11

The blank button in the upper left is not really a blank button but an icon made to look just like one. The button next to it is a variation made to look like a standard button but with rounded corners. You can put graphic designs or even hand created text like the "on/off" on the third button to create custom captions that the regular text captions cannot display. The four remaining icons were displayed using the "hidden border" checkbox option to make the original button outline invisible, the icon becomes the actual button outline. The ones displayed here are the full 64 x 40 dimensions of the button they cover, but they could be smaller or larger then the button they're on. Note however that no matter how small or large the displayed icon may be, the actual active area where the button press will be detected still corresponds to the original button object's size. Finally, creating a button with the hidden border box checked and assigning no icon or captions creates a totally invisible button, like the 8th button in the picture (…!). This can useful to access hidden menus and other such applications, or to create "hot spots" under the very large icon of another button as our next example will show:

A full screen image and hidden buttons

This second example shows how you can create one large icon to create the impression of a bitmap screen image with invisible buttons hidden underneath the "hot spots" in the picture. An icon can be created that is larger then the button it is displayed on; it's image will "spill over" the edges of the button but still be centered on it (although the actual "touch sensitive" area for that button remains the original button size, which is what makes this example possible). This time, we will create an image representing a home theater setup so that the user can easily find the buttons associated with each device. The first step is once again to start your favorite graphics editing program and layout your picture in such a way as to make the labeling of the hot spots coincide with the center of the hidden buttons that will respond to the screen presses. Here, we used Paintshop Pro because it allows us to set up a grid with adjustable gridline positions, which we set to 32 horizontally and 40 vertically. This gives us a drawing grid very similar to the one we see in the touchscreen editor. We can now start our screen design. For reasons you will see later, we chose to leave one horizontal row of touch squares out of this picture to use for a row of buttons along the bottom of the screen, so the bitmap here is actually 320 x 200 instead of 320 x 240 and is thus 5 touch squares high instead of 6 (Fig. 12). Notice how the various hand made captions aim to fall close to the center of the touch squares (except for the 4 buttons that make up the TV screen, we'll use 2 x 1 buttons for these).

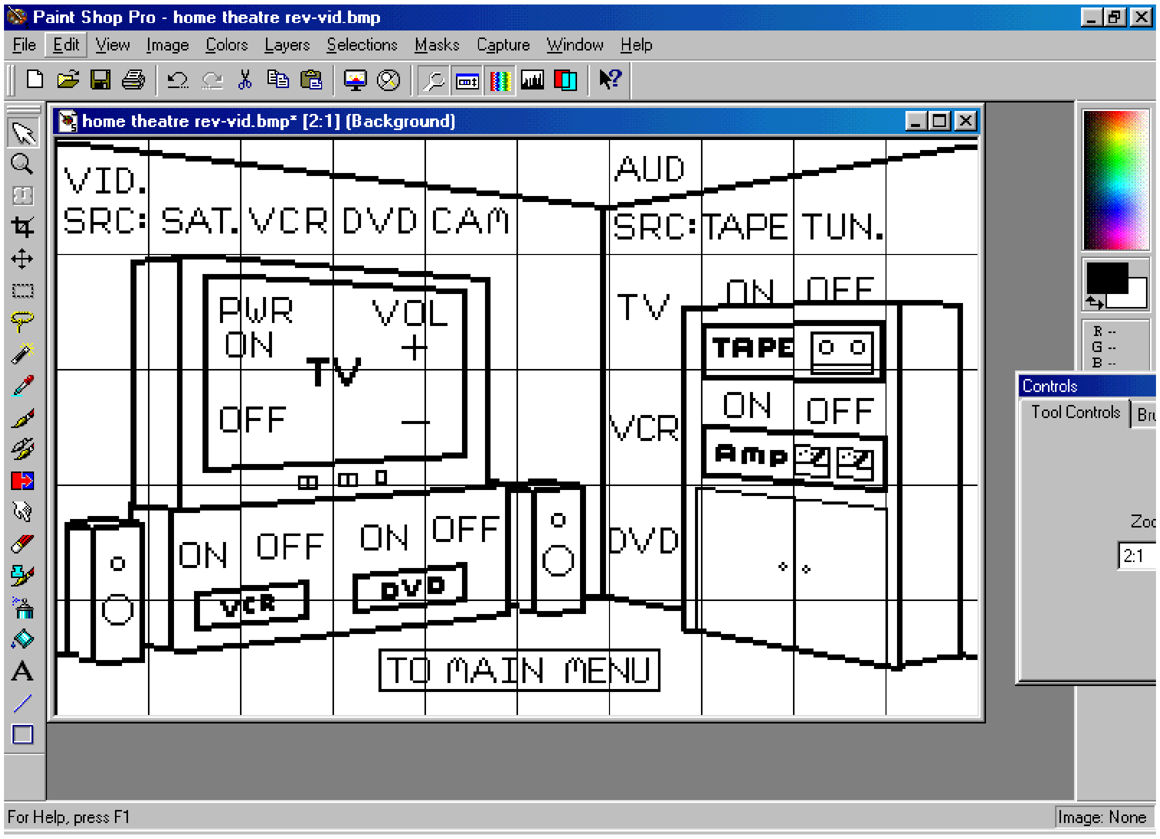

Paint Shop Pro - home theatre rev-vid.bmp
File Edit View Image Colors Layers Selections Masks Capture Window Help
home theatre rev-vid.bmp* [2:1] (Background)
VID. SRC: SAT. VCR DVD CAM
AUD SRC: TAPE TUN.
PWR ON
VOL +
TV
OFF
−
TV ON OFF
TAPE
VCR ON OFF
Amp
DVD
ON OFF VCR
ON OFF DVD
TO MAIN MENU
Controls
Tool Controls
Zoo
2:1
For Help, press F1
Image: None

Fig. 12

Next, we make some icons to put on buttons along the bottom of the screen to perform standard functions associated with playback devices, like "play", "rewind", etc. We'll use the industry standard graphic symbols for these. Here is the "fast forward" button designed with a thin outline and with one less pixel along the bottom and right edges so that these buttons will appear separate when displayed side by side. Also note that the colors have been inverted because we want a light colored button with a dark symbol to display on our Leopard I (Fig 13). The image in Fig 12 was also color-inverted before downloading to the Leopard to get dark lines on a light background:

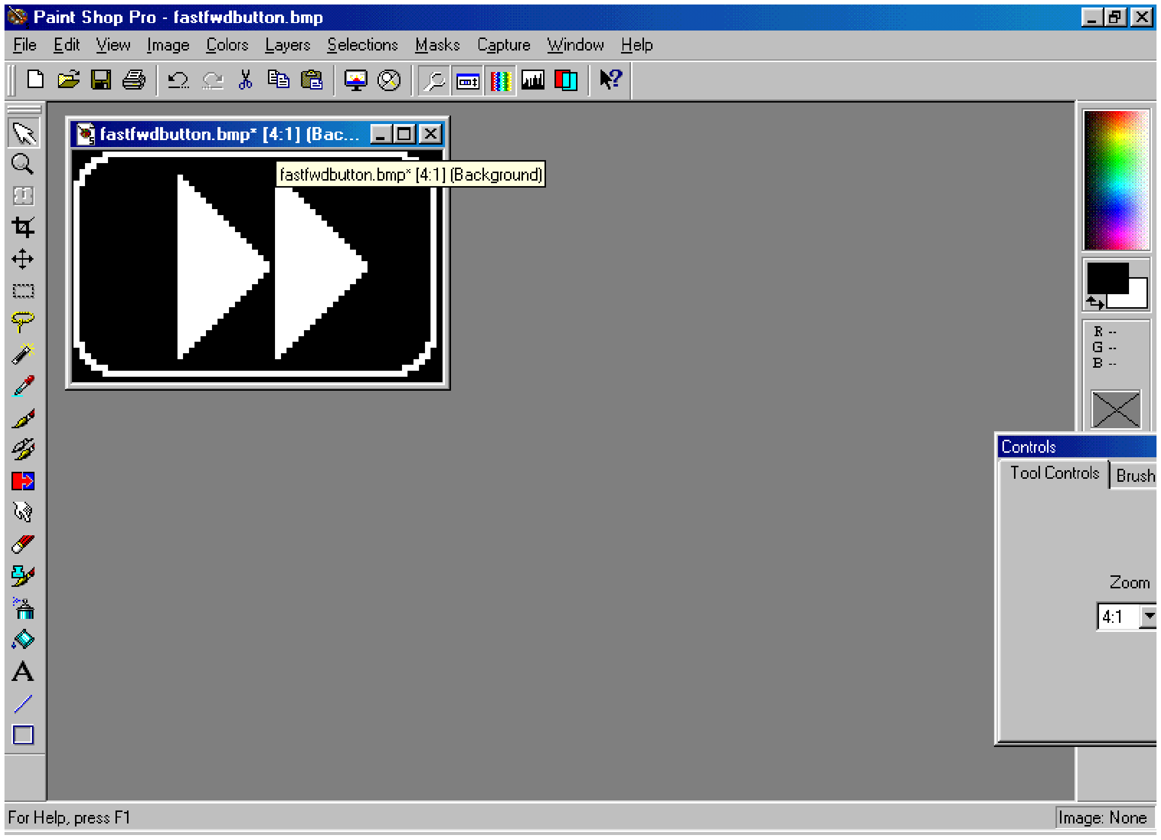

Paint Shop Pro - fastfwdbutton.bmp
File Edit View Image Colors Layers Selections Masks Capture Window Help
fastfwdbutton.bmp* [4:1] (Bac...
fastfwdbutton.bmp* [4:1] (Background)
R --
G --
B --
Controls
Tool Controls
Brush
Zoom
4:1
For Help, press F1
Image: None

Fig. 13

Next, we create the touchscreen layout over which the large image and the control buttons will be laid out (Fig. 14). All the buttons on this screen have the "hidden border" box checked and use no captions. All specify icon #0, which means display no icon, except for the 2 x 1 button to which the large icon image will be attached. This button is the one under the upper left corner of the "Enter Text" box in the picture.

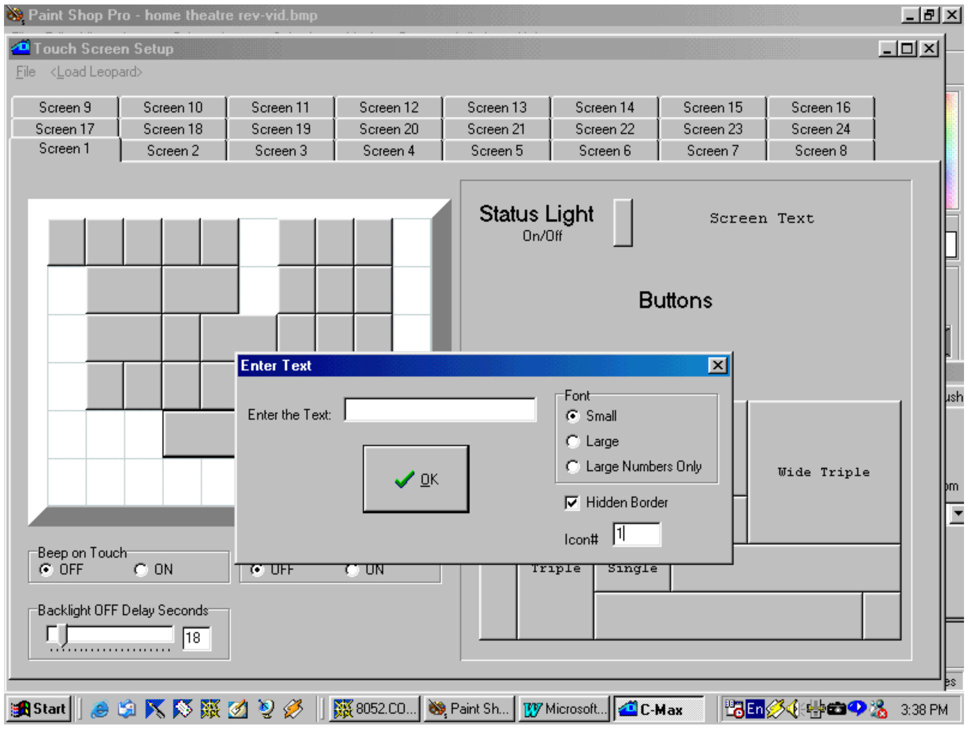
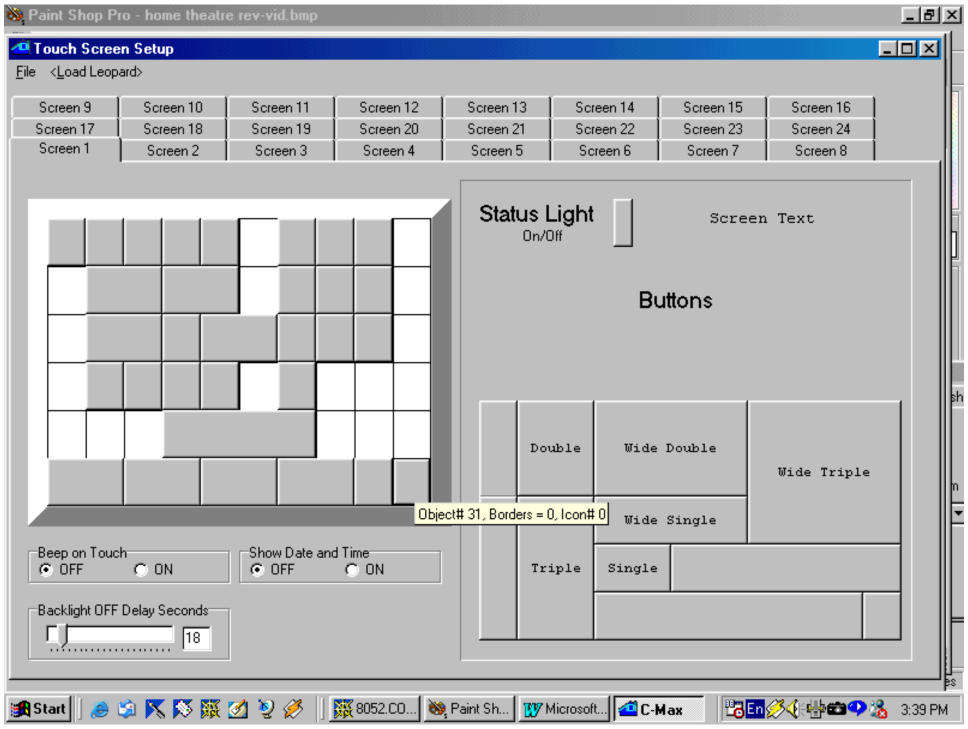


Fig. 14

This leaves the bottom row of touch squares on which we now add the playback control buttons. Each of these will also specify hidden borders and each one will be associated with the icon created for it (Fig. 15). It is also possible to actually display buttons or icons that overlap a large “spill over” icon (but not the button itself) on the screen. We could have done that here, but it is a waste of icon memory to do this if the image under the button(s) will not be visible anyway. If you do want to do this, make sure that the icon that is to appear “above” the other one has a higher object number within that screen in the touchscreen editor. If the button is totally hidden (as in this example) and only serves as a hot spot for the large icon, then the object number is not important and can be either higher or lower, but if your intent is to have a visible button with an icon or caption appear over the large icon then you must observe this requirement.

You can see the object numbers my moving the mouse cursor over the button. A small information window will appear showing you the object number, hidden border flag, and icon number for that object. This is shown over the bottom right button in Fig. 15. The best way to be sure to have the lowest object number for the large background image is to define that button first when creating that screen. Even then, you must be careful because if you then go to another screen containing lower-numbered objects, delete one or more items, and then go back to your “large icon” screen and continue adding new objects; these will now have lower object numbers then even the first button you put on the screen because any freed object numbers are reused before new higher ones are assigned. If you have any doubts as to the object numbers of your various objects, just pass the mouse pointer over every one and note the numbers assigned to each one.


Paint Shop Pro - home theatre rev-vid.bmp
Touch Screen Setup
File <Load Leopard>
Screen 9 Screen 10 Screen 11 Screen 12 Screen 13 Screen 14 Screen 15 Screen 16
Screen 17 Screen 18 Screen 19 Screen 20 Screen 21 Screen 22 Screen 23 Screen 24
Screen 1 Screen 2 Screen 3 Screen 4 Screen 5 Screen 6 Screen 7 Screen 8
Status Light On/Off
Screen Text
Buttons
Double
Wide Double
Wide Triple
Wide Single
Triple
Single
Object# 31, Borders = 0, Icon# 0
Start
3:39 PM

Beep on Touch [✓ OFF] ON

Show Date and Time [✓ OFF] ON

Backlight OFF Delay Seconds 18

Fig. 15

Fig. 16 shows an actual picture of the final result on a Leopard I screen:

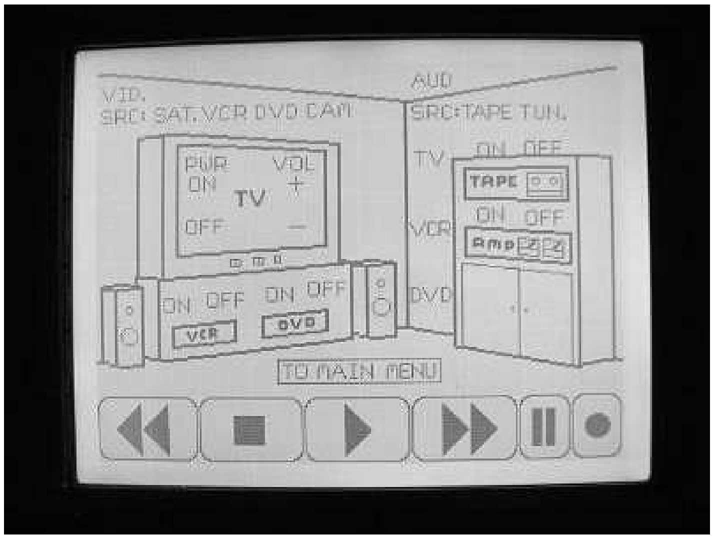


Fig. 16

Changing icons under program control

In this third and final example, we will show how you can have a running program actually change the displayed icon on buttons. You can use this to make dynamic screens where the illustration or hand created "captions" adapt to the choices the user makes as they happen. You can also use this feature to create graphical representations of data that you might prefer to see pictorally instead of just with numbers. The example we present here does just that: Imagine that you have an oil or water tank with a sensor that sends an analog signal between 0 and 5 volts to indicate the content's level from empty to full respectively. You could always display a number indicating the level as a percentage of full, but you are so used to seeing liquid levels displayed on mechanical guages that you would prefer a guage-like display. Here, we will read the 0 to 5 volt output of the tank sensor using a SECU16 module input configured for analog mode and display an automobile gas-guage like picture to indicate the level in the tank. We will do this by making six versions of the gas guage icon, each one with the pointer in a different position. One of these icons will always be displayed, and will be selected depending on the analog value read from the SECU16 input.

First we choose our icon size. The guage being vertical, we will make it 1 square wide by 3 squares high (32 x 120). We begin by designing the guage scale itself (Fig. 17) and then the pointer (also Fig. 17) as a separate image. This way, the pointer can be pasted to the scale in various positions to simplify the procedure.

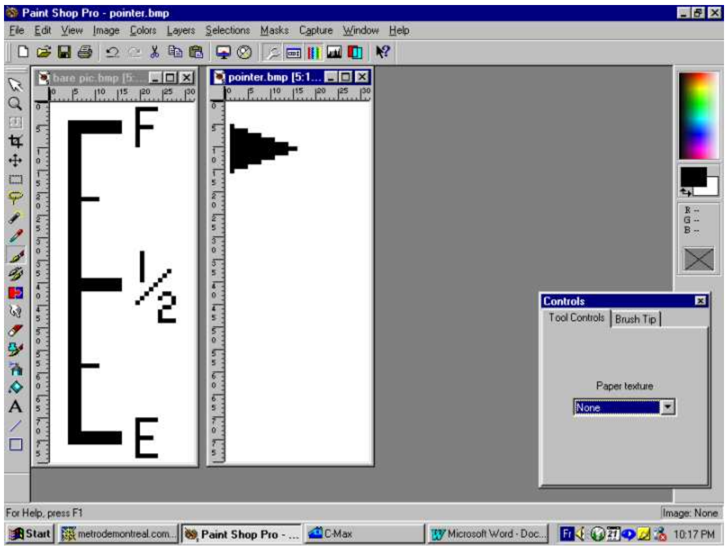


Fig. 17

With these two pictures we create the six pointer versions needed for our application (Fig, 18)

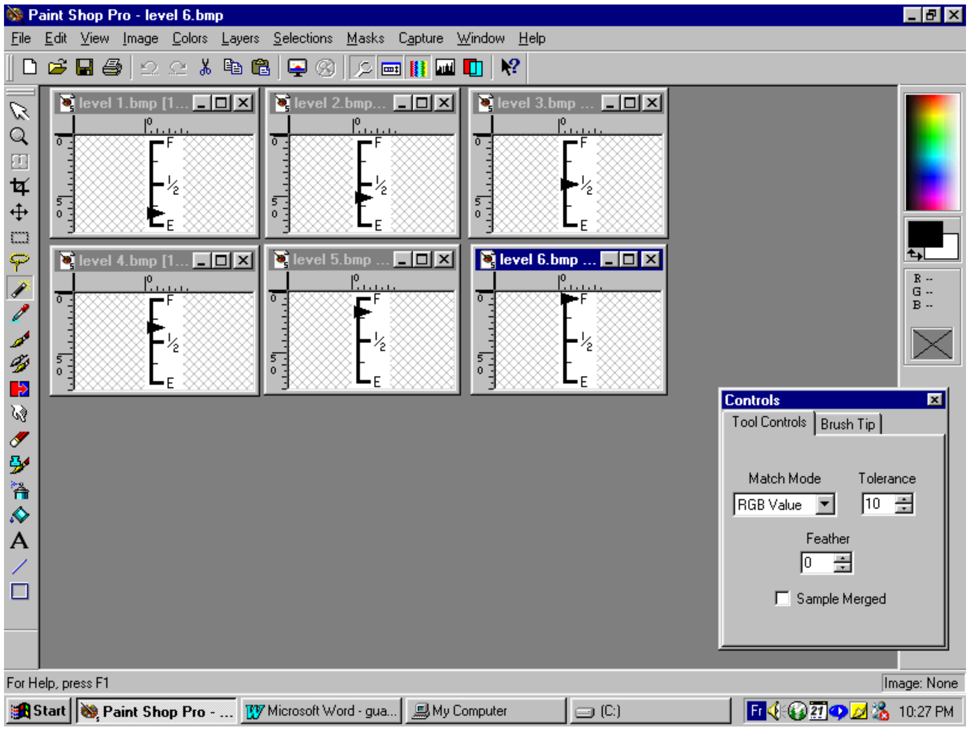


Fig 18

The six icons are then assigned in the System Map to six consecutive icon numbers (Fig. 19). Notice here that the first icon used happens to be icon# 5 and goes up to icon# 10 (any range can be used as long as the icons are in consecutive order and the illustrated levels rise with the higher icon numbers).

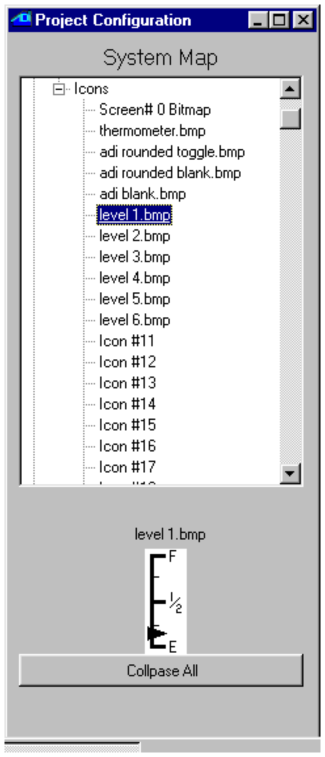


Fig. 19

The icons are loaded into the Leopard with the same procedure as in the other examples. A touchscreen is then defined to display a 1 wide x 3 high button with no caption, hidden border is checked, and assigned icon #5 by default. In our example, this button is object #1. Finally, a screen text object is also added to identify the purpose of our guage (Fig. 20). This screen definition is then also loaded into the Leopard.

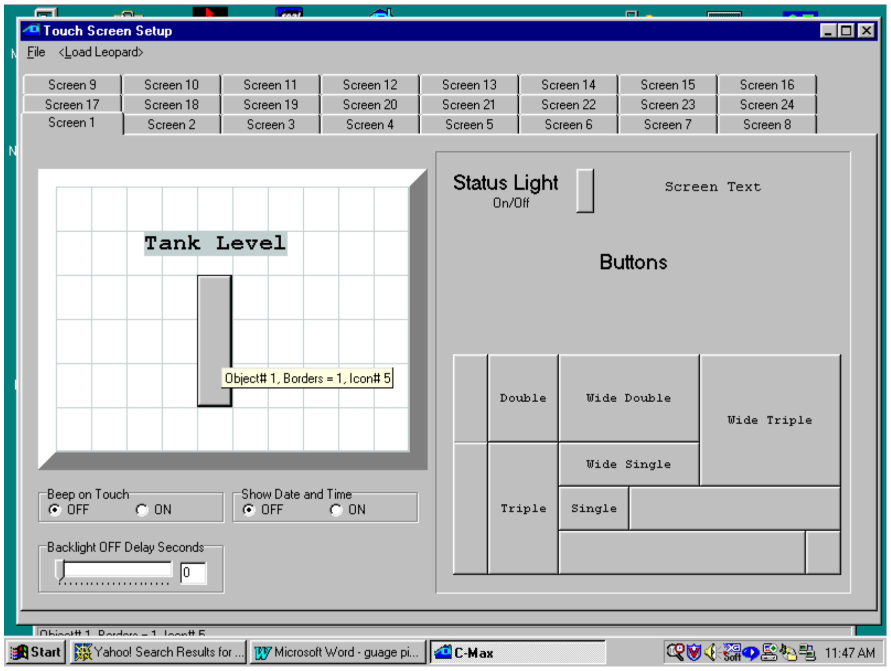


Fig. 20

Here is the program code used to read the SECU16 analog input and display the appropriate icon (Fig. 21)

```
0001 - IF Module #1   -SECU16 Tank-level-sensor  is > 0           // read tank level voltage
0002 -     THEN  Load Data to:  Variable #0                       // store in var#  0
0003 -     ELSE  Load Data to:  Variable #0                       // whether  > 0 or not
0004 - IF Variable #0 is > 0                                      // if > 0
0005 -     THEN Variable #0 / 43                                  // then scale from 0 to 5
0006 -     THEN Variable #0 + 5                                   // and add 5 for offset
0007 -     THEN Touch Object #1 displays  Icon# in Variable #0    // update displayed icon
0008 -     ELSE Touch Object #1 displays  Icon# 5                 // if level 0 then icon #5
0009 - End Program                                                //
```

Fig. 21

Lines #1 to #3 capture the analog value, which will range from 1 to 255 for signals of 0 to 5 volts respectively, into variable #0 .The variables were not mapped to names here for clarity. Line #5 takes the value from 0 to 255 and divides it by 43 to obtain a value from 0 to 5, since our guage can display six different levels. Thus input values from 215 to 255 will show the “full” icon, input values from 172 to 214 will show the next level down and so on. Line #6 adds 5 to our scaled value because our icon numbers for this example run from 5 to 10. Line #7 then displays the calculated icon number on the touch button. Line #8 exists because if line# 4 tests false, then the analog input is indeed zero and we want to show the lowest level icon in that case. Fig. 22 shows an actual picture of a Leopard running this program.

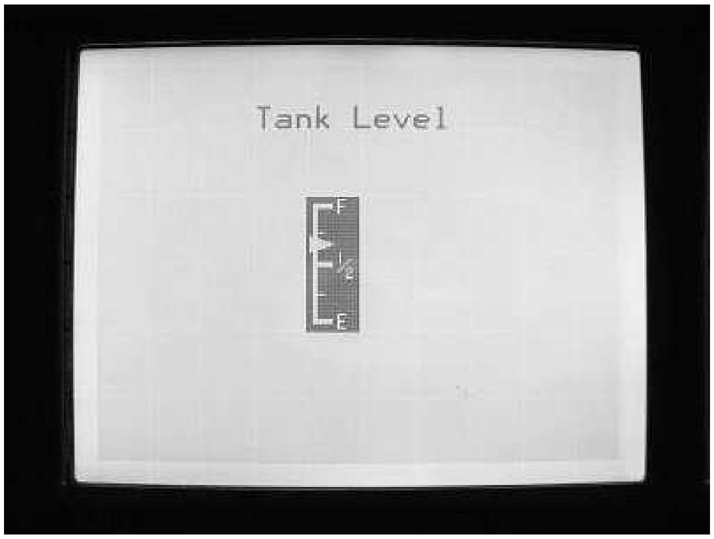


Fig. 22

As these three examples have shown, you can make highly customized screens using the new icon display capabilities and switching instruction in C-Max 2.0