Enhanced Apple //e Emulator
Leap into a new dimension with DHGR Images In Aztec C!
a demo by Bill Buckels
Waterpro
Caps

DMONDEMO – A DHGR Monochrome Image Demo Sampler in Aztec C

By Bill Buckels, February 16, 2013

Table of Contents

Introduction

In November 2012 I decided to create a relatively simple utility to convert back and forth between Apple II monochrome DHGR files and Windows monochrome BMP files. Hence DMONO.exe was conjured-up and completed.

Shortly thereafter, I decided that a complete overhaul of my AppleX Aztec C MS-DOS Cross-compiler Distribution for ProDOS 8 was long overdue. As I near completion and prepare to unleash a plethora of advice and a paucity of assistance upon my unsuspecting public, in matters related to the production of Aztec C programs for the Apple II, I have prepared a number of new programs (to demonstrate the many additions to the AppleX link-libraries), many new and improved utilities, and disk images of my new programs.

Today I finished one of these many new collections of disk image demos, "DOMONDEMO", and have decided to release this collection in advance of my new version of AppleX, as a preview of things yet to come. This then is the documentation for DMONDEMO; it is a "lite version" and not too terribly technical… really only meant to get the reader interested enough to download DMONDEMO and "muck with it".

DMONDEMO is available as a zip file from the following link:

http://www.aztecmuseum.ca/dmondemo.zip

I hope you enjoy the demo and I look forward to sharing the rest of the demos and other goodies that come with the new version of AppleX which should be online and available for download shortly.

Disk Images Included in DMONDEMO

4 disk images are included. 1 of these is a somewhat equivalent disk image of the WATSPEED.DSK Aztec C Demo and is provided for comparison purposes:

WATBASUC.DSK	– ProDOS 8 AppleSoft BASIC DHGR Monochrome Slideshow.

The other 3 disk images are Aztec C demos and are somewhat different from each other, but they too are DHGR Monochrome Slideshow programs. Each disk "showcases" 3 different DHGR image formats that will be supported by AppleX in ProDOS 8 when it is finally released (the Aztec C slideshow programs on all of the demo disks listed below support all the formats I have listed below and A2FC DHGR BSaved Images as well):

WATERPRO.DSK	– demo for DHR raw raster DHGR images.
WATSHELL.DSK	– demo for DHX run length encoded DHGR images.
WATSPEED.DSK	– demo for BIN/AUX BSaved DHGR images.

I am not going to get into too much detail at this point, (appendices are at the end of this document) but some features exist in the 2 Aztec C programs provided with these demos and some differences between them too. One of the differences is WATSHELL.DSK runs under the Aztec C Shell for ProDOS 8, and the other two disks (WATERPRO.DSK and WATSPEED.DSK) run in “raw” ProDOS 8.

WATSHELL.DSK – DHGR Graphics under the Aztec C Shell

The Aztec C Shell provides command line support in a “Unix-like” manner including the entry of “wildcards” and ProDOS 8 prefixed pathname support. The Aztec C Shell Program (DPLODE) on WATSHELL.DSK is both a single-image DHGR viewer and a slideshow program.

When more than one image name is entered on the Shell command line following the program name, DPLODE launches into slideshow mode, and DPLODE also works as an interactive viewer when DPLODE is launched by itself without command line “arguments”.

I’ll leave it to you to “rummage” around on WATSHELL.DSK to find-out a little more. FWIW DPLODE also displays usage when it is run in interactive mode.

Previous versions of AppleX did not provide support for making Aztec C Shell Programs but that is over. The Aztec C Shell should not be overlooked. It not only “sports” command line arguments, environment variables and shell scripting (in its own manner), but will also load BIN programs right from the command line in a manner similar to AppleSoft. I suspect I have already made my bias away from obstructured environments pretty clear (on many occasions), so suffice to say I like the Shell and more information on the Aztec C Shell (and other A2 shells) can be found at the following link:

http://www.aztecmuseum.ca/shell.htm

WATERPRO.DSK and WATSPEED.DSK contain almost the same demo except that WATERPRO.DSK “showcases” my DHR raw-raster format which loads from the top of the screen down in a civilized manner.

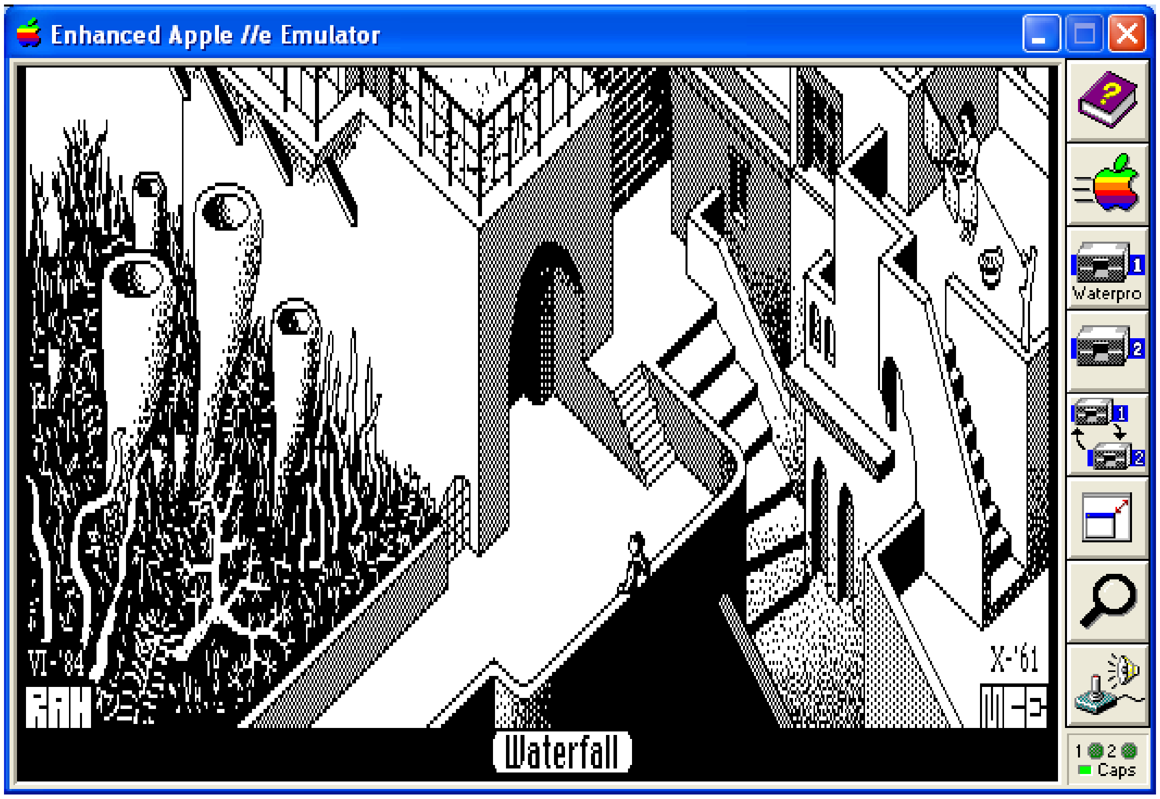
I am not going to get into DHGR image formats much here because you’ll see for yourself when you run these demos… and the utilities and C language library routines to work with these, and more (for DHGR and the other Apple II graphics modes), will be included in AppleX when I am done…

The slideshow program on these two disks is called DPSHOW. Unlike the shell program DPLODE, it is a single-purpose slideshow and not a viewer, and unlike DPLODE, DPSHOW accesses the ProDOS 8 directory system on its own thanks to routines for ProDOS directory services in the new AppleX libraries.

You can take the DPSHOW program by itself and “stick-it” in a ProDOS directory or on a disk and it will build, then run a slideshow, if it finds one or a bunch of DHGR images… it does this by building a text file of image names (a slideshow script) called a

"PACLIST". But it will not overwrite an existing PACLIST so you can edit these to your "heart's desire", adding and removing pathnames to images and so forth.

Slideshow Content - Waterfall (Waterval) – an Impossible Object



The reason that I decided to use monochrome DHGR images for these demos was to display (in pieces) one of my particular favorite digital images that has been around (in Mac Paint format) for quite a number of years… on March 25, 1985 (some 27 years ago) the following message appeared on Usenet (complete with Mac Paint file in Bin Hex) :

"I pulled this off the Club Mac bulletin board, and was impressed enough with it I decided to post it to the net. This doesn't appear to be a digitized image, so I can only assume it was drawn by hand, in which case it is very impressive indeed. Does anyone know any more about this?"

Tom Buckley

4 days later, on March 29, 1985, the Usenet group received the following response:

“The posted picture is a picture of the late Dutch artist M.C. Escher. ("MCE" in the lower right corner of the picture.) Escher also created "Metamorphosis" and many other pictures like the one posted here.”

Rick Jansen
Amsterdam, The Netherlands.

How I Converted Waterfall to DHGR

My utility DMONO was used to do the conversion from Windows Monochrome BMP files to DHGR Monochrome BSaved images without loss of detail. My new utility XPACK was used to create the rasterized and encoded versions from the BSaved images. I was already in possession of Waterfall in gif format so it was no problem for me to load the gif into Windows XP Paint and to double-stretch it, then chunk-it into 4 equal and smaller pieces (560 x 384 in size) and convert each of the 4 as noted above.

It was as simple as that… done over coffee this morning. I have included a full-size Waterfall in BMP format in DMONDEMO along with the disk images and this document.

Run-Length Encoded Images

This demo provides images in the DHX and DHR formats created by a not-yet released program called XPACK. When AppleX is released new utilities like XPACK will be available including utilities for creating plain-old HGR images and image fragments in the equivalent formats (RAG and RAX) from Windows BMP files. Space savings vary.

I also developed a very efficient compressed DHGR format created by a new utility called XCRUNCH but I am not including XCRUNCHED images today. While it’s true that XCRUNCH provides better compression than was available in a BEAGLE GRAPHICS crunched DHGR file the load speed is pretty slow…

For most purposes when it comes to Apple II image loading in a C program, the raw-raster image formats like DHR and RAG work best if one has the disk space. One reason that I provide output in BSaved Format from my utilities is to make them useful within AppleSoft’s bog and mire, and the other is that this is all fun to write and to play with.

Appendix A – Aztec C Shell Program DPLODE Source Code

```
/* ------------------------------------------------------------------------
System       : Manx Aztec C65 Version 3.2b
               MS-DOS cross-development environment
Platform     : Apple IIe 128K PRODOS 8
Program      : dplode(C) Copyright Bill Buckels 2013.
               All rights reserved.
Description  : Apple //e ProDOS 8 DHGR Command Line Picture Viewer
               for the Aztec C ProDOS Shell.

               Extended Version of dlode with Support for
               XPacked Images.

               dplode commandline usage: dplode -seconds DHGRimagename.2FC
                                         dplode -seconds DHGRimagename.BIN
                                         dplode -seconds DHGRimagename.DHX
                                         dplode -seconds DHGRimagename.DHR

               dplode interactive usage: dplode

               DHGR formats supported:  A2FC, BIN AUX pairs, and
                                        DHX and DHR xpacked images.

Written by   : Bill Buckels
Date Written : January 2013
Revision     : 1.0 First Release
Licence      : You may use this program for whatever you wish as long
               as you agree that Bill Buckels has no warranty or
               liability obligations whatsoever from said use.
------------------------------------------------------------------------ */

#include <g3.h>
#include <console.h>

char *usage =
"dplode(C) Copyright Bill Buckels 2013.All rights reserved.\n"
"Extended Version of dlode with Support for XPacked Images.\n"
"Command Line: dplode DHGRimagename or dplode -seconds DHGRimagename\n"
"  Multiple images can be entered for slideshow\n"
"  Slideshow Timeout is in seconds (default = 5)\n"
"DHGR formats supported: A2FC, BIN AUX pairs, DHX and DHR xpacked images.\n";


main (argc, argv)
int argc;
char *argv[];
{

    char picname[64], *ptr;
    int c= -3, cnt, color, idx, numpics = 1, timeout = 0, skip = 0, key;

    scr_clear();
    scr_curs (0, 0);

    if (argc == 1) {
        mode80();
        printf(usage);
        printf("Enter DHGR Pic Name: ");
        gets(picname); /* get the filename */
        /* blank line to end program */
        scr_clear();
        scr_curs (0, 0);
        if (picname[0] == 0)exit(c);
    }
    else {
        if (argc > 2) {
            /* slideshow timeout 1-100 if more than one image loaded from the
               commandline - default is 5 seconds (approx.)
               on a "stock" apple II
               and if the user has an accelerator or wishes something different
               this can be adjusted as necessary. */
            timeout = 5;

            for (cnt = 1; cnt < argc; cnt++) {
                /* switch character is optional */
                if (argv[cnt][0] == '-')ptr = (char *)&argv[cnt][1];
                else ptr = (char *)&argv[cnt][0];
                idx = atoi(ptr);
                if (idx > 0) {
                    skip = cnt;
                    if (idx > 100) idx = 100;
                    timeout = idx;
                    break;
                }
            }
        }
    }

    /* set double hires */
    dhireson();
    /* clear screen */
    dxdblock(0,191,0);

    cnt = 1;
    while (kbhit())clearkey();

    for (;;) {

        /* preload keycheck */
        key = kbhit();
        if (key!=0) {
            clearkey();
            /* if escape key is pressed end slideshow */
            if (key == ESCAPE) break;
            key = 0;
        }

        if (argc > 1) strcpy(picname,argv[cnt]);

        if (skip != cnt) {
            /* load picture */
            c = dplode(picname);
            if (c!=0) c = dlode(picname);

            /* if single picture loaded
                1. if a problem just exit,
                2. otherwise wait for key press and exit */
            if (argc < 3) {
                if (c!= 0) break;
                getch();
                break;
            }
            /* advance on timeout or advance, reverse or end on keypress */
            if (c==0) key = wait(timeout,XTIME,YTIME);

            /* special case - single image from command line with timeout */
            if ((argc-skip) < 3) break;

        }
        else {
            key = kbhit();
            if (key!=0)clearkey();

        }

        /* postload key check */
        if (key == ESCAPE) break; /* if escape key is pressed end slideshow */

        switch(key) {
            case UPARROW : /* reverse direction */
            case LTARROW : cnt--;
                           if (cnt < 1) cnt = (argc - 1);

                           /* the skip test - check twice
                              but even I can't think of everything */
                           if (cnt == skip) cnt--;
                           if (cnt < 1) cnt = (argc - 1);

                           if (cnt == skip) cnt--;
                           if (cnt < 1) cnt = (argc - 1);
                           break;
            default: cnt++;
        }

        if (cnt < argc) continue;
        cnt = 1;
    }

    while (kbhit())clearkey();

    dxdblock(0,191,0);

    dloresoff();
    scr_clear();
    scr_curs (0, 0);

    /* if the last picture did not load let them know
       before exiting */
    if (c!=0) {
      bellerr();
      if (c == -1) printf("! Can't open %s!\n",picname);
      else printf("! %s wrong format! Return code %d.\n",picname, c);
      printf("Press Any Key...");
      getch();
      scr_clear();
      scr_curs (0, 0);

     }

    exit(c);

}
```

Appendix B – Aztec C ProDOS Sys Program DPSHOW Source Code

```
/* ------------------------------------------------------------------------
System       : Manx Aztec C65 Version 3.2b
               MS-DOS cross-development environment
Platform     : Apple IIe 128K PRODOS 8
Program      : dpshow(C) Copyright Bill Buckels 2013.
               All rights reserved.

Description  : Apple //e ProDOS 8 DHGR Slideshow Program

               DHGR formats supported:  A2FC, BIN AUX pairs, and
                                        DHX and DHR xpacked images.

               Uses Page 2 Double Hi-Res to avoid
               ProDOS Sys Program Load Address at 0x2000

Written by   : Bill Buckels
Date Written : February 2013

Revision     : 1.0 First Release
Licence      : You may use this program for whatever you wish as long
               as you agree that Bill Buckels has no warranty or
               liability obligations whatsoever from said use.
------------------------------------------------------------------------ */

#include <g3.h>
#include <stdio.h>
#include <prodir.h>

#include "paclist.inc"

char *title = "dpshow";
char *copyright = "(C) Copyright Bill Buckels 2013.All rights reserved.";

int mystuff()
{
    FILE *fp;
    int cnt = 0, idx, valid = 1;
    char *buf = (char *)4192;

    if((fp=fopen("copyread","r"))==NULL) return cnt;


    /* allow 20 lines for copyright and usage */
    for (cnt = 0; cnt < 20;) {
        buf[0] = 0;
        if (fgets(buf,80,fp) == -1) break;

        /* signature line must match */
        if (cnt == 0) {
            for (idx = 0;copyright[idx] != 0; idx++) {
                if (copyright[idx] != buf[idx]) {
                   valid = 0;
                   break;
                }
            }
            if (valid == 0) break; /* outa' here */
            scr_curs(cnt,0);puts(title);cnt++;
            scr_curs(cnt,0);puts(copyright);cnt++;
        }
        else {
            for (idx = 0;buf[idx] != 0;) {
                if (buf[idx] == 13||buf[idx] == 10||buf[idx]==0) buf[idx] = 0;
                else {
                    if (buf[idx] < 32 || buf[idx] > 122) {
                        valid = 0;
                        break;
                    }
                    idx++;
                }
            }
            if (valid == 0) break; /* outa' here */
            scr_curs(cnt,0);puts(buf);cnt++;
        }
    }
    fclose(fp);
    scr_curs(cnt,0);
    return cnt;

}

/* the following is what I eventually ended-up with as a reasonable
   compromise for the structure of a simple slideshow program for
   ProDOS using a picture file list. It is a little more elegant than
   some of the other simple slideshow programs that I wrote in this
   same environment over 20 years back. I have done my best not to stop
   the show even if some of the pictures are missing or in an unsupported
   format. */

main()
{
    FILE *fp;
    unsigned filesize;
    int idx,status,started=0,key=0,c,cnt;
    char picname[82];

    scr_clear();
    scr_curs(0,0);

    /* set page 2 double hires - we need to use page 2 of course because
       ProDOS SYS programs load at the same address as the page 1 screen */
    d2hireson();

    /* clear screen by copying memory hole in sys program
    to page2 hi-res screen in auxiliary memory */
    maintoaux(0x4000,0x4000+8191,0x4000);

    /* loop through the slideshow and restart at the beginning until the
       user presses the escape key... */

/* on a clear display you can C for(;;) (forever)

At my age I prefer unconditional loops with clean breaks and a few flags
since the language supports it and I find it somewhat more readable realizing
of course that a break amounts to no more than a goto anyway.

And you guys who prefer while(1==0) can have it! That's a condition that
I'd prefer not to imagine. The highway, like the Apple II and this slideshow,
goes on forever. presumably. */

    for (;;) {

        status = -1;
        if((fp=fopen("PACLIST","r"))==NULL) {
            /* first time through try to create piclist if it does not exist */
            if (started == 0) {
               fp = makepaclist();
            }

            if (fp == NULL) break;
        }
        started = 1;
        picname[0] = 0; /* gigo - blank the line before calling fgets */
        /* get the first filename */
        if (fgets(picname,80,fp) == -1) {
            fclose(fp);
            break;
        }
        /* if garbage on the first line outa' here */
        if (picname[0] < 33 || picname[0] > 122) {
            fclose(fp);
            break;
        }

        /* clear keyboard */
        while (kbhit()!=0)clearkey();
        status = cnt = 0;

/* before I allow an image to load, I have established a rather lengthy
   interview process... basically for the hellery of it and to satisfy the
   arguably sadistic wish to deny entry unless ones' details are in in order
   that dwells in the heart of every good gatekeeper and passes for fun in
   the soul of every good programmer. */

/* if the first character on the line has an ascii value of a space character
or below
   this program will fail... I do not support blank lines and text files must
   be in Apple Format or Unix format... an MS-DOS paired 13,10 EOL will cause
   fgets to read a blank line after each valid line so only the first line will
   be read and if it is an image entry the first image will load repetively and
   frankly Scarlet, I don't give a damn. My main concern here is reading past
   eof so my main concern is keeping the garbage out of here or there is a
potential
   for a less-than spectacular crash possibly with the read loop going into a
   tailspin first... that is, if we don't have a garbage filter and some
   belt and suspender rules. a more cavalier approach could be applied here
   with the potential for disasterous results bur I also could have applied
   exhaustive format validation... since I am not Jack Crenshaw working on the
   Apollo Mission I am confident no lives will be lost in space with what I put
here. */

        /* line validation */
        while(picname[0] > 32 && picname[0] < 123) {

            /* preload integrity check */
            picname[64] = 0; /* truncate the line to the maximum length
                                of a ProDOS Pathname */
            idx=0;  /* replace carriage return with null, ignore lines beginning
                       with invalid characters etc. */

            while(picname[idx]!=0)
            {
        /* if I find an ascii value below the prefix character value (46)
           or any above a lower-case z I truncate the line there...
           comments can be on the line separated from the picname
           by an ascii value below 46 (a space or a pound sign are ok)
           but the line must not exceed 80 characters... I had to draw the
           line somewhere... */
                if(picname[idx] < 46 || picname[idx] > 122)picname[idx]= 0;
                else idx++;
            }

    /* comments can also be on a separate line. the rule here is almost
       the same as noted above for trailing comments on the picture entry line
       except that the first character on the line must be an ascii value
       from 33 to 45 (spaces are not allowed, pound signs work just fine) */

            if (picname[0] != 0) {

                /* preload keycheck - skip loading the image if
                   the user leans on the keyboard and try to avoid
                   partially loading raster based images because they
                   can look like heck */
                key = kbhit();
                if (key!=0) {
                    clearkey();
                    cnt++;
                }
                else {
                    c = d2pld(picname);
                    if (c< 0) c = d2lode(picname);
                    if (c > -1) cnt++;

                    /* if a load error skip this slide, otherwise... */
                    /* advance on timeout or advance or end on keypress */
                    if (c==0) key = wait(10,XTIME,YTIME);

                } /* nobody leaned on the keyboard
                     to advance to the next slide */
            } /* the format for the picture name entry in
                 the piclist is valid */
            else {
                /* picname[0] == 0 - truncated line - began with garbage */
                key = kbhit();
                if (key!=0)clearkey();

            }
            /* if escape key is pressed end slideshow */
            if (key == 27) break;
            picname[0] = 0; /* gigo - make sure no garbage is going in to fgets
                                trust nothing and no-one */
            if (fgets(picname,80,fp) == -1)break;
        }
        fclose(fp);

/* if all the pictures are missing or invalid or if
   the entire piclist is not correctly formatted
   the count will be zero...
   I report this as an error before the program exits.
   I don't bother reporting the occasional missing picture
   or one that didn't load vs. others that did...

   I'll leave that for the user to figure-out.
   I figured-out that enough is enough already so let them
   give their brains some exercise if they even notice
   some stuff not happening... empathy is no match for stupidity
   anyway and a good programmer like a good shepherd knows
   that it is pointless to guard a dead sheep, even a brain-dead one.
   */

        if (cnt == 0)status = -1;
        if (key == 27 || cnt == 0)break;

        /* bottom of loop - start over at the beginning of
           the slideshow piclist...
           on a clear display you can C for(;;) (forever)
           or is it on a clear disk you can seek forever? */

    }

    /* either the user has pressed the escape key to break our magic spell
       or the piclist is missing, or all the pictures are gone...
       but now it's time to say goodbye to all our company... */
    d2flood(0,0,139,191,0);
    d2hiresoff();
    scr_clear();
    scr_curs(0,0);

    /* clear keyboard */
    while (kbhit()!=0)clearkey();

    if (status!=0) {
        puts("!PACLIST missing, invalid, or could not be opened or created!");
        scr_curs(1,0);
        puts("Press Any Key...");
        getch();
        scr_clear();
        scr_curs(0,0);
    }

    /* display the copyright if one exists
       but if they just copy the program or remove it
       let them go ahead... there is only fame to be had from this old stuff
       anyway and certainly not fortune. enough crippleware exists in the world
       without me creating more. I am just happy to finish a program after
       all these years...
       (jk of course, but I am delighted that it's not crashing... so far:)

       hope you enjoyed the read even if the code is only k&r and I trust that
       something proved interesting or informative.

       for much much more, read the G2 library code and review some of
       my other fun little Aztec C (and other) projects.

       Bill Buckels
       February 4, 2013

       */
    if (mystuff() != 0) {
        puts("\nPress Any Key...");
        getch();
        scr_clear();
        scr_curs(0,0);
    }
    _exit();

/* Aztec C exits with a warm boot back to ProDOS
if everything has gone well... if one crashes into the
monitor there is still work to do. */
}
```

Appendix C – DPSHOW PACLIST.INC Source Code

```
/* structures for filefind */

/* point to unused memory below SYS programs
   that I also use for a transfer area between AUXMEM and MAINMEM */

struct filefind *ff = (struct filefind *)4192;
struct fileinfo *fi = (struct fileinfo *)4192;

/* make a piclist if it does not already exist... look for files in
   the current directory that match the criteria for our loader. */

FILE *makepaclist()
{
    FILE *fp = NULL;
    char *ext = NULL, *f=NULL, picname[20], filext[16];
    int filetype = 6, err, ferr=0, cnt=0, filematch, i, j;

    if((fp=fopen("PACLIST","w"))==NULL)return fp;

    /* look for Hi-Res Graphics files and add to piclist */
    err = gfindopen(f,fi);

    if (err > -1) {
        while (gfindnext(filetype, ext, picname,ff) == 0){

            if (ff->address != 0x2000 && ff->address != 0x4000) continue;
            if (ff->size < 1024 || ff->size > 16384) continue;

            strupr(picname); /* change all filenames to upper case */
            filematch = 0;

            /* get picname extension */
            j = -1;
            filext[0] = 0;
            for (i = 0; picname[i] != 0; i++) {
               if (j > -1) {
                   filext[j] = toupper(picname[i]);
                   j++;
                   filext[j] = 0;
               }
               if (picname[i] == '.') j = 0;
            }

            /* raw format */
            if (ff->size == 8192 || ff->size == 16384) {
                filematch = 1;
                /* bypass AUX files - I only want BIN files and A2FC files
                   in my list */
                if (ff->size == 8192 && filext[0] == 'A' && filext[1] == 'U' &&
                    filext[2] == 'X' && filext[3] == 0) filematch = 0;

            }
            else {
               /* XPACK raster format and RLE */
               if (filext[0] == 'D' && filext[1] == 'H' && filext[3] == 0) {
                    if (filext[2] == 'X') filematch = 1;
                    if (ff->size == 15365 && filext[2] == 'R') filematch = 1;
               }
           }
```

```
            if (filematch != 0) {
                /* if out of disk space quit writing to piclist */
                ferr = fputs(picname,fp);
                if (ferr != -1) ferr = aputc('\n',fp);
                if (ferr != -1) cnt++;
                else break;
            }
        }
    }

    fclose(fp);
    if (cnt > 0) fp=fopen("PACLIST","r");
    else {
        /* if no files found or no picnames written
           remove empty piclist */
        unlink("PACLIST");
        fp = NULL;
    }


    return fp;
}
```

Appendix D – DHGR BSaved Image Loader Source Code for DPSHOW

```
/* ------------------------------------------------------------------------
System       : Manx Aztec C65 Version 3.2b
               MS-DOS cross-development environment
Platform     : Apple IIe 128K PRODOS 8
Program      : d2lode.c
Description  : G2 Library Routine

               for ProDOS 8 Sys Programs
               Uses Page 2 Double Hi-Res to avoid
               ProDOS Sys Program Load Address at 0x2000

               Double Hi-Res 140 x 192 x 16 color Image Loader
               Loads two common non-compressed bsaved graphics
               image formats associated with double hires mode.

Written by   : Bill Buckels
Date Written : February 2013
Revision     : 1.0 First Release
Licence      : You may use this routine for whatever you wish as long
               as you agree that Bill Buckels has no warranty or
               liability obligations whatsoever from said use.
------------------------------------------------------------------------ */
#include <fcntl.h>


/* The convention of calling the second image a .AUX file is
   supported in this loader when loading a 2 part file... the load is
   split in the middle after loading the first half into auxiliary
   memory */

extern unsigned HB[]; /* page 2 scanline origins */

int d2lode(name)
char *name;
{
   int fh, status=-2;
   int c, fa = 0, fl = 8192, jdx, idx;
   char name2[64];

   jdx = 999;
   for (idx = 0; name[idx] != 0; idx++) {
        name2[idx] = name[idx];
        if (name[idx] == '.') jdx = idx;
   }
   name2[idx] = 0;

   if (jdx != 999) name2[jdx] = 0;
   strcat(name2,".AUX");

   fl = 8192;
   fh = open(name2,O_RDONLY,0x3C); /* open a binary file */
   if (fh == -1) {
       fl = 16384;
       fh = open(name,O_RDONLY,0x3C);
       if (fh == -1)return -1;
   }


   switch(fl) {
       case 16384:
       case 8192:
           /* read to main memory */
           c = read(fh,(char *)HB[0],8192);
           if (c != 8192)break;
           /* move to auxiliary memory */
           maintoaux(HB[0],HB[0]+8191,HB[0]);

           if (fl == 8192) {
                close(fh);
                fh = open(name,O_RDONLY,0x3C); /* open a binary file */
                if (fh == -1)return -1;

           }
           /* read to main memory */
           c = read(fh,(char *)HB[0],8192);
           if (c != 8192)break;
           status=0;
           break;
   }

   close(fh);

   return status;
}
```

Appendix E – DHGR Raster Image Loader Source Code for DPSHOW

```
/* ------------------------------------------------------------------------
System       : Manx Aztec C65 Version 3.2b
               MS-DOS cross-development environment
Platform     : Apple IIe 128K PRODOS 8
Program      : d2pld.c
Description  : G2 Library Routine

               for ProDOS 8 Sys Programs ONLY!
               Uses Page 2 Double Hi-Res to avoid
               ProDOS Sys Program Load Address at 0x2000

               Double Hi-Res 140 x 192 x 16 color Image Loader
               for dhx and dhr images.

Written by   : Bill Buckels
Date Written : February 2013
Revision     : 1.0 First Release
Licence      : You may use this routine for whatever you wish as long
               as you agree that Bill Buckels has no warranty or
               liability obligations whatsoever from said use.
----------------------------------------------------------------------- */
#include <fcntl.h>

extern unsigned HB[];

#define BLOCK_SIZE 3840

int d2pld(name)
char *name;
{
   unsigned int src, src2, target;
   int fh,x,x1=0,x2=0,xcnt,y1=0,offset=0,bytes=0,bank=0;
   unsigned char *ptra,*ptrm,ch,*buf,*mainbuf,*auxbuf;

   /* point to unused memory below SYS programs */
   buf = (unsigned char *)4192;
   mainbuf = (unsigned char *)&buf[BLOCK_SIZE];
   auxbuf = (unsigned char *)&mainbuf[40];


   fh = open(name,O_RDONLY,0x3C); /* open a binary file */
   if (fh == -1) return -1;

   x = read(fh,buf,5);
   if (x!= 5) {
       close(fh);
       return -1;
   }

   if (buf[0] == 'D' && buf[1] == 'H' && buf[3] == 80 &&
       buf[4] == 192 && (buf[2] == 'R' || buf[2] == 'X')) {
       ch = buf[2];
   }
   else {
       close(fh);
       return -2;
   }

   /* raster oriented raw data */
   if (ch == 'R') {
       /* read 4 blocks of 48 scanlines */
       for (x = 0; x < 4; x++) {
           if (read(fh,buf,BLOCK_SIZE) < BLOCK_SIZE) {
                close(fh);
                return -3;
           }
           /* display */
           for (x1 = 0, x2=40; x1 < BLOCK_SIZE; x1+=80,x2+=80) {

                /* read the keyboard buffer */
                /* stop if keypress */
                ptra = (unsigned char*)0xC000;
                if (ptra[0] > 127) {
                  x = 4;
                  break;
                }

                src = (unsigned int)&buf[x1];
                src2 = src+39;
                ptrm = (unsigned char *)&buf[x2];
                target = HB[y1];
                maintoaux(src,src2,target);
                movmem(ptrm,target,40);
                y1++;
           }
       }
       close(fh);
       return 0; /* done and outa' here */
    }


    /* 1280 is an arbitrary number borrowed from the shell version
       of this program... it is conceivable that a simple file
       could encode to some small size but I doubt if it would
       ever be below a 12th of the size of its uncompressed counterpart */
    if (read(fh,buf,BLOCK_SIZE) < 1280) {
       close(fh);
       return -3;
    }

    /* raster oriented run-length encoded raw data */
    target = HB[0];
    src = (unsigned int)&auxbuf[0];
    src2 = src+39;

    do{

        ch = buf[bytes]; bytes++;
        if (bytes == BLOCK_SIZE) {
            read(fh,buf,BLOCK_SIZE);
            bytes = 0;
        }

        /* check to see if its raw */
        /* if its not, run encoded */
        if(0xC0 == (ch & 0xc0)){
            xcnt = 0x3f & ch;
            ch = buf[bytes]; bytes++;
            if (bytes == BLOCK_SIZE) {
                read(fh,buf,BLOCK_SIZE);
                bytes = 0;
            }
        }
```

```
        else
          xcnt = 1;

        for(x=0;x<xcnt;x++){
            if (offset < 15360) {

                /* bank 0 - aux mem
                   bank 1 - main mem */
                if (bank == 0) {
                   auxbuf[x2] = ch;
                   bank = 1;
                }
                else {
                   mainbuf[x2] = ch;
                   x2++;
                   bank = 0;
                }

                x1++;

                /* move down one raster every 80 bytes */
                if (x1 >= 80) {
                    maintoaux(src,src2,target);
                    movmem(mainbuf,target,40);
                    x1 = x2 = bank = 0;
                    y1++;
                    if (y1 >= 192)break;
                    target = HB[y1];

                    /* read the keyboard buffer */
                    /* stop if keypress */
                    ptra = (unsigned char*)0xC000;
                    if (ptra[0] > 127) {
                      y1 = 192;
                      offset = 15360;
                      break;
                    }

                }
            }
            else break;

            offset++;
        }
        if (y1 >= 192)break;

     } while(offset<15360);

   close(fh);

   return 0;
}
```

Appendix F – DHGR Init Routine Source Code for DPSHOW

```
/* ------------------------------------------------------------------------
System       : Manx Aztec C65 Version 3.2b
               MS-DOS cross-development environment
Platform     : Apple IIe 128K PRODOS 8
Program      : d2init.c
Description  : G2 Library Routine
               Double Hi-Res Routines

               for ProDOS 8 Sys Programs
               Uses Page 2 Double Hi-Res to avoid
               ProDOS Sys Program Load Address at 0x2000

Written by   : Bill Buckels
Date Written : February 2013
Revision     : 1.0 First Release
Licence      : You may use these routines for whatever you wish as long
               as you agree that Bill Buckels has no warranty or
               liability obligations whatsoever from said use.
------------------------------------------------------------------------ */

/*
    SET80STORE=$C001 ;80STORE On- enable 80-column memory mapping (WR-only)
    CLR80STORE=$C000 ;80STORE Off- disable 80-column memory mapping (Write)

unsigned char *set80store = (unsigned char *)0xc001;
unsigned char *clr80store = (unsigned char *)0xc000;

*/


d2hireson()
{

    /* page 2 hires */
#asm
    jsr  $c300 ; TURN ON 80-COLUMN MODE
    sta  $c000 ; TURN OFF 80 STORE
    sta  $c050 ; GRAPHICS
    sta  $c055 ; PAGE TWO
    sta  $c052 ; GRAPHICS ONLY, NOT MIXED
    sta  $c057 ; HI-RES
    sta  $c05e ; TURN ON DOUBLE RES
 #endasm

}

d2hiresoff()
{
#asm
    sta  $c051 ; TEXT - HIDE GRAPHICS
    sta  $c05f ; TURN OFF DOUBLE RES
    sta  $c054 ; PAGE ONE
    sta  $c001 ; TURN ON 80 STORE
    jsr  $c300 ; TURN ON 80-COLUMN MODE
#endasm
}
```

Appendix G – Auxiliary Memory Routine Source Code

```
/* move a block of data from main to auxiliary memory */
/* do an absolute ml call */
#define A1L  0x3c
#define A1H  0x3d
#define A2L  0x3e
#define A2H  0x3f
#define A4L  0x42
#define A4H  0x43

maintoaux(src1,src2,dest)
unsigned int src1,src2,dest;
{
    /* load the registers and do the call */
    unsigned char *ptr=(unsigned char *)A1L;

    *ptr++=(unsigned char)(src1&0xff);
    *ptr++=(unsigned char)(src1>>8);
    *ptr++=(unsigned char)(src2&0xff);
    *ptr  =(unsigned char)(src2>>8);

     ptr=(unsigned char *)A4L;
     *ptr++=(unsigned char)(dest&0xff);
     *ptr  =(unsigned char)(dest>>8);
#asm
    sec
    jsr  $c311
#endasm
}
.
/* move a block of data from auxiliary to main memory */
/* do an absolute ml call */
#define A1L  0x3c
#define A1H  0x3d
#define A2L  0x3e
#define A2H  0x3f
#define A4L  0x42
#define A4H  0x43

auxtomain(src1,src2,dest)
unsigned int src1,src2,dest;
{
    /* load the registers and do the call */
    unsigned char *ptr=(unsigned char *)A1L;

    *ptr++=(unsigned char)(src1&0xff);
    *ptr++=(unsigned char)(src1>>8);
    *ptr++=(unsigned char)(src2&0xff);
    *ptr  =(unsigned char)(src2>>8);

     ptr=(unsigned char *)A4L;
     *ptr++=(unsigned char)(dest&0xff);
     *ptr  =(unsigned char)(dest>>8);
#asm
    clc
    jsr  $c311
#endasm

}
```

<u>**Closing Remarks**</u>

The preceding auxiliary memory routines have been used throughout my Aztec C programs since I first wrote those routines over 20 years back. They have been part of the AppleX distribution since its inception and resurrection, and I found them to be mighty handy for my DHGR routines in ProDOS 8 SYS programs so they are well deserving of a listing here despite their humble content and the fact that I haven't touched them since I wrote them.

Space and time prevents me from documenting any of this further, but I hope that you will have the time to review the disks in DMONDEMO and find something of interest throughout and fill-in some of the blanks that I have left-out.

I should mention as well that since AppleX (Aztec C) is an MS-DOS based cross-development environment for the Apple II and runs well in DOSBox even under Windows 7, and since I have continued to write my utilities for this environment as MS-DOS programs there is a pretty fair chance that all this will be around to play with for some time to come.

The 2013 release of AppleX will have many more disk images and demos like this one to play with, and lots of functional stuff as well including a much expanded link library.

Enjoy!

Bill Buckels
bbuckels@mts.net