********** Welcome to the wonderful world of 80's micros **********

The Colour Genie EG 2000 was one of these small, first home-computers
with color (or colour ;-) in the early eighties (I think it was 1982).
This Emulator is capable of running nearly every software that was
designed for the Colour Genies Z80 microprocessor and special hardware.
It does this by emulating the Zilog Z80 CPU instruction set (which
has several thousand different opcodes, compared to the Rockwell 6502
with less than 256 opcodes). Furthermore the emulator manages each
read and write from and to memory via function pointer tables.
For each address there can be a different handling, before
and/or after reading from or writing to it.

For those of you, who just want to take a look into the software:

You need a 486++ PC, a DOS Memory extender with the DPMI 0.9 standard
(e.g. DOS HIMEM.SYS), a VESA compatible VGA card with 800x600x256 mode
(VESA mode 0x0103) and about 1-2 megabytes of free extended memory.
For your convenience the freeware DPMI interface of the DJGPP package
(CWSDPMI.EXE) is included. If the emulator locks your machine, try
this DPMI software instead of your standard DPMI host.

I would not bet this emulator runs on a 386/16, so you better use
at least a 486/66. Do *NOT* run it from Windows 95 - WIN sucks!
The DPMI emulation of W95 is horribly slow and we use a lot of DPMI,
especially to hook the timer IRQ 08, keyboard IRQ 09 and floppy IRQ 0E
- under W95 you will have "stuck keys" in the emulation.
The best setup is to just load a DPMI server like HIMEM.SYS (or
CWSDPMI.EXE) and no expanded memory manager like EMM386.EXE.

If you have a Sound Blaster (should run on any original/compatible)
you can hear the Colour Genie sound through it. The base address is
derived from the environment variable BLASTER=Axxx - so it's setting
should really match your hardware.
There is also an emulation for the PC speaker - naturally quite a bit
less bearable than the original Colour Genie sound generator was.

You can optionally use a second video adaptor (MDA or HGC) with a
monochrome monitor to see some informations concerning the emulation.
(disabled in the current compilation; recompile it with -DDEBUG option)

Start the emulator by entering:
  CGENIE<ENTER>
You should hear your Floppy Drive B (if you have one) scratch around
a bit and finally be impressed by a virtual Colour Genie EG 2000.

First of all some words about the keyboard:
By default the keys are mapped to the keyboard layout for a german
102 key MF-keyboard (sorry aliens, this time the other way round).
This is switchable to a closer emulation of the Colour Genie keyboard
layout with the key combination <AltGr>+<K>. This is useful for games,
as there is no autorepeat interacting with you holding down some
fire or arrow keys. Take a look at the image at the bottom of the screen
(pops up if you hold <AltGr>) for the keyboard details.
The language for "nice" mode (this means autorepeat and layout far away
from the Colour Genie) is switchable between DE and US (Deutschland and
United States).

Colour Genie's Special keys are supported in the following way:
F1-F4   mapped to F1 - F4

```
CLEAR   mapped to F6
BREAK   mapped to F7
RST     mapped to F8 (NMI - non maskable interrupt)
CTRL    mapped to CTRL (STRG in german)
MOD SEL mapped to ALT (left ALT key only)
RPT     mapped to the key left of <1!> - <^ø> on DE keyboard

Some useful keys during emulation are:
F5      toggle keyboard between "nice mode" and "game mode"
F9      does a complete reset, including simulated hardware
F10     or ScrollLock freezes the CPU emulation and pops up info panel
F11     set emulation to original speed
F12     set emulation to maximum speed (no CPU cycle locking)
Grey -  decrease emulation speed by 1% of original speed
Grey +  increase emulation speed by 1% of original speed

Finally you can exit the emulator by pressing <AltGr>+<X>.
Other functions are listed in the popup area of the screen, which
is shown while you hold <AltGr> or pause the emulation.

Now, after you started the emulator you should read a message:

  DISK-BASIC
  FILES?

tap <ENTER>, and you will see a message:

  MEM SIZE?

tap <ENTER> again. After that you see a message:

  READY>

and this does mean the Colour Genie is ready to accept BASIC commands.
Some important commands you should know are:

  CLOAD"NAME"<ENTER>

Loads a BASIC image file into the emulators memory.
If tape emulation is turned on, it loads a file with name "name.cas"
from your hard disc into the emulators RAM. Try CLOAD"DEMO"<ENTER>
If tape emulation is turned off and a cassette player is connected to
your sound blasters line input, you could insert your Colour Genie tape,
press play and hope the best... I tried it with tapes written by the
emulation itself - and it worked. But I didn't try it with original
tapes. Maybe you will have to adjust the emulation speed a bit.

  CSAVE"NAME"<ENTER>
Stores the current BASIC image file from emulators memory.
Again if tape emulation is turned on, a .CAS BASIC image file with
the name "N.CAS" will be stored onto your hard disc.
Currently only the first character is taken for the filename.
This could be improved in this emulation by inspecting the input buffer
at Z80 memory [40A7] (normally 41E8 ...) for the complete string.

  SYSTEM<ENTER>
start or load a machine code tape.
You are asked to enter a name of up to six characters by
  *?
```

Enter the name (or at least the first character) of a machine code
tape image or *.CAS image file. For example enter
  *? PAINT<ENTER>
and the emulation will load "PAINT.CAS" (or the data from a real
tape) into memory.

If a filename is not matched with tape emulation turned on,
a scan for "NAME*.CAS" follows to match an abbreviated filename
in your current environment (path setting in CGENIE.INI).
If you have .CAS images with names of 8 characters in length,
this will help you out - because the Colour Genie supported only
6 character names for system tapes.
If that fails too, you probably will have to RESET the emulation.

After loading an image into the emulators memory, the prompt is
displayed again. Now you can enter
  *? /<ENTER>
to execute the loaded program from it's default starting address.
You can also give a decimal argument as starting address, if you
know, where you want the processor to go...
  *? /102<ENTER>
for example, would jump to the Z80 NMI vector at 0x0066 (= 102)

The Colour DOS ROM (which is loaded into the emulation by default)
extends the BASIC command set by several commands.
For example you can enter
  CMD"I"<ENTER>
to view a directory of the first floppy disc (or disc image file).
Enter CMD"I1", CMD"I2" for the other two discs (or image files).

If you enter "CMDI3" be prepared to hear your floppy drive scratching.
If you have a 5 1/4" 360K (or 180K) floppy drive B: in your machine,
you can read, write and format double density discs from and for
the Colour Genie! Other formats are supported, but you will have to
adjust the settings in the CGENIE.INI file first (take a look into
CGENIE.H for the DSK_xxx_DRV_yyy values for the Type= line).

This stuff is not tested very well but it gives you a chance to copy
files
from old discs. I recently read some 12 year old discs with a 360K drive
as drive B: on my Pentium 150... wow! what a MASS-storage medium ;-)
If you have got any old discs and they might still be readable... try it!
But better write protect them before you do anything inside the emulator.
It would be really sorry if you erase some of this old stuff by accident.

  LOAD"NAME"<ENTER> or LOAD"NAME/BAS"<ENTER>
will load a BASIC program with NAME resp. NAME/BAS from (any) floppy
disc.

  LOAD"NAME/BAS:1"
will load the BASIC program from floppy disc number 1 (or 2nd disc
image).

  SAVE"NAME:0"<ENTER> or SAVE"NAME/BAS:0"<ENTER>
will save a BASIC program in memory to floppy disc 0 (or 1st disc image).
Don't omit the trailing :drive number part, as otherwise the Colour DOS
will scan each drive for the existence of a file with that name.
And that will give you an error message for drive 3 (the real floppy disc
drive B:), if there is no readable and writable disc in it.

  CMD"S NAME/CMD"<ENTER>
starts the given machine code program from (any) floppy disc.
If you misspell a name, the Colour DOS searches all discs, and this includes your "drive 3" - the real floppy drive B: on your machine.
So you will have to wait a moment until you get an error.
The space between S and the filename can be omitted and the trailing doublequote too. So you could also type
  CMD"SNAME/CMD<ENTER>
for the same result.

  CMD"LNAME"<ENTER>
will try load an binary image from floppy disc (or any disc image file) into the emulators memory.

  KILL"NAME/BAS:0"<ENTER>
will erase the file NAME/BAS on drive 0. Sorry, no wildcards supported by Colour DOS :-(

  CMD"F3"<ENTER>
would format a floppy with 40 tracks, single sided, double density with 18 sectors of 256 bytes each in your physical floppy drive B:.
This is the default setting of Colour DOS.

You can change the formats of any disc drives by entering
  CMD"<n=x"<ENTER>

where "n" is the drive number and "x" is the format (for example CMD"<3=L" sets floppy drive 3 to 80 tracks, double sided, double density with 18 sectors of 256 bytes each - that is the maximum for Colour DOS!).

The available formats are listed now. Usable ones are marked with a "+" sign,
unusables are marked with a "-" sign:

```
A   - single density, 40 tracks, 1 side, 10 sectors
B   - single density, 40 tracks, 2 sides, 10 sectors
C   + double density, 40 tracks, 1 side, 18 sectors (default)
D   + double density, 40 tracks, 2 sides, 18 sectors
E   - single density, 40 tracks, 1 side, 10 sectors
F   - single density, 40 tracks, 2 sides, 10 sectors
G   + double density, 40 tracks, 1 side, 18 sectors
H   + double density, 40 tracks, 2 sides, 18 sectors
I   - single density, 80 tracks, 1 side, 10 sectors
J   - single density, 80 tracks, 2 sides, 10 sectors
K   + double density, 80 tracks, 1 side, 18 sectors
L   + double density, 80 tracks, 2 sides, 18 sectors
```

The unusable formats are because the PC floppy controller (NEC 765 or compatible) has no selectable clock speed for single density, which would be 125 kbps.
There are only 250 kbps, 500 kbps and sometimes even higher rates.
Maybe this would work, if one takes his solder and some tin !?
I never tried it, but the code is prepared to distinguish between single and double density.
If you really have some interesting stuff for the Colour Genie on SD discs,
you should try to make your NEC765 (or compatible) floppy disc controller run that slow. I would like to hear of it, if it works!
You will then have to recompile the whole thing and take a look into NEC765.C and enable the single density stuff.

The emulation sets the Colour DOS memory area for floppy formats at
a first access after a cold start to the corresponding information
out of the *.CGD image files. So you don't have to change the CMD"<x=y"
formats for bigger image files every time.
On the other hand it stores the information from this memory area
into the *.CGD image, if you "format" an image file.
So you can change the filename for [Drive3] in CGENIE.INI to something,
Set Type=0 (this is DSK_IMAGE) and format the new image from inside
the emulation by:
  CMD"<3=L"<ENTER>
  CMD"F3"
to a big image of 720K (80 tracks, 2 heads, 18 sectors per track).
Now you can copy some software from tapes to this "disc" by using
"COLOFF/CMD" or some similiar program to convert between tape and disc.

Now the hardware components and Z80 memory addresses shall be mentioned:

```
Memory       Type         Description
-----------------------------------------------------------------
0000-2FFF ROM R- 12 K Microsoft Basic (1st part of CGENIE.ROM)
3000-3FFF ROM R-  4 K Colour Extension Rom (2nd part of CGENIE.ROM)
4000-BFFF RAM RW 32 K RAM 2114-500 ns (non-EDO, not cacheable ;-)
C000-DFFF ROM R-  8 K Colour Genie DOS (optional from file CGDOS.ROM)
E000-EFFF ROM RW  4 K free ROM area (optional from file NEWE000.ROM)
F000-F3FF RAM RW  1 K colour RAM for 16 color text mode (4 bit only!)
F400-F7FF RAM RW  1 K font RAM for 128 definable 8x8 characters
F800-F8FF KBD R-  256 byte keyboard matrix for 64 keys (8 rows by 8
columns)
F900-FFE0 --- --  KBD or any other noise on the data bus ;-)
FFE0-FFE3 FDC RW  IRQ status (r) register and FDC motor- and head select
(w)
FFE4    I/O RW  memory mapped I/O - decoded but currently unused
FFE5    I/O RW  memory mapped I/O - decoded but currently unused
FFE6    I/O RW  memory mapped I/O - decoded but currently unused
FFE7    I/O RW  memory mapped I/O - decoded but currently unused
FFE8-FFEB PRT RW  memory mapped I/O (compatible to TRS80s printer port)
FFEC    FDC RW  FDC (WD179x) status (r) and command (w) register
FFED    FDC RW  FDC track number
FFEE    FDC RW  FDC sector number
FFEF    FDC RW  FDC input / output register
FFF0-FFFF --- RW   mirrored  FFE0-FFEF once again

I/O     Type         Description
-----------------------------------------------------------------------
00-F7   ---   this is unused I/O space
F8      PSG   Register select for the AY-3 8912
F9      PSG   Data read/write for the AY-3 8912
FA      CRT   Register select for the 6845 CRT controller
FB      CRT   Data read/write for the 6845 CRT controller
FC-FE   ---   this is unused I/O space
FF      XFF   multi function port with the following bits

FF.0    Cassette I/O (input with sound blaster only)
FF.1    ???
FF.2    ???
FF.3-FF.4 character generator select
FF.5    graphics/text mode select (FGR=1, LGR=0)
FF.6-FF.7 background colour select
```

Some more details about the hard- & software of a Colour Genie:

The text mode had default resolution of 40 x 24 (later 40 x 25) characters
of 8 by 8 pixels each. This is an amazing dot resolution of 320x192 pixels
(320x200 later). The 1 K * 4 colour RAM was used to generate one of
16 colours for every characters turned on pixels. Used with a TV set,
these colours were extremely poor and you could hardly distinguish one
from another. This is simulated very well in this emulation ;-)
The background colour (first there was only one, later four) was used
for the whole screen background (all the screen border and background of
any unset pixel in character or graphics mode) and was really used for
effects only like flashing screen at explosions by writing random
garbage to port FF (which forces the emulation to do much stuff).

The default memory address for text was 4400-47BF (4400-47E7 later).
The colour RAM mapped corresponding to the lower 10 bits of the video
address generated by the 6845 CRT controller - and so it was sometimes
repeated if the screen memory size exceeded 1024 bytes (which can be
seen in TEXTSYS/CMD if you use boldface, underline etc. for example).
Accessing the font RAM always produced bus conflicts with the video
hardware. Writing to the font RAM caused funny sparks to flicker all
over the screen (and there was no way to detect the vertical sync
phase for us programmers to achieve nicer character animation :-( )
This part of the hardware is not yet emulated ;-) but I'm thinking
about it...

The graphics mode was capable of 4 fixed colours
(black,blue,orange,green)
and could also be combined with the screen background colour(s).
The resolution was 160 x 100 pixels (160 x 102 later).

Each pixel was defined by 2 bits from top left to bottom right,
where bits 7+6 of each byte were the leftmost and bit 1+0 the
rightmost (very close to the CGA mode 4 colour mapping mode of our PC
hardware - let's better say ex-hardware, or do you have a CGA anymore?).
The default memory address for graphics was 4800-4BBF (4800-4BE7 later).
The colour and font RAM were unused in this mode.

Some games used a technique known under the term "page flipping". This
means they draw their next frame in a buffer currently invisible and
after that switched the register contents of the 6845 to that area.
The emulation tries to take care of this by buffering the current
video contents in a 16K area and comparing/updating only the changes
at page flipping. Otherwise the entire screen would have to be drawn
at every change to the CRT base address register.
The usable memory for video on the Colour Genie was limited to 4000-7FFF.
One of the games using FGR with three(!) frame buffers was ELIMIN/CMD,
the Eliminator clone by Harald Boegeholz. I can't guess, why he used
three buffers... But maybe it was because of the twinkling stars!?
CHOPPER used a small LGR screen with page flipping to avoid the need
of refilling the colour RAM - which could not have been done fast
enough without visible effects like "colour trailing".
You can see such effects in some other games that use side scroll
with no page flipping at all.

The kernel (a great word for a small piece of handmade code)
was almost compatible to that of the Tandy TRS80 Model I
Level II Basic. In fact, the first 12 K were nearly identical to

Microsoft's original 12K Basic and were only patched here and there
to extend the BASIC command set and store special functions for
supporting the slightly different hardware of a Colour Genie.
One amusing example is the BASIC token "FCOLOUR", which consits of
7 characters. This was too big for the original BASIC kernel to
handle (it was limited to 6 characters maximum per token) and
so the tokens name in the list was just "FCOLOU" and they tested
the BASIC program text each time this token was found, if a literal
"R" was following immediately.
Great! And really fast - of course... ;-)

I myself did also some hacks on this BASIC. The TRS80 tokens SET, RESET
and CHECK were first unused on the Colour Genie. We implemented them
as bit test and manipulation instructions - close to the Z80s
BIT, RES and SET opcodes. But nobody ever used them, because they were
incomaptible with old BASIC ROMs :-(

The compatibility to the TRS80 Level II Basic made it possible to
convert some of the programs written for the TRS80 to the Colour Genie.
But they all had to deal with the different screen layout (the TRS80 had
a 64 x 16 character screen) and some address range restrictions
(the TRS80 video ram was fixed at 3C00-3FFF, keyboard was mapped to
3800-38FF, FDC to 3FE0-3FFF).

The sound generator AY 3-8912 was one of that simple things used in
the early 80s. It was capable of mixing three channels with square
wave signals (which they just tied together, so it produced heavy
interferences every time you turned on more than one channel at a time
at frequencies with little difference) and one noise generator.
It supported 16 different "shapes" or amplitude envelopes with 16
amplitude steps each.
Crude and rough, but therefore relatively easy to emulate by software.
The emulation is close to perfect now, I would say. Everything sounds
as I "hear" it in my mind. Direct amplitude control is working now.
I used it in one of my last games for the Colour Genie "Crazy Paint".
There is digitized voice in it - listen! my voice digitized in 1984.
A somehow special year anyway...

The sound generator also provided two 8 bit bidirectional I/O ports.
These were used for the Colour Genies printer port (and reading the state
of an optional analogue joystick and its keypad too).
By the moment, only the printing capabilities are emulated.
There is a lot to do about reading some virtual joystick(s) from this
connection. But I have no idea, how these PC analogue sticks are working.
Maybe you are the one who knows??

Now some words about the reason I wrote this emulator:

I started my hobby and later profession - micro computers - in 1980
with a TRS80 model II compatible(!) computer called Video Genie II.
I bought it in Germany from the same company (Trommeschlaeger Computer
GmbH - not Trommelschlaeger as often spelled wrong - to his
inconvenience,
poor guy. He died in his own aeroplane crossing the atlantic some rumours
could be hear. Anybody knows more about that?) - that introduced the
Colour Genie in Germany some years later.
This company had it's main location just 10 minutes walk away from
my home, and so I ran to them every couple of days to get some new
software stuff for my Video Genie II (tapes with games, tools and so on).

Some guy known to them (Hi Frank!) realized the lack of software
availability in these days and started a great career as software
pirate :-o
He was selling all these American games for the Tandy TRS80
(like Big Five Software, Adventure International, Sega and so on)
for some bucks (eeeh Deutsch-Marks) and I spent my whole money
(which was not much at the age of 14 or 15) for this games.
As Trommeschlaeger introduced the Colour Genie, I was asked if I
would try to write some game program for this computer.
They generously supported me with a Colour Genie EG 2000 and a
cassette recorder and I wrote my first Program for this machine
(HEKTIK) in BASIC, using a fine BASIC Compiler they ripped from the
TRS80 sources (I think it was based on Z-Basic by Simutek - and I
later tried to rip it myself, sources are included in some image file).

But the real thing is machine code I knew and used to know before.
There was no assembler for the Colour Genie in these days and
without a floppy disc drive there was no way to develop programs
on a TRS80 (or Video Genie) and load them into a Colour Genie.
So I started programming games with a debugger (COLMON).
It allowed me to modify memory in hex and later disassemble it.
The biggest thing I ever wrote this way was CHOPPER.
It filled the whole RAM with its four levels and code.
And I wrote it in hex ;-)
I believe, I will never forget the Z80 opcodes... CD C9 01

So, you can probably understand why I glorify these days of micro-
computing. It felt a bit like exploring the "Wild West" with nothing
but a horse and a wagon with some tools and much hope and trust
in god.

Nowadays you start your computer, it loads a kernel of some megabytes
in a hurry and you don't even have an imagination of what is running
where and doing what in your computers huge amounts of real and virtual
memory - even if you are a *good* programmer you're not able to know
what makes your x-hardware not to work with your y-software...

So, if you laugh at the things that run on a Colour Genie
(or on its emulation today), remember the way the software was
developed and the time when this code was written. By the way:
CHOP32 and KONG32 do not mean they are 32-bit versions.. :-)))
They required the *full* memory extension to 32 KByte to run!
See this emulation as some trial to conservate history - it would be
sorry if all these old machines and programs were forgotten some day.

Have fun, make love, play CHOPPER

Juergen Buchmueller

P.S.:
Feel free to do what you want with this emulator - but don't make me
responsible for anything that might happen to your hardware and/or
other software if you really try it.
There is absolutely no warranty this program does anything at all.
And if it does anything, it must have nothing to do with what you
expected it to do. And if it does anything - be happy and smile.
The Z80 software contained in this package is not free of copyrights
in all parts. If you consider it a crime to spread one of your
thingies inside this package, inform me as soon as possible and
I will remove the corresponding software from future releases.

The Trommeschlaeger Computer GmbH does not exist anymore and even
EACA seems to have gone to the eternal hunting grounds of computing.
(is that the correct term for it?)

But besides this, if you want to support my work in any way, just
mail me what you have done or have changed. Even if you just find
some bug in my description, the emulation or the environment,
e-mail me at:
pullmoll@t-online.de

or take a look at my home page at the Deutsche Telekom home page server:
http://home.t-online.de/home/pullmoll

You can also send me an old fashioned letter or postcard:
Juergen Buchmueller
Koelnstrasse 429
53117 Bonn
Germany

And if you are in a hurry, you can try to call me by phone:
+49 228 9888860

P.P.S:
The whole thing is compiled using DJGPP Version 2.0, the DOS and DPMI
port, written by D.J. Delorie, of the *famous* GNU C/C++ compiler.
And so the GPL (General Public License) for GNU is also considered
valid for this project. Read the file COPYING in this archive for
more details about GNU, the GPL and the backgrounds.
The project is best managed with RHIDE, an integrated development
environment for DJGPP written by Robert Hoehne.
The project file CGENIE.GPR is included in the /SRC directory.
All this software is available via Internet FTP from the SimTel
archives. Use an Archie or FTP client to find out your closest
download location.
I got this stuff from FTP://ftp.zcu.cz/pub/gnu/djgpp.
But there might be faster locations to get the compiler and tools.
Take a look at D.J. Delories http://www.delorie.com first!