Default

gega

COLLABORATORS

	TITLE : Default	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	gega	August 26, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

Default

1.1 cp4 guide

```
            C= Commodore C+4 Emulator                                          ↩
               by gega

This documentation is still under development!


             Requirements

             Installation

             Video drivers

             Keyboard
            ..
             Keyboard layout

             Prefs file

             Command line options

             Prefs GUI

             Debug GUI

             Monitor GUI

             Stack GUI

             For Amiga programmers
            ..
             Video drivers

             For Plus4 programmers
            ..
             AssP4 help
            ..
             Statistics
```

```
                Screenshots
               Written by gega (Gáti Gergely)
E-Mail:   gatig@dragon.klte.hu
HomePage: www.klte.hu/~gatig/cp4.html

        Compiled with gcc and PhxAss, GUI designed with GadToolsBox.

Thanks to
        (in alphabetical order :-)
        Fehér Imre, for the Plus4 measurements and the programming advices,
        Ipacs Zsolt for the CPU-debugging and for the PC version,
        Kémeri Csaba /UnReal/ for testing, hardware support, icon drawing,
    and other stuff,
        Rajnai Álmos for encourageing me to start this project
        Zavacki Ferenc for a lot of help and for all of the C2Ps,
  ...and to everybody, who's interested in the Plus4
```

1.2 Requirements

```
  Min. 2M RAM - usable drivers: mono, windowmono
    3M RAM - recommended
  OCS,ECS - usable drivers: windowmono, mono, windowamiga, OCS
  OS3.0
  iec.library - only if you want to connect & use a real 1541
  Processor - 020+
  ROM images - Original C=+4 roms
```

1.3 Install

There is nothing special about the installation. If you want to store your .prefs file in the ENV: & ENVARC: directories, you have to make a dummy file called cp4.prefs in ENV:, before you start the program! (echo >ENV:cp4.prefs) Before starting, you have to copy the ROM images into the same directory using the following names:

```
        rom.basic               (16K)
        rom.kernal              (16K)
        rom.funclow             (16K)
        rom.funchigh            (16K)
```

A kernal and a basic ROM are required for using the emulator. If you only have these two, you will get a C16 emulation with 64 KBytes. At last, copy an icon next to the main program, and rename it appropriately.

If you haven't got a 1541 & a cable, to transfer the roms, download these from my homepage, www.klte.hu/~gatig/cp4_roms.lzx.

1.4 Video Driverek

After the first start, it is recommended to select a ↩ suitable video
driver. The most compatible and the default video driver is windowcard.c2p, because it works on most Amigas. This is a good choice if you have a video card, but it's very slow on AGA. Recommended video drivers for machines with more than 3M RAM:

Graphics	-030	040+
VideoCard	card, windowcard	+windowscale
AGA	zavacki	+windowamiga
ECS,OCS	OCS	+windowamiga

For less than 3M RAM, you must use the mono or the windowmono driver.

If you are a programmer, you can write a new external video driver for the emulator.

1.5 Keystrokes

You can use the following keys:

F5	Display OFF Useful on slow machines, when you want to listen to music
F6	Binary file loading (LOAD"program",8,1)
F7	Swap joystick port (See: Swap Joy -Gadget/Menu)
F8	Reset
F9	Hard Reset
F10	Debug
HELP	Prefs

1.6 Keyboard matrix

Keyboard layout (on a german keyboard)

```
C+4:  @ F3  F2  F1  HELP  £ Ret InstDel
Amiga:  ü f3  f2  f1  f4  RAmiga  Enter Backspace
C+4:  Shift E S Z 4 A W 3
Amiga:  Shift e s y 4 a w 3
C+4:  X T F C 6 D R 5
Amiga:  x t f c 6 d r 5
C+4:  V U H B 8 G Y 7
Amiga:  v u h b 8 g z 7
C+4:  N O K M 0 J I 9
Amiga:  n o k m 0 j i 9
C+4:  , - : . Up  L P Down
```

```
Amiga:  , \ ö . Up  l p Down
C+4:  / + = Esc Right ; * Left
Amiga:  - ' RAlt  Esc Right ä # Left
C+4:  Run/St  Q C=  Space 2 CTRL  Clr/H 1
Amiga:  Ctrl  q LAlt  Space 2 Tab Del 1
```

1.7 Prefs file

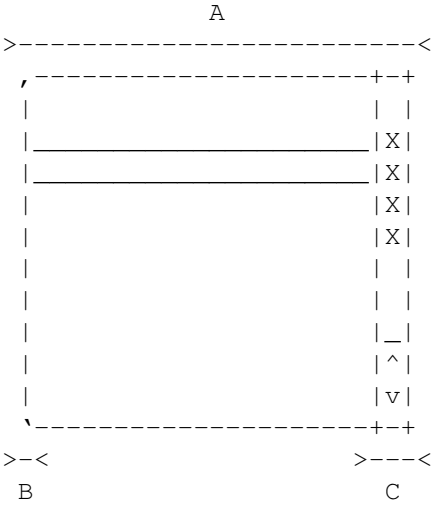
You can set most of the options in the program, via menuitems, gadgets or the keyboard, but there are some rarely used options which can be changed only by editing the .prefs file by hand. They are marked with a '!'. And there are some - marked with '+' - which are adjustable by cli options. You may edit the .prefs file by hand, for setting any option. Where you must enter a number, you can use one of the following templates:

```
decimal_digits          - decimal default
$hex_digits             - hexadecimal (old format)
0xhex_digits            - hexadecimal ('C' format)
```

The program will save all options at every exit, which was not caused by an error, so you do not have to save the modified options every time. These options include the position of every window.

```
TWOFRAME=NO                    See: Prefs,
        TwoFrame-
       Gadget/Menu
C2P=card.c2p                   See: Prefs,
        Chunky2Planar-
       Gadget
SPEEDLIMIT=YES                 See: Prefs,
        Limit-
       Gadget
SOUND=YES                      See: Prefs,
        Sound-
       Gadget/Menu
PERCENT=YES                    See: Prefs/
        Display_Percent-
       Menu
!PERCENTCOLOR1=$31             The body color of the percent display
!PERCENTCOLOR0=$51             The border color of the percent display
                               You must specify C+4 colors from 0 up to
                                  $7f
OVERSCAN=-1                    Overscan code
MODEID=-1                      ScreenmodeID
                               you have to select one, before the
                               first start of the program
P4PROGDIR=:                    Directory of C+4 programs (F6)
                               If you modify the path in the
                               filerequester, this will be modified
                               automatically
!INITMEM=0000FFFFFFFF0000      The initial contents of memory
                               Unlimited pattern of two digit hex numbers
IEC=YES                        Enable ROM patch for iec.library
```

```
 REALTIME=NO                    Skip frames until reach a real Plus4 speed
+KERNAL=rom.kernal
+FUNCTIONLOW=rom.funclow
+FUNCTIONHIGH=rom.funchigh
+CARTRIDGE1LOW=NO               Name of the cartridge image file | NO
+CARTRIDGE1HIGH=NO
+CARTRIDGE2LOW=NO
+CARTRIDGE2HIGH=NO
 MONWIN=NO                      Default state of Monitor window
                                NO  - closed
                                YES - opened
 FOLLOW=NO                      See: Monitor/Follow-Menu
 WITHDATA=NO                    See: Debug/???_With_Data-Menu
 ILLOPCODES=NO                  See: Debug/Illegal_OpCodes-Menu
 MONLEFT=30                     As I wrote above, the program stores
 MONTOP=11                      the coordinates of the windows
 PRFLEFT=0
 PRFTOP=11
 DBGLEFT=0
 DBGTOP=11
 STKTOP=25
 STKLEFT=80
!LISTINNERWIDTH=24              If you set the properties of the ListView
                                scroller to non standard values, then you
                                must calculate this value yourself:
```




```
 LISTINNERWIDTH = (A-(B+C))+2
```

1.8 Options

```
    Don't use these options, I will remove them!

-h  Help
-s  No sound
-t  No speed limit
-c  Select Chunky2Planar ('-c mono.c2p')
-v  ScreenModeID
-o  Overscan Mode (1-TEXT 2-STANDARD 3-MAX 4-VIDEO)
-f  Skip a frame
-p  No speed display
```

```
ROM-image options:      (All ROM-image 16K!)
  -b  Basic             (or -0)
  -k  Kernal            (or -1)
  -2  Function-Low
  -3  Function-High
  -4  Cartridge1-Low
  -5  Cartridge1-High
  -6  Cartridge2-Low
  -7  Cartridge2-High
```

1.9 Prefs GUI

```
        Open this window with the HELP key

        Gadgets
        Sound:          Sound enabled/disabled
        Limit:          Limit speed to the speed of a real Plus4 (100%)
        TwoFrame:       Skip every second frame
                        On an M68030/50 you can reach more than 70% using
                        this option
        Swap Joys:      Changes the interpretation of the second port of the
                        Amiga either to port 1 or to port 2 of the Plus4
                        Checked means port 1
        ScreenMode:     ScreenMode requester
        Percent:        Does not work yet
        ChunkyToPlanar: Video-driver selection
        Save:           Save current settings to ENV:/ENVARC: or to
                        PROGDIR: depending on the initial location of the
                        prefs file
        Use:            Use current settings
        Cancel:         Back to emulation, without using the new settings
        Quit:           Quit & Save

        Menus
        Project/
                Load_File       Load a binary file into memory (LOAD"...",8,1)
                Save_Prog       Save the file currently in memory (SAVE"...",8)
                Reset           Like the Debug/Reset-Menu
                Back            Like the Cancel gadget
                About           My address, etc
                Quit            Like the Quit gadget
        Prefs/
                Sound           See: Sound-Gadget
                Speed_Limit     See: Limit-Gadget
                TwoFrame        See: TwoFrame-Gadget
                Swap_Joy        See: Swap Joy-Gadget
                Display_Percent Speed display enabled/disabled
                IEC             iec.library enabled/disabled
                                With some programs the ROM-patch causes
                                C+4 crash or other bug. Then try to
                                disable this option.
                RealTime        Skip frames until reach a real Plus4 speed
```

1.10 Debug GUI

```
        Open this window with the F10 key

        Gadgets
        DisAsm ListView   Disassembly of the running program
                          The current statement under execution is marked
                          with '>'
                          Keys:
                                   crsr:          move
                                   sh+crsr:       page
        PC                The next statement's address
        A,X,Y,SP          Accu, X, Y-reg, Stack Pointer
        Flagek            7501-flags
        Run               Run the program
        Trace             Execute only one statement
        Step              One statement, but JSR is one too
        BreakPoints       Breakpoints enabled
        Clear             Clear all breakpoints
        Add               The selected address will be assigned to the actual
                          breakpoint

        Below  of  these  is  the  actual  Plus4  display mode, and some TED
information:

        Amiga_raster      Not implemented yet
        C+4_raster        Plus4 raster number
        Raster_request    The raster request line (Plus4 raster)
        Video_matrix      Video-Matrix address
        Gfx_base          Graphics memory address
        Ras/C#1/C#2/C#3   Enabled IRQs (+)

        Menus

        The menu is the same for the Debug, Monitor & Stack windows

        Project/
                Prefs           Open the Prefs window
                SaveMem         Save the memory
                Reset           Hard-Reset
                Back            Back to the emulation
                Quit            Quit & Save
        Debug/
                Trace           See: Trace-Gadget
                Step            See: Step-Gadget
                Run             See: Run-Gadget
                ???_With_Data   Value after unknown Opcodes
                BreakPoints     See: BreakPoints-Gadget
                Illegal_OpCodes Display all Opcodes
                Stack           Open the Stack window
        Monitor/
                OpenWin         Open the Monitor window
                Follow          The Monitor's cursor is set to the
                                executed statement's operand
                Ascii           Display in ascii
                ScreenCode      Display in screen-codes
```

1.11 Monitor GUI

```
                    Ascii/ScrCode     See: Monitor/Ascii|ScreenCode-Menu
        String gadget     Enter the string or hexa number to search for
                          Starting with '$' it will be a hex number else
                          it will be a string
        Search            Search down
        ListView          Plus4 memory
                          Useful keys:
                                   crsr:           move
                                   sh+crsr:        page
                                   numbers+'a-f'   modify
        Menus

        The
                 menu
                 is the same for the Debug, Monitor & Stack windows
```

1.12 Stack GUI

```
                        The  stack backtrace window shows the contents of the  ↩
                           stack. You can
examine the caller of the running subroutine.

        Menus

        The
                 menu
                 is the same for the Debug, Monitor & Stack windows
```

1.13 For Amiga Programmers

I will send the sources to everyone if requested. When I am satisfied with the state of the sources I will upload them to Aminet. Requests & bug-reports can be mailed to gatig@dragon.klte.hu.

1.14 Making Video drivers

Compile the modules without startup-code. GCC: -nostartup, SAS/C: NOSTARTUP. You have to open/close all libraries, if you need them, and you have to get the ExecBase from 4. The following 5 names may not be changed, and you must define all of them.

```
---CUT HERE---
#includes

#define C2P_INFO  "long story..."
```

```
#define C2P_VERSION "1"
#define C2P_REVISION  "0"
#define C2P_AUTHOR  "author..."
#define C2P_NAME  "drivername..."
// If you define the next one, the main program will not render the
// actual emulation speed
#define C2P_NOSPEED
// You can select the desired buffering (single,double,triple):
// none/0       - You'll get NULLs as delta
// 1,2,3        - You'll get the right delta buffers
#define C2P_BUFFERING 3

#include "c2p_module.c"

//------------------------------------------------------------------------
char *SAVEDS minit(ULONG scrmode, ULONG overscan, unsigned char *linedeltatab) {
//------------------------------------------------------------------------
/*
                Initializing

        Called only once, after LoadSeg()

        Input:
                scrmode         ModeID, given by user
                overscan        Overscan mode, given by user
                linedeltatab    LineSkipBuffer
                                283 bytes, one byte per line,
                                If a byte is not zero, you do not have to draw
                                to the screen memory. Usable only, if the
                                buffering was defined correctly
        Output:
                NULL            OK
                string          Error text, usable defines:
                                C2P_NOMSG       No error message
                                C2P_NOMEM       Not enough memory
                                "custom"        Module specific error
```

You have to open a window, and a screen, if you need it. Put the address of the Screen to vec.c2p_Scr. If your window is on an existing PubScreen, you have to specify the PubScreen's address. Write your window's address to vec.c2p_Win. You don't need to specify a ScreenTitle. You can use Gadgets, Menus, etc on your window, and you have to set the necessary IDCMPs. There're two exception to this rule; you will not get any IDCMP_RAWKEY & IDCMP_VANILLAKEY messages, because the main program eats them. About the messages, see the mdo() & mdofull() functions. The Window & Screen pointers are used by the main program to perform the following functions:

Open requesters on the screen, defined by window pointer, if it's a PubScreen,
Get messages from the Window's UserPort,
Calls to ActivateWindow(), ModifyIDCMP(), SetWindowTitles() and ScreenToFront()

There are 4 functions to handle the driver specific options. The GetOptionXXX() is used to get an existing option, or create a new one. The two versions are ...Int & ...Str for integers and strings. You must specify

```
a  default  value, and if the required option is not found, this one will be
returned.

         int   GetOptionInt(char *name,int   defval);
         char *GetOptionStr(char *name,char *defval);

         void AddOptionInt(char *name,int   value);
         void AddOptionStr(char *name,char *value);

With  the  last  two, you can create new options, or modify an existing one.
The option name is created by the C2P_NAME define, so if it is unique, there
will not be any name collosions.

*/
} // minit()

//------------------------------------------------------------------------
void SAVEDS mfree(void) {
//------------------------------------------------------------------------
/*
                Freeing resources

        Immediately before UnloadSeg()-ing

        You  have  to  free all your allocated resources. You have to verify
all  resources for successful allocation, before freeing them, and first you
have to call the mawake() function.

*/
} // mfree()

//------------------------------------------------------------------------
int SAVEDS mdo(unsigned char *chunky,unsigned char *delta,int numscreen) {
//------------------------------------------------------------------------
/*
                Drawing

        Called after every logical frame

        Input:
                chunky          The screen to draw to. 283x352 bytes of chunky
                                data. Ignore the MSB. These are Plus4
                                colors, $x0 are black.
                delta           Screen array, like above, but, this is the
                                current visible screen, if you gave the
                                buffering number correctly.
                numscreen       The ordinal number of the next screen,
                                from 0 to the C2P_BUFFERING-1.
                                ('3' ->  0 1 2 0 1...)

        Deltaconversion.  The IDCMP messages are in the vec.c2p_MsgList Exec
List, but it contains only the last frame's messages. If you don't handle an
event  here,  you will not get it in the future. See this code fragment, for
an example, for collecting all of the messages of the last logical frame:

  struct IntuiMessage *imsg;
  while(imsg=(struct IntuiMessage *)RemHead(&vec.c2p_MsgList)) {
```

```
    switch(imsg->Class) {...}
  }

In  the vec.c2p_Speed field, you find the actual percentage value, or -1, if
the  user  does  not  want  a  speed  display. You have to return one of the
following values:

        RET_OK          - Normal
        RET_PREFS       - Opens the Prefs window
        RET_DEBUG       - Opens the Debug window
        RET_RESET       - Normal reset
        RET_HRESET      - Hard-Reset
        RET_QUIT        - Quit
        RET_NEWWIN      - When you have opened a new window, you must notify
                          the main program about it
        RET_ERROR       - Fatal error! At smaller errors, return RET_PREFS
                          only

*/
} // mdo()

//------------------------------------------------------------------------
int SAVEDS mdofull(unsigned char *chunky,int numscreen) {
//------------------------------------------------------------------------
/*

        The same like mdo(), but there is no delta buffer.

*/
} // mdofull()

//------------------------------------------------------------------------
int SAVEDS mdont(void) {
//------------------------------------------------------------------------
/*

        This  one  is the UserBlank case routine. If the user pressed the F5
key, (no gfx required) this will be the frame handler. The vec.c2p_Speed and
the user input is still available.

*/
} // mdont()

//------------------------------------------------------------------------
void SAVEDS msleep(void) {
//------------------------------------------------------------------------
/*

        Set busy pointer, no more input until a mawake() call.

*/
} // msleep()

//------------------------------------------------------------------------
void SAVEDS mawake(void) {
//------------------------------------------------------------------------
/*
```

```
        Return to normal input handling.

*/
} // mawake()
---CUT HERE---
```

1.15 Plus4 programmers'

Send all compatibility & other errors to gatig@dragon.klte.hu, any attached files with tar/gzip/uuencode please!

If you know exactly how the up/down scrolling on the Plus4 works, please write me about it!

1.16 AssP4

There's a primitive, but easy to use assembler in this package. Fixed format, the separator is ONE tab, the start address must be at the very first line, usable only with hexadecimal numbers with the $ sign at the beginning.

```
---CUT HERE---
$start_addr

* remark

label MNM operand
  MNM operand   * remark
---CUT HERE---
```

For example:

```
---CUT HERE---
$7000

* example program

start lda #<routin
  sta c1
  lda #>routin
  sta c2
  dat #$02    * crash

routin  inc $ff19
  rti

c1  dat #$00
c2  dat #$00
---CUT HERE---
```

extra mnemonics:

```
  BSS #$num     * 'num' x #$00 bytes
  DAT #$num     * 'num' (one byte)
  STR string until '\n' in the source
```

1.17 Statistics

After about one hour playing:

```
OpCode Statistic
00  BRK IMP             -    -.-%
01  ORA (ZP,X)         87    0.0%
02 !CRA NONE            -    -.-%
03 !SLO (ZP,X)          -    -.-%
04 !NO2 IMP             -    -.-%
05  ORA ZP        1106916    0.3%
06  ASL ZP        1299123    0.3%
07 !SLO ZP              -    -.-%
08  PHP IMP         34719    0.0%
09  ORA #BYTE      307839    0.0%
0A  ASL ACC       1893801    0.5%
0B !ANC #BYTE           -    -.-%
0C !NO3 IMP             -    -.-%
0D  ORA ABS       1895561    0.5%
0E  ASL ABS          7714    0.0%
0F !SLO ABS             -    -.-%
10  BPL REL      20324363    6.0%
11  ORA (ZP),Y     672907    0.2%
12 !CRA NONE            -    -.-%
13 !SLO (ZP),Y          -    -.-%
14 !NO2 IMP             -    -.-%
15  ORA ZP,X            -    -.-%
16  ASL ZP,X          407    0.0%
17 !SLO ZP,X            -    -.-%
18  CLC IMP       3542836    1.0%
19  ORA ABS,Y        6216    0.0%
1A !NOP IMP             -    -.-%
1B !SLO ABS,Y           -    -.-%
1C !NO3 IMP             -    -.-%
1D  ORA ABS,X      279906    0.0%
1E  ASL ABS,X       24210    0.0%
1F !SLO ABS,X           -    -.-%
20  JSR ABS       5004621    1.4%
21  AND (ZP,X)          -    -.-%
22 !CRA NONE            -    -.-%
23 !RLA (ZP,X)          -    -.-%
24  BIT ZP         275222    0.0%
25  AND ZP         214057    0.0%
26  ROL ZP        1009465    0.3%
27 !RLA ZP              -    -.-%
28  PLP IMP         34760    0.0%
29  AND #BYTE     4493516    1.3%
2A  ROL IMP       1260950    0.3%
2B !ANC #BYTE           -    -.-%
2C  BIT ABS      16416421    4.8%
2D  AND ABS        163599    0.0%
```

```
2E  ROL ABS         11628    0.0%
2F !RLA ABS             -    -.-%
30  BMI REL       1883576    0.5%
31  AND (ZP),Y     161668    0.0%
32 !CRA NONE            -    -.-%
33 !RLA (ZP),Y          -    -.-%
34 !NO2 IMP             -    -.-%
35  AND ZP,X         7623    0.0%
36  ROL ZP,X            -    -.-%
37 !RLA ZP,X            -    -.-%
38  SEC IMP       1638565    0.4%
39  AND ABS,Y       40399    0.0%
3A !NOP IMP             -    -.-%
3B !RLA ABS,Y           -    -.-%
3C !NO3 IMP             -    -.-%
3D  AND ABS,X      328224    0.0%
3E  ROL ABS,X       53748    0.0%
3F !RLA ABS,X           -    -.-%
40  RTI IMP        109355    0.0%
41  EOR (ZP,X)          -    -.-%
42 !CRA NONE            -    -.-%
43 !SRE (ZP,X)          -    -.-%
44 !NO2 IMP             -    -.-%
45  EOR ZP         317623    0.0%
46  LSR ZP         448569    0.1%
47 !SRE ZP              -    -.-%
48  PHA IMP       1246910    0.3%
49  EOR #BYTE     1053301    0.3%
4A  LSR ACC       4225275    1.2%
4B !ASR #BYTE           -    -.-%
4C  JMP ABS       2643914    0.7%
4D  EOR ABS         24563    0.0%
4E  LSR ABS         39121    0.0%
4F !SRE ABS             -    -.-%
50  BVC REL         16263    0.0%
51  EOR (ZP),Y     607422    0.1%
52 !CRA NONE            -    -.-%
53 !SRE (ZP),Y          -    -.-%
54 !NO2 IMP             -    -.-%
55  EOR ZP,X        42320    0.0%
56  LSR ZP,X       189863    0.0%
57 !SRE ZP,X            -    -.-%
58  CLI IMP        174846    0.0%
59  EOR ABS,Y       16548    0.0%
5A !NOP IMP             -    -.-%
5B !SRE ABS,Y           -    -.-%
5C !NO3 IMP             -    -.-%
5D  EOR ABS,X      122675    0.0%
5E  LSR ABS,X       33535    0.0%
5F !SRE ABS,X           -    -.-%
60  RTS IMP       5005894    1.4%
61  ADC (ZP,X)          -    -.-%
62 !CRA NONE            -    -.-%
63 !RRA (ZP,X)          -    -.-%
64 !NO2 IMP             -    -.-%
65  ADC ZP        2472985    0.7%
66  ROR ZP        1117715    0.3%
```

```
67 !RRA ZP              -    -.-%
68  PLA ACC       1245200    0.3%
69  ADC #BYTE     2471303    0.7%
6A  ROR ACC       1039993    0.3%
6B !ARR #BYTE           -    -.-%
6C  JMP (ABS)       79673    0.0%
6D  ADC ABS        239556    0.0%
6E  ROR ABS         26309    0.0%
6F !RRA ABS             -    -.-%
70  BVS REL        397733    0.1%
71  ADC (ZP),Y       1603    0.0%
72 !CRA NONE            -    -.-%
73 !RRA (ZP),Y          -    -.-%
74 !NO2 IMP             -    -.-%
75  ADC ZP,X        14495    0.0%
76  ROR ZP,X       414059    0.1%
77 !RRA ZP,X            -    -.-%
78  SEI IMP        161303    0.0%
79  ADC ABS,Y      680902    0.2%
7A !NOP IMP             -    -.-%
7B !RRA ABS,Y           -    -.-%
7C !NO3 IMP             -    -.-%
7D  ADC ABS,X      495578    0.1%
7E  ROR ABS,X       86698    0.0%
7F !RRA ABS,X           -    -.-%
80 !NO2 IMP             -    -.-%
81  STA (ZP,X)         40    0.0%
82 !NO2 IMP             -    -.-%
83 !SAX (ZP,X)          -    -.-%
84  STY ZP        1231196    0.3%
85  STA ZP       12594358    3.7%
86  STX ZP        1414164    0.4%
87 !SAX ZP              -    -.-%
88  DEY IMP      12392369    3.6%
89 !NO2 IMP             -    -.-%
8A  TXA IMP       1962708    0.5%
8B !ANE #BYTE           -    -.-%
8C  STY ABS        252438    0.0%
8D  STA ABS       3505119    1.0%
8E  STX ABS        155158    0.0%
8F !SAX ABS             -    -.-%
90  BCC REL       4716846    1.4%
91  STA (ZP),Y    7737205    2.3%
92 !CRA NONE            -    -.-%
93 !SHA (ZP),Y          -    -.-%
94  STY ZP,X        58953    0.0%
95  STA ZP,X       896237    0.2%
96  STX ZP,Y        27471    0.0%
97 !SAX ZP,Y            -    -.-%
98  TYA IMP       1483139    0.4%
99  STA ABS,Y     2317664    0.6%
9A  TXS IMP           799    0.0%
9B !SHS ABS,Y           -    -.-%
9C !SHY ABS,X           -    -.-%
9D  STA ABS,X     4855300    1.4%
9E !SHX ABS,Y           -    -.-%
9F !SHA ABS,Y           -    -.-%
```

```
A0  LDY #BYTE    3292382    0.9%
A1  LDA (ZP,X)       230    0.0%
A2  LDX #BYTE    1318529    0.3%
A3 !LAX (ZP,X)         -    -.-%
A4  LDY ZP       1673542    0.4%
A5  LDA ZP      12822613    3.8%
A6  LDX ZP       2306028    0.6%
A7 !LAX ZP             -    -.-%
A8  TAY IMP      1807685    0.5%
A9  LDA #BYTE    3803109    1.1%
AA  TAX IMP      3633412    1.0%
AB !LXA #BYTE          -    -.-%
AC  LDY ABS      1255343    0.3%
AD  LDA ABS      9691488    2.8%
AE  LDX ABS       707963    0.2%
AF !LAX ABS            -    -.-%
B0  BCS REL      3030083    0.9%
B1  LDA (ZP),Y   4183610    1.2%
B2 !CRA NONE           -    -.-%
B3 !LAX (ZP),Y         -    -.-%
B4  LDY ZP,X        5094    0.0%
B5  LDA ZP,X      759896    0.2%
B6  LDX ZP,Y       44345    0.0%
B7 !LAX ZP,Y           -    -.-%
B8  CLV IMP           23    0.0%
B9  LDA ABS,Y    3285102    0.9%
BA  TSX IMP        28917    0.0%
BB !LAS ABS,Y          -    -.-%
BC  LDY ABS,X    1158530    0.3%
BD  LDA ABS,X    7208053    2.1%
BE  LDX ABS,Y     132547    0.0%
BF !LAX ABS,Y          -    -.-%
C0  CPY #BYTE    1907888    0.5%
C1  CMP (ZP,X)         -    -.-%
C2 !NO2 IMP            -    -.-%
C3 !DCP (ZP,X)         -    -.-%
C4  CPY ZP        353475    0.1%
C5  CMP ZP        384757    0.1%
C6  DEC ZP       2633324    0.7%
C7 !DCP ZP             -    -.-%
C8  INY IMP      3309487    0.9%
C9  CMP #BYTE    9403382    2.7%
CA  DEX IMP      9872566    2.9%
CB !SBX #BYTE          -    -.-%
CC  CPY ABS        94537    0.0%
CD  CMP ABS      3488928    1.0%
CE  DEC ABS       423934    0.1%
CF !DCP ABS            -    -.-%
D0  BNE REL     52716297   15.6%
D1  CMP (ZP),Y    272409    0.0%
D2 !CRA NONE           -    -.-%
D3 !DCP (ZP),Y         -    -.-%
D4 !NO2 IMP            -    -.-%
D5  CMP ZP,X         474    0.0%
D6  DEC ZP,X           -    -.-%
D7 !DCP ZP,X           -    -.-%
D8  CLD IMP         7134    0.0%
```

```
D9  CMP ABS,Y       5965    0.0%
DA !NOP IMP            -    -.-%
DB !DCP ABS,Y          -    -.-%
DC !NO3 IMP            -    -.-%
DD  CMP ABS,X     384237    0.1%
DE  DEC ABS,X      23223    0.0%
DF !DCP ABS,X          -    -.-%
E0  CPX #BYTE    1244094    0.3%
E1  SBC (ZP,X)         -    -.-%
E2 !NO2 IMP            -    -.-%
E3 !ISB (ZP,X)         -    -.-%
E4  CPX ZP        682812    0.2%
E5  SBC ZP        908867    0.2%
E6  INC ZP       2089052    0.6%
E7 !ISB ZP             -    -.-%
E8  INX IMP      1550920    0.4%
E9  SBC #BYTE   17356775    5.1%
EA  NOP IMP      8391166    2.4%
EB !SBC #BYTE          -    -.-%
EC  CPX ABS        17890    0.0%
ED  SBC ABS        44390    0.0%
EE  INC ABS      1327731    0.3%
EF !ISB ABS            -    -.-%
F0  BEQ REL      7511317    2.2%
F1  SBC (ZP),Y      1507    0.0%
F2 !CRA NONE           -    -.-%
F3 !ISB (ZP),Y         -    -.-%
F4 !NO2 IMP            -    -.-%
F5  SBC ZP,X         168    0.0%
F6  INC ZP,X        2262    0.0%
F7 !ISB ZP,X           -    -.-%
F8  SED IMP          156    0.0%
F9  SBC ABS,Y       5576    0.0%
FA !NOP IMP            -    -.-%
FB !ISB ABS,Y          -    -.-%
FC !NO3 IMP            -    -.-%
FD  SBC ABS,X     278592    0.0%
FE  INC ABS,X      39441    0.0%
FF !ISB ABS,X          -    -.-%
Total:    336142170 100.0%

Addressing Statistic
#BYTE   46652118   13.8%
ABS     47437926   14.1%
ZP      47355863   14.0%
ACC      8404269    2.5%
IMP     57650619   17.1%
(ZP,X)       357    0.0%
(ZP),Y  13638331    4.0%
ZP,X     2391851    0.7%
ABS,X   15371950    4.5%
ABS,Y    6490919    1.9%
REL     90596478   26.9%
(ABS)      79673    0.0%
ZP,Y       71816    0.0%
NONE           -    -.-%
Total:  336142170 100.0%
```

Instruction Statistic

ADC	6376422	1.8%
AND	5409086	1.6%
ASL	3225255	0.9%
BCC	4716846	1.4%
BCS	3030083	0.9%
BEQ	7511317	2.2%
BIT	16691643	4.9%
BMI	1883576	0.5%
BNE	52716297	15.6%
BPL	20324363	6.0%
BRK	-	-.-%
BVC	16263	0.0%
BVS	397733	0.1%
CLC	3542836	1.0%
CLD	7134	0.0%
CLI	174846	0.0%
CLV	23	0.0%
CMP	13940152	4.1%
CPX	1944796	0.5%
CPY	2355900	0.7%
DEC	3080481	0.9%
DEX	9872566	2.9%
DEY	12392369	3.6%
EOR	2184452	0.6%
INC	3458486	1.0%
INX	1550920	0.4%
INY	3309487	0.9%
JMP	2723587	0.8%
JSR	5004621	1.4%
LDA	41754101	12.4%
LDX	4509412	1.3%
LDY	7384891	2.1%
LSR	4936363	1.4%
NOP	8391166	2.4%
ORA	4269432	1.2%
PHA	1246910	0.3%
PHP	34719	0.0%
PLA	1245200	0.3%
PLP	34760	0.0%
ROL	2335791	0.6%
ROR	2684774	0.7%
RTI	109355	0.0%
RTS	5005894	1.4%
SBC	18595875	5.5%
SEC	1638565	0.4%
SED	156	0.0%
SEI	161303	0.0%
STA	31905923	9.4%
STX	1596793	0.4%
STY	1542587	0.4%
TAX	3633412	1.0%
TAY	1807685	0.5%
TSX	28917	0.0%
TXA	1962708	0.5%
TXS	799	0.0%

```
 TYA     1483139    0.4%
!CRA           -    -.-%
!NOP           -    -.-%
!NO2           -    -.-%
!NO3           -    -.-%
!SLO           -    -.-%
!ANC           -    -.-%
!RLA           -    -.-%
!SRE           -    -.-%
!RRA           -    -.-%
!ARR           -    -.-%
!SAX           -    -.-%
!ANE           -    -.-%
!SHA           -    -.-%
!SHS           -    -.-%
!SHY           -    -.-%
!SHX           -    -.-%
!LAX           -    -.-%
!LAS           -    -.-%
!DCP           -    -.-%
!SBX           -    -.-%
!ISB           -    -.-%
!SBC           -    -.-%
!ASR           -    -.-%
!LXA           -    -.-%
Total:  336142170 100.0%
```

1.18 ScreenShots

```
Picture #1
   No1  MultiView
   No1  PPShow

Picture #2
   No2  MultiView
   No2  PPShow
```