# Asm-Pro

**COLLABORATORS**

| | | | |
|---|---|---|---|
| | *TITLE* :<br><br>Asm-Pro | | |
| *ACTION* | *NAME* | *DATE* | *SIGNATURE* |
| WRITTEN BY | | January 23, 2025 | |

**REVISION HISTORY**

| NUMBER | DATE | DESCRIPTION | NAME |
|---|---|---|---|
| | | | |

# Contents

**1 Asm-Pro** **1**

1.1 Asm-Pro By Genetic . . . . 1
1.2 About Asm-Pro. . . . . 1
1.3 Asm-Pro - The story. . . . . 2
1.4 Contact Addresses . . . . 2
1.5 Requirements... . . . . 3
1.6 Installation . . . . 3
1.7 Programmer notes . . . . 3
1.8 mc680x0 Reference card . . . . 3
1.9 Programmer notes m68040/060 . . . . 8
1.10 Commandline. . . . . 9
1.11 Project . . . . 11
1.12 Editor . . . . 13
1.13 Memory . . . . 13
1.14 Insert . . . . 14
1.15 Assembler . . . . 15
1.16 Monitor . . . . 15
1.17 Diskette . . . . 16
1.18 Miscelaneous . . . . 17
1.19 Special features. . . . . 18
1.20 Special features - PCR in Debug window. . . . . 18
1.21 Env Preferences. . . . . 19
1.22 Asm Preferences. . . . . 21
1.23 Synt Preferences. . . . . 22
1.24 Special features - %Getdate/%Gettime. . . . . 23
1.25 Special features - IncIFF(p). . . . . 23
1.26 Special features - IncLINK. . . . . 26
1.27 Thanks . . . . 26
1.28 Disclaimer... . . . . 28
1.29 Frequently Asked Questions. . . . . 28
1.30 .... . . . . 28

# Chapter 1

# Asm-Pro

## 1.1 Asm-Pro By Genetic

Asm-Pro V1.16 - Coded by Solo/Genetic in 1998/99

Choose your destiny:

About Asm-Pro What is it?

The Story Background story.

Contact Addresses Our contact address.

Requirements Asm-Pro's system requirements.

Installation Installation notes.

Programmer notes Usefull notes for programmers.

Special features Special (new) features of Asm-Pro.

Commandline All commandmode options.

Preferences All Preferences options.

history file Detailed list of changed and fixed stuff.

FAQ Frequently Asked Questions.

Thanks The people that need to be thanked :).

Disclaimer The disclamer.

## 1.2 About Asm-Pro.

About Asm-Pro

Asm-Pro is a MC680x0 marco assembler with integrated editor, debugger, linker and monitor for our Amiga. With this program you have all the tools you need for the development of new software in assembly. It has full mc680x0, FPU and MMU support (and ppc support will be added asap).

This is a work in progress project and work will continue slowly but surely...

Feel free to tell me what you think of it and report me any bugs you might find. But first check the known bugs section in the history file and check my homepage for a newer version of Asm-Pro and/or the Historyfile.

Check out the story for more background info.

Solo/GnT

Credz:

All coding of Asm-Pro by Solo / Genetic

Asm-Pro Guide by Feanor / Genetic

## 1.3 Asm-Pro - The story.

Asm-Pro, the story...

Asm-Pro was inspired by the great Asm-One by TFA (v1.28). I started this project after I used Trashm'one (with it's nice inciff) for a while but it doesn't undestand 68020+ code and Asm-One does but had no inciff. So I desided to combine the two and also make it more system friendly.

There are a lot of changes and improvements over asm-one v1.28. The biggest change is that I wrote the entire editor and all other character output stuff 2 be system friendly so it also works on your favourite gfx-board. For a list of new features and updates see the major changes and for a more detailed description I suggest you read the history file.

Major Changes:

- Editor and other text output is system friendly now and works great on your favourite gfx-card.

- Redisign of the Asm-Pro look.

- Included a font requester and made the editor font sensitive.

- New gui for starting up Asm-Pro, workspace mem type and size can be saved with wp (write prefs).

- Multiple source file extentions (try "(.sl.asml.i)" as source extention to get all sources and includes in the file requester.)

- Debug registers have there own window.

- Stack size can be changed from shell or icon so even very big sources don't crash you system.

- Added the new mc68060 commands.

- Changed blok insert from Amiga-i to Amiga-v (amiga-i will still work though).

- Included my own IncIFFp routines, so it works now.

- Syntax coloring added (comments and labels).

- Added %getdate and %gettime it returns system time and date in ascii.

- Set mark keys should now also work with other keymaps (like swedish

or german) So pressing Amiga+Shift+2 will set Mark 2 etc. same goes

for delete conditions in debugmode.

Mapped AGA guide from Amiga-= to HELP-key so it won't get in the way.

- System friendly and custom scroll routine (for speed).

## 1.4 Contact Addresses

```
.____________________ :________________________.__________
| _______/ ________/ \| ___/ _________ _ /_| ______/
| \___ \ ___/ \ \ \ ___/ \ \_/ \ /\
| | \_ / \_\ \_ / \_ \ \_ / \
|__________/_________/_\______/_________/___/_______/_____/
```

Contact Adresses

Asm-Pro Homepage : http://surf.to/asmpro

E-Mail

Solo/Gnt : solognt@worldonline.nl

Feanor/Gnt : alexander.ramaker@achmea.nl

Surf to Genetic Homepage : http://surf.to/genetic

## 1.5 Requirements...

Requirements

Asm-Pro V1.16 needs the following configuration of hardware and software to be present:

· Any Amiga equipped with an MC680x0

· Kickstart version V2.04 or above

· Asl.Library

· ReqTools.Library

· MathTrans.Library

## 1.6 Installation

Installation

- Create an Asm-Pro directory on your favourite drive and copy all files/dirs from the archive to this directory.

- Optional you can assign "Asmpro:" to this directory so the Asm-Pro.guide and the Regsdata file can be found.

I am sure you've got the system include files so they are not included. If you don't you should get the Amiga developer CD.

I suggest you put the include files in AsmPro:Include/

## 1.7 Programmer notes

Programmer notes

Reference card All 680x0 instructions.

68040/060 emulation Some notes about software emulated instructions.

Debugger and 060 Debug window in 060 mode.

## 1.8 mc680x0 Reference card

Addressing mode Legends

<ea> Any Effective Address

<rea> Register Effective Address

<dea> Data Effective Address

<mea> Memory Effective Address

<cea> Control Effective Address

<aea> Alterable Effective Address (data or memory)

<adea> Alterable Data Effective Address

<amea> Alterable Memory Effective Address

<acea> Alterable Control Effective Address

Mode

Dn

An

(An)

(An)+

-(An)

d(An)

d(An,Xi)

$7fff.W

$7fffffff.L

d(PC)

d(PC,Xi)

Immediate

* 68020++ *

bd(An,Xi)

bd(PC,Xi)

[bd,An],Xi,od

[bd,An,Xi],od

[bd,PC],Xi,od

[bd,PC,Xi],od

Data Movement

Mnemonic Legal Modes Size

EXG Rm,Rn L

LEA <cea>,#<d16> L

LINK An,#<d16> unsized

MOVE <ea>,<adea> L,W,{B}

MOVEA <ea>,An L,W

MOVEM <reg.list>,<acea>+ L,W

<cea>+,<reg.list> L,W

MOVEP Dn,d(An) or d(An),Dn L,W

MOVEQ #<d8>,Dn L

PEA <cea> L

SWAP Dn W

UNLK An unsized

Integer Arithmetic

Mnemonic Legal Modes Size

ADD <ea>,Dn or Dn,<amea> L,W,{B}

ADDA <ea>,An L,W

ADDI #<data>,<adea> L,W,B

ADDQ #<d3>,<aea> L,W,{B}

ADDX Dm,Dn or -<Am),-(An) L,W,B

CLR <edea> L,W,B

CMP <ea>,Dn L,W,{B}

CMPA <ea>,An L,W

CMPI #<data>,<adea> L,W,B

CMPM (Am)+,(An)+ L,W,B

CMP2 <cea>,Rn L,W,B

DIVU/DIVS <dea>,Dn W

DIVU.L/DIVS.L <dea>,Dq L

DIVU.L/DIVS.L <dea>,Dr:Dq L

DIVUL.L/DIVSL.L <dea>,Dr:Dq L

EXT Dn L,W

EXTB Dn L

MULU/MULS <dea>,Dn W

MULU.L/MULS.L <dea>,Dl L

MULUL.L/MULSL.L <dea>,Dl:Dh L

NEG <adea> L,W,B

NEGX <adea> L,W,B

SUB <ea>,Dn or Dn,<amea> L,W,{B}

SUBA <ea>,An L,W

SUBI #<data>,<adea> L,W,B

SUBQ #<d3>,<aea> L,W,{B}

SUBX Dm,Dn or -<Am),-(An) L,W,B

Indivisible Read-Modify-Write Operations

Mnemonic Legal Modes Size

TAS <edea> B

CAS Dc,Du,<amea> L,W,B

CAS2 Dc1:Dc2,Du1:Du2, L,W,B

(Rn1):(Rn2)

TST <adea> L,W,B

Logical Operations

Mnemonic Legal Modes Size

AND <dea>,Dn or Dn,<amea> L,W,B

ANDI #<data>,<adea> L,W,B

EOR Dn,<adea> L,W,B

EORI #<data>,<adea> L,W,B

NOT <adea> L,W,B

OR <dea>,Dn or Dn,<amea> L,W,B

ORI #<data>,<adea> L,W,B

Shift and Rotate

Mnemonic Legal Modes Size

ASL/ASR Dm,Dn or #<d3>,Dn L,W,B (Dm mod 64)

ASL/ASR <amea> W (shift count=1)

LSL/LSR Dm,Dn or #<d3>,Dn L,W,B (Dm mod 64)

LSL/LSR <amea> W (shift count=1)

ROL/ROR Dm,Dn or #<d3>,Dn L,W,B (Dm mod 64)

ROL/ROR <amea> W (shift count=1)

ROXL/ROXR Dm,Dn or #<d3>,Dn L,W,B (Dm mod 64)

ROXL/ROXR <amea> W (shift count=1)

Bit Manipulation

Mnemonic Legal Modes Size

BCHG Dm,Dn or #<d5>,Dn L (Dm mod 32)

BCHG Dm,<amea> or B (Dm mod 8)

#<d3>,<amea>

BCLR Dm,Dn or #<d5>,Dn L (Dm mod 32)

BCLR Dm,<amea> or B (Dm mod 8)

#<d3>,<amea>

BSET Dm,Dn or #<d5>,Dn L (Dm mod 32)

BSET Dm,<amea> or B (Dm mod 8)

#<d3>,<amea>

BTST Dm,Dn or #<d5>,Dn L (Dm mod 32)

BTST Dm,<amea> or B (Dm mod 8)

#<d3>,<amea>

Binary Coded Decimal Operations

Mnemonic Legal Modes Size

ABCD Dm,Dn or -(Am),-(An) B

NBCD <edea> B

SBCD Dm,Dn or -(Am),-(An) B

PACK Dm,Dn,#<d16> or unsized

-(Am),-(An),#<d16>

UNPK Dm,Dn,#<d16> or unsized

-(Am),-(An),#<d16>

Bit-Field Operations

Mnemonic Legal Modes Size

BFCHG <ea>{offset:width} unsized

BFCLR <ea>{offset:width} unsized

BFEXTS <ea>{offset:width},Dn unsized

BEXTU <ea>{offset:width},Dn unsized

BFFFO <ea>{offset:width},Dn unsized

BFINS Dn,<ea>{offset:width} unsized

BFSET <ea>{offset:width} unsized

BFTST <ea>{offset:width} unsized

Program Control Operations

Mnemonic Legal Modes Size

Conditional

Bcc <label> 16 bit disp

Bcc.s <label> 8 bit disp

DBcc Dm,<label> 16 bit disp

Scc <adea> B

Unconditional

BRA <label> 8 or 16 bit disp

BSR <label> 8 or 16 bit disp

JMP <cea> unsized

JSR <cea> unsized

CALLM #d8,<cea> unsized

Returns

RTD #<d16> disp sign ext 32

RTR nop unsized

RTS nop unsized

RTM Rn unsized

System Control Operations

Mnemonic Legal Modes Size

Privileged

ANDI->SR #<d16>,SR W

EORI->SR #<d16>,SR W

MOVE<-SR SR,<dea> W (68010 only)

MOVE->SR <dea>,SR W

MOVE-USP USP,An or An,USP L

MOVEC Rc,Rn or Rn,Rc L

MOVES Rn,DFC<amea> or L,W,B

SFC<amea>,Rn

ORI->SR #<d16>,SR W

RESET nop unsigned

RTE nop unsigned

STOP #<d16> unsized

TRAP Generators

CHK <dea>,Dn W

CHK2 <cea>,Rn L,W,B

ILLEGAL nop unsized

TRAP #<d4> unsized

TRAPV nop unsized

TRAPcc #<d32> W,L

Condition Code Register

ANDI->CCR #<d8>,CCR B

EORI->CCR #<d8>,CCR B

MOVE<-CCR CCR,<adea> W(lower B only)

MOVE->CCR <dea>,CCR W(lower B only)

MOVE<-SR SR,<adea> W

ORI->CCR #<d8>,CCR B

Miscellaneous

Mnemonic Legal Modes Size

NOP nop unsized

BKPT #,d3> unsized

Coprocessor Operations

Mnemonic Legal Modes Size

cpBcc See Bcc

cpDBcc See DBcc

cpRESTORE <acea> + (An)+ unsized

cpSAVE <acea> + -(An) unsized

cpScc See Scc B

cpTRAPcc See TRAPcc W,L

## 1.9 Programmer notes m68040/060

Programmer notes m68040/060

Instructions hardware supported by the 040 that need to be emulated on the m68060 and could slow down execution when used:

- CMP2
- CHK2
- CAS2
- DIVU.L
- DIVS.L

- MULU.L
- MULS.L
- FBDcc
- FScc
- MOVEP (no support on 060)
- all the m68881/2 specific fp instructions
- and some more less commont instructions

Instructions that are hardware supported by the m68060 but are emulated on the m68040:

- FINT
- FINTRZ
- FTRAPcc

For more detailed info check out the m68060 user's manual.

## 1.10 Commandline.

Asm-Pro Command summary

The command line is the command centre of Asm-Pro,

recognized by the prompt :

>

Next are the commando's which can be used.

Project

ZS - Delete source text

O - Restore deletions of source text

R - Load source code

RB - Load Binair data

RO - Load Object module

W - Save source text

WB - Save Binaire data

WO - Save Object module

WL - Save Linkfile

I - Insert

U - Update file

ZF - Remove file

WP - Save Preferences

=M - Enlarge working memory

!/!! - Quit Asm-Pro

=C - Change colors

Editor

T[line] - Start of source (or line n)

B - End of source

L[Text] - Text search

ZL[line] - Delete line from cursor position

P [line] - Print lines from cursor position

EL - Expand Lables with....

Memory

M[.amount][address] - Memory edit

d[address] - Dissasemble

H[.amount][address] - View memory, hexadecimal

N[address] - View memory, ASCII text

S[.amount] - Search

F[.amount] - Fill

C[.amount] - Copy

Q - Compare

Insert

ID - Insert Disassembled code

IH[.amount] - Insert Hexadecimal code

IN - Insert ASCII code

IB - Insert Binary code

IS - Insert Sinus

Assembler

A - Assemble in Editor

AO - Assemble with Optimalisation

AD - Assemble with debug infromation

=S - View Symbol table

Monitor

J[address] - Jump to subroutine (JSR)

G[address] - Jump inmediately (JMP)

K[step] - One step modus

X[register] - View and/or change register

ZB - Remove Breakpoints

Diskette

RS[drive] - Read Sector

RT[drive] - Read Track

WS[drive] - Write Sector

WT[drive] - Write Track

CC[drive] - Calculate Bootblock-Checksum

W - Load External data

Miscelaneous

CS - Create Sinus in memory

Y [command] - Execute Dos command

V [path] - Show contents of path

> - Revert output to PRT: or Dfx:

?[expression] - Calculate number expression

return - Empties screen

=R - Custom chip information

PS - Parameter Set

## 1.11 Project

PROJECT

ZS - Zap Source

Deletes the program text, the copybuffer (clipboard) and the code.

The program text can be undeleted with the "o" (old source) command but

only if there is no new source text entered in the editor window.

O - Old Source

With this command you can retrieve the source text after a zs command.

R - Read

Reads a source text into the editor. Asm-Pro asumes it is a ASCII text.

If you want to alter the normal .s extension change "PROJECT/PREFERENCES/SOURCES.S".

RB - Read Binary

This command will load binary data into a memory address.

After the file name the start and end address will be asked.

BEG>

END>

BEG (Begin) is the first address that will be filled with data, END (End) the last.

BEG>$70000

END>$71000

Loads the first $1000(=4096 bytes) from a binary file into the memory, starting

at address $70000. To load the complete file leave the END> empty.

Example :

BEG>$70000

END>

RO - Read Object

Reads an executable into memory. It is sometimes possible

to execute it.

Some programms require input from the CLI. To get this right you must

fill in the registers A(pointer to parameters) and D (Length) yourself or use the PS command.

When loading an empty executable (" ") all memory is cleared.

W - Write

Writes the source code to a file. The file is a normal ASCII file so editing with
other text editors is also possible.

Usual the .S extension will be placed behind the name. If you want a different extension
switch of the "PROJECT/PREFERENCES/SOURCE.S". Now you can choose your
own extension with the !.[extension] commando. Example:

>!.asm

WB - Write binary

Writes a binary file to disk. You have to give the begin and end address.

Example:

BEG>$70000

END>$71000

Saves $1000(=4096 bytes) from address $70000 as a binairy file.

WO - Write Object

After assembling it is possible with the WO command to write the object/executable to disk.

Use WL to write a Linkfile.

I - Insert

With this command it is possible to insert a text file into the excisting
source text.

U - Update File

Writes the source text to disk, updating the file. Same as W (write).

ZF - Zap File

Deletes a file from disk.

Zi - Zap Include memory

Deletes include files from memory. The include files are kept in memory
to speed up the assembling proces.

Wp - Write Preferences

Creates or updates the Asm-Pro.Pref file.

=M - Expand Memory

This command makes it possible to expand the working memory
(as long as there is memory available) whithout losing anything.

! - Quit Assembler

Quits Asm-Pro. No changes are saved and memory is erased.

!! - Quit Assembler Fast

Same as Quit Assembler but no requesters are shown and nothing is saved.

=C - Change Colors

To overide the standard preferences colors you can use this command.

A window will appear making it possible to adjust the colors.

## 1.12 Editor

EDITOR

T- / Top of file

Jumps to the line number specified, otherwise jumps to the start of the source.

>T100 : Jumps to line 100

>T : Jumps to the start of the source

>t-1 : Jumps to the end of the source

L- Locate text

Searches for the string specified.

>Lmove : Searches for all move operands in the source.

To search again simply type : >L

NOTE : Between the L commando and the string cannot be a space.

ZL- Zap Lines

Deletes all lines from actual cursor position.

>ZL100 : deletes 100 lines from actual cursor position.

>ZL-1 : deletes all lines from actual cursor position.

P- Print Lines

Prints from cursor position the amount of lines specified.

To print the lines select "Project/Preferences/Printerdump" or press
Ctrl-P.

>P100 : Prints 100 lines from cursor position

>P-1 : Prints all lines from cursor position

EL- Extend Lables

If more than one programmer is working on the same source, there is a change
that lable names are used more than once. To avoid this the EL commando
is usefull

Asm-Pro will ask with what you want to extend the lables:

Extend Lables with>

## 1.13 Memory

MEMORY

M- Memory Edit

Gives direct edit possibility in memory. You can enter text or hexadecimal values.

To get word or longword mem dumps use the extensions .w or .l (or.b) like this:

```
>m.w$1200
00001200 FF60 F22E F0C0 FFDC 2D6E
```

D,@D- DisAssemble

Goes to the assemble modus, the memory is disassembled. You can scroll, jump and

edit. To undo changes press Escape (see 5.6 and 9)

With @D you only get 12 lines on your screen, so half-screen.

H,@H- HexDump

Goes to the hexdump modus. Again you can scroll, jump and

edit. If you want to edit longwords the H.L commando is more interesting.

With this you can edit complete longwords instead of just bytes. (see 5.6 and 9)

With @H you only get 12 lines on your screen, so half-screen.

See Memory Edit for output formats.

N@N - AsciiDump

Goes to the ASCII modus. Again you can scroll, jump and edit. The view

is in rows of 64 letters and again you only get 12 lines on your screen, so half-screen.

@A - Assemble mem

Assembles directly into memory.

@B - BinaryDump

Displayed is a view with 8 lines of binary data.

Also see Memory Edit for output formats.

S - Search in memory

Example:

>S

Beg>$10000

END>$20000

After entering the start and end address you can enter the data you want to search

DATA>123 /or 4321.I /or "hello" /or $4532.w /or %101001.b

Default search size is byte.

F - Fill Memory

Be carefull with this commando. If you fill the wrong part of the memory

you can crash the program or even your computer.

C- Copy Memory

Copies a part of the memory to an other part of the memory.

Q- Compare Memory

Compares two parts of the memory. When not equal the address of the first not

equal byte is returned.

## 1.14 Insert

INSERT

ID - Insert DisAssemble

This is a very powerful commando. It gives you the ability to disassemble memory and place it directly into the source code.

The new code will be supplied with lables (if possible), so it is ready for use.

The one disadvantage is that the commando is rather slow. For the real

disassemble work you can better use Resource or some other proper program.

IH - Insert Hexdump

Gives you the possibility to add hexadecimale text into your source code.

IN- Insert ASCII

Gives you the possibility to add text as DC.B strings into your source code.

It wil automatically use hexadecimales if not defined as ASCII.

IB - Insert Binary

Gives you the possibility to add binary as DC.B into your source code.

IS - Insert Sinus

Gives you the possibility to add a Sinus into your source code.

See "Rest : CS"

## 1.15 Assembler

ASSEMBLER

A- Assemble

This will start the assembling, same as Amiga-Shift-A.

@A - Assemble to memory

See paragraph 5.2 "memory"

AO - Assemble optimized

Normally assembles the source but will try to optimize the branch distance to

the shorter .S version. When the branch is more than -128 or +128 away,

the branch is changed into a .W version.

AD - Assemble debug

Assembles the source and then jumps to the debugger.

=S - Symbol table print

Shows all global variables after assembling. These variables can also

be printed, set "PROJECT/PREFERENCES/PRINTER DUMP" and select Listfile.

## 1.16 Monitor

MONITOR

J- Jump to address

Jumps to the specified address like a subroutine (JSR).

When no address is specified the jump will be to the start of the source.

Example :

>JStart

Jumps to the label "Start".

G- Go to address

Same as the J commando with the addition that it is possible to set a

breakpoint or stop at an illegal commando.

K- Single step, n steps

Steps for n steps through the source from actual program counter.

See Debugger for more information.

X- Change - edit registers

Can be used in two ways:

>X : view all registers

>XD2 : view one register (expl D2)

Just typing X will show all registers, including USP, SSP, SR and PC.

Flags of the statusregister will be shown as characters.

All changes in the registers since last view will be shown invers colored.

## 1.17 Diskette

DISKETTE

RS - Read Sector

Read sectors from a disk. An Amiga disk has 80 tracks on each side which are

seperated into 11 sectors. This makes 80*2*11 = 1760 sectors, each sector is

512 bytes large. The bootblock consist of 2 sectors, starting at sector 0.

The naame of the disk is in the rootblock which can be found on track 80.

Example :

To read the bootblock of DF1: into address $70000

>RS1 : diskdrive 1

RAM PTR>$70000 : Pointer to RAM-address

DISK PTR>0 : Sector number

LENGTH>2 : Amount of sectors to read

RT - Read Track

See RS

WS - Write Sector

See RS

WT - Write Track

See RS

CC- Calculate checksum

The first thing loaded by the system is the bootblock.

With this information it determinse what kind of disk it is, KICK=Kickstart, DOS=dos, BAD=unknow format.

The bootblock can also contain a executable. To check if the data is usable,

the systems checks the chekcksum. With CC this checksum can be recalculated.

This can be useful when programming your own bootblock or in case of a virus.

Example:

>CC1

Will calculate the checksum on the bootblock in DF1:

When a RS1 or RT1 commando is issued, the checksum will automatic

be calculated.

E - Extern files load

Data which is pointed at whith the external directieve, will be loaded

when the E commando is given. Specify a numner if you do not want to

load all external data. For more information see Chapter 7.

## 1.18 Miscelaneous

MISCELANEOUS

CS- Create sinus in memory

This commando creates a Sinus.

Example :

DEST> : Place in memory where sinus must be placed

BEG> : Starting Angle

END> : Ending Angle

AMOUNT> : number of entries in sinus table

AMPLITUDE> : Highest/Lowest value of the sinus

YOFFSET> : The middle of the sinus

SIZE> : B(ytes)/ W(ords) / L(ongwords)

MULTIPLIER> : Afterwards number multiplier

HCORRECTION> : Halve step correction

RCORRECTIE> : ROUND in stead of INT (0,7=1 in stead of 0)

Y - Execute dos commando

With this commando you can start a Dos program while working in Asm-Pro

Example :

>Ydiskmaster

V - view directory

Shows the contents of the path specified. Necessary when the Req.Library

is shut down.

Example

>VDF0:

Shows the contents of DF0:

Free space is in bytes.

> - Specify output

Reverts the output of a program to disk or printer. If the output is to

disk the file will get the .TXT extension. To stop the output repeat the

commando without a path.

? - Calculate value

Usage of all operators and defined lables is possible. The result is

shown in hexadecimal, decimal, ASCII and binary.

R - register information

Has the following options :

>=R<002> value search with address and register.

>=R DMACONR name, search with the original name.

>=R gives a list of all registers

PS - Parameter Set

Sets the parameters used in programs which have a CLI input.

Asm-Pro calculates the length and puts it into the correct register

## 1.19 Special features.

Special features.

IncIFF(p) Add iff pictures to your source

IncLINK Add LinkObject files to your source

%Getdate/%Gettime

PCR in Debug window

Don't know where to place this info so here it is:

I also implemented a custom scroll routine (using the cpu) to speed up scrolling in the editor. This is only usefull when you don't have a gfx-card as it is NOT system friendly..

The custom scroll routine ONLY works if you have a screen that is 640 wide.

It works great on a MULTISCAN:Productivity screen and scrolling is almost two times faster (on my 040) than with system routines.

This option can be activated in the Env Prefs window ("Custom Scrollr.").

If you have problems using the editor on your config, like key repeat problems etc. (if you do you'll know what I meen), try switching on the "scroll sync" option in the Env Prefs window. If this doesn't help please report back to me.

## 1.20 Special features - PCR in Debug window.

Special features - PCR in Debug window.

When you have a mc68060 the register window in the debugger will show the Processor Configuration Register (PCR).

The PCR is an 32-bit register which controls the operation of the mc68060 internal pipelines and contains a software readable revision number.

Bits 31-16 Identification:

These bits are configured with the value which id’s this device as an mc68060. These bits are ignored when writing to the PCR.

Bits 15-8 Revision Number:

Contains the 8-bit device revision number. The first revision is 00000000. these bits are ignored when writing to the PCR.

Bit 7 (EDEBUG) Enable debug features:

When this bit is set, the mc68060 outputs internal control infoirmation on the address bus and data busduring idle bus cycles. This bit is cleared at reset.

Bits 6-2 Reserved.

Bit 1 (DFP) Disable Floating-Point Unit:

When this bit is set the on-chip FPU is disabled and any attempt to execute a floatingpoint instruction generates a line F emulation. This bit is cleared at reset.

Bit 0 (ESS) Enable Superscalar Dispatch:

When this bit is set, the ability of the mc68060 to execute multiple instructions per machine cycle is enabled. This bit is cleared at reset.

For more info see the M68060 Microprocessors User’s Manual from motorola.

## 1.21 Env Preferences.

Asm-Pro Prefs summary

Environment Assembler SyntColors

Env Prefs

General Parameters

ReqTools Library

Activates the reqtools requesters

Save Marks

Save source marks at start of source

Source .ASM

Filter files from requester when active.

Filter is taken from "Source Extension".

Update Check

When checked source will only be saved when it was changed.

Printer Dump

Dump output to theprinter.

WB to front

Pops the workbench to front executing your program with ’j’

Resident Registers

If active the regsdata file will be resident in memory.

Safety

When active, there will be no default (enter) action on requesters.

Close Workbench

Close wb to save memory

Parameters

When active you can run your program like it was started from cli.

Set the parameters for it with 'PS'

Show Startup win.

When active it will show the startup window when starting Asm-Pro.

Monitor / Debugger

ASCII Only

Masks non printable characters in the hexdump with a '.'

Disassembly

Adds a disassembled version of the currect line at the bottom of the debugger

Show Source

Show source in the debugger

Enable/Permit

When active an Enable/Permit before/after executing your program

Libcalls dec

show library calls in decimal.

Realtime deb

activates the realtime debugger

Editor

Line Numbers

Shows line numbers in the source

CTRL up/down

Activates CTRL up/down :)

Keep x

Keeps x in the source.

Auto Indent

Well activates Auto Indent.

Ext. ReqTools

Activates the Search and replace reqtools requesters

otherwise this will be handled in the titlebar.

Syntax Coloring

Activates source coloring..

Custom Scrollr.

Activates custom scroll routine that uses the cpu

directly to copy display data. NOT a good idea when

you have a gfx-card.

(See "Special features" for more info)

Scroll Sync

When active, scrolling goes in sync with refresh rate.

Rest

Default Dir:

Default startup directory.

BootUp

Line of commands that are executed at startup of Asm-pro.

a <cr> is a '\'

eg.

"r coolsource.s\a\"

will read coolsource from the current dir and assamble it.

Source Extension

Set the source extension like: #?(.s|.asm|.i)

Select editor font

Set a font for Asm-pro.

Select screen mode

Choose a screenmode for Asm-Pro

## 1.22 Asm Preferences.

Asm-Pro Prefs summary

Environment Assembler SyntColors

Asm Prefs

Rescue

When active sources can be retrieved after a crash

with the 'O' command.

Level 7

When active and you have a level 7 switch, wait forever

loops can be interrupted and you return to Asm-Pro.

NumLock

Activates Numlock

Auto Alloc

Allocates the correct type of mem for sections.

Debug

If active source will not be reassembled when nothing

was changed.

List File

Lists the source file

Paging

Starts at top of screen when it is full.

Halt File

Halts when page is full (with paging).

All Errors

Shows all errors

Progress Indicator

Show progress of the assembler when assembling.

Progress by Line

Progress by line iso percent.

DS Clear

Clears DS datasections for you. Remember it is only

cleared inside Asm-Pro. Your stand alone program will

not have its ds cleared.

Label:

Please switch this OFF. Labels with ':' are always supported.

(might be usefull for some very old sources.)

UCase = LCase

Well uppercase is lowercase...

; Comment

When active comments must be preceded with ; or *

Processor Warn

When active Asm-pro gives warnings when instruction is

not for the processor set in "CPU".

FPU Present

Set is FPU is present.

68020++ Odd data

Will allow odd aligned data.

68851 Present

Set if MMU is present.

CPU:

Set the cpu type you need here (68000..68060).

## 1.23 Synt Preferences.

Asm-Pro Prefs summary

Environment Assembler SyntColors

Synt Prefs

Level

Not yet implemented.

Syntax Coloring

Set your prefered colors here..

## 1.24 Special features - %Getdate/%Gettime.

Special features - %Getdate/%Gettime.

Implemented %getdate and %gettime these commands will generate the

current date or timestring. this is usefull for a version string or about

window.

%getdate [dateformat]

%gettime

[dateformat] is optional and, when not specified, will default to the dos dateformat (FORMAT_DOS).

Dateformat accepts all dateformats as listed in dos/datetime.i and shown here:

FORMAT_DOS equ 0 ; dd-mmm-yy

FORMAT_INT equ 1 ; yy-mm-dd

FORMAT_USA equ 2 ; mm-dd-yy

FORMAT_CDN equ 3 ; <see below>

FORMAT_MAX equ FORMAT_CDN

There is one exception to the include and that is the FORMAT_CDN (or 3).

If you use FORMAT_CDN it wil produce a datestring like "dd.mm.yy" and this is very usefull for version strings: $VER <name> <version>.<revision> (dd.mm.yy)

It works like this:

versionstring:

dc.b "$VER: Asm-Pro v1.11a ("

%getdate FORMAT_CDN

dc.b ") By Solo/Genetic.",0

This will generate the following version string:

dc.b "$VER: Asm-Pro V1.11a (03.06.98) By Solo/Genetic.",0

OR:

About:

dc.b 'Assemble date: '

%getdate

dc.b 0

Will produce a string like this:

dc.b "Assemble date: 03-Jun-98",0

Same goes for %gettime (ex. "21:06:09").

## 1.25 Special features - IncIFF(p).

Special features - IncIFF(p).

==========================================================================

Example source IncIFF(p).

- INCIFF

Include iff picture as raw-blit or raw normal data with or without colormap

Syntax:

Picture: INCIFF {filename}[,conversion mode[,Cmap placing[,Cmap mode]]]

{filename} = The name of the IFF ILBM file to include.

{conversion mode} = "RN" or "RB" for RAW-NORMAL or RAW-BLIT format

{Cmap placing} = "A","B","N" for Cmap After,Before or None

{Cmap mode} = "ECS","AGA" for word or longword Cmap.

If no options are given Asm-Pro will convert to defaults : RAW-BLIT

No Cmap

Word Cmap

The Cmap in ECS mode will be decoded to WORDS.

The Cmap in AGA mode will be decoded to LONGWORDS.

ECS Cmap Example:

DC.W $0000,$0F00,$00F0,$000F ; COLOR00 - COLOR03

AGA Cmap Example:

DC.W $0000,$0000 ; COLOR00 High-RGB,Low-RGB

DC.W $0F00,$0F00 ; COLOR01 High-RGB,Low-RGB

DC.W $00F0,$00F0 ; COLOR02 High-RGB,Low-RGB

DC.W $000F,$000F ; COLOR03 High-RGB,Low-RGB

==========================================================================

- INCIFFP ((copper)palet picture)

Include pallet from iff picture in your program.

Syntax:

CopperColor: INCIFFP {filename}[,Coppermode[,Bankoffset][,coloroffset]

colorlist: INCIFFP {filename}[,list type]

{filename} = The name of the IFF file containing the CMAP hunk.

{copper mode} = "CE" or "CA" for ECS or AGA copper list

Default is "CE"

{Bankoffset} = [0-7] offset for start bank. (CA only)

Default is "0"

{coloroffset} = [$180-$1be] color offset inside a bank

Default is "$180"

{list type} = "12" or "24" for 12 bits or 24 bits color list

Default is "12"

If no options are given Asm-Pro will convert to default : colorlist 12bit

colorlist:

a inciffp adventure/pics/hall.ehb

b inciffp adventure/pics/hall.ehb,12

```
c inciffp adventure/pics/hall.ehb,24
coppercolors:
d inciffp adventure/pics/hall.ehb,CE
e inciffp adventure/pics/hall.ehb,CE,$190
f inciffp adventure/pics/hall.ehb,CA
g inciffp adventure/pics/hall.ehb,CA,$180
h inciffp adventure/pics/hall.ehb,CA,2
i inciffp adventure/pics/hall.ehb,CA,2,$180
12BitsColorlist:
dc.w $0000
dc.w $0F00
dc.w $00F0
dc.w $000F
24BitsColorlist:
dc.l $00000000
dc.l $00FF0000
dc.l $0000FF00
dc.l $000000FF
ECScoppercolorlist:
dc.w $0180,$0000
dc.w $0182,$0f00
dc.w $0184,$00f0
dc.w $0186,$000f
AGAcoppercolorlist:
dc.w $0106,$0C40
dc.w $0180,$0000,$0182,$0EEE
dc.w $0184,$0100,$0186,$0110
dc.w $0188,$0111,$018A,$0220
dc.w $018C,$0221,$018E,$0320
dc.w $0106,$2C40
dc.w $0180,$0981,$0182,$0777
dc.w $0184,$0982,$0186,$0984
dc.w $0188,$0A91,$018A,$0995
dc.w $018C,$0996,$018E,$0A92
```

## 1.26 Special features - IncLINK.

Special features - IncLINK.

===============================================================================

- With Inclink you can add linkobjects to the current section.

Like this:

test:

inclink "ram:testobj.o"

It reads a linkobject file, Relocs it and gets the symbols. Inclink can

only handle 1 section per file and it is put in current section.

These Commands Can be used to make definitions

- xref (external reference) a label you want from the main file
- xdef
- globl
- global
- extern (external definition) a label you will send to main file

Use the command write-link (WL) to save a linkfile..

E.g.

Try this little source and write it as an linkobject (with WL)

xdef start

one:

nop

nop

rts

now you can include it like this:

two:

inclink work3:test/link.o

and the symboltable (=S) will contain both the one and two labels :)) the link

object will be placed in the current section and thus only one section is

allowed..

## 1.27 Thanks

Thanks to

Special thanks go to the Beta testers for their constructive comments, bug reports and ideas.

- One/Genetic MC68060/50 MPC604/200 64MB CyberVision 64
- Tib/TFA MC68030/50 8MB
- P-O Yliniemi MC68060/50 MPC604/200 134MB CyberVision 64
- Thomas Wittwer MC68030/50/fpu 32MB

- Peter’ViTAL’Eriksson MC68060/50 36MB CyberVision 64

- Scorpion/Silicon MC68030/50 8MB

- Cliff Earl MC68000

(yes configurations may have changed.. but this is the hardware it was tested on :)

And ofcoz to all people sending me emails after Asm-Pro’s public release telling me they love it :).. Please continue to do so your opinion means a lot to me...

List of people that sent me (bug)reports:

Antibyte/Scoopex

booster.cromatics+loonies

Boushh/TFA

Christian Holmqvist

Cliff Earl

Emiliano Esposito

Erhan Bilgili

Erik Nordby

Frank (Copper) Pagels / Defect Softworks

Hitman/Code HQ

Joop de Jong

Jouni Korhonen

kERNAl/FAITH/LFC

Lay András

Mariusz Zielinski / Brainiax of Virtual Design

MaQ . floppy

Matteo Pedone

Maurice Maissan

Michael Knoke/Aliendesign

Mikkel Skovgaard

Morgan Johansson

Nicholas Clarke

P-O Yliniemi

Per Johansson

Peter Eriksson (ViTAl/MKS)

Rene Wunderlich

Rene W. Olsen / AC of Rebels

Roman Fierfas

Rune Brekke Stenslan (Spˆ Ctz)

Scorpion/Silicon

Sebastian Giese

Steen Lund Nielsen

Stefan Keilwert

Sven Thoennissen

The Welder / Faith & Darkage

Thomas Wittwer

Tomek 'VePaR' Malerczyk

Trevor Mensah (2-Cool LSd)

And some more but I lost a lot of emails.. send me another bugreport and you'll get in the list :)

## 1.28 Disclaimer...

Disclaimer

No guarantee of any kind is given that Asm-Pro is 100% reliable. You are using it at your own risk. The author takes no responsibility for any damage which is caused by using this program.

## 1.29 Frequently Asked Questions.

Frequently Asked Questions.

Q: Why is the editor so slow (on AGA)?

A: You should reduce the screensize and/or use less bitplanes for the Asm-Pro screen. You can also speed up the editor by turning off the syntax coloring.

If you have ECS/AGA you could turn on the custom scroll routine for more speed. This will NOT work on any gfx-card.

Q: Why does the instruction xxx not work or crash my Amiga?

A: Send me a bug report :)

Q: How come the colors prefs window shows 256 colors and the max should be 16?

A: This should only happen on gfx boards. Try selecting a 8-bit screenmode and it will work ok. Reqtools doesn't seem to like screensmodes with more than 256 colors.

Q: When I try to write the executable (WO) of my very big assembled source with lots of labels, Asm-Pro crashes?!

A: Raising the stacksize for Asm-Pro will fix this problem. The default stacksize is 20kb and is not enough for big sources with lots of labels. I use 60kb and it works fine for my 900kb source.

## 1.30 ....

Leeg.