HowToCode7

COLLABORATORS

	TITLE : HowToCode7		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		March 28, 2025	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

HowToCode7

1.1 HowToCode: Assembler

```
                Choosing and Using an Assembler
                -------------------------------

   1. Avoid K-Seka!
   2. Problems with Devpac?
   3. Problems with ArgAsm?
   4. Assembling within Cygnus Ed
```

1.2 Don’t use the K-Seka assembler!

It’s dead and buried. Get a real assembler. Hisoft Devpac is probably
the best all-round assembler, although I use ArgAsm which is
astonishingly fast. The same goes for hacked versions of Seka.

Is it any coincidence that almost every piece of really bad
code I see is written with Seka? No, I don’t think so :-)

When buying an assembler check the following:

1. That it handles standard CBM style include files without
alteration.

2. That it allows multiple sections

3. That it can create both executable and linkable code

4. 68020+ support (especially if you want to program for A1200,
writing A1200 code without 68020 instructions is real stupid!)

Devpac 3.0 is probably the best all-round assembler at the moment.
People on a tighter budget could do worse than look at the
public domain A68K (It’s much better than Seka!). I’d suggest
using Cygnus Ed as your Text Editor, or Turbo Text.

1.3 Devpac optimise mode produces crap code?

If you're using Devpac 2.x and have found that the OPT o+ flag produces crap code, then you need to add the option o3-. This is indeed fixed in >=V3.02

Under certain circumstances the optimiser will optimise relocate references in the first 64Kb of your code to word addressing, which obviously isn't very good if AmigaDOS then loads your code above 65535, which is quite likely!

o3- disables short-word address optimising.

This isn't necessary if you're creating linkable code (with l+), and indeed may be fixed in current versions of Devpac 3

My current option setup for Devpac is:

```
   opt   l+,o+,ow+,ow1-,ow2-,ow6-,d+,CHKIMM
```

l+ sets linkable code on (as I mix C and Assember in my current projects)

o+ enables optimise mode.

ow+ enables optimiser warnings (they act as errors with SLINK, so I edit my source when I get an optimiser warning)

ow1- disables warnings on short backwards branch optimising

ow2- disables warnings on address register indirect with displacement zero to address register indirect optimising, again I don't want to edit my code if I have (for example)

```
         move.l   vs_vscreen1b(a0),a1  ; vs_vscreen1b = 0
```

ow6- disables warnings if short branches forwards can be made

d+ debug information on

CHKIMM - Check Immediate values. This will report an error if any immediate addresses are used (the most common mistake in assembler is to leave the # from a value). Address 4 (EXECBASE) is allowed, and other fixed addesses (eg CUSTOM - $dff000) are allowed as long as you add a .L to the end.

```
         add.l 123,d0         ; This now gives an error!
         LEA   (CUSTOM).L,a0  ; This doesn't.
```

1.4 Argasm produces crap code, whatever happens

First, Argasm (unlike Devpac) from the Command Line or if called from Arexx using Cygnus Ed (my prefered system) defaults to writing linkable code, so if you want executable files you need to add

```
        opt l-    (disable linkable code)
```

If you find that your Argasm executables fail then check you haven't
got any BSR's across sections! Argasm seems to allow this, but of
course the code doesn't work. Jez 'Polygon' San from Argonaut software who
published ArgAsm says it's not a bug, but a feature of the linker...

Yeah right Jez...

But Argasm is *fast*, and it produces non-working code
faster than any other assembler :-)

Argonaut have abandoned ArgAsm so the last version (1.09d) is the
last. There will be no more, and it doesn't support 68020+
instructions, so I've stopped using it now.

1.5 Using Arexx with Cygnus Ed

Cygnus Ed is a wonderful system: Included in the the utils/ced
directory are a few macros I wrote that may help you configure
your assembler to run in CED.

To install:

1. Make sure Rexxmast is running and you have copied the macros
to your REXX: directory (make a sys:rexx directory rather than
assigning REXX: to S:)

2. Add the line

```
ced -r
```

to your s:user-startup. Copy the CygnusEd Activator (on Cygnus
Ed V2.1x distribution disk) as C:ed, yes, that's right. Right
over the top of the abysmal Commodore editor!! You lose 200Kb of
fastram doing this, but believe me, it's worth it. Whenever you
need to use Cygnus Ed, either type ed filename and it loads
in a flash, or just press Right-ALT/Right-Shift/Return to open
a new CED session.

The CygnusEd Activator is public domain, and is in the utils
directory of this archive.

3. Install the commands on the keys you want to use (under the
special menu).

I currently have mine set up:

```
F1 -  devpac.ced     - This calls Devpac to assemble the current file.
                        Output is file ram:test, errors to ram:errors

F2 -  argasm.ced     - Same, but for Argasm


F3 -  errors.ced     - Open and close the error window. The error
```

```
                         window is not editable.

F10 - ram:Test        - Execute the code!


Other keys are free for C, TeX or whatever else you want to use...
```