sss v1.0 Documentation

COLLABORATORS			
	TITLE : sss v1.0 Documentation		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		January 13, 2023	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

sss v1.0 Documentation

1.1 sss v1.0 doc (26.10.1998)

```
                                sss v1.0  (26.10.1998)

          © 1998 Simone Bevilacqua


DISCLAIMER & Distribution
some legal stuff

Requirements & Installation
what you need to get it working


Introduction
why it exists and what it's for...

Usage
how to use it

About the ESA sources
notes on the source code


History
what has happened till now

Author
some notes about me...

Greetz & Thanx
ciao!
```

1.2 DISCLAIMER and Distribution

```
                DISCLAIMER
```

```
 ************************************************************************
 * THIS PROGRAM IS PROVIDED "AS-IS" WITHOUT WARRANTY OF ANY KIND      *
 * EITHER EXPRESSED OR IMPLIED.                                       *
 *
                I
                 ACCEPT NO RESPONSABILITY OR LIABILITY FOR ANY DAMAGE OR PROBLEM *
 * DERIVING FROM THE USE OF THIS PROGRAM:  USE AT YOUR OWN RISK!!!    *
 ************************************************************************


Distribution

 This program is FREEWARE, therefore IT CANNOT BE SOLD FOR PROFIT.
 So, only the distribution charges (i.e.: disk, postage, handling, etc.)
 can be applied.

 No fee is required from
                me
                , but donations of any kind (something like
 the 1st original tankobon of "Dr.Slump & Arale chan" would be just a
 dream... ;) will be gladly accepted.

 If distributed on a coverdisk, please send a copy of the mag!!!

 ALL the following files *MUST* be included in the same package (regard-
 less of the form it comes in):

  sss/                 (progdir)

   sss                 executable
   sss.guide           this documentation
   do                 script for quick compiling
   defs.i
                standard asm source
                   main.esa
                ESA source code
                   misc.ei
                ESA include file
                   opt.ei
                ESA include file
                   split.ei
                ESA include file
                   data.i
                standard asm source
                 You can freely modify the program for your personal use, but  ↩
                    before
 spreading a modified version, please contact me.
```

1.3 Requirements & Installation

```
Requirements

 sss requires a 020+ CPU.
 It should work also on KS1.3, but I can give no guarantee.
```

```
Installation

 It doesn't need to be installed, just put it anywhere on your HD
 (preferably on your commands path).
```

1.4 Introduction

```
                Introduction

 This prog has been born mainly because ESA (Extended Syntax Assembly -
 you can find it in "dev/asm" on Aminet) lacked of a good, full working
 example: that's why you find also the
                source code
                 in this archive.
 A secondary reason is that I needed a simple file splitter and I could
 not bother finding the right one among the incredible amount available
 on Aminet.
 A third reason is that I wanted to replace (on Aminet) an old program
 of mine which - I fear - could fail to work well (I coded it when I was
 getting started with asm and patched it later, but...).

 OK, the preamble's over, and from the 2nd point you should have guessed
 what sss is: a simple file splitter.
 Being simple doesn't mean it is not smart, though...

 sss, in fact, will try to allocate a buffer as big as the chunk size to
 reduce the accesses to disk; should it fail, it will try to allocate
 the largest available contiguous block of memory.

 The output files produced will have a numerical extension in ascending
 order.
 Only the needed number of digits will be used: for instance, if there
 are only 9 chunks, the extension will be exactly 1 char long.
 If, instead, 9 < chunks < 100, then a 2 digits extension will be used.
 3 for chunks > 99, 4 for chunks > 999 and so on...

 Oh... not much more to say, why don't have a look at the
                usage
                ?
```

1.5 Usage

```
Usage

  SYNTAX

    sss [-q] InputFile ChunkSize [OutBase]

  ARGS

    -q        = quiet mode: don't print any message
```

```
    InputFile = name of the file to split
    ChunkSize = size in bytes of each chunk
                (0 < ChunkSize < $7fffffff = 2,147,483,647)
    OutBase   = output files will be called OutBase.x
                (000 <= x <= 99,999; by default OutBase = InputFile)
```

1.6 About the ESA sources

```
                Preliminary Info

 ************************************************************************
 *If you have already had a look at the sources and you never heard of *
 *ESA before, you're now surely wondering what language they're written*
 *in. The shortest answer I can give to you is: it's a mix of 68k asm  *
 *and higher level constructions. If you're interested and want to know*
 *more, I advice to download "ESA.lha" from dev/asm on Aminet (the ar- *
 *chive is really short).                                              *
 ************************************************************************


Tabulation

 All the sources have been written using this tabulation:

Label           instruction  operands                comment

T               T            T                       T

 That's why they look jerky on the window of the amigaguide viewer you
 are currently using, if you click on the gadgets
                here
                 or in this sec-
 tion.


Compiling and Assembling

 They are fully commented (even if not in the best way possible, I have
 to admit) and need no other external include file to be compiled and
 assembled.
 You can do this in two ways:
 - calling ESA to compile the file main.esa and then assembling the
   output asm source with your fave assembler
 - using the script do (please note that you could have to change so-
   me of the variables therein)
 The final source code is assembled correctly by phxass, maybe you could
 have to do minor changes to fit your assembler.


Function/Procedures Header Description

 Each subroutine has an header of the kind:

*************************************************************************
* RoutineName v a.c.r
```

```
*****************************************************************************
* INFO          ...
* SYNOPSIS      ...
*               ...
* IN            ...
* OUT           ...
* MODIFIES      ...
* REQUIRES      ...
* NOTE          ...
*****************************************************************************
```

- "RoutineName" indicates the name of the procedure/function

- "v a.c.r" indicates the version according to this versioning system:
  a=algorithm version (this changes when an algorithm replaces the previous, structurally different, one)
  c=compatibility (this increases when the modifications made to the source force changes also to the sources which call the func/proc; in practice when the proc/func can't be used [safely] in the same way/place as before the changes)
  r=revision (increased when bugfixes, optimizations, or any other operation which doesn't affect compatibility or doesn't represent a major change in the algorithm structure are made)

- "INFO" gives a brief description of what the routine does

- "SYNOPSIS" shows how to make the call.
  For example:
  OutValue = RoutineName[arg0,arg1]
  d0 a0 d1
  Means that "RoutineName" is a function which requires two parameters ("arg0" and "arg1") as input and returns a value, indicated as "OutValue" in d0

- "IN" is a list of the input arguments, describing their meaning and the expected/allowed values

- "OUT" gives a description of the value returned by a function

- "MODIFIES" lists all the variables/locations which are modified inside the routine and are not restored on exit

- "REQUIRES" here are listed all the variables,definitions, etc. needed by the routine to be compiled or work correctly

- "NOTE" additional info regarding important aspects of the routine

Any of these fields can be omitted.

Final Note

This sources are meant to be good examples, so no particular optimization has been done.
Moreover, I'm not much used to ESA yet, so they probably are not "the

```
 best" examples possible.
```

1.7 History

```
History

 v1.0 (26.10.1998)

  First public release
```

1.8 Hi there!

```
Hi there!

 If you have any problems or want to know more about ESA write to:

  bevilacq@cli.di.unipi.it

 I can also be reached by snail mail at the following addresses:

  (during "normal" periods)

  Simone Bevilacqua
  P.za  Garibaldi 9
  56100 Pisa (PI)
  ITALY

  (during uni vacation periods - "safer" address!!!)

  Simone Bevilacqua
  Via A.Volta 6
  86010 Ferrazzano (CB)
  ITALY
```

1.9 Greetz and Thanx

```
Greetz and Thanx

 Thanks to all the Amiga coders still around.

 Mega greetings to my family and all my friends!!!

 (the next time I'd like to scribble here something like:
 "WOOOOOOOOOOOoooooooooOOOOOOOOOOOOW!FINALLYMYPATIENCEHASBEENREWARDED!!!
 THANXALOTAMIGAINC./INT.FORHAVINGGIVENTHEWORLDANEWBEAUTYANDAHOPETOSAVEIT
 FROMTHOSEDIRTYM$OFT/INTEL'SHANDS!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!"

 Now I wonder: will that moment ever come before 2177a.d.?!?
 PLEASE, MAKE IT SO, MAKE IT SOON!!!)
```