ARM5194

EXTERNAL REFERENCE SPECIFICATION

for

SES MOTOROLA 68000 UTILITIES

Submitted: _____

Approved: _____

_____

_____

DISCLAIMER:

This document is an internal working paper only. It is subject to change, and does not necessarily represent any official intent on the part of CDC.

REVISION DEFINITION SHEET

REV	DATE	DESCRIPTION
A	09/30/82	Original ERS
B	05/10/84	New procedure descriptions added for GENGS68K, BLDMI68K, PURSYM68K, and TRANDILIB. Additional capabilities for local symbols have been added to REFORM68K.

1.0 PREFACE

1.0 PREFACE

1.1 INTRODUCTION

The Motorola 68000 Utilities are a collection of tools to help the software development process on the CYBER 170. TRAN68K generates absolute records for use by the MC68000 Simulator or anything that can read Motorola 'S' records. SIM68K runs the MC68000 Simulator. BIND68K will take a collection of modules generated by the CYBIL-CM compiler or the MC68000 assembler and create a new module from them. REFORM68K re-formats a module in CDC object text to a module in H-P relocatable object text and/or H-P local symbols records. GENGS68K generates an H-P linker symbols file. BLDMI68K builds an absolute memory image module that can be put on an object file or object library. PURSYM68K deletes debug symbol tables generated by the CYBIL and the MC68000 assembler from all the modules on all the files used as input. TRANDILIB translates an object file or object library into a packed file to be down loaded for a DI box to load into memory. The MC68000 Utilities run as stand-alone commands via the SES processor.

1.2 APPLICABLE DOCUMENTS

The following is a list of documents that either are referenced in this specification or are recommended to aid in understanding this specification.

1) SES User's Handbook (ARH1833)
2) SES Object Code Utilities (ARH2922)
3) SES MC68000 Absolute Linker (ARH4895)

2.0 MOTOROLA 68000 UTILITIES COMMANDS

The Motorola 68000 Utilities use System Control Language (SCL) syntax as the parameter interface. In the descriptions that follow, optional parameters are enclosed in brackets. All parameters of type 'name' can have a maximum length of 31 characters, except for parameters designated as file names (NOS file names cannot exceed 7 characters in length).

2.1 TRAN68K - GENERATE S RECORDS

TRAN68K generates a file of Motorola S records from the output files of the MC68000 Absolute Linker.

```
tran68k hdr=<filename>
        srec=<filename>
        [un=<username>]
```

hdr ! h :
Name of the header file from the Absolute Linker. The file name has the form xxxxHDR from the linker where xxxx is the name seed given by the linker. This file contains information about the other files needed to create the S records.

srec ! s :
Name of file to receive the S records.

un :
User number of catalog to search for hdr file and the other files needed. The default is the current user.

2.0 MOTOROLA 68000 UTILITIES COMMANDS
2.1 TRAN68K - GENERATE S RECORDS

Example

```
ses.tran68k segmhdr srecfil
REVERT.     END TRAN68K
```

This example shows TRAN68K creating srecfil from files that are specified by segmhdr in the current user's catalog.

2.2 SIM68K - RUN THE MOTOROLA 68000 SIMULATOR

SIM68K runs the MC68000 simulator and acquires the 'S' record file that has the absoulte memory image. Commands for the simulator can be entered at the terminal or entered through a command file. The output is either written at the terminal or to a specified file.

NOTE: Documentation for the simulator and its commands are in the M68000 Simulator Reference Manual, M68KOSIM(D2) from Motorola.

```
sim68k srec=<filename>
       [cf=<filename>]
       [output=<filename>]
       [un=<username>]
```

srec ! s :
Name of file that contains 'S' records, the absolute memory image to be loaded by the simulator.

cf :
Name of file that contains commands to be executed immediately upon ectivation of the simulator. The commend file consists of one or more valid simulator commands. If you do not specify this parameter, SIM68K takes commands from the input file assigned to your terminal.

output or o :
Name of file to receive the output from the simulator. If not specified, SIM68K writes to OUTPUT.

un :
User number of catalog to search for the srec and cf files. If not specified, the default is the current user.

2.0 MOTOROLA 68000 UTILITIES COMMANDS
2.2 SIM68K - RUN THE MOTOROLA 68000 SIMULATOR

Examples

```
ses.sim68k memory
1***** M68K BATCH SIMULATION   82/07/30   16.44.53. *****

?
```

This example shows SIM68K used to simulate the program on file memory. Any command may be entered after the ?

```
ses.sim68k cy001sr cf=command o=list
REVERT.     END SIM68K
```

This example shows SIM68K used with a command file and an output file. The simulator is going to load cy001sr into memory and then execute the commands on the file command. Output from the simulator will be on the file list.

2.3 BIND68K - BIND MC68000 MODULES

The purpose of this command is to create a single bound load module from the specified component object or load modules. The code sections of the old modules are combined into a single code section, data sections with identical attributes are combined into a single section and the binding sections are combined and reorganized for minimum redundancy. Entry point definitions which are referenced from one or more component modules are deleted unless they have been explicitly retained on this command or they have been previously retained.

```
bind68k file=list of <file[,username]>
        name=<module name>
        [module=list or range of <module name>]
        [retain=list of <externally declared procedure
            name>]
        [omit=list of <externally declared procedure
            name>]
        [starting_procedure=<externally declared procedure
            name>]
        upon=<file[,username]>
        [base=<file[,username]>]
        [lock[=file]]
        [output=<file>]
```

file ; f ; library ; lib :
This parameter may be coded as a multivalued list of sublists. Each element of the value list is a single name, refering to the name of a file already assigned to the job or in the current user's catalog; or the element is a sublist, the last sub-element in the sublist is a user name in whose catalog the files (or libraries) refered to by the other sub-elements may be found. An example at the end should help clear this up.

name ; n :
This parameter specifies the module name to be associated with the created module.

2.0 MOTOROLA 68000 UTILITIES COMMANDS
2.3 BIND68K - BIND MC68000 MODULES

module ! modules ! m :
This parameter specifies the modules or module subranges from files or libraries that are to be components of the created module. If not specified, all the modules on the specified files and libraries will be used as components.

retain ! r :
This parameter specifies the externally declared names referenced by a component of the created module that are to be retained in the created module. Procedures that have been retained previously need not be respecified in this parameter.

omit ! o :
This parameter specifies the externally declared names in the component modules whose definitions are to be removed from the output module.

starting_procedure ! sp :
This parameter specifies the externally declared name of the procedure at which execution is to begin. Omission causes last transfer symbol encountered to be used. The starting procedure is always retained in the new module.

upon ! up :
This parameter specifies the file to contain the created module and the user name in whose catalog the file is to be placed.

base ! b ! baselib ! bl :
This parameter specifies a file of object code modules that the upon file will be added. SES procedure GOF will be called if the keys b or base or just a file name is specified. GOL will be called if the keys baselib or bl are specified. GOF and GOL use the upon file as input and the base is rewritten with the created module as the last mdodule.

lock ; nolock :
these parameters determine whether the library update process is interlocked against simultaneous updates. Coding a filename for the lock parameter determines the name of the interlock file. interlocking is the default action when the upon file being updated in another user's catalog.

output ; out :
This parameter specifies the file name to which an object map of the newly created module is to be written.

Example
If you type in the following:

```
ses.bind68k f=(mylib,struct,(sine,cosine,tan,..
    ftnlib)) trig_functions sp=start up=bound
```

The files mylib and struct are either already local or pulled from your catalog. The files sine, cosine, and tan are in the ftnlib catalog or local. The name of the new module will be trig_functions and the new module will be on the file bound.

2.4 REFORM68K - RE-FORMAT MODULE TO H-P OBJECT TEXT FORMAT

REFORM68K takes a relocatable module in the CDC object text format and converts it to the H-P format. One CDC file with relocatable object text is reformatted into two files which conform to the H-P format: 1) a file containing relocatable object code (known in the H-P world as a "reloc" type file), and/or 2) a file containing the local symbols for each module (known to the H-P as an "asmb_sym" type file). One or bothe of these files can be generated, as needs dictate. The re-formatted module(s) can then be downloaded for use on an H-P 64000 workstation. This procedure will work on an input file with more than one module. Each module gets a separate record on each of the output files.

```
reform68k input=<filename>
          [output=<filename>]
          [symbols=<filename>]
          [un=<username>]
          userid=<H-P userid>
```

input ; i :
Name of file containing the module or modules to be re-formatted.

output ; o :
Name of file to receive re-formatted module or modules. If you don't code this parameter, the value of the profile variable 'hpreloc' will be used (if it exists in your "profile" file).

symbols ; sym :
Name of file to receive reformatted local symbols for a module or modules. If you don't code this parameter, the value of the profile variable 'hplocsym' will be used (if it exists in your "profile" file).

un :
User number of catalog to search for input file. The default is the current user.

userid ; uid :
A string of up to 6 characters that is the user id from an H-P development station.

2.0 MOTOROLA 68000 UTILITIES COMMANDS
2.4 REFORM68K - RE-FORMAT MODULE TO H-P OBJECT TEXT FORMAT

Example

```
ses.reform68k cyrel un=cy170 o=hprel sym=hplsym ..
              uid=hpfile
REVERT.  END REFORM68K  CYREL->HPREL,HPLSYM
```

The file cyrel is a file of CDC object text in the catalog cy170. The object text in H-P format will be on a file named hprel. The local symbols will be on a file named hplsym. The user id hpfile is part of the file hprel.

2.5 GENGS68K - GENERATE AN H-P LINKER SYMBOLS FILE

GENGS68K accepts an input the "dbg" file output from LINK68K (the debug keyword must be specified on the call to LINK68K or no "dbg"file will be generated). The file is reformatted into an H-P Linker Symbols File (also referred to in the H-P manuals as a "link_sym" type file, or "File Type 13"). The H-P formatted file can then be downloaded for use on an H-P 64000 workstation.

```
gengs68k input=<filename>
         [output=<filename>]
         [userid=<H-P userid>]
         [un=<username>]
```

input ; i :
(required) name of the linker symbols file output from LINK68K. The file name has the form 'xxxxD8G', where xxx is the name seed used by LINK68K.

output ; o :
(optional) name of a file to receive the H-P formatted Linker Symbols File. If not specified, GENGS68K writes to a local file named 'HPGSYM'.

userid ; uid :
(optional) 1-6 character H-P user ID. This is the same ID entered on the HP64000 in response to the userid prompt. If defaulted, then the current H-P userid will be used when the file is used on the H-P workstation.

un :
(optional) user number of catalog to search for the input file. If not specified, the default is the current user.

2.0 MOTOROLA 68000 UTILITIES COMMANDS
2.5 GENGS68K - GENERATE AN H-P LINKER SYMBOLS FILE

Example

```
ses.gengs68k i=junedbg uld=hpld
REVERT.     END GENGS68K  JUNDBG -> HPGSYM
```

In this example, the file named junedbg is a file of linker (i.e. global) symbols; the H-P formatted linker symbols will be on a file named hpgsym and the user id 'hpld' is recorded in that file.

2.6 BLDMI68K - BUILD MEMORY IMAGE

BLDMI68K generates an absolute module from the output files of the MC68000 Absolute Linker. The file specified by the hdr parameter is acquired by the procedure if it is not local.

```
bldmi68k name=<module name>
         hdr=<filename>
         [output=<filename>]
         [un=<username>]
```

input ! i :
This parameter specifies the module name to be associated with the created absolute module.

hdr ! h :
Name of the header file from the Absolute Linker. The file name has the form xxxxHDR from the linker where xxx is the name seed given by the linker. This file contains information about the other files needed to create the absolute module.

output ! o :
Name of file to receive the new absolute module. If you don't code this parameter, the default file name is MIBMOD.

un :
User number of catalog to search for hdr file and the other files needed. The default is the current user.

2.0 MOTOROLA 68000 UTILITIES COMMANDS
2.6 BLDMI68K - BUILD MEMORY IMAGE

Example

```
ses.bldmi68k abs_module segmhdr newmod
REVERT.     END BLDMI68K
```

This example shows BLDMI68K creating an absolute module named abs_module from files that are specified by segmhdr in the current user's catalog. The module will be on the file newmod.

2.7 PURSYM68K - PURGE SYMBOL TABLES

PURSYM68K removes debug symbol tables, generated by CYBIL/CM or the MC68000 assembler, from modules on object files or object libraries. The input files are acquired by the procedure and the output files are rewritten as permanent files. The same number of files must be specified for both the input and output parameters.

```
pursym68k input=<list of filenames>
          output=<list of filename>
          [un=<username>]
```

input ! i :
Name of file or list of files conteining debug symbol tables. These files are acquired if they are not local. These files can be object files or object libraries.

output ! o :
Name of file or list of files that do not contain debug symbol tables when this procedure is finished. All these files are rewritten into the catalog of the user number specified. If the input file is an object file, the output file will be an object file and the same holds true for object libraries.

un :
User number of catalog to search for input files and where to rewrite the output files. The default is the current user.

2.0 MOTOROLA 68000 UTILITIES COMMANDS
2.7 PURSYM68K - PURGE SYMBOL TABLES

Example

```
ses.pursym68k (exec monitor) (cexe mntr)
REVERT.     END PURSYM68K
```

This example shows PURSYM68K removing debug symbl tables from modules ont the file exec and putting the module without debug symbol tables on the file cexe. The same is done with monitor and mntr.

2.8 TRANDILIB - TRANSLATE DI LIBRARIES

TRANDILIB packs object modules on object files or object libraries into byte oriented data mappings so the new file or library can be downloaded to the DI box. The object modules will appear to have been generated by CYBIL on the C180. The number of input files specified must be the same as the number of output files specified.

```
trandilib input=<list of filenames>
          output=<list of filename>
          [un=<username>]
          [file|lib]
```

input | i :
Name of file or list of files to be translated. These files are acquired if they ar not local.

output | o :
Name of file or list of files that have been translated when this procedure is finished. All these files are rewritten into the catalog of the user number specified.

un :
User number of catalog to search for input files and where to rewrite the output files. The default is the current user.

file | f | lib | l :
This keyword specifies the format of the output files. If you code 'lib', the a module directory and an entry point directory are included in the output files. If you don't code either keyword, then the default is 'file'.

2.0 MOTOROLA 68000 UTILITIES COMMANDS
2.8 TRANDILIB - TRANSLATE DI LIBRARIES

Example

```
ses.trandilib (objfil1 objlib1) (objfil2 objfil3)
REVERT.     END TRANDILIB
```

This example shows TRANDILIB packing all the object modules on the object file, objfil1 and the object library, objlib1, into packed object files objfil2 and objfil3. If you had specified llb or l instead of file, objfil2 and objfil3 would have been packed object libraries

3.0 ERROR_MESSAGES

The following messages are output at the termination of commands. All errors cause command to abort.

20001 HEADER FILE file_name IS BAD
SEVERITY: Error
MEANING: file_name does not have the header file format

20002 SEGMENT FILE file_name NOT FOUND
SEVERITY: Error
MEANING: Self explanatory

20003 ACQUIRE ERROR file_name
SEVERITY: Error
MEANING: Internal error trying to acquire file_name

20004 USER ID long_user_id IS TOO LONG, NOW short_user_id
SEVERITY: Informational
MEANING: Self explanatory

20005 SYMBOL long_symbol HAS BEEN SHORTENED TO short_symbol
SEVERITY: Informational
MEANING: Self explanatory

20006 UNKNOWN OBJECT TEXT IN file_name
SEVERITY: Error
MEANING: Self explanatory

20007 UNEXPECTED EOF OR EOR FOUND
SEVERITY: Error
MEANING: Self explanatory

20008 TOO MANY EXTERNAL REFERENCES
SEVERITY: Error
MEANING: Self explanatory

20009 MODULE module_name IS NOT A MC68000 MODULE
SEVERITY: Error

3.0 ERROR MESSAGES

MEANING: Self explanatory

20010 DUPLICATE ENTRY POINT NAME
SEVERITY: Error
MEANING: Entry point name specified more than once

20011 MODULE module-name IS ALREADY ABSOLUTE AND CANNOT BE REFORMATTED
SEVERITY: Error
MEANING: REFORM68K is only for relocatable modules.

20012 ORG ADDRESS FOR section_name IS INVALID
SEVERITY: Error
MEANING: New module is made up of too many modules

20201 MULTIPLE IDENTIFICATION RECORDS FOUND ON MODULE module_name
SEVERITY: Error
MEANING: Self explanatory

20202 SECTION OF NEW MODULE IS TOO LONG
SEVERITY: Error
MEANING: New module is made up of too many modules

20204 UNKNOWN SECTION ORDINAL FOUND ON MODULE module_name
SEVERITY: Error
MEANING: Recompile module

20205 MISSING SECTION DEFINITION ON MODULE module_name
SEVERITY: Error
MEANING: Recompile module

20206 REFERENCING OUTSIDE SECTION ON MODULE module_name
SEVERITY: Error
MEANING: Recompile module

20207 TOO MANY LIBRARIES ENCOUNTERED
SEVERITY: Error
MEANING: Self explanatory

20211 STARTING PROCEDURE start_proc_name NOT IN CODE SECTION
SEVERITY: Error
MEANING: Self explanatory

3.0 ERROR MESSAGES

20217 ATTEMPTING TO BIND module_name, AN ABSOLUTE MODULE
SEVERITY: Error
MEANING: module_name cannot be bound.

20218 COMMON BLOCK common_block_name HAS 2 DIFFERENT LENGTHS, SECOND FOUND IN MODULE module_name
SEVERITY: Error
MEANING: Self explanatory

20219 ERROR ENCOUNTERED IN SYMBOL TABLE IN MODULE module_name
SEVERITY: Error
MEANING: Self explanatory, recompile module.

20220 ERROR ENCOUNTERED IN LINE TABLE IN MODULE module_name
SEVERITY: Error
MEANING: Self explanatory, recompile module.

20301 NUMBER OF INPUT FILES DOES NOT EQUAL NUMBER OF OUTPUT FILES
SEVERITY: Error
MEANING: Self explanatory

20302 MISSING IDENTIFICATION RECORD ON FILE file_name
SEVERITY: Error
MEANING: Self explanatory

20303 FOUND A TEXT RECORD THAT IS NOT SUPPORTED FOR MC68000 ON FILE file_name
SEVERITY: Error
MEANING: Self explanatory

Table of Contents