sc_6.2

COLLABORATORS			
	TITLE : sc_6.2		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		July 28, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

sc_6.2

1.1 SAS/C Version 6.3 Changes

This document describes bug fixes and enhancements made in the 6.3 release of the SAS/C Development System for AmigaDOS. The changes for the 6.3 release are a superset of the changes for the 6.1 and 6.2 releases; all 6.1 and 6.2 fixes are included with 6.3. Therefore, the patch process will work correctly whether you have installed the previous patches or not.

Our philosophy in reporting fixed bugs is that users have a need and a right to know what has changed in the latest release. This means publishing a list of fixed bugs with each update. Please don't be alarmed by the number of items on the list; many of the "bugs" reported here are not very serious, and others are very obscure. Some of them have only been reported once, others have been reported many times. Thanks to all the alert programmers out there who have brought problems to our attention and provided the detailed information necessary for us to reproduce and fix the problem.

If this update does not fix a problem that you have been having, please contact our Technical Support Division for help as soon as possible.

All reproducable reported CXERRs , Enforcer hits, and cases of bad code are fixed with each release. If you get one of these, please contact technical support immediately. It will help if you have a compilable test case.

In addition to the specific bugs listed, several code generation bugs were fixed in the optimizers and the code generator (sc2). Since the code generation bugs did not affect most people, and they are not easy to describe or reproduce, we will not describe them individually here. We suggest that you recompile all your code if it is behaving strangely.

A note about the release numbering system is probably in order. With version 6, we have adopted Commodore's shared library numbering system for the entire product. This means that the number to the right of the decimal in the version number should be considered a number in its own right, not merely as a fraction. For example, our first release was 6.0, then 6.1, then 6.2; this release is 6.3; the logical progression will be 6.4, ... 6.9, 6.10, 6.11, ..., 6.99, 6.100, 6.101

and so forth. This is a change from version 5, in which the number to the right of the decimal was considered a fraction, with possible leading zeroes. We reserve the right to skip some numbers to keep track of internal releases, but the number to the right of the decimal will always increase with a new release.

Also note that not all executables and shared libraries have been modified. Those that have not been modified keep their old version number (either 6.0, 6.1 or 6.2). This is normal. A list of modified files appears at the end of this document.

Users of Comeau C++ should be aware that Comeau Systems has produced a new version, version 3.0b, that has integrated support for SAS/C V6. Version 6.3 works well with Comeau C++. Contact Comeau Systems for information on obtaining the 3.0b patch, which is a free upgrade for registered Comeau C++ V3.0 customers.

********** IMPORTANT **********
Many customers are experiencing problems that could easily be avoided by referring to the correct sections of the manual. Specifically

1) There are two sections in the manual covering GSTs. The first is in the User's Guide Volume 2 in the Utilities section. The second starts on page 21 in the Library Reference Manual. Both sections contain information needed to use GSTs.

2) A guide to converting programs from version 5 to version 6 can be found in Appendix 5 of the User's Guide Volume 1.

3) With a floppy disk install, no icons will be copied to the created disks. This means that you will have to work from the CLI only, or that you will have to delete something off of the disks in order to make room for the icons. This is not mentioned in the manual and should be. For more information on workarounds for this, refer to Problem 5 under the Miscellaneous Problems button on the main help screen for Common Problems.

****

Click on one of the gadgets below for more information about specific changes.

1.2 Library and Header Changes

Fixed in 6.3

The difftime() function was returning incorrect values when using MATH=FFP.

Possible crash in startup code when using stack extension or the __stack external variable to set your stack size. NOTE: YOU SHOULD RELINK ALL YOUR PROGRAMS THAT USE STACK EXTENSION OR __stack!

The __assert function prototype in assert.h did not match the documentation in the library reference manual. However, __assert is intended for compiler and library internal use only, so the documentation should be considered incorrect.

The %i format identifier for scanf() did not work properly with 0. scanf() took the 0 as the beginning of an octal number.

Both the builtin strcmp() and the library strcmp() function did comparisons based on signed chars instead of unsigned chars.

ferror() indicated a read error if you attempted to read a file that was at EOF.

fscanf() did not accept the field width specifier for the %x format.

stco_i() caused a machine crash.

The file "LIB:utillib.with" discussed on page 117 of the User's Guide was inadvertently omitted from the product. As of the 6.3 patch, it has been added.

The stacksize() and stackused() functions did not work. They do now.

In math.h, the exception type codes _PLOSS and _RANGE were both defined to be 6. _PLOSS is not currently used, but was changed to 7 in case it is used in the future.

__timecvt() filled in the ds_Tick field with an incorrect value.

open() didn't correctly set errno when you opened a file that already exists in O_EXCL mode.

Floating point overflow was printed by printf() and company as "Inf0.00000" instead of just "Inf".

dfind() returned a '1' when asked to find a wildcard pattern and no files match the pattern. It should have returned -1 instead.

timer() function caused references to illegal addresses.

Fixed in 6.2

#pragmas were incorrect for CachePreDMA() and CachePostDMA().

errno was not set correctly if open() failed.

The datecmp() function did not return correct values.

The format specifier %06.2f did not properly pad with zeros.

The base of the system library keymap.library, KeymapBase, was not being auto-initialized properly.

fclose(stderr) caused the invoking shell to stay locked.

cback.a did not work under 1.3.

fputc() did not return the proper value for negative numbers.

The __aligned keyword did not work for global variables in a shared library.

Fixed in 6.1

The tmpnam() function used the first three characters of the command as invoked in constructing the name of the temporary file. This did not work if you invoked a command via an explicit path that contains a ':' or '/' in the first three characters. This was the cause of several strange problems in the SC and SLINK commands as well.

In __main() , the return value for the Open() which opens the stdio window was never checked. Also, __main() now checks the __stdiowin[] array and does not add the program name if __stdiowin[] does not end in a "/".

Some system tagcall routines did not have tagcall pragmas, most notably EasyRequest() and BuildEasyRequest().

<dos.h> did not include the definitions for REG_FP0 - REG_FP7. These definitions are used by getreg() and putreg() .

scanf() routine failed to scan a floating point number with a leading '+' character.

fread() did not set EOF flag properly. (It did return the correct return code, however.)

forkl() did not correctly open "*" for stdin and stdout.

If the UTILLIB option was used and an auto-open library failed to open, the __autoopenfail() function crashed the machine when invoked.

Autoopen libraries were opened twice and closed once because the autoopen functions were compiled with PARMS=BOTH .

<sys/dir.h> was not protected from multiple #includes.

m68881.h did not include definitions for fgetexp() and flog2().

<stat.h> did not contain definitions for the S_xxx bits. (They were previously defined in <fcntl.h>; now they are in both files.)

The bits returned by stat() in the st_mode field were incorrect. The 4 bits corresponding to the "rwed" attributes were OPPOSITE what

they should be. All other bits were correct.

<proto/all.h> did not include all <proto/xxx.h> files, notably
<proto/rexxsyslib.h>.

__builtin_memcmp() generated incorrect code when a variable
containing zero for the length to be compared was passed to it.

In <proto/layers.h>, LayersBase was declared struct LayersBase *; it is
now declared struct Library *.

Closing a file descriptor twice caused a crash. While this is arguably
an illegal operation, most systems accept it silently.

1.3 CPR Debugger Changes

Fixed in 6.3

CPR "grabbed" the input focus on whatever screen it was running
on when the user made a menu selection, even if it was not the
active window. This was most noticeable on the WorkBench
screen (CPR -w).

libinit.c failed to decrement the lib_OpenCnt field in the library
base if a memory allocation failed during library startup.

CPR now works with Enforcer on 68040 machines. A version
of Enforcer that supports the 68040 is required.

CPR left extra MsgPorts allocated on exit.

After using the SOURCE command, crashes were possible.

Executing a CATCH command followed immediately by an OPT TASK
command in a REXX script might have caused a crash.

CPR failed to take into account the full set of viewmodes from
the workbench in an AGA machine. This caused it to possibly
open in the wrong resolution when opening its own screen.
(CPR also now uses SA_LikeWorkbench when appropriate.)

Fixed in 6.2

CPR crashed when opened on a WorkBench screen in Productivity mode.

CPR got enforcer hits when using the MODULE command or window.

CPR crashed on exit under 1.3.

Typedefs of arrays were not handled properly.

Fixed in 6.1

CPR didn't pass argv[0] correctly when invoked from WorkBench.

CPR didn't get the address right for the first data item in the FAR or CHIP sections.

The -device , -unit and -speed parameters were not implemented on cross CPR.

Some programs required abnormally long times to load under CPR. This load time has been reduced.

Watch and register windows are no longer covered up by the source window if they are opened via command-line options .

Hitting the ESC key in the CPR WMSG window caused the window to go away. Additional WMSG commands after this crashed CPR.

Highlighting was messed up if a window was resized while something was highlighted.

The ECHO command didn't work reliably from an externally-invoked AREXX script.

The CPR screen stayed open if a visitor window was present, but CPR exited anyway.

The Calls window sometimes trashed its own bottom border.

CPR was doing incorrect pointer arithmetic for multidimensional arrays.

1.4 Parsing and Diagnostic Changes

Fixed in 6.3

NEW OPTION: the STRSECT option has been added to the SC command. If specified, STRSECT chooses the location that strings will be stored. Possible values are NEAR, FAR, CODE and DEFAULT. The default is DEFAULT. Acceptable abbreviations are N, F, C and D. If NEAR, FAR or CODE is specified, the strings will be placed into the appropriate section - near, far or code. If DEFAULT is specified and STRMERGE is on, strings will go into the code section. If DEFAULT is specified and STRMERGE is not on, strings will go into the default data section as set by the DATA= option. It doesn't matter whether the STRMERGE option appears before or after the STRSECT option.

Redundant keywords are now a warning instead of an error. (For example, "long long x;")

Enforcer hits were generated if using the ANSI option, a GST file, and misspelled union member names.

Enforcer hits when compiling with DEBUG=FF , a GST file, and the GENPROTO compiler option.

The PPONLY option stripped #pragma statements. These should

have been passed through.

The __REVISION__ preprocessor symbol was not updated in versions 6.1 or 6.2. It is now "3" to reflect revision 6.3.

New preprocessor symbols _OPT, _OPTINLINE, and _DEBUG have been added. They are defined to 1 if the corresponding SC command-line option is active. _DEBUG is defined to 1 if any debug optione except NODEBUG is specified.

The SC5 command attempted to invoke the code generator even if it was generating prototypes or preprocessor-only output.

The SC5 command returned a return code of zero when the compile step succeeded but the link step failed.

The SC command generated crashes and/or Enforcer hits when invoked with a filename of '*' and either XREF or LIST was on. This could have occurred if you compiled from within SE and selected LIST or XREF in your SCOPTIONS file.

Fixed in 6.2

There was a problem if a C source file was used to create a GST, and that file defined external data. IT IS NOW NOT POSSIBLE TO DEFINE EXTERNAL DATA if the MakeGST option is on. We recommend compiling the file twice, once with MakeGST and NoObjName , the second time with the normal GST option, or that you use a standalone .c file not part of your project as your GST generator file.

An extra quoted string at the end of a preprocessor statement (illegal according to the ANSI standard) was incorrectly concatenated to the next quoted string in the source.

The ObjName option did not accept the null string "", which should indicate the current directory. It now does.

String constants that appeared on a preprocessor statement in an illegal position sometimes were concatenated before the next string constant to appear in legal code.

The compiler was producing an invalid error if the GenProto option was on and an __asm function was declared that took no parameters, and the function so declared was used somewhere in the same file without otherwise being declared.

Error 72 , "conflict with previous declaration", is now a warning if the only conflict is in a "const" or "volatile" specification on the item declared or parameters to a function declared.

The compiler silently accepted the following illegal constructs:

```
char x[] = {"ab", "cd"};
char y[] = {'a', "bc"};
```

The compiler did not put the "const" or "volatile" keywords on prototypes generated with the GenProto option, or on types printed in error and warning messages.

If the NoMultipleIncludes option was on, but you included the same file with differences only in the case of the filename, the file was included twice anyway.

Fixed in 6.1

The __chip keyword produced an Error 82 for arrays declared that were larger than 64k-1 bytes. This restriction has been removed.

No diagnostic was produced for assigning a regargs function to a stdargs function pointer, or vice-versa. However, the warning "stdargs used for indirect function call" was produced inappropriately for calls through function pointers declared __stdargs. The inappropriate warning has been removed; the assignment now generates Warning 225 .

Error 15 was generated inappropriately for a call via a function pointer defined via a typedef.

Invoking a preprocessor macro with no parameters that is followed immediately by a parameter list containing the preprocessor macro name once again followed by a parameter list failed to expand the second invocation of the macro. (This is an extremely obscure case, so don’t worry if you didn’t understand the explanation.)

If an undefined member of a structure was accessed, that structure was no longer considered equivalent to other structures with identical member lists. This generated warning cascades.

National characters above 0x80 were not permitted in variable names.

When using the GenProto option for a file with more than one period in the name, (ex. a.b.c), the generated proto file was placed into a .h file with only the first part if the original name (ex. a.h).

The ERROR=ALL option to the SC command turned on all messages, including those normally suppressed. It should have just promoted the warnings that were already enabled to errors.

The -ca option on the SC5 command incorrectly enabled Warning 148 .

Error 67 was generated if the __inline keyword was placed on the definition of a function returning a pointer to a structure.

C++-style comments were not recognized if the ANSI option is on. While this is arguably correct, since C++ comments are non-ANSI, many users requested that they be made to work.

If the GST has only <proto/intuition.h> in it, then you #include <intuition/intuitionbase.h> in the .c file, IntuitionBase was considered to be an undefined structure tag.

Generating a GST when the file being compiled erroneously tried to access a structure that had not been defined resulted in a crash.

Generating a GST with static data defined in a header file now generates an error message. This is invalid; data cannot be defined in a header file, and there is no way to make a static data item into a simple declaration, as is done with external data.

1.5 Code Generation/Optimization Changes

Fixed in 6.3

When using the global optimizer on an Amiga 1200, the screen would blank during optimization and never recover. (The compilation would complete successfully, however.)

When using a #pragma tagcall with SHORTINT , integral constants were pushed as shorts as if for any other function call. However, tagcall functions always expect longs. The compiler now promotes all "int" values passed to tagcall functions to "long" automatically, even if SHORTINT is on.

Compiler sometimes generated byte immediate addressing when word immediate addressing was required.

Passing a structure consisting only of "char" elements by value caused the values to be received incorrectly by the called routine.

Taking the address of a static function previously defined in a module compiled with the ABSFUNCPTR and CPU=68030 options generated an incorrect result.

If the code section was too large to allow use of the STRMERGE compiler option, a CXERR was produced. This has been changed to an error message instead.

NOOPTINLINE did not turn off inlining if OPTDEPTH , OPTCOMP , or OPTRDEPTH were turned on. NOOPTINLINE should prevent any and all inlining. It now works correctly.

Fixed in 6.2

MATH=68881 was broken when dealing with constants.

With CODE=FAR and CPU=68020 , foward references to functions were not right.

Bitfields of 32 bits with CPU=68020 did not work.

Referencing formal register parameters when there were more than 32k of autos was broken.

SC2 could generate bad code for __asm functions with many parameters.

Fixed in 6.1

The return value of a function was lost when returning from a call in which the stack extension code allocated additional stack.

Auto variables sometimes got trashed when using longjmp() to jump up the stack.

When SHORTINT is selected, all numbers between 32k and 64k were translated improperly as negative numbers between -1 and -32k.

Initialization of automatic arrays and structures didn't work if the initializers did not contain as many elements as the array or struct and the STRMERGE option was used.

The DISASM option caused a crash if you specified a directory name as the output filename.

The assembler code produced by the DISASM option included MULSL.L and DIVSL.L instructions instead of MULS.L and DIVS.L.

The DISASM option produced bad assembly code if you had calls to __builtin_emit() in your code.

An incorrect function pointer was generated for a forward reference to a static function if the ABSFUNCPOINTER option was specified.

With the global optimizer active, calls to tagcall functions made via #pragma tagcall sometimes did not clean up the stack properly.

Functions declared __inline sometimes did not generate actual callable instances of themselves when they should have. This occurred if you did not run the global optimizer; if the function called itself recursively; or if the function was called from outside the current module. The functions would generate an unresolved symbol at link time.

FPrintf and other tagcall functions that take an OPTIONAL tagcall list generated an error message from phase 2. #pragma tagcall functions now silently accept empty tagcall lists.

Implicit references to substructures generated incorrect offsets if the substructure was not the first item inside the main structure.

Adjacent bitfields declared explicitly "short" or "long" were not merged properly. This caused the structures containing them to be larger than they needed to be. Bitfields declared "int" or "char" were not affected.

Some variables were listed with the wrong type in the compiler-generated cross-reference. This occurred after a typedef definition.

1.6 SE Editor Changes

Fixed in 6.3

SE created an incorrectly-named icon during a block write operation in some cases.

SE did not handle insertions for the clipboard correctly if an initial read succeeded for fewer bytes than requested, but additional data was available anyway.

SE did not allow a filename to be typed to the Open and Save As menu selections under AmigaDOS 1.3.

With case-sensitivity turned off, the SEARCH command failed to find some strings with embedded numerals unless the numbers were escaped with \.

Crash in SE if you tried to copy a block that overlapped two lines to the end of the second line.

If you tried to compile a file from the editor and there was a space in the device name the file existed on, the compiler did not receive the file name correctly and a spurious requester was raised.

If you created keystroke macros 1 through 0, saved them, and then reloaded them, macros 6 through 0 were not reloaded.

Fixed in 6.2

Busy pointer did not change back to ready after a compile until it was moved.

SE hung if the zoom gadget was hit with a large font.

SE did not save backup file and protection bit if rename backup option was selected.

Going from an interlaced screen to a non-interlaced screen would fail if the interlaced window was too small.

Fixed in 6.1

SE was not pure (as reported by WShell).

SE returned immediately when asked to compile the file in the editor buffer (F4 key) if the flag was enabled under AmigaDOS 2.0 that allows the "*" character to be a wild card (RNF_WILDSTAR).

Turning on the "Use TAB Character" option and the Autoindent option resulted in garbage characters being added to the file when you hit RETURN after entering an open curly brace.

SE did not turn the mouse pointer back on during time-consuming operations (like SEARCH/REPLACE).

SE sometimes opened a window smaller than its menus required when switching from interlaced to non-interlaced mode.

It was impossible to configure the screen colors and save them. You could cycle the colors while in SE, but when you exited, the colors weren't saved.

A bad version of the editor macro "findsym.se" was provided.

SE set no pointer during compilation. It now sets a busy pointer.

SE did not correctly add the backup file extension when saving backup files.

SE did not accept national characters that were entered through a REXX script.

1.7 SCMSG Message Browser Changes

Fixed in 6.3

The "delfile" REXX command caused a crash.

SCMSG allowed you to open the options window twice with two selections of the menu item, but then failed to close the first window.

SCMSG could trash the GOTOFILE option if NOGOTOFILE was set in the options file, then changed later.

Fixed in 6.2

The NEXT and PREV REXX commands didn't cycle back to the top or bottom of the message list as documented.

The PREV REXX command returned an incorrect return code regardless of the success status.

Fixed in 6.1

SCMSG menu colors were funny-looking on an 8-color workbench screen.

SCMSG didn't protect itself against the editor exiting while it was processing a message to the editor.

SCMSG did not open on the default public screen. It now does so; it also accepts a SCREEN <name> command-line parameter and an "opts screen <name>" AREXX command (see below).

SCMSG redrew its message lines even if they had not changed when receiving new messages. This caused unnecessary flicker.

SCMSG didn't set the AUTOEDIT option even if the options file said to. It also reset the GOTOFILE and GOTOLINE options to the SE defaults even if the user had erased those values.

SCMSG now accepts the following escape sequences in the REXX commands that it sends to the programmed editor. These are in addition to the %f, %l and so forth that are documented in the manual:

```
%# - message number
%m - message text
%e - alternate filename
%k - alternate line number
```

```
Note also that the %c escape sequence gives the message Class (ERROR/WARNING)
rather than the column number, as documented in the manual.  This was an
error in the manaul.

SCMSG now accepts the following new or modified REXX commands.  All old
commands still work.

   "build [options]"

      The "build" REXX command now passes any options specified on to the
      SMAKE utility.

   "hide" [option]
      The "hide" REXX command takes "rexxonly", "norexxonly", "autoedit", and
      "noautoedit" as options.  The options control when the window will
      reappear.  The "wait" option listed in the manual is the default, so
      there is no need to specify it.

   "number"
      synonym for "errnum"

   "newbld <compunit>"
      tells SCMSG to clear all messages for the specified compilation unit.
      Sent by the compiler when a new compile is begun to delete all old
      messages.  It also forces SCMSG to invoke the editor on the next new
      message if the AUTOEDIT option is on.

   "newmsg "<compunit>" "<file>" <line> 0 "" 0 <class> <errnum> <text>"
      adds a message to SCMSG's list.  <compunit> is the compilation unit
      the message is associated with (the .c filename), <file> and <line>
      indicate the filename and line number of the message, <class> is
      one of "Error", "Warning" or "Info", <errnum> is a positive error
      number, and <text> is the message text.  The 0 "" 0 in the middle
      must appear exactly as written.  If the message text contains the
      words 'See line <number> file "name"', the <number> is taken to be
      the alternate line number and the <name> is taken to be the
      alternate file name.

   "opts [option]"
      where [option] is one of the following:

         load <filename>  - Load options from the specified file
         save <filename>  - Save options to the specified file
         portname <xxx>   - Set the name of the editor's AREXX port to
                            the specified value
         screen <name>    - Close the SCMSG window and reopen it on the
                            specified public screen.

The default value for the GOTOLINE option, intended for use with the SE editor,
is now

   LL "%l\n" DM %m

instead of just

   LL "%l\n"
```

```
  This displays the message text in the message line of SE after taking you to the
  line of the message.
```

1.8 Other Utility Changes

```
Fixed in 6.3

 ASM could produce a BSS hunk with size 0, which confused
 the linker.

 ASM did not support the statement    NAME equ "string"

 Any time you had anything after a BSS section besides another
 BSS section, a bad executable was produced.

 ASM did not generate a relocation for an external accessed
 via the addressing mode used in the following instruction:

    move.w   extern_name(a7,d2.w*2),d0

 ASM did not detect that it is an error for bitfield instructions
 to use post-increment addressing.

 The "equr" directive in ASM incorrectly replaced strings which were
 substrings of a longer word.  equr should only replace exact matches
 of strings.

 The CPUSHA instruction in ASM did not handle parenthese around its
 parameter.  It now does.

 ASM did not receive the value of the CPU switch when invoked from SC.

 ASM gave "invalid effective address for opcode" for "pmove d0,tt0".
 This is a valid mode unless the CPR, DRP or SRP register is the
 target.

 BPKT instruction generated incorrect warnings from ASM.

 ASM incorrectly generated H_EXT32 references in some cases where it
 should have generated H_EXT16.

 ASM did not recognize valid 68040 registers ITT0 and ITT1.

 % was not recognized by ASM as the modulus operator.

 ASM put PC-relative displacement in the wrong location in the
 instruction for some cases.

Fixed in 6.2

 ASM's macro handling with respect to local labels was broken.

 REG was not implemented in ASM.
```

```
 ASM did not detect zero-length short branches.

Fixed in 6.1

 SMAKE didn't read the tooltypes array as documented; it also reported that
 A: is a "bad drive specifier" when it was used in a target.

 Closing the SMAKE window after SLINK started caused a system crash.

 SMAKE did not use the SYS_UserShell tag when calling the System() routine
 under AmigaDOS 2.0.  It now does.

 If multiple targets were passed to SMAKE on the command line and the first
 was up to date, the subsequent targets were not built.

 ASM didn't deal with packed decimal mode constants correctly.

 OML gave a "phase error" if you attempted to place object files in
 the library that contained chip data.

 LCTOSC and DUMPBOJ did not set the return code correctly.

 LCTOSC incorrectly translated '-Ln' to 'LINK DEBUGSTRIP' instead of 'LINK
 STRIPDEBUG'

 HyperGST could not locate header files obtained from the local project
 if invoked from WorkBench.

 SLINK did not always set the execute bit on the generated executables.

 LPROF/LSTAT were giving strange results when used on programs with
 multiple code hunks.  This is a very long-standing bug dating back to
 version 5.0.

 Scsetup did not correctly process the '?' parameter when invoked from
 the Shell.  It printed its command template, then started working on
 the current directory as if no parameters were entered.

 SCOPTS only saved to ENV: when asked to "Save Default".  It now saves to
 ENVARC: as well.
```

1.9 Documentation/Help File Changes

```
Fixed in 6.3

 The results of the COVER example in SC:EXAMPLES/COVER did not match
 the results specified in the read.me file.  The read.me file was
 updated to reflect reality.

Fixed in 6.2

 None

Fixed in 6.1
```

sc.guide hypertext file incorrectly listed 255 as the default value for IDENTIFIERLENGTH option, instead of correct value of 31. 255 is the maximum value.

Some new "Common Problems" were added to the Common Problems document.

The scmsg.guide hypertext file was modified to add Error 145 , which is new to this patch release, and to update or add descriptions for Warning 7 , Error 82 , Warning 184 , Warning 198 , and Error 200 .

In addition, the Global Optimizer warnings which were left out of the documentation were added to the Compiler Errors and Warnings section of scmsg.guide. The warnings added are

```
Warning 301
Warning 302
Warning 303
Warning 304
Warning 305
Note 306
```

Additional information was added to the ANSI compiler option description in sc.guide. The ANSI option has effects other than simply turning on warning messages. These effects are now described.

Three Data Names and one function were added to sc_lib.guide. All four additions are listed separately in the main Library Functions help screen under the button Newly Documented Functions and Data Names .

Additional corrections to the documentation can be found in Changes to the Documentation help screen (sc_changes.guide) under the button Version 6.1 (Error Correction Patch) Changes .

The "compile.ced" macro intended for use with CygnusEd had a bug in its use of the "jumpto" command. "jumpto" expects both a line number and a column number, and only the line number was provided. CygnusEd users may want to copy the new version of the macro into their REXX:CED drawer.

The TurboText macros referenced explicit directory paths, but the directory was renamed in the final version; therefore, they did not work correctly as installed. The TurboText directory has been renamed. Please reinstall these macros if you had problems with them in 6.0.

1.10 List of Modified Files

The following files were modified during the 6.3 patch installation process. If they have version numbers, the version should be 6.3.

Drawer	Files
SC:C	se, cpr, cprx, cprk, hypergst, asm, sc5, smake, scmsg, omd
SC:LIBS	sc1.library, sc2.library, schi.library, scpeep.library, scdebug.library, sclist.library scgo.library

SC:LIB	all .lib files, c.o, cback.o, cres.o, catch.o, catchres.o
SC:HELP	sc_6.3.guide
SC:SOURCE	c.a, cback.a
SC:INCLUDE	math.h

The following files were modified during the 6.3 patch installation process, but are the same as they were in version 6.2. If they have version numbers, the version should be 6.2.

Drawer	Files
SC:C	slink, lctosc
SC:INCLUDE	stdio.h
SC:HELP	sc_6.2.guide
SC:SOURCE	libinit.c

The following files were modified during the 6.3 patch installation process, but are the same as they were in version 6.1. If they have version numbers, the version should be 6.1.

Drawer	Files
SC:	read.me
SC:C	scompact, dumpobj, oml, lstat, lprof, tb
SC:ENV	se.dat
SC:LIBS	sekeymap.library
SC:INCLUDE	fcntl.h, stat.h, dos.h, m68881.h, mffp.h, mieeedoub.h, stdarg.h, sys/dir.h, proto/all.h, proto/layers.h, pragmas/intuition_pragmas.h
SC:REXX	findsym.se, showcli.cpr, showprocess.cpr
SC:HELP	sc_prob.guide, sc_lib.guide, sc_6.1.guide, scmsg.guide, sc_util.guide, sc.guide, cpr.guide
SC:SOURCE	autoopenfail.c, _main.c, _oserr.c, intuitlib.c, _cxferr.c, c.a
SC:EXTRAS	TTX/instructions, CED/rexx/ced/Compile.ced