Contents

S1C17 Family Software Development Tool (GNU17) Release history

The following states the changes since the last release and plus the release history.
(Descending order from newest release)

Tool Name	Ver 2.0.0 2010/02/26
Integrated development environment(IDE)	・Integration of GDB and Eclipse ・Compiler option addition (prototype warning) ・Synchronization of symbol registration C/C++ Project Paths -> path Containers was synchronized with registration/deletion of GNU33 Build Options -> Build Options -> Symbol. ・An initial value of %sp set by the boot process is made a variable. ・Correspondence of command (c17 flv,c17 flvs) for GDB MONOS Flash ・Target CPU change was enabled in the project property. ・Addition of Flash memory protecting bit setting function
C Compiler(gcc)	・Supported by default -Werror-implicit-function-declaration option By this, error is output when using a function without a prototype declaration in C source files.
Assembler(as)	・Speedup of two pass make ・Added filter for debug information of long long type
Linker(ld)	—
Library	・Improved listdio.a and libc.a for reducing the linking size
Debugger(gdb)	・"0x" display is deleted for the hex display form in the memory window. ・Allocation of short cut to function key ・The command reference is added to the Help menu. ・C17 flv (set voltage of MONOS flash) command is added. ・C17 flvs (clear voltage of MONOS flash) command is added. ・C17 chgclkmd (select clock source in break mode) command is added. ・The transrating data size is added to the parameter of the C17 fls command. ・To recognize it by enclosing it with a double quotation, the character string including the blank changes by the parameter of the c17 command.) ..(example: Comment parameter etc. of c17 fwlp command. ・Improvement of response speed when low-speed clock (OSC1) is debugged ・Correspondence of ICDmini hardware version 2.0 ・Fixed problems (Refer to GDB-08).
Mask ROM making tool (Winfog/Winmdc)	・Fixed problems When function (UAC) of the control of the user account of Vista is effective, the configuration file of DevTools (winfog and winmdc)(winfog17.ini and fog_sel17.ini, etc.) is not preserved.
LCD Panel Customize tool(LCDUtil17)	・When the model name is passed as a start argument, the function to make the dot matrix of the model is added.

Others	・I/O in surrounding simulator (ESSIM) Fixed Clock measurement value bug. Correction of S1C17705 simulator.

Tool Name	Ver 1.5.0 2009/2/20
Integrated development environment(IDE)	・Supported –O3 option ・Supported new Japanese character (Kanji) filter ・Fixed problems(Refer to IDE-05)
C Compiler(gcc)	・Supported –O3 option ・Improved speed for a process of double / long long type ・Improved speed for a process of multi array ・Implemented SJIS filter (or Japanese character (Kanji) filter) process The output of a wide character enclosed single quotation is changed from the previous version from Ver1.5.0 by this implementation, if SJIS filter process is enabled (-mno-sjis-filt option is not set). Refer to “filter function for Shift JIS code” in compiler package manual for details. And No.3 of “GNU17 C Compiler Known Issues” has been resolved by this implementation. ・Added –mno-sjis-filt option which disables SJIS filter (or Japanese character (Kanji) filter) process
Assembler(as)	—
Linker(ld)	—
Library	・Improved speed for emulation library(libgcc.a / libgccM.a / libgccMD.a) ・Improved speed for pow() function of ANS library(libc.a) ・Deleted dummy functions from ANSI library(libc.a)
Debugger(gdb)	・Added commands command ・The change of the default value of the Address item of the memory window was enabled. ・Fixed problems (Refer to GDB-07).
Mask ROM making tool (Winfog/Winmdc)	—
LCD Panel Customize tool(LCDUtil17)	・Increased restriction value of segments that can be read from a bitmap file.
Others	・Fixed problems (Refer to kanji-02). ・Fixed problems of VECTOR definition for sample programs ・moto2ff.exe added check size argument

Tool Name	Ver 1.4.0 2009/1/6
Integrated development environment(IDE)	・ Updated IDE base versions to Eclipse 3.4/CDT5.0/JavaVM5.0 ・ .cdtproject file replaced to .cproject when new project is created or project is imported ・ Supported new CPU models (17705) ・ Added objcopy parameter '-I elf32-little' in mak file ・ Added objcopy parameter '-I elf32-little' for elf context menu 'Object file conversion' ・ Fixed problems(Refer to IDE-04)
C Compiler(gcc)	—
Assembler(as)	—
Linker(ld)	—
Library	—
Debugger(gdb)	・Profiler/coverage function is added to the simulator mode. ・When the file name is specified, the c17 stdout command is displayed in the simulatedIO window. ・Fixed problems (Refer to GDB-06). ・I/O simulator (Essim17) support for new models (17705). ・Fixed problem : Essim17 hangs when reading an LCD file with no COM/SEG configuration.
Mask ROM making tool (Winfog/Winmdc)	—
Others	・ For Windows Vista in all the above tools. ・ Replaced following tools based on cygwin-1.5.25 cygwin1.dll/cygiconv-2.dll/cygintl-3.dll/cygintl-8.dll/ar.exe/cp.exe/make.exe/ objcopy.exe/rm.exe/sed.exe/sh.exe ・ Replaced fls17 module for S1C17704(¥mcu_model¥17704¥fls¥fls17704.elf)

Tool Name	Ver 1.3.0 2008/9/4
Integrated development environment(IDE)	・Supported multiplication and division co-processor library. ・Added Vector checker for Co-pro use and configuration dialog. ・Display build complete message in console when build is successful. ・Build goal set to ELF for imported projects that were created using earlier versions than GNU17v1.2.0. ・Changed parameter file settings for newly created projects. ・Added file filters *.dump/*.sa/*.saf/*.out for Navigator view.
C Compiler(gcc)	—
Assembler(as)	・Speed up of 2 pass make.
Linker(ld)	・Strengthening of overlap check of memory arrangement.
Library	・Added library corresponded to copro instruction of multiplication/ division (libgccMD.a).
Debugger(gdb)	・Addition of preservation and set again function of symbol registered in Watch Window. ・Added the division Copro instruction of core simulator. ・I/O simulator (Essim17) can read PSR register in S1C17702 and S1C17602. ・Fixed problems (Refer to GDB-05).
Mask ROM making tool (Winfog/Winmdc)	—
Others	・Optimization of boot part of sample program. ・Added Vector checker (vecChecker.exe) to package.

Tool Name	Ver 1.2.1　2008/6/30
Integrated development environment(IDE)	・Supported Linked Resources ・Supported Selection of file (ELF/PSA) made in Project properties->GNU17 Build Options. ・Supported new CPU models (17003) ・Fixed problems (Refer to IDE-03).
C Compiler(gcc)	・Fixed problems (Refer to GCC-01). And the size of a structure and a union is adjusted to be sure to become an even number byte as a result of the correction of < content -1> of GCC-01. Note that one byte's unused area might be added at the end so as not to become an odd number byte.
Assembler(as)	—
Linker(ld)	—
Library	・The prototype declaration of the header was changed to the ANSI-C conforming. ・Fixed problems (Refer to LIB-02).
Debugger(gdb)	・The number of the hardware break that was able to be set became four or less. ・Added Saving of breakpoints and loading breakpoints ・Changes binary form in Local/Watch Window. ・Fixed problems (Refer to GDB-04). ・I/O simulator (Essim17) support for new models (17003).
Mask ROM making tool (Winfog/Winmdc)	・Supported new CPU models (17003)
Others	・Delete PSR read/write sample ・Change S1C17602 flash writing program(cpu_model¥17602¥fls¥fls17602.elf) Fixed flash might not be deleted. ・Japanese character filter supports /cygdrive/ format paths for source files.

Tool Name	Ver 1.2.0　2008/4/28
Integrated development environment(IDE)	･Supported　-Wall and -O0 option. ･Supported new CPU models (17001･17002･17501･17602･17702･17704･17801). ･essim17_user.ini file generated at project creation. ･Added LcdUtil17 launch button. ･Added mask ROM generator tool launch buttons(Winfog･Winmdc) ･The command that generates the psa file (S2 record file) from the elf file is added to the Make file. ･The command that loads the psa file (S2 record file) is added to the command file. ･Flash writer command is added to the debugger command file. ･Co-processor library link setting generated at project creation. ･Disabled CPU change feature from the Project Properties.

C Compiler(gcc)	･The execution speed is sped up by the following optimization.
	(1) Optimized loop processing
	The loop processing (for sentence and while sentence, etc.) has been optimized.
	As a result, the code generation in the loop processing is subjected to change.
	(2) Corresponded to the delayed branch instruction
	It came to be able to generate the delayed branch instruction (call.d/ jpr.d etc.).
	The delayed branch instruction can be executed by fewer cycles than a usual branch instruction.
	(3) Deleted unnecessary cmp 0 instruction
	In the following patterns, “cmp %rd, 0” instruction is deleted because it is possible.
	pattern 1)
	and %rd, %rs/sign7
	cmp %rd, 0
	jreq / jrne / jrgt / jrge / jrlt / jrle
	pattern 2)
	add %rd, %rs/sign7
	cmp %rd, 0
	jreq / jrne
	･Supported -O0 (No optimization) option.
	･Supported -Wall (All warning is output) option.

Assembler(as)	—
Linker(ld)	･Fixed error message when object files generated with –mpointer16 and object files generated with –mpointer24 is linked with each other.
Library	Added library corresponded to copro instruction of multiplication (libgccM.a) ･Improved speed for comparison of floating points, sign reversing, addition, 64bit integer multiplication, and 16bit shift operations ･Fixed ANSI C library (Refer to LIB-01).

Debugger(gdb)	・Added ″c17 hbreakmd” command ・Added “set output-radix” command ・Supported asm(“brk″) of C source code; ・Added multiplication calculation Co-processor instruction to the core simulator. ・Added I/O simulator (Essim17). For models 001,602,701,702,704. ・Fixed problems (Refer to GDB-03).
Mask ROM making tool (Winfog/Winmdc)	・Supported new CPU models (17001・17002・17501・17602・17702・17704・17801)
Others	・Added LcdUtil17.exe (LCD file making utility). ・Fixed sconv32.exe to inhibit output message. ・Changed moto2ff to report error when program is in input file outside of specified address range. ・Added PSR read/write sample ・Added coprocessor library(libgccM.a) sample

Tool Name	Ver 1.1.5　2008/3/27	Ver 1.1.4　2008/3/17
Integrated development environment(IDE)	—	・Fixed problems (Refer to IDE-02).
C Compiler(gcc)	—	—
Assembler(as)	—	—
Linker(ld)	—	—
Library	—	—
Debugger(gdb)	—	—
Mask ROM making tool (Winfog/Winmdc)	—	—
Others	・Fixed problem concerning Japanese character filter (tool for Japanese) (Refer to Kanji-01).	—

Tool Name	Ver 1.1.2　2007/11/29	Ver 1.1.1 2007/10/24	Ver 1.1.0 2007/9/29
Integrated development environment(IDE)	—	—	・Changed parameter file wait configuration for S1C17701 ・Fixed problems (Refer to IDE-01).
C Compiler(gcc)	—	—	—
Assembler(as)	—	—	—
Linker(ld)	—	—	—
Library	—	・Fixed ANSI C library　text (utility¥lib_src¥ansilib¥doc)	—
Debugger(gdb)	・Fixed problems (Refer to GDB-02).	—	・Improved Flash ROM load speed ・ Changed Essim17 register default values, SVD comparison levels ・Fixed problems (Refer to GDB-01).
Mask ROM making tool (Winfog/Winmdc)	—	—	・Newly added.
Others	—	—	・Removed 512 characters per line restriction to none in Japanese character filter ・ Fixed S1C17701 Flash load/erase program.

Description of Problems

No	Problems

GCC-01	<Content-1>
	An address error exception occurs when an array element of the structure is passed by value to a subroutine.
	
	<Generation condition>
	When all the following conditions are filled.
	
	* Declaring the structure as an array.
	* An array of the structure is passed by value.
	* The size of the structure is 7 byte or less.
	* The size of the structure is an odd value.
	* The type of the structure member variable is only unsigned char / char.
	
	sample code:
	// The size of the structure is defined as 7byte, odd byte,
	// and the structure is constituted only by unsigned char /
	// char member variable.
	typedef struct s_tag {
	char m1 [3];
	char m2 ;
	unsigned char m3 ;
	}STR ;
	
	void sub(STR arg);
	
	int main(void)
	{
	// Declaring the structure as an array.
	STR s[2] = { { 0, 0, 0, 0, 0 }, { 0, 0, 0, 0, 0 } } ;
	int i ;
	
	// An array of the structure is passed by value.
	for(i = 0 ; i < 2 ; i++) sub(s[i]) ;
	
	Continued to next page

	Continued from previous page <Temporary Measures> This bug can be avoided by the above conditions are not filled. Concretely, it is as follows. Example of measures: * Add a char type member variable to the structure, and make the size of the structure into a even value. * Make the size of the structure into 8 bytes or more. * Pass the structure to a subroutine by reference(&).

<Content-2>
It cannot be accessed rightly to the array when operation using immediate is performed at the index expression of the global array.

<Generation condition>
When all the following conditions are filled.

* Compiled with SMALL model(-mpointer16).
* The index expression of the array has a operation using immediate and variables.
* The array is global.
* The type of the array is the following.
  unsigned char / unsigned short / unsigned int / unsigned long / char / short int / long / float / enum / structure / union

sample code:

```
unsigned int INPUT[20];
float f_Val;

int main(void)
{
    int i, y;

    for(i = 0; i <2; i++) {
        f_Val = INPUT[y + 16 - i];   // It cannot be accessed rightly
    };                               // to the array( INPUT[] ).
```

<Temporary Measures>
Do not operate using immediate at the index expression of the array with SMALL model(-mpointer16).
If you need to operate, assign the operation result to the global work variable, and use it as the index expression of the array.

No	Problems
IDE-01	When building within the IDE, the linker reports an “out of range error”. This occurred because the object files generated after the 2pass build (which is optimized) were used at the next 1pass build. To avoid this, the IDE outputs commands that restores the object files generated at 1pass build (which are not optimized) after the 2pass build is complete. Thus, “out of range error” will not occur and builds will correctly proceed.
IDE-02	When one library (libgcc.a) refers to a symbol in another library (libc.a), the executable does not link properly as mapped in the linker script file, causing an “overlap error”. To avoid this, the IDE outputs the linker script file with the section end labels (such as _END_text) located outside of the section definition scope (outside the curly braces).
IDE-03	Size of RAM and STACK was wrong in the project configuration file of S1C17702. (Mistake)0x000000-0x001FBF(8KB) (Correct)0x000000-0x002FBF(12KB)
	Fixed following problems when project is being copied& pasted and then renamed from C/C++ Project and Navigator views. ・ The linkerscript file name in the GNU17 Build Options does not change (-T new_project_gnu17IDE.lds) ・ “GDB17 launch for new_project” configuration is not created in External Tools Please note that project copying and pasting will not change a project. Rename project after pasting it.
IDE-04	Fixed problem : IDE hangs on Pack when the project is for S1C17701 and has a name longer than 57 characters
IDE-05	Fixed problem : On Windows Vista, building projects created on Windows 2000 or XP stops and is incomplete due to cp command error. Changed mak file commands as follows, where invocation of cp.exe is modified to invocation of copy. Old) for NAME in $(OBJS) ; do ¥ $(CP) -pf $$NAME obj1pass/$$NAME ; done ¥ New) for NAME in $(subst /,¥¥,$(OBJS)) ; do ¥ cmd /c ″copy /y $$NAME obj1pass¥¥$$NAME″ >nul ; done ¥ Thus, building on Windows Vista will complete.

No	Problems	GDB connect mode
GDB-01	GDB hangs after writing to flash and setting a software breakpoint.	ICD mode
	Hangs after continuously displaying "CPU is running" message when mouse is clicked on the Memory window while running program.	ICD mode
	unsigned long array displayed in 16bit in print command and Watch window.	Simulator mode /ICD mode
	High CPU usage while running program.	ICD mode
	The instruction, which was transferred to RAM and at which a software breakpoint is set, turns to 0xAAAA after go and break.	Simulator mode
	enum type symbol is displayed in 32bit instead of 16 bit.	Simulator mode /ICD mode
GDB-02	While simulated I/O is running, moving mouse cursor to or clicking on source window causes debugger to hang or display "CPU is running".	ICD mode
	Toolbar stays disabled after "step" command.	Simulator mode
	When the PC register value is the same as the software breakpoint, "continue" fails and stops.	ICD mode
	When a hardware break point is set between the current PC register and the address to where the until command is executed, the program does not stop at the hardware break point.	ICD mode
	Debugger sometimes hangs showing a sandglass after executing a command file (*.cmd).	Simulator mode /ICD mode
	When the source window is in MIXED mode, the green line cursor deviates from the current line when "set $pc=xxx" is executed from the console window.	ICD mode

No	Problems	GDB connect mode
GDB-03	If the memory window is greatly opened at a low-speed clock such as OSC1, it is likely to become a time-out error.	ICD mode
	Debugger hangs when selecting Japanese characters in the source window. Pasting Japanese characters may also cause hangs.	Simulator mode /ICD mode
	After starting debugger, executing “continue” and moving cursor to source window sometimes displays ”CPU is running” and causes a hang. (reproducibility low)	ICD mode
	After force break with STOP button, input from memory window fails to update.	Simulator mode
	When PC is at software break point address, and a hardware break point is set right after, the program does not stop at the hardware break point. ex: int main() { ←① char str[256]; str[0] = 0; ←② Does not break here int p=0; ←③ break ① hbreak ② break ③ When continued and stopped at ①, the next continue will not stop at ②, but at ③	ICD mode

	Setting a break point where it is unable to set in a C source file from the console window causes duplicate break points at the same address.
	
	ex:
	while(p<10)
	{ ←Unable to set break①(no ”-”)
	p++; ←②
	sub2();
	
	set break point at ①
	set break point at ② by clicking source window
	break points are set at same address when displayed with “info breakpoint” command.
		Simulator mode /ICD mode

No	Problems	GDB connect mode
	When setting a temporary break point, and executing step, finish or until commands, the program will stop at the break point but remains and does not get removed by itself.	Simulator mode /ICD mode

	When performing finish and step from within a function in a C source, and the destination is the same function, the program jumps to an invalid location.
	
	ex:
	int main() (continued to next page)
	{
	char str[256];
	str[0] = 0;
	int p=0;
	write_str("*** Test gdb simulated IO ***¥n"); ←①
	write_str("Please enter any string and <CR>¥n"); ←②
	▪
	▪
	
	void write_str(char *str)
	{
	int i;
	char *c;
	
	i = 0; ←③
	c = str; ←④
	while (*c != 0) { /* Check string length */
	▪
	▪
	}
	
	"step" from ① and PC stops at ③ in write_str() function.
	PC jumps to ② with "finish".
	When step is performed, the program does not stop at ③ in write_str() but stops at ④
		Simulator mode /ICD mode

No	Problems	GDB connect mode

	Performing “until” which stops at a function call line, and then “next”,
	the program does not stop at the same location as when performed right after
	the debugger launch and as after a software reset.
	
	int main()
	{
	char str[256];
	str[0] = 0;
	int p=0;
	write_str(″*** Test gdb simulated IO ***¥n″); ←①
	write_str(″Please enter any string and <CR>¥n″); ←②
	write_str(″Egggggg¥n″); ←③
	
	“until” stops at ①
	“next”
	PC value shows ②
	
	Then,
	c17 rst (PC changes by reset)
	“until” stops at ①
	“next”
	PC value shows ③ instead of ②
		Simulator mode

	“next” command with count argument operates as “step” when assembly source has ld and call instruction in line. ex: ld %r0,%r1 ←① call func nop ←③ ・ func: nop ←② When PC value is at ① and “next 2” is performed, PC value shows ② instead of ③.	Simulator mode /ICD mode

No	Problems	GDB connect mode
	When a software or hardware break point is set at a sub routine instruction (“call”) in the assembly source, and “finish” is performed, the program stops at the next line from the break point. ex: boot: xld.a %sp, 0xfc0 xcall init_lib ←①break point here nop ←② start program from boot and stop at ① “finish” command stops at ②.	ICD mode
	When the reset button on the tool bar is pressed, the mouse cursor displays a sandglass on the source window.	ICD mode
	When compiled with 24 bit pointer mode, print command displays wrong value for void pointer arrays. ex: void *pData[10]; pData[1] = 0x123456; // as pointer print /x pData[1] $1 = 0x3456aa ←0x123456 expected but upper8 bits are incorrect. The display is wrong in Watch and Local windows.	Simulator mode /ICD mode
	Tool bar buttons other than the stop button gets enabled on key inputs from the simulated I/O input.	ICD mode
	Source window break points occasionally do not turn disabled(black) when the break point is disabled in the Breakpoints window.	Simulator mode /ICD mode
	First an error is displayed when temporary hardware break points are set twice to the same address. Then the program is run and the temporary hardware break points gets released. Given this situation, the hardware break points to be set next time is displayed in disabled(black) state.	Simulator mode /ICD mode

No	Problems	GDB connect mode
	Print command for pointers shows incorrect value with invalid upper 8 bits, causing an error. ex: When pointer address for l_pcData is 0x2cc0 , "0xa5" in the upper 8 bits is incorrect. (GDB)print *l_pcData" Cannot access memory at address 0xa5002cc0"	ICD mode

	When a function call is at the end of a while loop, "finish" command does not stop at the start of the while loop.
	
	ex:
	<C source and assembly MIX>
	.L5:
	while(p<10)
	{
	p++;
	add %r4,0x1 ←①
	sub2();
	xcall sub2
	sub3();
	xcall sub3
	sub4();
	xcall sub4
	cmp %r4,0x9 ←②
	jrle .L5
	}
	
	"finish" command from within function sub4() stops at ② instead of ①.
		Simulator mode /ICD mode

	Debugger does not report an error when a break point is set to an address that are not in the memory mappings of the parameter file.	Simulator mode
GDB-04	Fixed problem:do not break even if passing excluding the break number specified by ignore command in the simulator mode.	Simulator mode
	Loading the PSA file (write) might not be normally done to flash ROM.	ICD mode
	GDB might freeze when the Japanese character displayed in the source window is clicked with the mouse, and it end.	Simulator mode /ICD mode
	EUC code of Japanese characters (ShiftJIS code) in the source window becomes blank.	Simulator mode /ICD mode

No	Problems	GDB connect mode
	It becomes impossible might do the compulsion break with the STOP button when the while sentence is executed in STEP. ex: <C source> while (*(unsigned long*)0x400 == 0){ }	Simulator mode
GDB-05	There is no [Disable breakpoint] in the menu displayed by right-clicking the temporary break setting of the source window.	Simulator mode /ICD mode
	When verify error on loading to flash memory, do not show error message.	ICD mode
	When there are two or more stack areas in parameter file, It becomes effective only the first one.	Simulator mode
	When continue command executed in command file, The mouse cursor doesn't become an hourglass. And in ICD mode, GDB freezes it.	Simulator mode /ICD mode
	In core simulator,, When an undefined instruction code is executed, it doesn't become an error.	Simulator mode
	In Watch/Local Window, the value of the symbol of the long type of the register allocation doesn't display high 16bit.	Simulator mode /ICD mode
GDB-06	When the line - number that doesn't exist in the until command is specified, the button of the menubar and the toolbar of the Source window becomes like invalidity.	ICD mode
	A wrong character to the changing line part of Source Window and Simulated I/O Window might be displayed.	Simulator mode /ICD mode
	When changing line of the file is CRLF, it inputs it as two characters in the place where the file input is done with Simulated I/O Window by the getc function etc.	Simulator mode /ICD mode
	Even if the address exceeds 0xffffff by the 'set' command and 'x' command, it doesn't become an error.	Simulator mode /ICD mode
	When data that exceeds the display specification size is input to the cell of Memory Window, the value for a specified size is not input.	Simulator mode /ICD mode
	The value of the Volatile long type variable becomes Local Window by the Until command at the blank after the break.	Simulator mode /ICD mode
	When the structure in 1024 bytes or more is displayed to the Watch window, the value of the member variable is not correct occasionally.	ICD mode
	When OK/Apply of SourcePreferences is pressed with the variable of the Watch/Local window edited, the registered variable is not displayed.	Simulator mode /ICD mode
GDB-07	When big size (1024 bytes or more) structure is displayed in the Watch window, the value of the member variable might be not correct.	ICD mode
	If the number of resource steering wheels not opened in RUN increases, and RUN is done for a long time, resource Leake might be generated.	ICD mode

No	Problems	GDB connect mode
	The program is not executed when a current PC address is the same as the address of the until command.	ICD mode
GDB-08	When x command is executed from the Console window with the symbol registered in the Watch window, neither the value nor the address are correctly displayed.	Simulator mode /ICD mode
	When loading it into the flash memory by the load instruction, it doubly writes it at the same address. 。	ICD mode

LIB-01	Corrected portions of ANSI C library function prototypes. <string.h> char *memcpy(/* char *, char *, int */) → void *memcpy(/* char *, char *, int */) ; char *memmove(/* char *, char *, int */) ;→ void *memmove(/* char *, char *, int */) ; char *memset(/* char *, int, int */) ; → void *memset(/* char *, int, int */) ;
	<stdio.h> int sscanf(char *, const char *, ...); → int sscanf(const char *, const char *, ...); int puts(const char *); → int puts(char *); int fputs(const char *, FILE *); → int fputs(char *, FILE *); <ctype.h> int isalnum(char c) ; → int isalnum(int c) ; int isalpha(char c) ; →int isalpha(int c) ; int iscntrl(char c) ; →int iscntrl(int c) ; int isdigit(char c) ; →int isdigit(int c) ; int isgraph(char c) ; →int isgraph(int c) ; int islower(char c) ; →int islower(int c) ; int isprint(char c) ; →int isprint(int c) ; int ispunct(char c) ; →int ispunct(int c) ; int isspace(char c) ; →int isspace(int c) ; int isupper(char c) ; →int isupper(int c) ; int isxdigit(char c) ; →int isxdigit(int c) ; int tolower(char c) ; →int tolower(int c) ; int toupper(char c) ; →int toupper(int c) ;
LIB-02	As for malloc(), securing the heap area might not be able to be corrected.
	Corrected definition part of ANSI C library gmtime(). struct tm *gmtime(time_t *t); →struct tm *gmtime(const time_t *t);

No	Problems
Kanji-01	Filter execution occasionally fails when built from the IDE with the Japanese character filter feature enabled.
Kanji-02	When build is cancelled from the IDE, the Kanji filter process gets interrupted and the source files get deleted. This results in source files(*.c) and filtered files(*.kanji_filt) to be lost.

GNU17 C Compiler Known Issues

The following shows the case of bugs recognized in GNU17 C Compiler.

<table>
<tr><td rowspan="8">No.1</td><td>content of bug</td></tr>
<tr><td>The following compile error occurs, when declaring a huge array(several hundred thousand bytes).

cc1.exe: out of memory allocating mmmmmmmm bytes after a total of nnnnnnnn bytes</td></tr>
<tr><td>workaround</td></tr>
<tr><td>Be small the memory domain which a compiler secures at once by dividing the array and the source code.</td></tr>
<tr><td>reappearance code</td></tr>
<tr><td>unsigned char uc_array[] = { 0x00,0x01, ... };

int main()
{

->The size of array is more than several hundred thousand bytes.</td></tr>
<tr><td>cause</td></tr>
<tr><td>This is the error that the memory domain which the compiler has secured becomes insufficient
at the time of compile.
Because the size of the array without dimension is too large.
The same error may occur when compiling the source file with many lines.</td></tr>
<tr><td rowspan="4">No.2</td><td>content of bug</td></tr>
<tr><td>The result does not become the right value.
Because sign extension of char type variable and addition / subtraction are carried out
at once by optimization.

This bug occurs when all the following conditions are filled.
* First the value which is more than 128(=0x80) is set to the variable which is bigger than char type.
Second substitute the result which addition / subtraction are carried out to this variable for char type variable.
Last substitute the result which addition / subtraction are carried out to this char type variable for the variable which is bigger than char type.
Then the error occurs.
* It is necessary that the result of one of substitution is within 0 - 127.</td></tr>
<tr><td>workaround</td></tr>
<tr><td>Declare volatile to char type variable in order not to sign extension and addition / subtraction
are carried out at once by optimization.</td></tr>
</table>

No.2	reappearance code

	signed int big_type_val ;
	
	int main(void)
	{
	signed char char_val ;
	
	big_type_val = 128 ;
	char_val = big_type_val - 1 ; -- (1)
	big_type_val = char_val - 1 ; -- (2) // big_type_val should be 126, but is -130.

	cause
	It is an error by optimization. The process of (1) & (2) is collected into one and compiled by optimization. For this reason, sign extension and operation are carried out at once. Then the result does not become the right value.
No.3	content of bug
	The result of strcmp() between a Kanji string sequence which is defined by the macro of stringification operator and a Kanji string sequence which is enclosed by double quotation mark does not become equal. Kanji are Japanese characters. The error occurs when Kanji filter is effective. ➔ This bug has been resolved in Ver 1.5.0 or after.
	workaround
	Invalidate Kanji filter. When compiling from a command line, change "CC=xgcc_filt" into "CC=xgcc" in makefile(*.mak). When compiling from IDE, invalidate the item of Kanji filter use in project property.
	reappearance code

	#include <string.h>
	
	#define str(a) #a // macro of stringification operator
	
	int main(void)
	{
	if(strcmp(str("字"), "¥"字¥"")) { // The result of compare should be equal, but is not equal.

No.3	cause
	Kanji filter which changes a Kanji string sequence into ASCII sequence at the time of compile is effective by the default. When using macro of stringification operator, the compare of a Kanji string sequence does not become equal. Because a Kanji string sequence is changed in the order of the following at the time of compile. source code str(″字″) ″¥″字¥″″ Conversion by Kanji filter str(″¥x8e¥x9a″) ″¥″¥x8e¥x9a¥″″ Conversion by preprocessor ″¥″¥¥x8e¥¥x9a¥″″ ″¥″¥x8e¥x9a¥″″
No.4	content of bug
	The following compile error occurs. error: unable to find a register to spill in class This bug may occur when all the following conditions are filled. * Compiled with REGULAR Model or MIDDLE Model. * The pointer argument is passed by %r3 register to a function. * Referencing the pointer argument passed by %r3 register in a function. See the compiler package manual “registers for passing arguments” at ″6.4.3 Method of Using Registers″ about the allocation of registers for passing arguments.
	workaround
	Don’t pass the pointer argument which is cause of the error by %r3 register. So change the order of parameters, or add the dummy argument.

No.4	reappearance code

	void sub(int arg1, int arg2, int arg3, long *arg4)
	{
	static long long int num ;
	
	num = *arg4 ;
	}
	
	※In this case the pointer argument 'arg4' is passed by %r3.
	So for example, add the dummy argument as follows,
	
	void sub(int arg1, int arg2, int arg3, int dummy, long *arg4)
	{
	static long long int num ;
	
	num = *arg4 ;
	}

	cause
	This is compiler internal error when failing to secure registers needed to process.

No.5	content of bug
	Values for global variables in subroutine are not set correctly by inline expansion. This bug occurs when all of the following conditions apply: ・ The address of a global variable is passed as a parameter to a subroutine. ・ A value is set to the global variable via the pointer, which is a subroutine parameter. ・ The subroutine is expanded inline. A number of conditions must be met for inline expansion, as shown below. - An inline statement is added and compiled and optimized to at least -O1 or -O3. - The subroutine definition section precedes the main function. - The subroutine is small. Inline expansion can be checked in Disassembly view by checking whether the subroutine is called.
	workaround
	workaround (1): Declare the global variable (g2 in the example below) with volatile added. workaround (2): Disable inline expansion by placing the subroutine after the main function.
	reappearance code

	int g1, g2;
	int i_Val;
	void write_at (int *addr, int off)
	{
	addr[off] = 1000; // specifies g2.
	}
	int main(void)
	{
	g2 = 12;
	write_at (&g1, &g2 - &g1); // This function is expanded inline.
	i_Val = g2; // i_Val should be 1000 but is actually 12.

	cause
	Error due to optimization.

No.6	content of bug
	The following internal compiler error occurs if a function call is made after casting the immediate value as a function pointer. internal compiler error: Segmentation fault
	workaround
	First substitute the immediate value for the function pointer global value before function calling is made for the global variable.
	reappearance code

	typedef void *(*T)(void);
	void f(void)
	{
	  ((T) 10000000)() // Internal compiler error occurs.
	}
	* This can be avoided in this case by function calling the function pointer global variable as shown below.
	typedef void *(*T)(void);
	T p_Pt; // Function pointer global variable
	void f(void)
	{
	  p_Pt = (void *(*)(void))10000000;
	  p_Pt();
	}

	cause
	Caused by a bug in processing when an immediate value is directly assigned for function calling.

No.7	content of bug
	An illegal assembler instruction is issued if a function call is made after casting the parameter address as a function pointer.
	workaround
	First substitute the parameter for the global variable before assigning the global variable address for making the function call.
	reappearance code

	void f(int x)
	{
	  (*(void (*)())&x)(); // Illegal assembler instruction is generated.
	}
	* This can be avoided in this case by assigning the global variable address to make the function calling as shown below.
	int ip_Pt; // Global variable
	void f(int x)
	{
	  ip_Pt = x;
	  (*(void (*)())&ip_Pt)();
	}

	cause
	Caused by a bug in processing for direct function calling from a parameter address.

No.8	content of bug
	Calculations between subroutines nested in a while() statement conditional expression and local variables are not performed correctly by inline expansion. This bug occurs when all of the following conditions apply: ・ Compiled using optimization exceeding -O1 (gnu17 supports -O3). ・ Subroutine in a while() statement conditional expression is expanded inline. ・ It is nested within at least nine functions. ・ The local variable calculated in the while() statement conditional expression is calculated in the same way in the while() statement block.
	workaround
	Declare by adding volatile to the local variable calculated in the while() statement conditional expression.
	reappearance code

	int f(int x)
	{
	return (x + 1);
	}
	int main(void)
	{
	int a = 1;
	// Calculations performed with local variable a and with 9 f() functions nested in while() statement
	// conditional expression.
	while ((f(f(f(f(f(f(f(f(f(1)))))))))) - a < 10){
	a--; // Same calculation processing as in conditional expression.
	exit (0); // The required exit(0) is not actually executed.
	}
	abort(); // abort() is executed without entering the while() block.

	cause
	Error due to optimization.

GNU17 IDE Known Issues

Below are the known issues of GNU17 IDE.

No.1	content of bug
	Build hangs inside CDT Scanner Config Builder
	workaround
	In Project Properties->C/C++ Make Project->Discovery Options ->Enable generate scanner info command is turned OFF. Please do not turn this ON, otherwise your build might fail in cases.
No.2	content of bug
	File filters for C/C++ Projects view does not work
	workaround
	The filters may not function properly until you have closed and open the project or restart GNU17 IDE.
No.3	content of bug
	In Problems View, Description column sometimes displays a blank
	workaround
	When there is a warning in an included file after a build, the Problems View displays a error mark with a blank in the Description column. The build sequence is successful however, we recommend you to perform a build again after correcting those warnings.
No.4	content of bug
	When saving a file that is currently opened in the IDE from an external editor, the IDE refreshes the shown file from the file system regardless of selecting 'No' in the File Changed dialog.
	workaround
	According to Window->Preferences->General->Workspace->Refresh automatically switch defaults to ON, selecting No gets ignored and the editor refreshes itself. Please refrain from concurrently editing the same file by an external editor and one in the IDE.
No.5	content of bug
	When selecting Window->Reset Perspective, the Outline view may display an error.
	workaround

	You can avoid this by closing the Outline view, double click a new file in the C/C++ Projects view and open that file in the editor area, and selecting Window->Show View->Outline. The Outline view should reopen.
No.6	content of bug
	C/C++Projects view does not show syntax tree(labes) for assembly source files properly.
	workaround
	C/C++Projects view does not show syntax trees for assembly source files. Please note this when referring to the tree display.
No.7	content of bug
	The source window sometimes shows 2 colored lines for the current line.
	workaround
	When a source files is shown in the source window, the colored line which indicates the current line is shown in 2 lines. In this case, please reopen the source file in the editor.
No.8	content of bug
	X icon is not cleared even when there are no errors in the source file.
	workaround
	When an source file with an error gets built, an error mark with an X icon appears in the left side of the editor. Sometimes this icon is not cleared even when the error is eliminated from the source file and rebuilt. In this case, please select Delete C/C++ Markers in the Problems view to clear the error marks and perform a rebuild.
No.9	content of bug
	On Windows Vista, project delete fails when [Delete project contents on disk] is selected in the [Delete Resources]. I get the message, [An exception has been caught while processing the refactoring 'Delete Resource'], and cannot delete the project folder.
	workaround
	When building a project, conime.exe(enables Japanese character input for Command Prompt) starts in the project folder as its current directory, and remains after the build has completed. Therefore, the project folder cannot be deleted. In this case, kill the conime.exe process from the Task Manager, or restart your PC and then delete the project folder.