Bugs Open as of Dynamic C 7.04P3

Reference Number	**Description**	**Work-Around**	**Version(s) Affected**
6	Run-time math exceptions in watch expressions cause the target program to crash when debugging.	None, except don’t evaluate floating point watch expressions with bad domain arguments.	6.04-current
21	After using the print preview option with Dynamic C in full-screen mode, the taskbar will no longer automatically pop up (if set to "auto hide") until Dynamic C is exited.	Avoid using print preview if you prefer to keep your task bar as “autohide.”	6.04-current

42	Compiler decrements the address of month, rather than the value stored in hl.
	int dom[12] ={31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
	void main(){
	  auto char month;
	  unsigned int day;
	  day = 0;
	  month = 0;
	  // only a problem if month is an auto char.
	  // hl is not restored after the increment
	  // address of month (-1) gets added as the
	  // offset (rather than the value of month).
	  day = dom[month++];
	  // Pre-increment is also incorrect.
	  day = dom[++month];
	}
		To avoid this, either do not declare the char as auto, or do the incrementing before or after the array access.	7.02P-current
			fixed in a pending release

50	djnz not decrementing b register when single stepping if there is no code between the djnz and the jump destination. Example: #asm ld b, 2 wait: djnzwait #endasm	Add a NOP or don’t try to single step this busy wait code: #asm ld b, 2 wait: nop djnzwait #endasm	6.04-current
100	No error warning is given for indirect calls to cofunctions at compile time.	None	6.04-current fixed in a pending release

Bugs Open as of Dynamic C 7.04P3

Reference Number	Description	Work-Around	Version(s) Affected

105	Incorrect handling of float constants at compile time causes crash.
	For example:
	main(){
	unsigned long c;c = 2e32 - 1;
	}
		do it like this:
		main(){
		float f;unsigned long c;
		f = 2e32 - 1;
		c = (unsigned long)f;
		}
			6.04-current
			fixed in a pending release

148	When tabbing through the Function Lookup/Insert window selections after the mode radio buttons, the parameter # from the "Insert Call" mode can also be seen and set from the "View Only" mode.	None	6.04 - current fixed in a pending release

151	The initializers in the following code are semantically equivalent. The expression for p1 compiles correctly (assuming bug fixes apply for defects #113-115). The expression for p2 generates a compile time error.
	struct foo {int x; int y;} foovar;
	main() {
	int *p1= &((*(&foovar)).x);
	int *p2 = &((&foovar)->x);
	}
		None	6.04-current
			fixed in a pending release

160	Constant folding does not work properly for expressions that cast int* to an integral type and then add. This defect therefore causes initializers to constant data to be evaluated incorrectly and misaligned
	int z =10;;
	main() {
	(long)&z+1; // compiler should evaluate to constant value bug instead generates call to L_add
	(long)((char*)&z + 1); // semantically equivalent expression that folds correctly
	(char)&z+1; // compiler should evaluate to constant value bug instead generates call to L_add
	(char)((char*)&z + 1); // semantically equivalent expression that folds correctly
	}
		None	6.04-current

Bugs Open as of Dynamic C 7.04P3

Reference Number	Description	Work-Around	Version(s) Affected

161	In the following program, a function address/pointer is used in arithmetic expressions and it compiles. This usage should be flagged as an error by the compiler.
	
	In the generated code for the initializers, only the address is generated and the arithmetic is ignored. The increment treats it as though the type where int*.
	
	`int foo();`
	`main() {`
	`int x = foo + 5; // no error reported`
	`int (*fp)() = 5+foo+3; // no error reported`
	`int (*fp2)() = foo;`
	`int (*fp3)();`
	`fp3 = foo;`
	`fp3++; // no error reported`
	`}`
	`foo() {return 10;}`
		None	6.04- current
			fixed in a pending release

168	The following program does not compile, but should.
	`int (*fp)();`
	`main() {`
	`   fp = &main; // won't compile`
	`   fp = main;    // work-around`
	`}`
		The ‘&’ is redundant here anyway, since main is evaluated as an address	6.04-current
			fixed in a pending release

169	The address of operator '&' causes compilation errors when used with arrays.
	`main() {`
	`   int ia[10];`
	`   int (*pa)[10]; // pointer to an array of int`
	`   int *p;`
	`   // "p = &ia" does not compile, but should with a warning`
	`   // since "pointer to int" and "pointer to array of int" don't match`
	`   p = &ia;`
	`   p = *&ia; // is semantically equivalent to next line, but will not compile`
	`   p = ia;`
	`}`
		None	6.04-current
			fixed in a pending release

Bugs Open as of Dynamic C 7.04P3

Reference Number	**Description**	**Work-Around**	**Version(s) Affected**

194	Given a label, if a space exists between colon and label, DC will give error.
	main(){
	goto label_a;
	//label_a : // will not work
	label_a: // works
	}
		None	6.04-current
			fixed in a pending release

207	Because no warning is generated when a global assembly label is redefined, a user application can inadvertently override library code or data.	It is a good idea not to use short global label names.	6.04-current

210	// calling a function with more than one
	// indirect function call as an argument fails
	// if the function pointed to takes arguments
	int intfunc(int x);
	typedef int (*func2)();
	main(){
	func2 fp1,fp2;
	fp1 = fp2 = intfunc;
	foo((*fp1)(1),(*fp2)(2));
	}
	int intfunc(int x){}
	foo(int a, int b){}
		None	6.04-current
			fixed in a pending release

220	The code for BitWrPortI does not protect against a race condition with an ISR that is updating the same register. This can be worked around by blocking interrupts during calls to BitWrPortI()	None	6.04-current fixed in a pending release

226	The following program generates "internal error: invalid register store"
	main() {
	   auto word sequence_mode;
	   auto int debug_on;
	   sequence_mode=debug_on=0;
	}
		sequence_mode=0;
		debug_on=0;
			6.53-current
			fixed in a pending release

229	The Print Options have a default margin of 1" but will actually about print a 0.3" margin.	None	6.04-current fixed in a pending release

Bugs Open as of Dynamic C 7.04P3

Reference Number	Description	Work-Around	Version(s) Affected

232	Sizeof operator behaves incorrectly with structs, typedefs and unions.
	typedef struct {
	  int a;
	  int b;
	  char buf[10];
	} rec;
	char byteaccessible[sizeof(rec)];
	main() {
	}
		None	6.04-current
			fixed in a pending release

237	Print Preview artifact 1. Open a sample program. 2. Open print preview. 3. Click on the printer icon. 4. Open properties and adjust any of the parameters. 5. Accept the new parameters and close the properties window. 6. A print preview artifact remains.	To clear the artifact grab the Dynamic C window from behind the print preview artifact. Position the Dynamic C window so that you can click the "print preview" icon on Dynamic C. An error message, "Exception #32739 (no message abailable). Ok to resume?", will appear and click "Yes". Only make adjustments to the print properties OUTSIDE of the print preview screen (i.e. File->Print->Properties).	6.04-current

238	The runwatch mechanism takes snapshots of each watch in the watch list at different points in time.
	The compiler/libraries should take a single snapshot of the variables in the watch list and update the display
	from a single point in time. Adding watches for x and y in the following program demonstrates the problem.
	main() {
	  int x,y;
	  x = y;
	  while(1) {
	    x++; y++;
	    runwatch();
	  }
	}
		None	6.04-current

Bugs Open as of Dynamic C 7.04P3

Reference Number	Description	Work-Around	Version(s) Affected
246	The expression (x--) is not evaluated correctly if x is an auto char. The error only happens if in functions that are not "useix".	make the function using the expression useix or make x static or make x an int	6.04-current fixed in a pending release
249	Incorrect parameter type causes DC to crash HttpSpec http_flashspec[] = { HTTPSPEC_FILE, "/", index_html, NULL, 0, NULL, &view}, // Remove ampersand to cause crash	None	7.02P - current
253	If help on a function is obtained via Control-H and then Insert Call is selected the source file location gets corrupted.	None	6.57-current fixed in a pending release

263	The printf line should generate an error (due to the extra semicolon), but instead the compiler passes the value of i to printf.
	void main()
	{
	int i;
	i = 0;
	printf("i = %d\n", i++;);}
		None	7.02P - current

266	local data in cofunctions limited to 128 bytes	use global data	6.04-current
272	Dynamic C crashes when loading initial loader window is stopped prematurely 1. Open Dynamic C and close the "Loading Initial Loader" window as soon as it appears. 2. Open a sample and compile. Again close the "Loading Initial Loader" window as soon as it appears. 3. GPF	None	6.04 - current fixed in a pending release
274	Exceptions: 230, 231, 232, 239, 242 are not used but are documented. Exception 241 is not implemented but should be exceptions: 245, 246, 247, 248 are used but not documented. Exception 255 is undocumented and thrown for multiple reasons non fatal exceptions are thrown from several places also This is both a documetation defect and a bug	None	6.57-current fixed in a pending release

Bugs Open as of Dynamic C 7.04P3

Reference Number	Description	Work-Around	Version(s) Affected
279	A missing EndHeader line in PPPOE.LIB causes the compiler to crash.	None	7.02P-current fixed in a pending release
282	A do-while bug has been uncovered in DCRTCP.LIB, in the function packdom(). packdom() recently changed, which has triggered this bug. A bad jump is generated for the end of the do-while statement.	None	6.52-current fixed in a pending release
291	The RS232 function serXwrite() will block until all or the data to be written is copied on to the port buffer.	None	6.04-current

294	Dynamic C crashes when a program is compiled which "#use's a library which contains the following:
	
	/*** BeginHeader***/
	int MyErrors[6];
	int s;
	void InitErrorCodes (void) {
	s = sizeof(MyErrors);
	}
	/*** EndHeader */
		None	6.04-current
			fixed in a pending release

297	when you INSERT(not append) a block of code and you decide to undo your changes, the undo/redo command "picks up" an extra line of code.	None	6.04-current fixed in a pending release
299	If the harddisk fills up while you are editing a file, and you try and save your work, DynamicC will ignore the problem, and not give any indication that it didn't save your work.	None	6.04-current fixed in a pending release
301	The comments in coremodule Keylcd.c indicate that the user connect PA4..PA7 to D0..D3 of the LCD. This is not correct for any LCD which is compatible with the HD44780. The data lines should be connected to D4..D7 when using the 4 bit programming mode.	None	6.04-current fixed in a pending release

Bugs Open as of Dynamic C 7.04P3

Reference Number	Description	Work-Around	Version(s) Affected

304	ldp (ix),hl and ex af,af' do not expand in multiline assembly macros
	#define MAC $\\
	ex af,af' $\\
	ldp (ix),hl $\\
	nop
	main(){
	;
	#asm
	MAC
	#endasm
	}
	this expands to just nop. If no blank space precedes any instruction in the multi-line assembly macro definition, compilation fails.
		None	6.04-current
			fixed in a pending release

310	There is a low limit on the number of characters of data which can appear in the watch window. This limit did not previously exist.	None	6.55 - current fixed in a pending release
311	RS232 - opening with baud rate 0 causes divide by zero error	None	6.52 - current fixed in a pending release

312	Assigning the return value of of an indirect call to an auto long breaks the indirect call.
	long (*fptr)();
	long foo(unsigned x);
	main(){
	auto unsigned long x; // take away the auto or the long and this works
	fptr = foo;
	printf ("x=%08lx\n",x = (*fptr)(10));
	}
	long foo(unsigned x){
	return 0x100ul * x;
	}
		None	6.04 - current
			fixed in a pending release

Bugs Open as of Dynamic C 7.04P3

Reference Number	Description	Work-Around	Version(s) Affected
313	The timeout functions set_timeout() and chk_timeout() are implemented incorrectly in DCRTCP.LIB. They are based on MS_TIMER. set_timeout() adds the requested number of seconds to MS_TIMER. chk_timeout() then compares MS_TIMER to the given timeout value. This means that if, when the number of seconds is added to MS_TIMER, this causes the 32-bit unsigned value to roll over, then the next chk_timeout will trigger the timeout early (as long as MS_TIMER has also not yet rolled over). With TCP/IP connections, this could cause prematurely dropped connections around the rollover point, which will occur every 49.7 days. These same timeout functions are used throughout the TCP/IP libraries (such as HTTP.LIB), so similar problems could occur elsewhere.	None	6.51 - current

314	Calling a function via a function pointer and assigning the return value to a dereferenced pointer does not work correctly. Splitting it out so that assig;b(o)-2-12.6(o)-261.501(iv)0o

Bugs Open as of Dynamic C 7.04P3

Reference Number	Description	Work-Around	Version(s) Affected
321	1: Run "COF ECHOBLK.C" from the serial folder with the following defined: #class static #memmap xmem. 2: Once the program is compiled do not enter data into the serial window. After 20 seconds a "Timed Out" message should be displayed. Instead a "Ti" message is displayed. The rest of the message is lost. The sample is not affected in any other way.	None	7.02-current
323	If you delete the user defined BIOS file name from the compiler options dialog, but leave the "Use" check box checked, DC be crashes on BIOS compile.	None	6.04 - current fixed in a pending release
325	If you compile to a file with the "Include debug code..." option unchecked, then compile to target with option checked, it compiles without debug code the first time.	Compile twice	7.02 - current fixed in a pending release
329	1: Using RabbitLink run " Cof EchoBlk.c" from the Samples\Serial folder and Dynamic C will display an error as follows: "Target communication state: Compiling User Program Error receiving write acknowledgment" 2: Attempt to recompile the sample and Dynamic C will crash with an "Abnormal Termination" message. 3: No data or parameters will be saved.	None	7.02-current fixed in a pending release
330	Using fshift() can cause a lookup table in RAM to be corrupted. This prevents other files from being opened until the filesystem is reformatted or the program restarts.	None	7.02 - current

331	Optimization bug with using the <= operator as shown below.
	short nI, mI;
	main() {
	  nI= 16; mI= 1;
	  if (nI <= 0 && mI <= 0) {
	      // shouldn't enter here but does
	    }
	}
		add parenthesis
		
		if ((nI <= 0) && (mI <= 0)) {
			6.19 - current
			fixed in a pending release

Bugs Open as of Dynamic C 7.04P3

Reference Number	Description	Work-Around	Version(s) Affected
333	The following piece of code, although incorrect, crashes the compiler instead of generating an error: `typedef struct foo foostruct;` `foostruct foolist = { };` `main(){}` If the 'struct' keyword is removed OR 'foolist' is not initialized, the proper errors show up instead.	None	6.04 - current fixed in a pending release
334	The LCD driver appends two bytes to a message which has been previously declared via char[] = "text";	None	6.04-current fixed in a pending release
337	The "Include debug code/RST 28 Instruction" is useless when compiling via TCP/IP. Although the program is successfully compiled, Dynamic C will not start running it and the "disconnect and press reset to run" message doesn't really help the typical RabbitLink user, especially since between the lack of RST28s and the necessary reset, console communications between the RabbitLink and the target will be lost.	None	7.03-current fixed in a pending release
346	There is a C conversion programming error in fs_block_init in fs_flash.lib which cause a sector boundary check to fail.	None	7.02-current fixed in a pending release
348	Declarative expressions inside a cast will be compiled, when they should generate an error.	None	6.04-current fixed in a pending release

Bugs Open as of Dynamic C 7.04P3

Reference Number	Description	Work-Around	Version(s) Affected

349	Assume a variable is externed in one library (the extern is in a BeginHeader/EndHeader block), then defined outside the Begin-Header/EndHeader block in the same library. If the address of that variable is used in another library, the address is not assigned correctly. Instead, it is assigned as the address 0x0000.	Define the variable directly in the Begin-Header/EndHeader block instead of using extern. or extern the void pointer as below
		/*** BeginHeader barfunc */
		void barfunc(void);
		extern void* const bardata;
		
		/*** EndHeader */
		void* const bardata = &(foodata);
		
		void barfunc(void)
		{
		printf("Entered bar-func()\n");
		}
			7.03 - current
			fixed in a pending release

355	RFU can overwrite the ID block	None	6.53-current
357	In FLASHWR.LIB the structure FlashData is declared in the header as _FlashData(). There should be no underscore.	Remove the under-score	7.02-current fixed in a pending release
358	printf hex format always prints %x in caps (3AE instead of 3ae) and does not recognize %X at all.	None	6.04-current fixed in a pending release

359	Trying to compile the following program makes Dynamic C crash:
	
	#use "dcrtcp.lib"
	void SendMessage(mssg)
	{
	tcp_Socket tcpSock;
	sock_write(&tcpSock, mssg, strlen(mssg)+1);
	}
	main()
	{
	while(1){
	}
		None	7.02-current
			fixed in a pending release

Bugs Open as of Dynamic C 7.04P3

Reference Number	Description	Work-Around	Version(s) Affected
360	Can not use 33+ character functions in BeginHeader/EndHeader blocks	Shorten the function name to 32 or fewer characters.	6.04-current
361	If the watch expression list is empty and watches are added, you have to close the watch expression window and reopen it before you can delete them. In other words, when you add a watch expression to an empty list, the delete button should become active.	None	7.02-current fixed in a pending release
370	The result of x%1 gave incorrect result of x.	None	6.57-current fixed in a pending release
376	OP6700: Bug in dispContrast function. The dispContrast function does not work for many values. The values 45..60 put garbage or a blank screen.	None	6.57-current

377	This program causes a GPF because of the [] on bananas.
	
	#define MAX_CARDS 100
	typedef struct
	{
	    unsigned long stuff;
	    unsigned long morestuff;
	    unsigned long hello;
	}mystruct;
	mystruct bananas[MAX_CARDS];
	mystruct apples[MAX_CARDS];
	mystruct *Apples[]={bananas[]};// DC won't like this
	
	void main()
	{
	    printf("Hello");
	}
		Do not use this incorrect syntax	7.02-current
			fixed in a pending release

381	Exceeding 249 lines on an editor line confuses the debugger when setting break points.	Don't make the line so long.	6.04-current

Bugs Open as of Dynamic C 7.04P3

Reference Number	Description	Work-Around	Version(s) Affected
382	Inspecting a long variable via RabbitLink does not show the correct value The lower 16 bits seem to be OK but the upper 16 bits are not correct.	None	7.03P - current
384	The watch window assumes that a root function called from an xmem function is also in xmem -- the address it uses to access a passed variable is off by one byte (as if the XPC value was pushed as well).	None	6.04 - current

Bugs Open as of Dynamic C 7.04P3

Reference Number	Description	Work-Around	Version(s) Affected

388	You can not allocate more than 32k of socket buffers, even if the RAM is available.	Change the following lines:
		#ifdef DISABLE_DNS
		_sock_buf_area = xalloc(MAX_SOCKETS * SOCK_BUF_SIZE);
		for (r = 0; r < MAX_SOCKETS; r++) {
		#else
		_sock_buf_area = xalloc((MAX_SOCKETS + 1) * SOCK_BUF_SIZE);
		for (r = 0; r < (MAX_SOCKETS + 1); r++) {
		#endif
		
		to this:
		#ifdef DISABLE_DNS
		_sock_buf_area = xalloc(MAX_SOCKETS * (long)SOCK_BUF_SIZE);
		for (r = 0; r < MAX_SOCKETS; r++) {
		#else
		_sock_buf_area = xalloc((MAX_SOCKETS + 1) * (long)SOCK_BUF_SIZE);
		for (r = 0; r < (MAX_SOCKETS + 1); r++) {
		#endif
			6.57 - current

Bugs Open as of Dynamic C 7.04P3

Reference Number	Description	Work-Around	Version(s) Affected
398	Disconnecting (or disabling polling) while a transfer of the printf buffer from the target to Dynamic C is occurring will lock up the RabbitLink while it waits endlessly for the transfer to continue. This can also occur when F4 is pressed.	Don't use printfs.	7.03P - current
399	Execution cursor not updated in library source file when single stepping through pure assembly function.	None	6.04 - current