Software Development Products

Product Errata

Intel® C++ Compiler for Linux* and Windows*

10th February 2003

Number of entries − 69

DISCLAIMER

Linux*

28571	Debugger cannot display the contents of virtual base classes correctly
27115	Size of shared lib created using icc is ~100 times larger than that created using gcc
26994	Profile Guide Optimization ineffectual on files containing inline assembly
27428	Link warning when building code with _asm directives and a shared library
27633	libraries built using the −g swtich are enormous
26850	The std::cos function behaves like std::sin
27478	Error caused by __attribute__((__unused__))
29341	The Itanium(R) compiler hangs compiling lame MP3 encoder at −O2 optimization level
27807	Linux* C++ Compiler for Itanium(R) Architecture requires −prof_gen to link
27919	The port@host syntax is not supported when ICID determines location of the license file
26927	ios_base::ate does not work properly with ostringstream
28049	Scope of types wrong in C++ debug information
28512	GNU gcc source compatibility issue: some functions not defined in the std namespace
28513	Intel C++ Compiler defines _STD macro, causes conflicts
28784	Cannot make a symbol "weak"
28946	ICC does not initialize static constant members within the class scope
29661	A volatile void pointer parameter dereferenced in a function causes segmentation fault
28766	Errors with /usr/include/linux/byteorder/swab.h

Windows*

18282	Derived class enumeration bug
19189	No option to turn off "targeted for automatic cpu dispatch" remarks.
20109	printf invalid format string: warning #269
21906	Itanium(R) compiler unaligned access error with __unaligned keyword used on a structure
22666	Compiler does not support double casting
22910	Dependency information not generated under certain circumstances
23572	Large value (>=0x80000000) cast from double to int produces incorrect results
24227	The C++ Compiler compiles an illegal member−like constructor call
22544	Native 64−bit EDB cannot see processes
25485	Compiler issues error message when compiling in−line asm code using __asm_emit
25664	Default Include Directory path does not change back to Microsoft* Visual C++* .NET compiler
25728	Alignment not carried over to derived class

26318	Runtime errors when using Stingray* Objective Studio 2000 with the C++ Compiler for Windows*
26058	Incorrect handling of __declspec(dllexport) in static functions
26167	Intel C++ Compiler 6.0 installation does not complete when using Netscape* 6.2 as default browser
26177	2+ GB addresses with /3GB cause runtime problems
26131	Can't find how to select/deselect the Intel Compiler in Microsoft* Visual C++* .NET
27053	Compiler fails to destroy objects processing nested exception handling code
26137	threadprivate OpenMP pragmas ignored
26319	Handling of char * comparisons incorrect
27874	"Update User's Registry" tool and Admin rights
28335	Templates with /Zi
28272	icl ignores the Using−declaration of base class assignment operator
28297	/Qoption,c,−ip_ninl_min_stats incorrect
28342	linker directive −export not generated in presence of _declspec(dllimport)
29035	Visual Studio .NET crashes when selecting properties of multiple projects
29228	UUID not allowed on enumerated types
29660	Integration tool causes link error when Qipo active &project dependencies exist
29344	The program prints incorrect results when compiled with the /QxK or /QxW switches
29945	No warning message issued when 'delete' is called on a local array

Linux*

Reference #	Product	Version	Operating System	Title	Last Update
21517	Intel(R) C++ Compiler for Linux*	6.0, 7.0	Red Hat* 7.0	Template function with too few arguments is accepted	28−Oct−02

Symptom	The following program compiles without any errors, although the function "wibble" can never be called with only one argument. Also, the default arguments from the function "bar" are taken over into the call of "wibble", which is incorrect.
	
	template<typename F>
	void foo(F f)
	{
	f(1);
	}
	
	void bar(int i, int j = 5, int k = 7)
	{
	std::cout << "bar(" << i << ", " << j << ", " << k << ")" << std::endl;
	}				

```
void wibble(int i, int j, int k = 9)
{
std::cout << "wibble(" << i << ", " << j << ", " << k << ")" << std::endl;
}

int main()
{
foo(
foo( // should give an error, "wibble" cannot be called with one arg
}
```

Current Status/Solution

This is a known issue that may be resolved in a future product release.

Reference #	Product	Version	Operating System	Title	Last Update
21755	Intel(R) C++ Compiler for Linux*	6.0	Red Hat* 7.0	The compiler does not issue a warning for the lack of include file for the malloc function	30−Sep−02

Symptom

The C++ compiler does not issue a warning message if a program calls the "malloc()" function without including the malloc.h or stdlib.h header files.

Current Status/Solution

This problem has been resolved in a product update with package ID l_cc_pu_7.0.068 or higher. You may download and install the latest product update from the premier support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support

Reference #	Product	Version	Operating System	Title	Last Update
22064	Intel(R) C++ Compiler for Linux*	6.0, 7.0	Red Hat* 7.1	FlexLM* cannot find license file if directory path contains @	3−Oct−02

Symptom

If the Intel(R) C++ Compiler for Linux* is installed into a directory path that contains '@' (for example, /opt/@cell/compiler), FlexLM* cannot find the license file.

Current Status/Solution

This is a known issue that may be resolved in a future product release. As a workaround remove the '@' symbol from the directory path.

Reference #	Product	Version	Operating System	Title	Last Update
23192	Intel(R) C++ Compiler for Linux*	6.0, 7.0	Red Hat* 7.1	Compile errors when compiling pre−processed source files generated by GNU C++	29−Oct−02

Symptom

Compiling a GNU C++ pre−processed source file with icc generates numerous "undefined identifier" errors with headers in
/usr/include/g++−3/ such as streambuf.h and iostream.h:

	"/usr/include/g++−3/streambuf.h", line 339: error: identifier "__null" is undefined int have_backup() { return _IO_save_base != __null; } There is also an error generated for /usr/lib/gcc−lib/i386−redhatlinux/2.96/include/stdarg.h: "/usr/lib/gcc−lib/i386−redhat−linux/2.96/include/stdarg.h", line 43: error: identifier "__builtin_va_list" is undefined typedef __builtin_va_list __gnuc_va_list; g++ will compile this normally
Current Status/Solution Because the pre−processed source file was generated by the GNU C++ compiler (g++), gcc/g++ headers were included. Currently icc does not compile all of the gcc/g++ headers. This is a known issue that may be resolved in a future product release.	

Reference #	**Product**	**Version**	**Operating System**	**Title**	**Last Update**
23916	Intel(R) C++ Compiler for Linux*	6.0, 7.0	Red Hat* 7.1	Double data types are 80 bits rather than 64 bits − no way to get 64−bit double data types	28−Oct−02
Symptom	The switch −long_double is on by default, enabling 80−bit long double type. There is no way to disable this switch and get 64−bit double type.				
Current Status/Solution This is a known issue that may be resolved in a future product release.					

Reference #	**Product**	**Version**	**Operating System**	**Title**	**Last Update**
24094	Intel(R) C++ Compiler for Linux*	E5,0,E6.0,P5.0,P6.0	Red Hat* 7.1	Problem with object references in constructor	26−Nov−02
Symptom	The methods and data members of an object should be accessible from the constructor of an object. ecc produces SEGV and incorrect results for various references to object methods and data members in the object constructor.				
Current Status/Solution This is a known issue that may be resolved in a future product release.					

Reference #	**Product**	**Version**	**Operating System**	**Title**	**Last Update**
25004	Intel(R) C++ Compiler for Linux*	6.0, 7.0	Red Hat* 7.1	Loop with a switch statement is not auto−parallelized	27−Sep−02
Symptom	A loop containing a switch statement is not auto−parallelized, when compiling with the −parallel option. An example is provided below.				

```
main() {
int i;
for (i=0; i<50; i++) {
switch(i){}
}

}

icc −parallel −par_report3 test.c
test.c
procedure: main
serial loop: line 3: not a parallel candidate due to the loop being lexically discontinuous
```

Current Status/Solution

This is a known issue that may be resolved in a future product release.

Reference #	Product	Version	Operating System	Title	Last Update
25005	Intel(R) C++ Compiler for Linux*	6.0, 7.0	Red Hat* 7.1	Loop with a conditional operator(?:) is not parallelized	27−Sep−02

Symptom

If conditional operator (?:) appears in a loop, the loop is not auto−parallelized.

```
#include <stdio.h>
main() {
int j;
int a[100];
for (j=0; j<50; j++) {
a[j]=(j>25)? 1 : 0;
}
printf("%d\n",a[44]);

}

icc −parallel −par_report2 test.c
test.c
procedure: main
serial loop: line 5: not a parallel candidate due to unknown reasons
serial loop: line 5
output data dependence assumed from line 6 to line 6
output data dependence assumed from line 6 to line 6
flow data dependence assumed from line 6 to line 6
output data dependence assumed from line 6 to line 6
output data dependence assumed from line 6 to line 6
flow data dependence assumed from line 6 to line 6
anti data dependence assumed from line 6 to line 6
anti data dependence assumed from line 6 to line 6
```

Current Status/Solution

This is a known issue that may be resolved in a future product release.

Reference #	Product	Version	Operating System	Title	Last Update
25430	Intel(R) C++ Compiler for Linux*	6.0, 7.0	Red Hat* 6.2	C++ compiler incompatible with SGI* hash_set, hash_map implementation	20−Nov−02
Symptom	Code written to use the hash_map and hash_set interfaces provided by GNU* and SGI* C++ compilers does not compile with the Intel C++ compiler.				

Current Status/Solution

The hash_set and hash_map classes are not part of the C++ standard library. Therefore, there are multiple conflicting implementations. The Intel(R) C++ Compiler uses the Dinkumware* implementation. To compile under the Intel C++ compiler, use the interface provided by Dinkumware*. This interface is described in documentation available at their web site at http://www.dinkumware.com/htm_cpl/hash_map.html.

Reference #	Product	Version	Operating System	Title	Last Update
25416	Intel(R) C++ Compiler for Linux*	6.0, 7.0	Red Hat* 7.1	Compiler reports incorrect error message when compiling with the invalid switch −Xk	22−Nov−02

Symptom

If you use the invalid compiler switch "−Xk" to compile a simple C program containing a reference to the stdio.h header file, the compiler issues the following invalid error messages:

```
icc −c −Xk ers.c

catastrophic error: #error directive: "You need a
ISO C conforming compiler to use the glibc headers"
# error "You need a ISO C conforming compiler to use the glibc headers"

error: expected a ";" typedef signed char __int8_t;

error: invalid combination of type specifiers
__extension__ typedef signed long long int __int16_t;

error: invalid combination of type specifiers
__extension__ typedef signed long long int __int32_t;
^
error: invalid combination of type specifiers
__extension__ typedef signed long long int __int64_t;


"/usr/include/gconv.h", line 71: error: identifier "const" is undefined
__const unsigned char **, __const unsigned char *,
^
```

Current Status/Solution

This is a known issue that may be resolved in a future product release. The correct compiler switch to use is −xK (lower case x and capital K).

Reference #	**Product**	**Version**	**Operating System**	**Title**	**Last Update**
25001	Intel(R) C++ Compiler for Linux*	6.0	Red Hat* 7.0	eccfilt/iccfilt utilities produce incorrect results	6−Dec−02
Symptom	The iccfilt/eccfilt utilities are C++ name demanglers utilities provided with the Intel C++ Compiler for Linux. The syntax is as follows: $ echo _ZN10NewShareme3fooEv | iccfilt The iccfilt/eccfilt utilities report the symbol in mangled form rather than demangled form.				
Current Status/Solution This problem has been resolved in the 7.0 compiler product with package ID l_cc_p_7.0.073 or higher. You may download and install the latest product update from the premier support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support					

Reference #	**Product**	**Version**	**Operating System**	**Title**	**Last Update**
26085	Intel(R) C++ Compiler for Linux*	P6.0	Red Hat* 7.1	The icc compiler generates shared object files that are 3 to 6 times larger than gcc on STL code.	6−Dec−02
Symptom	When code that make heavy use of STL is compiled to produce a shared object, the code size of the shared object is three to six times bigger than that produced by gcc.				
Current Status/Solution This is a known issue that may be resolved in a future product release. As a workaround do not use shared objects and use static linking.					

Reference #	**Product**	**Version**	**Operating System**	**Title**	**Last Update**
26654	Intel(R) C++ Compiler for Linux*	6.0, 7.0	Caldera*	Configuration scripts fail to setup man path; man −w does not work	28−Oct−02
Symptom	The MANPATH variable is required to display the man page for the Intel Compiler. If MANPATH is not defined when the configuration script executes, the script uses the "man −w" command to acquire the location of the man files. Some versions of the man function do not recognize the "−w" switch, ignoring the switch and displaying the prompt "What manual page do you want?". This causes the script to terminate prematurely, without setting the MANPATH variable. This is known to be a problem with man 2.3.19, which has come with SuSE* Linux* 7.3.				
Current Status/Solution This is a known issue that may be resolved in a future product release.					

Reference #	Product	Version	Operating System	Title	Last Update
26420	Intel(R) C++ Compiler for Linux*	6.0,7.0	Red Hat* 7.2	gprof is unable to find call graph data when profiling with ecc	28−Oct−02

Symptom

The call graph functionality is not completely working with ecc.

```
$ ecc −p x.c y.c −o x
$ ./x
$ gprof x
gprof: gmon.out file is missing call−graph data
```

Current Status/Solution

This is a known issue that may be resolved in a future product release. As a workaround, first compile the source module(s) into assembly modules, then complete the build from the assembly modules:

```
$ ecc −p −S x.c
$ ecc −p −S y.c
$ ecc −p x.s y.s −o x
$ ./x
$ gprof x
```

Reference #	Product	Version	Operating System	Title	Last Update
26298	Intel(R) C++ Compiler for Linux*	6.0	Caldera*	LDB assertion failure	30−Oct−02

Symptom

ldb crashes at startup on Red Hat* Linux* 7.2 and 7.3.

```
$ ldb ./bin/qmake
Linux Application Debugger for 32−bit applications, Version 6.0, Build 20020325.
Copyright (c) 2001−2002 Intel Corporation. All Rights Reserved.

Assertion failure: low_pc, file ../../src/ST/intel/iArange.C, line 7
$
```

Current Status/Solution

This problem has been resolved in the Intel(R) C++ Compiler 7.0. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support.

Reference #	Product	Version	Operating System	Title	Last Update
25543	Intel(R) C++ Compiler for Linux*	6.0	Red Hat* 7.2	Virtual base classes seem to be broken in Intel(R) C++ Compiler Version 6.0	22−Nov−02

Symptom

Virtual base classes (at least destructors) don't work with Intel(R) C++ Compiler version 6.0. The following test program when compiled with icc gives segmentation fault at run time.

```
class base
{
```

```
public:
virtual ~base() {}
private:
int m_base;
};

class c1 : virtual public base{};
int
main(int, char**)
{
base* abase = new c1();
delete abase;
}
```

Current Status/Solution

This problem has been resolved in a product update with package ID l_cc_pu_6.0.166 or higher. You may download and install the latest product update from the premier support web site at https://premier.intel.com.

Reference #	Product	Version	Operating System	Title	Last Update
26702	Intel(R) C++ Compiler for Linux*	6.0, 7.0	Red Hat* 7.2	Cannot do multi−file IPO when using a library.	29−Oct−02

Symptom

If code is compiled separately (with −ipo) and placed in a library, the compiler will not perform any multi−file optimization when a program links against the library (and is itself also compiled with −ipo). Instead, everything degrades to single−file optimization.

The following example illustrates this problem and shows the diagnostic message output by the compiler:

Compile main.cpp by itself with −ipo, use xild to put the resulting object in a static library, and then link with ecc and only single−file optimizations are indicated:

```
$ ecc −c −ipo main.cpp
main.cpp

$ xild −lib cru mylib.a main.o
IPO: using IR for main.o
IPO: performing single−file optimizations
xiar: executing 'ar'

$ ecc −ipo shrtvect.cpp mylib.a
IPO: using IR for shrtvect.o
IPO: performing single−file optimizations
```

Current Status/Solution

This is a known issue that may be resolved in a future product release.

Reference #	Product	Version	Operating System	Title	Last Update

26591	Intel(R) C++ Compiler for Linux*	6.0	Caldera*	Program produces incorrect results when compiled with the −xM compiler switch	30−Sep−02

Symptom

The following program produces incorrect results when it's compiled with the −xM compiler switch to vectorize the loops. The code produces correct results when vectorization is disabled using the "−xM −vec−" compiler switches.

icc: Intel(R) C++ Compiler for 32−bit applications, Version 6.0 Build 020426Z
Code:

```
––––––––––––––––––––––––––––

#include <stdio.h>
#include <stdlib.h>

static int f(int k)
{
static int n = 0; /* number of elements in array v[] */
static int * v = NULL; /* array of size n */
int i;

if (k > n)
{
i = n;
n = k;
v = (int *) realloc(v, n * sizeof(*v));
for(;i < n;i++) /* no bug if for(i=0; i v[i]=i;
}

printf("Now:\n");
for(i=0; i < n; i++)
printf("v[%d]=%d\n", i, v[i]);

return 0;
}

int main(int argc, char ** argv)
{
return f( argc); /* to avoid constant propagation */
}
–––––––––––––––––––––––––––––––
```

Problem:
call to f(3) should print "0", "1", "2"

However:

```
>icc −xM bug3.c // produces "0", "0", "0" as seen below
bug3.c
bug3.c(15) : (col. 7) remark: LOOP WAS VECTORIZED.
>./a.out a a
Now:
v[0]=0
v[1]=0
v[2]=0

icc −xM −vec− bug3.c // Disabling the vectorizer produces correct results
bug3.c
>./a.out a a
Now:
```

```
v[0]=0
v[1]=1
v[2]=2

>gcc−3.1 bug3.c // The GNU gcc compiler also produces the correct results
>./a.out a a
Now:
v[0]=0
v[1]=1
v[2]=2
```

Current Status/Solution

This problem has been resolved in a product update with package ID l_cc_pu_6.0.166 or higher. You may download and install the latest product update from the premier support web site at https://premier.intel.com.

Reference #	Product	Version	Operating System	Title	Last Update
26589	Intel(R) C++ Compiler for Linux*	6.0,7.0	Red Hat* 7.2	Incorrect DWARF debug information generated for C99 adjustable arrays	28−Oct−02

Symptom

The compiler generates incorrect DWARF debug information for C99 adjustable arrays. Specifically:

− It describes all of the array objects as references to adjustable arrays, rather than as arrays themselves;

− all of the variables which are dynamic arrays are described in the DWARF as arguments to the subroutine in which they occur, even if they're declared as locals.

Current Status/Solution

This problem has been resolved. The solution will be available in a future product release.

Reference #	Product	Version	Operating System	Title	Last Update
26638	Intel(R) C++ Compiler for Linux*	6.0	Caldera*	The compiler reports: Unsupported encoding mode 10.	30−Sep−02

Symptom

When an specific application is run, the following error message is displayed:

Unsupported encoding mode 10. Please report back to Intel.

This is an intermittent problem that mostly occurs after the application has run for a while. This runtime problem has been seen when running building and running the application under the RedHat Linux 7.3 operating system.

Current Status/Solution

This problem has been resolved in a product update with package ID l_cc_pu_6.0.166 or higher. You may download and install the latest product update from the premier support web site at https://premier.intel.com.

Reference #	Product	Version	Operating System	Title	Last Update
21735	Intel(R) C++ Compiler for Linux*	6.0	Red Hat* 7.2	DWARF information for classes describes them as structs	29−Oct−02
Symptom	The DWARF debug information emitted for a class uses the DW_TAG_structure_type tag (describing them as a struct) rather than the DW_TAG_class_type tag (to describe them as a class).				
Current Status/Solution This problem has been resolved in a product update with package ID l_cc_p_6.0.1.304 or higher. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support.					

Reference #	Product	Version	Operating System	Title	Last Update
28571	Intel(R) C++ Compiler for Linux*	6.0,7.0	Red Hat* 7.2	Debugger cannot display the contents of virtual base classes correctly	5−Dec−02
Symptom	Incorrect DWARF addressing information is emitted for virtual base class members of derived classes. This prevents a debugger from being able to display the contents of the virtual base class correctly.				
Current Status/Solution This is a known issue that may be resolved in a future product release.					

Reference #	Product	Version	Operating System	Title	Last Update
27115	Intel(R) C++ Compiler for Linux*	6.0	Red Hat* 7.1	Size of shared lib created using icc is ~100 times larger than that created using gcc	29−Oct−02
Symptom	A library created by the Intel(R) Linux* C++ Compiler is about 100 times larger than the equivalent library created by the GCC compiler.				
Current Status/Solution The Intel−compiled library will be larger than the equivalent GCC−compiled library because by default, libcprts is statically linked for the Intel Compiler. GCC links dynamically by default. If the GCC−compiled library is built statically, then the libraries produced by both compilers will be about the same size.					

Reference #	Product	Version	Operating System	Title	Last Update
26994	Intel(R) C++ Compiler for Linux*	6.0, 7.0	Red Hat 7.2	Profile Guide Optimization ineffectual on files containing inline assembly	28−Oct−02

<table>
<tr><td><b>Symptom</b></td><td>Compiling a module containing inline assembly with profile−guided optimization (PGO) options is ineffectual. The resulting object file shows no benefit from the PGO options.</td></tr>
<tr><td colspan="2"><b>Current Status/Solution</b>

This is a known issue that may be resolved in a future product release.</td></tr>
</table>

<table>
<tr><th>Reference #</th><th>Product</th><th>Version</th><th>Operating System</th><th>Title</th><th>Last Update</th></tr>
<tr><td>27428</td><td>Intel(R) C++ Compiler for Linux*</td><td>6.0</td><td>Red Hat* 7.1</td><td>Link warning when building code with _asm directives and a shared library</td><td>6−Dec−02</td></tr>
<tr><td><b>Symptom</b></td><td colspan="5">When building a source code that contains _asm directives and using a shared library, there is a warning(s) of the form

ld: warning: type and size of dynamic symbol `_ZN6TfTypeC1Ev' are not defined</td></tr>
<tr><td colspan="6"><b>Current Status/Solution</b>

This problem has been resolved in the Intel(R) C++ Compiler 7.0. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support.</td></tr>
</table>

<table>
<tr><th>Reference #</th><th>Product</th><th>Version</th><th>Operating System</th><th>Title</th><th>Last Update</th></tr>
<tr><td>27633</td><td>Intel(R) C++ Compiler for Linux*</td><td>6.0,7.0</td><td>Red Hat* 7.2</td><td>libraries built using the −g swtich are enormous</td><td>26−Nov−02</td></tr>
<tr><td><b>Symptom</b></td><td colspan="5">Building libraries with −g result in enormous file size of over 10x versus not using −g.</td></tr>
<tr><td colspan="6"><b>Current Status/Solution</b>

This is a known issue and may be resolved in a future product release.</td></tr>
</table>

<table>
<tr><th>Reference #</th><th>Product</th><th>Version</th><th>Operating System</th><th>Title</th><th>Last Update</th></tr>
<tr><td>26850</td><td>Intel(R) C++ Compiler for Linux*</td><td>6.0</td><td>Red Hat* 7.2</td><td>The std::cos function behaves like std::sin</td><td>29−Oct−02</td></tr>
<tr><td><b>Symptom</b></td><td colspan="5">The std::cos function with argument of float or double type returns the sine of the argument, not the cosine.</td></tr>
<tr><td colspan="6"><b>Current Status/Solution</b>

This problem has been resolved in a product update with package ID l_cc_p_6.0.1.304 or higher. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support.</td></tr>
</table>

Reference #	Product	Version	Operating System	Title	Last Update

<table>
<tr><td>27478</td><td>Intel(R) C++ Compiler for Linux*</td><td>6.0</td><td>Red Hat* 7.1</td><td>Error caused by __attribute__((__unused__))</td><td>21−Nov−02</td></tr>
<tr><td><b>Symptom</b></td><td colspan="5">C++ code like the following:

void foo()
{
void (*x)() __attribute__ ((__unused__));
}

results in an error:

test.cpp(4): error: identifier "x" is undefined
void (*x)() __attribute__ ((__unused__));
^

test.cpp(4): error: expression must have (pointer−to−) function type
void (*x)() __attribute__ ((__unused__));
^

test.cpp(4): error: expected a ";"
void (*x)() __attribute__ ((__unused__));
^

If you compile this as C code, or with gcc/g++, you will not get this error.</td></tr>
<tr><td colspan="6"><b>Current Status/Solution</b>

This problem has been resolved in the Intel(R) C++ Compiler 7.0. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support.</td></tr>
</table>

<table>
<tr><th>Reference #</th><th>Product</th><th>Version</th><th>Operating System</th><th>Title</th><th>Last Update</th></tr>
<tr><td>29341</td><td>Intel(R) C++ Compiler for Linux*</td><td>6.0,7.0</td><td>Red Hat* 7.2</td><td>The Itanium(R) compiler hangs compiling lame MP3 encoder at −O2 optimization level</td><td>28−Oct−02</td></tr>
<tr><td><b>Symptom</b></td><td colspan="5">The Itanium(R) compiler hangs when compiling the "lame MP3 encoder Linux* application" at the −O2 optimization level, but compiles correctly at −O0, −O1, and −O3. The problem occurs during the compilation of the decode_i386.c module.</td></tr>
<tr><td colspan="6"><b>Current Status/Solution</b>

This problem has been resolved. The solution will be available in a future product release. As a workaround, do not compile the decode_i386.c module at −O2 optimization level.</td></tr>
</table>

<table>
<tr><th>Reference #</th><th>Product</th><th>Version</th><th>Operating System</th><th>Title</th><th>Last Update</th></tr>
<tr><td>27807</td><td>Intel(R) C++ Compiler for Linux*</td><td>6.0</td><td>Red Hat* 7.2</td><td>Linux* C++ Compiler for Itanium(R) Architecture requires −prof_gen to link</td><td>28−Oct−02</td></tr>
<tr><td><b>Symptom</b></td><td colspan="5">When building object modules (discretely) using the PGO switch −prof_gen, the Itanium compiler also requires that the −prof_gen switch be included in the link command. The C++ Compiler for IA32 does not require the switch for the link command.</td></tr>
</table>

	Itanium(R) architecture example: ecc −O2 −c −prof_gen test.c −o test.o ecc test.o −o test displays errors of the form: test.o: In function `main': test.o(.text+0x42): undefined reference to `_PGOPTI_Prof_Begin' test.o: In function `addstr': test.o(.text+0x242): undefined reference to `_PGOPTI_Prof_Begin' Using the command ecc test.o −o test −prof_gen will successfully build. IA32 architecture example: icc −O2 −c −prof_gen test.c −o test.o icc test.o −o test will successfully build.
Current Status/Solution This problem has been resolved in the Intel(R) C++ Compiler 7.0. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support.	

Reference #	Product	Version	Operating System	Title	Last Update
27919	Intel(R) C++ Compiler for Linux*	6.0, 7.0	Red Hat* 7.1	The port@host syntax is not supported when ICID determines location of the license file	25−Nov−02
Symptom	The following syntax for locating the license key is not supported: LM_LICENSE_FILE=port@host Using this syntax will cause a segmentation fault in the ICID utility when reading the license key information.				
Current Status/Solution This problem has been resolved in a product update with package ID l_cc_pu_7.0.078 or higher. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support.					

Reference #	Product	Version	Operating System	Title	Last Update
26927	Intel(R) C++ Compiler for Linux*	6.0	Red Hat* 7.1	ios_base::ate does not work properly with ostringstream	25−Nov−02
Symptom	The "ios_base::ate" openmode does not work properly when used with ostringstream. For example: #include <iostream>				

```
#include <sstream>

int main(int, char *[])
{
const char str1[] = "The quick brown fox jumped ",
str2[] = "over the lazy dog.";

std::ostringstream phrase(str1,std::ios_base::ate);
phrase << str2;

// Should print: "The quick brown fox jumped over the lazy dog."

std::cout << phrase.str() << std::endl;

// With Intel(R) C++ compiler we have instead: "over the lazy dog.x jumped"

return 0;
}
```

Current Status/Solution

This problem has been resolved in a product update with package ID l_cc_pu_6.0.1.308 or higher. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support

Reference #	Product	Version	Operating System	Title	Last Update
28049	Intel(R) C++ Compiler for Linux*	6.0	Red Hat* 7.2	Scope of types wrong in C++ debug information	28−Oct−02

Symptom

The compiler emits type definitions in the wrong scope in the DWARF debug information.

```
struct outer {
struct inner {
struct innerinner {
int x;
} inner_innerinner;
} outer_inner;
};
```

In this example, the scope of the definition of struct inner is struct outer, not global scope as reported by the compiler.

Current Status/Solution

This is a known issue that may be resolved in a future product release.

Reference #	Product	Version	Operating System	Title	Last Update

28512	Intel(R) C++ Compiler for Linux*	6.0	Red Hat* 7.1	GNU gcc source compatibility issue: some functions not defined in the std namespace	28−Oct−02

Symptom

The Intel C++ Compiler is not gcc source compatible for some functions in the std namespace. The following functions are not defined in std namespace for the C++ compiler:

fwide;
fwprintf;
fwscanf;
swprintf;

Current Status/Solution

This problem has been resolved in the Intel(R) C++ Compiler 7.0. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support.

Reference #	Product	Version	Operating System	Title	Last Update
28513	Intel(R) C++ Compiler for Linux*	6.0	Red Hat* 7.1	Intel C++ Compiler defines _STD macro, causes conflicts	28−Oct−02

Symptom

The Intel C++ Compiler defines the _STD macro as
#define _STD std::
in yvals.h

This may result in compile time errors with applications or standard libraries that also define the _STD macro.

Current Status/Solution

This problem has been resolved in the Intel(R) C++ Compiler 7.0. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support.

Reference #	Product	Version	Operating System	Title	Last Update
28784	Intel(R) C++ Compiler for Linux*	7.0	Red Hat* 7.2	Cannot make a symbol "weak"	23−Jan−03

Symptom

The Intel(R) C++ Compiler for Linux* documentation states that the compiler does not support "Function Attributes Declarations" such as the use of the "weak" attribute shown below:

```
extern void foo(void) __attribute__((weak));
```

Current Status/Solution

The documentation is in error. This documentation problem may be resolved in a future product release. The Intel(R) C++ Compiler for Linux* does support function attributes. Below is an example of using the "weak" attribute.

```
$ cat foo.c
extern void foo(void) __attribute__((weak));
extern void foo(void)
{
int i;
i=2;
}
$ ecc −c foo.c
$ nm foo.o
0000000000000000 t .l_foo
0000000000000000 ? __fte_foo
0000000000000000 ? __udt_foo
0000000000000000 W foo
```

Reference #	Product	Version	Operating System	Title	Last Update
28946	Intel(R) C++ Compiler for Linux*	6.0,7.0	Red Hat* 7.2	ICC does not initialize static constant members within the class scope	23−Jan−03

Symptom

When compiling the code:

```
class A
{
static const char* const a = "aaa";
};

void foo()
{
}
```

The compiler will give the following error:

```
−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−

foo.cc(3): error: data member initializer is not allowed
static const char* const a = "aaa";
^

compilation aborted for foo.cc (code 2)

−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
```

Current Status/Solution

This is a known issue that may be resolved in a future product release. As a workaround, initialize the data member as you would for an object of class type (i.e. outside the class definition):
const char* const A::a = "aaa"; // put this in a source file

Reference #	Product	Version	Operating System	Title	Last Update
29661	Intel(R) C++ Compiler for Linux*	6.0, 7.0	Red Hat* 7.1	A volatile void pointer parameter dereferenced in a function causes segmentation fault	26−Nov−02

Symptom	Consider the following test case:
	`void t(volatile void *x, int i)`
	`{`
	`if (i) *x;`
	`}`
	When the above code is compiled, the compiler reports a segmentation fault. If volatile type qualifier is removed from the parameter or if the pointer is not dereferenced in the statement, the code is compiled successfully.

Current Status/Solution

This problem has been resolved in a product update with package ID l_cc_pu_7.0.082 or higher. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support

Reference #	Product	Version	Operating System	Title	Last Update
28766	Intel(R) C++ Compiler for Linux*	6.0	Other (specify below)	Errors with /usr/include/linux/byteorder/swab.h	23−Nov−02

Symptom	The compiler issues the following errors when compiling a program that uses the <linux/cdrom.h> header: error: expected an expression return __arch__swab16(x) error: expected an expression return __arch__swab16p(x) error: expected an expression return __arch__swab16s(x) error: expected an expression return __arch__swab32(x) error: expected an expression return __arch__swab32p(x) error: expected an expression return __arch__swab32s(x)

Current Status/Solution

This problem has been resolved in a product update with package ID l_cc_pu_7.0.068 or higher. You may download and install the latest product update from the premier support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support

Windows*

Reference #	Product	Version	Operating System	Title	Last Update

18282	Intel(R) C++ Compiler for Windows*	5.0,6.0,7.0	Windows* 2000 Server	Derived class enumeration bug	27−Sep−02

Symptom	The construct below results in a compiler error.
	
	class A
	{
	public:
	enum {Enum};
	};
	
	class B : public A
	{
	public:
	enum {Enum=Enum};
	};
	
	error: "Enum" has already been declared in the current scope
	enum {Enum = Enum};
	^
	A workaround is to change one of the enumerated names:
	class A
	{
	public:
	enum {Foo};
	};
	
	class B : public A
	{
	public:
	enum {Enum=Foo};
	};				

Current Status/Solution This is a known issue that may be resolved in a future product release.					

Reference #	Product	Version	Operating System	Title	Last Update
19189	Intel(R) C++ Compiler for Windows*	5.0,6.0,7.0	Windows NT* 4.0 Service Pack 4	No option to turn off "targeted for automatic cpu dispatch" remarks.	27−Sep−02

Symptom	With foo.cpp:
	−−−− begin foo.cpp −−−−
	void addOne(unsigned char *pData, unsigned int length)
	{
	for(unsigned int i = 0; i < length; i++)
	{
	pData[i] += 1;
	}
	}				

```
−−−− end foo.cpp −−−−

..."icl /c /QaxM foo.cpp" gives:
foo.cpp
foo.cpp(3) : (col. 5) remark: LOOP WAS VECTORIZED.
foo.cpp(2) : (col. 1) remark: ?addOne@@YAXPAEI@Z has been targeted for automatic cpu dispatch.
```

There is no option to turn off the remarks about functions being targeted for automatic cpu dispatch.

Current Status/Solution

This is a known issue that may be resolved in a future product release.

Reference #	Product	Version	Operating System	Title	Last Update
20109	Intel(R) C++ Compiler for Windows*	5.0.1, 6.0,7.0	Windows* 2000 Server	printf invalid format string: warning #269	27−Sep−02

Symptom

There are cases where the compiler reports invalid format string conversion in printf/fprintf/etc. functions when they are valid. This happens when using "%S" to specify wide character strings when using the C runtime library's printf (also "%ws", "%wS", "%ls" and "%lS") or to specify single character strings when using the C runtime library's wprintf (also "%hs" and "%hS").

```
#include <stdio.h>

int main(int argc, char *argv[])
{
wchar_t widestring[] = L"A Wide String";
char narrowstring[] = "A Narrow String";

char widedest[100];
char narrowdest[100];

// Convert a wide string to a narrow string − 5 different ways
sprintf(narrowdest, "%S", widestring); // because we are a non−wide char app, "%S" means a wide string.
// If we were a wide char app, "%S" would mean a normal char string
sprintf(narrowdest, "%ls", widestring);
sprintf(narrowdest, "%lS", widestring);
sprintf(narrowdest, "%ws", widestring);
sprintf(narrowdest, "%wS", widestring);

// Convert a narrow string to a wide string − 2 different ways
sprintf(widedest, "%hs", narrowstring);
sprintf(widedest, "%hS", narrowstring);

return 0;
}
```

Current Status/Solution This is a known issue that may be resolved in a future product release.	

Reference #	Product	Version	Operating System	Title	Last Update
21906	Intel(R) C++ Compiler for Windows*	5.0,6.0.7.0	Other (specify below)	Itanium(R) compiler unaligned access error with __unaligned keyword used on a structure	26−Nov−02
Symptom	The Itanium(R) comipiler doesn't create the proper unaligned stores when trying to store unaligned data from a structure. This causes an unaligned access error when the compiled program is run.				
Current Status/Solution This is a known issue that may be resolved in a future product release. As a workaround, try one of the following: 1) copy each member of the struct separately (so the operator= does not come into play.) 2) Define a user defined operator= that takes an unaligned object. For example: __unaligned Object&operator=(const Object)__unaligned {}					

Reference #	Product	Version	Operating System	Title	Last Update
22666	Intel(R) C++ Compiler for Windows*	6.0,7.0	Windows* 2000 Professional	Compiler does not support double casting	28−Oct−02
Symptom	The Intel(R) C++ compiler for Itanium(R)−based applications does not support double casting. When you try to double cast with the Intel compiler, the compiler gives an "integer conversion resulted in truncation" error. Here is a code example that reports this error: const unsigned __int32 addr32 = (unsigned __int32) (unsigned __int64) // Note the double cast (first to __int64 then to __int32)				
Current Status/Solution This is a known issue that may be resolved in a future product release.					

Reference #	Product	Version	Operating System	Title	Last Update
22910	Intel(R) C++	5.0.1,	Windows*	Dependency information not generated	25−Nov−02

	Compiler for Windows*	6.0, 7.0	2000 Professional	under certain circumstances	
Symptom	If a header file is updated in an application, pressing F7 in Microsoft* Visual Studio 6 does not rebuild the application. The problem is that initial dependency information is not created because the Microsoft compiler finds an error in the header file but the Intel compiler does not. Updating the header file further will also not be recognized since there is no dependency information to reference.				
Current Status/Solution This is a known issue that may be resolved in a future product release.					

Reference #	Product	Version	Operating System	Title	Last Update
23572	Intel(R) C++ Compiler for Windows*	6.0,7.0	Windows* 2000 Professional	Large value (>=0x80000000) cast from double to int produces incorrect results	25−Nov−02

Symptom	A large double precision value that is cast from double to int produces incorrect results.
	For example, the following code fragment:
	double v=2164195328.0000; // (0x80ff0000)
	int optval = (int)(v+0.5);
	Results in optval=0x80000000				

Current Status/Solution
This is a known issue that may be resolved in a future product release. As a workaround first cast to unsigned int and then to int.
Example:
double v=2164195328.0000; // (0x80ff0000)
int optval;
if (v optval = −(int)(unsigned int)(−v+0.5);
else
optval = (int)(unsigned int)(v+0.5);					

Reference #	Product	Version	Operating System	Title	Last Update
24227	Intel(R) C++ Compiler for Windows*		Windows* 2000 Professional	The C++ Compiler compiles an illegal member−like constructor call	9−Dec−02

Symptom	ICL compiles successfully the code where a class constructor is called within the body of different consructor of the same class.
	Other compilers (VC6, gcc) mark this code as error.
	
	class A {
	public:
	A();
	A(int);
	int i_;
	};
	
	A::A(int i) {
	i_ = i;
	}
	
	A::A() {
	this−>A(2);
	}

Current Status/Solution This problem has been resolved in a product update with package ID W_CC_PU_7.0.073 or higher. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support	

Reference #	Product	Version	Operating System	Title	Last Update
22544	Intel(R) C++ Compiler for Windows*	6.0, 7.0	Other (specify below)	Native 64−bit EDB cannot see processes	27−Sep−02
Symptom	Using the 64−bit version of Intel Enhanced Debugger (EDB) running on an Itanium processor system to debug an application it is not possible to debug processes. If you go to "File−>Attach Process...", the "Attach To Process" box comes up but all but a few user process names show up under the "Process Name" section as "system". If you know the PID of the process you want to attach to, trying to attach comes back with the error "Unable to load \system. Access is denied."				
Current Status/Solution This is a known issue that may be resolved in a future product release.					

Reference #	Product	Version	Operating System	Title	Last Update
25485	Intel(R) C++ Compiler for Windows*	6.0, 7.0	Windows* 2000 Professional	Compiler issues error message when compiling in−line asm code using __asm_emit	22−Nov−02

Symptom

The Intel C++ Compiler for Windows (icl), issues an error when compiling the following code:

```
#define CPUID_INSTRUCTION \
__asm _emit 0x0f \
__asm _emit 0xa2
#endif

int __glQueryCPUID()
{
__asm CPUID_INSTRUCTION
return 0;
}

error: label "_emit" was referenced but not defined
```

Current Status/Solution

This is a known issue that may be resolved in a future product release. As a workaround, do not specify the __asm keyword on "_emit 0x0f" as shown below:

```
#define CPUID_INSTRUCTION \
_emit 0x0f \
__asm _emit 0xa2
#endif

int __glQueryCPUID()
{
__asm CPUID_INSTRUCTION
return 0;
}
```

Reference #	Product	Version	Operating System	Title	Last Update
25664	Intel(R) C++ Compiler for Windows*	6.0,7.0	Windows* XP Professional	Default Include Directory path does not change back to Microsoft* Visual C++* .NET compiler	29−Oct−02

Symptom

When switching back to the Microsoft* Visual C++* .NET compiler, the Default "Include Directory Path" still includes the Intel(R) C++ Compiler include directory.

Current Status/Solution

This problem has been resolved in a product update with package ID W_CC_PU_6.0.108 or higher. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com.

Reference #	Product	Version	Operating System	Title	Last Update
25728	Intel(R) C++ Compiler for Windows*	6.0,7.0	Other (specify below)	Alignment not carried over to derived class	30−Sep−02
Symptom	The alignment of a base class, specified with __declspec(align(n)), isn't carried over to the derived class.				
Current Status/Solution This problem has been resolved in a product update with package ID W_CC_P_7.0.072 or higher. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support					

Reference #	Product	Version	Operating System	Title	Last Update
26318	Intel(R) C++ Compiler for Windows*	6.0,7.0	Windows* 2000 Server	Runtime errors when using Stingray* Objective Studio 2000 with the C++ Compiler for Windows*	30−Oct−02
Symptom	Run−time errors such a invalid memory accesses may result when using the Intel(R) C++ Compiler for Windows* to compile an application that makes use of the Stingray* Objective Studio 2000 class library.				
Current Status/Solution This is a known issue that may be resolved in a future product release.					

Reference #	Product	Version	Operating System	Title	Last Update
26058	Intel(R) C++ Compiler for Windows*	6.0	Windows* 2000 Server	Incorrect handling of __declspec(dllexport) in static functions	30−Oct−02

Symptom	When a static function is declared with the '__declspec(dllexport)' statement incorrectly put before the declaration the compiler generates an .obj file instead of reporting an error. The function in the generated .obj does not actually have external linkage. See the test case below:
	
	C:\temp>copy con test.c
	__declspec(dllexport) static void foo()
	{
	int i;
	i=0;
	i++;
	}
	^Z				

Current Status/Solution This problem has been resolved in a product update with package ID W_CC_P_7.0.072 or higher. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support

Reference #	Product	Version	Operating System	Title	Last Update
26167	Intel(R) C++ Compiler for Windows*	6.0	Windows* 2000 Professional	Intel C++ Compiler 6.0 installation does not complete when using Netscape* 6.2 as default browser	29−Oct−02
Symptom	The Intel C++ compiler installation is not compatible with Netscape* 6.2. Files are extracted from the self−extracting archive but the installation does not complete.				
Current Status/Solution This problem has been resolved in a product update with package ID W_CC_PU_6.0.099 or higher. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support.					

Reference #	Product	Version	Operating System	Title	Last Update
26177	Intel(R) C++ Compiler for Windows*	5.0,6.0	Windows* 2000 Professional	2+ GB addresses with /3GB cause runtime problems	2−Oct−02
Symptom	When running Microsoft* Windows* NT* 4.0 or Windows 2000 with the /3GB option enabled, you may run into this problem. During optimization the comparisons for loop exit conditions may be based on memory addresses. However, if the addresses straddle the 2GB boundary of memory, the comparison done with an address above 2GB and one below 2GB will result in the overflow bit being set. The compiler may then have a jump that is sensitive to the overflow bit causing the application to behave erratically.				
Current Status/Solution This problem has been resolved in a product update with package ID W_CC_P_6.0.1.305 or higher. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support					

Reference #	Product	Version	Operating System	Title	Last Update

26131	Intel(R) C++ Compiler for Windows*	6.0	Windows* XP Professional	Can't find how to select/deselect the Intel Compiler in Microsoft* Visual C++* .NET	21−Nov−02
Symptom	Currently, the information on how to select or deselect the Intel compiler in .NET is somewhat buried in the section on "Creating a New Visual C++ Project" under "Using the Intel(R) C++ Compiler with Microsoft* Visual C++* .NET*". This information should be given its own section with a more obvious title like "Selecting the Intel(R) C++ Compiler".				
Current Status/Solution This problem has been resolved in the Intel(R) C++ Compiler 7.0. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http:/www.intel.com/software/products/support.					

Reference #	Product	Version	Operating System	Title	Last Update
27053	Intel(R) C++ Compiler for Windows*	6.0	Windows NT* 4.0 Service Pack 6	Compiler fails to destroy objects processing nested exception handling code	30−Sep−02
Symptom	While executing in a catch scope − trying to handle a thrown exception − a nested try−catch block is visited, and an exception is thrown and caught by this nested try−catch block. After returning from the original try−catch block, certain objects may not be destroyed.				
Current Status/Solution This problem has been resolved in a product update with package ID W_CC_PU_6.0.108 or higher. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support					

Reference #	Product	Version	Operating System	Title	Last Update
26137	Intel(R) C++ Compiler for Windows*	6.0	Windows* 2000 Professional	threadprivate OpenMP pragmas ignored	29−Oct−02
Symptom	The compiler ignores the "threadprivate" OpenMP pragma when it is placed by itself in a file. For example, it is ignored when being used to declare a global variable "threadprivate" like so: −−−test.c−−− int n; #pragma omp threadprivate(n) int foo(void) {} −−−end test.c−−−				

Current Status/Solution

This problem has been resolved in a product update with package ID W_CC_P_7.0.072 or higher. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support

Reference #	Product	Version	Operating System	Title	Last Update
26319	Intel(R) C++ Compiler for Windows*	5.0.1, 6.0	Windows* XP Professional	Handling of char * comparisons incorrect	28−Oct−02
Symptom	Unsigned pointer comparisons seem to be done as 'signed' expressions. Such comparisons should be done as unsigned expressions.				

Current Status/Solution

This problem has been resolved in a product update with package ID W_CC_P_6.0.1.305 or higher. You may download and install the latest product update from the premier support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support

Reference #	Product	Version	Operating System	Title	Last Update
27874	Intel(R) C++ Compiler for Windows*	6.0	Windows NT* 4.0 Service Pack 6	"Update User's Registry" tool and Admin rights	9−Dec−02
Symptom	The Update user's registry tool available with the Intel(R) C++ Compiler version 6.0 for Windows*, enables the select compiler tool in IDE only if the user has administrative rights on the system.				

Current Status/Solution

This problem has been resolved in a product update with package ID W_CC_P_7.0.072 or higher. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support

As a workaround:

1. Grant administrative rights to the user
2. Log into the system as the user and run the Update Users registry program
3. Disable the users administrative rights.

<table>
<tr><th>Reference #</th><th>Product</th><th>Version</th><th>Operating System</th><th>Title</th><th>Last Update</th></tr>
<tr><td>28335</td><td>Intel(R) C++ Compiler for Windows*</td><td>6.0</td><td>Windows* XP Professional</td><td>Templates with /Zi</td><td>29−Oct−02</td></tr>
<tr><td><b>Symptom</b></td><td colspan="5">While evaluating the C++ compiler I find that a test program that makes extensive use of template metaprogramming compiles successfully in under a second, however the same program fails during compile when any of the /Zi family of switches is used.</td></tr>
<tr><td colspan="6"><b>Current Status/Solution</b>

This problem has been resolved in a product update with package ID W_CC_P_7.0.072 or higher. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support</td></tr>
</table>

<table>
<tr><th>Reference #</th><th>Product</th><th>Version</th><th>Operating System</th><th>Title</th><th>Last Update</th></tr>
<tr><td>28272</td><td>Intel(R) C++ Compiler for Windows*</td><td>7.0</td><td>Windows* 2000 Professional</td><td>icl ignores the Using−declaration of base class assignment operator</td><td>30−Oct−02</td></tr>
<tr><td><b>Symptom</b></td><td colspan="5">icl ignores the using−declaration of the base class assignment operator. As a result, the following code code will not compile:

struct A { A&operator =(const A) };
struct B : A { B&operator =(const B) using A::operator =; };
int main() { B b; A a; b = a; }

t.cpp
t.cpp(2): warning #784: using−declaration of function "A::operator=" ignored
struct B : A { B&operator =(const B) using A::operator =; };
^

t.cpp(3): error: no operator "=" matches these operands
operand types are: B = A
int main() { B b; A a; b = a; }</td></tr>
<tr><td colspan="6"><b>Current Status/Solution</b>

This is a known issue that may be resolved in a future product release.</td></tr>
</table>

Reference #	Product	Version	Operating System	Title	Last Update
28297	Intel(R) C++ Compiler for Windows*	6.0	Windows* 2000 Professional	/Qoption,c,−ip_ninl_min_stats incorrect	2−Oct−02

<table>
<tr><td><b>Symptom</b></td><td>The option −ip_ninl_min_stats is spelled incorrectly in the compiler users guide as /ip_ninl_min_stats. It should be −ip_ninl_min_stats. There should be a −, rather than a / in front of the option.</td></tr>
<tr><td colspan="2"><b>Current Status/Solution</b>

This problem has been resolved in a product update with package ID W_CC_P_7.0.072 or higher. You may download and install the latest product update from the Premier Support web site at https://premier.intel.com. You need to be a registered user to access Premier Support. For registration information, please visit http://www.intel.com/software/products/support</td></tr>
</table>

<table>
<tr><th>Reference #</th><th>Product</th><th>Version</th><th>Operating System</th><th>Title</th><th>Last Update</th></tr>
<tr><td>28342</td><td>Intel(R) C++ Compiler for Windows*</td><td>6.0</td><td>Windows* 2000 Server</td><td>linker directive −export not generated in presence of _declspec(dllimport)</td><td>28−Oct−02</td></tr>
<tr><td><b>Symptom</b></td><td colspan="5">Given the following test case:

extern "C" __declspec(dllimport) char foo;
char foo;

Doing dumpbin /directives bar.obj no −export:_foo,data will be found in the object file.

If the test case is rewritten as:

extern "C" __declspec(dllimport) char foo;
char foo = 'A';

This will generate the correct −export directive.</td></tr>
<tr><td colspan="6"><b>Current Status/Solution</b>

This problem has been resolved. The solution will be available in a future product release.</td></tr>
</table>

<table>
<tr><th>Reference #</th><th>Product</th><th>Version</th><th>Operating System</th><th>Title</th><th>Last Update</th></tr>
<tr><td>29035</td><td>Intel(R) C++ Compiler for Windows*</td><td>6.0.1</td><td>Windows* XP Professional</td><td>Visual Studio .NET crashes when selecting properties of multiple projects</td><td>28−Oct−02</td></tr>
<tr><td><b>Symptom</b></td><td colspan="5">When you select multiple projects in the solution explorer and go to Properties −> C/C++ −> Command Line, Visual Studio.NET will create an error report and shut down.</td></tr>
<tr><td colspan="6"><b>Current Status/Solution</b>

This is a known issue that may be resolved in a future product release.</td></tr>
</table>

Reference #	Product	Version	Operating System	Title	Last Update
29228	Intel(R) C++ Compiler for Windows*	6.0	Windows* 2000 Professional	UUID not allowed on enumerated types	26−Nov−02

Symptom	The Intel(R) C++ Compiler prints an error message and does not allow the inclusion of a UUID in the declaration of an enumerated type.
	
	Consider the following code:
	
	test.cpp:
	#include <comdef.h>
	
	enum __declspec(uuid("00000546−0000−0010−8000−00aa006d2ea4"))
	TestEnum
	{
	x=0,
	y=1,
	z=2
	};
	
	With the compile command
	icl −c test.cpp
	this error message is generated:
	
	test.cpp(3): error: expected either a definition or a tag name
	enum __declspec(uuid("00000546−0000−0010−8000−00aa006d2ea4"))				

Current Status/Solution This is a known issue that may be resolved in a future product release.					

Reference #	Product	Version	Operating System	Title	Last Update
29660	Intel(R) C++ Compiler for Windows*	6.0, 7.0	Windows* XP Professional	Integration tool causes link error when Qipo active &project dependencies exist	23−Nov−02
Symptom	Given two projects P1 and P2 in the Microsoft* Visual Studio* .NET where P1 has a dependency on P2, if you try to compile both P1 and P2 projects with the −Qipo switch, the project P2 compiles correctly, but project P1 does not and produces the following warnings and errors: warning LNK4229: invalid directive 'Intel(R) compiler non−linkable IL object file' "unresolved function _DllMainstartup".				
Current Status/Solution This is a known issue that may be resolved in a future product release. As a workaround, remove the P1 project dependency on P2 and rebuild project P1. Note that if you now add back the project dependency, the whole project builds without errors, but the problem reappears if you exit and restart Visual Studio* .NET.					

Reference #	Product	Version	Operating System	Title	Last Update
29344	Intel(R) C++ Compiler for Windows*	6.0	Windows* 2000 Professional	The program prints incorrect results when compiled with the /QxK or /QxW switches	23−Nov−02

Symptom

The compiler produces incorrect code when compiling the following program with the −QxK or −QxW options:

```
#include <stdio.h>

int main (void)
{
unsigned int index, totalPageCount = 5, pageSize = 0x200000U;

for (index = 4; index < totalPageCount; index++)
{
if (index < 0x800000U / pageSize || index >= 0xFEE00000U / pageSize)
printf ("unexpected result\n");
}

return 0;
}
```

```
icl /QxK test.c
```

The above program incorrectly prints "unexpected result" when compiled with the /QxK option.

Current Status/Solution

This is a known issue that may be resolved in a future product release. As a workaround, use two local variables for the lower and upper bound values for the index as shown below:

```
#include

int main (void)
{
unsigned int index, totalPageCount = 5, pageSize = 0x200000U;
unsigned int nHigh = 0x800000U / pageSize;
unsigned int nLow = 0xFEE00000U / pageSize;

for (index = 4; index < totalPageCount; index++)
{
if (index < nHigh || index >= nLow)
printf ("unexpected result\n");
}

return 0;
}
```

Reference #	Product	Version	Operating System	Title	Last Update
29945	Intel(R) C++ Compiler for Windows*	6.0, 7.0	Windows* 2000 Professional	No warning message issued when 'delete' is called on a local array	23−Nov−02

Symptom

In the following code the 'delete' operator is called on a local array that has not been created dynamically. Therefore, the compiler should issue a warning message, but it doesn't.

```
main()
{
char test[10];
delete [] test; // Compiler should issue warning for this statement because the array 'test' is not dynamic (e.g. was not created with the 'new' operator).
};
```

```
icl −c test.cpp
Intel(R) C++ Compiler for 32−bit applications, Version 6.0.1 Build 20021015Z
Copyright (C) 1985−2002 Intel Corporation. All rights reserved.

test.cpp // Code compiled with no warning !
```

Current Status/Solution

This is a known issue that may be resolved in a future product release. In the mean time, please ensure that you do not use "delete" on an array as shown in the above example.

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. EXCEPT AS PROVIDED IN INTEL’S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY RELATING TO SALE AND/OR USE OF INTEL PRODUCTS, INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT, OR OTHER INTELLECTUAL PROPERTY RIGHT.

Intel products are not intended for use in medical, life saving, life sustaining, critical control or safety systems, or in nuclear facility applications.
Intel may make changes to specifications and product descriptions at any time, without notice.Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.
Intel, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.