

Intel C/ C++ Compiler w nowej wersji 9.0

WYŚCIG KOMPILATORÓW

W czerwcu Intel zaprezentował wersję 9.0 kompilatora C++ dla procesorów Intel, umożliwiającą jeszcze dokładniejsze optymalizowanie kodu. **RENÉ REBE**

Przejściowa wersja 8.1 miała na celu przygotowanie oprogramowania Intel C++ Compiler ICC [1] do obsługi architektury AMD64/ x86-64 (EM64T w procesorach Intel). Wersja 9 jest w pełni funkcjonalnym produktem kolejnej generacji, udostępniającym nowe rozszerzenia i optymalizacje [2]. Podobnie jak we wcześniejszych wersjach, kompilator obsługuje architektury IA-32, x86-64 oraz Intel Itanium. Pakiet zawiera własny debuger Intela, narzędzie typu code coverage

oraz środowisko programistyczne Eclipse. Oprócz tego dostępny jest assembler dla procesorów Itanium CPU, jednak w tym artykule nie będziemy go omawiać. Nie zapewniono na razie integracji assemblera Itanium ze środowiskiem Eclipse.

Nowy kompilator jest objęty podobną licencją do wersji poprzedniej. Licencja niekomercyjna, bez pomocy technicznej, jest dostępna bezpłatnie na potrzeby programistów realizujących projekty Open Source. Progra-

mów binarnych utworzonych z użyciem tej wersji nie można sprzedawać. Do przygotowywania programów komercyjnych wymagana jest licencja. W zależności od wielkości instalacji rejestracja może polegać na podaniu numeru seryjnego lub wskazaniu pliku z licencją. Kompilator może także korzystać z usług sieciowego menedżera na licencji Flex. Kompilator C++ kosztuje ok. 300 EUR u dystrybutorów i ok. 400 USD w firmie Intel.

Ramka 1: OpenMP

OpenMP [4] jest szerokim standardem rozszerzeń języków C, C++ i Fortran, mających na celu jawne przekazywanie kompilatorowi informacji na temat sposobu dystrybucji instrukcji programu na równolegle wykonywane wątki.

Centralnym elementem są tutaj dyrektywy `#pragma`, w których przekazywane są instrukcje o sposobie rozdzielania kodu na współbieżne części. Na przykład, zapis `#pragma omp parallel` oznacza blok programu do wykonania równoległego; `shared ()` określa zmienne wspólne, zaś `private ()` – zmienne do wyłącznego użytku danego procesu.

Dyrektywa `#pragma omp for schedule` określa sposób dystrybucji między wątkami. Dyrektywy `#pragma` oznaczają więc pętle do wykonania równoległego.

Wątki korzystają ze wspólnych zmiennych `a`, `b`, `c` oraz `chunk`; zmienna iteracyjna `i` ma w każdym wątku charakter prywatny. Wyrażenie to informuje kompilator o konieczności równoległego wykonania pętli `for` i rozdzielenia przestrzeni iteracji na indywidualne bloki o wielkości `chunk`.

W standardzie OpenMP zdefiniowano znacznie więcej instrukcji związanych z równoległym wykonywaniem złożonych programów; obecnie instrukcje te są jednak zaimplementowane tylko w specjalistycznych kompilatorach – głównie tych przeznaczonych do budowania wysokowydajnych aplikacji klastrowych. Na Wydruku 1 zaprezentowano prosty przykład użycia mechanizmu OpenMP.

Optymalizacja oparta na profilach

Wprowadzenie funkcji Software-based Speculative Pre-Computation (SSP), włączanej opcją `-ssp`, było poprzedzone gorącą dyskusją. Mechanizm ten dodaje do programu pewną liczbę wątków pomocniczych, które z wyprzedzeniem obliczają niektóre instrukcje zawarte w programie tak, aby wypełnić pamięć rozkazów procesora. Metoda ta ma być szczególnie efektywna w przypadku zastosowania technologii hyper-threading.

SSP jest tylko jedną z tak zwanych optymalizacji opartych na profilach. Optymalizacja taka polega na tym, że programista najpierw kompiluje program za pomocą specjalnego instrumentu (`-prof-gen-sampling`), aby następnie uruchomić go z reprezentatywną próbką danych wejściowych (narzędziem `profrun`). Potem program musi zostać skompilowany

Pierwsza w Polsce gazeta-forum Małych i Średnich Przedsiębiorstw

GAZETA MSP Małych i Średnich Przedsiębiorstw



Informuje
o najważniejszych wydarzeniach
Edukuje i radzi
jak najefektywniej prowadzić firmę
Prezentuje
produkty i usługi obniżające koszty działalności

po raz drugi z uwzględnieniem danych zgromadzonych w poprzednich etapach.

W przeciwieństwie do poprzednich wersji, opcja `-O2` lub wyższa włącza teraz różne optymalizacje międzyproceduralne, a kompilator optymalizuje więcej pętli. ICC dostarcza programiście jeszcze więcej informacji: raporty o optymalizowanych pętlach oraz inne ostrzeżenia są teraz bardziej szczegółowe. Do pracy na wielu plikach kompilator udostępnia dodatkowe opcje optymalizujące. Ponadto opcja `-ipo` nowego kompilatora pozwala tworzyć indywidualne pliki obiektowe przy kompilacji wielu plików.

Instalacja z pakietu RPM

Wersja 8 kompilatora Intel'a miała ok. 65 MB; najnowsza ma aż 192 MB. W paczce

Wydruk 1: Przykład zastosowania mechanizmu OpenMP

```
01 #include <omp.h>
02
03 main ()
04 {
05     const int N = 10000;
06     int i;
07     float a[N], b[N], c[N];
08     const int chunk = 100;
09
10     #pragma omp parallel shared
11     (a,b,c,chunk) private (i)
12     {
13         #pragma omp for schedule (dynamic,chunk) nowait
14         for (i=0; i < N; i++)
15             c[i] = a[i] + b[i];
16     } /* Koniec bloku wykonywanego równolegle */
17 }
```

tar znajdziemy pakiety RPM dla różnych architektur: I-386, EM64T oraz IA64. Jest tu także kompletna platforma Eclipse z pakietem programistycznym C Development Toolkit (CDT) w wersjach dla bibliotek GTK i Motif.

Instalację pakietów wykonuje skrypt powłoki; jest wiadomo o problemach występujących w dystrybucjach innych niż Red Hat i Suse (oparte na systemie pakietów RPM i objęte wsparciem firmy Intel). W niektórych systemach skrypt ogranicza się do zakomunikowania, że nie rozpoznał typu systemu, biblioteki glibc i jądra. Natomiast bez problemów przebiegła instalacja w dystrybucji Ubuntu opartej na Debianie. Bez względu na posiadaną dystrybucję trzeba mieć dostęp do polecenia `rpm` wymaganego do rozpakowania archiwum RPM.

W każdym razie kompilator ten nie jest tak łatwy w obsłudze jak GCC, który tak płynnie integruje się z systemem. ICC domyślnie nie przeszukuje katalogów z bibliotekami i plikami nagłówkowymi własnej instalacji; w praktyce oznacza to, że w wielu przypadkach trzeba użyć opcji `-I/opt/intel/cc/9.0/include` i ewentualnie `-I/opt/intel/cc/9.0/include/c++`. Ponieważ programy są zazwyczaj linkowane z bibliotekami uruchomieniowymi ICC, przed uruchomieniem kompilatora ICC chyba najkorzystniej jest ustawić zmienną środowiskową `LD_LIBRARY_PATH` na wartość `/opt/intel/cc/9.0/lib`.

Zazwyczaj wystarczy w tym celu wykonać polecenie `source /opt/intel/cc/9.0/bin/iccvars.sh` lub dodać taką linijkę do pliku `.profile`. Aby sprawę ułatwić, te ustawienia domyślne można zapisać w plikach konfiguracyjnych `.../bin/icc.cfg` oraz `.../bin/icpc.cfg`. Ustawienie zmiennej środowiskowej `LD_LIBRARY_PATH` pozwala uniknąć konieczności wprowadzania specjalnego wpisu w pliku `/etc/ld.so.conf`. Domyślnie ICC korzysta z sys-

Wydruk 2: Ostrzeżenie ICC

```
01 lib/_dtostr.c (47): remark #1572:
02 floating-point \
03 equality and inequality comparisons
04 are unreliable
05 if (d==0.0) {
06     ^
```

Tabela 1: Pętle poddawane wektoryzacji

ICC-O2	-O2- <i>ip</i>	GCC	
Botan	(171) 36	0	2
Bzip2	0	58	6
GnuPG	132	192	6
Gzip	48	62	2
Lame	118	112	22
Libmad	24	24	4
OpenSSL	148	170	30
Tramp-3D	(30) 8	0	2

temowej biblioteki C++, choć własną implementację też ma. Aby jawnie wymusić jedną z tych bibliotek, używamy odpowiednio opcji `-cxxlib-icc` lub `-cxxlib-gcc`.

Od wersji 8.1 kompilator obsługuje opcje kompilatora GNU `-march` oraz `-mcpu`. Jednak w rozwiązaniu Intel'a zastosowano nieco enigmatyczną notację `-x{K|W|N|B|P}` opartą na wewnętrznych nazwach kodowych produktów firmy: K oznacza Pentium III (Katmai), B oznacza Pentium M (Banias). Opcja `-ax` umożliwia wskazanie więcej niż jednego typu procesora; wtedy `-` tam gdzie jest to uzasadnione – ICC tłumaczy kod wielokrotnie, raz dla każdego typu. Wtedy po uruchomieniu programu zoptymalizowane bloki są uruchamiane odpowiednio do wykrytego typu procesora.

Współpraca z GCC

W zasadzie jest możliwe kompilowanie w jednym programie mieszanych plików obiektowych z użyciem kompilatorów ICC i GCC (np. wtedy, gdy ICC odmawia skompilowania pliku lub gdy GCC produkuje kod znacznie lepszej jakości). Jeśli podczas linkowania użyjemy polecenia `icc`, linker automatycznie dowieże żądane biblioteki uruchomieniowe wyprodukowane przez kompilator Intel'a:

```
icc -o prog main.o math.o
```

Jeśli jednak zechcemy powiązać pliki obiektowe programem `gcc`, musimy jawnie wskazać biblioteki dodatkowe. Linker GCC wymaga dodatkowych parametrów:

```
-L/opt/intel/cc/9.0/lib/ -lirc
```

Zresztą za kulisami kompilator Intel dla architektury IA-32 korzysta z biblioteki GNU Binutils.

Paralelizacja ręczna i automatyczna

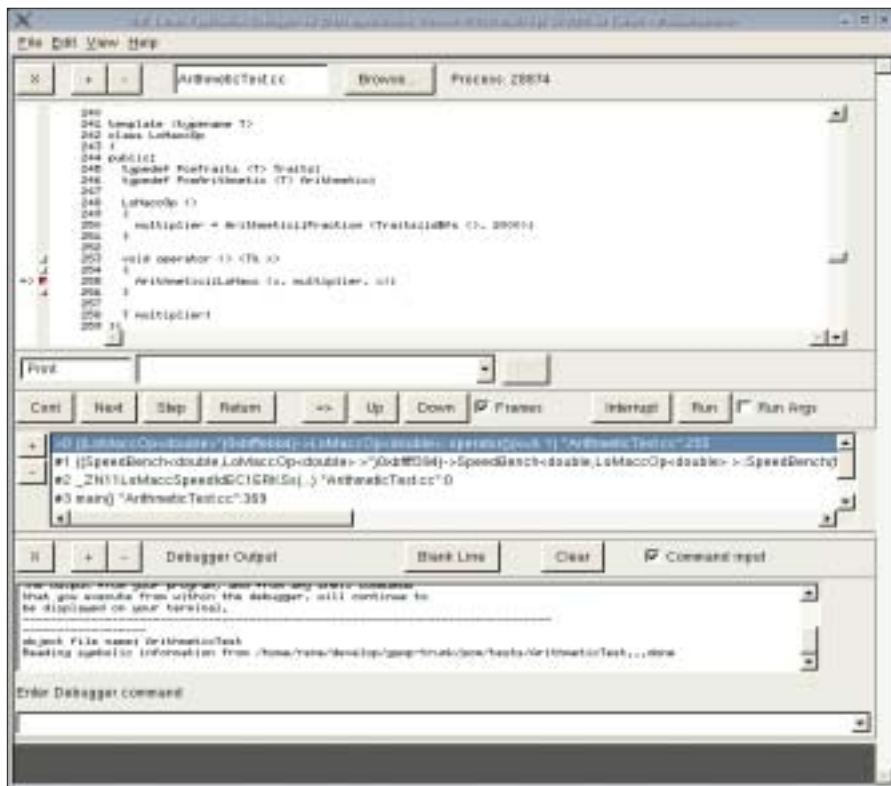
Zespół roboczy OpenMP [3] opracowuje specyfikacje, które mają na celu udostępnienie narzędzi ułatwiających pisanie programów równoległych w językach C, C++ i Fortran. Opcja *-openmp* wymusza na kompilatorze Intel generowanie programów, w których w systemach uniksowych istotne fragmenty (np. pętle) są wykonywane równolegle. Po uruchomieniu programu wyodrębnia on kolejne wątki za pomocą biblioteki Pthread; następnie poszczególne wątki są przetwarzane. To pozwala lepiej wykorzystać możliwości systemów wieloprocesorowych (patrz Ramka „OpenMP”).

ICC samodzielnie, bez pomocy OpenMP, wykrywa segmenty kodu, w których indywidualne wątki mogą być wykonywane równolegle. Do włączania automatycznej paralelizacji służy opcja *-parallel*. Teraz programiści, którym zależy na tym, aby aplikacja działała na wielu procesorach lub jądrach procesorów, mogą - bez modyfikowania kodu - oczekiwać od tego kompilatora lepszej optymalizacji. W Tabeli 1 pokazano, ile pętli kompilator poddaje wektoryzacji w poszczególnych testach. W nawiasach podano wartości dotyczące wyrażeń korzystających z kodu biblioteki C++ STL.

Podpowiedzi podczas debugingu

Pomocne przy programowaniu są także szczegółowe ostrzeżenia kompilatora Intel. W niektórych przypadkach zgłaszają znacznie więcej ostrzeżeń niż kompilator GNU; szczególnie dotyczy to wyrażeń stałych, użycia obiektów tymczasowych, porównań między liczbami zmiennopozycyjnymi oraz zmniejszenia precyzji podczas obliczeń i alokacji. Podobnie jak w poprzednich wersjach, kompilator Intel przytacza odpowiednie fragmenty kodu oraz wskazuje znak, przy którym wykryto ostrzeżenie lub błąd. To przydatna pomoc podczas debugowania (Wydruk 2, linijki 3 i 4).

Debugger dołączony do pakietu ICC ma tekstowy interfejs, tak jak jego odpowiednik GDB; przygotowano jednak również minimalny fronton graficzny. Standardowo debugger IDB działa w trybie DBX, w którym oczekuje danych wejściowych w składni zdefiniowanej przez firmę Intel. Można go jednak uruchomić w sposób zgodny z rozwiązaniem GDB (wersją 6.1 lub nowszą) - należy zastosować opcję *-gdb*.



Rysunek 1: Fronton kompilatora Intel jest funkcjonalny, ale niezbyt modnie wygląda.

Spartański interfejs graficzny

Uruchomienie programu IDB z opcją *-gui* powoduje wyświetlenie nieciekawie wyglądającego interfejsu graficznego (Rysunek 1). Interfejs ten nie zapewnia odpowiedniej automatyzacji i nie upraszcza pracy w porównaniu z wierszem poleceń. Debugger GUI nie zapewnia wizualizacji.

W przeciwieństwie do GDB, IDB nie obsługuje automatycznego uzupełniania klawiszem Tab. Można natomiast ustawiać punkty przerwań przy tworzeniu szablonów C++, o ile nie są one oznaczane wewnątrz kodu przez kompilator. GDB akceptuje tak wprowadzone punkty przerwań, ale po ich napotkaniu nie przerywa wykonywania programu.

Podsumowanie

Porównaliśmy kompilator ICC 9.0 z oprogramowaniem GCC 3.4 i 4.0 na reprezentatywnych narzędziach Open Source (w tym OpenSSL, Libmad, Bzip2 oraz Gzip) i stwierdzamy, że najnowsze wersje GCC są już bliskie lub równe wydajnością kompilatorowi ICC, który kiedyś wyraźnie je wyprzedzał. W naszych testach nie uwzględniliśmy najnowocześniejszego sprzętu ani oprogramowania zoptymalizowanego pod jego kątem, z którym czasem kojarzy się zastosowa-

nia ICC. ICC 9.0 wciąż wiodzie prym na rynku pod względem automatycznej wektoryzacji i pokazuje, jak optymalnie może być skompilowany kod C i C++ po wykonaniu dystrybucji na poszczególne jednostki SIMD (Single Instruction, Multiple Data) współczesnych procesorów. Obsługa OpenMP i automatycznej paralelizacji jest przydatna podczas budowania aplikacji do systemów wysokowydajnych. Programiści docenią również szczegółowe ostrzeżenia zwracające uwagę na potencjalne problemy na etapie programowania, a nie debugingu. ■

Dodatkowe informacje

[1] Kompilator Intel C++ <http://www.intel.com/software/products/compilers/clin>

[2] Przegląd optymalizacji ICC:

http://www.intel.com/software/products/compilers/docs/qr_guide.htm

[3] OpenMP: <http://www.openmp.org>

O Autorze

René Rebe jest aktywnym programistą linuksowym od 1997 r. Jest głównym programistą pakietu narzędzi T2.