# インテル(R) C++ コンパイラー 10.0 日本語版 スペシャル・エディション 入門ガイド

目次

### <u>概要</u>

インテル(R) C++ コンパイラーの設定 はじめに 検証用ソースファイル 適切なインストールの確認 コンパイラーの起動 (コマンドライン) コンパイル (最適化オプションなし) 実行/プログラムの検証 コンパイル (最適化オプションあり) 実行/パフォーマンスの比較 コンパイラーの起動 (Microsoft\* Visual Studio\* .NET IDE) コンパイル (最適化オプションなし) 実行/プログラムの検証 コンパイル (最適化オプションあり) 実行/パフォーマンスの比較 既存ソースのコンパイル 追加情報



インテル(R) C++ コンパイラー 10.0 Windows\* 版は、Windows ホストシステムでコマンドラインまた は Microsoft\* Visual Studio\* .NET 統合開発環境 (IDE) のいずれかを使用して C/C++ ソースファ イルをコンパイルし、Windows プラットフォームで実行するアプリケーションを作成します。この製品、 および他のインテル(R) ソフトウェア開発製品に関する詳細は、<u>インテル・ソフトウェア開発製品</u> Web サイトを参照してください。

注: C++ コンパイラーのデフォルトのインストール・パスは C:¥Program Files¥Intel¥Compiler¥C++¥10.0.025 です。このガイドでは、 パス C:¥Program Files¥Intel は <install-dir> と表記するた め、デフォルトの IA-32 インストール・パスは <install-dir>¥Compiler¥C++¥10.0.025¥IA32 と表記されます。

# インテル(R) C++ コンパイラーの設定

インテル C++ コンパイラー 10.0 Windows 版が正常にインストールされたら、環境変数を設定する 必要があります。「インストール・ガイド」を参照して、インテル C++ コンパイラー 10.0 Windows 版 が正しく設定されていることを確認してください。

 注: Microsoft Visual Studio 環境を先にインストールした場合、インテル C++ コンパイラー 10.0 Windows 版を Microsoft Visual Studio IDE 環境 で使用するための追加の手順は必要ありません。インテル C++ コンパイ ラー 10.0 Windows 版を先にインストールした場合は、「インストール・ガイ ド」の手順に従ってください。

[スタート] > [プログラム] > [Intel(R) Software Development Tools (インテル(R) ソフトウェア開発 ツール)] > [インテル(R) C++ コンパイラー 10.0.025] > [IA-32 対応アプリケーション用インテル(R) C++ コンパイラーのビルド環境] を選択して、コマンド・ウィンドウを開くと、環境が自動的に設定され ます。バッチ・ファイル (IA-32 アーキテクチャーの場合、

<install-dir>¥Compiler¥C++¥10.0.025¥IA32¥bin¥iclvars.bat)を実行して環境を 設定することもできます。

Microsoft Visual Studio IDE を使用している場合、プロジェクトを開いた後で、[プロジェクト] > [インテ  $\mu(R)$  C++ プロジェクトへ変換] を選択してください。IDE を使用してプロジェクトを設定する方法の詳 細は、インストールされるインテル(R) C++ コンパイラーのドキュメントの「アプリケーションのビルド」を 参照してください。

## はじめに

インテル C++ コンパイラーのインストールが完了したら、適切なインストール、設定、およびコンパイ ラーの動作を確認する基本的な検証作業を実行してください。この検証には、コンパイラーのインス トールの一部としてインストールされる検証用ソースファイル

<install-dir>¥Compiler¥C++¥10.0.025¥samples¥optimize¥int\_sin.c を使用しま す。

#### 検証用ソースファイル

検証用ソースファイルは、1 サイクル 2 パイラジアンの正弦曲線の絶対値を統合する数値演算プロ グラムです。次の図は、計算に使用される方法を示しています。この方法は、曲線と上辺の中央部分 が一致するように長方形を連続的に追加します。長方形の数が増えると (長方形の幅が狭くなると)、 計算される領域は 4 (4.0) に近づきます。次の図は、2<sup>4</sup> 内点と 2<sup>5</sup> 内点の最初の 8 片で何が計算 されているかを示しています。



プログラムには、stdio、stdlib、時間ライブラリーに加えてインテルの数値演算ライブラリーが含まれて いるため、インテル・ライブラリーがインストールされているかどうかを確認します。プログラムのタイミン グ関数は、プログラム実行の開始から終了までを測定したアプリケーション・クロックの数を返します。 この時間測定は不正確で、プロセッサーとその作業量に依存して変わります。

#### 適切なインストールの確認

検証用ソースファイルをコンパイルして実行し、出力が既知の正しい値である 4 に収斂するかどうか をチェックすることで、コンパイラーが適切にインストールされたかどうかを確認できます 次の手順を 実行して、インストールを確認します。

1. この検証用ソースファイルは、

<install-dir>\u00e4Compiler\u00e4C++\u00e410.0.025\u00e4samples\u00e4optimize \u00e74\u00e4b9

ます。ディレクトリへの書き込み権限があることを確認してから、ディレクトリに移動し、次のよう に検証用ソースファイルをコンパイルします。

prompt> icl int\_sin.c

<install-dir>¥Compiler¥C++¥10.0.025¥samples¥optimize ディレクトリに、プログラム int sin.exe が生成されます。次のようにプログラムを実行します。

prompt> int\_sin

3. プログラムを起動したコマンド・プロンプト・ウィンドウに次のような出力が表示されます。

Number of Interior Points		Computed Integral
4		3.141593e+000
8		3.792238e+000
16		3.948463e+000
32		3.987141e+000
64		3.996787e+000
128		3.9910.0.02597e+000
256		3.999799e+000
512		3.999950e+000
1024		3.999987e+000
2048		3.999997e+000
4096		3.999999e+000
8192		4.000000e+000

16384 4.000000e+000 32768 4.00000e+000 -----4.000000e+000 65536 -----131072 4.000000e+000 262144 4.000000e+000 -----524288 4.000000e+000 -----1048576 4.000000e+000 -----4.000000e+000 2097152 -----4194304 4.000000e+000 \_\_\_\_\_ 8388608 4.00000e+000 \_\_\_\_\_ 16777216 4.000000e+000 \_\_\_\_\_ 33554432 4.000000e+000 67108864 4.000000e+000

Application Clocks = 5.500000e+003

コンパイラーの動作が異なる場合、またはコンパイルしたプログラムの出力が正しくない場合は、インストールと設定を確認してください。必要であれば、再インストールを行ってください。問題がなければ、インテル C++ コンパイラーは適切にインストールされ、正しく動作しています。

## コンパイラーの起動(コマンドライン)

インテル C++ コンパイラー 10.0 Windows 版は、icl を使用してコマンドラインから起動できます。 ここでは、前述した int\_sin.c 検証用ソースファイルを使用します。作業の大部分をコマンドライン からではなく、IDE インターフェイスを使用して行っている場合は、このセクションをスキップして、「コン パイラーの起動 (Microsoft\* Visual Studio\* .NET IDE)」セクションに進んでください。

int sin.c 検証用ソースファイルは、

<install-dir>\u00e4Compiler\u00e4C++\u00e410.0.025\u00e4samples\u00e4optimize \u00e74\u00e4v\u00e4bles\u00e4optimize \u00e4v\u00e4v\u00e4bles\u00e4bles\u00e4optimize \u00e4v\u00e4v\u00e4bles\

このセクションを開始する前に、Windows の [スタート] メニューから [プログラム] > [Intel(R) Software Development Tools (インテル(R) ソフトウェア開発ツール)] > [インテル(R) C++ コンパイ ラー 10.0.025] > [IA-32 対応アプリケーション用インテル(R) C++ コンパイラーのビルド環境] を開 き、次のようにディレクトリを移動してください。

prompt> cd <install-dir>¥Compiler¥C++¥10.0.025¥samples¥optimize

#### コンパイル(最適化オプションなし)

最初に、最適化オプションを使用しないでコンパイルし、パフォーマンスの基準を確立します。次のよう にインテル C++ コンパイラーを起動します。

prompt> icl int sin.c /Od

/Zi デバッグ・オプション (デフォルトの最適化が /O2 から /Od に変更される) を使用してもかま いません。

#### 実行/プログラムの検証

プログラムは、ソースと同じディレクトリに int\_sin.exe という名前で生成されます。次のようにプロ グラムを実行します。

prompt> int\_sin

各計算で消費される実行時間 (プロセッサー・クロック・サイクルの数) は、内点の数が増えると、計算 された整数値 4.0 に近く (または等しく) なります。次のような出力が表示されます。

Number of	Computed Integral	
Interior Points		

4 3.141593e+000 3.792238e+000 8 \_\_\_\_\_ 3.948463e+000 16 -----32 3.987141e+000 -----64 3.996787e+000 \_\_\_\_\_ 128 3.9910.0.02597e+000 -----256 3.999799e+000 512 3.999950e+000 1024 3.999987e+000 \_\_\_\_\_ 2048 3.999997e+000 3.999999e+000 4096 -----8192 4.000000e+000 16384 4.00000e+000 32768 4.00000e+000 -----65536 4.000000e+000 \_\_\_\_\_ 4.000000e+000 131072 4.000000e+000 262144 -----524288 4.00000e+000 1048576 4.000000e+000 2097152 4.000000e+000 

			_
4194304		4.000000e+000	
8388608		4.000000e+000	
16777216		4.000000e+000	-
33554432		4.000000e+000	-
67108864		4.000000e+000	-

Application Clocks = 1.184300e+004

#### コンパイル(最適化オプションあり)

インテル C++ コンパイラーのいくつかの最適化オプションを使用して実現されるパフォーマンス向上 は非常に重要です。他のオプションは、異なる領域で演算やパフォーマンスを向上します。次のように (デフォルトの最適化を使用して) コンパイラーを起動します。

prompt> icl int\_sin.c

デフォルトでは、コンパイラーはコードの実行速度を優先するレベル 2 の最適化 (/O2) を行います。

#### 実行/パフォーマンスの比較

次のように int sin プログラムの最適化バージョンを実行します。

prompt> int sin

最適化を行わなかった場合と、アプリケーション・クロックの数を比較します。実際の時間の差は使用 するアーキテクチャーに依存します。次の出力は、一般的な IA-32 システムのものです。

Number of	C	computed Integral	
Interior Points			
4		3.141593e+000	
8		3.792238e+000	

-----16 3.948463e+000 32 3.987141e+000 3.996787e+000 64 3.9910.0.02597e+000 128 256 3.999799e+000 \_\_\_\_\_ 512 3.999950e+000 \_\_\_\_\_ 1024 3.999987e+000 2048 3.999997e+000 -----4096 3.999999e+000 -----8192 4.00000e+000 16384 4.00000e+000 -----32768 4.000000e+000 -----65536 4.000000e+000 \_\_\_\_\_ 131072 4.000000e+000 -----262144 4.00000e+000 524288 4.000000e+000 -----1048576 4.000000e+000 \_\_\_\_\_ 4.000000e+000 2097152 ------4194304 4.000000e+000 \_\_\_\_\_



Application Clocks = 5.500000e+003

注: この例における (最適化なしから最適化ありにした場合の) 実行時間の 大幅な向上はすべてのプログラムにあてはまるものではありませんが、通常 は、適切な最適化を行うことで、インテル(R) プロセッサー上で実行するプロ グラムの実行時間を向上できます。インテル C++ コンパイラーは、デフォル トでは /O2 レベルでプログラムを最適化する点に注意してください。

# コンパイラーの起動 (Microsoft\* Visual Studio\* .NET IDE)

インテル C++ コンパイラーをインストールするシステムに Microsoft Visual Studio .NET 2003 また は Microsoft Visual Studio 2005 がインストールされている場合、追加の手順なしで Microsoft Visual Studio .NET IDE 環境でインテル C++ コンパイラーを使用できます。Visual Studio .NET の 実行中にインテル C++ コンパイラーをインストールした場合、Visual Studio を再起動すると、インテ ル C++ コンパイラーが選択できるようになります。インテル C++ コンパイラーをインストールしたら、 次の手順を実行します。

 Visual Studio .NET で [ファイル] - [新規作成] - [プロジェクト] を選択して新規プロジェクトを 作成します。[新しいプロジェクト] ダイアログ・ボックスで、次に示すように、Win32 コンソー ル プロジェクトを選択します。プロジェクト名として int\_sin を使用します。

新しいプロジェクト			? 🔀
プロジェクトの種類( <u>P</u> ):		テンプレート①:	
Visual Basic     √isual C#		Visual Studio にインストールされたテンプレート	
Visual J#		Win32 コンソール アプリケーション	Win32 プロジェクト
ATL		マイテンプレート	
- 全般 - MFC - スマートデパ - Win32 - その他のプロジェ - インテル(R) Fort	イス クトの種類 tran プロジェクト	□□オンライン テンプレートの検索	
Win32 コンソール アナ	りケーションを作成するた(	」  bのプロジェクトです	
プロジェクト名(N):	int_sin		
場所( <u>L</u> ):	C:¥Documents and S	Settings¥xlpt¥My Documents¥Visual Studio 2005¥Projects	◎ 参照(B)
ソリューション名( <u>M</u> ):	[int_sin	עוע	のディレクトリを作成(D)
		□ソース管理(2	追加(山)
			OK キャンセル

2. Win32 アプリケーション ウィザードが次のウィンドウを表示します。



3. [アプリケーションの設定] を選択して、[追加のオプション] で [空のプロジェクト] を選択しま す。[完了] をクリックして、Win32 コンソール・プロジェクトを作成します。

Win32 アプリケーション ウィザー	· ≍ - int_sin		?×
アプリケー:	ションの設定		
概要 アブリケーションの設定	アプリケーションの種類: <ul> <li>Windows アプリケーション(W)</li> <li>コンソール アプリケーション(Q)</li> <li>DLL(Q)</li> <li>スタティック ライブラリ(S)</li> <li>追加のオプション:</li> <li>逆のプロジェクト(C)</li> <li>シンボルのエクスボート(S)</li> <li>プリコンパイルバ済みヘッダー(P)</li> </ul>	共通ヘッダー ファイルを追加: ATL( <u>A</u> ) MFC( <u>M</u> )	
		く前へ 次へ 一 完了 日本	キャンセル

 次に、[プロジェクト] メニューから [インテル(R) C++ プロジェクトへ変換] を選択、またはイン テル(R) C++ ツール バーの [インテル(R) C++ プロジェクトへ変換] ボタンをクリックします。 Microsoft Visual C++ プロジェクトがインテル C++ プロジェクトに変換され、新しいインテル C++ プロジェクト・ファイル (.icproj) が作成されます。

🏶 int_sin – Microsoft Visua	al S	tudio							
ファイル(E) 編集(E) 表示(⊻)	プロ	ジェクト(Ⴒ)	ビルド( <u>B</u> )	デバッグ( <u>D</u> )	ツール①	ウィンドウ(₩)			
🖥 • 🛅 • 💕 📕 🥔 🐰	₹\$	クラスの追加	ⓑⓒ)…			n32			
0 5	₹\$	リソースのネ	追加( <u>R</u> )						
スタート ページ	80	3 新しい項目の追加(W)… Ctrl+Shift+A							
	:::	既存項目(	の追加(G)…		Shift+Al	t+A			
Microsoft	-*	新しいフィノ	レタ( <u>F</u> )						
Visual :	F	Compag \	/isual Fortra	an プロジェクト <sup>3</sup>	項目を抽出し	9			
	5	インテル(R	) C++ プロジ	ェクトへ変換の	I				
最近使ったブロジェクト	ð	すべてのフォ	pイルを表示(	0					
<mark>int_sin</mark> ■att		プロジェクト	のアンロード	Û		stem			
Win App8		参照( <u>F</u> )							
🦉 int_sin_fort		Web 参照	の追加( <u>E</u> )…			າລາວ			
i interst isoptimization test		スタートアップ プロジェクト(z設定( <u>A</u> )							
		カスタム ビルド規則( <u>B</u> )							
開: フロジェクト 作成: プロジェクト		ツールのビ	ルド順序(止)…			か、最新			
	e	プロパティ(	<u>P</u> )		Alt	+F7 ition 2			

または、



5. [Confirmation (確認)] ダイアログで [はい] をクリックします。 変換が成功するとソリューショ ン エクスプローラー内にインテル(R) C++ プロジェクトのロゴが表示されます。



 IDE で [プロジェクト] - [既存項目の追加...] を選択、またはソリューション エクスプローラか ら ソースファイル を右クリックして表示されるメニューから [追加] – [既存の項目] を選択し、 表示される「既存項目の追加」ダイアログで以下のファイルを選択して [追加] ボタンをクリッ クします。

<install-dir>¥Compiler¥C++¥10.0.025¥samples¥optimize¥int\_sin.c

🏈 int_sin – Microsoft Visual Studio									
ファイル(Ё) 編集(Ё) 表示(⊻)	プロジェクト・	P) ビルド( <u>B</u> )	デバッグ( <u>D</u> )	ツール(工)	<u> ウィンドウ₩</u>				
i 🛅 • 🛅 • 📂 🔒 🕼 i 🐰	🍇 クラスの	😝 クラスの追加( <u>C</u> )							
0	🔧 אין דע	な リソースの追加(R)							
<u> スタート ページ</u>	🛅 新しい	新しい項目の追加(W) Ctrl+Shift+A							
	💴 既存功	〔目の追加( <u>G</u> )…		Shift+Al	t+A				
Microsoft	🧉 新しい	新しいフィルタ(E)							
Visual :	F Comp	Compaq Visual Fortran プロジェクト項目を抽出⊗							
	💀 Visua	C++ プロジェク	∽へ変換(⊻)		10000				
最近使ったプロジェクト	Web ₹	)照の追加( <u>E</u> )…							
<mark>int_sin</mark> ≇test	🛅 int_sin	のプロパティ( <u>P</u> )		Alt	+F7 stem ;				

または、



既存項目の追加	🛛 - int_sin								?×
探す場所①:	🚞 optimize		~	<b>(</b> ) •	Q X	<b>1</b>	<b>#</b> #	ツール(止)・	
<b>ご</b> デスクトップ	int_sin.c								
עניפר איז דעניפר איז									
ער דאר אד די בארב אד									
	ファイル名( <u>N</u> ):						~	追加	I( <u>A</u> )
	ファイルの種類(工):	すべてのファイル (*.*	)				*	- キャン	ren 🛛

7. 新しいプロジェクト int\_sin に、サンプルのソースファイル int\_sin.c が追加されます。



### コンパイル(最適化オプションなし)

次の手順を実行します。

VS.NET でソリューションをビルドする前に、ソリューションを消去します。[ビルド] - [ソリューションのクリーン] を選択します。次のレポートが表示されます。



2. 次に、[ビルド] - [ソリューションのビルド] を選択します。次のレポートが表示されます。



デフォルトのプロジェクト構成は Debug 構成なので、ソリューションは最適化なし、シンボリック・デバッグ情報付きでビルドされる点に注意してください。

これは、コマンドラインで icl /Od /Zi int\_sin.c と入力した場合と同じです。

### 実行/プログラムの検証

1. [デバッグ] - [デバッグなしで開始] を選択します。コマンド・ウィンドウに次の出力が表示されま す。

🚳 "C:¥Documents	and	Settings¥xlpt¥My	Documents¥Visual Studio 2005¥Pr 🗕 🗖 🗙
1024		3.999987e+000	
2048		3.999997e+000	
4096		3.9999999e+000	
8192		4.000000e+000	
16384		4.000000e+000	
32768		4.000000e+000	
65536		4.000000e+000	
131072		4.000000e+000	
262144		4.000000e+000	
524288		4.000000e+000	
1048576		4.000000e+000	
2097152		4.000000e+000	
4194304		4.000000e+000	
8388608		4.000000e+000	
16777216		4.000000e+000	
33554432		4.000000e+000	
67108864		4.000000e+000	
Application C	locł	ks = 1.090600	e+004
続行する には何か ▲	‡	を押してくださ	

2. 最適化を行わなかった場合のアプリケーション・クロックの数をメモします。

### コンパイル(最適化オプションあり)

最適化を行って再コンパイルします。次の手順を実行します。

1. [プロジェクト] - [int\_sin のプロパティ] を選択します。次のダイアログ・ボックスが表示されま す。

int_sin フロパティページ		2 🛛
構成(C): アクティブ(Debug)	プラットフォーム(P): アクティブ(Win32)	<ul> <li>構成マネージャ(Q)</li> </ul>
<ul> <li>■ 構成プロパティ</li> <li>- 全般</li> <li>- デバッヴ</li> <li>- C/G++</li> <li>- 全般</li> <li>- 一字化</li> <li>- 全般</li> <li>- 一字化</li> <li>- 一字ビキ成</li> <li>- 言語</li> <li>- ブリコンパイル済みヘッ</li> <li>- 出力フィル</li> <li>- ブラウザー情報</li> <li>- 診断</li> <li>- 詳細</li> <li>- コマンドライン</li> <li> コマンドライン</li> <li></li></ul>	<ul> <li>□ インテル固有 グローバルな最適化 洋動小数点精度の改善 Windows アブリケーションの最適化 デノーマル結果のフラッシュ 洋動小数点のスペキュレーション インテル(R) ブロセッサー拡張コードの使用 インテル(R) ブロセッサー拡張コードの要求 ルーブのアンロール回数 並列化</li> <li>□ 全般 最適化 インライン関数の展開 組み込み関数を使用する 速度またはサイズを優先 フレームポインターなし ファイバー保護の最適化</li> </ul>	いいえ なし いいえ いいえ 標準 (/Qfp-speculationsafe) なし なし なし いいえ <b>無効 (/Od)</b> 既定値 いいえ どちらも優先しない いいえ いいえ
	<b>最適化</b> コードの最適化オプションを指定します。特定の最近 /02, /03, /0x)	箇化オプションを使用する場合は、「カスタム」を選択します。 (/Od, /O1,
		OK キャンセル 適用(A)

2. [構成マネージャ...] をクリックして、[アクティブ ソリューション構成] を Release に変更しま す。[閉じる] をクリックします。

構成マネージャ						?×
アクティブ ソリューション構成( <u>C</u> ):		アクティブ ソリ	ューション プラットフォーム(Ⴒ):			
Release 🗸 🗸 🗸		Win32				*
プロジェクトのコンテキスト(ビルドまたは配置するプロ	ジェクト構成をチェック)	( <u>R</u> ):				
プロジェクト	構成		プラットフォーム		ビルド	
int_sin	Release	~	Win32	*	<ul> <li>Image: A set of the set of the</li></ul>	
					開じ	3

3. [プロパティ ページ] ダイアログ・ボックスに戻り、[構成プロパティ] - [C/C++] - [最適化] を選 択して、[最適化] が実行速度 (/O2) に設定されていることを確認します。[コマンドライン] を 使用してオプションを定義することもできます。

int_sin プロパティ ページ		? 🗵
構成( <u>C</u> ): アクティブ(Release)	プラットフォーム(P): アクティブ(Win32)	✓ 構成マネージャ() )
<ul> <li>● 構成フレバティ</li> <li>● 全般</li> <li>● デバッヴ</li> <li>● C/C++</li> <li>● 全般</li> <li>● 最適化</li> <li>● フリブロセッサー</li> <li>● コード生成</li> <li>● 言語</li> <li>● ブリコンパイル済みヘッ</li> <li>● 出力ファイル</li> <li>● ブラウザー情報</li> <li>● 診断</li> <li>● 副和</li> <li>● コマンドライン</li> <li>● コマンドライン</li> <li>● - フェマンドライン</li> <li>● ジステム</li> <li>● 最適化</li> <li>● コマンドライン</li> <li>● コマンドライン</li> <li>● コマンドライン</li> <li>● コマンドライン</li> <li>● コマンドライン</li> <li>● コマンドライン</li> </ul>	<ul> <li>日 インテル固有 グローバルな最適化 浮動小数点粘度の改善 Windows アブリケーションの最適化 デノーマル結果のフラッシュ 浮動小数点のスペキュレーション インテル(R) ブロセッサー拡張コードの使用 インテル(R) ブロセッサー拡張コードの要求 ループのアンロール回数 並列化</li> <li>日 全般 最適化 インライン関数の展開 組み込み関数を使用する 速度またはサイズを優先 フレームポインターなし ファイバー保護の最適化</li> </ul>	<ul> <li>いいえ なし</li> <li>いいえ いいえ</li> <li>いいえ</li> <li>高速 (/Qfp-speculationfast) なし</li> <li>なし</li> <li>いいえ</li> <li>実行速度 (/O2)</li> <li>既定値</li> <li>いいえ</li> <li>どちらも優先しない</li> <li>いいえ</li> <li>いいえ</li> <li>いいえ</li> </ul>
	<b>最適化</b> コードの最適化オプションを指定します。特定の最) 702, 703, 70x)	適化オプションを使用する場合は、「カスタム」を選択します。 (/Od, /O1,

4. [ビルド] - [ソリューションの消去] を選択してから、[ビルド] - [ソリューションのビルド] を選択 して、最適化ありで int\_sin プロジェクトをビルドします。

### 実行/パフォーマンスの比較

1. [デバッグ] - [デバッグなしで開始] を選択します。コマンド・ウィンドウに次の出力が表示されま す。



- 2. 最適化を行った場合のアプリケーション・クロックの数をメモして、最適化を行わなかった場合と 比較します。
  - 注: この例における (最適化なしから最適化ありにした場合の) 実行時間の 大幅な向上はすべてのプログラムにあてはまるものではありませんが、通常 は、適切な最適化を行うことで、インテル・プロセッサー上で実行するプログラ ムの実行時間を向上できます。インテル C++ コンパイラーは、デフォルトで は /O2 レベルでプログラムを最適化する点に注意してください。

最適化設定とその他のオプションを変更することで、プログラムの実行速度、コンパイル時間、 コードサイズに与える影響を判断できます。

# 既存ソースのコンパイル

このガイドでは、コンパイル、最適化の適用/削除、引数の指定、およびプログラム出力をモニタする手順を説明しました。既存の C/C++ ソース・プログラムがある場合、例の代わりにそれらのソースファイルをコンパイルして実行してみてください。

# 追加情報

このガイドで提供した例は、インテル C++ コンパイラーの機能の一部を紹介しただけに過ぎません。

その他の機能は、インストールされるドキュメント・インデックス doc\_index.htm から参照できます。このインデックスには、本製品に含まれているすべてのドキュメントへのリンクが含まれています。

インテル C++ コンパイラー 10.0 Windows 版の詳細は、インストールされるインテル(R) C++ コン パイラーのドキュメントを参照してください。

テクニカル・サポートや制限事項については、製品のリリースノートを参照してください。

インテル・ソフトウェア開発製品の詳しい情報については、次のインテル Web サイトを参照してください。

http://www.intel.co.jp/jp/software/products/index.htm