
# インテル® MPI ライブラリー

# スタンドアロンもしくは Cluster Edition スイートで利用可能

**MPI クラスター・アプリケーション向けの高いパフォーマンスを達成**

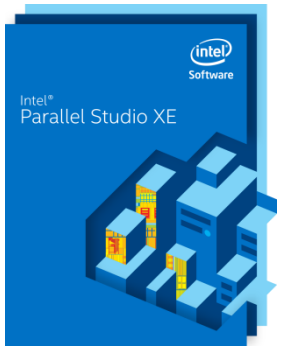
**+**

インテル® MPI ライブラリー

- MPI ライブラリー - MPICH ベース
- MPI 3.0 標準化
- ベンチマークとチューニング

**開発ツールを包括的な共有、分散およびハイブリッド・アプリケーション向けの開発スイートに進化**

- インテル® MPI ライブラリー
- **+**
- コンパイラー
- パフォーマンス・ライブラリー
- 解析ツール




インテル® Parallel Studio XE 2015 Cluster Edition[+] でも利用可能

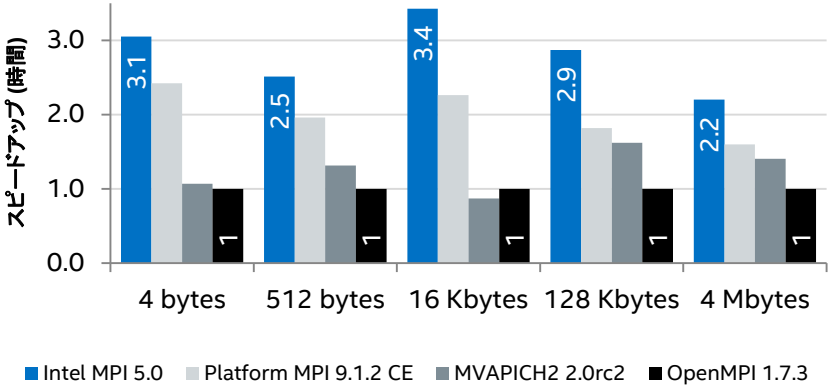
[+]Windows* と Linux* 向け

# 提供する価値
## インテル® MPI ライブラリー

| | |
|---|---|
| 何を | • インテルのハイパフォーマンス MPI ライブラリー |
| なぜ | • スケール・パフォーマンスのため - 最新のインテル® アーキテクチャー向けに最適化<br>• スケールフォワード (将来に向けた) - マルチコアとメニーコアに対応済み<br>• 効率良くスケール - 柔軟なファブリックの選択と互換性 |
| どのように | • 標準化ベース - オープンソース MPICH 実装をベースに構築<br>• 継続的な拡張性 - 低レイテンシー、高帯域幅そしてプロセスの増加に向けてチューニング<br>• 複数ファブリックのサポート - 一般的な高パフォーマンス・ネットワーク・ファブリックをサポート |

# レイテンシーの軽減はよりパフォーマンスを高める
## インテル® MPI ライブラリー

**インテル® MPI ライブラリー 5.0 による優れたパフォーマンス**
192 プロセス、8 ノード（InfiniBand + 共有メモリー）、Linux* 64
相対（幾何学的）、MPI レイテンシー・ベンチマーク（高い方が良い）

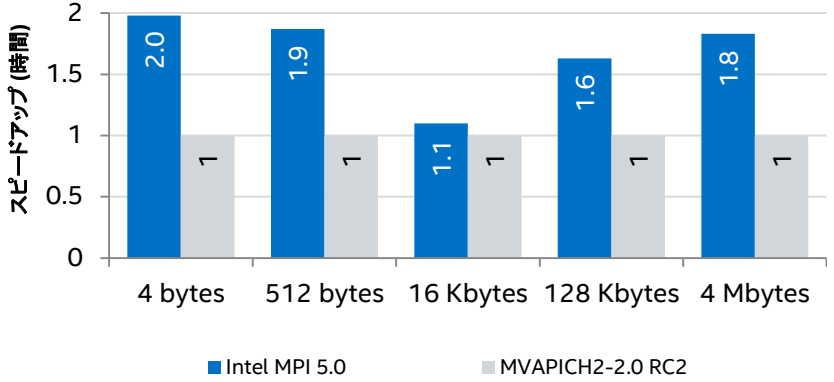



システム構成: ハードウェア: CPU: デュアル インテル® Xeon® プロセッサー E5-2697v2 @ 2.70GHz、64GB RAM。インターコネクト: Mellanox Technologies* MT27500 Family [ConnectX*-3] FDR。ソフトウェア: RedHat* RHEL 6.2、OFED 3.5-2、インテル® MPI ライブラリー 5.0、インテル® MPI ベンチマーク 3.2.4 (デフォルトのパラメーター、インテル® C++ コンパイラー XE 13.1.1 Linux* 版でビルド)。

性能に関するテストに使用されるソフトウェアとワークロードは、性能がインテル® マイクロプロセッサー用に最適化されていることがあります。SYSmark* や MobileMark* などの性能テストは、特定のコンピューター・システム、コンポーネント、ソフトウェア、操作、機能に基づいて行ったものです。結果はこれらの要因によって異なります。製品の購入を検討される場合は、他の製品と組み合わせた場合の本製品の性能など、ほかの情報や性能テストも参考にして、パフォーマンスを総合的に評価することをお勧めします。
* その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。ベンチマークの出典: インテル コーポレーション

最適化に関する注意事項: インテル® コンパイラーは、互換マイクロプロセッサー向けには、インテル製マイクロプロセッサー向けと同等レベルの最適化が行われない可能性があります。これには、インテル® ストリーミング SIMD 拡張命令 2 (インテル® SSE2)、インテル® ストリーミング SIMD 拡張命令 3 (インテル® SSE3)、ストリーミング SIMD 拡張命令 3 補足命令 (SSSE3) 命令セットに関連する最適化およびその他の最適化が含まれます。インテルでは、インテル製ではないマイクロプロセッサーに対して、最適化の提供、機能、効果を保証していません。本製品のマイクロプロセッサー固有の最適化は、インテル製マイクロプロセッサーでの使用を目的としています。インテル® マイクロアーキテクチャーに非固有の特定の最適化は、インテル製マイクロプロセッサー向けに予約されています。この注意事項の適用対象である特定の命令セットの詳細は、該当する製品のユーザー・リファレンス・ガイドを参照してください。改訂 #20110804

**インテル® MPI ライブラリー 5.0 による優れたパフォーマンス**
64 プロセス、8 ノード (InfiniBand* + 共有メモリー)、Linux* 64
相対 (幾何学的)、MPI レイテンシー・ベンチマーク (高い方が良い)




システム構成: ハードウェア: インテル® Xeon® プロセッサー E5-2680 @ 2.70GHz、RAM 64GB。インターコネクト: InfiniBand*、ConnectX* アダプター、FDR。MIC: C0-KNC 1238095 kHz、61 コア。RAM: カードごとに 15872MB。ソフトウェア: RHEL 6.2、OFED 1.5.4.1、インテル® MPSS 3.2、インテル® C/C++ コンパイラー XE 13.1.1、インテル® MPI ベンチマーク 3.2.4。

性能に関するテストに使用されるソフトウェアとワークロードは、性能がインテル® マイクロプロセッサー用に最適化されていることがあります。SYSmark* や MobileMark* などの性能テストは、特定のコンピューター・システム、コンポーネント、ソフトウェア、操作、機能に基づいて行ったものです。結果はこれらの要因によって異なります。製品の購入を検討される場合は、他の製品と組み合わせた場合の本製品の性能など、ほかの情報や性能テストも参考にして、パフォーマンスを総合的に評価することをお勧めします。
* その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。ベンチマークの出典: インテル コーポレーション

最適化に関する注意事項: インテル® コンパイラーは、互換マイクロプロセッサー向けには、インテル製マイクロプロセッサー向けと同等レベルの最適化が行われない可能性があります。これには、インテル® ストリーミング SIMD 拡張命令 2 (インテル® SSE2)、インテル® ストリーミング SIMD 拡張命令 3 (インテル® SSE3)、ストリーミング SIMD 拡張命令 3 補足命令 (SSSE3) 命令セットに関連する最適化およびその他の最適化が含まれます。インテルでは、インテル製ではないマイクロプロセッサーに対して、最適化の提供、機能、効果を保証していません。本製品のマイクロプロセッサー固有の最適化は、インテル製マイクロプロセッサーでの使用を目的としています。インテル® マイクロアーキテクチャーに非固有の特定の最適化は、インテル製マイクロプロセッサー向けに予約されています。この注意事項の適用対象である特定の命令セットの詳細は、該当する製品のユーザー・リファレンス・ガイドを参照してください。改訂 #20110804


最適化に関する注意事項

# インテル® MPI ライブラリー概要

## 合理化された製品のセットアップ

- `root` もしくは一般ユーザー ID でインストール
- `mpivars.(c)sh` スクリプトによる簡単なパスの設定

## 簡単なプロセス管理

- `mpirun` スクリプトが自動的に Hydra プロセス管理を使用します
- システム、ユーザー、セッションごとの設定ファイル

## 実行時に制御できる環境変数

- プロセスのピニング
- 最適化された集合操作
- ドライバーごとのプロトコルしきい値
- 集合アルゴリズムのしきい値
- 強化されたメモリー登録キャッシュ

などなど…

# コンパイルとリンクコマンド
## インテル® MPI ライブラリー

コンパイラーを使用

```
mpiicc, mpiicpc, mpiifort
```

GNU* コンパイラー
(インテル® MPI ライブラリーと同じ場所にある) を使用

```
mpicc, mpicxx, mpif77, …
```

使いやすさ

- コマンドはインテル® MPI ライブラリーの include ファイルを自動的に検出
- コマンドはインテル® MPI ライブラリーを自動的にリンク

```
mpiifort -o testf test.f
```

コマンドは、直接パスではなく PATH (もしくはオプションで指定) のコンパイラーを起動

例: インテル® Fortran コンパイラーを使用してコンパイル

# 実行コマンド
## インテル® MPI ライブラリー

### すべて込み

```
mpirun -f hostfile -n #  // # 個のプロセスで実行
```

- 一般的なシナリオ
  - 利便性が高い
  - デフォルトで新しい Hydra プロセス管理を使用
  - バッチシステムのジョブに最適
    "セッション中" モード: `mpirun` は、バッチシステムからノードリストを取得

### 例:

- プログラムを実行

```
$ mpirun -f hosts.file -n 2 ./testc
Hello world: rank 0 of 2 running on node1
Hello world: rank 1 of 2 running on node1
```

# プロセスの配置
## インテル® MPI ライブラリー

### 簡単なプロセス配置

```
mpirun [-perhost #ppn] -n # // # 個のプロセスで実行
```

### 引数セットを使用してプロセスを配置

```
mpirun -n #p1 -host node1 exe1 :-n #p2 -host node2 exe2
```

- 引数セット (":" で区切る) は、"ローカル" オプションのセットを定義
  - ローカルオプションは、現在の引数セットにのみ適用
  - グローバルオプションは、すべての引数セットに適用

### 設定ファイルでプロセスを配置

```
$ cat theconfigfile
-n #p1 -host node1 exe1
-n #p2 -host node2 exe2
# -n #p3 -host dead_node3 exe3
-n #p4 -host node4 exe4

$ mpirun -configfile
theconfigfile
```

# ファブリックの選択
## インテル® MPI ライブラリー

環境変数 I_MPI_FABRICS は、実行時にインターコネクト・デバイスを選択します

I_MPI_FABRICS の値:

- shm (共有メモリーのみ)
- dapl (DAPL ファブリック)
- tcp (ソケット)
- tmi
- ofa

shm:dapl ファブリックがデフォルトです

例:

- I_MPI_DEBUG=2 で選択されたデバイスを確認

```
$ mpirun -f hosts.file -genv I_MPI_DEBUG 2 -n 2 ./testc
```

- I_MPI_FABRICS=shm:ofa で選択されたデバイスを変更

```
$ mpirun -f hosts.file -genv I_MPI_DEBUG 2 -genv I_MPI_FABRICS shm:ofa -n 2 ./testc
```

# 軽量の統計
## インテル® MPI ライブラリー

I_MPI_STATS をゼロ以外の整数値にすると、MPI 通信の統計情報を収集 (最大値は 10)

解析の有効性を高めるため、I_MPI_STATS_SCOPE で結果を操作

推奨値: I_MPI_STATS=3、I_MPI_STATS_SCOPE=coll

```
$ mpirun -genv I_MPI_STATS 3 -genv I_MPI_STATS_SCOPE coll …
```

```
~~~~ Process 0 of 16 on node compute-0-
0.local
lifetime = 751561.16

Data Transfers
Src     Dst       Amount(MB)       Transfers
-----------------------------------------
--
000 --> 000      0.000000e+00       0
000 --> 001      1.398373e-02       277
000 --> 002      1.371384e-02       240
[…]
==========================
Totals               6.158352e-02    1031

Communication Activity
Operation       Volume(MB)       Calls
-----------------------------------------
Collectives
Allgather        7.629395e-06      2
Allgatherv       0.000000e+00      0
Allreduce        1.678467e-03      216
[…]
==============
```

どこが通信のホットスポットであるか判断

最大の消費者は?

実際の引数による通信アクティビティー

```
Collectives
Operation   | Context | Algo | Comm size  | Message size | Calls | Cost(%)
-----------------------------------------------------------------------------
Allgather
1           0         4      16           4              1       0.20
2           96        4      16           4              1       0.21
Allreduce
1           124       1      16           72             1       0.03
2           124       1      16           8              6       0.13
3           124       1      16           4              7       0.13
4           112       1      6            8              1       0.09
5           112       1      6            4              1       0.00
6           96        1      16           8              1       7.02
Barrier
1           124       2      16           0              1       0.01
2           108       2      3            0              1       0.05
[…]
=============================================================================
```

最大のコストは?

# パフォーマンス・チューニング: mpitune

## インテル® MPI ライブラリー

クラスターやアプリケーション向けにインテル® MPI ライブラリーをチューニングするため自動チューニング機能を使用 (一度だけ行う、時間がかかる場合あり)

モード (mpitune -h オプションを参照)

- クラスター全体のチューニング

```
mpitune …
```

- アプリケーション固有のチューニング

```
mpitune -application ¥"mpirun -n 32 ./exe¥" …
```

-tune フラグとともに使用されるオプション設定を作成

```
mpirun -tune …
```

# インテル® MPI ライブラリーは、インテル® Xeon Phi™ コプロセッサーをサポート

# MPI + オフロード
## インテル® MPI ライブラリー

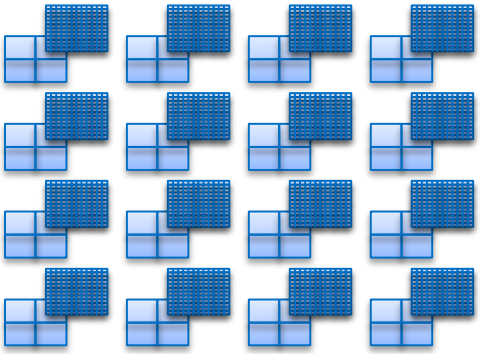
インテル® Xeon® プロセッサー上 (のみ) の MPI ランク

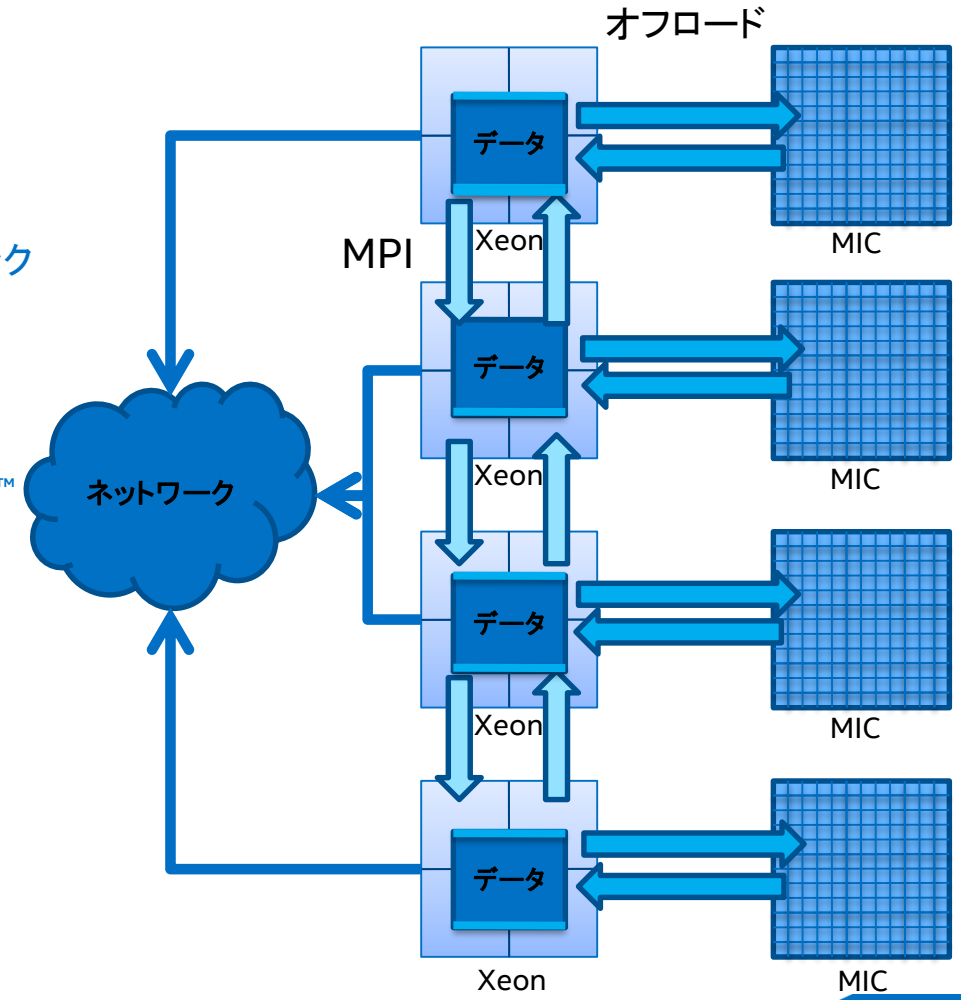
プロセッサー間とのすべてのメッセージ

オフロードモデルは、MPI ランクを加速します

インテル® MIC アーキテクチャー上のインテル® Cilk™ Plus、OpenMP*、インテル® TBB、Pthread*

ハイブリッドの同種ネットワーク:





# MPI + オフロード - どのように実行するか

## インテル® MPI ライブラリー

### オフロード宣言を持つコードをコンパイル

```
$ mpiifort -openmp test.f -o test.offload
```

### hosts ファイルを作成 (インテル® Xeon プロセッサーのみ)

```
$ cat hosts
node0
node1
```

### アプリケーションを実行 (インテル® Xeon プロセッサーのみ)

```
$ mpirun -f hosts -n 2 ./test.offload
```

# メニーコアホスト (ネイティブ)

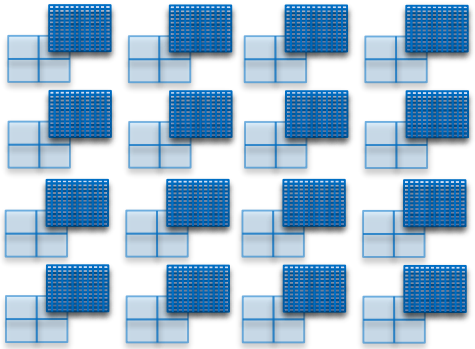
## インテル® MPI ライブラリー

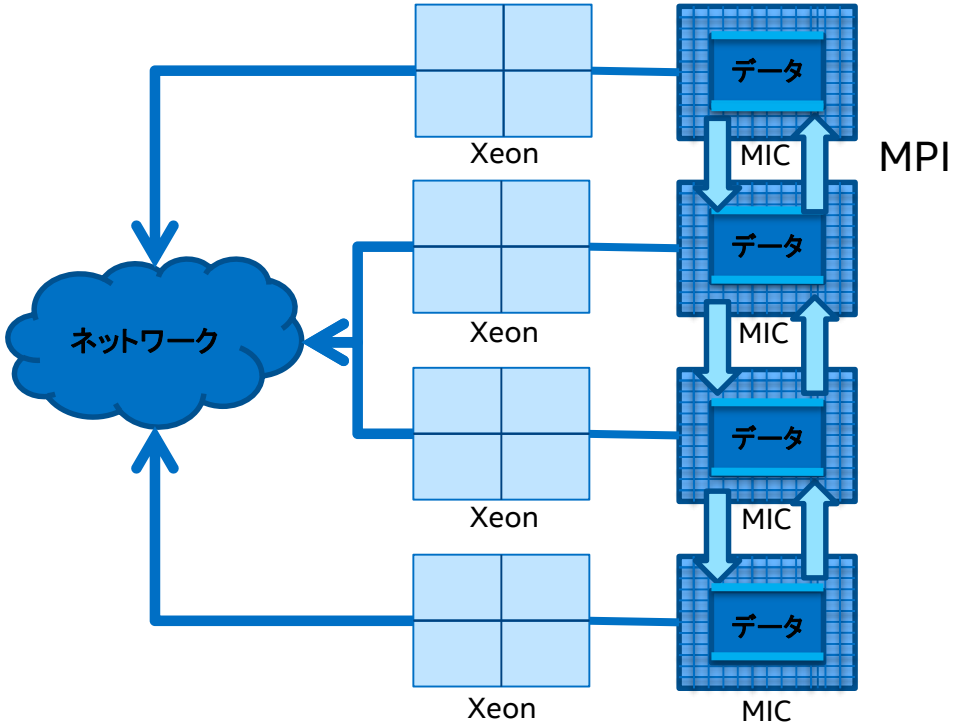
インテル® Xeon Phi ™ コプロセッサー上 (のみ) の MPI ランク

インテル® Xeon Phi™ コプロセッサーとのすべてのメッセージ

MPI プロセス内で直接使用されるインテル® Cilk™ Plus、OpenMP*、インテル® TBB、Pthread*

メニーコア CPUの同種ネットワークとしてプログラム:





# メニーコアホスト (ネイティブ) - どのように実行するか

## インテル® MPI ライブラリー

インテル® Xeon Phi™ コプロセッサー向けアプリケーションをコンパイル

```
$ mpiifort -mmic test.f -o test.mic
```

MIC で実行可能なバイナリーをコプロセッサーにコピー

```
$ scp test.mic mic0:/home/user/
$ scp test.mic mic1:/home/user/
```

`hosts` ファイルを作成 (MIC のみ)

```
$ cat hosts
mic0
mic1
```

MIC 上で実行することをライブラリーに通知

```
$ export I_MPI_MIC=1
```

アプリケーションを実行 (インテル® Xeon プロセッサー上で)

```
$ mpirun -f hosts -n 4 /home/user/test.mic
```

# シンメトリック

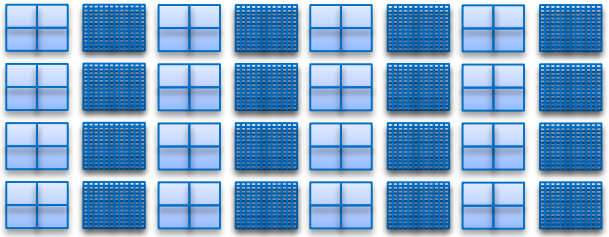
## インテル® MPI ライブラリー

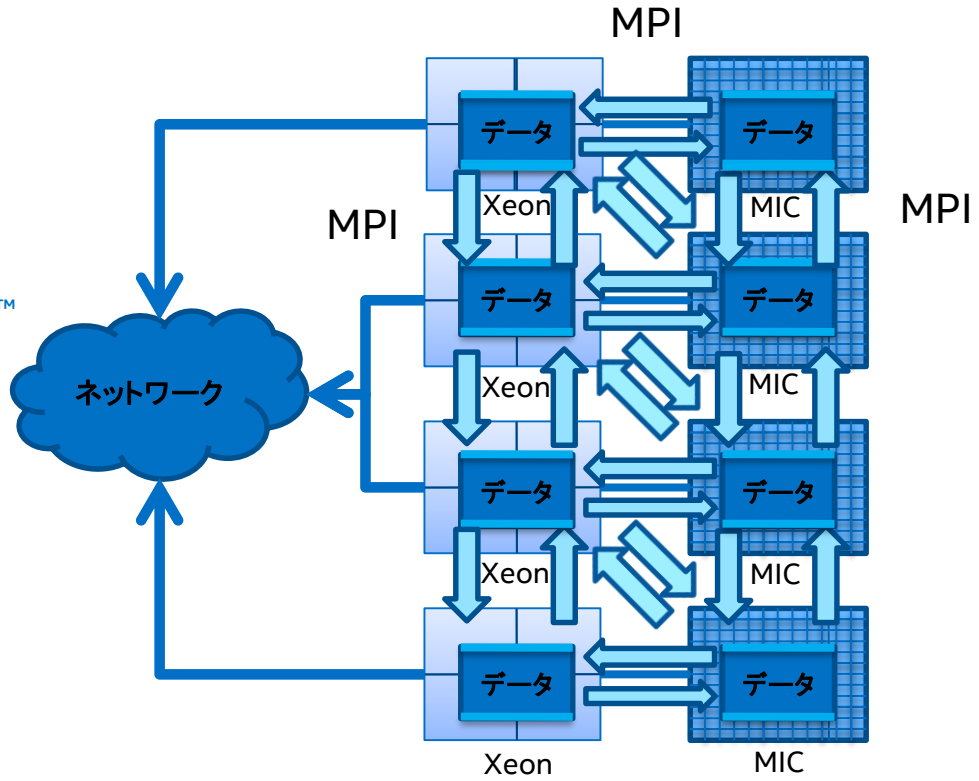
インテル® Xeon Phi™ コプロセッサーとインテル® Xeon® プロセッサー上の MPI ランク

任意のコアとのメッセージ送受信

MPI プロセス内で直接使用されるインテル® Cilk™ Plus、OpenMP*、インテル® TBB、Pthread*

同種ノードの異種ネットワークとしてプログラム:





# シンメトリック – どのように実行するか

## インテル® MPI ライブラリー

インテル® Xeon® プロセッサーとインテル® Xeon Phi™ コプロセッサー向けにコンパイル

```
$ mpiifort test.f -o /home/user/test
$ mpiifort -mmic test.f -o test.mic
```

MIC で実行可能なバイナリーをコプロセッサーにコピー (コピー中にリネーム)

```
$ scp test.mic mic0:/home/user/test
$ scp test.mic mic1:/home/user/test
```

`hosts` ファイルを作成 (インテル® Xeon プロセッサー + MIC)

```
$ cat hosts
node0
mic0
mic1
```

MIC 上で実行することをライブラリーに通知する

```
$ export I_MPI_MIC=1
```

アプリケーションを実行 (インテル® Xeon プロセッサー上で)

```
$ mpirun -f hosts -n 4 /home/user/test.mic
```

# 環境変数を介した NFS サポート
## インテル® MPI ライブラリー

### コプロセッサー上で NFS をサポートするには、2 つの環境変数が使用可能

- I_MPI_MIC_PREFIX – 実行可能ファイル名の先頭に値を追加 (直接)
- I_MPI_MIC_PREFIX – 実行可能ファイル名の最後に値を追加 (拡張子)

### プロシージャー:

- Set I_MPI_MIC=1
- 通常のようにジョブを実行

```
mpirun … ./app args
```

- ホストノードは、指定されたコマンドを起動

```
./app args
```

- コプロセッサー・ノードは、変更されたコマンドを起動

```
$I_MPI_MIC_PREFIX./app$I_MPI_MIC_POSTFIX args
```

# 複雑な実行向けの設定ファイル
## インテル® MPI ライブラリー

設定ファイルは、異なる MPI オプション、異なる実行ファイル、異なるプログラム引数などを許可

行ごとに 1 つの引数セット、# でコメントアウト

実行コマンドは、設定ファイルのみを指定

```
$ cat theconfigfile
-n 1 -host node1 ./master
-n 3 -env OMP_NUM_THREADS 8 -host node1 ./worker
-n 4 -env OMP_NUM_THREADS 60 -host node1-mic0 ./worker.mic
-n 4 -env OMP_NUM_THREADS 8 -host node2 ./worker
-n 4 -env OMP_NUM_THREADS 60 -host node2-mic0 ./worker.mic
$ mpirun -configfile theconfigfile
```

# 新機能

## インテル® MPI ライブラリー 5.0

### 最新の標準化をサポート (MPI-3.0)

- 非ブロッキング集合操作は、通信と計算の完全なオーバーラップを可能にする
- 隣接集合操作による通信ネットワークのサポートを強化
- キャッシュ・コヒーレントなシステムで効率良く動作するように一方向操作が向上
- 非常に大きなメッセージ (2GB 超) をサポートする新しいデータ型の追加

### 既存の MPI-2.x およびインテル® MPI ライブラリー 4.x アプリケーションと下位互換

# オンライン・リソース

## インテル® MPI ライブラリー

インテル® MPI ライブラリー製品ページと 30 日の無料評価版

- http://www.isus.jp/article/idz/hpc/intel-mpi-library/

インテル® Trace Analyzer & Collector 製品ページ

- www.intel.com/go/traceanalyzer

インテル® Cluster と HPC テクノロジー・フォーラム

- http://software.intel.com/en-us/forums/intel-clusters-and-hpc-technology

インテル® Xeon Phi™ コプロセッサー・デベロッパー・コミュニティー

- http://www.isus.jp/article/idz/mic-developer/

# 法務上の注意書きと最適化に関する注意事項

本資料の情報は、現状のまま提供され、本資料は、明示されているか否かにかかわらず、また禁反言によるとよらずにかかわらず、いかなる知的財産権のライセンスを許諾するものではありません。製品に付属の売買契約書『Intel's Terms and Conditions of Sale』に規定されている場合を除き、インテルはいかなる責任を負うものではなく、またインテル製品の販売や使用に関する明示または黙示の保証 (特定目的への適合性、商品適格性、あらゆる特許権、著作権、その他知的財産権の非侵害性への保証を含む) に関してもいかなる責任も負いません。

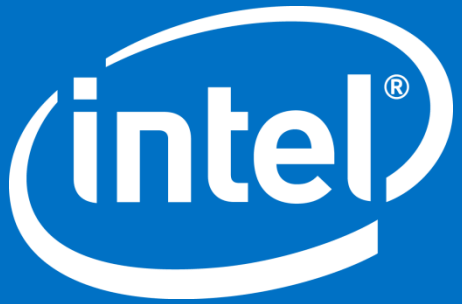
性能に関するテストに使用されるソフトウェアとワークロードは、性能がインテル® マイクロプロセッサー用に最適化されていることがあります。SYSmark* や MobileMark* などの性能テストは、特定のコンピューター・システム、コンポーネント、ソフトウェア、操作、機能に基づいて行ったものです。結果はこれらの要因によって異なります。製品の購入を検討される場合は、他の製品と組み合わせた場合の本製品の性能など、ほかの情報や性能テストも参考にして、パフォーマンスを総合的に評価することをお勧めします。



**最適化に関する注意事項**

インテル® コンパイラーは、互換マイクロプロセッサー向けには、インテル製マイクロプロセッサー向けと同等レベルの最適化が行われない可能性があります。これには、インテル® ストリーミング SIMD 拡張命令 2 (インテル® SSE2)、インテル® ストリーミング SIMD 拡張命令 3 (インテル® SSE3)、ストリーミング SIMD 拡張命令 3 補足命令 (SSSE3) 命令セットに関連する最適化およびその他の最適化が含まれます。インテルでは、インテル製ではないマイクロプロセッサーに対して、最適化の提供、機能、効果を保証していません。本製品のマイクロプロセッサー固有の最適化は、インテル製マイクロプロセッサーでの使用を目的としています。インテル® マイクロアーキテクチャーに非固有の特定の最適化は、インテル製マイクロプロセッサー向けに予約されています。この注意事項の適用対象である特定の命令セットの詳細は、該当する製品のユーザー・リファレンス・ガイドを参照してください。

改訂 #20110804


intel®