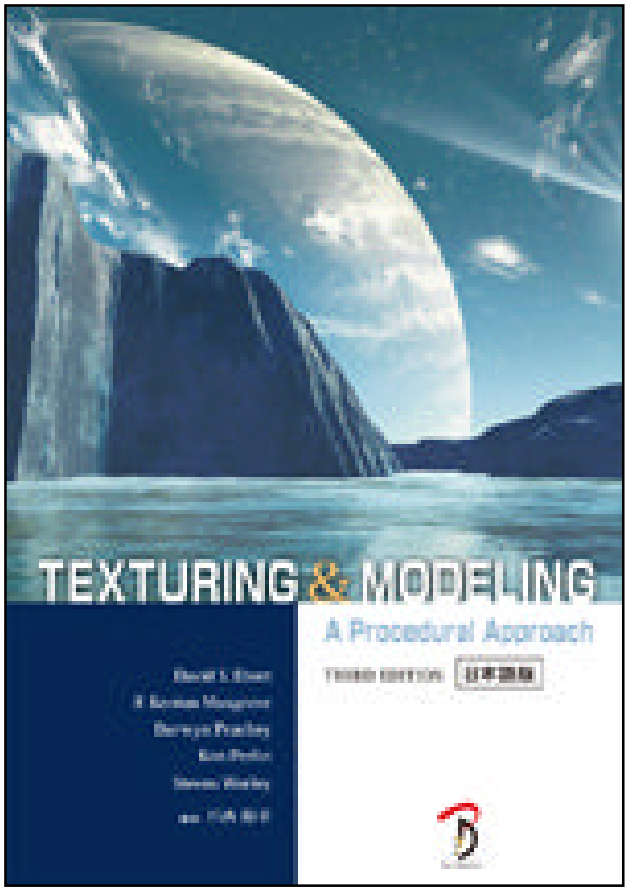
Texturing & Modeling A Procedural Approach 日本語版

- ISBN-13: 978-4862460868
- 著者
  - David S. Ebert
  - F. Kenton Musgrave
  - Darwyn Peachey
  - Ken Perlin
  - Steven Worley
- 翻訳
  - 川西 裕幸



スケーラブルな並列化

並列化は CPU にも GPU にも必須

川西 裕幸
http://blogs.msdn.com/hiroyuk/
マイクロソフト株式会社

Agenda

- はじめに
- 並列101
  - 並行性（Concurrency）
  - 並列化（Parallelism）
- 並列201
  - 共有データ
- デバッグ ツール
- GPGPU
- 結論
- Appendix

はじめに

未来のゲームテクノロジー

by Tim Sweeney @CEDEC 2008

- 2012年から2020年を考えたとき、課題は「メニーコア」と「より汎用的なグラフィックス」
  - 前者への対応は、「ソフトウェア トランザクション メモリー」や「純粋関数型プログラミング言語」の活用
  - 後者については、汎用的なC/C++で、映画に使われるような**ソフトウェア レンダリング**をハードウェアで実行するようになるだろう
- HWが20倍高速になっても、開発予算を2倍以上にすることはできないので、**生産性**のために性能を犠牲にせざるを得ないし、そのために「すごい」ツールが必要になる。

Future of Gaming Graphics

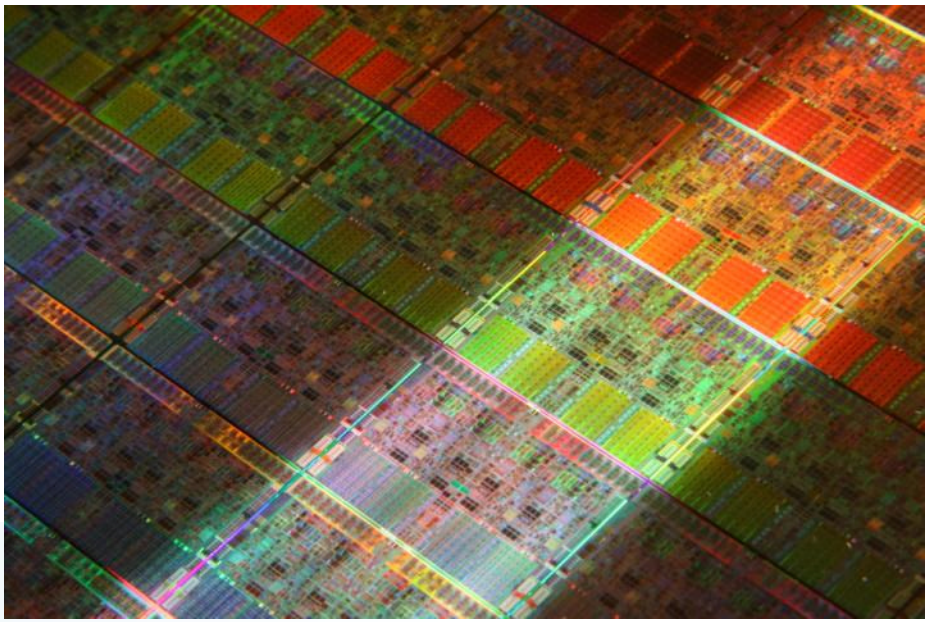
by Carl Jones (Crytek) @ CEDEC 2009

- 2012年までは大きな変化はない
  - スタイルやAI、物理に注力すべき
  - フォトリアルだけでは皆同じに見える
- 2013年以降に変化
  - GPUもCPUも高度に並列化・汎用化し、競合する
  - 並列化が課題
  - 新しいレンダリング アルゴリズムが要求される
    - Point Based Rendering
    - Ray Trace
    - Rasterizer（REYESを含む）
    - Sparse Volume Structure
    - …

ムーアの法則

「すなわち、1975年までには、最小コストで得られる集積回路の部品数は6万5千に達するであろう。私は、それほどにも大規模な回路が1個のウェハー上に構築できるようになると信じている。」
-- Intel 共同創業者 Gordon Moore in 1965

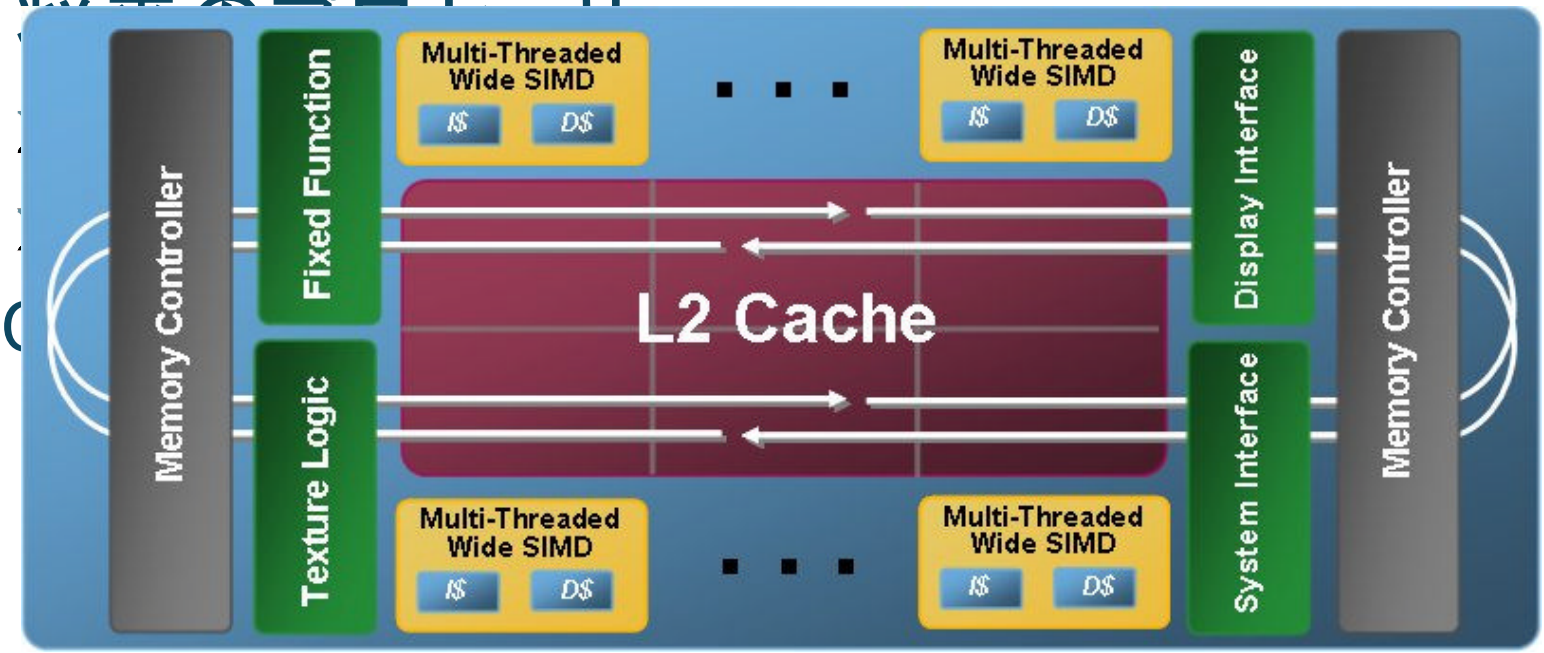
2007年IDFでアナウンスされた4コアNehalem:
7億31百万トランジスタ



メニーコア シフト

CPU	コア	HyperThread	スレッド	周波数（GHz）
Pentium 4	1	○	2	3.4（2004）
Core2 Duo	2	X	2	3.0（2006）
Core i7	4	○	8	3.2（2008）

- 将来のプロセッサ
- C



課題

- アムダールの法則
  - どんなに多数のプロセッサーがあっても、
    コンピュータ プログラムは
    並列で実行しない（直列）部分の合計より
    決して速くなることはない。
- 自動並列化コンパイラーはない
- 並列化にはコストがかかる
  - 並列化によって遅くなる場合もある

並列プログラミングの挑戦

HOT-CHIPS 2006 by Yuan Lin @Sun Microsystems

1. 並行タスクの発見と作成
2. タスクのスレッドへのマッピング
3. 同期プロトコルの定義と実装
4. 競合状態の処理
5. デッドロックの処理
6. メモリーモデルの処理
7. 並列タスクの合成
8. スケーラビリティの実現
9. 移植可能で予測可能な性能の実現
10. エラーからの回復
11. 全ての個々のスレッド問題の処理

並列101

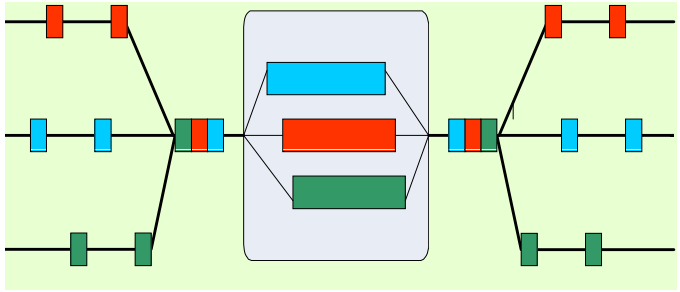
並列プログラミング 101

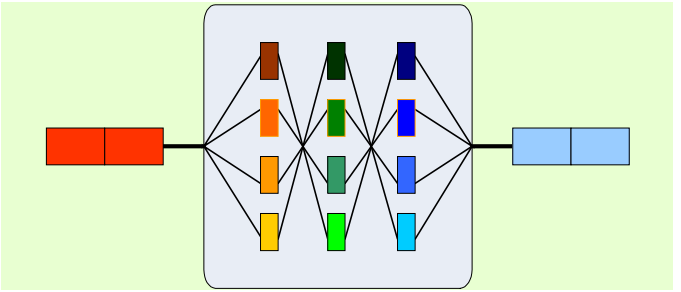
- 並行（Concurrency） vs 並列（Parallelism）
- タスク並列 vs データ並列
- 共有メモリー（SMP）vs メッセージ渡し（MPI）
- スレッド vs タスク
- 協調型 vs プリエンプティブ

並行(Concurrency) vs 並列(Parallelism)

並行: 独立な要求
(ほとんどのサーバーアプリ)

並列: 1つのタスクを可能な並行実行に分解





複数ノードの「オーナーシップ」を仲裁

並行検索を開始…

スレッドの「独立性」をシミュレーション

タスクのスケジュール化

マルチスレッド、非同期、…

タスク並列 vs データ並列

- タスク並列
  - 複数の処理を並行に実行するように並列化
- データ並列
  - 異なるデータに同じ処理を並行に実行するように並列化
- 並列化を考慮するときに焦点を当てる対象
  - 排他な概念ではない
- GPUは主にデータ並列

共有メモリー vs メッセージ渡し

- 共有メモリー（SMP）
  - 並列に処理されるタスクが同一のメモリー空間にアクセスする
  - 複数CPUシステム（nウェイ）
  - メニーコア CPU
  - GPU
- メッセージ渡し（MPI）
  - 並列に処理されるタスクが同一のメモリー空間にアクセスしない
  - 共有データをメッセージとして渡す
  - 複数ノードのHPC
  - CELL

スレッド vs タスク

- スレッド
  - ハードウェアを管理するための単位
  - システムで同時に動作するスレッド数以上にスレッドを生成しても無駄
- タスク
  - ソフトウェアが処理するための単位
  - 個々のスレッドにタスクを割り当てるしくみが必要
- ファイバー
  - 軽量でコンテキストスイッチを発生しない
  - スレッド アフィニティを持つ（あるスレッドに属する）
- プロセス
  - アプリケーション起動時に生成されるタスク
  - 利用可能な1つのスレッドに割り当てられる
- ジョブ？

協調型 vs プリエンプティブ

- 協調型
  - 自分で終了しない限り、タスクは切り替わらない
  - ユーザーモード スケジューラー
- プリエンプティブ
  - 割り込み
  - OSがタスクを横取り
  - カーネルモード スケジューラー

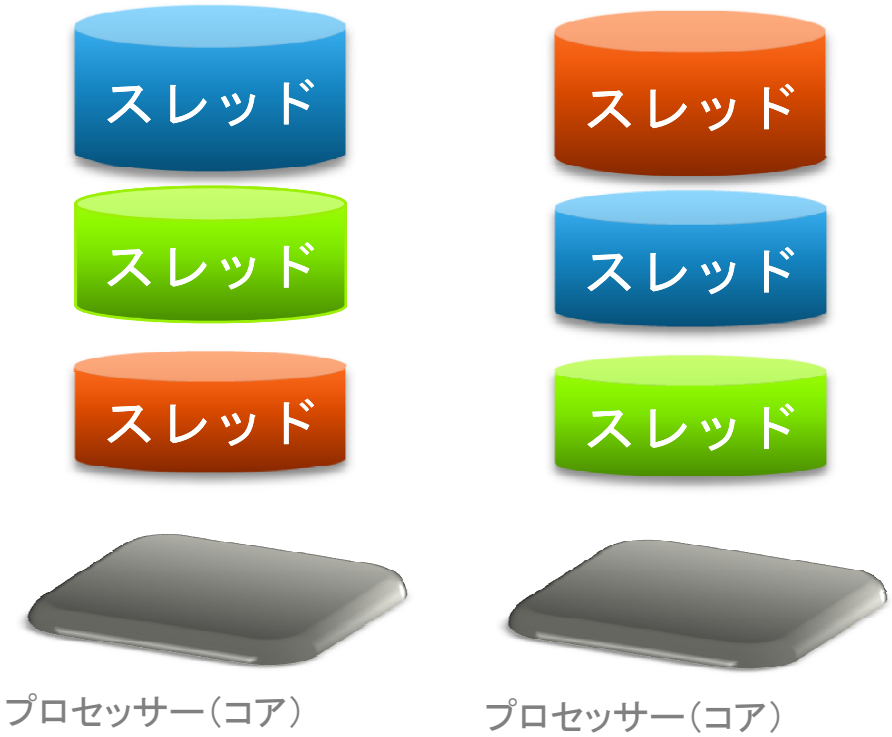
Visual Studio 2010
ツール / プログラミング モデル / ランタイム

並行性

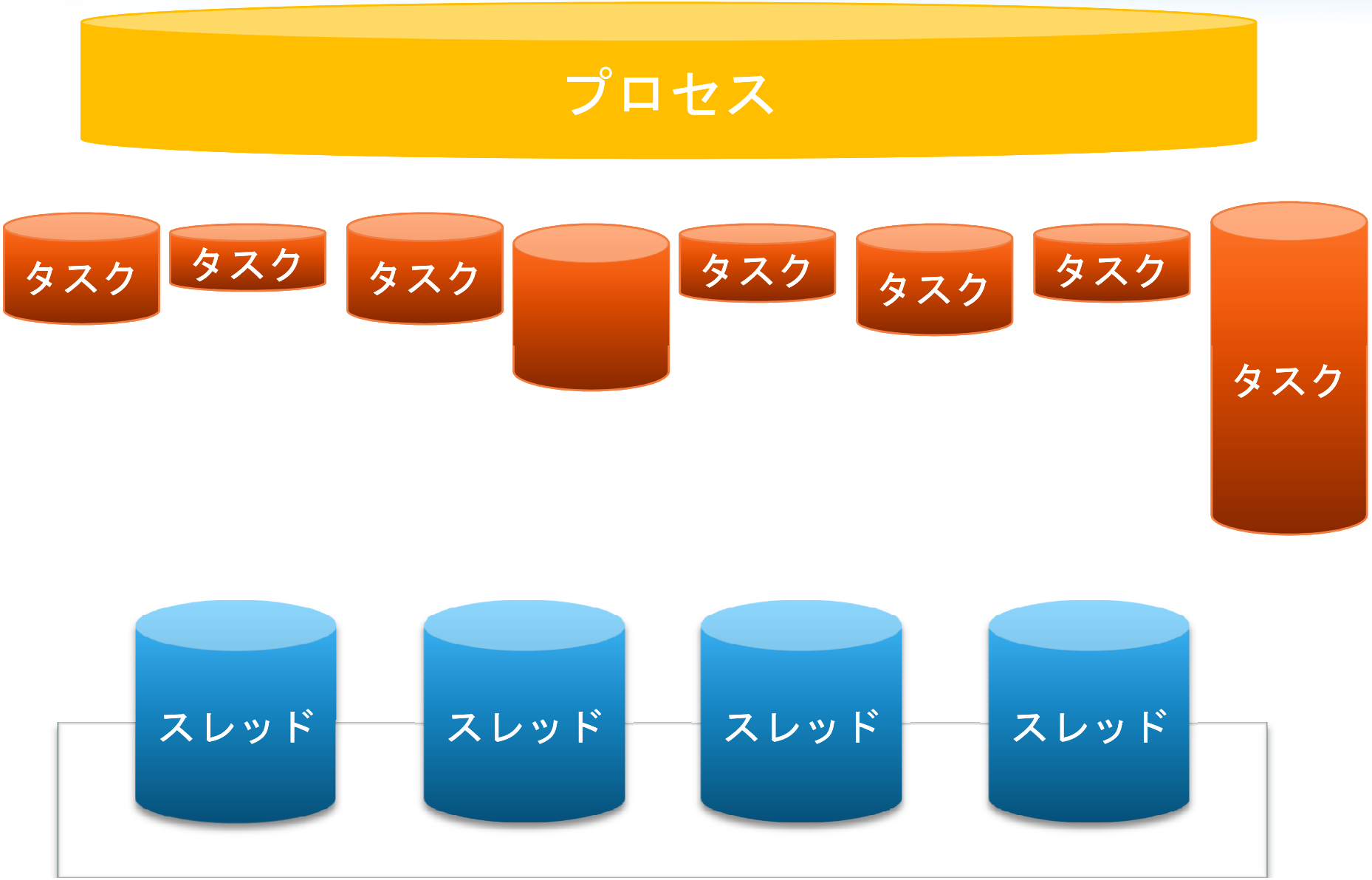
並行ランタイム

カーネルモード スケジューラー

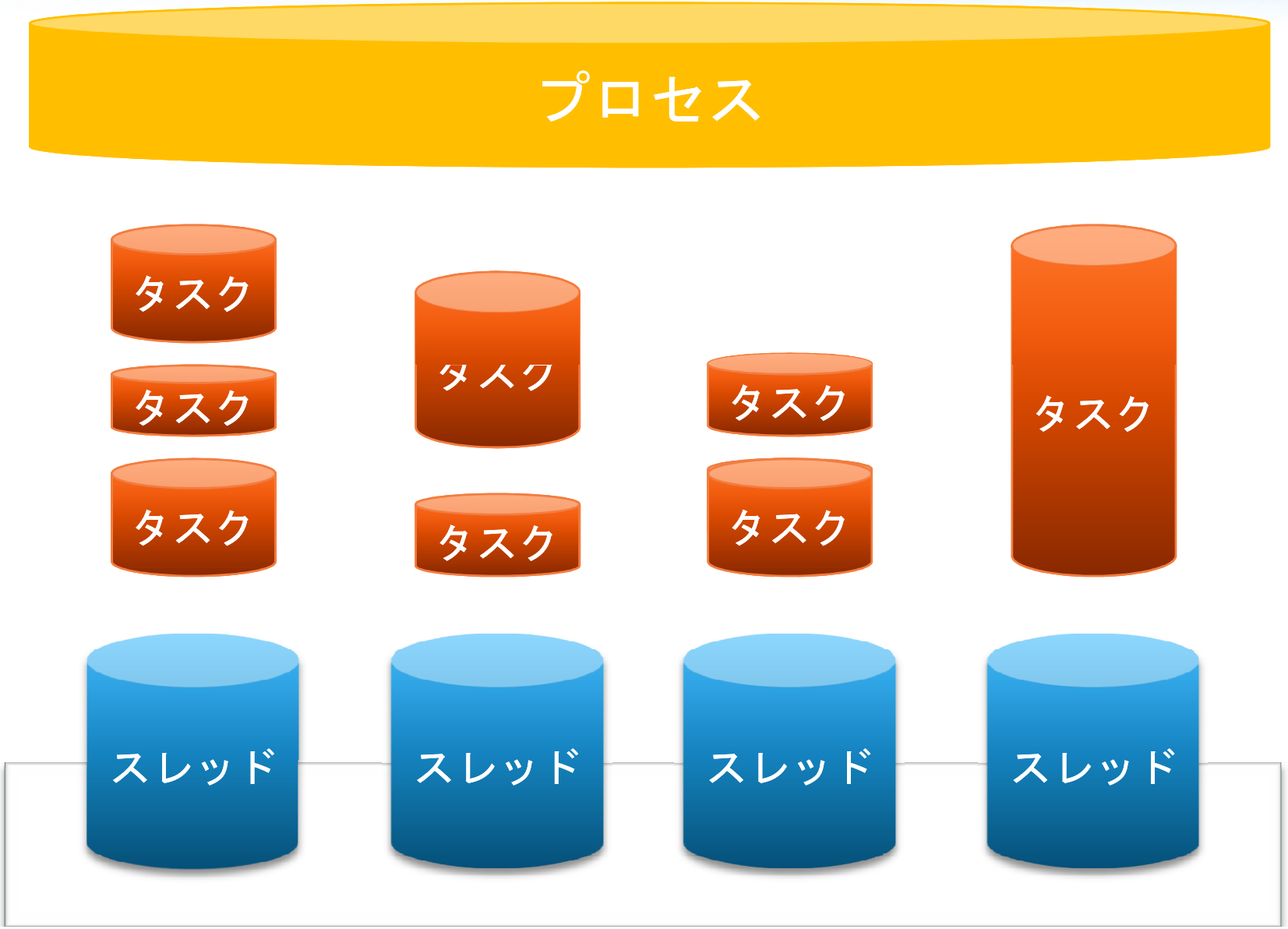
- プリエンプティブなマルチタスクを実現
- スレッドを、プロセッサーに、ある時間間隔で割り当てる
  - タイムスライス
  - コンテキスト スイッチ



並列化

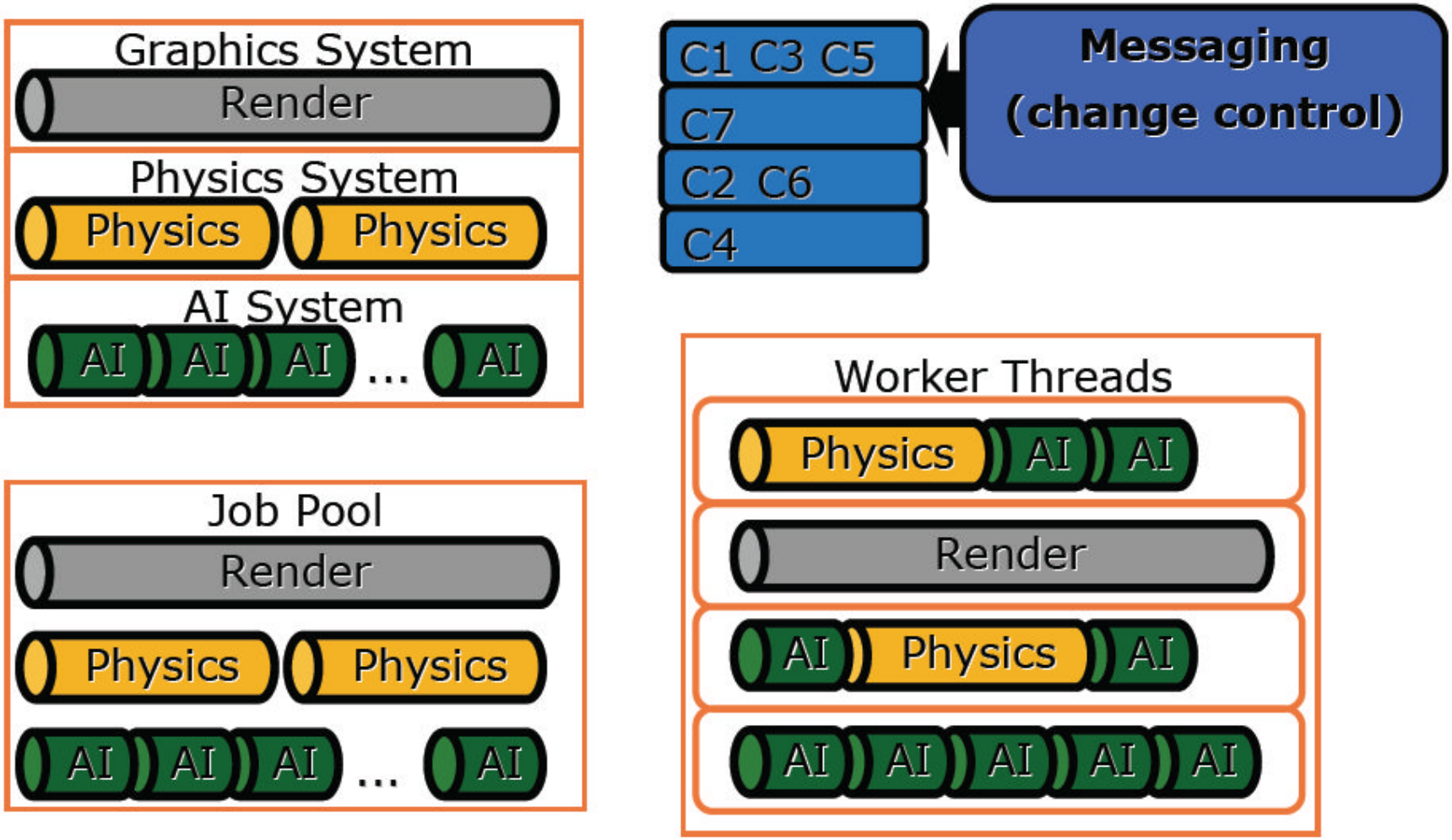


スケジューリング



ゲームのスレッド戦略

@IDF SF 2008



http://intel.wingateweb.com/US08/published/sessions/VCTS002/SF08_VCTS002_100r.pdf

ユーザーモード スケジューラ