特集
Special Feature

［自動運転元年］

❹自動運転用プロセッサの要求性能・機能・方式
―複合的処理特性を持つアプリケーションへの適応―

杉本英樹　（株）エヌエスアイテクス

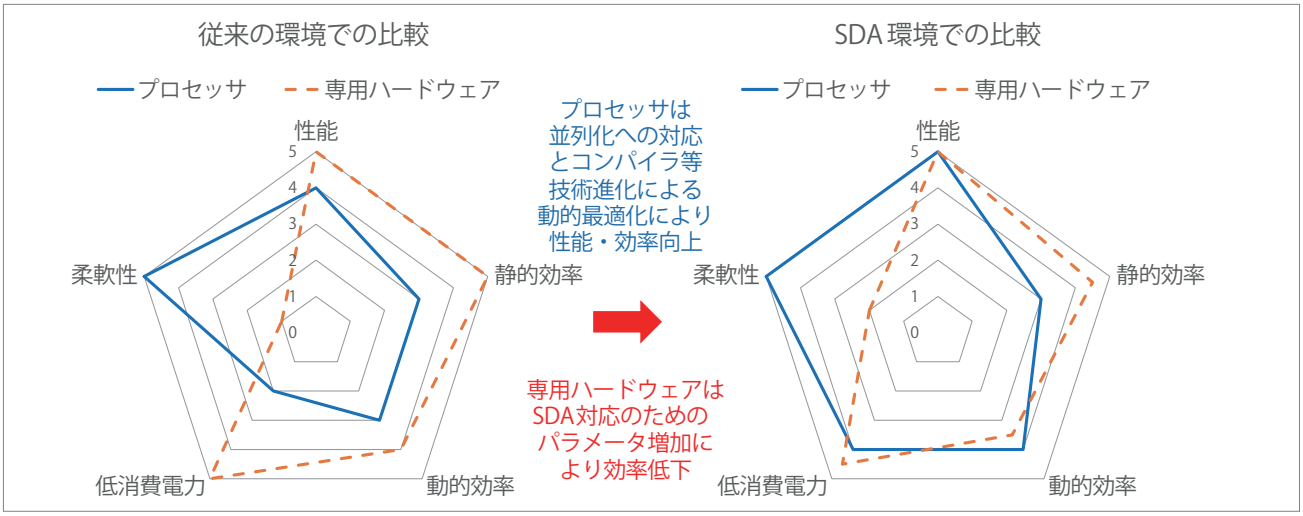
自動運転技術の現状

自動運転におけるプロセッサ

高度運転支援を含む自動運転技術の進歩は，自動車の開発スタイルにまで影響を及ぼしている．すなわちシステムに最適な，かつ高品質・高信頼性のハードウェアの実現にある程度時間を掛けられた時代から，進化速度の速いアルゴリズムやソフトウェア技術に追従でき，SDA（Software Defined Architecture）に対応可能なハードウェアが求められる時代への変化である．

そこで重要になるのがプロセッサである．先に述べた変化により，従来は効率（性能／消費電力）を重視して多用されていた専用ハードウェアがプロセッサで置き換えられる事例が増加している．さらに，プロセッサ技術の進化により，専用ハードウェアとプロセッサの機能実現の効率差が小さくなっていることもこの傾向に拍車をかけている（図 -1）．

しかし，単に専用ハードウェアを従来のプロセッサに置き換えただけでは，システムのすべての機能を実現するは困難である．たとえば，自動運転は反応速度（反応時間の短さ）が必要とされるが，元来直列処理が得意なプロセッサは短時間に処理するのを苦手とする場合がある．以降，実際にどのような



■図 -1　専用ハードウェアとプロセッサの比較

困難が存在し，どのように解決されようとしているか，現在の技術動向と将来への展望について述べる．

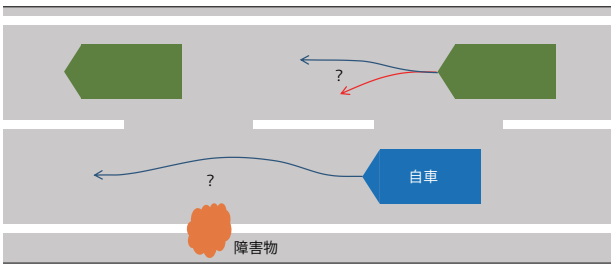
自動運転の難しさ

プロセッサについて述べる前に，自動運転技術の難しさ，すなわちプロセッサに求められる特性の背景について説明する．

自動運転の難しさはさまざまであるが，その 1 つに一度に非常に多くの走行シナリオを検討・取捨選択しなければならないことがある．また，そのシナリオ自体が自車の行動，選択したシナリオによって変化することも難しさを増す．

よくありそうな走行シナリオ（図 -2）を例に考える．左側の障害物を避けつつ車線内を進行することは，現時点で物理的には可能な状況である．

さて，障害物を避けて車線の右側ぎりぎりを走行しようとしたときに発生し得るシナリオにはどのようなものがあるであろうか．そのまま通過できることが多いかもしれない．しかし，左から車が寄ってきたことに驚いた右側の車のドライバが挙動を乱すかもしれない．また，障害物が風などで右に移動してくる可能性も 0 ではない．



■図 -2　自動運転での走行シーンの例

以上のようなさまざまな想定，しかも時間軸に沿ったケースの分岐を含めた非常に多くのシナリオから最も適切なものを選択する必要がある．リスクが少しでもある場合は停止するということも考えられるが，いくら安全であっても少しも動けないアルゴリズムでは実用にならないのである．

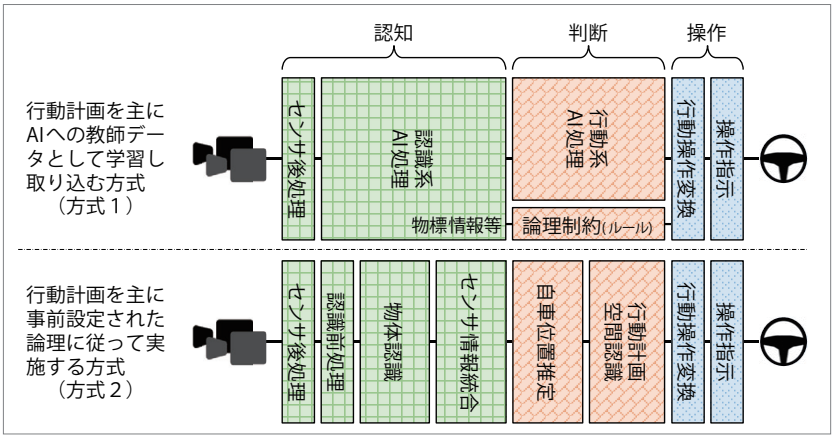
このように，非常に多くの，かつそれぞれパターン・特性の異なるシナリオを評価し，さらに短時間で適切なものを選択するのには，非常に多くの処理を同時に行う必要がある．このことが自動運転のリアルタイムにかつ大量のデータ処理が必要というやや特異的な特性の元となっている．

自動運転処理の特性

次に，自動運転の具体的な処理特性の例を見ていこう．実際使用されるアルゴリズムは多種多様であるが，センサ入力からアクチュエーションに至る処理とデータの流れを図に例示した（図 -3）．細部ではさまざまな方式が存在するが，ここに示すように大きくは 2 種類程度に代表される．

左側のセンサ入力からは，画像を中心に非常に多くのデータが入力される．この大量のデータから必要な情報，たとえば物の種類や自車を含む位置の特定（物標情報と呼ぶ）を抽出したりするのが認知の工程である．この工程の特徴は大量のデータを扱う一方で，出力となる情報は相対的に少ないことにある．たとえばカメラの画像は非常に多くの画素情報（色・明度等）を入力してくるが，その中で本当に必要な情報は相対的に少ないというような事情である．

次の工程は判断である．この部分が先に述べたようにさまざまなシナ



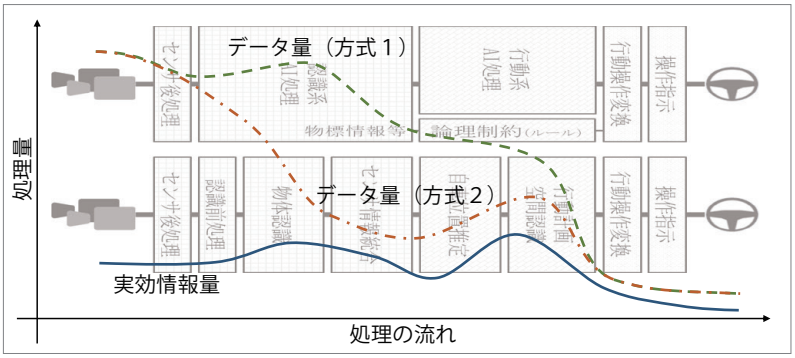
■図 -3　自動運転の処理パイプラインの例

リオを想定して適切な行動を選択する部分である．このシナリオ検討を認知と同じ空間（データセット）で主に AI（Artificial Intelligence）を用いて処理する方式と，物標情報とあらかじめ用意された論理的な知見（ルールセット）を用いて探索する方式が存在する．前者の場合は認知に似た特性，後者の場合にはデータ量よりも処理フローの複雑度が処理時間を支配する特性となる．

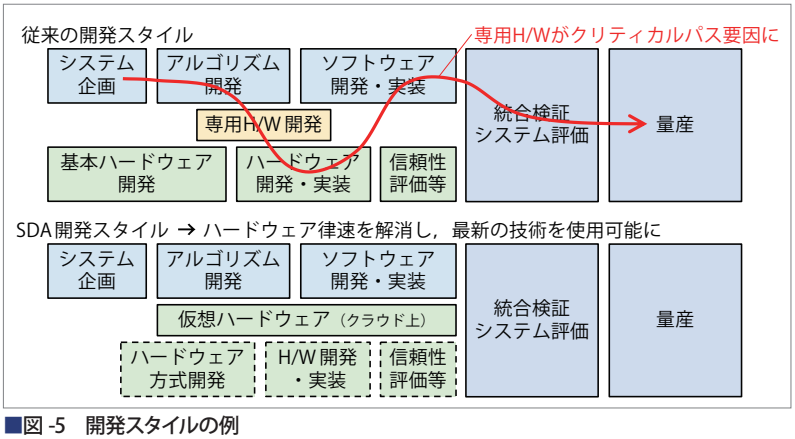
最後の工程は操作であり，判断工程で導出した行動シナリオに沿って車両に制御指示を出す部分である．この工程は，現時点では判断工程の出力の再解釈（行動から操作指示への変換）にすぎない場合が多く，データ量も情報量も大きくない場合が多い．

これまで説明してきた３つの工程の流れについて，データ量と情報量および処理の複雑度（規則性・相互依存性の低さ）をグラフに表す（図 -4）．

このように，工程によってデータ量と情報量のバランス，そしてその処理特性が異なることが，これまで専用ハードウェアが使われることが多かった理由であり，同時にプロセッサで処理する場合の難しさにもつながる．



■図 -4　自動運転の処理特性



■図 -5　開発スタイルの例

開発の現場では

システム開発の動向

プロセッサの要件として，処理特性と同時に考えなければならないのは，システムの開発スタイルまたはメソドロジへの対応である．

自動運転に使用されるアルゴリズムの進化は依然速く，アルゴリズムの進化による価値向上は，ハードウェアの最適化による価値向上よりも大きい場合が多い．このため，アルゴリズムの開発は，組込みシステムで多いターゲットシステム（または相当する疑似環境）上ではなく，PC 上など汎用的かつ性能余力のある環境で行われる．したがって，開発されたアルゴリズムは，ターゲットシステムよりも，むしろ PC に最適化された実装となる場合が多い．さらに，PC 上での開発は機能ごとに行われることが多く，システムとして統合した場合の性能（特に実時間性）は検証されていないこともある．

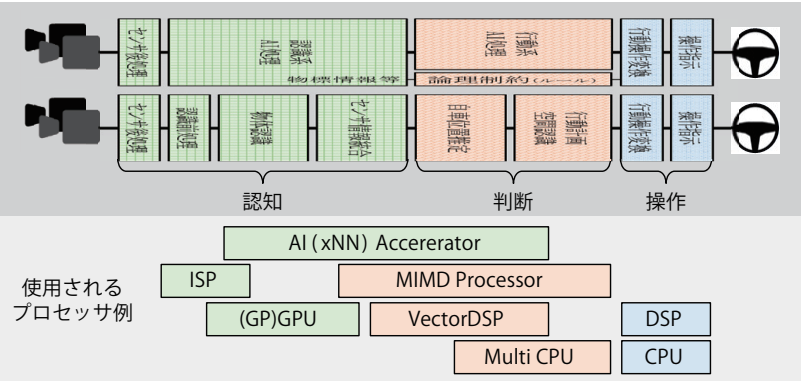
結果として，ターゲットシステム上での開発ではあまり発生しなかった，ターゲットへのソフトウェアの最適化とターゲットシステム（主にハードウェア）のソフトウェアへの最適化のトレードオフ関係が，開発フェーズの後期で発生するようになってしまった．この開発スタイルの違いにより，図示（図 -5）するように，従来であればある程度固まったアルゴリズムを想定してハードウェアを開発できたものが，あらかじめアルゴリズムの変化，およびそれによる処理特性の変化を

想定してターゲットシステム／ハードウェアを開発することが必要になったのである．

ハードウェアの動向

では，アルゴリズムの変化，およびそれによる処理特性の変化想定したハードウェアをどう実現するのか．結論を言ってしまうと，ヘテロジニアスなマルチプロセッサシステムというのがその答えの 1 つであり，1 つの主流になりつつある．

先に述べた，自動運転の処理パイプラインに，処理特性から見て適切と思われるプロセッサを当てはめたものを図示する（図 -6）．実際には各工程の中でもさまざまな処理があり，単純な 1 対 1 対応が最適解ではない．しかし，処理特性のバリエーションという意味では，ほぼすべての処理が何らかのプロセッサでカバーできる．



■図 -6　自動運転の処理パイプラインとプロセッサ

この構成の良いところは，個々のアルゴリズムが変化して処理特性が変化しても，総量の見積もりさえ大きく誤らなければ同じハードウェアが使用できるという点である．正に SDA のコンセプトに合った構成なのである．さらに専用ハードウェアを使用した構成に比べても，同じ条件（処理特性・開発スタイル）であれば，大きく効率が下がることもなくなっている．

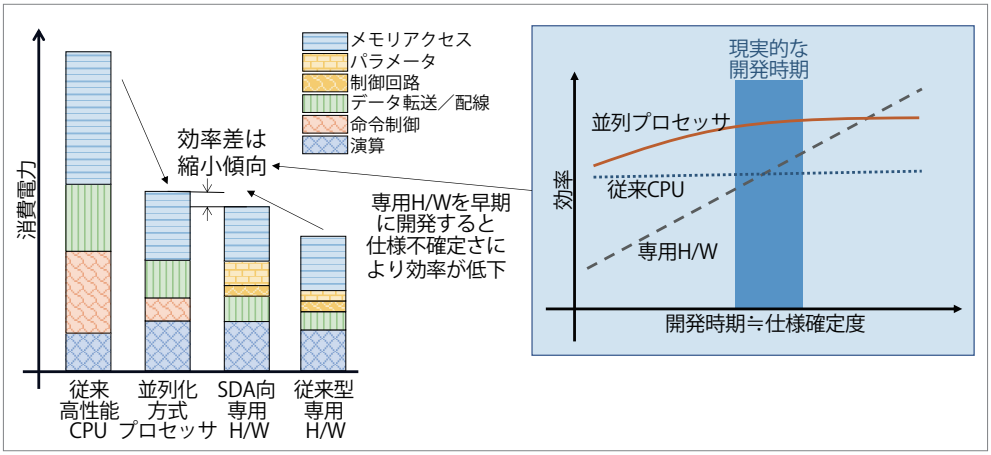
これには以下のような理由がある（図 -7）．専用ハードウェアは，アルゴリズムの変化にある程度追従しようとすると，処理パラメータ（ハードウェア的には設定レジスタ）やそれに伴い条件処理を実施するための回路が増加し，効率が下がること．対して，プロセッサの場合は，コンパイラ技術の進化や性能向上により，動的にハードウェアを最大限に活かすことができること．さらに，半導体の微細化により，トランジスタコストは低下し，同時に消費電力の大半が演算ではなくデータ移動によるものになっていることである．

プロセッサ要求と実現方式

自動運転を支えるプロセッサ

それでは，具合的に自動運転システムで実際に使用されているプロセッサについて見ていこう．

先に述べたように，ヘテロジニアス・マルチコア構成となっており，特性のほぼ全域をカバーした構成となっている（図 -8）．これにより，アルゴリズムやソフトウェアの変化に対する対応力を高めているのである．



■図 -7　専用ハードウェアとプロセッサの効率比較

実際の構成は図 -3 で説明した主に判断工程の処理方式の差異に応じて，AI（DNN ; Deep Neural Network や RNN ; Recurrent Neural Network）の性能を特に重視したものと，AI を意識しつつも論理的（ルールセット）処理にも利用できるようにしたもの等に分かれている．また，変化が少ないと思われる処理には，依然専用のハードウェアが割り当てられる場合もある．

プロセッサとして注目すべきなのは，さまざまな方式で並列処理を実現している点であり，実時間処理が必要なシステムへの適応を求めた結果であると言える．

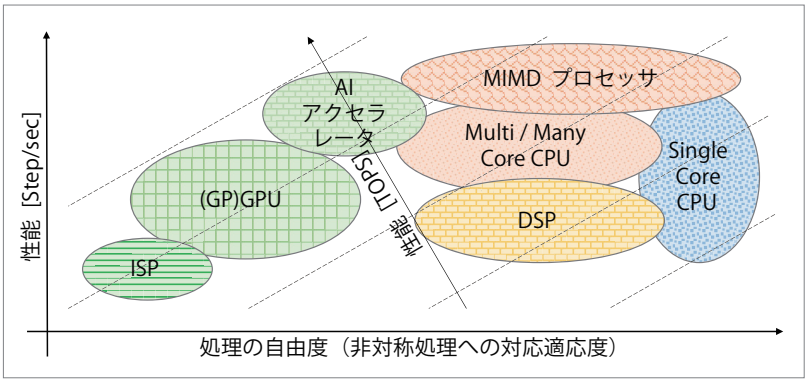
プロセッサの方式と性能

では，各プロセッサはどのような並列処理化の方式を採用しているのであろうか．処理の並列化を考えるときに重要となるのは，処理グラフ（ある機能を実現するのに必要な処理の集合を，その依存関係で結んだグラフ）である．

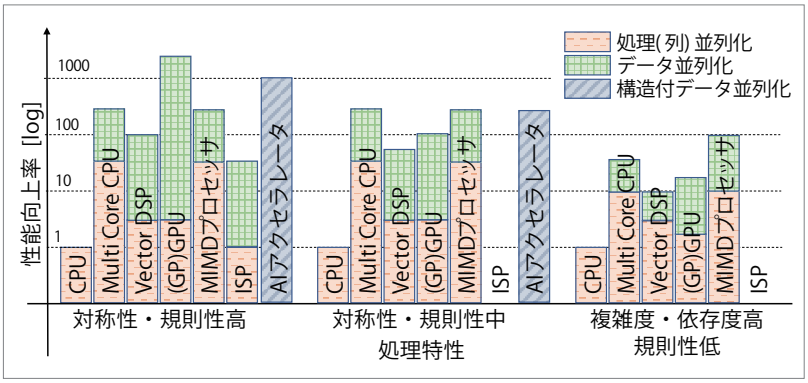
たとえば，センサ入力はデータ量が非常に大きいが，データの内容は対称的，すなわちデータの素片が持つ情報・特性が均一である．このため，データ並列という複数のデータに対して同一の処理を同時並行して行う多用される．この方式を採用する代表的なプロセッサとしては，ISP（Image Signal Processor）や GPU（Graphics Processing Unit）があり，一度に（1 命令で）数千のデータを同時に処理することができる．これにより，同じデータ量を処理するのに要する時間を数千分の一にすることができる．さらに，AI 処理のデータ構造に特化して，複数のデータ並列処理を同時に実現して並列度を向上させた AI プロセッサも開発競争の中で実用化が始まっている．

一方で，判断し工程で使われる経路探索やその判断材料となる論理的な知見（ルールセット）を探索・適用する処理などは，相対的にデータ量が少なく，一方で複数の複雑な処理グラフの組合せ処理が必要になる．このため，CPU（Central Processing Unit）を多数並べて複数の処理グラフを同時に実行するマルチコア構成や，最近では後述する MIMD 系の構造を持つプロセッサが使用される場合が多い．

これらのプロセッサの並列化方式と性能を，処理特性ごとに図示する（図 -9）．



■図 -8　自動運転で使用されているプロセッサ



■図 -9　プロセッサの並列化方式と性能

処理グラフと SIMD と MIMD

ここで，最近増加している MIMD（Multiple Instruction stream, Multiple Data stream）系のプロセッサについて，その特徴を述べる．

命令粒度での並列化手法としては，大別して SIMD（Single Instruction

stream, Multiple Data streams）方式と MIMD 方式がある．

ここで，各方式の処理の違いを図で説明する（図 -10）．SIMD は，センサ入力のようなデータ並列に適した内容の処理（図左に示す）だけでなく，同じ処理を異なるデータに対して繰返す内容の処理（図中央に示す）にも使用される．この方式は，1 命令で多数のデータを一度に処理できるために命令数が少なくて済み，効率が良いという特徴がある．

しかし，実時間性を要する処理の場合，図中央に示したような，本来異なる時間で扱うべき処理を同時にまとめてしまうのでは，実時間性が失われてしまい都合が悪い．このために，図右に示すような，複数の処理グラフを同時に並行して処理できる MIMD 方式のプロセッサが使用される．この方式は，並列化できる処理内容の自由度が高いという特徴がある．一方，処理ごとに別の命令が必要となる．実際には，どちらかの方式だけで適切に並列化できることは少なく，データ並列も併用した SIMD 命令を持つ MIMD プロセッサが使用される場合も増加している．

今後の展望とプロセッサへの期待

自動運転の技術は，まだまだ進化途中であり変化の激しい状態がしばらくは続くと思われる．このような中で，従来専用ハードウェア指向であった組込み業界でも SDA 化，プロセッサ化の流れが始まっている．一方で，実時間処理の多い組込み系特有の制約もあり，プロセッサに求められる要件も多種多様である．

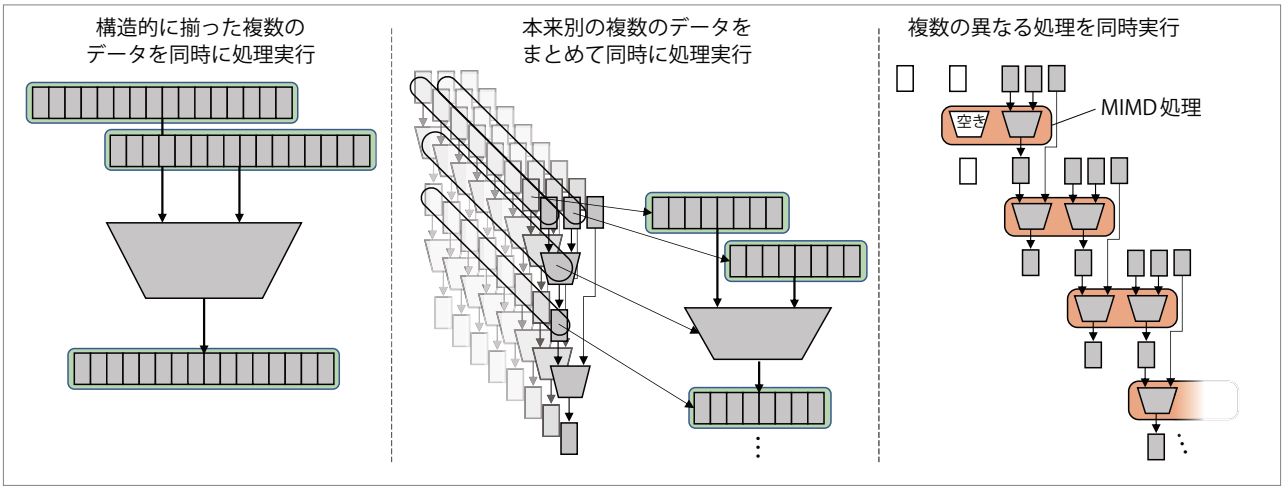
しかしながらそれらの要件に合った新しいプロセッサを導入しようとしても，プロセッサ，そしてそれを取り巻く基本ソフトウェアや開発環境などのエコシステムの開発・普及には多大な費用がかかり，安易に新しい方式が使用できるとは限らない．

システムの要件を満たすために，既存のエコシステムを活用しつつより適したプロセッサをどう実現していくのか．これまでのハードウェア開発とは異なる視点での発想力と開発力が，今後の自動運転システムの進化・発展のためには重要であると考える．

（2021 年 9 月 25 日受付）

■杉本英樹　hideki_sugimoto@nsitexe.co.jp

（株）エヌエスアイテクス取締役兼 CTO．長年プロセッサ・アーキテクトとして主に組込み向けプロセッサの開発を牽引．現在はユーザ視点から，将来の自動車を始めとする組込み系システムに求められるプロセッサの検討および新規技術の研究開発に従事．



■図 -10　SIMD と MIMD での並列処理