SYNOPSYS®
Silicon to Software™

ASIP Designerを用いた RISC-V ISAプロセッサの実現とその拡張

RISC-V Day Tokyo 2020

日本シノプシス合同会社
シニア・アプリケーションエンジニア
伴野　充

2020/11/5,6

“ドメインに特化したプロセッサ*“の時代

(*): term coined by David Patterson

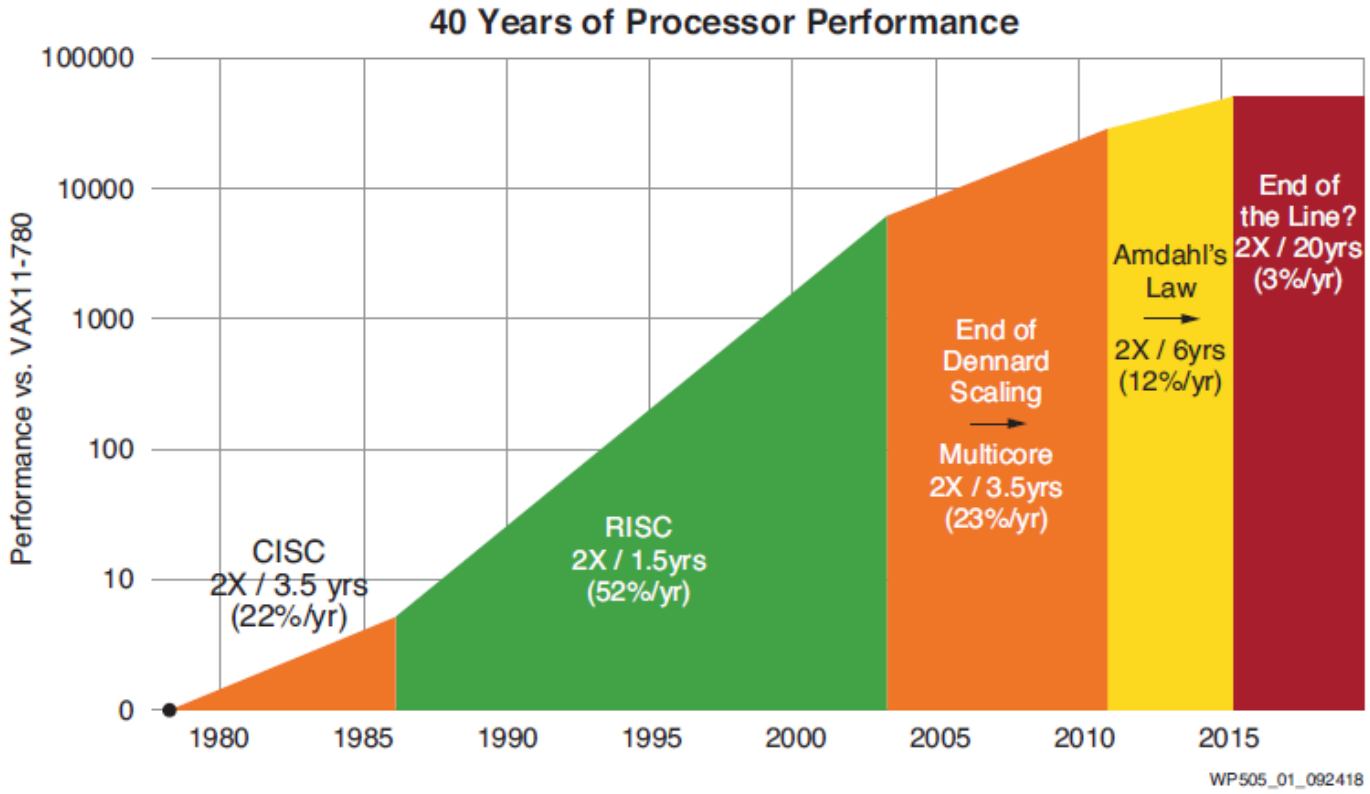


Figure 1: **Processor Performance vs. Time**

Source: Xilinx Whitepaper 505 – Versal ACAP
(reference to J. Hennessy, D. Patterson, *Computer Architecture: A Quantitative Approach* (6th Edition, 2019)

- 1995 – 2003:

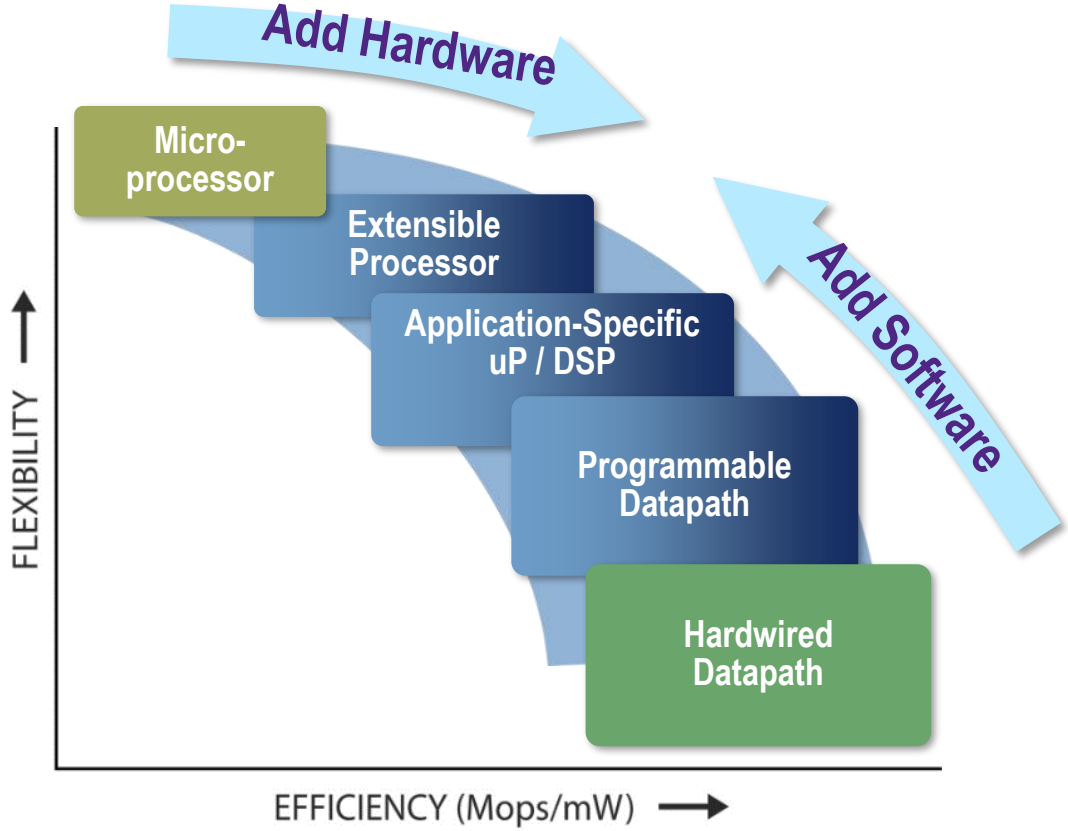
  プロセス技術による性能（より高い周波数だが同じ電力消費）およびメモリアクセスの最適化（Dennard Scaling）
- 2003: Multicore

  タスクの並列化制限に到達（Amdahlの法則）
- 2015: Domain-Specific Processors

  ヘテロジニアス・マルチコア設計への移行

Application-Specific Processors (ASIP)

ハードウェアとソフトウェアの融合



Application-specific instruction set processor

From Wikipedia, the free encyclopedia

An **application-specific instruction set processor** (**ASIP**) is a component used in system-on-a-chip design. The instruction set of an ASIP is tailored to benefit a specific application. This specialization of the core provides a tradeoff between the flexibility of a general purpose CPU and the performance of an ASIC.

2019年11月19日発表：分子動力学シュミレータ向けASIP

SYNOPSYS®

SUCCESS STORY

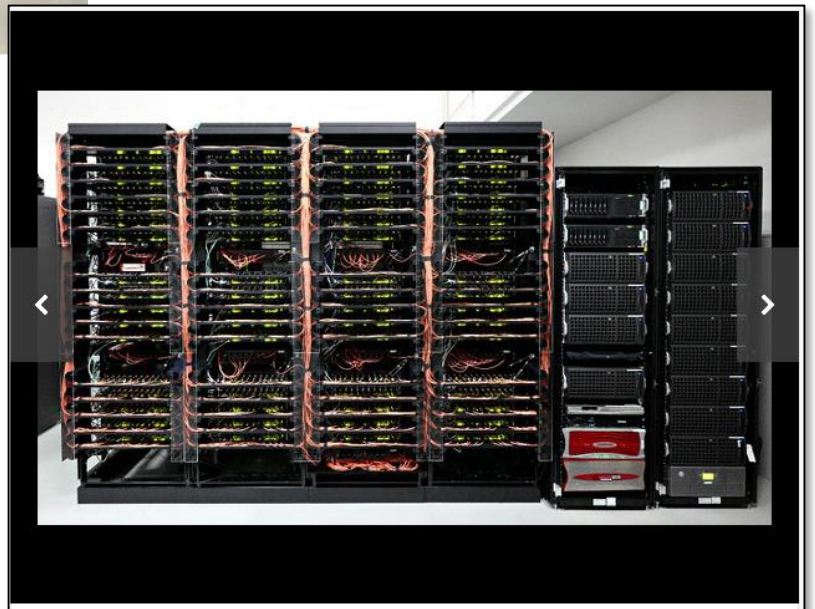
シノプシスと理化学研究所

シノプシスの ASIP Designer を用いて理研が分子動力学シミュレータ用カスタム・プロセッサを6ヶ月以内で開発

“今回の開発プロジェクトに与えられた期間は非常に厳しいものでしたが、シノプシス社のASIP Designerを使えば我々の非常に特殊なアーキテクチャの実装が可能になると信じていました。ASIP Designerにより、命令セットをチューンナップし、我々の特殊なアルゴリズムを既存のプロセッサよりも30倍高速に実行することが可能になりました。これにより、重要な生体分子の相互作用のシミュレーションに必要な計算時間を年単位から数週間単位へと大幅に短縮することができるのです”

国立研究開発法人 理化学研究所 生命機能科学研究センター 計算分子設計研究チーム チームリーダー
泰地 真弘人（たいじ・まこと）氏

RIKEN



理研、創薬専用スパコン開発 「RISC-V」アーキテクチャ採用、10万原子の挙動再現

ITmedia NEWS 11/19(火) 19:41配信

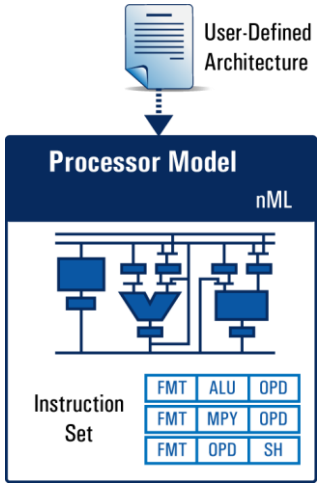
水やタンパク質など分子の動きのシミュレーションに特化した専用計算機「MDGRAPE-4A」

ASIP Designer

ASIP Designer

アーキテクチャ構造記述言語(nML)により、

アプリケーションに特化した命令セットを持つ
専用プロセッサ/ DSP を自動生成する設計環境



nML記述

```
action {
  stage E1:
    vecd =
     vsub(vecr=V[r],vecs=V[t])@vec;
    V[t] = vect = vabs(vecd) @vabs;
  }
  syntax : "vadiff v"t ",v"r ",v"t;
  image  : t::r;
```

ASIP設計開発ツール
“ASIP Designer”

命令セット
シミュレータ(ISS)

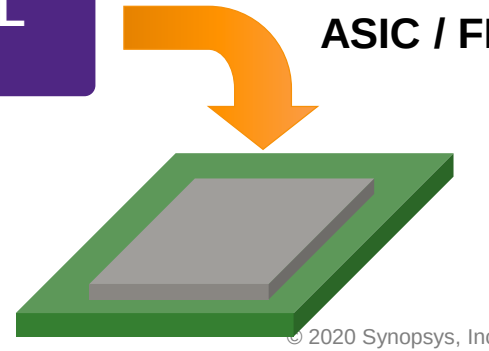
SystemC ベース
System Simulator

C/C++ コンパイラ
アセンブラ/リンカー

デバッガ/プロファイラ

Verilog / VHDL
RTL コード

ASIC / FPGA



ご提供可能なExampleプロセッサ

種類	説明
マイクロコントローラ	
Tnano	16-bit microcontroller, lightweight and configurable
Tmicro	16-bit microcontroller, fully featured
DLX (family)	Variants of 32-bit microcontroller
Tmcu	32-bit microcontroller
Trv32 (family)	32-bit microcontroller featuring RISC-V ISA, 3 or 5 pipeline stages
Trv64 (family)	64-bit microcontroller featuring RISC-V ISA, 3 or 5 pipeline stages
PD_Triop	32-bit microcontroller with 64-bit address spaces
DSP及び並列演算エンジン	
Tdsp	16/32-bit DSP
Tvec (family)	Variants of SIMD processor
Tvliw (family)	Variants of VLIW processor
暗号処理エンジン	
Tsec	High throughput SHA/RSA/AES accelerator
Tcript	High throughput AES accelerator
その他	
Tmotion	Accelerator of motion estimation kernel
Tcom8	SIMD processor optimized for some communication kernels
FFTcore	Scalar implementation of complex FFT
Mxcore	Matrix processing ASIP for communication kernels
Primecore	SIMD implementation of prime-factor algorithm for FFT & DFT
Tgauss	Vectorization and memory management for image processing
Tvox	Accelerator for SLAM (simultaneous localization and mapping)

nML記述
ソースコード
ご提供

豊富なExample

- マイクロコントローラ
- DSPs
- SIMD、VLIW
- マルチスレッド
- セキュリティ（AES、SHA等）
- メモリI/F、AXIバス等

Example使用のメリット

- 基本的な機能は既に設計開発済み
- 動作可能な例を参照可能
- モデリングの概念を効率的に習得
- nMLソースコードで提供されるため、ユーザーが自由に追加変更可能

RISC-V 命令セットプロセッサExample

	32ビット　データパス	64ビット　データパス
3段ステージパイプライン	Trv32p3 / Trv32p3x	Trv64p3 / Trv64p3x
5段ステージパイプライン	Trv32p5 / Trv32p5x	Trv64p5 / Trv64p5x

- サポート命令: RV64IM, RV32IM
  - 整数命令（Integer）
  - 乗算命令（Multiply）
- オプション拡張: TrvXXpYx
  - (標準) 圧縮命令(Compressed：IM**C**)
  - (カスタム) ゼロオーバヘッドループ
  - (カスタム) アドレス更新後のロード/ストア

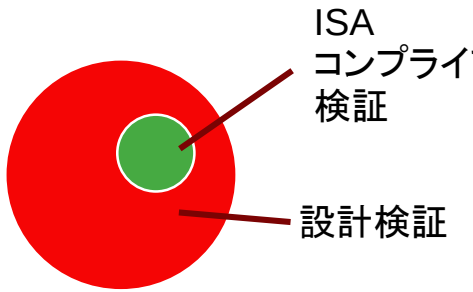
- マイクロアーキテクチャの特長
  - 5段ステージパイプライン: IF, ID, EX, ME, WB
  - 3段ステージパイプライン: IF, ID, EX
  - 保護されたパイプライン（Protected pipeline）
    - 可能な場合はレジスタをバイパス
    - バイパスが出来ない時はStall (bubble)
  - ハードウェア乗算器
    - 64x64→128→select 64bit, 32x32→64→select 32bit
  - インターリーブ除算器

目標とするASIPを実現するためのスタートポイントとして活用できます。

Trv Example モデル: RISC-V ISA Compliant

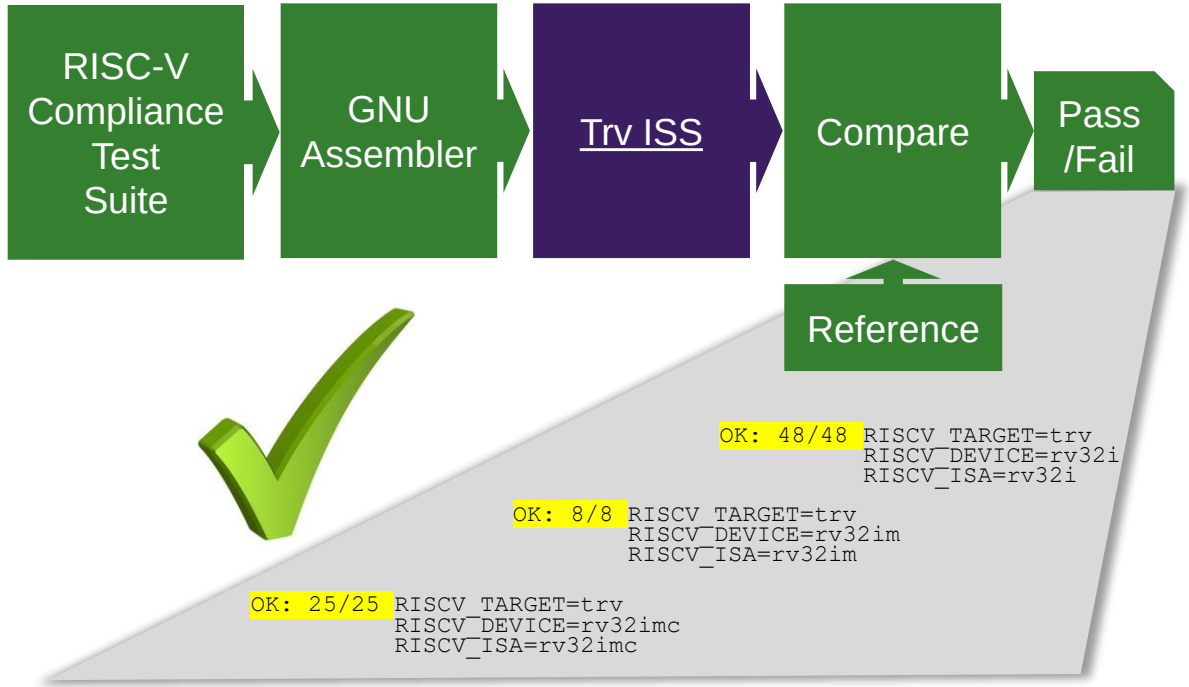
フレームワーク

- ISAコンプライアンス検証とは？
  - RISC-Vの実装がISA仕様に準拠しているかを確認
  - ISAコンプライアンスはRTL検証ではない
    - ISAテストスイートがベース
    - RTL実装にバグがないことを保障するものではない



- RISC-Vコンプライアンスフレームワーク
  - https://github.com/riscv/riscv-compliance/
  - ”このフレームワークは任意のモデルをリファレンス シグネチャに対して比較する”
  - “現状ではRV32IMC unprivileged specのみをカバー”

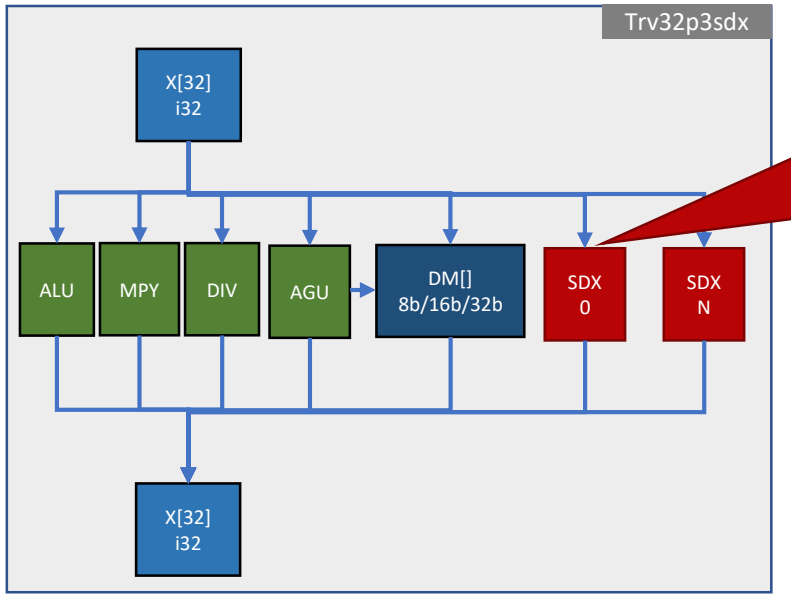
- セットアップ
  - RISC-Vテストフレームワーク内にてテストが実行される
  - ターゲットの環境はTrvのnMLモデルからASIP Desinger によって生成されてRISC-Vの命令セットシミュレータ



SDX: シンプルデータパス拡張

コンセプト

- SDXは（RISC-V）プロセッサモデルに単純な拡張命令を追加するメカニズム
  - 拡張命令スケルトンは、プロセッサモデルの一部として提供
    - ユーザーは特別なnML記述を学ぶ必要がない
  - ユーザーが拡張機能の動作をPDGとしてコーディング
    - PDGは、データパス関数の動作を定義するために使用されるCのような言語
  - 拡張命令をターゲットとするコンパイラ組み込み関数を提供
- RISC-Vの“custom-2” 命令としてエンコード
- 命令バリエーション
  - 3つのreg 32ビットおよび64ビット、累積、追加のシングルレジスタ入力および出力等々, …
- 例: SHA256 (オペレーション融合), FFT (コンプレックス処理), CNN (パッケージ化されたSIMD)



Behavioral model (PDG)

```
w32 sdx0(w32 a, w32 b)
{
  w32 r;
  r[15:0] = a[15:0] + b[15:0];
  r[31:16] = a[31:16] + b[31:16];
  return r;
}
```

Compiler intrinsics

```
int sdx0(int,int);
int add2(int,int);
```

SDX Examples

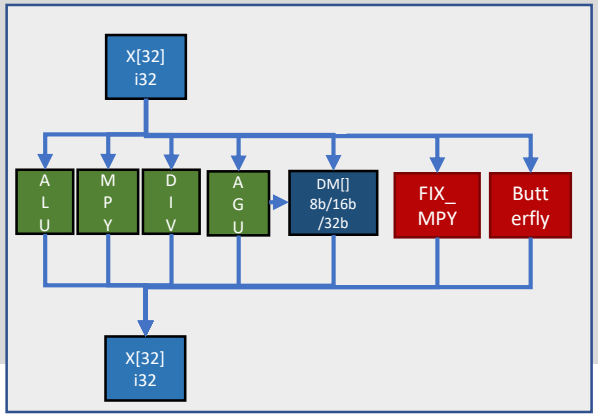
ASIP Designerに付属

FFT

SDX instructions accelerating
- Complex fixed-point multiplication & scaling
  sdx1 rd,rs1,rs2
- FFT Butterfly : *sdx5 rd,rs1,rs1*

Specialization:
- Fractional data types
- Complex numbers (16bit/16bit -> 32bit register)

Speedup: 77% Area increase: 31%



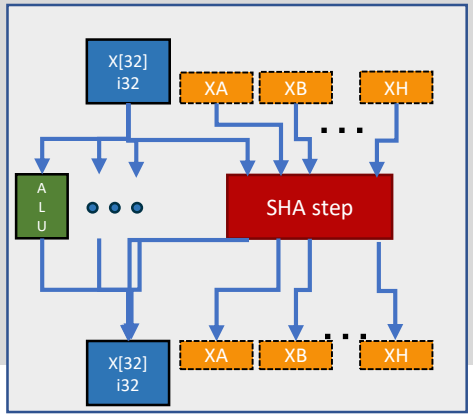
SHA256

Computes a hash of message W using bitwise AND, OR, XOR operations, shift operations and additions

Custom data path is ideal to implement the complex hash function in one instruction

Additional state of the hash functions (8 state variables) require an SDX variant that supports **8 additional register reads and writes**

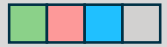
sdx7 rd, rs1, rs2, x24,x25,…, x32



Keyword Spotting

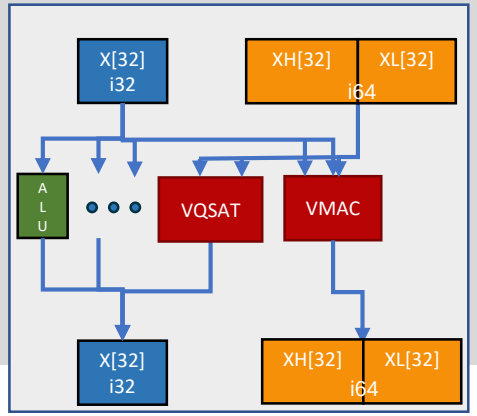
Based on small sized Neural Network (3.3M MACs)

SDX architecture feature: **packed SIMD**
32 bit register contains vector of 4x 8-bit values



Use of register pairs, enabling 64bit access

sdx4a_dr dd,rs1,rs2,mode // vmac
sdx0_dd rd, ds1,ds2 // vqsat



Tmoby アーキテクチャ

アーキテクチャの特長

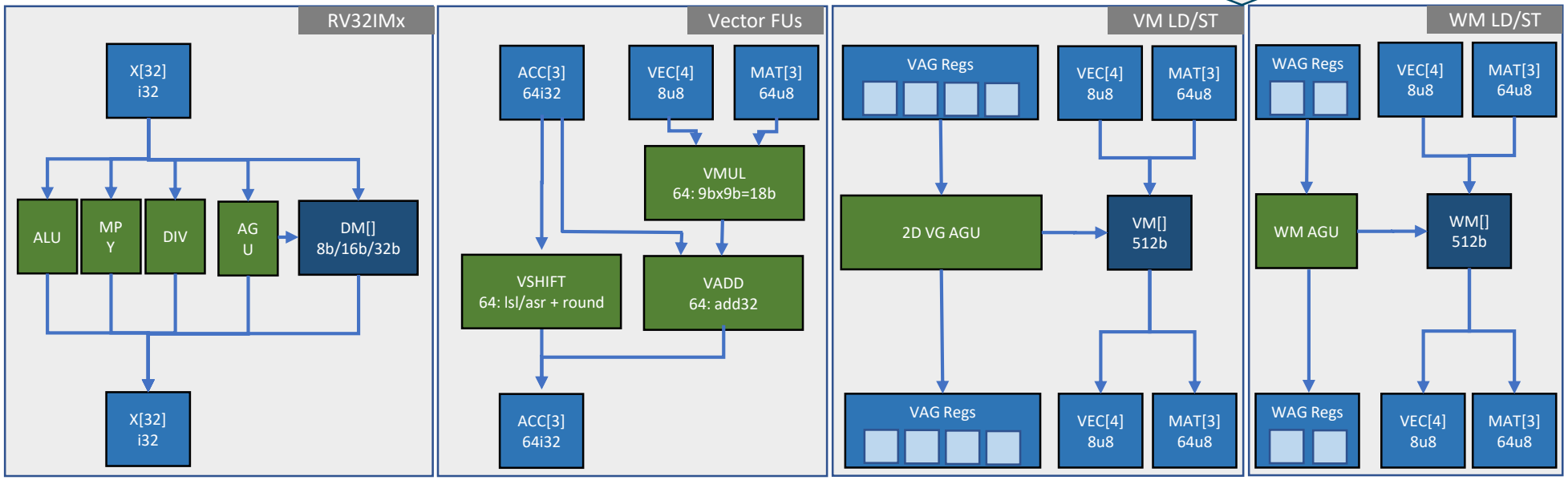
- ISA: 4並列命令
- ベクタデータパス
  - SIMD64
  - MAC 8x8→32
- メモリ
  - VM: フューチャー
  - WM: ウェイト
  - ベクタアドレッシング

アプリケーション: MobileNet V3

- TensorFlowコードを使用
- Tmoby固有のベクトル組込み関数を用いてCコードに変換

主な特長
- Graphの実装
- メモリコピー
- カーネル

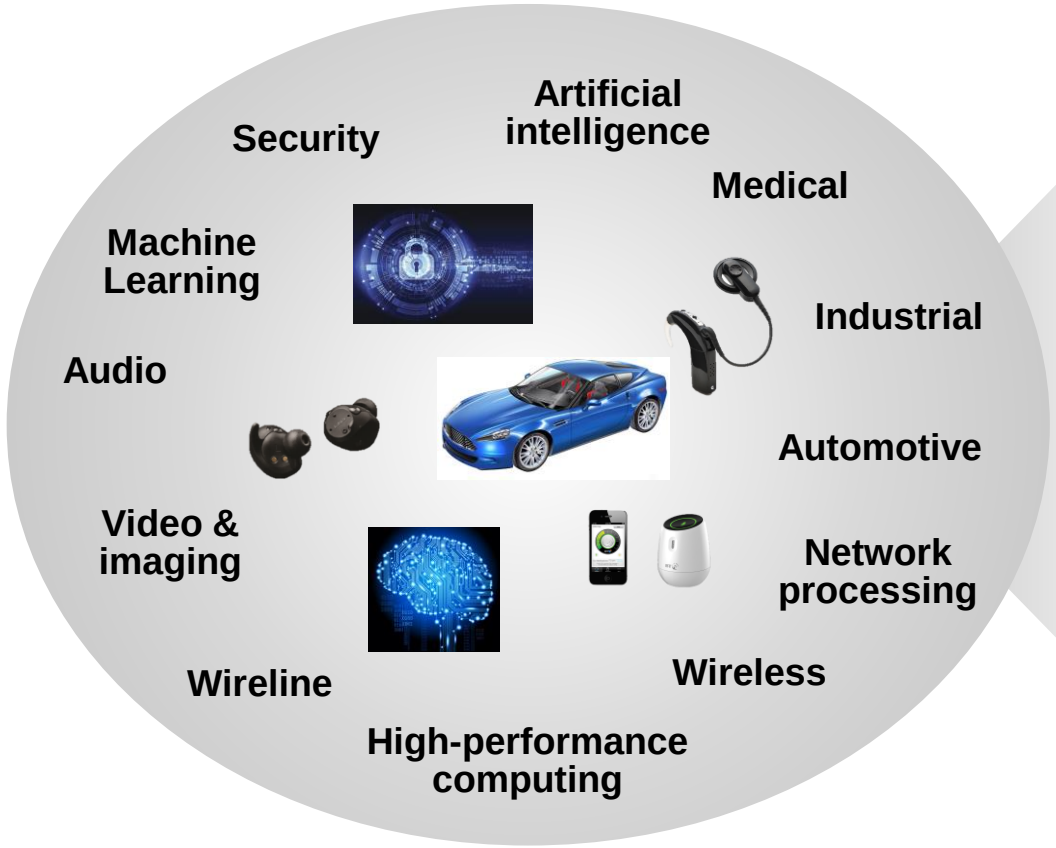
カーネル
- 2D畳込み (point-wise)
- 2DのDepth-Wise
- 加算
- 2Dのアベレージプーリング
- ソフトマックス



まとめ

シノプシスの ASIPソリューション

幅広い市場分野での採用 – 業界で実証済み



Atmel
富士ゼロックス
CONEXANT
CEVA
COGNITIVE
dialog
CohIear
HUAWEI
GN ReSound
ETRI
NOKIA
NXP
IMEC
京セラドキュメントソリューションズ
ICT
リコー
理化学研究所
生命機能科学研
NSI-TEXE
Sensata
ON Semiconductor
ST
TEXAS INSTRUMENTS
ヤマハ
SILICON LABS
富士通研究所
横河電機

幅広いアプリケーションにわたる広範な展開

ASIP Designer ー まとめ

- 差別化実現の鍵 ー 独自プロセッサ/DSP設計開発
  - nML言語によるトップダウン設計手法で効率的なASIP開発を実現
  - 複数アルゴリズムや機能をASIPに統合することでゲート数を削減
  - 必要最低限な構成と命令で超低消費電力プロセッサを実現

- ASIP Designer(AD)によるASIP開発のメリット
  - 差別化技術を含むロイヤリティフリーなRISC-Vプロセッサを実現
  - 設計開発後期のアルゴリズム変更など急な仕様変更への対応
  - 製品出荷後でもプログラムによるアップデートが可能
  - 言語設計の為、次世代の機種展開が容易（高い設計流用性）
  - 汎用DSP/プロセッサベンダによる生産中止トラブルを回避

SYNOPSYS®
Silicon to Software™

Thank You