Close to the Edge

How Neural Network inferencing is migrating to specialised DSPs in State of the Art SoCs

Marcus Binning
Sept 2018
Lund

Science Fiction → Science Fact (or Consumer Device)

The Babel Fish

"The **Babel fish** is small, yellow, leech-like - and probably the oddest thing in the universe. It feeds on brain wave energy, absorbing all unconscious frequencies and then excreting telepathically a matrix formed from the conscious frequencies and nerve signals picked up from the speech centres of the brain, the practical upshot of which is that if you stick one in your ear, you can instantly understand anything said to you in any form of language: the speech you hear decodes the brain wave matrix."

© From: "*The Hitchhiker's Guide to the Galaxy*", Douglas Adams

- One of the latest focus areas for AI is automatic language translation
  – It's a really hard problem

What is AI ?

- Merriam-Webster defines artificial intelligence this way:
  - A branch of computer science dealing with the simulation of intelligent behavior in computers.
  - The capability of a machine to imitate intelligent human behavior.
- English Oxford Living Dictionary gives this definition:
  - “The theory and development of computer systems able to perform tasks normally requiring human intelligence, such as visual perception, speech recognition, decision-making, and translation between languages.”

What are Neural Networks ?

- neural network
  - noun
  - a computer system modelled on the human brain and nervous system.

- The building blocks of AI
  - Convolutional (CNN)
  - Recurrent (RNN)
  - Long Short Term Memory (LSTM)
    - → Alphabet Soup ….

- Diverse, targeted at solving different type of problem
  - Spatial, temporal ..

The Basics of Real-Time Neural Networks:

Training vs Inferencing in embedded systems

Training: Runs once per database, *server-based*, **very** compute intensive

Server Farm

Labeled dataset

Selection of layered network

Iterative derivation of coefficients by stochastic descent error minimization

10^4-10^{10} TMAC/s

Set of coefficients (1M-1B weights)

Embedded

0.5 to 10TMAC/s

Single-pass evaluation of input image

Most probable label

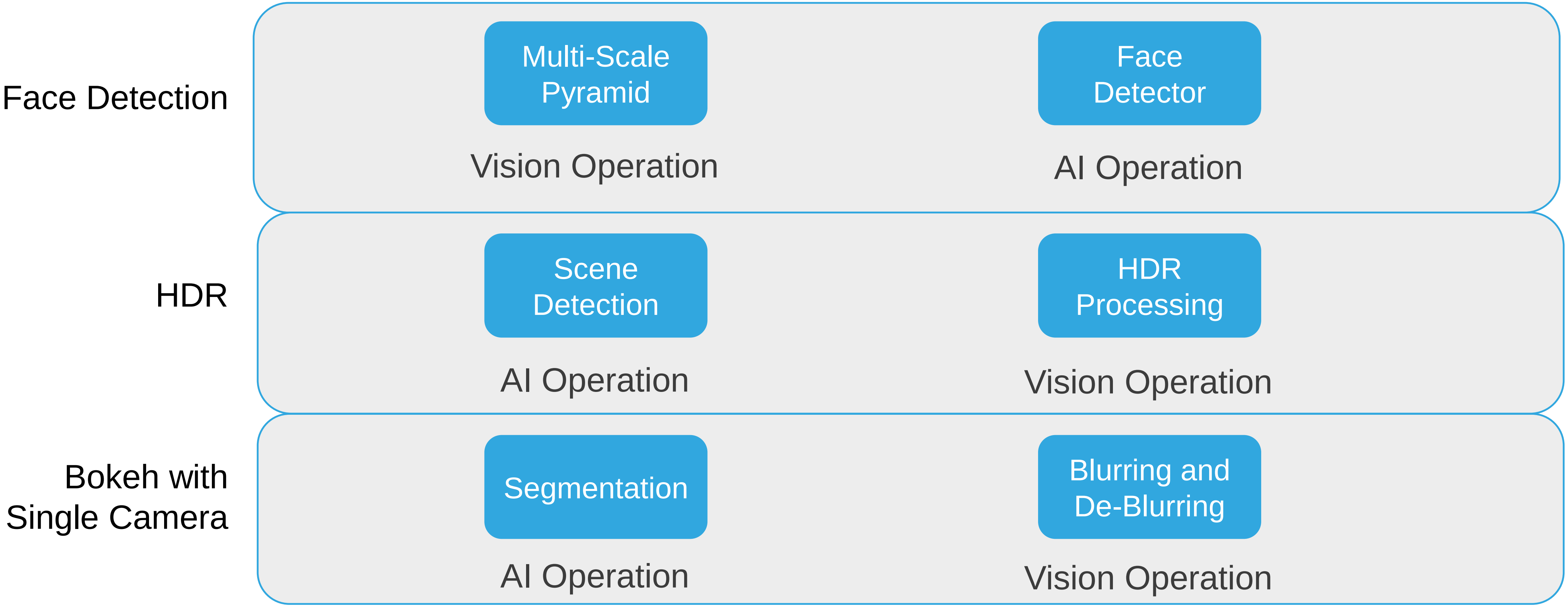
Deployment (“Inference”): Runs on every image, device based, compute intensive

Most important for embedded systems: Power

Integrated with other Processing

- AI does not exist in isolation
- Pre-processing, post processing
- Solutions need to reflect this, especially for embedded
  - The energy cost of moving data is relatively high
- Examples …

AI-Based Application Trends: Mix of Vision and AI

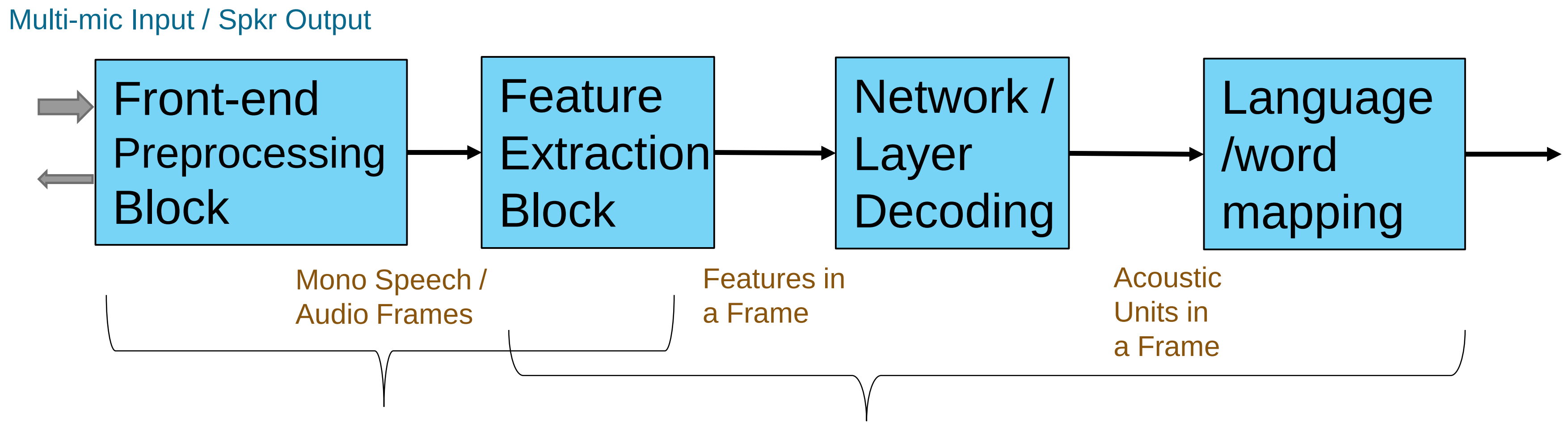


Face Detection	Multi-Scale Pyramid Vision Operation	Face Detector AI Operation
HDR	Scene Detection AI Operation	HDR Processing Vision Operation
Bokeh with Single Camera	Segmentation AI Operation	Blurring and De-Blurring Vision Operation

- Moving from traditional feature-based embedded vision to AI-based algorithm
- All use cases still have mix of vision and AI operations
- Need for both vision and AI processing in e.g. the camera pipeline

Smart Speaker Processing Chain – Audio/Voice

Mix of traditional DSP processing and AI mapped to CDNS SW components


Multi-mic Input / Spkr Output
Front-end Preprocessing Block
Feature Extraction Block
Network / Layer Decoding
Language /word mapping
Natural Language / Key Words
Mono Speech / Audio Frames
Features in a Frame
Acoustic Units in a Frame

DSP Processing

- Low level kernel library containing many commonly used components
- Includes support for math functions like tanh() etc

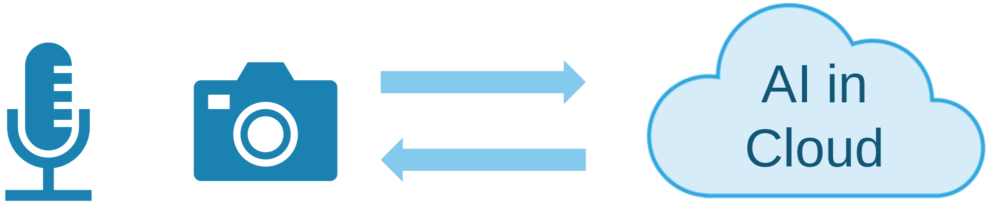
AI Processing

- Additional SW library, primarily Focused on Network/Layer decoding functions
- Also contains “few” modules to provide basic support for Viterbi/Hashing/Beam search kernels
- Imported from NDSP lib, this also contains “specific” modules instances of feature extractions modules + Sigmoid/tanh/SoftMax

Example DSPs which are commonly used in this domain:

- HiFi 3/3z
- HiFi 4
- …

Majority of AI Inferences Are in the Cloud today



AI in Cloud

“Alexa, when is my new camera arriving?”

Smart Assistant
Voice search

Travel Assistant
Translation

Navigation Assistant
Store finder

On-Device ("At the Edge") AI – Why?

Low latency requirements

- Natural dialogue in speech assistants – requires less than 200msec latency
- Real-time decision making in automotive, robots, AR/VR, etc. needs low latency

Lack of good connectivity

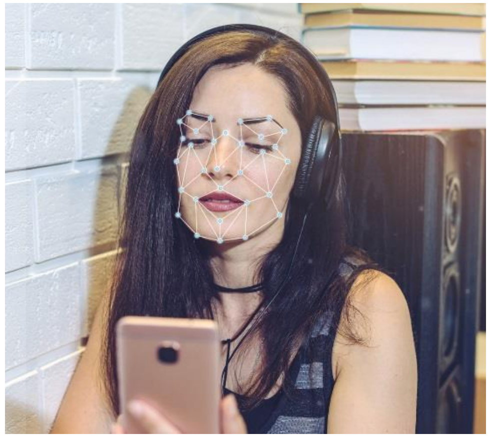
- Smart city cameras difficult to connect to existing network
- Inspection drones for wind turbines and power lines operate in rural areas

Privacy

- Smart home video cameras and smart assistants—consumers desire privacy

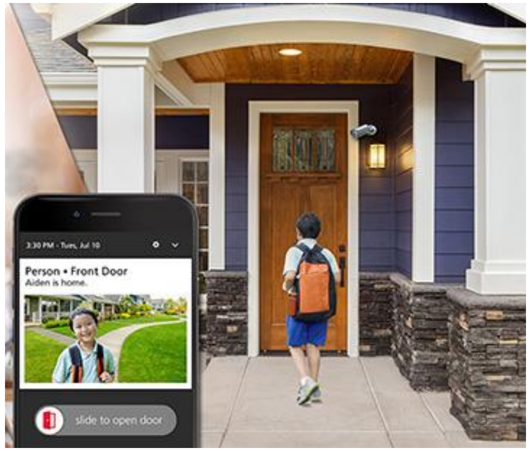
Target Markets for On-Device AI Inferencing

IoT
< 0.5TMAC/s

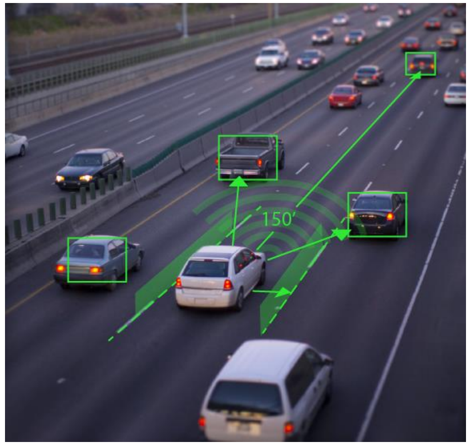


Mobile
0.5 - 2TMAC/s

AR/VR
1 - 4TMAC/s



Smart Surveillance
2 - 10TMAC/s



Autonomous Vehicles
10s - 100s TMAC/s

On-Device AI Processing Needs Are Increasing

Mobile

- On-device AI experiences like face detection and people recognition at video capture rates

AR/VR headsets

- On-device AI for object detection, people recognition, gesture recognition, and eye tracking

Surveillance cameras

- On-device AI for family or stranger recognition and anomaly detection

Drones and robots

- On-device AI to recognize subjects, objects, obstacles, emotions, etc.

Automotive

- On-device AI to recognize pedestrians, cars, signs, lanes, driver alertness, etc. for ADAS and AV

CNN Algorithm Development Trends

Increasing Computational Requirements

(~16X in <4 years)

- AlexNet (2012)
- Inception (2015)
- ResNet (2015)

NETWORK	MACS/IMAGE
ALEXNET	724,406,816
INCEPTION V3	5,713,232,480
RESNET-101	7,570,194,432
RESNET-152	11,282,415,616

Network Architectures Changing Regularly

- AlexNet (bigger convolution); Inception V3 and ResNet (smaller convolution)
- Linear network vs. branch

New Applications and Markets

- Automotive, server, home (voice-activated digital assistants), mobile, surveillance

Low Power

How do you pick an inference hardware platform today (2018) for a product shipping in 2020-2022+? How do you achieve low-power efficiency yet be flexible?

AI = Big Problem size, Requires Big (SoC) Solutions

What is realistic to deploy ?

- Kirin 980 from HiSilicon
  - “The Kirin 980 integrates 6.9 billion transistors in an area of less than 1 square centimeter”
  - “Kirin 980 can quickly adapt to AI scenes such as face recognition, object recognition, object detection, image segmentation and intelligent translation with the power of a dual-core NPU achieving 4500 images per minute”

    https://consumer.huawei.com/en/campaign/kirin980/
- A12 Bionic from Apple
  - “The company says it's the industry's first 7-nanometer chip and contains 6.9 billion transistors”

    https://www.engadget.com/2018/09/12/apple-a12-bionic-7-nanometer-chip/
  - “The Neural Engine is incredibly fast, able to perform five trillion operations per second. It’s incredibly efficient, which enables it to do all kinds of new things in real time”

    https://www.apple.com/uk/iphone-xr/a12-bionic/

“When all you have is a hammer, everything looks like a nail”

Abraham Maslow

Edge Processing of AI Requires NEW Architectures

- AI is fundamentally a Software Problem **TRUE**
- Software can easily run on Standard Processors **TRUE**
- Software performance can be scaled up on GPUs **TRUE**
- AI Software can run at the required performance/power levels on Standard processor/GPU platforms **FALSE**
- It’s Obvious what the next generation of AI platforms should look like **FALSE**

What Characteristics Does a “NN DSP” need ?

- High Computation Throughput at low power
  - Different number representations – from (typically) 8b fixed point to Floating point
  - “Lots of MACs” ! Arranged in a flexible way to be able to handle the different kernels
- Supporting ISA
  - “Just Enough” → usually means arithmetics, logical, some shifts and vector shuffling,
  - Don’t need all the “bells and whistles” associated with traditional Computer Vision
  - Keep it lean and focussed with “enough” flexibility to handle all visible and predicted (!) requirements
- High Data Throughput from L1 memory
  - Need to be able to feed data to the computation units in a sustained manner
  - There may be some tricks (e.g. “Compression” → Taking advantage of zeroes)
- Connection to Bulk Memory – DMA
  - The Network Model will not fit in L1 memory (pretty much a given)
  - Access and timing of memory fetch from off-chip memory to L1 usually handled by DMA

Quantisation / Tiling

- Fundamentals for embedded processing of NN

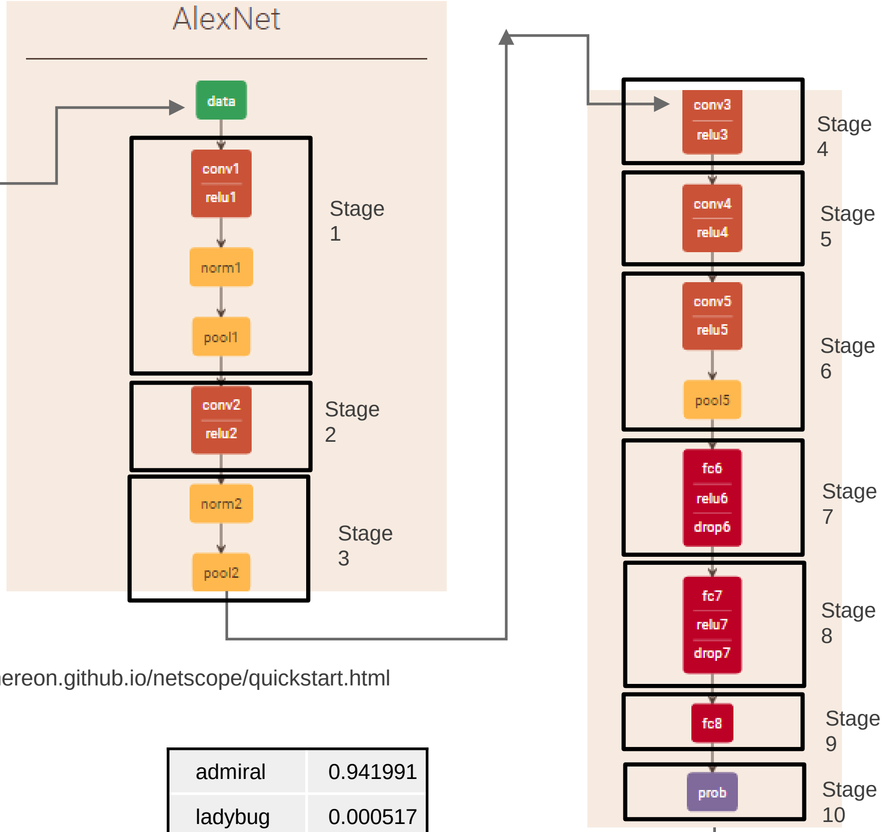
- In reference trained model (e.g. Caffe, Tensorflow etc)
  - – Number representation usually floating point, single precision
  - – Entire image available/visible to the model

- In an embedded solution
  - – Number representation usually much lower – “Quantised” → 16b, 8b, even lower
  - – Inferencing system can only process parts of the image at a time “Tiles”

- Embedded systems must handle long latencies to bulk memory (off chip)
- Embedded systems must intelligently quantise to avoid degradation of NN performance (accuracy)

Vision P6 DSP Running Alexnet Convolutional Neural Network

227x227 RGB image



Alexnet visualization from http://ethereon.github.io/netscope/quickstart.html

Vision P6 Detection Result

admiral	0.941991
ladybug	0.000517
monarch	0.000287
tench	0.000057
goldfish	0.000057

Alexnet:

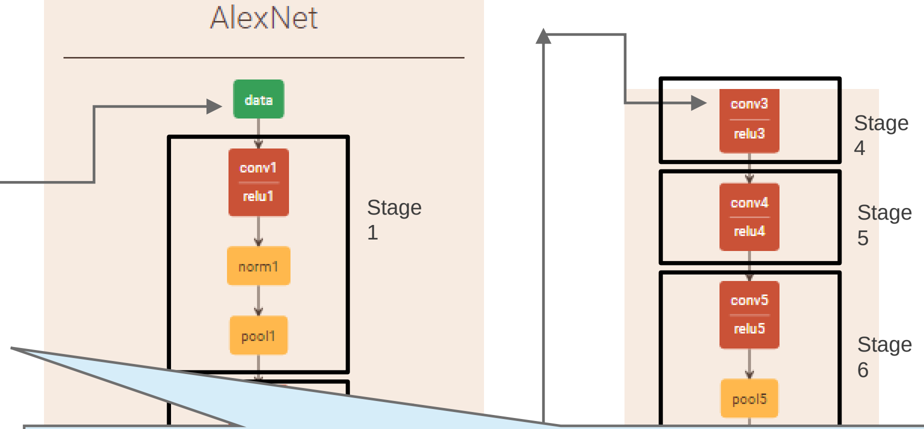
Winner of the ImageNet (ILSVRC) 2012 Contest
Trained for 1000 different classes (images)
Most often quoted benchmark for CNN
Classifier CNN Example
5 Conv & 3 FC layers
Input image: 227x227 image patch (ROI)

Cadence Alexnet Implementation

Based on Caffe 32b floating point Alexnet model
Use 8 bit coefficients, 8 bit data computations
Pure C P6 implementation,
No library dependencies such as BLAS, NumPy, etc

Vision P6 DSP Running Alexnet Convolutional Neural Network

227x227 RGB image



Alexnet:

Winner of the ImageNet (ILSVRC) 2012 Contest
Trained for 1000 different classes (images)
Most often quoted benchmark for CNN
Classifier CNN Example
5 Conv & 3 FC layers
Input image: 227x227 image patch (ROI)

t Implementation

2b floating point Alexnet model
ents, 8 bit data computations
mentation,
dencies such as BLAS, NumPy, etc

Alexnet visualization from http://e

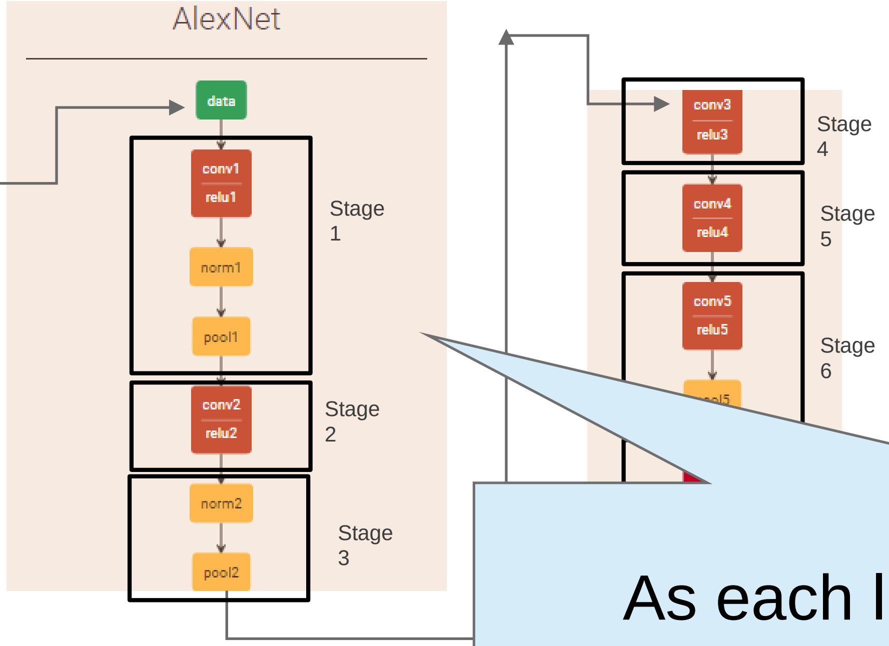
Visi

Patches ("Tiles") fetched from memory as 227x227
The Network does not "decide" which patches to fetch from the main image – decision made by other systems

goldfish	0.000057

Vision P6 DSP Running Alexnet Convolutional Neural Network

227x227 RGB image



Alexnet visualization from http://ethereon.github.io/netscope/quickstart.html

Alexnet:

Winner of the ImageNet (ILSVRC) 2012 Contest
Trained for 1000 different classes (images)
Most often quoted benchmark for CNN
Classifier CNN Example
5 Conv & 3 FC layers
Input image: 227x227 image patch (ROI)

etc

As each layer is processed, the related coefficents (“weights”) must be fetched from memory (usually off chip) by the DMA systems, in time for processing by the MAC units

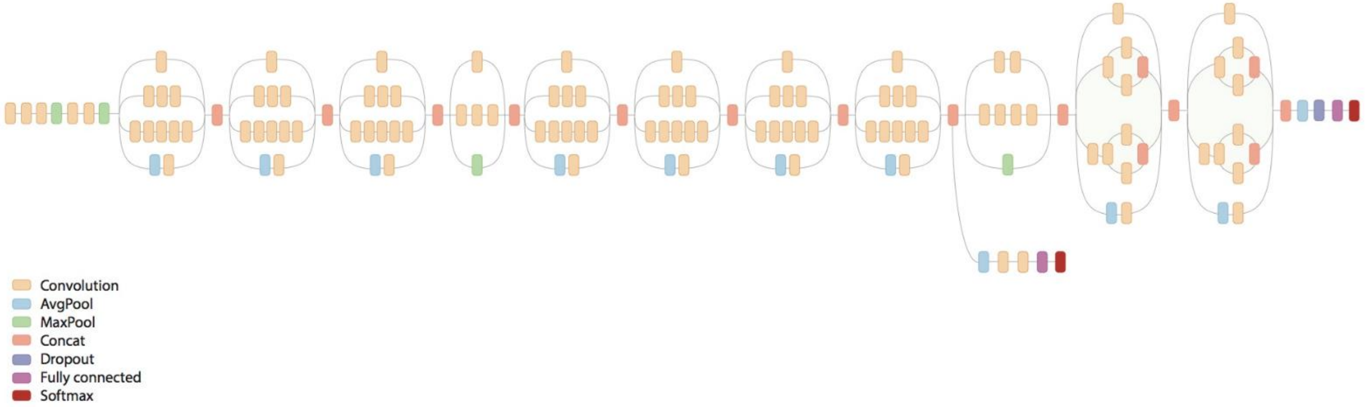
Vision P6 Detection Result

admiral	0.941991
ladybug	0.000517
monarch	0.000287
tench	0.000057
goldfish	0.000057

Inception V3 Accuracy on Vision P6

(8bit Data and 8 bit Weights)

Inception V3 Details	
Input ROI	299x299x3
Number of Layers	110
Compute Requirement	5.78 GMAC
Bandwidth Requirement (8 bit Weights)	19.4 MB



Vision P6 Accuracy (Loss <1%)
Using 8bit Quantized Data & Weights

Accuracy*	Float	8bit Fixed Point
Top-1 Accuracy	74.00%	73.29%
Top-5 Accuracy	91.62%	91.18%

*Accuracy tested over 50K images in ImageNet Val set

Marginal loss of Network accuracy due to quantisation

Examples of (Vision + NN) Specialised DSPs from Cadence

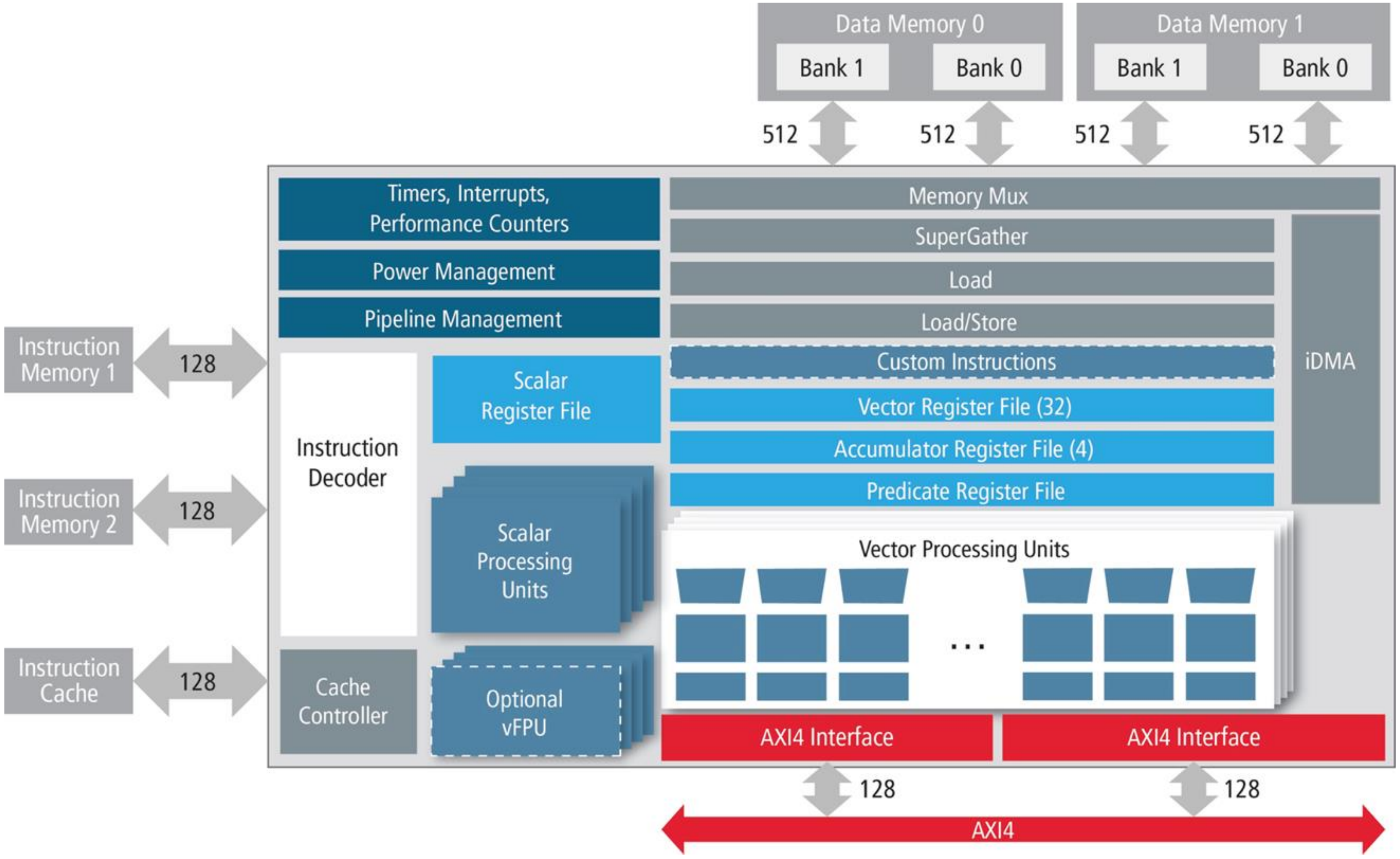
- Sometimes you want “Computer Vision + NN Capable” in 1 core

- Sometimes you want only “NN Capable”, but more efficient.

- Usually (our philosophy) you need lots of flexibility, but the opportunity to differentiate

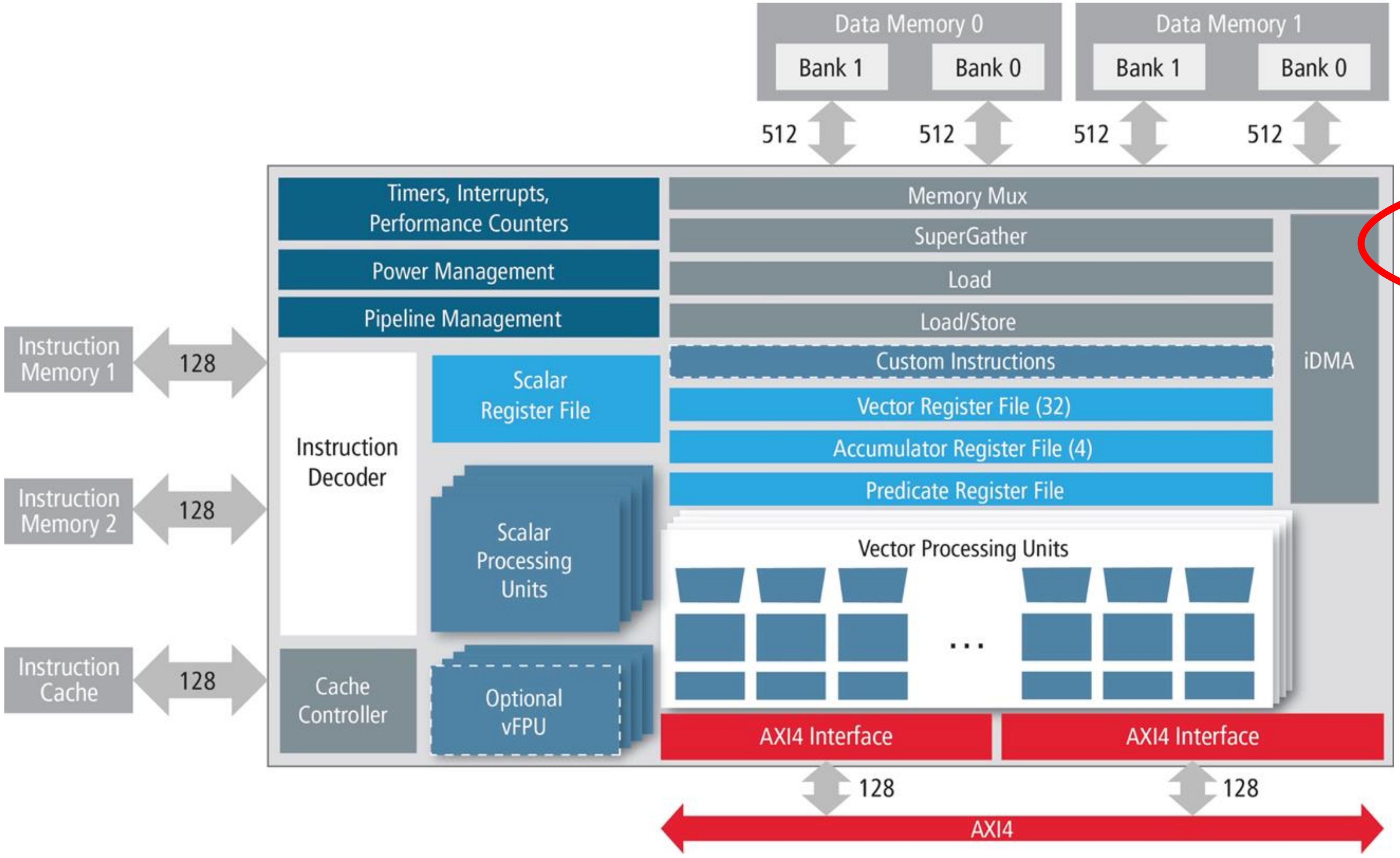
- This means
  - Program in ‘C’ using convenient vector types
  - Good debug, modelling, integration, libraries, supporting SW, compilers
  - Ability to add ISA features (custom instructions) if desired

Vision P6 Architecture



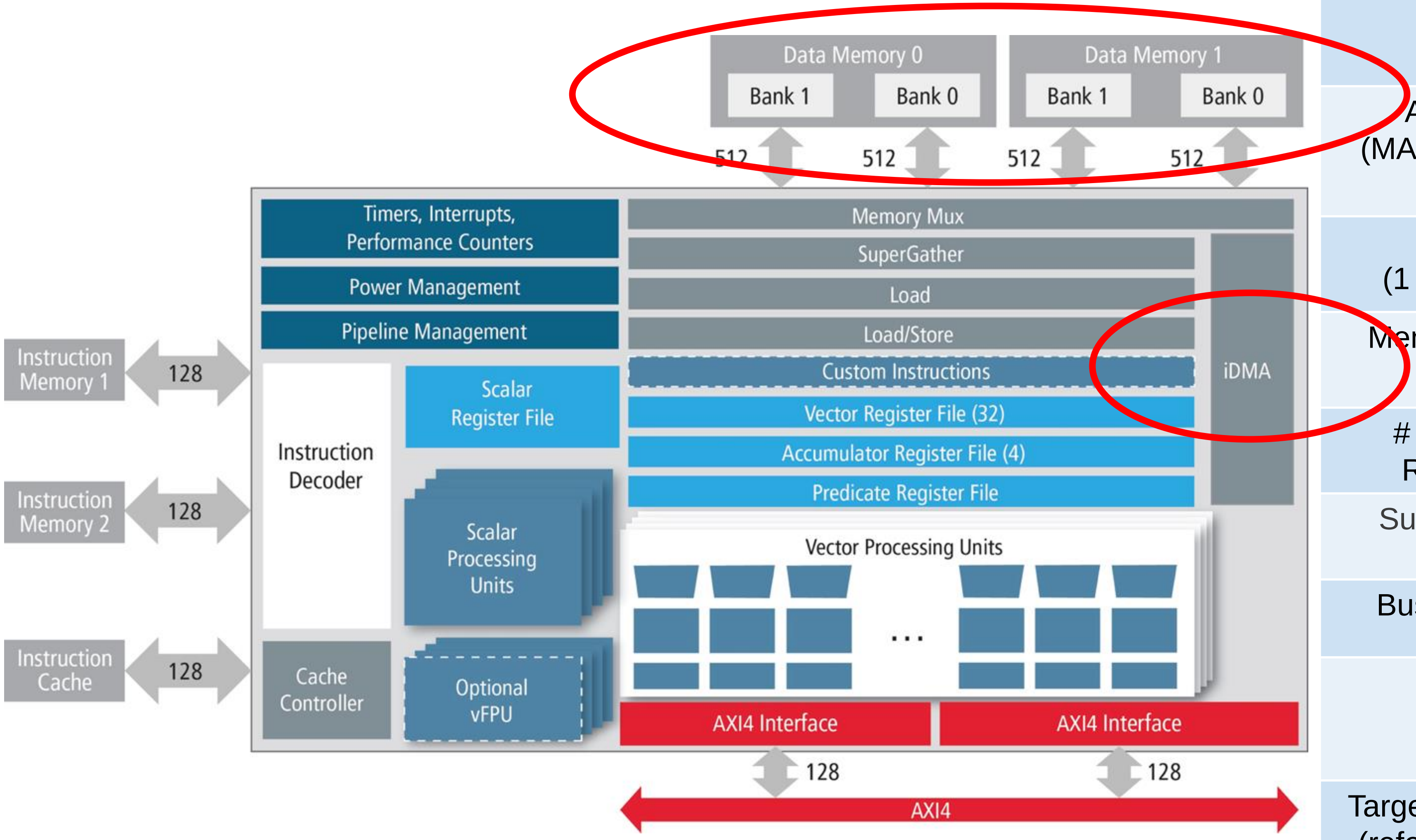
VLIW & SIMD	5 slots 64way 8-bit 32way 16-bit 16way 32-bit
ALU Ops (MAX 4 out of 5 slots)	256 8-bit 128 16-bit 64 32-bit
MAC (1 of 5 slots)	Vision P6: 256 (8x8), 128 (8x16), 64 (16x16)
Memory Width	1024-bits 2 vector load/store units
# of Vector Registers	32
SuperGather	32 non-contiguous locations read/ written per instruction
Bus Interface	AXi4
iDMA	no alignment restrictions, local memory to local memory transfers, …
Target Frequency (reference core)	800Mhz @28nm 1.1 GHz @16nm (with overdrive)
Optional	Vector Floating Point, ECC

Vision P6 Architecture



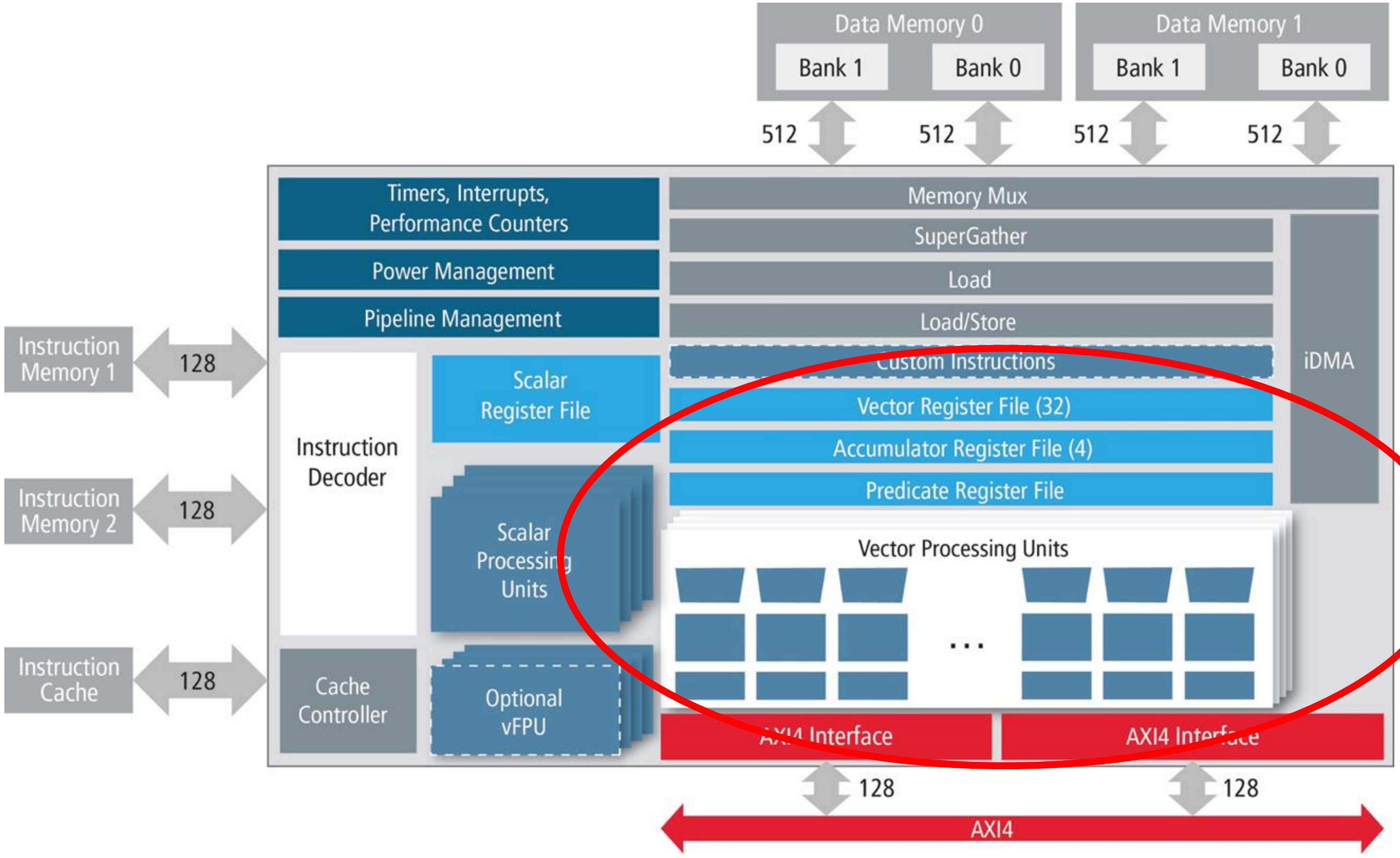
VLIW & SIMD	5 slots 64way 8-bit 32way 16-bit 16way 32-bit
ALU Ops (MAX 4 out of 5 slots)	256 8-bit 128 16-bit 64 32-bit
MAC (1 of 5 slots)	Vision P6: 256 (8x8), 128 (8x16), 64 (16x16)
Memory Width	1024-bits 2 vector load/store units
# of Vector Registers	32
SuperGather	32 non-contiguous locations read/ written per instruction
Bus Interface	AXi4
iDMA	no alignment restrictions, local memory to local memory transfers, …
Target Frequency (reference core)	800Mhz @28nm 1.1 GHz @16nm (with overdrive)
Optional	Vector Floating Point, ECC

Vision P6 Architecture



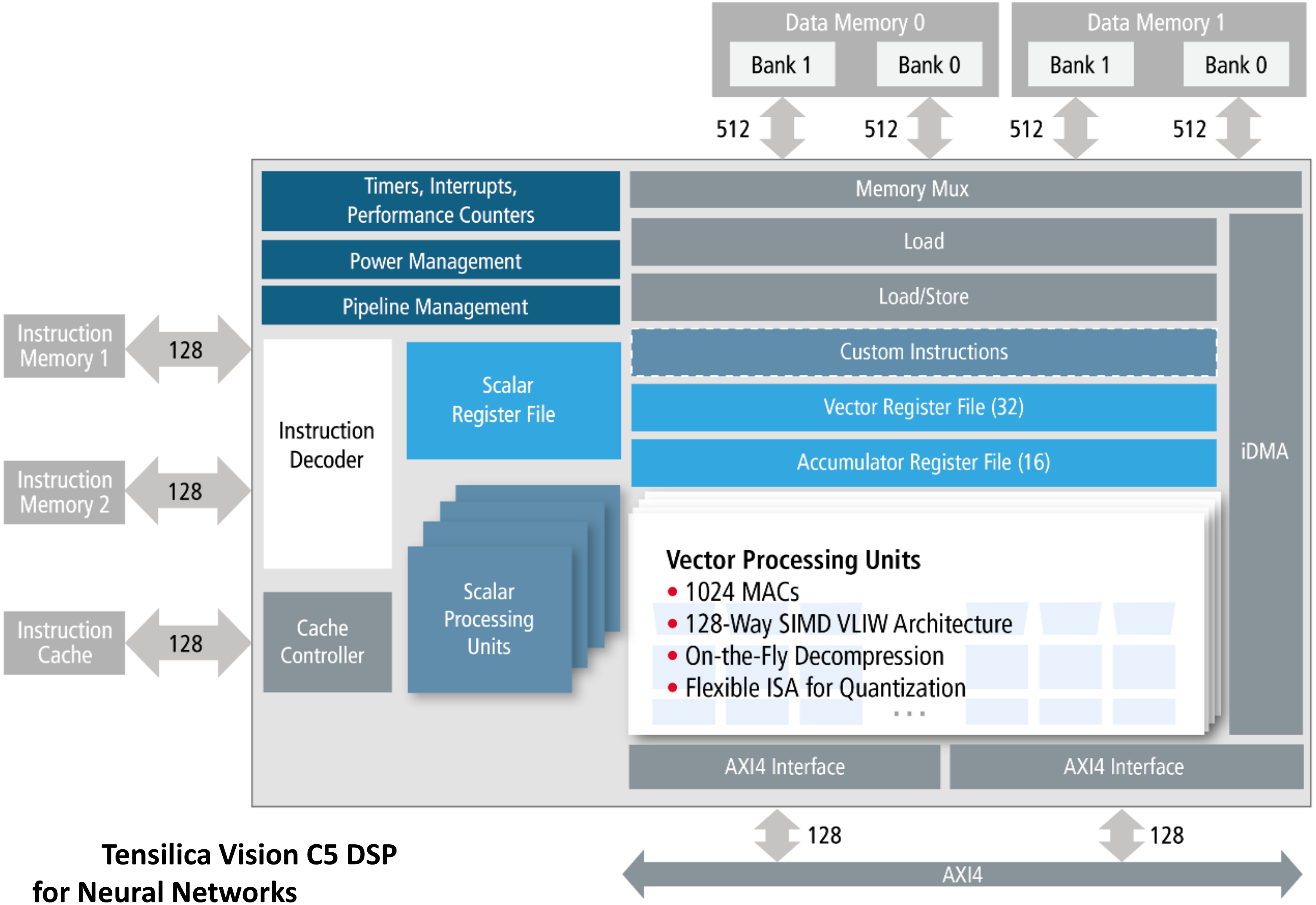
VLIW & SIMD	5 slots 64way 8-bit 32way 16-bit 16way 32-bit
ALU Ops (MAX 4 out of 5 slots)	256 8-bit 128 16-bit 64 32-bit
MAC (1 of 5 slots)	Vision P6: 256 (8x8), 128 (8x16), 64 (16x16)
Memory Width	1024-bits 2 vector load/store units
# of Vector Registers	32
SuperGather	32 non-contiguous locations read/ written per instruction
Bus Interface	AXi4
iDMA	no alignment restrictions, local memory to local memory transfers, …
Target Frequency (reference core)	800Mhz @28nm 1.1 GHz @16nm (with overdrive)
Optional	Vector Floating Point, ECC

Vision P6 Architecture



VLIW & SIMD	5 slots 64way 8-bit 32way 16-bit 16way 32-bit
ALU Ops (MAX 4 out of 5 slots)	256 8-bit 128 16-bit 64 32-bit
MAC (1 of 5 slots)	Vision P6: 256 (8x8), 128 (8x16), 64 (16x16)
Memory Width	1024-bits 2 vector load/store units
# of Vector Registers	32
SuperGather	32 non-contiguous locations read/ written per instruction
Bus Interface	AXi4
iDMA	no alignment restrictions, local memory to local memory transfers, …
Target Frequency (reference core)	800Mhz @28nm 1.1 GHz @16nm (with overdrive)
Optional	Vector Floating Point, ECC

Vision C5 Architecture



Tensilica Vision C5 DSP for Neural Networks

- Complete Stand Alone DSP to run all NN Layers
- General Purpose, Programmable and Flexible
- Scalable to multi-TMAC design

- Fixed point DSP with 8-bit and 16-bit support

- 1024 8x8 MACs per cycle
- 512 16x16 MACs per cycle

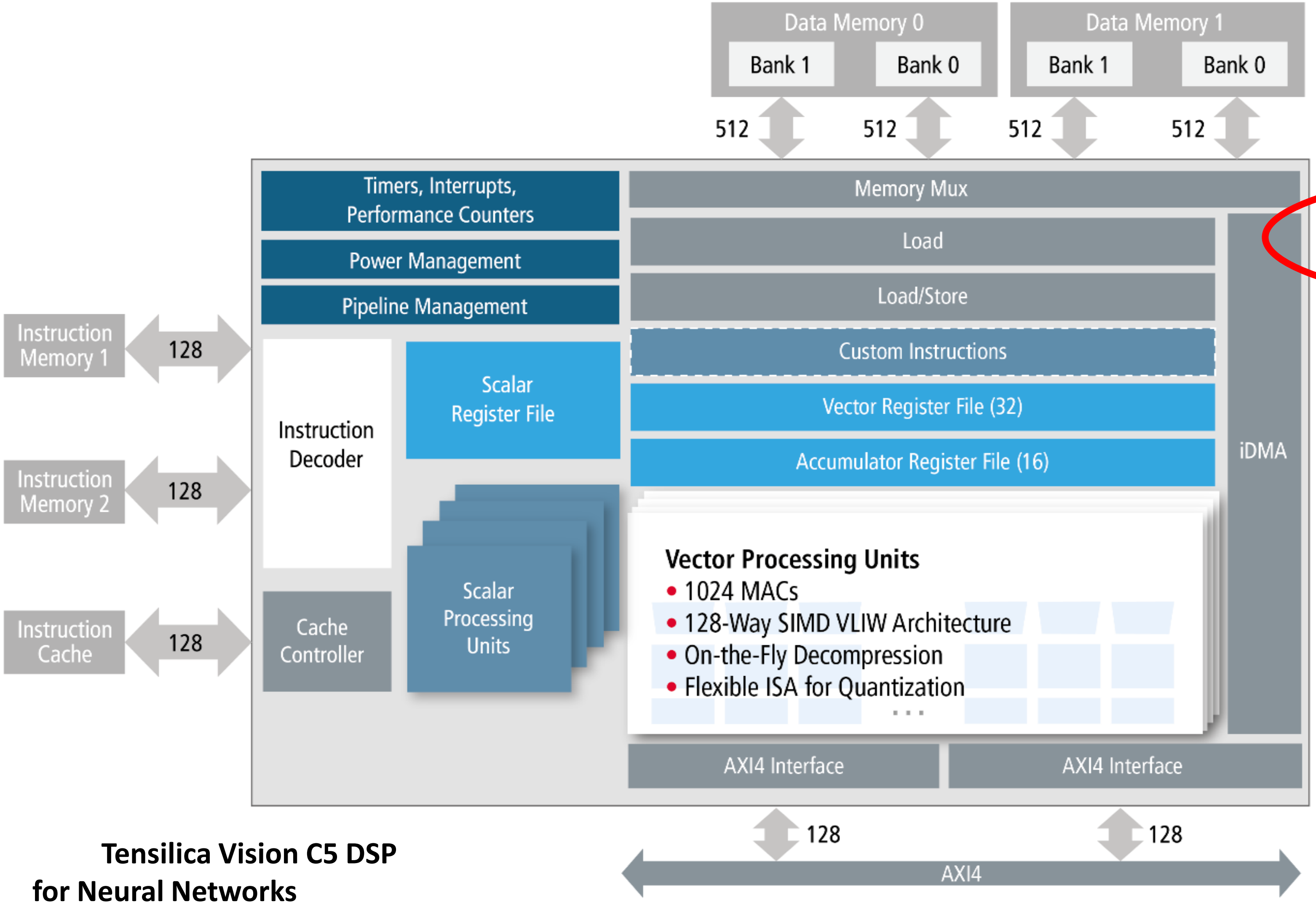
- 512-bit vector register file
- 1024-bit register (pairing 2 512-bit registers)
  - 128 way 8bit SIMD
  - 64 way 16bit SIMD

- 3 or 4 slot VLIW
  - Load/store/pack pairs with ALU/MAC ops

- 1024 bit Memory Width
  - Dual LD/ST Support including aligned data

- On the Fly Decompression Support
- Special addressing mode for efficient access of 3-D data
- Richer set of convolution multipliers (signed and unsigned)
- Extensive data rearrangement and selection

Vision C5 Architecture



Vector Processing Units
- 1024 MACs
- 128-Way SIMD VLIW Architecture
- On-the-Fly Decompression
- Flexible ISA for Quantization
- . . .

Tensilica Vision C5 DSP for Neural Networks

- Complete Stand Alone DSP to run all NN Layers
- General Purpose, Programmable and Flexible
- Scalable to multi-TMAC design

- Fixed point DSP with 8-bit and 16-bit support

- 1024 8x8 MACs per cycle
- 512 16x16 MACs per cycle

- 512-bit vector register file
- 1024-bit register (pairing 2 512-bit registers)
  - 128 way 8bit SIMD
  - 64 way 16bit SIMD

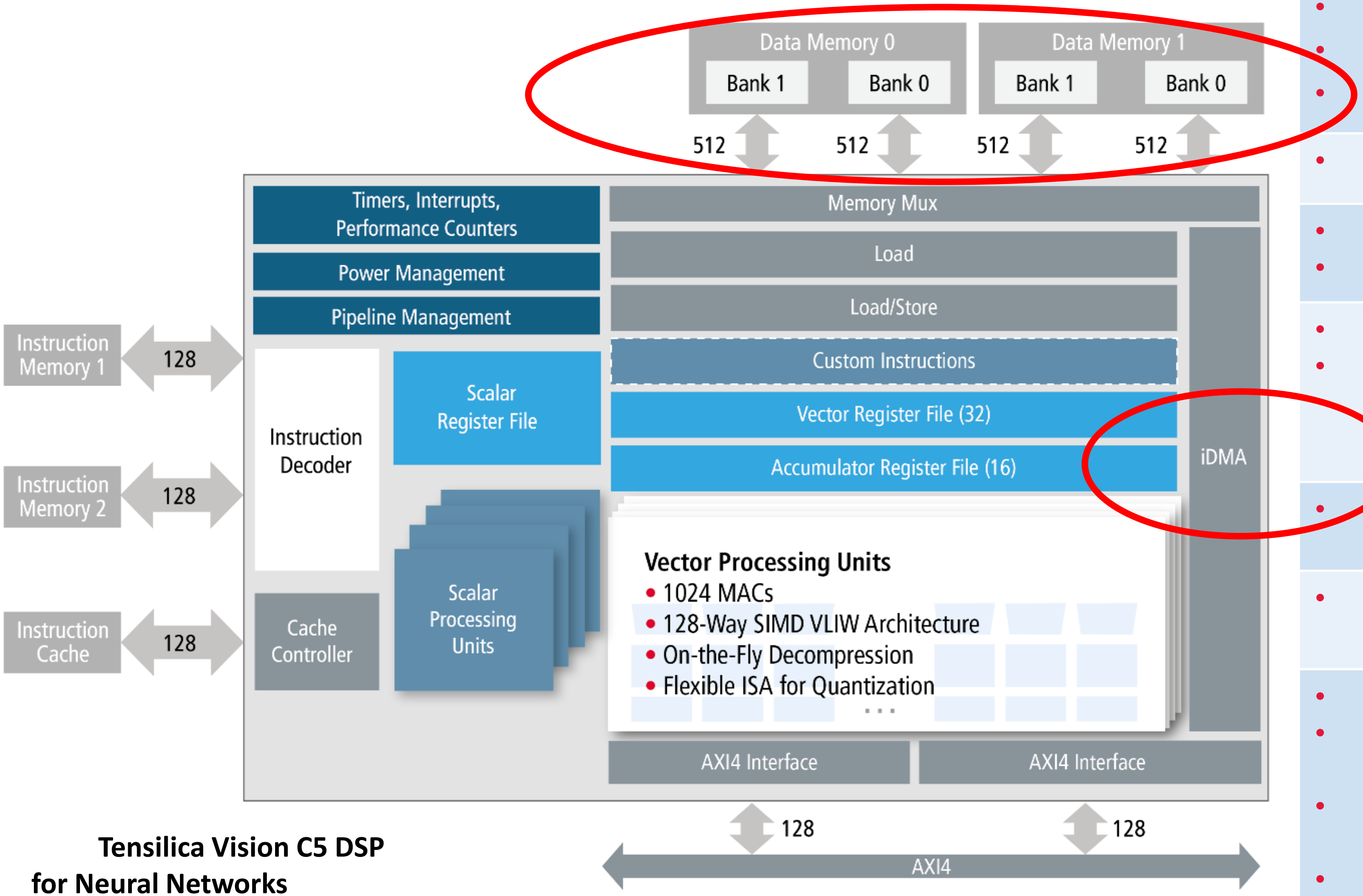
- 3 or 4 slot VLIW
  - Load/store/pack pairs with ALU/MAC ops

- 1024 bit Memory Width
  - Dual LD/ST Support including aligned data

- On the Fly Decompression Support
- Special addressing mode for efficient access of 3-D data
- Richer set of convolution multipliers (signed and unsigned)
- Extensive data rearrangement and selection

cadence®

Vision C5 Architecture



Tensilica Vision C5 DSP for Neural Networks

- Complete Stand Alone DSP to run all NN Layers
- General Purpose, Programmable and Flexible
- Scalable to multi-TMAC design

- Fixed point DSP with 8-bit and 16-bit support

- 1024 8x8 MACs per cycle
- 512 16x16 MACs per cycle

- 512-bit vector register file
- 1024-bit register (pairing 2 512-bit registers)
  - 128 way 8bit SIMD
  - 64 way 16bit SIMD

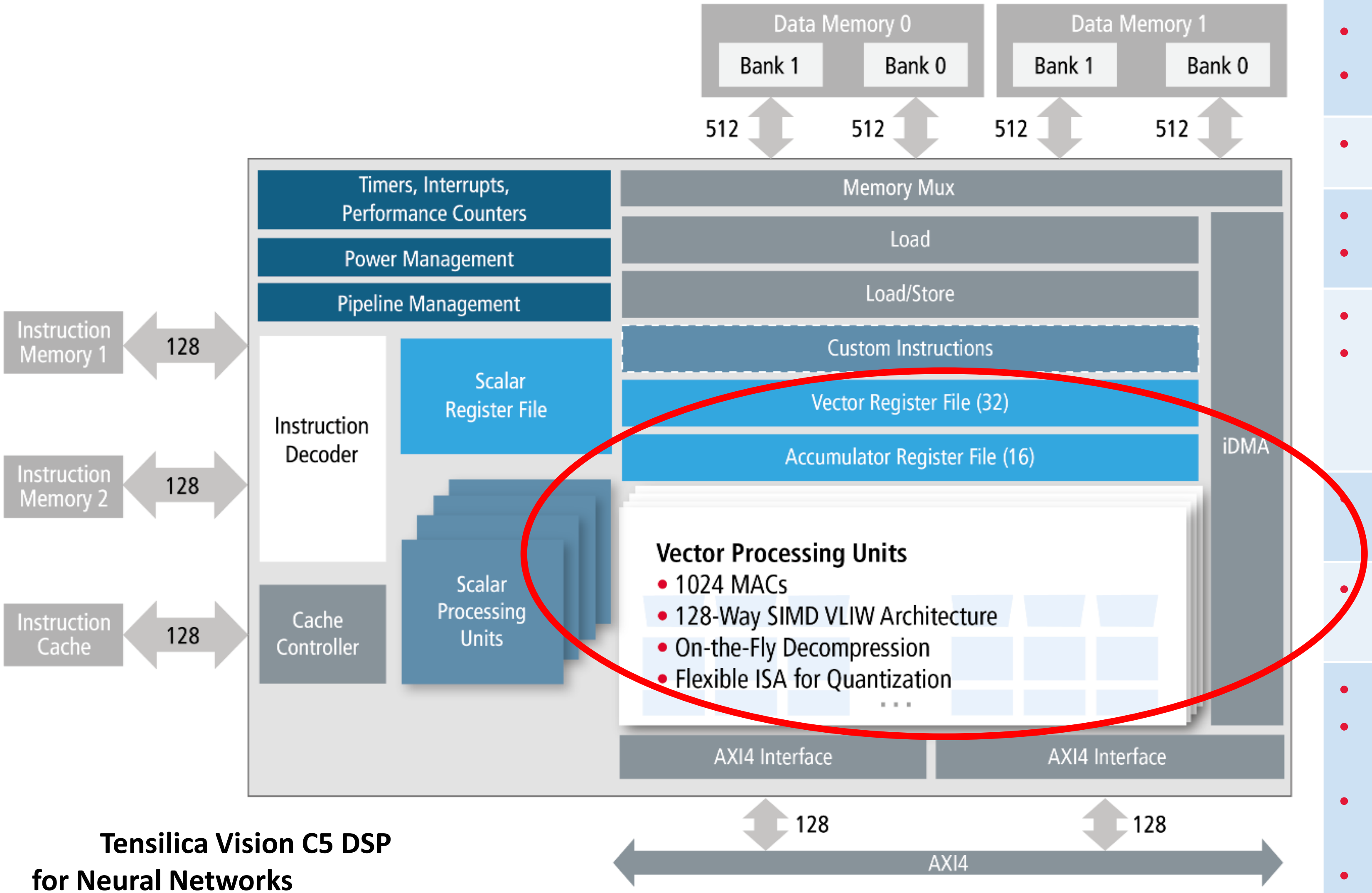
- 3 or 4 slot VLIW
  - Load/store/pack pairs with ALU/MAC ops

- 1024 bit Memory Width
  - Dual LD/ST Support including aligned data

- On the Fly Decompression Support
- Special addressing mode for efficient access of 3-D data
- Richer set of convolution multipliers (signed and unsigned)
- Extensive data rearrangement and selection

cadence®

Vision C5 Architecture



Vector Processing Units

- 1024 MACs
- 128-Way SIMD VLIW Architecture
- On-the-Fly Decompression
- Flexible ISA for Quantization
- . . .

Tensilica Vision C5 DSP for Neural Networks

- Complete Stand Alone DSP to run all NN Layers
- General Purpose, Programmable and Flexible
- Scalable to multi-TMAC design

- Fixed point DSP with 8-bit and 16-bit support

- 1024 8x8 MACs per cycle
- 512 16x16 MACs per cycle

- 512-bit vector register file
- 1024-bit register (pairing 2 512-bit registers)
  - 128 way 8bit SIMD
  - 64 way 16bit SIMD

- 3 or 4 slot VLIW
  - Load/store/pack pairs with ALU/MAC ops

- 1024 bit Memory Width
  - Dual LD/ST Support including aligned data

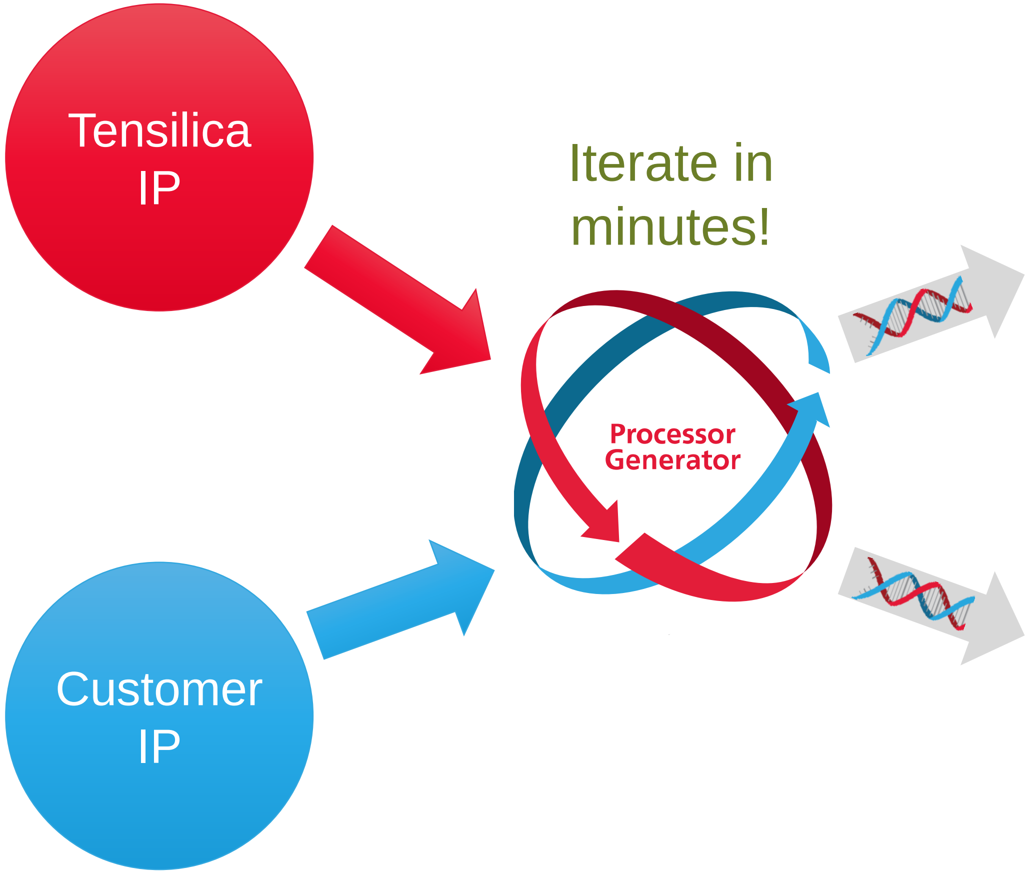
- On the Fly Decompression Support
- Special addressing mode for efficient access of 3-D data
- Richer set of convolution multipliers (signed and unsigned)
- Extensive data rearrangement and selection

Automated Tool, ISS, Model, RTL, and EDA Script Generation...

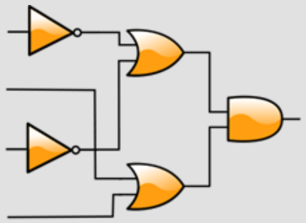
Base Processor
Dozens of Templates for many common applications

Pre-Verified Options
Off-the-shelf DSPs, interfaces, peripherals, debug, etc.

Optional Customization
Create your own instructions, data types, registers, interfaces



Complete Hardware Design



Pre-verified
Synthesizable RTL
EDA Scripts
Test suite…

Advanced Software Tools

IDE
C/C++ Compiler
Debugger
ISS Simulator
SystemC Models
DSP code libraries

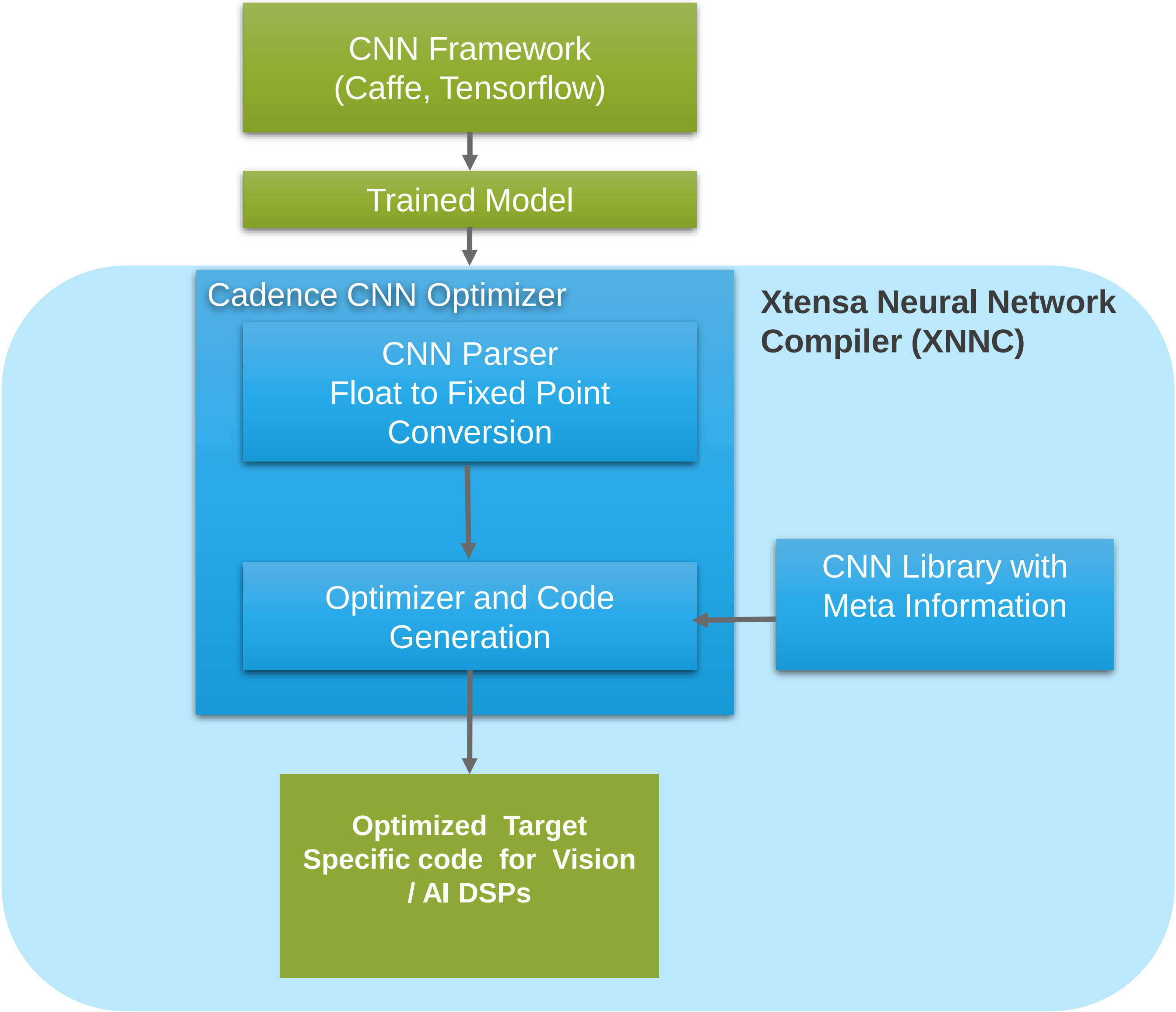
Hardware is Not Enough

- Hardware must be
  - Programmed
  - Debugged
  - Integrated
  - Modelled

- Embedded Software must
  - Integrate at a high level with other software
  - Be self-sufficient – usually libraries are required
  - Integrate computations with data fetch
  - Be easily maintained – no assembly programming thank you very much!

- Vision based example …

Xtensa Neural Network Compiler (XNNC)


CNN Framework
(Caffe, Tensorflow)
Trained Model
Cadence CNN Optimizer
Xtensa Neural Network Compiler (XNNC)
CNN Parser
Float to Fixed Point Conversion
Optimizer and Code Generation
CNN Library with Meta Information
Optimized Target Specific code for Vision / AI DSPs

- Connects to existing industry CNN frameworks by using their Trained Model descriptions
- and **auto-generates Trained Model optimized code for Cadence CNN DSPs**

- Three Major components to XNNC
- CNN Parser: Float to Fixed Point conversion
- CNN Code Generation and Optimization
- CNN Library for Vision DSP

- First CNN Framework support: Caffe, followed by Tensorflow
- For Vision DSPs

Summary – What does it take to run AI Inferencing at the Edge ?

- The right architecture ..
- … with the right High Level Software integration
- … and the right scalability and flexibility
- … right now

Sep 19th 2018: *Cadence Launches New Tensilica DNA 100 Processor IP Delivering Industry-Leading Performance and Power Efficiency for On-Device AI Applications*
http://www.cadence.com/go/dna100

cādence®