Linking Application Description with Efficient SIMD Code Generation for Low-Precision Signed-Integer GEMM

Günther Schindler[1], Manfred Mücke[2], Holger Fröning[1]

[1] Heidelberg University, Germany
[2] Materials Center Leoben, Austria

10th Workshop on UnConventional High Performance Computing
August 29, 2017

DEEPCHIP PROJECT

Collaboration with Franz Pernkopf (TU Graz) and Manfred Mücke (MCL)

Research goal: enable the inference of Deep Neural Networks (DNNs) on mobile devices and embedded systems with limited power consumption and computational resources

- Processor selection: ARM CPUs (Xilinx FPGAs, NVIDIA's Tegra and ARM's Mali GPUs)
- Methods: reduced Precision (sparsity, and asynchronity)

MOTIVATION

Reduced-precision representation in machine learning is promising

- Reduce computational complexity and memory footprint

Machine learning tool stacks and BLAS libraries usually don't support reduced-precision representation (except TensorFlow: 8 bit)

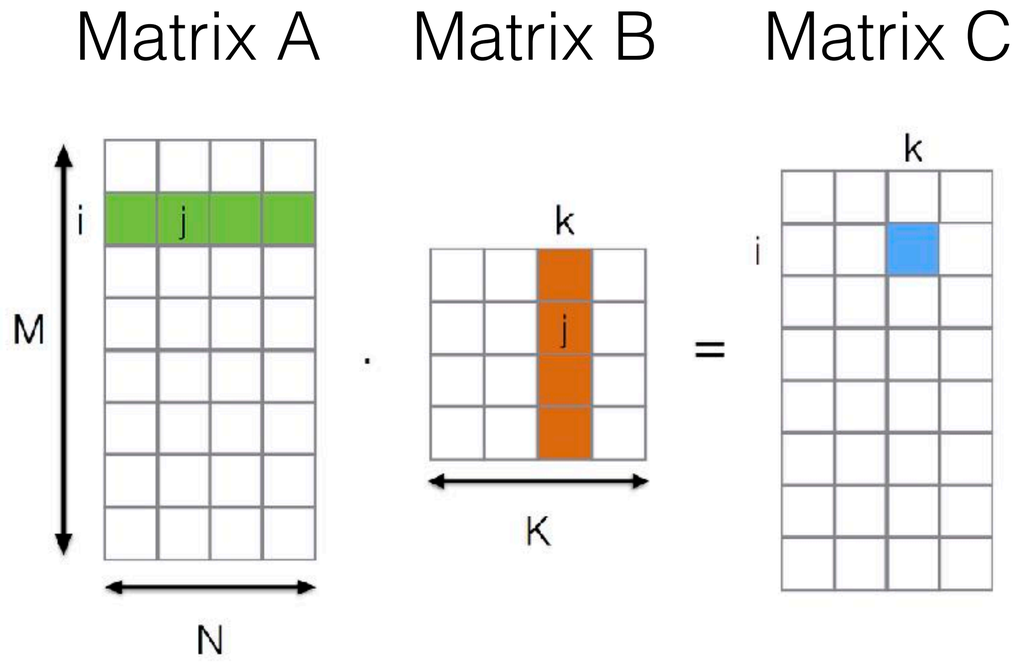
- Approach: extend tool stacks by leveraging existing BLAS libraries for certain processors

Today: extending Theano and Eigen for reduced-precision signed-integer General Matrix Multiply (GEMM) for NEON capable ARM processors

GEMM (1)

```
Input: matrices A and B
Output: matrix C

For i from 1 to M
 For j from 1 to K
  sum = 0
  For k from 1 to N
   sum = sum + Aik * Bkj
  Cij = sum
```



Multiply-Accumulate (MAC) Operation

GEMM (2)

GEMM is the key operation of neural networks

GEMM is difficult to implement efficiently

- Caching, parallelization, vectorization, loop unrolling, register usage, etc.

Approach

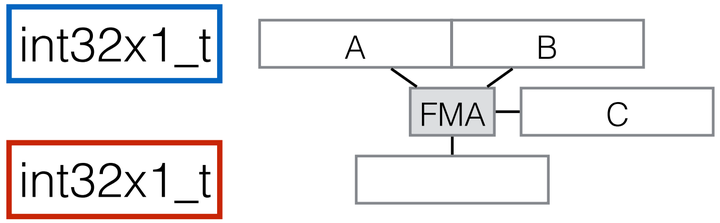
- Use existing high-level transformations of the highly optimized Eigen library
- Optimize MAC operation for targeted precision and processor

Design issues:

- Types: 16-/8-/4-/2-/1-bit signed integer
- Bit-width doubling when multiplying - avoids integer overflows
- Reduction when accumulating - SIMD vectorization
- Bitpacking of several low-precision values into a 32-bit vector - allows easy integration into Theano

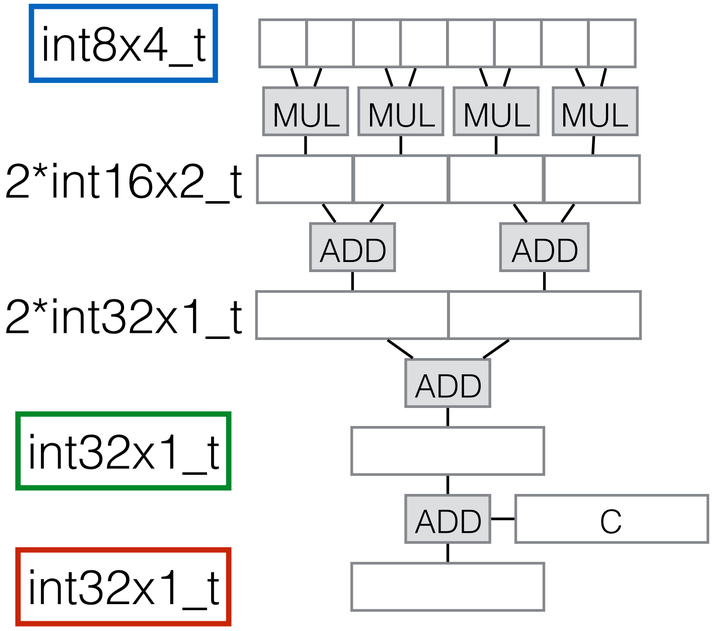
SIMD-MAC IMPLEMENTATION

32-Bit MAC



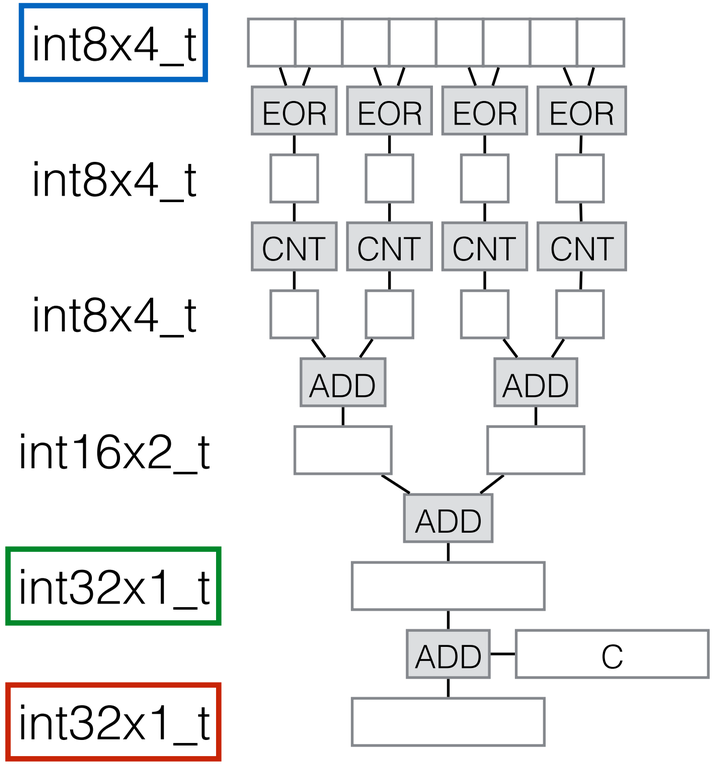
6 Cycles

8-Bit MAC



36 Cycles

1-Bit MAC



15 Cycles

Input Representation

Intermediate Representation

Output Representation

SIMD-MAC works for 32-bit and 128-bit SIMD width

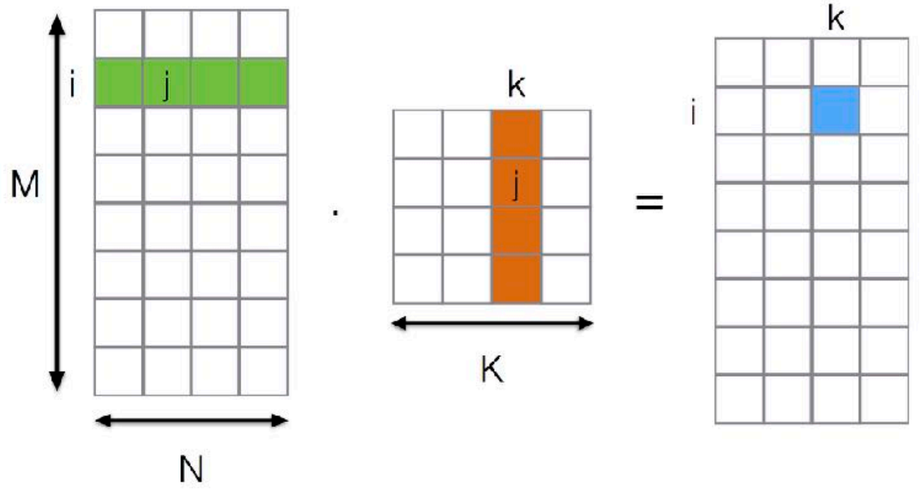
REDUCED INTERMEDIATE REPRESENTATION (1)

Reducing the Intermediate Representation (IR) eases reduction within MAC operation

- But reduced IR has lower integer range
- Output value of 1-bit and 2-bit multiplications is constrained to +1, (0), and -1

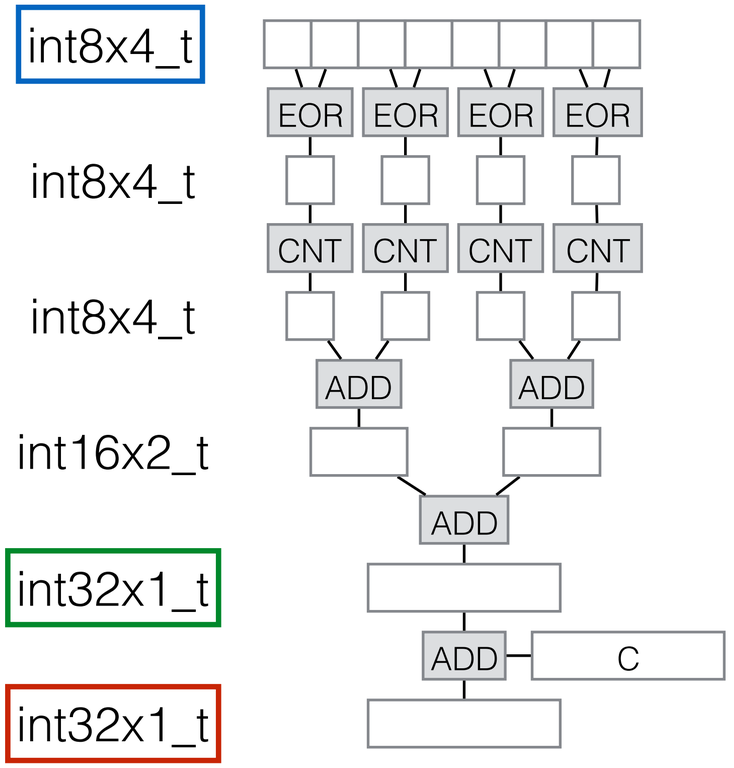
N accumulations per output value

- 32-bit IR for N>4096
- 16-bit IR for 32<N<=4096
- 8-bit IR for N<=32



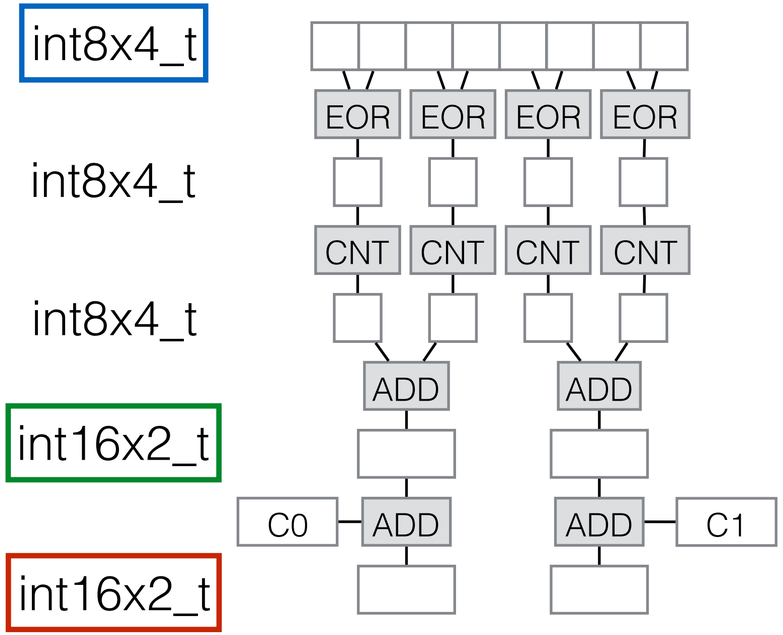
REDUCED INTERMEDIATE REPRESENTATION (2)

32-Bit IR



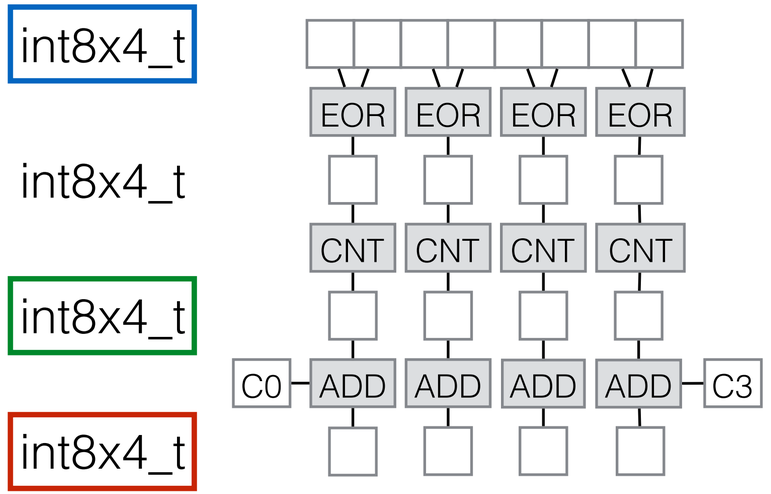
15 Cycles

16-Bit IR



12 Cycles

8-Bit IR



9 Cycles

Input Representation

Intermediate Representation

Output Representation

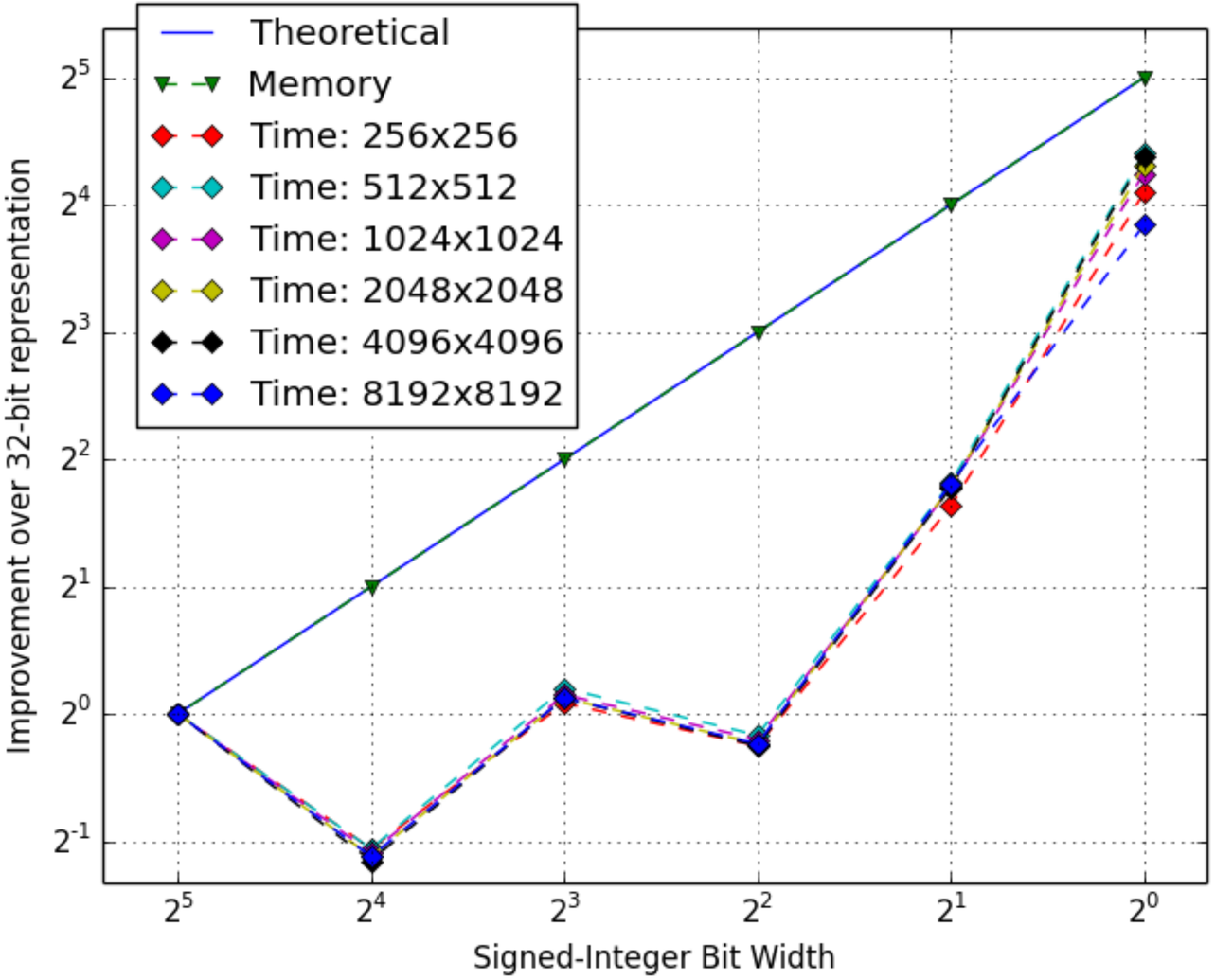
RESULTS - LOW-PRECISION GEMM ON ARM CORTEX A-15

Doubling the bit width when multiplying is very expensive (16/8/4 bit)

Reduction increases computational complexity

Handling of 4 bit and 2 bit values is difficult

1 bit values are easy to interpret and easy to compute



CONCLUSION

Reduced precision is the most promising approach to improve NNs on embedded devices

ARM’s ISA is well suited for reduced-precision operations

Bit-width doubling is crucial to the overall performance

Based on our results: binary and ternary computations are clearly superior

OUTLOOK

Reduced precision in neural networks

- Tradeoff between time/energy and accuracy
- Fewer bits usually need more neurons or layers to obtain comparable accuracy

Example: 32-bit vs. 1-bit Multilayer Perceptron on MNIST

	32 Bit	1 Bit
Classification Ratio	0.3 kFPS	5.7 kFPS
Accuracy	99.09 %	98.83 %

MNIST is simple. What about other datasets (ImageNet, Cifar10)?

How big is the potential of different quantization strategies?