RC1 - NEON-v7 programming

This course explains how to use ARMv7 NEON SIMD instructions to boost multimedia algorithms

Objectives

- This course has been designed for programmers wanting to run multimedia algorithms on NEON Single Instruction Multiple Data execute units on ARMv7 processors.
- Each instruction family is detailed, first at assembly level, and then at C level using macros developed present in arm_neon.h file.
- Several tricky usage of processing instructions are provided.
- Vector and vector element load / store instructions are studied and guidelines for organizing data in memory are provided to minimize the number of memory accesses.
- The underlying cache operation as well as preload mechanisms (instruction and hardware prefetch) are detailed to explain how a processing can be pipelined .
- The course shows how DSP typical algorithms such as FIR and FFT can be vectorized and then optimized to be executed on NEON unit.

Labs are compiled with GCC and run on a Linux Cortex-A9 board or a simulator
A more detailed course description is available on request at training@ac6-training.com

Prerequisites

- Knowledge of ARMv7 instruction sets.

Environnement du cours

- Cours théorique
  - Support de cours au format PDF (en anglais) et une version imprimée lors des sessions en présentiel
  - Cours dispensé via le système de visioconférence Teams (si à distance)
  - Le formateur répond aux questions des stagiaires en direct pendant la formation et fournit une assistance technique et pédagogique
- Activités pratiques
  - Les activités pratiques représentent de 40% à 50% de la durée du cours
  - Elles permettent de valider ou compléter les connaissances acquises pendant le cours théorique.
  - Exemples de code, exercices et solutions
  - Pour les formations à distance:
    - Un PC Linux en ligne par stagiaire pour les activités pratiques, avec tous les logiciels nécessaires préinstallés.
    - Le formateur a accès aux PC en ligne des stagiaires pour l'assistance technique et pédagogique
    - Certains travaux pratiques peuvent être réalisés entre les sessions et sont vérifiés par le formateur lors de la session suivante.
  - Pour les formations en présentiel::
    - Un PC (Linux ou Windows) pour les activités pratiques avec, si approprié, une carte cible embarquée.
    - Un PC par binôme de stagiaires s'il y a plus de 6 stagiaires.
  - Pour les formations sur site:
    - Un manuel d'installation est fourni pour permettre de préinstaller les logiciels nécessaires.
    - Le formateur vient avec les cartes cible nécessaires (et les remporte à la fin de la formation).
- Une machine virtuelle préconfigurée téléchargeable pour refaire les activités pratiques après le cours
- Au début de chaque session (demi-journée en présentiel) une période est réservée à une interaction avec les stagiaires pour s'assurer que le cours répond à leurs attentes et l'adapter si nécessaire

Audience visée

- Tout ingénieur ou technicien en systèmes embarqués possédant les prérequis ci-dessus.

Plan du cours

Day 1

Introduction to NEON/VFPv3

- Clarifying the resources shared by NEON and VFP
- Register bank, Q registers, D registers
- Data types
- Vector vs scalar
- Related system registers
- Alignment issues
- Enabling NEON/VFP
- Differences between NEONv7 and NEONv8

NEON instruction syntax

- Instructions producing wider / narrower results
- Instructions modifiers
- Selecting the shape
- Selecting the operand / result type
- Syntax flexibility
- Declaring initialized vectors in C language
- Using unions with vectors and arrays of vectors to simplify the debug
- Casting vectors

LOAD and STORE instructions

- Addressing modes
- Vector load / store
- Vector load / store multiple
- Element and structure load / store instructions
- Multiple single elements
- Single element to 1 lane
- Single elements to all lanes
- Optimizing the ordering of data in memory to take benefit of 2-, 3- and 4- element structures
  Exercise : Example: managing audio samples
- Processor acceleration mechanisms: store merging buffers
  Exercise : Using load with de-interleaving instructions to store all right lane samples into a vector and left lane samples into another vector

Day 2

Data transfer instructions

- Move
- Swap
- Table lookup
- Vector transpose
- Vector zip / unzip

- Data transfer between NEON and integer unit
  Exercise : Clarifying narrow and long instructions, building a vector from bytes selected from a pair of vectors

Logical and bitfield instructions

- Logical AND, Bit Clear, OR, XOR
- Operations with immediate values
- Bitwise insert instructions, avoiding branches
- Count Leading zeros, ones, signs
- Normalizing floating point numbers when VFP is not implemented
- Scalar duplicate
- Extract
- Shift with possible rounding and saturation
- Bitfield reverse
  Exercise : Transposing a matrix, shifting a large bitmap using vector instructions

Data processing Instructions

- Arithmetic instructions
- Add, modulo vs saturated arithmetic
- Halving / Doubling the result
- Rounding
- Subtract
- Multiply
- Multiply accumulate / Multiply subtract
- Absolute value
- Min / Max
  Exercise : Implementing a complex multiply accumulate with NEON
- Conversion instructions
- Converting Floating Point numbers into Fixed point numbers
- Converting Fixed point numbers into Floating point numbers
  Exercise : Converting fixed-point elements into single precision floating point values and adding the resulting elements
- Advanced arithmetic instructions
- Reciprocal estimate, reciprocal square root estimate, Newton-raphson algorithm
- Pairwise instructions
- Element comparison

NEON coding examples

- FIR filter
- Converting the scalar algorithm into a vector algorithm
- Finding the NEON instructions to encode the vector algorithm
- Optimizing the code
- Using the performance monitor to tune the algorithm
- FFT (DFT)
- Converting the scalar algorithm into a vector algorithm, understanding how circle properties can be used to process 4 angles concurrently
- Finding the NEON instructions to encode the vector algorithm
- Optimizing the code
- Using the performance monitor to tune the algorithm