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 formation@ac6-formation.com

Prerequisites

- Knowledge of ARMv7 instruction sets.

Course Environment

- Theoretical course
  - PDF course material (in English) supplemented by a printed version for face-to-face courses.
  - Online courses are dispensed using the Teams video-conferencing system.
  - The trainer answers trainees' questions during the training and provide technical and pedagogical assistance.
- Practical activities
  - Practical activities represent from 40% to 50% of course duration.
  - Code examples, exercises and solutions
  - For remote trainings:
    - One Online Linux PC per trainee for the practical activities.
    - The trainer has access to trainees' Online PCs for technical and pedagogical assistance.
    - QEMU Emulated board or physical board connected to the online PC (depending on the course).
    - Some Labs may be completed between sessions and are checked by the trainer on the next session.
  - For face-to-face trainings:
    - One PC (Linux ou Windows) for the practical activities with, if appropriate, a target board.
    - One PC for two trainees when there are more than 6 trainees.
  - For onsite trainings:
    - An installation and test manual is provided to allow preinstallation of the needed software.
    - The trainer come with target boards if needed during the practical activities (and bring them back at the end of the course).
- Downloadable preconfigured virtual machine for post-course practical activities
- At the start of each session the trainer will interact with the trainees to ensure the course fits their expectations and correct if needed

Target Audience

- Any embedded systems engineer or technician with the above prerequisites.

Evaluation modalities

- The prerequisites indicated above are assessed before the training by the technical supervision of the traineein his company, or by the trainee himself in the exceptional case of an individual trainee.
- Trainee progress is assessed in two different ways, depending on the course:
  - For courses lending themselves to practical exercises, the results of the exercises are checked by the trainer while, if necessary, helping trainees to carry them out by providing additional details.
  - Quizzes are offered at the end of sections that do not include practical exercises to verifythat the trainees have assimilated the points presented
- At the end of the training, each trainee receives a certificate attesting that they have successfully completed the course.
  - In the event of a problem, discovered during the course, due to a lack of prerequisites by the trainee a different or additional training is offered to them, generally to reinforce their prerequisites,in agreement with their company manager if applicable.

Plan

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

Renseignements pratiques

Inquiry : 2 days