RC1 - Programmation NEON-v7

Ce cours explique comment utiliser les instructions SIMD NEON de l'ARMv7 pour accélérer les algorithmes multimédia

Objectifs

- Ce cours a été conçu pour les programmeurs souhaitant exécuter des algorithmes multimédia sur les unités d'exécution NEON Single Instruction Multiple Data des processeurs ARMv7.
- Chaque famille d'instructions est détaillée, d'abord au niveau assembleur, puis au niveau C à l'aide des macros présentes dans le fichier arm_neon.h.
- Plusieurs usages subtils des instructions de traitement sont présentés.
- Les instructions de load / store de vecteurs et d'éléments de vecteurs sont étudiées, et des recommandations d'organisation des données en mémoire sont fournies afin de minimiser le nombre d'accès mémoire.
- Le fonctionnement sous-jacent du cache ainsi que les mécanismes de préchargement (instruction et prefetch matériel) sont détaillés pour expliquer comment un traitement peut être pipeliné .
- Le cours montre comment des algorithmes typiques du traitement du signal tels que le FIR et la FFT peuvent être vectorisés puis optimisés pour s'exécuter sur l'unité NEON.

Prérequis

- Connaissance des jeux d'instructions ARMv7.

Environnement du cours

- Cours théorique
  - Support PDF (en anglais), imprimé en présentiel ; à distance via Teams.
  - Assistance du formateur tout au long de la formation.
- Chaque session débute par un point avec les stagiaires.

Audience visée

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

Plan du cours

Premier jour

Introduction à NEON/VFPv3

- Clarification des ressources partagées par NEON et VFP
- Banc de registres, registres Q, registres D
- Types de données
- Vecteur et scalaire
- Registres système associés
- Problèmes d'alignement
- Activation de NEON/VFP
- Différences entre NEONv7 et NEONv8

Syntaxe des instructions NEON

- Instructions produisant des résultats plus larges / plus étroits
- Modificateurs d'instructions
- Choix de la forme
- Choix du type de l'opérande / du résultat
- Souplesse de la syntaxe
- Déclaration de vecteurs initialisés en langage C
- Utilisation d'unions avec des vecteurs et des tableaux de vecteurs pour simplifier le débogage
- Conversion de type des vecteurs

Instructions LOAD et STORE

- Modes d'adressage
- Load / store de vecteurs
- Load / store de vecteurs multiples
- Instructions de load / store d'éléments et de structures
  - Éléments simples multiples
  - Un élément vers 1 lane
  - Un élément vers toutes les lanes
- Optimisation de l'ordre des données en mémoire pour tirer parti des structures à 2, 3 et 4 éléments
- Mécanismes d'accélération du processeur : store merging buffers

Deuxième jour

Instructions de transfert de données

- Move
- Swap
- Table lookup
- Transposition de vecteurs
- Zip / unzip de vecteurs
- Transfert de données entre NEON et l'unité entière

Instructions logiques et de champs de bits

- AND logique, Bit Clear, OR, XOR
- Opérations avec valeurs immédiates
- Instructions d'insertion bit à bit, pour éviter les branchements
- Comptage des zéros, des uns, des signes en tête
- Normalisation des nombres flottants lorsque VFP n'est pas implémenté
- Duplication de scalaire
- Extract
- Décalage avec arrondi et saturation possibles
- Inversion de champ de bits

Instructions de traitement de données

- Instructions arithmétiques
  - Add, arithmétique modulo et arithmétique saturée
  - Division / multiplication par deux du résultat
  - Arrondi
  - Subtract
  - Multiply
  - Multiply accumulate / Multiply subtract
  - Valeur absolue

    - Min / Max
- Instructions de conversion
    - Conversion de nombres flottants en nombres à virgule fixe
    - Conversion de nombres à virgule fixe en nombres flottants
- Instructions arithmétiques avancées
    - Estimation de l'inverse, estimation de l'inverse de la racine carrée, algorithme de Newton-Raphson
    - Instructions pairwise
- Comparaison d'éléments

Exemples de code NEON

- Filtre FIR
    - Conversion de l'algorithme scalaire en algorithme vectoriel
    - Recherche des instructions NEON permettant de coder l'algorithme vectoriel
    - Optimisation du code
    - Utilisation du performance monitor pour affiner l'algorithme
- FFT (DFT)
    - Conversion de l'algorithme scalaire en algorithme vectoriel, en comprenant comment les propriétés du cercle permettent de traiter 4 angles simultanément
    - Recherche des instructions NEON permettant de coder l'algorithme vectoriel
    - Optimisation du code
    - Utilisation du performance monitor pour affiner l'algorithme