A System-on-Chip-based CABAC Encoder Design for H.264/AVC Video Coding Standard

Thinh M. Le - SMIEEE, X.H. Tian, B.L. Ho, X. Jiang, S.P. Krishnamurthy, Yong Lian - SMIEEE
Department of Electrical and Computer Engineering
National University of Singapore

BACKGROUND

H.264/AVC has been introduced to improve coding efficiency and network friendliness. Adopted for the main and higher profiles of H.264/AVC, the Context-based Adaptive Binary Arithmetic Coding (CABAC) entropy coding tool, combined with the Rate-Distortion Optimization (RDO) tool significantly improve video coding efficiency. However, high computational complexity and power consumption prevent SW implementation of CABAC in real-time and high compression video applications. The project aims at designing an efficient, SoC-based CABAC encoder which consists of three major functional blocks: binarizer, context modeler, and binary arithmetic coder (BAC) (Fig. 1b).

DESIGN OBJECTIVES

- Achieve high throughput SoC-based CABAC coding
  - Minimize the computation on the host processor
  - Constant throughput in different coding configurations
- Support different coding tools fully, including RDO
- Utilize the context memory efficiently
- Reduce RDO context operation delay

DESIGN STRATEGY

- Conduct performance-complexity analyses at the beginning of the SoC design flow (Fig. 1a)
- Reduce cycle length by pipelining context access & BAC
- Enable parallel processing of functional blocks
- Reduce context RAM access frequency by:
  - Context line access and buffering of context models
  - Reduce context RAM size and RDO operation delay by:
    - Utilizing 3 small RAM blocks to backup intermediate context state during P8×8 RDO coding
    - Pipelined operations of context state backup and restoration

CABAC ENCODER ARCHITECTURE

- Block 1: SE binarization & context model selection (Fig. 2)
  - 3 FWFT FIFOs buffer intermediate results
- Block 2: context model access, binary arithmetic coding, and output bit packing
  - 3-stage pipeline, throughput of 1 bin/cycle
- Context manager: context model initialization, context state backup & restoration in P8×8 RDO coding
- WISHBONE System bus interface
- Enhance design portability & reusability
- Total memory: 27.1Kb ROM, 7.37Kb context RAM, 0.78Kb FIFO buffers, 11.0Kb SE RAM for coded MBs

REFERENCES


Contact: Dr. Le M. Thinh (elelmt@nus.edu.sg)