

An AVS Video Decoder Design and Implementation Based On Parallel Algorithm

Chunchun Sui, Ning Wang, Jing Chen, Xixin Cao

(Chunchun Sui, Jing Chen, Xixin Cao, School of Software and Microelectronic, Peking University, Beijing, China)

(Ning Wang, Institute of Microelectronics, Xidian University, Xi'an, China)

cxx@ss.pku.edu.cn

Abstract—With the development of communication, digital video compression technology turns into one of the most flourishing realm. In this paper the author introduces an AVS decoder design based on a multimedia chip-platform. In order to obtain the optimal performance, the structure of decoder adopts parallel algorithm with the centre processor and the coprocessor. The performance of the decoder which is about ten times faster than that of the software mode rm52j_r1, meets the requirement of real-time broadcasting (30 frames per second). Some analyses about the optimization are also included in this paper.

Keywords—Video decoder, Parallelism processing, CC1100, AVS;

I. INTRODUCTION

Nowadays, with the rapid development of digital communication, more and more digital media come into our lives. At the same time, a large number of multi-medium programs with high quality are in demand. If these original data information are transported directly, the current network and storage device cannot afford such a large number of data bulk[1]. Therefore we adopt video encoding and decoding criteria with high compression ratio to compress data information, and this is the key point to solve the issue.

AVS Working Group (Advanced Audio Video Coding Standard of China was released in China, which combined with the advantages of H.264 and MPEG2) is responsible for constituting an audio and video compression standard with its own intellectual property rights for the domestic industrial sector.

C2 platform based on the CC1100 chip is a video development platform which is designed by C2 Microsystem Company CC1100 single-chip integrated RISC processor, 256-bit vector processors, ME (Motion Estimation) Engine programmable coprocessor and ENE (Entropy Engine) programmable coprocessor. However, most of the available codes aim for the realization of general processor or DSP. So there are no ready-made examples based on the platform of CC1100 implement AVS decoder for reference. It is also a big challenge for the developers.

This paper finishes the assignment that involved implementation and verification and optimization of the AVS

decoder in the CC1100 platform, and the performance of the decoder meet the requirement of real-time broadcasting (30 frames per second).

II. AN ANALYSIS OF THE STRUCTURE OF PLATFORM CC1100

As shown in Figure 1, CC1100 is composed of one MPU and three DSP processors. MPU is RSIC CPU, which used the similar MIPS structure. In the core of structure of CC1100, there are three DSP aiming at optimization of usual calculating in three kinds of encoding and decoding arithmetic respectively. The 3 kinds of calculating were as follows:

1. Motion Estimation, ME [2]: is a calculator which spent most time on video processing. In ME engine, proprietary instructions were used for calculating accelerating in block matching and movement vector searching.

2. Entropy Encode/Decode: is a calculator which was very frequently used in image processing. ENE engine provided relevant hardware support for this calculating.

3. Vector Processing: image is a kind of two dimension data. When it comes to an 8×8 image block, a row or line pixels can be processed as just a vector. Vector calculator was very frequently used. In addition, this part can also be used in audio data processing.

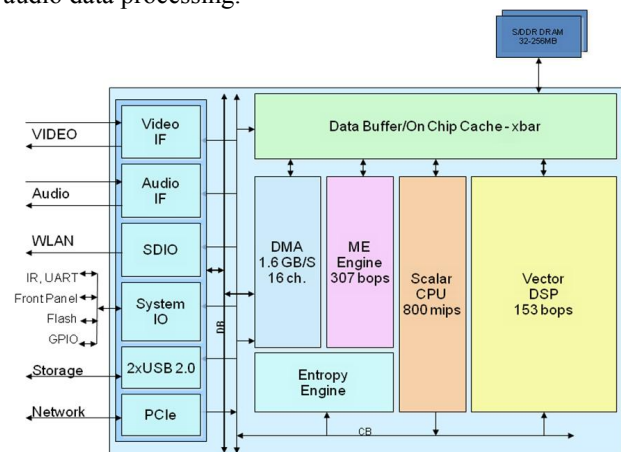


Figure 1. Framework of CC1100

CC1100 includes not only its own arithmetic and logical operations instructions such as addition, minus, multiplication, and division, but also hardware design about the largest

calculation out of the above-mentioned three types of operations, together with appropriate instruction library. With the support of dedicated hardware, software instructions provided by calling the library package, efficiency of encoding and decoding can be greatly enhanced.

III. AVS DECODER DESIGN

A. The reference framework of the decoder

The decoder was based on the official reference decoder rm52j_r1 for the design and optimization. The overall structure of the rm52j_r1 as shown in Figure 2:

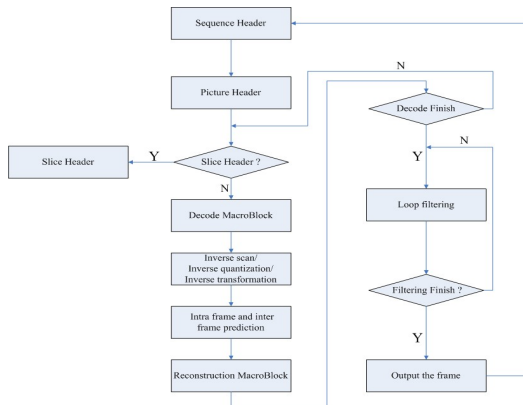


Figure 2. Flow chart in RM52j_r1

As presented above, it can be divided into several parts:

- syntactic element analysis of the framework, which mainly included SequenceHeader and PictureHeader
- The macro block SliceHeader data will be identified and managed at the start of the macro block row.
- macroblock reconstruct, consisting of macro block syntactic element, residuals data and revert macro block data, as well as the following steps: Inverse scan, Inverse quantization, Inverse transformation, Intra frame and inter frame prediction, Loop filtering.

Related to the previous code standard, the complexity of the AVS Audio Video Coding Standard has substantially reduced. But it is impossible for the software to meet the needs of real-time encoding and decoding. Therefore, it is necessary to use the hardware acceleration in the chip.

B. Overall process design for the decoder

In order to use hardware acceleration module for stream parsing ENE, we need to consider the synergy of hardware and software rendering time. Three programs can be chosen from:

- Take Macroblock as the smallest unit of hardware and software into consideration. This program modifies process of the original code to the smallest degree, but because of the little amount of macroblocks and macroblock quantity, task preparation, switching costs will far exceed their time dealing with the macroblock.
- Take Slice as the smallest unit of hardware and software into consideration. The head of Slice is disposed in

the main program, and the disposed data hands to the ENE, and then continue to analyse macroblock. Due to different length of Slice, task processing time of Slice may vary greatly, which greatly influences the effect of parallel algorithm.

3. Take Picture (image frame or field) as the smallest unit of hardware and software into consideration. MPU processes Sequence Header and Picture Header, the remaining part is processed by ENE, including the Slice Header and the macro block data.

According to analysis, the decoder uses the third consideration, which requires the image level of the bit stream parsing code and image data processing code to be separated.

The new decoder process is shown in Figure 3. The part inside the dashed finishes hardware acceleration.

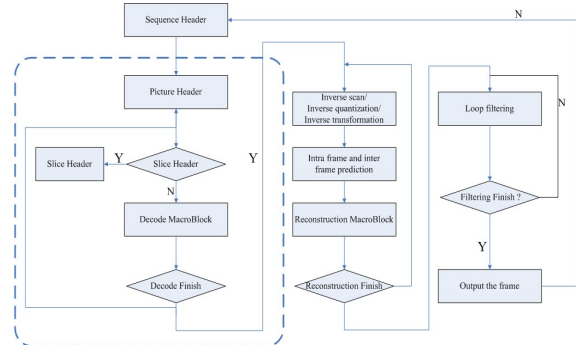


Figure 3. Flow chart of AVS decoder process

In this process: Sequence header / image header parsing, as the reference code; macroblock parsing parts, including the Slice header parsing and macro block parsing. The most important part of Slice header parsing data is the weighted prediction parameters, the data is stored into each macroblock of the Slice, while in each macroblock using Slice_id to identify and distinguish between different Slices. The output of this step is the arrays of the control structures macroblock and the residual data of the macroblock; loop filter section, filtering all borders to the unit of macro block, outputs of this step is decoded YUV data, for the using of reference frame and display of output.

C. Functional division of MPU and ENE in decoding process

CC1100 is a heterogeneous multi-core structure. It provides the corresponding programmable ENE and vector operation coprocessor for present video code. These coprocessors are relatively independent from to MPU, and can autonomously work under the control of MPU. Therefore, by reasonably dividing the functional modules, the running time between the modules can be parallel and data processing can be pipelining to achieve high-efficient decoding.

Design schemes of this paper: ENE deals with the entropy decoding. The vector processor deals with the part of complex operation. The MIPS's core deals with the decoding control, storage management, ENE, the control of the vector modules and input/output of data.

Form the ENE functional perspective, the factors must be considered as:

ENE is the specialized hardware accelerator dealing with entropy encoding/decoding. It supports fixed-length code, variable length code (VLC), unified variable length code, context-adaptive variable length code (CAVLC) and context-adaptive binary arithmetic code (CABAC). ENE can parse bits of random length and determine the element values by looking up the VLC table, which can be done in one instruction cycle.

The storage space of ENE is limited. Therefore the 16K U-Code RAM stores micro codes, the 2K Data RAM stores data and stacks and the 6K VLC RAM stores VLC code lists.

The input and output of ENE use DMA and FIFO, which differs from that of common co-processors. Figure 4 shows the interface between ENE and MPU:

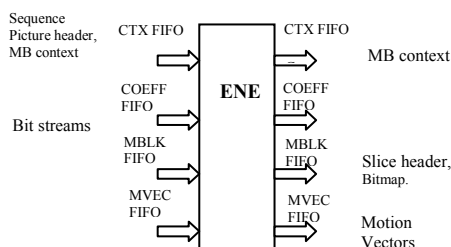


Figure 4. The interface between ENE and MPU

D. Design and implementation of prediction-modules of decoders

The intra-frame prediction of AVS predicts the pixel of the present block in units of 8×8 blocks with surrounding pixels, which has 5 brightness prediction modes and 4 colour prediction modes; the inter-frame prediction has 4 kinds of segmentation of 16×16 , 16×8 , 8×16 , 8×8 , which obtains the pixel of the present block by interpolating fraction pixels with reference pixels. The following factors should be considered in module design: the storage formats of input data; the unification of interpolating process; the determining order of interpolating branches.

1) Parsing design of prediction modes in ENE: This paper has designed an algorithm including both of the two parsing modes for intra-frame and inter-frame having a unified format of data transmission.

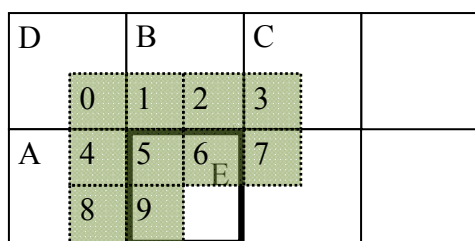


Figure 5. Distribution of Macroblock in Prediction Mode

As Figure 5 shows, the present block is E with A, B, C and D being the left, top, upper right and top left macro blocks respectively:

With the present block is 16×16 mode, 16×8 mode, 8×16 mode or 8×8 mode, the corresponding parameters of $4/1/3/0$, $4/1/3/0$ or $8/5/4/7$, $4/1/2/0$ or $5/2/3/1$, $4/1/3/0$, $5/2/3/1$, $8/5/6/4$

or $9/6/7/5$ in surrounding blocks are needed for the prediction of parameters of the present block[3].

2) Parsing design of prediction modes in MPU:

AVS adopts brightness interpolation prediction with an accuracy of $1/4$ pixel and color interpolation prediction with an accuracy of $1/8$ pixel to lower the code rate. Interpolation process takes up most of the time of motion compensation. AVS_P2 introduces 4 different block sizes (16×16 , 16×8 , 8×16 and 8×8). The motion vectors of small-size mode blocks can be predicted with that of larger-size mode blocks.

The reference graphs are stored in units of 4×4 block (a TILE) in the prediction process. The advantage of this format is, data of 4 TILES of 64bytes can be chosen as input and generate an output of 4×4 TILE for the line interpolation or column interpolation of macro blocks. Each TILE of the reference graph is taken from top field or bottom field and the TILE line of top field and that of bottom field are alternating arranged in view of the mix of frame fields.

The brightness interpolation process can be divided into three steps: four-order wave filtering of the level half-pixel; four-order wave filtering of the vertical half-pixel; four-order wave filtering of the $1/4$ pixel.

E. Design of parallel processing of ENE and MPU modules

As Figure 6 shows, parallel processing is achieved for ENE and MPU modules. The parallel gershgorim band and image layer both have start code flags. At the same time, both of them can be used as the implementation method of pipeline, because communication capacities of them are smaller than that of macro blocks and the load distributions of MPU and ENE operation are relatively balanced.

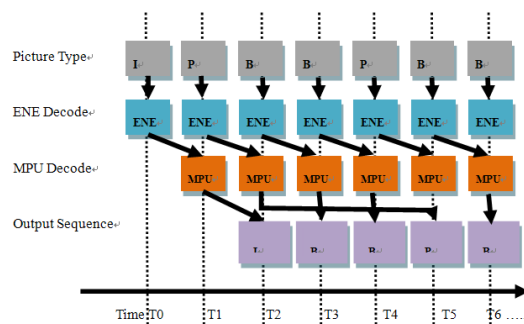


Figure 6. Parallel Mode between ENE and MPU

IV. FUNCTIONAL TEST AND PERFORMANCE OPTIMIZATION OF THE AVS DECODE

1) Test environment

On the CC100 PVR platform, by using Cop2 register of C2 JAZZB to count up the statistical runtime, the un-optimized, optimized and original RM52j_r codes are run respectively to compare their performance. At the same time, the unit test program of each core function is also run to measure its performance. The test sequence is the official published one with image resolution of D1 (720×576 pixels).

Target platform: a C2 CC1100 board (CPU: 300MHz)

Host platform: a personal computer installed with Fedora Core 7 Linux

Software for the target platform:

c2-sdk-1.3_11-devtools/kernel/ sw-media[4]

CC1100 has many kinds of bootstrapping modes. The mode the development board uses is to start from the SD card. Linux kernel and the frame of encoding and decoding are stored in the SD card. Therefore, after realizing the AVS decoder, it is needed to integrate into the program of the frame and store in the SD card. After these, tests can start to run.

2) Optimization of the hardware accelerate module ENE

Because of the restriction of hardware resources, the memory spaces of data and programs are very small. Therefore, the first problem we face is the number of codes and how to optimize the memory spaces when to optimize codes.

A .Optimization of data storage

For the optimization of data storage, the primary method is to arrange global variables and constants properly so as to reduce memory spaces. Used spaces:

- Jazz AVS ENE Code Size(.data)
1.695k Used 84.7%
- Jazz AVS ENE Code Size(.vlttable)
5.718k Used 95.3%

From the data above we can observe that almost all storage spaces have been occupied. The data are obtained under the condition that the data spaces are large enough at the expense of partial performance.

B .Optimization of program storage

For the optimization of program storage, there are mainly two aspects: 1. the modification of C languages to make the compiler to compile compacted codes; 2. assembly language are used for some compact codes which can not be generated by the compiler or are difficult to realize using C language.

The outputs are generated by the present using spaces:

- Jazz AVS ENE Code Size(.text)
-15.343k Used 95.9%.

Moreover, the assembly language when needed can also be used to increase the code efficiency and improve the code size.

3)The optimization mode in CPU

1. The inverse quantization, inverse transform and reconstruction are realized in the same module.

2. During the reconstruction[5], it is desired to open the inner loop of the 4 two-lever loops to realize the pipeline of the software.

3. Minimize the number of function calls of the core functions for all macro blocks.

4. Core functions, such as reconstruction, prediction and loop filtering, deal with two 8*8 blocks at one time. On the one hand, the number of loops can be reduced. On the other hand, there is more room for the compiler to realize the optimization so as to assemble instructions better.

4) Results analysis

The performance data of the decoder in Figure 7:

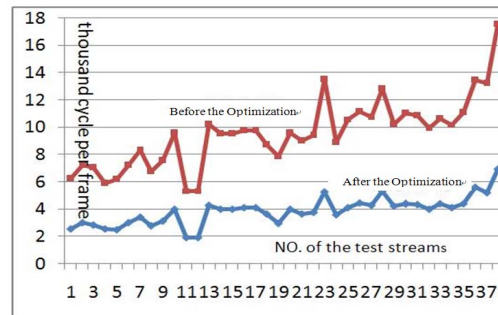


Figure 7. Comparison in the hardware accelerate module

From the test data above, it is observed that the optimized AVS decoder has a larger improve over the referent one.

The optimized AVS decoder has one time than the un-optimized one in speed, ten times than rm52j_r1; a large decrease in stall. These showed the optimization of the program.

According to the calculation of D1 (720*576) in 30 fps, the occupancy of CPU is about 200MHz. Cooperating with the other modules (Demux, audio decoder, reprocessing, broadcast), it entirely meet the requirement of real-time broadcast, and can support fast-forward, slow-motion and positioning etc. at a certain degree.

V. CONCLUSIONS

A deep research has been done in the paper on the design of the structure, realization of the key module and optimization of the AVS video decoder. We obtained some experiences based on the measurement of performances to find out the bottleneck and realize the optimization.

1. Make full use of on-chip high-speed storage and minimize the access time cooperating with the prefetched instruction or DMA;

2. Make full use of SIMD, and design the instruction algorithm based on the platform;

3. Properly arrange the codes according to cache to enhance the performance;

4. Properly design the pipeline and make full use of the special hardware and processor to transmit and deal with data in parallel.

There remain other modifications in the proposed decoder for the further study, such as a higher expansibility system structure. Although the proposed scheme can provide the optimum performance, expansibility will be affected because the proposed scheme coupled with standards is too close. A further research is needed on how to support the current standards by a common high-efficiency design with the more flexible division of the software and hardware and a higher expansibility.

REFERENCES

- [1] Zhang Chunyuan, *Stream Processor design and the relevant research*,
- [2] C2 Microsystems, CC1100 Data Book
- [3] Wen Gao, Cliff Reader, *AVS - The Chinese Next-Generation Video Coding Standard*.
- [4] C2 Microsystems, C2 Vector Programming Guide.
- [5] C2 Microsystems, CC1x00 Instruction Set.