Brief

Based on the paper “An efficient adaptive binary range coder and its VLSI architecture” published on IEEE Transactions on circuit and system for video technology, vol. 25, no.8, pp.1435-1446, 2015, we merged our arithmetic coder (AC) in the JPEG2000 framework to replace MQ. We implemented software and hardware on FPGA. Then we decided to publish it for research and evaluation.

Software

The compression software consists encode and decode execution programs which are compiled by visual studio 2010 on Windows 7 64 bits environment. The software only supports RAW images with 10 bits/pixel, little endian. The example usage is as follows:

To encode s001_10_q_s_t.raw,

xd_encode s001_10_q_s_t.raw 1024 1024 5

The first argument s001_10_q_s_t.raw is source image which will be compressed.

The second argument is width of image, which is 1024 in the example.

The third argument is height of image, which is 1024 in the example.

The fourth argument is target bit rate in bits per pixel. In this example, 5 means that the compression ratio is 2:1.

The compressed file is by default named test.dat in binary mode.

To decode test.dat

xd_decode test.dat rec.raw 1024 1024

The first argument test.dat is compressed file.

The second argument rec.raw is reconstruction image in RAW format.

The source image and reconstruction image can be viewed by Adobe PhotoShop for subject quality.

The third argument is width of image, which is 1024 in the example.

The fourth argument is height of image, which is 1024 in the example.

For object quality, we provide a PSNR tool, like:

psnr s001_10_q_s_t.raw rec.raw 1024 1024 1 10 ratio_s010.txt

The ratio_s010.txt is log file for writing PSNR value.

To compare with other compressions, we use JPEG2000 and HEVC standards. For JPEG2000, KDU7.1 is used. For HEVC, X265 v2.6 10bits is used.

KDU usage:

kdu_compress -i s001_10_q_s_t.rawl -o l.jp2 Nprecision=10 Nsigned=no Creversible=no Sdims={1024,1024} -rate 5

kdu_compress -i s001_10_q_s_t.rawl -o l.jp2 Nprecision=10 Nsigned=no Creversible=no Sdims={1024,1024} -rate 2.5

kdu_compress -i s001_10_q_s_t.rawl -o l.jp2 Nprecision=10 Nsigned=no Creversible=no Sdims={1024,1024} -rate 1.666667

kdu_compress -i s001_10_q_s_t.rawl -o l.jp2 Nprecision=10 Nsigned=no Creversible=no Sdims={1024,1024} -rate 1.25

kdu_expand -i l.jp2 -o rec.rawl

s001_10_q_s_t.rawl is same with s001_10_q_s_t.raw, just change the extension name according to the requirement of KDU softeware.

After decoding, we can also use PSNR tool to record PSNR value between two images.

X265 usage:

x265 --input s001_10_q_s_t400.yuv --input-depth 10 --frames 1 --input-res 1024x1024 --input-csp 0 --fps 1 --output test.mp4 -D 10 --psnr --tune psnr --bitrate 5243 --vbv-bufsize 5243 --vbv-maxrate 5243 --preset veryslow --strict-cbr

ffmpeg -i test.mp4 -y rec.yuv

x265 --input s001_10_q_s_t400.yuv --input-depth 10 --frames 1 --input-res 1024x1024 --input-csp 0 --fps 1 --output test.mp4 -D 10 --psnr --tune psnr --bitrate 2621 --vbv-bufsize 2621 --vbv-maxrate 2621 --preset veryslow --strict-cbr

ffmpeg -i test.mp4 -y rec.yuv

x265 --input s001_10_q_s_t400.yuv --input-depth 10 --frames 1 --input-res 1024x1024 --input-csp 0 --fps 1 --output test.mp4 -D 10 --psnr --tune psnr --bitrate 1748 --vbv-bufsize 1748 --vbv-maxrate 1748 --preset veryslow --strict-cbr

ffmpeg -i test.mp4 -y rec.yuv

x265 --input s001_10_q_s_t400.yuv --input-depth 10 --frames 1 --input-res 1024x1024 --input-csp 0 --fps 1 --output test.mp4 -D 10 --psnr --tune psnr --bitrate 1311 --vbv-bufsize 1311 --vbv-maxrate 1311 --preset veryslow --strict-cbr

ffmpeg -i test.mp4 -y rec.yuv

As the input of X265 is *.yuv, the image name is changed to s001_10_q_s_t400.yuv but the content keeps unchanged. For fair comparison, the option –strict-cbr is used to test the CBR control of HEVC. For decode, ffmpeg is used to expand code stream.

PSNR performance

Ten remote images with resolution 1024x1024, 10 bits/pixel are employed for comparison. Four common used ratios are tested in the satellite scenario, i.e., 2:1, 4:1, 6:1, and 8:1. The target bit rates are 5 bpp, 2.5 bpp, 1.67 bpp and 1.25 bpp. Table 1-3 shows the real bit rate of three softwares.

Table 1 the real bit rates of xd_encode, in bpp

Image	5	2.5	1.67	1.25
001	4.9709549	2.5589676	1.6987228	1.2533035
002	5.0410385	2.5042267	1.6655884	1.2536545
003	4.9867554	2.4858398	1.7053986	1.253479
004	4.9940796	2.4873505	1.6700668	1.2575912
005	5.0919266	2.4719086	1.674675	1.2519302
006	5.0132141	2.5214386	1.674263	1.2526245
007	5.1329117	2.5043411	1.6748505	1.2640457
008	4.9857483	2.4789429	1.6507416	1.2525787
009	4.9879761	2.5228271	1.617012	1.2520294
010	4.8835449	2.4729156	1.6873627	1.273674

Table 2 the real bit rates of KDU, in bpp

Image	5	2.5	1.67	1.25

001	4.972145081	2.490753174	1.652252197	1.247756958
002	4.998962402	2.465065002	1.638038635	1.227455139
003	4.934768677	2.464653015	1.666847229	1.239089966
004	4.990745544	2.497070313	1.6510849	1.234550476
005	4.997688293	2.50062561	1.650817871	1.250419617
006	4.935852051	2.488113403	1.654212952	1.228401184
007	4.923248291	2.475708008	1.650779724	1.23236084
008	4.998474121	2.466026306	1.649841309	1.249885559
009	4.966110229	2.489311218	1.663673401	1.231872559
010	4.971183777	2.50050354	1.652954102	1.23550415

Table 3 the real bit rates of X265, in bpp

Image	5	2.5	1.67	1.25
001	5.01554871	2.51501465	1.68245697	1.26570129
002	5.01554871	2.51501465	1.68245697	1.26570129
003	5.01554871	2.51501465	1.68245697	1.26570129
004	5.01554871	2.51501465	1.68245697	1.26570129
005	5.01554871	2.51501465	1.68245697	1.26570129
006	5.01554871	2.51501465	1.68245697	1.26570129
007	5.01554871	2.51501465	1.68245697	1.26570129
008	5.01554871	2.51501465	1.68245697	1.26570129
009	5.01554871	2.51501465	1.68245697	1.26570129
010	5.01554871	2.51501465	1.68245697	1.26570129

Table 4-6 show the PSNR of these softwares.

Table 4 the PSNR of xd_encode, in dB

Image	5bpp	2.5bpp	1.67bpp	1.25bpp
001	64.57	50.67	45.88	43.54
002	63.03	48.95	44.25	41.77
003	64.15	49.72	45.19	42.8
004	65.77	52.76	48.59	46.4
005	65.38	51.3	47.17	44.75
006	65.03	51.05	46.46	44.12
007	65.03	50.9	46.61	44.36
008	64.53	50.11	45.42	43.37
009	64.27	50.28	45.24	43.3
010	64.8	51.67	47.72	45.37
AVE	64.656	50.741	46.253	43.978

Table 5 the PSNR of KDU, in dB

Image	5bpp	2.5bpp	1.67bpp	1.25bpp
001	64.22	50.09	45.49	43.42
002	62.46	48.56	43.99	41.49
003	63.2	49.44	44.83	42.61
004	65.89	52.69	48.44	46.24
005	64.93	51.26	46.95	44.65

006	64.44	50.64	46.22	43.9
007	64.38	50.57	46.41	44.13
008	64.19	49.82	45.28	43.26
009	63.82	49.91	45.39	43.1
010	65.06	51.66	47.48	45.09
AVE	64.259	50.464	46.048	43.789

Table 6 the PSNR of X265, in dB

Image	5bpp	2.5bpp	1.67bpp	1.25bpp
001	57.04	47.22	44.97	42.74
002	54.51	47.09	43.19	40.87
003	56.18	47.47	44.29	42.45
004	59.34	50.17	46.86	45.7
005	57.84	49.02	44.04	42.85
006	57.03	47.98	45.6	43.16
007	57.15	48.53	44.52	43.67
008	57.92	47	44.63	41.48
009	56.29	48.14	42.87	41.32
010	58.53	51.39	47.25	43.98
AVE	57.183	48.401	44.822	42.822

Table 7-10 list the PSNR difference between xd_encode and KDU, X265 under ratio 2, 4, 6, 8.

Table 7 the PSNR difference between xd_encode and KDU, X265 under ratio 2

	KDU7.1	X265
MAX	0.95	8.52
MIN	-0.26	6.27
AVE	0.397	7.473

Table 8 the PSNR difference between xd_encode and KDU, X265 under ratio 4

	KDU7.1	X265
MAX	0.58	3.45
MIN	0.01	0.28
AVE	0.277	2.34

Table 9 the PSNR difference between xd_encode and KDU, X265 under ratio 6

	KDU7.1	X265
MAX	0.39	3.13
MIN	-0.15	0.47
AVE	0.205	1.431

Table 10 the PSNR difference between xd_encode and KDU, X265 under ratio 8

	KDU7.1	X265
MAX	0.28	1.98
MIN	0.1	0.35
AVE	0.189	1.156

From above tables, the xd_encode is better in PSNR than KDU and X265 under constant bit rate tests.

Hardware

After software implementation, hardware is then designed with verilog code. We use Xilinx ISE 14.6 as platform and provide the netlist (*.ngc) and post synthesis simulation model (*synthesis.v) for verification and simulation. In order to adapt different width of images, we realize 1024x1024, 2048x1024, 4096x1024 and 8192x1024 images resolutions. For throughput scalability, the internal cores for context and AC can be set to different numbers. For space application, only some FPGA chips and DDR1 can be considered with radiation harden methods. Thus we just use XQ5VFX130T-1 FPGA and one DDR1 (3D1D2G32TS2268IB, 64Mx32 bits) to store wavelet coefficients and code stream. Table 11 illustrates the resource utilization and synthesis frequency.

Table 11 the resource utilization of FPGA

device	resolution	Pixel precision (bits)	Slice Register	LUTs	BRAM	Clock Freq.(MHz)
xq5vfx130t	1024x1024	10	41196 (50%)	65382 (79%)	172 (57%)	93.056
xq5vfx130t	2048x1024	10	41357 (50%)	65852 (80%)	206 (69%)	93.025
xq5vfx130t	4096x1024	10	41783 (51%)	66287 (80%)	290(97%)	93.025
xq5vfx130t	8192x1024	10	42118 (51%)	67468 (82%)	465 (156%)	93.069

From table 11, the BRAM usage is overflowed under 8192x1024 resolution. Some bigger FPGA can be selected under this condition. Because of bandwidth limit of DDR1, the actual throughput cannot reach the clock speed. We test 1024x1024 image, the actual throughput is 56 MSPS (MSamples Per Second).

Simulation

A test bench file is provided in order to simulation. Here a header file named macro_spiht.h is used to define the width and height parameters and source file to be simulated. In order to simulate images, the binary file is converted to text mode file. Then the text mode image file is read by SOURCE_IMAGE_STIMUL.v, which sends the source image data to compression module. Table 12 lists the compression module ports.

Table 12 the compression module ports

Port	Direction	Width	Description
clock	IN	1	Main clock for compression
reset	IN	1	Async reset, low active
start	IN	1	Start on frame, high active, one clock cycle at least
target_rate	IN	20	Target rate in bytes
din	IN	10	Source image data input
din_valid	IN	1	Image data valid, high active
code_stream	OUTPUT	16	Compressed data output
code_stream_valid	OUTPUT	1	Compressed data valid, high active

cntrl0_ddr_dq	INOUT	32	DDR data
cntrl0_ddr_a	OUTPUT	14	DDR address
cntrl0_ddr_ba	OUTPUT	2	DDR bank address
cntrl0_ddr_cke	OUTPUT	1	DDR CKE
cntrl0_ddr_cs_n	OUTPUT	1	DDR CS
cntrl0_ddr_ras_n	OUTPUT	1	DDR RAS
cntrl0_ddr_cas_n	OUTPUT	1	DDR CAS
cntrl0_ddr_we_n	OUTPUT	1	DDR WE
cntrl0_ddr_dm	OUTPUT	1	DDR DM
cntrl0_ddr_dqs	INOUT	2	DDR DQS
cntrl0_ddr_ck	OUTPUT	1	DDR CLK
cntrl0_ddr_ck_n	OUTPUT	1	DDR CLK
cntrl0_reset_tb	OUTPUT	1	DDR control reset output
locked	INPUT	1	Lock of external clock, low active, for DDR control
clk0	INPUT	1	DDR control clock, 120MHz
clk90	INPUT	1	DDR control clock, 120MHz
clk200	INPUT	1	DDR control clock, 200MHz

In macro_parameter.h, image resolution should be selected for simulation. We define four different macros for 1024x1024, 2048x1024, 4096x1024 and 8192x1024. Each time, we can open or close these macros for the corresponding simulation.

The output of simulation is written into the code files in text mode. It is very easy to compare them with the software output.

FILE List

FILE name	Usage
\netlist\high_perform_top_1024x1024.ngc	Netlist for 1024x1024 images
\netlist\high_perform_top_2048x1024.ngc	Netlist for 2048x1024 images
\netlist\high_perform_top_4096x1024.ngc	Netlist for 4096x1024 images
\netlist\high_perform_top_8192x1024.ngc	Netlist for 8192x1024 images
\netlist\ high_perform_top_1024x1024.syr	Synthesis report for 1024x1024 images
\netlist\ high_perform_top_2048x1024.syr	Synthesis report for 2048x1024 images
\netlist\ high_perform_top_4096x1024.syr	Synthesis report for 4096x1024 images
\netlist\ high_perform_top_8192x1024.syr	Synthesis report for 8192x1024 images
high_perform_top_1024x1024_synthesis.v	Post synthesis simulation model for 1024x1024 images
high_perform_top_2048x1024_synthesis.v	Post synthesis simulation model for 2048x1024 images
high_perform_top_4096x1024_synthesis.v	Post synthesis simulation model for 4096x1024 images
high_perform_top_8192x1024_synthesis.v	Post synthesis simulation model for 8192x1024 images

tb_high_perform.v	Test bench
SOURCE_IMAGE_STIMUL.v	Source image generator for 1024x1024 images
SOURCE_IMAGE_STIMUL2048.v	Source image generator for 2048x1024 images
SOURCE_IMAGE_STIMUL4096.v	Source image generator for 4096x1024 images
SOURCE_IMAGE_STIMUL8192.v	Source image generator for 8192x1024 images
macro_parameter.h	Header file for image resolution selection
macro_spiht.h	Header file for image file for 1024x1024 images
macro_spiht2048.h	Header file for image file for 2048x1024 images
macro_spiht4096.h	Header file for image file for 4096x1024 images
macro_spiht8192.h	Header file for image file for 8192x1024 images
ddr_model_parameters.vh	DDR model header file
source_sim_1.dat	Source image file in text mode for 1024x1024 image
s001_003_10_q_s_t_2048.dat	Source image file in text mode for 2048x1024 image
s001_005_10_q_s_t_4096.dat	Source image file in text mode for 4096x1024 image
s001_10_q_s_t_8192.dat	Source image file in text mode for 4096x1024 image
\Cmodel\xd_encode.exe	Encode program
\Cmodel\xd_decode.exe	Decode program
\Cmodel\kdu_compress.exe	KDU7.1 encode program
\Cmodel\kdu_expand.exe	KDU7.1 decode program
\Cmodel\x265.exe	X265 v2.6 10bit
\Cmodel\ffmpeg.exe	ffmpeg program
\Cmodel\s001_10_q_s_t.raw	Source image in binary mode, little endian
\Cmodel\s001_10_q_s_t.rawl	Source image in binary mode, little endian for KDU7.1
\Cmodel\s001_10_q_s_t400.yuv	Source image in binary mode, little endian for X265

Contact

Kai Liu, Xidian University

Email: kailiu@mail.xidian.edu.cn

Cell : +86 13892810532