Release Notes for JPEG Decoder on ARM11 ELINUX

ABSTRACT:

Release Notes for JPEG Decoder on ARM11 ELINUX

KEYWORDS:

Multimedia codecs, JPEG

APPROVED:

Wang Zening

Revision History

Version	Date	Author	Change Description
1.0	14-Oct-2004	B Venkatarao	Release 1.0
2.0	17-Jan-2005	B Venkatarao	Release 2.0
3.0	28-Mar-2005	B Venkatarao	ARM11 release 3.0
3.1	05-Sep-2005	Shankar Narayan P.S	Release with provision to build shared library
4.0	06-Feb-2006	Lauren Post	Using new format
4.1	28-March-2006	Sriram Shankar	Document review
4.2	04-March-2009	Eagle Zhou	Document update
4.3	25-May-2009	Eagle Zhou	Add VPU decoding
4.4	19-Oct-2009	Eagle Zhou	Refine
5.0	14-Jan-2010	Eagle Zhou	Modify Downscale scheme Add orientation tag

Table of Contents

Introduction

1.1 Purpose

The purpose of this document is to provide information on the package contents, instructions on building library and test applications and test execution on ARM11 ELINUX, RVDS and Linux x86

1.2 Scope

The scope is restricted to information on the package contents and instructions for building and testing. This document does not provide architecture or details about the APIs provided in the package. Performance data will be provided in another document as detailed in the Requirements Book.

1.3 Audience Description

The reader is expected to have basic understanding of Image processing and JPEG Sequential and Progressive decoding.

1.4 References

1.4.1 Standards

- DIS 10918-1 and draft DIS 10918-2
- "JPEG Still Image Data Compression Standard" by William B. Penne baker and Joan L. Mitchell published by Van No strand Reinhold, 1993, ISBN 0-442-01272-1. 638 pages, price US$59.95. This book includes the complete text of the ISO JPEG standards (DIS 10918-1 and draft DIS 10918-2).

1.4.2 General References

- Wallace, Gregory K. "The JPEG Still Picture Compression Standard", Communications of the ACM, April 1991 (vol. 34 no. 4), pp. 30-44.

1.4.3 Freescale Multimedia References

- JPEG Decoder Application Programming Interface – jpeg_dec_api.doc
- JPEG Decoder Release notes – jpeg_dec_release_notes.doc
- JPEG Decoder Interface header – jpeg_dec_interface.h
- JPEG Decoder Test Application – jpeg_dec_app.c

1.5 Definitions, Acronyms, and Abbreviations

TERM/ACRONYM	DEFINITION
API	Application Programming Interface
ARM	Advanced RISC Machine
Data Unit	JPEG proposal defines a data unit as a sample in predictive codecs and a [8x8] block in case of DCT based codecs
DCT	Discrete Cosine Transform
FSL	Freescale
IDCT	Inverse Discrete Cosine Transform
IJG	Independent JPEG Group
JPEG	Joint Photographic Experts Group
MCU	Minimum Coded unit. JPEG proposal defines an MCU as the smallest group of interleaved data units
RVDS	ARM RealView Development Suite
TBD	To Be Determined
UNIX	Linux PC x/86 C-reference binaries

1.6 Document Location

docs/jpeg_dec

2 Release History

RELEASE NUMBER	DELIVERABLES	FEATURES
1.0		• Engineering Release
2.0	• Documentation • Application Interface header file • ELINUX and RVDS libraries and test applications • UNIX/Linux x/86 Reference library and test application • Makefiles and Source code for library and test application including optimized assembler for the ELINUX and RVDS libraries. • Test vectors	• Supports Baseline and Progressive modes. • Supports all the integer sampling formats allowed by the Baseline DCT coding. (4:4:4, 4:2:2, 4:2:0, 4:1:1 etc) • Supports any image size upto 64Kx64K. • Supports the output format to be same as encoded image format. • Supports color conversion from YUV to RGB-565 or RGB-888. • Supports resizing (down) upto integer factor of 32. This is supported only for RGB outputs. • Supports restart markers. • EXIF file format supported • Compressed Thumbnails supported. Both JFIF and EXIF thumbnails are supported
3.1	Same	• Shared Library Support • Upgrade to RVDS 2.2
4.2	Same	• Decode frame API support
4.3	Same	• Support VPU decoding(baseline)
5.0	Same	• Modify Downscale: Remove original integer resize (in post process) Add Downscale for 1/2 x 1/2, 1/4 x 1/4, 1/8 x 1/8 (in decoding process) • Add orientation tag

Table 1. Details of the release

2.1 Assumptions and Known Problems

None

2.2 Contacts

Please report any problems to the following email address: mmsw@freescale.com

3 List of Deliverables

3.1 Documentation

Base directory: /ARM11/

Subdirectory	Files
docs/jpeg_dec	jpeg_dec_api.doc jpeg_dec _release_notes.doc jpeg_dec_data_sheet.doc

3.2 Public Headers

Base directory: /ARM11/

Subdirectory	File
API_include	jpeg_dec_interface.h

3.3 Test Application Source

Base directory: /ARM11/src/image/jpeg_dec

Subdirectory	Files
test/	"Makefile" makefile for building RVDS, UNIX and ELINUX board executables.
test/c_source	*.c, a) application code. b) Debug log code. It contains implementation of functions used for logging data and messages.
test/test_util/scripts	Batch files to be run on the board

3.4 Library Source

Base directory: /ARM11/src/image/jpeg_dec

Subdirectory	Files
library	Makefile "Makefile" for building RVDS, UNIX, and ELINUX libraries. libjpeg_dec_arm11_RVDS.a – Special options for simulator testing libjpeg_dec_arm11_ELINUX.a - static library for board libjpeg_dec_arm11_ELINUX.so – shared library for board libjpeg_dec_UNIX.a – library for Linux x/86 – c reference code
library/c_source	*.c, JPEG decoder source code
library/asm_source	*.s, JPEG decoder assembler code
library/include	*.h, JPEG decoder library header files

3.5 Common Makefiles

Base Directory: /ARM11/common

Makefile	Description
common.mk	This is a common makefile included in the codec library makefile for building the libraries. This file includes common options used by all codecs. Following flags can be overwritten or added to in the codec library makefile 1. Path to toolchain tools (TC_ROOT) 2. GNU header file path (HEADER_PATHS) 3. GNU library path (LIB_PATHS) 4. GNU Compiler/Assembler Options (GNU_CFLAGS, GNU_AFLAGS) 5. Endian Flags 6. Optimization Flags(OPTIM_LEVEL, OPTIM_TYPE) 7. Common options for RVDS,UNIX and ELINUX (CFLAGS,AFLAGS) 8. Build specific flags 9. Source directory of 'C' code 10. Source directory of 'assembly(.s)' code 11. Object directory for .o files 12. RVDS Compilation Tools 13. Codec header path 14. Arguments for librarian for UNIX builds 15. SHARED_ELINUX builds for libraries that must be linked using the toolchain because of external library includes.
common_testapp.mk	This is the common makefile included in the codec test makefile for building the test application. This file includes the common options used by the all the codecs. Following flags can be overwritten or added to in the codec test makefile 1. Toolchain path depending on the build option 2. Compiler Flags 3. Linker flags 4. Paths for c_source, exe and object directories 5. Codec header files' INCLUDES path 6. Endian Flags 7. CODEC_LIB generation

3.6 Test Vectors

Base Directory: multimedia_vectors/test_vectors

The test vectors are provided in another location from the library and test source

Subdirectory	Files	Description
jpeg2k_dec/ input	*.jpg	Input test vectors
jpeg_dec/ref/yuv jpeg_dec/ref/ rgb_888 jpeg_dec/ref/ rgb_565 jpeg_dec/ref/ scale	*.bin	Previously generated reference outputs which can be used as a reference.

4 Software Setup & Tools used

- ARM RVDS 2.2 (build 503) should be installed in the PC.
- Freescale Linux OS Release L26.1.15 must be running on the evaluation board.
- Intel based Red Hat Linux Machine must have the Montavista toolchain installed on it.
    - `MontaVista 3.4.3-25.0.36.0501313 2005-08-21`
- 'Cygwin' **Version** CYGWIN_NT-5.1, a freely downloadable linux emulator is installed in PC - **http://www.cygwin.com/**.
- 'make' utility available for targeted platforms

5 Build Procedure

All the required makefiles are provided under individual directories. The library can be built for windows / target processor (ARM1136J-S). The details for the build procedure are described below.

5.1 Library

To build the library, run 'make' on 'Makefile' from library directory. The makefile shall create the required directory to hold the object files. The makefile can be used if you want to build the library only. The same makefile can used to build libraries for both board, Unix/Linux and RVDS with different build options. The following options are available to build the library.

Options
a) **BUILD options**:
- o **BUILD= ELINUX** : This is the default option and builds both static library 'libjpeg_dec_arm11_ELINUX.a' and shared library 'libjpeg_dec_arm11_ELINUX.so' , for testing on the board.

- o **BUILD=RVDS**: This option builds the static library 'libjpeg_dec_arm11_RVDS.a', for testing on RVDS (Armulator).

- o **BUILD=UNIX**: This option builds the static library 'libjpeg_dec_UNIX.a', for testing on UNIX/Linux machine.

     Eg: make BUILD= ELINUX
                       make BUILD=RVDS
                       make BUILD=UNIX

b) **ENDIAN options for RVDS**:
- o **TARGET_ENDIAN=LITTLE**: This is the default option and sets the endian-ness to 'little'
- o **TARGET_ENDIAN=BIG**: This option sets the endian-ness to big
         Eg: make BUILD=RVDS TARGET_ENDIAN=BIG

c) **clean options**:
- o **clean_RVDS**:   Deletes all the object files and the RVDS library 'libjpeg_dec_arm11_RVDS.a'.
- o **clean_ELINUX**: Deletes all the object file and the ELINUX libraries libjpeg_dec_arm11_ELINUX.a and libjpeg_dec_arm11_ELINUX.so.
- o **clean_UNIX**:   Deletes all the object files and the UNIX library 'libjpeg_dec_UNIX.a'.
- o **clean**: Deletes all the object files and RVDS,UNIX and ELINUX libraries.

Note: Make appropriate changes in file 'common.mk' at directory 'ARM11/common' for the location of toolchains.

The library that is built is saved as libjpeg_dec_arm11_RVDS.a for RVDS build, and libjpeg_dec_arm11_ELINUX.a and libjpeg_dec_arm11_ELINUX.so for board build. These libraries are saved in the current directory (the same directory in which the source and assembly directories are listed).

Target	Compilation Environment	Build Options	Library Name
Board	PC (Using Cygwin)	BUILD= ELINUX	libjpeg_dec_arm11_ELINUX.a libjpeg_dec_arm11_ELINUX.so
RVDS	PC (Using Cygwin)	BUILD=RVDS TARGET_ENDIAN= BIG/LITTLE	libjpeg_dec_arm11_RVDS.a
Unix/ Linux	Unix/Linux machine	BUILD=UNIX TARGET_ENDIAN= BIG/LITTLE	libjpeg_dec_UNIX.a

5.2 Test Application

To build the test application, run 'make' on 'Makefile' from the test directory. This makefile can create executables for testing on Linux x86, the ARM11board and RVDS for ARM11. The executables jpeg_dec_arm11_RVDS' for RVDS, jpeg_dec_arm11_ELINUX for board and jpeg_dec_UNIX for UNIX are stored under test/exe directory. The makefile shall create the required directory structure to hold the object files and executables. The following commands should be invoked so as to build the executables.

Options
1) **BUILD options**:
   o **BUILD=ELINUX**: This is the default option and builds the executable 'jpeg_dec_arm11_ELINUX', for the board.

   o **BUILD=RVDS**: This option builds the executable 'jpeg_dec_arm11_RVDS' for the RVDS (Armulator).

   o **BUILD=UNIX**: This option builds the executable 'jpeg_dec_UNIX' for the Unix/Linux machine.

   Eg: make BUILD=ELINUX (for board)
                    make BUILD=RVDS   (for Armulator)
                    make BUILD=UNIX   (for Unix/Linux machine)

2) **ENDIAN options for RVDS**:

o **TARGET_ENDIAN=LITTLE**: This is the default option and sets the endian-ness to 'little'

o **TARGET_ENDIAN=BIG**: This option sets the endian-ness to big
Eg: make BUILD=RVDS TARGET_ENDIAN=BIG

3) **LIBRARY options**:
o **LIB= STATIC**: This option builds the ELINUX test application linked with the ELINUX static library 'libjpeg_dec_arm11_ELINUX.a'.If nothing is specified ,the executable links with shared library 'libjpeg_dec_arm11_ELINUX.so'

   Eg: make LIB=STATIC

4) **clean options**:
o **clean_RVDS**: Deletes all the object files and the RVDS executable 'jpeg_dec_arm11_RVDS' .
o **clean_ELINUX**: Deletes all the object file and the ELINUX 'jpeg_dec_arm11_ELINUX'.
o **clean_UNIX**: Deletes all the object files and the Unix/Linux executable 'jpeg_dec_UNIX'.
o **clean**: Deletes all the object files and RVDS,UNIX ELINUX executables.

Note:
In 'common_testapp.mk' at directory 'ARM11/common', the paths for the compiling and linking tools are hard coded for the current set-up. These paths may not be the same in the user's directory set up. Hence, the 'common_testapp.mk' should be modified to point to the directories where the linking and compilation tools are present before building the application for board.

The following table summarises the build options,

Target	Compilation Environment	Build Options	Executable Name
Board	Redhat Linux Machine	BUILD=ELINUX LIB= STATIC	jpeg_dec_arm11_ELINUX
RVDS	PC (Using Cygwin)	BUILD=RVDS TARGET_ENDIAN= LITTLE/BIG	jpeg_dec_arm11_RVDS
UNIX/ Linux	Unix/Linux machine	BUILD=UNIX TARGET_ENDIAN= LITTLE/BIG	jpeg_dec_UNIX

6 Test Application Execution

To know the options provided by the test application, run the executable without any argument. It shall print a brief summary of all the options available.

6.1 ELINUX

test_jpeg_dec_arm11_elinux –scale <scale_val> –vpu <vpu_enable> -mode <mode> -exif_info <exif> -ofmt <output format> -outfile <output file> <input file>

scale option:

  1 - no scale (default)

  2 - 1/(2x2) downscale

  4 - 1/(4x4) downscale

  8 - 1/(8x8) downscale

vpu option:

  1: enable vpu decoding

  0: disable vpu decodeing (default)

mode option:

  0: primary image

  1: thumbnail image

  2: both (default)

exif_info option:

  0: don't require exif information (default)

  1: require exif information

ofmt option:

  0: Same as Encoded image format (default)

  1: RGB 888 format

  2: RGB 565 format

  3: BGR 888 format

  4: BGR 565 format

7 Pre compilation Options

The following C options need to be set

C Defines	Description	Remarks
LOG_TIMING	To log performance timing results	