RELEASE NOTES

Real Audio Decoder Test FWK Algorithm Release Notes OMAP1710 (C55x)

Document Revision: 1.0

Issue Date: 01 March 2004

Table of Contents

Revision History

REV	DATE	AUTHOR	NOTES
1.0	01 Mar 2004	Srinivasa M.K	Initial Release

1. Introduction

This package contains the following items:

- Texas Instruments Real Audio Decoder Algorithm Test Framework on OMAP1710 (C55x)
- Input Test streams
- Reference Test streams
- Utilities for comparing the Decoder test streams with Reference Streams
- Batch File to build the project, and Compare the Outputs with references
- Command file to Run the application, and testing with All Stack /Interrupt mode Combinations

Note: The term “C55x” refers to the Texas Instruments TMS320C55xxseries of DSP platforms. These DSP are members of the TI TMS320C5000™ family.

1.1. Purpose

This document describes the contents of this package and provides information on using the package.

1.2. Scope

This document is intended to be used as a starting point for working with this test framework release. It describes the content, dependencies and verification procedure for this package. Some environment setup information may be provided.

This document is not intended to be a user guide for the Real Audio Decoder algorithm or the test framework.

1.3. Audience

The intended audiences for this document are developers; test & quality engineers who wish to use the Real Audio Decoder test framework to perform tests on Real Audio Decoder Algorithm; or those looking for a sample application source code on how to use the algorithm.

2. Release Content

2.1. Package Directory Structure

2.1.1. RADECODER Application dir Description

`OMAPSW_DSP\test\algorithm\audio\ra\decoder\fw_c\build`

Has the Code composer studio project file & linker command file

`\OMAPSW_DSP\test\algorithm\audio\ra\decoder\fw_c\include`

Has the header files used by the test framework.

`\OMAPSW_DSP\test\algorithm\audio\ra\decoder\fw_c\src\c`

Contains the C source code of the test framework/application

`\OMAPSW_DSP\test\algorithm\audio\ra\decoder\fw_c\src\asm`

Contains the Assembly source code of the test framework/application

2.1.2. Test Directory Description

The test directory contains all test files required for testing the Real Audio Decoder algorithm library. The directory layout of this folder is as follows:

`\test patterns\output`

Contains output files generated by the Real Audio Decoder algorithm.

`\testpatterns\utils`

Contains the utilities for viewing final output.

Note: Test streams are kept in the tool server, hence the folders input reference are moved to **the tool server from test patterns directory .**

The system variable RA_TEST_STREAM has to be set with the path of the Real Audio Decoder test streams in tool server.

Example: In this case assumption is that D drive as tool server, the environment variable RA_TEST_STREAM has to be set D:\TestPatterns\audio\ ra\decoder With this modification the config_file.txt and the difference calculator batch files are generated when the command file execute the algorithm for testing

3. Validation

3.1. Testing methodology

Two modes of algorithm validation can be performed using test framework:

1. Automated testing: By executing the script file *RAdec.cmd* complete Decoder Application is tested the Build reports of both algorithm and application are captured in the text RA_DEC_build_log.txt, and the testing reports of the application are captured in the text document *RA_DEC_test_log.txt*
2. Manual Testing: Manual testing is done by building the Library and application projects followed by the running the application for particular test stream and comparing the output with respective reference output:

3.2. Test Environment

The test procedure assumes that the following software/hardware requirements are met.

- Code Composer Studio™ version 2.21.
- Cgtools version 2.74
- Windows® 2000 OS.
- OMAP1710 TEB
- In order to be able to build correctly, please make sure that the environment variables are set correctly. In order to compile correctly with Code Composer Studio™, make sure that the user variable TI_DIR is set to Drive:\ti in the Environment Variables of Windows.
- Make sure that External memory is enabled before start working with DSP if the code is using external memory.
  - Initialize the external memory of the DSP by using the appropriate gel file

Note:

Make sure that Range of the external memory mentioned in the command file is matches with the gel file initialized external memory range.

3.3. Test Tools Description

Build_RA_dec.bat

This batch file is used to build the both algorithm and test frame -work projects ,this batch file required one parameter to specify Projects has to build in the *debug* mode or *release* mode. The procedure to run this batch file is given below

To Build the library and Application project in debug mode, the syntax is:

```
Build_RA_dec.bat  DEBUG
```

To Build the library and Application project in release mode, the syntax is

```
Build_RA_dec.bat  RELEASE
```

file_diff.exe

This application is used to compare the to data files ,and it gives the difference if it is not equal to zero , and it compares the sizes of output file and corresponding reference file . the syntax to run this application is

```
file_diff     outputfile name       reference file name
```

radiff.ba

This is batch file is used to compare *the set of* decoded generated output files with the corresponding reference files. and store the compared result in the test log text document. Syntax to run this batch file is

```
radiff.bat
```

radec.cmd

This script file is used to test the decoder application, this script file uses the batch files ***build_RA_dec.bat*** for building the project in different modes i.e. debug and release modes, and ***radiff.bat*** for testing the output files with that of reference files. and this script file is used to test the decoder application in different modes such as interrupt on/off and stack(three different stack) modes.

Config_file.txt

This text file contains the input parameters for the decoder algorithm these parameters are explained below.

The format of the *Config_file.txt* file is:

```
InputFilename OutputFileName nSamples nFrameBits sampRate nRegions nChannels
lossRate cplStart cplQbits
```

i) InputFilename with path (relative to *application\build\debug* folder)

ii) OutputFileName with path (relative to *application\build\debug* folder)

iii) **nSamples:** Samples per frame :/* 1024, 512 or 256 */

iv) **nFrameBits:** No of bits per frame

v) **sampRate**: Sampling rate

vi) **nRegions** :No of regions per frame

vii) **nChannels:** No of channels

viii) **lossRate**: Loss rate

ix) **cplstart**: start of coupling info

x) **cplQbits**: coupling Q bits

3.4. Automated Test Procedure

All automatic test cases are executed using a Code Composer Studio™ test script *radec.cmd* (provided in utils folder of the Real Audio Decoder package). The procedure for running this script is described below.

2. Set an environment variable named TI_DIR in windows, to the directory where Code Composer Studio™ is installed (e.g. c:\ti or d:\ti). Setting of environment variable may be dependent on the windows OS. For windows 2K, right click my computer icon on desktop, select *advanced tab* and select *environment variables* button. On the new pop up window select *New*. In the new window set *Variable Name* as TI_DIR, and *Variable Value* to Code Composer Studio™ folder name (c:\ti etc).

Open Code Composer Studio™ (DSP side)

From *Tools* menu select *Command Window*.

On the command window that comes up now, type *chdir RA_dec _ test_ directory. RA_dec _ test_ directory* is the *full directory path name* of the folder where *Real Audio decoder test* folder resides.

For example in this case the chdir is

```
Chdir Z:\OMAPSW_DSP\test\algorithm\audio\RA\decoder\testpatterns\utils
```

Now type *take* ***radec.cmd*** in the command window. Test will take approximately 2 hrs to complete. Once the test is over a message “HAVE A NICE DAY” will be displayed in the command log window.

After the completion of script file execution, the test results can be obtained from log file ***RA_DEC_test_log.txt*** (located in *utils* folder). All build results will be logged in ***RA_DEC_build_log.txt*** (release and debug modes) file.

The execution flow of the *radec.cmd* test script is given below.

1. Builds the *Real Audio* decoder algorithm and application project Debug mode.
2. Enables timer interrupt (this script is meant for OMAP1710).
3. Runs the application for all test vectors in all three-stack modes (fast 16, slow 16 and slow 32 modes). After completion of run in each stack modes, outputs are compared against reference output (located in *\testpatternst\reference* folder) streams.
4. Disables the interrupt.
5. Runs all tests mentioned in step 3.
6. Build *Real Audio* Decoder algorithm and application in release mode Repeats steps 2 to 5.

3.5. Manual Test Procedure

3.5.1. Build Procedure

Manual testing involves testing the *Real Audio* Decoder algorithm, The first step in any of these testing is building the algorithm in debug mode or release mode This task can be easily performed using a DOS bat file, *build_RA_dec.bat,* provided in the *utils* folder

Note that like earlier test scripts, this bat file also requires the user to set TI_DIR environment variable.

The usage is, under DOS command window type

- Build_RA_dec.bat DEBUG
- This will build the algorithm and application projects in debug mode
- Build_RA_dec.bat RELEASE

This will build the algorithm and application projects in release mode

3.5.2. Verification Procedure

After completion of each manual test, the test resource should compare the outputs generated against reference output. A DOS bat file, RA*diff.bat* (in *utils* folder) is provided to automate this task of comparing the outputs against reference outputs. The test resource can modify (to specify the test streams to be compared) and run this bat file after completion of each test to verify the outputs generated against the reference outputs. This batch file uses *file_diff.exe* (*utils* folder) utility for comparison purpose. The usage of *file_diff.exe* is

```
File _diff   file1  file2
```

Where *file1* and *file2* are the two files are to be compared against each other.

4. Issues and constraints

Refer to documents OMAPSW_Assumptions_and_Constraints.xls in