G.711 Waveform Coder

With Appendices 1 & 2

PRODUCT DESCRIPTION

G.711 is, by far, the most commonly supported voice companding algorithm used in telephony. It has become the de facto standard used to ensure interoperability in voice over Internet protocol (VoIP) applications. Compression is performed on a per sample basis with each uniformly quantized sample producing an 8-bit pulse code modulated (PCM) or companded value.

In addition to companding, appendices have been added to the G.711 standard that allow telephony equipment to reduce network traffic by transmitting minimal data during periods of silence and to improve playback quality by synthesizing speech during periods when data has been lost or corrupted.

Adaptive Digital's G.711 waveform coder is a bit exact implementation of the ITU G.711 PCM standard. Adaptive Digital's G.711 coder converts between 8-bit mu-law (American standard) or a-law (European standard) companded values and 16-bit uniformly quantized values. Although many DSPs support G.711 conversion via hardware interface to TDM highways, it is often necessary to perform G.711 conversion in software, particularly when voice and signaling applications share a common line. Unlike the G.711 companding standard, the two G.711 appendices are not bit-exact. As a result, not all implementations perform identically.

G.711 **Appendix 1** (optional) coder is a high quality low-complexity algorithm for packet loss concealment (PLC). It is a highly effective algorithm for concealing lost packets of G.711 data. Speech remains intelligible even under conditions where up to 30% of the packets are lost.

G.711 **Appendix 2** (optional) provides voice activity detection (VAD), discontinuous transmission (DTX), and comfort noise generation (CNG). When combined, these algorithms provide an efficient method for reducing packet bandwidth during portions of a conversation when no voice activity is present. During silence periods, only the bits necessary to model the background noise characteristics

FEATURES

- Supports both mu-law and a-law
- ITU G.711 Compliant
- C Callable
- Multi-channel capable
- Robust packet loss concealment improves voice quality under packet loss conditions

AVAILABILITY

ADT G.711 is available on the following Platforms: Other configurations are available upon request.

Product	Platform	Memory Model	Endian	Code Gen Tool Version
ADT_g711_c54x	TI TMS320C54x	Far	N/A	N/R
ADT_g711_c55x	TI TMS320C55x	L3	Little	N/R
ADT_g711_c64x	TI TMS320C64x	L3	Little	N/R
ADT_g711_arm7	ARM7	N/A	Little	N/R
ADT_g711_arm9	ARM9	N/A	Little	N/R
ADT_g711_arm11	ARM11	N/A	Little	N/R

ADT G.711 Appendices 1 and 2 are available on the following Platforms: Other configurations are available upon request.

Product	Platform	Memory Model	Endian	Code Gen Tool Version
ADT_g711a1_c54x	TI TMS320C54x	Far	N/A	N/R
ADT_g711a1_c55x	TI TMS320C55x	Large	Little	N/R
ADT_g711a1_c64x	TI TMS320C64x	L3	Little	N/R
ADT_g711a1_arm9	ARM9	N/A	Little	N/R
ADT_g711a1_arm11	ARM11	N/A	Little	N/R
ADT_g711a2_c55x	TI TMS320C55x	Large	N/A	N/R
ADT_g711a2_c64x	TI TMS320C64x	L3	Little	N/R
ADT_g711a1a2_c54x	TI TMS320C54x	Far	N/A	N/R
ADT_g711a1a2_c64x	TI TMS320C64x	L3	Little	N/R
ADT_g711a1a2_c64x+	TI TMS320C64x+	L3	Little	N/R
ADT_g711a1a2_arm9	ARM9	N/A	Little	Code Sourcery 2011.09-70
ADT_g711a1a2_arm11	ARM11	N/A	Little	N/R
ADT_g711a1a2_m3	ARM Cortex-M3	N/A	Little	Code Sourcery 2011.09-70
ADT_g711a1a2_m4	ARM Cortex-M4	N/A	Little	GCC v 4.5.2*
ADT_g711a1a2_a8	ARM Cortex-A8	N/A	Little	Code Sourcery 2011.09-70
ADT_g711a1a2_a9	ARM Cortex-A9	N/A	Little	Android NDK r6b
ADT_g711a1a2_a15	ARM Cortex-A15	N/A	Little	Android NDK r6b
ADT_ g711a1a2_win32dll	Win32	N/A	Little	VS2010
ADT_ g711a1a2_win32lib	Win32	N/A	Little	VS2010
ADT_g711a1a2_i686	i686 linux	N/A	Little	GCC

*GCC v 4.5.2 (Sourcery G++2011.03-41)

Endian, byte order: “Little Endian” means that the low-order byte of the number is stored in memory at the lowest address, and the high-order byte at the highest address. "Big Endian" means that the high-order byte of the number is stored in memory at the lowest address, and the low-order byte at the highest address.

Acronyms

Mm – Memory Model: Memory Model is specific to Texas Instruments processors.

N/A – Not Applicable

N/R – Not Recorded

SPECIFICATIONS

TI TMS320

C54x

CPU UTILIZATION & MEMORY REQUIREMENTS

All Memory usage is given in units of 16-bit word.

Function	MIPS (Peak)	Program Memory	Data Memory	Per-Channel Data Memory
Mu-law Encode	0.4	--	--	--
Mu-law Decode	0.1	--	--	--
A-Law Encode	0.25	--	--	--
A-Law Decode	0.1	--	--	--
G.711 Common/Tables	--	644	4	--
G.711 Appendix 1	0.37	1593	49	836
VAD - G.711Appendix 2	3.2	4935	1447	434
CNG – G.711 Appendix 2	1.3	877	181	210

C55x

CPU UTILIZATION & MEMORY REQUIREMENTS

All Memory usage is given in units of byte.

Function	MIPS (Peak)	Program Memory	Data Memory	Per-Channel Data Memory
Mu-law Encode	0.14	--	--	--
Mu-law Decode	0.07	--	--	--
A-Law Encode	0.14	--	--	--
A-Law Decode	0.07	--	--	--
G.711 Common/Tables	--	262	1032	--
G.711 Appendix 1	0.27	2162	98	1680
VAD - G.711Appendix 2	2.9	10388	3464	1352
CNG – G.711 Appendix 2	0.9	1942	682	568

C64x

CPU UTILIZATION & MEMORY REQUIREMENTS

All Memory usage is given in units of byte.

Function	MIPS (Peak)	Program Memory	Data Memory	Per-Channel Data Memory
Mu-law Encode	0.05	--	--	--
Mu-law Decode	0.02	--	--	--
A-Law Encode	0.05	--	--	--
A-Law Decode	0.02	--	--	--
G.711 Common/Tables		1696	1024	--
G.711 Appendix 1	0.36	4990	98	1680
VAD - G.711Appendix 2	1.73	16032	1548	1644
CNG – G.711 Appendix 2	0.44	3584	272	104

ARM

ARM7

CPU UTILIZATION & MEMORY REQUIREMENTS

All Memory usage is given in units of 8 bit byte.

Function	Per-Sample Cycles	Program Memory	Table Memory	Per-Channel Data Memory
Mu-law (Compress) Encode	69	--	--	--
Mu-law (Expand) Decode	20	--	--	--
A-Law (Compress) Encode	70	--	--	--
A-Law (Expand) Decode	15	--	--	--
G.711 Common/Tables	--	900	1024	0

ARM9/ARM11

CPU UTILIZATION & MEMORY REQUIREMENTS

All Memory usage is given in units of 8bit bytes.

Product	Function	MIPS	Program Memory	Data Memory	Per-Channel Data Memory	Scratch Memory
G.711A1A2	Encode	12.1	61k	5.26k	1640	624
	Decode	3.8			1776	624

ARM CORTEX-M3 / CORTEX-M4

CPU UTILIZATION & MEMORY REQUIREMENTS

All Memory usage is given in units of byte.

Product	Function	MIPS	Program Memory	Data Memory	Per-Channel Data Memory
G.711A1A2	Encode	7.33	22700	2886	1636
	Decode	2.5			1776

ARM CORTEX-A8/A9/A15

CPU UTILIZATION & MEMORY REQUIREMENTS

All Memory usage is given in units of byte.

Product	Function	MIPS	Program Memory	Data Memory	Per-Channel Data Memory	Scratch Memory
G.711A1A2	Encode	7.9	92K	5.26K	1640	624
	Decode	2.3			1776	624

LINUX

i686 LINUX

CPU UTILIZATION & MEMORY REQUIREMENTS

All Memory usage is given in units of bytes.

Function	MIPS	Program Memory	Data Memory	Per Channel Data Memory	Scratch Memory
Encode	9.72	62k	5.3k	1640	624
Decode	1.26			1776	624

PC-WINDOWS

Windows DLL

CPU UTILIZATION & MEMORY REQUIREMENTS

All Memory usage is given in units of bytes.

Function	MIPS	Program Memory	Data Memory	Per Channel Data Memory	Scratch Memory
Encode	8.58	61k	5.2k	1640	624
Decode	2.64			1776	624

Windows LIB

CPU UTILIZATION & MEMORY REQUIREMENTS

All Memory usage is given in units of bytes.

Function	MIPS	Program Memory	Data Memory	Per Channel Data Memory	Scratch Memory
Encode	7.59	61k	5.2k	1640	624
Decode	2.64			1776	624

We specify MIPS (Millions of Instructions Per Second) as MCPS (Millions of Instruction Cycles Per Second). Unless otherwise specified, peak MIPS are indicated.

FUNCTIONS

API function call summary

Standard functions

```
G711_Encode(. . .)      Performs G.711 encode function

G711_Decode(. . .)      Performs G.711 decode function
```

Appendix 1 (concealment) functions

```
G711ErasureInit(. . .)  Initializes data structures for per channel packet loss
                        concealment

G711A1Decode(. . .)     Generates synthesized voice samples to cover missing
                        data.
```

Appendix 2 (voice activity/comfort noise) functions

```
Vad_ADT_Init(. . .)     Initializes data structures for voice activity detection

Cng_ADT_Init(. . .)     Initializes data structures for comfort noise generation

Vad_ADT_Run(. . .)      Detects voice or silence and returns comfort noise
                        parameters during silence periods

Cng_ADT_Run(. . .)      Generate comfort noise based upon received parameters
```

API function call summary cont'd

Integrated functions (appendix 1 and 2)

```
G711_ADT_initEncode (. . .)   Initializes G.711 Encode Channel

G711_ADT_initDecode (. . .)   Initializes G.711 Decode Channel

G711_ADT_encode (. . .)       Perform G.711 encode

G711_ADT_decode (. . .)       Perform G.711 decode
```

Deliverables

The deliverable items are platform dependent. In general, there is one library. (Sometimes multiple variants of the library are included in the deliverables.) There are also header files, some of which are specific to the product and others are common across many of Adaptive Digital's products. Also included in the deliverables is product documentation, which includes a users guide and usually includes release notes and a data sheet. Sample/test code may be included as well.

Adaptive Digital is a member of the Texas Instruments Developer Network, and ARM Connected Community.

CONTACT INFORMATION

Web: www.adaptivedigital.com
Email: information@adaptivedigital.com
Tel: 610.825.0182 ~ Toll Free: 1.800.340.2066
Fax: 610.825.7616
Address: 525 Plymouth Road, Suite 316
Plymouth Meeting, PA 19462