Bad frame masking and comfort noise generator (for G.711) for TMS320C6201† Overview Guide

www.radisys.com

World Headquarters
5445 NE Dawson Creek Drive • Hillsboro, OR
97124 USA
Phone: 503-615-1100 • Fax: 503-615-1121
Toll-Free: 800-950-0044

International Headquarters
Gebouw Flevopoort • Televisieweg 1A
NL-1322 AC • Almere, The Netherlands
Phone: 31 36 5365595 • Fax: 31 36 5365620

007-01201-0001
November 2000

November 2000

Bad frame masking and comfort noise generator for TMS320C6201

The Bad Frame Masking (BFM) and Comfort Noise Generator (CNG) algorithm is designed to operate with speech communication system using G.711 coder. The Bad Frame Masking and Comfort Noise Generator are optimally implemented using Texas Instruments TMS3206201 Digital Signal Processor.

In packetized voice communication, when a network is congested, packets are held in queues of switching nodes, causing delays in delivery of packets. Long delays can result in unacceptable quality of service. A widely-used method for relieving network congestion is dropping packets. However this leads to gaps in the received voice signal, i.e. packet missing. In Fig. 1, for example, speech packets in buffer memory in System B are not available due to network congestion so that a bad speech frame occurs at the decoder of the coder of System B.

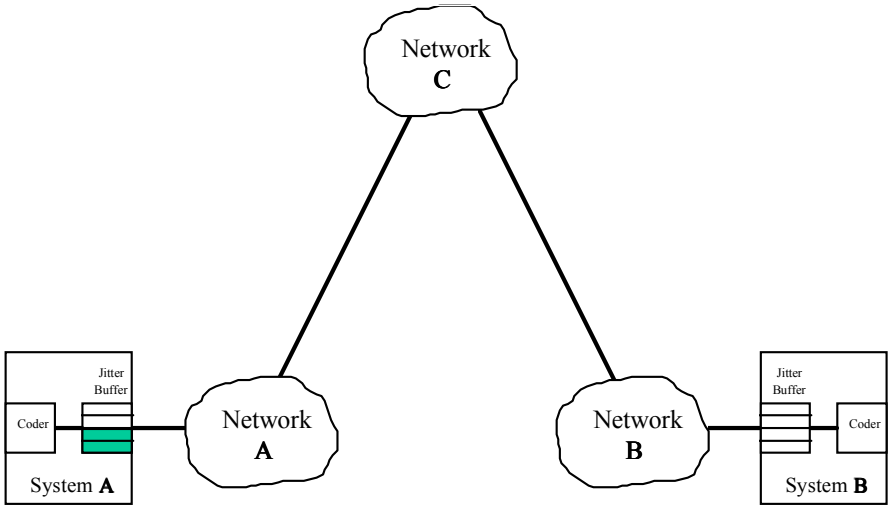

Figure 1. Speech packet missing in System B

The BFM module operates on frames of 80 linear 16-bit PCM samples (10 ms at 8000 Hz sampling rate) and a 10 ms lookahead (80 samples) is used. Note that the Bad Frame Masking module has no mechanism for detecting bad frames and the system must notify frame losses with a Bad Frame Indicator (*bfi_in*). The values of *bfi_in* are defined in the bfmasking.h header file. GOOD_FRAME (0) is defined as the flag for good frames and BAD_FRAME (1) as the flag for bad frames.

The main objective of the Comfort Noise Generator (CNG) engine is to generate comfort noise at the decoder of G.711 when no packets are received or when the received VAD (Voice Activity Detector) field indicates silence or background noise frames. The VAD decision and the G.711 encoder packet generated at the transmitter form the inputs to the Comfort Noise Generator which is invoked at the

receiver in conjunction with the G.711 decoder. The possible VAD outputs can be VOICE (1), PERIODIC NOISE UPDATE (2), and SILENCE (0).

For information about installing and using the Bad Frame Masking and Comfort Noise Generator software, see the topics listed below.

Installation

To make the Bad Frame Masking and Comfort Noise Generator software fully functional, follow these steps:

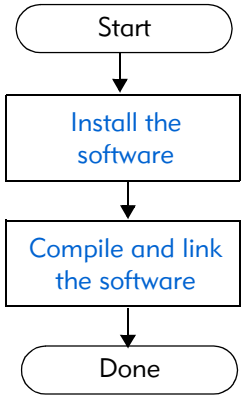

Figure 2. Installing and building the Bad Frame Masking and Comfort Noise Generator software

Requirements

Before you can install this software, you need the following:

- A Microsoft† Windows† 95, Windows 98, or Windows NT† compatible PC platform.
- The following software installed on your computer:
  - TMS320C6x Code Generation Tools Release 3.01 for Windows 95, Windows 98, or Windows NT

    —or—

    Code Composer Studio Tools, version 1.2, for Windows 95, Windows 98, or Windows NT.
  - NMAKE utility (available from Microsoft).
  - PKUNZIP utility.

Install the software

To install the Bad Frame Masking and Comfort Noise Generator software:

1. Create a directory on the target system. For example:

   ```
   mkdir c:\g711_bfmcng
   ```

2. Copy the product files from the CD-ROM to the directory you just created.

   

   Use a copy utility that retains the directory structure, such as Xcopy.

Compile and link the software

To compile and link the Bad Frame Masking and Comfort Noise Generator software:

1. Open an MS DOS command prompt window and go to the target directory, then to the \c6xsim sub-directory.
2. If you've already installed the TI tools, you can build the c6x version of the Bad Frame Masking and Comfort Noise Generator by entering this command at the MS DOS command prompt:

   ```
   NMAKE -f G711BFMCNG.mak
   ```

Code organization

When installed, the Bad Frame Masking and Comfort Noise Generator software files reside in this directory structure:

Directory	Extension	File type	Description
\bfmcng\bfm	.h	Header files	Header files used to build the Bad Frame Masking and Comfort Noise Generator. Applications can share these files as well. The directory includes these files: bfmasking.h bfmbasic_op.h bfmtab.h bfmtypedef.h
\bfmcng\c6xsim		Object code and build files. Source code for test applications.	
	.c	C files	g711bfmcng.c, the control source file.
	.cmd	Linker command files	Linker command files control object modules, linking options, and memory maps. These files include: • bfm.lib: Library of all BFM/CNG object modules • g711bfmcng.cmd: Linker command file
	.mak	Build file	The g711bfmcng.mak file[1] controls the build process.
\bfmcng\doc	.pdf	Manual	Includes this guide, in PDF format.
\bfmcng\input			Test vectors.
\bfmcng\results			Fixed point C-simulation test results.

[1] This file is tested to work with the Microsoft NMAKE utility, and can be modified to work with other make utilities.

Application programmer interface

The functional prototypes for the BFM/CNG modules are as follows.

Once you've installed and built the Bad Frame Masking and Comfort Noise Generator software, you can use these functions in your C application code.

The Bad Frame Masking and Comfort Noise Generator functions accept buffers passed to the module (for example, no stack variables, and no direct access of globals).

Interface

The next figure shows BFM/CNG module usage for the G.711 coder.

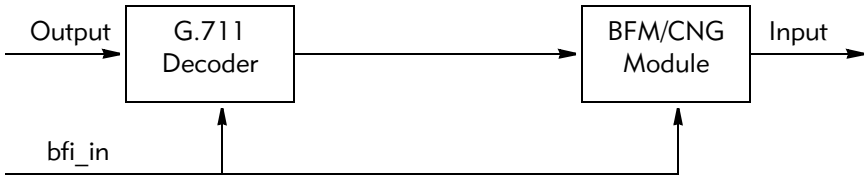

Figure 3. Usage of the BFM/CNG module for G.711 coder

The BFM/CNG module can support VAD (Voice Activity Detector) function. In case of silence, the BFM/CNG module generates comfort noise at appropriate levels. For burst packet loss, comfort noise is generated up to 400ms (this is to make sure that the lost packets are not updates from the VAD) and after that the most recent frame is taken as the reconstructed frame with an amplitude attenuation of 18 dB.

Init_BFMasking

For BFM/CNG tables/coefficient structure (*BfmTab*) initialization and API executions, please see the demo program (g711bfmcng.c), included in the CDROM. In the demo program the Tables Structure BfmTab is initialized with bfmtab.h in g711_BfmCng\bfm directory. The BFM/CNG module should be initialized before the G.711 decoding process by calling the following function. This initializes all the state variables/buffers (BfmBufStruct) used in the BFM/CNG module.

Syntax

```
void Init_BFMasking (
   BFMBUF *BfmBufStruct/* input/output : Buffer*/
          /*  2084 bytes, 32-bit aligned*/
);
```

Parameters

BfmBufStruct

A pointer to data buffer structure BfmBuf. The data buffer structure BfmBuf is defined in the bfmasking.h header file

BFMasking

The BFM/CNG algorithm is executed by calling the following function. This uses the *bfi_in* and *out_buf* as input from application and generates the output *out_buf*.

Syntax

```
void BFMasking (
   World16 bfi_in,/* input : Bad Frame Indicator*/
          /*  0 : Good Frame;  1: Bad Frame*/
```

```
    World16 out_buf[],/* input/output : Speech Signal*/
           /*  160 bytes*/
    BFMTBL *BfmTab,/* input : BFM table*/
           /*  882 bytes*/
    BFMBUF *BfmBufStruct/* input/output : Buffer*/
           /*  2084 bytes*/
);
```

Parameters

The parameters for the API is defined as follows. All the type definitions are defined in bfmtypedef.h in g711_BfmCng\bfm directory. For multiple channels multiple instances of BfmBufStruct have to be allocated on a per-channel basis. The same set *BfmTab* coefficients can be used for all the channels.

bfi_in Bad Frame Indicator from the system. The values of *bfi_in* are defined in the bfmasking.h header file in g711_BfmCng\bfm directory.

out_buf[] Input/Output speech vector (80 linear 16-bit PCM samples), 16-bit aligned.

BfmTab Pointer to the initialized coefficient table defined in the bfmtab.h header file in g711_BfmCng\bfm directory, with Pragma Data alignment of 8.

BfmBufStruct
Pointer to the data buffer structure BfmBuf. The data buffer structure BfmBuf is defined in the bfmasking.h header file in g711_BfmCng\bfm directory, with Pragma Data alignment of 8.

Pseudo C code for API calling sequence

```
Init_BFMasking(pBfmBuf);

While(1) {
       bfi_in = GOOD_FRAME;
       if(rtpStatus == -1) {  /* System packet (RTP) return status */
          bfi_in = BAD_FRAME;
       }
if(bfi_in  == GOOD_FRAME)
       {
          g711Decode(in_buf,out_buf,80,MULAW,LawTbl);
       }
       BFMasking(bfi_in, out_buf, BfmTbl, BfmBuf);
   }
```

Testing and performance specifications

Tests on the BFM/CNG fixed-point C simulation

The Radisys fixed-point C simulation of the BFM/CNG algorithm was tested with different speech sequences. The speech material was two 7-second sentences spoken by a male speaker and a female speaker, respectively. In the test, packets were discarded randomly and there were six missing packet rates: 1 percent, 2 percent, 3 percent, 4 percent, 5 percent, and 10 percent. The results should satisfy the following requirements:

- The Bad Frame Masking module should provide better speech quality than that of “silence substitution” technique in which bad frames are replaced by silence. Satisfactory speech quality (i.e.: with slight noise or speech clipping) should be obtained with random frame loss rates of up to 5 percent.
- The comfort noise should be perceptually equivalent to the background noise in the voice encoder. The transitions between active voice and inactive voice should be imperceptible.

Input speech files for the test are in the g711_BfmCng\input directory and the corresponding output files are in the g711_BfmCng\results directory.

Tests on the BFM/CNG c6x implementation

The c6X implementation of the BFM/CNG algorithm should produce bit-exact results compared with the RadiSys fixed-point C simulation code. This test can be run using Texas Instruments EVM with Code Composer Studio Tools, version 1.2, to load g711bfmcng.out file from g711_BfmCng\c6xsim directory. Make sure that proper path is set to the input/output files in g711_BfmCng\c6xsim\ g711bfmcng.c file and use this command from g711_BfmCng\c6xsim to generate g711bfmcng.out file.

```
nmake -f g711bfmcng.mak
```

Bit exactness between the C and c6x binary output files can be verified by using this command in a DOS window:

```
fc/b
```

BFM/CNG c6x implementation test results

The tests were conducted on code in g711_BfmCng\c6xsim. The input files for the test were TestM.inp, TestF.inp, TestM.vad, and TestF.vad from g711_BfmCng\input directory. It was confirmed that the c6x implementation was a bit-exact match of the Radisys fixed-point C simulation code. The Bad Frame Masking module has reduced the degradation in speech quality in the presence of packet loss. Both the speech reconstructed by the BFM and the comfort noise generated by the CNG have met the above-mentioned requirements.

MCPS measurements for the BFM/CNG module

The current implementation of the BFM/CNG module takes 10500 instruction cycles per frame (peak value). This figure of 10500 instruction cycles per (10.0 msec.) frame translates to about 1.05 MCPS (million cycles per second) per channel.

Memory measurements for the BFM/CNG module

The following table contains memory measurements for the current c6x implementation of the BFM/CNG module. The BFM/CNG module also needs a 512 bytes of stack memory for local variables.

Table 1. Memory requirements for the BFM/CNG module

Program memory (Kbytes)	Data memory (Kbytes)	Tables (Kbytes)
25.024	2.084	0.882

Where to get more information

About the Bad frame masking and comfort noise generator for TMS320C6201

You can find out more about the Bad Frame Masking and Comfort Noise Generator from these sources:

- **Readme file:** Lists features and issues that arose too late to include in other documentation.
- **World Wide Web:** RadiSys maintains an active site on the World Wide Web. The site contains current information about the company and locations of sales offices, new and existing products, contacts for sales, service, and technical support information. You can also send e-mail to RadiSys using the web site.

When sending e-mail for technical support, please include information about both the hardware and software, plus a detailed description of the problem, including how to reproduce it.

To access the RadiSys web site, enter this URL in your web browser:
`http://www.radisys.com`

Requests for sales, service, and technical support information receive prompt response.

About related products

SP6040

The RadiSys SP6040 (SPIRIT™-6040 CompactPCI† board) is a high-performance intelligent I/O subsystem designed for telecom and datacom applications. Based on Texas Instruments† devices, the SP6040 has a 200MHz DSP engine with an I/O connector that provides interface-to-digital network interfaces (DNI) such as E1/T1 and ATM. The SP6040 contains up to four TMS320C6201 digital signal processors.

SP6020

The RadiSys SP6020 (SPIRIT-6020 board) features two Texas Instruments TMS320C6201B fixed-point DSPs that run at 200MHz and serve as the main processing engines. Each ’C6x can deliver up to 1600 MIPS of processing power,

contingent on the available parallelism within the application code. The DSPs are used for voice and data processing, compression, and decompression.

TASK-6000

TASK-6000 software is a tool set that provides a framework for composing, executing and dynamically configuring optimized real-time TMS320C62x DSP applications. The Voice Gateway Platform (VGP), available from RadiSys, is used as the reference hardware platform.

For detailed information about TASK-6000 products, see the following publications:

TASK-6000 User's Manual, (07-0992-xx)

TASK-6000 Installation Guide, (07-1000-xx)

TI tools

Software tools from Texas Instruments used to build DSP executables.

For more information about Texas Instruments products, enter this URL in your web browser:
`http://www.ti.com/dsp`