

# **$\mu$ SAP77016-B05**

## **MPEG-4 CELP Speech Codec Middleware**

---

### **Target Devices**

**$\mu$ PD77110**

**$\mu$ PD77111**

**$\mu$ PD77112**

**$\mu$ PD77113**

**$\mu$ PD77114**

**$\mu$ PD77115**

**[MEMO]**

#### Patent for MPEG-4 CELP

A number of patents exist for systems related to MPEG-4 CELP.

NEC would request that customers ensure they comply with the relevant rights for these patents. NEC does not accept any responsibility for infringement of any patent rights by the customer.

**MS-DOS and Windows are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.**

- **The information in this document is current as of October, 2000. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC's data sheets or data books, etc., for the most up-to-date specifications of NEC semiconductor products. Not all products and/or types are available in every country. Please check with an NEC sales representative for availability and additional information.**

- No part of this document may be copied or reproduced in any form or by any means without prior written consent of NEC. NEC assumes no responsibility for any errors that may appear in this document.

- NEC does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from the use of NEC semiconductor products listed in this document or any other liability arising from the use of such products. No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC or others.

- Descriptions of circuits, software and other related information in this document are provided for illustrative purposes in semiconductor product operation and application examples. The incorporation of these circuits, software and information in the design of customer's equipment shall be done under the full responsibility of customer. NEC assumes no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.

- While NEC endeavours to enhance the quality, reliability and safety of NEC semiconductor products, customers agree and acknowledge that the possibility of defects thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects in NEC semiconductor products, customers must incorporate sufficient safety measures in their design, such as redundancy, fire-containment, and anti-failure features.

- NEC semiconductor products are classified into the following three quality grades:

"Standard", "Special" and "Specific". The "Specific" quality grade applies only to semiconductor products developed based on a customer-designated "quality assurance program" for a specific application. The recommended applications of a semiconductor product depend on its quality grade, as indicated below. Customers must check the quality grade of each semiconductor product before using it in a particular application.

"Standard": Computers, office equipment, communications equipment, test and measurement equipment, audio and visual equipment, home electronic appliances, machine tools, personal electronic equipment and industrial robots

"Special": Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-disaster systems, anti-crime systems, safety equipment and medical equipment (not specifically designed for life support)

"Specific": Aircraft, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems and medical equipment for life support, etc.

The quality grade of NEC semiconductor products is "Standard" unless otherwise expressly specified in NEC's data sheets or data books, etc. If customers wish to use NEC semiconductor products in applications not intended by NEC, they must contact an NEC sales representative in advance to determine NEC's willingness to support a given application.

(Note)

(1) "NEC" as used in this statement means NEC Corporation and also includes its majority-owned subsidiaries.

(2) "NEC semiconductor products" means any semiconductor product developed or manufactured by or for NEC (as defined above).

M8E 00.4

# Regional Information

Some information contained in this document may vary from country to country. Before using any NEC product in your application, please contact the NEC office in your country to obtain a list of authorized representatives and distributors. They will verify:

- Device availability
- Ordering information
- Product release schedule
- Availability of related technical literature
- Development environment specifications (for example, specifications for third-party tools and components, host computers, power plugs, AC supply voltages, and so forth)
- Network requirements

In addition, trademarks, registered trademarks, export restrictions, and other legal issues may also vary from country to country.

## **NEC Electronics Inc. (U.S.)**

Santa Clara, California  
Tel: 408-588-6000  
800-366-9782  
Fax: 408-588-6130  
800-729-9288

## **NEC Electronics (Germany) GmbH**

Duesseldorf, Germany  
Tel: 0211-65 03 02  
Fax: 0211-65 03 490

## **NEC Electronics (UK) Ltd.**

Milton Keynes, UK  
Tel: 01908-691-133  
Fax: 01908-670-290

## **NEC Electronics Italiana s.r.l.**

Milano, Italy  
Tel: 02-66 75 41  
Fax: 02-66 75 42 99

## **NEC Electronics (Germany) GmbH**

Benelux Office  
Eindhoven, The Netherlands  
Tel: 040-2445845  
Fax: 040-2444580

## **NEC Electronics (France) S.A.**

Velizy-Villacoublay, France  
Tel: 01-30-67 58 00  
Fax: 01-30-67 58 99

## **NEC Electronics (France) S.A.**

Madrid Office  
Madrid, Spain  
Tel: 91-504-2787  
Fax: 91-504-2860

## **NEC Electronics (Germany) GmbH**

Scandinavia Office  
Taeby, Sweden  
Tel: 08-63 80 820  
Fax: 08-63 80 388

## **NEC Electronics Hong Kong Ltd.**

Hong Kong  
Tel: 2886-9318  
Fax: 2886-9022/9044

## **NEC Electronics Hong Kong Ltd.**

Seoul Branch  
Seoul, Korea  
Tel: 02-528-0303  
Fax: 02-528-4411

## **NEC Electronics Singapore Pte. Ltd.**

United Square, Singapore  
Tel: 65-253-8311  
Fax: 65-250-3583

## **NEC Electronics Taiwan Ltd.**

Taipei, Taiwan  
Tel: 02-2719-2377  
Fax: 02-2719-5951

## **NEC do Brasil S.A.**

Electron Devices Division  
Guarulhos-SP Brasil  
Tel: 55-11-6462-6810  
Fax: 55-11-6462-6829

J00.7

### Major Revisions in This Edition

Page	Description
Throughout	Addition of $\mu$ PD77111, 77112, and 77115.
p.11	Modification of description in <b>1.2 MPEG-4 CELP Speech Codec</b>
p.12	Modification of <b>1.3 Product Overview</b>
p.19	Modification of Return value, Function, and Hardware resources in <b>2.3.2 mpc_InitDec function</b>
p.20	Modification of Hardware resources in <b>2.3.3 mpc_Enc function</b>
pp.22, 23	Modification of Function and Hardware resources in <b>2.3.4 mpc_Dec function</b>
p.25	Addition of description to <b>2.4 Compressed Data Format</b>
p.34	Modification of <b>APPENDIX SAMPLE PROGRAM SOURCE</b>

The mark ★ shows major revised points.

## PREFACE

<b>Target Readers</b>	<p>This manual is intended for users who wish to design and develop application systems using the <math>\mu</math>PD77016 Family.</p> <p>The <math>\mu</math>PD77016 Family includes the <math>\mu</math>PD77015, 77016, 77017, 77018, 77019, 77110, 77111, 77112, 77113, 77114, and 77115. This manual, however, only covers the <math>\mu</math>PD77110, 77111, 77112, 77113, 77114, and 77115.</p>												
<b>Purpose</b>	<p>The purpose of this manual is for users to gain an understanding of the middleware used for support when designing and developing application systems using the <math>\mu</math>PD77016 Family.</p>												
<b>Organization</b>	<p>This manual consists of the following.</p> <ul style="list-style-type: none"><li><b>CHAPTER 1 INTRODUCTION</b></li><li><b>CHAPTER 2 LIBRARY SPECIFICATIONS</b></li><li><b>CHAPTER 3 INSTALLATION</b></li><li><b>CHAPTER 4 SYSTEM EXAMPLE</b></li><li><b>APPENDIX SAMPLE PROGRAM SOURCE</b></li></ul>												
<b>How to Read This Manual</b>	<p>It is assumed that the reader of this manual has general knowledge in the fields of electrical engineering, logic circuits, and microcontrollers.</p>												
<b>Conventions</b>	<table><tr><td>Data significance:</td><td>Higher digits on the left and lower digits on the right</td></tr><tr><td><b>Note:</b></td><td>Footnote for item marked with <b>Note</b> in the text</td></tr><tr><td><b>Caution:</b></td><td>Information requiring particular attention</td></tr><tr><td><b>Remark:</b></td><td>Supplementary information</td></tr><tr><td rowspan="3">Number representation:</td><td>Binary xxxx or 0bxxxx</td></tr><tr><td>Decimal xxxx</td></tr><tr><td>Hexadecimal 0xxxxx</td></tr></table>	Data significance:	Higher digits on the left and lower digits on the right	<b>Note:</b>	Footnote for item marked with <b>Note</b> in the text	<b>Caution:</b>	Information requiring particular attention	<b>Remark:</b>	Supplementary information	Number representation:	Binary xxxx or 0bxxxx	Decimal xxxx	Hexadecimal 0xxxxx
Data significance:	Higher digits on the left and lower digits on the right												
<b>Note:</b>	Footnote for item marked with <b>Note</b> in the text												
<b>Caution:</b>	Information requiring particular attention												
<b>Remark:</b>	Supplementary information												
Number representation:	Binary xxxx or 0bxxxx												
	Decimal xxxx												
	Hexadecimal 0xxxxx												

**Related Documents**

The related documents indicated in this publication may include preliminary versions. However, preliminary versions are not marked as such.

**Documents Related to Devices**

Document Name Part Number	Pamphlet	Data Sheet	User's Manual		Application Note
			Architecture	Instructions	Basic Software
μ PD77110	U12395E	U12801E	U14623E	U13116E	U11958E
μ PD77111					
μ PD77112					
μ PD77113		U14373E			
μ PD77114					
μ PD77115	–	U14867E	–	–	–

**Documents Related to Development Tools**

Document Name		Document No.
IE-77016-98, IE-77016-PC User's Manual	Hardware	U13044E
IE-77016-CM-LC User's Manual		U14139E
RX77016 User's Manual	Function	U14397E
	Configuration Tool	U14404E
RX77016 Application Note	HOST API	U14371E

**Caution** The related documents listed above are subject to change without notice. Be sure to use the latest version of each document for designing.

# CONTENTS

<b>CHAPTER 1 INTRODUCTION</b> .....	<b>11</b>
<b>1.1 Middleware</b> .....	<b>11</b>
<b>1.2 MPEG-4 CELP Speech Codec</b> .....	<b>11</b>
<b>1.3 Product Overview</b> .....	<b>12</b>
1.3.1 Features .....	12
1.3.2 Operating environment.....	13
1.3.3 Performance.....	13
1.3.4 Directory structure .....	14
<b>CHAPTER 2 LIBRARY SPECIFICATIONS</b> .....	<b>15</b>
<b>2.1 Application Processing Flow</b> .....	<b>15</b>
<b>2.2 Timing Diagrams</b> .....	<b>17</b>
<b>2.3 Function Specifications</b> .....	<b>18</b>
2.3.1 mpc_InitEnc function .....	18
2.3.2 mpc_InitDec function.....	19
2.3.3 mpc_Enc function .....	20
2.3.4 mpc_Dec function.....	22
2.3.5 mpc_GetVersion function .....	24
<b>2.4 Compressed Data Format</b> .....	<b>25</b>
2.4.1 Header .....	25
2.4.2 Frame.....	27
<b>2.5 Relationship Between Bit Rate and Number of Samples</b> .....	<b>28</b>
<b>CHAPTER 3 INSTALLATION</b> .....	<b>30</b>
<b>3.1 Installation Procedure</b> .....	<b>30</b>
<b>3.2 Sample Creation Procedure</b> .....	<b>30</b>
<b>3.3 Symbol Naming Regulations</b> .....	<b>30</b>
<b>CHAPTER 4 SYSTEM EXAMPLE</b> .....	<b>31</b>
<b>4.1 Simulation Environment in Which Timing File Used</b> .....	<b>31</b>
<b>4.2 Operation</b> .....	<b>31</b>
<b>APPENDIX SAMPLE PROGRAM SOURCE</b> .....	<b>34</b>



## LIST OF FIGURES

Figure No.	Title	Page
2-1	Application Processing Flow (Encoder).....	15
2-2	Application Processing Flow (Decoder) .....	16
2-3	Encoder Timing Diagram .....	17
2-4	Decoder Timing Diagram .....	17
2-5	Header Format .....	25
2-6	Frame Configuration .....	27

## LIST OF TABLES

Table No.	Title	Page
1-1	Supported Functions .....	12
1-2	Required Memory Size .....	13
1-3	MIPS Values Required for Compression/Decompression Processing .....	13
2-1	Relationship Between Narrow Band Bit Rate and Number of Samples.....	28
2-2	Relationship Between Wide Band Bit Rate and Number of Samples .....	29

## CHAPTER 1 INTRODUCTION

### 1.1 Middleware

Middleware is the name given to a group of software that has been tuned so that it draws out the maximum performance of the processor and enables processing that is conventionally performed by hardware to be performed by software. The concept of middleware was introduced with the development of a new high-speed processor, the DSP, in order to facilitate operation of the environments integrated in the system.

By providing appropriate speech codec and image data compression/decompression-type middleware, NEC is offering users the kind of technology essential in the realization of a multimedia system for the  $\mu$ PD77016 Family, and is continuing its promotion of system development.

The  $\mu$ SAP77016-B05 is middleware that supplies the functions of speech compression/decompression.

### ★ 1.2 MPEG-4 CELP Speech Codec

The MPEG-4 CELP<sup>Note 1</sup> speech codec is a CELP speech compression/decompression function standardized as ISO/IEC<sup>Note 2</sup> 14496-3 (MPEG-4 version 1) Audio Part. In addition, the error robustness and silence compression functions are expanded by ISO/IEC 14496-3/Amd1 (MPEG-4 version 2) Audio Part. This MPEG-4 CELP speech codec employs an algorithm proposed by NEC.

The speech input/output data processed by the MPEG-4 CELP speech codec is 16-bit linear PCM data resulting from sampling an analog input signal at 8 or 16 MHz. If the signal is sampled at 8 kHz (narrow band: NB), 28 types of bit rates, 3.85 to 12.2 Kbps, are supported. If the signal is sampled at 16 kHz (wide band: WB), 30 types of bit rates, 10.9 to 23.8 Kbps, are supported (refer to 2.5 Relationship Between Bit Rate and Number of Samples).

- Notes**
1. CELP: Code Excited Linear Prediction
  2. ISO: International Organization for Standardization  
IEC: International Electrotechnical Commission

★ 1.3 Product Overview

1.3.1 Features

- (1) Speech input/output data: 16-bit linear PCM data
- (2) Coding/decoding 80 to 320 samples/frame at sampling frequency of 8 or 16 kHz
- (3) 28 types of bit rates (3.85 to 12.2 Kbps) at sampling frequency of 8 kHz or 30 types of bit rates (10.9 to 23.8 Kbps) at 16 kHz
- (4) High-quality speech coding
  - Wide band (16 kHz sampling)
  - Quality equivalent to ITU-T G.729<sup>Note</sup> at 8 Kbps and sampling frequency of 8 kHz

**Note** International Telecommunication Union – Telecommunication Standardization Sector

Table 1-1 lists the supported functions.

**Table 1-1. Supported Functions**

	Function	Encoder	Decoder
Version 1	Basic coding function (base layer)	√	√
	Bit rate hierarchy function (Bit Rate Scalable)	×	×
	Band hierarchy function (Band Width Scalable)	×	×
	Multi-pulse sound source (Multi Pulse Excitation)	√	√
	Regular pulse sound source (Regular Pulse Excitation)	×	×
	Fine rate control	×	×
	Version 1 compressed data format	√	√
Version 2	Error concealment <sup>Note</sup> : Bit error	–	√
	Error concealment <sup>Note</sup> : Frame erasure	–	√
	Silence compression	×	√
	Version 2 compressed data format	×	√

**Note** Error concealment is a function of version 2, but this middleware allows it to be used with data of version 1.

**Caution** “Version” in the above figure indicates the version of MPEG-4, not the version of the middleware.

**Remark** √: Supported, ×: Not supported, –: Not subject to support

### 1.3.2 Operating environment

#### (1) Target DSP:

$\mu$ PD77110, 77111<sup>Note</sup>, 77112<sup>Note</sup>, 77113, 77114, 77115<sup>Note</sup>

**Note** Applicable when these processors operate only as decoders.

#### (2) Required memory size:

This middleware can be used as a codec, or separately as an encoder or decoder. Table 1-2 shows the required memory size.

**Table 1-2. Required Memory Size**

Memory	Type	Size (Words)	Encoder [Words]	Decoder [Words]
Instruction memory	–	14.6 K	8.0 K	8.0 K
X memory	RAM (Work)	Encoder: 3 K / Decoder: 1.5 K	3.0 K	1.5 K
	(Static)	3.2 K	2.7 K	0.6 K
	ROM	5.5 K	2.4 K	5.0 K
Y memory	RAM (Work)	Encoder: 4.7 K / Decoder: 0.7 K	4.7 K	0.7 K
	(Static)	2.5 K	2.5 K	0.1 K
	ROM	13.1 K	10.1 K	9.3 K

#### (3) Software tools (Windows™ version):

DSP tools

- WB77016 (workbench assembler)
- HSM77016 (high-speed simulator)
- IE77016 (debugger)

### 1.3.3 Performance

[Condition] DSP:  $\mu$ PD77016 Family (33 MIPS @ 33 MHz operation)

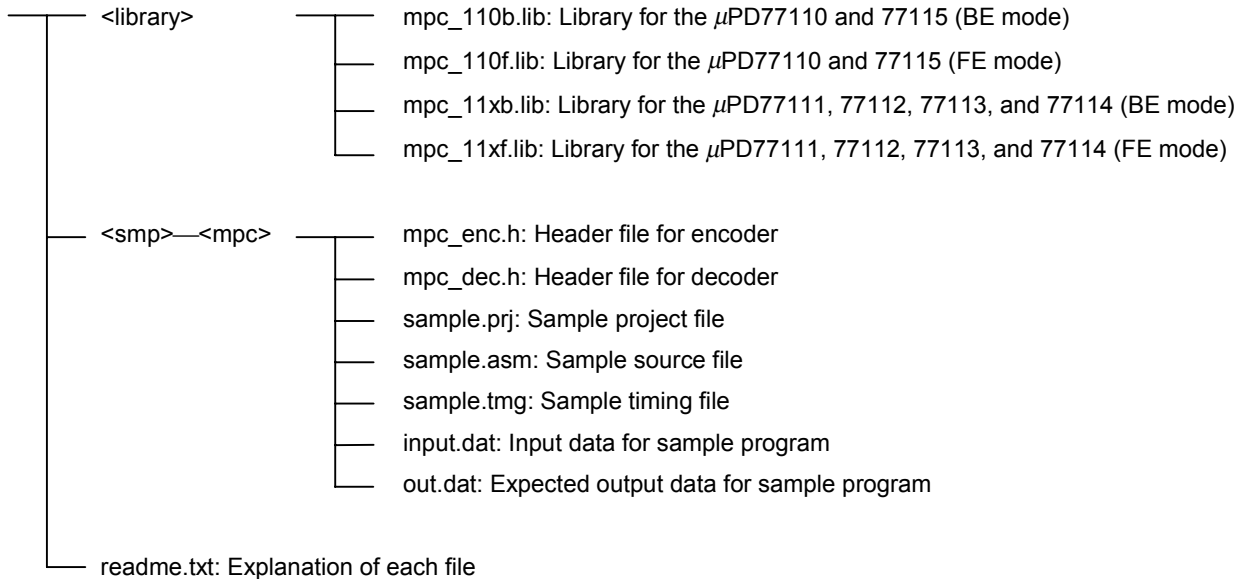
[The MIPS values required to execute the processing of 1 frame in real time]

**Table 1-3. MIPS Values Required for Compression/Decompression Processing**

Bit Rate [bps]	Compression Time [MIPS]	Decompression Time (Postprocessing Filter On) [MIPS]	Total (Postprocessing Filter On) [MIPS]
4650 (NB)	16.5	2.5	19.0
5500 (NB)	17.5	2.6	20.1
7300 (NB)	19.5	2.6	22.1
10700 (NB)	20.8	3.3	24.1
12200 (NB)	22.3	3.5	25.8
14300 (NB)	52.5	9.1	61.6
17000 (NB)	55.3	9.3	64.6
21100 (NB)	58.2	12.3	70.5
23800 (NB)	60.8	12.5	73.3

### 1.3.4 Directory structure

The contents of the packages are shown below.



Each directory is outlined below.

#### (1) library

Stores the library files.

Two types of libraries, for BE (Bit Error) mode and FE (Frame Erasure) mode, are available depending on the differences in the concealment processing of the decoder, and either of them is selected when the middleware is embedded into a system. They cannot be selected during reception.

- BE (Bit Error) mode

This mode is used if the system can receive error frame information.

The past frame data and part of the current frame data are used for decoding processing in this mode.

- FE (Frame Erasure) mode

This mode is used if the system can receive no error frame information.

In this mode, decoding processing is performed by using only the past frame data.

#### (2) <smp>--<mpc>

Stores the source files of the sample program, and the header files. A timing file, to be described later, is also available.

## CHAPTER 2 LIBRARY SPECIFICATIONS

MPEG-4 CELP provides the following 5 functions.

Function Name	Function
mpc_InitEnc	Encode processing initialization
mpc_InitDec	Decode processing initialization
mpc_Enc	Encode processing
mpc_Dec	Decode processing
mpc_GetVersion	Version information acquisition

### 2.1 Application Processing Flow

Examples of application processing using the MPEG-4 CELP speech codec are shown in Figures 2-1 and 2-2 below.

**Figure 2-1. Application Processing Flow (Encoder)**

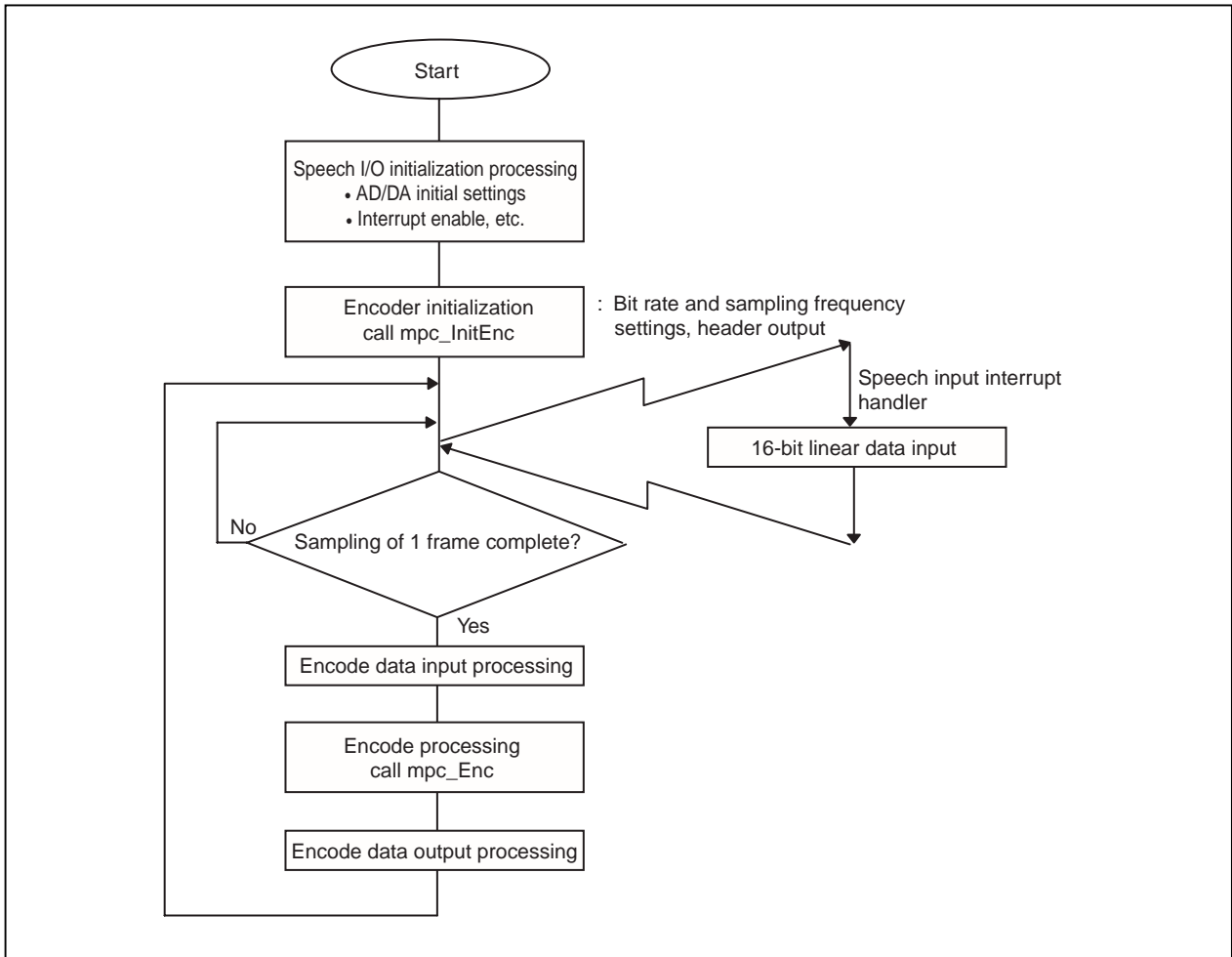
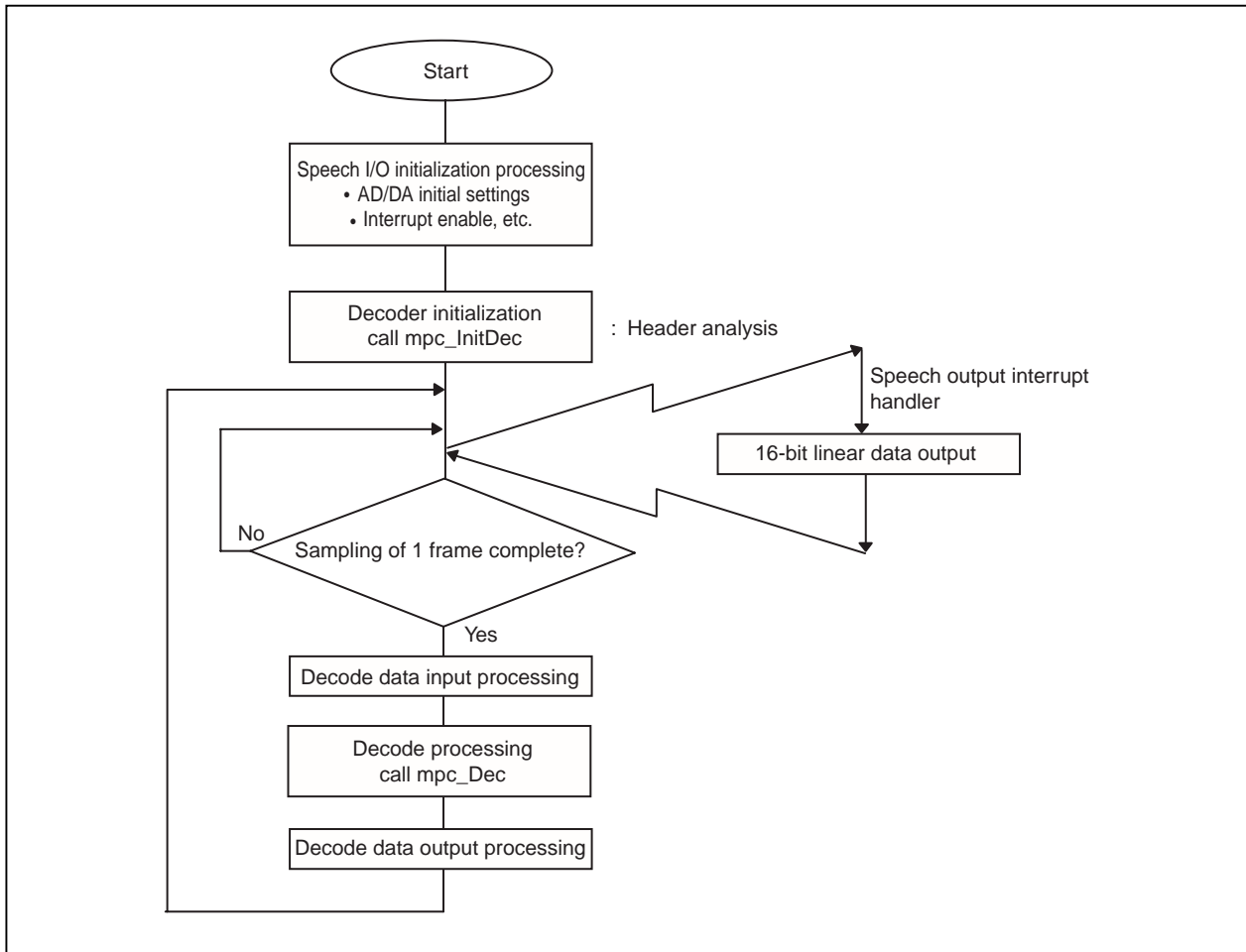


Figure 2-2. Application Processing Flow (Decoder)

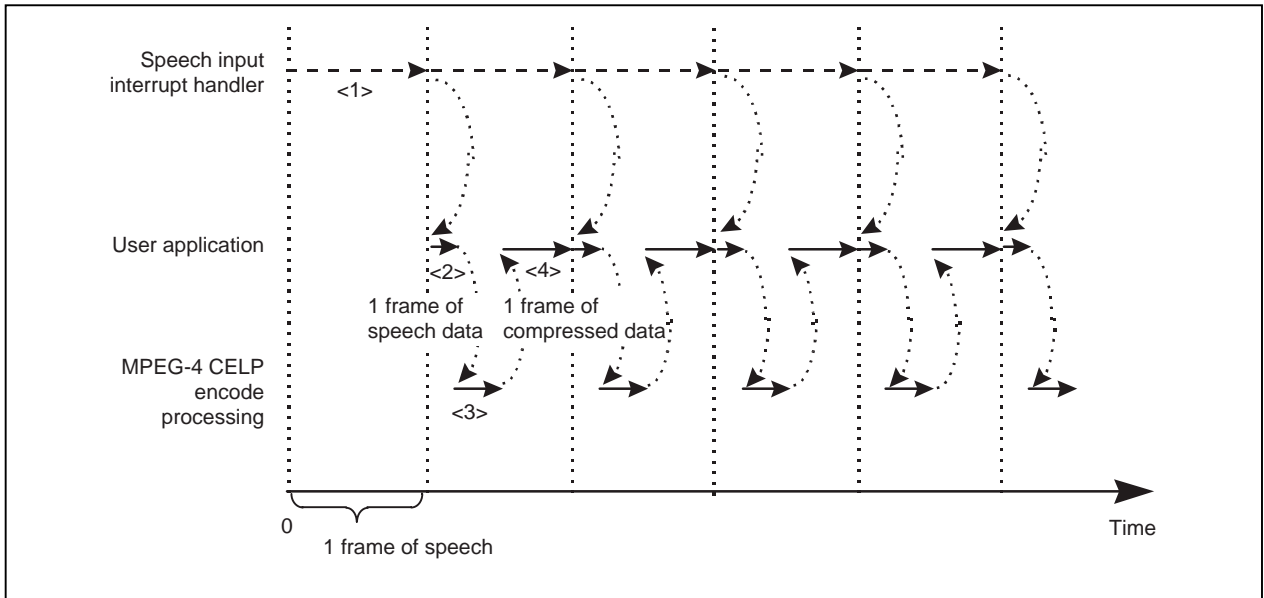


The speech data I/O processing is dependent on the target system's hardware, so make sure the design accords with the target system.



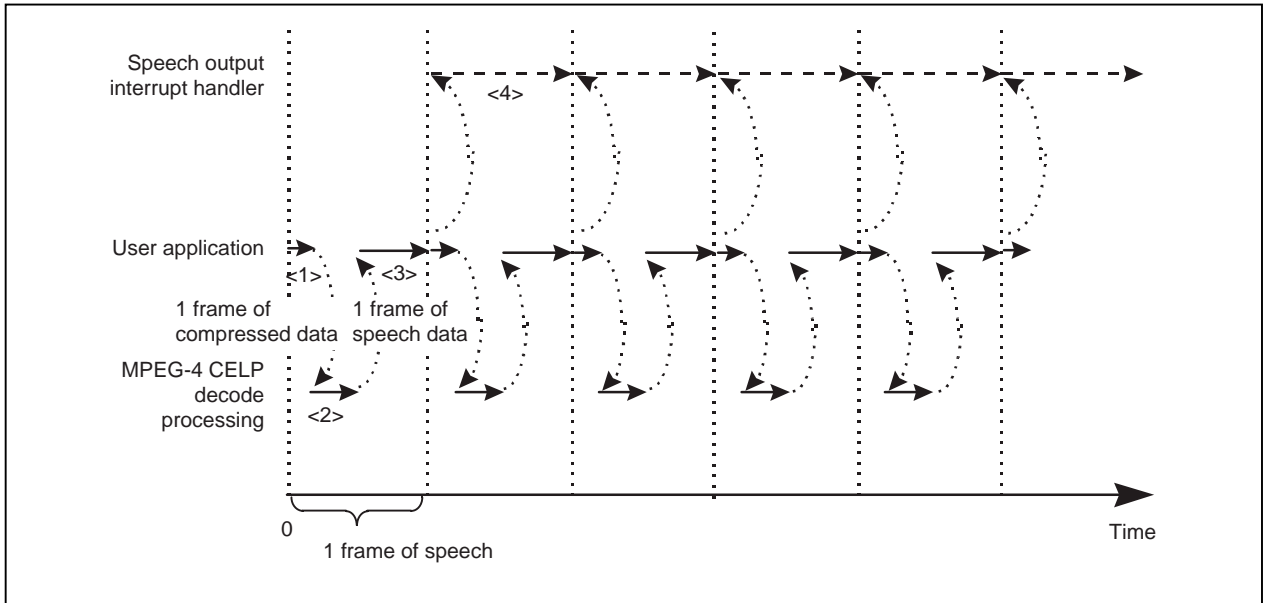
2.2 Timing Diagrams

Figure 2-3. Encoder Timing Diagram



- <1> Sampling frequency: 8 or 16 kHz, Precision: 16 bits; A/D converted to 1 frame of PCM data
- <2> Buffering via the user application
- <3> 1 frame of sample data is compressed
- <4> 1 frame of compressed data is saved. The user can use the rest of the time for processing another application.

Figure 2-4. Decoder Timing Diagram



- <1> 1 frame of compressed data is read and passed to decompression processing
- <2> 1 frame of compressed data is decompressed into 1 frame of sample data

<3> The decompressed speech data is buffered. The user can use the rest of the time for processing another application.

<4> Sampling frequency: 8 or 16 kHz, Precision: 16 bits; 1 frame of PCM data is D/A converted

## 2.3 Function Specifications

### 2.3.1 mpc\_InitEnc function

- **Classification** MPEG-4 CELP encoder initialization processing
- **Function name** mpc\_InitEnc
- **Summary of function** Makes the parameter settings and initializes the RAM area used by the MPEG-4 CELP encoder.
- **Format** call mpc\_InitEnc
- **Arguments**
  - \*mpc\_E\_ANA\_BUFF\_SADR:x Output data buffer start address (X memory)
  - \*mpc\_E\_RATE\_NUM:x Bit rate setting.  
Sets the 28 types/30 types of bit rates in the No. column in the tables in **2.5 Relationship Between Bit Rate and Number of Samples**.
  - \*mpc\_E\_WB:x Sets the MPE configuration of the header.  
Sets the sampling frequency. 0: NB/1: WB set in the sample rate mode of the header.
  - R6 Start address of the X memory work area
  - R5 Start address of the Y memory work area
- **Return value**
  - \*mpc\_E\_PCM\_NUM:x Number of samples per frame
  - \*mpc\_E\_ANA\_BITS:x Number of compressed data bits in 1 frame
- **Function**

Makes the parameter settings necessary for the MPEG-4 CELP encoder, initializes the calculation area, and outputs the header to the top of the buffer specified by mpc\_E\_ANA\_BUFF\_SADR. (11 bits with MSB first).  
The bit rate cannot be changed after the initialization processing.  
For details of the memory size required for the work area, refer to the value of the RAM (Work) in **1.3.2 (2) Required memory size**.

★ The encoded data can be output in version 1 only. It cannot be output in version 2.
- **Registers used** R0, R1, R2, R3, R4, R5, R6, R7, DP0, DP1, DP2, DP3, DP4, DP5, DN0, DN1
- **Hardware resources**

The hardware resources for this function are as follows.

Maximum stack level:	3
Maximum loop stack level:	1
Maximum number of repeats:	800
Maximum number of cycles:	2700

**Caution** Between when the `mpc_InitEnc` function was called and the end of compression processing, the RAM area must not be destroyed. If the RAM area is destroyed, operation is not guaranteed.

### 2.3.2 `mpc_InitDec` function

- **Classification** MPEG-4 CELP decoder initialization processing
- **Function name** `mpc_InitDec`
- **Summary of function** Makes the parameter settings and initializes the RAM area used by the MPEG-4 CELP decoder.
- **Format** call `mpc_InitDec`
- **Arguments**

<code>*mpc_D_ANA_BUFF_SADR:x</code>	Input data buffer start address (X memory)
<code>*mpc_D_PF_FLAG:x</code>	Post-processing filter flag. 0: OFF/1: ON
R6	Work area X memory start address
R5	Work area Y memory start address
- **Return value**

<code>*mpc_D_PCM_NUM:x</code>	Number of samples per frame
<code>*mpc_D_ANA_BITS:x</code>	Number of <code>CelpBaseFrame</code> bits among compressed data in 1 frame (this is the same as the number of compressed bits in 1 frame if silence compression is not used.)
- ★ **Function**

Analyses the compressed data by the header, sets the parameters required for the MPEG-4 CELP decoder, and initializes the calculation area. The header is input to the top of the buffer specified by `mpc_D_ANA_BUFF_SADR`. (11 bits with MSB first in version 1, and 12 bits with MSB first in version 2). Whether the speech data output is passed through a post-processing filter is determined by `mpc_D_PF_FLAG`. The post-processing filter is a filter configured on the output side of the decoder to improve the sound quality of the decoded speech signals. Refer to **2.4.1 Header** for the format of the header. For details of the memory size required for the work area, refer to the value of the RAM (Work) in **1.3.2 Operating environment**.
- **Registers used** R0, R1, R2, R3, R4, R5, R6, R7, DP0, DP1, DP2, DP4, DP5, DN0
- **Hardware resources**

The hardware resources for this function are as follows.

Maximum stack level:	3
Maximum loop stack level:	1
Maximum number of repeats:	50
Maximum number of cycles:	800
- ★

**Caution** Between when the `mpc_InitDec` function was called and the end of decompression processing, the RAM area must not be destroyed. If the RAM area is destroyed, operation is not guaranteed.

### 2.3.3 `mpc_Enc` function

- **Classification** MPEG-4 CELP encoder processing
- **Function name** `mpc_Enc`
- **Summary of function** Compresses 1 frame of specified speech data.
- **Format** call `mpc_Enc`
- **Arguments**
  - `*mpc_E_PCM_BUFF_SADR:x` Input data buffer start address (X memory)
  - `*mpc_E_ANA_BUFF_SADR:x` Output data buffer start address (X memory)
- **Return value** None
- **Function**

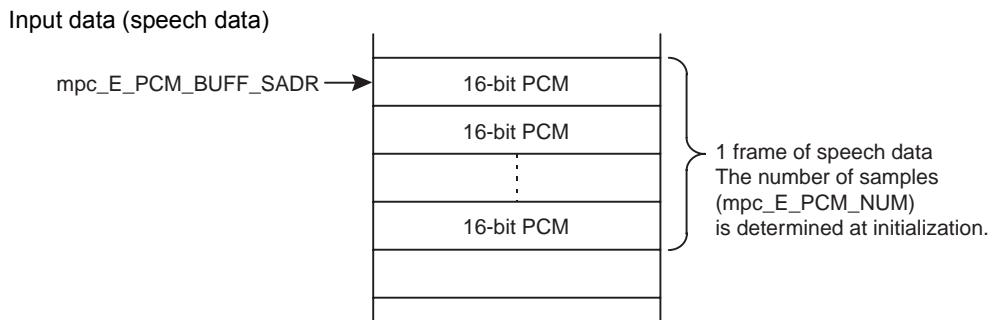
Performs compression processing for speech data specified by `mpc_E_PCM_BUFF_SADR` and writes and returns this data to the compressed data buffer specified by `mpc_E_ANA_BUFF_SADR`. If the final word of the compressed data frame is less than 1 word, this data is word-aligned by outputting it after filling the remainder with 0s.

The number of samples input in 1 frame is determined by the bit rate, and is assigned by `mpc_E_PCM_NUM`, which is the return value of the encoder initialization processing, `mpc_InitEnc`.

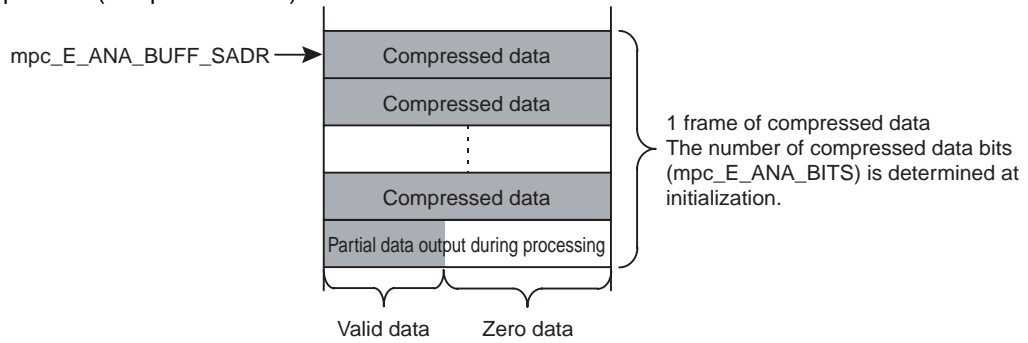
The size of the compressed data output is also determined by the bit rate.

Neither the bit rate nor NB/WB can be changed.

The compressed data is a stream of bit units.



Output data (compressed data)



- **Registers used**      R0, R1, R2, R3, R4, R5, R6, R7, DP0, DP1, DP2, DP3, DP4, DP5, DP6, DP7, DN0, DN1, DN2, DN3, DN4, DN5, DN6, DN7, DMX, DMY

- **Hardware resources**

The hardware resources for this function are as follows.

Maximum stack level:	6
Maximum loop stack level:	4
Maximum number of repeats:	340
Maximum MIPS value:	60.8

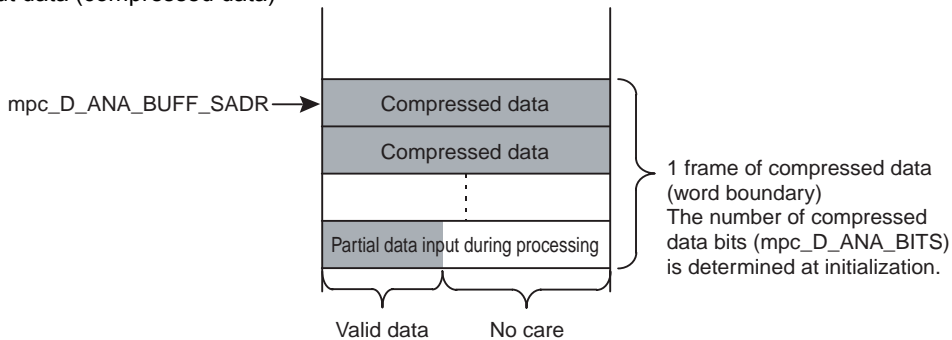
★

2.3.4 mpc\_Dec function

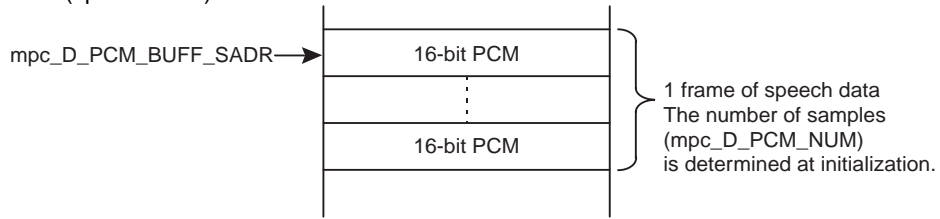
- **Classification** MPEG-4 CELP decoder processing
- **Function name** mpc\_Dec
- **Summary of function** Decompresses 1 frame of the specified compressed speech data.
- **Format** call mpc\_Dec
- **Arguments**
  - \*mpc\_D\_ANA\_BUFF\_SADR:x Input data buffer start address (X memory)
  - \*mpc\_D\_PCM\_BUFF\_SADR:x Output data buffer start address (X memory)
  - \*mpc\_D\_ERR\_FLAG:x Recover processing flag. 0: OFF/1: ON
- **Return value** None

★ **Function** Performs decompression processing for compressed data specified by mpc\_D\_ANA\_BUFF\_SADR and writes and returns this data to the speech data buffer specified by mpc\_D\_PCM\_BUFF\_SADR. When the mpc\_D\_ERR\_FLAG is on, that frame is regarded to have an error, and recovery is performed by error concealment processing using the data of the previous frame.  
 A frame error must be detected by the calling system.  
 Recovery processing is performed in two modes, BE (Bit Error) mode and FE (Frame Erasure) mode, and is determined depending on the library to be embedded (refer to **1.2 MPEG-4 CELP Speech Codec** and **1.3.4 Directory structure**).  
 If the final word of the compressed data frame is less than 1 word, the valid data is input starting from the MSB.  
 The size of the compressed data input and the number of samples output in 1 frame is determined by the bit rate. The bit rate cannot be changed.  
 If silence compression is used, the size of the compressed data input varies depending upon the frame.  
 The compressed data is a stream of bit units.

Input data (compressed data)



Output data (speech data)



- **Registers used** R0, R1, R2, R3, R4, R5, R6, R7, DP0, DP1, DP2, DP3, DP4, DP5, DP6, DP7, DN0, DN1, DN2, DN3, DN4, DN5, DN6, DN7, DMX, DMY

- **Hardware resources**

The hardware resources for this function are as follows.

Maximum stack level:	7
Maximum loop stack level:	3
Maximum number of repeats:	400
Maximum MIPS value:	12.5 (with no error code)

★

### 2.3.5 mpc\_GetVersion function

- **Classification**           Version information acquisition
  
- **Function name**           mpc\_GetVersion
  
- **Summary of function**   Returns the version of the library.
  
- **Format**                   call mpc\_GetVersion
  
- **Arguments**               None
  
- **Return Value**           R0H           Major version number  
                              R0L           Minor version number
  
- **Function**               Returns the version number of the MPEG-4 CELP speech codec library as a 32-bit value.  
                              When R0 = 0x00'0x0001'0x0100:  
                              Version: V1.01
  
- **Registers used**         R0
  
- **Hardware resources**  
The hardware resources for this function are as follows.
  - Maximum stack level:       1
  - Maximum loop stack level:  0
  - Maximum number of repeats: 0
  - Maximum number of cycles:  6



★ **2.4 Compressed Data Format**

The MPEG-4 CELP speech codec has two types of compressed data formats: version 1 and version 2.

With this middleware, the encoder outputs compressed data of version 1. The decoder can read compressed data of both versions 1 and 2.

Version 1: Conforms to MPEG-4 CELP object

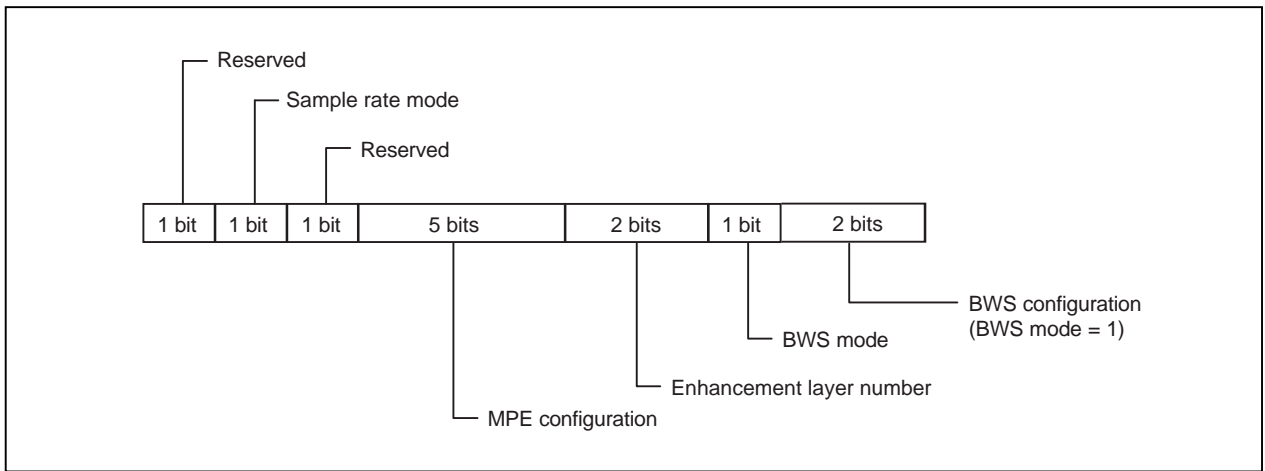
Version 2: Conforms to MPEG-4 ER-CELP object

**2.4.1 Header**

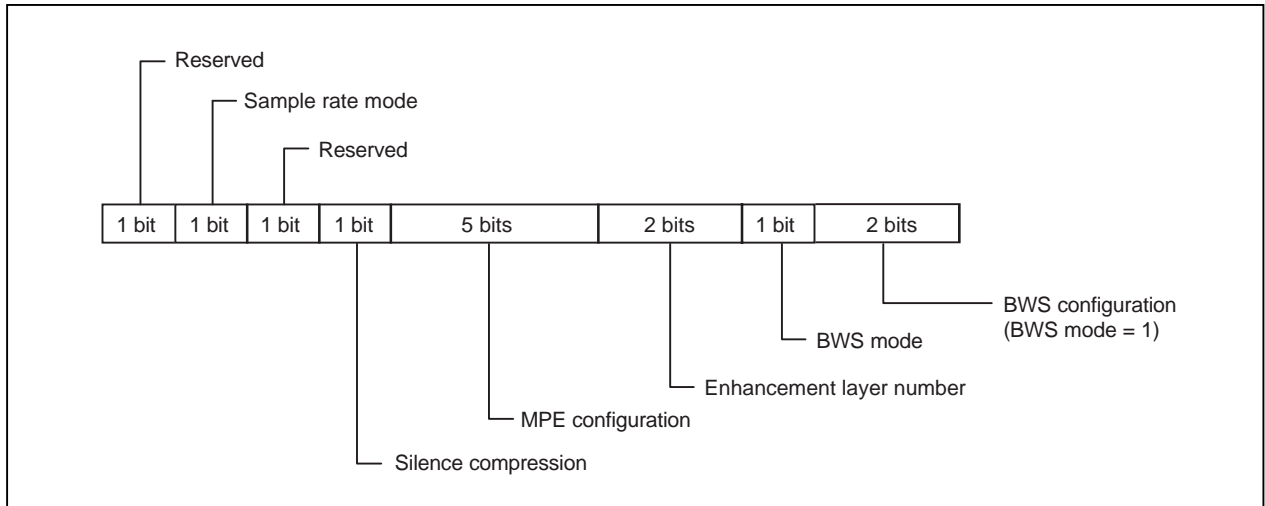
With version 1, the header size is 11 bits (BWS = OFF) or 13 bits (BWS = ON), and the header configuration is as shown in Figure 2-5 (a). With version 2, the header size is 12 bits (BWS = OFF) or 14 bits (BWS = ON), and the header configuration is as shown in Figure 2-5 (b).

**Figure 2-5. Header Format**

(a) Version 1



(b) Version 2

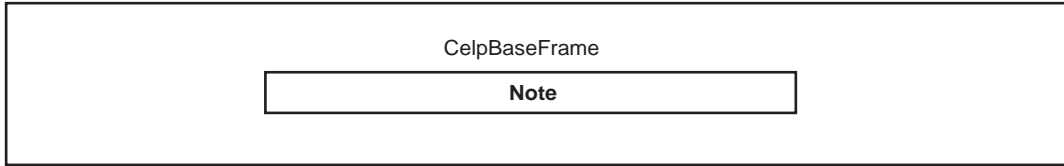


- (1) Sample rate mode:  
Determines the sampling frequency of MPEG-4 CELP. 8 kHz (narrow band): 0, 16 kHz (wide band): 1
- (2) Silence compression (provided in version 2 only):  
Specifies silence compression.  
Off: 0, on: 1
- (3) MPE configuration:  
Determines the compression bit rate of MPEG-4 CELP. (28 types/30 types)
- (4) Number of enhancement layers:  
Not supported in this middleware (always 0).
- (5) BWS mode:  
Not supported in this middleware (always 0).
- (6) BWS configuration:  
Not supported in this middleware (this bit does not exist).

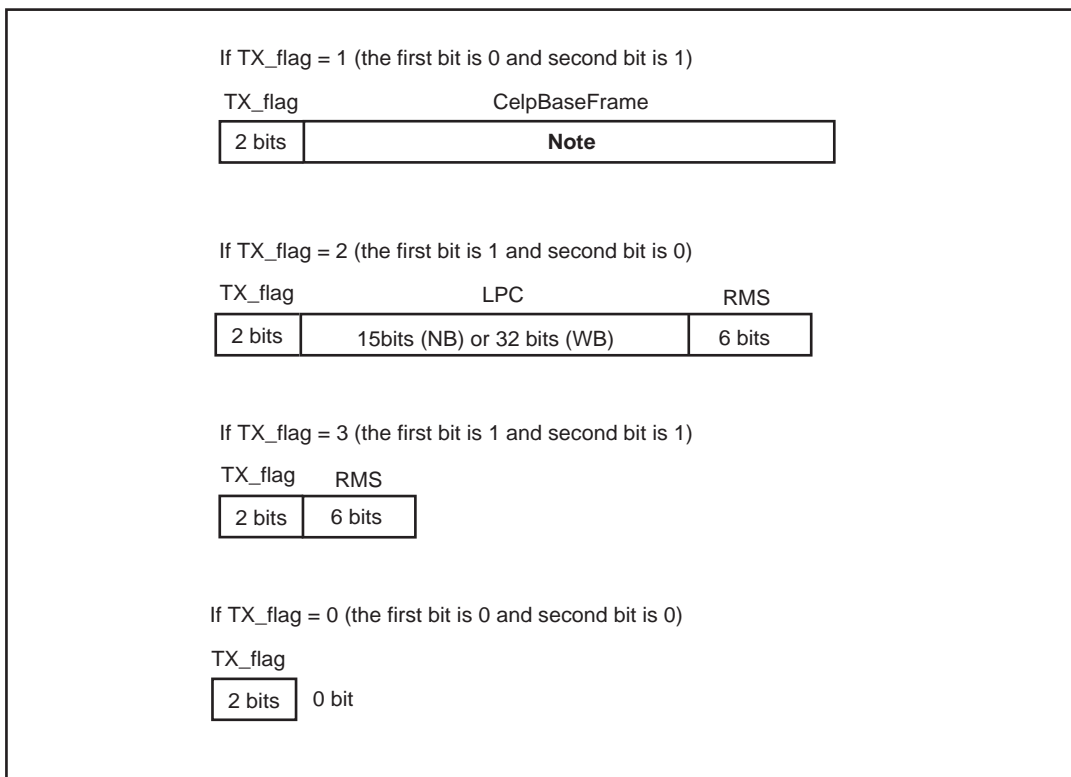
2.4.2 Frame

Figure 2-6. Frame Configuration

(a) With version 1 or version 2 when Silence Compression = OFF



(b) With version 2 when Silence Compression = ON



**Note** The bit length of the CelpBaseFrame differs depending upon the bit rate and can be calculated by the following expression (refer to **2.5 Relationship Between Bit Rate and Number of Samples**):

- For Narrow Band  
 Compressed bit rate (X) × Number of samples per frame (L) ÷ 8000
- For Wide Band  
 Compressed bit rate (X) × Number of samples per frame (L) ÷ 16000

The bit length can be obtained by the return value "mpc\_E\_ANA\_BITS:x" of the mpc\_InitEnc function or the return value "mpc\_D\_ANA\_BITS:x" of the mpc\_InitDec function.

## 2.5 Relationship Between Bit Rate and Number of Samples

The relationship between the number of samples and the 28 types of compression bit rates for the narrow band (8 kHz sampling) and 30 types of compression bit rates for the wide band (16 kHz sampling) is shown in the tables below.

**Table 2-1. Relationship Between Narrow Band Bit Rate and Number of Samples**

No.	Compression Bit Rate (bps): X	Number of Samples per Frame: L
0	3850	320
1	4250	320
2	4650	320
3	4900	240
4	5200	240
5	5500	240
6	5700	160
7	6000	160
8	6300	160
9	6600	160
10	6900	160
11	7100	160
12	7300	160
13	7700	160
14	8300	160
15	8700	160
16	9100	160
17	9500	160
18	9900	160
19	10300	160
20	10500	160
21	10700	160
22	11000	80
23	11400	80
24	11800	80
25	12000	80
26	12200	80
27	6200	240

**Table 2-2. Relationship Between Wide Band Bit Rate and Number of Samples**

No.	Compression Bit Rate (bps): X	Number of Samples per Frame: L
0	10900	320
1	11500	320
2	12100	320
3	12700	320
4	13300	320
5	13900	320
6	14300	320
7	Reserve	—
8	14700	320
9	15900	320
10	17100	320
11	17900	320
12	18700	320
13	19500	320
14	20300	320
15	21100	320
16	13600	160
17	14200	160
18	14800	160
19	15400	160
20	16000	160
21	16600	160
22	17000	160
23	Reserve	—
24	17400	160
25	18600	160
26	19800	160
27	20600	160
28	21400	160
29	22200	160
30	23000	160
31	23800	160

## CHAPTER 3 INSTALLATION

### 3.1 Installation Procedure

The MPEG-4 CELP speech codec middleware is supplied on a 3.5-inch floppy disk (1.44 MB). The procedure for installing the  $\mu$ SAP77016-B05 in the host machine is outlined below.

- (1) Set the floppy disk in the floppy disk drive and copy the files to the directory where tools from ATAIR are used (e.g. C:\DSPTools). The following is an example of when files are copied from the A drive to the C drive.

```
A:\>xcopy /s *.* c:\DSPTools <CR>
```

- (2) Confirm that the files have been copied. Refer to **1.3.4 Directory structure** for details on the directories.

```
A:\>dir c:\DSPTools <CR>
```

### 3.2 Sample Creation Procedure

The sample program is stored in the sample directory of the supplied medium. With the sample program, it is possible to simulate the external I/O of speech and compressed data by using a timing file to be described later (in **CHAPTER 4 SYSTEM EXAMPLE**) on the SM77016 ATAIR simulator.

The following is an explanation of how to build the MPEG-4 CELP speech codec middleware sample program.

- (1) Start up the WB77016 (workbench).
- (2) Open the sample.prj project.  
**Example** Specify sample.prj from Open Project on the Project menu.
- (3) Execute Build and confirm that sample.lnk has been created.  
**Example** The sample.lnk file is generated by selecting Build All from the Make menu.
- (4) Start up the SM77016 (simulator)
- (5) Open sample.lnk  
**Example** Specify sample.lnk by clicking Open on the File menu.
- (6) Open the timing file sample.tmg.  
**Example** Specify sample.tmg by clicking Open on the File menu.

### 3.3 Symbol Naming Regulations

The section names used in this library are shown below.

Classification	Regulation
Function name, variable name	mpc_xxxx
Macro, constant name	MPC_XXXX
Section name	__MPC_XXXX (Two underscores at the start)

## CHAPTER 4 SYSTEM EXAMPLE

### 4.1 Simulation Environment in Which Timing File Used

An example in which the speech codec compression/decompression processing simulator and a timing file are used is shown below. Speech data is input, and then output frame by frame after each frame has undergone compression/decompression processing.

Software environment:

- High-speed simulator: HSM77016
- Sample program: sample.lnk (created in **3.2 Sample Creation Procedure**)
- Timing file: sample.tmg

### 4.2 Operation

- (1) Start up the HSM77016 (high-speed simulator)
- (2) Open sample.lnk, which was created in **3.2 Sample Creation Procedure**.  
**Example** Specify sample.lnk by clicking Open on the File menu.
- (3) Open the timing file sample.tmg.  
**Example** Specify sample.tmg by clicking Open on the File menu.
- (4) Make the wait settings.  
**Example** Set waits in the DWTR/IWTR registers in the setting windows opened by clicking Periphery Register on the Window menu.
- (5) Execute with Run.

The timing file sample.tmg is described below.

The high-speed simulator (HSM77016) provides a function for simulating external I/O using a timing file.

#### (a) Data file input (16-bit data)

Data is input from a file via the host interface. An example of the description format is shown below.

##### • Preparation

```
open input "input.dat"      ; Input file specification (encode data)
input format hex           ; Input file format specification
```

##### • Input processing (16-bit data)

```
wait cond pin hwe == 0     ; Waiting until the host can write to HDT(in).
wait cond pin hcs == 1     ; Waiting until the chip select signal becomes inactive.

set pin hcs = 0            ; Input starts when the chip select signal becomes active.
; (A)
set port ha = 0            ; The lower 8 bits of the HDT register are selected.
set pin hwr = 0            ; The host write strobe is activated.

input data                  ; Data values are input from the data file and are assigned as "data".
set port hd = data&0xFF    ; The 8-bit "data" is input from the HD port.
wait 100ns                 ; Waiting until the data is transferred.
```

```

set pin hwr = 1           ; Input ends when hwr becomes inactive.
wait 5ns                 ; delay
; (B)

;The processing from (A) to (B) is repeated for higher 8 bits.
set port ha = 1          ; The higher 8 bits of the HDT register are selected.
wait 5ns                 ; delay
set pin hwr = 0          ; start output
set port hd = (data>>8)&0xFF ; input high byte to host port
wait 100ns               ; access duration
set pin hwr = 1          ; end input

set pin hcs = 1          ; Input ends when the chip select signal becomes inactive.
    
```

- Termination

```

close input              ; The data file is closed.
    
```

(b) Data file output (16-bit data)

- Preparation

```

open output "out.dat"    ; Output file specification (encode data)
output format HIDEBASE unsigned hex ; Output file format specification
    
```

- Output processing (16-bit data)

```

wait cond pin hre == 0   ; Waiting until the host can read from HDT(out).
wait cond pin hcs == 1   ; Waiting until the chip select signal becomes inactive.

set pin hcs = 0          ; Output starts when the chip select signal becomes active.
; (A)
set port ha = 0          ; The lower 8 bits of the HDT register are selected.
set pin hrd = 0          ; The host read strobe is activated.

wait 50ns                ; access duration
set lowbyte = port hd&0xff ; 8 bits are read out from the HD port and are assigned as "lowbyte".
set pin hrd = 1          ; Output ends when hrd becomes inactive.
wait 5ns                 ; delay
; (B)

;The processing from (A) to (B) is repeated for higher 8 bits.
set port ha = 1          ; The higher 8 bits of the HDT register are selected.
wait 5ns                 ; delay
set pin hrd = 0          ; start output
wait 50ns                ; access duration
output ((port hd&0xFF)<<8)|lowbyte; output word data to file
set pin hrd = 1          ; end output

set pin hcs = 1          ; Output ends when the chip select signal becomes inactive.
    
```



- Termination

```
close input          ; The data file is closed.
```

★

## APPENDIX SAMPLE PROGRAM SOURCE

```

/*-----*/
/* File Information */
/*-----*/
/* Name : sample.asm for Timing File */
/* Type : SPX Assembler Code */
/* Version : 2.00a */
/* Date : 2000.06.21 */
/* CPU : uPD7701x */
/* Compiler: Atair uPD77016 Workbench */
/* About : encoder function module */
/*-----*/
/* Copyright (C) NEC Corporation 1998,1999 */
/* NEC CONFIDENTIAL AND PROPRIETARY */
/* All rights reserved by NEC Corporation. */
/* Use of copyright notice does not evidence publication */
/*-----*/

#include "mpc_enc.h"
#include "mpc_dec.h"

#define STACK_SIZE_E 0x1500
#define STACK_SIZE_D 0x700

#define HST 0x3807
#define HDT 0x3806
#define HDO 0x3806

#define START_ADDRESS 0x200
#define VECTR_ADDRESS 0x210
#define BEGIN_ADDRESS 0x240

#define WB_FLAG 1 /* 0:NB / 1:WB */
#define RATE_NUMBER 6 /* bit rate number */
#define FRAME_SIZE 320 /* PCM frame size */
#define POSTFILTER_FLAG 1 /* PostFilter 0:off/1:on */
#define VERSION 1 /* Bitstream version */

%DEFINE(_read_host(reg))
(
    r0l=*HST:x;
    r0=r0 & 0x1;
    if(r0 != 0) jmp $-2;
    r@reg = *HDT:x;
)
%DEFINE(_write_host(reg))
(
    r0l=*HST:x;
    r0=r0 & 0x2;
    if(r0 != 0) jmp $-2;
    *HDT:x=r@reg;
)
%DEFINE(_VECTOR_DEF)

```

```

(
    nop;
    reti;
    nop;
    nop;
)

/*-----*/

USER_DATA XRAMSEG
    InputData:
        ds 320;
    CodeData:
        ds 30;
    OutputData:
        ds 320;

USER_FREE_X xramseg
    _USER_Free_X_Area_D:
        ds STACK_SIZE_D;
    _USER_Free_X_Area_E:
        ds STACK_SIZE_E;

USER_FREE_Y yramseg
    _USER_Free_Y_Area_D:
        ds STACK_SIZE_D;
    _USER_Free_Y_Area_E:
        ds STACK_SIZE_E;

/*-----*/
/* Function Name : main */
/*-----*/
/* others: r_      [* , , , , , , , *]  dmx,dmy [ , ] */
/*           dp_    [* , , , , , , , ]  loops/stacks [1/0] */
/*           dn_    [ , , , , , , , ]  cycles      ??? */
/*-----*/

Startup imseg at START_ADDRESS
    jmp _main;
Vector imseg at VECTR_ADDRESS
;0x210 INT1
    %_VECTOR_DEF;
;0x214 INT2
    %_VECTOR_DEF;
;0x218 INT3
    %_VECTOR_DEF;
;0x21c INT4
    %_VECTOR_DEF;
;0x220 Serial Input 1
    %_VECTOR_DEF;
;0x224 Serial Output 1
    %_VECTOR_DEF;
;0x228 Serial Input 2
    %_VECTOR_DEF;

```

```

;0x22c Serial Output 2
  %_VECTOR_DEF;
;0x230 Host Input
  %_VECTOR_DEF;
;0x234 Host Output
  %_VECTOR_DEF;

main imseg at BEGIN_ADDRESS
_main:
  clr(r0);
  call  _Init_Int;      /* Initialize interrupt */

  /* ----- */
  /* clear memory */
  /* ----- */
  clr(r0);
  clr(r1);
  clr(r2);
  r0l=_USER_Free_X_Area_E;
  r1l=_USER_Free_Y_Area_E;
  r2l=STACK_SIZE_E;
  call  _Zero_Mem;
  clr(r0);
  clr(r1);
  clr(r2);
  r0l=_USER_Free_X_Area_D;
  r1l=_USER_Free_Y_Area_D;
  r2l=STACK_SIZE_D;
  call  _Zero_Mem;
  /* ----- */
  /* ----- */

  /* ----- */
  /* Setting for Encoder */
  /* ----- */
  r7l = CodeData;
  *mpc_E_ANA_BUFF_SADR:x = r7l;
  r7l = RATE_NUMBER;
  *mpc_E_RATE_NUM:x = r7l;
  r7l=WB_FLAG;
  *mpc_E_WB:x=r7l;
  clr(r6);
  clr(r5);
  r6l=_USER_Free_X_Area_E;
  r5l=_USER_Free_Y_Area_E;
  call  mpc_InitEnc;      /* For Encoder Function */

  /* ----- */
  /* Setting for Decoder */
  /* ----- */
  r7l= CodeData;
  *mpc_D_ANA_BUFF_SADR:x = r7l;
  r7l = POSTFILTER_FLAG;
  *mpc_D_PF_FLAG:x=r7l;
  r7l = VERSION;

```

```

*mpc_D_VER:x=r71;
clr(r6);
clr(r5);
r6l=_USER_Free_X_Area_D;
r5l=_USER_Free_Y_Area_D;
call    mpc_InitDec;

/* ----- */
/* Begin Main Loop      */
/* ----- */
_main_loop:
/* ----- */
/* Read Input Data      */
/* ----- */
r0l=InputData;
call    _Read_Code_Data;

/* ----- */
/* Encode one Frame     */
/* ----- */
r7l = InputData;
*mpc_E_PCM_BUFF_SADR:x = r7l;
call    mpc_Enc;

/* ----- */
/* Decode one Frame     */
/* ----- */
r7l = OutputData;
*mpc_D_PCM_BUFF_SADR:x = r7l;
clr(r0);
*mpc_D_ERR_FLAG:x = r0l;
call    mpc_Dec;

/* ----- */
/* Write Output Data    */
/* ----- */
r0l=OutputData;
r1l=*mpc_D_PCM_NUM:x;
call    _Write_Code_Data;

    jmp _main_loop;

/*-----*/
/* Function Name : _Init_Int          */
/* disable internal interrupt HO,HI  */
/*-----*/
/* others: r_      [*, , , , , , , ]  dmx,dmy [ , ]          */
/*          dp_    [ , , , , , , , ]  loops/stacks [0/0]     */
/*          dn_    [ , , , , , , , ]  cycles      ???        */
/*-----*/
_init_Int:
    r0l=sr;
    r0=r0 | 0x0300;
    sr=r0l;
    r0l=0x0001;
    *HST:x=r0l;

```

```

ret;

/*-----*/
/* Function Name : _Write_Code_Data */
/* [argv] */
/* r0l:Write Data Ponter : Xmem */
/* r1l:Number of Write Data */
/*-----*/
/* others: r_ [* , * , , , , , , *] dmx,dmy [ , ] */
/* dp_ [* , , , , , , , ] loops/stacks [1/0] */
/* dn_ [ , , , , , , , ] cycles ??? */
/*-----*/
_Write_Code_Data:
    dp0=r0l;
    if(r1 == 0) jmp _end_Write_Code_Data;
    r7l=*dp0++;
    loop r1l{
        %_write_host(7l);
        r7l=*dp0++;
        nop;
    };
_end_Write_Code_Data:
    r0l=*HST:x;
    r0=r0 & 0x2;
    if(r0 != 0) jmp $-2;
    ret;

/*-----*/
/* Function Name : _Read_Code_Data */
/* [argv] */
/* r0l:Read Data Ponter : Xmem */
/* [ret] */
/*-----*/
/* others: r_ [* , , , , , , , ] dmx,dmy [ , ] */
/* dp_ [* , , , , , , , ] loops/stacks [1/0] */
/* dn_ [ , , , , , , , ] cycles ??? */
/*-----*/
_Read_Code_Data:
    dp0=r0l;
    r0l=FRAME_SIZE;
    loop r0l{
        %_read_host(0);
        *dp0++=r0h;
        nop;
    };
    ret;

/*-----*/
/* Function Name : _Zero_Mem */
/* clear memory */
/* [argv] */
/* r0l:Start Address : Xmem */
/* r1l:Start Address : Ymem */
/* r2l:Size */
/* [ret] */
/* r0:0 :Error !=0:OK */
/*-----*/

```

```
/*-----*/
/* others: r_      [*,* , , , , , ]  dmxdmy [ , ]          */
/*           dp_   [* , , , , * , , ]  loops/stacks [1/0]    */
/*           dn_   [ , , , , , , , ]  cycles      ???        */
/*-----*/
  _Zero_Mem:
    dp0=r0l;
    dp4=r1l;
    clr(r1);
    clr(r0);
    rep r2l;
    *dp0++=r0l  *dp4++=r1l;
    ret;
```

END

## Facsimile Message

Although NEC has taken all possible steps to ensure that the documentation supplied to our customers is complete, bug free and up-to-date, we readily accept that errors may occur. Despite all the care and precautions we've taken, you may encounter problems in the documentation. Please complete this form whenever you'd like to report errors or suggest improvements to us.

From:

Name

Company

Tel.

FAX

Address

**Thank you for your kind support.**

**North America**

NEC Electronics Inc.  
Corporate Communications Dept.  
Fax: 1-800-729-9288  
1-408-588-6130

**Hong Kong, Philippines, Oceania**

NEC Electronics Hong Kong Ltd.  
Fax: +852-2886-9022/9044

**Asian Nations except Philippines**

NEC Electronics Singapore Pte. Ltd.  
Fax: +65-250-3583

**Europe**

NEC Electronics (Europe) GmbH  
Technical Documentation Dept.  
Fax: +49-211-6503-274

**Korea**

NEC Electronics Hong Kong Ltd.  
Seoul Branch  
Fax: 02-528-4411

**Japan**

NEC Semiconductor Technical Hotline  
Fax: 044-435-9608

**South America**

NEC do Brasil S.A.  
Fax: +55-11-6462-6829

**Taiwan**

NEC Electronics Taiwan Ltd.  
Fax: 02-2719-5951

I would like to report the following error/make the following suggestion:

Document title: \_\_\_\_\_

Document number: \_\_\_\_\_ Page number: \_\_\_\_\_

If possible, please fax the referenced page or drawing.

Document Rating	Excellent	Good	Acceptable	Poor
Clarity	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Technical Accuracy	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Organization	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>