E-GOLDradio

GSM/GPRS Single Chip Solution

PMB 7870

V1.00

Secure Mobile Solutions

Infineon technologies

Never stop thinking.

Edition 2005-12-07

Published by Infineon Technologies AG,
St.-Martin-Strasse 53,
81669 München, Germany

Attention please!

The information herein is given to describe certain components and shall not be considered as a guarantee of characteristics.

Terms of delivery and rights to technical change reserved.

We hereby disclaim any and all warranties, including but not limited to warranties of non-infringement, regarding circuits, descriptions and charts stated herein.

Information

For further information on technology, delivery terms and conditions and prices please contact your nearest Infineon Technologies Office (**www.infineon.com**).

Warnings

Due to technical requirements components may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies Office.

Infineon Technologies Components may only be used in life-support devices or systems with the express written approval of Infineon Technologies, if a failure of such components can reasonably be expected to cause the failure of that life-support device or system, or to affect the safety or effectiveness of that device or system. Life support devices or systems are intended to be implanted in the human body, or to support and/or maintain and sustain and/or protect human life. If they fail, it is reasonable to assume that the health of the user or other persons may be endangered.

E-GOLDradio

GSM/GPRS Single Chip Solution

PMB 7870

V1.00

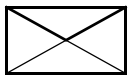
Secure Mobile Solutions

Never stop thinking.

We Listen to Your Comments

Any information within this document that you feel is wrong, unclear or missing at all?
Your feedback will help us to continuously improve the quality of this document.
Please send your proposal (including a reference to this document) to:

smsdocu.comments@infineon.com



1 Introduction

Attention: This document is valid starting from Startup Code G16 V1.3

E-GOLDradio CONFIDENTIAL		
Revision History:	**2005-12-07**	Rev. 1.01
Previous Version:	Rev. 1.00, 2005-05-16	
Page	Subjects (major changes since last revision)	
	Initial Version based on *E-GOLDradio G14 Firmware Manual*	
Changes for Rev. 1.01		

Remarks on the Usage of This Document

- The digital signal processor is called TEAKlite or DSP. The microcontroller is called MCU.
- All references to shared memory locations are made in code-style letters like **SM_BOOT_DATA**. All names for these shared memory locations have the sequence "SM_.." as their leading letters. The exact position and size of these shared memory locations can be read from **Section 7.1 “Contents of Shared Memory” on Page 149**.
- The bitstream in a shared memory block always has the same order. The first bit of the bitstream is in the first word at bit position ’0’ (LSB). The second bit is at bit position ’1’. The 17th bit is located in the second word at bit position ’0’ and so on.
- All RMS values in this document are given in [dB/16]. Therefore, if the signal level is reduced to 50% (6 dB) the corresponding RMS value is decreased by 6*16 = 96_D. The maximum value that can be reached with a full scale signal in the baseband buffer is about 1400_D.
- All metric references in this document (**SM_SYNC_METRIC**, **SM_TCH_METRIC**, etc.) are output values of the Viterbi-Algorithm in the Channel-Decoder. These values indicate how many bits have been corrected by the Viterbi-Algorithm. Therefore, a metric value of ’0’ means that no bits have been corrected, a value of n means that n bits have been corrected. The maximum value of n depends on the channel type (for example, <= 78 for Synch Bursts, <= 378 for Full Speech).

Note: The Infineon interfaces contain links to MP3 algorithms, which are proprietary to the Fraunhofer Gesellschaft. Infineon does not grant any license or right to use MP3. For detailed information about MP3 or the right to use such Intellugale Property, please contact the Fraunhofer Gesellschaft directly.

1.1 Overview

The E-GOLDradio DSP subsystem consists of a 16-bit TEAKlite core (running at 104MHz) and hardware peripherals. It is a coprocessor for the MCU. On the DSP, there are algorithms and an operating system called the scheduler.

1.2 Algorithm Partitioning

The signal processing algorithms listed below are implemented in Firmware modules. These modules are in a 80 k-word program ROM and a 4 k-word program RAM:

- Scanning of channels for measurement of field strengths of neighboring base stations
- Detection and evaluation of Frequency Correction Bursts
- Equalization of GMSK Normal Bursts and Synchronization Bursts with bit-by-bit soft-output
- Synch burst channel decoder
- Channel encoding and soft-decision decoding for fullrate, enhanced-fullrate, halfrate, and AMR speech
- Support for fullrate (F14.4, F9.6, F4.8, and F2.4) and halfrate (H2.4 and H4.8) data channels
- Control channels as well as Rach and Prach
- GPRS coding schemes (CS1-CS4)
- Fast USF detection algorithms for the Medium Access Control (MAC) software layer
- Fullrate, enhanced fullrate, and halfrate speech encoding and decoding
- Adaptive multi-rate (fullrate and halfrate) speech encoding and decoding
- Mandatory sub-functions such as:
  - Discontinuous transmission, DTX (GSM 46.031, 46.041, 46.081, and 46.093 standards)
  - Voice activity detection, VAD (GSM 46.032, 46.042, 46.082, and 46.094 standards)
  - Background noise calculation (GSM 46.012, 46.022, 46.062, and 46.092 standards)
- Generation of tone and side tone
- Hands-free function
- Support for voice memo
- Handling of vocoder and voice-paths for type approval testing
- Synthesizer with up to 40 voices at 16 kHz sampling rate and 21 voices at 32 kHz.
- TTY/CTM converter
- PCM/ADPCM Player
- MP3 Player (optional).

1.3 Scheduler Functions

The scheduler is based on an operating system and is triggered by interrupts generated by hardware peripherals or commands from the MCU.

The scheduler features:

- Communications between DSP and MCU
- Semi-automatic handling of control channels
- Fully automatic handling of speech and data traffic channels
- Support of the GSM ciphering algorithms (A51, A52, and A53) in combination with a hardware accelerator
- Support for High Speed Circuit Switched Data (HSCSD) with a maximum of 4 RX and 1 TX or 3 RX and 2 TX timeslots (Class 10 mobile)
- Support for General Packet Radio Services GPRS with up to 4 RX and 1 TX or 1 RX and 4 TX (Class 12 mobile).

Note: The E-GOLDradio GPRS Class depends on the MCU workload.

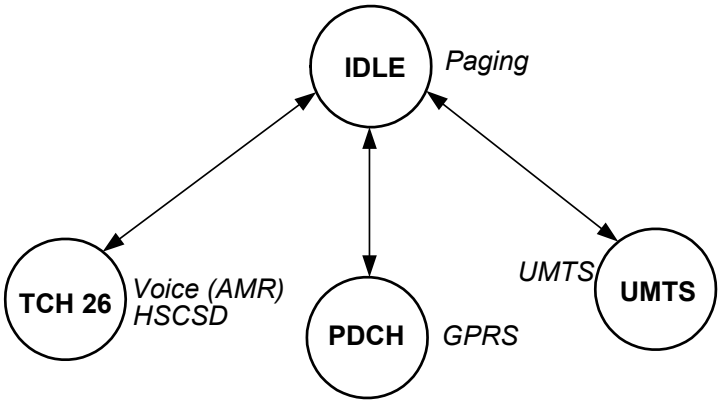
- Monitoring of paging blocks for packet switched and circuit switched services simultaneously (GPRS MS in Class-B mode of operation)
- Loop-back functions (according to GSM 11.10 standard)
- Audio scheduler that supports various sample rates (including handling of MP3, Synthesizer, TTY/CTM, PCM/ADPCM player, and echo cancellation algorithms)
- Voice Memo for Full Rate Speech and AMR.

For the modem the scheduler has four states:

1. IDLE State:
   – Receiving and transmitting frames that do not belong to a circuit or packet switched channel
   – Receiving paging channels and broadcast channels on the 51 multiframe.
2. TCH26 State:
   – Speech circuit switched services with narrowband AMR
   – Data services and HSCSD (By-Pass-Mode).
3. PDCH State:
   – GPRS packet switched services.
4. UMTS State:
   – Only speech processing support for an external UMTS modem processor.

The transition from one state to another can only be forced by the MCU. As it is shown in **Figure 1-1** the transition between the states TCH26, UMTS, and PDCH can only be made by going through IDLE state.

Figure 1-1 Firmware States



The DSP modem functions belong to all four states: DSP IDLE, TCH26, PDCH, UMTS.

Table 1-1 shows the applications that can be used during each of the four DSP states.

Table 1-1 Usable Applications during the Different FW States at 104 MHz

DSP Status (Modem Functions)	Voiceband Processing Functions (Applications)							
	Internal Voice Memo	Number of Synthesizer Voices/16 kHz	Number of Synthesizer Voices/32 kHz	MP3	Hands free	TTY-CTM	I^2S_y external Mode Circular Buffer	PCM Player
Idle	yes	40	21	yes	-	-	yes	yes
Tch26 Signal Only	yes	32	17	yes	-	-	yes	yes
Tch26 Data Services Single Slot	yes	-	-	-	-	-	yes	yes
Tch26 HSCSD	-	-	-	-	-	-	yes	yes
Tch26 Voice (Speech Calls such as AMR, etc.)	yes	-	-	-	yes	yes	yes	yes
PDCH GPRS	yes	14	7	Class 9	-	-	yes	yes
UMTS	yes	-	-	-	yes	-	yes	yes

Table 1-2 Combination of Audio Applications

Audio Application	I2Sy_MMS	I2Sy_External	Synthesizer	PCM Player	Internal Voice Memo	Handsfree	TTY/CTM	Tone Generator
I2Sy_MMS		no	yes	yes	yes	yes	yes	yes
I2Sy_External	no		no	no	yes	yes	yes	yes
Synthesizer	yes	no		no	no	no	yes	yes
PCM Player	yes	no	no		yes	yes	no	yes
Internal Voice Memo	yes	yes	no	yes		no	no	yes
Handsfree	yes	yes	no	yes	no		no	yes
TTY/CTM	yes	yes	yes	no	no	no		yes
Tone Generator	yes	yes	yes	yes	yes	yes	yes	

Notes:

1. *Modem related applications like Frequency Correction, SynchBurst and Monitoring always work in parallel.*
2. *The Audio Scheduler for 8 kHz must be started for every application!*
3. *The use of the Audio Scheduler for 16 kHz is only allowed in the Idle-mode. The Internal Voice Memo at 16 kHz is not possible.*

2 Booting

E-GOLDradio CONFIDENTIAL		
Revision History:	**2005-12-07**	Rev. 1.01
Previous Version:	Rev. 1.00, 2005-05-16	
Page	Subjects (major changes since last revision)	
	Initial Version based on *E-GOLDradio G14 Firmware Manual*	
Changes for Rev. 1.01		

2.1 Boot Concept

After Reset, the startup code for the DSP has to be booted: the program RAM (and, if needed, some parts of the data RAM) has to be downloaded from the MCU. The data is written by the MCU to the shared memory and read by the DSP from the shared memory to the DSP-internal memory (for details on the shared memory refer to **Chapter 7 Shared Memory**).

Two different boot options are supported by the DSP:

1. Normal Boot procedure
2. Fast Boot procedure used after the Standby Power Down mode.

The following Boot Commands are available:

- `PLOAD:` Load program RAM
- `DLOAD:` Load data RAM
- `BRANCH:` Branch to a specific address
- `FAST:` Fast Boot process after Standby Power Down
- `PREAD:` Read from program address space
- `DREAD:` Read from data address space.

*Note: The commands listed in **Section 3.2 "List of Commands" on Page 21** can NOT be applied before the DSP has been completely booted by the MCU. Only the boot commands listed above are allowed during this phase.*

*Note: After booting the startup-code, the MCU can use the shared memory locations **`SM_FW_VERSION`** and **`SM_HW_VERSION`** for other things, refer to **Section 7.1 "Contents of Shared Memory" on Page 149**. (If an error causes the DSP to reset, the first thing to do is verify that the correct versions of the Firmware and Hardware are being used by reading these two shared memory locations.)*

2.2 Normal Boot Procedure

This procedure is used for the first boot process of the DSP. During this process the DSP obtains the startup code from the MCU.

The following steps are done during the boot phase of the DSP:

1. Immediately after reset the DSP sets communication flag #0 to ’1’ to indicate to the MCU that the DSP is not ready to receive the first boot command.
2. After having some internal initialization (~2-3µs) the DSP resets communication flag #0 to ’0’ to indicate to the MCU that the DSP is ready to receive the first boot command.
3. The MCU may read the shared memory locations **SM_FW_VERSION** and **SM_HW_VERSION** to check the firmware version and the hardware version of the DSP-subsystem.
4. The MCU fills up the shared memory at address **SM_BOOT_DATA** with data belonging to the first boot command.
5. The MCU sets communication flag #0 to ’1’ and activate interrupt line #0 to the DSP. When this interrupt is received, the DSP starts accepting the boot command.
6. The MCU waits until the DSP has accepted the boot command by polling communication flag #0: as soon as the DSP has completely accepted the boot command it resets communication flag #0 to ’0’ again.
7. If this is not the last boot command (BRANCH), the MCU fills up the shared memory at address **SM_BOOT_DATA** with data belonging to the next boot command. Go back to step **5**.

The boot commands PLOAD and DLOAD, program and data RAM respectively, can be loaded. These commands can be called several times to load different parts of program RAM and data RAM. After loading all the necessary program RAM and data RAM locations, the command BRANCH has to be given to force DSP out of the boot loop to start normal program flow. The BRANCH command is always the last boot command.

Note: When the branch command is acknowledged by the DSP this guarantees that the DSP is ready to receive the first command (refer to ***Section 3.2 “List of Commands” on Page 21****) from the MCU. This means that the DSP Firmware is completely initialized.*

To give a boot command, the shared memory, starting at offset **SM_BOOT_DATA**, has to be filled as indicated in **Table 2-1**

Table 2-1 Load Boot commands

Shared Memory address	Boot Command PLOAD	Boot Command DLOAD	Boot Command BRANCH
SM_BOOT_DATA	0	1	2
SM_BOOT_DATA+1	P-RAM Destination	D-RAM Destination	Branch-Address
SM_BOOT_DATA+2	Block-Length1	Block-Length2	
SM_BOOT_DATA+3	Data-Word #1	Data-Word #1	
SM_BOOT_DATA+4	Data-Word #2	Data-Word #2	
...	...	...	
SM_BOOT_DATA+ Block_Length2+2	...	Data-Word #Block-Length2	
...	...		
SM_BOOT_DATA+ Block_Length1+2	Data-Word #Block-Length1		

Note: The maximum block-length for downloading program or data memory is limited to 507 (512-2-3) words.

Table 2-2 Read Boot commands

Shared Memory address	Boot Command PREAD	Boot Command DREAD	Boot Command FAST
SM_BOOT_DATA	3	4	5
SM_BOOT_DATA+1	P-Space Source	D-Space Source	
SM_BOOT_DATA+2	Block-Length	Block-Length	

Note: The maximum block-length for reading program or data memory is (512-2-3) words. On exit of the PREAD and DREAD commands the read data are located in shared memory beginning at **SM_BOOT_DATA***+3 and ending at* **SM_BOOT_DATA***+Block_Length+2.*

2.3 Fast Boot Procedure

This procedure wakes up the DSP from the Standby Power Down mode where the DSP and all peripherals are switched off. The shared memory that is switched on during the Standby Power Down must contain the startup code.

The Standby Power Down mode can be forced by applying the command **PW_DOWN**. The wake-up procedure is similar to the normal BOOT procedure. Only two MCU commands are necessary to wake-up the DSP:

1. Immediately after reset the DSP sets communication flag #0 to '1' to indicate to the MCU that the DSP is not yet ready to receive the first boot command.
2. After some internal initialization (~2-3µs), the DSP resets communication flag #0 to '0' to indicate to the MCU that the DSP is ready to receive the first boot command.
3. The DSP quickly restores the data from the shared memory by giving the command FAST. In the worst case, the DSP needs 6000 cycles for 1 kwords. For applying the FAST command, the MCU must set communication flag #0 to '1' and activate interrupt #0 to the DSP. Receiving this interrupt, the DSP starts accepting the FAST command.
4. After having finished the fast FAST-command, the DSP resets communication flag #0 and is now ready for the next command.
5. The MCU gives the BRANCH command.

The shared memory locations **SM_SBPD_INFO** and **SM_SBPD_BOOT_ADD** contain the Standby Power Down (SBPD) related data. Under all circumstances, it must be guaranteed that MCU does not overwrite those shared memory locations in the time period between giving the **PW_DOWN** command and the end of the fast boot procedure.

This SBPD support can only be used if the size of the Data and Program Code of the Startup Code does not exceed the size of the Shared Memory Field **SM_SBPD_BOOT_ADD**.

Note: The Firmware Group does not guarantee that this requirement is fulfilled for startup codes after startup version 1.0.

2.4 Firmware ID and DSP Subsystem ID

Table 2-3 Version Information Locations in SM

PMB 7870-Version	SM_FW_VERSION	SM_HW_VERSION	SCU_CHIPID.CHREV
PMB 7870 V1.0 G14	0604_H	$E001_H$	07_H
PMB 7870 V1.0 G16	0606_H	$E001_H$	07_H

The Firmware Mask version can be read from the Shared Memory location **SM_FW_VERSION** that is used for the PMB 7870 V1.x.

The hardware DSP Subsystem version can be read from the Shared Memory location **SM_HW_VERSION** that is used for the PMB 7870 V1.x.

The Chip Revision Number can be read from the chip identification bit field **SCU_CHIPID.CHREV**. For more information, refer to the *E-GOLDradio Design Specification*.

2.5 Startup-Code Version

From Startup-Code 1.0.0 the information about the Startup-Code version is written to the shared memory location **SM_STARTUP_CODE_VERSION**. The version number has three parts:

1. Startup-code version: This number starts from 1 and is increased for major changes (for example, feature extension, interface change, etc.). When this number is increased the other parts of the version number have to be reset to zero.

2. Startup-code subversion: This number starts from 0 and is increased for non-major changes (changes that are not covered by part 1). When this number is increased the third part of the version number has to be reset to zero.
3. Debug-code version: This number starts from 0 (for the official release) and is increased for every debug patch based for the current release.

Table 2-4 shows the location of the different parts in `SM_STARTUP_CODE_VERSION`.

Table 2-4 Bit ordering in `SM_STARTUP_CODE_VERSION`

Bit Number	Bit15...Bit12	Bit11...Bit14	Bit3...Bit0
Part	1	2	3
Name	Startup-Code Version	Startup-Code Sub-version	Debug-Code Version
Example	1	5	0

3 Commands

E-GOLDradio CONFIDENTIAL		
Revision History:	**2005-12-07**	Rev. 1.01
Previous Version:	Rev. 1.00, 2005-05-16	
Page	Subjects (major changes since last revision)	
	Initial Version based on *E-GOLDradio G14 Firmware Manual*	
Changes for Rev. 1.01		
Page 32	Update number of user commands	
Page 23	Add command parameter description for external audio output by I^2S_2	
Page 32	Modify default value of the **TX_DIG** command	

3.1 DSP Subsystem Command Interface to MCU

Commands are used to give the MCU control over the DSP. For example, the MCU can order the DSP to switch on or off DTX-mode or start a new traffic channel.

Commands are given via interrupts. The MCU writes parameters to a command block in the shared memory and give an interrupt to the DSP. After having received this interrupt the DSP will read the command block and interpret it.

There are three command pipes. Each of them has a separate interrupt line from the MCU to the DSP and its own command block in the shared memory. These three command pipes use the bits #0, #1, and #2 of register **SCU_MCUTODSP** to send an interrupt from the MCU to the DSP and the MCU must load the appropriate parameters into the shared memory locations **SM_MCU_CMD_0**, **SM_MCU_CMD_1**, and **SM_MCU_CMD_2**.

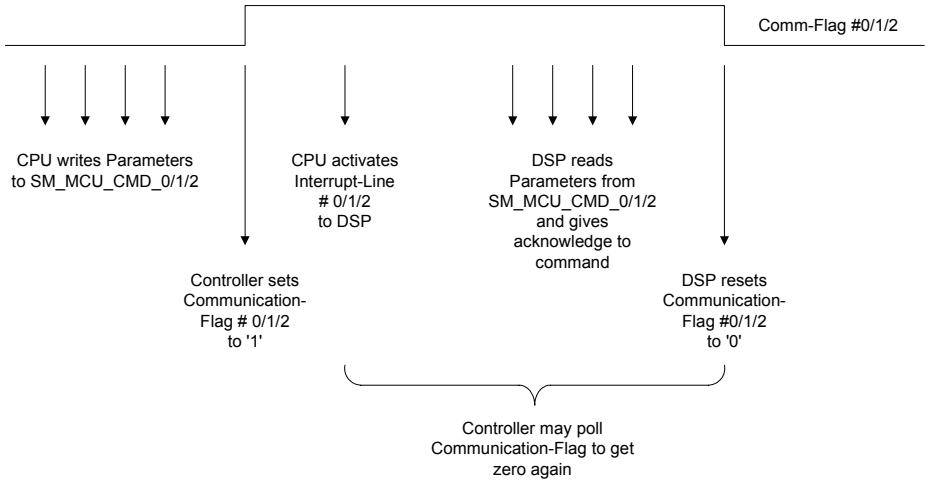
The three command pipes are orthogonal. This means that any command may be given via any of the command pipes #0, #1, and #2.

At the DSP all three command pipes have higher interrupt priority than all the other interrupts (except CODONHI, BBHI and I2S2RX by default or I2S1RX), but command pipe #0 has higher priority than command pipe #1, and command pipe #1 has higher priority than command pipe #2. This does not mean that command pipe #1 may interrupt command pipe #0; it means that, if pipe #0 and pipe #1 are active at the same time, pipe #0 will be served first.

A command is given in same way a boot command (boot commands are allowed in the boot phase only). The following steps have to be done (see **Figure 3-1**):

1. The MCU has to write the command code and the command parameters to the shared memory. This means that the command code has to be written to the location **SM_MCU_CMD_0**, **SM_MCU_CMD_1**, or **SM_MCU_CMD_2**, and the command parameters have to follow at the addresses **SM_MCU_CMD_0**/1/2+1, **SM_MCU_CMD_0**/1/2+2, etc.
2. The MCU sets communication flag #0, #1 or #2 to ’1’ and activates interrupt #0, #1 or #2 using register **SCU_MCUTODSP**. Upon receiving this interrupt the DSP starts accepting the command on the corresponding pipe.
3. After sending this interrupt the MCU waits until the DSP has accepted the command. This is done by (periodically) polling the communication flags because as soon as the DSP has accepted the command it will reset the communication flag to ’0’ again.
4. If the command has been accepted correctly, a value of "negative command number#" is returned by the DSP in the corresponding shared memory location **SM_MCU_CMD_0**, **SM_MCU_CMD_1** or **SM_MCU_CMD_2**. If the command failed, a value of ’0’ is returned.

Figure 3-1 Sending a Command from MCU to DSP



The procedure for the MCU to apply a command is always the same as described above, but the DSP distinguishes between two different types of commands:

1. Asynchronous commands

These commands are valid immediately after acceptance by the DSP. These are most commands are of this type.

2. Synchronous commands

These commands are accepted immediately, but do not become valid before the next frame interrupt. Only few commands belong to this type: **CCH_RX**, **CCH_TX**, **TCH_26**, and **PDCH**.

Note: The DSP subsystem guarantees that each command sent by the MCU is accepted within 70us.

3.2 List of Commands

Table 3-1 is the list of commands. A detailed description of these commands is given in Section 3.3 “Command Description” on Page 33..

Table 3-1 List of commands

Name	Command Number	Operation Description
	Parameters	**Parameter Descriptions**
FC_INIT	1	Initialization for FCB Search
	MODE	0: Stand-By 1: Traffic
	THRSH	Threshold value
	HYST	Hysteresis Value
MODU_INIT	2	Initialization for Modulator
	IOFFS	I-Offset Pre-Correction for the GMSK Modulator
	QOFFS	Q-Offset Pre-Correction for the GMSK Modulator
	IAMP	I-Amplitude Pre-Correction for the GMSK Modulator
	QAMP	Q-Amplitude Pre-Correction for the GMSK Modulator
	GMSK_ FREQ	Frequency Pre-Correction for the GMSK Modulator
	DUMMY	Additional number of dummy bits for the GMSK Modulator
	IQ_SETUP	Additional IQ Swap for the GMSK Modulator
IQ_SWAP_1	3	IQ-Swap for GMSK Normal Burst Rx and Tx
	SWITCH	0: IQ-Swap is off 1: IQ-Swap is on
IQ_SWAP_2	4	IQ-Swap for FCB-Search and Sync Burst Detection
	SWITCH	0: IQ-Swap is off 1: IQ-Swap is on
DEC_INIT	5	Initialization for Channel Decoder
	SWITCH	CODEC Switch 0: Full Rate Speech (FS) 1: Enhance Full Rate Speech (EFR) 2: Half Rate Speech (HS) 3: Adaptive Multi Rate (AMR)
	PAR 0 - 25	Number of parameters depends on SWITCH

Table 3-1 List of commands

Name	Command Number	Operation Description
	Parameters	**Parameter Descriptions**
CIPH_KEY	6	Set Cipher Mode and/or Cipher Keys
	SWITCH	1: Set Cipher Mode only 2: Set Cipher Keys only 3: Set both Cipher Mode and Cipher Keys
	CIPH	Ciphering: Off/A51/A52/A53
	SS	Sub-Stream Number
	CKEY0	Cipher Key 0
	CKEY1	Cipher Key 1
	CKEY2	Cipher Key 2
	CKEY3	Cipher Key 3
	CKEY4	Cipher Key 4 for A53, otherwise same as Cipher Key 0
	CKEY5	Cipher Key 5 for A53, otherwise same as Cipher Key 1
	CKEY6	Cipher Key 6 for A53, otherwise same as Cipher Key 2
	CKEY7	Cipher Key 7 for A53, otherwise same as Cipher Key 3
CCH_RX	7	Receive Control Channel
	CIPH	Ciphering: don't change/Off/A51/A52/A53
	TSC	Training Sequence Number
	EARLY	Defines channel decoding mode: 0: After receiving complete data frame (4 bursts) 1: After receiving partial frame (only 2 or 3 bursts)
CCH_TX	8	Transmit Control Channel
	CIPH	Ciphering: do not change/Off/A51/A52/A53
	TSC	Training Sequence Number
TCH_26	9	Start (or modify) TCH-26 Channel
	CHT	Channel Type
	CIPH	Ciphering: don't change/Off/A51/A52/A53
	TSC	Training Sequence Number
	INIT	Initialization Flag
	PAR1	Explained in detail in **Chapter 3.3.9 "TCH_26" on Page 43**
	PAR2	
	PAR3	
	PAR4	
	PAR5	
	PAR6	
	PAR7	
	PAR8	

Table 3-1 List of commands

Name	Command Number	Operation Description
	Parameters	**Parameter Descriptions**
LOOP	10	Close and Open Loops for circuit switch TCH26 connections (FR, HR, EFR, AMR and data services)
	SWITCH	0: Open Loop 1: Signalling FER *TCH Loop including signalling of erased frames* (Loop A) 2: No Signalling FER *Speech TCH loop without signalling of erased frames* (Loop B). 3: Signalling erased and unreliable frames (HS only) *TCH loop including Signalling erased and unreliable* frames (Loop D) 4: Signalling erased SID frames (HS only) *TCH loop including Signalling erased SID frames* (Loop E) 5: Signalling erased valid SID frames (HS only) *TCH Loop including Signalling erased valid SID frames* (Loop F) 6: *TCH Burst-by-Burst Loop* (Loop C) 7: *TCH Loop without signalling of erased frames for* in-band channel error rate (Loop I) (AMR only)
	UL-SRC#0	Source for Uplink Substream #0 (0 ... 3)
	UL-SRC#1	Source for Uplink Substream #1 (0 ... 2)
PDCH	11	Start or modify the Packet Data Channel (PDCH) mode (only in the first TDMA frame of a radio block)
	TSC	Training Sequence Number
	MODE	0 = GPRS
BB_OFF	13	Leave Modem related algorithms and go to Idle state (Audio unaffected). Refer to **Section 3.3.12 “BB_OFF” on Page 46**.
IDLE	14	Same as command **BB_OFF**. Additionally all algorithms running are stopped and go to Idle state. The Audio scheduler sample based is not influenced. Refer to **Section 3.3.13 “IDLE” on Page 48**.
VB_ON	15	Configuration of the I^2S_x interface and the AFE.

Table 3-1 List of commands

Name	Command Number	Operation Description
	Parameters	**Parameter Descriptions**
	SWITCH	0: Switch AFE off and I^2S_x off 1: Copy parameters VRXCTRL1, VRXCTRL2, and VTXCTRL to AFE hardware. 2: Switch AFE in/out on. Switch I^2S_x off. 3: Switch I^2S_x on and AFE off. 4: Switch I^2S_x on and AFE out on. 5: Switch I^2S_x on, AFE in on, and AFE out off 6: Switch I^2S_x on, AFE in on, and AFE out on 7: Copy parameters VRXCTRL1, VRXCTRL2, VTXCTRL, and OUT_MODE to AFE hardware 8: Switch I^2S_2 on for external audio output, AFE in on, and AFE out off 9: Switch I^2S_2 on for external audio output, AFE in and out on.
	VRXCTRL1	Receive Control 1 register same as hardware register AFE_VRXCTRL1, refer to **[1]**
	VRXCTRL2	Receive Control 2 register same as hardware register AFE_VRXCTRL2, refer to **[1]**
	VTXCTRL	Transmit Control register same as hardware register AFE_VTXCTRL, refer to **[1]**
	RATESW	0: Audio scheduler sample rate 8 kHz (must be used in combination with I^2S_x) 1: Audio scheduler sample rate 16 kHz
	OUT_MODE	0: AFE and external audio by I^2S_2 output path mode set to mono 1: AFE and external audio by I^2S_2 output path mode set to stereo
	CSEL	Clock select register, refer to **[1]**
	NUM0	Divider 0 numerator register, refer to **[1]**
	DEN0	Divider 0 denominator register, refer to **[1]**
	NUM1	Divider 1 numerator register, refer to **[1]**
	DEN1	Divider 1 denominator register, refer to **[1]**
	RXCONF	Receiver configuration register, refer to **[1]**
	TXCONF	Transmitter configuration register, refer to **[1]**
VB_SET_BIQUAD	16	Setup biquad filter coefficients
	PAR 1-20	1-5: Biquad In 1 6-10: Biquad In 2 11-15: Biquad Out 1 16-20: Biquad Out 2

Table 3-1 List of commands

Name	Command Number	Operation Description	
	Parameters	**Parameter Descriptions**	
VB_SET_GAIN	17	Setup voiceband gains	
		Default (reset)	Format: q15, range: [0,0x7FFF] 0 dB corresponds to the following values
	Scal_In	0x2000	0x1FFF
	Scal_Out	0x2000	0x1FFF
	Side_Ton	0x2000	0x3FFF
	Mic_Mute	0x0000	0x0000
	Scal_Mic	0x2000	0x1FFF
	Gain_Out	0x2000	0x1FFF
	Scal_Rec	0x2000	0x1FFF
	Speech_Mix_DL	0x1000	0x7FFF
	Ton_Mix	0x0000	0x7FFF
	Delta0	0x0000	0x7FFF
	Delta1	0x0000	0x7FFF
	Kappa0	0x7FFF	0x7FFF
	Kappa1	0x0000	0x7FFF
	Lambda0	0x7FFF	0x7FFF
	Lambda1	0x0000	0x7FFF
	Scal_AFE	0x0000	0x1FFF
	Scal_Mic2	0x0000	0x1FFF
	AFE_tone	0x0000	0x3FFF
	Ton_Mix_UL	0x0000	0x7FFF
	Ton_Mix_DL	0x0000	0x7FFF
	Speech_Mix_UL	0x1000	0x7FFF
VB_START_TONE	18	Start a new Tone (frequency, amplitude duration, and interrupt value are read from fixed locations in the shared memory). Tone is started if duration parameter is not equal to “0”	
VB_STOP_TONE	19	Emergency Stop for Tone Generation (Normally Tone Generation is stopped by not reactivating the above mentioned DURATION Parameter)	
VB_READ_DURATION	20	Read DSP internal DURATION Parameter and write it to shared memory location **SM_TONE_DUR_OUT**	
VB_RESET	21	Reset Voiceband(Reset States of Voiceband Filters and switch off Tone Generation)	

Table 3-1 List of commands

Name	Command Number	Operation Description
	Parameters	**Parameter Descriptions**
VB_DAI	22	Configuration of DAI mode
	MODE	0: Normal mode 1: Vocoder test 2: Acoustic test 3: Voiceband test
HF_SET_PAR	23	Setup Handsfree Parameter
	PAR1	GAIN_ANALOG
	PAR2	STEP_WIDTH
	PAR3	LMS_LENGTH
	PAR4	LMS_OFFSET
	PAR5	BLOCK_LENGTH
	PAR6	RXTX_RELATION
	PAR7	NR_SW_2
	PAR8	NR_U_Fak_0
	PAR9	NR_U_Fak
	PAR10	ADD_ATTEN
	PAR11	MIN_ATTEN
	PAR12	MAX_ATTEN
HF_ON	24	Switch Handsfree on and off
	SWITCH	0x0000: Handsfree is switched off Bit #0 set: Echo Canceller (EC) initialization Bit #1 set: EC restart (without coefficient initialization) Bit #2 set: EC on Bit #3 set: EC adaptation on Bit #4 set: Noise reduction initialization Bit #5 set: Noise reduction on Bit #6 set: Noise reduction works with additional AGC Bit #7 set: Automatic Gain Control (AGC) initialization Bit #8 set: AGC on Setting the bits is not mutually exclusive, more than one bit can be set at the same tine.

Table 3-1 List of commands

Name	Command Number	Operation Description	
	Parameters	**Parameter Descriptions**	
VM_CMD	25	Configure Voice Memo	
	VM_MODE	0: Switch off Voice Memo mode Other: For switching on Voice Memo refer to **Section 3.3.24 "VM_CMD" on Page 52**	
		Default (reset)	Format: q15, range: [0,0x7FFF] 0 dB corresponds to the following values
	alpha0	0x0000	0x7FFF
	alpha1	0x0000	0x7FFF
	beta0	0x7FFF	0x7FFF
	beta1	0x0000	0x7FFF
	gamma0	0x7FFF	0x7FFF
	gamma1	0x0000	0x7FFF
VB_SET_CBUF_GAIN	26	Setup circular buffer gains	
		Default (reset)	Format: q15, range: [0,0x7FFF] 0 dB corresponds to the following values
	Scal_SAPP	0x7FFF	0x3FFF
	Scal_Ext	0x7FFF	0x3FFF
	Mix_AFE	0x0000	0x3FFF
	Mix_I2Sx	0x0000	0x3FFF
	Scal_PCM	0x7FFF	0x3FFF
DTX_ON	30	Switch DTX on and off	
	SWITCH	0: Switch DTX off 1: Switch DTX on	
PW_DOWN	31	Immediately initiates stand by power down procedure, DSP copies startup code to shared memory, command is only allowed if the DSP is in idle state, after this command no further command is accepted by the DSP so Reset has to be applied.	
WRITE_DSP	32	Write Data Memory in DSP	
	S_ADDR	Source Address in shared memory (offset)	
	D_ADDR	Destination Address in DSP Memory	
	LEN	Length of Block to be written	

Table 3-1 List of commands

Name	Command Number	Operation Description
	Parameters	**Parameter Descriptions**
READ_DSP	33	Read out Data Memory in DSP
	S_ADDR	Source Address in DSP Memory
	D_ADDR	Destination Address in shared memory (offset)
	LEN	Length of Block to be read
WRITE_PROG	34	Write Program Memory in DSP
	S_ADDR	Source Address in shared memory (offset)
	D_ADDR	Destination Address in DSP Memory
	LEN	Length of Block to be written
READ_PROG	35	Read out Program Memory in DSP
	S_ADDR	Source Address in DSP memory
	D_ADDR	Destination Address in shared memory (offset)
	LEN	Length of Block to be written
MCU_INT	36	Enable/disable the PDCH interrupts (USF, channel decoder, RLC decoder, header decoder)
	SWITCH	0: Switch this feature off (default) Other: Switch this feature on
VB_I2Sy	37	Configuration of the I^2S_y interface and DAI mode
	SWITCH	0: Switch I^2S_y Tx and Rx off 1: Switch I^2S_y Tx and Rx on 2: Switch I^2S_y on and enable DAI mode 3: Switch I^2S_y for external mode (circular buffer)
	CSEL	Clock select register
	NUM0	Divider 0 numerator register
	DEN0	Divider 0 denominator register
	NUM1	Divider 1 numerator register
	DEN1	Divider 1 denominator register
	RXCONF	Receiver configuration register
	TXCONF	Transmitter configuration register
	I2Sy_RATE	Data sampling rate (refer to **Table 5-30 "Input Sampling Rate Index Values" on Page 134**)
	I2Sy_MODE	Data mode mono/dual mono/stereo

Table 3-1 List of commands

Name	Command Number	Operation Description	
	Parameters	**Parameter Descriptions**	
TTY_CTM	38	Setup TTY-CTM	
	SWITCH	0: Switch off 1: Switch on	
	NO_NEG	0: Switch on negotiation phase between UL and DL (Default). 1: Switch off negotiation phase (useful for testing with 3GPP scripts)	
	AUT_PREAMB	0: TTY preamble is always used to path loudspeaker/TTY device (only useful in test mode) 1: TTY preamble is used depending if the TTY device connected to the microphone path uses a preamble	
VB_SYNC	39	Control of voiceband synchronization	
	SWITCH	0: No synchronization required 1: Synchronization required only for DL 2: Synchronization required only for UL 3: Synchronization required for UL and DL	
	SYNC	1: First synchronization 2: Re-synchronization	
	PTR_VAL	Voiceband Write/Read pointer	
	SYNC_LIM	Synchronization limit	
UMTS_ON	40	Start UMTS speech support	
MP3	41	Controls MP3 decoder	
	SWITCH	0: Switch off MP3 1: Switch on and initialize MP3 2: Get the first part of the input frame (1 ... 360 words) and copy them to the first part of the internal buffer 3: Get the second part of the input frame (361 ... 720 words) and copy them to the second part of the internal buffer	
	PAR1	If switch = 1	Sample rate index (refer to **Table 5-30 “Input Sampling Rate Index Values” on Page 134**)
		If switch = 2/3	Pointer (offset) to the first word of the input frame located in shared memory

Table 3-1 List of commands

Name	Command Number	Operation Description	
	Parameters	**Parameter Descriptions**	
SYNTH	42	Controls Synthesizer	
	SWITCH	0: Switch off Synthesizer 1: Switch on and init Synthesizer 2: Feed a new frame (maximum 65 words, 20 ms frame-based) to DSP via shared memory	
	PAR 1	If switch = 1	Sample rate index (refer to **Table 5-30 "Input Sampling Rate Index Values" on Page 134**)
		If switch = 2	Headroom (number of applied right shifts for each generator output)
	PAR 2	If switch = 1	Par 2 not used
		If switch = 2	Pointer (offset) to the first word of the input frame located in DSP shared memory
RF_ADAPT	43	RF adaptations and adjustments of the adaptive FW filter	
	BB_NB_FILTER_SWITCH	Switch on/off the adaptive filter for a Normal Burst (NB) <= 0: NB filter or decimation filter is used in adaptive filter, > 0: NB filter is always used Default: 0	
	BB_ADAPT_THRES	Change threshold for 'DCS/PCS Rx sensitivity' measurements for the function 'CheckPower' in case of a Normal Burst (only valid if BB_NB_FILTER_SWITCH <= 0). *Note: If the BB_NB_FILTER_SWITCH is <= 0 and the BB_ADAPT_THRES = 0 then only the decimation is done.* Default: 0.7 (0x02CD)	
	SMARTIPATCH	Initialization of the NB filter: 0: Off 1: On Default: 1	
	BB_CTRL	Setting of the Baseband Filter Control Register The init value after HW reset is 0x0110. For detailed information, refer to the E-GOLDradio Design Specification.	
	UPDATE	If Update != 0, the filter coefficients of the FW narrow band filter are updated.	
	PAR0	Filter coefficients of the FW narrow band filter	
	...		
	PAR12		

Table 3-1 List of commands

Name	Command Number	Operation Description
	Parameters	**Parameter Descriptions**
AUDIOPOSTPROC	44	Switch on/off the postprocessing for Synthesizer and change the coefficients of the high frequency shelving filter and the parameters of the audio compressor.
	AudioPostProc Switch	Bit_0: Enable the shelving filter 0: Off 1: On (Default) Bit_1: Enable the audio compressor 0: Off 1: On (Default) Bit_2: Update filter coefficients and parameters of audio compressor: 0: Off 1: On (Default)
	b_exp	High Frequency Shelving Filter Coefficients (Refer to **Section 3.3.40 "AUDIOPOSTPROC" on Page 60** for default values at 16 and 32 kHz)
	b1	
	b0	
	a1	
	INITDATA_mono_flag	Audio Compressor Parameters (Refer to **Section 3.3.40 "AUDIOPOSTPROC" on Page 60** for default values at 16 and 32 kHz)
	INITDATA_m_bufflen	
	INITDATA_m_inv_bufflen	
	INITDATA_m_hp_coeff_exp	
	INITDATA_mlp1_coef	
	INITDATA_mlp2_coef	
	INITDATA_mlp4_coef	
	INITDATA_mlp3_coef	
	INITDATA_m_L_A	
	INITDATA_m_L_B	
	INITDATA_m_G_Comp	
	INITDATA_um_R_infA	
	INITDATA_um_R_AB	
	INITDATA_um_R_B0	

Table 3-1 List of commands

Name	Command Number	Operation Description	
	Parameters	**Parameter Descriptions**	
PCMPLAY	45	Controls PCM Player	
	SWITCH	0: Switch off PCM Player 1: Switch on and init PCM Player 2: Feed new PCM data to DSP via shared memory	
	PAR 1	If switch = 1	0 PCM Mode 16-bit format 1 ADPCM Mode 2 PCM Mode 8-bit format
		If switch = 2	Depends om PAR 3: Number of sample pairs (stereo, dual mono) or number of samples (mono)
	PAR 2	If switch = 1	Sample rate index, refer to **Table 5-30 "Input Sampling Rate Index Values" on Page 134**
		If switch = 2	Not used
	PAR 3	If switch = 1	0: Mono 1: Dual Mono 2: Stereo
		If switch = 2	Not used
DTW	46	Controls Speech Recognition algorithm	
	SWITCH	0: Initialization 1: Normalized pattern computation 2: Distance calculation 3: Adaptation of one reference	
TX_DIG	47	Initialization of TX path	
	SWITCH	0: Analog TX path 1: Digital TX path (Default)	
I2S_SWAP	48	I^2S_1 and I^2S_2 Swap	
	SWITCH	0: I^2S_x functionnalities are mapped on I^2S_1 and I^2S_y functionnalities are mapped on I^2S_2 (Default). 1: I^2S_x functionnalities are mapped on I^2S_2 and I^2S_y functionnalities are mapped on I^2S_1.	
USER_15	49	User Function 15	
USER_14	50	User Function 14	
USER_13	51	User Function 13	
USER_12	52	User Function 12	

Table 3-1 List of commands

Name	Command Number	Operation Description
	Parameters	**Parameter Descriptions**
USER_11	53	User Function 11
USER_10	54	User Function 10
USER_9	55	User Function 9
USER_8	56	User Function 8
USER_7	57	User Function 7
USER_6	58	User Function 6
USER_5	59	User Function 5
USER_4	60	User Function 4
USER_3	61	User Function 3
USER_2	62	User Function 2
USER_1	63	User Function 1
USER_0	64	User Function 0

3.3 Command Description

This section gives a detailed description of the commands listed in **Table 3.2 List of Commands**.

3.3.1 FC_INIT

This command changes the mode and threshold values for FCB search. Frequency evaluation and RMS value calculation are done in every mode.

- MODE (Default value: 0 / Range: 0,1)
  0. This mode is recommended for the mobile synchronization phase. The maximum deviation between carrier frequencies of the mobile and base station that can be measured is ±**15 kHz** under normal conditions. In ideal conditions (GSM tester) the range for a successful FCB search is ±**25 kHz,** but this value cannot be guaranteed under real conditions.
  1. This mode is recommended for neighbor cell monitoring. The maximum deviation between carrier frequencies of the mobile and base station that can be measured is ±**2 kHz**.
- THRSH (Default value: 125 / Range: THRSH <= 143 (146))

  – THRSH is a quality measure for FCB bursts. If THRSH is less than 120 FCB, bursts can be found even with bad receive conditions, but there is a chance of false detections. If THRSH is larger than 140 FCB, bursts can be found only with good receive conditions, but there is almost no spurious detections.
- HYST (Default value: 30 / Range: 1 <= HYST <= 35)
  – HYST is the hysteresis value for the FCB search window. As soon as a new FC burst has been found the FCB algorithm waits for HYST more samples. If the quality of the detected FC burst gets better within these HYST samples, the position of the burst is re-adapted.

Note: Frequency evaluation and RMS value calculations are done in both modes.

Note: The maximum value for THRSH with 'Mode = 0' is 146, the maximum value for THRSH with 'Mode = 1' is 143.

Recommended parameter values

- MODE = 0:
  – HYST = 30
  – THRSH:
  a) 125 good FCB detections
  b) 140 good FCB detections only with very good signals
- MODE = 1:
  – HYST = 30
  – THRSH:
  a) 115 good FCB detections
  b) 130 good FCB detections only with very good signals

3.3.2 MODU_INIT

This command sets up certain modulator related values.

- IOFFS I-Offset Pre-Correction for the GMSK Modulator
- QOFFS Q-Offset Pre-Correction for the GMSK Modulator
- IAMP I-Amplitude Pre-Correction for the GMSK Modulator
- QAMP Q-Amplitude Pre-Correction for the GMSK Modulator
- GMSK_FREQ Frequency Pre-Correction for the GMSK Modulator
- DUMMY Number of dummy symbols (DUMMY must be > 0).
  Whenever the modulator is started by the System Timer (rising edge of signal CODON) the modulator transmits DUMMY symbols followed by the first burst, see **Figure 4-4 "Dummy Symbols" on Page 69**. This is independent of the current DSP mode. There are no dummy symbols between two subsequently transmitted bursts.
- IQ_SETUP IQ Swap for the GMSK Modulator
  – 0: Rx swap is equal to Tx swap.
  – 1: additional Tx swap due to hardware RF interfaces.

By default, all these values are set to '0' after reset except of DUMMY which is set to 21.

3.3.3 IQ_SWAP_1

This command changes the IQ-Swap for Normal Burst Rx and Tx.

- SWITCH:
  – 0: IQ-Swap is off
  – 1: IQ-Swap is on

3.3.4 IQ_SWAP_2

This command changes the IQ-Swap for FCB Search and Sync Burst Detection.

- SWITCH:
  - 0: IQ-Swap is off
  - 1: IQ-Swap is on

3.3.5 DEC_INIT

This command must be used to initialize certain channel decoder thresholds for all speech channels - Fullrate, Enhanced-Fullrate, Halfrate, and AMR:

- SWITCH
  0: Fullrate
  1: Enhanced Fullrate (EFR)
  2: Halfrate (HS)
  3: Adaptive Multi Rate (AMR).
- PAR 0...25 A description of the thresholds for Halfrate, Fullrate, and Enhanced Fullrate is given in **Table 3-2**, for AMR see **Table 3-4 “PAR 0 If SWITCH = 3 (AMR)” on Page 37**.

Table 3-2 PAR 0...11 If SWITCH = 0, 1, or 2

PAR	Name	Description
0	SID1	Threshold for setting SID Parameter (input to speech decoder) to ’1’: If a received frame differs from a SID frame (only the bits in the SID field) in less than SID1 bit positions but in more than SID2 bit positions, the SID parameter is set to ’1’.
1	SID2	Threshold for setting SID Parameter (input to speech decoder) to ’2’: If a received frame differs from a SID frame in less than SID2 bit positions, the SID parameter is set to ’2’.
2	BFI	Threshold for setting BFI Flag (input to speech decoder) to ’1’: If more than BFI bit errors (Decoder Metric > BFI) have been found in the channel decoder run, the BFI Flag is set to ’1’.
3	UFI	Threshold for setting UFI Flag (input to halfrate speech decoder) to ’1’: If more than UFI bit errors (Decoder Metric > UFI) have been found in the channel decoder run, the UFI Flag is set to ’1’. The UFI Flag is always set to ’1’ if BFI Flag is set to ’1’. (This UFI parameter is needed for halfrate only. In the case of fullrate and enhanced fullrate this parameter is don’t care.)
4	XRandRF	Threshold for setting the BFI and UFI Flags (input to speech decoder) to '1': If the high resolution metric of the received frame is less or equal than the XRandomRF threshold, the BFI and the UFI flags are set to '1'. Valid only in case of random RF. Range: 0...3165 for halfrate speech Range: 0...5670 for fullrate speech
5	YRandRF	Threshold to detect the random RF case (according to GSM 11.10 standard, §14.1.2.1): If the accumulated high resolution metric of the previous 30 frames is less or equal than the YRandomRF threshold, the XRandomRF threshold is valid for BFI/UFI setting, otherwise the BFI/UFI thresholds for normal RF are valid for setting the BFI/UFI flags. Range: 0...11868 for halfrate speech Range: 0...21262 for fullrate speech
6	Same meaning as PAR 0 but this parameter is used for loop-back mode	
7	Same meaning as PAR 1 but this parameter is used for loop-back mode	
8	Same meaning as PAR 2 but this parameter is used for loop-back mode	
9	Same meaning as PAR 3 but this parameter is used for loop-back mode	
10	Same meaning as PAR 4 but this parameter is used for loop-back mode	
11	Same meaning as PAR 5 but this parameter is used for loop-back mode	

Sets of recommended parameter values are given in Table 3-3.

Table 3-3 Recommended Parameter Values for E-GOLDradio

Parameters	Fullrate Enhanced Fullrate	Halfrate
SID1	16	18
SID2	2	3
BFI	5365	2950
UFI	Don't care, but must passed to MCU	2990
XRandRF	5300	3050
YRandRF	19000	11000
SID1 (loop-back)	16	18
SID2 (loop-back)	2	3
BFI (loop-back)	5365	2950
UFI (loop-back)	Don't care, but must passed to MCU	2990
XRandRF	5300	3050
YRandRF	19000	11000

Note: These six parameters (SID1, SID2, BFI, UFI, XRandRF, and YRandRF) have to be given twice. Once for normal operation mode and once for the loop-back mode.

Table 3-4 PAR 0 If SWITCH = 3 (AMR)

Value of PAR 0 = Sub Type	Description	Refer to
0	Afs_Marker_Thrsh,	Table 3-5, Page 38
1	Ahs_Marker_Thrsh,	Table 3-6, Page 38
2	Afs_Metric_Thrsh (low),	Table 3-7, Page 38
3	Afs_Metric_Thrsh (high),	Table 3-8, Page 38
4	Ahs_Metric_Thrsh (low),	Table 3-9, Page 39
5	Ahs_Metric_Thrsh (high),	Table 3-10, Page 39
6	SU_Metric_Thrsh,	Table 3-11, Page 39
7	Inband_Thrsh,	Table 3-12, Page 39
8	AFS_eBFI_PARAM,	Table 3-13, Page 40
9	AHS_eBFI_PARAM,	Table 3-14, Page 41

Table 3-5 Afs_Marker_Thrsh (Sub Type = 0)

PAR	Meaning	Actual Value	Maximum Value
1	Sid_First	4134	6360
2	Sid_Update	4134	6360
3	Onset	4446	6840
4	RATSCCH	4350	6360

Table 3-6 Ahs_Marker_Thrsh (Sub Type = 1)

PAR	Meaning	Actual Value	Maximum Value
1	Sid_First P1	4770	6360
2	Sid_First INH	4770	6360
3	Sid_First P2	2385	3420
4	Sid_Update	4770	6360
5	Sid_Update INH	4770	6360
6	Onset	2385	3420
7	RATSCCH	4350	6360

Table 3-7 Afs_Metric_Thrsh low (Sub Type = 2)

PAR	Meaning	Actual Value	Maximum Value
1	AFS 4,75 (Thrsh1)	116	535
2	AFS 4,75 (Thrsh2)	535	535
3	AFS 5,15 (Thrsh1)	114	565
4	AFS 5,15 (Thrsh2)	565	565
5	AFS 5,90 (Thrsh1)	107	520
6	AFS 5,90 (Thrsh2)	520	520
7	AFS 6,70 (Thrsh1)	103	576
8	AFS 6,70 (Thrsh2)	576	576

Table 3-8 Afs_Metric_Thrsh High (Sub Type = 3)

PAR	Meaning	Actual Value	Maximum Value
1	AFS 7,40 (Thrsh1)	98	474
2	AFS 7,40 (Thrsh2)	474	474
3	AFS 7,95 (Thrsh1)	93	513
4	AFS 7,95 (Thrsh2)	513	513
5	AFS 10,2 (Thrsh1)	83	642
6	AFS 10,2 (Thrsh2)	642	642
7	AFS 12,2 (Thrsh1)	77	508
8	AFS 12,2 (Thrsh2)	508	508

Table 3-9 Ahs_Metric_Thrsh Low (Sub Type = 4)

PAR	Meaning	Actual Value	Maximum Value
1	AHS 4,75 (Thrsh1)	37	285
2	AHS 4,75 (Thrsh2)	285	285
3	AHS 5,15 (Thrsh1)	36	303
4	AHS 5,15 (Thrsh2)	303	303
5	AHS 5,90 (Thrsh1)	32	224
6	AHS 5,90 (Thrsh2)	224	224

Table 3-10 Ahs_Metric_Thrsh High (Sub Type = 5)

PAR	Meaning	Actual Value	Maximum Value
1	AHS 6,70 (Thrsh1)	28	240
2	AHS 6,70 (Thrsh2)	240	240
3	AHS 7,40 (Thrsh1)	26	260
4	AHS 7,40 (Thrsh2)	260	260
5	AHS 7,95 (Thrsh1)	23	266
6	AHS 7,95 (Thrsh2)	266	266

Table 3-11 SU_Metric_Thrsh (Sub Type = 6)

PAR	Meaning	Actual Value	Maximum Value
1	Sid Update Metric	212	212

Table 3-12 Inband_Thrsh (Sub Type = 7)

PAR	Meaning	Actual Value	Maximum Value
1	AFS_INBAND_THRSH	0	960 + LA
2	AHS_INBAND_THRSH	0	480 + LA
3	SID_INBAND_THRSH	0	480

Table 3-13 Afs_eBFI_PARAM (Sub Type = 8)

PAR	Meaning	Actual Value	Maximum Value
1	AFS 4,75 NormalRF BFI (THRSH)	5685	8025
2	AFS 4,75 RandomRF BFI (THRSH)	7020	8025
3	AFS 4,75 Hi ResMetric Scaling Factor	8350	32767
4	AFS 5,15 NormalRF BFI (THRSH)	5700	8475
5	AFS 5,15 RandomRF BFI (THRSH)	7020	8475
6	AFS 5,15 Hi ResMetric Scaling Factor	8250	32767
7	AFS 5,90 NormalRF BFI (THRSH)	5700	7800
8	AFS 5,90 RandomRF BFI (THRSH)	6850	7800
9	AFS 5,90 Hi ResMetric Scaling Factor	7855	32767
10	AFS 6,70 NormalRF BFI (THRSH)	5710	8640
11	AFS 6,70 RandomRF BFI (THRSH)	6650	8640
12	AFS 6,70 Hi ResMetric Scaling Factor	7685	32767
13	AFS 7,40 NormalRF BFI (THRSH)	5850	7110
14	AFS 7,40 RandomRF BFI (THRSH)	6650	7110
15	AFS 7,40 Hi ResMetric Scaling Factor	7455	32767
16	AFS 7,95 NormalRF BFI (THRSH)	5850	7695
17	AFS 7,95 RandomRF BFI (THRSH)	6555	7695
18	AFS 7,95 Hi ResMetric Scaling Factor	7265	32767
19	AFS 10,2 NormalRF BFI (THRSH)	5880	9630
20	AFS 10,2 RandomRF BFI (THRSH)	6425	9630
21	AFS 10,2 Hi ResMetric Scaling Factor	6950	32767
22	AFS 12,2 NormalRF BFI (THRSH)	5940	7620
23	AFS 12,2 RandomRF BFI (THRSH)	6385	7620
24	AFS 12,2 Hi ResMetric Scaling Factor	6720	32767
25	AFS RandomRF Detection (THRSH)	22850	26662

Table 3-14 Ahs_eBFI_PARAM (Sub Type = 9)

PAR	Meaning	Actual Value	Maximum Value
1	AHS 4,75 NormalRF BFI (THRSH)	2840	4275
2	AHS 4,75 RandomRF BFI (THRSH)	3210	4275
3	AHS 4,75 Hi ResMetric Scaling Factor	3470	32767
4	AHS 5,15 NormalRF BFI (THRSH)	2850	4545
5	AHS 5,15 RandomRF BFI (THRSH)	3210	4545
6	AHS 5,15 Hi ResMetric Scaling Factor	3435	32767
7	AHS 5,90 NormalRF BFI (THRSH)	2860	3360
8	AHS 5,90 RandomRF BFI (THRSH)	3200	3360
9	AHS 5,90 Hi ResMetric Scaling Factor	3340	32767
10	AHS 6,70 NormalRF BFI (THRSH)	2870	3600
11	AHS 6,70 RandomRF BFI (THRSH)	3200	3600
12	AHS 6,70 Hi ResMetric Scaling Factor	3270	32767
13	AHS 7,40 NormalRF BFI (THRSH)	2900	3900
14	AHS 7,40 RandomRF BFI (THRSH)	3150	3900
15	AHS 7,40 Hi ResMetric Scaling Factor	3205	32767
16	AHS 7,95 NormalRF BFI (THRSH)	2920	3990
17	AHS 7,95 RandomRF BFI (THRSH)	3150	3990
18	AHS 7,95 Hi ResMetric Scaling Factor	3180	32767
19	AHS RandomRF Detection (THRSH)	11220	12600

3.3.6 CIPH_KEY

This command is used to change the Cipher Mode and to set (or to modify) the Cipher Keys. It can be used either in traffic channel or in idle state:

- SWITCH Action to be done:
  1. Set Cipher Mode
  2. Set Cipher Keys
  3. Set Cipher Mode and Cipher Keys
- CIPH Cipher Mode:
  0. No Ciphering
  1. Ciphering with A51
  2. Ciphering with A52
  3. Ciphering with A53
- SS Substream Index [#0 - #3]
  (If not HSCSD, a ’0’ has to be given here)
- CKEY0-3 Only for A51/A52 Cipher Key := Byte0 - Byte7:
  CKEY0 = Byte1 Byte0
  CKEY1 = Byte3 Byte2
  CKEY2 = Byte5 Byte4
  CKEY3 = Byte7 Byte6
- CKEY0-7 Only for A53 Cipher Key := Byte0 - Byte15:
  CKEY0 = Byte14 Byte15
  CKEY1 = Byte12 Byte13
  CKEY2 = Byte10 Byte11
  CKEY3 = Byte8 Byte9

CKEY4 = Byte6 Byte7
CKEY5 = Byte4 Byte5
CKEY6 = Byte2 Byte3
CKEY7 = Byte0 Byte1

*Note: If the DSP subsystem is in TCH26 mode and only the Cipher Mode is to be changed, this command should be used instead of the **TCH_26** command (Channel Mode Modify command).*

3.3.7 CCH_RX

This command is used to start receiving a Control Channel Block in Mode 51:

- CIPH Type of Ciphering:
  -1: Does not change preset Ciphering mode
  0: No Ciphering
  1: Ciphering with A51
  2: Ciphering with A52
  3: Ciphering with A53
- TSC Training Sequence Number:
  0-7: Corresponds to the indices of GSM 45.002 standard.
- EARLY Early start of the channel decoder:
  This parameter defines if the channel decoder is started after receiving the 2nd and 3rd burst. If the channel decoder procedure has been successful, the MCU may switch off the baseband functions for power consumption.
  0: No early start. The Channel Decoder is started after 4th burst.
  1: Channel Decoder is started after 2nd and 3rd burst.

*Note: This command is also used for a GPRS mobile which camps on the CCCH. It can be used if a GPRS mobile is on PCCCH (52 multiframe structure) and the mobile only has to listen to the Packet Paging Channels (PPCH) and Packet Broadcast Channels (PBCCH). In this case, the **CCH_RX.CIPH** parameter has to be set to 0.*

*Note: In any cases the first received burst is written to the CCH_RX_DATA location (refer to **Control Channel Receive on CCCH (Mode 51) or PCCCH**) and the communication-flag #7 is set to '1' as soon as the burst is complete. This process enables very fast CCH decoding from the MCU side.*

3.3.8 CCH_TX

This command is used to start transmitting a Control Channel Block in Mode 51:

- CIPH Type of Ciphering:
  -1: Does not change preset Ciphering mode
  0: No Ciphering
  1: Ciphering with A51
  2: Ciphering with A52
  3: Ciphering with A53
- TSC Training Sequence Number:
  0-7 corresponds to the indices of GSM 45.002 standard.

Note: This command is also used for a GPRS mobile which camps on the CCCH.

3.3.9 TCH_26

This command can be used to start or to modify a TCH 26 Channel:

- CHT Channel Type:
  0. Signalling Only Fullrate
  1. Fullrate Speech
  2. Enhanced Fullrate Speech
  3. Adaptive Multi Rate Full Speech (AFS)
  4. Fullrate Data 2.4kBit/s
  5. Fullrate Data 4.8kBit/s
  6. Fullrate Data 9.6kBit/s
  7. Fullrate Data 14.4kBit/s
  8. Signalling Only Halfrate
  9. Halfrate Speech
  10. Adaptive Multi Rate Half Speech (AHS)
  11. Halfrate Data 2.4kBit/s
  12. Halfrate Data 4.8kBit/s
  13. HSCSD 4.8kBit/s
  14. HSCSD 9.6kBit/s
  15. HSCSD 14.4kBit/s
- CIPH Type of Ciphering:
  -1. Does not change preset Ciphering mode
  0. No Ciphering
  1. Ciphering with A51
  2. Ciphering with A52
  3. Ciphering with A53
- TSC Training Sequence Number:
  0-7 corresponds to the indices of GSM 45.002.
- INIT Initialization Switch
  By means of this flag the user can decide which bits of the interleaving and de-interleaving memory are initialized. Details are given in **Table 3-17 “Bits of the Initialization Switch” on Page 45**.
- PAR1..8 Additional Parameters #1 ... #8
  The meanings of these additional parameters PAR1 .. PAR8 depend on the channel type, refer to **Table 3-15 “Additional Parameters PAR1 - PAR8” on Page 44**.

Table 3-15 Additional Parameters PAR1 - PAR8

Channel Type (0 ... 15)	PAR1	PAR2	PAR3	PAR4	PAR5	PAR6	PAR7	PAR8
Signalling Only Fullrate	TS							
TCH/FS	TS							
TCH/EFR	TS							
TCH/AFS	TS							
TCH/F2.4	TS	Mode						
TCH/F4.8	TS	Mode						
TCH/F9.6	TS	Mode						
TCH/F14.4	TS	Mode						
Signalling Only Halfrate	TS		Sub					
TCH/HS	TS		Sub					
TCH/AHS	TS		Sub					
TCH/H2.4	TS	Mode	Sub					
TCH/H4.8	TS	Mode	Sub					
HSCSD 4.8		Mode	RX 0	RX 1	RX 2	RX 3	TX 0	TX 1
HSCSD 9.6		Mode	RX 0	RX 1	RX 2	RX 3	TX 0	TX 1
HSCSD 14.4		Mode	RX 0	RX 1	RX 2	RX 3	TX 0	TX 1

These parameters are defined as:

- TS Time Slot [#0 - #7]
- Mode 0: Transparent data
  1: Transparent fax without bit reversion
  2: Transparent fax with bit reversion
  3: Non transparent data
  4: Data is passed directly to and from Channel Codec
- Sub Sub-Channel Number for Halfrate [#0, #1]

The parameters RX0/1/2/3 and TX0/1 are only used for HSCSD channel types. The following has to be taken into account:

- The parameter TX1 has to be set to “-1” if only one uplink timeslot is allocated. All parameters that are not used have to be set to “-1”.
- RX0 is used for the first timeslot where the EQON window is set, RX1 for the second timeslot, and so on. The EQON signal from the system interface has to be continued between two allocated timeslots even if there is a gap of one or two timeslots in between.
- The EQON signal can be cleared after the last allocated timeslot.

Table 3-16 Values of Parameters RX 0..3 and TX 0/1

Value	Meaning for this Timeslot
-1	no allocation on this timeslot
0	mainstream
1	substream index 1
2	substream index 2
3	substream index 3

More information about how to use the **TCH_26** is given in **Section 4.11.5 “Data Channels” on Page 83**.

.

Table 3-17 Bits of the Initialization Switch

Bit Number of INIT Flag	Initialization of Interleaving and De-interleaving Data (if Bit is set)
#0	Sacch Data on Mainstream (Uplink and Downlink)
#1	Facch Data on Mainstream (Uplink and Downlink)
#2	Tch Data on Mainstream (Uplink and Downlink)
#3	Sacch Data on Substream #1 Downlink
#4	Tch Data on Substream #1 Downlink
#5	Sacch Data on Substream #2 Downlink
#6	Tch Data on Substream #2 Downlink
#7	Sacch Data on Substream #3 Downlink
#8	Tch Data on Substream #3 Downlink
#9	Sacch Data on Substream #1 Uplink
#10	Tch Data on Substream #1 Uplink

3.3.10 LOOP

This command is used to open and close test loops for Type Approval testing:

- SWITCH
  0. Open Loop
  1. Signalling FER [*TCH loop including signalling of erased frames* (LOOP A)]
  2. No Signalling FER [*TCH loop without signalling of erased frames* (LOOP B)]
  3. Signalling erased and unreliable frames (Halfrate Speech only)
     [*TCH loop including Signalling erased and unreliable frames* (Loop D)]
  4. Signalling erased SID frames (Halfrate Speech only)
     [*TCH loop including Signalling erased SID frames* (Loop E)]
  5. Signalling erased valid SID frames (Halfrate Speech only)
     [*TCH Loop including Signalling erased valid SID frames* (Loop F)]
  6. TCH Burst-by-Burst Loop (Loop C)
  7. TCH loop without signalling of erased frames for inband channel error rate (AMR only)
- UL_SRC#0 Source for uplink substream #0.
  The Index of that downlink substream [#0...#3] that should be looped back to uplink substream #0 has to be given here.
  Notes:
  1.For non-HSCSD channels always a '0' is given here.
  2.If no loop-back is wanted for uplink substream #0, a '-1' has to be given here.
- UL_SRC#1 Source for uplink substream #1.
  The Index of that downlink substream [#0...#2] that should be looped back to uplink substream #1 has to be given here.
  Notes:
  1.For non-HSCSD channels or when uplink substream #1 is not active this parameter is don't care.
  2.If no loop-back is wanted for uplink substream #1, a '-1' has to be given here.
  3.According GSM 44.014 standard the round trip delay (RTD) in case of Burst-by-Burst Loop is 12 for fullrate and 16 for halfrate.

3.3.11 PDCH

This command is used to start the scheduler for the Packet Data Channel mode:

- TSC Training Sequence Number:
  0-7: Corresponds to the indices of GSM 45.002 standard
- MODE GPRS mode
  0: GPRS (No other value allowed as E-GOLDradio does not support EGPRS)

3.3.12 BB_OFF

The Operating System of the DSP has five interruptible task levels with different priorities and one interrupt level. The different levels are called Level Interrupt, Level A, Level B, Level C, Level D and Level E. The Level Interrupt has the highest priority and is not interruptible by tasks of a lower task level. From Level A to Level E the priority is decreasing. **Table 3-15** shows which task belongs to which task level.

Table 3-18 Task Level

Task	Level Interrupt	Level A	Level B	Level C	Level D	Level E
Audio Scheduler Sample Based and I^2S MMS mode	x					
Hardware peripherals (AFE,I^2S_1,I^2S_2)	x					
Asynchronous commands	x					
Synchronous commands		x				
Frame Task		x				
Equalizer			x			
FCB / Sync / Monitoring			x			
PDCH Encoder				x		
Channel CODECS				x		
Data Services				x		
Audio Scheduler Frame Based						x
Speech CODECS						x
MP3						x
TTY-CTM						x
Handsfree						x
Synthesizer						x
Audio Scheduler Circular Buffer					x	
Voice Memo						x
I^2S External Mode					x	
PCM Player						x
DTW Speech Recognizer						x

The command **BB_OFF** is used to switch off immediately the baseband related algorithms and to force the DSP subsystem into the IDLE state. It kills all running tasks with the same or a higher priority than Level C except for the Level Interrupt.

This means that:

- CCH_RX, CCH_TX, PDCH, or TCH26 mode is switched off if it is in use.
- Latched asynchronous commands that have to be executed during the next Frame Interrupt are deleted.
- The current and pending equalizers are killed
- FCB search, Synch, and monitoring are killed
- All channel CODECS are interrupted and killed
- Voiceband is not influenced
- Voice Memo, and speech CODECS are not influenced and still run
- Ciphering is switched off
- BER loop is opened

This command is always recommended to use for the transition from:

- PDCH to Idle
- Tch26 Signaling Only to Idle.

Note: The hardware peripherals AFE, I^2S_1 and I^2S_2 are not switched off.

3.3.13 IDLE

This command is used to force the DSP into the Idle Mode. The command IDLE kills all running tasks with the same or a higher priority than Level E, except for the Level Interrupt. Differences with the command **BB_OFF** are:

- All Speech CODECs are killed.
- All Audio Applications (such as MP3, Voice Memo, Synthesizer, Speech Recognizer, TTY/CTM, Handsfree, and I^2S_y external mode) are terminated and switched off.

Immediately after giving this command to the DSP, the DSP subsystem is in IDLE state. This means the DSP core clock is switched off.

Notes:

1. *If the DSP subsystem is in a speech TCH26 state (for example, EFR, HR, FR, or AMR) it is recommended to use this IDLE command.*
2. *The hardware peripherals AFE, I^2S_1 and I^2S_2 are not switched off.*

3.3.14 VB_ON

This command is used to start or to stop the firmware audio scheduler. When the audio scheduler is enabled the AFE input (microphone) path is switch on to trigger the sample based audio scheduler processing. This command is used to control the I^2S_x interface.

Depending on the SWITCH parameter, the command provides seven different functions:

- SWITCH:
  0. Both AFE and I^2S_x interface are switched off.
     No further parameters are used.
  1. No change in AFE state.
     Only the parameters VRXCTRL1, VRXCTRL2, and VTXCTRL are copied to the appropriate AFE registers.
  2. AFE is switched on.
     The parameters VRXCTRL1, VRXCTRL2, and VTXCTRL are copied to the appropriate AFE registers.
  3. AFE is switched off and I^2S_x is switched on.
     The parameters VRXCTRL1, VRXCTRL2, VTXCTRL, CSEL, NUM0, DEN0, NUM1, DEN1, RXCONF, and TXCONF are copied to the appropriate AFE and I^2S_x registers.
  4. AFE output (loudspeaker) and I^2S_x are switched on.
     The parameters VRXCTRL1, VRXCTRL2, VTXCTRL, CSEL, NUM0, DEN0, NUM1, DEN1, RXCONF, and TXCONF are copied to the appropriate AFE and I^2S_x registers.
  5. AFE input (microphone) and I^2S_x are switched on.
     The parameters VRXCTRL1, VRXCTRL2, VTXCTRL, CSEL, NUM0, DEN0, NUM1, DEN1, RXCONF, and TXCONF are copied to the appropriate AFE and I^2S_x registers.
  6. AFE output (loudspeaker) as well as input (microphone) and I^2S_x are switched on.
     The parameters VRXCTRL1, VRXCTRL2, VTXCTRL, CSEL, NUM0, DEN0, NUM1, DEN1, RXCONF, and TXCONF are copied to the appropriate AFE and I^2S_x registers.
  7. No change in AFE state.
     Only the parameters VRXCTRL1, VRXCTRL2, VTXCTRL, and OUT_MODE are copied to the appropriate AFE registers.
  8. AFE input (microphone) and external audio output by I^2S_2 are switched on.
     The parameters VRXCTRL1, VRXCTRL2, VTXCTRL, CSEL, NUM0, DEN0, NUM1, DEN1, RXCONF, and TXCONF are copied to the appropriate AFE and I^2S_2 registers.
  9. AFE output (loudspeaker) as well as input (microphone) and external audio output by I^2S_2 are switched on. The parameters VRXCTRL1, VRXCTRL2, VTXCTRL, CSEL, NUM0, DEN0, NUM1, DEN1, RXCONF, and TXCONF are copied to the appropriate AFE and I^2S_2 registers:
     Hardware registers, refer to **[1]**

- RATESW
  Determines the sample rate of the audio scheduler sample based processing. This parameter is always used, except the SWITCH parameter is 0. The sample rate of the AFE input (microphone) path is always the same as for the audio scheduler.
  0. 8 kHz
  1. 16 kHz
- OUT_MODE:
  Output (loudspeaker) path of AFE and external audio output by I^2S_2.
  0. Mono.
  1. Stereo (the right channel of circular buffer passed to the right channel of output path, the same is done for the left channel, and sample based samples are mixed on both right and left output channels).
- CSEL, NUM0, DEN0, NUM1, DEN1, RXCONF, TXCONF:
  Hardware registers, refer to **[1]**

Whenever this command is given (except when SWITCH = 0) the following has to be considered:

- If the AFE output (loudspeaker) path is switched on, it is always based on a 47,619 kHz sample rate.
- Since the AFE input (microphone) path is used for triggering the audio scheduler sample based processing, the parameters VRXCTRL1, VRXCTRL2, VTXCTRL, and RATESW are always used.
- If the I^2S_x is used, it has to be configured in either the burst slave or burst master mode. If the burst master mode is used, it has to be guaranteed that the clock on the master side is exactly the same as the internal clock.
- If the AFE input is switched on in parallel with the I^2S_x (switch 5 or 6), the AFE input is only mixed at the output, not in the uplink data.
- If the external audio output by I^2S_2 is used, it has to be configured in the continuous master mode.

3.3.15 VB_SET_BIQUAD

This command sets the coefficients for the biquad filters within the DSP. Five coefficients for each filter are used:

- Parameters 1-5 are reserved for Biquad_In_1
- Parameters 6-10 for Biquad_In_2
- Parameters 11-15 for Biquad_Out_1
- Parameters 16-20 for Biquad_Out_2.

The correspondence of the command parameters and the coefficient names on the Biquad structure is given in **Table 5-1 “VB_SET_BIQUAD Parameters” on Page 109**.

3.3.16 VB_SET_GAIN

This command is used to set the gains in the different parts of the voice band processing system. The default settings of the gains are described in **Section 3.2 List of Commands VB_SET_GAIN**.

A description of their position and when they are used by the audio scheduler is described in **Chapter 5 Voiceband Processing Functions**.

3.3.17 VB_START_TONE

This command is used to start a new tone (built up by three sine waves).

The tone related parameters are read from the shared memory:

- Frequencies of the 3 waves:
  - **SM_TONE_FREQ_1**
  - **SM_TONE_FREQ_2**
  - **SM_TONE_FREQ_3**
- Amplitudes of the 3 waves:
  - **SM_TONE_AMP_1**

  - SM_TONE_AMP_2
  - SM_TONE_AMP_3
- Parameter Duration:
  - SM_TONE_DUR_IN
- Interrupt Duration:
  - SM_TONE_DUR_INTER
- Fading:
  - SM_TONE_FADIN_DUR
  - SM_TONE_FADOUT_DUR

The tone itself is started with the next voice band sample. The shared memory locations SM_TONE_DUR_IN and SM_TONE_DUR_INTER are cleared by the DSP after they are read.

3.3.18 VB_STOP_TONE

This command is used to stop a tone immediately. It resets the internal duration counter.

3.3.19 VB_READ_DURATION

This command is used to read out the DSP internal duration counter for tone generation. This counter is copied into the shared memory location SM_TONE_DUR_OUT.

3.3.20 VB_RESET

This command is used to reset the audio scheduler. Following buffers and variables are reset:

- All state buffers of each biquad filter
- Internal duration counter of tone generator. Hence the tone generator is switched off
- Sample Rate Converters
- Circular mixing buffer
- Voiceband uplink and downlink buffers
- Voice memo transfer buffers

Note: It is not recommended from FW side to send a VB_RESET command while AFE is ON, since this would reset the sample rate converters without updating the AFE Read and Write Pointers. The VB_RESET command should be sent before switching on AFE.

3.3.21 VB_DAI

This command has only one parameter MODE. It selects one of various Voiceband Signal Paths. This is needed for type approval and production testing. Depending on this parameter, four different modes can be started:

1. Normal operation
2. Vocoder test
3. Acoustic test
4. Voiceband test.

A description of the DAI functions is in Section 5.5 “DAI Functions” on Page 134.

3.3.22 HF_SET_PAR

This command is used to set the handsfree parameters within the DSP, refer to Table 3-19.

Table 3-19 Handsfree Parameters

Handsfree parameters	Value range	Default value	Short description
GAIN_ANALOG	0 ... 32767	0	External (for example GAIM) analog gain
STEP_WIDTH	0 ... 32767	13107	LMS adaptation speed (step size)

Handsfree parameters	Value range	Default value	Short description
LMS_LENGTH	2 ... 400	300	LMS filter length (number of coefficients)
LMS_OFFSET	0 ... 400	8	LMS filter offset (number of skipped taps)
BLOCK_LENGTH	{2, 4, 5, 8}	4	LMS block update vector length
RXTX_RELATION	-960 ... 960	-300	Speaker to micro signal power relation
NR_SW_2	0 ... 32767	4096	Noise reduction max attenuation
NR_U_Fak_0	0 ... 16384	4096	Noise reduction factor of overestimation for band 0
NR_U_Fak	0 ... 16384	4096	Noise reduction factor of overestimation for band 1 to 7
ADD_ATTEN	0 ... 960	0	AGC additional attenuation
MIN_ATTEN	0 ... 960	0	AGC minimal attenuation
MAX_ATTEN	0 ... 960	491	AGC maximal attenuation

3.3.23 HF_ON

This command is used to switch Handsfree on and off.

- SWITCH:

  0000_H: Handsfree/Noise reduction is switched off

  If not 0000_H, the following functions can be enabled according to the bits that are set:

  Bit #0: Echo Canceller (EC) initialization

  Bit #1: EC restart initialization
  Already adapted echo representing coefficients remain unchanged
  Does not care if Bit#0 is set.

  Bit #2: EC on, echo canceller is activated

  Bit #3: EC adaptation on, coefficients adaptation is activated

  Bit #4: Noise reduction initialization

  Bit #5: Noise reduction on

  Bit #6: Noise reduction works with additional AGC

  Bit #7: Automatic Gain Control (AGC) initialization

  Bit #8: AGC on, gain control is activated.

3.3.24 VM_CMD

Most of the Voice Memo module is controlled by the parameter VM_MODE.

The additional parameters are used to mix the uplink/downlink path to the recording input buffer or to mix the playback samples to the uplink/downlink path respectively. For a more detailed description refer to **Section 5.3.4 "Voice Memo" on Page 117**.

- VM_MODE: Description how the bits are used, refer to **Table 3-20 "Description of the Parameter VM_MODE" on Page 52**
- alpha0: Mixing uplink path to record buffer
- alpha1: Mixing downlink path to record buffer
- beta0: Gain for downlink path
- beta1: Mixing playback output buffer to uplink path
- gamma0: Gain for uplink path
- gamma1: Mixing playback output buffer to uplink path.

Note: An update of only coefficients alpha0, alpha1, through gamma1 during a voice memo procedure does not change the state of the voice memo if the parameter VM_MODE is the same as in the previous VM_CMD command.

Table 3-20 Description of the Parameter VM_MODE

Bit	Name	Description	Value
0	FR_REC	Full Rate record Voice Memo	0: OFF 1: ON
1	FR_ PL	Full Rate play Voice Memo	0: OFF 1: ON
2	PCM_ REC	PCM record Voice Memo	0: OFF 1: ON
3	PCM_ PL	PCM play Voice Memo	0: OFF 1: ON
4	---	Not used	---
5	---	Not used	---
6	ADPCM_REC	Record audio data with ADPCM codec	0: OFF 1: ON
7	---	Not used	---
8	AMR_REC	Adaptive Multirate record Voice Memo	0: OFF 1: ON
9	AMR_PL	Adaptive Multirate play Voice Memo	0: OFF 1: ON
10	---	Not used	---
11	---	Not used	---
12	CR0	Code Rate Bit 0	Refer to **Table 3-21 "AMR CODEC Rates" on Page 53**
13	CR1	Code Rate Bit 1	
14	CR2	Code Rate Bit 2	
15	DTX	DTX	0: OFF 1: ON

Table 3-21 AMR CODEC Rates

CODEC Mode	Frame Content	Mirrored Mode Indicator	CR2	CR1	CR0
0	4.75 kbit/s	b000	0	0	0
1	5.15 kbit/s	b100	0	0	1
2	5.9 kbit/s	b010	0	1	0
3	6.7 kbit/s	b110	0	1	1
4	7.4 kbit/s	b001	1	0	0
5	7.95 kbit/s	b101	1	0	1
6	10.2 kbit/s	b011	1	1	0
7	12.2 kbit/s	b111	1	1	1

If the AMR_REC bit is set, the upper 4 bits (12..15) of the VM_MODE word are interpreted as:

- DTX:
  1. DTX during recording is off (default)
  2. DTX during recording is on.
- The combination of CR2..0 defines the CODEC rate for recording with AMR:
  0. 4.75 kbit/s
  1. 5.15 kbit/s
  2. 5.90 kbit/s
  3. 6.70 kbit/s
  4. 7.40 kbit/s
  5. 7.95 kbit/s
  6. 10.2 kbit/s
  7. 12.2 kbit/s.

Notes:

1. *Voiceband initialization:*
   If any Voice Memo mode is selected (any VM_MODE bit except TRANSCODING), it is necessary to initialize the Voiceband and/or I^2S_x and/or AFE interface before starting the Voice Memo function.
2. *Voice Memo off:*
   To stop the Voice Memo (in any mode) the MCU needs to send the VM_CMD with VM_MODE set to 0.
3. *Operation mode change:*
   It is assumed that any of the Voice Memo modes is cancelled by the MCU if the mobile changes from Idle/ PDCH to dedicated mode or vice versa. This has to be done by sending the VM_CMD with VM_Mode set to 0.
4. *ADPCM_REC:*
   The built in sound ringer supports the playing and recording of Intel DVI ADPCM coded audio data. To generate an ADPCM content the Voice Memo system shall support the recording of audio data with this CODEC type.
5. *PCM_REC and PCM_PL:*
   *PCM Voice Memo Record and Playback. The 160 PCM samples are copied into the shared memory location **SM_VM_PCM_BUFFER_0** for Voice Memo record. For Playback the 160 samples are copied out of the shared memory location **SM_VM_PCM_BUFFER_1**.*

Table 3-22 shows the bits that may be set at the same time for different modes.

Table 3-22 Bits That May Be Set at the Same Time in Different VMemo Modes

	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Voice Memo OFF																
AMR Record	X	X	X	X				X								

	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
AMR Play							X									
AMR Record and Play	X	X	X	X			X	X								
FR Play															X	
FR Record																X
FR Record and Play															X	X
ADPCM Record										X						
PCM Record														X		
PCM Play													X			

3.3.25 VB_SET_CBUF_GAIN

This command is used to set the gains before and after the circular mixing buffer:

- Scal_SAPP: Gain applied to the SAPP data. This gain is applied at the input side of the circular buffer before the SRC. This gain is used for both MP3 and Polyringer data (SAPP).
- Scal_Ext: Gain applied to the external data received from the I^2S_y Rx. This gain is applied at the input side of the circular buffer before the SRC.
- Mix_AFE: Gain applied to the output samples of the circular mixing buffer. The result is mixed with the voiceband downlink samples to deliver the AFE Rx samples.
- Mix_I2Sx: Gain applied to the output samples of the circular mixing buffer. The result is mixed with the voiceband downlink samples to deliver the I^2S_x Tx samples.
- Scal_PCM: Gain applied to the PCM Player data. This gain is applied at the input side of the circular buffer before the SRC.

3.3.26 DTX_ON

This command is used to switch DTX on and off:

- SWITCH:
  0. Switch DTX off
  1. Switch DTX on

3.3.27 PW_DOWN

This command is used to prepare the DSP for the Stand By Power Down mode. Immediately after receiving this command the DSP copies the relevant program data into the shared memory, which is not switched off during the Stand By Power Down mode. The data can be restored during the next boot phase, refer to **Chapter 2 “Booting” on Page 13**. The startup code contains the information about which data has to be stored.

Before this command can be applied the MCU has to make sure that the DSP is in IDLE state and no applications are still running: it is recommended to give the IDLE command before this command.

This Stand By Power Down support can only be used if the size of the Data and Program Code of the Startup Code does not exceed the size of the Shared Memory Field **SM_SBPD_BOOT_ADD**.

Note: It cannot be guaranteed by the Firmware Group that this requirement can be fulfilled for startup codes after startup version 1.0.

3.3.28 WRITE_DSP

This command writes to the specified DSP internal data memory:

- S_ADDR Source address in shared memory (offset)

- D_ADDR Destination address in DSP internal data memory
- LEN Length of block to be written. This parameter is restricted by the size of the shared memory.

3.3.29 READ_DSP

This command reads from the specified DSP internal data memory:

- S_ADDR Source address in DSP internal data memory
- D_ADDR Destination address in shared memory (offset)
- LEN Length of block to be read. This parameter is restricted by the size of the shared memory.

3.3.30 WRITE_PROG

This command writes to the specified DSP internal program memory:

- S_ADDR Source address in shared memory (offset)
- D_ADDR Destination address in DSP internal data memory
- LEN Length of block to be written. This parameter is restricted by the size of the shared memory.

3.3.31 READ_PROG

This command reads from the specified DSP internal program memory:

- S_ADDR Source address in DSP internal data memory
- D_ADDR Destination address in shared memory (offset)
- LEN Length of block to be read. This parameter is restricted by the size of the shared memory.

3.3.32 MCU_INT

This command is used to switch on/off the PDCH interrupt:

- SWITCH:
  0. Switch PDCH interrupts off
  1. Switch PDCH interrupts on

This command can be given before going into PDCH mode or if the DSP subsystem is already in PDCH mode.

The parameter SWITCH enables the interrupt from DSP to MCU. For interrupt initialization the shared memory contains four addresses for both **SM_PDCH_DEC_INT** and **SM_PDCH_USF_INT** that have to be configured before SWITCH enables the interrupts

- Values less than zero with these eight addresses mean interrupts are not configured.
- Values greater or equal zero mean an interrupt can generated.

The following algorithms support an interrupt after processing is finished:

- USF detection, which is dependent on shared memory location **SM_PDCH_USF_INT** (signal: TOMCU0)
- Channel decoder or header decoder, which is dependent on shared memory location **SM_PDCH_DEC_INT** (signal: TOMCU1).

3.3.33 VB_I2Sy

This command is used to control and configure the I^2S_y interface. The I^2S_y always has to be configured in the normal master mode. Depending on the SWITCH parameter the command has different functions:

- SWITCH:
  0. The I^2S_y interface is switched off.
     None of the other parameters are copied or evaluated.
  1. The I^2S_y in MMS mode is switched on.
     The parameter I2Sy_MODE is used (only value 0 and 1 are allowed), but I2Sy_RATE is not used. The parameters CSEL, NUM0, DEN0, NUM1, DEN1, RXCONF, and TXCONF are copied to the appropriate I^2S_y registers.
  2. The I^2S_y interface is switched on and configured for DAI mode.
     The parameters I2Sy_RATE and I2Sy_MODE are not used. The parameters CSEL, NUM0, DEN0, NUM1, DEN1, RXCONF, and TXCONF are copied to the appropriate I^2S_y registers.
  3. The I^2S_y interface is switched on.
     The Rx path is set up for the external mode as input for the circular buffer (see **Figure 5-7**). For this mode the two parameters I2Sy_RATE and I2Sy_MODE are relevant. All values for I2Sy_MODE are allowed. The parameters CSEL, NUM0, DEN0, NUM1, DEN1, RXCONF, and TXCONF are copied to the appropriate I^2S_y registers.
- CSEL, NUM0, DEN0, NUM1, DEN1, RXCONF, TXCONF:
  Hardware registers, refer to **[1]**.
- I2Sy_RATE:
  Refer to **Table 5-30 “Input Sampling Rate Index Values” on Page 134**.
- I2Sy_MODE:
  0. Mono mode. Every sample is used.
  1. Dual mono mode. Every second sample (left channel) is used.
  2. Stereo mode. Only allowed if I^2S_y is used as input for circular buffer (SWITCH = 3).

Notes:

1. *If I^2S_y MMS mode is running in the IDLE mode and the TCH26 mode is to be enabled, the I^2S_y interface has to be switched off before activating the TCH26.*
2. *Except for DAI mode (SWITCH = 2), the I^2S_y is limited to 16-bit samples in the Tx and Rx directions.*

3.3.34 TTY_CTM

This command is used to enable/disable the TTY/CTM feature:

- SWITCH:
  0. TTY/CTM off
  1. TTY/CTM on
- NO_NEG:
  0. Switches on negotiation between uplink and downlink (normal mode). If the DSP receives TTY signals from the external device, the decoded ASCII strings are stored in an internal buffer. The CTM module first sends a special sequence and waits until it receives a response sequence from the far end CTM module. After that, it starts transmitting the ASCII strings that are still stored in the buffer.
  1. Switches negotiation off. Recommended for testing the TTY/CTM module in just one direction.
- AUT_PREAMB:
  0. Switches off auto preamble. The TTY/CTM preamble is always used independently if the DSP receives a preamble in the microphone path or not. This is only useful in the test mode.
  1. Switches on auto preamble. The TTY module uses a preamble to the connected TTY device (loudspeaker path) only if it receives a preamble in the microphone path from this device.

The following has to be considered when using the TTY/CTM feature:

- The voice band has to run in 8kHz mode.
- Before giving this command it has to be guaranteed that the audio scheduler is already running by giving the **VB_ON** command before the command.
- All biquad filters have to be switched off by setting the appropriate parameters a1, b1, a2, b2, and a0 to 0,0,0,0, and 0x7FFF (VB_SET_BIQUAD)
- The command TTY_CTM can be given either before or after Tch26.
- The tone generator cannot be used.
- Handsfree has to be switched off

3.3.35 VB_SYNC

This command is used to synchronize the uplink and/or downlink processing. Depending on the value of the SWITCH parameter, the command can have one of four different modes:

- SWITCH:
  0. No synchronization required.
  1. Synchronization is required only in downlink direction
  2. Synchronization is required only in uplink direction
  3. Synchronization is required in downlink and uplink direction.
- SYNC:
  0. First synchronization
  1. Re-synchronization.
- PTR_VAL
- SYNC_LIM.

Note: The range of the parameters PTR_VAL and SYNC_LIM depends on the synchronization type (first or re- synchronization) and direction (uplink or downlink). Refer to ***Section 6.5 "Data Interface Format" on Page 142****.*

3.3.36 UMTS_ON

This command is used to force the FW into UMTS mode. The command does not contain any parameters. The data and control information is exchanged between the MCU and DSP via the shared memory (refer to **"**UMTS Mode" on Page 153** shared memory location). For a description of the UMTS mode refer to **Chapter 6 "UMTS Audio Interface" on Page 139**.

3.3.37 MP3

This command is used to control and configure the MP3 decoder. Depending on the SWITCH parameter the command has different functions:

- SWITCH:
  0. The last input frame given with SWITCH = 2 or 3 is still MP3 decoded and fed into the circular buffer. After that, the MP3 is switch off and no further MP3 decoder is started.
  1. The MP3 decoder is switched on, the PAR1 must contain the right value to specify the MP3 sample rate. Before the MP3 is switched on it has to be guaranteed the first frame has been copied to the internal memory using SWITCH = 2 or 3.
  2. The first part of the MP3 input frame (word0,...word359) written by the MCU into the shared memory starting at the address specified by PAR1 is copied by the DSP to its internal memory.
  3. The second part of the MP3 input frame (word360,...word719) written by the MCU to the shared memory starting at the address specified by PAR1 is copied by the DSP to its internal memory.
- PAR1:
  SWITCH = 1: Sample rate, refer to **Table 5-30 “Input Sampling Rate Index Values” on Page 134**
  SWITCH = 2 or 3: Pointer to the shared memory address, where the first or second part of the MP3 input frame is stored.

For a description refer to **Section 5.4.2 “MP3” on Page 127**.

3.3.38 SYNTH

This command is used to control and configure the Synthesizer. Depending on the SWITCH parameter the command has different functions:

- SWITCH:
  0. The last input frame given with SWITCH = 2 or 3 is still used for the Synthesizer input and fed into the circular buffer. After that the Synthesizer is switched off.
  1. The Synthesizer is switched on, the PAR1 must contain the right value to specify the sample rate (only 16kHz and 32kHz are possible). Before the Synthesizer is switched on it has to be guaranteed the first frame has been copied to the internal memory using SWITCH = 2.
  2. The input frame (maximum of 65 words) written by the MCU into the shared memory starting at the address specified by PAR1 is copied by the DSP to its internal memory.
- PAR1:
  If SWITCH = 1: Sample rate (1 for 16 kHz or 4 for 32 kHz)
  If SWITCH = 2: Contains the Headroom of the Synthesizer, this means the amount of right shifts the synthesizer applies before summing up the output data of all voices to prevent clipping
  (0 <= Headroom <= 16).
- PAR2:
  If SWITCH =2: Pointer to the shared memory address, where the first input frame is stored.

For more information refer to **Section 5.4.3 “Synthesizer” on Page 129**.

3.3.39 RF_ADAPT

For the support of different RF solutions it can be helpful to change the thresholds and filter coefficients of the BB HW Filter and the BB FW Narrow Band Filter. This can be done via the command **RF_ADAPT**. It is recommended to only change the coefficients immediately after booting the DSP:

- BB_NB_FILTER_SWITCH
  Switch on/off the adaptive mode:
  <=0: The adaptive mode is used (either Narrowband or Decimation Filter - depending on power threshold, see BB_ADAPT_THRES)
  >0: Narrowband Filter is always used
- BB_ADAPT_THRES:
  Threshold for the decision, if the Narrowband or the Decimation Filter shall be used in case of Normal Burst
- SMARTIPATCH: switch for the initialization of the NB filter
  1: Switch on initialization
  0: Switch off
- BB_CTRL: Setting of the Baseband Filter Control Register
- UPDATE: Switch value, to decide, if the following 13 filter coefficients are updated
  0: Coefficients are not updated
  !=0: Coefficients are updated
- PAR0..12: Filter coefficients of the FW narrowband filter.

To get the address for the coefficients of the BB HW Filter, refer to **[1]**. For the BB FW Narrow Band Filter internal parameter fields, refer to **Table 3-23**.

Table 3-23 BB FW Narrow Band Filter Internal Parameter Fields

Name	Address in G14 Mask	Default Value After Reset
BB_Filter+0	D:0x5859	570
BB_Filter+1	D:0x585A	-315
BB_Filter+2	D:0x585B	-2750
BB_Filter+3	D:0x585C	-2237
BB_Filter+4	D:0x585D	6080
BB_Filter+5	D:0x585E	18839
BB_Filter+6	D:0x585F	25162
BB_Filter+7	D:0x5860	18839
BB_Filter+8	D:0x5861	6080
BB_Filter+9	D:0x5862	-2237
BB_Filter+10	D:0x5863	-2750
BB_Filter+11	D:0x5864	-315
BB_Filter+12	D:0x5865	570
BB_Adapt_Thres	D:0x5866	0x02CD = 717= FIX(0.7*1024)
BB_NB_Filter_Switch	D:0x5867	0x0000

Parameter definitions:

- BB_Adapt_Thres:
  An internal function checks the power of the base channel and the adjacent channel against this threshold. Range: [0, ..., 0x7fff] = [0, ..., 32)*1024 with [0, ..., 32) == floating point value of the threshold. If the floating point value of the threshold is less or equal to zero, decimation is always done. If the floating point value is 31,999, then the filtering is narrow band filtering.
- BB_NB_Filter_Switch:
  If the narrowband filter is to be used as adaptive filter, it must be set to <=0. If it is to be always switched on, the value must be >0. In this case, the function for checking the power is not called.
- BB_Filter:
  13 filter coefficients of the FW Baseband Filter. The filter is T/2 Based.

Note: The parameters BB_Adapt_Thres and BB_NB_Filter_Switch are only used in Normal Bursts and Sync Bursts. For FCB search and Monitoring the narrowband filter is always switched on.

3.3.40 AUDIOPOSTPROC

This command is used to switch on/off and control the Audio postprocessing modules of the sound ringer **Section 5.4.3.1 “Audio Postprocessing for Synthesizer” on Page 130**:

- High Frequency Shelving Filter
- Audio Compressor.

Depending on the first parameter, AudioPostProcSwitch, the two modules can be switched on/off separately. The parameter contains three bits with the following functions:

- Bit_0:
  0: The High Frequency Shelving Filter is switched off
  1: The High Frequency Shelving Filter is switched on. The filter is on by default.
- Bit_1:
  0: The Audio Compressor is switched off
  1: The Audio Compressor is switched on. The Compressor is on by default.
- Bit_2:
  0: No update is done
  1: The following parameters are updated for 16 kHz sampling rate.
- Bit_3:
  0: No update is done
  1: The following parameters are updated for 32 kHz sampling rate.

Table 3-24 describes the default settings for the four coefficients of the High Frequency Shelving Filter at 16 kHz and 32kHz.

Table 3-24 Default Settings for Filter Coefficients

Parameter Name	Default Value at 16 kHz	Default Value at 32 kHz
b_exp	2	1
b1	0xE19A	0xAE66
b0	0x4B33	0x7E66
a1	0x3333	0xD99A

Table 3-25 describes the default settings for the fourteen parameters of the Audio Compressor at 16 kHz and 32 kHz.

Table 3-25 Default Settings for Audio Compressor

Parameter Name	Default Value at 16 kHz	Default Value at 32 kHz
INITDATA_mono_flag	0	0
INITDATA_m_bufflen	40	80
INITDATA_m_inv_bufflen	819	410
INITDATA_m_hp_coeff_exp	6	7
INITDATA_m_lp1_coeff	20713	20713
INITDATA_m_lp2_coeff	1985	1985
INITDATA_m_lp4_coeff	12893	12893
INITDATA_m_lp3_coeff	809	809
INITDATA_m_L_A	-7680	-7680
INITDATA_m_L_B	-2560	-2560
INITDATA_m_G_comp	512	512
INITDATA_um_R_infA	256	256
INITDATA_um_R_AB	128	128
INITDATA_um_R_B0	0	0

3.3.41 PCMPLAY

This command is used to control and configure the PCM Player. Depending on the SWITCH parameter the command has different functions:

- SWITCH:
  0. The last input data given with SWITCH = 2 is fed into the circular buffer. After that the PCM Player is switched off.
  1. The PCM Player is switched on, PAR1 must contain the format for the data that is put in the shared memory (PCM or ADPCM), PAR2 must contain the right value to specify the sample rate. PAR3 must contain the Mode of the data (Mono, DualMono, Stereo). About 6ms after the Init Command, the first Interrupt Request for data is raised.
  2. The input data (maximum of 600 words) written by the MCU into the shared memory starting at **SM_VM_PCM_BUFFER_1** is copied by the DSP to its internal memory. PAR1 must contain the amount of sample pairs for DualMono or Stereo or the amount of Samples for Mono).
- PAR1:
  If SWITCH = 1: Data Format (0 for PCM 16-bit, 1 for ADPCM or 2 for PCM 8-bit).
  If SWITCH = 2: Contains the amount BLOCK_LENGTH of sample pairs in case of DualMono or Stereo or the amount of samples in case of Mono. This length included the two synchronization words in case they are sent (see **Table 5-27 “ADPCM Data Structure without Synchronization Double-Word” on Page 133**and **Table 5-28 “ADPCM Data Structure with Synchronization Double-Word” on Page 133**).
- PAR2:
  If SWITCH = 1: specifies the Sample Rate according **Table 5-30 “Input Sampling Rate Index Values” on Page 134**.
- PAR3:
  If SWITCH = 1: specifies the Mode (0 = Mono, 1 = DualMono, 2 = Stereo).

Note: DualMono is handled like stereo inside the DSP.

For more information refer to **Section 5.4.5 “PCM Player” on Page 131**.

3.3.42 DTW

This command is used to enable the speech recognition processing in the audio scheduler. Depending on the SWITCH parameter the command has different functions:

- SWITCH:
  0. The DSP obtains its configuration parameters for the detection of word boundaries and for adaptation
  1. Calculation the normalized pattern.
  2. Calculation of Distance between Actual Normalized Pattern and One Reference Pattern.
  3. Adaptation of One Reference.

For each command certain parameters must be written in shared memory at **SM_DTW_PAR**. The results are overwritten at the same shared memory location by the DSP.

For more information refer to **Section 5.3.5 “DTW Speech Recognizer” on Page 121**.

3.3.43 TX_DIG

This command is used to enable TX analog or digital path. By default the digital path is set. For more information refer to **Section 4.7.2 “Digital Transmit Path” on Page 69** and **[1]**.

- SWITCH:
  0. TX Analog path.
  1. TX Digital path.

3.3.44 I2S_SWAP

This command is use to swap I^2S functionnalities between I^2S_1 and I^2S_2 hardware interface.

- SWITCH:
  0. I^2S_1 used for Bluetooth functionality and I^2S_2 for MMS, DAI and external audio.
     In the FW manual I^2S_x interface is mapped on I^2S_1 hardware peripheral and I^2S_y interface is mapped on I^2S_2 hardware peripheral.
  1. I^2S_2 used for Bluetooth functionality and I^2S_1 for MMS, DAI and external audio.
     In the FW manual I^2S_x interface is mapped on I^2S_2 hardware peripheral and I^2S_y interface is mapped on I^2S_1 hardware peripheral.

This command cannot be sent if one of the I^2S is used. The best way to use it is to send the command after boot processing or **IDLE** command.

4 Modem Functions

E-GOLDradio CONFIDENTIAL		
Revision History:	**2005-12-07**	Rev. 1.01
Previous Version:	Rev. 1.00, 2005-05-16	
Page	Subjects (major changes since last revision)	
	Initial Version based on *E-GOLDlite Firmware Manual*	
Changes for Rev. 1.00		
Page 63	Update **Section 4.1 FCB Search**	
Page 66	Update **Section 4.6.1 Equalizer Output Parameters**	
Changes for Rev. 1.01		

This chapter explains the behavior of the baseband scheduler.

4.1 FCB Search

For FCB Search control the **FC_INIT** command and the FCON line from the GSM system interface unit are needed. The command is used for parameter setup, the FCON line starts and stops the FCB Search itself.

The results of the FCB Search are written to the following shared memory locations:

- **SM_FC_STATUS** FCB Search-status
  0. No FCB has been found
  1. FCB has been found
  2. FCB has been found and evaluated
  4. FCB has been found (quality of FCB greater than TRSH), but there were not enough samples to receive HYST samples to check the absolute maximum
- **SM_FC_START** Number of bits (3.69µs) from the rising edge of FCON to the beginning of the found FCB
- **SM_FC_QUAL** Quality of the found FCB - the larger this value the better the quality of the received FCB
  - **SM_FC_QUAL** > 135 (High quality FCB)
  - **SM_FC_QUAL** < 120 (Low quality FCB)
- **SM_FC_RMS** RMS value of the received FCB given in [dB/16]
- **SM_FC_FREQ** Frequency offset of the received FCB given in [Hz]

In the start of the FCB Search (FCON goes high) the shared memory location **SM_FC_STATUS** is set to ’0’, and communication flags #3 and #4 are set to ’1’.

If no FCB is found before the falling edge of the GSM Timer Signal FCON, the RMS value of the last 128 IQ sample pairs is estimated. The RMS is saved in the Shared Memory at **SM_FC_RMS**.

If an FCB is found, the two results, **SM_FC_START** and **SM_FC_QUAL**, are written to the shared memory, **SM_FC_STATUS** is set to ’1’, and communication flag #3 is reset to ’0’. If there were not enough samples to receive HYST samples to check the absolute maximum, **SM_FC_STATUS** is set to ’4’ and communication-flags #3 and #4 are reset to ’0’ (no evaluation of the RMS-value and frequency offset is done).

The RMS value and Frequency offset are computed. After this computation (~ 340µs@104MHz) the two results are written to the shared memory locations **SM_FC_RMS** and **SM_FC_FREQ**, **SM_FC_STATUS** is set to ’2’, and communication flag #4 is reset to ’0’.

To wait for the result of the FCB Search, the user (MCU) should not poll the shared memory location **SM_FC_STATUS**. Instead, poll communication flags #3 and #4. The reason is that the MCU must not read the same

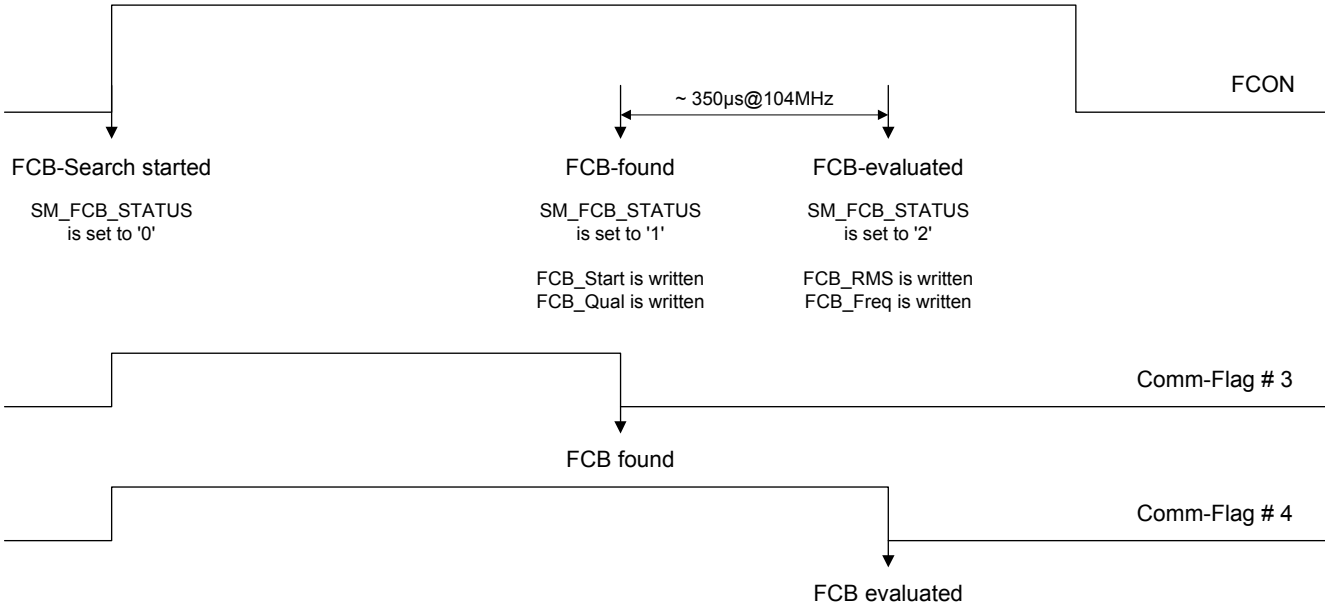
shared memory location at the same time when the DSP writes to it, and this might be happen in this case because the MCU does not know when the DSP writes to the shared memory location **SM_FC_STATUS**.

A timing diagram of a successful FCB Search with frequency evaluation is given in **Figure 4-1**.

The next FCB Search can be started after Comm Flag #4 is cleared by the DSP.

Note: In the case of zero inputs the value of RMS in the Shared Memory is 0xFF80.

Figure 4-1 FCB Search



4.2 Sync Burst Detection

Sync Burst Detection is controlled by the GSM system interface signal SCON. If SCON goes high and remains high for at least 160 bit-durations (= 160*3.69µs = 591µs) the Sync Burst equalizer and the Sync Burst decoder is started.

The Sync Burst equalizer and Sync Burst decoder, together, have a runtime of about 630µs@104MHz. The output of the Sync Burst decoder is written to the following shared memory locations:

- **SM_SYNC_METRIC** Sync Burst decoder metric (max. value 78)
- **SM_SYNC_STATUS** Sync Burst decoder status
  0. CRC OK, the decoded data is valid
  1. CRC not OK, the decoded data is not valid
- **SM_SYNC_DATA** Sync Burst output data (25 information bits contained in two 16-bit words)

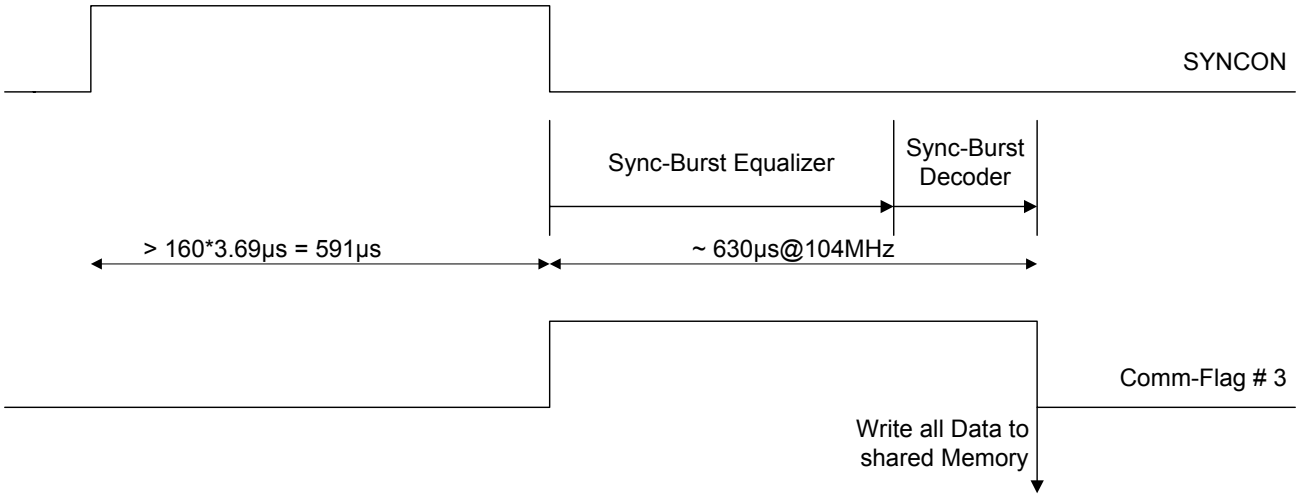
The shared memory location **SM_SYNC_EQU** contains the equalizer output parameters for the sync burst (refer to **Section 4.6.1 “Equalizer Output Parameters” on Page 66**).

At the beginning of the Sync Burst equalizer run communication flag #3 is set to ’1’. At the end of the Sync Burst decoder run, after the results have been written to the shared memory, this flag is reset to ’0’ again. The MCU can poll this flag.

A timing diagram of a Sync Burst detection run (equalizer and decoder) is given in **Figure 4-2**.

Note: The 32-bit error probability is in q31 notation. The maximum value is 0.5. The higher the value the higher the error probability of the signal.

Figure 4-2 Sync Burst Detection



4.3 Monitoring

Monitoring on E-GOLDradio is controlled by the GSM system interface signal MONON. At the end of a MONON window (the falling edge of MONON) the DSP gets an interrupt and immediately starts the calculation of the RMS value. For this calculation the DSP takes the samples that the Baseband filter has written to the DSP internal memory during the MONON window. The result of the MONON algorithm is written to the shared memory. This RMS value is normalized to the length of the window and is given in units of dB/16.

When writing the result to the shared memory the DSP evaluates the shared memory location **SM_MON_INDEX**:

- If **SM_MON_INDEX** is in the range 0...7, the DSP writes the result to the shared memory location **SM_MON_VALS.SM_MON_INDEX** and it is increased by 1 and reset to 0 if it has reached a value of 8 (counting modulo 8).
- If **SM_MON_INDEX** is not in the range 0...7, the DSP writes the result to the shared memory location **SM_MON_VALS**[0] and **SM_MON_INDEX** is set to -1.

For getting a useful RMS value the window size must at least be 60 symbols (60*3,69µs = 221,4µs). The maximum window length that may be given is 190 bits (190*3.69µs = 701,1µs) long. For this maximum window length the DSP requires less than 25µs@104MHz to compute the result and to write it to the shared memory.

Notes:

1. *Because of the firmware implementation the range of the monitoring window size must be between 7 and 190 symbols. The gap between two consecutive monitoring windows must be at least* 70µs.
2. *In the case of zeros inputs the value of RMS in the Shared Memory is 0xFF80.*

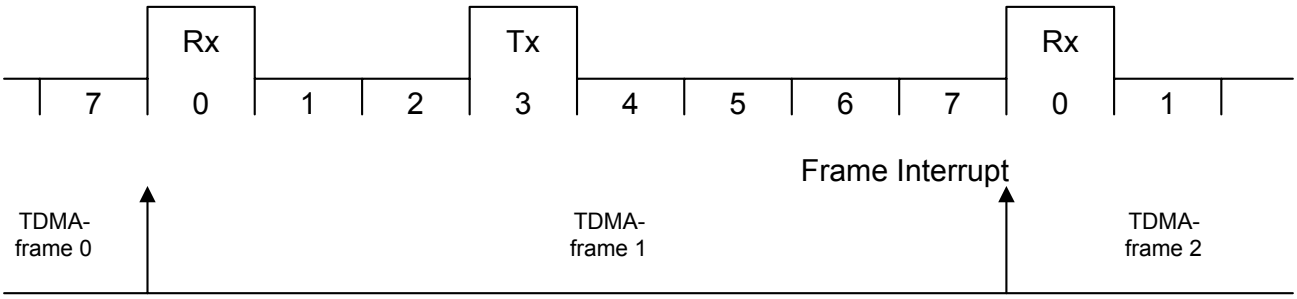
4.4 Frame Interrupt

For proper operation many parts of the scheduler require a Frame Interrupt to be given once per TDMA frame (for example, the DSP internal TDMA counters are updated with every Frame Interrupt). This Frame Interrupt has to be given by the GSM system interface unit using the FRAME signal:

- This frame interrupt has to be raised once per TDMA frame. At the latest, the frame interrupt always should be raised with the beginning of the first Rx timeslot or three timeslots minus timing advance before the first Tx timeslot in the TDMA frame, see **Figure 4-3**.
- In Stand-By mode it has to be raised in every frame where the Rx or Tx timeslot is active.
- In TCH26 mode it has to be raised in every frame. For halfrate or AMR halfrate this frame interrupt is also required in those frames that belong to the other subchannel.
- In PDCH mode a 52-multiframe structure is used. The Frame Interrupt has to be raised in each TDMA even if there is no Rx or Tx timeslot allocated.

- The Frame Interrupt has also to be raised in an IDLE frame.

Figure 4-3 Frame Interrupt



4.5 TDMA Counters

The DSP internal TDMA counters are updated once per TDMA frame from the TDMA counter shared memory locations **SM_COUNTER_104**, **SM_COUNTER_51**, and **SM_SFNUM**. At every frame interrupt (refer to **Section 4.4**) these shared memory TDMA counters are copied on to the DSP internal counters, then the shared memory TDMA counters are incremented (even in the IDLE frame). That is, the shared memory TDMA counters are post-incremented (using the C language.)

In this way, the MCU can cross check (and *modify*) the TDMA counters used by the DSP. As described above, the shared memory TDMA counters always hold those values that are used by the DSPs in the **next** TDMA frame

In the shared memory TDMA counters:

- **SM_COUNTER_104** TDMA frame number modulo 104
- **SM_COUNTER_51** TDMA frame number modulo 51
- **SM_SFNUM** TDMA frame number DIV 26*51 where DIV stands for integer division.

4.6 Equalizer

The DSP subsystem supports two different equalizers, one for the sync burst (see **Section 4.2 “Sync Burst Detection” on Page 64**) and one for a normal burst. The modulation type is automatically decided by the DSP subsystem according to the internal mode. The output parameters described in the following sections can only be read by the MCU in the gap one timeslot before the frame interrupt and one timeslot after the frame interrupt. The equalizer output parameters are guaranteed to be available in the shared memory at the latest one timeslot before the frame interrupt.

4.6.1 Equalizer Output Parameters

After every sync burst equalizer, 14 parameters are written to the shared memory location **SM_SYNC_EQU**.

After every normal burst equalizer run, a block of 14 words is copied to the shared memory location **SM_EQUAL_0**/1/2/3 for the respective time slot. Some examples are:

- For a single slot connection speech call or CCH_RX, the equalizer output parameters are written to **SM_EQUAL_0**.
- For a multislot connection HSCSD or GPRS where subsequent timeslots are allocated, the equalizer parameters of the first timeslot are written to **SM_EQUAL_0** and the output of the second to **SM_EQUAL_1**, and so on.
- For a multislot connection HSCSD or GPRS where arbitrary timeslots are allocated, the shared memory location which contents the equalizer parameters depends on the MCU use of "EQON continuous" or "EQON with gaps" (see **Section 4.11.5 “Data Channels” on Page 83** and **Section 4.12.2.1 “Receiving Radio Blocks” on Page 88**).

The Equalizer Output parameters are:

- **EQON Continuous**
  The procedure is similar to that used for subsequently allocated timeslots. The only difference is that the shared memory locations of those timeslots containing gaps are not used. For example, if timeslots #0, #1, #3 are used (a gap in #2) the parameters of timeslot 0 are in **SM_EQUAL_0**, timeslot 1 in **SM_EQUAL_1**, timeslot 3 in **SM_EQUAL_3**, whereas **SM_EQUAL_2** is not used.
- **EQON with Gaps**
  Only the allocated time slots are counted. For example, if timeslots #0, #1, #3 are used (a gap in #2) the parameters of timeslot 0 are in **SM_EQUAL_0**, timeslot 1 in **SM_EQUAL_1**, timeslot 3 in **SM_EQUAL_2**, whereas **SM_EQUAL_3** is not used.
- **Shared Memory Parameters**
  Each shared memory output block **SM_SYNC_EQU** and **SM_EQUAL_0**/1/2/3 contains the following 14 parameters:

EQ_MODUL:	Modulation Type (’1’ for GMSK).
EQ_DC_R:	Inphase offset subtracted from input signal.
EQ_DC_I:	Quadrature offset subtracted from input signal.
EQ_RMS:	RMS value of received burst given in [dB/16].
EQ_NSR:	Noise to Signal Ratio (logarithmic).
EQ_POS:	Position of received burst within SCON/EQON window given in units of Bit/8.
EQ_BEP_32H:	High word of 32 bit error probability.
EQ_BEP_32L:	Low word of 32 bit error probability.
EQ_QUAL:	Range 0...78x15 for sync bursts, 0...116x15 for normal bursts.
EQ_FREQ_OFF:	Estimated frequency offset of received burst given in [Hz].
CHANORD_FLAG:	0: Channel length not shortened 1: Channel length shortened.
MEAN_BEPH:	High word of mean value of 32 bit error probability. For details refer to **Section 4.6.2**
MEAN_BEPL:	Low word of mean value of 32 bit error probability. For details refer to **Section 4.6.2**
CV_BEP:	Constant variation of bit error probability. For details serefer toe **Section 4.6.2**

The output parameters MEAN_BEPH, MEAN_BEPL, and CV_BEP are only calculated for the normal burst. For a sync burst these paremeters are set to 0.

Additional output:

- **SM_BB_IF_FLAG**
  The output of the power estimator P_{Base}-P_{Adj} (determines if the baseband input samples were decimated (<= 0) or narrow band filtered (> 0)) is copied to the shared memory location **SM_BB_IF_FLAG**. The field contains 4 values for up to 4 Rx time slots.

Note: For multislot configurations all the shared memory output locations for, for example, the enhanced measurement, channel decoder, USF detection, etc. each timeslot are handled in the same way as the equalizer output.

4.6.2 Enhanced Measurement Reporting Support

For enhanced measurement reporting the Firmware provides:

- Mean_BEP calculations. This is the mean value of the BEPs (bit error probabilities) of 4 bursts.
- CV_BEP calculations. This is the normalized standard deviation of the BEPs of 4 bursts.

For the different channels, Mean_BEP and CV_BEP are calculated in the following way:

- TCH_RX: Mean_BEP and CV_BEP of the last 4 bursts of the speech frame.
- PDCH_RX: Mean_BEP and CV_BEP of the 4 bursts of a RBL.
- CCH_RX: Mean_BEP and CV_BEP of the 4 bursts of the CCH.
- FACCH_RX: Mean_BEP and CV_BEP of the 4 bursts of the FACCH.
- RATSCCH_RX: Mean_BEP and CV_BEP of the 4 bursts of the RATSCCH.
- SACCH_RX: Mean_BEP and CV_BEP of the 4 bursts of the SACCH.
- PTCCH_RX: Mean_BEP and CV_BEP of the 4 bursts of the PTCCH.

For multi-slot configurations, Mean_BEP and CV_BEP calculations are done separately for each timeslot.

Mean_BEP and CV_BEP calculations are started after the equalizer run.

Mean_BEP and CV_BEP calculations are done separately for the SACCH/PTCCH and non-SACCH channels (TCH, PDCH, CCH, FACCH, RATSCCH):

- Non-SACCH channels:
  Calculations are done after every equalizer run always taking the BEPs of the last 4 bursts. The controller has to be careful when the results are read: only when the last burst of a block (RBL, speech frame, etc.) has been received are the calculated results valid.
- SACCH/PTCCH:
  In TDMA frame #12, Mean_BEP and CV_BEP calculations are done for the SACCH/PTCCH (in addition to the calculations for non-SACCH), taking the last 4 BEPs of bursts received in TDMA frame #12. The results are only valid after the last burst of the SACCH/PTCCH has been received.

Since the output is only valid at the end of a block (RBL, speech frame, etc.), the controller must read Mean_BEP and CV_BEP from the shared memory together with the decoder output data. In the case of a CCH with early decoding switched on, the enhanced measurement output might not be valid when the decoder output is ready because the Firmware might have decoded the CCH already after the 2nd or 3rd burst. In this case, there is no valid enhanced measurement output because not all bursts of the CCH were received.

All Mean_BEPs are stored in Q31 notation with the high word stored at the lower address. Mean_BEPs can have values of 0-0,5 (0-0x40000000).

All CV_BEPs are stored in Q14 notation. CV_BEPs can have values of $0-(2-2^{-14})$ (0-0x7FFF).

4.7 Modulator

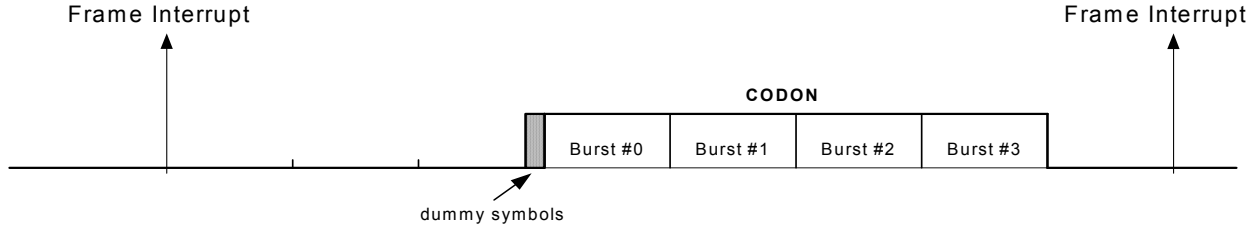
4.7.1 Analog Transmit Path

Starting with the rising edge of CODON, the modulator hardware reads out the first symbol and starts the modulation. In parallel, the rising edge of CODON is also used to trigger the DSP to build up the first burst of this TDMA frame in the modulator RAM (tail bits, data bits, training sequence, etc.). Due to timing restrictions in the DSP subsystem, a certain amount of dummy symbols are required before the first symbol of the burst starts: the number of DUMMY symbols must be set to a value >= 21 by sending command **MODU_INIT**.

In **Figure 4-4** the CODON signal has to be set DUMMY symbols earlier by the system interface. When the modulator is started at rising edge of signal CODON, it transmits some dummy symbols followed by the first burst. This is independent of the current DSP mode.

Before the burst is completely transmitted the DSP is awaken again for copying the next burst into the modulator RAM. There are no dummy symbols between subsequently transmitted bursts.

Figure 4-4 Dummy Symbols



I/Q Swap

Table 4-1 is overview IQ Swap. The IQ Swap settings are mainly used for correction of wiring errors on PCB's between BB chip and RF part. Normally these settings are only set once by the MCU. With the parameter switch of IQ_SWAP Command you can swap the signals.

Table 4-1 I/Q Swap Function

		MODU_INIT.iq_setup	
		0	**1**
IQ_SWAP.switch	**0**	Rx'.q = Rx.q Rx'.i = Rx.i Tx'.q = Tx.q Tx'.i = Tx.i	Rx'.q = Rx.q Rx'.i = Rx.i Tx'.q = Tx.i Tx'.i = Tx.q
	1	Rx'.q = Rx.i Rx'.i = Rx.q Tx'.q = Tx.i Tx'.i = Tx.q	Rx'.q = Rx.i Rx'.i = Rx.q Tx'.q = Tx.q Tx'.i = Tx.i

4.7.2 Digital Transmit Path

The Digital Transmit Path is activated by the **TX_DIG** command. From the DSP software point of view it is completely transparent, refer to **Section 4.7.1 "Analog Transmit Path" on Page 69**. Because of the omitted latency in the digital TX interface, in contrast to the analog TX path, the software timing has to be changed by either:

- GSM timer adjustment (TXON compare to CODON)
- Timing compensation on the HW delay line.

Refer to **[1]** for more details. By default the analog transmit path is set.

4.8 Control Channel Receive on CCCH (Mode 51) or PCCCH

The **CCH_RX** command is a synchronous command, the command is latched and accepted immediately by the DSP but does not become valid before the next frame interrupt.

To start a CCH Rx procedure the **CCH_RX** command has to be sent before the first Rx timeslot and, therefore, before the first frame interrupt in the block to be received (see **Figure 4-5**), but it must be given after the frame interrupt of the previous TDMA frame.

If ciphering (A51, A52, A53) is switched on, the shared memory TDMA counters have to be set up correctly before the first frame interrupt.

The results of the four equalizer runs for the four Rx timeslots is written to the shared memory location **SM_EQUAL_0**, for more information refer to **Section 4.6.1 “Equalizer Output Parameters” on Page 66**.

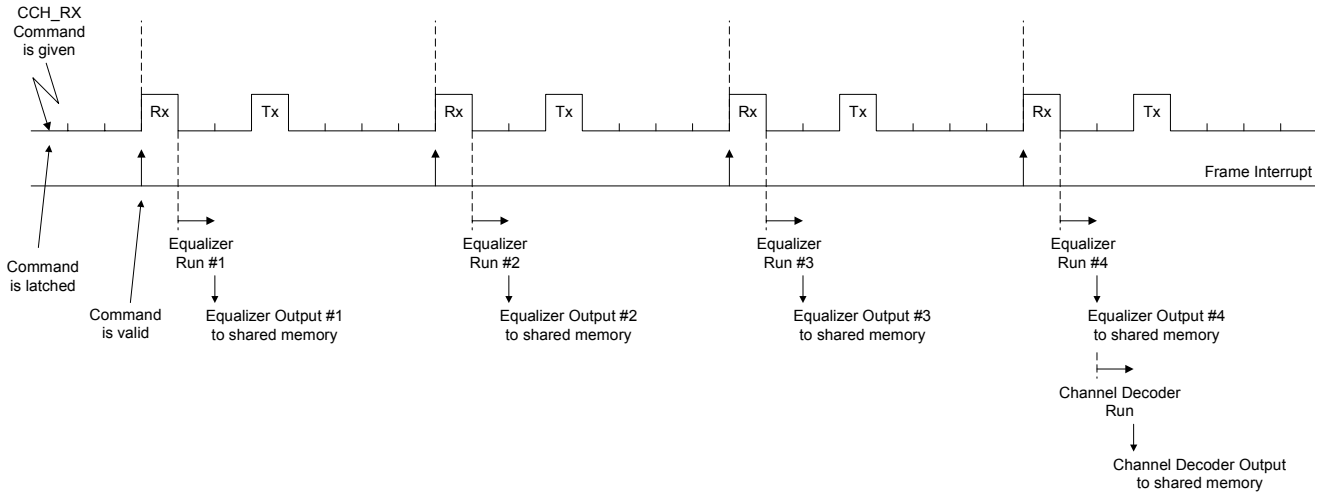
After the equalizer run of the fourth burst, the CCH channel decoder is started in the DSP. The results of this channel decoder run are written to the shared memory location **SM_SDCCH_RX_DATA**. This 14-word shared memory location block consists of the following data in order:

1. CCH_RX_METRIC Channel decoder metric (1 word, max. value: 456)
2. CCH_RX_STATUS Channel decoder status (1 word), the possible values are:
   0. No error in CRC, decoded data is valid
   1. Error in CRC, but decoded data is valid (error has been corrected)
   2. Error in CRC, decoded data is not valid (error could not be corrected)
3. CCH_RX_DATA 184 bit output data stored in 12 16-bit words.
   a) Word 0 = D(15:0)
   b) Word 1 = D(31:16)
   – ...
   l) Word 11 = D(183:176).

If two CCH Rx blocks have to be received in a sequence, the CCH_RX command for the second block can be given at any time in the last TDMA frame of the first block.

Figure 4-5 is a timing diagram for a CCH Rx procedure.

Figure 4-5 Control Channel Receive on CCCH (Mode 51) or PCCCH



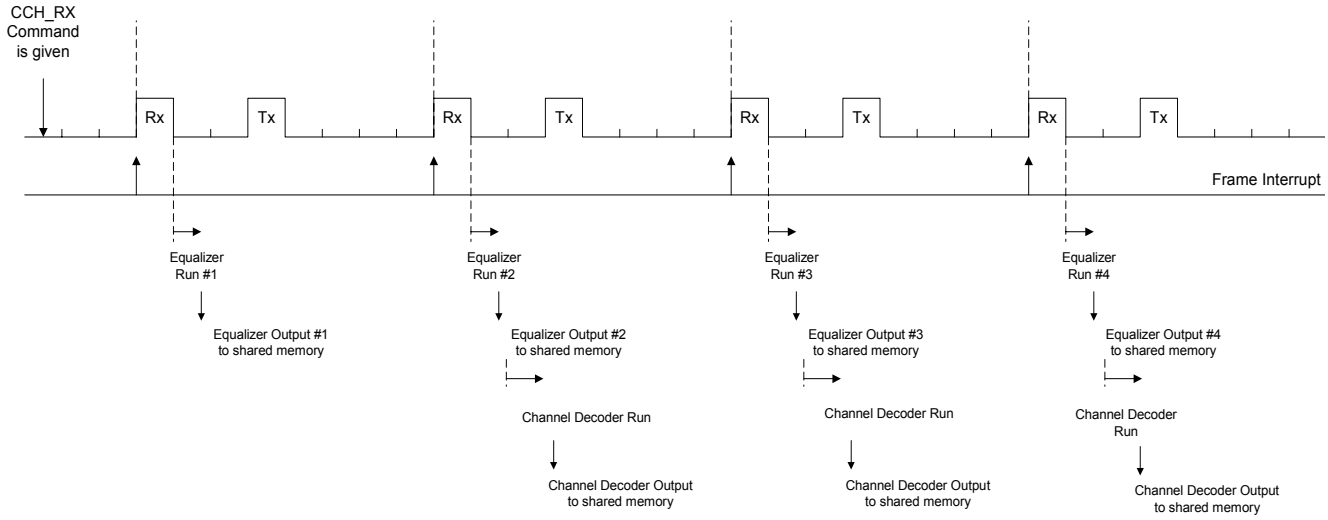
From a theoretical point of view the CCH channel decoder run can be successful with only two or three received bursts with high quality. In this case an early decoder start (decoder starts after second or third burst) can be forced by setting the parameter value ’EARLY’ in the **CCH_RX** command to ’1’.

To use this early decoding feature and to avoid a multiple channel decoder run, the MCU has to check the shared memory location CCH_RX_STATUS after the second and the third burst.
If this shared memory location contains the value ’0’, the DSP has successfully decoded the received control channel. In this case, the MCU can read the data from CCH_RX_DATA and immediately switch off all Rx signals to

save power consumption.
If the MCU does not switch off the Rx path reading the data, a new channel decoder run occurs in the next frame.

Note: By default the first received burst is written to the CCH_RX_DATA location and the communication-flag #7 is set to '1' as soon as the burst is complete. This process enables very fast CCH decoding from the MCU side.

Figure 4-6 Control Channel Receive on CCCH (Mode 51) or PCCCH



4.9 Control Channel Transmit on CCCH (Mode-51) or PCCCH

The **CCH_TX** command is a synchronous command, the command is latched and accepted immediately by the DSP but does not become valid before the next frame interrupt.

To start a CCH Tx procedure the **CCH_TX** command has to be sent before the first frame interrupt at the beginning of the block to be transmitted (see **Figure 4-7**), but it must be given after the frame interrupt of the previous TDMA frame.

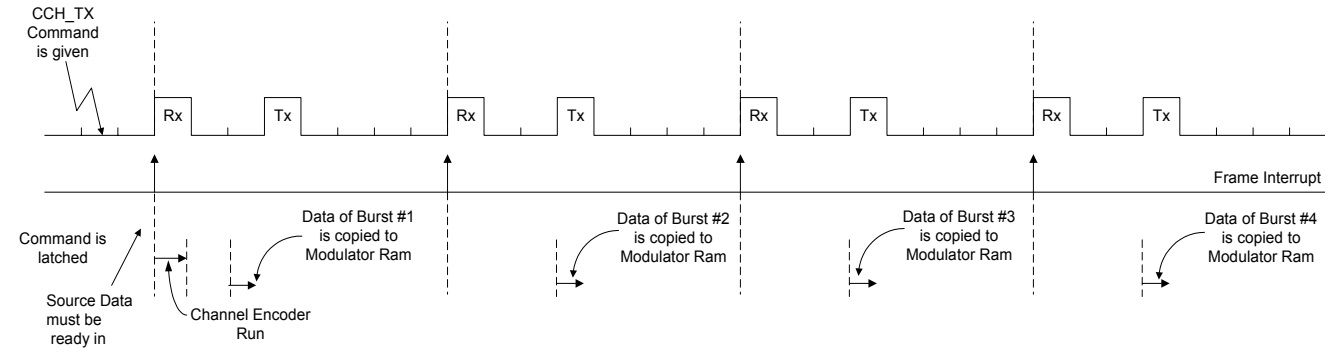
If ciphering is switched on, the shared memory TDMA counters have to be set up correctly before the first frame interrupt.

The source data is expected in the shared memory location **SM_SDCCH_TX_DATA** before the next frame interrupt after the command has been sent. The 184 information bits have to be stored in this 12-word shared memory location.

If two CCH Tx blocks have to be transmitted in a sequence, the **CCH_TX** command for the second block can be given at any time in the last TDMA frame of the first block.

Figure 4-7 is a timing diagram for a CCH Tx procedure.

Figure 4-7 Control Channel Transmit on CCCH (Mode-51) or PCCCH



4.10 Access Burst (RACH)

(P)RACH Encoding

With every Frame Interrupt the DSP checks if an Access Burst has to be transmitted instead of the normal burst by checking the value of the shared memory location **SM_RACH_FLAG**:

- 0: Normal operation goes on.
- 1: The DSP starts the 8 bit RACH encoding procedure.
- 2: The DSP starts the 11 bit extended RACH encoding procedure.

By starting the encoder procedure, the DSP reads the Access Burst information **SM_RACH_DATUM** and **SM_RACH_BSIC** and sets a DSP internal flag for Rach transmission within this frame. The shared memory flag **SM_RACH_FLAG** is reset to '0', the MCU has to set it a second time for another RACH to be transmitted in the next frame. **Table 4-2** shows how the shared memory location for the RACH and extended RACH is used.

Table 4-2 Shared Memory Block for RACH

Shared Memory	Contents	Description
SM_RACH_FLAG	1	8 bit RACH
	2	11 bit extended RACH
SM_RACH_TSC	0	training sequence
	1	alternative training sequence TS1
	2	alternative training sequence TS2
SM_RACH_TIM_ADV	0...63	timing advance value
SM_RACH_DATUM	Bit[7:0] = d[7:0]	Information bits for RACH
	Bit[11:0] = d[11:0]	Information bits for extended RACH
SM_RACH_BSIC	Bit[2:0] = BS[2:0]	Base Station color code
	Bit[5:3] = PLMN[2:0]	PLMN color code

*Note: The DSP accesses the Rach related shared memory locations (**SM_RACH_FLAG**, **SM_RACH_TSC**, **SM_RACH_TIM_ADV**, **SM_RACH_DATUM**, and **SM_RACH_BSIC**) only in the interrupt service routine for the frame interrupt. The MCU may read and write these flags at any other time.*

Training Sequence

The shared memory word **SM_RACH_TSC** allows the use of the alternative trainings sequences TS1 and TS2. The DSP fetches the TSC index only during the frame interrupt if a (P)RACH is requested by the MCU. The DSP does not overwrite the TSC index in the shared memory.

Timing Advance

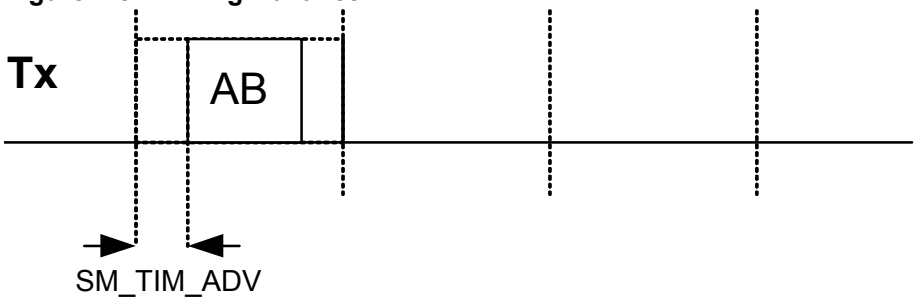
The DSP uses the timing advance value to transmit a (P)RACH. This is done no matter if the DSP subsystem is in the IDLE mode, TCH26 or PDCH mode.

When a request is made to encode and to transmit a (P)RACH by the MCU, the shared memory location **SM_RACH_TIM_ADV** is used. In addition to the (P)RACH data the MCU has to write the timing advance value to this shared memory location. (range of 0 to 63). At the beginning of every frame interrupt when a (P)RACH is requested, this value is fetched by the DSP and is used for the (P)RACH which is transmitted during the current TDMA frame. The DSP does not change the shared memory value so the MCU can overwrite the value only when there is any change in timing advance.

Figure 4-8 shows an example. The DSP shifts the PRACH right within the timeslot according to the value of the shared memory location **SM_RACH_TIM_ADV**.

The bits before and after the (P)RACH are filled with “dummy” bits. So when the MCU starts the modulator by giving the right signal from the system interface the modulator starts transmitting the dummy bits and follows with the encoded (P)RACH bits.

Figure 4-8 Timing Advance



For normal bursts it is not necessary that the DSP has any information about timing advance. It just writes the bursts into the modulator RAM and the MCU takes care of the timing on the radio interface.
A random access burst is transmitted without timing advance except in the case of a single slot connection. This can be handled by the MCU by shifting the modulator window. In this case, the shared memory location **SM_RACH_TIM_ADV** must be ’0’.

Alternatively, in a single slot connection the MCU may use this timing advance feature: there is no need to change the modulator window and to reprogram the modulator start signal in the system interface. Therefore, the MCU may or may not use the parameter **SM_RACH_TIM_ADV**.

Notes:

1. *If the DSP subsystem is in PDCH mode this method of sending a (P)RACH can only be used if only one Tx timeslot is allocated. In this case, the complete memory field* **SM_TX_INFO** *has to be set to ’-1’.*
2. *If the DSP subsystem is in TCH26 mode with channel type HSCSD, the DSP always transmits the RACH in the mainstream.*

4.11 TCH 26 Mode

The user (MCU) applies the **TCH_26** command to start or to modify a TCH26 mode on the DSP subsystem. For information about command and its parameters refer to **Section 3.3.9 “TCH_26” on Page 43**. When using this command:

- If the DSP is in Stand-By mode, the MCU can apply the **TCH_26** command to force the DSP into the TCH26 mode. Since this is a synchronous command, it is accepted immediately but does not become valid before the frame interrupt in the next TDMA frame (the command is latched).
- If the DSP is already in TCH26 mode, the **TCH_26** command can be used to modify the channel type. The command is latched, but does not become valid before the next frame interrupt. The Init flag (see parameter list of **TCH_26** command) can be used to select which parts of the interleaving and de-interleaving memory are initialized (for details refer on **Table 3-17 “Bits of the Initialization Switch” on Page 45**). This is valid for TCH Start for TCH Modify usage of the **TCH_26** command.
- When using a proper value for the Init flag (no initialization of Facch and Sacch data) in a TCH Modify command it is guaranteed that no Facch and no Sacch is lost due to this TCH Modify command.
- The shared memory TDMA counters have to be set up correctly before the first frame interrupt after the **TCH_26** command.
- When sending the **TCH_26** command the Dtx mode, handsfree, voice-memo, all test loops, and DAI (Digital Audio Interface) test modes are switched.
- The TCH26 mode is terminated by sending the **IDLE** or **BB_OFF** command.
- Audio scheduler can be switched on either before or after **TCH_26** command is given.
- Switching off the audio scheduler while in TCH26 mode is not allowed.
- Switching between AFE, I^2S_1, and I^2S_1+AFE output during TCH26 mode is allowed.

4.11.1 Sacch

Sacch Tx

For all single slot channel types and for the mainstream in a HSCSD constellation the source data for an Sacch Tx is expected in the shared memory location **SM_SACCH_TX_DATA_0**. For HSCSD uplink sub-stream #1 the source data is expected in the shared memory location **SM_SACCH_TX_DATA_1**. The 184 information bits are stored in these 12-word shared memory locations.

The encoding of the Sacch data is always done in the TDMA frame before the first Sacch frame. The encoding process starts with the frame interrupt, so the MCU has to write the source data to the shared memory before the frame interrupt of the frame preceding the first Sacch frame.

Table 4-3 lists all Sacch constellations and indicates when the source data for the Sacch Tx has to be available in the shared memory.

Table 4-3 Sacch Tx

Sacch Is Transmitted in Frames	Sacch Tx Data Must Be Available Before Frame Interrupt in Frame
12,38,64,90	11
25,51,77,103	24
38,64,90,12	37
51,77,103,25	50
64,90,12,38	63
77,103,25,51	76
90,12,38,64	89
103,25,51,77	102

Sacch Rx

When receiving an Sacch the results are written to the shared memory blocks **SM_SACCH_RX_DATA_0**/1/2/3. **SM_SACCH_RX_DATA_0** is used for all single slot channel types and for the mainstream in HSCSD. The downlink sub-streams #1, #2, and #3 in HSCSD results is written to the shared memory blocks **SM_SACCH_RX_DATA_1**/2/3.

The structure of the 14-word shared memory blocks **SM_SACCH_RX_DATA_0**/1/2/3 is identical to the CCH Rx result **SM_SDCCH_RX_DATA**. This 14-word shared memory location block consists of the following data in order:

1. SACCH_RX_METRIC Channel decoder metric (1 word, max. value: 456)
2. SACCH_RX_STATUS Channel decoder status (1 word), the possible values are:
   0. No error in CRC, decoded data is valid
   1. Error in CRC, but decoded data is valid (error has been corrected)
   2. Error in CRC, decoded data is not valid (error could not be corrected)
3. SACCH_RX_DATA 184 bit output data stored in 12 16-bit words.
   a) Word 0 = D(15:0)
   b) Word 1 = D(31:16)
   – ...
   l) Word 11 = D(183:176).

Sacch Decoding is done after the last burst of an Sacch block has been received. In all channel types (including HSCSD) with more than one Rx slot, the results are available by the end of that last Sacch frame. **Table 4-4** shows when the source data for the Sacch Rx has to be available in the shared memory.

Table 4-4 Sacch Rx

Sacch Is Transmitted in Frames	Sacch Rx Data Must Be Available Before Frame Interrupt in Frame
12,38,64,90	90
25,51,77,103	103
38,64,90,12	12
51,77,103,25	25
64,90,12,38	38
77,103,25,51	51
90,12,38,64	64
103,25,51,77	77

4.11.2 Facch

Facch Tx

To decide if an Facch has to be transmitted or not the DSP checks the shared memory flag **SM_FACCH_TX_FLAG**. At the beginning of every encoder run where an Facch is allowed to start, the DSP checks the flag:

0: TCH encoding is done

1: Facch is encoded and the shared memory flag **SM_FACCH_TX_FLAG** is reset to ’0’ again.

Note: When using a Signalling Only Channel the shared memory flag **SM_FACCH_TX_FLAG** *is not checked. In this case, an Facch is always transmitted, independently of the flag* **SM_FACCH_TX_FLAG**.

The Facch source data is stored in the shared memory block **SM_FACCH_TX_DATA** with 184 bits in 12 words as follows:

- Word 0 = D(15:0)
- Word 1 = D(31:16)
- ...
- Word 11 = D(183:176)

Table 4-5 shows when the Facch data (**SM_FACCH_TX_FLAG** and **SM_FACCH_TX_DATA**) has to be available in the shared memory.

Table 4-5 Facch Tx (Fullrate and Halfrate)

Facch Is Transmitted in Frames	Facch Tx Data Must Be Available Before Frame Interrupt in Frame
Fullrate	
0,1,2,3,4,5,6,7	24
4,5,6,7,8,9,10,11	3
8,9,10,11,13,14,15,16	7
13,14,15,16,17,18,19,20	11
17,18,19,20,21,22,23,24	16
21,22,23,24,0,1,2,3	20
Halfrate Subchannel-0	
0,2,4,6,8,10	24
8,10,13,15,17,19	7
17,19,21,23,0,2	16
Halfrate Subchannel-1	
1,3,5,7,9,11	24
9,11,14,16,18,20	7
18,20,22,24,1,3	16

Facch Rx

When receiving an Facch the results are written to the shared memory location **SM_FACCH_RX_DATA**. This 14-word shared memory location block consists of the following data in order:

1. FACCH_RX_METRIC Channel decoder metric (1 word, max. value: 456)
2. FACCH_RX_STATUS Channel decoder status (1 word), the possible values are:
   0. No error in CRC, decoded data is valid
   1. Error in CRC, but decoded data is valid (error has been corrected)
   2. Error in CRC, decoded data is not valid (error could not be corrected)
3. FACCH_RX_DATA 184 bit output data stored in 12 words.
   a) Word 0 = D(15:0)
   b) Word 1 = D(31:16)
   – ...
   l) Word 11 = D(183:176).

When running the channel decoder the DSP checks the stealing flags of the received burst:

- If less than four stealing flags (4 out of 8) are set, normal TCH decoding is done. The shared memory flag **SM_FACCH_RX_FLAG** is set to ’0’.
- If four or more than four stealing flags are set, Facch decoding is done. The Facch decoder is started, the results are written to **SM_FACCH_RX_DATA**, and the shared memory flag **SM_FACCH_RX_FLAG** is set to ’1’.

Note: In the case of a signalling only channel, the flag **SM_FACCH_RX_FLAG** *is always set to ’1’.*

The DSP tries to start the Facch Decoder in every frame where an Facch in downlink direction is allowed to end. **Table 4-6** shows when the Facch Rx data (**SM_FACCH_RX_FLAG** and **SM_FACCH_RX_DATA**) is available in the shared memory to be read out by the MCU.

Table 4-6 Facch Rx (Fullrate and Halfrate)

Facch Is Received in Frames:	Facch Rx Data Is Available After the End Of Slot #7 in Frame:
Fullrate	
0,1,2,3,4,5,6,7	7
4,5,6,7,8,9,10,11	11
8,9,10,11,13,14,15,16	16
13,14,15,16,17,18,19,20	20
17,18,19,20,21,22,23,24	24
21,22,23,24,0,1,2,3	3
Halfrate Subchannel-0	
4,6,8,10,13,15	16
13,15,17,19,21,23	24
21,23,0,2,4,6	7
Halfrate Subchannel-1	
5,7,9,11,14,16	16
14,16,18,20,22,24	24
22,24,1,3,5,7	7

4.11.3 Speech Channels

In the speech channel the decoder metric and the decoder status are written to the shared memory locations **SM_TCH_METRIC** and **SM_TCH_STATUS**, where the decoder status can have one of the following values:

0. No error
1. FS, EFR and HS: Error in 3-bit CRC, AFS and AHS: Bad frame (CRC and metric are used)
2. Error in 8-bit CRC (EFR only)
3. Error in 3-bit CRC and 8-bit CRC (EFR only)

These two output values are written to the shared memory at the end of each channel decoder run for speech data: **SM_TCH_METRIC** and **SM_TCH_STATUS** are available in the shared memory after the end of slot #7 in TDMA frames 3, 7, 11, 16, 20, 24. This is the same for fullrate and halfrate speech.

Note: These two output values are not written by the DSP if an Facch has been received.

For DTX uplink control the DSP supplies the two shared memory values **SM_DTX_FLAG** and **SM_DTX_USED**.

SM_DTX_FLAG

SM_DTX_FLAG indicates if the next four TCH bursts (fullrate) or the next two TCH bursts (halfrate) have to be transmitted or not. **Table 4-7** shows when the **SM_DTX_FLAG** for uplink DTX control is available in the shared memory.

Table 4-7 Dtx Flag for Uplink Dtx Control

Bursts to be transmitted			SM_DTX_FLAG is valid after slot #3 in frame
Fullrate	Halfrate Subchannel-0	Halfrate Subchannel-1	
0,1,2,3	0,2	1,3	24
4,5,6,7	4,6	5,7	3
8,9,10,11	8,10	9,11	7
13,14,15,16	13,15	14,16	11
17,18,19,20	17,19	18,20	16
21,22,23,24	21,23	22,24	20

SM_DTX_USED

This control flag tells the MCU if DTX has ever been used within the Sacch reporting period or not. It is written to the shared memory 18 TDMA frames before the beginning of the next Sacch block. For example, if an Sacch block is transmitted in TDMA frames 25, 51, 77, and 103 the flag **SM_DTX_USED** is written to the shared memory in frame 7 (it is available in the shared memory after slot #3 in frame 7).

4.11.4 AMR Channels (AFS and AHS)

For AMR type speech channels (AFS for fullrate calls and AHS for halfrate calls) the same rules as introduced in **Section 4.11.3 “Speech Channels” on Page 77** for FS, EFR, and HS speech channels apply. This means the MCU must do the SACCH and FACCH control and the shared memory locations **SM_TCH_METRIC**, **SM_TCH_STATUS**, **SM_DTX_FLAG**, and **SM_DTX_USED** may be evaluated.

For an AMR type speech channel the MCU has to do the following tasks:

1. **Link Adaptation**:
   The MCU has to do the overall data rate control of the AMR connection on the mobile station side.
2. **RATSCCH Messages**:
   Similar to FACCH messaging, the MCU has to take control of RATSCCH messaging used in the AMR connection.
3. **AMR DRX Flag**:
   A flag has to be set/reset according to the DRX mode.

4.11.4.1 Link Adaptation

Active CODEC Set

The link adaptation approach in this firmware mask is based on GSM recommendation 05.09. One AMR speech connection parameters is the ’Active CODEC Set’, which has to be agreed upon before or in the beginning of the AMR call. The ’Active CODEC Set’ describes which subset of data rates (up to 4 different data rates) is used for the actual call. In AMR fullrate (AFS) there is an overall set of 8 data rates where this ’Active CODEC Set’ is selected from. In AMR halfrate (AHS) the overall set comprises 6 different data rates. **Table 4-8** shows the overall data rates allowed for AFS or AHS.

Table 4-8 Data rates used in AMR (Fullrate and Halfrate)

AFS	AHS
4,75 kbit/s	4,75 kbit/s
5,15 kbit/s	5,15 kbit/s
5,9 kbit/s	5,9 kbit/s
6,7 kbit/s	6,7 kbit/s
7,4 kbit/s	7,4 kbit/s
7,95 kbit/s	7,95 kbit/s
10,2 kbit/s	
12,2 kbit/s	

As described in the previous paragraph the 'Active CODEC Set' has to be selected from the overall set of different data rates allowed in an AFS or AHS call, respectively. In the firmware solution described here this agreement on the 'Active CODEC Set' has to be done by means of two shared memory locations that describe the 'Active CODEC Sets':

- **SM_AMR_ACS_UL** for the uplink direction
- **SM_AMR_ACS_DL** for the downlink direction.

Note: The uplink and downlink 'Active CODEC Sets' may be different.

The values for **SM_AMR_ACS_UL** and **SM_AMR_ACS_DL** have to be set up in the following way:

- Each data rate of **Table 4-8 Data rates used in AMR (Fullrate and Halfrate)**corresponds to one bit of the shared memory locations **SM_AMR_ACS_UL** or **SM_AMR_ACS_DL**. The data rate 4,75 kbit/s corresponds to bit #0 in **SM_AMR_ACS_UL**/**SM_AMR_ACS_DL**, data rate 5,15 kbit/s corresponds to their bit #1 and so on.
  If for example the 'Active CODEC Set' in uplink direction consists of the data rates 4,75 kbit/s, 6,7 kbit/s, and 7,95 kbit/s, a value of 29_H (= 0010 1001 in binary notation) must be written to the shared memory location **SM_AMR_ACS_UL**.
- Since the maximum number of data rates defined in the 'Active CODEC Set' is four the number of bits set to '1' in **SM_AMR_ACS_UL**/**SM_AMR_ACS_DL** must be less than or equal to four.
- The highest data rate allowed in an AHS connection is 7,95 kbit/s (see **Table 4-8**). Therefore, in an AHS call no bit more significant than bit #5 must be set in **SM_AMR_ACS_UL**/**SM_AMR_ACS_DL**.
- The shared memory location **SM_AMR_ACS_DL** can be written by the MCU in TDMA frames 0,1,2,4,5,6,8,9, and 10.
- The shared memory location **SM_AMR_ACS_UL** can be written by the MCU in TDMA frames 2,6, and 10 (valid for the IFX audio scheduler).

CODEC Mode Indication/Request/Command

For proper operation of the link adaptation mechanism in an AMR call there are in-band signaling parameters in GSM recommendation 05.09:

- In uplink direction, the mode indication parameter indicates the data rate used in the uplink direction and the mode request parameter informs the base station about the preferred data rate to be used in the downlink direction. These two parameters are also in shared memory locations **SM_AMR_MI_UL** and **SM_AMR_MR_UL**.
- In downlink direction, the mode indication parameter indicates the data rate used in the downlink direction and the mode command parameter informs the mobile station about the data rate that has to be used in the uplink direction. These two parameters are also in shared memory locations **SM_AMR_MI_DL** and **SM_AMR_MC_DL**.
- The values of these parameters (**SM_AMR_MI_UL**/**SM_AMR_MR_UL** and **SM_AMR_MI_DL**/**SM_AMR_MC_DL**) must not exceed the number of data rates that have been agreed in the 'Active CODEC set'.

For example, if three data rates have been defined in the 'Active CODEC Set' for the uplink direction **SM_AMR_MI_UL** must have a value of 0,1 or 2.

- The mode indication in downlink direction **SM_AMR_MI_DL** is an output parameter of the channel decoder. With every second speech frame (period of 20 ms) the channel decoder delivers an update for this parameter. For those speech frames where no update for the mode, indication is given by the channel decoder **SM_AMR_MI_DL** is also used as an input. Therefore, the MCU has to be careful that this parameter is initialized in a proper way.
- The downlink mode command parameter **SM_AMR_MC_DL** is also transmitted in the downlink direction and using this parameter the base station informs the mobile station what data rate to use in the uplink. This parameter also is updated by the channel decoder every second speech frame and, therefore, the MCU also has to initialize this parameter.
- The downlink mode command **SM_AMR_MC_DL** indicates what data rate in the uplink must be used. Since the data rate indicated by **SM_AMR_MC_DL** has not always to be used immediately the MCU has to update the shared memory location **SM_AMR_MI_UL** (mode indication in uplink) every frame. This guarantees that the uplink uses the right data rate even if the mobile is in handover or the base station requests a step more than one in the Active Code Set.
- The uplink mode request **SM_AMR_MR_UL** has to be calculated by the MCU. The MCU has to evaluate the receive parameters (equalizer quality, channel decoder metric, etc.) and to determine the optimum data rate for the downlink connection. If the downlink quality is good, a higher data rate my be selected. If the downlink quality is poor, a lower data rate has to be selected. This optimum data rate to be used in downlink direction has to be transmitted to the base station as a mode request parameter.

Table 4-9 Summary on Mode Indication/Command/Request

Shared Memory Location	Description of Contents
SM_AMR_MI_DL	Mode Indication in Downlink Direction • Update done by channel decoder • Initialization required by MCU
SM_AMR_MC_DL	Mode Command in Downlink Direction (requested data rate for uplink) • Update done by channel decoder • Initialization required by MCU
SM_AMR_MI_UL	Mode Indication in Uplink Direction • Used by speech encoder (DSP internally forwarded to the channel encoder) • Update required by MCU within every frame • Must be updated continuously
SM_AMR_MR_UL	Mode Request in Uplink Direction • By means of measurements on downlink quality (equalizer output, decoder metric, etc.) MCU has to determine mode request for downlink data rate. • Used by channel encoder • Written by MCU • Must be updated continuously

Since **SM_AMR_MR_UL** and **SM_AMR_MI_UL** are used by the channel encoder the MCU has to update this parameter before the frame interrupt in TDMA frames 3, 7, 11, 16, 20, and 24 (when counting modulo 26) and it must not modify **SM_AMR_MR_UL** during these TDMA frames.

The other two shared memory locations **SM_AMR_MI_DL** and **SM_AMR_MI_UL** (**SM_AMR_MI_UL** is identical to **SM_AMR_MC_DL**) only have to be initialized by the MCU before a new AMR call is started.

Table 4-10 gives an overview when these three shared memory locations have to be written by the MCU.

Table 4-10 MCU Modify on Mode Indication/Command/Request

	MCU Modify required
SM_AMR_MI_DL	MCU initialization before new AMR call
SM_AMR_MC_DL	
SM_AMR_MI_UL	MCU update before each channel encoder run (TDMA frames 3, 7, 11, 16, 20, 24)
SM_AMR_MR_UL	

CODEC Mode Phase (Even or Odd)

The transmitter - receiver synchronization parameter called 'CODEC Mode Phase' (GSM recommendation 05.09) decides if the CODEC information (mode indication or mode command) is transmitted with even or odd phase in the downlink direction. Its shared memory location is **SM_AMR_MI_EVEN**.

This parameter has to be set to '0' if the 'CODEC Mode Phase' is odd, and it has to be set to '1' if the 'CODEC Mode Phase' is even. This must done (only once) before the start of the AMR call or after a phase change due to a RATSCCH message.

4.11.4.2 RATSCCH Messages

The basic approach of handling Ratscch messages is identical to handling Facch messages:

Uplink

If a Ratscch message has to be transmitted in the uplink, the MCU has to **SM_RATSCCH_TX_FLAG** to '1' and it has to write the Ratscch information to **SM_RATSCCH_TX_DATA**. The size of the field **SM_RATSCCH_TX_DATA** is 3 words and the first bit of the 35 bit Ratscch message has to be written in the LSB of its first word.

The Ratscch flag and the Ratscch information have to be written to the shared memory by the MCU before the frame interrupt in TDMA frames 3, 7, 11, 16, 20 and 24 (when using modulo 26 to count in the TDMA counter).

Notes:

1. *Sending an Facch has higher priority than a RATSCCH*
2. **SM_RATSCCH_TX_FLAG** *is reset to '0' when the Ratscch has been transmitted.*
3. *A Ratscch message might be delayed due to a high priority Facch message. In this case, the MCU polls* **SM_RATSCCH_TX_FLAG** *until it becomes '0' again to find out when the Ratscch has be*en transmitted.

Downlink

If a Ratscch message has been received, the Ratscch Receive Flag **SM_RATSCCH_RX_FLAG** is set to '1' and the Ratscch Receive Data is written to the 5-word shared memory block **SM_RATSCCH_RX_DATA** that holds the following information (in order):

1. RATSCCH_RX_METRIC Channel decoder metric (1 word, max. value: 212)
2. RATSCCH_RX_STATUS Channel decoder status (1 word)
   0. No error in CRC, decoded data is valid
   1. Error in CRC, but decoded data is valid (error has been corrected)
2. RATSCCH_RX_DATA 35 bit output data stored in three 16-bit words.
   a) Word 0 = D(15:0)
   b) Word 1 = D(31:16)
   c) Word 2 = D(34:32)

The Ratscch Rx information (**SM_RATSCCH_RX_FLAG** and **SM_RATSCCH_RX_DATA**) is written at the end of the channel decoder run. This means: this kind of information is available at the end of TDMA frames 3, 7, 11, 16, 20 and 24 (when counting the TDMA counter modulo 26).

Note: If no Ratscch has been detected the flag **SM_RATSCCH_RX_FLAG** *is reset to '0'.*

4.11.4.3 Frame Types in AMR

In the AMR implementation described here there are two frame types (one for uplink, Tx Frame Type, and one for downlink, Rx Frame Type). These two frame types are mainly for DSP internal use but they are written to the shared memory for debugging purposes. They may be read by the MCU at the locations **SM_AMR_TX_TYPE** and **SM_AMR_RX_TYPE**. Reading these two frame types tells the MCU what kind of frame type the DSP has processed internally:

1. Tx Frame Type
   Evaluating **SM_AMR_TX_TYPE** the MCU can check what kind of data the DSP has encoded in the previous frame. **Table 4-11** shows all values of **SM_AMR_TX_TYPE** that are allowed for an AFS call, **Table 4-12** shows the set of possible values for an AHS call.
2. Rx Frame Type
   Evaluating **SM_AMR_RX_TYPE** the MCU can check what kind of data the DSP has decoded in the previous frame. **Table 4-13** shows all values of **SM_AMR_RX_TYPE** that are allowed for an AFS call, **Table 4-14** shows the set of possible values for an AHS call.

Note: For the meaning of 'SID-First' frames, 'SID-Update' frames and 'No-Data' frames refer to GSM recommendation 06.93.

Table 4-11 Possible Tx Frame Types in an AMR Fullrate Call (AFS)

SM_AMR_RX_TYPE Value	Description
0	Speech Frame
1	'SID-First' Frame
2	'SID-Update' Frame
3	'No-Data' Frame
4	'FACCH' Frame
5	'RATSCCH' Frame

Table 4-12 Possible Tx Frame Types in an AMR Halfrate Call (AHS)

SM_AMR_TX_TYPE Value	Description
0	Speech Frame
1	First part of 'SID-First' Frame
2	First part of 'SID-Update' Frame
3	'No-Data' Frame
4	First part of 'FACCH' Frame
5	First part of 'RATSCCH' Frame
6	Second part of 'SID-First' Frame
7	Second part of 'SID-Update' Frame
8	Second part of 'FACCH' Frame
9	Second part of 'RATSCCH' Frame

Table 4-13 Possible Rx Frame Types in an AMR fullrate call (AFS)

SM_AMR_RX_TYPE Value	Description
0	Speech Frame
1	'SID-First' Frame
2	'SID-Update' Frame
3	'RATSCCH' Frame
4	'FACCH' Frame

Table 4-14 Possible Rx Frame Types in an AMR halfrate call (AHS)

SM_AMR_RX_TYPE Value	Description
0	Speech Frame
1	First part of 'SID-First' Frame
2	Inhibit of 'SID-First' Frame
3	Second part of 'SID-First' Frame
4	'SID-Update' Frame
5	Inhibit of 'SID-Update' Frame
6	First part of 'RATSCCH' Frame
7	Second part of 'RATSCCH' Frame
8	First part of 'FACCH' Frame
9	Second part of 'FACCH' Frame

Note: According to the ETSI specification, with the first speech frame after a silence period an ONSET frame is also transmitted by the BTS. If the DSP detects an ONSET frame, it signals to the MCU by setting bit **SM_AMR_RX_TYPE**[8] *to 1.*

The channel encoder and the channel decoder are run in TDMA frames 3, 7, 11, 16, 20 and 24 counting with modulo 26. Therefore. the shared memory parameters **SM_AMR_TX_TYPE** and **SM_AMR_RX_TYPE** must not be read before the end of these TDMA frames (to find out what has been encoded or decoded in the previous run).

Note: The 'SID-Update' frame has to be delayed by four frames.

4.11.4.4 AMR DRX Flag

The shared memory location **SM_AMR_DRX_FLAG** is used for AMR channels and has to be set in DRX mode, that is, when the BTS does not transmit speech frames due to a silence frame phase.

After a silence frame has been detected, the MCU must set the flag to '1'. After an ONSET, the MCU must reset the flag to '0'.

In bad channels the SID_FIRST or ONSET might not be detected correctly. In this case, the recommended procedure is:

- If a SID_FIRST or a SID_UPDATE has been detected, set **SM_AMR_DRX_FLAG** to '1'.
- If an ONSET or a speech frame has been detected, set **SM_AMR_DRX_FLAG** to '1'.

4.11.5 Data Channels

Setup of a default Data Channel in the E-GOLDradio means data is directly passed from the MCU to the DSP via shared memory to the channel encoder input and the channel decoder output is directly passed via the shared memory to the MCU.

Uplink

The source data for the uplink direction is read from:

- **SM_TCH_TX_DATA_0**, which is used for single slot channel types and the mainstream in HSCSD
  SM_TCH_TX_DATA_1, which is used for uplink sub-stream #1 in HSCSD.

The contents of these 23-word shared memory blocks depends on the data service mode and the data service speed selected in the TCH26 channel.

The source data for TCH uplink has to be available in the shared memory before the frame interrupt in the TDMA frames 3,7,11,16,20,24.

Downlink

The results of the downlink direction are written to:

SM_TCH_RX_DATA_0, which is used for single slot channel types and the mainstream in HSCSD

SM_TCH_RX_DATA_1/2/3, which are used for downlink sub-streams #1/2/3 in HSCSD.

The first word of these 24-word shared memory blocks always contains:

- `TCH_RX_METRIC` Channel decoder metric (1 word). The maximum possible value of `TCH_RX_METRIC` is < 456 and is independent of the data channel type.
- `TCH_RX_DATA` Output data (23 words), which depends on the data service mode and the data service speed

In downlink direction the availability of the output data depends on the selected channel type:

1. For interleaving-8 in fullrate (F2.4) and interleaving-19 in halfrate (H2.4 and H4.8), the output data is available in the shared memory at the end of timeslot #7 in TDMA frames 3,7,11,16,20,24.
2. For interleaving-19 in fullrate (F4.8, F9.6, and F14.4) and for HSCSD with more than one Rx timeslot, the output data is available in the shared memory at the end of timeslot #7 in TDMA frames 1,5,9,14,18,22.

Table 4-15 shows when the source data for uplink direction is required and when the results are available in downlink direction.

Table 4-15 Tx and Rx for Data Channels

Source data necessary in uplink direction	Before frame interrupt in TDMA frames 3,7,11,16,20,24
Results available in downlink direction for F2.4, H2.4 and H4.8	After slot #7 in TDMA frames 3,7,11,16,20,24
Results available in downlink direction for F4.8, F9.6 and F14.4 (even for HSCSD with more than one Rx timeslot)	After slot #7 in TDMA frames 1,5,9,14,18,22

When a new data service channel is setup, the parameter 'Mode' has to be set to '4', which means data is passed directly to and from Channel CODEC.

The following examples show how the **TCH_26** parameters have to be set for the setup of an HSCSD channel.

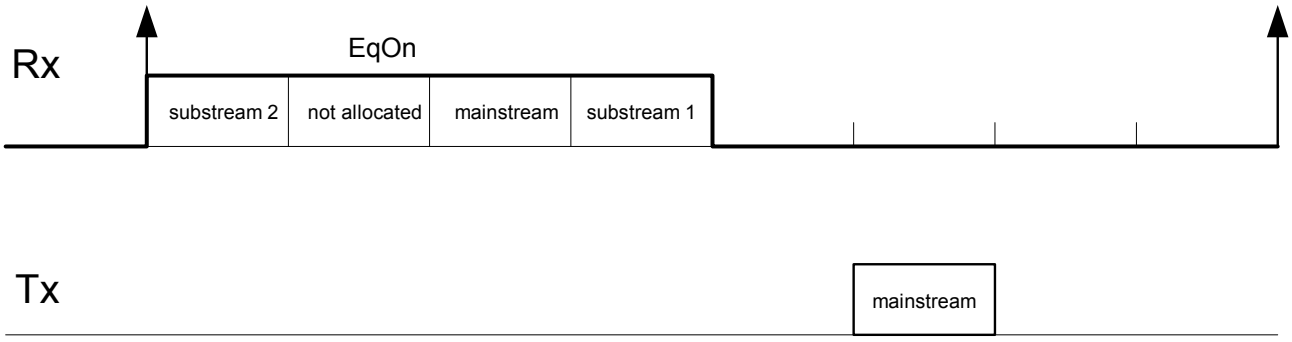
HSCSD Example 1

In this example the parameters RX0/1/2/3 and TX0/1 have to be set as follows (see **Figure 4-9**):

- RX0: 2
- RX1: -1 (not used)
- RX2: 0 (mainstream)
- RX3: 1
- TX0: 0
- TX1: -1 (not used).

*Note: The equalizer output parameters of sub-stream 2 is written to **SM_EQUAL_0**, the equalizer output parameters of the mainstream to **SM_EQUAL_2**, and the equalizer output parameters of the sub-stream 1 to **SM_EQUAL_3**. The shared memory field **SM_EQUAL_1** is not used in this example.*

Figure 4-9 HSCSD Example 1



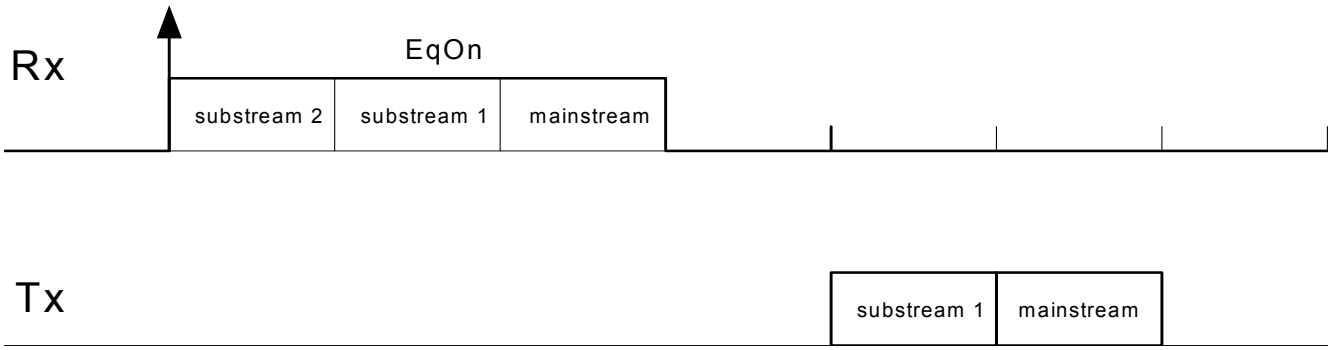
HSCSD Example 2

In this example the parameters RX0/1/2/3 and TX0/1 have to be set as follows (see **Figure 4-10**):

- RX0: 2
- RX1: 1
- RX2: 0 (mainstream)
- RX3: -1 (not used)
- TX0: 1
- TX1: 0 (mainstream)

*Note: The equalizer output parameters of sub stream 2 is written to **SM_EQUAL_0**, the equalizer output parameters of the sub-stream 1 to **SM_EQUAL_1**, and the equalizer output parameters of the mainstream to **SM_EQUAL_2**. The shared memory field **SM_EQUAL_3** is not used in this example.*

Figure 4-10 HSCSD Example 2



4.12 PDCH Mode

The PMB 7870 firmware supports a packet switched mobile class 12 (4 Rx and 1 Tx or 1 Rx and 4 Tx) in the class B mode of operation (paging of circuit switched and packet switched services simultaneously) through additional scheduler functions. The PDCH mode is suitable for GPRS. All coding schemes CS1-CS4 are available.

The MCU can switch the DSP subsystem to PDCH mode by giving the command **PDCH**. This command and the parameter related to this command are explained in **Section 3.3.11 “PDCH” on Page 46**. When using this command:

- If the DSP is in Stand-By mode, the MCU can apply the **PDCH** command to force the DSP subsystem into PDCH mode. Since this is a synchronous command, it is accepted immediately but does not become valid until the frame interrupt in the next TDMA frame.
- **PDCH** has a parameter "MODE" that tells the DSP that the PDCH mode is GPRS.

- The MCU must insure that the shared memory TDMA counters (**SM_COUNTER_104**, **SM_COUNTER_51**, and **SM_SFNUM**) are set correctly before the first Frame Interrupt occurs after the command is given. The PDCH mode does not work correctly without these counters.
- Before the MCU applies the **PDCH** command, it must initialize the shared memory locations **SM_RX_INFO**, **SM_PDTCH_TX_DATA_0**/1, and **SM_TX_INFO**.
- The DSP subsystem has only one PDCH mode scheduler which covers the Packet Idle mode and the Packet Transfer mode.
- The PDCH mode can be terminated by sending the **IDLE** or **BB_OFF** command.

The PDCH mode has two independent state machines:

1. One to handle radio blocks (for example PACCH, PDTCH)
2. One for the Timing Advance Control Channel (PTCCH).
   A so called radio block can either be a MAC/RLC control block or an RLC data block. For the DSP there is no difference between data blocks and control blocks.

- A small and flexible interface reduced to four TDMA frames (radio block size) handles radio blocks, therefore, every radio block the DSP subsystem needs pre-information about the number of Rx timeslots (Rx constellation), the number of Tx timeslots (Tx constellation), and the data for the next radio block.
- The PTCCH state machine receives and transmits the Timing Advance Control Channel while the mobile is in packet transfer mode. This state machine is controlled by the MCU signals CODON and EQON.

4.12.1 Packet Idle Mode

- In Packet Idle mode, the mobile regularly decodes Broadcast Channels (PBCCH) and Paging Channels (PPCH): during the login phase to the network, the mobile listens to the Common Control Channel (CCCH) of the base station. A 51 multiframe carries the CCCH.
- After receiving some network information the network can force the GPRS mobile to switch to the Packet Common Control Channel (PCCCH), which is based on a 52-multiframe.

CCCH

If the GPRS mobile camps on the CCCH, the MCU can only use the **CCH_RX** (**Page 42**) and **CCH_TX** (**Page 42**) commands. The MCU gets access to the CCCH in the same way as in the circuit switched mode.

PCCCH

If the GPRS mobile has to camps on the PCCCH, the DSP provides two possibilities for receiving and transmitting a radio block which contains control information:

- **CCH_RX and CCH_TX**
- **PDCH Mode**.

4.12.1.1 CCH_RX and CCH_TX

These commands are also usable in a non 51-multiframe. As the Ciphering algorithms A51, A52 and A53 are not used in GPRS, the MCU has to switch them off by setting the parameter CIPH to ’0’. In this situation the DSP does not use the TDMA counters.

If there are any subsets of blocks statically allocated on a PCCCH reserved for PRACHs, the MCU can use the process in **Section 4.10 “Access Burst (RACH)” on Page 72** to send PRACHs.

Note: If the PRACH on a PCCCH is dynamically allocated by indicating blocks with the USF = FREE, the MCU cannot use the commands CCH_RX and CCH_TX. The MCU must switch to the PDCH mode.

4.12.1.2 PDCH Mode

When using the PDCH mode for accessing the PCCCH the receiving and transmitting radio blocks procedure is the same as in Packet Transfer mode. This means that the PDCH mode expects a Frame Interrupt every TDMA

frame including IDLE frames (refer to **Section 4.4 "Frame Interrupt" on Page 65**). To reduce power consumption during DTX mode, it is not necessary wakes up the DSP on every TDMA frame with a Frame Interrupt if the MCU doesn't want to receive anything. For this use the following procedure is recommended:

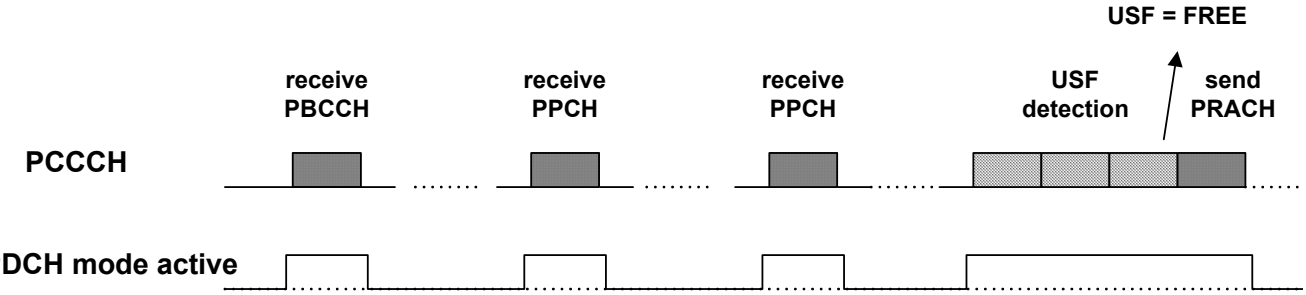
- To start receiving a Packet Paging Channel (PPCH) or a Packet Broadcast Control Channel (PBCCH) the MCU forces the DSP into PDCH mode by sending the command **PDCH**

Note: The Frame counters must have been correctly set before sending the command.

- After receiving and decoding the PPCH or PBCCH the MCU sends the command **IDLE** or **BB_OFF** to the DSP.
- The same process can be used to send a PRACH if the blocks are dynamically allocated. In this case the PDCH mode has to be active at the beginning of the first downlink radio block for USF detection. Depending on the result of this USF detection, the PRACH may or may not be transmitted.

Figure 4-11 is a timing diagram for paging on the PCCCH and sending a PRACH in dynamic allocation.

Figure 4-11 Paging On the 52 Multiframe in PDCH Mode



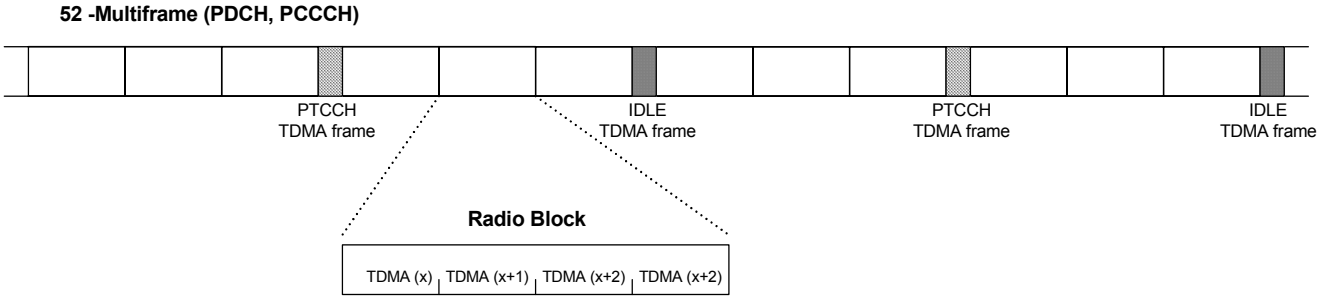
*Note: The procedure for receiving and transmitting a radio block on the PCCCH is the same as in Packet Transfer mode (refer to **Section 4.12.2 Packet Transfer Mode**).*

4.12.2 Packet Transfer Mode

To provide a flexible interface, the PDCH mode is reduced to a "GPRS Packet Data Unit". This radio block is completely independent of the previous radio block and the next radio block.

The PDCH is mapped on a 52-multiframe consisting of two idle frames, two PTCCH frames, and 12 radio blocks. In the enlarged radio block of a 52-multiframe shown in **Figure 4-12** a radio block consists of 4 bursts, each in one TDMA frame. The DSP subsystem needs a counter 4 and a counter 104 that locates a radio block in the 52-multiframe and processes the PTCCH (refer to **Section 4.12.5 "PTCCH" on Page 100**). The DSP subsystem also uses the counter 104 for the internal memory management.

Figure 4-12 Radio Block



The interface is based on the MCU giving the DSP pre-information about the next radio block. After processing this radio block, the DSP subsystem gets the pre-information for the next radio block and so on. With this pre-information, the DSP gets information about the channel allocation (number of Rx and Tx timeslots), the channel encoder type, and what is to be done with the received data (for example, a USF detection). The DSP must write the results of the previous radio block to the shared memory before a guaranteed point in time.

The channel constellation can be completely different from one radio block to another. For instance, there might be 4 Rx and no Tx allocated during one radio block but only 1 Rx and 4Tx in the next radio block.

Note: Within a radio block, there is no need for a change in the channel constellation.

The interface is optimized for MCU access to the shared memory. That means the DSP fetches the pre-information data from the shared memory just before it is used. That gives the MCU a long time to prepare the shared memory data. The exact timing for transmitting and receiving radio blocks is given in **Section 4.12.2.1 "Receiving Radio Blocks" on Page 88** and **Section "Transmitting Radio Blocks" on Page 95**.

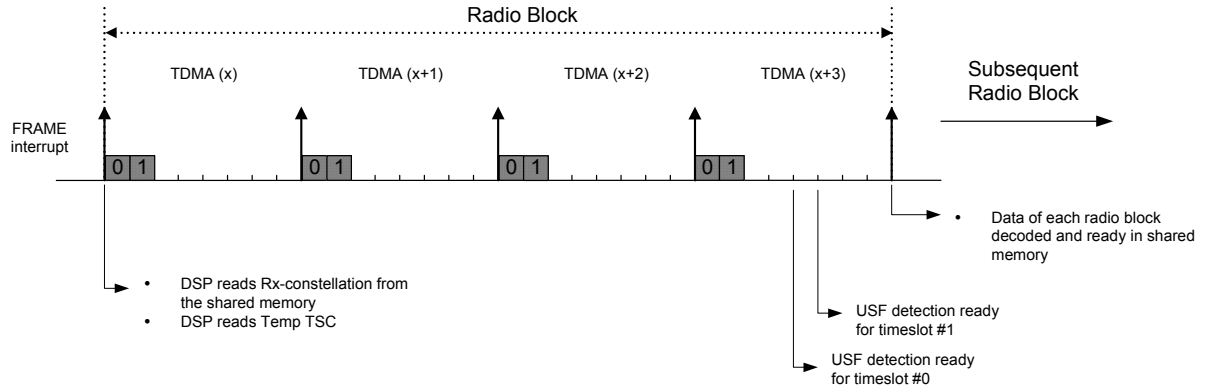
Note: There is no difference between Control Blocks (PACCH) and Data Blocks (PDTCH) for the DSP.

4.12.2.1 Receiving Radio Blocks

This section is an overview about receiving radio blocks. It describes the time when the data has to be written in the shared memory and specifies the time when the MCU can read the results.

As an example, consider four TDMA frames that correspond to the length of a radio block (see **Figure 4-13** for GPRS). As there are two Rx timeslots allocated the MCU can receive two radio blocks. It does not matter which radio block we are looking at, Since the timing of each radio block on a 52-multiframe is the same, it does not matter which radio block is used in this example.

Figure 4-13 GPRS Timing for Receiving Radio Blocks



For GPRS there are three points over a time of 4 TDMA frames where the DSP or the MCU expects data or control information:

1. The DSP reads the **Rx Constellation** (the Rx information and the temporary training sequence) from the shared memory provided by the MCU.
2. The **USF Detection Result** is available in the shared memory.
3. The **Channel Decoder Output Data** (received radio blocks) are available in the shared memory.

Rx Constellation

The DSP always fetches the Rx information from the shared memory on the first Frame Interrupt of a radio block (TDMA x). **Table 4-16** shows when the Rx constellation must be available in the shared memory locations **SM_RX_INFO**, and **SM_RX_TEMP_TSC**.

Table 4-16 Rx Info (Within the 52-Multiframe)

Radio Blocks Are Received in Frames:	Rx Constellation Must Be Available Before Frame Interrupt in Frame:
0,1,2,3	0
4,5,6,7	4
8,9,10,11	8
13,14,15,16	13

Table 4-16 Rx Info (Within the 52-Multiframe)

Radio Blocks Are Received in Frames:	Rx Constellation Must Be Available Before Frame Interrupt in Frame:
17,18,19,20	17
21,22,23,24	21
26,27,28,29	26
30,31,32,33	30
34,35,36,37	34
39,40,41,42	39
43,44,45,46	43
47,48,49,50	47

The shared memory location **SM_RX_INFO** consists of four words. Each word is reserved for exactly one Rx timeslot. From this memory location the DSP gets the information about what to do with the received bursts of this timeslot (refer to **Table 4-17**).

Table 4-17 Tasks (words of SM_RX_INFO)

Value	Tasks for Received Bursts
-1	Do nothing with the received bursts: the equalizer and channel decoder are not started if the timeslot is activated
0	Check which coding scheme has been used by the base station for this allocated timeslot so that can the DSP start the appropriate channel decoder, For GPRS: CS1, CS2, CS3, or CS4.
	Start USF detection depending on the recognized coding scheme.

Note: The first activated downlink timeslot belongs to the first word of the shared memory block **SM_RX_INFO**, *the second activated downlink timeslot belongs to the second word of* **SM_RX_INFO**, *and so on.*

The location of the output data of the USF detection, the channel decoder, and the output parameter of the equalizer depends on the allocated timeslots. If there are subsequent timeslots, then the equalizer, channel decoder, and USF output of the first allocated timeslot is written to **SM_EQUAL_0**/**SM_PDTCH_RX_DATA_0**/**SM_USF_RESULT_0** and the output of the second activated timeslot to **SM_EQUAL_1**/**SM_PDTCH_RX_DATA_1**/**SM_USF_RESULT_1** and so on.

According to the GSM 45.002 standard, the number of Rx timeslots shall be allocated within a four-timeslot window (class 12 mobile). This means even if there are only two timeslots allocated, there can only be two gaps between them. The DSP subsystem supports two different applications to handle this case by using the Rx information field **SM_RX_INFO**:

1. **EQON Continuous**
2. **EQON with Gaps**.

EQON Continuous

The EQON signal from the system interface is set over the whole window size even if the MCU does not receive bursts. The MCU has to be careful that the baseband filter is switched on because the DSP must receive a baseband buffer full interrupt every timeslot.

In a baseband buffer full interrupt, the DSP refers to the fetched Rx information field to know what has to be done with the received burst.

Figure 4-14 shows an example for this application and **Table 4-18** contains the appropriate usage of the **SM_RX_INFO** field.

Figure 4-14 EQON Continuous

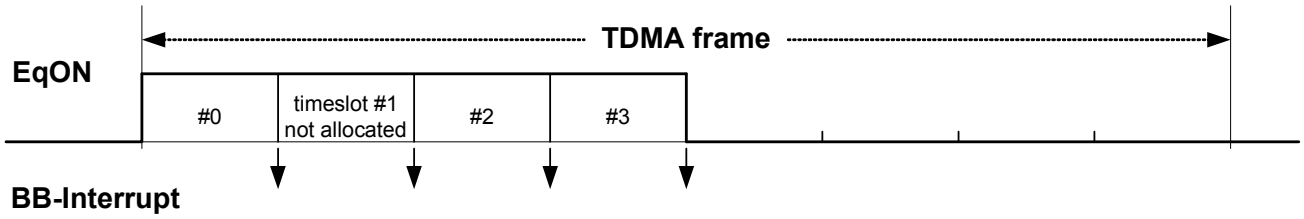


Table 4-18 Usage of SM_RX_INFO for EQON Continuous

Word	Contents	Tasks Description	Used Shared Memory Output Buffers
1	0	Start USF detection and channel or header decoding	SM_EQUAL_0 SM_USF_RESULT_0 SM_PDTCH_RX_DATA_0
2	-1	Nothing to do within this interrupt	
3	0	Start USF detection and channel or header decoding	SM_EQUAL_2 SM_USF_RESULT_2 SM_PDTCH_RX_DATA_2
4	0	Start USF detection and channel or header decoding	SM_EQUAL_3 SM_USF_RESULT_3 SM_PDTCH_RX_DATA_3

EQON with Gaps

The EQON signal is only set in those timeslots where the MCU wants to receive a burst. As shown in **Figure 4-15** there is a gap in the EQON signal during timeslot #1. Therefore, the DSP subsystem gets a baseband buffer full interrupt on every timeslot except for timeslot #1.

The algorithm inside the DSP subsystem is the same for **EQON Continuous**. It uses the fetched Rx information field and to know what it does with the received bursts. Since in this case the DSP subsystem does not get an interrupt in timeslot #1, the **SM_RX_INFO** field is used differently as shown in **Table 4-19**.

The MCU cannot write any information for timeslot #1 in the Rx constellation field **SM_RX_INFO**.

Figure 4-15 EQON with Gaps

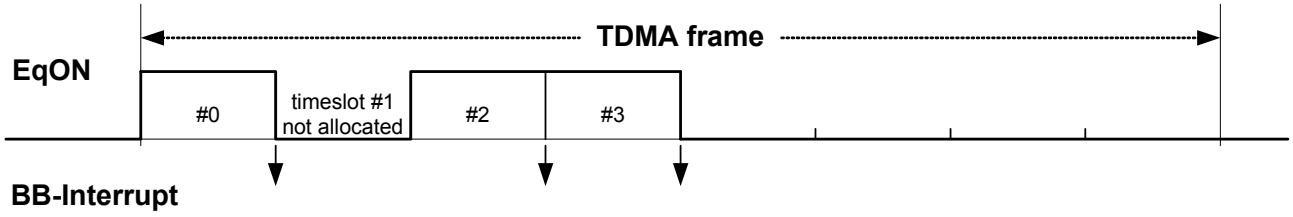


Table 4-19 Usage of SM_RX_INFO In the Case of EQON with Gaps

Word	Content	Tasks Description	Used Shared Memory Output Buffers
1	0	Start USF detection and channel decoding	SM_EQUAL_0 SM_USF_RESULT_0 SM_PDTCH_RX_DATA_0
2	-1	Do nothing	Shared memory not used
3	0	Start USF detection and channel or header decoding	SM_EQUAL_1 SM_USF_RESULT_1 SM_PDTCH_RX_DATA_1
4	0	Start USF detection and channel or header decoding	SM_EQUAL_2 SM_USF_RESULT_2 SM_PDTCH_RX_DATA_2

These two applications can be mixed by the MCU. For example, if the GPRS mobile is not allowed to send in timeslot three (see **Figure 4-15**), in the next radio block the MCU may leave the EQON signal as it is. But, it has to change the third word of **SM_RX_INFO** to '-1' (nothing to do).

The shared memory location **SM_RX_TEMP_TSC** is used to temporarily set another training sequence as given by the MCU in the **PDCH** command. This new training sequence might be useful if the MCU wants to receive, for example, a PBCCH (currently using another training sequence) without leaving the PDCH mode. This is an exception, usually there is no need for a change of the training sequence within a PDCH mode.

The procedure is as follows:

1. At the same time that the shared memory location **SM_RX_INFO** is read, the DSP also reads **SM_RX_TEMP_TSC**.
2. If the value in **SM_RX_TEMP_TSC** is -1, the DSP uses the training sequence specified by the **PDCH** command.
3. Otherwise, the value is interpreted by the DSP as a temporarily training sequence index for the current radio blocks. All received bursts are equalized with this temporarily training sequence. It is also used for the uplink bursts if any timeslots are allocated. Allowed values are 0...7, refer to **Section 3.3.11 "PDCH" on Page 46**. This training sequence is valid until the next time the Rx constellation is read.

USF Detection Result

The DSP starts the equalization algorithm immediately after receiving a burst if the appropriate word of the **SM_RX_INFO** field contains '0'. The equalizer parameter is written to the shared memory according to the allocated timeslot, refer to **Table 4-18** and **Table 4-19**. When the equalization of the last burst of a radio block is done the DSP starts the decoding of the radio block as follows:

1. The DSP looks at the stealing flags and determines the channel CODEC type used by the base station.
2. The DSP starts the USF detection algorithm. USF detection for CS1 is based on a short VITERBI algorithm, USF detection for CS2, CS3, and CS4 is done by correlation.
3. The DSP writes the results of the USF detection to the proper shared memory location **SM_USF_RESULT_0**/1/2/3 (refer to **Table 4-18** and **Table 4-19**). Four words are reserved for each shared memory location (refer to **Table 4-20**).
4. The appropriate channel decoder is initiated.

These steps (1...4) are repeated for each allocated timeslot.

Table 4-20 Contents of SM_USF_RESULT_0/1/2/3

Word	Description	Contents	
0	Coding scheme	0	CS1
		1	CS2
		2	CS3
		3	CS4
1	USF	0,1,2,...,7	Detected USF flag
2	Reliability	CS1	Metric value; range [0...28]; best quality: 0
		CS2-CS4	Correlation value; range [-180...180]; best quality: 180
3		CS1	Sum of the 6 soft bits, which represents the USF; range [0...90]; best quality 90
		CS2-CS4	next lower correlation value; range [-180...<<180]

In Table 4-21 are the times when the USF detection results are available in the shared memory.

Table 4-21 Results of USF Detection

Radio Blocks Received in Frames (Maximum 4 Ts)	USF Detection Is Available After the End of Frame/#slot			
	TS 0	TS 1	TS 2	TS 3
0,1,2,3	3, #3	3, #4	3, #5	3, #6
4,5,6,7	7, #3	7, #4	7, #5	7, #6
8,9,10,11	11, #3	11, #4	11, #5	11, #6
13,14,15,16	16, #3	16, #4	16, #5	16, #6
17,18,19,20	20, #3	20, #4	20, #5	20, #6
21,22,23,24	24, #3	24, #4	24, #5	24, #6
26,27,28,29	29, #3	29, #4	29, #5	29, #6
30,31,32,33	33, #3	33, #4	33, #5	33, #6
34,35,36,37	37, #3	37, #4	37, #5	37, #6
39,40,41,42	42, #3	42, #4	42, #5	42, #6
43,44,45,46	46, #3	46, #4	46, #5	46, #6
47,48,49,50	50, #3	50, #4	50, #5	50, #6

The PDCH mode provides an interrupt after USF detection is finished. The command **MCU_INT** has to be sent (once) when PDCH mode is started to switch on or off the interrupt feature in the PDCH mode. The decision to send an interrupt depends on the shared memory location **SM_PDCH_USF_INT**. If there is a gap between two timeslots, the same rules are valid as for the shared memory indices for the USF/decoder output data (refer to **Section 4.12.2.1 “Receiving Radio Blocks” on Page 88**).

Table 4-22 SM_PDCH_USF_INT

Word	Related Timeslot	Description	
0	#0	-1	No interrupt after USF detection
		0	Interrupt is generated by DSP after USF detection
1	#1	-1	No interrupt after USF detection
		0	Interrupt is generated by DSP after USF detection

Table 4-22 SM_PDCH_USF_INT

Word	Related Timeslot	Description	
2	#2	-1	No interrupt after USF detection
		0	Interrupt is generated by DSP after USF detection
3	#3	-1	No interrupt after USF detection
		0	Interrupt is generated by DSP after USF detection

The signal TOMCU0 to the MCU is used for the USF interrupt. Since the same interrupt signal is given for each USF interrupt independent of the timeslot, it is recommended to enable only one USF interrupt per TDMA frame.

*Note: The shared memory location **SM_PDCH_USF_INT** is not fetched by the DSP at the beginning of the first TDMA frame of a radio block. To avoid any shared memory conflicts between the DSP and MCU these shared memory locations must be written by the MCU before the PDCH mode is applied. Alternatively, it can be written by the MCU just after reading the last USF result within a TDMA frame.*

Channel Decoder Output Data

After the coding scheme and USF detection as described in **USF Detection Result**, the appropriate channel/header decoder is started by the DSP.

GPRS:

- The radio block consists of the header and the RLC data. Both are decoded with one of the channel decoders CS1, CS2, CS3, or CS4. When the channel decoder is finished the data is written to the shared memory location **SM_PDTCH_RX_DATA_0**/1/2/3 according to the allocated timeslots as it is shown in **Table 4-18** and **Table 4-19**.
- The size of the output data blocks depends on the use of the data rate. The maximum size for CS4 is 30 words and holds the information as described in **Table 4-23**.

Table 4-23 Contents of Channel Decoder Output Data Blocks

Word	Description	Contents		
0	Metric	Maximum value: CS1: 456/ CS2: 588/ CS3: 676/ CS4: undefined)		
1	BCS	0	BCS check ok	
		1	CS1	Error occurred and corrected (with fire decoder)
			others	Error occurred
		2	CS1	Error occurred
			CS4	Error occurred and corrected
2	Decoder type	0	Decoded with CS1	
		1	Decoded with CS2	
		2	Decoded with CS3	
		3	Decoded with CS4	
3	Data	bit[15:0]		
...				
14	Data	bit[183:176] for CS1		
...				
19	Data	bit[270:256] for CS2		
...				
22	Data	bit[314:304] for CS3		

Table 4-23 Contents of Channel Decoder Output Data Blocks

Word	Description	Contents
...		
29	Data	bit[430:416] for CS4

Note: The MCU has always to write the value '-1' to the first word of the shared memory locations **SM_PDTCH_RX_DATA_0**/1/2/3 *immediately after reading the results. This enables the MCU i to detect if there has been a channel decoder run or not.*

The channel and header decoder is repeated for each allocated timeslot. It is guaranteed that the results are available at the end of the last TDMA frame of a radio block even if there are more than one Rx timeslot allocated. **Table 4-24** indicates the time when channel/header decoder results are available in the shared memory.

Table 4-24 Results of the Channel Decoder

Radio Blocks Received In Frames: (Max. 4 Ts)	Channel Decoder Data of Four Radio Blocks Are Available After the End of Frame/#slot:
0,1,2,3	3, #7
4,5,6,7	7, #7
8,9,10,11	11, #7
13,14,15,16	16, #7
17,18,19,20	20, #7
21,22,23,24	24, #7
26,27,28,29	29, #7
30,31,32,33	33, #7
34,35,36,37	37, #7
39,40,41,42	42, #7
43,44,45,46	46, #7
47,48,49,50	50, #7

The PDCH mode provides an interrupt after channel/header decoding is finished. The command **MCU_INT** has to be sent (once) when PDCH mode is started to switch on or off the interrupt feature. The decision to send an interrupt depends on the shared memory location **SM_PDCH_DEC_INT**. If there is a gap between two timeslots, the same rules are valid as for the shared memory indices for the USF/decoder output data (refer to **Section 4.12.2.1 "Receiving Radio Blocks" on Page 88**).

Table 4-25 SM_PDCH_DEC_INT

Word	Related Timeslot	Description	
0	#0	-1	No interrupt after channel decoder
		0	Interrupt is generated by DSP after channel decoder
1	#1	-1	No interrupt after channel decoder
		0	Interrupt is generated by DSP after channel decoder
2	#2	-1	No interrupt after channel decoder
		0	Interrupt is generated by DSP after channel decoder
3	#3	-1	No interrupt after channel decoder
		0	Interrupt is generated by DSP after channel decoder

The signal TOMCU1 to the MCU is used for the channel/header interrupt. Since the same interrupt signal is given for each decoder interrupt independent of the timeslot, it is recommended to enable only one interrupt per TDMA frame.

Note: The shared memory location **SM_PDCH_DEC_INT** *is not fetched by the DSP at the beginning of the first TDMA frame of a radio block. To avoid any shared memory conflicts between the DSP and MCU, these shared memory locations must be written by the MCU before the PDCH mode is applied. Alternatively, it can be written by the MCU just after reading the last channel/header result within a TDMA frame.*

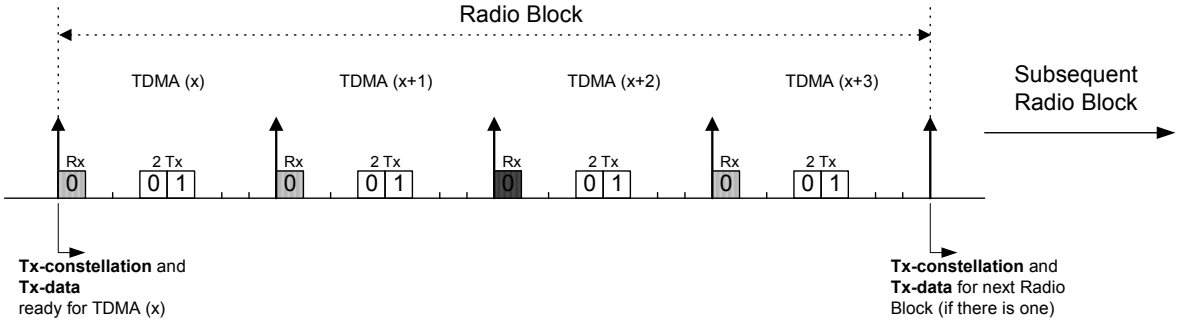
Transmitting Radio Blocks

This section describes the procedure for transmitting radio blocks and illustrates when the DSP expects data from the MCU.

Consider four TDMA frames corresponding to the length of a radio block in **Figure 4-16**, Unlike **Figure 4-13** two additional Tx timeslots are allocated. With this channel constellation (1 Rx and 2 Tx) the GPRS mobile is able to receive one radio block and to transmit two radio blocks at the same time. The number of Tx timeslots are restricted to a class 12 mobile which handles a four maximum Tx. The timing for each radio block on a 52-multiframe is the same.

If the MCU wants to receive radio blocks or if the MCU has to be careful about the USF detection for transmitting radio blocks, the procedure described in **Section 4.12.2.1 “Receiving Radio Blocks” on Page 88** is used.

Figure 4-16 Timing for Transmitting Radio Blocks



Over a period of four TDMA frames, there are different points where the DSP expects from the MCU:

- **Encoding** data for the next radio block
- Control information about **Tx Constellation**.

Encoding

The data that the MCU wants to transmit in the subsequent radio block must be ready in the shared memory before the first TDMA frame of the radio block. That is, the DSP always looks at the shared memory starting with the first Frame Interrupt of a radio block. After that it starts the encoding procedure if it is requested. That means the data must be written to the shared memory before the Frame Interrupt TDMA(x). In **Table 4-26** are times the data have to be ready in the shared memory.

Table 4-26 Encoding Data

Radio Blocks Are Transmitted in Frames:	Encoding Data and SM_TX_INFO Must Be Available before Frame Interrupt
0,1,2,3	0
4,5,6,7	4
8,9,10,11	8
13,14,15,16	13
17,18,19,20	17
21,22,23,24	21
26,27,28,29	26
30,31,32,33	30
34,35,36,37	34
39,40,41,42	39
43,44,45,46	43
47,48,49,50	47

The shared memory provides one location for each Tx timeslot, each consisting of 28 words. The locations SM_PDTCH_TX_DATA_0/1/2/3 contain all the information that the DSP needs for the encoding procedure. The first word (refer to Table 4-27) of SM_PDTCH_TX_DATA_0/1/2/3 signals the DSP the channel encoder type, the following words are reserved for the data itself. The data contents for GPRS can be seen in Table 4-28. The highest data throughput in GPRS is supported by the coding scheme CS4. The encoder type values '4' and '5' are necessary to encode a PACCH which consists of four PRACHs. For more information refer to Section 4.12.4 "PACCH" on Page 99.

Table 4-27 Contents of First Word of SM_PDTCH_TX_DATA_0/1/2/3

Word	Description	Contents	
0	Encoder type	00_H	Use coding scheme CS1
		01_H	Use coding scheme CS2
		02_H	Use coding scheme CS3
		03_H	Use coding scheme CS4
		04_H	Encoding with 8 bit PRACH (PACCH)
		05_H	Encoding with 11 bit extended PRACH (PACCH)

Table 4-28 Contents of SM_PDTCH_TX_DATA_0/1/2/3 for GPRS

Word	Description	Contents
1	data	bit[15:0]
2	data	bit[31:16]
...		
12	data	bit[183:176] for CS1
...		
17	data	bit[270:256] for CS2
...		
20	data	bit[314:304] for CS3

Table 4-28 Contents of SM_PDTCH_TX_DATA_0/1/2/3 for GPRS

Word	Description	Contents
...		
27	data	bit[430:416] for CS4

The shared memory locations do not correspond to a predefined timeslot. The location SM_PDTCH_TX_DATA_0/1/2/3 can be filled even if the MCU does not know if it is allowed to send any timeslots in the next four TDMA frames. After getting the USF detection, the MCU decides which encoded data it wants to send on a determined timeslot and writes that information to SM_TX_INFO.

After reading the data from the shared memory locations the MCU may not be able to transmit the encoded data within the next four TDMA frames because the detected USF does not belong to its mobile. In this case, the MCU may force the encoding of the data by setting the first word of the shared memory locations SM_PDTCH_TX_DATA_0/1 to the appropriate value. It is not necessary to copy the data words into the shared memory again.

Tx Constellation

The Tx constellation shared memory location SM_TX_INFO is provided by the MCU to the DSP to indicate the bursts that the DSP has to copy to the modulator RAM.

The DSP fetches SM_TX_INFO starting with the first frame interrupt at the beginning of a radio block. This is done for every radio block. As the DSP does not overwrite SM_TX_INFO the MCU only has to sets this field if there are any changes.

The shared memory field SM_TX_INFO contains four words. One word is reserved exactly for one Tx timeslot and includes the information which burst the DSP has to copy to the modulator RAM as shown in **Table 4-29**.

Table 4-29 Values of SM_TX_INFO

Possible Values	Transmit Encoded Data Of
-1	Nothing to do within this time slot
0	SM_PDTCH_TX_DATA_0
1	SM_PDTCH_TX_DATA_1
2	SM_PDTCH_TX_DATA_2
3	SM_PDTCH_TX_DATA_3

Note: The first word of SM_TX_INFO belongs to the first activated uplink timeslot, the second word of SM_TX_INFO belongs to the second activated uplink timeslot, and so on.

According to the GSM 45.002 standard, the number of Tx timeslots shall be allocated in a window of size four (class 12 mobile). The DSP subsystem supports two different applications with the Tx constellation field SM_TX_INFO:

1. **CODON Continuous**
2. **CODON with Gaps**.

CODON Continuous

The CODON signal from the system interface is set over the whole window size even if the MCU does not want to transmit bursts. That means the modulator is active in the whole window. The MCU has to make sure that the RF signals are switched off.

The rising edge of CODON has to occur before starting the burst transmission (refer to **Section 3.3.2 “MODU_INIT” on Page 34**). Consider two allocated Tx timeslots for CODON Continuous within the whole window in **Figure 4-17**. **Table 4-30** contains the appropriate usage of the Tx constellation field SM_TX_INFO.

Figure 4-17 CODON signal continuous

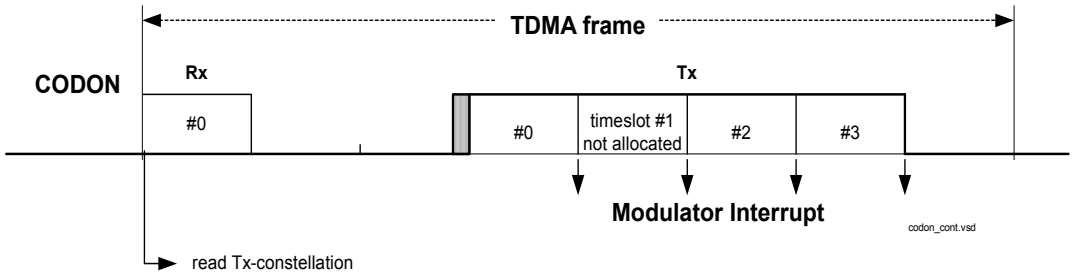


Table 4-30 Usage of SM_TX_INFO for CODON Continuous

Word	Related Timeslot	Contents	Description
0	#0	1	Transmit encoded data of SM_PDTCH_TX_DATA_1
1	#1	-1	Nothing to do within this timeslot
2	#2	2	Transmit encoded data of SM_PDTCH_TX_DATA_2
3	#3	3	Transmit encoded data of SM_PDTCH_TX_DATA_3

CODON with Gaps

The CODON signal from the system interface is set when the MCU wants to transmit bursts. The algorithm in the DSP is the same as in **CODON Continuous**, but it gets an additional CODON high interrupt and it misses a modulator interrupt. For that reason the Tx constellation field has a different meaning from that of the first application. Consider two allocated timeslots and a gap during timeslot #1 in **Figure 4-18**. The appropriate setting of **SM_TX_INFO** is described in **Table 4-31**.

Figure 4-18 CODON Signal with Gaps

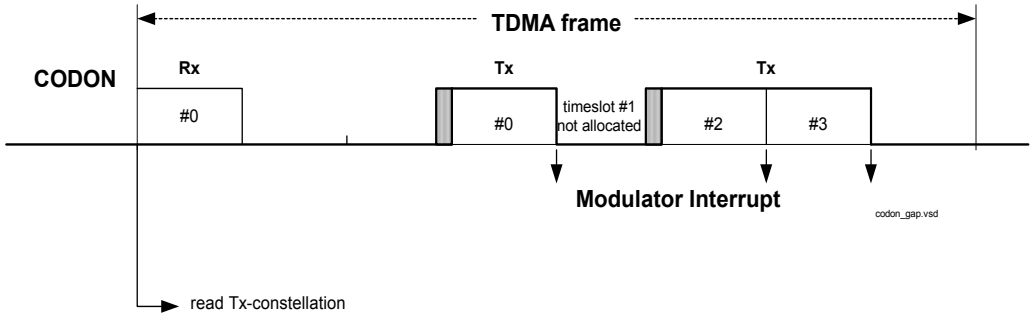


Table 4-31 Usage of SM_TX_INFO for CODON with Gaps

Word	Related Timeslot	Contents	Description
0	#0	0	Transmit encoded data of SM_PDTCH_TX_DATA_0
1	#2	1	Transmit encoded data of SM_PDTCH_TX_DATA_1

Table 4-31 Usage of SM_TX_INFO for CODON with Gaps

Word	Related Timeslot	Contents	Description
2	#3	2	Transmit encoded data of SM_PDTCH_TX_DATA_2
3	x	-1	Nothing to do within this timeslot

These two applications can be mixed by the MCU. For instance, if the MCU wants to interrupt the transmission in timeslot #3 in **Figure 4-18**, it can leave the CODON signal as it is. But it has to change the word related to timeslot #3 to '-1' (nothing to do in this timeslot).

Fixed Allocation, Dynamic Allocation, and Extended Dynamic Allocation

The DSP subsystem interface is used for each allocation. From the MAC layer point of view, the main difference between the allocation types is that the DSP subsystem may watch for the USF. For instance, the MCU may not evaluate the USF detection if the GPRS mobile transmits in a fixed allocation mode. The DSP subsystem is not allowed to decide if the GPRS mobile may send or not. It just gives the results (USF detection, radio blocks) back to the MCU. With this interface the DSP does not need to know the allocation type.

Uplink and Downlink Packet Transfer Simultaneously

From the DSP subsystem point of view there is no difference between the GPRS mobile receiving data in the packet downlink transfer or control information in the packet uplink transfer. It do not know what kind of data are inside the radio blocks.

4.12.3 PRACH

In GPRS a new PRACH, the extended PRACH with 11 information bits, has been specified. The PRACH in GPRS can be used in two different cases and, therefore, has two different meanings:

- As an access burst which is known from the circuit switched mode.
  By using the PRACH as an access burst the MCU controls the PRACH encoding by writing proper values to the shared memory locations SM_RACH_FLAG/SM_RACH_TSC/SM_RACH_TIM_ADV/ SM_RACH_DATUM/SM_RACH_BSIC before the *Frame Interrupt* occurs (refer to **Section 4.10 "Access Burst (RACH)" on Page 72**).
  If the DSP subsystem is in the PDCH mode, it is mandatory that the SM_TX_INFO field is set to '-1' for the current radio block. Additionally, sending a PRACH by the MCU is only possible in a single Tx timeslot. This means only the first timeslot can contain the PRACH. The DSP does not copy any bursts to the subsequent timeslots if allocated.

Note: The MCU has always to repeat this process if it wants to transmit a PRACH in the subsequent TDMA frame.

- An uplink PACCH that consists of 4 PRACHs.
  Sending a PACCH with 4 PRACHs in a radio block is completely different. The MCU does this by writing appropriate values to the shared memory locations SM_PDTCH_TX_DATA_0/1/2/3 (refer to **Section 4.12.4 PACCH** below).

4.12.4 PACCH

The DSP subsystem cannot see any difference between transmitting or receiving an RLC data block instead of an RLC/ MAC control block (PACCH). It only sees the coding scheme of each radio block. Control channels like PACCH are always coded with CS1. As the CS1 coding scheme is also used for RLC data blocks and, therefore, the DSP subsystem is not able to distinguish between an RLC data block and an RLC/ MAC control block.

Downlink

After getting an interrupt from the baseband filter the DSP begins to check the stealing flags. If the DSP detects the CS1 coding scheme and if the appropriate word of the shared memory location **SM_RX_INFO** contains the value ’0’, the DSP immediately starts the channel decoder for CS1. The results are written to the shared memory location as described in **Section 4.12.2.1 “Receiving Radio Blocks” on Page 88**.

Uplink

Starting with the Frame Interrupt in the first TDMA frame of a radio block, the DSP checks what it has to encode for this current radio block. It is the same process as explained in **Section “Transmitting Radio Blocks” on Page 95**.

The MCU may transmit a PACCH which consists of four PRACHs. The MCU signals this by writing the appropriate values to the shared memory location of **SM_PDTCH_TX_DATA_0**/1/2/3. But in this case, the shared memory locations have another usage (refer to **Table 4-32**).

Table 4-32 PACCH Consisting of four PRACHs

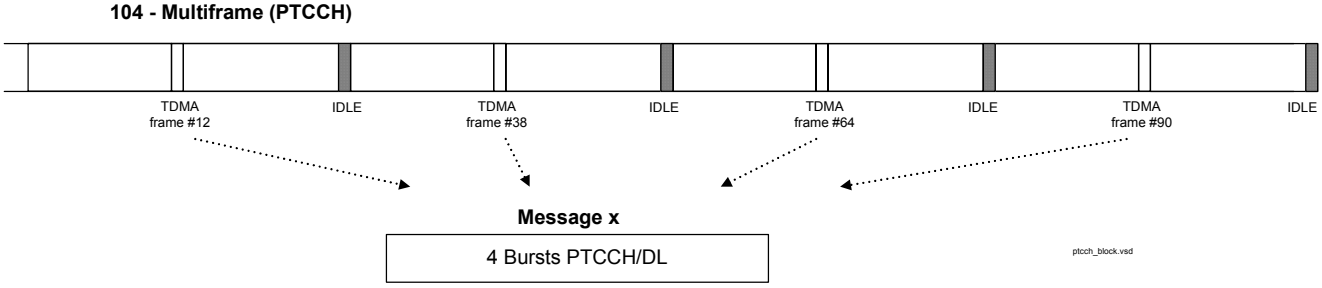
SM_PDTCH_TX_DATA_0/1	Contents	Description
Word 0	Same meaning as in **Table 4-27**	
Word 1	0	Training sequence
	1	Alternative training sequence TS1
	2	Alternative training sequence TS2
Word 2	0...63	Timing advance value
Word 3	Bit[7:0] = BS[7:0]	Information bits for RACH
	Bit[11:0] = BS[11:0]	Information bits for extended RACH
Word 4	Bit[2:0] = BS[2:0]	Base Station color code
	Bit[5:3] = PLMN[2:0]	PLMN color code
Word 5-27	not used	

4.12.5 PTCCH

In GPRS the Packet Timing Advanced Control Channel (PTCCH) is mapped onto a multiframe consisting of 416 TDMA frames. The TDMA frames (12, 38, 64, 90,...,402) are exclusively allocated for the PTCCH downlink messages according to the GSM 45002 standard.

The PDCH mode supports a 104-multiframe (416/4) for receiving exactly one PTCCH/DL message, see **Figure 4-19**. To get four PTCCH messages, the MCU initiates the receiving procedure once every 104-multiframe. Since the DSP subsystem periodically works with a 104 TDMA frame, all consecutive PTCCH messages (in a multiframe of 416 TDMA frames) are received automatically.

Figure 4-19 PTCCH



For locating the PTCCH frames within the 104-multiframe the PDCH mode uses the shared memory counter 104. In contrast to the PDTCH radio block, it is not necessary to give any information to the DSP. The MCU controls the Timing Advanced Control Channel by the signals EQON and CODON as described below.

Note: The number of supported Packet Timing Advanced Control Channels for uplink and downlink packet transfer is restricted to one by the DSP interface.

PTCCH Downlink

The MCU can initiate the receiving procedure by giving the EQON signal in every PTCCH TDMA frame. The DSP always starts the Equalizer if it gets a baseband buffer full interrupt within the PTCCH related timeslots. The PTCCH downlink message is always encoded with the coding scheme CS1. Channel decoding is done after the last burst of a PTCCH message has been received. This means the DSP automatically starts the channel decoder with coding scheme CS1 if it gets a baseband buffer full interrupt in TDMA frame 90. The result is written to **SM_PTCCH_RX_DATA** and is available at the end of TDMA frame 90.

The structure of the shared memory block **SM_PTCCH_RX_DATA** is identical to the SACCH Rx result as described in **Section 4.11.1 “Sacch” on Page 74**. This 14-word shared memory location block consists of the following data in order:

1. PTCCH_RX_METRIC Channel decoder metric (1 word, max. value: 456)
2. PTCCH_RX_STATUS Channel decoder status (1 word), the possible values are:
   0. No error in CRC, decoded data is valid
   1. Error in CRC, but decoded data is valid (error has been corrected)
   2. Error in CRC, decoded data is not valid (error could not be corrected)
3. PTCCH_RX_DATA 184 bit output data stored in 12 16-bit words.
   a) Word 0 = D(15:0)
   b) Word 1 = D(31:16)
   – ...
   l) Word 11 = D(183:176).

PTCCH Uplink

The PTCCH uplink is based on sending a PRACH in the TDMA frames 12, 38, 64, 90. But not more than one PRACH can be sent in these TDMA frames.

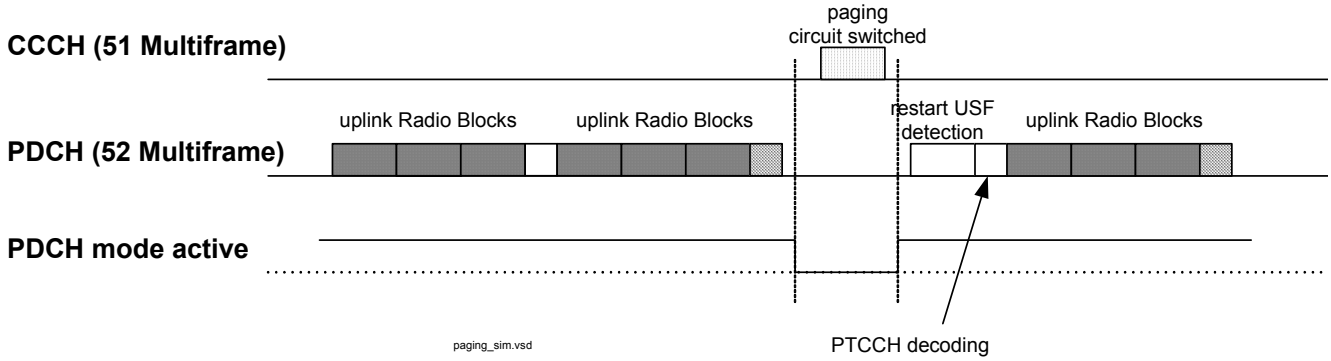
Sending a PRACH is provided by appropriately setting the shared memory locations **SM_RACH_FLAG**/ **SM_RACH_TSC**/**SM_RACH_TIM_ADV**/**SM_RACH_DATUM**/**SM_RACH_BSIC** in the previous TDMA frame. The encoding of the PRACH data is always done in the Frame Interrupt. It is the same process as described in **Section 4.12.3 “PRACH” on Page 99**.

4.12.6 Paging of Circuit Switched Services during PDCH Mode

A GPRS mobile in the class B mode of operation can page circuit switched services while it is in PDCH mode. Paging of these services is done on the CCCH, which consists of a 51-multiframe structure. It does not fit into the PDCH mode because it is based on a 52-multiframe.

The DSP subsystem provides paging of circuit switched services by releasing the PDCH mode for a certain time. Consider the example in **Figure 4-20**.

Figure 4-20 Paging of Circuit Switched Services



To perform the paging procedure completely, the following steps have to be done by the MCU:

1. After the end of the last TDMA frame of a radio block or after the end of the timing advance TDMA frame, the MCU can initiate the paging procedure by sending the command **IDLE**, or **BB_OFF** to the DSP.
2. While the DSP subsystem is in idle mode, the MCU can camp on the CCCH by using the **CCH_RX** and **CCH_TX** commands that were introduced in **Section 3.3.7 “CCH_RX” on Page 42** and **Section 3.3.8 “CCH_TX” on Page 42**
3. After having received the paging channel the MCU forces the DSP subsystem back to the PDCH mode by sending the command **PDCH**.
4. Starting with the next radio block after the **PDCH** command has been sent, the GPRS mobile can receive data and the USF. Thus transmitting of data is not possible before the subsequent radio block except if the mobile is in the fixed allocation mode.

Note: In the packet transfer mode the mobile may under some circumstances lose two radio blocks due to a paging procedure of circuit switched services. When the MCU forces the DSP subsystem back to PDCH mode it might be that the current PTCCH downlink message can not be received correctly. This can be recognized by checking the CRC value.

5 Voiceband Processing Functions

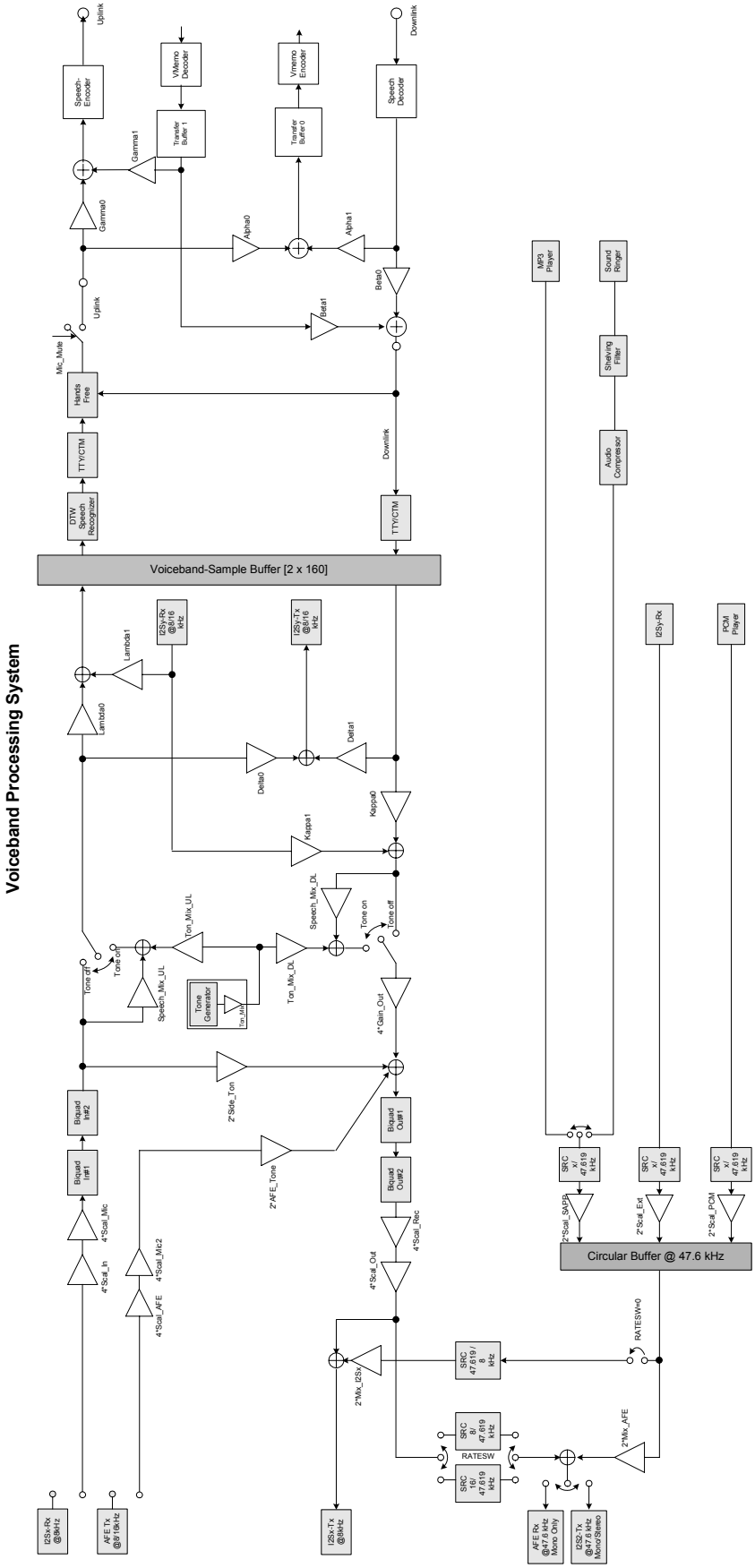
E-GOLDradio CONFIDENTIAL		
Revision History:	**2005-12-07**	Rev. 1.01
Previous Version:	Rev. 1.00, 2005-05-16	
Page	Subjects (major changes since last revision)	
	Initial Version based on *E-GOLDradio G14 Firmware Manual*	
Changes for Rev. 1.01		
Page 114	Add noise reduction comments.	
Page 104	Update **Figure 5-1** with external I^2S_2 stereo output.	
Page 105	Add description of external I^2S_2 stereo output.	
Page 108	Update **Figure 5-2** with external I^2S_2 stereo output.	
Page 135	Update **Figure 5-10** with external I^2S_2 stereo output.	
Page 136	Update **Figure 5-11** with external I^2S_2 stereo output.	
Page 137	Update **Figure 5-12** with external I^2S_2 stereo output.	
Page 128	Modify the byte order for the MP3 data stream input.	

In this chapter the behavior of the Audio scheduler and related applications are described. The voiceband processing system is split into:

- Sample-based Voiceband Processing (125µs/62,5µs) (refer to **Section 5.2 “Sample-Based Voiceband Processing” on Page 106**)
- Frame-based Voiceband Processing (20ms) (refer to **Section 5.3 “Frame-Based Voiceband Processing” on Page 112**)
- Circular Mixing Buffer, to mix different sound signals (MP3 Player, Sound Ringer, I^2S_y-RX) with the downlink signal (refer to **Section 5.4 “Circular Mixing Buffer” on Page 126**).

Figure 5-1 is an overview of the voiceband processing system with these three subsystems. The frame-based voiceband processing is called by the Operating System (OS) when 160 voiceband input samples are ready (microphone path), that is 160 voiceband samples have already been processed by sample-based voiceband processing. The sample-based voiceband processing is done on a hardware interrupt level and its main task is to transfer the voiceband samples from either AFE-Tx or I^2S_x-RX to the Voiceband-Sample-Buffer (microphone path) and from the Voiceband-Sample-Buffer to the AFE-Rx and/or I^2S_x-TX (loudspeaker path).

Figure 5-1 Voiceband Processing System



5.1 Hardware Interfaces

Three HW interfaces are supported:

- **Audio Front-End Interface (AFE) and External Audio Output (by I2S2)** controlled by the **VB_ON** command.
- **I2Sx Interface** controlled by the **VB_ON** command.
- **I2Sy Interface** controlled by the **VB_I2Sy** command.

The processing of input samples is always triggered by the AFE-Tx interrupt.

The mapping of I^2S_1 and I^2S_2 hardware peripherals on I^2S_x or I^2S_y interfaces is controlled by the **I2S_SWAP** command. The external audio output by I^2S_2 HW interface is not available if I^2S_x is switched on.

For a detailed description of these interfaces and how they have to be configured refer to **[1]**.

5.1.1 Audio Front-End Interface (AFE) and External Audio Output (by I^2S_2)

The AFE and external audio output by I^2S_2 interfaces are switched on/off and configured by the **VB_ON** command. This command must be sent before any function of the voiceband Processing can be started. The AFE-Tx (Microphone) input always runs in the mono mode.

The input sampling rate can be switched between 8 and 16 kHz by the parameter RATESW in the **VB_ON** command. The AFE-Rx (Loudspeaker) output and the external audio output interface always run at a 47.619 kHz sampling rate. The output mode can be switched between mono and stereo mode using the parameter OUT_MODE in the **VB_ON**.

The output of the circular mixing buffer is always stereo; the output of the sample-based processing is always mono. Depending on the OUT_MODE (mono or stereo), the mixing of these signals can be done in two different ways:

Mono

1. AFE Mono Output: the left and right channel samples from the circular mixing buffers (CBleft and CBright) are mixed together to build a mono signal that is then mixed with the sample-based mono output (SBout):

$$\mathrm{AFEmonoout} = \frac{1}{2}\left(\mathrm{SBout} + \frac{1}{2}(\mathrm{CBleft} + \mathrm{CBright})\right)$$

2. I^2S_2 Mono Output: the left and right channel samples from the circular mixing buffers (CBleft and CBright) are mixed together to build a mono signal that is then mixed with the sample-based mono output (SBout):

$$\mathrm{I2S2monoout} = \frac{1}{2}\left(\mathrm{SBout} + \frac{1}{2}(\mathrm{CBleft} + \mathrm{CBright})\right)$$

Stereo

1. AFE Stereo Output: the sample-based mono output (SBout) is separately mixed with the left and right channel samples out of the circular mixing buffer (CBleft and CBright) to provide the two AFE output channels (AFEleft and AFEright):

$$\mathrm{AFEleft} = \frac{1}{2}(\mathrm{SBout} + \mathrm{CBleft})$$

$$\mathrm{AFEright} = \frac{1}{2}(\mathrm{SBout} + \mathrm{CBright})$$

2. I^2S_2 Stereo Output: the sample-based mono output (SBout) is separately mixed with the left and right channel samples out of the circular mixing buffer (CBleft and CBright) to provide the two I^2S_2 output channels (I2S2left and I2S2right)

$$I2S2left = \frac{1}{2}(SBout + CBleft)$$

$$I2S2right = \frac{1}{2}(SBout + CBright)$$

There are seven different operational modes:

1. AFE input and output on. I^2S_x input and output are off.
2. AFE input and output off. I^2S_x input and output are on.
3. AFE input off and output on. I^2S_x input and output are on.
4. AFE input on and output off. I^2S_x input and output are on.
5. AFE input and output on. I^2S_x input and output are on.
6. AFE input on and output off. External audio output by I^2S_2 on.
7. AFE input and output on. External audio output by I^2S_2 on.

In modes **4** and **5**, the AFE input is only mixed for the I^2S_x and AFE outputs, but not for the uplink path to the network.

5.1.2 I^2S_x Interface

The I^2S_x interface can be used as input and/or output for the voiceband processing system. This interface is used to connect the E-GOLDradio to a Bluetooth device. It is switched on/off and configured by the **VB_ON** command. The I^2S_x interface always runs in the burst mode in the mono mode at 8 kHz.

It can be run in the master or slave mode. In the slave mode the E-GOLDradio has to provide the clock source.

5.1.3 I^2S_y Interface

The I^2S_y interface allows connecting external input/output devices to the voiceband processing system and can be used at two different locations on the system (see **Figure 5-1**):

- At the end of the sample processing part (as input and output), that is, for I^2S MMS or DAI (refer to **Section 5.5 “DAI Functions” on Page 134**). In this mode the interface has to run at the same sampling rate than the AFE input (8 or 16 kHz). Therefore, the selection of the sampling rate is done in the DSP and not via the I2Sy_RATE parameter in the **VB_I2Sy** command. The mode can be switched by the parameter I2Sy_MODE in the **VB_I2Sy** command between mono (every sample used) and dual mono (every second sample used).
- As an input for the circular mixing buffer (refer to **Section 5.4 “Circular Mixing Buffer” on Page 126**). In this mode the interface can run at different sampling rates (8, 16, 22.1, 24, 32, 44.1 or 48 kHz). The selection of the sampling rate is done by the MCU via the I2Sy_RATE parameter in the **VB_I2Sy** command. The mode can be switched by the parameter I2Sy_MODE in the **VB_I2Sy** command between mono (every sample used), dual mono (every second sample used), or stereo.

The I^2S_y is switched on/off and configured by the **VB_I2Sy** command and always runs in the normal mode.

It can be run in master or slave mode. In slave mode the E-GOLDradio has to provide the clock source.

5.2 Sample-Based Voiceband Processing

This section is a description of each voiceband processing component:

- In the microphone path the treatment of the samples (scaling and double filtering)
- In the loudspeaker path with the generation of the side tone as well as the tone generator, double filtering as well as scaling.

The voiceband parameters of the different blocks [filter parameters and scaling factors] indicated in **Figure 5-2** is set using the commands **VB_SET_BIQUAD** and **VB_SET_GAIN**.

5.2.1 Overview

Figure 5-2 shows the sample-based part for processing samples in Tx and Rx direction.

The Tx path (uplink) is processed first: the input sample is scaled, filtered, and it is used to generate the sidetone for the Rx path processing and the input for the I^2S_y interface before it is copied into the Voiceband input buffer for the speech frame based processing.

Then the Rx path (downlink) is processed: the sample from voiceband output buffer is mixed with the output of the I^2S_y interface and the side tone. The sample is then added to the sidetone, filtered, scaled, and passed through the AFE or I^2S_x.

Uplink

The following the steps of the sample based voiceband processing in Tx direction are briefly described:

1. The output of the AFE or I^2S_x is scaled with two consecutive gains *4*Scal_In* and *4*Scal_Mic* (refer to **VB_SET_GAIN**).
2. The signal is filtered with two consecutive biquad filters (refer to **Section 5.2.2 “Biquad Filters” on Page 109**).
3. The output of the filters is used to generate the side tone for the Rx path (see **Figure 5-2 “Sample Based Voiceband Processing” on Page 108**).
4. If a tone is generated, then the signal is scaled by *Speech_Mix_UL* and mixed with the tone signal, which is scaled by *Ton_Mix* and by *Ton_Mix_UL*.
5. The signal is scaled with the factor:
   a) *lambda0* and mixed with the by *lambda1* scaled I^2S_y-Rx signal to provide the input to the voiceband-sample buffer.
   b) *delta0* and mixed with the by *delta1* scaled downlink signal to build the I^2S_y-Tx signal.
6. If the AFE input in parallel to I^2S_x input is selected in the **VB_ON** command, the input of the AFE is scaled with two consecutive gains 4**Scal_AFE* and 4**Scal_Mic2* (see **VB_SET_GAIN**) and is stored for adding in the downlink path.

Downlink

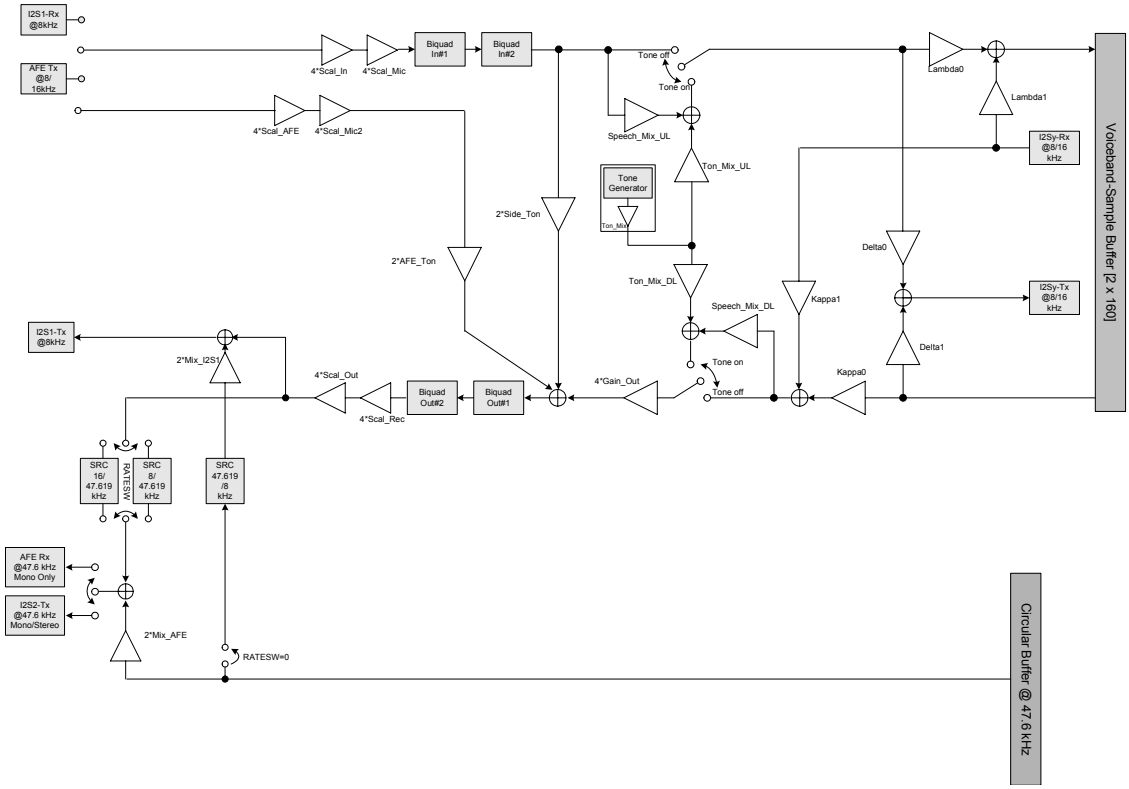
The following the steps of the sample based voiceband processing in Rx direction are briefly described.

The working buffer for this task is the voiceband-sample buffer (160 sample length).

1. See Tx-direction step **5b**.
2. The signal is scaled by *kappa0* and mixed with the by *kappa1* scaled I^2S_y-Rx signal.
   – If a tone is generated, then the signal is scaled by *Speech_Mix_DL* and mixed with the tone signal, which is scaled by *Ton_Mix* and by *Ton_Mix_DL*.
3. The signal is scaled by Gain_Out and mixed with the by *2*side_Ton* scaled side tone (refer to the Tx direction step **3**).
4. If the AFE input in parallel to I^2S_x input is selected in the **VB_ON** command, the signal is mixed to the 2**AFE_Ton* scaled AFE input. (refer to the Tx direction step **6**)
5. The signal is filtered with two consecutive biquad filters (refer to **Section 5.2.2**).
6. The signal is scaled with two consecutive gains *4*Scal_Rec* and *4*Scal_Out* (refer to **VB_SET_GAIN**).
7. If the AFE input rate is 8 kHz, the downlink sample is mixed with the output of the **Circular Mixing Buffer** (after the buffer is decimated from 47.619 to 8 kHz and then scaled by 2**Mix_I2Sx*) to deliver the I^2S_x-Tx output. Else the downlink sample is simply output as I^2S_x-Tx sample.
8. To get the AFE Rx output, the downlink sample is interpolated (refer to **Section 5.2.4 “Sample-Based Sample Rate Converter” on Page 112**) from 8/16 to 47.619 kHz and then mixed with the by 2**Mix_AFE* scaled circular buffer output.

The gains (*Scal_In*, *Scal_Out, Side_Ton, Mic_Mute, Scal_Mic, Gain_Out, Scal_Rec, Ton_Mix, Ton_Mix_DL, Speech_Mix_DL, Ton_Mix_UL, Speech_Mix_UL, delta0*, *delta1, lambda0, lambda1, kappa0, kappa1*) for this part are set by the command **VB_SET_GAIN**.

Figure 5-2 Sample Based Voiceband Processing



5.2.2 Biquad Filters

The structure of single Biquad-filter used for this processing is shown in **Figure 5-3**, where it can be seen that in both directions two single Biquad-filters in a row are used. The coefficients of these filters are set with the command **VB_SET_BIQUAD**. Five coefficients for each filter are used. Parameters:

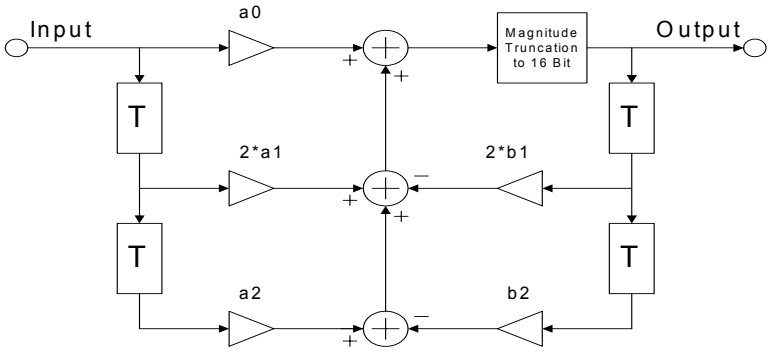
- 1-5 are reserved for Biquad_In_0
- 6-10 are reserved for Biquad_In_1
- 11-15 fare reserved or Biquad_Out_1
- 16-20 are reserved for Biquad_Out_2.

The correspondence of the command parameters and the coefficient names on the Biquad structure is in **Table 5-1**:

Table 5-1 VB_SET_BIQUAD Parameters

Parameters	Biquad Coefficient	Default Value
1, 6, 11, 16	a1	0
2, 7, 12, 17	b1	0
3, 8, 13, 18	a2	0
4, 9, 14, 19	b2	0
5, 10, 15, 20	a0	0x7FFF

Figure 5-3 Single Biquad Filter



5.2.3 Tone Generator

The tone generator is built up by three sine generators. The three sine waves are summed up and scaled with a tone-mix factor Tone_Mix. The output is used to mix a tone into uplink and/or downlink path. Therefore it is scaled by a second Tone-Mix gain (*Tone_Mix_UL* for uplink and *Tone_Mix_DL* for downlink) and added to the speech signal (if present), that previously has been scaled with the speech-mix factor (*Speech_Mix_UL* for uplink and *Speech_Mix_DL* for downlink) (see **Figure 5-4**). The five scaling factors *Tone_Mix*, *Tone_Mix_UL*, *Tone_Mix_DL*, *Speech_Mix_UL* and *Speech_Mix_DL* are set via the command **VB_SET_GAIN**. The frequency coefficients Freq1/2/3 directly correspond to the shared memory locations **SM_TONE_FREQ_1**/2/3 (refer to **Chapter 7 “Shared Memory” on Page 149**).

When a new tone should be started with the command **VB_START_TONE**, the tone generator parameters (frequencies and amplitudes) are copied from the shared memory to an internal coefficient buffer and a state buffer. The DSP sets then the internal duration counter according to the shared memory location **SM_TONE_DUR_IN**.

The Sine generators have the following output relation:

$$y_i(n) = 2Freq_i y_i(n-1) - y_i(n-2) \quad (5.1)$$

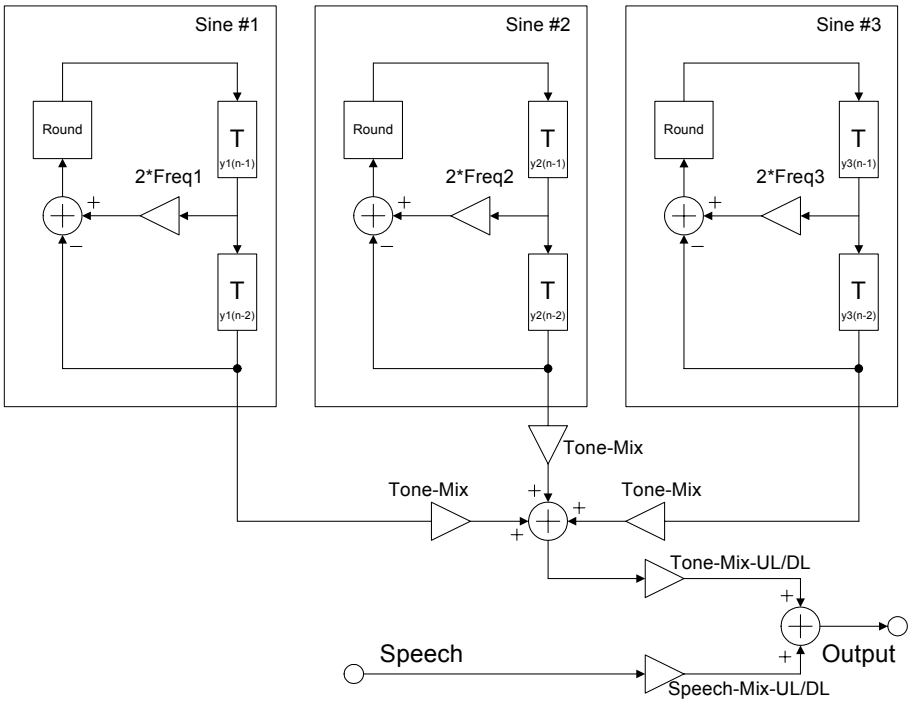
where i = 1, 2, or 3.

At every sample interrupt (every 125µs), If the duration counter is greater than 0, the output of the tone generator (including the mixing of speech and tone) is built as:

$$y_{ULtone}(n) = \left(ToneMixUL \cdot \sum_{i=1}^{3} y_i(n) \right) + (Speech \cdot SpeechMixUL) \quad (5.2)$$

$$y_{DLtone}(n) = \left(ToneMixDL \cdot \sum_{i=1}^{3} y_i(n) \right) + (Speech \cdot SpeechMixDL) \quad (5.3)$$

Figure 5-4 Tone Generator



When the internal counter reaches the value ’1’, the DSP will check, if a new tone has been prepared by copying the tone generator parameters again out of the shared memory. The value of **SM_TONE_DUR_IN** is set to zero. If the internal counter is not ’-1’, it is decremented. When the counter reaches the value ’0’ no tone generation is done anymore. When a new tone is started, the delay line elements y1/2/3(n-2) in **Figure 5-4** get an initialization value of 0,i.e. the first sample of a new tone will always have a value of 0. The delay line elements y1/2/3(n-1) are initialized with the contents of the shared memory locations **SM_TONE_AMP_1**/2/3 (refer to **Chapter 7**). In this way, the amplitude of the tones can be set. If the tone generator is not active, the speech signal is not scaled down by the speech-mix factor.

If the MCU wants to switch on a certain frequency, Freq*i*, the corresponding values **SM_TONE_FREQ_1**/2/3 can be calculated as follows:

$$SM\_TONE\_FREQ\_i = 0x7FFF \bullet \cos\left(\frac{2 \cdot \Pi \cdot Freq_i}{F_S}\right) \quad (5.4)$$

where i = 1, 2, or 3 and Fs is the sampling rate (8 or 16 kHz). The maximum tone frequency that can be generated is 4kHz.

The MCU can set the amplitude Amp*i* by writing to the shared memory location **SM_TONE_AMP_1**/2/3 a value calculated with the equation:

$$SM\_TONE\_AMP\_i = Amp_i \bullet \sin\left(\frac{2 \cdot \Pi \cdot Freq_i}{F_S}\right) \quad (5.5)$$

where i = 1, 2, or 3 and Fs is the sampling rate (8 or 16 kHz). The range for the value Amp*i* is 0...0x7FFF.

To avoid saturation of the tone generator samples (and a distorted and non-linear rise of the output amplitude), the sum of the amplitudes of the three tones should not exceed $7FFF_H$:

$$Amp_1 + Amp_2 + Amp_3 \leq 7FFF_H \quad (5.6)$$

Example:

The following example gives an overview how the tone generator can be used for generating tones. In **Table 5-2** values are given for typical DTMF frequencies.

Table 5-2 Generating DTMF Tones

Frequency	SM_TONE_FREQ_1/2/3
697	27979
770	26956
852	25701
941	24219

Tone Duration Control

An internal duration counter is used for tone duration control. If this tone duration control has a value of *0*, the tone generation unit is not activated.

Using the **VB_START_TONE** command the MCU can set a new value for the duration counter:

1. The shared memory location **SM_TONE_DUR_IN** is copied to the internal duration counter and then **SM_TONE_DUR_IN** is reset to ‘0’.
2. The amplitude and frequency values for the new tone are read from the shared memory locations **SM_TONE_AMP_1**/2/3 and **SM_TONE_FREQ_1**/2/3 (refer to **Chapter 7 “Shared Memory” on Page 149**).
3. When the duration counter is not equal to zero, the tone generation unit is activated in the next voiceband interrupt.
4. At the end of the tone generation (after one sample of the tone has been generated) the internal duration counter is decremented by one, unless the duration counter has a value of *zero*.

*Note: If the MCU sets **SM_TONE_DUR_IN** to -1, the counter is never decremented and the tone never stops.*

5. When the tone duration counter reaches a value of ‘0’, the shared memory tone parameters (**SM_TONE_AMP_1**/2/3, **SM_TONE_FREQ_1**/2/3, and **SM_TONE_DUR_IN**) are read once again.
   a) If the MCU has not changed the **SM_TONE_DUR_IN** value, this parameter is still ‘0’ and when this value is copied to the internal duration counter the tone stops.
   b) If the MCU has changed **SM_TONE_DUR_IN**, a new tone is started.

The **VB_READ_DURATION** command is used to read out the internal duration counter by copying it to **SM_TONE_DUR_OUT**.

The MCU can immediately stop the tone generation at any time by running the command **VB_STOP_TONE**, which resets the internal duration counter.

Interrupt Generation

If the DSP generates a DSP_INT3 interrupt to the MCU before the tone duration counter expires, then the MCU must write a non-zero value, N, into the shared memory location **SM_TONE_DUR_INTER**, which determines the

number of samples before the tone stops. This means that the interrupt is given to the MCU N-samples before the tone stops. Every time the MCU reads the tone duration parameter in the shared memory, **SM_TONE_DUR_INTER** is also read and cleared.

Fading At the Beginning and End

If the DSP should fade the beginning and/or the end of the generated tone, then the MCU has to write the fading duration (in samples) for the beginning to the shared memo location **SM_TONE_FADIN_DUR** and/or for the end to the shared memo location **SM_TONE_FADOUT_DUR**.

Since these two shared memory locations are checked by the DSP for every new tone, the MCU has always (when tone generator is activated) to take care about a meaningful content. If fading (beginning/end) is not required, then 0 has to be written to the corresponding shared memory locations.

5.2.4 Sample-Based Sample Rate Converter

Three different sample-based sample rate converters (SRC) are used in the sample based part of the voiceband processing system:

- Interpolator from 8 kHz to 47.619 kHz:
  For an input sampling rate of 8 kHz, a rate conversion to the AFE output rate of 47.619 kHz is required.The SRC delivers a variable number of output samples for each input sample. The output samples are written twice into the output buffer to get a stereo signal.
- Interpolator from 16 kHz to 47.619 kHz:
  For an input sampling rate of 16 kHz, a rate conversion to the AFE output rate of 47.619 kHz is required.The SRC delivers a variable number of output samples for each input sample. The output samples are written twice into the output buffer to get a stereo signal.
- Decimator from 47.619 kHz to 8 kHz:
  Before the output of the circular buffer, which is sampled at 47.619 kHz, can be mixed with the I^2S_x-Tx running at 8 kHz, a sample rate conversion is required. The SRC delivers one output sample for a variable number of input sample pairs. The mean value of each input sample pair is calculated to get a mono signal.

5.3 Frame-Based Voiceband Processing

This section describes each component of the Frame Based Voiceband processing:

- The microphone path with the handsfree, TTY/CTM block and Voice Memo
- The loudspeaker path with the TTY/CTM block and the Voice Memo.

5.3.1 Overview

Tx-Direction

The following the steps of the voiceband processing in Tx direction are shown in the **Figure 5-5** uplink path:

The working buffer for this task is filled by the sample-based task (160 sample length).

1. If **DTW** recognition is on, the uplink **DTW** algorithm is executed.
2. If TTY/CTM is on (refer to **TTY_CTM** command) then the uplink TTY/CTM algorithm is executed.
3. If handsfree is switched on (depending on **HF_ON**), the speech frame goes through the handsfree block, which consists of two independent operating tasks: the Echo Canceller and the Automatic Gain Control unit (AGC). For a more information refer to **Section 5.3.2 “Handsfree” on Page 114**.
4. After this data processing,
   a) If the parameter *Mic_Mute* in the command **VB_SET_GAIN** is set, the microphone mute is activated.
   b) If TCH26 mode and/or voice memo (Play Mode) is active, the input signal is again scaled by *gamma0* and mixed with the voice memo transfer buffer that was previously scaled by *gamma1*. The resulting signal is then sent to the speech encoder.

c) If TCH26 mode and/or voice memo (Record Mode) is active, the input signal is scaled by *alpha0* and mixed with the speech decoder output that was previously scaled by *alpha1*. The resulting signal is copied into the transfer buffer of the voice memo encoder.

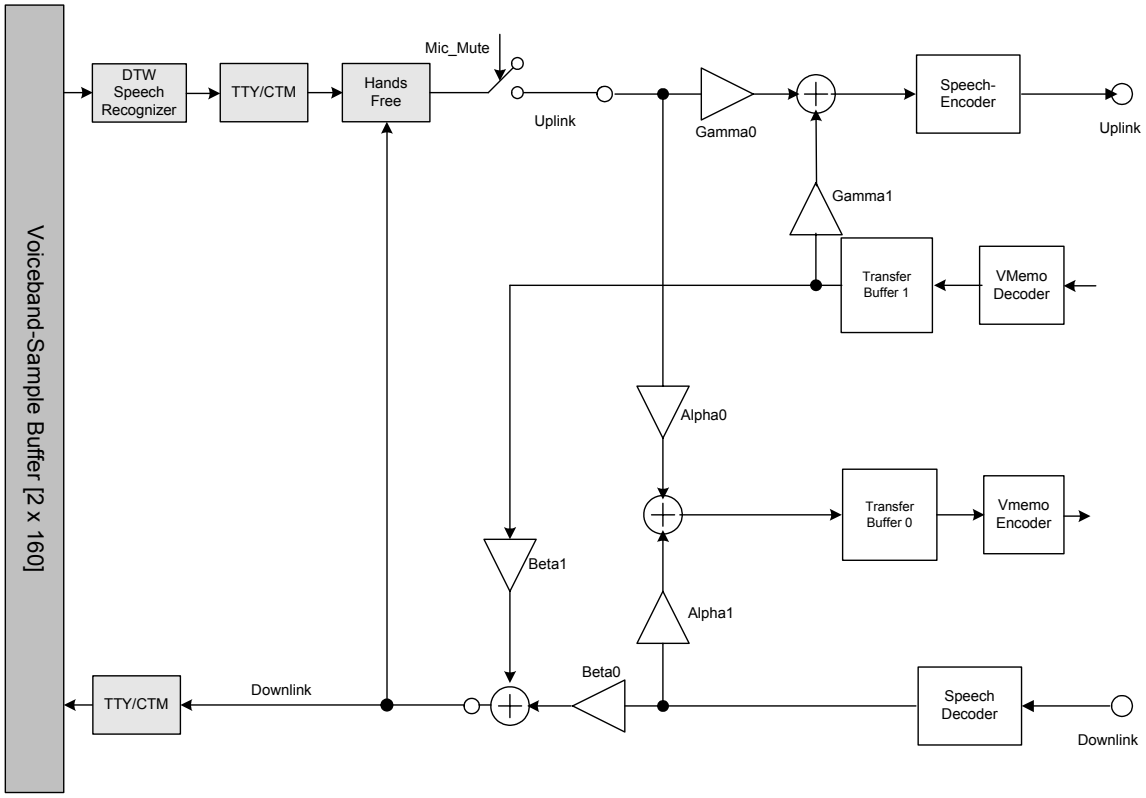
Rx-Direction

The following the steps of the voiceband processing in Rx direction are shown in the **Figure 5-5** downlink path:

1. Refer to Tx direction step **4c**.
2. If TCH26 mode and/or voice memo is active then the speech decoder output is scaled by *beta0* and mixed with the by *beta1* scaled voice memo decoder output.
3. If handsfree is switched on, the samples are used to provide a delay line for the handsfree algorithm.
4. If TTY/CTM is on (see **TTY_CTM**), the downlink TTY/CTM algorithm is executed.
5. The data is copied into the voiceband sample buffer as input for further sample based processing.

The gains (*alpha0*, *alpha1, beta0, beta1, gamma0, gamma1*) for this part are set by the command **VM_CMD**.

Figure 5-5 Frame-Based Voiceband Processing



5.3.2 Handsfree

Handsfree consists of two independent operating parts:

- Echo Canceller (EC), based on Block-NLMS (normalized least mean square) algorithm
- Automatic Gain Control (AGC).

Both parts only affect signals in voiceband TX path. Handsfree functional blocks EC and AGC are controlled by the **HF_ON** command. The **HF_ON** SWITCH bits control the Handsfree blocks as follows:

- Bit#0 (EC init bit)
  When set the EC does a full initialization. This mode is recommended if an initialization has never been done (that is, when the EC is switched on for the first time) or if it is very likely that the echo characteristic changed too much since the handsfree operation was stopped. As long as this bit is set, the Bit#1 (EC reset bit) is not interpreted.
- Bit#1 (EC reset bit)
  When set, initialization does not affect the former - by the block NLMS part - estimated echo impulse response. This helps to avoid a complete re-adaptation of the LMS coefficients after a short interruption of handsfree operation and if it is very likely that the echo characteristic did not change too much in the meantime. The EC reset bit is not interpreted as long as the EC init bit is set.
- Bit#2 (EC on bit)
  Setting this bit switches the EC into the running mode. This bit can be set at the same time as the EC init bit or the EC reset bit.
- Bit#3 (EC adaptation bit)
  This controls the LMS coefficients update. If set, echo adaptation can take place. The EC on bit must be set to enable the adaptation.
- Bit#4 (Noise reduction init bit)
  When set, the noise reduction algorithm does a full initialization.
- Bit#5 (Noise reduction on bit)
  Setting this bit switches the noise reduction algorithm into the running mode. This can be set at the same time as the noise reduction init bit.
- Bit#6 (Noise reduction with additional AGC bit)
  When set, the Wiener filter coefficients are done. The final result is amplified with the AGC algorithm. The noise reduction on bit must be set to enable this update.
- Bit#7 (AGC init bit)
  When set the AGC does a full initialization.
- Bit#8 (AGC on bit)
  When set AGC full operation is enabled. This bit can be set simultaneously to the AGC init bit.

Note: Bit#4 to bit#6 in the SWITCH are reserved for future use and have to be cleared.

*Note: The Handsfree function can be stopped by the **HF_ON** command with SWITCH = 0.*

Handsfree behavior is controlled by following parameters given via command **HF_SET_PAR**:

- GAIN_ANALOG
  The current setting of the external switchable analog gain has to be passed to the handsfree blocks. The highest selectable analog gain must always correspond to the maximum GAIN_ANALOG value of 32767_D. From this maximum lower gains lead to lower GAIN_ANALOG values.

Note: Whenever the current gain in the AFE gets changed for volume control, the echo characteristic seems to change too. Therefore, the volume changes must be passed to both AFE and handsfree simultaneously.

- STEP_WIDTH
  This controls the adaptation speed. The higher this value is, the faster the echo characteristic gets adapted. The maximum limit (stability point) for STEP_WIDTH is related to BLOCK_LENGTH as follows:

$$\text{STEP\_WIDTH} <= 2 * 32767 / \text{BLOCK\_LENGTH} \quad (5.7)$$

Note: The given limitation for STEP_WIDTH is theoretical, often it is necessary to reduce this value far below this limit to guarantee stability under real conditions.

- LMS_LENGTH
  This is the number of consecutive LMS coefficients that get updated. Therefore, it is also the maximum length (in time units of 125µs) of echo impulse response handsfree is able to cancel. This value has a linear influence on the handsfree total runtime (refer to **Table 5-3**).

  Table 5-3Relative Dependence of Runtime from LMS_LENGTH

LMS_LENGTH	100	200	300	400
Runtime relative to maximum	32%	55%	77%	100%

- LMS_OFFSET
  This is the number of taps in the LMS signal delay line that are skipped. With this value it is possible to avoid adaptation of LMS coefficients within an intrinsic delay. This value does not influence the runtime.

Note: Due to memory restrictions, there is a basic limitation:

$$LMS\_LENGTH + LMS\_OFFSET <= 400 \quad (5.8)$$

Note: although both, LMS_LENGTH and LMS_OFFSET can be changed in a running handsfree, the user has to be aware that, when reducing (LMS_LENGTH) and/or shifting (LMS_OFFSET) the updated LMS coefficients vector, those coefficients not being used do not get initialized. Shifting back to those coefficients may introduce a short additional echo caused by the handsfree function itself.

- BLOCK_LENGTH
  This can only be set to one of four distinct values. Any other setting leads to an immediate termination of a LMS coefficient update. This parameter influences the handsfree behavior in various ways:
  – Runtime, refer to **Table 5-4**.

  Table 5-4 Dependence of Runtime from BLOCK_LENGTH

BLOCK_LENGTH	2	4	5	8
Runtime[1)] (*) [ms]	6.32	4.50	4.13	3.58
Relative runtime [%]	100	71	65	57

  1) The absolute runtime calculated with LMS_LENGTH = 400.

  – Adaptation precision of LMS coefficients, increasing BLOCK_LENGTH increases the precision.
  – LMS robustness, increasing BLOCK_LENGTH increases robustness against instability.
  – Adaptation speed, increasing BLOCK_LENGTH decreases speed.

Note: A modification of BLOCK_LENGTH during a running handsfree needs also a modification of STEP_WIDTH.

- RXTX_RELATION
  This value tells the handsfree about the real world signal power relation between the loudspeaker and microphone and their acoustic coupling. For example, when a given signal power level x in the digital Rx path is applied to the D/A interface, the maximum possible signal power level y in the digital Tx path at the A/D interface (Tx signal due to worst case echo situation and in total absence of any additional near end signal) has to meet the following constraint (x, y given in dB):

$$y < x - 3/32 * RXTX\_RELATION \quad (5.9)$$

The handsfree function has an AGC in addition to the echo canceller. Based on the current signal power relation between RX and TX path a decision is made to attenuate the TX signal or not and the required attenuation gets calculated.

When the AGC decides to attenuate, ADD_ATTEN is added to the calculated attenuation. This summed attenuation is bounded to the limits given by MIN_ATTEN and MAX_ATTEN.

Note: For the attenuation level values and RXTX_RELATION, there is a relation to dB levels:

$$RXTX\_RELATION = RxTxRelation\ [dB] * 32/3 \quad (5.10)$$

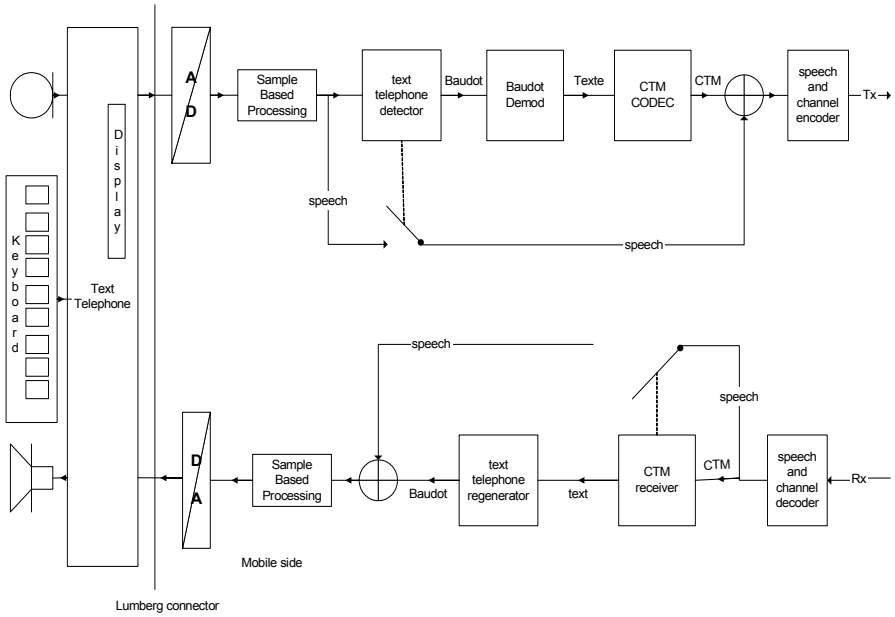
5.3.3 TTY/CTM

The TTY/CTM module allows the conversion of TTY signals from external terminals to adequate CTM signals for reliable text-telephony over mobile phones. TTY and CTM signals are designed to use the voice path, therefore, the TTY/CTM block is before the speech CODEC block.

Alternating voice and text is supported to allow the end user to speak and hear (see **Figure 5-6**):

- A user who cannot talk can hear his correspondent while sending him text: Hearing Carry Over
- A user who cannot hear can receive text from his correspondent while talking: Voice Carry Over.

Figure 5-6 Signal Flow in a TTY/CTM System



5.3.3.1 TTY Signals

TTY signals are analogue Baudot codes modulated by a Frequency Shift Keying (FSK) modem. The A/D converter sees these signals as if they were speech signals and communicated to the TTY-device through the normal speech path via the Lumberg connector.

Two tones are used to represent the asynchronous serial data. A binary ONE is represented by 1400 Hz +/-1% tone and a binary ZERO is represented by 1800 Hz+/-1% tone. A bit duration of 22 ms +/- 0.40 provide nominal data signalling rates of 45.45 bits/s.

Characters are preceded by a start bit (binary ZERO) and followed by a stop bit (binary ONE). The start bit is one bit-time in duration. The stop bit is 1.5 bit-times long. Each key depression causes the transmission of a complete Baudot character including the start and stop bits. 150 ms of binary ONE (1400 Hz) is transmitted as a preamble to the first character. A binary ONE hold tone also follows the last key depression. In this case, the hold tone is transmitted for a period of 150ms to 300ms after the end of the stop bit(s).

5.3.3.2 CTM Signals

To get more robustness, CTM signals are modulated differently from TTY signals.

The bit-stream is grouped in pairs of two bits. Each pair of two bits is modulated into a sine waveform of length 5 ms (40 samples) starting with a phase value of zero. The relation between the bits and the modulated waveform is as follows:

$$s(k) = 8 \cdot \text{round}\left(2047 \cdot \sin\left(2\pi \frac{f}{8000} k \right) \right) \quad \text{for } 0 \le k < 40, \tag{5.11}$$

where the audio samples are PCM-coded with 2's complement representation and with a minimum resolution of 13 significant bits, left-justified in a 16-bit word. The three least significant bits are to be set to '0' if not used for greater resolution. The amplitude is set to a value lower than maximum to avoid saturation of speech CODECS within the transmission path.

Depending on the values of bit#0 and bit#1, one of the frequencies 400 Hz, 600 Hz, 800 Hz, and 1000 Hz is used (refer to **Table 5-5**), provided that at least one of the two bits is not marked to be muted. If both bits are marked to be muted, a sequence of 40 zero-valued samples is generated.

The output signal is also zero if no valid bits are available at the CTM modulator's input.

Table 5-5 Frequency Parameter f for the CTM Modulator

	bit1 = 0	bit1 = 1	bit1 = mute
Bit0 = 0	400	600	600
Bit0 = 1	800	1000	1000
Bit0 = mute	800	1000	0

5.3.3.3 Switching between Speech and Data

if there are no text characters for transmission, the cellular text telephone modem includes adaptive switching between CTM signal generation and transparent transmission. This automatic switching is provided in both directions by means of the switch S1 in **Figure 5-6**.

Switch S1 is controlled by the functional blocks of the CTM transmitter, which can prevent forwarding the signal from the speech input of the CTM transmitter while the CTM modulator is active. Switch S2 is controlled by the functional blocks of the CTM receiver, which can block any CTM signal.

The switching between speech and TTY / CTM tones introduces a delay in the speech path. This delay is critical for the round trip delay of the end-to-end communication link:

Speech delay < 5ms (5.12)

this additional delay in the TTY / CTM converter must be the lowest possible value.

5.3.4 Voice Memo

Figure 5-5 Frame-Based Voiceband Processing shows the signal flow for the VoiceMemo/MMS functionality. The main VoiceMemo parts are the TransferBuffer 0 and 1 and the VoiceMemo-En-/Decoder (with the IF2 (interface 2) blocks in case of AMR):

Note: If any Voice Memo mode is selected (any VM_MODE *bit), the Voiceband (I^2S_x and/or AFE interface) must be initialized before starting the Voice Memo function.*
To stop the Voice Memo (in any mode) the MCU sends the **VM_CMD** *with* VM_MODE *set to 0.*

1. If a recording mode is selected, the uplink data are mixed with the TransferBuffer0 (scaled by *alpha0*), which already contains the speech samples from the downlink direction (scaled by *alpha1*).
2. If in the dedicated mode, the speech encoder processes the uplink data for transmission over the speech channel.
3. The data in TransferBuffer0 is processed according to the mode selected in the VM_MODE- word (refer to **VM_CMD**).

4. After the VM speech encoding is finished, the resulting speech frame bits need to be reordered according to the [26.101] Annex A+B standard for AMR recording (IF2 block) and packed to fit into the first 16 words of Voice-Memo-Buffer-0. For Fullrate recording they are packed in 17 words of the Voice-Memo-Buffer-0.
5. The Voice Memo data result is written to the Voice-Memo-Buffer-0 **SM_VM_BUFFER_0**.

There is one interrupt generated from the Voice Memo Module to the MCU. For VoiceMemo the Interrupt DSP_INT2 is used. To be able to distinguish the different interupt sources for the DSP_INT2 (MP3, Synth, PCM, VoiceMemo) the communication flag 8 is set by the Voice Memo Module at each interrupt. The flag has to be cleared by the MCU.

Note: All voice memo blocks are read and/or written by the DSP at approximately the same time within ~20ms.

To play VoiceMemo/MMS content, the MCU writes one block of VoiceMemo data to the Voice-Memo-Buffer-1 (**SM_VM_BUFFER_1**) before sending the appropriate Voice Memo command to the DSP with the correct parameter set. When the Voice Memo decoding task is processed, the data is read from Voice-Memo-Buffer-1. For AMR frames, the inverse bit re-ordering is applied to the data (IF2 block) before the speech decoder task is set up. No special decoding mode is chosen, as this information is available in the frame header. If the voice memo is in dedicated mode, swapping the CODEC static variables is necessary and the decoding result is written to TransferBuffer1. The data inside this buffer may be mixed into the uplink data by scaling with *gamma1* and to the downlink by scaling with *beta1,* which is controlled by the frame-based scheduler.

Different modes are supported by the Voice Memo function. These modes and the corresponding bits to set in the VM_MODE parameter are described in **VM_CMD**.

5.3.4.1 Data Interface

The data interface of the Voice Memo feature is in the three buffers described in **Table 5-6** (in the memory shared between the DSP and the MCU).

Table 5-6 Shared Memory Layout for Voice Memo

Shared Memory Layout	Word	AMR_PL	AMR REC	FR_PL	FR_REC	TRANSCODE
SM_VM_BUFFER_0	0	AMR-encoded audio frames (16 words)		FR-encoded audio frames (17 words)		AMR-encoded audio frames (16 words)
	1					
	...					
	15					
	16					
SM_VM_BUFFER_1	0		AMR frames to be de-coded (16 words)		FR frames to be de-coded (17 words)	FR frames to be transcoded (17 words)
	1					
	...					
	15					
	16					

Shared Memory Layout	Word	AMR_PL	AMR REC	FR_PL	FR_REC	TRANSCODE
DUMMY	0					
	1					
	...					
	8					
	9					

To record audio signals:

1. The MCU starts the VoiceMemo by sending the appropriate command to the DSP.
2. The DSP starts to encode the audio data and transmits the result to the Voice-Memo-Buffer-0.
3. The DSP issues an interrupt to the MCU when all of the encoded data has been written to the Voice-Memo-Buffer-0.
4. The MCU can immediately read the data from VoiceMemoBuffer0 without checking any communication flags as long as it can be guaranteed that the data is read within 20ms.
5. Recording continues until the MCU sends the command to stop the VoiceMemo.

For playing recorded or received VoiceMemo/MMS data:

1. The MCU writes one complete encoded frame (17 words) to Voice-Memo-Buffer-1 via the Shared Memory, which is considered to be the receiving buffer for playing VoiceMemo data.
2. The MCU sends the command to start the VoiceMemo play mode.
   The DSP issues an interrupt to the MCU after having read the data from the Voice-Memo-Buffer-1 to allow the MCU to refill the buffer with new data.
3. This continues until the MCU stops VoiceMemo by sending the appropriate command.

The size of Voice-Memo-Buffer-0 and -1 is 17 words (x16 bits) each, which is equivalent to 34 bytes.

Although AMR encoded speech frames may be shorter than 17 words, it is assumed, that the Voice-Memo-Buffer-1 (receiving buffer) is always filled with 17 words because padding bits set to zero fill words that are not used. The DSP always reads 17 words from this buffer, decodes the frame type and decides, how many words of the buffer are valid. The same is valid for the uplink: The Voice-Memo-Buffer-0 is always filled with zeros for the words that are not used.

As the frame duration is 20 ms and the maximum frame size is 17 words, the resulting maximum bit rate from MCU to the DSP is ((17word) x 16 bits/word)/20ms = 13.6kBit/s).

5.3.4.2 Storage Format

The storage format of AMR encoded audio content is defined in the RFC3267 Chapter 5 standard. Hence, the storage format is the concatenation of the file header (magic number "#!AMR\n") followed by the speech frames, each consisting of an octet frame header defined in the RFC3267 chapter 5.3 and the AMR core frame as defined in the 3GPP TS 26.101 Chapter 4.2/4.3 (IF1) standard. The AMR core frame is filled up with stuffing bits for octet alignment.

Note: As comparisons with existing implementations show, the format used is the AMR IF2 but using an 8-bit frame header in front of each AMR core frame instead of the 4-bit header defined in the AMR IF2!

The first octet of a frame defines the frame header:

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P	FT(3)	FT(2)	FT(1)	FT(0)	Q	P	P

The first bit is a padding bit (padding bits must be set to zero). The next four bits define the Frame Type Field.

Table 5-7 Frame Type

Frame Type Index	Frame Content
b0000 = 0	4.75 kbit/s
b0001 = 1	5.15 kbit/s
b0010 = 2	5.90 kbit/s
b0011 = 3	6.70 kbit/s (PDC-EFR)
b0100 = 4	7.40 kbit/s (IS 641)
b0101 = 5	7.95 kbit/s
b0110 = 6	10.20 kbit/s
b0111 = 7	12.20 kbit/s
b1000 = 8	AMR Comfort Noise
b1001 = 9	GSM-EFR Comfort Noise
b1010 = 10	IS-641 Comfort Noise
b1011 = 11	PDC-EFR Comfort Noise
b11xx = 12...14	For future use
b1111 =15	No transmission/reception

The Q bit works as a signalling flag for the payload quality:

0: Payload is severely damaged (the receiver should set the RX frame type to 'SPEECH_BAD' or 'SID_BAD' depending on the Frame Type).

1: Payload is valid.

After the Q-bit two additional padding bits have to be inserted.

The following bits form the AMR core format, which is the sequence of class A, class B, and class C bits. As the format is octet aligned stuffing bits are needed to fill up the last octet.

As an example **Table 5-8** shows the bit-, octet- and word sequence in the VoiceMemoBuffers:

Example CodecMode 6.70kBit/s:

`FrameTypeHeader`: 8 bit

`ClassA bits`: 58 bit

`ClassB bits`: 76 bit

`ClassC bits`: 0 bit

`Padding bits`: 2 bit

Table 5-8 VoiceMemoBuffer Bit alignment for AMR 6.70 kBit/s

Word	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
...	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	d(128)	...	...	...	...	d(133)	P	P	d(120)	...	...	...	...	...	...	d(127)
7	d(112)	...	...	...	...	...	...	d(119)	d(104)	...	...	...	...	...	...	d(111)
6	d(96)	...	...	...	...	...	...	d(103)	d(88)	...	...	...	...	...	...	d(95)
5	d(80)	...	...	...	...	...	...	d(87)	d(72)	...	...	...	...	...	...	d(79)
4	d(64)	...	...	...	...	...	...	d(71)	d(56)	d(57)	d(58)	...	...	...	...	d(63)
3	d(48)	...	...	...	...	...	...	d(55)	d(40)	...	...	...	...	...	...	d(47)
2	d(32)	...	...	...	...	...	...	d(39)	d(24)	...	...	...	...	...	...	d(31)
1	d(16)	...	...	...	...	...	...	d(23)	d(8)	...	...	...	...	...	...	d(15)
0	d(0)	d(1)	d(2)	d(3)	d(4)	d(5)	d(6)	d(7)	P	0	0	1	1	Q	P	P

Table 5-8 shows the appearance of the bits in the shared memory of the DSP for a 6.70 kBit/s AMR frame. The frame header is located in the high byte of the first shared memory word (note the order of the AMR core frame bits). For the 6.70 kBit/s AMR mode the number of octets is 18 (header plus AMR core frame) which fit into 9 words. It is assumed that the bits in words 9..17 are set to zero.

In Table 5-9 is the classification of the bits for all AMR modes.

Table 5-9 Speech CODEC Bit Classification Plus One Byte for Frame Header

AMR Speech CODEC FR	Coefficient Number	Bit Number	Class A	Class B	Class C	Number of Octets	Number of Words in Shared Memory
12.2	57	244	81	103	60	32	16
10.2	39	204	65	99	40	27	14
7.95	23	159	75	84	0	21	11
7.40	19	148	61	87	0	20	10
6.70	19	134	58	76	0	18	9
5.90	19	118	55	63	0	16	8
5.15	19	103	49	54	0	14	7
4.75	17	95	42	53	0	13	7
SID	5	35				5	3

5.3.4.3 Voice Memo Use-Cases

In VM_CMD in addition to the VM_MODE parameter, the Voice Memo parameters Par1 to Par6 (*alpha0*, *alpha1*, *beta0*, *beta1*, *gamma0* and *gamma1*) are used to mix the speech signals with the voice memo data streams (uplink and/or downlink). With these gains the type (playing or record) and the direction (Uplink and/or downlink) is set. Table 5-10 is a summary of the supported use-cases for the different FW states with the corresponding settings of the voice memo gains.

Table 5-10 Voice Memo Gain Settings for Different Use-Cases

Operation / Gain	alpha 0	alpha 1	beta 0	beta 1	gamma 0	gamma 1	Supported in FW states
Playing a voice memo and mixing it into the uplink path	0	0	0	0	x	x	TCH26, UMTS
Playing a voice memo and mixing it into the downlink path	0	0	x	x	0	0	TCH26, UMTS, IDLE, PDCH
Recording a voice memo out of the uplink path	x	0	0	0	0	0	IDLE, PDCH
Recording a voice memo after uplink and downlink signal have been mixed together	x	x	0	0	0	0	TCH26, UMTS

Example: If just the downlink signal is to be recorded, *alpha0* is set to 0 and *alpha1* is set to $7FFF_H$.

5.3.5 DTW Speech Recognizer

The DTW speech recognition SW is designed for the recognition of isolated words. Each word is represented by a normalized pattern (prototype) consisting of 15 feature vectors (FV), each composed of 12 components. The collection of references contains 2 prototypes per word in the "speaker dependent" mode and 4 prototypes per word in the "speaker independent" mode. The "dynamic time warping method" (DTW) is used to calculate the distances between normalized patterns.

The SW is split into a controller part and a DSP part. Data are exchanged between the two parts via the shared memory.

In this section only describes the DSP part and the interface needed for the communication between the CPU and the DSP.

5.3.5.1 Initialization

By calling the **DTW** command with SWITCH = 0 the DSP gets its configuration parameters for the detection of word boundaries and for adaptation. The configuration parameters are in **Table 5-11**.

Table 5-11 Description of DSP Configuration Parameters

PAR	Meaning	Description	Default value
1	MIN_LAENGE	For word boundary detection	6
2	MAX_LAENGE	For word boundary detection	100
3	MIN_DIFF	For word boundary detection	28
4	ENDE_MIN	For word boundary detection	15
5	ANF_MIN	For word boundary detection	30
6	WA_DEC	For word boundary detection	12
7	WA_INC	For word boundary detection	13
8	ANF_PAUSE	For word boundary detection	10
9	ENDE_MAX	For word boundary detection	20
10	ME_SCHWELLE	For word boundary detection	22
11	MIN_ANFEN	For word boundary detection	5
12	MIN_FREMD	For adaptation	1900

5.3.5.2 Normalized Pattern Calculation

DTW (SWITCH = 1) has to be called several times to calculate the normalized pattern. The core of algorithm is based on a state machine and each state is reached by calling one or several times the Normalized Pattern Calculation sub-routine (SR). **Table 5-12** shows the states used.

Table 5-12 Description of DSP Configuration Parameters

Job Code	Value	Description	Remarks
JC_GETFEATURE_INIT1	80_H	Initialization 1	1 SR call
JC_GETFEATURE_INIT2	81_H	Initialization 2	1 SR call
JC_GETFEATURE_WORK0	82_H	Initialization 3 and 1 FV calculation	1 SR call
JC_GETFEATURE_WORK0	83_H	FV calculation, looking for preliminary end of word	SR called as long as no word end found and timer not expired
JC_GETFEATURE_OFFLINE	84_H	Final end of word	1 SR call
JC_SEGMENTIERUNG_INIT	90_H	Initialization for segmentation	1 SR call
JC_SEGMENTIERUNG_NEXT	91_H	Calculate normal pattern	Several SR calls
JC_DPERK_INIT	$A0_H$		End of normal pattern calculate
JC_ABBRUCH	FF_H	Abortion of word processing	error, see table

The initialization is distributed over three states because only 80 new input speech samples are available with each SR call and one analysis frame consists of 200 speech samples (and 56 appended zeroes). Thus, enough samples for the FV calculation are not available before the third SR call.

On the controller part the processing is based on 160 new input speech samples, this means two SR are executed by the **DTW** command when SWITCH = 1.

A SR call in state JC_GETFEATURE_WORK calculates one FV and the frame energy that is needed for the word boundary detection. At every second call (20 ms) an averaged FV and frame energy are stored in buffers holding the latest three seconds. Additionally the preliminary word boundaries are looked for and the state JC_GETFEATURE_WORK is left as soon as the preliminary word end has been found. The maximum length of stay in state JC_GETFEATURE_WORK is determined by an internal timer that expires after 65534 x ANA_PAUSE[1)] = ~11 min.

For the FV calculation, frames of 200 input samples are (optionally) filtered by a pre-emphasis filter, windowed by a Hamming window, and completed by a block of 56 zeroes. An FFT calculates the spectrum of this frame. An (optional) noise reduction can be performed in the frequency domain by spectral subtraction. For each discrete frequency the logarithm of the absolute value of the FFT result is calculated. 11 of the 12 components of the FV are calculated by a linear combination of a number of these logarithms around certain center frequencies (filter bank). The first component is the logarithm of the AC energy of the frame.

In state JC_GETFEATURE_OFFLINE the final word boundaries are determined.

In state JC_SEGMENTIERUNG_NEXT the 15 final FVs of the normalized pattern are calculated by an averaging process over the stored FVs between the word boundaries (refer to **Table 5-16**).

Table 5-13 Description of DSP Input Parameters for Normalized Pattern Calculation

Variable	Length in Words	Description
job_state	1	One of the job codes from **Table 5-17 “Description of Error Codes for Normalized Pattern Calculation” on Page 124**
flg_noisered	1	0: Noise reduction off !=0: Noise reduction on
flg_prefilter	1	0: Pre-emphasis filter off !=0: Pre-emphasis filter on

At the end of two SR executions the variables in **Table 5-14** are returned to the controller via the shared memory (independent of the state). After the first SR execution, the output of the first SR is internally directly transferred to input for the second SR.

Table 5-14 Description of DSP Output Parameters for Normalized Pattern Calculation

Variable	Length in Words	Description
job_state	1	New job state for next SR call
err_status	1	Error status (refer to **Table 5-17 “Description of Error Codes for Normalized Pattern Calculation” on Page 124**)

1) By default ANA_PAUSE = 10 ms.

If job_state = JC_SEGMENTIERUNG_INIT after the SR execution, the parameters in **Table 5-15** are also returned via the shared memory.

Table 5-15 Description of Additional DSP Output Parameters for Normalized Pattern Calculation

Variable	Length in Words	Description
word_start	1	In 20 ms intervals
word_end	1	In 20 ms intervals
anf_energie	1	Silence energy
max_fram	1	Interval # of energy maximum
max_energie	1	Energy maximum
en_mw	1	Average word energy

If job_state = JC_DPERK_INIT after the SR execution, the end of the calculation of the normalized pattern has been reached and the parameters in **Table 5-16** are also returned via the shared memory.

Table 5-16 Description of DSP output parameters for normalized pattern calculation

Variable	Length in Words	Description
fv_norm	15*6	Normal pattern, packed (2 FV components in 16 bits)

The variable err_status indicates the occurrence of an error during program execution. **Table 5-17** contains the error codes.

Table 5-17 Description of Error Codes for Normalized Pattern Calculation

Name	Value	Description
NO_ERR	1	No error
ERR_WORD_SHORT	1	Word too short
ERR_WORD_LONG	2	Word too long
ERR_WORD_BORDER	3	Beginning of word >= end of word
ERR_FR_COUNT_OVERFLOW	4	Maximum analysis time expired
ERR_INVALID_JC	5	Invalid job code

5.3.5.3 Calculation of Distance between Actual Normalized Pattern and a Reference Pattern

The distance is calculated by calling the **DTW** command with SWITCH = 2. Before each call the input parameters in **Table 5-18** must be written in the shared memory.

Table 5-18 Description of DSP Input Parameters for Distance Calculation

Variable	Length in Words	Description
fv_norm	90	Actual normalized pattern, packed
anzproto	1	Number of prototypes per reference entry
referenz	184 or 364	Reference pattern, length is depending on the mode (number of prototypes): Speaker dependent: 184 (2 prototypes) Speaker independent: 364 (4 prototypes)

At the end of the SR execution the calculated distance (score) is returned to the controller via the shared memory (refer to **Table 5-19**).

Table 5-19 Description of DSP Output Parameter for Distance Calculation

Variable	Length in Words	Description
score	1	Distance

One SR call is performed for each entry in the collection of reference data to calculate its distance to the actual normalized pattern. If the SR is called with an empty reference (number of trained utterances for this word equals zero), MAX_SCORE ($3FFF_H$) is returned as distance.

5.3.5.4 Adaptation of a Reference

The reference pattern is adapted by calling the **DTW** command with SWITCH = 3. Before each call the input parameters in **Table 5-20** must be written in shared memory.

Table 5-20 Description of DSP Input Parameters for Adaptation

Variable	Length in Words	Description
fv_norm	90	Actual normalized pattern, packed
word_start	1	In 20 ms intervals
word_end	1	In 20 ms intervals
n_hits	1	Number of valid entries in min_score5[]
min_score5[0]	1	Training: distance to "home" reference Recognition: best score
min_score5[1]	1	Training: distance to the nearest "not home" reference Recognition: second best score
anzproto	1	Number of prototypes per reference entry
referenz	184 or 364	Reference pattern, length is depending on the mode (number of prototypes): Speaker dependent: 184 (2 prototypes) Speaker independent: 364 (4 prototypes)

At the end of the SR execution the adapted reference pattern is returned to the controller via the shared memory (refer to **Table 5-21**).

Table 5-21 Description of DSP output parameter for adaptation

Variable	Length in Words	Description
referenz	184 or 364	Reference pattern

5.3.5.5 DTW Speech Recognizer Use-Cases

Each switch case of **DTW** command requests input parameters that written in shared memory at **SM_DTW_PAR** before launching the command itself. After execution the input parameters are overwritten in **SM_DTW_PAR** by the output parameters.

The 3 main DTW speech recognizer algorithms are controlled by the MCU using the **DTW** command. To start one of them, the following sequence has to be used:

1. The initialization is executed by the DSP with the **DTW** command (SWITCH = 0). Refer to **Section 5.3.5.1 "Initialization" on Page 122** for the input parameters.
2. The input parameters are written in shared memory and the specific algorithm is enabled by sending the command **DTW** to the DSP with the parameter SWITCH set to 1,2, or 3.
3. The algorithm itself is executed as soon as the command **VB_ON** is sent with the appropriate parameters. Only in the case of normalized pattern calculation (SWITCH=1) is the AFE output really used. In the other cases (SWITCH = 2 and SWITCH = 3), even though no used, the AFE configuration must be switched on.
4. The DSP generates DSP_INT3 interrupts for the MCU at the end of the execution. The output parameters are ready to be fetched by the MCU in **SM_DTW_PAR**.
5. For calculation of distance or adaptation of one reference (SWITCH = 2 and SWITCH = 3), the sub-routines terminate.
6. The audio scheduler is switched off by **VB_ON** command.
7. For normalized pattern calculation the **DTW** command (SWITCH = 1) is sent again with the previous output job state parameter as the input job state parameter. From JC_GETFEATURE_INIT1 to JC_GETFEATURE_OFFLINE job state each command must be within 20 ms after the interrupt.
   a) If the current state is not a final state (JC_DPERK_INIT or JC_ABBRUCH), the MCU goes back to step **7**.
   b) If the current state is a final state, the audio scheduler is switched off by **VB_ON** command.

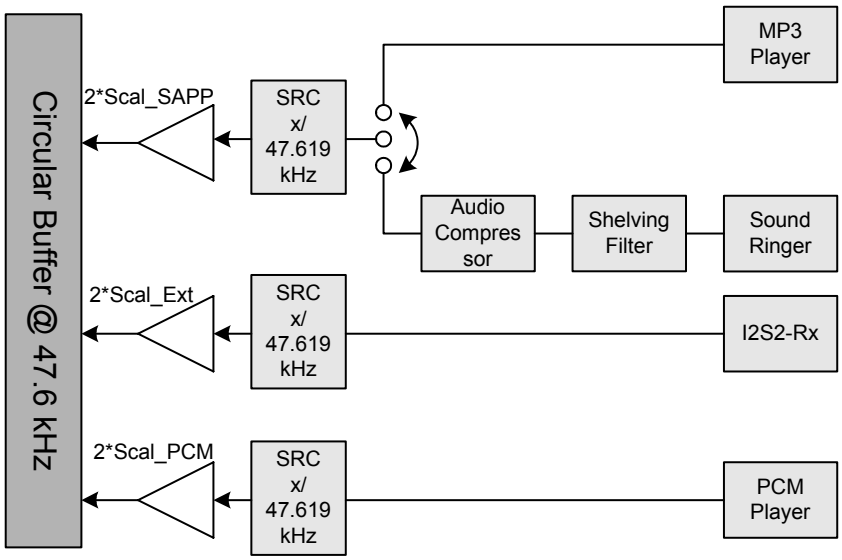
Note: Step 2 and 3 can be swapped.

5.4 Circular Mixing Buffer

The circular buffer is a 3000-word buffer that stores and mixes the voiceband samples from external sources. The signals to be stored must be stereo signals with a sampling rate of 47.619 kHz. The buffer has a circular structure, so that when the write pointer reaches the end of the buffer, it is wrapped to the beginning address of the buffer. The buffer has three inputs:

1. MP3 and Synthesizer data:
   a) The sound application data can have anyone of a set of different sampling rates, therefore, the sampling rate of the input signal is first converted to the circular buffer sampling rate 47.619 kHz (refer to **Section 5.4.6 "Block-Based Sample Rate Converter" on Page 133**).
   b) The signal is scaled by 2*Scal_SAPP.
   c) The signal is mixed into the circular buffer (see **Figure 5-7**).
2. External data from I^2S_y-Rx:
   a) The I^2S_y-Rx data can have anyone of a set of different sampling rates, therefore, the sampling rate of the input signal is first converted to the circular buffer sampling rate 47.619 kHz (see **Section 5.4.6**).
   b) The signal is scaled by 2*Scal_Ext.
   c) The signal is mixed into the circular buffer (see **Figure 5-7**).
3. PCM data from the PCM Player:
   a) The PCM data can have anyone of a set of different sampling rates, therefore, the sampling rate of the input signal is first converted to the circular buffer sampling rate 47.619 kHz (see **Section 5.4.6**).
   b) The signal is scaled by 2*Scal_PCM.
   c) The signal mixed into the circular buffer (see **Figure 5-7**).

Figure 5-7 Circular Mixing Buffer



5.4.1 Circular Mixing

The input samples of the circular buffer are not simply written into the buffer. They are added to the already stored sample and the result is then written into the buffer.

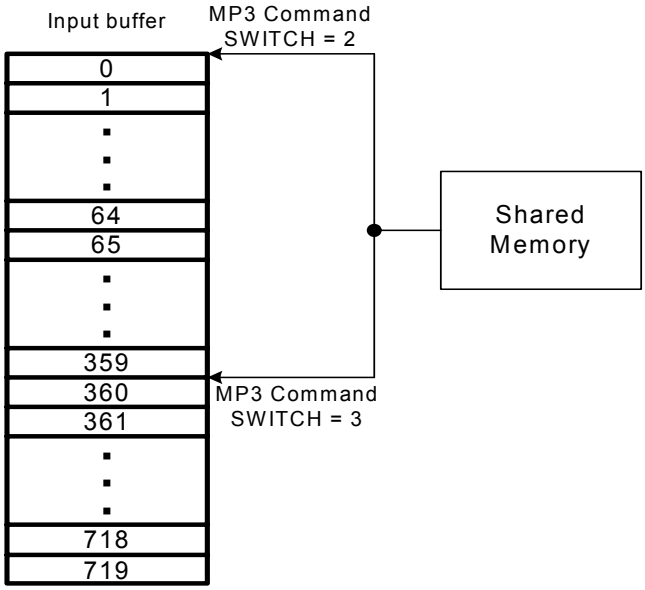
The number of sample pairs that have to be read out has to be the same as the number of the sample pairs delivered by the 8 (or 16)-to-47.619 kHz interpolator at the downlink output of the audio scheduler, since these pairs are then mixed together to deliver the AFE-Rx output.

After reading out the sample pairs, they have to be cleared in the circular buffer. The number of free samples in the buffer is then checked. If more than 320 samples are free (= 6.72 ms) and the MP3 or Synthesizer is initialized, new data is requested.

If the PCM Player is initialized and more than 600 samples are free, a new data request is sent to the MCU.

5.4.2 MP3

Figure 5-8 Writing in the MP3 Internal DSP Input Buffer



The MP3 decoder is controlled by the MCU using the **MP3** command. To start the decoder the following sequence has to be used:

1. The first frame is sent to the DSP with the **MP3** command (SWITCH = 2 or 3). The DSP always expects 2*360 words. The data words must be written by the MCU in the little endian format, the first byte is written in the LSB.
2. The MP3 is enabled and the sampling rate is set up by sending the command **MP3** to the DSP with SWITCH = 1.
3. The DSP generates DSP_INT2 interrupts for the MCU to get new frames.
4. To be able to distinguish the different interupt sources for the DSP_INT2 (MP3, Synth, PCM, VoiceMemo) the communication flag 9 is set by the MP3. This flag has to be cleared by the MCU.
5. At every Interrupt, the MCU sends the next frame to the DSP with the **MP3** Command (SWITCH = 2 or 3). The frame must always start at the position in where the pointer in the command points to the shared memory. The first 360 words are sent with the **MP3** command SWITCH = 2 and the second 360 words with SWITCH = 3.
6. After the communication flag for the Switch = 2 command is cleared by the DSP, the same memory locations for the frame data on the shared memory can be used for the Switch = 3 command.
7. When the end of the file is reached (the last frame is sent to the DSP), the **MP3** command with SWITCH = 0 is sent to stop the MP3 decoder.

When the command MP3 with the parameter SWITCH = 2 or 3 is received by the DSP, the shared memory location where the data is stored is identified from the PAR1 in the command and the 360 data words are copied from the shared memory to an internal 720-word buffer. Depending on the value of SWITCH, a different offset in the internal buffer is used to store the data (see **Figure 5-8**):

- SWITCH = 2: Data copied at beginning of internal buffer.
- SWITCH = 3: Data copied with an offset of 360 words in the internal buffer.

The MP3 decoder writes the MP3 parameters to **SM_MP3_PAR_OUT** (refer to **Table 5-22**).

Table 5-22 Description of MP3 Parameters

Name	Value	Description
Number of copied bytes	>1	Number of bytes in actual frame (depends on sampling frequency, bitrate and padding bit in header)
	1	Invalid header data
Error State	0	No error in header
	> 0	Error number for severe numbers
	< 0	Error number for light numbers
Mono/Stereo flag	0	Mono
	1	Stereo

Each MP3 decoder run provides 2*576 = 1152 samples as output and passes them to the circular buffer. Every time the samples are passed to the buffer the algorithm is started again. Thus, the number of interrupts expected depends on the sampling rate. For example, at a sampling rate of 48 kHz the MP3 decoder runs every 12 ms, that is, the DSP generates an interrupt to MCU every 12 ms. **Table 5-23** gives the number of granules and the times between interrupts for different sampling rates.

These time periods refer to maximal times. The real values can vary due to the low priority of the MP3 decoder in the DSP. If the DSP has to process another task, the MP3 might be delayed by worst case cycles, that is, the MCU has less time to provide new data to the DSP. Nevertheless, the interrupt is given for the 48kHz example every 12 ms on the average.

To switch off the MP3 decoder, the **MP3** command with SWITCH = 0 is sent. It is guaranteed that the last received data frame is processed before the DSP deactivates the MP3 decoder.

Table 5-23 Number of Granules and Interrupts Depending on Sampling Rate

Sampling Rate (kHz)	Number of Granules	Time between Interrupts (ms)
16	1	36
22.1	1	26
24	1	24
32	2	18
44.1	2	13
48	2	12

5.4.3 Synthesizer

The Synthesizer is controlled by the MCU using the **SYNTH** command. To start the Synthesizer the following sequence has to be used:

1. The first frame is send to the DSP with the **SYNTH** command (SWITCH = 2). The frames have a maximum length of 65 words for 20 ms.
2. The Synthesizer is enabled and the sampling rate is set up in the DSP by sending the command **SYNTH** with SWITCH = 1.
3. The DSP generates DSP_INT2 Interrupts for the MCU to get new frames. To be able to distinguish the different interrupt sources for the DSP_INT2 (MP3, Synth, PCM, VoiceMemo) the communication flag 10 is set by the Synthesizer. This flag is cleared by the MCU.
4. At every Interrupt, the MCU sends the next frame to the DSP with the **SYNTH** Command (SWITCH = 2). The frame always starts at the position where the pointer in the command points to in the shared memory.
5. When the end of the file is reached (the last frame is sent to the DSP), the **SYNTH** command with Switch = 0 is sent to stop the Synthesizer.

When the command **SYNTH** with SWITCH = 2 is received by the DSP, the shared memory location where the data is stored is identified from the PAR2 in the command. Then the 65 data words are copied from the shared memory to the internal buffer.

After each frame the following parameters (refer to **Table 5-24**) are calculated and written to shared memory:

1. Synthesizer Status:
   a) This parameter is calculated for each granule.
   b) The four values are combined (by using the OR operation) to build the status for the whole frame.
   c) The status is copied to **SM_SYNTH_PAR_OUT**. Only two bits are used in the status word (the rest are reserved for future use):
   – Bit#0 Limit flag:
   Using this bit the MCU decides whether the headroom should be increased or not via the parameter Headroom in the **SYNTH** Command (SWITCH = 2). If the MCU sends a new headroom value to the DSP, this value is used for the next Synthesizer run even if the previous one is still being processed (see **Figure 5-9**).
   – Bit#1 Metronom flag:
   This is set if a metronom-click occurred in current frame.
2. Low RMS value:
   a) A low pass is applied to the Synthesizer output samples (including audio post-processing).
   b) An RMS calculation is done over the resulting signal.
   c) The result is written to t **SM_SYNTH_RMS_LOW**.
3. High RMS value:
   a) A high pass is applied to the Synthesizer output samples (including audio post-processing).
   b) An RMS calculation is done over the resulting signal.
   c) The result is written to the shared memory location **SM_SYNTH_RMS_HIGH**.

Figure 5-9 Timing Diagram for Synthesizer

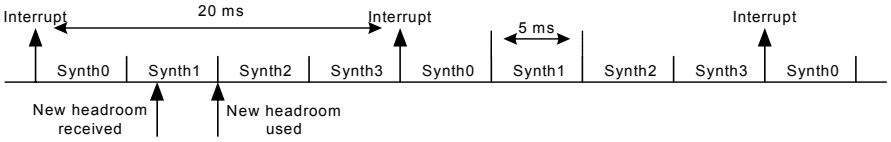


Table 5-24 Description of Synthesizer Parameters

Name	Value	Description	SM Location
Limit flag	0	No overflow	Bit 0 of SM_SYNTH_PAR_OUT
	1	Overflow (clipping) occurred during summing up of the output data of all voices. The Headroom should be increased (refer to Section 3.3.38)	
Metronom flag	0	No Metronom-Click occurred	Bit 1 of SM_SYNTH_PAR_OUT
	1	Metronom-Click occurred in current frame	
RMS low	Log.	RMS value of low frequency part of Synthesizer output	SM_SYNTH_RMS_LOW
RMS high	Log.	RMS value of high frequency part of Synthesizer output	SM_SYNTH_RMS_HIGH

Each Synthesizer run provides 5 ms of output and passes the samples to the Circular Buffer. The data frames provided by the MCU correspond to 20 ms of output. Therefore, the Synthesizer is called four times to process one MCU data frame. When all samples are passed to the Circular Buffer the algorithm is started again.

If the DSP has to process another task, the interrupts might be delayed by worst case cycles. In this case, the MCU has less time to provide new data to the DSP. Nevertheless, the interrupt is sent every 20 ms on the average.

To switch off the Synthesizer, the **SYNTH** command with SWITCH = 0 is sent. It is guaranteed that the last received data frame is processed before the DSP deactivates the Synthesizer.

5.4.3.1 Audio Postprocessing for Synthesizer

The output samples of the Synthesizer are post-processed by two modules:

- High Frequency Shelving Filter:
  This module is implemented as a first order IIR Filter, which is used to boost the audio signal high frequencies. Its transfer function is given by:

$$H_{SZH}(z) = \frac{b_0 \cdot 2^{b\_exp} + b_1 \cdot 2^{b\_exp} \cdot z^{-1}}{1 + a_1 \cdot z^{-1}} \quad (5.13)$$

where b_exp, b0, b1, and a1 are the filter coefficients set by the **AUDIOPOSTPROC** command. For more details refer to **[5]**.

- Audio Compressor:
  The audio compressor is a device for manipulating the dynamic range of mono or stereo audio signals. The audio compressor is controlled by 14 configuration parameters (refer to **Table 5-25**).

Table 5-25 Description of the 14 Parameters for the Audio Compressor

Parameter Name (16-bit format)	Range and Number Representation	Description
mono_flag	0: If IO-buffer is stereo 1: If IO-buffer is mono	Flag, which indicates if input buffer is stereo or mono. (the maximal valid length is limited by the define)
m_bufflen	Range: 1 to 32767	Length of IO-buffer in number of samples.

Parameter Name (16-bit format)	Range and Number Representation	Description
m_inv_bufflen	Range: 1 to 32767	Inverse of buffer length in Q1.15 representation.
m_hp_coeff_exp	Range: 0 to 15	Determines coefficients for HP-Filter, which is used for DC-removal: b0 = -b1 = -a1 = 1 - 1/2^coeff_exp $$H(z) = \frac{b_0 + b_1 z^{-1}}{1 - a_1 z^{-1}}$$
m_lp1_coeff	0 to 32767	Filter coefficient determining the attack time of the RMS-level measurement.
m_lp2_coeff	0 to 32767	Filter coefficient determining the release time of the RMS-level measurement.
m_lp4_coeff	0 to 32767	Filter coefficient determining the attack time of the compressor.
m_lp3_coeff	0 to 32767	Filter coefficient determining the release time of the compressor.
m_L_A	-96*256 to 0	Lower compressor input threshold. Must fulfill m_L_A < m_L_B
m_L_B	-96*256 to 0	Upper compressor input threshold. Must fulfill m_L_A < m_L_B
m_G_comp	-96*256 to 96*256	Output compensation gain: Sets output level of L_B: L_B_out = L_B+G_comp
um_R_infA	0 to (2^16 -1)	Gradient of compression curve for input levels in the range of -96 dB to L_A (only positive gradients between 0 and 90 degrees are possible).
um_R_AB	0 to (2^16 -1)	Gradient of compression curve for input levels in the range of L_A to L_B (only positive gradients between 0 and 90 degrees are possible).
um_R_B0	0 to (2^16 -1)	Gradient of compression curve for input levels in the range of L_B to 0dB (only positive gradients between 0 and 90 degrees are possible).

For more details refer to **[6]**.

5.4.4 I^2S_y External Mode

This mode is described in **Section 5.1.3 “I2Sy Interface” on Page 106**.

5.4.5 PCM Player

Note: In the following section, the term ’PCM Player’ refers to both the PCM and ADPCM Player.

5.4.5.1 Interface to Controller

The PCM Player is controlled by the MCU using the **PCMPLAY** command. To start the PCM Player, the following sequence has to be used:

1. First the PCM Player is enabled and the format, sampling rate, and mode is set up in the DSP by sending the **PCMPLAY** with SWITCH = 1.
2. The DSP generates DSP_INT2 Interrupts for the MCU to get data.To be able to distinguish the different interrupt sources for the DSP_INT2 (MP3, Synth, PCM, VoiceMemo) the communication flag 11 is set by the PCM Player. This flag is cleared by the MCU.
3. At every Interrupt, the MCU sends the next data to the DSP with the **PCMPLAY** (SWITCH = 2).

4. To stop the playback of PCM Data **PCMPLAY** (SWITCH = 0) is sent by the MCU instead of a new data command after an interrupt to stop the PCMPlayer.
5. If during a playback of data, the format, sample rate, or mode has to be changed, the MCU has to send a **PCMPLAY** (SWITCH = 1) with the appropriate settings for the Parameters 1-3 instead of a new data command after the DSP raised the interrupt

When **PCMPLAY** (SWITCH = 2) is received by the DSP, the amount of data identified by the PAR1 in the command is copied from **SM_VM_PCM_BUFFER_1** to the internal buffer. After the communication flag of the command is cleared, the memory is free again.

To reduce the interrupt load and the DSP load, it is recommended that the MCU always sends the maximum allowed amount of data to the DSP with each **PCMPLAY** command. The maximum Data size and the resulting interrupt load is shown in **Table 5-26**.

Table 5-26 Interrupt Rate for Different Sample Rates

Sampling Rate (kHz)	Time between interrupts (ms)	Maximum Data Package size		
		PCM (8-bit and 16-bit)[1)]	ADPCM No Sync[2)]	ADPCM with Sync[3)]
8	6.25	2*50	2*12	2*12
11.025	6.26	2*69	2*17	2*17
12	6.25	2*75	2*18	2*18
16	6.25	2*100	2*25	2*24
22.025	6.26	2*138	2*34	2*34
24	6.25	2*150	2*37	2*37
32	6.25	2*200	2*50	2*49
44.1	6.26	2*276	2*69	2*68
47.619	6.25	2*300	2*75	2*74
48	6.25	2*300	2*75	2*74

1) For PCM 8-bit format one Data Word corresponds to two PCM samples. Therefore the number of words copied to shared memory has to be half the number of PCM samples.
2) This is only the number of Data Words. Additionally two header words have to be sent at the beginning of the frame.
3) This is only the number of Data Words. Additionally two header words and two Synchronization words have to be sent.

Note: For Mono, the maximum Data size is always 1 the given amount.*

Note: For ADPCM the content of the frame is not allowed to exceed the given sample amount.

For stereo playback, the MCU has to provide 2 Frames (one for the left and one for the right channel) in the shared memory. These two frames have to contain the same amount on PCM samples. The position of both frames in the shared memory is different between the codes:

- PCM 16-bit: The left channel and right channel 16-bit samples are written alternately into the shared memory.
- PCM 8-bit: The left channel and right channel 8-bit samples are written alternately into the shared memory (one word on shared memory contains one byte for left channel and one byte for right channel).
- ADPCM: First the left channel has to be provided and immediately after the right channel is stored without a gap in between.

5.4.5.2 ADPCM Decoder

The ADPCM data structure is shown in following tables. The first word of the ADPCM data is called SYNC_INDEX.

If it is set to:

- $FFFF_H$, there is no synchronization double-word contained in the ADPCM data (refer to **Table 5-27**).

- If it is set to any other value, it denotes the position of the synchronization double-word in the following data (refer to **Table 5-28**).

Table 5-27 ADPCM Data Structure without Synchronization Double-Word

Word Index	Content
0	SYNC_INDEX ($FFFF_H$)
1	BLOCK_LENGTH = ADPCM_DATA_WORDS
2...BLOCK_LENGTH+1	ADPCM data

Table 5-28 ADPCM Data Structure with Synchronization Double-Word

Word Index	Content
0	SYNC_INDEX
1	BLOCK_LENGTH = ADPCM_DATA_WORDS+2 Synch words
2...BLOCK_LENGTH+1	ADPCM data and synchronization double-word

The synchronization Double-word consists of 32 bit as shown in **Table 5-29** and corresponds to the synchronization word contained in the ADPCM 'wav' stream. The synchronization index SYNC_INDEX has been introduced to the data stream to allow re-synchronization in case where a data packet is missing.

Table 5-29 ADPCM Synchronization Word

Word Index	Content
0	PCM_SAMPLE
1	STEP_TABLE_INDEX (lower byte used only)

The number of words used in the ADPCM data area (refer to **Table 5-26**) depends on:

- Sample rate
- Inclusion of a synchronization word.

If a sync double-word is included, 1 ADPCM Data Word less has to be sent in the frame (otherwise, the internal buffers in the DSP would have an overflow). This means the maximum length of the frame would then be 1 Data Word more than without a sync double-word (sync double-word adds 2 Words, but one ADPCM Data Word less is sent in this frame).

5.4.6 Block-Based Sample Rate Converter

The signal, which builds the input for the circular buffer (MP3 and Synthesizer from the DSP and the external input from the I^2S_y-Rx) can run at one of different possible sampling rates (8, 16, 22.05, 24, 32, 44.1, and 48 kHz). The buffer has to have a fixed sampling rate of 47.619 kHz. Therefore, two SRCs are used, one for each input path (Sound Applications and I^2S_y-Rx). Unlike the sample based SRC, which can only process one fixed input/output sampling rate pair, the block based SCR is a general function, where the input sampling rate is a parameter and the output rate is fixed to 47.619 kHz. The input sample rate is handled as an index (refer to **Table 5-30**).

Table 5-30 Input Sampling Rate Index Values

Index	Sampling Rate (kHz)
0	8
1	16
2	22.05
3	24
4	32
5	44.1
6	48
7	47.619
8	11.025
9	12

5.5 DAI Functions

For type approval testing of audio and vocoder functions E-GOLDradio has to be connected to the system simulator. The interface between the system simulator and the E-GOLDradio is described in the GSM 04.14 standard.

The DAI mode can be activated by using the command **VB_DAI**. The mobile station may get the DAI test mode either via a layer 3 message or directly via two pins from the system simulator. The MCU has to check for Reset and the two pins from the system simulator or a Layer 3 message.

To activate the appropriate DAI mode the MCU sends two messages to the DSP in the following order:

1. **VB_I2Sy**: The I^2S_y has to be activated and configured for DAI operation (that is, SWITCH = 2).
2. **VB_DAI**: The parameter MODE has to be set to 1, 2, or 3.

Normal Mode

This is the normal operational mode. The samples computed by Uplink path of the sample-based process part (see **Figure 5-1**) are written to the 160-sample voiceband buffer. One sample is copied out of the 160 sample voiceband buffer to build the input for the Downlink path of the sample-based process part (see **Figure 5-1**).

Vocoder Test

E-GOLDradio reads the sample from the DAI-Rx register and transfers it to the voiceband buffer. It reads a voice sample from the voiceband buffer and writes the sample to the DAI-Tx register (see **Figure 5-10**). The contents of the voiceband filter are not changed in this mode, so the microphone signal is looped back to the loudspeaker.

Acoustic Test

E-GOLDradio reads the sample from the DAI-Rx register and transfers it to the hardware voiceband filter. It reads a voice sample from the hardware voiceband filter and writes the sample to the DAI-Tx register. The contents of the voiceband buffer are not changed in this mode, so the downlink signal is looped back to the uplink (far end speaker will hear a loop) as shown in **Figure 5-11**.

Voiceband Test

The Downlink sample at the output of the voiceband buffer is copied into the input of the voiceband buffer in the Uplink direction (see **Figure 5-12**), so the downlink signal is looped back to the uplink (far end speaker will hear a loop). The content of the voiceband filter is not changed, so the microphone signal is looped back to the loudspeaker (near end speaker will hear a loop).

Return to Normal Mode

To return to the normal mode (to stop the Acoustic test mode, Vocoder test mode, or Voiceband test mode) **VB_DAI** (with MODE = 0) is sent (see **Figure 5-10**, **Figure 5-11**, and **Figure 5-12**).

Figure 5-10 Vocoder Test

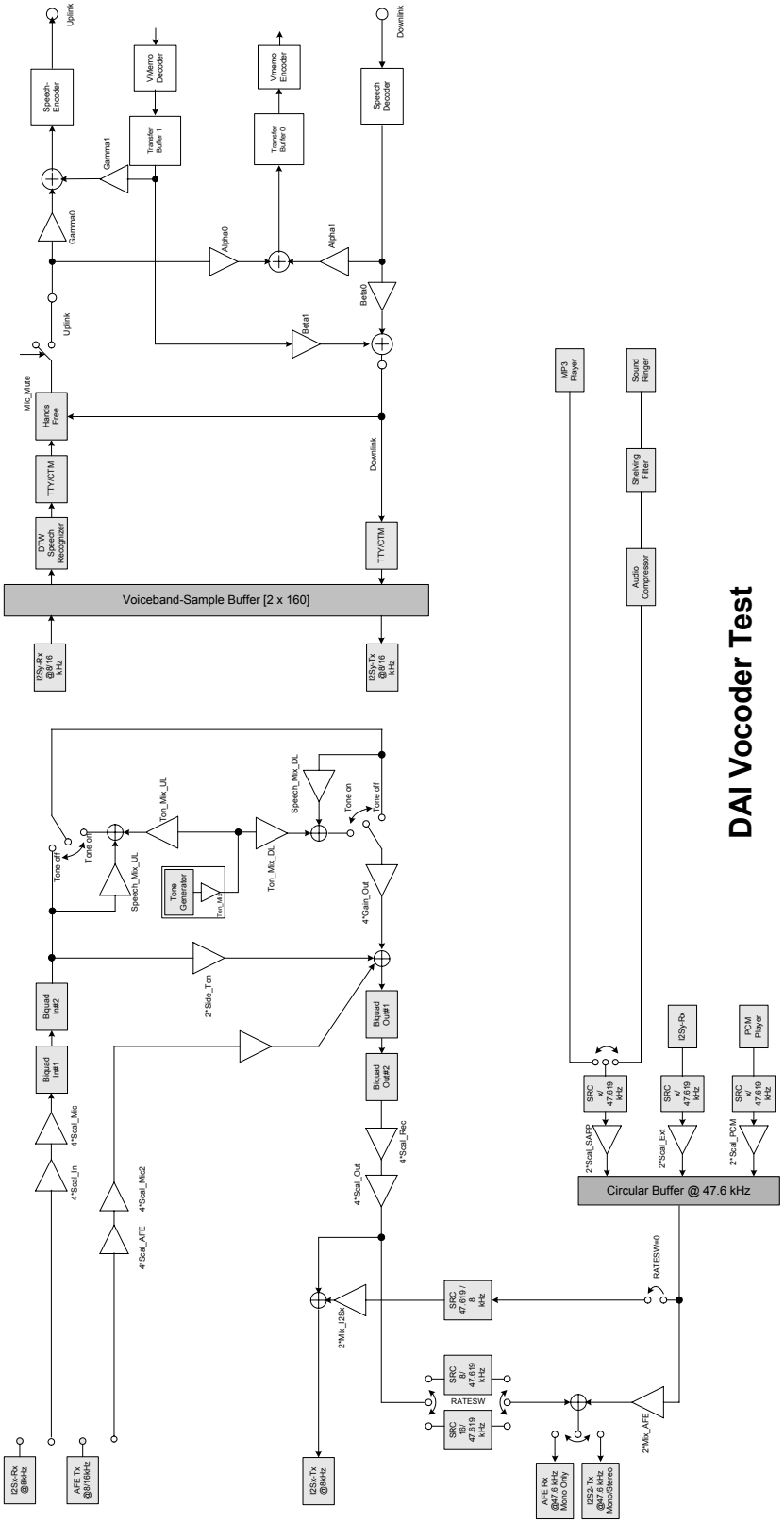


Figure 5-11 Acoustic Test

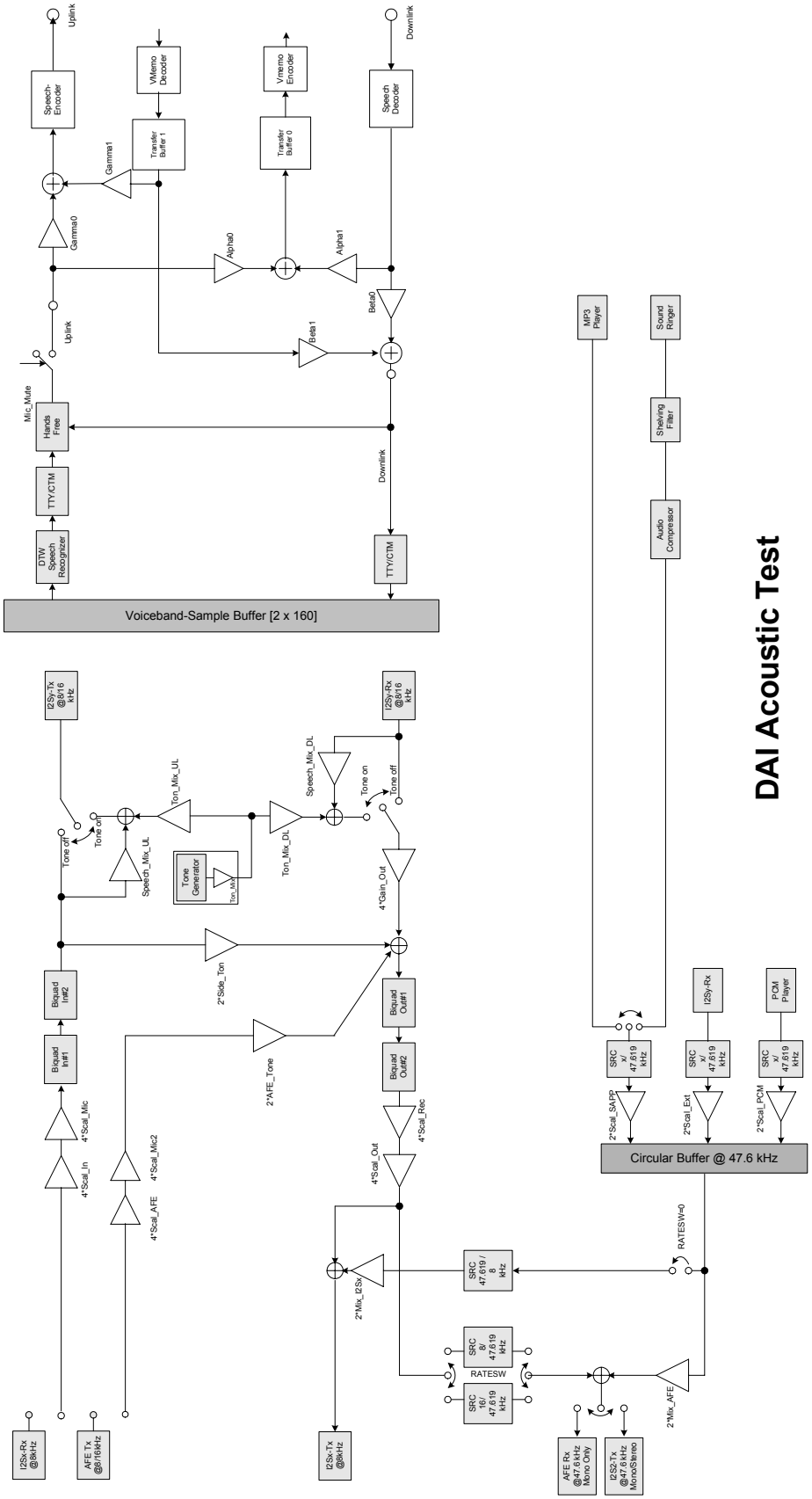
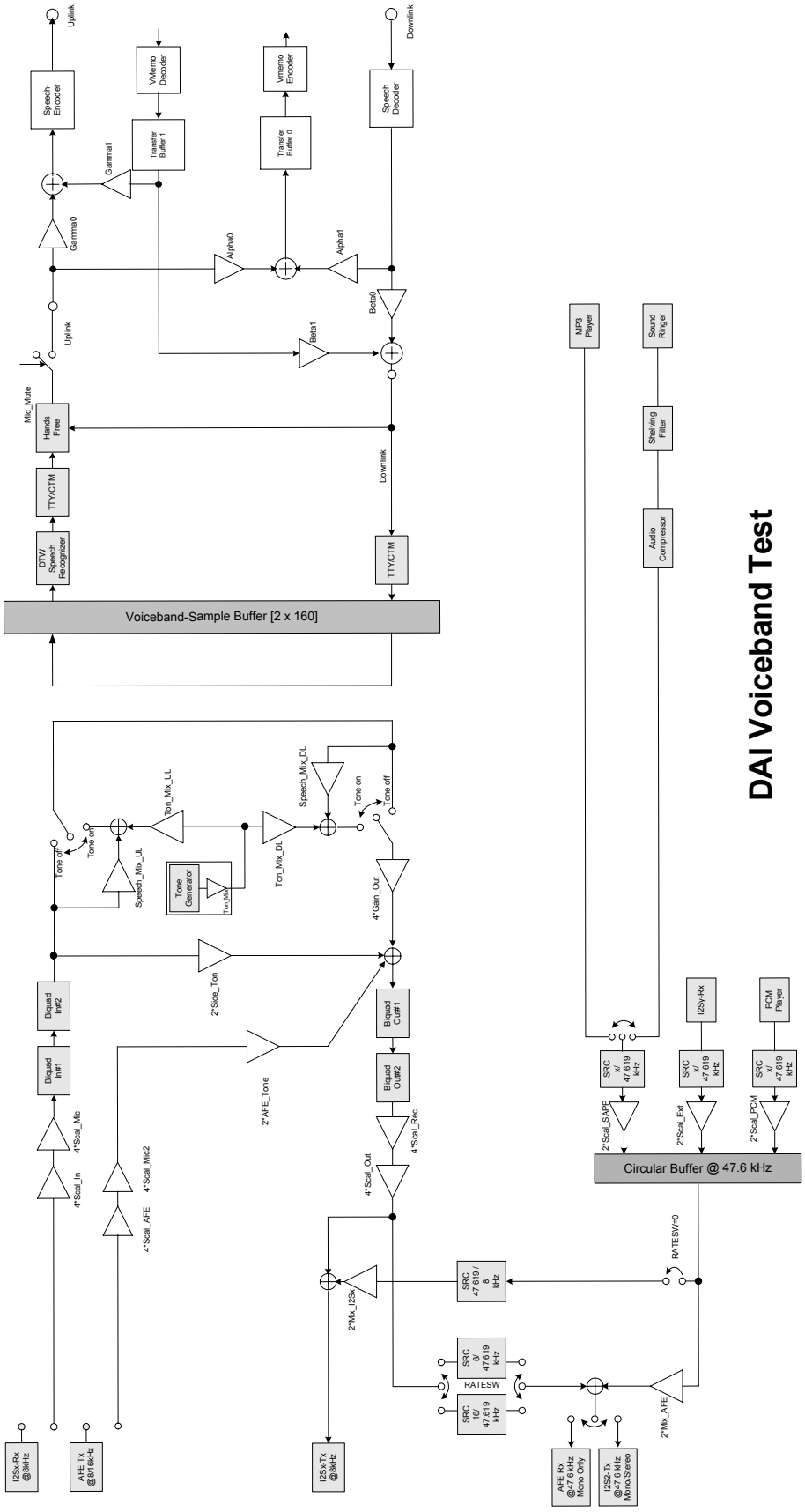


Figure 5-12 Voiceband Test



6 UMTS Audio Interface

E-GOLDradio CONFIDENTIAL		
Revision History:	**2005-12-07**	Rev. 1.01
Previous Version:	Rev. 1.00, 2005-05-16	
Page	Subjects (major changes since last revision)	
	Initial Version based on *E-GOLDradio G14 Firmware Manual*	
Changes for Rev. 1.01		

This chapter gives information about the UMTS-scheduler.

6.1 System Overview

The UMTS mode can be activated by the MCU sending the **UMTS_ON** command to the DSP subsystem. **UMTS_ON** only activate the UMTS mode. The control information and the UL/DL speech data is made available for the DSP by the MCU via the shared memory. To enter the UMTS mode following points have to be considered:

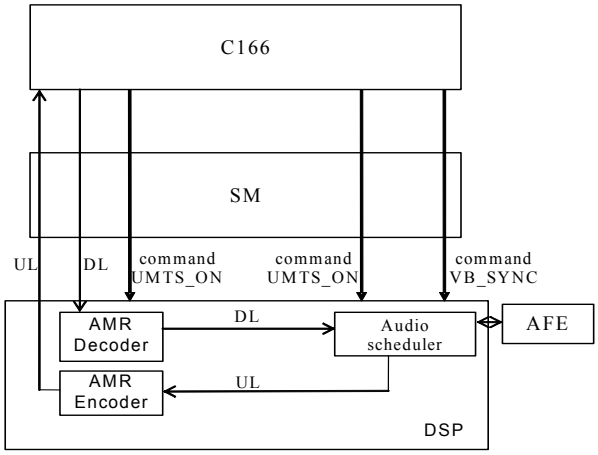
- The transition to the UMTS mode can only be done from the IDLE mode, that is, in the TCH26 or PDCH mode the MCU must send an **IDLE** command before sending **UMTS_ON**.
- The information needed by the DSP in UMTS mode has to be provided in time by the MCU in the shared memory locations **SM_UMTS_UL_CTRL**, **SM_UMTS_UL_DATA** and **SM_UMTS_DL_DATA** (refer to **Section 7.1 "Contents of Shared Memory" on Page 149**). These addresses need to be initialized before the **UMTS_ON** is given.
- The Audio Scheduler can be switched on either before or after **UMTS_ON** is sent.
- The Audio Scheduler cannot be switched off while in the UMTS mode.
- It is allowed to switch between AFE, I^2S_x and I^2S_x+AFE output during the UMTS mode.
- The **VB_SYNC** command cannot be sent before the Audio Scheduler is enabled (via **VB_ON**).

The UMTS mode does not require a special speech CODEC implementation on the DSP, since the speech CODEC defined for UMTS is based on the GSM AMR speech CODECs. The speech processing in the DSP is controlled by the MCU. **Figure 6-1** shows the data flow in UMTS mode.

To start speech processing in the UMTS mode

1. Send **UMTS_ON** to the DSP.
2. Synchronize the voiceband by sending **VB_SYNC** to the DSP.
3. Check and maintain the voiceband synchronization by sending **VB_SYNC** to the DSP at regular intervals.

Figure 6-1 Data Flow Diagram



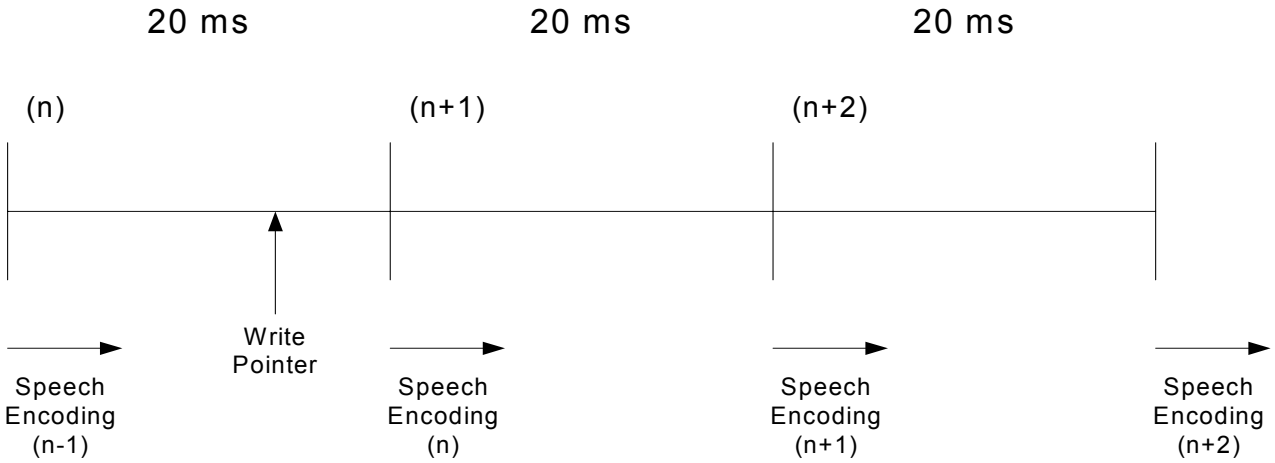
6.2 UMTS Uplink

Speech encoding starts at the frame boundary as shown in **Figure 6-2**. The samples from the AFE hardware are passed to the sample-based processing block. Once 160 samples are collected after the sample-based processing the speech encoding is done. Hence, the speech encoding is done at the frame boundary. The data rate to be used for encoding is specified by the MCU in **SM_UMTS_UL_CTRL** at the frame boundary. After encoding the output data is written to **SM_UMTS_UL_DATA** according to the data formats described in **Chapter 5**. The data format selected depends on the frame type. The encoded data must be available **SM_UMTS_UL_DATA** about 5 ms (worst case) after the frame boundary.

*Note: The frame boundary and the point when the speech encoder starts is defined by CPU [**VB_SYNC**].*

Note: If the speech decoder is running when speech encoder has to start, the speech encoder can only start after the completion of speech decoder. the speech encoder itself takes about 4 ms in the worst case. A safety margin of 1 ms is included considering the runtime for speech decoder. So, the anticipated time for the encoded data to be available in the shared memory is a total of 5 ms for the worst case.

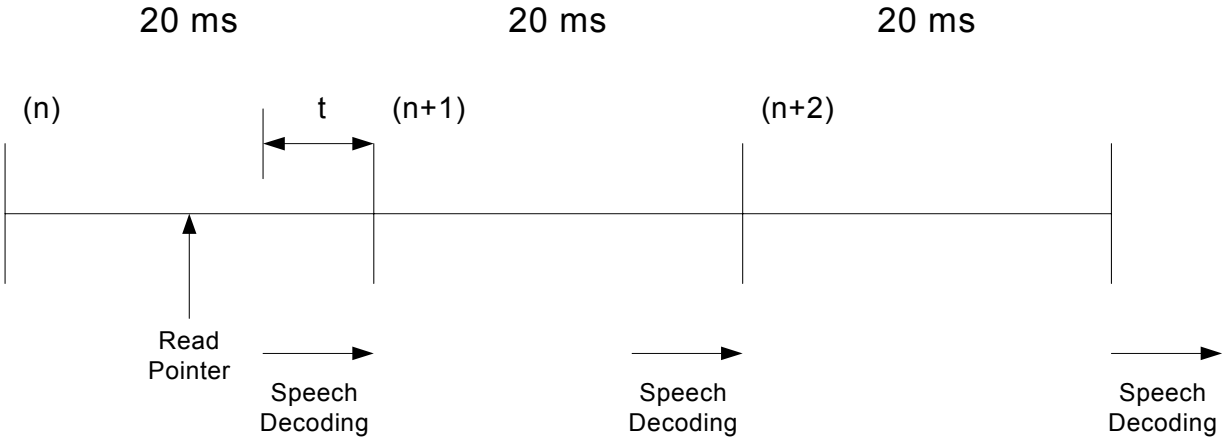
Figure 6-2 Timing of Speech Encoding



6.3 UMTS Downlink

Speech decoding starts 5 ms before the frame boundary. If the speech encoding is running when speech decoding is to start, the decoder waits until the encoding is completed. When the speech decoding starts 160 samples from the decoder output are available to the sample-based processing block. The samples are then passed to the hardware block. The samples must be ready at the frame boundary. Hence, the speech decoding must start 5 ms before the frame boundary. The speech decoder itself needs 1 ms in the worst case for the runtime. So, the MCU has to guarantee that the data is available in **SM_UMTS_DL_DATA** 5 ms before the frame boundary.

Figure 6-3 Timing of the Speech Decoding



Note: The frame boundary and when the speech decoder starts is defined by MCU in **VB_SYNC**.

Note: If it is guaranteed from the voiceband timing controlled by the MCU that the encoder is not running while speech decoder is requested by the DSP than 5 ms can be reduced to 1ms. i.e., the 4 ms safety margin can be excluded only if the caller can make sure that a processing overlap does not occur.

Note: If MMS is activated, the voicememo is started after speech decoder is completed. So, it takes 10 ms in the worst case for the voicememo to be completed.

6.4 Voiceband Synchronization

The Voiceband Synchronization uses two 160-samples buffers, which represents 20 ms with a sampling rate of 8 kHz. There us a read buffer and a write buffer. The encoder starts at the end of the 160-sample buffer for the uplink. The decoder starts at 160th - 8 sample for the downlink. **VB_SYNC** sets the read- and write-pointers to the correct start addresses to ensure the synchronization as described below. **VB_SYNC** also checks the right positions of the pointers mentioned before.

6.4.1 First Synchronization

After issuing the first **VB_SYNC** command with the parameter SYNC = 1, the first synchronization is done by the DSP. Immediately after the frame boundary, the speech encoder/decoder (depending on whether the synchronization is required in the uplink or downlink direction) is started. The write or read pointer can have any value in the range [0,159], the pointer is directly set to the specific value given by **VB_SYNC**.

6.4.2 Re-synchronization

When the speech processing is started, **VB_SYNC** (SYNC = 2) is sent to the DSP at regular intervals to check and maintain the voiceband synchronization.

If the drift between the positions of the actual pointer and the expected read or write pointer in PTR_VAL parameter is greater than the threshold defined by the parameter SYNC_LIMIT, the re synchronization is to done:

- In the downlink direction if

  |Read_pointer - PTR_VAL| > SYNC_LIMIT, the read pointer is set to PTR_VAL value.
- In the uplink direction if

  |Write_pointer - PTR_VAL| > SYNC_LIMIT, the write pointer is set to PTR_VAL value.

The range of the parameter PTR_VAL in the case of re-synchronization is limited and cannot have the entire range of [0,159] as in the case of first synchronization.

6.4.3 Parameter Range

The range of the parameters PTR_VAL and SYNC_LIM of command **VB_SYNC** depends on the synchronization type (first or re-synchronization) and direction (uplink or downlink).

The parameter PTR_VAL depends on both factors as described in **Table 6-1**:

Table 6-1 Range of Parameter PTR_VAL

SWITCH	SYNC	Range of PTR_VAL
1	1	[0,159]
	2	[SYNC_LIMIT+5,119- SYNC_LIMIT-5]
2	1	[0,159]
	2	[SYNC_LIMIT+5,159- SYNC_LIMIT-5]

The parameter SYNC_LIM is relevant only for the re-synchronization mode and depends only on SWITCH as described in Table 6-2:

Table 6-2 Range of Parameter SYNC_LIM

SWITCH	Range of SYNC_LIM
1	< 55
2	< 75

Note: Latency of the DSP until the command is processed can be <= 100 ms. This means, when the command is given to the DSP, in worst case it might be that the synchronization is checked up to 100 ms later.

6.5 Data Interface Format

Table 6-4 gives the number of words for UMTS AMR Frames for all frame types. The total number of words includes one byte for the frame header. Although most of AMR encoded speech frames are shorter than 16 words, it is assumed that the receiving buffer is always filled with 16 words. the padding bits should be set to zero. The DSP always reads 16 words from this buffer, decodes the frame type, and decides the number of words of the buffer that are valid. The same holds good for the uplink.

SM_UMTS_UL_CTRL is size of one 16-bit word and contains the information about the data rate to be used for encoding. The valid values for the data rate are from zero up to seven as shown in Table 6-3.

Table 6-3 UMTS_UL_CTRL

Data Rate	SM_UMTS_UL_CTRL
4.75	0
5.15	1
5.90	2
6.15	3
7.40	4
7.95	5
10.20	6
12.20	7

SM_UMTS_DL_DATA has the same format as **SM_UMTS_UL_DATA**.

6.5.1 Frame Types

Table 6-4 Speech CODEC Words for Various Frame Types

Frame Type Index	Frame Content	Coefficient Number	Bit Number	Number of Octets	Number of Words in Shared Memory
7	12.2	57	244	32	16
6	10.2	39	204	27	14
5	7.95	23	159	21	11
4	7.4	19	148	20	10
3	6.7	19	134	18	9
2	5.9	19	118	16	8
1	5.15	19	103	14	7
0	4.75	17	95	13	7
8	SID	5	39	5	3
15	No Data	0	0	1 (frame Header)	1 (frame Header)

6.5.2 Frame Header

The data interface format of the AMR encoded speech frames is the concatenation of the frame header and the AMR core frame format plus necessary stuffing bits to achieve octet alignment. The first octet of a frame defines the frame header as described by Table 6-5.

Table 6-5 Description of Frame Header

Bit Number	7	6	5	4	3	2	1	0
Content	P	FT(3)	FT(2)	FT(1)	FT(0)	Q	P	P

- Bits 0, 1 and 7 are with padding bits (that is, zeros).
- The *Q* bit (bit 2) is a signal flag for the payload quality:
  0: Payload is severely damaged (receiver should set the RX frame type to 'SPEECH_BAD' or 'SID_BAD' depending on the Frame Type)
  1: Payload is valid.
- Bits 3 to 6 define the Frame Type Field.

Table 6-6 Description of Frame Type Field in AMR Header

Frame Type Index	FT(3)	FT(2)	FT(1)	FT(0)	Frame Content
0	0	0	0	0	4.75 kbit/s
1	0	0	0	1	5.15 kbit/s
2	0	0	1	0	5.9 kbit/s
3	0	0	1	1	6.7 kbit/s
4	0	1	0	0	7.4 kbit/s
5	0	1	0	1	7.95 kbit/s
6	0	1	1	0	10.2 kbit/s
7	0	1	1	1	12.2 kbit/s
8	1	0	0	0	AMR Comfort Noise
15	1	1	1	1	No Data

6.5.3 Storage Format

The following bits form the AMR core format, which are the sequence of class A, class B and finally class C bits. As the format is octet aligned, some stuffing bits are needed to fill up the last octet.
Table 6-6 to Table 6-15 show the bit-, octet- and word sequence in the buffers for various frame_types:

- The number of padding bits depends on the Frame_Type_Index (see Table 6-6). Possible values for bit Q for Frame_Type_Index 0...7:
  - Q-bit uplink: Always 1
  - Q-bit downlink: 1 (speech good) or 0 (Speech bad)
- Possible values for bit Q and STI (d(35)) for Frame_Type_Index 8:
  - Q-bit uplink: Always 1
  - Q-bit downlink: 1 (SID First, SID Update) or 0 (SID bad)
  - STI (d(35))-bit uplink: 1 (SID update) or 0 (SID first)
  - STI (d(35))-bit downlink: 1 (SID update, SID bad) or 0 (SID first)
- Possible values for bit Q for Frame_Type_Index 15 (no date):
  - Q-bit uplink: Always 1
  - Q-bit downlink: Always 1.

Table 6-7 Bit Alignment in Shared Memory for AMR Mode 4.75 kbit/s

Word/Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
15	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
...	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
7	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
6	P	P	P	P	P	P	P	P	d(88)	...	...	...	...	...	d(94)	P
5	d(80)	...	...	...	...	...	...	d(87)	d(72)	...	...	...	...	...	...	d(79)
4	d(64)	...	...	...	...	...	...	d(71)	d(56)	...	...	...	...	...	...	d(63)
3	d(48)	...	...	...	...	...	...	d(55)	d(40)	...	...	...	...	...	...	d(47)
2	d(32)	...	...	...	...	...	...	d(39)	d(24)	...	...	...	...	...	...	d(31)
1	d(16)	...	...	...	...	...	...	d(23)	d(8)	...	...	...	...	...	...	d(15)
0	d(0)	d(1)	d(2)	d(3)	d(4)	d(5)	d(6)	d(7)	P	0	0	0	0	Q	P	P

Table 6-8 Bit Alignment in Shared Memory for AMR Mode 5.15 kbit/s

Word/Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
15	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
...	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
7	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
6	d(96)	...	...	...	...	...	d(102)	P	d(88)	...	...	...	...	...	...	d(95)
5	d(80)	...	...	...	...	...	...	d(87)	d(72)	...	...	...	...	...	...	d(79)
4	d(64)	...	...	...	...	...	...	d(71)	d(56)	...	...	...	...	...	...	d(63)
3	d(48)	...	...	...	...	...	...	d(55)	d(40)	...	...	...	...	...	...	d(47)
2	d(32)	...	...	...	...	...	...	d(39)	d(24)	...	...	...	...	...	...	d(31)
1	d(16)	...	...	...	...	...	...	d(23)	d(8)	...	...	...	...	...	...	d(15)
0	d(0)	d(1)	d(2)	d(3)	d(4)	d(5)	d(6)	d(7)	P	0	0	0	1	Q	P	P

Table 6-9 Bit Alignment in Shared Memory for AMR Mode 5.9 kbit/s

Table 6-10

Word/Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
15	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
...	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
8	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
7	d(112)	...	...	...	...	d(117)	P	P	d(104)	...	...	...	...	...	...	d(111)
6	d(96)	...	...	...	...	...	...	d(103)	d(88)	...	...	...	...	...	...	d(95)
5	d(80)	...	...	...	...	...	...	d(87)	d(72)	...	...	...	...	...	...	d(79)
4	d(64)	...	...	...	...	...	...	d(71)	d(56)	...	...	...	...	...	...	d(63)
3	d(48)	...	...	...	...	...	...	d(55)	d(40)	...	...	...	...	...	...	d(47)
2	d(32)	...	...	...	...	...	...	d(39)	d(24)	...	...	...	...	...	...	d(31)
1	d(16)	...	...	...	...	...	...	d(23)	d(8)	...	...	...	...	...	...	d(15)
0	d(0)	d(1)	d(2)	d(3)	d(4)	d(5)	d(6)	d(7)	P	0	0	1	0	Q	P	P

Table 6-11 Bit Alignment in Shared Memory for AMR Mode 6.7 kbit/s

Word/Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
15	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
...	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
9	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
8	d(128)	...	...	...	...	d(133)	P	P	d(120)	...	...	...	...	...	...	d(127)
7	d(112)	...	...	...	...	...	...	d(119)	d(104)	...	...	...	...	...	...	d(111)
6	d(96)	...	...	...	...	...	...	d(103)	d(88)	...	...	...	...	...	...	d(95)
5	d(80)	...	...	...	...	...	...	d(87)	d(72)	...	...	...	...	...	...	d(79)
4	d(64)	...	...	...	...	...	...	d(71)	d(56)	...	...	...	...	...	...	d(63)
3	d(48)	...	...	...	...	...	...	d(55)	d(40)	...	...	...	...	...	...	d(47)
2	d(32)	...	...	...	...	...	...	d(39)	d(24)	...	...	...	...	...	...	d(31)
1	d(16)	...	...	...	...	...	...	d(23)	d(8)	...	...	...	...	...	...	d(15)
0	d(0)	d(1)	d(2)	d(3)	d(4)	d(5)	d(6)	d(7)	P	0	0	1	1	Q	P	P

Table 6-12 Bit Alignment in Shared Memory for AMR Mode 7.4 kbit/s

Word/Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
15	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
...	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
10	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
9	d(144)	...	...	d(147)	P	P	P	P	d(136)	...	...	...	...	...	...	d(143)
8	d(128)	...	...	...	...	...	...	d(135)	d(120)	...	...	...	...	...	...	d(127)
7	d(112)	...	...	...	...	...	...	d(119)	d(104)	...	...	...	...	...	...	d(111)
6	d(96)	...	...	...	...	...	...	d(103)	d(88)	...	...	...	...	...	...	d(95)
5	d(80)	...	...	...	...	...	...	d(87)	d(72)	...	...	...	...	...	...	d(79)
4	d(64)	...	...	...	...	...	...	d(71)	d(56)	...	...	...	...	...	...	d(63)
3	d(48)	...	...	...	...	...	...	d(55)	d(40)	...	...	...	...	...	...	d(47)
2	d(32)	...	...	...	...	...	...	d(39)	d(24)	...	...	...	...	...	...	d(31)
1	d(16)	...	...	...	...	...	...	d(23)	d(8)	...	...	...	...	...	...	d(15)
0	d(0)	d(1)	d(2)	d(3)	d(4)	d(5)	d(6)	d(7)	P	0	1	0	0	Q	P	P

Table 6-13 Bit Alignment in Shared Memory for AMR Mode 7.95 kbit/s

Word/Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
15	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
...	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
11	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
10	P	P	P	P	P	P	P	P	d(152)	...	...	...	...	...	d(158)	P
9	d(144)	...	...	...	...	...	...	d(151)	d(136)	...	...	...	...	...	...	d(143)
8	d(128)	...	...	...	...	...	...	d(135)	d(120)	...	...	...	...	...	...	d(127)
7	d(112)	...	...	...	...	...	...	d(119)	d(104)	...	...	...	...	...	...	d(111)
6	d(96)	...	...	...	...	...	...	d(103)	d(88)	...	...	...	...	...	...	d(95)
5	d(80)	...	...	...	...	...	...	d(87)	d(72)	...	...	...	...	...	...	d(79)
4	d(64)	...	...	...	...	...	...	d(71)	d(56)	...	...	...	...	...	...	d(63)
3	d(48)	...	...	...	...	...	...	d(55)	d(40)	...	...	...	...	...	...	d(47)
2	d(32)	...	...	...	...	...	...	d(39)	d(24)	...	...	...	...	...	...	d(31)
1	d(16)	...	...	...	...	...	...	d(23)	d(8)	...	...	...	...	...	...	d(15)
0	d(0)	d(1)	d(2)	d(3)	d(4)	d(5)	d(6)	d(7)	P	0	1	0	1	Q	P	P

Table 6-14 Bit Alignment in Shared Memory for AMR Mode 10.2 kbit/s

Word/Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
15	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
14	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
13	P	P	P	P	P	P	P	P	d(200)	...	...	d(203)	P	P	P	P
12	d(192)	...	...	...	...	...	...	d(199)	d(184)	...	...	...	...	...	...	d(191)
11	d(176)	...	...	...	...	...	...	d(183)	d(168)	...	...	...	...	...	...	d(175)
10	d(160)	...	...	...	...	...	...	d(167)	d(152)	...	...	...	...	...	...	d(159)
9	d(144)	...	...	...	...	...	...	d(151)	d(136)	...	...	...	...	...	...	d(143)
8	d(128)	...	...	...	...	...	...	d(135)	d(120)	...	...	...	...	...	...	d(127)
7	d(112)	...	...	...	...	...	...	d(119)	d(104)	...	...	...	...	...	...	d(111)
6	d(96)	...	...	...	...	...	...	d(103)	d(88)	...	...	...	...	...	...	d(95)
5	d(80)	...	...	...	...	...	...	d(87)	d(72)	...	...	...	...	...	...	d(79)
4	d(64)	...	...	...	...	...	...	d(71)	d(56)	...	...	...	...	...	...	d(63)
3	d(48)	...	...	...	...	...	...	d(55)	d(40)	...	...	...	...	...	...	d(47)
2	d(32)	...	...	...	...	...	...	d(39)	d(24)	...	...	...	...	...	...	d(31)
1	d(16)	...	...	...	...	...	...	d(23)	d(8)	...	...	...	...	...	...	d(15)
0	d(0)	d(1)	d(2)	d(3)	d(4)	d(5)	d(6)	d(7)	P	0	1	1	0	Q	P	P

Table 6-15 Bit Alignment in Shared Memory for AMR Mode 12.2 kbit/s

Word/Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
15	d(240)	...	...	d(243)	P	P	P	P	d(232)	...	...	...	...	...	...	d(239)
14	d(224)	...	...	...	...	...	...	d(231)	d(214)	...	...	...	...	...	...	d(223)
13	d(208)	...	...	...	...	...	...	d(215)	d(200)	...	...	...	...	...	...	d(207)
12	d(192)	...	...	...	...	...	...	d(199)	d(184)	...	...	...	...	...	...	d(191)
11	d(176)	...	...	...	...	...	...	d(183)	d(168)	...	...	...	...	...	...	d(175)
10	d(160)	...	...	...	...	...	...	d(167)	d(152)	...	...	...	...	...	...	d(159)
9	d(144)	...	...	...	...	...	...	d(151)	d(136)	...	...	...	...	...	...	d(143)
8	d(128)	...	...	...	...	...	...	d(135)	d(120)	...	...	...	...	...	...	d(127)
7	d(112)	...	...	...	...	...	...	d(119)	d(104)	...	...	...	...	...	...	d(111)
6	d(96)	...	...	...	...	...	...	d(103)	d(88)	...	...	...	...	...	...	d(95)
5	d(80)	...	...	...	...	...	...	d(87)	d(72)	...	...	...	...	...	...	d(79)
4	d(64)	...	...	...	...	...	...	d(71)	d(56)	...	...	...	...	...	...	d(63)
3	d(48)	...	...	...	...	...	...	d(55)	d(40)	...	...	...	...	...	...	d(47)
2	d(32)	...	...	...	...	...	...	d(39)	d(24)	...	...	...	...	...	...	d(31)
1	d(16)	...	...	...	...	...	...	d(23)	d(8)	...	...	...	...	...	...	d(15)
0	d(0)	d(1)	d(2)	d(3)	d(4)	d(5)	d(6)	d(7)	P	0	1	1	0	Q	P	P

Table 6-16 Bit Alignment in Shared Memory for SID Mode

Word/Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
15	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
...	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
3	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
2	d(32)	...	...	d(35)	mi(0)	mi(1)	mi(2)	P	d(24)	...	...	...	...	...	...	d(31)
1	d(16)	...	...	...	...	...	...	d(23)	d(8)	...	...	...	...	...	...	d(15)
0	d(0)	d(1)	d(2)	d(3)	d(4)	d(5)	d(6)	d(7)	P	1	0	0	0	Q	P	P

Note: (MI(0), M(1) and MI(2): d(36), d(37) and d(38))

Mirrored Mode Indication: AMR CODEC mode according to the first eight entries in **Table 6-4**. The detailed description of the minored mode indication can be found in **Table 6-17**.

Table 6-17 Mirrored CODEC Mode

CODEC Mode	Frame Content	Mirrored CODEC Mode
0	4.75 kbit/s	b000
1	5.15 kbit/s	b100
2	5.9 kbit/s	b010
3	6.7 kbit/s	b110
4	7.4 kbit/s	b001
5	7.95 kbit/s	b101
6	10.2 kbit/s	b011
7	12.2 kbit/s	b111

Table 6-18 Bit Alignment in Shared Memory for NO_DATA Mode

Word/Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
15	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
...	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
0	P	P	P	P	P	P	P	P	P	1	1	1	1	Q	P	P

7 Shared Memory

E-GOLDradio
CONFIDENTIAL

Revision History:	**2005-12-07**	Rev. 1.01
Previous Version:	Rev. 1.00, 2005-05-16	
Page	Subjects (major changes since last revision)	
	Initial Version based on *E-GOLDradio G14 Firmware Manual*	
Changes for Rev. 1.00		
Page 149	Update **Table 7-1**	
Changes for Rev. 1.01		

7.1 Contents of Shared Memory

Table 7-1 Shared Memory

SM_Name	Offset	Size
** Firmware Version		
SM_FW_VERSION	0	1
** Hardware Version		
SM_HW_VERSION	1	1
** OCEM Status		
SM_OCEM_STATUS_1	2	1
** Boot Data		
SM_BOOT_DATA	2	512
** TDMA Counters		
SM_COUNTER_104	2	1
SM_COUNTER_51	3	1
SM_SFNUM	4	1
** MCUSM_ Commands		
SM_MCU_CMD_0	5	28
SM_MCU_CMD_1	33	28
SM_MCU_CMD_2	61	28
** FCB-Detection Output		
SM_FC_STATUS	89	1
SM_FC_START	90	1
SM_FC_QUAL	91	1
SM_FC_RMS	92	1
SM_FC_FREQ	93	1

Table 7-1 Shared Memory

SM_Name	Offset	Size
** BB-Filter Output		
SM_BB_IF_FLAG	94	4
** Sync-Burst Output		
SM_SYNC_EQU	98	14
SM_SYNC_METRIC	112	1
SM_SYNC_STATUS	113	1
SM_SYNC_DATA	114	2
SM_SYNC_EQ_EOTD_POS	116	1
SM_SYNC_EQ_EOTD_CORR	117	18
** NB-Equalizer Output		
SM_EQUAL_0	135	14
SM_EQUAL_1	149	14
SM_EQUAL_2	163	14
SM_EQUAL_3	177	14
** Monitoring Output		
SM_MON_INDEX	191	1
SM_MON_VALS	192	8
** Rach Control		
SM_RACH_FLAG	200	1
SM_RACH_TSC	201	1
SM_RACH_TIM_ADV	202	1
SM_RACH_DATUM	203	1
SM_RACH_BSIC	204	1
** Dedicated Control-Channels		
SM_SDCCH_TX_DATA	205	12
SM_SDCCH_RX_DATA	217	14
** Tone Parameters		
SM_TONE_FREQ_1	231	1
SM_TONE_AMP_1	232	1
SM_TONE_FREQ_2	233	1
SM_TONE_AMP_2	234	1
SM_TONE_FREQ_3	235	1
SM_TONE_AMP_3	236	1
SM_TONE_DUR_IN	237	1
SM_TONE_DUR_OUT	238	1

Table 7-1 Shared Memory

SM_Name	Offset	Size
SM_TONE_DUR_INTER	239	1
SM_TONE_FADIN_DUR	240	1
SM_TONE_FADOUT_DUR	241	1
** Voice-Memo Arrays		
SM_ADPCM_ENC	242	44
SM_VM_BUFFER_0	242	17
SM_VM_BUFFER_1	259	17
Circuit Switched Data (Overlay)		
** Control Channels Uplink		
SM_FACCH_TX_FLAG	286	1
SM_FACCH_TX_DATA	287	12
SM_SACCH_TX_DATA_0	299	12
SM_SACCH_TX_DATA_1	311	12
** Control Channels Downlink		
SM_FACCH_RX_FLAG	323	1
SM_FACCH_RX_DATA	324	14
SM_SACCH_RX_DATA_0	338	14
SM_SACCH_RX_DATA_1	352	14
SM_SACCH_RX_DATA_2	366	14
SM_SACCH_RX_DATA_3	380	14
** Speech Channel Control		
SM_TCH_METRIC	394	1
SM_TCH_STATUS	395	1
SM_DTX_FLAG	396	1
SM_DTX_USED	397	1
** Adaptive Multi Rate (AMR) Tx-Data		
SM_AMR_ACS_UL	398	1
SM_AMR_MI_UL	399	1
SM_AMR_MR_UL	400	1
SM_AMR_TX_TYPE	401	1
SM_RATSCCH_TX_FLAG	402	1
SM_RATSCCH_TX_DATA	403	3
** Adaptive Multi Rate (AMR) Rx-Data		
SM_AMR_ACS_DL	406	1
SM_AMR_MI_DL	407	1

Table 7-1 Shared Memory

SM_Name	Offset	Size
SM_AMR_MC_DL	408	1
SM_AMR_MI_EVEN	409	1
SM_AMR_RX_TYPE	410	1
SM_RATSCCH_RX_FLAG	411	1
SM_RATSCCH_RX_DATA	412	5
SM_AMR_RX_DATA	417	18
** Data Channel Uplink		
SM_TCH_TX_DATA_0	435	23
SM_TCH_TX_DATA_1	458	23
** Data Channel Downlink		
SM_TCH_RX_DATA_0	481	24
SM_TCH_RX_DATA_1	505	24
SM_TCH_RX_DATA_2	529	24
SM_TCH_RX_DATA_3	553	24
** DRX Flag for AMR		
SM_AMR_DRX_FLAG	580	1
Packet Switched (Overlay)		
** Rx Info during Frame-Interrupt		
SM_RX_INFO	286	4
SM_RX_TEMP_TSC	290	1
** Tx Info for CODON Interrupt		
SM_TX_INFO	291	4
** Timing Advance PTCCH Downlink		
SM_PTCCH_RX_DATA	295	14
** USF Detection Result		
SM_USF_RESULT_0	309	4
SM_USF_RESULT_1	313	4
SM_USF_RESULT_2	317	4
SM_USF_RESULT_3	321	4
** Packet Data Channel Uplink		
SM_PDTCH_TX_DATA_0	325	28
SM_PDTCH_TX_DATA_1	353	28
SM_PDTCH_TX_DATA_2	381	28

Table 7-1 Shared Memory

SM_Name	Offset	Size
SM_PDTCH_TX_DATA_3	409	28
** Packet Data Channel Downlink		
SM_PDTCH_RX_DATA_0	437	30
SM_PDTCH_RX_DATA_1	467	30
SM_PDTCH_RX_DATA_2	497	30
SM_PDTCH_RX_DATA_3	527	30
**Interrupts for PDCH mode		
SM_PDCH_USF_INT	561	4
SM_PDCH_DEC_INT	565	4
Common		
** IO-Transfer-Buffer		
SM_IO_TRANSFER	581	576
**UMTS Mode		
SM_UMTS_UL_CTRL	581	1
SM_UMTS_UL_DATA	582	16
SM_UMTS_DL_DATA	598	16
** MP3-/Soundringerframes		
SM_MP3_PAR_OUT	614	3
SM_SYNTH_PAR_OUT	614	1
SM_SYNTH_RMS_LOW	615	1
SM_SYNTH_RMS_HIGH	616	1
** Voice Memo for PCM Data		
SM_VM_PCM_BUFFER_0	617	160
SM_VM_PCM_BUFFER_1	777	600
** DTW Speech Recognizer		
SM_DTW_PAR	937	460
** Startup-Code Version		
SM_STARTUP_CODE_VERSION	1535	1
** Stand By Power Down Mode		
SM_SBPD_INFO	96	90
SM_SBPD_BOOT_ADD	186	1334
RESERVED_NOT_USABLE	1520	16

8 Run Times Of DSP Algorithms

E-GOLDradio CONFIDENTIAL		
Revision History:	**2005-12-07**	Rev. 1.01
Previous Version:	Rev. 1.00, 2005-05-16	
Page	Subjects (major changes since last revision)	
	Initial Version based on *E-GOLDlite Firmware Manual*	
Changes for Rev. 1.02		

8.1 Run Times Of DSP Algorithms

In the following tables the number of clocks required for most procedures running on E-GOLDradio are given. For these tables interrupts and scheduler functions have not been taken into account. As a result, the voiceband interrupts and baseband receive processing lengthens the run times. For the speech CODECs the worst case numbers found in the ETSI test vectors are used.

Table 8-1 Equalizer and FCB-Search

Function	Number of Clocks
Normal Burst Equalizer (GMSK)	< 46194+8000 = 54194
Sync Burst Equalizer	< 47758+8000 = 55758
FCB Search (150 IQ-pairs)	4832
FCB Evaluation	31133

Table 8-2 Channel CODECs

Function	Encoder	Decoder
Speech		
TCH/ EFR	12153	37253
TCH/ FR	11979	36783
TCH/ HR	8154	23710
Control		
FACCH/ HR, FR	12350	27277
RACH	1110	-
PRACH	1590	-
SCH	-	13559
Data Services		
TCH/ F14,4	12642	24246
TCH/ F9,6	11557	21138
TCH/ F4,8	10615	18022
TCH/ F2,4	9449	15014
TCH/ H4,8	11304	21098
TCH/ H2,4	10275	17594
GPRS		

Table 8-2 Channel CODECs

Function	Encoder	Decoder
CS1	12362	23918
CS2	15252	27756
CS3	16150	30437
CS4	11483	22287
USF6	-	6936
USF12	-	1185
AMR fullrate speech		
AFS 12,2	16699	40901
AFS 10,2	19524	42586
AFS 7,95	16017	41441
AFS 7,40	14914	35329
AFS 6,70	15816	36263
AFS 5,90	14514	37819
AFS 5,15	14530	33679
AFS 4,75	13790	35594
AMR fullrate signalling		
Sid Update	11724	10740
Sid First	20945	-
Ratscch	9830	12382
Onset	4469	-
Sid First	4468	-
Afs Marker Check RATSCCH	-	20883
Afs Marker Check SID Update	-	19244
Afs Marker Check SID First	-	20945
Afs Marker Check Onset	-	20735
AMR halfrate speech		
AHS 7,95	10865	21725
AHS 7,40	10441	21400
AHS 6,70	9773	20271
AHS 5,90	9160	19517
AHS 5,15	10659	20909
AHS 4,75	10085	22537
AMR halfrate signalling		
Sid Update	11594	9175
Sid Update Inhibit	2898	2898
Sid First P1	5023	-
Sid First P2	2690	-
Sid First Inhibit	2898	-
Onset	2691	-
Ratscch Marker	4991	-
Ratscch Data	6614	9022

Table 8-2 Channel CODECs

Function	Encoder	Decoder
Ahs Marker Check SID First P1	-	13229
Ahs Marker Check SID First P2	-	12361
Ahs Marker Check SID First Inhibit	-	12221
Ahs Marker Check Onset	-	13665
Ahs Marker Check RATSCCH Data	-	13229
Ahs Marker Check SID Update Inhibit	-	11969
Ahs Marker Check SID Update	-	12713

Table 8-3 Speech CODEC

Function	Encoder	Decoder
Fullrate Speech Encoder (+Vad)	52176	18321
Halfrate Speech Encoder (+Vad)	273340	42597
AMR 12,2 (Enc.: +DTX, Dec.: non DTX)	346946	48299
AMR 10,2 (Enc.: +DTX, Dec.: non DTX)	327270	48273
AMR 7,95 (Enc.: +DTX, Dec.: non DTX)	335329	53004
AMR 7,4 (Enc.: +DTX, Dec.: non DTX)	319145	47396
AMR 6,7 (Enc.: +DTX, Dec.: non DTX)	346731	52543
AMR 5,9 (Enc.: +DTX, Dec.: non DTX)	270550	51643
AMR 5,15 (Enc.: +DTX, Dec.: non DTX)	218974	51605
AMR 4,75 (Enc.: +DTX, Dec.: non DTX)	274287	51452
AMR EFR	347016	48965

9 Document List and Glossary

The documents in **Table 9-1** are referred to in this document. They are either available in a separate attachments directory or are general standards.

Table 9-1 Document List

Number	Filename	Version	Date	Document Type
[1]	PMB 7870 Design Specification	Latest	-	Design Specification
[2]	ETSI Recommendations, namely 45.00x (not attached)	-	-	GSM Standard
[3]	TEAKLite_Overview_v1.0	1.0	-	TeakLite
[4]	UMTS Speech Interface Specification Infineon Technologies AG	1.0	2003-07-07	Firmware Concept
[5]	Interface Description for the High Frequency Shelving Filter (OAK/TeakLite-Assembler-Code Hi_shelving_asm00), , Siemens AG Austria	-	2003	Firmware Concept
[6]	AUDIO COMPRESSOR: Description of the Configuration Parameters for the Audio Compressor (DSP-ASM-C-Reference Code Compressor_c04, Compressor_asm03) and Interfaces of the OAK/TEAKLite-Implementation), , Siemens AG Austria	-	2004	Firmware Concept

The abbreviations in Table 9-2 are used in this document.

Table 9-2 Glossary Type Descriptions

Type	Description
ADPCM	Adaptive Differential Pulse Code Modulation
AMR	Adaptive Multi-Rate
BEP	Bit Error Probability
CC	Channel Codecs
CPU	Control Processing Unit
CRC	Cyclic Redundancy Check
CTM	Cellular Text telephone Modem
CV BEP	Coefficient of Variance of the Bit Error Probability
DAC	Digital Analog Converter
DAI	Digital Audio Interface
DMA	Direct Memory Access
DSP	Digital Signal Processor
DTW	Dynamic Time Warping
DTX	Discontinuous Transmission
EFR	Enhanced Full-Rate
EGPRS	Enhanced GPRS
E-OTD	Enhanced Observed Time Difference
EQ	Equalizer
FR	Full Rate
GEA	GPRS Encryption Algorithm
GMSK	Gauss Minimum Shift Keying
GPRS	General Packet Radio Service
GSM	Global System for Mobile communication
HR	Half Rate
HSCSD	High Speed Circuit Switched Data
I2S	Inter IC Sound
JTAG	Joint Test Action Group
LLC	Logical Link Control
MAC	Multiply Accumulate Module
MCS	Modulation and Coding Scheme
MCU	Micro Controller Unit
MP3	MPEG Audio Layer III
MS	Mobile Station
OCDS	On Chip Debug Support
PCB	Printed Circuit Board
PCM	Pulse Code Modulation
RBL	Radio Block
RLP	Radio Link Protocol
RTOS	Real-Time Operating System
SBPD	StandBy Power Down
SEIB	Serial Emulation Interface Block
SIM	Subscriber Identity Module
SM	Shared Memory

Table 9-2 Glossary Type Descriptions

Type	Description
SSC	Serial Synchronous Interface Controller
STM	System Timer Module
TDMA	Time Division Multiple Access
TEAKLite	DSP Core
TTY	TeleTYpewriters known also as a TDD (Telecommunications Device for the Deaf)
VB	Voice Band
VM	Voice Memo

Total Quality Management

Qualität hat für uns eine umfassende Bedeutung. Wir wollen allen Ihren Ansprüchen in der bestmöglichen Weise gerecht werden. Es geht uns also nicht nur um die Produktqualität – unsere Anstrengungen gelten gleichermaßen der Lieferqualität und Logistik, dem Service und Support sowie allen sonstigen Beratungs- und Betreuungsleistungen.

Dazu gehört eine bestimmte Geisteshaltung unserer Mitarbeiter. Total Quality im Denken und Handeln gegenüber Kollegen, Lieferanten und Ihnen, unserem Kunden. Unsere Leitlinie ist jede Aufgabe mit „Null Fehlern“ zu lösen – in offener Sichtweise auch über den eigenen Arbeitsplatz hinaus – und uns ständig zu verbessern.

Unternehmensweit orientieren wir uns dabei auch an „top“ (Time Optimized Processes), um Ihnen durch größere Schnelligkeit den entscheidenden Wettbewerbsvorsprung zu verschaffen.

Geben Sie uns die Chance, hohe Leistung durch umfassende Qualität zu beweisen.

Wir werden Sie überzeugen.

Quality takes on an allencompassing significance at Semiconductor Group. For us it means living up to each and every one of your demands in the best possible way. So we are not only concerned with product quality. We direct our efforts equally at quality of supply and logistics, service and support, as well as all the other ways in which we advise and attend to you.

Part of this is the very special attitude of our staff. Total Quality in thought and deed, towards co-workers, suppliers and you, our customer. Our guideline is “do everything with zero defects”, in an open manner that is demonstrated beyond your immediate workplace, and to constantly improve.

Throughout the corporation we also think in terms of Time Optimized Processes (top), greater speed on our part to give you that decisive competitive edge.

Give us the chance to prove the best of performance through the best of quality – you will be convinced.

www.infineon.com

Published by Infineon Technologies AG