BP30
FM Radio driver Specification

Edition 2006

Published by Neonseven s.r.l.,
Viale Stazione di Prosecco, 15

34010 Sgonico (Trieste) Italy

© Neonseven.
All Rights Reserved.

For questions on technology, delivery and prices please contact the Neonseven Offices in Italy Sgonico and Gorizia

Attention Please!

The information herein is given to describe certain components and shall not be considered as warranted 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.

Warnings

Due to technical requirements components may contain dangerous substances. For information on the types in question please contact Neonseven.

Neonseven technologies may only be used in life-support devices or systems with the express written approval of Neonseven, if a failure of such technologies 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.

Table of Contents

1 Document Mission/Scope

1.1 Mission

This document contains the specification of the FM Radio module, its SW interface and how it works.

1.2 Scope

This document is addressed to SW developers who want to include the FM radio module inside their SW.

2 List of Acronyms

Abbreviation / Term	Explanation / Definition
FM	Frequency Modulation
BP30	Basic Platform 30
SDL	Specification Description Language
ADC	Analog to Digital Converter

3 Introduction

This document describes in details the interface, which the FM radio driver offers to the general layer 1. The interface is a function interface. The description consists of three parts: MMI functions, Test functions and other functions. The parameters and the specific features in the document, depend on the specific chip radio used. For the BP30 platform the chip is TEA5761 UK from Philips that supports all the functionality describe in this document.

4 Interface description

In this chapter, there is a detailed description of all the interface functions. Driver consists of a main state machine with only two states: *idle* and *started*. The state machine and valid functions are illustrated in the figure below.

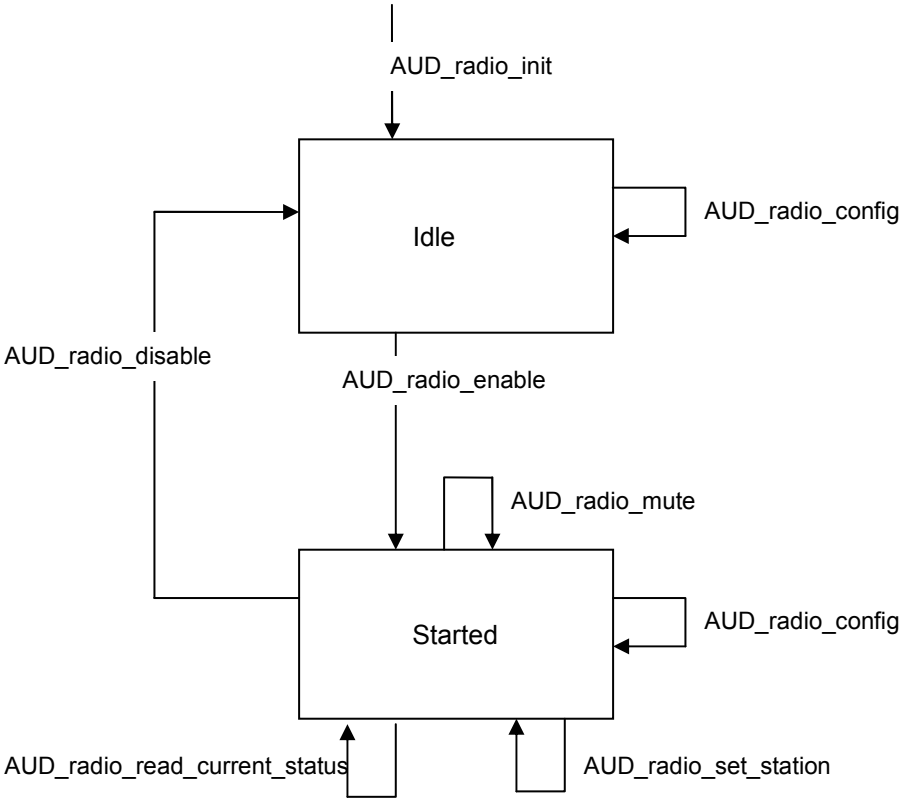

Figure 5-1 FM radio State Machine

4.1 Common interface functions

In this section is described the common used interface in the MS.

4.1.1 AUD_radio_init

Prototype: SINT8 AUD_radio_init (UINT8 handle)

Parameters: UINT8 handle

Returns: Return code, see Audio Interface documentation

Description: This function initializes the FM radio driver, during start up (at boot point). All hardware used by the FM radio driver is initialized in this function. All internal variables and state machines will be reset, when this function is called. Finally configuration parameters are loaded from none volatility memory.

4.1.2 AUD_radio_config

Prototype: SINT8 AUD_radio_config (UINT8 handle, AUD_radio_config_data_type* AUD_radio_config_data);

Parameters: UINT8 handle

AUD_radio_config_data_type* AUD_radio_config_data: Configuration parameters used to setup the radio. See table.

Returns: Return code, see Audio Interface documentation

Description: This function configures the FM radio driver. The configuration parameters to the radio are updated with this function. The configuration parameter settings is given by the *AUD_radio_config_data_type* struct. It is possible to only set some of the parameters in the struct by giving the rest of the parameters the value –1. The behavior of the function is decided of the state of the driver. If the state is *idle* is the parameters not updated on radio immediately but they are stored in ram and send to the radio when changing from *idle* state to *started* state. If the function is called in *started* state is the radio configured immediately. It's optional for the application to call the function. If not the function is called then the default values are used.

Struct: AUD_radio_config_data_type		
Name	Default	Description
SSL	Level_5	Search Stop Level, see table
HLSI	0	HIGH/LOW Side Injection. If HLSI = 1, then HIGH side LO injection. If HLSI = 0, then LOW side LO injection
MS	0	Mono/Stereo. If MS = 1, then forced mono. If MS = 0, then stereo ON
ML	0	Mute Left. If ML = 1, then left audio channel muted and forced mono. If ML = 0, then not muted
MR	0	Mute Right. If MR = 1, then right audio channel muted and forced mono. If MR = 0, then not muted
BL	0	Band Limits. If BL = 1, then Japan FM band. If BL = 0, then US/Europe FM band
HCC	0	High Cut Control. If HCC = 1, then high cut control is ON. If HCC = 0, then high cut control is OFF

Table 5-1 Data Configuration Radio

Enum: AUD_radio_search_stop_level_enum	
Enum name	Description
Not_allowed	not allowed in search mode
level_5	low; level ADC output = 5
level_7	mid; level ADC output = 7
level_10	high; level ADC output = 10

Table 5-2 Search level

Copyright (C) 2006NeonSeven S.R.L. All rights reserved
- Exclusive property of Infineon Technologies AG –

4.1.3 AUD_radio_enable

Prototype: SINT8 AUD_radio_enable (UINT8 handle);

Parameters: UINT8 handle

Returns: Return code, see Audio Interface documentation

Description: This function power on the FM radio and load the settings given by the AUD_radio_config function or the settings from last radio power up.

4.1.4 AUD_radio_disable

Prototype: SINT8 AUD_radio_disable (UINT8 handle);

Parameters: UINT8 handle

Returns: Return code, see Audio Interface documentation

Description: This function is used to power down the radio. The function stores the current settings in ram. The settings are used next time the function *AUD_radio_enable* is called. The settings are erased if the MS is turned off or if the function *AUD_radio_config* is called.

4.1.5 AUD_radio_mute

Prototype: SINT8 AUD_radio_mute (UINT8 handle, UINT8 mute)

Parameters: UINT8 handle

UINT8 mute: if 1 then L and R audio muted. If 0 then audio not muted.

Returns: Return code, see Audio Interface documentation

Description: This function is used to turn off the audio from the radio. The function can for instance be used when searching for a station and the noise should be turned off.

4.1.6 AUD_radio_set_station

Prototype: SINT8 AUD_radio_set_station (UINT8 handle, AUD_radio_seek_mode_enum seekmode, UINT32 frequency)

Parameters: UINT8 handle

AUD_radio_seek_mode_enum seekmode, see table:

UINT32 frequency

Copyright (C) 2006NeonSeven S.R.L. All rights reserved
- Exclusive property of Infineon Technologies AG –
Confidential

Enum: AUD_radio _seek_mode_enum	
Enum name	Description
off	Auto seek disabled and the frequency from parameter *frequency* is used to set the station
manualUp	Manual seek one frequency step up
manualDown	Manual seek one frequency step down
autoUp	Auto station seek up
autoDown	Auto station seek down

Table 5-3 Seek mode

Returns: Return code, see Audio Interface documentation

Description: This function is used to set or find a station. With the parameter *seekmode* it's possible to decide whether it's a manual or a automatic station seek. If it's an automatic station seek the parameter *frequency* doesn't matter.

A SDL message is used to inform the application when a station is found. The SDL message is send when a station is found or when the current band end has been reached without finding any station[1]. The SDL message is only used as trigger that indicates to the application that a station has been found. The boolean parameter *station_found* in the SDL message indicates whether a station is found or not. The radio application must use the *AUD_radio_read_current_*status function if it wants information about station quality.

4.1.7 AUD_radio_read_current_status

Prototype: SINT8 AUD_radio_read_current_status (UINT8 handle);

Parameters: UINT8 handle

Returns: Return code, see Audio Interface documentation

Description: This function is used from the application to get the status of FM radio. A SDL message returns the status parameters. The status parameters are:
UINT16 sender_name[0..8]
UINT8 type_of_station[0..8]
UINT32 frequency[2]
UINT8 level
UINT8 stereo

[1] The SDL message is sent when for the second time the band limit is reached. This indicates that no station is found during search.
[2] The frequency is represented by an integer. The frequency parameter is station's frequency in kHz e.g. 89.7 MHz => frequency = 89700

Copyright (C) 2006NeonSeven S.R.L. All rights reserved - Exclusive property of Infineon Technologies AG – Confidential

4.2 Interface functions for test purpose

In this section there is a description of the functions used for test purposed.

4.2.1 AFR_ptest_generic_func

Prototype: word AFR_ptest_generic_func (atctst_afr_generic_func_req_type *func_req_ptr);

Parameters: atctst_afr_generic_func_req_type *func_req_ptr

Returns: Return code, see Table 5-4

Description: This function is used by the Phonetool to test all the FM radio features described before. PhoneTool have to be version 9.9, otherwise the return code from the interface functions are incompatible with the PhoneTool

Enum: AUD_radio_return_code	
Enum name	Description
aud_rc_handle_not_used	Returned when calling AUD_radio_disable and radio not started
aud_rc_resource_in_use	Returned when calling AUD_radio_enable and radio not stopped
aud_rc_no_hw_support	
aud_rc_ok	success

Table 5-4 Return Test code

4.3 Other interface functions

4.3.1 AUD_set_volume

Prototype: SINT8 AUD_set_volume (UINT16 Level);

Parameters: UINT16 Level

Returns: Return code, see Audio Interface documentation

Description: This function is used from the application to increase or decrease the volume of FM radio. Inside the source file, the real function that has an action on the amplifier is FM_Volume (UINT16 Level).

Copyright (C) 2006NeonSeven S.R.L. All rights reserved
- Exclusive property of Infineon Technologies AG –
Confidential

5 References

5.1 External

TEA 5761UK Preliminary IC Specification V2.4

5.2 Internal

Title	Doc ID

6 Document change report

	Change Reference		Record of changes made to previous released version	
Rev	**Date**	**CR**	**Section**	**Comment**
1.0	18/06/2004	N.A	Document Creation	
1.1	27/01/2006	N.A	Document updated to BP30 platform	

7 Approval

Revision	Approver(s)	Date	Source/signature
1.0	Stefano Godeas	18/06/2004	Document stored on server
1.1	Stefano Godeas	27/01/2006	Document stored on server

Copyright (C) 2006NeonSeven S.R.L. All rights reserved
- Exclusive property of Infineon Technologies AG –

8 Annex 1

None