

**Firmware Design Specifications**

**iMX8X-ML**

|  |  |
| --- | --- |
| **Version**  **Status**  **Date** | 1.0  Baselined  31-Jan-2019 |
|  |  |

**Confidentiality Notice**

Copyright (c) 2019 eInfochips. - All rights reserved

This document is authored by eInfochips and is eInfochips intellectual property, including the copyrights in all countries in the world. This document is provided under a license to use only with all other rights, including ownership rights, being retained by eInfochips. This file may not be distributed, copied, or reproduced in any manner, electronic or otherwise, without the express written consent of eInfochips

Contents

[1. Document Details 4](#_Toc536797907)

[1.1 Revision & Approval History 4](#_Toc536797908)

[1.2 Definition, Acronyms and Abbreviations 4](#_Toc536797909)

[1.3 References 4](#_Toc536797910)

[2. Introduction 5](#_Toc536797911)

[2.1 Purpose of the Document 5](#_Toc536797912)

[2.2 Intended Audience 5](#_Toc536797913)

[3. Moule overview 6](#_Toc536797914)

[3.1 Functional description 6](#_Toc536797916)

[3.1.1 Features 6](#_Toc536797917)

[3.1.2 Functional Block Diagram 7](#_Toc536797918)

[3.1.3 Module Work Flow 7](#_Toc536797919)

[3.2 Hardware description 8](#_Toc536797920)

[3.2.1 Schematic 8](#_Toc536797921)

[4. SOFTWARE Overview 10](#_Toc536797922)

[4.1 Software Implementation 10](#_Toc536797923)

[4.1.1 Driver details 10](#_Toc536797924)

[4.1.2 Installation instructions 10](#_Toc536797925)

[4.1.3 Getting the driver 10](#_Toc536797926)

[4.1.4 Including the driver to the kernel 13](#_Toc536797927)

[4.1.5 Creating the device node 13](#_Toc536797928)

[5. Design Limitation 15](#_Toc536797929)

[6. Risk, ASSUMPTIONS AND CONSTRAINTS 16](#_Toc536797930)

[7. REFERENCES 17](#_Toc536797931)

[I2S Output Digital Microphone - Adafruit Industries : https://cdn-shop.adafruit.com/product-files/3421/i2S+Datasheet.PDF 17](#_Toc536797932)

[8. Annexure 18](#_Toc536797933)

FIGURES

[Figure 1 Block diagram of SPH0645LM4H-B 6](#_Toc535051996)

[Figure 2 Block diagram of I2S digital MIC 7](#_Toc535051997)

[Figure 3 Schematic interface 8](#_Toc535051998)

# Document Details

## Revision & Approval History

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Version** | **Author** | | **Reviewer** | | **Approver** | | |
| **Name** | **Date**  **(DD-MM-YYYY)** | **Name** | **Date**  **(DD-MMM-YYYY)** | **Name** | **Date**  **(DD-MM-YYYY)** |
| Draft  0.1 | Saurav Pandya | 11-Jan-2019 | Parth Modi | 28-Jan-19 | Prajose John | 31-Jan-19 |
| 1.0 | Baselined |  |  |  |  |  |

| **Version** | **Description of Change** |
| --- | --- |
| Draft  0.1 | Draft Version |
| 1.0 | Baselined |
|  |  |
|  |  |
|  |  |
|  |  |

## Definition, Acronyms and Abbreviations

| Definition/Acronym/Abbreviation | Description |
| --- | --- |
| DMIC | Digital Mic |
| I2S | Inter IC-sound |
| SNR | Signal to Noise ratio |
| RF | Radio Frequency |

## References

| No. | Document | Version | Remarks |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose of the Document

This design note describes how to add support for SPH0645LM4H-B Codec to a Linux system from software perspective.

## Intended Audience

The document is intended for developers, project team members.

# Moule overview

The SPH0645LM4H-B is a miniature, low power, bottom port microphone with an I2S digital output. The solution consists of a proven high performance SiSonic™ acoustic sensor, a serial Analog to Digital convertor, and an interface to condition the signal into an industry standard 24 bits I2S format. The I2S interface simplifies the integration in the system and allow direct interconnect to digital processors, application processors and microcontroller. Saving the need of an external audio codec, the SPH0645LM4H-B is perfectly suitable for portable applications where size and power consumption are a constraint



## Functional description

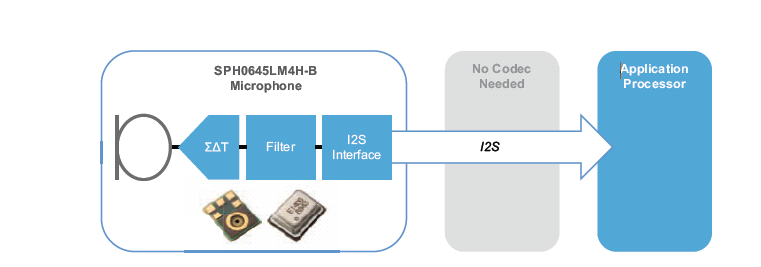


Figure 1 Block diagram of SPH0645LM4H-B

### Features

* High SNR of 65dB(A)
* Low Current of Type 600μA
* I2s Output: Direct attach to μA
* Multi modes: standard > 1Khz 600μA / sleep < 900Khz 10μA
* Flat frequency Response
* RF shielded
* Support dual Microphones
* Ultra-Stable Performance
* Standard SMD Reflow
* Omnidirectional
* Zero Height Mic
* Packaged in SPH 3.50x2.65x0.98mm

### Functional Block Diagram

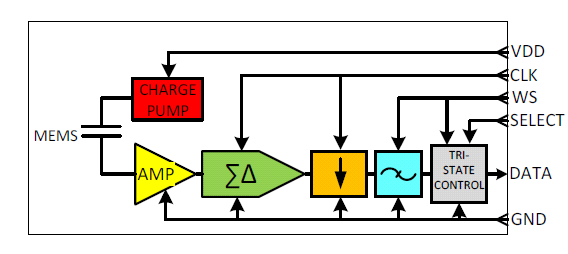


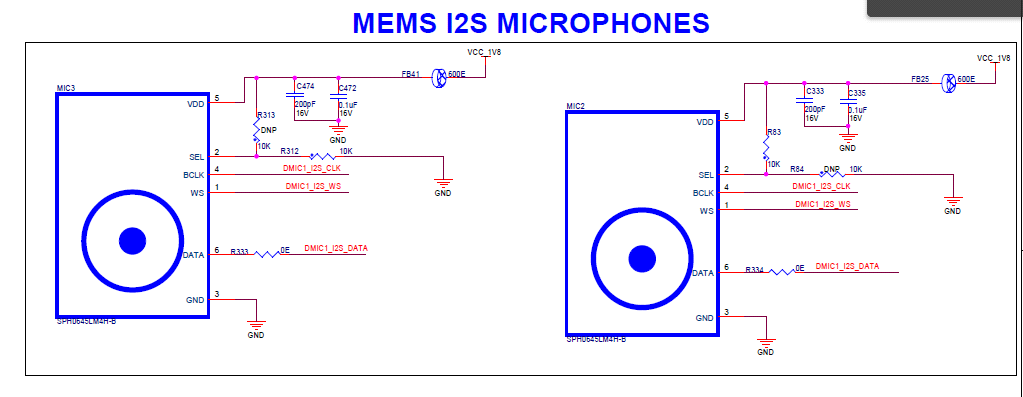
Figure 2 Block diagram of I2S digital MIC

### Module Work Flow

*NA*

## Hardware description

### Schematic



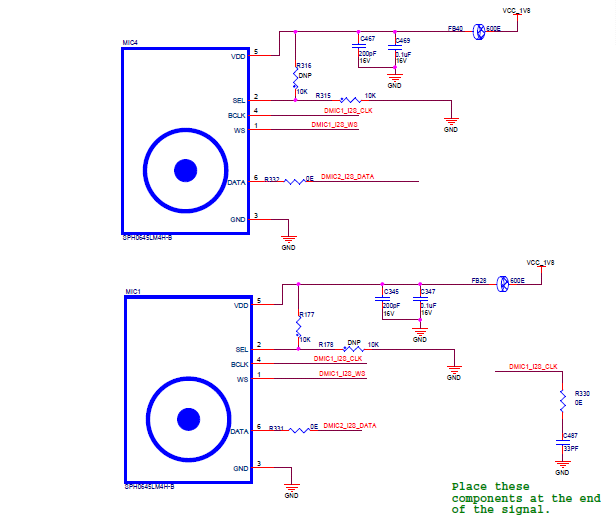


Figure 3 Schematic interface

# SOFTWARE Overview

## Software Implementation

## Driver details

The SPH0645LM4H-B driver offer communication to the microphones with I2S digital output.

## Installation instructions

The following instructions assume the driver being installed under the **sound/soc/fsl/imx-sph0645.c** source sub-folder. Below instructions may have to be adapted accordingly in case another path is chosen for the driver installation.

## Getting the driver

The SPH0645LM4H-B driver is available at below

/\*

\* Copyright (C) 2013-2015 Freescale Semiconductor, Inc.

\*

\* The code contained herein is licensed under the GNU General Public

\* License. You may obtain a copy of the GNU General Public License

\* Version 2 or later at the following locations:

\*

\* http://www.opensource.org/licenses/gpl-license.html

\* http://www.gnu.org/copyleft/gpl.html

\*/

#include <linux/module.h>

#include <linux/of\_platform.h>

#include <sound/soc.h>

#include <sound/pcm\_params.h>

#include "fsl\_esai.h"

static int imx\_sph0645\_hw\_params(struct snd\_pcm\_substream \*substream,

struct snd\_pcm\_hw\_params \*params)

{

struct snd\_soc\_pcm\_runtime \*rtd = substream->private\_data;

struct snd\_soc\_dai \*cpu\_dai = rtd->cpu\_dai;

u32 channels = 2; //ALWAYS 2 CHANNELS params\_channels(params);

u32 rate = params\_rate(params);

u32 bclk = rate \* channels \* 32;

int ret = 0;

printk(" ++++ sph0645 \n ");

/\* set cpu DAI configuration \*/

ret = snd\_soc\_dai\_set\_fmt(cpu\_dai, SND\_SOC\_DAIFMT\_I2S | SND\_SOC\_DAIFMT\_NB\_NF | SND\_SOC\_DAIFMT\_CBS\_CFS);

if (ret) {

dev\_err(cpu\_dai->dev, "failed to set dai fmt\n");

return ret;

}

//ret = snd\_soc\_dai\_set\_fmt(codec\_dai, SND\_SOC\_DAIFMT\_I2S | SND\_SOC\_DAIFMT\_NB\_NF | SND\_SOC\_DAIFMT\_CBS\_CFS);

//if (ret) {

// dev\_err(cpu\_dai->dev, "failed to set codec dai fmt: %d\n", ret);

// return ret;

//}

ret = snd\_soc\_dai\_set\_tdm\_slot(cpu\_dai, 3, 3, 2, 32);

if (ret) {

dev\_err(cpu\_dai->dev, "failed to set dai tdm slot\n");

return ret;

}

/\*ret = snd\_soc\_dai\_set\_sysclk(cpu\_dai, ESAI\_HCKT\_EXTAL, bclk , SND\_SOC\_CLOCK\_OUT);\*/

if (substream->stream == SNDRV\_PCM\_STREAM\_PLAYBACK)

{

ret = snd\_soc\_dai\_set\_sysclk(cpu\_dai, ESAI\_HCKT\_EXTAL, 12288000 , SND\_SOC\_CLOCK\_OUT);

if (ret){

dev\_err(cpu\_dai->dev, "failed to set cpu sysclk\n");

return ret;

}

}

else

{

ret = snd\_soc\_dai\_set\_sysclk(cpu\_dai, ESAI\_HCKR\_EXTAL, 12288000 , SND\_SOC\_CLOCK\_OUT);

if (ret){

dev\_err(cpu\_dai->dev, "failed to set cpu sysclk\n");

return ret;

}

}

///ret = snd\_soc\_dai\_set\_sysclk(codec\_dai, 0, bclk, SND\_SOC\_CLOCK\_IN);

///if (ret)

/// dev\_err(cpu\_dai->dev, "failed to set codec sysclk\n");

printk(" --- sph0645 \n ");

return ret;

};

static struct snd\_soc\_ops imx\_sph0645\_ops = {

.hw\_params = imx\_sph0645\_hw\_params,

};

static struct snd\_soc\_dai\_link imx\_dai = {

.name = "imx-sph0645",

.stream\_name = "imx-sph0645",

.codec\_name = "snd-soc-dummy",

.codec\_dai\_name = "snd-soc-dummy-dai",

.ops = &imx\_sph0645\_ops,

.dai\_fmt = SND\_SOC\_DAIFMT\_I2S | SND\_SOC\_DAIFMT\_NB\_NF | SND\_SOC\_DAIFMT\_CBS\_CFS,

.dpcm\_capture = 1,

.dpcm\_playback = 1,

};

static struct snd\_soc\_card snd\_soc\_card\_imx\_3stack = {

.name = "imx-audio-sph0645",

.dai\_link = &imx\_dai,

.num\_links = 1,

.owner = THIS\_MODULE,

};

static int imx\_sph0645\_probe(struct platform\_device \*pdev)

{

struct snd\_soc\_card \*card = &snd\_soc\_card\_imx\_3stack;

struct device\_node \*ssi\_np, \*np = pdev->dev.of\_node;

struct platform\_device \*ssi\_pdev;

int ret;

/\* int int\_port, ext\_port, ret;

ret = of\_property\_read\_u32(np, "mux-int-port", &int\_port);

if (ret) {

dev\_err(&pdev->dev, "mux-int-port missing or invalid\n");

return ret;

}

ret = of\_property\_read\_u32(np, "mux-ext-port", &ext\_port);

if (ret) {

dev\_err(&pdev->dev, "mux-ext-port missing or invalid\n");

return ret;

}

imx\_audmux\_config(int\_port, ext\_port);

\*/

ssi\_np = of\_parse\_phandle(pdev->dev.of\_node, "ssi-controller", 0);

if (!ssi\_np) {

dev\_err(&pdev->dev, "phandle missing or invalid\n");

return -EINVAL;

}

ssi\_pdev = of\_find\_device\_by\_node(ssi\_np);

if (!ssi\_pdev) {

dev\_err(&pdev->dev, "failed to find SSI platform device\n");

ret = -EINVAL;

goto end;

}

card->dev = &pdev->dev;

card->dai\_link->cpu\_dai\_name = dev\_name(&ssi\_pdev->dev);

card->dai\_link->platform\_of\_node = ssi\_np;

platform\_set\_drvdata(pdev, card);

ret = snd\_soc\_register\_card(card);

if (ret)

dev\_err(&pdev->dev, "Failed to register card: %d\n", ret);

end:

if (ssi\_np)

of\_node\_put(ssi\_np);

return ret;

}

static int imx\_sph0645\_remove(struct platform\_device \*pdev)

{

struct snd\_soc\_card \*card = &snd\_soc\_card\_imx\_3stack;

snd\_soc\_unregister\_card(card);

return 0;

}

static const struct of\_device\_id imx\_sph0645\_dt\_ids[] = {

{ .compatible = "fsl,imx-audio-sph0645", },

{ /\* sentinel \*/ }

};

MODULE\_DEVICE\_TABLE(of, imx\_sph0645\_dt\_ids);

static struct platform\_driver imx\_sph0645\_driver = {

.driver = {

.name = "imx-mic-sph0645",

.pm = &snd\_soc\_pm\_ops,

.of\_match\_table = imx\_sph0645\_dt\_ids,

},

.probe = imx\_sph0645\_probe,

.remove = imx\_sph0645\_remove,

};

module\_platform\_driver(imx\_sph0645\_driver);

/\* Module information \*/

MODULE\_AUTHOR("Freescale Semiconductor, Inc.");

MODULE\_DESCRIPTION("ALSA SoC i.MX sph0645");

MODULE\_LICENSE("GPL");

MODULE\_ALIAS("platform:imx-mic-sph0645");

## Including the driver to the kernel

The SPH0645LM4H-B driver put into the **sound/soc/fsl/Kconfig** and **sound/soc/fsl//Makefile** in kernel source.

## Creating the device node

Below is the SPH0645LM4H-B node implemented inside platform device tree file. (.dts file located for instance under arch/arm64/boot/dts/freescale/fsl-imx8qxp-mek.dts)

**sound-dmic {**

**compatible = "fsl,imx-audio-sph0645",**

**"fsl,imx-mic-sph0645";**

**model = "imx-sph0645";**

**ssi-controller = <&esai0>;**

**status = "okay";**

**};**

# Design Limitation

*NONE*

# Risk, ASSUMPTIONS AND CONSTRAINTS

*NONE*

# REFERENCES

|  |
| --- |
| I2S Output Digital Microphone - Adafruit Industries : <https://cdn-shop.adafruit.com/product-files/3421/i2S+Datasheet.PDF> |
|  |

# Annexure

NONE