

# Acoustic Event Detection MI User Manual

© 2017 MStar Semiconductor, Inc. All rights reserved.

MStar Semiconductor makes no representations or warranties including, for example but not limited to, warranties of merchantability, fitness for a particular purpose, non-infringement of any intellectual property right or the accuracy or completeness of this document, and reserves the right to make changes without further notice to any products herein to improve reliability, function or design. No responsibility is assumed by MStar Semiconductor arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights, nor the rights of others.

MStar is a trademark of MStar Semiconductor, Inc. Other trademarks or names herein are only for identification purposes only and owned by their respective owners.

## REVISION HISTORY

| Revision No. | Description | Date       |
|--------------|-------------|------------|
| 1.0          | • Created   | 08/07/2017 |

## TABLE OF CONTENTS

|  |           |
|--|-----------|
| REVISION HISTORY .....                                   | i         |
| TABLE OF CONTENTS.....                                   | ii        |
| LIST OF FIGURES .....                                    | iii       |
| <b>1. INTRODUCTION.....</b>                              | <b>1</b>  |
| 1.1. Purpose .....                                       | 1         |
| <b>2. Specification .....</b>                            | <b>2</b>  |
| <b>3. API REFERENCE .....</b>                            | <b>4</b>  |
| 3.1. API Overview .....                                  | 4         |
| 3.2. API Lists.....                                      | 4         |
| <b>MI_LSD_Init</b> .....                                 | 4         |
| <b>MI_LSD_Uninit</b> .....                               | 5         |
| <b>MI_LSD_SetLsdThreshold</b> .....                      | 5         |
| <b>MI_LSD_GetdBResult</b> .....                          | 6         |
| <b>MI_LSD_Run</b> .....                                  | 6         |
| <b>MI_LSD_GetResult</b> .....                            | 7         |
| <b>4. DATA TYPE .....</b>                                | <b>8</b>  |
| 4.1. Overview.....                                       | 8         |
| 4.2. Structure Lists .....                               | 8         |
| <b>LSD_PARAMS</b> .....                                  | 8         |
| 4.3. Enumeration Lists .....                             | 8         |
| <b>MI_RET</b> .....                                      | 8         |
| <b>5. FLOW.....</b>                                      | <b>10</b> |
| 5.1. Loud Sound Detection.....                           | 10        |
| <b>6. CODE/DATA SIZE INFORMATION.....</b>                | <b>11</b> |
| <b>7. DRAVE USAGE INFORMATION (WORKING BUFFER) .....</b> | <b>12</b> |
| <b>8. CPU MIPS/CLOCK CYCLES ESTIMATION .....</b>         | <b>13</b> |

## **LIST OF FIGURES**

|   |    |
|---|----|
| Figure 1: Audio example of aliasing. ....                           | 2  |
| Figure 2: Audio example of clipping. ....                           | 2  |
| Figure 3: Audio example of effective sample rate is below 8kHz .... | 3  |
| Figure 4: Code/data size information .....                          | 11 |

## 1. INTRODUCTION

---

### 1.1. Purpose

Loud sound detection (LSD) is a function used for detecting dBFS from audio streams.

## 2. SPECIFICATION

---

1. For best performance, background environment should be quiet
2. If you are using audio files as the sound source, you should make sure
  - 甲、 There is no aliasing in audio files, see Figure 1
  - 乙、 There is no signal clipping in audio files, see Figure 2
  - 丙、 Effective sample rate is larger than 8 kHz, see Figure 3
  - 丁、 Speaker volume and mic gain is high enough

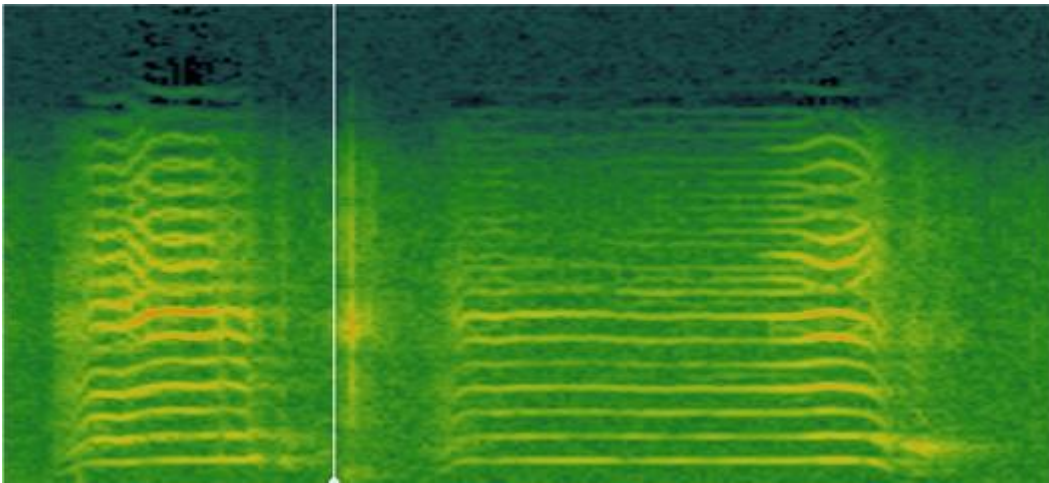


Figure 1: Audio example of aliasing.

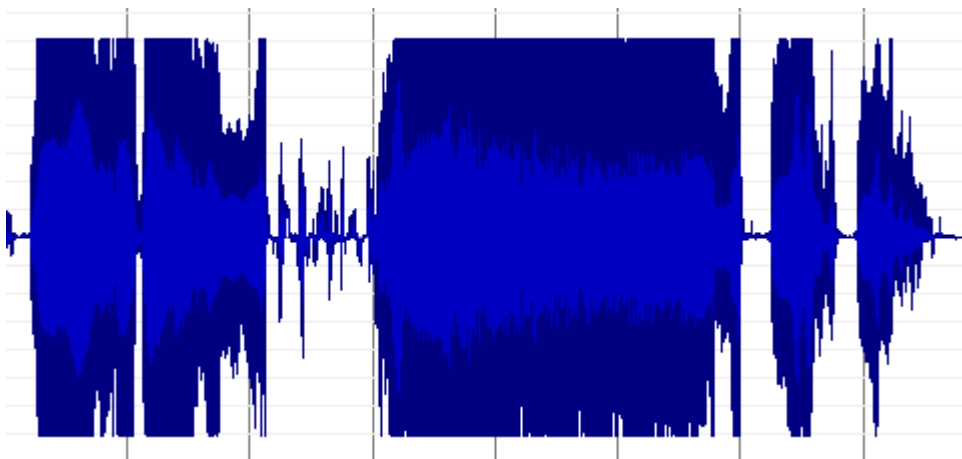


Figure 2: Audio example of clipping.

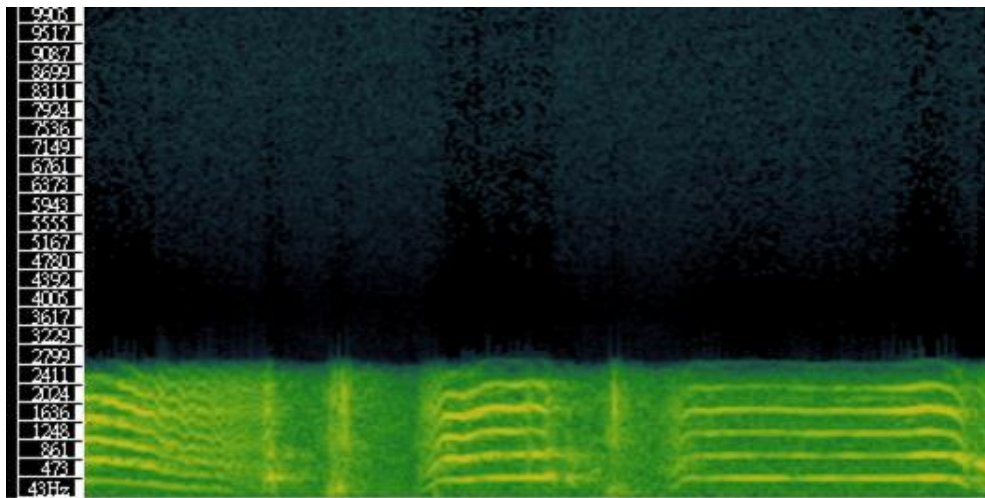


Figure 3. Audio example of effective sample rate is below 8kHz



## 3. API REFERENCE

---

### 3.1. API Overview

- [MI\\_LSD\\_Init](#): Initialize LSD library
- [MI\\_LSD\\_Uninit](#): To exit the lib function and release memory
- [MI\\_LSD\\_GetLsdResult](#): Get dBFS result of LSD process
- [MI\\_LSD\\_Run](#): Perform LSD
- [MI\\_LSD\\_GetResult](#): Get result of LSD library
- [MI\\_LSD\\_SetThreshold](#): Set LSD threshold in dBFS

### 3.2. API Lists

#### MI\_LSD\_Init

##### Purpose

Initialize LSD library

##### Function Prototype

```
LSD_HANDLE MI_LSD_Init(LSD_PARAMS *lsd_params, S32 *point_length);
```

##### Arguments

| Name         | Description             |
|--------------|-------------------------|
| lsd_params   | Structure of lsd_params |
| point_length | Input data length       |

##### Return value

| Return value | Description                |
|--------------|----------------------------|
| LSD_HANDLE   | LSD handle pointer address |
| NULL         | Initialization failure     |

##### Requirement

Header files: mi\_lsd.h

Library files: libLSD\_Linux.a or libLSD\_Linux.so

## MI\_LSD\_Uninit

### Purpose

To exit the lib function and release memory

### Function Prototype

```
MI_RET MI_LSD_Uninit(LSD_HANDLE lsd_handle);
```

### Arguments

| Name       | Description               |
|------------|---------------------------|
| lsd_handle | Pointer to the LSD_HANDLE |

### Return value

| Return value              | Description        |
|---------------------------|--------------------|
| MI_RET_SUCCESS            | Success            |
| MI_LSD_RET_INVALID_HANDLE | Invalid LSD handle |

### Requirement

Header files: mi\_lsd.h

Library files: libLSD\_Linux.a or libLSD\_Linux.so

## MI\_LSD\_SetLsdThreshold

### Purpose

Set LSD threshold in dBFS

### Function Prototype

```
MI_RET MI_LSD_SetLsdThreshold(LSD_HANDLE lsd_handle, S32 threshold_db);
```

### Arguments

| Name         | Description                     |
|--------------|---------------------------------|
| lsd_handle   | Pointer to the LSD_HANDLE       |
| threshold_db | Default threshold is -15 (dBFS) |

### Return value

| Return value              | Description        |
|---------------------------|--------------------|
| MI_RET_SUCCESS            | Success            |
| MI_LSD_RET_INVALID_HANDLE | Invalid LSD handle |

### Requirement

Header files: mi\_lsd.h

Library files: libLSD\_Linux.a or libLSD\_Linux.so

## MI\_LSD\_GetdBResult

### Purpose

Get dBFS result of LSD

### Function Prototype

*MI\_RET* MI\_LSD\_GetdBResult(*LSD\_HANDLE* lsd\_handle, *S16* \*audio\_input, *S16* \*lsd\_db\_result);

### Arguments

| Name          | Description  |
|---------------|--|
| lsd_handle    | Pointer to the LSD_HANDLE  |
| audio_input   | Audio input address. The input array should have point_number*channel (fields of LSDProcessStruct) elements. For example, for 8 kHz stereo, the input array should have 256*2 elements; for 32 kHz mono, the input array should have 1024*1 elements |
| lsd_db_result | Pointer to the value of dBFS   |

### Return value

| Return value              | Description        |
|---------------------------|--------------------|
| MI_RET_SUCCESS            | Success            |
| MI_LSD_RET_INVALID_HANDLE | Invalid LSD handle |

### Requirement

Header files: mi\_lsd.h

Library files: libLSD\_Linux.a or libLSD\_Linux.so

### Note

- MI\_LSD\_GetdBResult should be called before MI\_LSD\_Run for each frame

## MI\_LSD\_Run

### Purpose

Perform LSD

### Function Prototype

*MI\_RET* MI\_LSD\_Run(*LSD\_HANDLE* lsd\_handle, *S16* \*lsd\_db\_result);

### Arguments

| Name          | Description               |
|---------------|---------------------------|
| lsd_handle    | Pointer to the LSD_HANDLE |
| lsd_db_result | Pointer to the dBFS value |

#### Return value

| Return value          | Description    |
|-----------------------|----------------|
| MI_RET_SUCCESS        | Success        |
| MI_LSD_RET_INIT_ERROR | LSD Init error |

#### Requirement

Header files: mi\_lsd.h

Library files: libLSD\_Linux.a or libLSD\_Linux.so

## MI\_LSD\_GetResult

#### Purpose

Get result of LSD

#### Function Prototype

```
MI_RET MI_LSD_GetResult(LSD_HANDLE lsd_handle, S16 *lsd_result);
```

#### Arguments

| Name       | Description                                  |
|------------|--|
| lsd_handle | Pointer to the LSD_HANDLE                    |
| lsd_result | Pointer to the result of loud sound detected |

#### Return value

| Return value              | Description        |
|---------------------------|--------------------|
| MI_RET_SUCCESS            | Success            |
| MI_LSD_RET_INVALID_HANDLE | Invalid LSD handle |

#### Requirement

Header files: mi\_lsd.h

Library files: libLSD\_Linux.a or libLSD\_Linux.so

## 4. DATA TYPE

### 4.1. Overview

|                            |  |
|----------------------------|--|
| <a href="#">LSD_PARAMS</a> | Define the audio sample rate and channel number of LSD |
| <a href="#">MI_RET</a>     | Define error code of LSD                               |

### 4.2. Structure Lists

#### LSD\_PARAMS

##### Description

Define the audio sample rate and channel number of LSD

##### Syntax

```
typedef struct {
    unsigned int sample_rate;
    unsigned int channel;
} LSD_PARAMS;
```

##### Member

| Member      | Description                    |
|-------------|--------------------------------|
| sample_rate | The sample rate of audio input |
| channel     | Channel number                 |

### 4.3. Enumeration Lists

#### MI\_RET

##### Description

Define error code of LSD

##### Syntax

```
typedef enum {
    MI_RET_SUCCESS = 0x00000000,
    MI_LSD_RET_INIT_ERROR = 0x10000701,
    MI_LSD_RET_IC_CHECK_ERROR = 0x10000702,
    MI_LSD_RET_INVALID_HANDLE = 0x10000703,
    MI_LSD_RET_INVALID_SAMPLERATE = 0x10000704
} MI_LSD_RET;
```

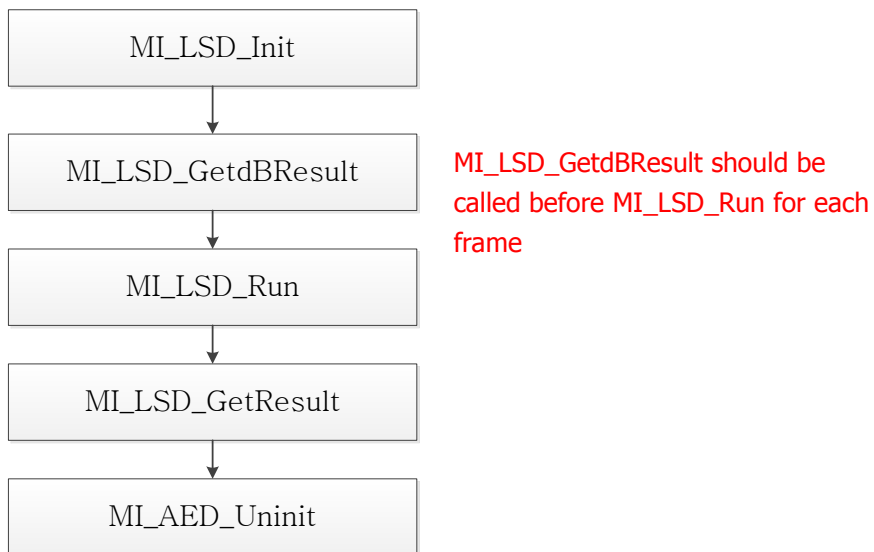
Member

| Member                        | Description                      |
|-------------------------------|----------------------------------|
| MI_RET_SUCCESS                | Success                          |
| MI_LSD_RET_INIT_ERROR         | LSD init error                   |
| MI_LSD_RET_IC_CHECK_ERROR     | Incorrect platform check for LSD |
| MI_LSD_RET_INVALID_HANDLE     | Invalid LSD handle               |
| MI_LSD_RET_INVALID_SAMPLERATE | Invalid Sample rate of LSD       |

## 5. FLOW

---

### 5.1. Loud Sound Detection



## 6. CODE/DATA SIZE INFORMATION

| Code      | RO Data                        | RW Data | ZI Data | Debug   |              |
|-----------|--------------------------------|---------|---------|---------|--------------|
| 42228     | 1038                           | 420     | 10600   | 71756   | Grand Totals |
| =====     |                                |         |         |         |              |
| Total RO  | Size(Code + RO Data)           |         |         | 43266 ( | 42.25kB)     |
| Total RW  | Size(RW Data + ZI Data)        |         |         | 11020 ( | 10.76kB)     |
| Total ROM | Size(Code + RO Data + RW Data) |         |         | 43686 ( | 42.66kB)     |
| =====     |                                |         |         |         |              |

Figure 4: Code/data size information



## 7. Drame Usage Information (Working Buffer)

---

| Sample Rate of Audio Input | Buffer Size (bytes) |
|----------------------------|---------------------|
| 8kHz                       | 0                   |
| 16kHz                      | 16992               |
| 32kHz                      | 16992               |

## 8. CPU MIPS/CLOCK CYCLES ESTIMATION

---

### ■ Loud sound detection

run MI\_LSD\_GetdBResult , MI\_LSD\_Run every 32 msec

I3, CPU freq = 400 MHz

- 8 kHz/mono
  - 0.05 ms
- 16 kHz/mono
  - 0.20 ms
- 32 kHz/mono
  - 0.22 ms