

# Face Detection User Manual

---

v1.5

© 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, 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 user 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.	12/07/2015
1.1	• Add the API of direction of face detection	17/06/2016
1.2	• Add specification for face detection/recognition	11/08/2016
1.3	• Add the parameter of init and exist API	31/08/2016
1.4	• Add set FR mode of API	20/01/2017
1.5	• Add set FR face width of API	13/02/2017

## TABLE OF CONTENTS

REVISION HISTORY .....	ii
TABLE OF CONTENTS .....	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>3</b>
3.1. API Overview .....	3
3.2. API Lists.....	3
<b>MI_FD_Init</b> .....	3
<b>MI_FD_Uninit</b> .....	4
<b>MI_FR_SetFrMode</b> .....	4
<b>MI_FR_SetFrFaceWidth</b> .....	5
<b>MI_FD_EnableFD</b> .....	5
<b>MI_FR_EnableFR</b> .....	6
<b>MI_FD_SetOption</b> .....	6
<b>MI_FD_Run</b> .....	9
<b>MI_FD_GetFaceInfo</b> .....	10
<b>MI_FR_CalFeatureFromImg</b> .....	11
<b>MI_FR_GetFeatureSizes</b> .....	12
<b>MI_FR_GetFeatureData</b> .....	12
<b>MI_FR_SetFeatureData</b> .....	13
<b>MI_FR_EnableFR</b> .....	13
<b>MI_FR_CalFeatureFromRawY</b> .....	14
<b>MI_FR_CalcImageScore</b> .....	14
<b>4. DATA TYPE .....</b>	<b>16</b>
4.1. Overview.....	16
4.2. Struct Lists .....	16
<b>PFID_FACE_POSITION</b> .....	16
<b>PFID_RECT</b> .....	18
<b>PFID_FACE_DETECT</b> .....	18
4.3. Enumeration Lists .....	19
<b>FDFR_ROMFILES</b> .....	19
<b>5. FACE DETECTION FLOW.....</b>	<b>20</b>
<b>6. steps for generating face recognition database .....</b>	<b>21</b>
<b>7. Code Size Information .....</b>	<b>22</b>
<b>8. DRAM Usage Information (Working Buffer).....</b>	<b>23</b>
<b>9. CPU MIPS/Clock Cycles Estimation .....</b>	<b>24</b>

## **1. INTRODUCTION**

---

### **1.1. Purpose**

Face detection is a function used for detecting human face in the scene captured by video camera. The applications for this function are, for example, security monitoring.

## 2. SPECIFICATION

---

1. Suggesting the system to use in the **uniform lighting conditions**. **Backlight** and **low lighting** cases have poor performance.
2. **No glasses** or **masks** on the face.
3. Face detection number up to **10**.
4. **Face angle limitations** for **face detection/recognition**

Axis	Max. Range	Comment
Roll	$\pm 45^\circ$	Rotation in plane
Yaw	$\pm 30^\circ$	Rotation out of plane horizontally
Pitch	$\pm 20^\circ$	Rotation out of plane vertically

Performance degrades if exceeds the maximum ranges.

5. The **width** of **face image** should be **20 pixels or more** for face detection case.
6. **The maximum number** of **face recognition database** is **21**, and each **background-image** should be **simple**.
7. The **input face image** should be **20 pixels or more between right and left eyes** for face recognition case.

## 3. API REFERENCE

---

### 3.1. API Overview

- [MI\\_FD\\_Init](#): Initialize FD library.
- [MI\\_FD\\_Uninit](#): To exit the lib function and release memory.
- [MI\\_FR\\_SetFrMode](#): Set FR mode.
- [MI\\_FR\\_SetFrFaceWidth](#): To set a threshold to decide whether do FR or not.
- [MI\\_FD\\_EnableFD](#): To enable/disable the face detection function.
- [MI\\_FR\\_EnableFR](#): To enable/disable the face recognition function.
- [MI\\_FD\\_SetOption](#): To set face detection/tracking mode of the face detection. To set different face/camera direction of the face detection. To set minimum face width in detection and tracking mode. To set minimum face width in partial detection mode.
- [MI\\_FD\\_Run](#): To process fd/fr one frame.
- [MI\\_FD\\_GetFaceInfo](#): To get result of the face detection.
- [MI\\_FR\\_CalFeatureFromImg](#): To generate FR data base from one image.
- [MI\\_FR\\_GetFeatureSizes](#): To get size of FR data base.
- [MI\\_FR\\_GetFeatureData](#): To get FR data base.
- [MI\\_FR\\_SetFeatureData](#): To set FR data base.
- [MI\\_FR\\_EnableFR](#): To set FR enable.
- [MI\\_FR\\_CalFeatureFromRawY](#): To calculate feature from image.
- [MI\\_FR\\_CalcImageScore](#): To get suitable face frame while generating FR database.

### 3.2. API Lists

#### MI\_FD\_Init

##### Purpose

Initialize FD library.

##### Function Prototype

```
int MI_FD_Init(FDFR_ROMFILES romfiles);
```

##### Arguments

Name	Description
romfiles	Enumeration of FDFR_ROMFILES

##### Return value

Return value	Description
0	Success
-1	Fail

**Requirement**

Header files: FDFR.h

Library files: libFDFR\_LinuxC3.so

**MI\_FD\_Uninit****Purpose**

To exit the lib function and release memory

**Function Prototype**

int MI\_FD\_Uninit(FDFR\_ROMFILES romfiles);

**Arguments**

Name	Description
romfiles	Enumeration of FDFR_ROMFILES

**Return value**

Return value	Description
0	Success
-1	Fail

**Requirement**

Header files: FDFR.h

Library files: libFDFR\_LinuxC3.so

**MI\_FR\_SetFrMode****Purpose**

Set FR mode.

**Function Prototype**

int MI\_FR\_SetFrMode(int mode)

**Arguments**

Name	Description
mode	0 : Low accuracy but speed is fast. 1 : Middle accuracy but speed is middle.



Name	Description
	2 : High accuracy but speed is slow. (default)

Return value

Return value	Description
0	Success
-1	Fail

Requirement

Header files: FDFR.h

Library files: libFDFR\_LinuxC3.so

## MI\_FR\_SetFrFaceWidth

Purpose

To set a threshold to decide whether do FR or not.

Function Prototype

int MI\_FR\_SetFrFaceWidth(int face\_width)

Arguments

Name	Description
face_width	Setting a threshold, If the Face width of Face is bigger than this threshold, the process will do FR. Otherwise, the process will still do FD.

Return value

Return value	Description
0	Success
-1	Fail

Requirement

Header files: FDFR.h

Library files: libFDFR\_LinuxC3.so

## MI\_FD\_EnableFD

Purpose

To enable/disable the face detection function.

Function Prototype

MI\_RET MI\_FD\_EnableFD(int enable)

Arguments

Name	Description
enable	1 : enable face detection function 0 : disable face detection function

Return value

Return value	Description
MI_FDFR_RET_SUCCESS	Success
MI_FDFR_RET_FD_ENABLE_ERROR	Fail

Requirement

Header files: FDFR.h

Library files: libFDFR\_LinuxC3.so

**MI\_FR\_EnableFR**Purpose

To enable/disable the face recognition function.

Function Prototype

int MI\_FR\_EnableFR (int enable)

Arguments

Name	Description
enable	1 : enable face recognition function 0 : disable face recognition function

Return value

Return value	Description
0	Success
-1	Fail

Requirement

Header files: FDFR.h

Library files: libFDFR\_LinuxC3.so

**MI\_FD\_SetOption**

**Purpose**

To set face detection/tracking mode of the face detection.  
To set different face/camera direction of the face detection.  
To set minimum face width in detection and tracking mode.  
To set minimum face width in partial detection mode.

**Function Prototype**

MI\_RET MI\_FD\_SetOption(FDOption\_e opt, S32 val)

**Arguments**

Name	Description
opt	FD_OPTION_DETECT_MODE
option	0: pure detection mode. Each frame starts new detecting process, and don't use previous result.  1: pure tracking faces detected in previous frame. Only start new detection process when there is no face in previous frame.  2: tracking faces in previous frame and partially detect other region

Name	Description
opt	FD_OPTION_FACE_DIRECTION
fdoption	0: detecting the face/camera in each frame was rotated 0 degree in counterclockwise(roll) direction (equivalently pure detection mode).  1: detecting the face/camera in each frame was rotated 90 degree in counterclockwise(roll) direction.  2: detecting the face/camera in each frame was rotated 0 and 90 degrees in counterclockwise(roll) direction.  3: detecting the face/camera in each frame was rotated 180 degree in counterclockwise direction.  4: detecting the face/camera in each frame was rotated 0 and 180 degrees in counterclockwise(roll) direction.

Name	Description
	<p>5: detecting the face/camera in each frame was rotated 90 and 180 degrees in counterclockwise(roll) direction.</p> <p>6: detecting the face/camera in each frame was rotated 0, 90 and 180 degrees in counterclockwise(roll) direction.</p> <p>7: detecting the face/camera in each frame was rotated 270 degree in counterclockwise(roll) direction.</p> <p>8: detecting the face/camera in each frame was rotated 0 and 270 degrees in counterclockwise(roll) direction.</p> <p>9: detecting the face/camera in each frame was rotated 90 and 270 degrees in counterclockwise(roll) direction.</p> <p>10: detecting the face/camera in each frame was rotated 0, 90, and 270 degrees in counterclockwise(roll) direction.</p> <p>11: detecting the face/camera in each frame was rotated 180 and 270 degrees in counterclockwise(roll) direction.</p> <p>12: detecting the face/camera in each frame was rotated 0, 180 and 270 degrees in counterclockwise(roll) direction.</p> <p>13: detecting the face/camera in each frame was rotated 90, 180 and 270 degrees in counterclockwise(roll) direction.</p> <p>14: detecting the face/camera in each frame was rotated 0, 90, 180 and 270 degrees in counterclockwise(roll) direction.</p>

Name	Description
------	-------------

Name	Description
opt	FD_OPTION_FACE_WIDTH
face_width	Minimum face width to detect. (Note: this values should be bigger than 20)

Name	Description
opt	FD_OPTION_PARTIAL_WIDTH
face_width	Minimum face width to partial detect. (Note: this values should be bigger than 20)

#### Return value

Return value	Description
MI_FDFR_RET_SUCCESS	Success
MI_FDFR_RET_INVALID_PARAMETER	Fail

#### Requirement

Header files: FDFR.h

Library files: libFDFR\_LinuxC3.so

## MI\_FD\_Run

#### Purpose

To process fd/fr one frame

#### Function Prototype

```
int MI_FD_Run(unsigned char* imgPtr, int width, int height);
```

#### Arguments

Name	Description
imgPtr	Source image (gray level image)
width	Width of input image
height	Height of input image

#### Return value

Return value	Description
-1	Fail
other	number of faces detected

#### Requirement

Header files: FDFR.h

Library files: libFDFR\_LinuxC3.so

## MI\_FD\_GetFaceInfo

### Purpose

To get result of the face detection

### Function Prototype

int MI\_FD\_GetFaceInfo(unsigned short\* fd\_info\_buf)

### Arguments

Name	Description
info	<p>Result buffer address, and the content is as follows</p> <pre> #define      PFID_MAX_FACE_NUM (20)      /* Max numbers of faces which can bedetected */ typedef struct _pfid_face_position {     signed short      flag;                                 /* Setting flag(PFID_SFLG_*) */     signed short      conf;                                 /* Face confidence [low?F0?`high?F100] */     signed short      rotate;                                 /* Face rotation flag(PFID_ FACE_ROLL_*) */     signed short      rect_l;                                 /* Left face frame (X coord) / negative: invalid */     signed short      rect_r;                                 /* Right face frame (X coord) / negative: invalid */     signed short      rect_t;                                 /* Top face frame (Y coord) / negative: invalid */     signed short      rect_b;                                 /* Bottom face frame (Y coord) / negative: invalid */     signed short      eye_lx;                                 /* Left eye (X coord) / negative: invalid */     signed short      eye_ly;                                 /* Left eye (Y coord) / negative: invalid */     signed short      eye_rx;                                 /* Right eye(X coord) / negative: invalid */     signed short      eye_ry;                                 /* Right eye(Y coord) / negative: invalid */     /* Other necessary information may be appended */ } PFID_FACE_POSITION; </pre>

Name	Description
	<pre>typedef struct _pfid_face_detect {     signed short        num;                         /* Numbers of detected faces (Max:PFID_MAX_FACE_NUM) */     PFID_FACE_POSITION     pos[PFID_MAX_FACE_NUM]; /* Face position */ } PFID_FACE_DETECT;.</pre>

#### Return value

Return value	Description
0	Success
-1	Fail

#### Requirement

Header files: FDFR.h

Library files: libFDFR\_LinuxC3.so

## MI\_FR\_CalFeatureFromImg

#### Purpose

To generate FR data base from one image

#### Function Prototype

MI\_RET MI\_FR\_CalFeatureFromImg(char\* filename, int store\_idx)

#### Arguments

Name	Description
filename	image file name (PGM format)
store_idx	index to store in data base

#### Return value

Return value	Description
MI_FDFR_RET_SUCCESS	Success
MI_FDFR_RET_IN_IMAGE_ERROR	Fail

#### Requirement

Header files: FDFR.h

Library files: libFDFR\_LinuxC3.so

## MI\_FR\_GetFeatureSizes

### Purpose

To get size of FR data base

### Function Prototype

int MI\_FR\_GetFeatureSizes(void)

### Arguments

Name	Description
None	

### Return value

Return value	Description
size	size of FR data base

### Requirement

Header files: FDFR.h

Library files: libFDFR\_LinuxC3.so

## MI\_FR\_GetFeatureData

### Purpose

To get FR data base

### Function Prototype

MI\_RET MI\_FR\_GetFeatureData(short idx, char\* feat\_data, char\* feat\_name)

### Arguments

Name	Description
idx	index for FR data base
feat_data	array to save feature data
feat_name	array to save feature name

### Return value

Return value	Description
MI_FDFR_RET_SUCCESS	Success
MI_FDFR_RET_FR_GET_FEATURE_DATA_ERROR	Fail

### Requirement

Header files: FDFR.h

Library files: libFDFR\_LinuxC3.so



## MI\_FR\_SetFeatureData

### Purpose

To set FR data base

### Function Prototype

MI\_RET MI\_FR\_SetFeatureData(short idx, char\* feat\_data, char\* feat\_name)

### Arguments

Name	Description
idx	index for FR data base
feat_data	array to save feature data
feat_name	array to save feature name

### Return value

Return value	Description
MI_FDFR_RET_SUCCESS	Success
MI_FDFR_RET_FR_SET_FEATURE_DATA_ERROR	Fail

### Requirement

Header files: FDFR.h

Library files: libFDFR\_LinuxC3.so

## MI\_FR\_EnableFR

### Purpose

To set FR enable

### Function Prototype

MI\_RET MI\_FR\_EnableFR(int enable)

### Arguments

Name	Description
enable	Enable or disable FR

### Return value

Return value	Description
MI_FDFR_RET_SUCCESS	Success
MI_FDFR_RET_FR_ENABLE_ERROR	Fail

### Requirement

Header files: FDFR.h

Library files: libFDFR\_LinuxC3.so

## MI\_FR\_CalFeatureFromRawY

### Purpose

To calculate feature from image

### Function Prototype

MI\_RET MI\_FR\_CalFeatureFromRawY(unsigned char\* imgPtr, int width,int height, int store\_idx)

### Arguments

Name	Description
imgPtr	image
width	image width
height	image height
store_idx	feature index

### Return value

Return value	Description
MI_FDFR_RET_SUCCESS	Success
MI_FDFR_RET_FR_GET_FEATURE_DATA_ERROR	Fail

### Requirement

Header files: FDFR.h

Library files: libFDFR\_LinuxC3.so

## MI\_FR\_CalcImageScore

### Purpose

To get suitable face frame while generating FR database

### Function Prototype

int MI\_FR\_CalcImageScore(unsigned char\* imgPtr, int width, int height, FDFR\_RECT\* prect\_img\_center)

### Arguments

Name	Description
imgPtr	image
width	image width
height	image height
prect_img_center	valid range of face

### Return value

Return value	Description
0	No face detect or more than one face detect
0-70	face is not in valid range
70	face in valid range & 3-axis rotate
80	face in valid range & 2-axis rotate
90	face in valid range & 1-axis rotate
100	face in valid range & 0-axis rotate

**requirement**

Header files: FDFR.h

Library files: libFDFR\_LinuxC3.so

## 4. DATA TYPE

### 4.1. Overview

<a href="#">PFID_FACE_POSITION</a>	Define the face position
<a href="#">PFID_RECT</a>	Define the rectangle of FD
<a href="#">PFID_FACE_DETECT</a>	Define the result of FD.

### 4.2. Struct Lists

#### PFID\_FACE\_POSITION

##### Description

Define the face position.

##### Syntax

```
typedef struct _pfid_face_position {
    signed short    flag;
    signed short    conf;
    signed short    rotate;
    signed short    rect_l;
    signed short    rect_r;
    signed short    rect_t;
    signed short    rect_b;
    signed short    eye_lx;
    signed short    eye_ly;
    signed short    eye_rx;
    signed short    eye_ry;
} PFID_FACE_POSITION;
```

##### Member

Member	Description
flag	Setting flag(PFID_SFLG_*)
conf	Face confidence [low?F0?` high?F100]
rotate	Face rotation flag(PFID_FACE_ROLL_*)
rect_l	Left face frame (X coord) / negative: invalid
rect_r	Right face frame (X coord) / negative: invalid
rect_t	Top face frame (Y coord) / negative: invalid
rect_b	Bottom face frame (Y coord) / negative: invalid

Member	Description
eye_lx	Left eye (X coord) / negative: invalid
eye_ly	Left eye (Y coord) / negative: invalid
eye_rx	Right eye(X coord) / negative: invalid
eye_ry	Right eye(Y coord) / negative: invalid

**Note**

None.

## PFID\_RECT

### Description

Define the rectangle of FD.

### Syntax

```
typedef struct _pfid_rect {
    signed short    sx;
    signed short    sy;
    signed short    ex;
    signed short    ey;
} FDFR_RECT;
```

### Member

Member	Description
sx	Start point of x.
sy	Start point of y.
ex	End point of x.
ey	End point of y.

## PFID\_FACE\_DETECT

### Description

Define the result of FD.

### Syntax

```
typedef struct _pfid_face_detect {
    signed short    num;
    PFID_FACE_POSITION pos[PFID_MAX_FACE_NUM];
    signed char     flg[PFID_MAX_FACE_NUM];
    PFID_RECT       rect;
} PFID_FACE_DETECT;
```

### Member

Member	Description
num	Numbers of detected faces (Max:PFID_MAX_FACE_NUM)
pos[PFID_MAX_FACE_NUM]	Face position
flag	reserved
rect	reserved

#### Note

Don't change the follow values.

```
#define    PFID_MAX_FACE_NUM          (20)    /* Max numbers of faces which can bedetected
*/
```

### 4.3. Enumeration Lists

#### **FDFR\_ROMFILES**

#### Description

Define the romfiles of FDFR

#### Syntax

```
typedef enum {
    FD_ROMFILES = 0,
    FR_ROMFILES
}FDFR_ROMFILES;
```

#### Member

Member	Description
FD_ROMFILES	romfiles of face detection
FR_ROMFILES	romfiles of face recognition

## 5. FACE DETECTION FLOW

---

Call Only Once

fdip\_ctl\_initial

Call Every Frame  
(Y only)

fdip\_ctl\_process



fdip\_getfd\_info  
fdip\_getrecognize\_info



## **6. STEPS FOR GENERATING FACE RECOGNITION DATABASE**

---

1. Registered face image firstly saved in Portable Grayscale Map (PGM) format, and then put the file in /bin/\_IMAGES\_/DB.
2. Edit the file /bin/DB\_list.txt which mainly let FR lib know how many registered files should be read.

## 7. CODE SIZE INFORMATION

---

Code + Data: 2.7 MByte

## 8. DRAM USAGE INFORMATION (WORKING BUFFER)

---

For 640x480 input, FD+FR is around 16MBytes

*Note: It is estimated by real time AIT8428 (600MHz) linux system.*

## 9. CPU MIPS/CLOCK CYCLES ESTIMATION

---

function combination	Image Size	
	320X240	640X480
fd	120 (ms)	440 (ms)
fd+fr	660 (ms)	990 (ms)

**Note: It is estimated by real time AIT8428 (600MHz) linux system.**