Alarm provides the mechanism to schedule activities for the future which are either repeating or are executed in one shot. This section provides information about the alarm APIs and data structures that can be used by user in applications.

# Features and Limitations

Following are the salient features of the alarm implementation:

1. Supports weekly and daily alarms
2. Supports periodic and one short alarms.
3. Provides functions to set system time.
4. Alarm can be deleted.
5. Automatically synchronize the alarm after changing system time.

Following are the limitations:

1. A maximum 6 alarms are supported.

# Header file/s

*Components/alarm/inc/alarm.h*.

# Data Structure Definitions

## alarm\_tm

Store the alarm date and time from user API.

|  |  |
| --- | --- |
| ***tm\_year*** | Year |
| ***tm\_mon*** | Month of the year (1-12) |
| ***tm\_day*** | Day of the month (0-31) |
| ***tm\_hour*** | Hour of the day (0-23) |
| ***tm\_min*** | Minute of the hour (0-59) |
| ***tm\_sec*** | Seconds of minute(0-59) |

Table : alarm\_tm - Data structure definitions

## alarm\_info

Store alarm information. .

|  |  |
| --- | --- |
| ***alarm\_timesec*** | Alarm time in seconds |
| ***alarm\_type*** | Alarm type  0-> Daily alarms  1-> Weekly alarms |
| ***alarm\_id*** | Alarm ID generated after setting the alarm |
| ***alarm\_repeat*** | Repeat the alarm  0-> Once  1-> Repeat |
| ***alarm\_name*** | Alarm name |

Table : alarm\_info - Data structure definitions

# API Reference

## alarm\_init

### Overview

This API initializes the alarm.

### Definition

|  |
| --- |
| void (\* alarm\_callback)(uint32\_t, void \*) |

### Parameters

None.

### Return

Success: 0

Error: 1 (already initialized).

## alarm\_set

### Overview

This function is used to set the alarm with given configuration.

### Definition

|  |
| --- |
| int32\_t  alarm\_set(struct alarm\_tm \*alarm\_ts, uint8\_t alarm\_type, uint8\_t periodic, uint32\_t cb\_func, uint8\_t\* arg) |

### Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *alarm\_ts* | Pointer to alarm\_tm structure. Contains date and time |
| *alarm\_type* | Alarm type |
| *periodic* | Whether the alarm has to be repeated or not |
| *cb\_func* | Alarm callback function. The definition for this function is as follows:   |  | | --- | | void (\* alarm\_callback)(uint32\_t alarm\_id, uint8\_t \*alarm\_name) |   where,  alarm\_id – current alarm ID  alarm\_name – current alarm name |
| *arg* | Call back argument. Used to store alarm name |

Table : alarm\_set - parameter descriptions

### Return

Success: 0

Error:

-1 -> Alarm not initialized.

-2 -> Alarm memory allocation failed

-3 -> Invalid alarm type

-4 -> Invalid time

-5 -> Max alarms present

## alarm\_set\_time

### Overview

Set system time.

### Definition

|  |
| --- |
| void  alarm\_set\_time(uint64\_t time\_toset) |

### Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *time\_tosett* | Time to be set in seconds. |

Table : alarm\_set\_time - parameter descriptions

### Return

None.

## alarm\_delete

### Overview

Delete the alarm.

### Definition

|  |
| --- |
| int8\_t  alarm\_delete(uint32\_t alarm\_id) |

### Parameters

|  |  |
| --- | --- |
| **Parameters** | **Description** |
| *alarm\_id* | Alarm ID. This is obtained after setting the alarm. |

Table : alarm\_delete - parameter description

### Return

Success: 0

Error: 1

## alarm\_display

### Overview

Display all alarms. User defined call back will be triggered for each alarm with the alarm information.

### Definition

|  |
| --- |
| void alarm\_display(uint32\_t cb) |

### Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *cb* | Call back to trigger alarm display. The definition for this callback is as follows:   |  | | --- | | typedef void (\* alarm\_dsiplay\_callback)(struct alarm\_infio \*ainfo) |   where,  ainfo – pointer to alarm to alarm\_info structure, which contains the information for the current alarm. |

Table : alarm\_display - parameter description

### Return

None.

## alarm\_info\_get

### Overview

Get information about a specific alarm.

### Definition

|  |
| --- |
| struct alarm\_info \* alarm\_info\_get(uint32\_t alarm\_id) |

### Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *alarm\_id* | Alarm ID |

### Return

Success: Return valid pointer to alarm\_info structure.

Error: NULL.

# Example Application

For the example codes, refer: apps\alarm\alarm\_test.c application.