M190 Module

Library Development Guide

V1.00

EasyLinkIn

Sensing the World

CONTENTS

Revision History

Edition	Revision Date	Revision history	Editor
V1.00	2022.04.22	New create	SBX

1. Document Description

1.1. Document Function

This document describe the use and precautions of secondary development library for customers.

1.2. Document Object

Technicians who use libraries for secondary development.

1.3. Document Convention

Except for custom types, Other types are defined in two public header files shown in following table.

stdint.h	uint8_t uint16_t uint32_t int8_t int16_t int32_t
stdbool.h	bool

1.4. Development Environment

Lib library files are obtained in Keil integrated development, customers need to use the same environment for development.

2. Library Structure

2.1. Packaged Components

Resources	Description
lorawan stack	Lorawan protocol stack
loratimer	High precision timer, used for lorawan receive window timing
lorawan app core	Lorawan core application
env manager	Environment variable management unit
contiki os	Basic job-scheduling system

3. Library Usage Resources

This section describes the MCU resources have been used in library. System exceptions may occur if customers reconfigure these resources.

3.1. Resources Usage of Internal Chip

Recourse	Description
HSI	Master clock, HS116 fixed, modification is not allowed
lptimer1	
lptimer2	
rtc alarm/wakeup	Library long time timer, software timer will be provided to customer
radio gpio	PB2 PB12
flash	Flash Programming, used for saving parameters
wdt	Watch dog
rng	Nonce
systick	Running in free counting mode, can be read, can not be reconfigured

4. Resources Provided by Library

4.1. Software Timer

Software timer	Description
etimer	Event timer, maximum timing precision: 8ms
ctimer	Callback timer, maximum timing precision:8ms
rtc_timer	Long time timer， timer precision:1s

Note: the timers mentioned in above table are non-real-time software timers. Affected by the current operating load and interrupt condition, the maximum timing precision is 1 tick(8ms), which is not suitable for ultra-high time precision environment.

4.2. Contiki OS Scheduler

Contiki is a small,open source ,easily portable multi-task OS. It is designed for a memory-constrained network systems, including embedded systems with 8-git microprocessors.

Please check the relevant information for details.

5. How to Use Library

5.1. Simplest Demo

Call two library function can initiate the system. It is best not to modify the code that initialize the clock to 24M.Otherwise an exception may occur. If you need modification ,please contact the technical personnal.

You only need to call two library functions to start the system. It is best not to modify the code that initialates the clock to 24M, otherwise an exception may occur. If you need to modify, please contact the technical personnel.

```
int_t main(void) {
        SystemClkInit_RCH(SysctrlRchFreq24MHz);
        eli_lib_init();
        // Start user processed
        process_start(&user_process, NULL);
        while(1){
            lib_loop_main();
          }
    }
```

For detailed, please see software demo.

6. Library Usage Precautions

6.1. Rules of use

As Contiki is a collaborative scheduling system and does not support preemption, please keep the while loop is unblocked and the blocking time is no longer than 1 tick (8ms) to ensure normal task scheduling in the system.

6.2. Suggested Use

- Minimize interruption use. If use interruption, please keep the interruption as short as possible. The core task of lorawan requires precise timing, frequent interruption may cause abnormal packet loss.
- Refers to the format in demo code to split the task into threads, and the task scheduling is more uniform and reasonable
- When executing long time task, split the task into steps and allocate CPU time to ensure that the other tasks no not get stuck.

6.3. Use Skill

```
PROCESS_THREAD(led_process, ev, data)
{
    static struct etimer et;

    PROCESS_EXITHANDLER(goto EXIT);
    PROCESS_BEGIN();
    while (1) {
        PROCESS_WAIT_EVENT_UNTIL(ev == PROCESS_EVENT_CONTINUE);
        led_blink_st *led = (led_blink_st *)data;
        etimer_set(&et, MS2TICK(led->on_ms));
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        LED_OFF();
    }
EXIT:
    data = data;
    PROCESS_END();
}
```

6.3.1. Make good use of PROCESS_WAIT_EVENT_UNTIL blocking to wait for the task to be triggered.

6.3.2. Make good use of etimer_set to configure delayed waiting , during the delayed time, allocate CPU to avoid stuck.

Appendix

Contiki website：
http://www.contiki-os.org/

Lorawan Alliance website：
http://lora-alliance.org/about-lorawan/