HCRTOS watchdog timer userguide

1. 文档履历

版本号	日期	制/修订人	制/修订记录
1.0	2023.2.16	邱浩佳	新增文档说明

HCRTOS watchdog timer userguide

---

2. 概述

2.1 编写目的

介绍watchdog timer的功能和指导开发

2.2 读者对象

软件开发工程师和技术支持工程师

3. 模块介绍

watchdog timer有两种模式，可通过ioctl进行配置为不同模式。

1. watchdog模式，超时或达到计数值会产生复位中断；watchdog模式提供自动喂狗功能，需要在配置中打开；
2. timer模式，超时或达到计数值会超时timer中断；
3. 所有板子的默认配置不开启watchdog驱动；
4. watchdog timer当前最大计时为4,000,000,000us；

！！！在自动喂狗模式下，串口console会提供start和stop watchdog两个命令，注意，在开启自动喂狗模式下，用jtag进行debug时，需要先停止watchdog，不然在os被停止状态下，不会进行喂狗，watchdog会超时，从而导致芯片复位！！！

3.1 模块配置

3.2 看门狗模式

在sdk根路径下输入：make menuconfig，根据下面路径选中watchdog模式。

```
There is no help available for this option.|
  | Symbol: CONFIG_WDT_MODE_WATCHDOG [=y]|
  | Type  : bool|
  | Prompt: watchdog mode|
  |   Location:|
  |     -> Components|
  |       -> kernel (BR2_PACKAGE_KERNEL [=y])|
  |         -> Drivers|
  |           -> watchdog (CONFIG_DRV_WDT [=y])|
  |             -> watchdog timer running mode (<choice> [=y])|
  |   Defined at watchdog:5|
  |   Depends on: <choice>
    ------------------------------------------------------------------------------
-------

                       --- watchdog
                       watchdog timer running mode (watchdog mode)  --->
                       [ ]   auto feed watchdog timer in idle task
                       (10000) default timeout (in millisecond) of watchdog
timer
```

输入：make kernel-rebuild all，编译选中的驱动。

3.3 自动喂狗

在sdk根路径下输入：make menuconfig，根据下面路径选中auto feed watchdog timer in idle task。

```
There is no help available for this option.|
  | Symbol: CONFIG_WDT_AUTO_FEED [=y]|
  | Type  : bool|
  | Prompt: auto feed watchdog timer in idle task|
  |   Location:|
  |     -> Components|
  |       -> kernel (BR2_PACKAGE_KERNEL [=y])|
  |         -> Drivers|
  |           -> watchdog (CONFIG_DRV_WDT [=y])|
  |   Defined at watchdog:12|
  |   Depends on: BR2_PACKAGE_KERNEL [=y] && CONFIG_DRV_WDT [=y] && CONFIG_WDT_MODE_WATCHDOG [=y]

  ------------------------------------------------------------------------------

                       --- watchdog
                            watchdog timer running mode (watchdog mode)  -->
                       [*]    auto feed watchdog timer in idle task
                       (10000) default timeout (in millisecond) of watchdog timer
```

times(millisecond) of watchdog timer：该时间可选范围为1-120000毫秒，默认为10000毫秒，超过时间还未进行喂狗，则芯片会进行复位。

编译指令：make kernel-rebuild all

3.4 看门狗模式下串口控制台命令

```
hc1600a@dbc3000v10(wdt)# help

Commands available:
  help                    Show available cmds
  exit                    Exit from current cmd set
  history                 Show history cmds
  settimeout     set watchdog timeout
  start                   start watchdog
  stop                    stop watchdog
   status                  get watchdog status
hc1600a@dbc3000v10(wdt)# settimeout -h
-----------------------------------
wdt settimeout cmds help
for example : settimeout -t 10000
't'   10000 means 10s
settimeout range is 1ms-120s
-----------------------------------
hc1600a@dbc3000v10(wdt)# settimeout -t100
```

3.5 timer模式

在sdk根路径下输入：make menuconfig，根据下面路径选中timer mode。

```
There is no help available for this option.|
  | Symbol: CONFIG_WDT_MODE_TIMER [=y]|
  | Type  : bool|
  | Prompt: timer mode|
  |   Location:|
  |     -> Components|
  |       -> kernel (BR2_PACKAGE_KERNEL [=y])|
  |         -> Drivers|
  |           -> watchdog (CONFIG_DRV_WDT [=y])|
  |             -> watchdog timer running mode (<choice> [=y])|
  |   Defined at watchdog:8|
  |   Depends on: <choice>
    --------------------------------------------------------------------------
-------

                       --- watchdog
                             watchdog timer running mode (timer mode)  --->

                       [*]   auto feed watchdog timer in idle task
                       (10000) default timeout (in millisecond) of watchdog
timer
```

编译指令：make kernel-rebuild all；timer模式可以通过ioctl命令进行配置。

4. 模块接口说明

介绍本模块相关的API接口说明

4.1 应用层使用

通过open /dev/watchdog 节点，然后就可以使用ioctl，用不同的命令对watchdog timer进行操作。

- 开启watchdog timer

```
ioctl(fd, WDIOC_START, 0);
```

- 关闭watchdog timer

```
ioctl(fd, WDIOC_STOP, 0);
```

- 重置watchdog timer计数器的值为设置的值

```
ioctl(fd, WDIOC_KEEPALIVE, 0);
```

- 设置watchdog timer模式，有WDT_MODE_TIMER 和WDT_MODE_WATCHDOG两个可选。

```
ioctl(fd, WDIOC_SETMODE, WDT_MODE_WATCHDOG);  //设置为watchdog模式

ioctl(fd, WDIOC_SETMODE, WDT_MODE_TIMER);     //设置为timer模式
```

- 设置watchdog timer的定时值，以us为单位

```
uint32_t watchdog_timeout = 1000000; //us
ioctl(fd, WDIOC_SETTIMEOUT, (uint32_t)watchdog_timeout)
```

- 获取watchdog timer的定时值以us为单位

```
uint32_t watchdog_timeout;
ioctl(fd, WDIOC_GETTIMEOUT, (uint32_t)&watchdog_status);
```

- 获取watchdog timer的剩余时间，以us为单位

```
uint32_t watchdog_timeout;
ioctl(fd, WDIOC_GETTIMELEFT, (uint32_t)&watchdog_status);
```

5. 模块测试用例与Sample Code

参考程序路径：components/cmds/source/watchdog/watchdog_test.c

```
#include <stdint.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <getopt.h>
#include <fcntl.h>
#include <sys/ioctl.h>
#include <kernel/delay.h>
#include <kernel/lib/console.h>

#include <freertos/FreeRTOS.h>
#include <freertos/task.h>
#include <freertos/semphr.h>
#include <freertos/queue.h>
#include <kernel/lib/console.h>

#include <nuttx/wqueue.h>
#include <hcuapi/iocbase.h>
#include <hcuapi/watchdog.h>

#define WATCHDOG_TIMEOUT 400000

static const char *device = "/dev/watchdog";

static void notify_watchdog_call(void *arg, unsigned long param)
{
    printf("%s:%d:receive watchdog timer notify\n", __func__, __LINE__);
```

```c

    return ;
}

struct work_notifier_s notify_watchdog;

int watchdog_test(int argc, char * argv[])
{
    int ret = 0;
    int fd;
    uint32_t watchdog_value = WATCHDOG_TIMEOUT;

    notify_watchdog.evtype = WDIOC_NOTIFY_TIMEOUT;
    notify_watchdog.qid = HPWORK;
    notify_watchdog.remote = false;
    notify_watchdog.oneshot = false;
    notify_watchdog.qualifier = NULL;
    notify_watchdog.arg = NULL;
    notify_watchdog.worker2 = notify_watchdog_call;
    work_notifier_setup(&notify_watchdog);

    fd = open(device, O_RDWR);
    if (fd < 0) {
        printf("can't open %s\n",device);
        return -1;
    }

    ret = ioctl(fd, WDIOC_SETTIMEOUT, (uint32_t)watchdog_value);
    if (!ret) {
        printf("%d set watchdog timer timeout = %ld\n",__LINE__, watchdog_value);
    }

    ret = ioctl(fd, WDIOC_GETTIMEOUT, (uint32_t)&watchdog_value);
    if (!ret) {
        printf("%d get watchdog timer timeout = %ld\n",__LINE__, watchdog_value);
    }

    ret = ioctl(fd, WDIOC_SETMODE, WDT_MODE_TIMER );
    if (!ret) {
        printf("%d set watchdog timer mode to timer\n",__LINE__);
    }

    ret = ioctl(fd, WDIOC_START, 0);
    if (!ret) {
        printf("%d start watchdog timer\n",__LINE__);
    }

    usleep(WATCHDOG_TIMEOUT + 1000000);

    ret = ioctl(fd, WDIOC_GETTIMELEFT, (uint32_t)&watchdog_value);
    if (!ret) {
        printf("%d get watchdog timer residual value = %ld\n",__LINE__, watchdog_value);
    }

    ret = ioctl(fd, WDIOC_GETTIMELEFT, (uint32_t)&watchdog_value);
    if (!ret) {
        printf("%d get watchdog timer residual value = %ld\n",__LINE__, watchdog_value);
    }

    ret = ioctl(fd, WDIOC_STOP, 0);
    if (!ret) {
            printf("%d stop watchdog timer\n",__LINE__);
    }

    ret = ioctl(fd, WDIOC_KEEPALIVE, 0);
    if (!ret) {
        printf("%d reset watchdog timer timerout value\n",__LINE__);
    }

    usleep(WATCHDOG_TIMEOUT / 2);

    ret = ioctl(fd, WDIOC_GETTIMELEFT, (uint32_t)&watchdog_value);
    if (!ret) {
        printf("%d get watchdog timer residual value = %ld\n",__LINE__, watchdog_value);
    }

    close(fd);

    return ret;
}

CONSOLE_CMD(watchdog_test,NULL,watchdog_test,CONSOLE_CMD_MODE_SELF,"test watchdog function app")

```

6. 模块调试方法

watchdog 模式下，提供串口控制台的命令进行debug。

7. 常见问题

Q：在开启看门狗后怎么获取看门狗状态?

A：通过串口控制台的命令。

Q：开启看门狗后，在用jtag进行debug一半时，出现GDB无法读取板子信息。

A：应该先关闭看门狗，再进行的jtag，具体参考watchdog debug注意事项