

32-bit microcontrollers

Stop Mode Precautions for HC 32F460 Series

Applicable objects

F Series	HC 32F460
-----------------	-----------

Table of Contents Table of Contents

1	Abstract	3
2	Stop Mode of HC32F460 Series	3
	2.1 Introduction.....	3
	2.2 Register Introduction.....	3
	2.3 Operation flow	4
	2.3.1 Enter stop mode	4
	2.3.2 Release stop mode	4
	2.4 Cautions.....	6
3	Sample Code	7
	3.1 Code Introduction.....	7
	3.2 Code Run.....	9
4	Version Information & Contact	11

1 Abstract

This application note introduces the stop mode notes of HC32F460 series chips.

2 Stop Mode of HC32F460 Series

2.1 Introduction

Stop (STOP) mode is one of the three low-power modes of the HC32F460 series chips. In STOP mode, the CPU, most peripherals and clock sources are stopped, and the chip maintains the CPU internal registers and SRAM data, peripheral status and pin status.

2.2 Register Introduction

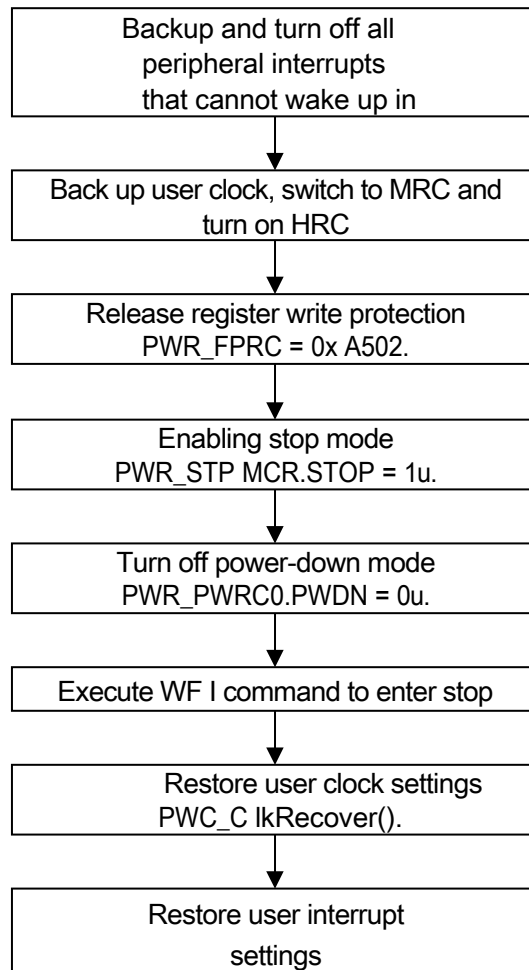
- 1) PWR_PWRC1: Power mode control register, set the drive capability when different modes go in STOP mode.
- 2) PWR_STPMCR: Stop mode controller to enable STOP mode and set the CLK and FLASH states when STOP mode wakes up.
- 3) INT_WUPEN: Stop mode wake-up event

enable register. Caution:

- Before entering stop mode, you must set INT_WUPEN to enable the corresponding wake-up event, otherwise you cannot wake up the stop mode.

2.3 Operation process

2.3.1 Enter stop mode



The diagram above lists the steps to enter the stop mode, in which both the peripherals and the CPU of the chip stop working.

Note: When the system clock is HRC and MPL, you need to switch the system clock source to MRC, other clock sources just need to turn on HRC.

2.3.2 Release stop mode

Stop mode can be released by resetting and interrupting. The resets that can be used to disengage the stop mode are

- Pin Reset

- Power-on reset
- Undervoltage Reset (BOR)

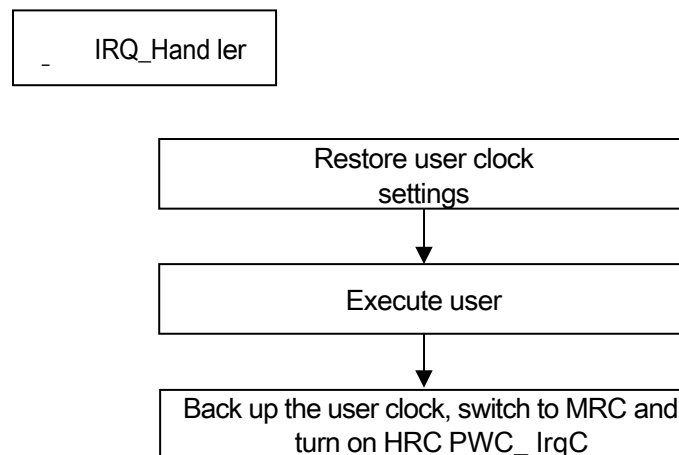
- Voltage detection 1/2 reset
- Dedicated watchdog reset

The interrupt events that can be used to release the stop mode are

- Pin interrupt EIRQ0~15
- Voltage detection 1/2 interrupt
- Dedicated watchdog underflow interrupt
- Periodic interrupt of the real time clock
- Alarm clock interruption
- Wake-up timer interrupt
- Comparator 1 Interrupt
- UART1_RXD0 Interrupt
- TIMER01_A Compare

Match Interrupt Attention:

- To select the interrupt event unstop mode, you need to operate the INT_WUPEN register to set the corresponding interrupt event wake-up permission before entering the stop mode.



The diagram above illustrates the steps to wake up the stop mode.

2.4 Cautions

- 1) Before executing WFI into stop mode, make sure that the FLASH is not programmed or erased (i.e. EFM_FSR.RDY=1), and the oscillation stop monitoring function is invalid, otherwise the chip cannot enter the stop mode.
- 2) Before executing WFI into stop mode, make sure that the DMA is in stop state (i.e. DMA_EN.EN=0), otherwise the chip may act in an unguaranteed way.
- 3) Before executing WFI into stop mode, the digital filtering of EIRQ must be set to invalid, otherwise the interrupt cannot be used for STOP wakes up.
- 4) Select the interrupt event to release the stop mode and enable the corresponding interrupt event to wake up the stop mode (INT_WUPEN) before executing the WFI into the stop mode.
- 5) Before executing WFI into stop mode, make sure that all other peripheral interrupts (non-STOP mode wakeup interrupts) are turned off. Otherwise, the triggering of other interrupts may lead to unguaranteed actions of the chip. After waking up, restore other peripheral interrupts to avoid missing interrupt events.
- 6) Before executing WFI into stop mode, you must ensure that HRC oscillates, and if the system clock is HRC and MPLL, you also need to switch the system clock to MRC, and then restore the system clock after waking up to restore the HRC state.
- 7) To resume the MCU from stop mode by an interrupt event, restore the system from MRC to the user clock setting and restore the interrupt configuration
 - Points 4, 5, 6 and 7 can be achieved by turning off the corresponding registers of the NVIC and clock module by yourself or by calling the API interface provided by HDSC:
 - `en_result_t enIntWakeupEnable (uint32_t u32WakeupSrc);` (Enables interrupt event wakeup stop mode)
 - `void PWC_EnterStopMd (void);` (Enter stop mode)

This function takes up about 100us@168MHz

extra

- void **PWC_IrqClkBackup** (void); (Backup user clock settings) This function takes about 50us@168MHz extra
- void **PWC_IrqClkRecover** (void) (Restore user clock settings) This function takes about 50us@168MHz extra

3 Sample Code

3.1 Code Introduction

Users can write their own code to learn and verify the module according to the above operation flow and notes, or download the sample code of Device Driver Library (DDL) directly through the website of UW Semiconductors and use the sample of stop mode in the LPM to verify.

The following section briefly describes the configuration involved in this sample DDL-based LPM module `lpm_stop_wkup` code.

1) LED and PORT initialization:

This sample uses an external pin interrupt to wake up the stop mode, so you need to initialize the port and make sure the corresponding port interrupt is enabled.

```
Led_Init();  
Port_Init();
```

2) STOP mode configuration:

```
/* Config stop mode. */  
stcPwcStopCfg.enStpDrvAbi = StopHighspeed;  
stcPwcStopCfg.enStopClk = ClkFix;  
stcPwcStopCfg.enStopFlash = Wait; PWC_  
StopModeCfg(&stcPwcStopCfg).
```

3) Interrupt configuration:

This sample uses the external pin 0 interrupt, which is valid on the rising edge and **invalid for digital filtering**.

```
/* EIRQ0 config. */  
stcExintCfg.enExitCh = ExtiCh00;  
stcExintCfg.enFilterEn = Disable;  
stcExintCfg.enExtiLvl = ExIntRisingEdge;  
EXINT_Init(&stcExintCfg).  
  
/* Register EIRQ0.*/  
stcPortIrqCfg.enIntSrc = INT_PORT_EIRQ0;  
stcPortIrqCfg.enIRQn = PORT_IRQn;  
stcPortIrqCfg.pfnCallback = Port_Handle;  
enIrqRegistration(&stcPortIrqCfg).
```

4) Set STOP mode interrupt wake-up source

```
/* Set wake up source EIRQ0. */  
enIntWakeupEnable(Extint0WU).
```

5) Enables wake-up source interrupts:

```
/* Enable EIRQ. */  
enIntEnable(Int0).  
NVIC_ClearPendingIRQ(PORT_IRQn);  
NVIC_SetPriority(PORT_IRQn,DDL_IRQ_PRIORITY_DEFAULT).  
NVIC_EnableIRQ(PORT_IRQn).
```

6) Peripheral status before entering STOP mode:

Ensure that FLASH is not programmed or erased, that DMA is stopped, and that other peripheral interrupts are turned off.

```
/* Ensure DMA disable */  
while((0 != M4_DMA1 ->EN_f.EN) && ((0 != M4_DMA2 ->EN_f.EN))).  
/* Ensure FLASH is ready */  
while(1 != M4_EFM ->FSR_f.RDY).  
  
PWC_EnterStopMd().
```

7) Post-wakeup interrupt processing flow:

```
IRQ_Handler.  
  
PWC_IrqClkRecover(); // in pairs use  
LED0_TOGGLE().      // user  
application Ddl_Delay1ms(1000).  
PWC_IrqClkBackup().  // in pairs use
```

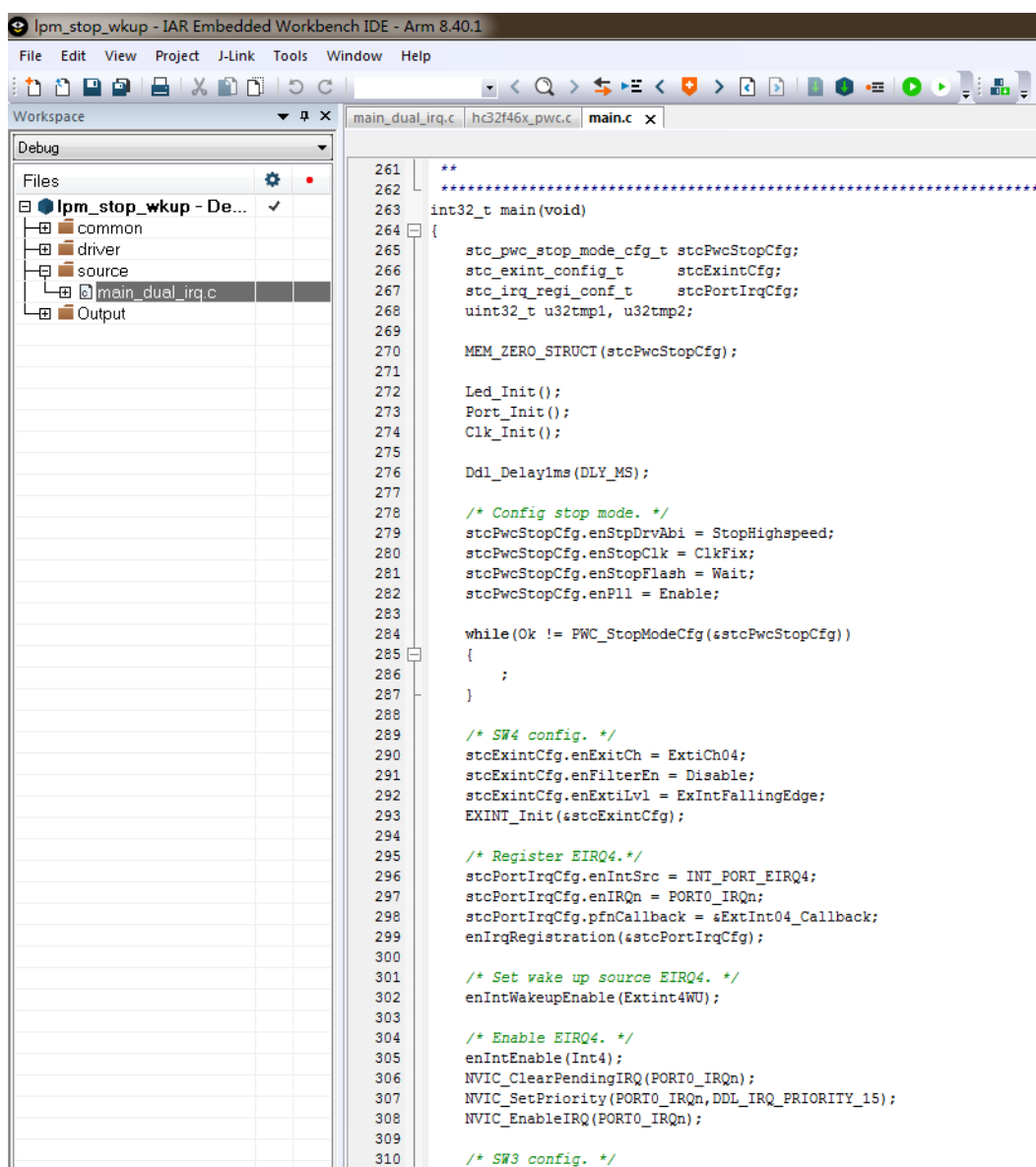
Interrupts woken up in non-stop mode do not need to be processed for clock-related configuration.

3.2 Code Run

Users can download the sample code (lpm_stop_wkup) of the HC32F460 DDL from the UW website and run the code with the evaluation board (EV-HC32F460-LQFP100-050-V1.1) to learn to use STOP mode.

The following section describes how to run the STOP sample code on the evaluation board and observe the results:



- Verify that the correct IAR EWARM v7.7 tool is installed (please download the appropriate installation package from the official IAR website and refer to the user manual for installation).
- Download the HC32F460 DDL code from the UW Semiconductors website.
- Download and run the project file in lpm\lpm_stop_wkup \:
- 1) Open the lpm_stop_wkup \ project and open the 'main.c' view as follows:



```

261  /**
262  ****
263  int32_t main(void)
264  {
265      stc_pwc_stop_mode_cfg_t stcPwcStopCfg;
266      stc_extint_config_t      stcExtIntCfg;
267      stc_irq_regi_conf_t      stcPortIrqCfg;
268      uint32_t u32tmp1, u32tmp2;
269
270      MEM_ZERO_STRUCT(stcPwcStopCfg);
271
272      Led_Init();
273      Port_Init();
274      Clk_Init();
275
276      Ddl_Delay1ms(DLY_MS);
277
278      /* Config stop mode. */
279      stcPwcStopCfg.enStpDrvAbi = StopHighspeed;
280      stcPwcStopCfg.enStopClk = ClkFix;
281      stcPwcStopCfg.enStopFlash = Wait;
282      stcPwcStopCfg.enPll = Enable;
283
284      while(Ok != PWC_StopModeCfg(&stcPwcStopCfg))
285      {
286          ;
287      }
288
289      /* SW4 config. */
290      stcExtIntCfg.enExitCh = ExtiCh04;
291      stcExtIntCfg.enFilterEn = Disable;
292      stcExtIntCfg.enExtiLvl = ExIntFallingEdge;
293      EXINT_Init(&stcExtIntCfg);
294
295      /* Register EIRQ4. */
296      stcPortIrqCfg.enIntSrc = INT_PORT_EIRQ4;
297      stcPortIrqCfg.enIRqn = PORT0_IRQn;
298      stcPortIrqCfg.pfnCallback = &ExtInt04_Callback;
299      enIrqRegistration(&stcPortIrqCfg);
300
301      /* Set wake up source EIRQ4. */
302      enIntWakeUpEnable(Extint4WU);
303
304      /* Enable EIRQ4. */
305      enIntEnable(Int4);
306      NVIC_ClearPendingIRQ(PORT0_IRQn);
307      NVIC_SetPriority(PORT0_IRQn, DDL_IRQ_PRIORITY_15);
308      NVIC_EnableIRQ(PORT0_IRQn);
309
310      /* SW3 config. */

```

- 2) Click  to recompile the entire project.
- 3) Click  Download the code to the evaluation board and run it at full speed.
- 4) Press SW2 to put the chip into stop mode.
- 5) Press SW4, the chip is woken up and the red light flashes. The chip continues to enter stop mode.
- 6) Press SW3, the chip is woken up and the green light flashes. The chip continues to enter stop mode.

The clock, red light and green light waveforms can be observed on the oscilloscope, where SW3 interrupt priority is higher than SW4, during the execution of SW4 interrupt, SW3 interrupt will be triggered and SW4 interrupt will be preempted, at which time the red light and green light can be observed to be on at the same time, and then the green light and red light will be off one after another.

4 Version Information & Contact

Date	Versions	Modify records
2019/3/20	Rev1.0	Initial Release
2020/8/26	Rev1.1	Add instructions for clock switching into and out of stop mode; update supported models



If you have any comments or suggestions in the process of purchase and use, please feel free to contact us.

Email: mcu@hdsc.com.cn

Website: <http://www.hdsc.com.cn/mcu.htm>

Address: 10/F, Block A, 1867 Zhongke Road, Pudong

New Area, Shanghai, 201203, P.R. China

