



32-bit Microcontrollers

Interrupt Controller INTC for HC 32 F 460 Series

Applicable objects

Series	Product Model
HC32F460	HC32F460JEUA
	HC32F460JETA
	HC32F460KEUA
	HC32F460KETA
	HC32F460PETB

Table of Contents Table of Contents

1	Abstract	3
2	INTC Introduction	3
3	INTC for HC32F460 Series	4
3.1	NMI non-maskable interrupts.....	4
3.1.1	NMI Key Features	4
3.1.2	NMI Register Description.....	4
3.1.3	NMI Configuration Process Description	5
3.2	External pin interrupt.....	6
3.2.1	External pin interrupt register.....	6
3.2.2	External pin interrupt configuration flow description.....	6
3.3	Software Interruptions	7
3.4	Interrupt source selection.....	7
3.4.1	Interrupt selection register.....	7
3.4.2	Description of interrupt selection register method.....	8
4	Sample Code	9
4.1	Code Introduction	9
4.2	Code Run.....	11
5	Summary	12
6	Version Information & Contact	13

1 Abstract

This application note introduces the Interrupt Controller (INTC) module of HC32F460 series chips, and briefly explains how to use INTC module by showing the sample code of external pin interrupt, software interrupt and NMI interrupt.

2 INTC Introduction

The HC32F460 series interrupt controller (INTC) module is rich in features, including the ability to mask non-maskable interrupts.

(NMI), external pin interrupt (EXINT), software interrupt (SWI), interrupt, event enable configuration; peripheral module interrupt source is freely set to any entry except system interrupt vector entry.

INTC Key Features:

- Peripheral interrupt vector entry interrupt source is configurable
- 16 programmable interrupt priority levels
- Multiple selectable NMI interrupt sources
- 16 external pin interrupts
- 32 software interrupts
- System sleep mode wake-up source configuration
- System stop mode wake-up source configuration
- Support wake up after WFI, WFE

3 INTC for HC32F460 Series

3.1 NMI non-maskable interrupts

The Non-Maskable Interrupt (NMI) has the highest priority. The NMI of HC32F460 series can select multiple interrupt event requests, and the application can determine the source of the NMI interrupt by querying the NMIFR register and clear the corresponding flag bit by using the NMICFR register.

3.1.1 NMI Key Features

- Multiple interrupt requests can be selected as NMI signal sources:
 - NMI Pin Interrupt
 - External high-speed XTAL oscillation stop interrupt
 - External low-speed XTAL 32 oscillation stop interrupt
 - WDT underflow, refresh error interrupt
 - SWDT underflow, refresh error interrupt
 - Low voltage monitoring PVD1 interrupt
 - Low voltage monitoring PVD2 interrupt
 - SRAM parity error interrupt
 - SRAM ECC checksum error interrupt
 - MPU bus error interrupt
- NMI pin interrupt digital filtering function and filter clock can be set
- NMI pin interrupt rising edge, falling edge triggering

3.1.2 NMI Register Description

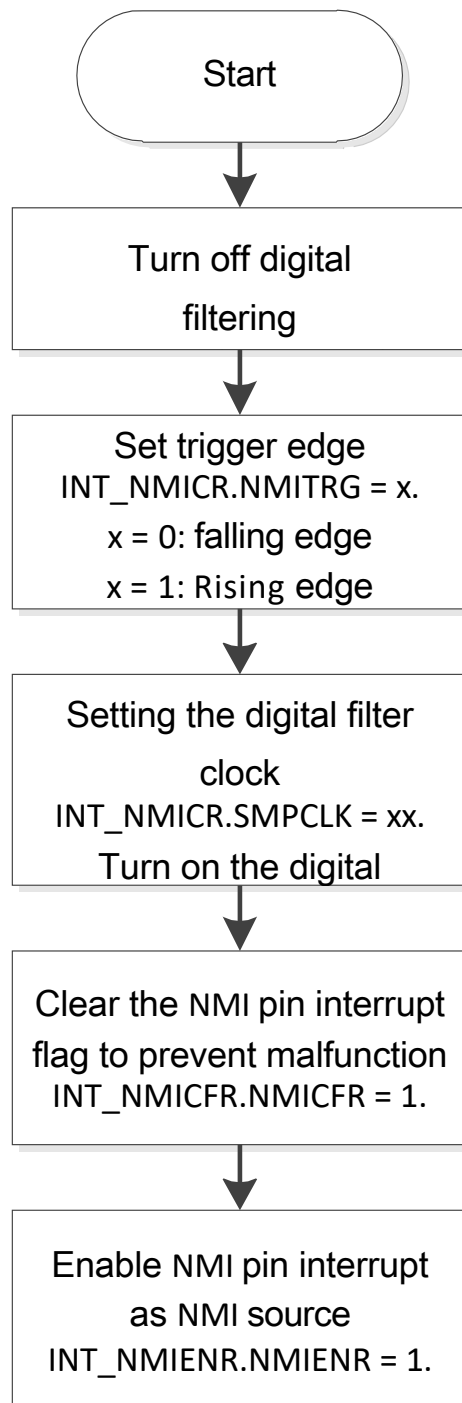
English description (abbreviation)	Chinese Description tion
NMI Control Register(INT_NMICR)	NMI pin non-maskable interrupt control register
NMI Enable Register (INT_NMIENR)	Non-maskable interrupt enable register
NMI Flag Register (INT_NMIFR)	Non-maskable interrupt flag register

NMI Clear Flag Register (INT_NMICFR)

Non-maskable interrupt flag clear register

3.1.3 NMI configuration process description

The following flowchart gives the configuration flow using the NMI pin interrupt as an example.



If you want to use the NMI pin interrupt function, you need to configure the interrupt selection register and the NVIC section, which will be explained in detail in the **interrupt source selection** section.

3.2 External pin interrupts

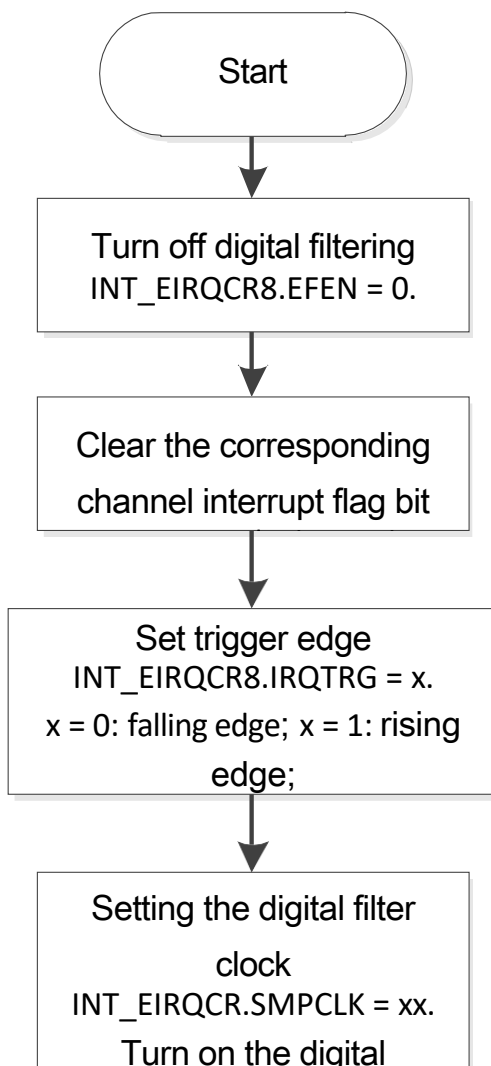
The HC32F460 series has 16 external pin interrupt events, and the attributes of each channel can be configured via registers, including digital filtering function, trigger level selection.

3.2.1 External pin interrupt register

English description (abbreviation)	Chinese Description
External Interrupt Control Register(INT_EIRQCRx), x = 0~15	NMI pin non-maskable interrupt control register
External Interrupt Flag Register (INT_EIFR)	External pin interrupt flag register
External Interrupt Clear Flag Register (INT_EICFR)	External pin interrupt flag clear register

3.2.2 External pin interrupt configuration flow description

The following flowchart gives the configuration flow using the external pin interrupt 8 as an example.



If you want to use the external pin interrupt function, you need to configure the pin as external interrupt enable (please refer to the General IO section in Chapter 11 of the chip manual to set `PCRxy.INTE = 1;`); in addition, please refer to the **Interrupt Source Selection** section of this document to configure the interrupt selection register and the NVIC section.

3.3 Software Interruptions

The HC32F460 series has 32 software interrupt requests, and they correspond to interrupt vectors 0~31 one by one, which can be set by the software reset register.

The corresponding bit of `INT_SWIER` is used to generate a software interrupt event request.

3.4 Interrupt source selection

The HC32F460 series has 16 system interrupt vector entries and 144 peripheral interrupt vector entries, which can be accessed through the interrupt sources.

Select registers to configure interrupt requests from the chip's 239 peripherals to 144 interrupt vector entries for flexible management of interrupt service routines.

3.4.1 Interrupt selection register

English description (abbreviation)	Chinese Description
Interrupt Select Register(<code>INT_SELx</code>), $x = 0 \sim 31$, total 32	Interrupt selection register, all interrupt event requests can correspond to
Interrupt Select Register (<code>INT_SELy</code>), $y = 32 \sim 127$, 96 in total	Interrupt selection registers, divided into 16 groups of 6 each, are used to The event request number 32 corresponds to the module.
Interrupt Vector Share Select Register (<code>INT_VSSELz</code>), $z = 128 \sim 143$, total 16	Interrupt vector shared selection register, each bit corresponds to an interrupt event request, please refer to the chip manual for the specific correspondence 12.3.2 Interrupt event sequence section

3.4.2 Description of the interrupt selection register method

The following is an example of the interrupt selection register, also using the external pin interrupt 8.

Before configuring the interrupt selection register, consult the 12.3.2 Interrupt Event Request Sequence Number section of the chip manual to obtain the interrupt event sequence number you want to configure, and the following figure is extracted from this section of the manual.

编号	中断事件请求序号	功能	功能名称	是否可选择为中断	可否选择为AOS触发源	对应NVIC向量的中断选择寄存器 ¹⁾		
						NVIC向量 0~31	NVIC向量 32~127	NVIC向量 128~143
0	000h	PORT	PORT_EIRQ0	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[0]
1	001h		PORT_EIRQ1	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[1]
2	002h		PORT_EIRQ2	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[2]
3	003h		PORT_EIRQ3	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[3]
4	004h		PORT_EIRQ4	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[4]
5	005h		PORT_EIRQ5	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[5]
6	006h		PORT_EIRQ6	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[6]
7	007h		PORT_EIRQ7	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[7]
8	008h		PORT_EIRQ8	√	√	INT_SEL0~31	INT_SEL32~37	INT_VSSEL128[8]

From the above table, we can see that the serial number of external pin interrupt 8 is 8, and the available interrupt selection registers are INT_SEL0~31, INT_SEL32~37, INT_VSSEL128[8].

If you set INT_SEL10 = 8; the program will respond to interrupt vector 8 when external pin 8 is generated; if you set

INT_VSSEL 128[8] = 1; The program will respond to interrupt vector number 128. If the same interrupt request number is set to multiple interrupt selection registers at the same time, when this interrupt request comes, if the same interrupt priority is set, the program will respond according to the interrupt vector number, from the smallest to the largest, until all the configured interrupt selection registers are executed.

4 Sample Code

4.1 Code Introduction

Users can write their own code to learn to verify the module according to the above workflow, or download the sample code of Device Driver Library (DDL) directly from the website of UW Semiconductors and use the sample of INTC to verify.

The following section briefly describes the configuration involved in this sample AN DDL-based INTC module `exint_nmi_swi` code.

- 1) Set the NMI pin interrupt initialization structure variable:

```

/*****
/* NMI Pin */
/*****
*****/
/* Filter setting */
stcNmiConfig.enFilterEn = Enable;
stcNmiConfig.enFilterClk = Pclk3Div8.
/* Falling edge */
/* Callback function */
stcNmiConfig.pfnNmiCallback = Nmi_IrqCallback;
stcNmiConfig.u16NmiSrc = NmiSrcNmi.

```

- 2) NMI interrupt initialization

```
NMI_Init(&stcNmiConfig).
```

- 3) Set the software interrupt initialization structure variable

```

/*****
/* SWI 31 configuration */
/*****
*****/
/* SWI Ch.31 */
stcSwiConfig.enSwiCh = SwiCh31.
/* Software interrupt */
stcSwiConfig.enSwiType = SwiInt.
/* Software interrupt callback function */
stcSwiConfig.pfnSwiCallback = SWI31_Callback.

```

- 4) Software interrupt initialization

```
SWI_Init(&stcSwiConfig).
```

- 5) Set external pin interrupt initialization structure variable

```

/***** */
/* External Int Ch.3 */
/***** */
stcExtiConfig.enExitCh = ExtiCh03.
/* Filter setting */
stcExtiConfig.enFilterEn = Enable;
stcExtiConfig.enFltClk = Pclk3Div8.
/* Both edge */
stcExtiConfig.enExtiLvl = ExIntBot hEdge.

```

6) Initialize external pin interrupt pins:

```

/* Set PD03 as External Int Ch.3 input */
MEM_ZERO_STRUCT(stcPortInit);
stcPortInit.enExInt = Enable;
PORT_Init(SW2_PORT, SW2_PIN, & stcPortInit).

```

7) Interrupted registration:

```

/* Select External Int Ch.3 */
stcIrqRegiConf.enIntSrc = INT_PORT_EIRQ3.
/* Register External Int to Vect.No.000 */
stcIrqRegiConf.enIRQn = Int000_IRQn.
/* Callback function */
stcIrqRegiConf.pfnCallback = ExtInt03_Callback.
/* Registration IRQ */
enIrqRegistration(&stcIrqRegiConf).

```

8) NVIC Configuration:

```

/* Clear pending */
NVIC_ClearPendingIRQ(s tclrqRegiConf.enIRQn).
/* Set priority */
NVIC_SetPriority(stcIrqRegiConf.e nIRQn, DDL_IRQ_PRIORITY_15).
/* Enable NVIC */
NVIC_EnableIRQ(stcIrqRegiConf.enIRQn).

```

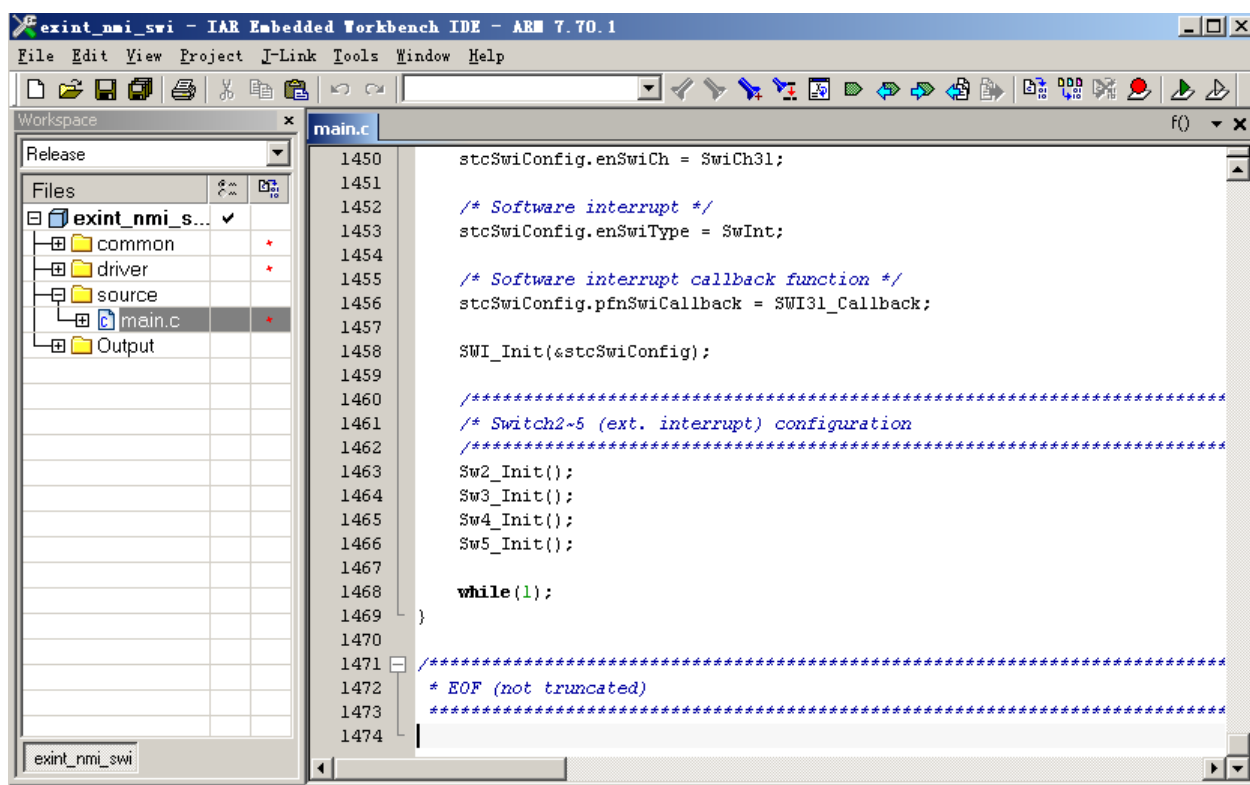
4.2 Code Run



Users can download the sample code (exint_nmi_swi) 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 the INTC module.

The following section describes how to run the INTC 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 exint_nmi_swi\ at

1) Open the exint_nmi_swi\ project and open the 'main.c' view as follows:



- 2) Click  to recompile the entire project.
- 3) Click  Download the code to the evaluation board and run it at full speed.
- 4) By pressing SW2~SW5 on the evaluation board and shorting the NMI jumper caps, the LEDs are observed to change to determine the execution of each interrupt service program.

5 Summary

The above section briefly introduces the INTC of HC32F460 series, explains the registers and part of the operation flow of INTC module, and demonstrates how to use INTC sample code, so that users can configure and use INTC module according to their needs in actual development.

6 Version Information & Contact

Date	Versions	Modify records
2019/3/15	Rev1.0	Initial Release



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: 39, Lane 572, Bibo Road, Zhangjiang Hi-

Tech Park, Shanghai, 201203, P.R. China

