EM9305 INTERRUPT SYSTEM

Product Family: **EM BLE System-On-Chip solution**

Part Number: EM9305
Keywords: Interrupt, priority

INTRODUCTION

EM9305 low level firmware system implements a hardware abstraction layer (HAL) providing individual drivers for each of the interrupts sources. The driver provides a callback mechanism where a designated or registered function is called whenever the interrupt is triggered. The designated function referred to as a callback function is registered with the driver as a function pointer. The callback function is usually called in the interrupt handler context between register context save and restore operation becoming transparent for the user.

PURPOSE

The purpose of current application note is to provide a more thorough overview of internal interrupt system integrated on EM9305. The interrupt management system inherits from ARC core architecture providing flexible configurations of interrupt system at hardware level and at software level.

SCOPE

This document is applicable to available EM9305 SDK versions such as EMBLUE. This document does not cover exceptions deserving a dedicated documentation.
EM9305 integrates an EM7D ARC core from Synopsys based on ARCV2_EM4_CORE4 as defined in sdk/common/em9305/includes/hw_versions.h.

HARDWARE ARCHITECTURE

The ARC core integrates a hardware configurable interrupt unit able to support a powerful interrupt system supporting nested interrupts with up to 16 priority levels from 0 to 15 and up to 240 interrupt sources (HW configuration). The scalable architecture is managing various sources of interrupt:

- ARC core interrupts (ARC Timer0/1, watchdog, DMA, …)
- Peripherals (GPIO, Universal Timer, SPI Master/Slave, I2C…)
- Software triggered Interrupts (SWI0, SWI1…)

INTERRUPT VECTORISATION TABLE

An interrupt vectorization table located in fetch able persistent memory defines for each interrupt entry index a 32-bit address pointing to associated interrupt handler. This table can updated, copied and relocated if needed using INT_VECTOR_BASE auxiliary register.

INTERRUPT UNIT SOFTWARE CONFIGURATION

A global interrupt Enable/Disable control is available to protect some critical area of code from being interrupted. In addition, each interrupt source can be individually configured in the interrupt unit seen as a register bank using IRQ_SELECT register to set IRQ_Level from 0 to 15, IRQ_Enable, IRQ_Trigger (edge or level). Each interrupt source can be individually sensed using IRQ_STATUS and IRQ_PENDING register. An interrupt threshold priority level is defined by application to enable interrupt greater or equal to specified threshold level.

INTERRUPT API

<irq.h> header file from SDK located in sdk/libs/em_hw_api/include provides access to interrupt controller functions. <irq.h> file includes <interrupt.h> header file integrating source level inline functions managing interrupts.

IRQ controller allows also to individually clear each IRQ source as well as to set each IRQ request to generate IRQ by SW. Some IRQ sources (after enabling and masking) are then ORed together to one final interrupt request going into IRQ manager (ARC part). IRQ manager is responsible of capturing interrupt requests from IRQ controller, setting priority for each IRQ source (after ORing).

Basic functions of drivers are:

- Set priority level of each IRQ (IRQ_SetPriority)
- Get priority level of each IRQ (IRQ_GetPriority)
- Enable/disable interrupt globally (IRQ_EnableInterrupts)
- Enable/disable each IRQ individually (IRQ_Enable)
- Mask/unmask each IRQ individually (IRQ_Mask)
- Read IRQ status (IRQ_GetStatus)
- Get current interrupt index (IRQ_GetInterruptCause)
- Clear/set interrupt request (IRQ_Clear)
- Trig SW interrupt (IRQ_Trigger)

EM9305 DEFAULT CONFIGURATION

Default configuration coming from ROM firmware (version1.0)
Priority level = 0 (Highest) to 9 (Lowest)
Default IRQ Threshold = 10

Irq #	Source	Priority
Irq0-15	ARC core Allocated Exceptions (reset, illegal instructions,…)	Highest
Irq16-20	ARC core interrupts predefined: T0, T1, WD, DMA done, DMA error	5
Irq21-30	PROTOCOL TIMER	1
Irq31-35	SLEEP TIMER	2
Irq36-37	RADIO TX, RX	0
Irq38-39	SPIS TX, RX	4
Irq40-41	UART TX, RX	4
Irq42	GPIO	6
Irq43-44	UT2, UT3	7
Irq45	SPIM	7
Irq46	I2C	7
Irq47	RC CALIB	3
Irq48	ADC	6
Irq49	PML_SVLD	5
Irq50	PML_CLOCK	3
Irq51	NVM	7
Irq52	QDEC	7
Irq53	USB	7
Irq54	CRYPTO UNIT	6
Irq55	I2S	6
Irq56-65	SWI0-9	8

CONCLUSIONSc

The level of information is provided for information mainly to have better understanding about EM9305 platform, it is not recommended that application is modifying default settings in particular priority set to take in account platform real time requirements. Please be aware that NVM embedded (EM core firmware) is susceptible to change default interrupt system configuration however exposed mechanism remain valid.
Most of the driver provide a call back function allowing application to insert its dedicated processing in the interrupt handler. This approach should provide a level of flexibility discouraging application to modify low level IRQ service routines.
In case you identified specific needs to modify interrupt system please contact EM to get recommendations.

More information can be find in DesignWare® ARC v2 ISA Programmer's Reference Manual for ARC EM Processors from Synopsys.