Porting Kinetis Interrupt from CodeWarrior to KDS

By Jennie Zhang

Recently I received two cases from Kinetis users. They both meet problem when porting Kinetis Interrupt project from CodeWarrior 10.x to KDS. There is a document KDS_Porting_Guide.pdf under KDS install folder which covers this topic. However I know many customers still have difficulty even they follow the document steps. This scenario is normal; because users' projects are various, we can't expect one porting document solve all kinds of problem of individuals. For me, honestly, I seldom use this porting guide to port my project. I prefer porting my project all by hand. Thus I can know my project changes from up to bottom. Thus even if I meet problem later, I can still position it easily and quickly.

For the reason of time, in this article, I will focus on how to port Kinetis Interrupt from CodeWarrior to KDS by hand.

Some NVIC register definition name and file structure are different in CodeWarrior and KDS. One big difference is that KDS uses core file from ARM limited directly. I summarized the difference as below table. It's good to know the basic difference before we start the porting.

		CodeWarrior	KDS
NVIC Register Definition :	Where	MCU header file. Eg: MK60N512VMD100.h	ARM Core file. Eg: core_cm4, core_cm0plus.h
	Written by	NXP(former Freescale)	ARM Limited
	Sample of usage	NVICISERx = 0x01	NVIC->ISER[x] = 0x01 Packaged in NVIC_EnableIRQ(n)
Vector Table Definition:	Where	Kinetis_sysinit.c	Startup_MK60D10.S
	Language Using	C	Assembly
	Interrupt handler name	Defined by user	Defined by KDS. Eg, PORTA_IRQHandler Allow user modify.

1. Review basic Kinetis NVIC knowledge.

General steps for enabling an interrupt on NVIC:

1) Enable the peripheral to be used
2) Set the proper bit on the NVICSERx to enable the interrupt on the NVIC
3) Clear any pending interrupt by writing to the NVICCPRx to avoid any spurious interrupt
4) Configure the interrupt priority by writing to the NVICIPn
5) Write the ISR
6) Enable global interrupts

Here, there are two indexes "x" "n" suffixed. Remember what are they? If not, see below example.

This Fig1 is extracted from Interrupt Vector Assignment Table list from K60 user manual.

Port A pin detect interrupt is highlighted. This interrupt IRQ number is 87. Thus we set n = 87. NVIC non-IPR register number is 2, we set x = 2

Here is a formula for the relation of x and n: x = n/32. Also take previous example: 87/32=2, BINGO!

Address	Vector	IRQ[1]	NVIC non-IPR register number [2]	NVIC IPR register number [3]	Source module	Source description
0x0000_0198	102	86	2	21	—	—
0x0000_019C	103	87	2	21	Port control module	Pin detect (Port A)
0x0000_01A0	104	88	2	22	Port control module	Pin detect (Port B)
0x0000_01A4	105	89	2	22	Port control module	Pin detect (Port C)
0x0000_01A8	106	90	2	22	Port control module	Pin detect (Port D)

Fig. 1 Interrupt Vector Assignment

2. interrupt initialization: CodeWarrior vs. KDS

CodeWarrior		KDS	
NVICISERx	Interrupt Set Enable Register	NVIC_EnableIRQ(n)	Enable External Interrupt : Enable a device-specific interrupt in the NVIC interrupt controller
NVICICERx	Interrupt Clear Enable Register	NVIC_DisableIRQ(n)	Disable External Interrupt : Disables a device-specific interrupt in the NVIC interrupt controller
NVICICPRx	Interrupt Clear Pending Register	NVIC_ClearPendingIRQ(n)	Clear Pending Interrupt : Clears the pending bit of an external interrupt
NVICISPRx	Interrupt Set Pending Register	NVIC_GetPendingIRQ(n)	Get Pending Interrupt : Read the pending register in the NVIC and returns the pending bit for the specified interrupt
		NVIC_SetPendingIRQ(n)	Set Pending Interrupt : Sets the pending bit of an external interrupt
NVICIABRx	Interrupt Active bit Register	NVIC_GetActive(n)	Get Active Interrupt : Get Active Interrupt: reads the active register in NVIC and returns the active bit.
NVICIPn	Interrupt Priority Register	NVIC_SetPriority(n,priority)	Set Interrupt Priority: Sets the priority of an interrupt.

► Example: Set up the PORTA interrupt:

1) Locate the interrupt vector that you want on the Interrupt Vector Assignment Table list from the Kinetis device used. See Fig.1. Port A pin detect interrupt is highlighted.

   From the Fig1, we can know: n = 87. x = 2. So as example NVICISERx is NVICISER2 in this case.

2) Enable PORTA interrupt:
   87%32 = 23
   - CodeWarrior: NVICISER2 |=(1<<23);
   - KDS: NVIC_EnableIRQ(87);

3) Clear any pending interrupts :
   - CodeWarrior: NVICICPR2|=(1<<23);
   - KDS: NVIC_ClearPendingIRQ(87);
4) Set the interrupt priority. Just the 4 most significant bits are used.
   - CodeWarrior: NVICIP87 = 0x80;
   - KDS: NVIC_SetPriority(87, 8);

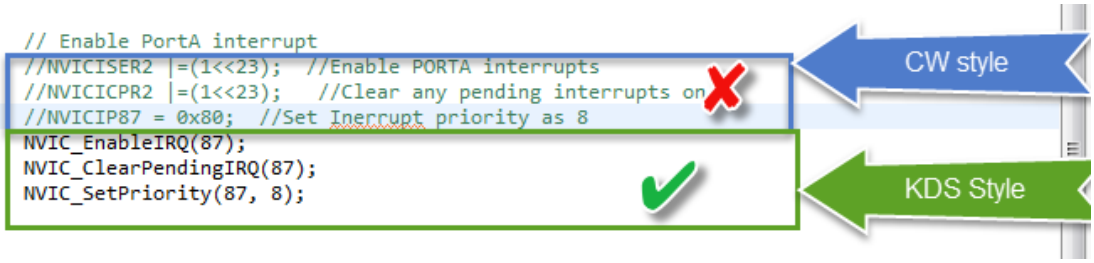
3. Porting Kinetis interrupt project from CodeWarrior to KDS

Enclosed CodeWarrior demo code test_interrupt_cw.zip based on board TWR-K60N512.

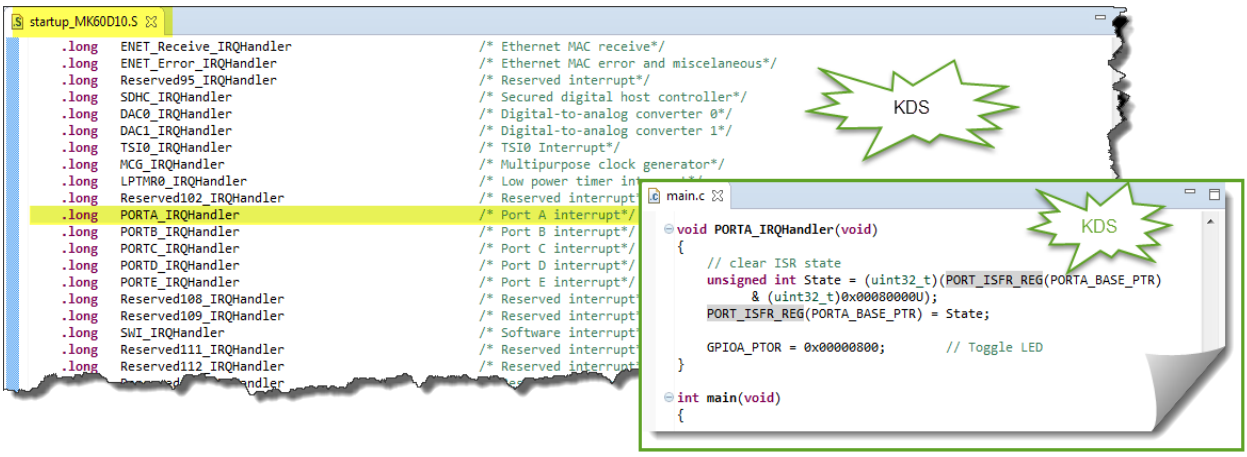
Function: press SW1, it triggers PORTA interrupt then toggles led E1.

Porting steps:

1) Under KDS, create a new project named ***test_interrupt_kds,*** no processor expert, no sdk supported.
2) Copy main(),ConfigureClocks(),GpioInitK60(),OnPortEvent() from CodeWarrior to KDS project.
3) Under KDS, Change NVIC initialization code from CodeWarrior style to KDS style.



4) Under KDS, Change interrupt handler function name from OnPortEvent to PORTA_IRQHandler() which is KDS style handler name.



Then build the KDS project and download program. We will see there is no error and porting is done successfully.