Ctimer Trigger ADC

ADC trigger includes software trigger and hardware trigger. Software trigger just set the START bit to 1 in the SEQA_CTRL(or SEQB_CTRL) register, that will launch one pass through this conversion sequence. Hardware triggers are either a signal from an external pin or an internal signal.

There are many hardware trigger sources, for example LPC804, there are 8 inputs triggers sources.

Table 271 **ADC hardware trigger inputs**

Input #	Source	Description
0	-	No hardware trigger.
1	PININT0_IRQ	GPIO_INT interrupt 0.
2	PININT1_IRQ	GPIO_INT interrupt 1.
5	T0_MAT3	CTIMER match 3.
6	CMP0_OUT_ADC	Analog comparator output.
7	GPIO_INT_BMAT	GPIO_INT bmatch.
8	ARM_TXEV	Arm core TXEV event.

This article mainly introduces how to config CTIMER match 3 trigger ADC in LPC804, includes how to config related registers, and the code under SDK. Other LPC serials, also can refer to this DOC.

1.How To Configure ADC Part.

2.How to Configure CTIMER Part

3.Project Basic Information

4.Reference

1. How To Configure ADC Part.

1.1. Register:

Through “Table 271. ADC hardware trigger inputs” , we know when set Input# to 5 can select CTIMER match3 as ADC trigger source. so set the **TRIGGER** bits (15:12)of SEQA_CRTL to 5 to config CTIMER match 3 as hardware trigger source (If sue A/D conversion sequence B, need config SEQB_CRTL).

Table 276. A/D Conversion Sequence A Control Register (SEQA_CTRL, address 0x4001 C008) bit description

Bit	Symbol	Value	Description	Reset value
11:0	CHANNELS		Selects which one or more of the twelve channels will be sampled and converted when this sequence is launched. A 1 in any bit of this field will cause the corresponding channel to be included in the conversion sequence, where bit 0 corresponds to channel 0, bit 1 to channel 1 and so forth. When this conversion sequence is triggered, either by a hardware trigger or via software command, A/D conversions will be performed on each enabled channel, in sequence, beginning with the lowest-ordered channel. **Remark:** This field can ONLY be changed while the SEQA_ENA bit (bit 31) is LOW. It is allowed to change this field and set bit 31 in the same write.	0x00
14:12 ✗ 15:12 ✓	TRIGGER		Selects which of the available hardware trigger sources will cause this conversion sequence to be initiated. Program the trigger input number in this field. **Remark:** In order to avoid generating a spurious trigger, it is recommended writing to this field only when the SEQA_ENA bit (bit 31) is low. It is safe to change this field and set bit 31 in the same write.	0x0
17:16	-		Reserved.	-

Please pay attention that, the TRIGGER should be 15:12 of SEQA_CTRL, total 4 bits, it is wrong in User Manual (UM11065 Rev. 1.3).

1.2. SDK Code:

In function `static void ADC_Configuration(void);`

```
  adcConvSeqConfigStruct.triggerMask = 5U;
// CTIMER_MAT3 trigger ADC
  adcConvSeqConfigStruct.triggerPolarity = kADC_TriggerPolarityPositiveEdge;
//positive edge launches the conversion
```

This function includes the ADC Configuration and work mode.

2. How to Configure Ctimer Part.

2.1. Register:

In lpc804, there are four Matches, External Match0, External Match1, External Match2 and External Match3,only External Match3 can trigger ADC. The Match register(MR) values are continuously compared to the Timer Counter(TC) value. When the two values are equal, actions can be triggered automatically. Even though User Manal haven't statement how Match3 trigger ADC, we can config External Match 3 output refer to below rules to trigger ADC:

1) If A/D Conversion Sequence A Control Register SEQA_CTRL->TRIGPOL set to 1, a positive edge launches the conversion sequence on the selected trigger input, config EM3 to0x2 can trigger ADC.
2) If A/D Conversion Sequence A Control Register SEQA_CTRL->TRIGPOL set to 0, a negative edge launches the conversion sequence on the selected trigger input, config EM3 to 0x1 can trigger ADC.

From "UM->Table233 Timer external match registers (EMR, offset 0x03C) bit description" , we need config EMC3 to control External Match 3 output.

Table 233. Timer external match registers (EMR, offset 0x03C) bit description

Bit	Symbol	Value	Description	Reset value
0	EM0	-	External Match 0. This bit reflects the state of output MAT0, whether or not this output is connected to a pin. When a match occurs between the TC and MR0, this bit can either toggle, go LOW, go HIGH, or do nothing, as selected by EMR[5:4]. This bit is driven to the MAT pins if the match function is selected via SWM.	0
1	EM1	-	External Match 1. This bit reflects the state of output MAT1, whether or not this output is connected to a pin. When a match occurs between the TC and MR1, this bit can either toggle, go LOW, go HIGH, or do nothing, as selected by EMR[7:6]. This bit is driven to the MAT pins if the match function is selected via SWM.	0
2	EM2	-	External Match 2. This bit reflects the state of output MAT2, whether or not this output is connected to a pin. When a match occurs between the TC and MR2, this bit can either toggle, go LOW, go HIGH, or do nothing, as selected by EMR[9:8]. This bit is driven to the MAT pins if the match function is selected via SWM.	0
3	EM3	-	External Match 3. This bit reflects the state of output MAT3, whether or not this output is connected to a pin. When a match occurs between the TC and MR3, this bit can either toggle, go LOW, go HIGH, or do nothing, as selected by MR[11:10]. This bit is driven to the MAT pins if the match function is selected via SWM.	0

11:10	EMC3		External Match Control 3. Determines the functionality of External Match 3.	00
		0x0	Do Nothing.	
		0x1	Clear. Clear the corresponding External Match bit/output to 0 (MAT3 pin is LOW if pinned out).	
		0x2	Set. Set the corresponding External Match bit/output to 1 (MAT3 pin is HIGH if pinned out).	
		0x3	Toggle. Toggle the corresponding External Match bit/output.	

So the more detailed description of EMC3 should be:

11:10	EMC3		External Match Control 3. Determines the functionality of External Match 3.	00
		0x0	Do Nothing.	
		0x1	Clear. Clear the corresponding External Match bit/output to 0 (MAT3 pin is LOW if pinned out). If TRIGGER is 5, TRIGPOL bit is 0, MATCH3 can trigger ADC.	
		0x2	Set. Set the corresponding External Match bit/output to 1 (MAT3 pin is HIGH if pinned out). If TRIGGER is 5, TRIGPOL bit is 1, MATCH3 can trigger ADC.	
		0x3	Toggle. Toggle the corresponding External Match bit/output. If TRIGGER IS 5, half of MATCH3 can trigger ADC	

Pay attention:

- If in ADC_Configuration() function, config a positive edge launches the conversion sequence on the trigger, as below code:

```
adcConvSeqConfigStruct.triggerPolarity = kADC_TriggerPolarityPositiveEdge;
```

Set EMC3 to 0x2: Set the corresponding External Match bit to 1. In Match3 interrupt callback function, clear EM3 to 0.

- If in ADC_Configuration() function, config a negative edge launches the conversion sequence on the trigger, as below code:

```
adcConvSeqConfigStruct.triggerPolarity = kADC_TriggerPolarityNegativeEdge;
```

Set EMC3 to 0x1. Clear the corresponding External Match bit to 0. In Match3 interrupt callback function, set EM3 to 1.

- If config EMC3 to 0x3, Toggle the corresponding External Match bit, also can work no matter set positive or negative edge to launch ADC conversion. While the ADC trigger frequency is half of the Match3, so if use this mode, need pay attention to MATCH register, make the frequency of Match3 is double of trigger

ADC. Neither need set nor clear EM3 on interrupt callback function.

- If config EMC3 to 0x0, won't trigger ADC.

In my project, I have config a positive edge launches the conversion sequence on the trigger, so need set EMC3 to 0x2.

2.2 SDK Code:

```
matchConfig1.outControl =  kCTIMER_Output_Set;
//Config EMC3 = 0x2, Set the External Match bit to 1.
```

Ctimer code:

```
  /*Initialize Ctimer */
CTIMER_GetDefaultConfig(&config);
CTIMER_Init(CTIMER, &config);

/* Configuration Ctimer MATCH3 */
 matchConfig1.enableCounterReset = true;
 matchConfig1.enableCounterStop = false;
 matchConfig1.matchValue = CTIMER_CLK_FREQ ;
 matchConfig1.outControl =  kCTIMER_Output_Set;
 matchConfig1.outPinInitState = true;
 matchConfig1.enableInterrupt = true;
 CTIMER_SetupMatch(CTIMER, CTIMER_MAT3_OUT, &matchConfig1);
 CTIMER_RegisterCallBack(CTIMER, &ctimer_callback_table[0],
kCTIMER_MultipleCallback);


/*Start Ctimer*/
CTIMER_StartTimer(CTIMER);
```

3. Project Basic information

Project in attachment is developed on MCUXpresso IDEv11.1.1,

LPCXpresso804 board.

4. Reference

< LPC804 User Manual-UM11065>