灵活多通道dma扫描

使用lptmr触发adc采集，adc采集完成触发dma通道1开始传输到指定缓存，dma通道1传输完成触发链接，链接dma通道0，dma通道0将adc配置传给adc配置寄存器。

这样可以灵活采集各种通道，并且对资源占用较小。

板子:frdm-k64

原理如下

Adc配置

使能adc中断，配置adc通道

EnableIRQ(*ADC0\_IRQn*);

ADC16\_GetDefaultConfig(&adc16ConfigStruct);

ADC16\_Init( ADC0, &adc16ConfigStruct);

ADC16\_DoAutoCalibration( ADC0 );

ADC16\_EnableHardwareTrigger( ADC0, true);

ADC16\_EnableDMA( ADC0, true);

adc16ChannelConfigStruct.channelNumber = g\_ADC\_mux[2];

adc16ChannelConfigStruct.enableInterruptOnConversionCompleted = true;

adc16ChannelConfigStruct.enableDifferentialConversion = false;

ADC16\_SetChannelConfig( ADC0, 0, &adc16ChannelConfigStruct);

Lptmr配置

lptmr\_config\_t lptmrConfig;

LPTMR\_GetDefaultConfig(&lptmrConfig);

lptmrConfig.bypassPrescaler = false;

lptmrConfig.prescalerClockSource = *kLPTMR\_PrescalerClock\_1*;

/\* Initialize the LPTMR \*/

LPTMR\_Init(LPTMR0, &lptmrConfig);

/\* Set the LPTimer period \*/

LPTMR\_SetTimerPeriod( LPTMR0,USEC\_TO\_COUNT(200000, CLOCK\_GetFreq(*kCLOCK\_LpoClk*)));

SIM->SOPT7 |= 0x8EU; //配置adc硬件触发源

Dma配置

edma\_transfer\_config\_t transferConfig\_ch0;

edma\_transfer\_config\_t transferConfig\_ch1;

edma\_config\_t userConfig;

/\* Configure DMAMUX \*/

DMAMUX\_Init(DMAMUX0);

DMAMUX\_SetSource(DMAMUX0, DMAChannel\_0, 60); /\* Channel 0 Source 60: DMA always enabled \*/

DMAMUX\_EnableChannel(DMAMUX0, DMAChannel\_0);

DMAMUX\_SetSource(DMAMUX0, DMAChannel\_1, 40); /\* Channel 1 Source 40: ADC COCO trigger \*/

DMAMUX\_EnableChannel(DMAMUX0, DMAChannel\_1);

EDMA\_GetDefaultConfig(&userConfig);

userConfig.enableHaltOnError = false;

EDMA\_Init(DMA0, &userConfig);

EDMA\_CreateHandle(&g\_EDMA\_Handle\_1, DMA0, DMAChannel\_1);

EDMA\_SetCallback(&g\_EDMA\_Handle\_1, EDMA\_Callback\_1, NULL);

//设置dma通道1的adc值传到g\_ADC0\_resultBuffer

EDMA\_PrepareTransfer(&transferConfig\_ch1, /\* Prepare TCD for CH1 \*/

(uint32\_t\*) (ADC0->R), /\* Source Address (ADC0\_RA) \*/

**sizeof**(uint16\_t), /\* Source width (2 bytes) \*/

g\_ADC0\_resultBuffer, /\* Destination Address (Internal buffer)\*/

**sizeof**(g\_ADC0\_resultBuffer[0]), /\* Destination width (2 bytes) \*/

**sizeof**(uint16\_t), /\* Bytes to transfer each minor loop (2 bytes) \*/

B\_SIZE \* 2, /\* Total of bytes to transfer (12\*2 bytes) \*/

*kEDMA\_PeripheralToMemory*); /\* From ADC to Memory \*/

/\* Push TCD for CH1 into hardware TCD Register \*/

EDMA\_SubmitTransfer(&g\_EDMA\_Handle\_1, &transferConfig\_ch1);

//将dma通道1链接到通道0

EDMA\_SetChannelLink(DMA0, DMAChannel\_1, *kEDMA\_MinorLink*, DMAChannel\_0);

EDMA\_SetChannelLink(DMA0, DMAChannel\_1, kEDMA\_MajorLink, DMAChannel\_0);

EDMA\_CreateHandle(&g\_EDMA\_Handle\_0, DMA0, DMAChannel\_0);

EDMA\_SetCallback(&g\_EDMA\_Handle\_0, EDMA\_Callback\_0, NULL);

//设置dma通道0，将adc的配置传给寄存器adc\_sc1

EDMA\_PrepareTransfer(&transferConfig\_ch0, /\* Prepare TCD for CH0 \*/

&g\_ADC\_mux[0], /\* Source Address (ADC channels array) \*/

**sizeof**(g\_ADC\_mux[0]), /\* Source width (1 bytes) \*/

(uint32\_t\*)(ADC0->SC1),/\* Destination Address (ADC\_SC1A\_ADCH)\*/

**sizeof**(uint8\_t), /\* Destination width (1 bytes) \*/

**sizeof**(uint8\_t), /\* Bytes to transfer each minor loop (1 bytes) \*/

CHANNELS, /\* Total of bytes to transfer (3\*1 bytes) \*/

*kEDMA\_MemoryToPeripheral*);/\* From ADC channels array to ADCH register \*/

/\* Push TCD for CH0 into hardware TCD Register \*/

EDMA\_SubmitTransfer(&g\_EDMA\_Handle\_0, &transferConfig\_ch0);

//传输完后修正，通道0倒退3，通道1倒退24

DMA0->TCD[0].SLAST = -1 \* CHANNELS;

DMA0->TCD[1].DLAST\_SGA = -2 \* B\_SIZE;

中断函数，以及edma的回调函数，这个程序执行一遍就结束了，如果想一直执行下去，将**EDMA\_Callback\_1里的**EDMA\_StartTransfer(&g\_EDMA\_Handle\_1);注释去掉，就可以一直运行

**void** **ADC0\_IRQHandler**(**void**)

{

g\_Adc16ConversionDoneFlag = true;

}

**void** **EDMA\_Callback\_0**(edma\_handle\_t \*handle, **void** \*param, bool transferDone, uint32\_t tcds)

{

**if** (transferDone)

{

g\_Transfer\_Done\_ch0 = true;

}

}

**void** **EDMA\_Callback\_1**(edma\_handle\_t \*handle, **void** \*param, bool transferDone, uint32\_t tcds)

{

**if** (transferDone)

{

// EDMA\_StartTransfer(&g\_EDMA\_Handle\_1);

g\_Transfer\_Done\_ch1 = true;

}

}

实验结果

# 参考文章：[Using DMA to Emulate ADC Flexible Scan Mode with SDK 2.x](https://community.nxp.com/docs/DOC-335320)