Freescale Semiconductor
Application Note

Document Number:AN4639
Rev. 0, 12/2012

How to Configure DMA Periodic Trigger Function on Kinetis Devices

by: Jimmy Cheng

Contents

1 Introduction

The enhanced direct memory access (eDMA) controller is integrated in most of Kinetis families and it performs data movement operations with high efficiency. Most of the time a user wants to implement data movement operations periodically. This peripheral request may request only a DMA transfer when the PIT trigger is produced, because the PIT trigger works by gating the request form the peripheral to DMA channel until a trigger event has been seen.

This application note is aimed at configuring DMA Periodic Trigger mode function. Example code is built on KE15 device.

DMA MUX routes 53 peripheral DMA sources/slots and 10 always-on slots to any of the 16 channels. DMA MUX has three operating modes:

- Disabled mode: In this mode, the DMA channel is disabled. Because disabling and enabling of DMA channels is done primarily via the DMA configuration registers, this mode is used mainly as the reset state for a DMA channel in the DMA channel MUX. It may also be used to temporarily suspend a DMA channel while reconfiguration of the system takes place, for example, changing the period of a DMA trigger.

- Normal mode: In this mode, a DMA source is routed directly to the specified DMA channel. The operation of the DMA MUX in this mode is completely transparent to the system.
- Periodic Trigger mode: In this mode, a DMA source may request only a DMA transfer periodically by PIT trigger. This mode is available only for DMA for the first four channels from channel 0 to channel 3. The PIT generates periodic trigger events to the number of DMA MUX as shown in Table 1.

In Periodic Trigger mode, the number of PIT channel is corresponding to the number of DMA and DMA MUX channel. For example , if the customer wants to use PIT2 as trigger, the corresponding DMA2 register and DMA MUX channel 2 register should be configured.

Table 1. PIT channel assignments for periodic DMA trigger

DMA channel number	DMA MUX number	PIT channel
DMA Channel 0	DMA MUX channel 0	PIT Channel 0
DMA Channel 1	DMA MUX channel 1	PIT Channel 1
DMA Channel 2	DMA MUX channel 2	PIT Channel 2
DMA Channel 3	DMA MUX channel 3	PIT Channel 3

Before configuring the DMA MUX channel registers, the user has to know the peripheral source number. For instance, if the RDRF flag of UART0 module is peripheral request, the source number should be 2. See Table 2 for details.

DMA request is produced when the peripheral request and trigger are effective simultaneously. If the DMA request has been serviced, and next peripheral request does not appear, even though new PIT trigger arrives, this trigger will be ignored. See Figure 1 and Figure 2.

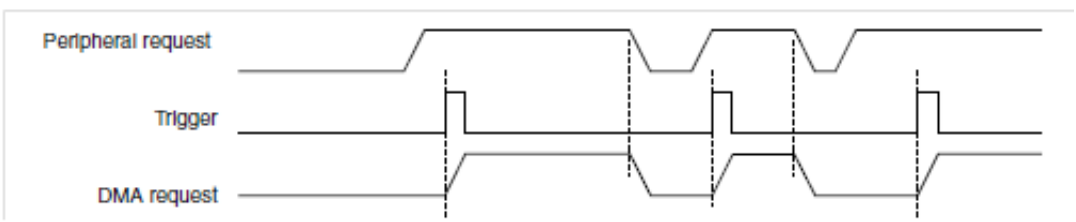

Figure 1. DMA MUX channel triggering: normal operation

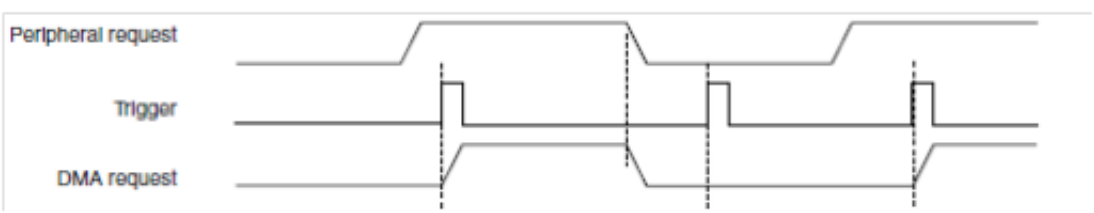

Figure 2. DMA MUX channel triggering: ignored trigger

Table 2. DMA request sources—MUX

Source number	Source module	Source description
0	—	Channel disable[1]

Table continues on the next page...

Table 2. DMA request sources—MUX (continued)

Source number	Source module	Source description
1	Reserved	Not used
2	UART0	Receive
3	UART0	Transmit
4	UART1	Receive
5	UART1	Transmit
6	UART2	Receive
7	UART2	Transmit
8	UART3	Receive
9	UART3	Transmit
10	SPI0	Receive
11	SPI0	Transmit
12	SPI1	Receive
13	SPI1	Transmit
14	I_2C0	—
15	I_2C1	—
16	Reserved	—
17	Reserved	—
18	FTM0	Channel 0
19	FTM0	Channel 1
20	FTM0	Channel 2
21	FTM0	Channel 3
22	FTM0	Channel 4
23	FTM0	Channel 5
24	FTM3	Channel 0
25	FTM3	Channel 1
26	FTM3	Channel 2
27	FTM3	Channel 3
28	FTM3	Channel 4
29	FTM3	Channel 5
30	FTM1	Channel 0
31	FTM1	Channel 1
32	FTM2	Channel 0
33	FTM2	Channel 1
34	PDB0	—
35	PDB1	—
36	PDB2	—
37	PDB3	—
38	Reserved	—
39	ADC0	—

Table continues on the next page...

Table 2. DMA request sources—MUX (continued)

Source number	Source module	Source description
40	ADC1	—
41	ADC2	—
42	ADC3	—
43	CMP0	—
44	CMP1	—
45	CMP2	—
46	CMP3	—
47	RTC	—
48	CMT	—
49	Port control module	Port A
50	Port control module	Port B
51	Port control module	Port C
52	Port control module	Port D
53	Port control module	Port E
54	DMA MUX	Always enabled
55	DMA MUX	Always enabled
56	DMA MUX	Always enabled
57	DMA MUX	Always enabled
58	DMA MUX	Always enabled
59	DMA MUX	Always enabled
60	DMA MUX	Always enabled
61	DMA MUX	Always enabled
62	DMA MUX	Always enabled
63	DMA MUX	Always enabled

1. Configuring a DMA channel to select source 0 or any of the reserved sources disables that DMA channel.

2 DMA MUX request sources

KE15 includes a DMA request mux that allows up to 64 DMA request signals to be mapped to any of the first eight DMA channels plus eight source inputs of iEVENT module.

Four iEVENT outputs (ch0 and ch1 to NVIC, ch2 and ch3 to DMAC) will be created by combinational boolean logic on each four EVENT inputs (A, B,C, D). iEVENT module has eight source inputs available, which are driven by 8 outputs of DMA request mux (number 15 to 8). These 8 source inputs are specifically routed to the four event inputs (A,B,C,D) of each channel. Source inputs 0-3 are routed to the inputs (A,B,C,D) of Ch0 and Ch2 respectively, and source inputs 4–7 to the inputs (A,B,C,D) of Ch1 and Ch3 respectively.

The DMA request signals from ADC0–3 are directly routed to DMAC channel 8–11. The iEvent out signals (channel 2 and 3) from iEVENT are directly routed to DMAC channel 12–13. See Figure 3.

The channle 0–3 of DMA requested MUX provides periodic triggering capability. The trigger is generated by Periodic Interrupt Timer (PIT). See Table 1.

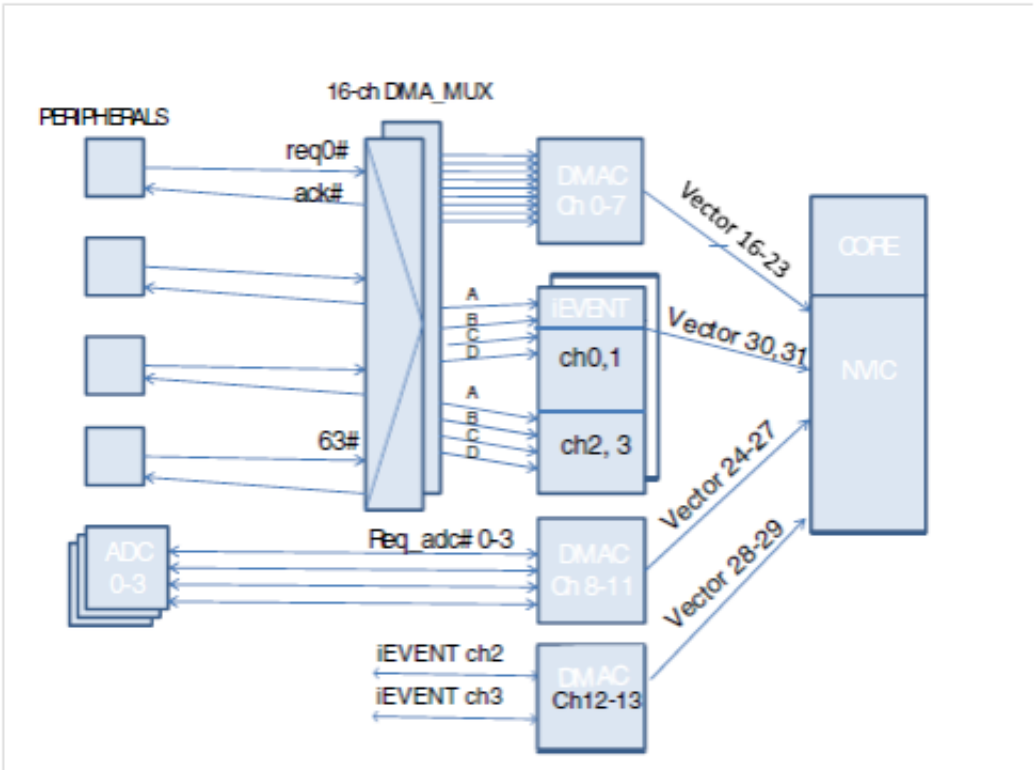

Figure 3. DMA request sources routing to DMA MUX

3 Application

The Periodic Trigger mode example in this application note is DMA tranferred fixed data from source address to destination address in RAM, and DMA MUX is configured to Periodic Trigger mode. The peripheral slot is 18, it is FTM0 Channel 0 flag. FTM0 works in Output compare-toggle output on match mode, when channel value register (FTM0_C0V) matches counter register (FTM0_CNT) value. The channel flag will be set, which is the peripheral request signal for DMA MUX. Periodic trigger signal is produced by PIT0, and data will be moved when DMA request appears. If one DMA transfer is completed, DMA ISR will disable DMA MUX ENBL bit and terminate DMA MUX Trigger mode.

Furthermore, Periodic Trigger mode application can be used and DMA MUX always enables slot number from 54 to 63, that is, total 10 sources. Once the source number to these sources is set, the peripheral request is not necessary, the trigger mode is concluded by PIT trigger period. See Table 2. The initialization of DMA MUX channel register is shown below:

```
DMA_MUX0_CHCFG2 = DMA_MUX_CHCFG_ENBL_MASK //enables DMAMUX channel
                 |DMA_MUX_CHCFG_TRIG_MASK //enables Periodic Trigger mode
                 |DMA_MUX_CHCFG_SOURCE(63);//63 is always enable
```

4 Initialization

The initialization includes the register configuration for FTM, PIT, DMA MUX, and DMA. It also enables the FTM channel interrupt and PIT interrupt, and the user has to define the source data and the destination address for data movement. Before configuring the register of these modules, make sure the relevant clock gate bit is enabled in System Clock Gating Control Register (SIM_SCGC5~ SIM_SCGC7).

Below is an introduction to initialization of these modules:

Initialization

Before configuring DMA MUX trigger and source number, the ENBL bit should be disabled. Select DMA MUX channel 0 for PIT Trigger mode.

```
/*DMA MUX initialization*/
SIM_SCGC6 |= SIM_SCGC6_DMAMUX_MASK; //enables DMA MUX clock gate
DMA_MUX0_CHCFG0 = 0; //clears register for changing source and trigger
DMA_MUX0_CHCFG0 = DMA_MUX_CHCFG_ENBL_MASK //enables DMA MUX channel
                  |DMA_MUX_CHCFG_TRIG_MASK //enables Periodic Trigger mode
                  |DMA_MUX_CHCFG_SOURCE(18);//source number is 18, FTM0 channel 0
```

Before initializing DMA register, define the source data and the address of source and destination.

```
uint8_t Data_source[8]={0x01,0x23,0x45,0x67,0x89,0xAB,0xCD,0xEF}; // source data space
uint8_t Data_Desti[8];//destination data space
```

So, the source address is &Data_source[0]), and the destination address is &Data_Desti[0]).

```
/*eDMA module initialization*/
//SIM_SCGC7 |= SIM_SCGC7_DMA_MASK;//enables DMA clock gate, default value is enable
DMA_ERQ = 0x01;//enables DMA0 request
DMA_TCD0_SADDR = (uint32_t)(&Data_source[0]);//defines source data address
DMA_TCD0_SOFF = 1;//Source address signed offset
DMA_TCD0_DADDR = (uint32_t)(&Data_Desti[0]);//defines destination data address
DMA_TCD0_CITER_ELINKNO = 0x01;//CITER=1
DMA_TCD0_BITER_ELINKNO = 0x01;//BITER=1
DMA_TCD0_NBYTES_MLNO = 8;//byte number
DMA_TCD0_DOFF = 1;//destination address signed offset
DMA_TCD0_ATTR = 0;//8 bit transfer size, register default value is undefined
DMA_TCD0_SLAST = -8;//restores the source address to the initial value
DMA_TCD0_DLASTSGA = -8;//restores the destination address to the initial value
DMA_TCD0_CSR = DMA_CSR_INTMAJOR_MASK;//The end-of-major loop interrupt is enabled
```

Configure FTM0 that works under Output Compare Toggle Out mode. Set up the appropriate channel value after calculating period of the FTM channel flag. Usually, FTM0_SC register is initialized at the end.. FTM0 channel flag CH0F bit is cleared when the channel DMA transfer is done.

```
/*FTM0 initialization*/
/*FTM0 initialization*/
SIM_SCGC6 |= SIM_SCGC6_FTM0_MASK;//enables FTM0 clock gate
PORTE_PCR3 = PORT_PCR_MUX(4);//sets PTE4 as FTM0_CH0 output pin
FTM0_C0SC = 0x55;//output compare toggle output on match, enables DMA transfer for the
channel
FTM0_C0V = 35;// channel value, channel flag is set when FTM0_CNT value is 35
FTM0_MOD = 72;// modulo value, period is 1us
FTM0_SC = FTM_SC_CLKS(1);//selects system clock as FTM0 clock source, starts FTM0 counter
PIT module is the periodic trigger.
/*PIT initialization*/
SIM_SCGC6 |= SIM_SCGC6_PIT_MASK; //enables PIT clock gate
PIT_MCR &= ~PIT_MCR_MDIS_MASK;//turns on PIT
PIT_LDVAL0 = 71; //PIT trigger is 1 µs
PIT_TCTRL0 |= PIT_TCTRL_TIE_MASK;//enables PIT timer interrupt
PIT_TCTRL0 |= PIT_TCTRL_TEN_MASK;//enables timer0
```

See the following flow chart for application code:

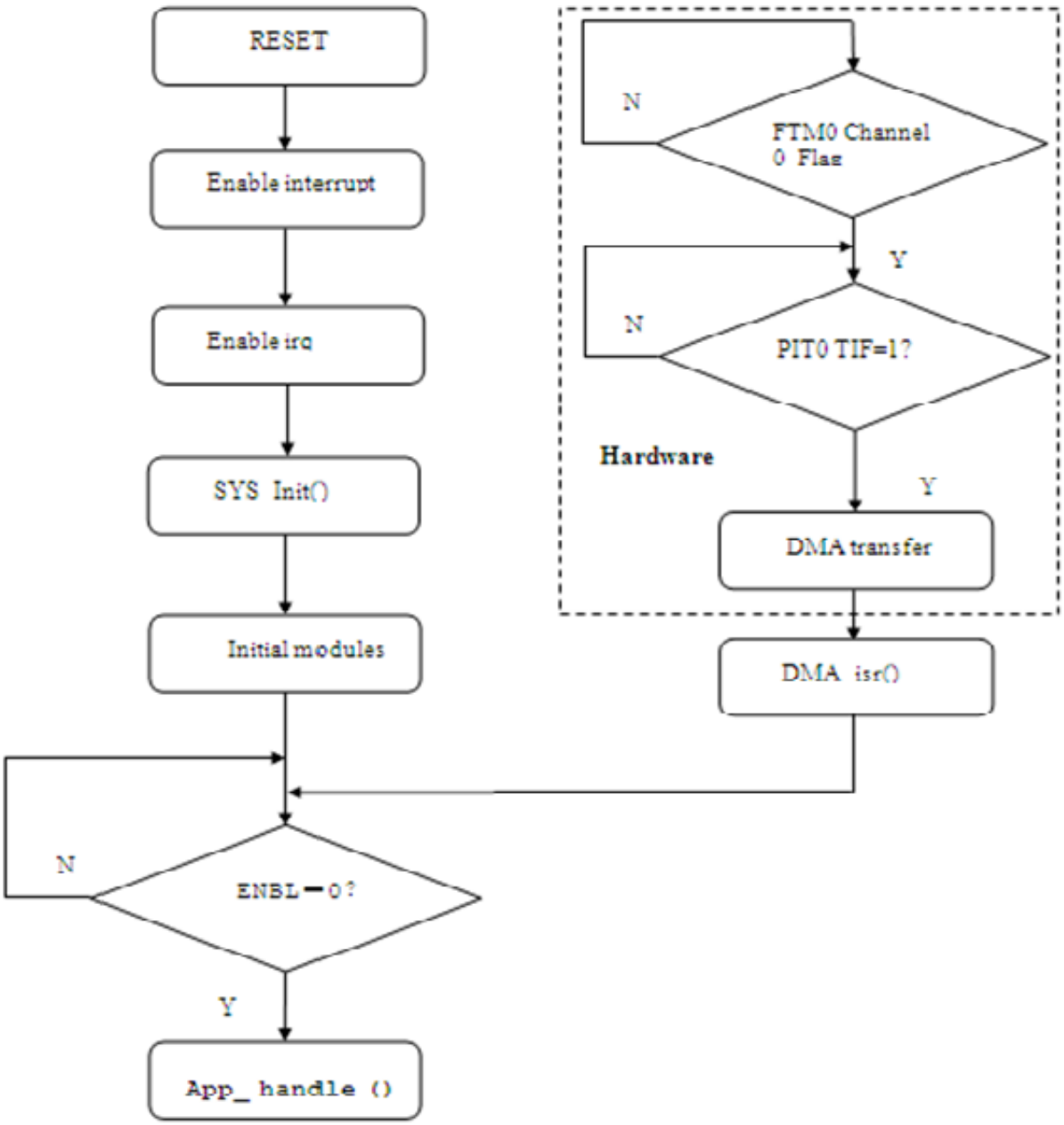

Figure 4. Flow chart for application code

5 Conclusion

The Periodic Trigger mode function of DMA MUX is completed by internal hardware logic. The user has to caculate only the intended trigger period. Usually, data movements are controlled in RAM and not by software. If the trigger period is too long, a DMA transfer is not completed before next peripheral request is valid. The source data may be covered by the new data, but it leads to losing data.

6 Example code

```
uint8_t Data_Source[8]={0x01,0x23,0x45,0x67,0x89,0xAB,0xCD,0xEF};//defines source data space
uint8_t Data_Desti[8]; //defines destination data space
void main(void)
{
EnableInterrupts;
enable_irq(INT_DMA0-16);
Sys_Init();//configures ICS/MCG,UART module,system clock is 72 MHz, bus clock is 36 MHz
/*PTE6 initial*/
PORTE_PCR6 |= PORT_PCR_MUX(1);
GPIOE_PDDR |= GPIO_PDDR_PDD(0x40);
GPIOE_PDOR |= 0x40;
/*DMA MUX initialization*/
SIM_SCGC6 |= SIM_SCGC6_DMA MUX_MASK;//enables DMA MUX clock gate
DMA MUX0_CHCFG0 = 0;//clears register for changing source and trigger
DMA MUX0_CHCFG0 = DMA MUX_CHCFG_ENBL_MASK //enables DMA MUX channel
                                          |DMA MUX_CHCFG_TRIG_MASK //enables Periodic Trigger
mode,
                                          |DMA MUX_CHCFG_SOURCE(18);//source is FTM0 channel 0
/*eDMA module initialization*/
//SIM_SCGC7 |= SIM_SCGC7_DMA_MASK;//enables DMA clock gate,default is enable
DMA_ERQ = 0x01;//enables DMA0 request
DMA_TCD0_SADDR = (uint32_t)(&Data_Source[0]);//defines source data address
DMA_TCD0_SOFF = 1;//Source address signed offset
DMA_TCD0_DADDR = (uint32_t)(&Data_Desti[0]);//defines destination data address
DMA_TCD0_CITER_ELINKNO = 0x01;//CITER=1
DMA_TCD0_BITER_ELINKNO = 0x01;//BITER=1
DMA_TCD0_NBYTES_MLNO = 8;//byte number
DMA_TCD0_DOFF = 1;//destination address signed offset
DMA_TCD0_ATTR = 0;//8 bit transfer size, register default value is undefined
DMA_TCD0_SLAST = -8;//restores the source address to the initial value
DMA_TCD0_DLASTSGA = -8;//restores the destination address to the initial value
DMA_TCD0_CSR = DMA_CSR_INTMAJOR_MASK;//The end-of-major loop interrupt is enabled
/*FTM0 initialization*/
SIM_SCGC6 |= SIM_SCGC6_FTM0_MASK;//enables FTM0 clock gate
PORTE_PCR3 = PORT_PCR_MUX(4);//sets PTE4 as FTM0_CH0 output pin
FTM0_C0SC = 0x55;//output compares toggle output on match, enables DMA transfer for the
channel
            FTM0_C0V = 35;// channel value, channel flag is set when FTM0_CNT value is 35
            FTM0_MOD = 72;// modulo value, period is 1 µs
FTM0_SC = FTM_SC_CLKS(1);//selects system clock as FTM0 clock source, starts FTM0 counter
/*PIT initialization*/
SIM_SCGC6 |= SIM_SCGC6_PIT_MASK; //enables PIT clock gate
PIT_MCR &= ~PIT_MCR_MDIS_MASK;//turns on PIT
PIT_LDVAL0 = 71; //PIT trigger is 1 µs
PIT_TCTRL0 |= PIT_TCTRL_TIE_MASK;//enables PIT timer interrupt
PIT_TCTRL0 |= PIT_TCTRL_TEN_MASK;//enables timer0
//PIT_TFLG0 |= PIT_TFLG_TIF_MASK;
App_handle();
for(; ;)
 {
 }
}

void App_handle(void)
{
 uint8_t i,error;
 while(DMA MUX0_CHCFG0&DMA MUX_CHCFG_ENBL_MASK);
  for(i=0;i<8;i++)
  {
if(Data_Source[i]==Data_Desti[i])
{
  error=0;
}
```

```
else
{
  error++;
}
  }
  if(error>0)
  {
for(i=0;i<8;i++)
{
  printf("Data_Desti[%d]=0x%02x\n",i,Data_Desti[i]);
  printf("Data_Source[%d]=0x%02x\n\n",i,Data_Source[i]);
  GPIOE_PDOR &= ~0x40;//turn on LED
}
  }
  else
  {
for(i=0;i<8;i++)
{
  printf("Data_Desti[%d]=0x%02x\n",i,Data_Desti[i]);
  printf("Data_Source[%d]=0x%02x\n\n",i,Data_Source[i]);
}
  }

  while(1);
}

void DMA0_isr(void)
{
DMA_MUX0_CHCFG0 = 0;
DMA_INT =0x01;
DMA_CDNE = 0x01;
}
```

NOTE

Sys_Init() is a basic system for initializing function, including configuration of MCG/ ICS, UART, and PORT. This application code is based on 72 MHz core clock(system clock) and 36 MHz bus clock.

7 References

- Cortex®-M4 Technical Reference Manual, Revision r0p1, available at http://infocenter.arm.com/help/index.jsp

8 Glossary

Table 3. Glossary of terms used in this application note

FTM	FlexTimer Module
ISR	Interrupt Service Routine
PIT	Periodic Interrupt Timer
DMA MUX	Direct Memory Access Multiplexer

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

Document Number: AN4639
Rev. 0, 12/2012