PIT Module Timer 0 for Timer Driver

The PIT module on the KL05 has two timers: timer 0 and timer 1. Each timer is a 32-bit down counter that loads its timer start value (*TSV*), counts down on every bus clock cycle until the counter value is zero, and then repeats by loading *TSV*. When timer 0 is configured to generate an interrupt request, the PIT timer 0 interrupt will occur after the count has reached 0 when *TSV* is loaded, so an interrupt occurs with a predictable period, (i.e., *TSV* + 1 bus clock cycles). Keeping track of the number of these interrupts that have occurred can then be used to determine timing with an approximate resolution of the interrupt period. Of the PIT registers, which map to the memory space of the KL05, the following registers are used for this functionality.

Symbol[1]	**Register Name**	**Address**	**Base**[2]	**Displacement**[3]
PIT_MCR	PIT Module Control Register	0x40037000	PIT	PIT_MCR
PIT_LDVAL0	Timer Load Value Register 0	0x40037100	PIT_CH0	PIT_LDVAL
PIT_TCTRL0	Timer Control Register 0	0x40037108	PIT_CH0	PIT_TCTRL
PIT_TFLG0	Timer Flag Register 0	0x4003701C	PIT_CH0	PIT_TFLG

[1]Defined in MKL05Z4.s include file
[2]Append _BASE to the name given in the column; defined in MKL05Z4.s include file
[3]Append _OFFSET to the name given in the column; defined in MKL05Z4.s include file

Using the PIT timer 0 in a program for timing involves configuring timer 0 for the desired time. Since it is a 32-bit counter, the desired time in terms of bus clock cycles must be able to be expressed as a 32-bit unsigned number. This document first describes initializing the PIT for a timer 0 interrupt every 0.01s, and then it discusses the PIT interrupt service routine.

PIT initialization

How the PIT is initialized depends on the KL05 clock configuration since the PIT timers count on every cycle of the bus clock. The Start.s startup module provided for this course has a subroutine Startup to configure a 48-MHz core clock (47,972,352 Hz)for the KL05 and a 24-MHz bus clock (23,986,176 Hz) before any PIT initializations. (Note that this subroutine call from main is part of the program template for hardware programs provided for this course.)

Also before the PIT can be initialized, its clock must be enabled. The assembly language code that follows enables the PIT module clock.

```
;Enable clock for PIT module
            LDR   Ri,=SIM_SCGC6
            LDR   Rj,=SIM_SCGC6_PIT_MASK
            LDR   Rk,[Ri,#0]
```

```
ORRS  Rk,Rk,Rj
STR   Rk,[Ri,#0]
```

In case PIT timer 0 has been generating interrupts with a different interrupt period, timer 0 should be disabled to prevent any further interrupts with the old period. Clearing bit 0 (TEN) of Timer Control Register 0 (PIT_TCTRL0) disables timer 0, as accomplished by the following assembly language code.

```
;Disable PIT timer 0
LDR   Ri,=PIT_CH0_BASE
LDR   Rj,=PIT_TCTRL_TEN_MASK
LDR   Rk,[Ri,#PIT_TCTRL_OFFSET]
BICS  Rk,Rk,Rj
STR   Rk,[Ri,#PIT_TCTRL_OFFSET]
```

Next, the KL05 NVIC must be configured for PIT interrupts. The interrupt priority for the PIT interrupt should be set, and the PIT interrupt should be unmasked to the CPU. The following assembly language code sets PIT Timer 0 for priority 0 (highest) and unmasks the PIT interrupt request. All labels not defined below are from the provided MKL05Z4.s include file.

```
;NVIC_ICPR
;31-00:CLRPEND=pending status for HW IRQ sources;
;             read:   0 = not pending;  1 = pending
;             write:  0 = no effect;
;                     1 = change status to not pending
;22:PIT IRQ pending status
NVIC_ICPR_PIT_MASK  EQU  PIT_IRQ_MASK
;--------------------------------------------------------
;NVIC_IPR0-NVIC_IPR7
;2-bit priority:  00 = highest; 11 = lowest
;--PIT
PIT_IRQ_PRIORITY    EQU  0
NVIC_IPR_PIT_MASK   EQU  (3 << PIT_PRI_POS)
NVIC_IPR_PIT_PRI_0  EQU  \
                    (PIT_IRQ_PRIORITY << PIT_PRI_POS)
;--------------------------------------------------------
;NVIC_ISER
;31-00:SETENA=masks for HW IRQ sources;
;             read:   0 = masked;     1 = unmasked
;             write:  0 = no effect;  1 = unmask
;22:PIT IRQ mask
NVIC_ISER_PIT_MASK  EQU  PIT_IRQ_MASK
;--------------------------------------------------------
;Set PIT interrupt priority
LDR     Ri,=PIT_IPR
LDR     Rj,=NVIC_IPR_PIT_MASK
;LDR     Rk,=NVIC_IPR_PIT_PRI_0
LDR     Rl,[Ri,#0]
BICS    Rl,Rl,Rj
;ORRS    Rl,Rl,Rk
STR     Rl,[Ri,#0]
;Clear any pending PIT interrupts
```

```
LDR     Ri,=NVIC_ICPR
LDR     Rj,=NVIC_ICPR_PIT_MASK
STR     Rj,[Ri,#0]
;Unmask PIT interrupts
LDR     Ri,=NVIC_ISER
LDR     Rj,=NVIC_ISER_PIT_MASK
STR     Rj,[Ri,#0]
```

The MDIS field (bit 1) of the PIT Module Control Register (PIT_MCR) must be configured to enable the module before any other setup is done. To enable the module, MDIS must be cleared. If debugging will occur with the timer driver, it may also be desirable to stop timers while in debug mode by setting bit 0 (FRZ).

PIT timer 0 functions as a modulo-*n* down counter if the 32-bit TSV in Timer Load Value Register 0 (PIT_LDVAL0) is set to $n - 1$. *TSV* can be calculated using the following formula.

$$TSV = \frac{t_{Interrupt}}{T_{TimerClock}} - 1 = \frac{t_{TimerClockRate}}{T_{InterruptRate}} - 1$$

For an interrupt every 0.01s with a timer clock rate of 23,986,176 Hz, *TSV* should be 239,861.

Finally, PIT timer 0 needs to be enabled and set to generate an interrupt. Setting bit 0 (TEN) of Timer Control Register 0 (PIT_TCTRL0) enables timer channel 0. Setting bit 1 (TIE) enables timer channel 0 to generate an interrupt request after the timer value reaches 0 when loading *TSV*.

The following assembly language code configures the PIT to generate an interrupt from timer 0 every 0.01 s. All labels not defined below are from the provided MKL05Z4.s include file.

```
;PIT_LDVALn:  PIT load value register n
;31-00:TSV=timer start value (period in clock cycles - 1)
;Clock ticks for 0.01 s at ~24 MHz count rate
;0.01 s * ~24,000,000 Hz = ~240,000
;TSV = ~240,000 - 1
;Clock ticks for 0.01 s at 23,986,176 Hz count rate
;0.01 s * 23,986,176 Hz = 239,862
;TSV = 239,862 - 1
PIT_LDVAL_10ms  EQU  239861
;-------------------------------------------------------
;PIT_MCR:  PIT module control register
;1-->0:FRZ=freeze (continue'/stop in debug mode)
;0-->1:MDIS=module disable (PIT section)
;           RTI timer not affected
;           must be enabled before any other PIT setup
PIT_MCR_EN_FRZ  EQU  PIT_MCR_FRZ_MASK
```

```
;-------------------------------------------------------
;PIT_TCTRLn:  PIT timer control register n
;0-->2:CHN=chain mode (enable)
;1-->1:TIE=timer interrupt enable
;1-->0:TEN=timer enable
PIT_TCTRL_CH_IE  EQU  (PIT_TCTRL_TEN_MASK :OR: \
                       PIT_TCTRL_TIE_MASK)
;-------------------------------------------------------
;Enable PIT module
LDR   Ri,=PIT_BASE
LDR   Rj,=PIT_MCR_EN_FRZ
STR   Rj,[Ri,#PIT_MCR_OFFSET]
;Set PIT timer 0 period for 0.01 s
LDR   Ri,=PIT_CH0_BASE
LDR   Rj,=PIT_LDVAL_10ms
STR   Rj,[Ri,#PIT_LDVAL_OFFSET
;Enable PIT timer 0 interrupt
LDR   Rj,=PIT_TCTRL_CH_IE
STR   Rj,[Ri,#PIT_TCTRL_OFFSET]
```

PIT Channel 0 Interrupts

After PIT timer 0 (down counter) reaches the value 0 and loads *TSV*, the timer interrupt flag (TIF) of Timer Flag Register 0 (PIT_TFLG0) is set to 1. If PIT channel 0 interrupts are enabled, TIF's being set causes a PIT interrupt. Writing a 1 to this flag clears the flag and thus the current PIT timer 0 interrupt condition.