TPM PWM Waveform Generation to Control LED Brightness

The timer/PWM module (TPM) of the KL05 is based on a 16-bit counter. The counter can be configured for free-running or modulo operation, and it can function as an up counter or an up-down counter. The KL05 has two TPMs: TPM0 and TPM1. Each TPM has six channels, some of which can connect to physical pins on the KL05Z board. There are general registers for each TPM, as well as specific registers for each channel of the module.

Each TPM and its channels can provide several functions. One function provided by the overall module is detection of timer overflow, which is determined by the module's MOD value in its control and status register. In addition, each channel of the module can be configured for one of three functions: input capture, output compare, or PWM, (a square wave). Input capture for a channel refers to recording the counter value when the signal connected to that channel's pin has a rising edge and/or falling edge. Output compare for a channel refers to setting a flag when the counter value reaches the value of the channel's VAL value in its counter value register with the option to set, clear, pulse, or toggle the output of that channel's pin. PWM pulses for a channel can be set as center- or edge-aligned as well as either low or high.

Although an LED is either fully on or fully off, if a PWM signal is applied to an LED, the perceived brightness of the LED can be controlled by the PWM signal's duty cycle: the pulse duration relative to the signal's period. If the period is short enough, (or equivalently the frequency is high enough), the human vision system perceives the flickers fused together as a constant light source with an intensity equal to the average brightness. Since the KL05Z circuit for the LED PWM connected to one side and a pull-up resistor connected to the other side, adjusting the LED's brightness from full illumination to no illumination is achieved by varying the duty cycle from 0% to 100%, respectively. (Given the complexity of the human vision system, a comprehensive analysis of all the factors and possible effects involved with pulsed light are beyond the scope of this document, but this superficial analysis is sufficient to support varying the brightness of the red LED on the KL05Z board using PWM.) Figure 1 shows the waveform of a PWM signal. The pulse time t_{Duty} relative to the period determines the duty cycle, as follows.

$$DutyCycle = 100\left(\frac{t_{Duty}}{t_{Period}}\right)\%$$

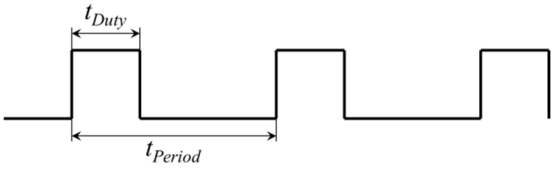

Figure 1. PWM Waveform

Using the TPM in a program to control the brightness of the red LED on the KL05Z involves configuring a single TPM channel connected to the LED for a PWM waveform. The clock source for the TPM must be specified, and the TPM's clock rate must be set. The period of the waveform and the duration of its duty cycle must then be specified in terms of the TPM's clock period. Also, the polarity and alignment of the PWM signal from the channel must be specified. On the KL05Z board, TPM0 channel 3 connects through KL05 port B pin 8 to the red LED.

Program initialization

Lab Exercise Eleven explores several PWM frequencies for controlling red LED brightness. The initialization derived here results in the red LED on at full brightness and configures the lowest of these PWM frequencies, which is 60 Hz. This rate corresponds to a PWM signal with a $16^2/_3$-ms period from TPM0 channel 3 that can be accessed through port B pin 8 connected to the red LED on the KL05Z. The following settings are required for this initialization.

- TPM0 module clock enabled in SIM_SCGC6: TPM0 = 1
- Port B module clock enabled in SIM_SCGC5: PORTB = 1
- TPM0 channel 3 connected to port B pin 8 in PORTB_PCR8:
  MUX = 010_2
- TPM clock source in SIM_SOPT2 (see analysis that follows):
  TPMSRC = 01
- TPM0 counter modulo value in TPM0_MOD (see analysis that follows):
  MOD = 49,970
- TPM0 channel 3 edge-aligned PWM, no overflow, and no DMA in TPM0_C2SC:
  CHIE = 0; MSB = 1; MSA = 0; ELSB = 1; ELSA = 0; DMA = 0
- TPM0 channel 3 value in TPM0_C2V (see analysis that follows):
  VAL = 0
- TPM0 counter clock configuration, (see analysis that follows), no DMA, no overflow interrupt, and edge-aligned PWM in TPM0_SC:
  DMA = 0; TOIE = 0; CPWMS = 0; CMOD = 01; PS = 100_2

Analysis. Since the TPM has a 16-bit counter, the TPM counter clock frequency must be low enough that the ratio of it to 60 Hz can be represented in 16 bits, (i.e., the number of TPM counter clock periods in $16^2/_3$ ms is less than 65536). On the other hand, it is desirable for the clock frequency to be as high as possible for greatest resolution of the LEDs brightness.

The TPM counter clock source and frequency are determined by three settings: one field of the system integration module (SIM) system options register 2 SIM_SOPT2 and two fields of the TPM0 status and control register TPM0_SC. TPMSRC = 01 in SIM_SOPT2 select a 47972352-Hz clock input (~48 MHz) to the TPM. CMOD = 1 in TPM0_SC sets the TPM counter to count, (i.e., increment or decrement, depending on up or up/down configuration), every TPM counter clock cycle. PS in TPM0_SC determines the prescale

factor applied to the TPM clock input to produce the TPM counter clock, according to the following formula.

$$TPM\_CounterClockRate = \frac{TPM\_ClockInputRate}{2^{PS}} \quad (1)$$

There are constraints to guide the selection of the value used for PS. First, the period and the minimum pulse duration must both be representable as unsigned 16-bit integers. For a 60-Hz PWM signal, the period is the number of TPM counter clock periods in $16^2/_3$ ms, (the PWM period). (This number is the same as the ratio of the TPM counter clock rate to the PWM rate of 60 Hz.) If a minimum brightness of 5% is desired, then the minimum PWM pulse duration is the number of TPM counter clock periods in 5% of $16^2/_3$ ms. (This number is the same as the ratio of 5% of the TPM counter clock rate to the PWM rate of 60 Hz.) These two numbers establish minimum and maximum limits for PS, as expressed by the following inequality based on frequency ratios.

$$1 \leq \frac{5\% \times TPM\_CounterClockRate}{60\text{ Hz}} < \frac{TPM\_CounterClockRate}{60\text{ Hz}} < 2^{16}$$

Substituting (1) gives the following.

$$1 \leq \frac{TPM\_ClockInputRate}{(1200\text{ Hz})(2^{PS})} < \frac{TPM\_ClockInputRate}{(60\text{ Hz})(2^{PS})} < 2^{16}$$

Multiplying throughout by 2^{PS} results in the following inequality.

$$2^{PS} \leq \frac{TPM\_ClockInputRate}{1200\text{ Hz}} < \frac{TPM\_ClockInputRate}{60\text{ Hz}} < 2^{PS}2^{16}$$

Solving the leftmost inequality for PS, gives the following.

$$\begin{aligned} PS &\leq \log_2\left(\frac{TPM\_ClockInputRate}{1200\text{ Hz}}\right) \\ &\leq \log_2\left(\frac{48{,}000{,}000\text{ Hz}}{1200\text{ Hz}}\right) = \log_2(40{,}000) \\ &\leq 15.288 \end{aligned}$$

Then solving the rightmost part for PS gives the following.

$$\begin{aligned} PS &> \log_2\left(\frac{TPM\_ClockInputRate}{60\text{ Hz}}\right) - 16 \\ &> \log_2\left(\frac{48{,}000{,}000}{60\text{ Hz}}\right) - 16 = \log_2(800{,}000) - 16 = 19.6096 - 16 \\ &> 3.6096 \end{aligned}$$

Thus, the following inequality expresses the constraint on PS for PWM timing, given that PS must be an integer.

$$4 \leq PS \leq 15 \quad (2)$$

Second, PS is an unsigned 3-bit integer field in TPM0_SC, which further constrains its possible values.

$$0 \leq PS \leq 7 \quad (3)$$

Therefore, PS must satisfy the following inequality to comply with both (2) and (3).

$$4 \leq PS \leq 7 \quad (4)$$

Table 1 summarizes the PWM timing parameters for each possible value of PS. Note that the number of periods in $16^2/_3$ ms for $0 \leq PS < 4$ is too large to express in a 16-bit integer, as predicted by (4). Also note that none of the periods can be expressed as an integer, so the TPM periods will be slightly off from the desired calculated periods.

Table 1. Parameters for TPM 60-Hz PWM with 48-MHz Clock Input

PS	2^{PS}	*TPM Count Rate* (Hz)	*5% Duty: $833^1/_3$ ns (periods)*	*$16^2/_3$ ms (periods)*
0	1	47972352	39976.96	799,539.2
1	2	23986176	19988.48	399,769.6
2	4	11993088	9994.24	199,884.8
3	8	5996544	4997.12	99,942.4
4	16	2998272	2498.56	49,971.2
5	32	1499136	1249.28	24,985.6
6	64	749568	624.64	12,492.8
7	128	374784	312.32	6,246.4

Thus, one way to get an acceptable TPM count rate is to set PS to $100_2 = 4_{10}$. Of the acceptable values for PS, (i.e., $4 \leq PS < 7$), choosing a lower value for PS allows more precise timing resolution, whereas choosing a higher value of PS could facilitate a longer PWM period. Since there will be a fixed PWM period for this application, the lowest usable value of PS will be used.

The 16-bit MOD field in the modulo register TPM0_MOD determines the period of the PWM signal. The value in this field corresponds to the number of TPM counter clock periods in one period of the PWM signal as specified in the following equation.

MOD = *TPM counter clock periods* − 1

As seen in Table 1, for PS = 4 this value should be 49,971 – 1 = 49,970.

Similarly, the 16-bit VAL field in the channel 3 value register TPM0_C3V determines the duty cycle of the PWM signal. The value in this field corresponds to the number of TPM counter clock periods during the pulse, (i.e., duty cycle or high portion), of the PWM signal. For maximum LED brightness, which corresponds to the LED always on at a 0% duty cycle, the value should be 0. For minimum LED brightness, (i.e., LED off at 100% duty cycle), the value should be 49,970. Values between 0 and 49,970 would appear to illuminate the LED at brightness levels between these two extremes.

Assembly language code. The following assembly language code generates a PWM signal with a $16^2/_3$-ms period from TPM0 channel 3 that can be accessed through port B pin 8 connected to the red LED on the KL05Z. All symbols not defined in this code are from the RIT CMPE-250 include file MKL05Z4.s.

```
;PWM frequencies
PWM_FREQ          EQU  60
TPM_SOURCE_FREQ   EQU  47972352
TPM_SC_PS_VAL     EQU  4
PWM_PERIOD        EQU  ((TPM_SOURCE_FREQ \
                         (1 << TPM_SC_PS_VAL)) / PWM_FREQ)
;PortB pin 8 symbols
PTB8_MUX_TPM0_CH3_OUT     EQU  (3 << PORT_PCR_MUX_SHIFT)
SET_PTB8_TPM0_CH3         EQU  (PIN_ISF_MASK :OR: \
                                PTB8_MUX_TPM0_CH3_OUT)
;SIM_SOPT2 symbols
SIM_SOPT2_TPMSRC_MCGFLLCLK  EQU (1 << SIM_SOPT2_TPMSRC_SHIFT)
;TPM0_CONF symbol
TPM_CONF_DEFAULT          EQU  0
;TPM0_CNT symbol
TPM_CNT_INIT              EQU  0
;TPM0_MOD symbol
TPM_MOD_PWM_PERIOD        EQU  (PWM_PERIOD - 1)
;TPM0_SC symbols
TPM_SC_CMOD_CLK           EQU  1
TPM_SC_PS_VAL             EQU  0x4
TPM_SC_CLK_DIV16          EQU  ((TPM_SC_CMOD_CLK << \
                                 TPM_SC_CMOD_SHIFT) | \
                                TPM_SC_PS_VAL)
;TPM0_C3SC symbol
TPM_CnSC_PWMH             EQU  (TPM_CnSC_MSB_MASK | \
                                TPM_CnSC_ELSB_MASK)
;TPM0_C3V symbols
TPM_CnV_PWM_DUTY_LED_OFF  EQU  TPM_MOD_PWM_PERIOD
TPM_CnV_PWM_DUTY_LED_ON   EQU  0
```

```
;Enable TPM0 module clock
            LDR    Ri,=SIM_SCGC6
            LDR    Rj,=SIM_SCGC6_TPM0_MASK
            LDR    Rk,[Ri,#0]        ;current SIM_SCGC6 value
            ORRS   Rk,Rk,Rj          ;only TPM0 bit changed (set)
            STR    Rk,[Ri,#0]        ;update SIM_SCGC6
;Enable port B module clock
            LDR    Ri,=SIM_SCGC5
            LDR    Rj,=SIM_SCGC5_PORTB_MASK
            LDR    Rk,[Ri,#0]        ;current SIM_SCGC5 value
            ORRS   Rk,Rk,Rj          ;only PORTB bit changed (set)
            STR    Rk,[Ri,#0]        ;update SIM_SCGC6
;Connect TPM0 channel 3 to port B pin 8
            LDR    Ri,=PORTB_PCR8
            LDR    Rj,=SET_PTB8_TPM0_CH3
            STR    Rj,[Ri,#0]        ;PortE_PCR29 = TPM0_CH2
;Set TPM clock source
            LDR    Ri,=SIM_SOPT2
            LDR    Rj,=SIM_SOPT2_TPMSRC_MCGFLLCLK
            LDR    Rk,=SIM_SOPT2_TPMSRC_MASK
            LDR    Rh,[Ri,#0]        ;current SIM_SOPT2 value
            BICS   Rh,Rh,Rk          ;clear TPMSRC field
            ORRS   Rh,Rh,Rj          ;only TPM bits changed
            STR    Rk,[Ri,#0]        ;update SIM_SOPT2
;Set TPM0 configuration to default value
            LDR    Ri,=TPM0_BASE
            LDR    Rj,=TPM_CONF_DEFAULT
            STR    Rj,[Ri,#TPM_CONF_OFFSET]
;Set TPM0 counter modulo value
            LDR    Ri,=TPM0_BASE
            LDR    Rj,=TPM_CNT_INIT
            LDR    Rk,= TPM_MOD_PWM_PERIOD
            STR    Rj,[Ri,#TPM_CNT_OFFSET]
            STR    Rk,[Ri,#TPM_MOD_OFFSET]
;Set TPM0 channel 3 edge-aligned PWM
            LDR    Rj,=TPM_CnSC_PWMH
            STR    Rj,[Ri,#TPM_C3SC_OFFSET]
;Set TPM0 channel 3 value
            LDR    Rj,= TPM_CnV_PWM_DUTY_LED_ON
            STR    Rj,[Ri,#TPM_C3V_OFFSET]
;Set TPM0 counter clock configuration
            LDR    Rj,= TPM_SC_CLK_DIV16
            STR    Rj,[Ri,#TPM_SC_OFFSET]
```

C code. The following C code generates a PWM signal with a $16^2/_3$-ms period from TPM0 channel 3 that can be accessed through port B pin 8 connected to the red LED on the KL05Z. All symbols not defined in this code are from the Keil Freescale/NXP include file MKL05Z4.h.

```
/* PWM frequencies */
#define PWM_FREQ        (60u)
#define TPM_SOURCE_FREQ (47972352u)
/* Port B pin 8 symbols */
#define PTB8_MUX_TPM0_CH3_OUT (2u << PORT_PCR_MUX_SHIFT)
#define SET_PTB8_TPM0_CH3_OUT (PORT_PCR_ISF_MASK | \
                               PTB8_MUX_TPM0_CH3_OUT)
/* SIM_SOPT2 symbols */
#define SIM_SOPT2_TPMSRC_MCGFLLCLK (1u << SIM_SOPT2_TPMSRC_SHIFT)
/* TPM0_CONF symbol */
#define TPM_CONF_DEFAULT (0)
/* TPM0_CNT symbol */
#define TPM_CNT_INIT (0)
/* TPM0_MOD symbol */
#define TPM_MOD_PWM_PERIOD ((TPM_SOURCE_FREQ / \
                             (1 << TPM_SC_PS_VAL)) / \
                            PWM_FREQ)
/* TPM0_SC symbols */
#define TPM_SC_CMOD_CLK (1u)
#define TPM_SC_PS_VAL  (0x4u)
#define TPM_SC_CLK_DIV ((TPM_SC_CMOD_CLK << \
                         TPM_SC_CMOD_SHIFT) | \
                        TPM_SC_PS_VAL)
/* TPM0_MOD symbols */
#define TPM_MOD_PWM_PERIOD ((TPM_SOURCE_FREQ / \
                             (1 << TPM_SC_PS_VAL)) / \
                            PWM_FREQ)
/* TPM0_C3SC symbol */
#define TPM_CnSC_PWMH (TPM_CnSC_MSB_MASK | TPM_CnSC_ELSB_MASK)
/* TPM0_C3V symbol */
#define TPM_CnV_PWM_DUTY_LED_OFF (TPM_MOD_PWM_PERIOD)
#define TPM_CnV_PWM_DUTY_LED_ON  (0)
```

```
/* Enable TPM0 module clock */
SIM->SCGC6 |= SIM_SCGC6_TPM0_MASK;
/* Enable port B module clock */
SIM->SCGC5 |= SIM_SCGC5_PORTB_MASK;
/* Connect TPM0 channel 3 to port B pin 8 */
PORTB->PCR[8] = SET_PTB8_TPM0_CH3_OUT;
/* Set TPM clock source */
SIM->SOPT2 &= ~SIM_SOPT2_TPMSRC_MASK;
SIM->SOPT2 |= SIM_SOPT2_TPMSRC_MCGFLLCLK;
/* Set TPM0 configuration register to default values */
TPM0->CONF = TPM_CONF_DEFAULT;
/* Set TPM0 counter modulo value */
TPM0->CNT = TPM_CNT_INIT;
TPM0->MOD = TPM_MOD_PWM_PERIOD - 1;
/* Set TPM0 counter clock configuration */
TPM0->SC = TPM_SC_CLK_DIV;
/* Set TPM0 channel 3 edge-aligned PWM */
TPM0->CONTROLS[3].CnSC = TPM_CnSC_PWMH;
/* Set TPM0 channel 3 value */
TPM0->CONTROLS[3].CnV = TPM_CnV_PWM_DUTY_LED_ON;
```

Program change of PWM to adjust LED brightness

During the course of lab, the brightness of the red LED on the KL05Z needs to be adjusted to each of five levels, which range from no illumination (1) to full illumination (5). Since the brightness of the LED depends on the duty period of the TPM0 channel 2 signal, changing the brightness of the LED means writing a new value to TPM0_C3V. The following constants create a lookup table in ROM for LED brightness duty periods, which can be accessed as an array with an index [0, 4] one less than the desired brightness level [1, 5].

```
PWM_DUTY_MAX  EQU  (PWM_PERIOD - 1)
              EXPORT  PWM_duty_table_0       ;include if accessed from C
PWM_duty_table
PWM_duty_table_0                             ;include if accessed from C
;LED brightness from dimmest (1) to brightest (5)
              DCW  PWM_DUTY_MAX              ;0% bright (off)
              DCW  (75 * PWM_DUTY_MAX / 100) ;25% bright
              DCW  (50 * PWM_DUTY_MAX / 100) ;50% bright
              DCW  (25 * PWM_DUTY_MAX / 100) ;75% bright
              DCW  0                         ;100% bright (always on)
```

Assembly language code. The following assembly language code could be used to set the red LED to brightness level 5.

```
   MOVS  Ri,#5         ;Desired brightness level
   SUBS  Ri,Ri,#1      ;Corresponding look-up table entry index
   LSLS  Ri,Ri,#1      ;Convert to byte index
   LDR   Rj,=PWM_duty_table
   LDRH  Rj,[Rj,Ri]    ;Corresponding look-up table entry
   LDR   Rk,=TPM0_C3V
   STR   Rj,[Rk,#0]    ;Change PWM duty cycle for desired position
```

C code. The following C declarations and code could be used to set the red LED at brightness level 5.

```
/* Prototype to enable access to ROM table from assembly language object */
extern const UInt16 PWM_duty_table_0; /* Provided in header file */
/* Initialization for C assess to assembly language ROM table */
const UInt16 *PWM_duty_table = &PWM_duty_table_0;

int    BrightLevel;

   BrightLevel = 5;                /* Desired brightness level */
/* Set duty cycle of TMP0 channel 3 for desired brightness level */
   TPM0->CONTROLS[3].CnV = PWM_duty_table [BrightLevel - 1];
```