Timer Controller (TMR):

- Four 32-bit timers, **TIMER0~TIMER3**
- Can be used for
  - frequency measurement
  - event counting
  - interval measurement
  - clock generation
  - delay timing
  - and so on

Timer Controller (TMR):

Features

- 4 sets of 32-bit timers with 24-bit up-timer and one 8-bit pre-scale counter
- Independent clock source for each timer
- Provides one-shot, periodic, toggle and continuous counting operation modes
- Support event counting function to count the event from external pin
- Support input capture function to capture or reset counter value
- 24-bit timer value is readable through TDR (Timer Data Register)

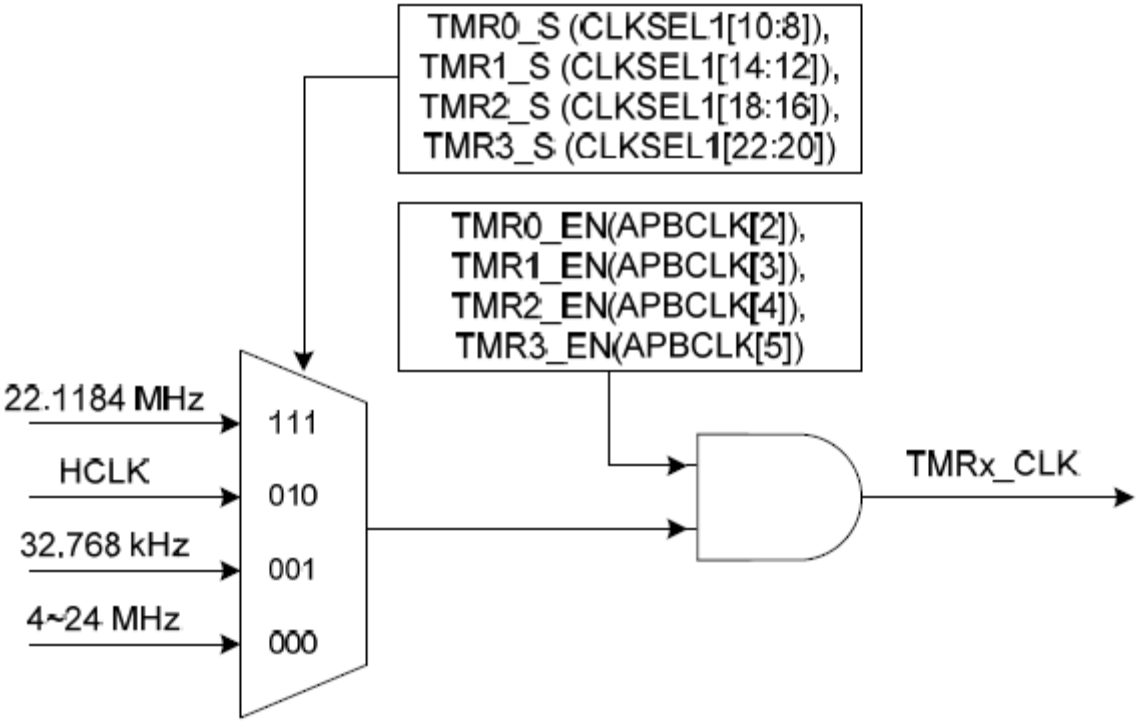
Timer Controller (TMR):

Features

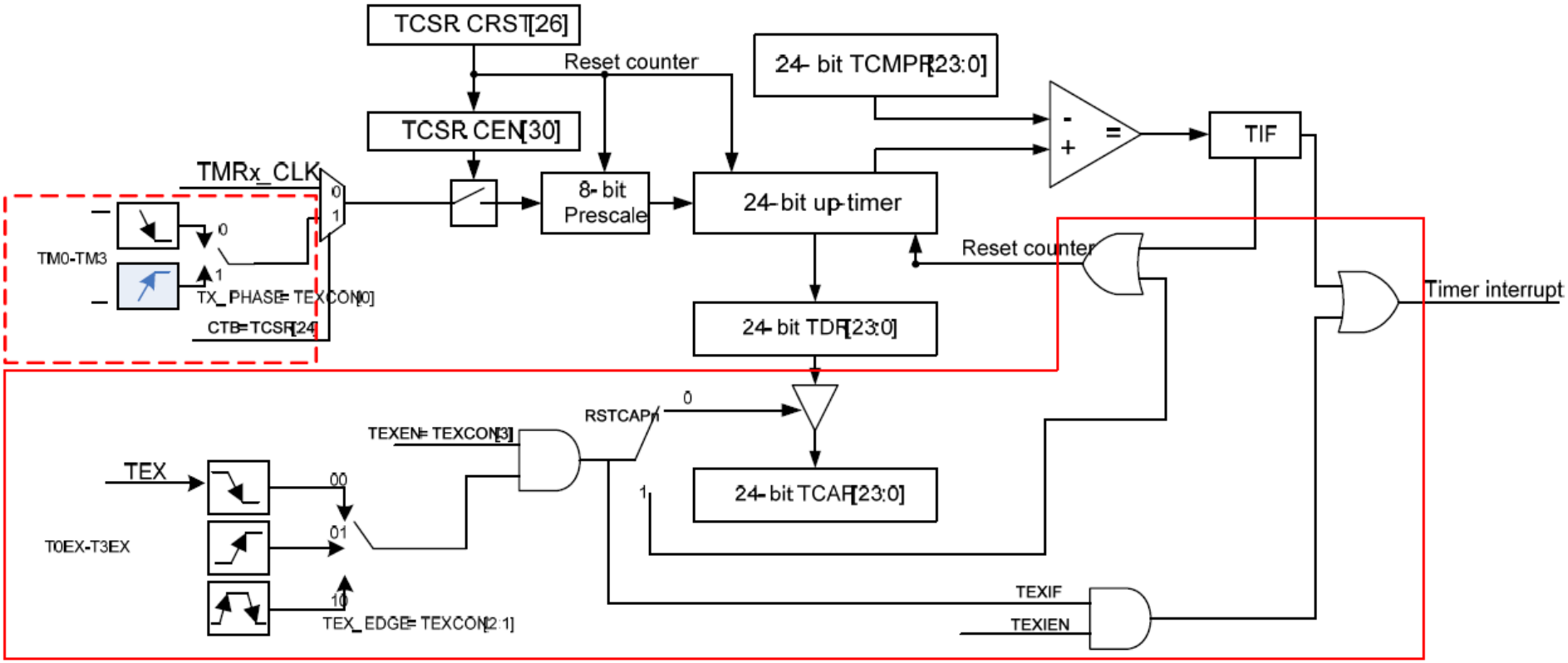
- Time out period = (Period of timer clock input) * (8-bit pre-scale counter + 1) * (24-bit **TCMP**)

- Maximum counting cycle time = (1 / T MHz) * (2^8) * (2^{24}), T is the period of timer clock

Timer Controller: Block Diagram



Timer Controller: Block Diagram



Timer Controller: Register Map

Register	Offset	R/W	Description	Reset Value
TMR_BA01 = 0x4001_0000 TMR_BA23 = 0x4011_0000				
TCSR0	TMR_BA01+0x00	R/W	Timer0 Control and Status Register	0x0000_0005
TCMPR0	TMR_BA01+0x04	R/W	Timer0 Compare Register	0x0000_0000
TISR0	TMR_BA01+0x08	R/W	Timer0 Interrupt Status Register	0x0000_0000
TDR0	TMR_BA01+0x0C	R	Timer0 Data Register	0x0000_0000
TCAP0	TMR_BA01+0x10	R	Timer0 Capture Data Register	0x0000_0000
TEXCON0	TMR_BA01+0x14	R/W	Timer0 External Control Register	0x0000_0000
TEXISR0	TMR_BA01+0x18	R/W	Timer0 External Interrupt Status Register	0x0000_0000

Timer Controller: One–Shot Mode

Once the timer counter value reaches timer compare register (TCMPR) value, if IE (TCSR[29] interrupt enable bit) is set to 1, then the timer interrupt flag is set and the interrupt signal is generated and sent to NVIC to inform CPU.
> It indicates that the timer counting overflow happens.

If IE (TCSR[29] interrupt enable bit) is set to 0, no interrupt signal is generated.

In this operating mode, once the timer counter value reaches timer compare register (TCMPR) value, the timer counter value goes back to counting initial value and CEN (timer enable bit) is cleared to 0 by timer controller.

Timer counting operation stops, once the timer counter value reaches timer compare register (TCMPR) value.

That is to say, timer operates timer counting and compares with TCMPR value function only one time after programming the timer compare register (TCMPR) value and CEN (timer enable bit) is set to 1.

Timer Controller: Periodic Mode

Once the timer counter value reaches timer compare register (TCMPR) value, if IE (TCSR[29] interrupt enable bit) is set to 1, then the timer interrupt flag is set and the interrupt signal is generated and sent to NVIC to inform CPU.
> It indicates that the timer counting overflow happens.

If IE (TCSR[29] interrupt enable bit) is set to 0, no interrupt signal is generated.

In this operating mode, once the timer counter value reaches timer compare register (TCMPR) value, the timer counter value goes back to counting initial value and CEN (timer enable bit) is kept at 1 (counting enable continuously). The timer counter operates up counting again.

The timer counting operation doesn't stop until the CEN is set to 0.
The interrupt signal is also generated periodically.

Timer Controller: Toggle Mode

Once the timer counter value reaches timer compare register (TCMPR) value,
if IE (TCSR[29] interrupt enable bit) is set to 1, then the timer interrupt flag is set
and the interrupt signal is generated and sent to NVIC to inform CPU.
> It indicates that the timer counting overflow happens.

The associated toggle output (tout) signal is set to 1.

If IE (TCSR[29] interrupt enable bit) is set to 0, no interrupt signal is generated.

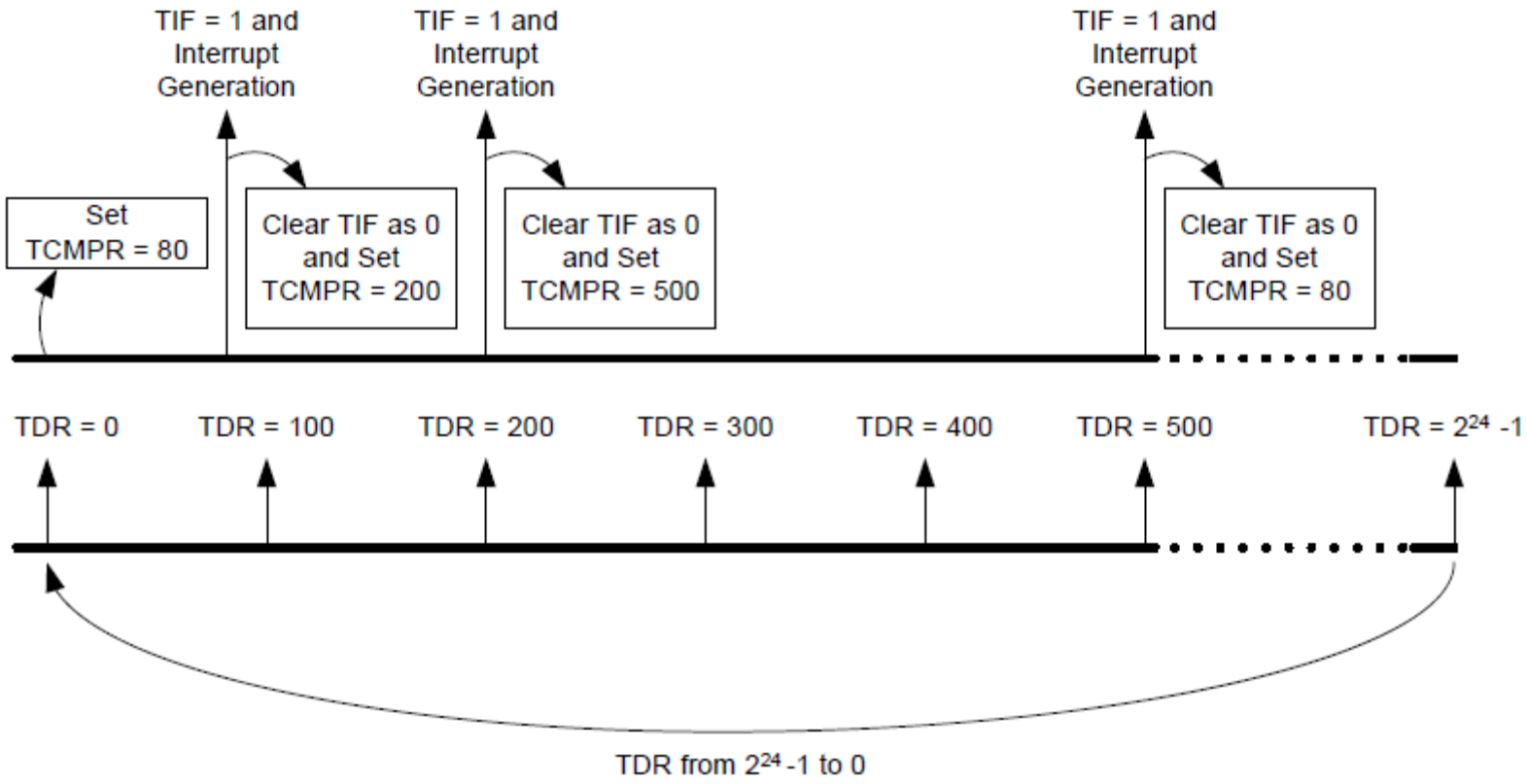
In this operating mode, once the timer counter value reaches timer compare
register (TCMPR) value, the timer counter value goes back to counting
initial value and CEN (timer enable bit) is kept at 1 (counting enable continuously).
The timer counter operates up counting again.

If the interrupt flag is cleared by software, once the timer counter value reaches timer compare register (TCMPR) value and IE (interrupt enable bit) is set to 1, then the timer interrupt flag is set and the interrupt signal is generated and sent to NVIC to inform CPU again. The associated toggle output (tout) signal is set to 0.

The timer counting operation doesn't stop until the CEN is set to 0.

Timer Controller: Continuous Counting Mode

User can change different TCMPR value immediately without disabling timer counting and restarting timer counting.



Timer Controller: Event Counting Function

from TM0~TM3 pins

In event counting function, the clock source of timer controller, TMRx_CLK should be set as HCLK.
It provides TM0~TM3 enabled or disabled de-bounce function by TEXCONx[7] and TM0~TM3 falling or rising phase counting setting by TEXCONx[0]. And, the event count source operating frequency should be less than 1/3 HCLK frequency if disable counting de-bounce or less than 1/8 HCLK frequency if enable counting de-bounce. Otherwise, the returned TDR value is incorrect.

Timer Controller: Input Capture Function

to capture or reset timer counter value

If TEXEN (Timer External Pin Enable) is set to 1 and RSTCAPSEL is set to 0, the timer counter value (TDR) will be captured into TCAP register when TEX (Timer External Pin) pin trigger condition occurred.
There are four TEX sources form specified pins, T0EX~T3EX pins.

If TEXEN is set to 1 and RSTCAPSEL is set to 1, the TDR will be reset to 0 when TEX pin trigger condition happened.
The TEX trigger edge can choose by TEX_EDGE.
When TEX trigger occurred, TEXIF (Timer External Interrupt Flag) is set to 1, and if enabled TEXIEN (Timer External Interrupt Enable Bit) to 1, the interrupt signal is generated then sent to NVIC to inform CPU.

It also provides T0EX~T3EX enabled or disabled capture de-bounce function by TEXCONx[6]. And, the TEX source operating frequency should be less than 1/3 HCLK frequency if disable TEX de-bounce or less than 1/8 HCLK frequency if enable TEX de-bounce.

Timer Controller: Example

```
void Timer_initial(void)
{
  /* Step 1. Enable and Select Timer clock source */
  SYSCLK->CLKSEL1.TMR0_S = 0; //Select 12Mhz for Timer0 clock source
  //DK: 0 = 12M, 1 = 32k, 2 = HCLK, 7 = 22M

  SYSCLK->APBCLK.TMR0_EN =1;  //Enable Timer0 clock source

  /* Step 2. Select Operation mode */
  TIMER0->TCSR.MODE=1;    //Select periodic mode for operation mode
  //DK: 0 = One shot, 1 = Periodic, 2 = Toggle, 3 = continuous counting mode

  /* Step 3. Select Time out period
    = (Period of timer clock input) * (8-bit Prescale + 1) * (24-bit TCMP)*/
  TIMER0->TCSR.PRESCALE=0;  // Set Prescale [0~255]
  TIMER0->TCMPR  = 1000000;   // Set TICR(TCMP) [0~16777215] : 24bits
                 // (1/12000000)*(0+1)*(1000000)= 83 msec or 12 Hz
                 // (1/22000000)*(0+1)*(1000000)= 45 msec (for 22MHz)

  /* Step 4. Enable interrupt */
  TIMER0->TCSR.IE = 1;
  TIMER0->TISR.TIF = 1;        // Write 1 to clear
  NVIC_EnableIRQ(TMR0_IRQn);  // Enable Timer0 Interrupt

  /* Step 5. Enable Timer module */
  TIMER0->TCSR.CRST = 1;     // Reset up counter
  TIMER0->TCSR.CEN = 1;      // Starts counting

  TIMER0->TCSR.TDR_EN=1;     // updated continuously with
  // the 24-bit up-timer value as the timer is counting.
}
```