SIOC嵌入式軟體實驗
實驗三：Timer

WU-YANG
Technology Co., Ltd.

support.wuyang@gmail.com

實驗目的

- Timer在嵌入式系統中常用於計時和PWM控制訊號輸出本章將探討ARM Cortex-M3 Timer ，並使讀者瞭解其應用方式。

實作重點

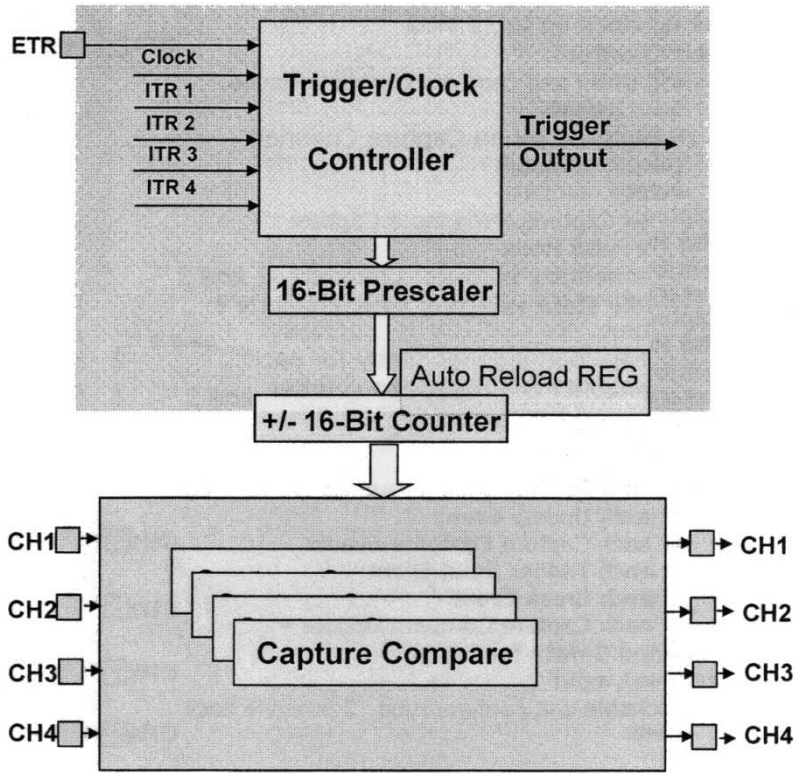
- Timer的控制
- 計時碼表設計
- Timer產生PWM訊號

STM32 timer分類

- Advanced Control Timer (TIM1)
- General-Purpose Timers(TIM2-TIM4)

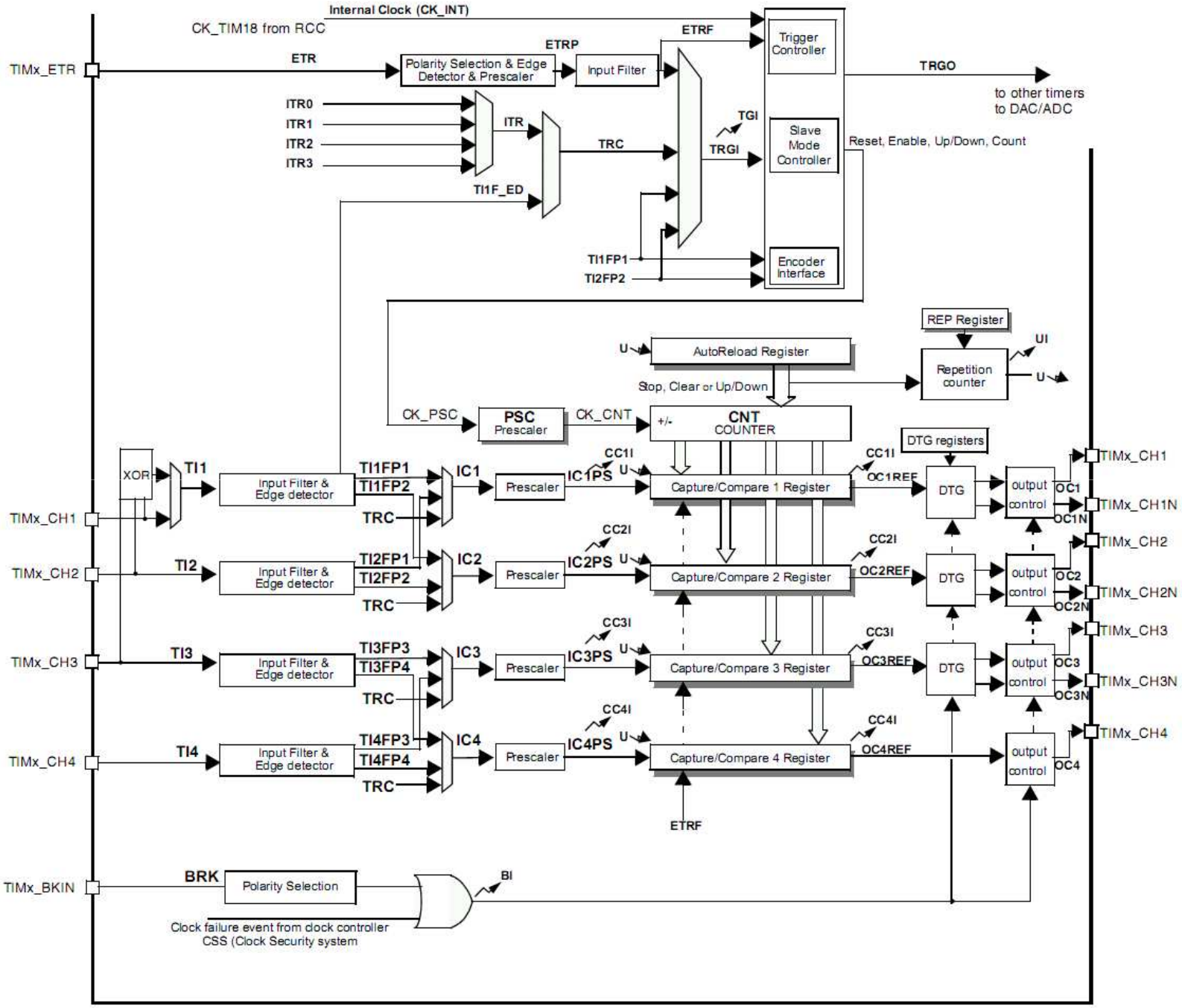
General Purpose timer

- 16bit Counter
  - ✓ Up counting mode
  - ✓ Down counting mode
  - ✓ Up /Down mode
- 四個獨立通道
  - ✓ 輸入捕獲
  - ✓ 輸出比較
  - ✓ PWM生成
  - ✓ 單脈衝模式輸出
- 使用外部信號控制定時器和定時器互聯的同步電路
- 如下事件發生時產生中斷/DMA
  - ✓ 更新
  - ✓ 觸發事件
  - ✓ 輸入捕獲
  - ✓ 輸出比較



Advanced control timer

- Complementary Outputs with programmable dead-time
- Break input to put the timer's output signals in reset state or in a known state.


Internal Clock (CK_INT)
CK_TIM18 from RCC
TIMx_ETR
ETR
ETRP
ETRF
Polarity Selection & Edge Detector & Prescaler
Input Filter
Trigger Controller
TRGO
to other timers
to DAC/ADC
ITR0
ITR1
ITR2
ITR3
ITR
TRC
TGI
TRGI
Slave Mode Controller
Reset, Enable, Up/Down, Count
TI1F_ED
TI1FP1
TI2FP2
Encoder Interface
REP Register
UI
U
AutoReload Register
Repetition counter
Stop, Clear or Up/Down
CK_PSC
PSC Prescaler
CK_CNT
+/-
CNT COUNTER
DTG registers
XOR
TI1
TI1FP1
TI1FP2
IC1
Input Filter & Edge detector
Prescaler
IC1PS
CC1I
Capture/Compare 1 Register
OC1REF
DTG
output control
OC1
OC1N
TIMx_CH1
TIMx_CH1N
TIMx_CH2
TIMx_CH2N
TIMx_CH3
TIMx_CH3N
TIMx_CH4
TI2
TI2FP1
TI2FP2
IC2
IC2PS
CC2I
Capture/Compare 2 Register
OC2REF
OC2
OC2N
TI3
TI3FP3
TI3FP4
IC3
IC3PS
CC3I
Capture/Compare 3 Register
OC3REF
OC3
OC3N
TI4
TI4FP3
TI4FP4
IC4
IC4PS
CC4I
Capture/Compare 4 Register
OC4REF
OC4
TIMx_BKIN
BRK
Polarity Selection
BI
Clock failure event from clock controller
CSS (Clock Security system

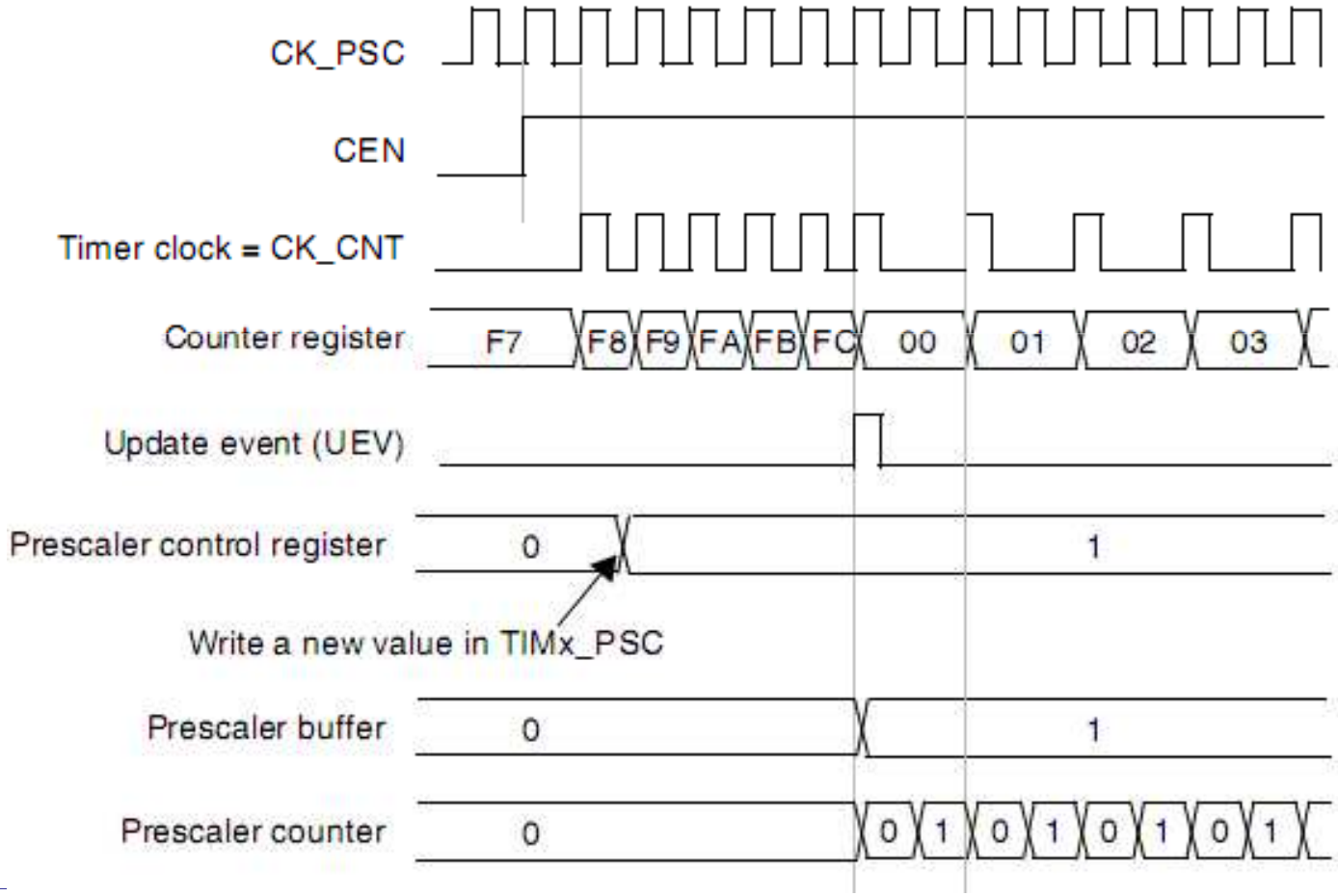
Timer Register

Time-base unit

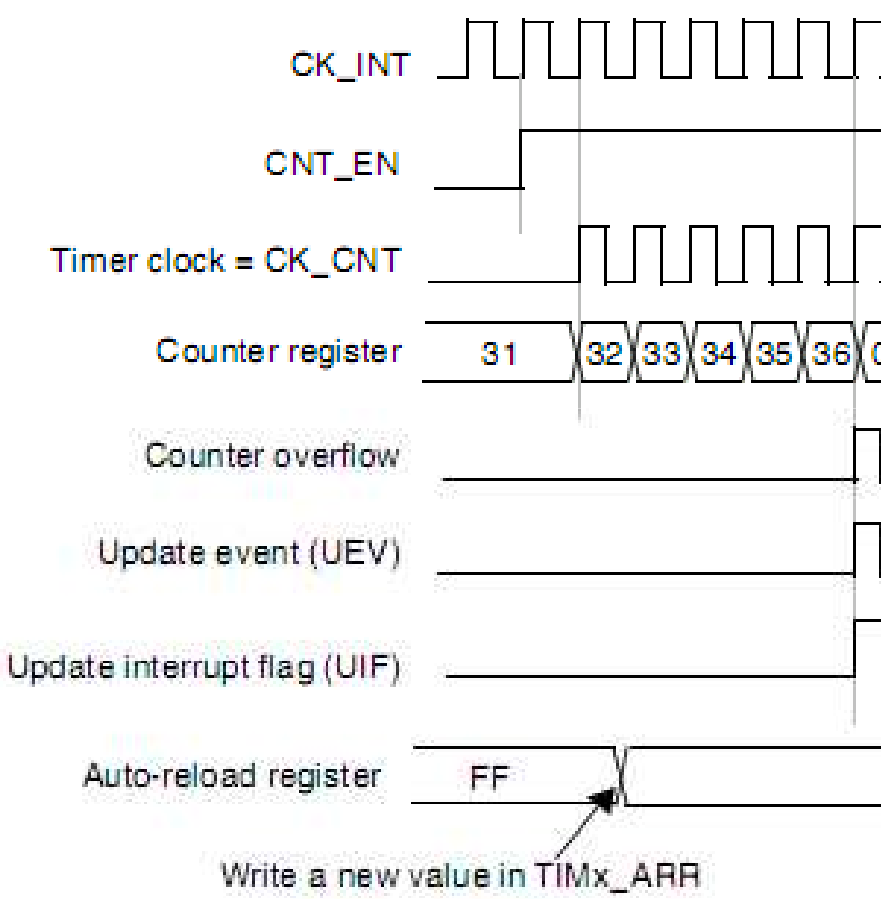
- Counter register (TIMx_CNT)
- Prescaler register (TIMx_PSC)
- Auto-reload register (TIMx_ARR)
- Repetition counter register (TIMx_RCR)

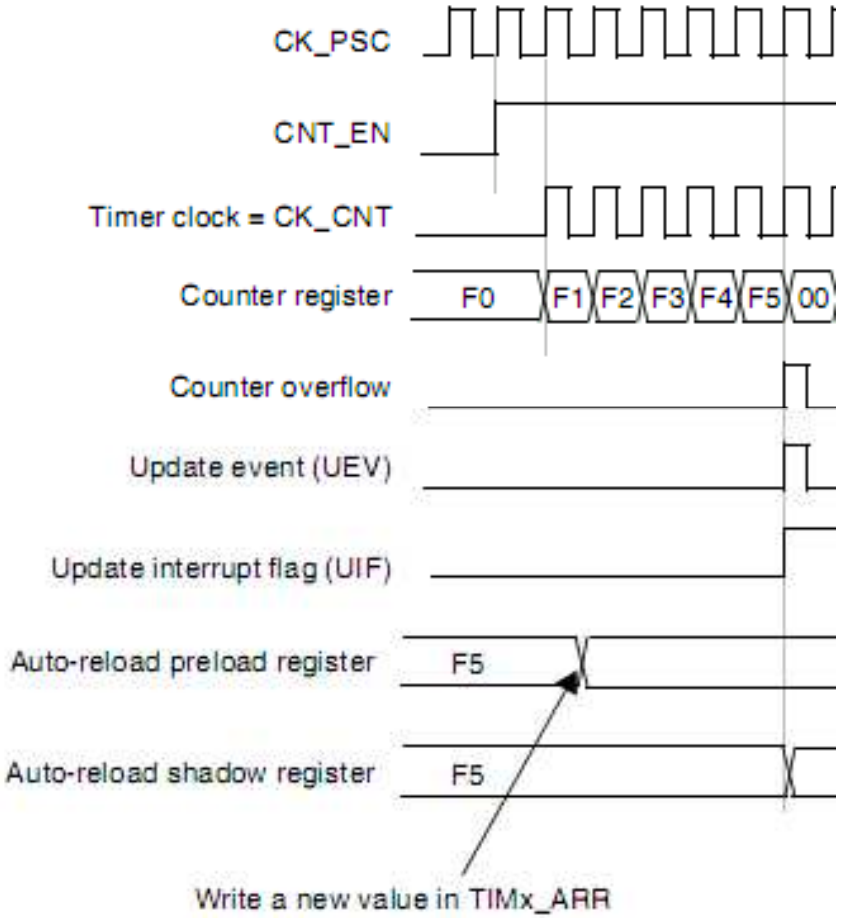
Prescaler register

Counter timing diagram with prescaler division change from 1 to 2



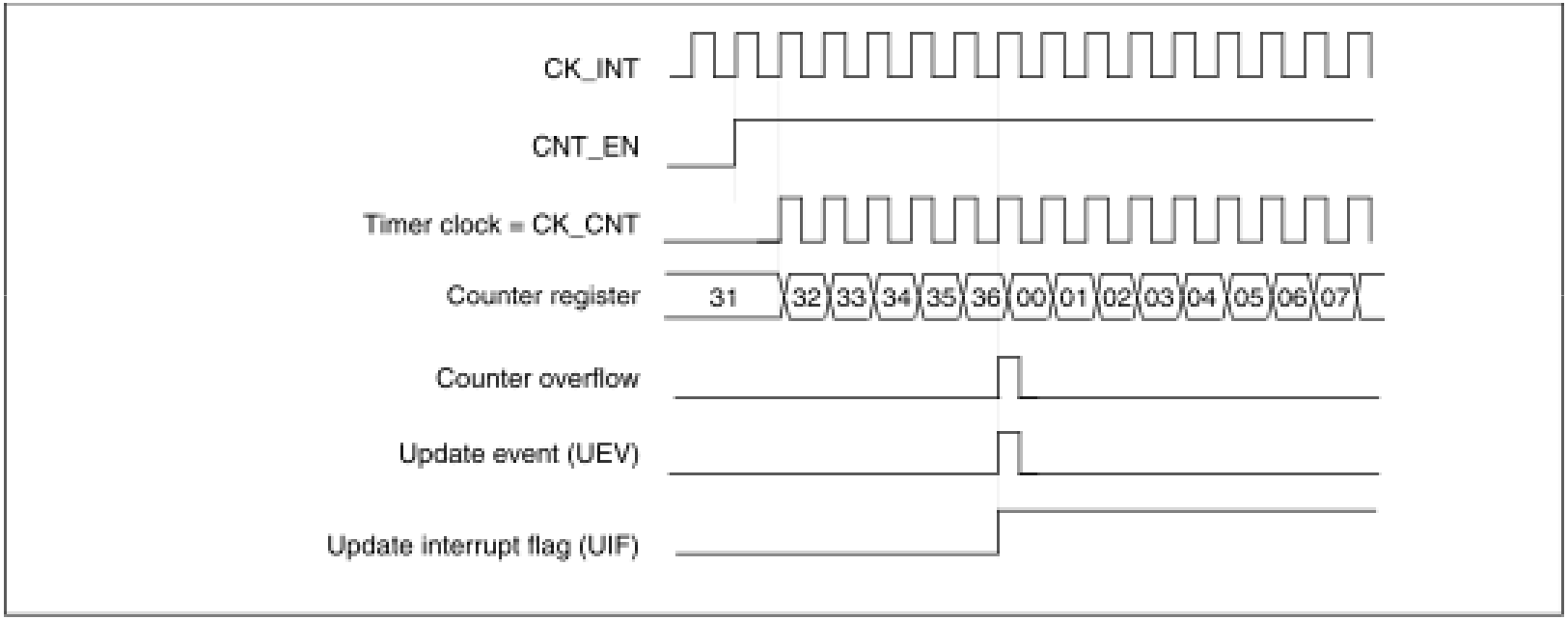
Auto-reload register buffer





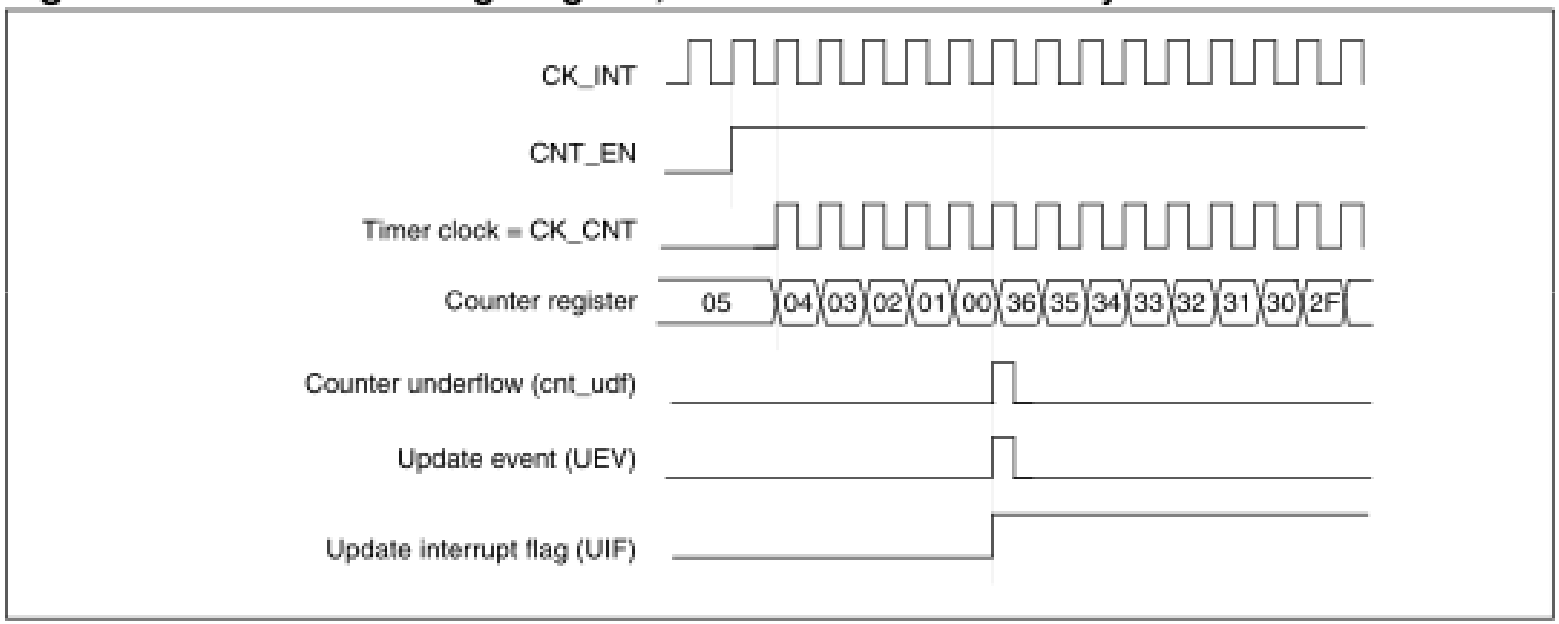
Counter Modes

Up counting modes

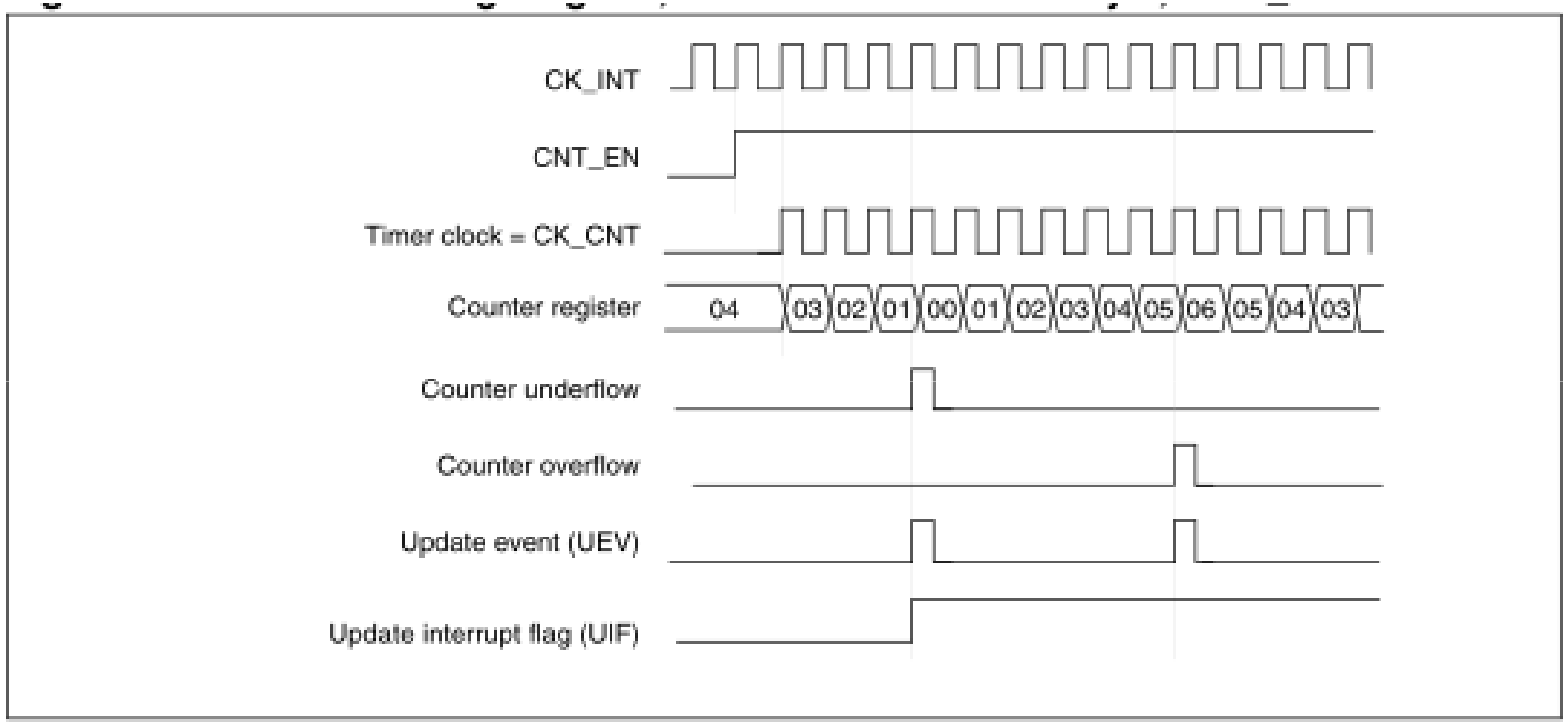


Down counting modes

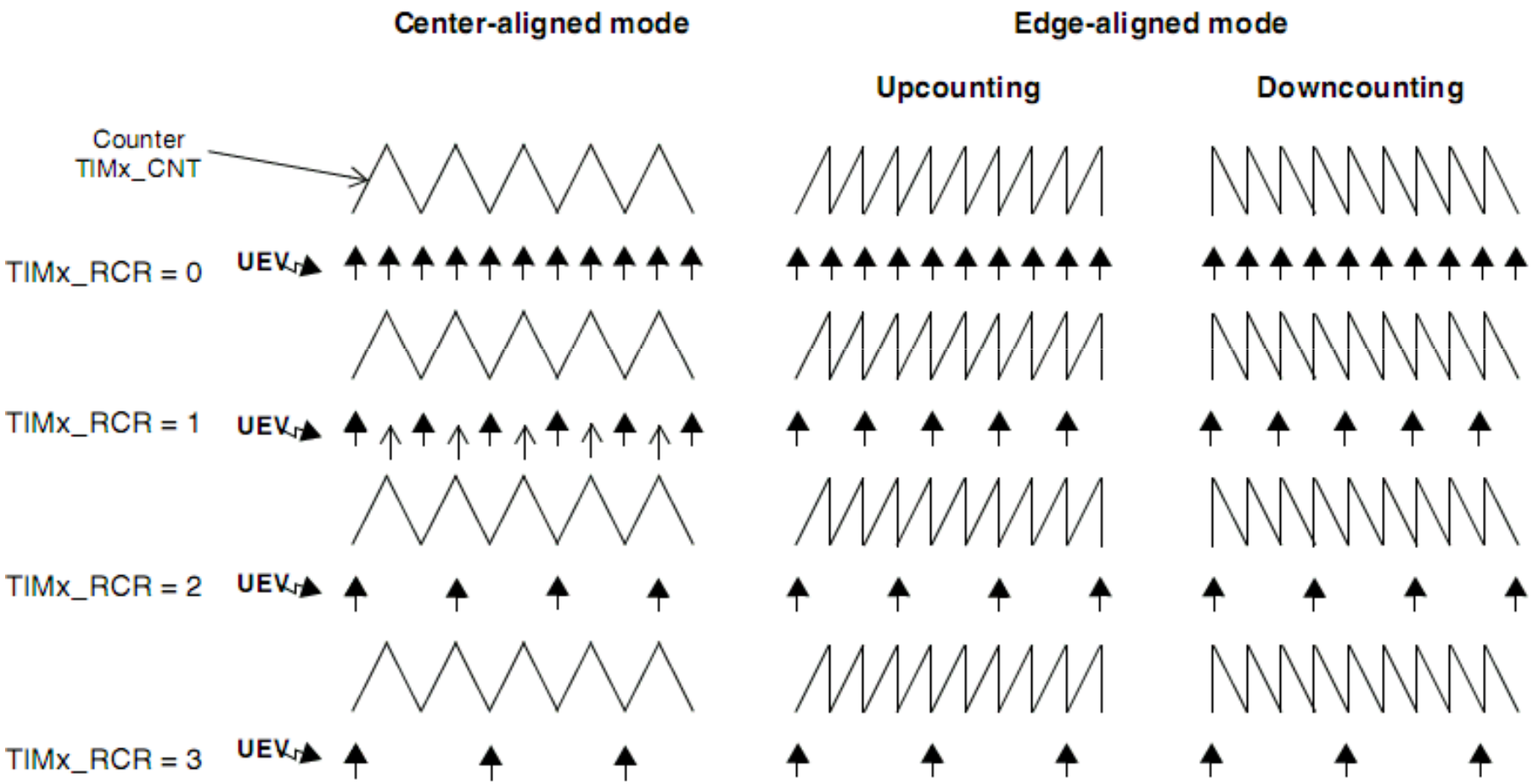
Figure 109. Counter timing diagram, internal clock divided by 1



Center-aligned modes (Up/ Down counting)



Counter modes

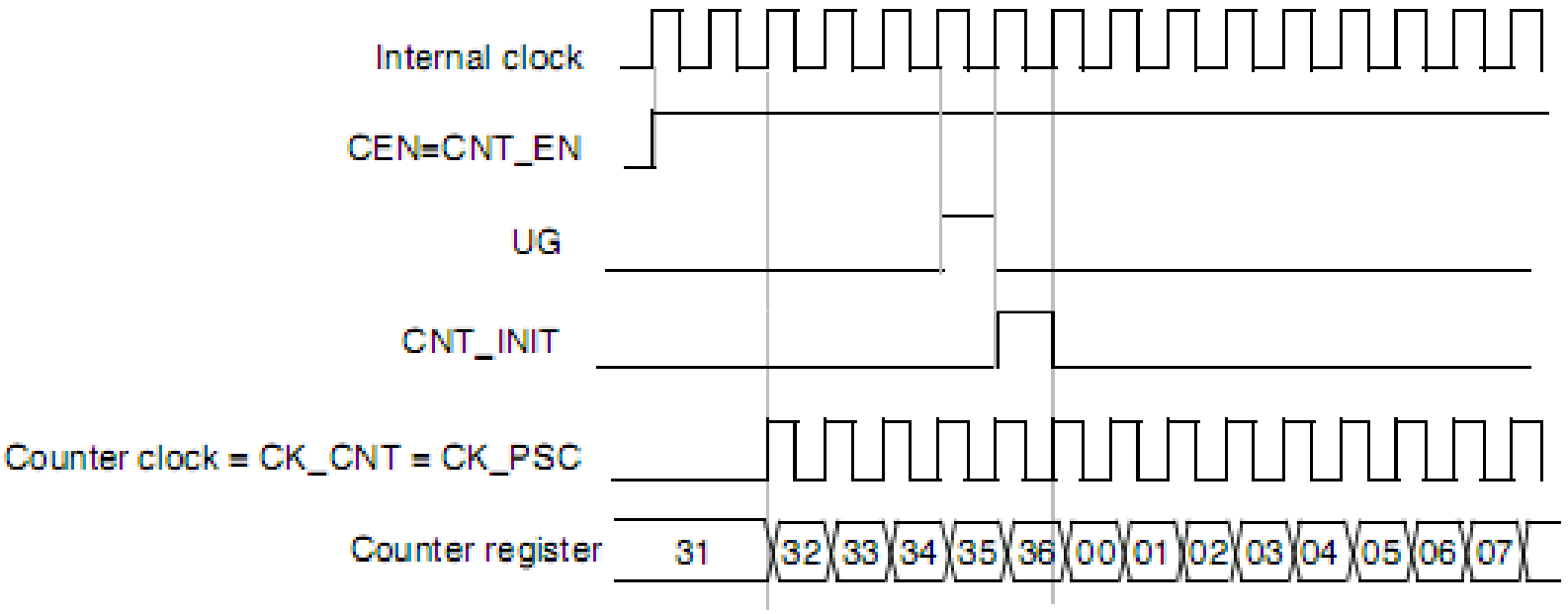


Clock selection

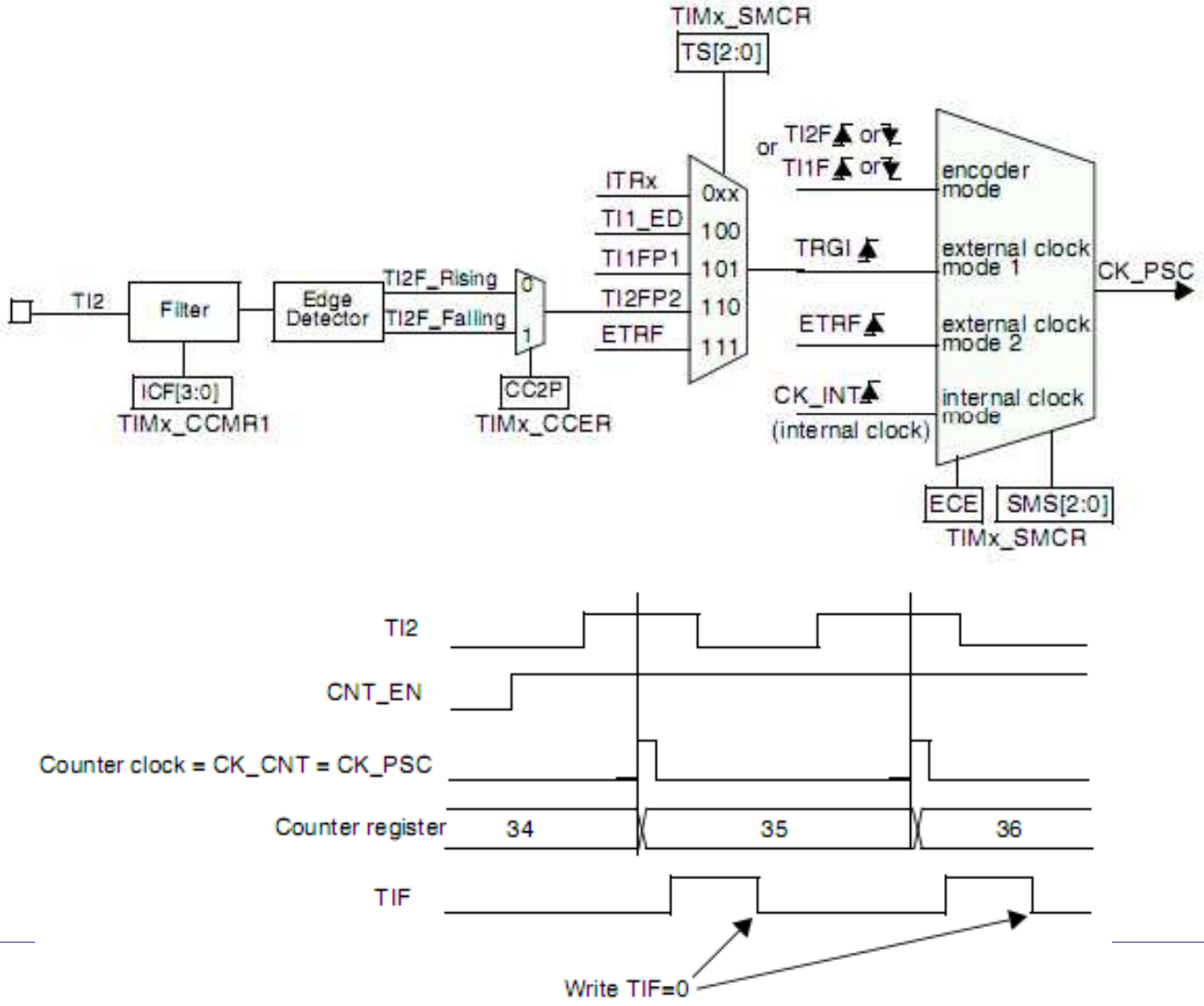
Clock selection

- Clock can be selected out of following sources
  - ✓ Internal clock (CK_INT)
  - ✓ External clock mode1:External input pin(TIx)
  - ✓ External clock mode2: external trigger input (ETR)
  - ✓ Internal trigger inputs (ITRx): using one timer as prescaler for another timer

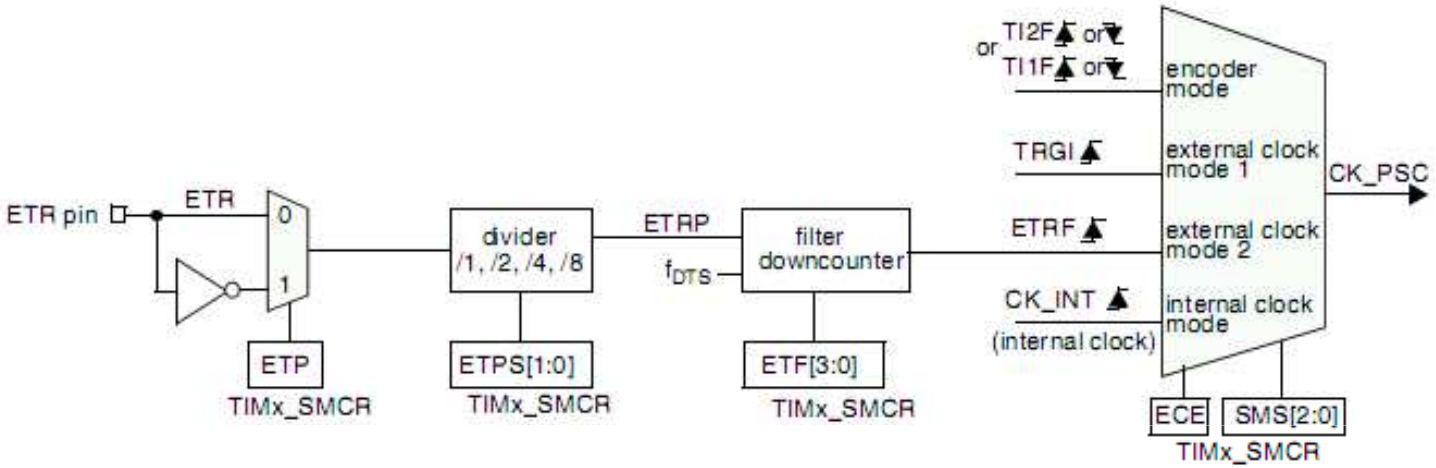
Internal clock source

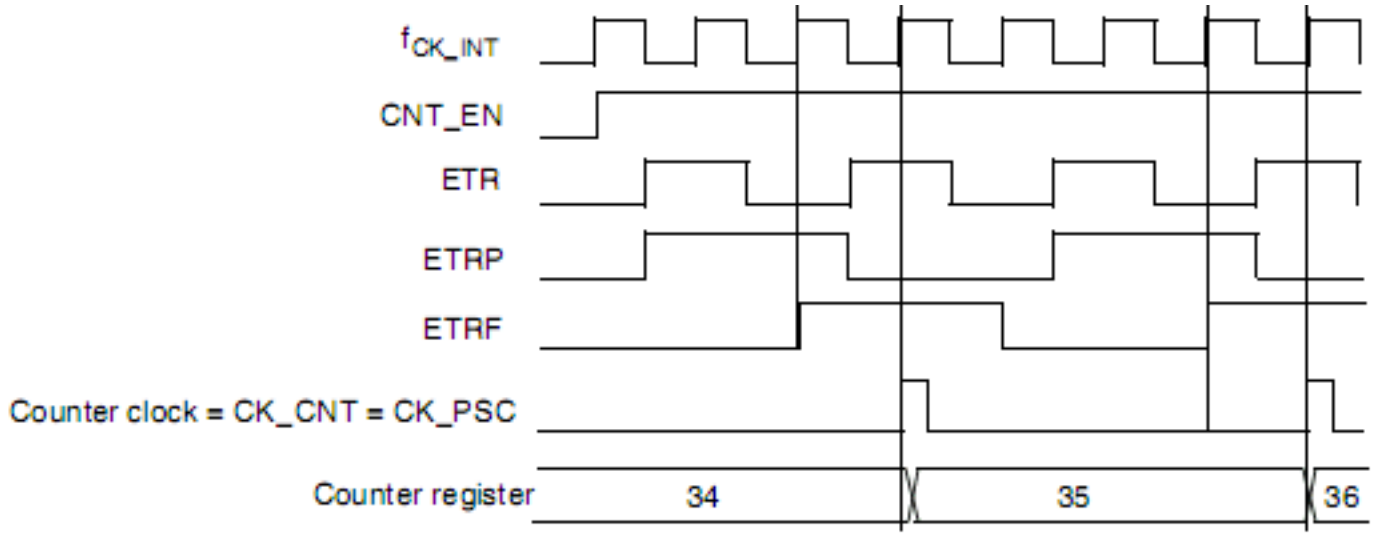


External clock source mode 1



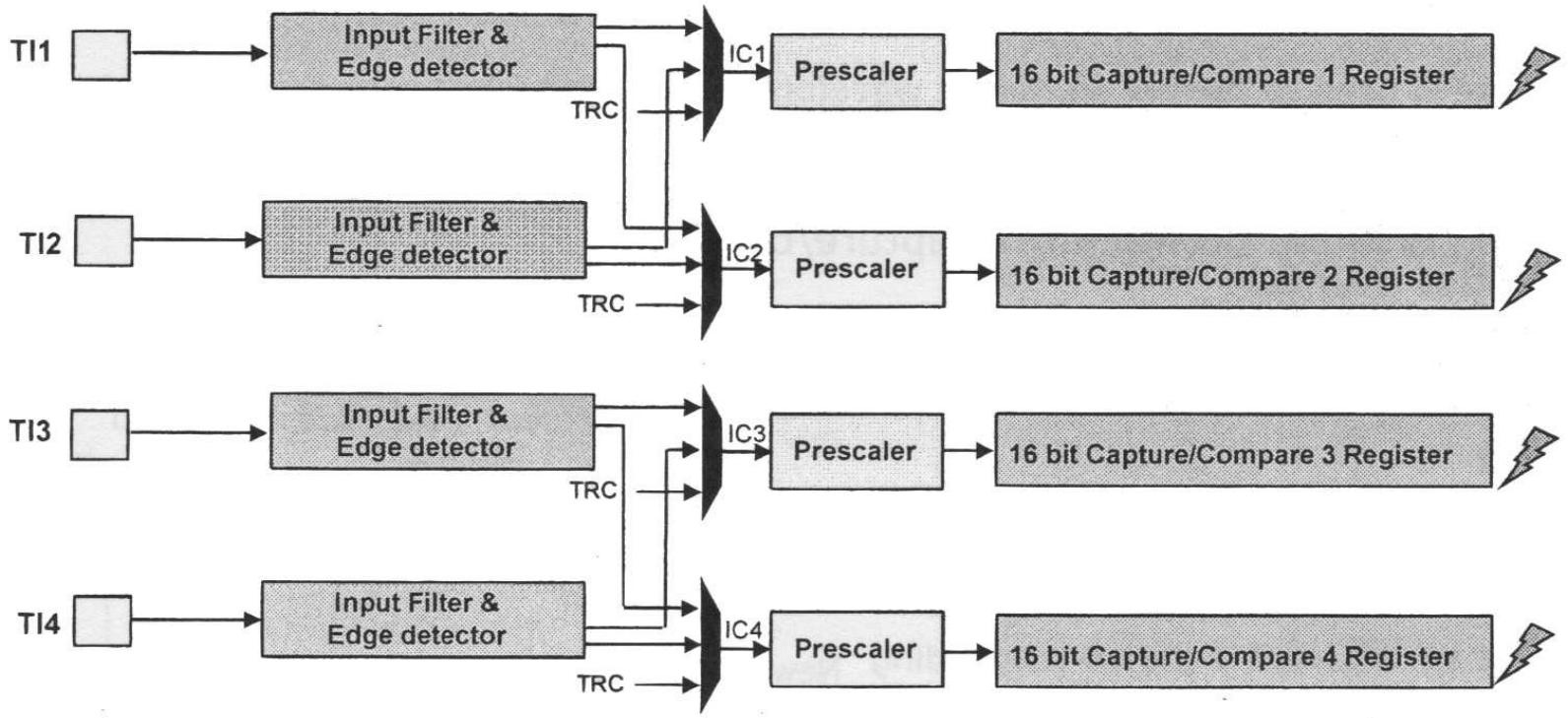
External clock source mode 2



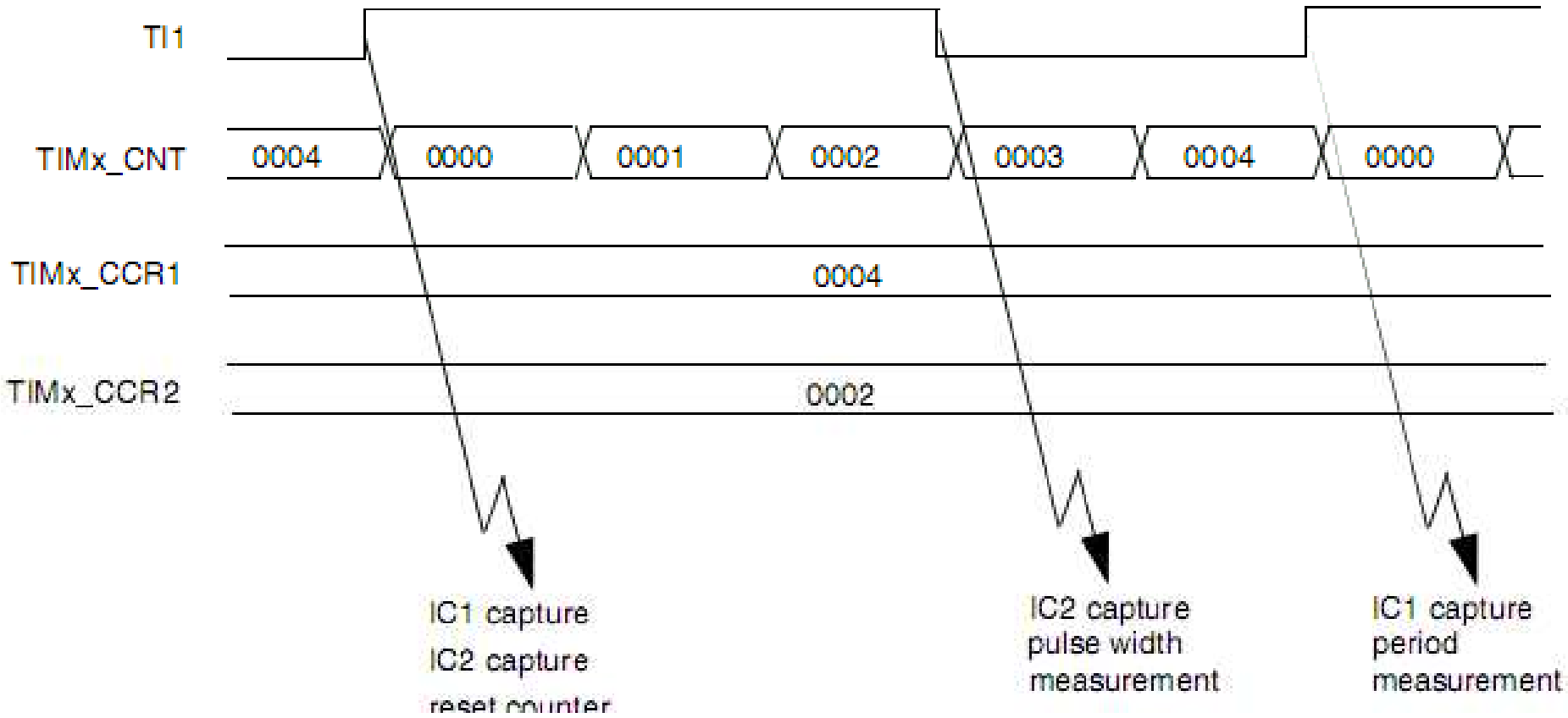


4 Independent Channels

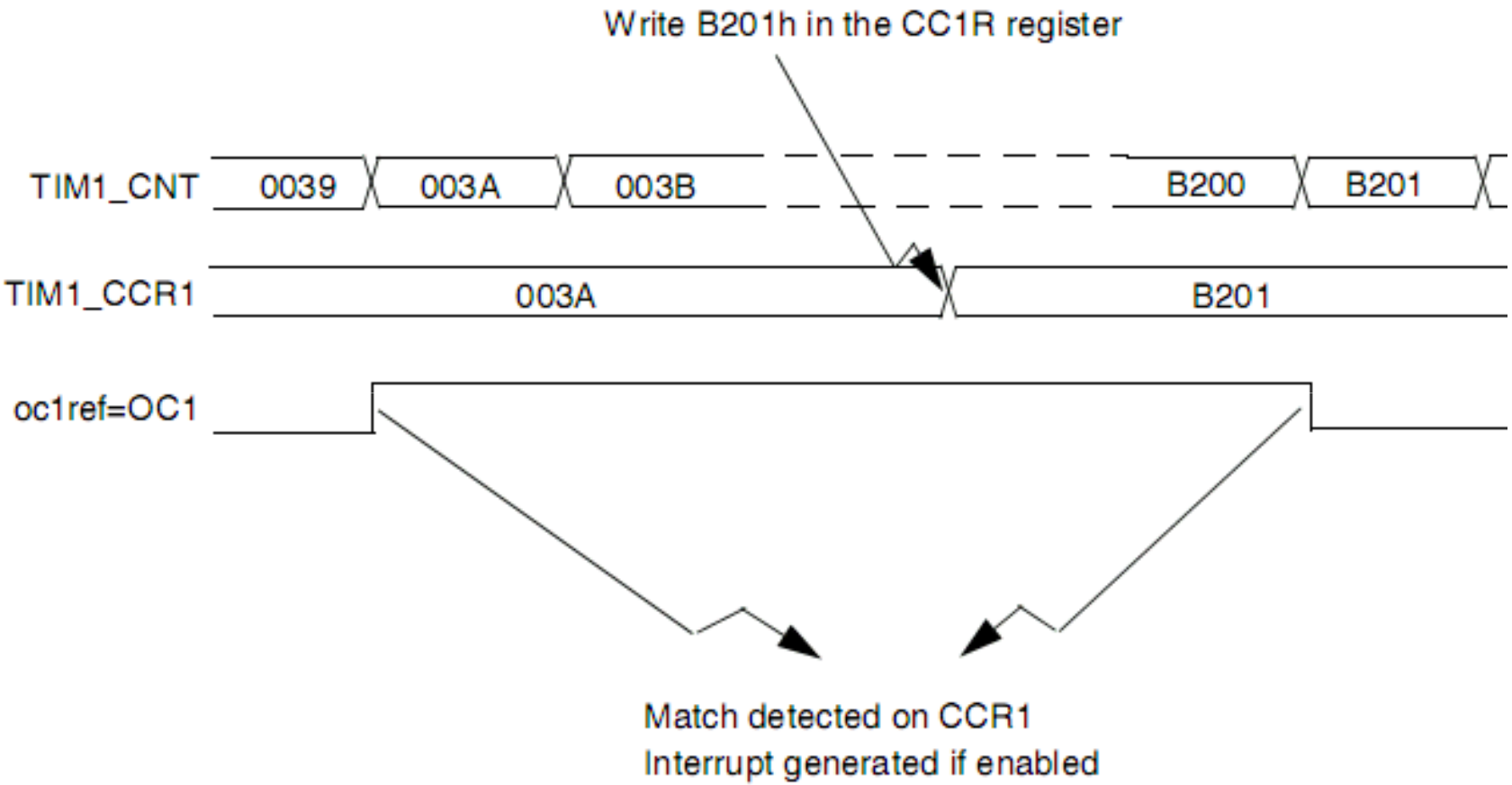
Input capture mode



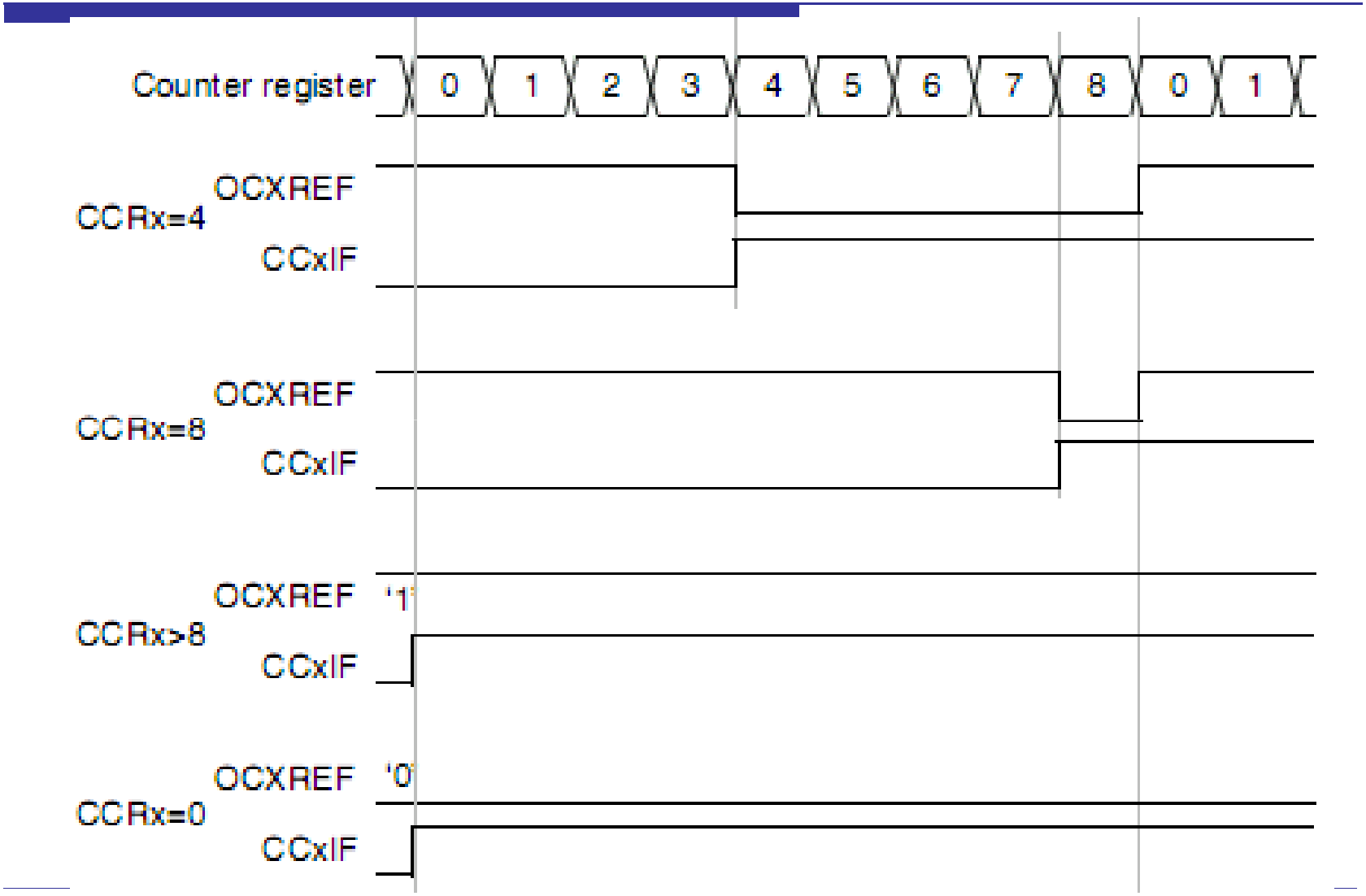
PWM input mode



Output compare mode

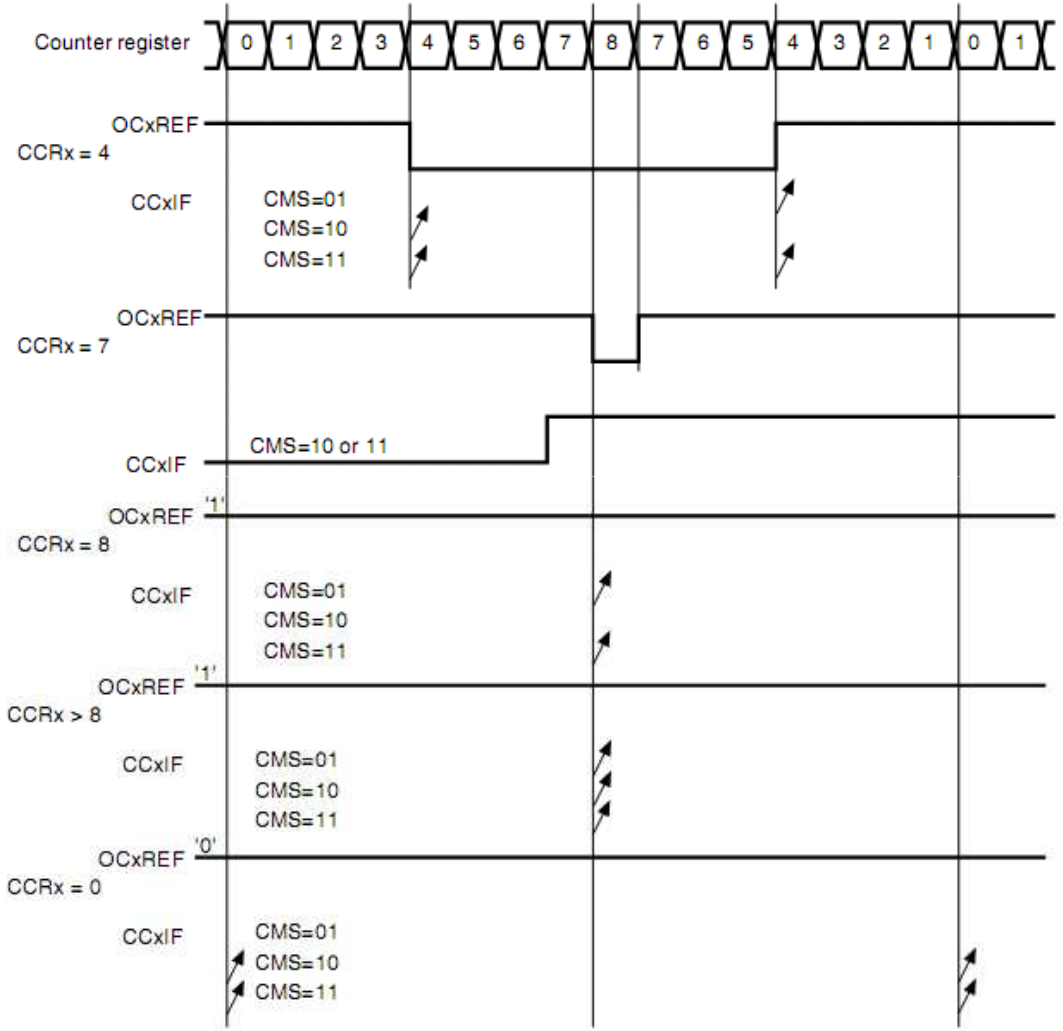


PWM edge-aligned mode



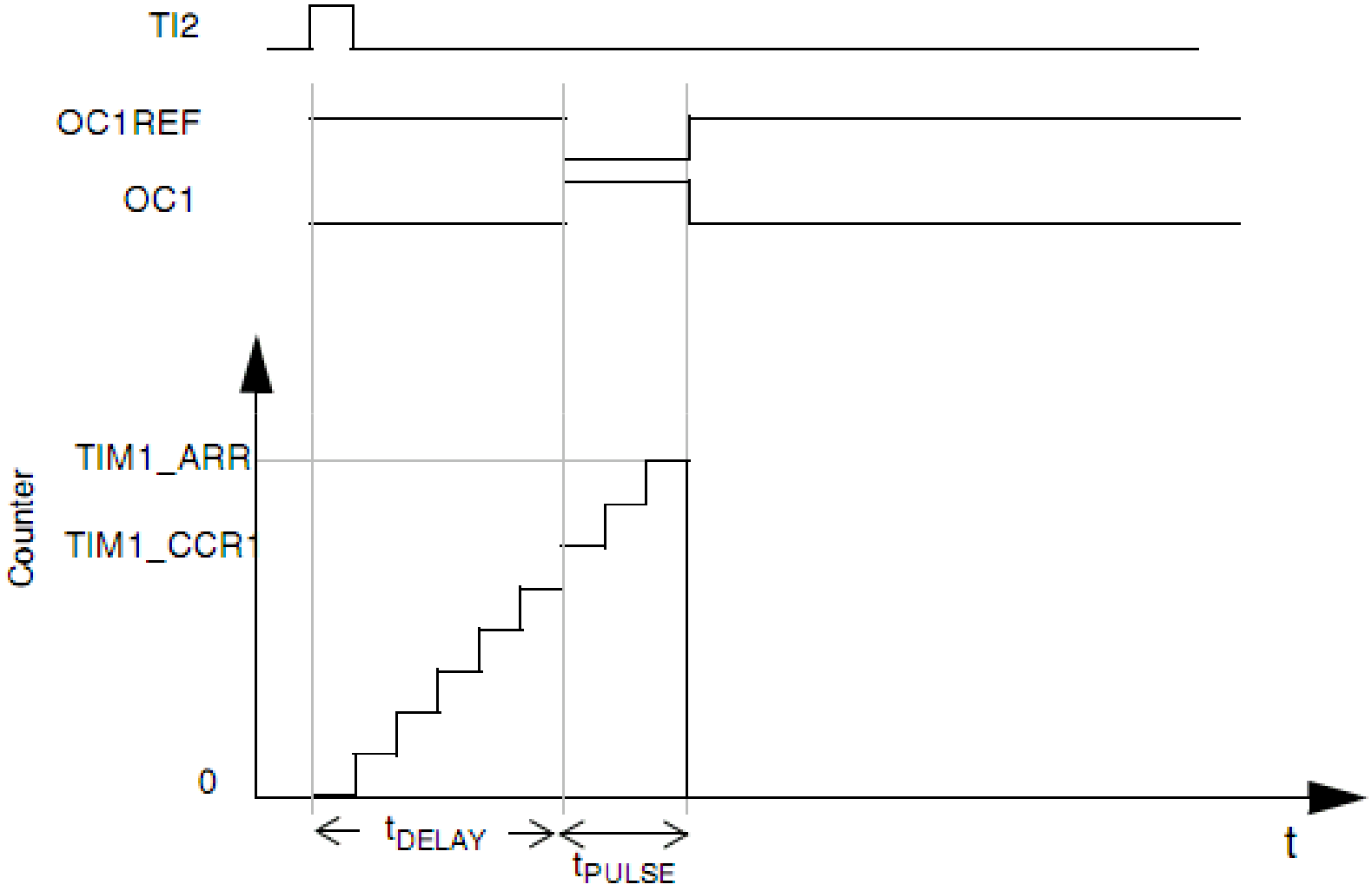
Edge-aligned PWM waveforms (ARR=8)

PWM center-aligned mode



Center-aligned PWM waveforms (ARR=8)

One-pulse mode

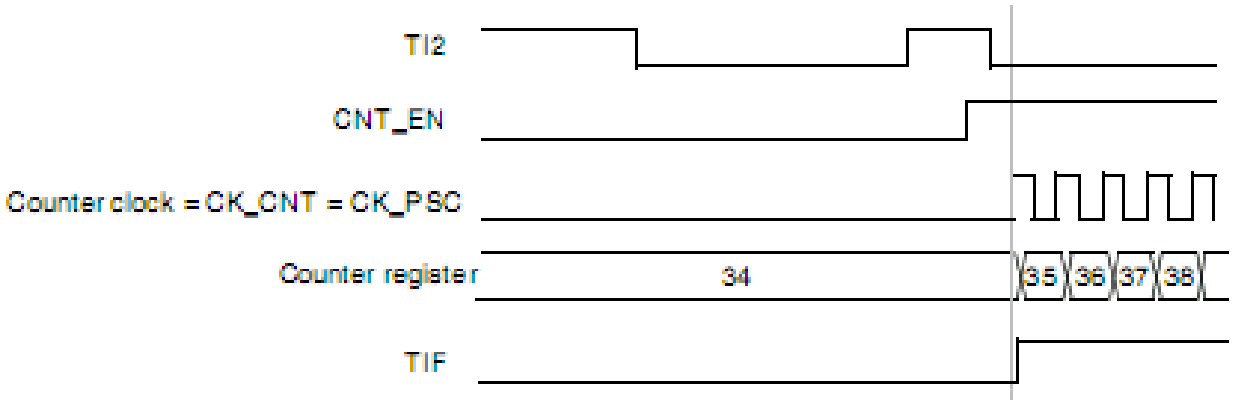


Synchronization

Timers and external trigger synchronization

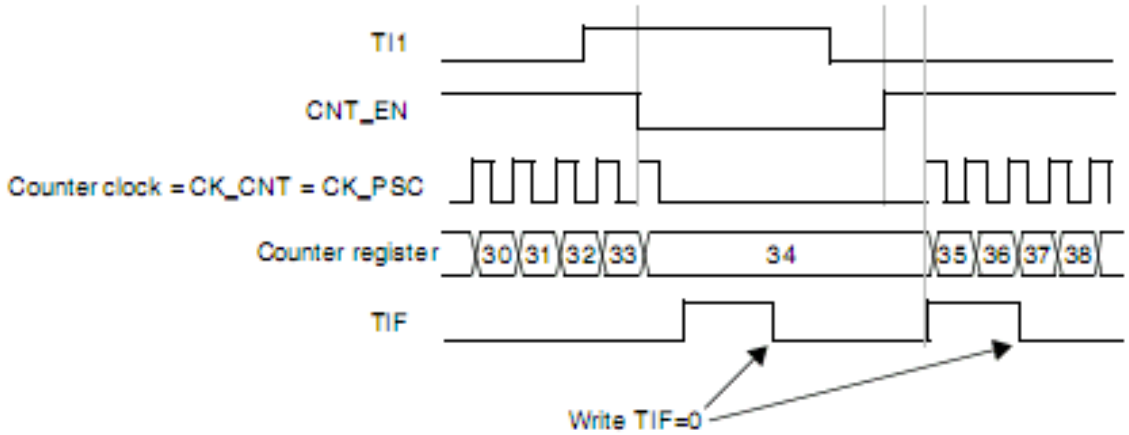
- Slave mode:

  Trigger mode



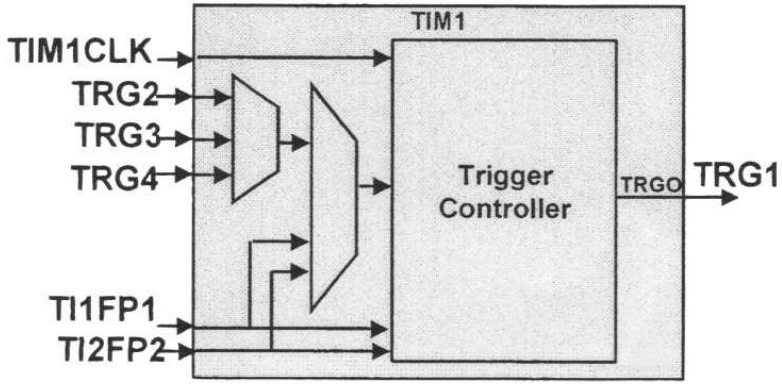
- Slave mode:

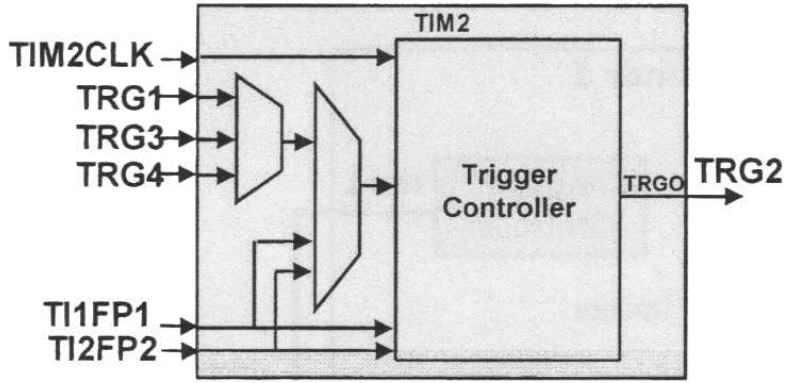
  Gated mode

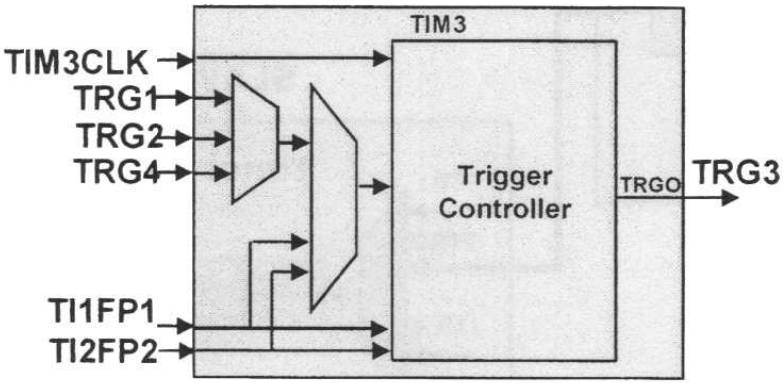


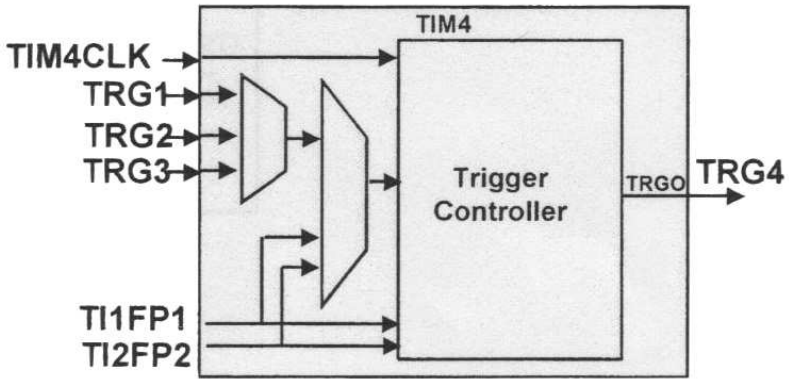
Timer synchronization

- The four Timers are link together for timers synchronization or chaining



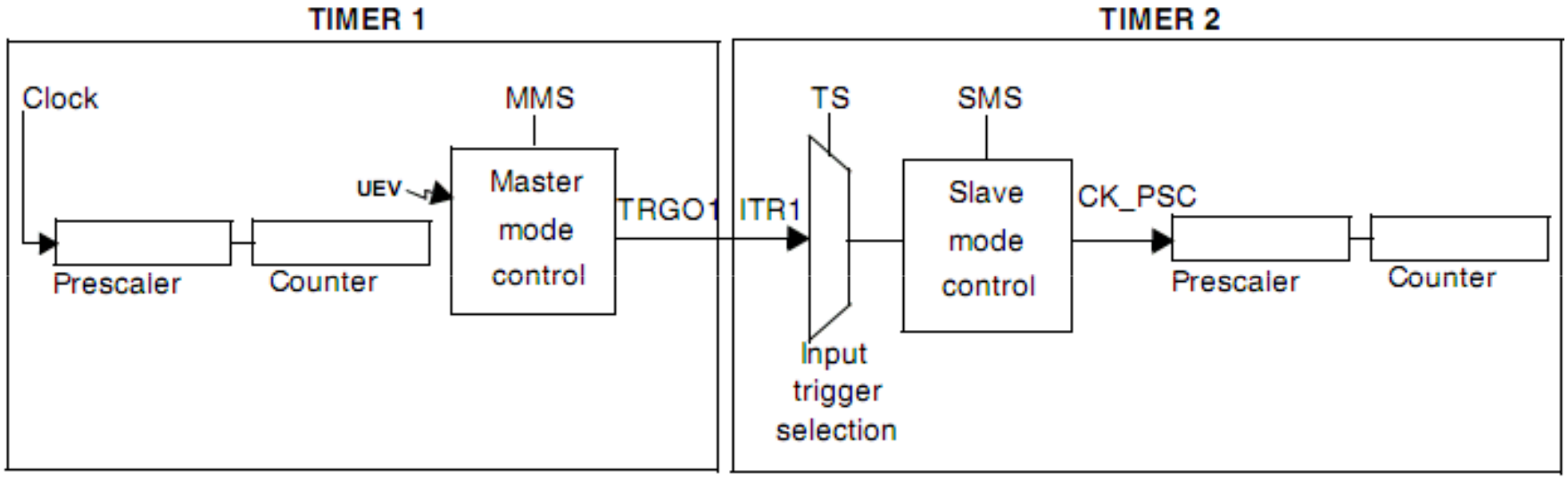






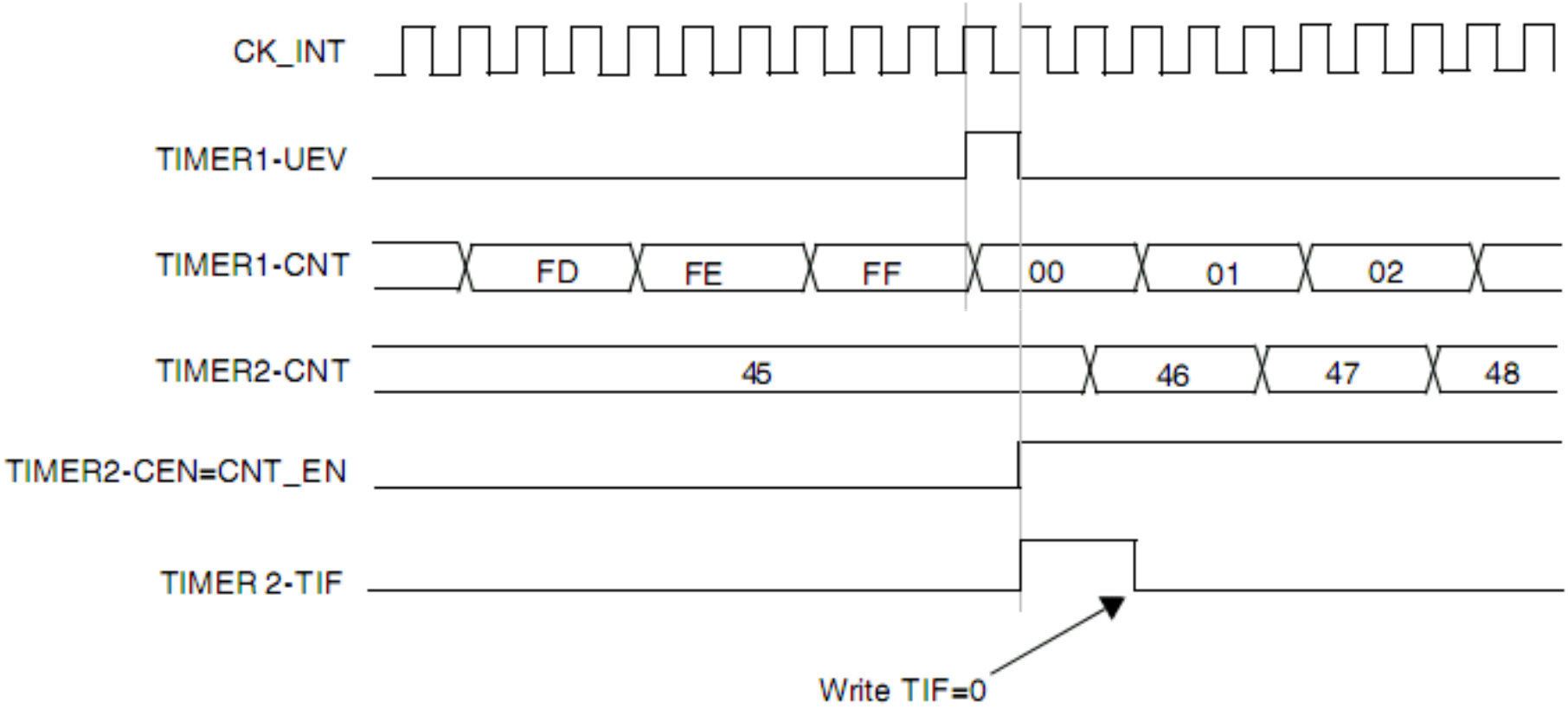
Timer synchronization(cont.)

- Using one timer as prescaler for the another



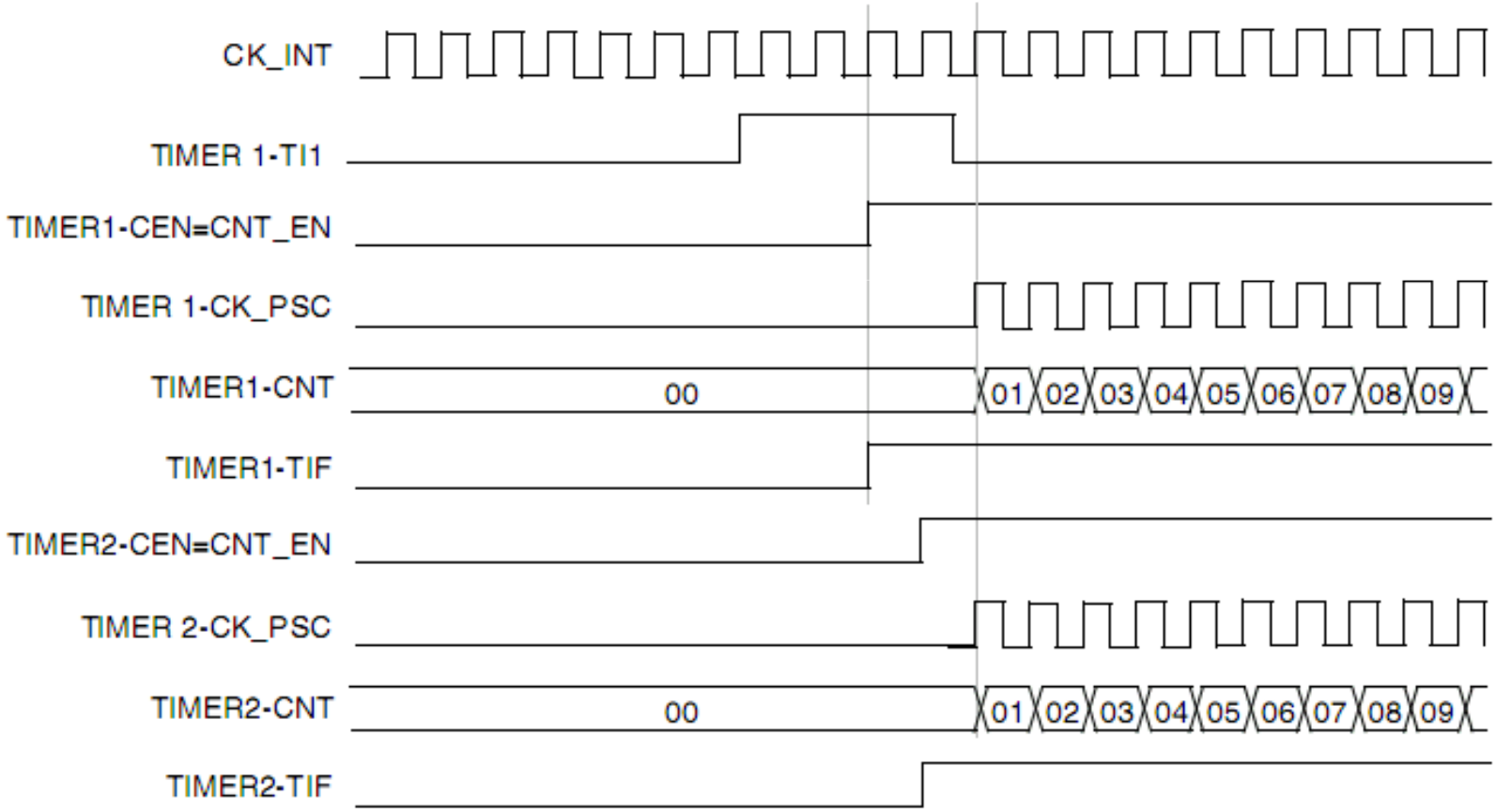
Timer synchronization(cont.)

- Using one timer to start another timer



Timer synchronization(cont.)

- Starting 2 timers synchronously in response to an external trigger



實驗

1、TIMER的控制

2、計時碼表設計

3、使用timer產生PWM訊號，控制LED燈亮度
(或蜂鳴器音量、或直流馬達)

1.Timer控制實驗

說明：

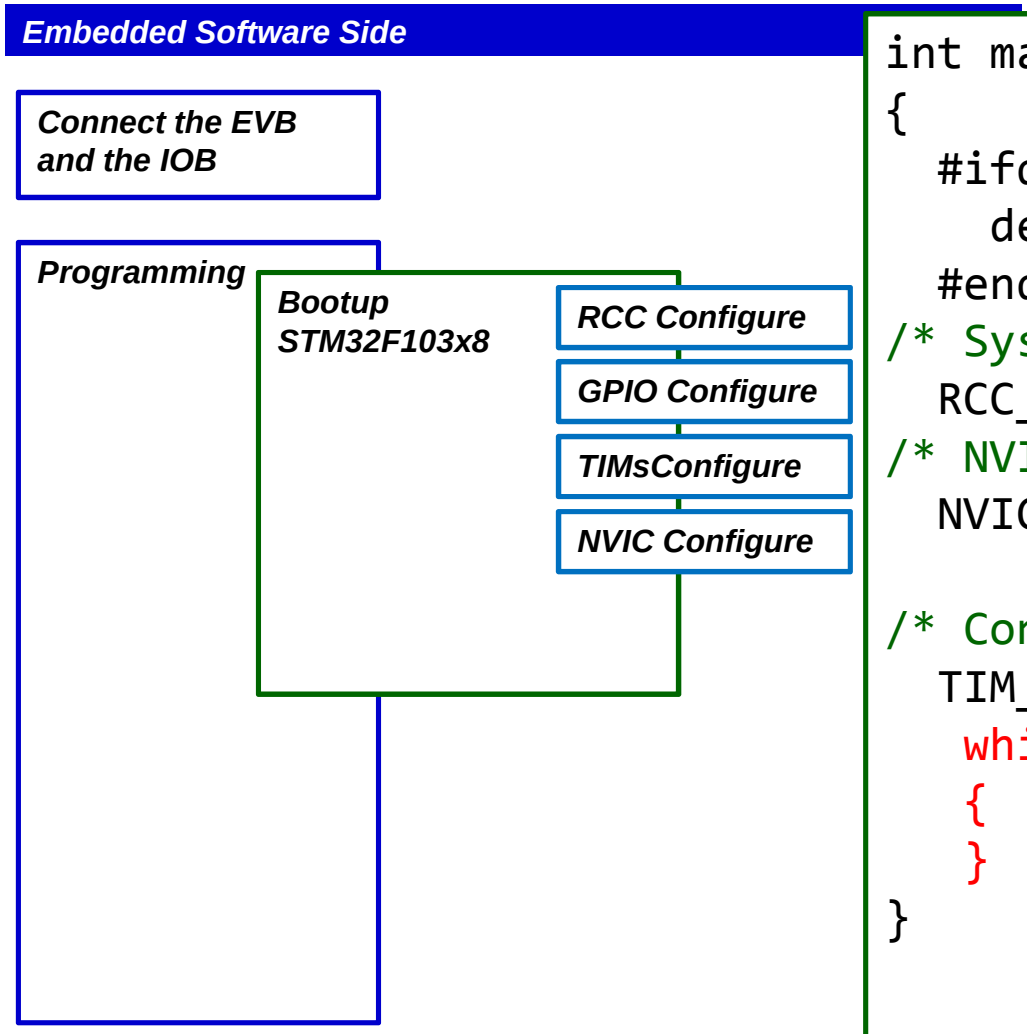
用3個timer(TIM2, TIM3, TIM4)分別產生1 sec, 2 sec及3 sec等3個不同時間訊號,輸出訊息至螢幕

Step1 修改程式碼

- 檔案目錄結構

	<..\Timer_Counter\E1>
<project>	單元實驗**Project**目錄
<source>	程式碼目錄
<include>	引入檔目錄
<library>	函式庫目錄
<image>	燒錄配置檔目錄
	<..\\Timer_Counter\E1\image>
Lab.dfu	燒錄配置檔
	<..\Timer_Counter\E1\source>
main.c	硬體配置程式
stm32f10x_it.c	中斷服務程式
hw_config.c	Enable clock

Development Flow




```
int main(void)
{
  #ifdef DEBUG
    debug();
  #endif
/* System clocks configuration -*/
  RCC_Configuration();
/* NVIC configuration   -----*/
  NVIC_Configuration();

/* Configure TIMs -----------*/
  TIM_Configuration();
   while(1)
   {
   }
}
```

Configure RCC

RCC FwLib Functions List

Function name	Description
RCC_DeInit	Resets the RCC clock configuration to the default reset state
RCC_HSEConfig	Configures the External High Speed oscillator (HSE).
RCC_WaitForHSEStartUp	Waits for HSE start-up.
RCC_HCLKConfig	Configures the AHB clock (HCLK).
RCC_PCLK1Config	Configures the Low Speed APB clock (PCLK1).
RCC_PCLK2Config	Configures the High Speed APB clock (PCLK2).
RCC_PLLConfig	Configures the PLL clock source and multiplication factor.
RCC_PLLCmd	Enables or disables the PLL.
RCC_SYSCLKConfig	Configures the system clock (SYSCLK).
RCC_APB2PeriphClockCmd	Enables or disables the High Speed APB (APB2) peripheral clock.

```
void Set_System(void)
{
 .
 .
 .
#ifndef USE_STM3210C_EVAL
  /* Enable USB_DISCONNECT GPIO clock */
  RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIO_DISCONNECT, ENABLE);

  /* Configure USB pull-up pin */
  GPIO_InitStructure.GPIO_Pin = USB_DISCONNECT_PIN;
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out_OD;
  GPIO_Init(USB_DISCONNECT, &GPIO_InitStructure);
#endif /* USE_STM3210C_EVAL */
  Set_USBClock();
  USB_Interrupts_Config();
  USB_Init();
  RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOA, ENABLE);
  /* Enable TIM2, TIM3 and TIM4 */
  RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM2 | RCC_APB1Periph_TIM3 |
                         RCC_APB1Periph_TIM4, ENABLE);
}
```

Configure TIMs

```
void TIM_Configuration(void)
{
  TIM_TimeBaseInitTypeDef  TIM_TimeBaseStructure;
  TIM_OCInitTypeDef  TIM_OCInitStructure;
  /* TIM2 configuration */
  TIM_TimeBaseStructure.TIM_Period = 0x4AF;
  TIM_TimeBaseStructure.TIM_Prescaler = 0xEA5F;
  TIM_TimeBaseStructure.TIM_ClockDivision = 0x0;
  TIM_TimeBaseStructure.TIM_CounterMode = TIM_CounterMode_Up;
  TIM_TimeBaseStructure.TIM_RepetitionCounter = 0x0000;
  TIM_TimeBaseInit(TIM2, &TIM_TimeBaseStructure);
  TIM_OCStructInit(&TIM_OCInitStructure);
  /* Output Compare Timing Mode configuration: Channel1 */
  TIM_OCInitStructure.TIM_OCMode = TIM_OCMode_Timing;
  TIM_OCInitStructure.TIM_Pulse = 0x0;
  TIM_OC1Init(TIM2, &TIM_OCInitStructure);
  /* TIM3 configuration */
  ..........
  /* TIM2 enable counter */
  TIM_Cmd(TIM2, ENABLE);
   ..........
  /* Immediate load of TIM2 Precaler value */
  TIM_PrescalerConfig(TIM2, 0xEA5F, TIM_PSCReloadMode_Immediate);
   ..........
  /* Clear TIM2 update pending flag */
  TIM_ClearFlag(TIM2, TIM_FLAG_Update);
   ..........
  /* Enable TIM2 Update interrupt */
  TIM_ITConfig(TIM2, TIM_IT_Update, ENABLE);
}
```

TIM4設成0xE0F
(約3sec)

Configure NVIC

```
/* Configure one bit for preemption priority */
NVIC_PriorityGroupConfig(NVIC_PriorityGroup_2);

/* Enable the TIM2 Interrupt */
NVIC_InitStructure.NVIC_IRQChannel = TIM2_IRQChannel;
NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 0;
NVIC_InitStructure.NVIC_IRQChannelSubPriority = 0;
NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
NVIC_Init(&NVIC_InitStructure);

/* Enable the TIM3 Interrupt */
/* Enable the TIM4 Interrupt */
```

IRQ Service

```
void TIM2_IRQHandler(void)
{
  /* Clear TIM2 update interrupt */
  TIM_ClearITPendingBit(TIM2, TIM_IT_Update);

  printf("TIMER2\r\n");
}
```

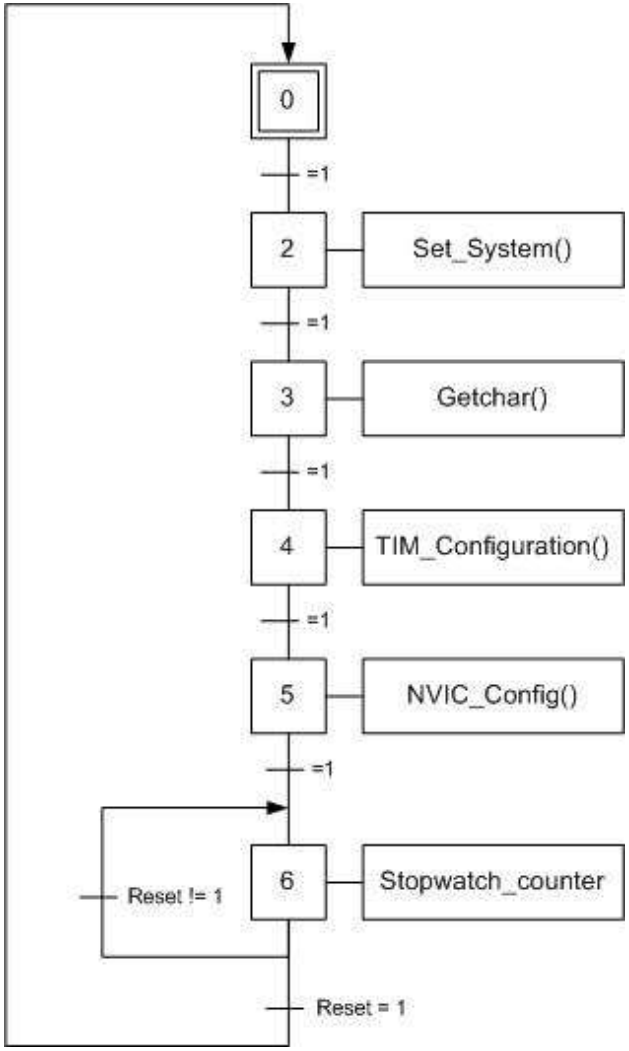
Step 2 編譯燒錄程式並觀察結果

- 編譯
- 將編譯後的hex檔轉換為dfu
- 透過USB 燒錄dfu檔
- Timer2, Timer3及Timer4是否Delay 1s, 2s, 3s後印出其TIMER字樣

2.計時馬錶實驗

- 說明：
  - 使用者輸入”1”馬錶開始計時並印出
  - 使用者輸入”2”馬錶暫停計時並印出
  - 使用者輸入”3”馬錶繼續計時並印出
  - 使用者輸入”4”馬錶停止計時並印出
- 請修改
  - main()加入使用者輸入指令
  - TIM2_IRQHandler();
    - 呼叫Stopwatch_counter();
    - 清除TIM2的更新中斷
  - Stopwatch_counter();作指令上的counter判斷

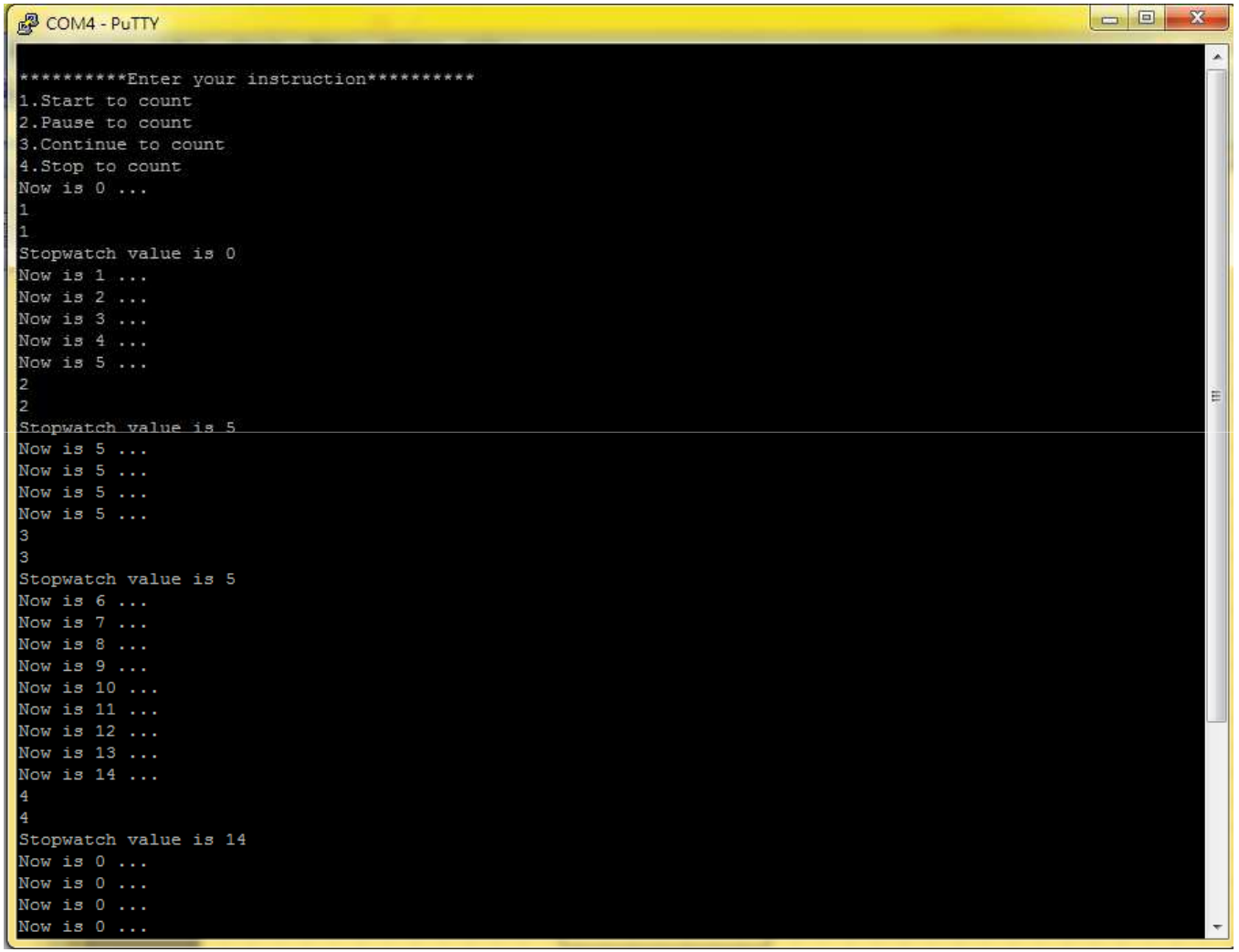
Grafcet


0
=1
2
Set_System()
=1
3
Getchar()
=1
4
TIM_Configuration()
=1
5
NVIC_Config()
=1
6
Stopwatch_counter
Reset != 1
Reset = 1

Stopwatch_counter();

```
if( State_flag == 1 )
{
      //add your code
}
else if(State_flag == 2)
{
       //add your code
}
else if(State_flag == 3)
{
       //add your code
}
else if(State_flag == 4)
{
       //add your code
}
printf("Now is %d ...\r\n", Stopwatch);
```

DEMO


COM4 - PuTTY
**********Enter your instruction**********
1.Start to count
2.Pause to count
3.Continue to count
4.Stop to count
Now is 0 ...
1
1
Stopwatch value is 0
Now is 1 ...
Now is 2 ...
Now is 3 ...
Now is 4 ...
Now is 5 ...
2
2
Stopwatch value is 5
Now is 5 ...
Now is 5 ...
Now is 5 ...
Now is 5 ...
3
3
Stopwatch value is 5
Now is 6 ...
Now is 7 ...
Now is 8 ...
Now is 9 ...
Now is 10 ...
Now is 11 ...
Now is 12 ...
Now is 13 ...
Now is 14 ...
4
4
Stopwatch value is 14
Now is 0 ...
Now is 0 ...
Now is 0 ...
Now is 0 ...

3.TIMER產生PWM訊號

- 說明：

  使用timer產生PWM訊號，控制LED燈亮度、或蜂鳴器音量、或直流馬達

- 請修改
  - 主程式呼叫PWM_output();並修改其輸入duty cycle參數大小控制LED亮度
  - PWM_output();
    設定頻率給TIM_TimeBaseStructure.TIM_Prescaler
    設定Duty Cycle給TIM_OCInitStructure.TIM_Pulse
- Ex:

  PWM_output(1~999, 1000)

  Duty Cycle 值越高越亮?還是越暗?

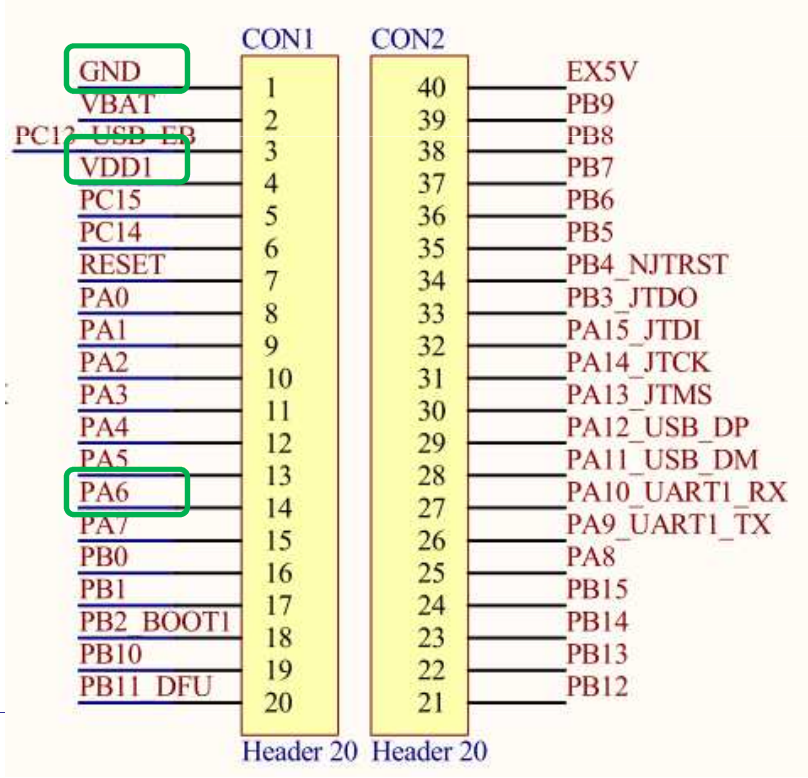
void PWM_output(u16 DutyCycle,u32 Frequency)

```
{
//TIM3->PSC = (36000/Frequency)-1;
// add your code
TIM_TimeBaseInit(TIM3, &TIM_TimeBaseStructure);

//TIM3->CCR1= DutyCycle;
// add your code
TIM_OC1Init(TIM3, &TIM_OCInitStructure);
}
```

硬體電路配置

子板腳位名稱	子板腳位編號	SIOC腳位名稱	SIOC腳位編號
VCC3.3V	CON2.29	VDD1	CON1.4
GND	CON2.30	GND	CON1.1
LEDR1	CON2.27	PA6	CON1.14



DEMO

- 播放DEMO影片

參考資料

- 參考資料

  [1] STM32F10xxx reference manual_2011.pdf

  [2] STM32F103x8.pdf

Q & A