

---

# Microcontroller Timing Functions

---

Reference:

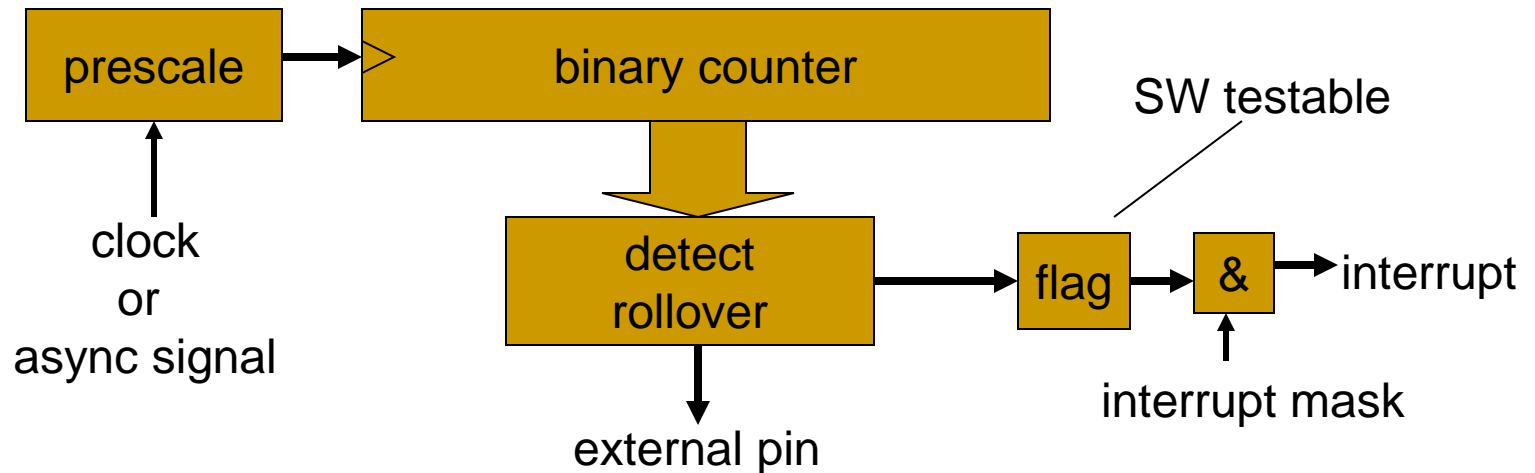
LPC 2292 User Manual

# LPC 2000 Timing Functions

- General purpose timer/counters (2 to 4)
  - 32-bit counter + 32-bit prescaler
  - 4 capture channels
  - 4 match registers
- Watchdog
  - Reset CPU if no SW reset within designated time
  - Recover from program crashes
- Real Time Clock (RTC)
  - Clock calendar
  - Trigger events at designated date/time
- PWM modulator
  - Generate up to 6 PWM signals
  - Structure similar to general purpose timer

# Timers and counters

- Very similar in most  $\mu$ Cs:
  - a **timer** is incremented by a clock/periodic signal;
  - a **counter** is incremented by an asynchronous external signal.
- Rollover causes interrupt or sets a testable “flag”
- Programmable: count, prescale, operating mode



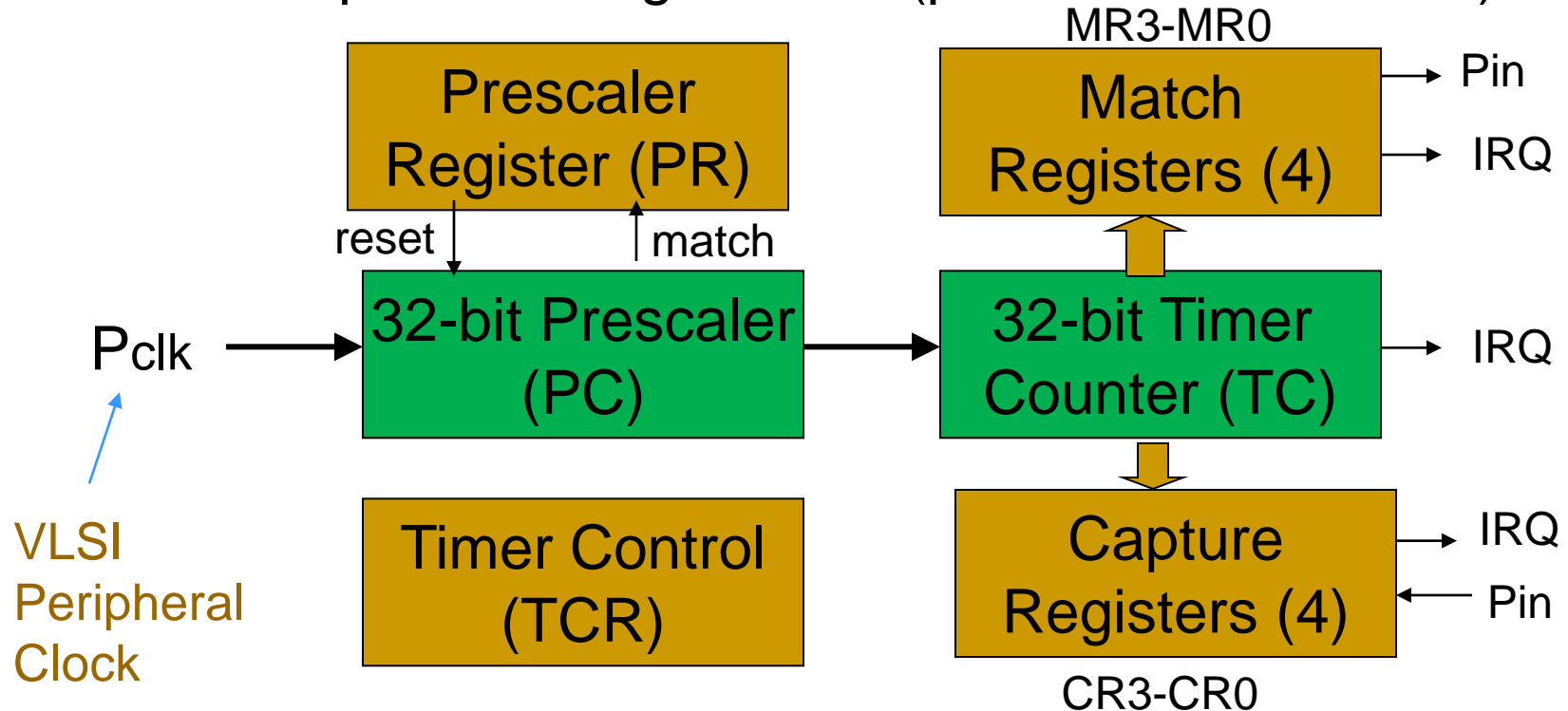
---

# Timer uses

- Interval timer for measuring internal events
  - Pulse width demodulator for external signals via “capture” events
  - Signal generator via “match” events
  - Free running timer
  - External event/clock counter
-

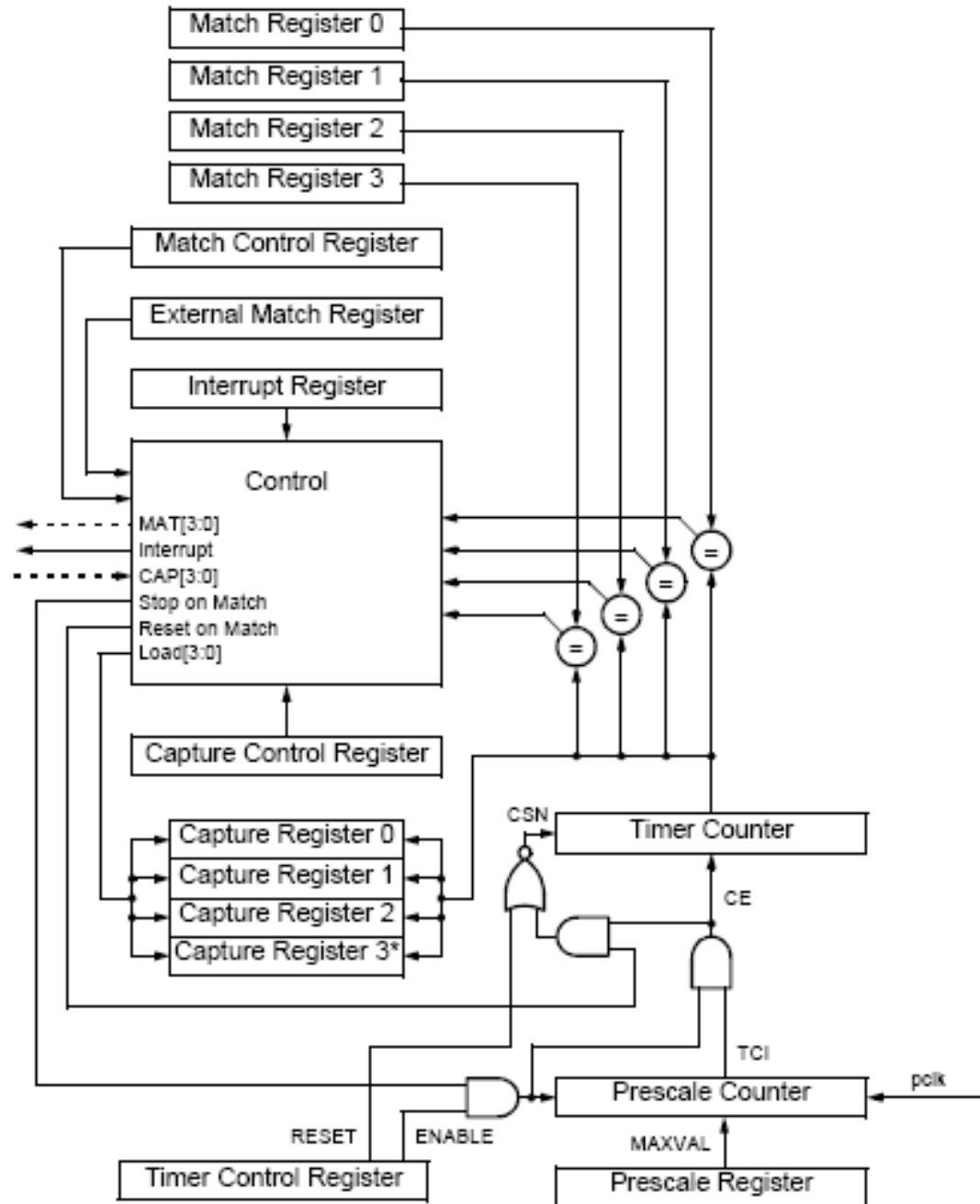
# General purpose timer

Counter TC increments when prescale counter PC matches prescale register PR (prescaler resets to 0)



- Prescale counter increments on Pclk
- Timer counter increments on Prescaler rollover

# LPC 2000 Timer



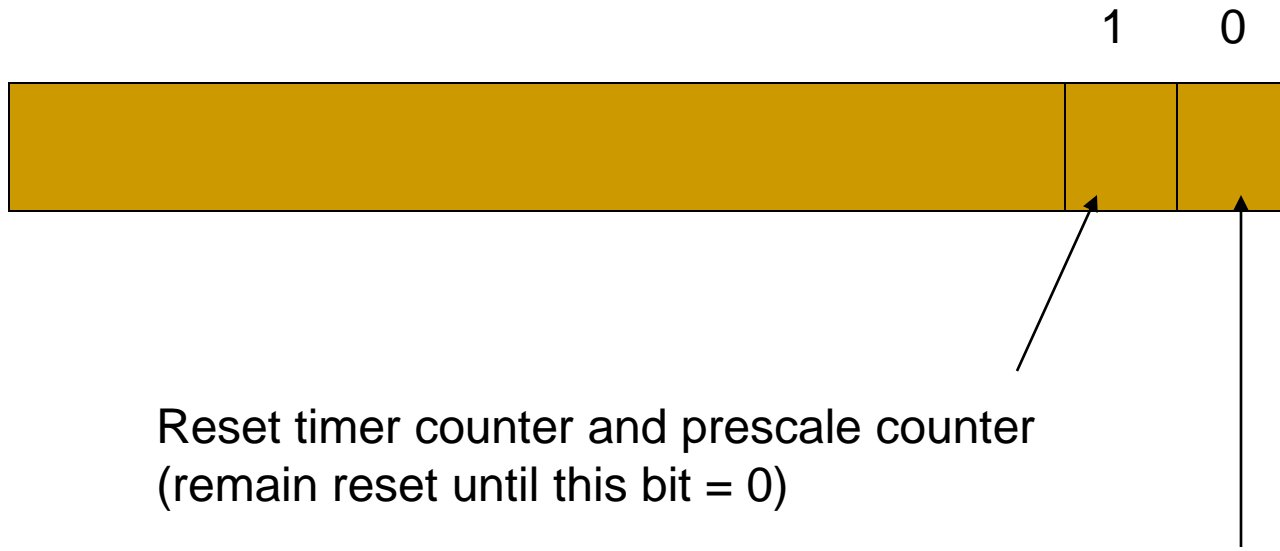
---

# Main Timer/Counter Registers

- Timer Count (TC) : T0TC, T1TC
    - 32-bit count
  - Prescale Counter (PC) : T0PC, T1PC
    - 32-bit prescale counter, increments on Pclk
    - Resets and increment TC when PC = PR
  - Prescale Register (PR): T0PR, T1PR
    - Maximum 32-bit prescale count
  - Timer Control Reg. (TCR): T0TCR, T1TCR
    - Bit0=TC/PC enable, Bit1=TC/PC reset
-

# Timer Control Register

Timer0: T0TCR, Timer1: T1TCR



Reset timer counter and prescale counter  
(remain reset until this bit = 0)

Enable timer counter and prescale counter  
1 = enable, 0 = disable

Usually do 2 writes to TxTCR: (1) reset, (2) enable

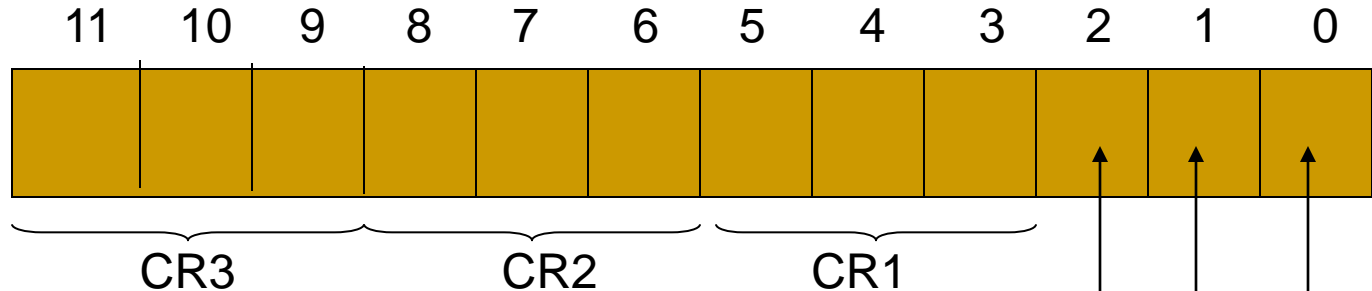
---

# Capture channels

- An input pin is designated as a “trigger”
    - Pins: CAP0.0, CAP0.1, CAP0.2, CAP0.3  
CAP1.0, CAP1.1, CAP1.2, CAP1.3
  - When a signal transition occurs on a pin:
    1. Take “snapshot” of timer (capture the time of the input event in a “capture register”)
      - Copy time from TC register
      - Capture registers: CR0, CR1, CR2, CR3
    2. Set flag; signal IRQ interrupt if “enabled”
-

# Capture Control Register (CCR)

Timer0: T0CCR, Timer1: T1CCR



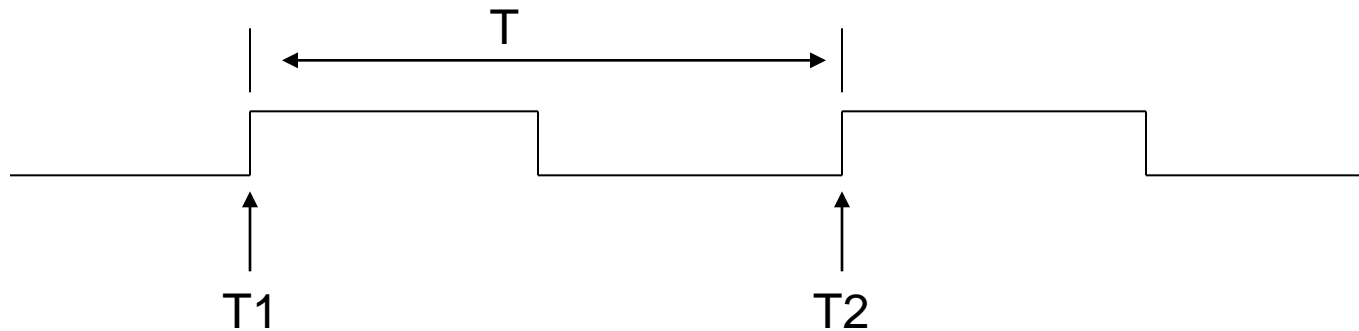
Interrupt on CAPn.0 "event" - CAP0I

Load CR0 with TC on CAPn.0 falling edge - CAP0FE

Load CR0 with TC on CAPn.0 rising edge - CAP0RE

# Capture example

- Measure period of a waveform
- Capture time at each rising edge
  - 1<sup>st</sup> edge = time T1
  - 2<sup>nd</sup> edge = time T2
  - Waveform period  $T = T2 - T1$



## Ex. Capture & interrupt on pin 0.2 rising

```
#include <lpc22xx.h>

void T0isr (void) __irq {
    static int value;
    value = T0CR0;           // Read the capture value
    T0IR = 0x00000010;      // Clear capture 0 interrupt
    VICVectAddr = 0;        // Dummy write to VIC to signal end of interrupt
}

int main(void) {
    VPBDIV = 0x00000002;    // Set pclk to 30 MHz (Assume cpu clk=60MHz)
    PINSEL0 = 0x00000020;   // Enable pin 0.2 as capture channel0
    TOPR = 0x00007530;      // Load prescaler for 1 Msec tick
    TOTCR = 0x00000002;     // Reset counter and prescaler
    T0CCR = 0x00000005;     // Capture on rising edge of channel0
    TOTCR = 0x00000001;     // Enable timer

    VICVectAddr4 = (unsigned) T0isr; // Set timer ISR vector address
    VICVectCntl4 = 0x00000024;       // Set channel
    VICIntEnable = 0x00000010;      // Enable the interrupt
    while (1);
}
```

---

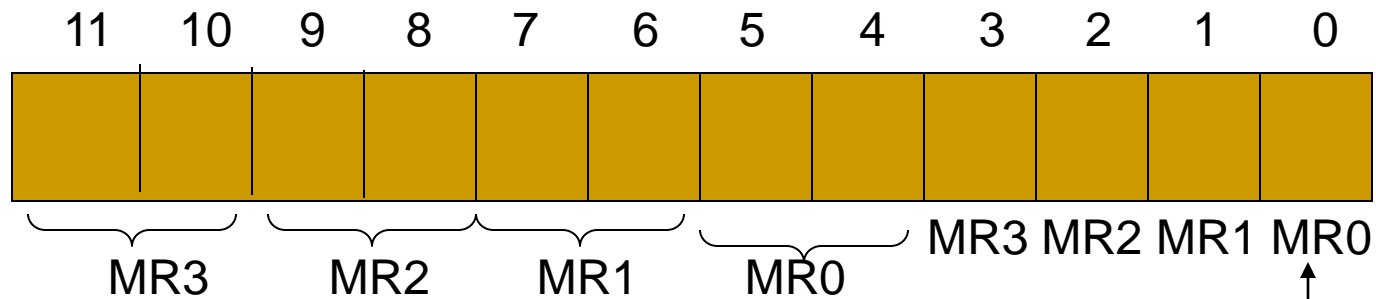
# Match registers

- Program match register with desired time to trigger an “event”
    - Trigger & set flag when match register = TC
    - Interrupt on match event (if enabled)
    - TC can continue, reset, or stop on the event
    - Can force state of output pin MATn.k (low, high, toggle)
  - Match registers MR0,MR1,MR2,MR3
  - Output pins (MATn.k controlled via MRk):
    - MAT0.0, MAT0.1, MAT0.2, MAT0.3  
MAT1.0, MAT1.1, MAT1.2, MAT1.3
    - Configure in External Match Register (EMR)
-

# External Match Control Register

Control and status of external match pins

Timer0: T0EMR, Timer1: T1EMR



Control MAT0.0 on match

00 – nothing

01 – reset

10 – set

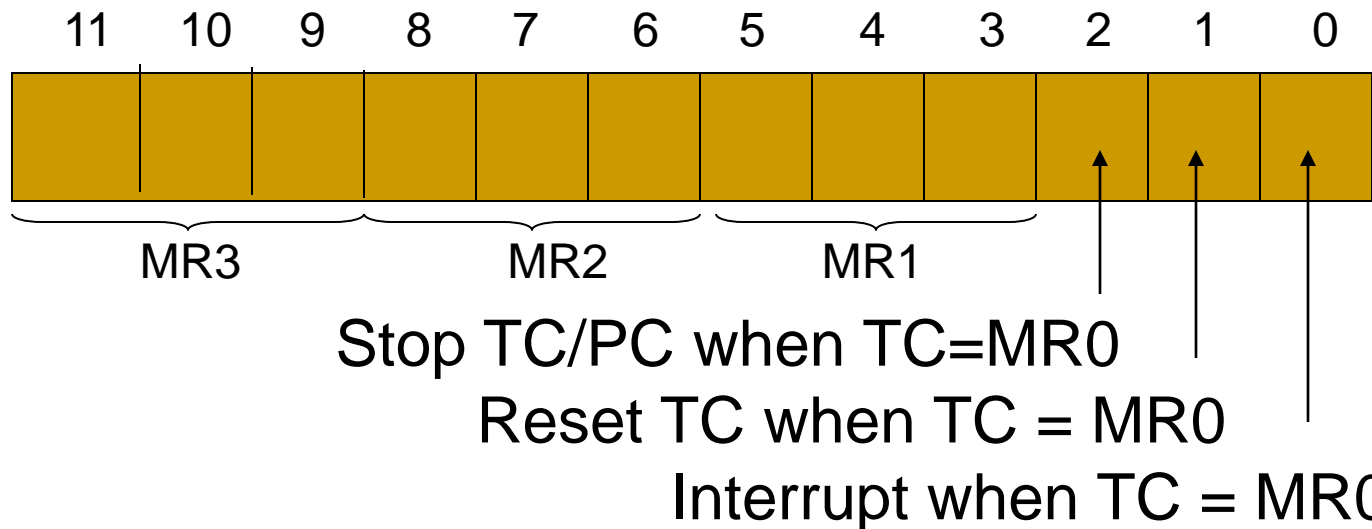
11 - toggle

External match 0:  
state of pin MAT0.0  
on match of MR0

# Match Control Register

Select action for TC/PC on match

Timer0: T0MCR, Timer1: T1MCR



# Match example

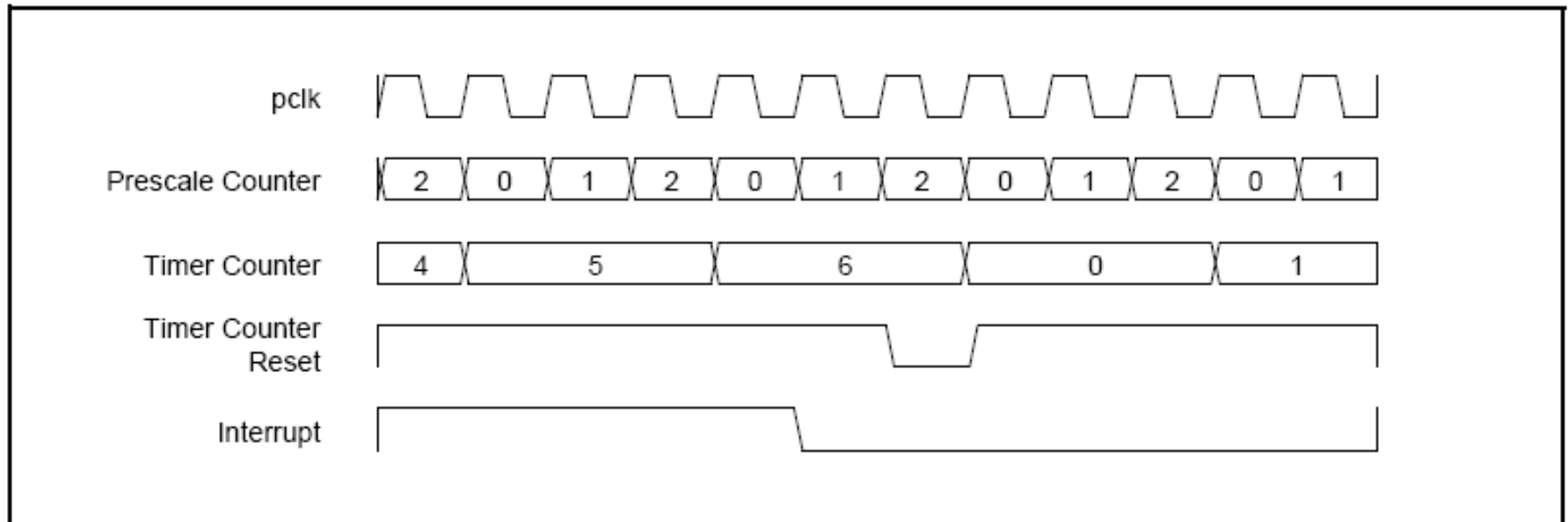


Figure 39: A timer cycle in which PR=2, MRx=6, and both interrupt and reset on match are enabled.

# Match example

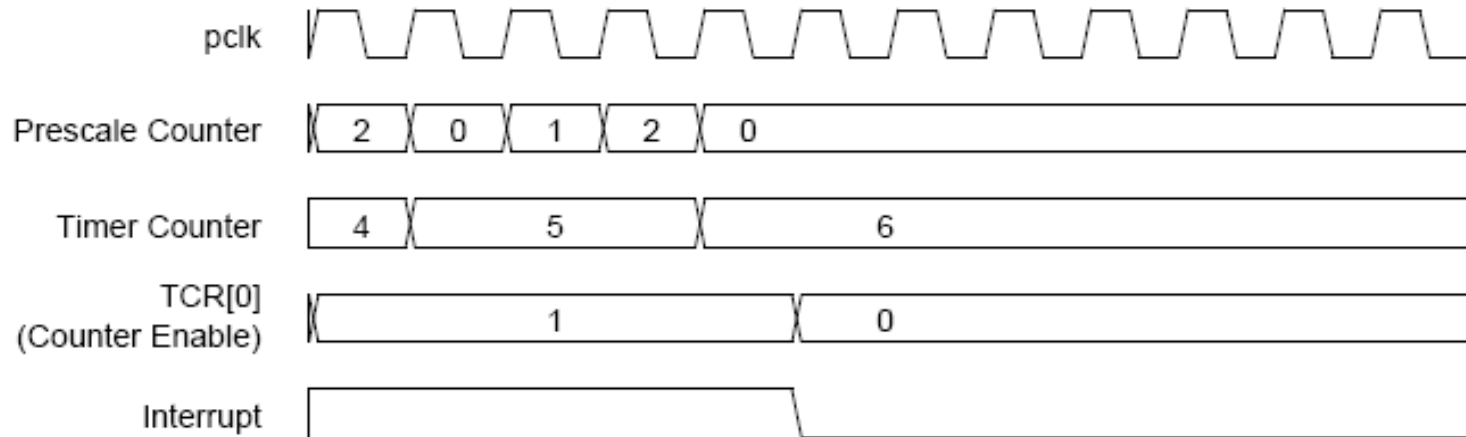
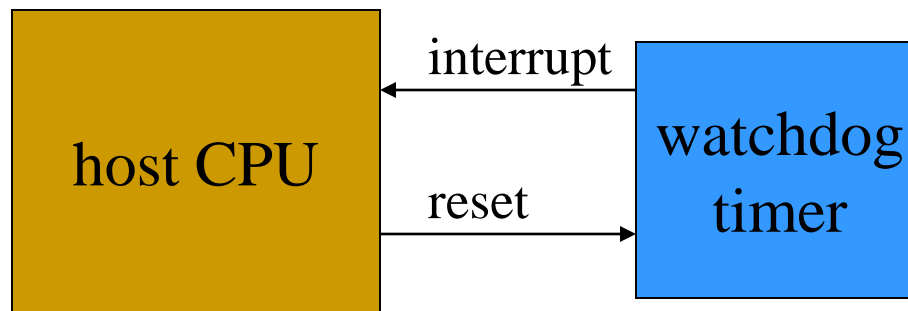


Figure 40: A timer cycle in which PR=2, MRx=6, and both interrupt and stop on match are enabled.



# Watchdog timer

- Watchdog timer must be periodically reset by system software (write “feed sequence”)
- If watchdog not reset, it generates an interrupt to **reset the host**.
- Use to detect problems with host



Program timeout value

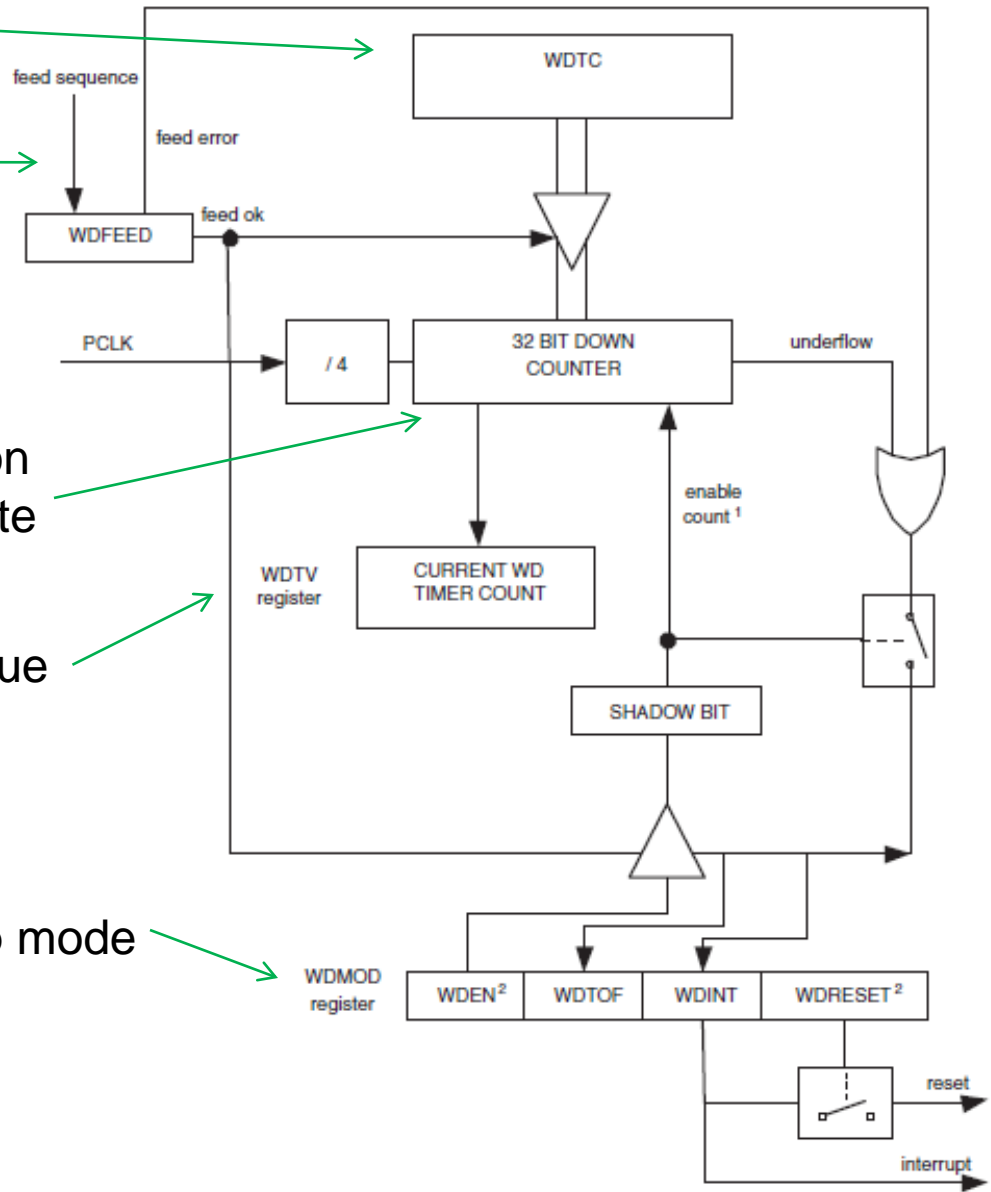
Write feed sequence  
0xAA – 0x55

Counter – resets on  
feed sequence write

Read timer value

Set op mode

# Watchdog Timer

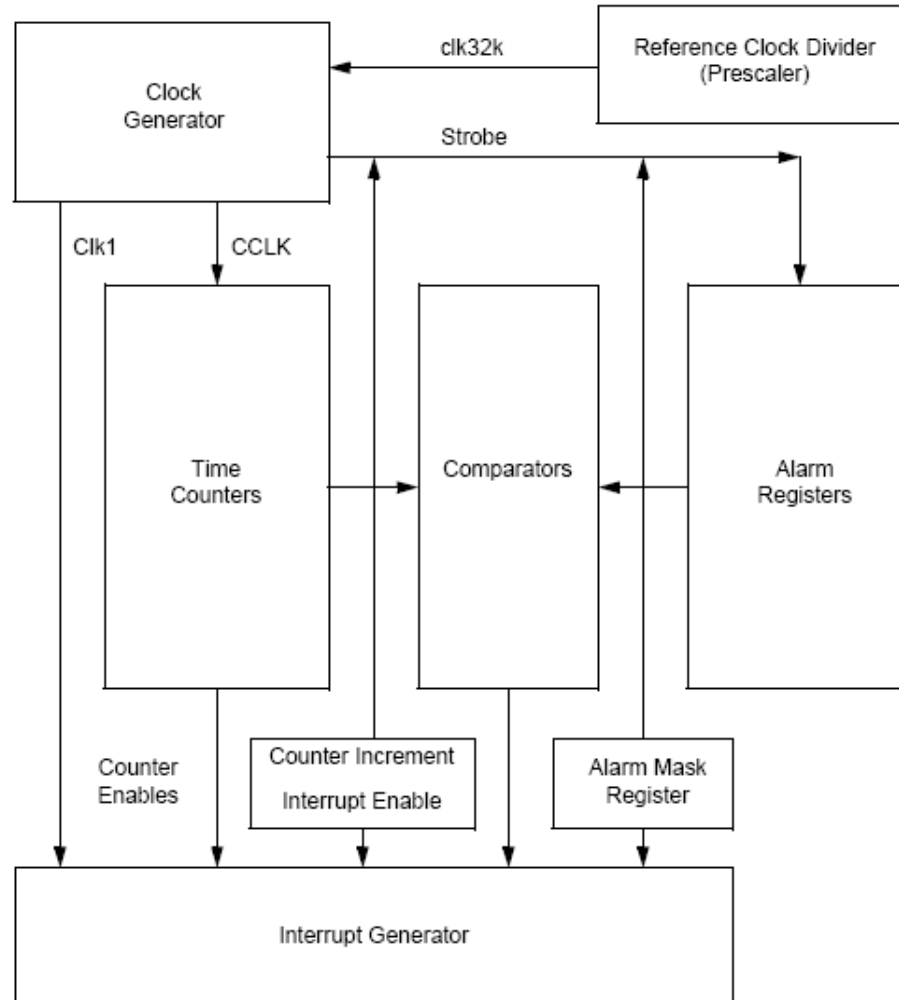


---

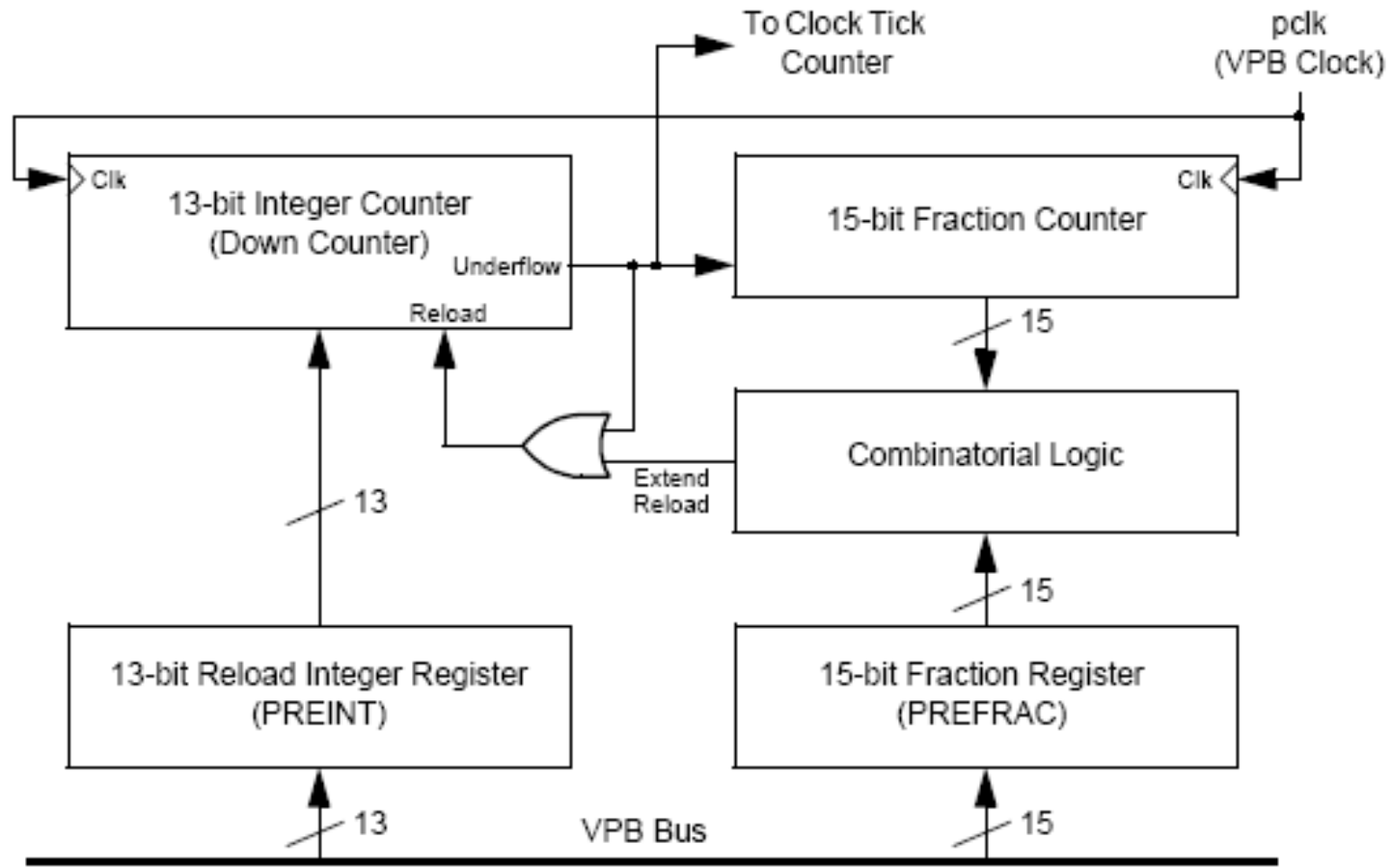
# Real time clock

- Set of counters to measure time
  - Maintains calendar and clock
    - Seconds, minutes, hours, day of month, month, year (to 2099), day of week, day of year
    - Alarm registers trigger events at designated times
  - Low power mode
    - Battery operation while CPU idle
-

# Real time clock structure



# Reference clock divider (prescaler)



# RTC prescaler registers

- RTC needs reference clock = 32.768 kHz
- Divide Pclk to get 32,768 pulses/sec
  - Fraction makes some pulses longer
- PREINT (13 bits)– integer prescale
  - Set to  $\text{PREINT} = (\text{int}) (\text{Pclk}/32768) - 1$
- PREFRAC (15 bits)– fractional prescale
  - Set to  $\text{PREFRAC} = \text{Pclk} - (\text{PREINT} + 1) \times 32768$

Example (Start RTC with Pclk=30 MHz)

```
PREINT = 0x00000392;    //914
PREFRAC = 0x00004380;    //17280
CCR = 0x00000001;       //start RTC
```

# Time counter registers

- Write to set; Read for current time
  - SEC = 0..59
  - MIN = 0..59
  - HOUR = 0..23
  - DOM (Day of month) = 1..28/29/30/31
  - DOW (Day of week) = 0..6
  - DOY (Day of year) = 1..365/366
  - MONTH = 1..12
  - YEAR = 0..4095
- Above also packed into 3 “consolidated time registers” (CTIME0,CTIME1,CTIME2)

---

# RTC alarm/interrupt registers

- Alarm reg: ALxxx (one per time register)
    - Time at which alarm to be signaled
    - Interrupt when time count registers match alarm reg's
  - Alarm mask reg (AMR)
    - Bit AMRxxx = 1 enables timer reg xxx to be included in the alarm check (0 excludes that timer reg)
  - Counter increment interrupt register (CCIR)
    - Bit IMxxx =1 to enable interrupt when time reg xxx is incremented
-

---

# RTC miscellaneous registers

- Clock control reg (CCR) – bits
    - CLCEN (enable the RTC)
    - CTCRES (reset the RTC)
  - Interrupt location (ILR) - 2 bits
    - RTCCIF: counter block interrupt
    - RTCALF : alarm function interrupt
    - Write 1 to reset the bit
  - Clock tick counter (CTC) (read only, 15 bits)
    - Clock divider counter value (< 1 second)
-