香 港 中 文 大 學（深 圳）
The Chinese University of Hong Kong, Shenzhen

ECE3080
Microprocessors and Computer Systems

Timer and Counter

Instructor：Tin Lun LAM

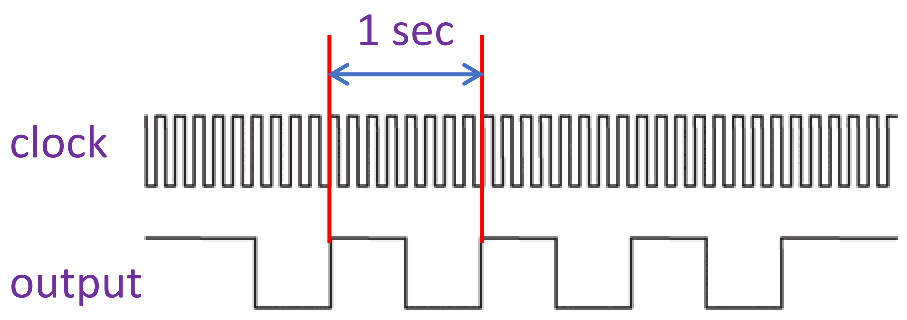
E-mail: tllam@cuhk.edu.cn

URL: https://myweb.cuhk.edu.cn/tllam

What is Timer/Counter?

What is timer/counter of microcontroller?

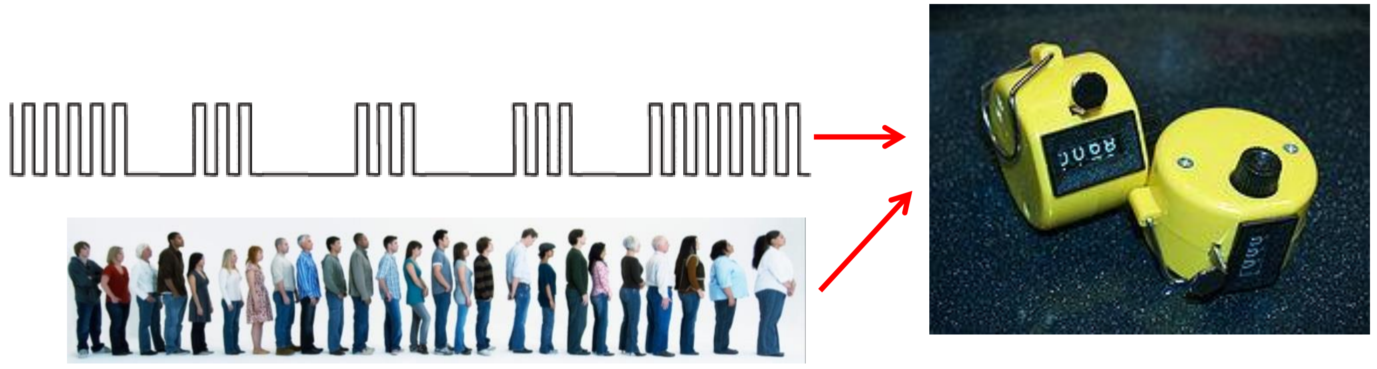
◆ (I). In computing and in embedded systems, a **programmable interval timer** (**PIT**) is a counter that generates an output signal when it reaches a programmed count. The output signal may trigger an interrupt.



Pictures from http://edcompassblog.smarttech.com, http://www.theguardian.com

What is timer/counter of microcontroller?

- (II). You can also use it to count the number of non-periodic pulses.
- E.g., you can count how many people (car, birds, …) entering/exiting a space by generating a pulse to the counter whenever there is an arrival/departure and letting the counter count the number of pulses.

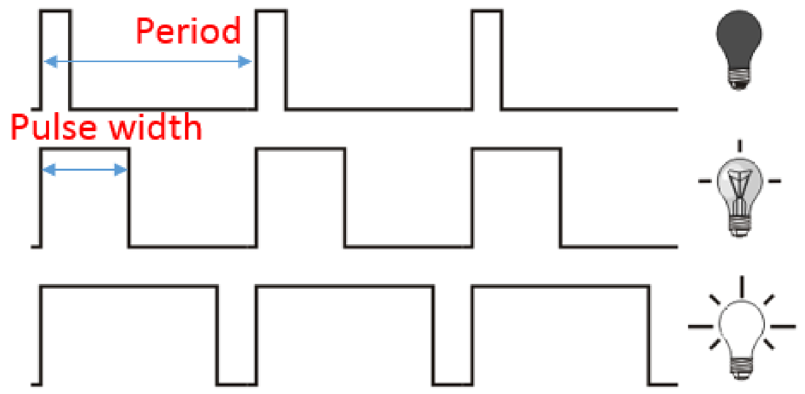


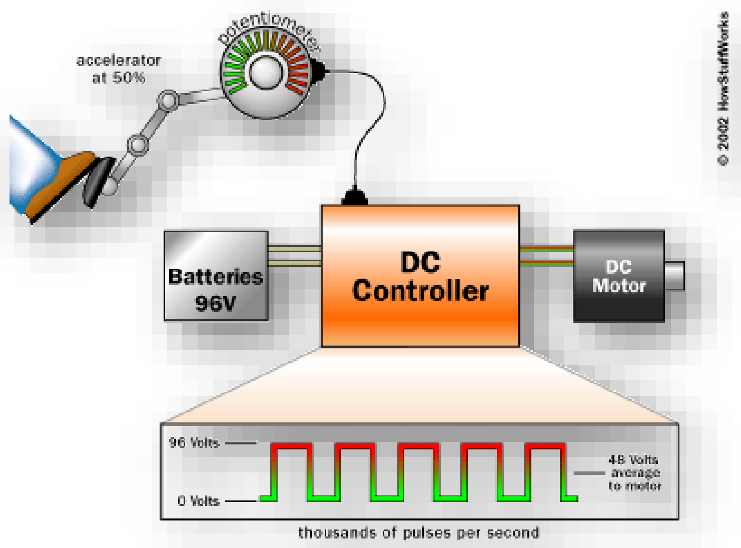
Pictures from http://en.wikipedia.org/wiki/Tally_counter

Timer's applications

Timer/counter applications

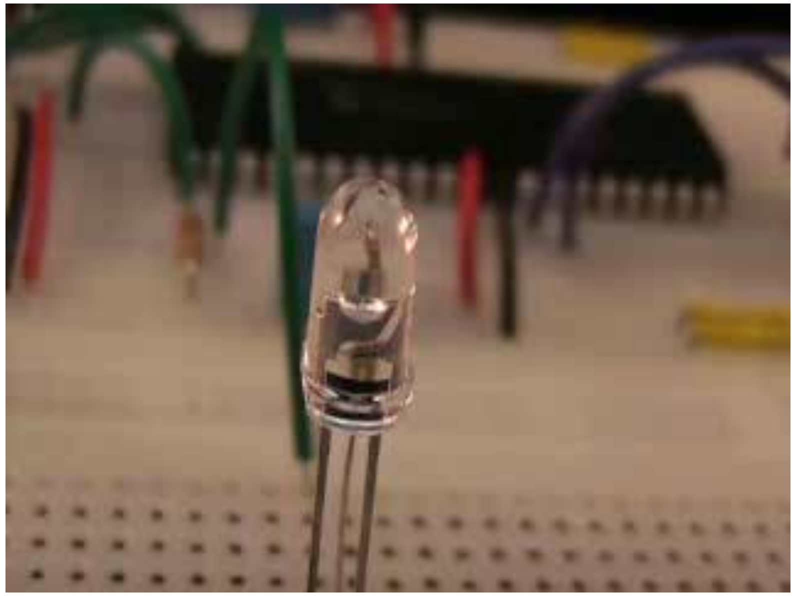
- One advanced feature of time/counter is to generate or capture Pulse Width Modulation (PWM) signals.
- PWM is a technique for various controls, e.g., motor speed and power control and light intensity control.

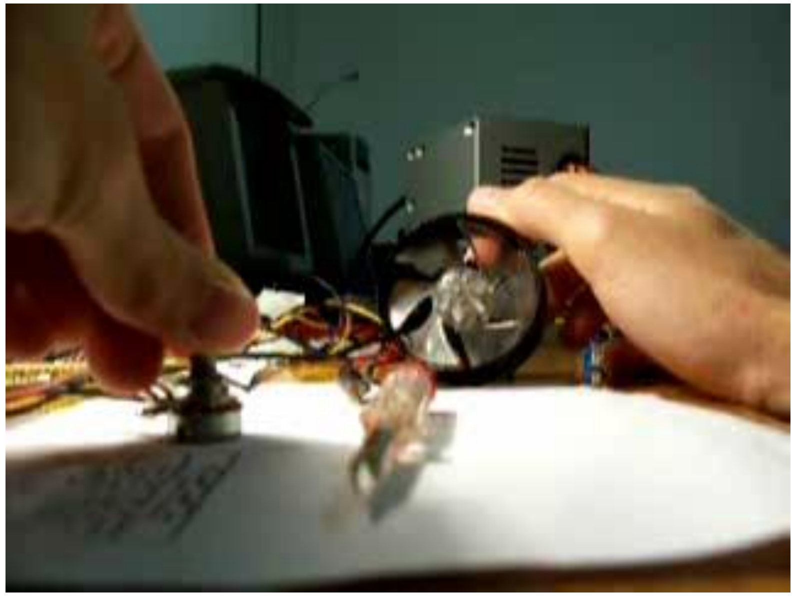


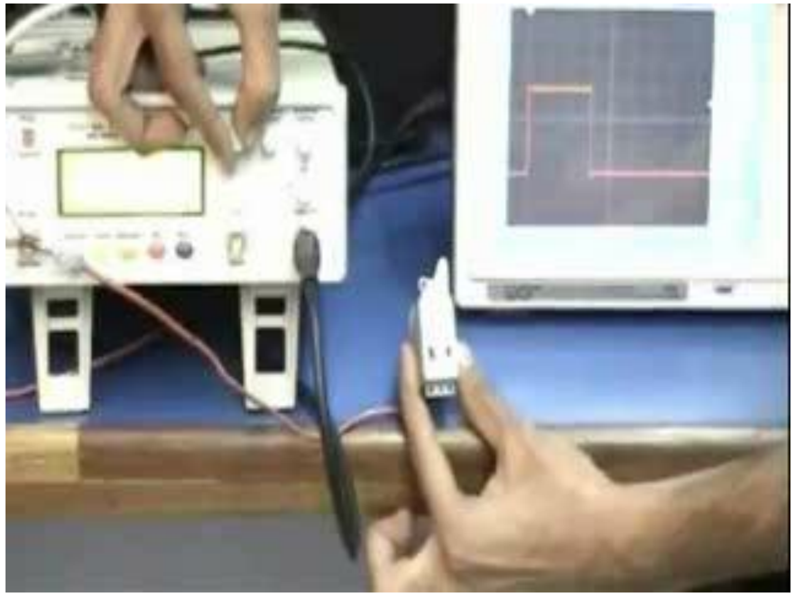


Pictures from http://ww.rs-online.com, http://www.eepw.com.cn
http://en.wikipedia.org, http://auto.howstuffworks.com, http://www.litfurniture.com

Timer/counter applications



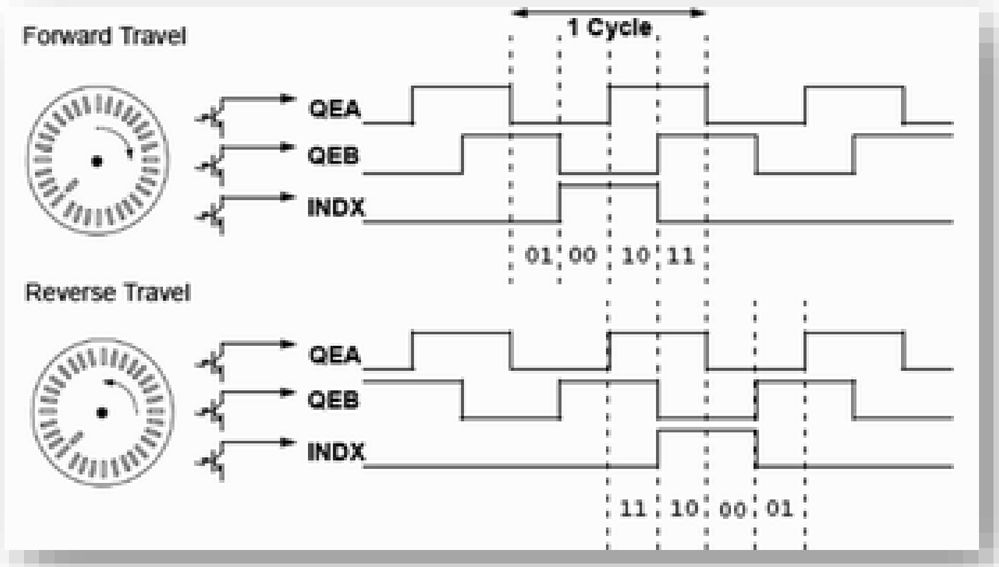




Quadrature encoding

- Quadrature encoding is a technique that measures rotation directions and speed of motors and tuning knobs. E.g. microwave oven time/power control. (http://en.wikipedia.org/wiki/Rotary_encoder)
- Two sensors are placed 90° out of phase and the sensors output, QEA and QEB, can be input to a timer to decode the rotation direction and speed using timer' s quadrature encoder function (pp.57).

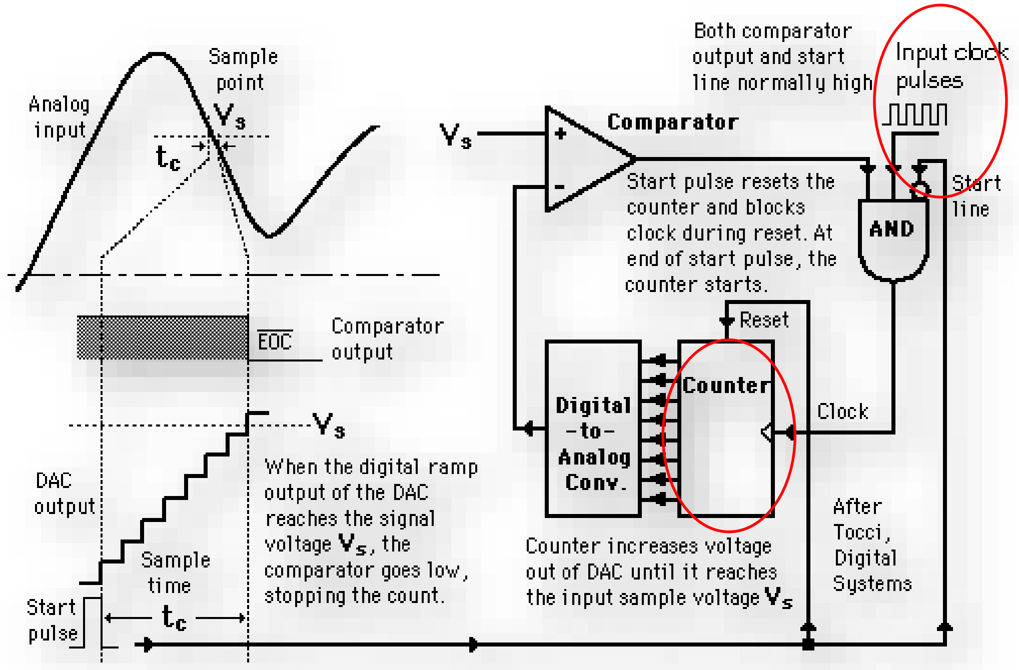
E.g. microwave oven panel.



Pictures from http://ww.rs-online.com, http://www.eepw.com.cn http://www.amazon.com

Timer/counter applications – A/D and D/A

- Analog-to-digital (A/D) and digital-to-analog (D/A) also need a time base for conversion.


Analog input
Sample point
V_s
t_c
$\overline{EOC}$
Comparator output
V_s
DAC output
When the digital ramp output of the DAC reaches the signal voltage V_s, the comparator goes low, stopping the count.
Sample time
Start pulse
t_c
V_s
Comparator
Both comparator output and start line normally high
Input clock pulses
Start pulse resets the counter and blocks clock during reset. At end of start pulse, the counter starts.
AND
Start line
Reset
Digital -to- Analog Conv.
Counter
Clock
Counter increases voltage out of DAC until it reaches the input sample voltage V_s
After Tocci, Digital Systems

A Digital Ramp ADC

Pictures from http://hyperphysics.phy-astr.gsu.edu

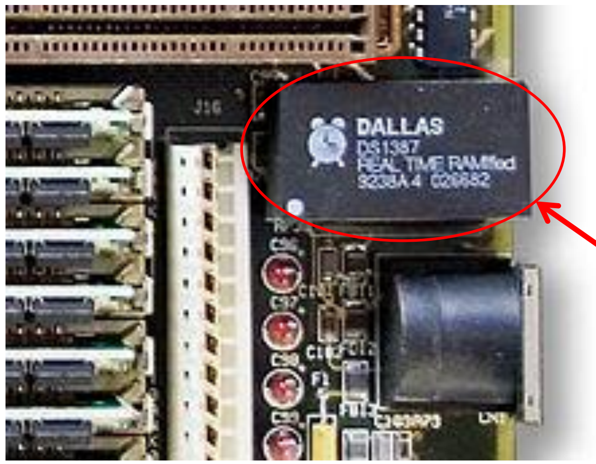
Timer/counter applications – multitasking

- **Multitasking** – a method where multiple tasks are performed during the same period of time. The tasks share common processing resources, such as central processing units (CPUs) and main memory.
- For single-CPU computer, only one task is running at any point in time, meaning that the CPU is actively executing instructions for that task.
- Multitasking solves the problem by scheduling (using a timer) which task may be running at any given time, and when another waiting task gets a turn.
- The act of reassigning a CPU from one task to another one is called a context switch. When context switches occur frequently enough, the illusion of parallelism is achieved.

Contents from http://en.wikipedia.org/wiki/Computer_multitasking
Pictures from http://blog.intercom.io, http://mercercognitivepsychology.pbworks.com

Timer/counter applications – Real time clock

- A **real-time clock** (**RTC**) is a computer clock (often in the form of an IC) that keeps track of the current time.
- Although the term often refers to the devices in personal computers, servers and embedded systems, RTCs are present in almost any electronic device which needs to keep accurate time.
- E.g., Cortex-M3 has an embedded RTC inside.
- Benefits:
  - Low power consumption (important when running from alternate power)
  - Frees the main system for time-critical tasks
  - Sometimes more accurate than other methods



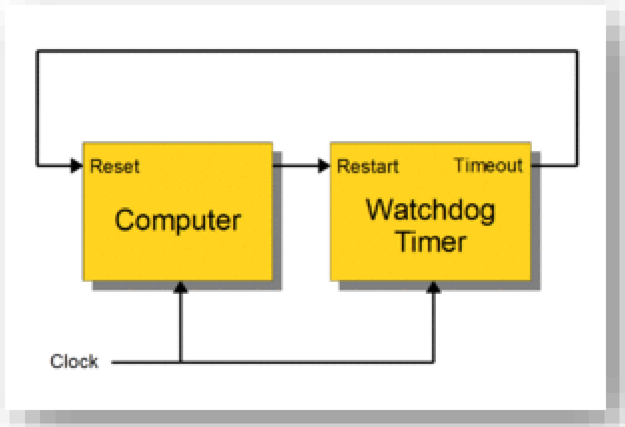
An external RTC chip for Intel x86 system

Mo	Tu	We	Th	Fr	Sa	Su
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

Contents from http://en.wikipedia.org/wiki/Real-time_clock
Pictures from http://hyperphysics.phy-astr.gsu.edu, http://www.amigaos.net

Timer/counter applications – Watch dog timer

- A **watchdog timer** (WDT; sometimes called a *computer operating properly timer* or *COP* timer, or simply a *watchdog*) is an electronic timer that is used to detect and recover from computer malfunctions. Q: How?
- The computer shall regularly restart the watchdog timer during normal operation to prevent it from elapsing, or "timing out".
- Q: What If a hardware fault or program error occurs and the computer fails to restart the watchdog?
- A: The timer will elapse and generate a *timeout signal* that can be used to initiate corrective action or actions to place the computer system in a safe state and restore normal system operation.



Contents from http://en.wikipedia.org/wiki/Watchdog_timer

Pictures from http://en.wikipedia.org/

Basic knowledge of Timer

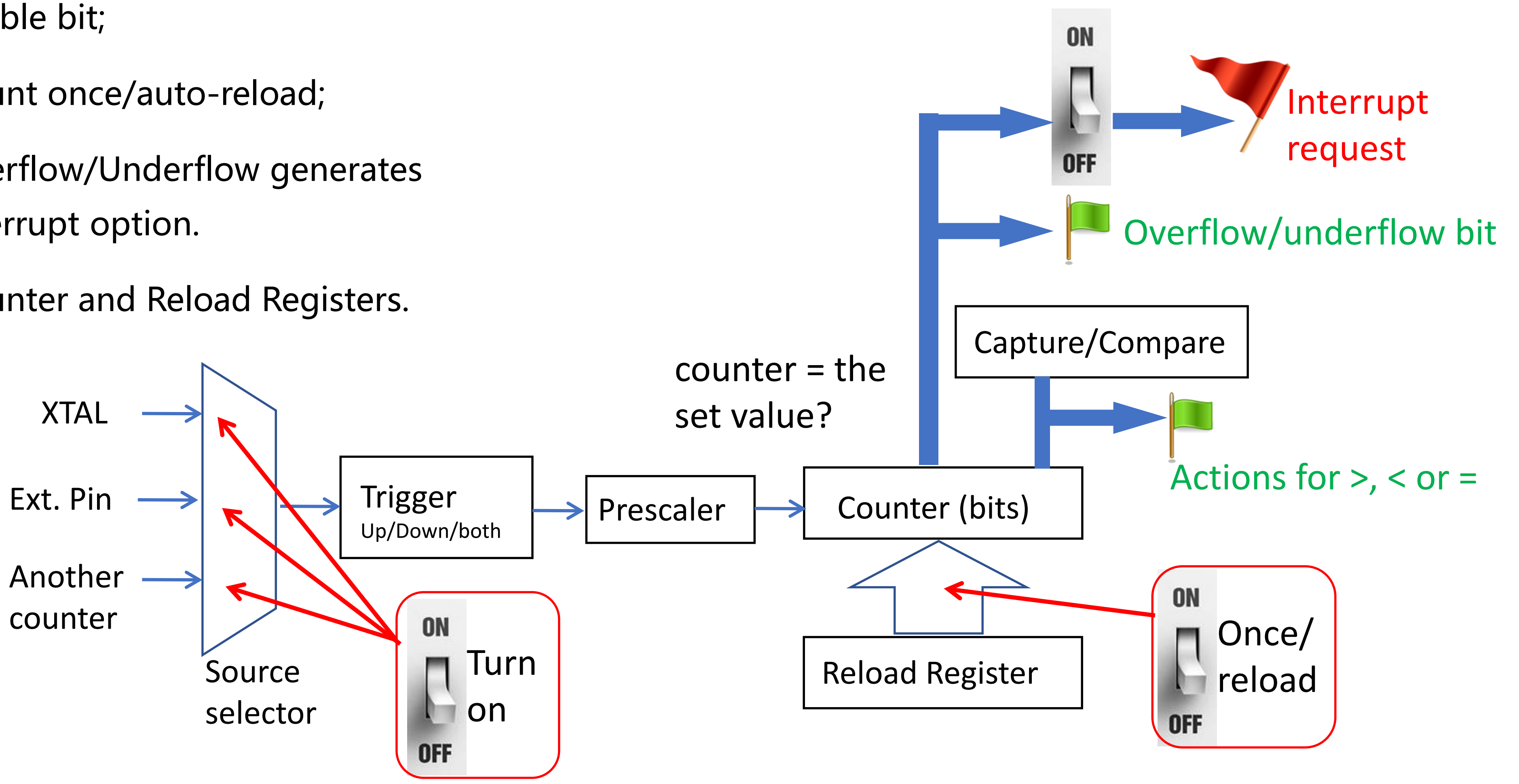
Timer Categories

- **Systick**
  - System tick timer, used to provide a periodic timing function. This timer can generate interrupt signals periodically, serving as the heartbeat clock of the system, which plays a role in task switching or time slicing for task allocation.
- **Basic timer**
  - No external input/output, mainly used for time base counting and timing.
- **General-purpose timer**
  - In addition to the basic timer functions, it can provide functions such as input capture, output compare, and connection to other sensor interfaces.
- **Advanced timer**
  - In addition to the general-purpose timer functions, it can provide functions such as motor control, digital power application, and programmable dead time for complementary output.
- **RTC (Real-Time Clock)**
  - Mainly used to provide real-time time and date.
- **Watchdog**
  - Mainly used to monitor whether the system has encountered exceptions.

Basic knowledge about Timer

Timer always have:

- Source selection (from Xtal, external source or other counters);
- Enable bit;
- Count once/auto-reload;
- Overflow/Underflow generates interrupt option.
- Counter and Reload Registers.



Cortex-M3 and STM32's Timers

Cortex-M3 timers

- Advanced-Control Timers (Tim1, 8). RM0008 Ch-14.
- **General Purpose Timers (Tim2, 3, 4, 5). RM0008 Ch-15.**
- Basic Timers (Tim6, 7). RM0008 Ch-17.
- General Purpose Timers (Tim9, 10, 11, 12, 13, 14). RM0008 Ch-16.
- **Real Time Clock (RTC). RM0008 Ch-18.**
- Independent Watch Dog Timers (IWDG). RM0008 Ch-19.
- Window Watch Dog Timers (WWDG). RM0008 Ch-20.
- **SysTick timer (SYSTICK). Text book Ch-8.5.**

General-purpose Timers of Cortex-M3 (TIM 2, 3, 4 and 5)

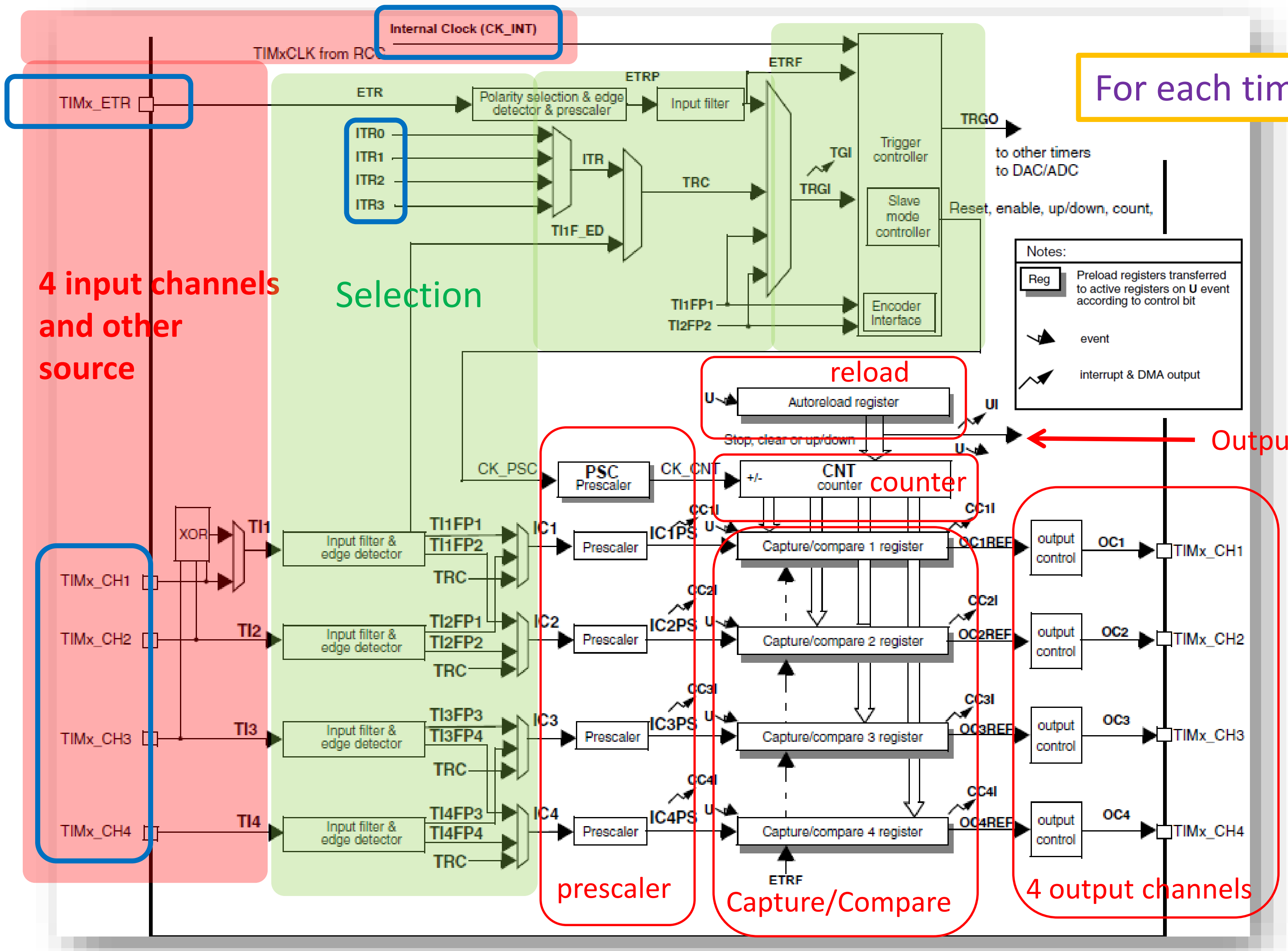
General-purpose timers (Tim2 to Tim5)

- 16-bit up, down, up/down auto-reload counter.
- 16-bit programmable prescaler used to divide the counter clock frequency by any factor between 1 and 65535.
- For each timer, up to 4 independent channels that can be used for:
  - Input capture
  - Output compare
  - PWM generation (Edge- and Center-aligned modes)
  - One-pulse mode output
- Synchronization circuit to control the timer with external signals and to interconnect several timers.
- Generation of Interrupt/DMA on the following events:
  - Update: counter overflow/underflow, counter initialization (by software or internal/external trigger)
  - Trigger event (counter start, stop, initialization or count by internal/external trigger)
  - Input capture
  - Output compare
- Supports incremental (quadrature) encoder and hall-sensor circuitry for positioning purposes
- Trigger input for external clock or cycle-by-cycle current management

Contents from RM0008 Ch-15

General-purpose timers (Tim2 to Tim5)

香港中文大學(深圳) The Chinese University of Hong Kong, Shenzhen



Pictures from RM0008 Ch-15

Clock selection

- The counter clock can be provided by the following clock sources:

1. Internal clock (CK_INT)
2. External clock mode1: external input pin (TIx)
3. External clock mode2: external trigger input (ETR)
4. Internal trigger inputs (ITRx): using one timer as prescaler for another timer. For example, you can configure Timer 1 to act as a prescaler for Timer 2, Timer 1 generates a trigger pulse at the end of the count and the pulse is used as the clock for Timer 2.

Contents from RM0008 Ch-15

The time-base unit

- The time-base unit includes 16-bit counter registers.
  - Counter Register (TIMx_CNT) (This x denotes the timer #)
  - Prescaler Register (TIMx_PSC):
  - Auto-Reload Register (TIMx_ARR)
- The main block of the programmable timer is a 16-bit counter and its auto-reload register.
- The counter can count (1) up, (2) down or (3) up and down.
- The counter input clock can be divided by a prescaler (1 to 65535).
- The counter, the auto-reload register and the prescaler register can be written or read by software. Note that this can take place even when the counter is running.

Contents from RM0008 Ch-15

The time-base unit

- The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register.
- The content of the preload register are transferred into the shadow register permanently or at each Update Event (UEV), depending on the value of the auto-reload preload enable bit (ARPE) in the time control register (TIMx_CR1 register). (see p.65)
- The update event (UEV) is sent when (i) the counter reaches the overflow (or underflow when downcounting) and (ii) if the UDIS bit equals 0 in the TIMx_CR1 register. It can also be generated by software.

General-purpose timers - Up-counting mode

Upcounting mode

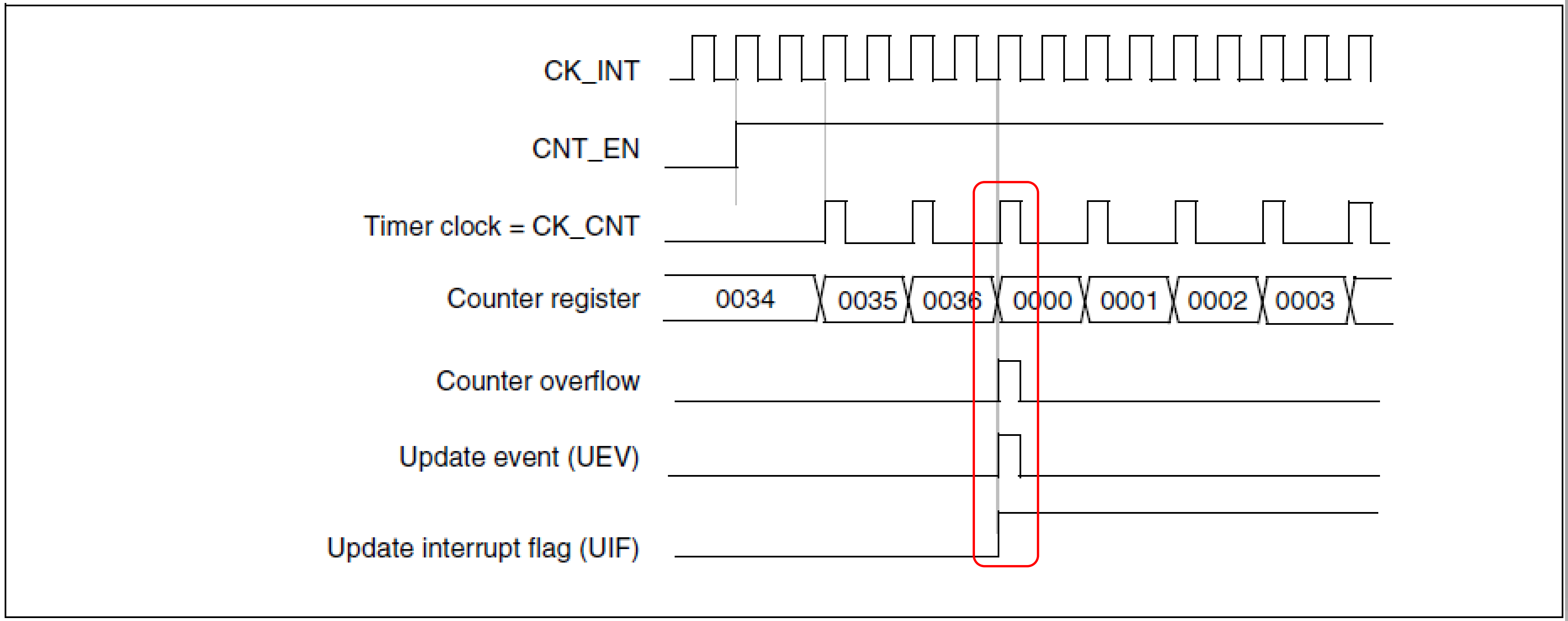
- In upcounting mode,
  - the counter counts from 0 to the auto-reload value (content of the TIMx_ARR register);
  - then restarts from 0 and generates a counter overflow event.
- An update event (UEV) signal can be generated at each counter overflow or by setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller).
  (UG: update generation bit)
- When an UEV occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set, depending on the URS (update request selection) bit:
  - The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register)
  - The auto-reload shadow register is updated with the preload value (TIMx_ARR)

Contents from RM0008 Ch-15

Upcounting mode

- Assume the prescaler control register is set to 1 (=divided by 2).

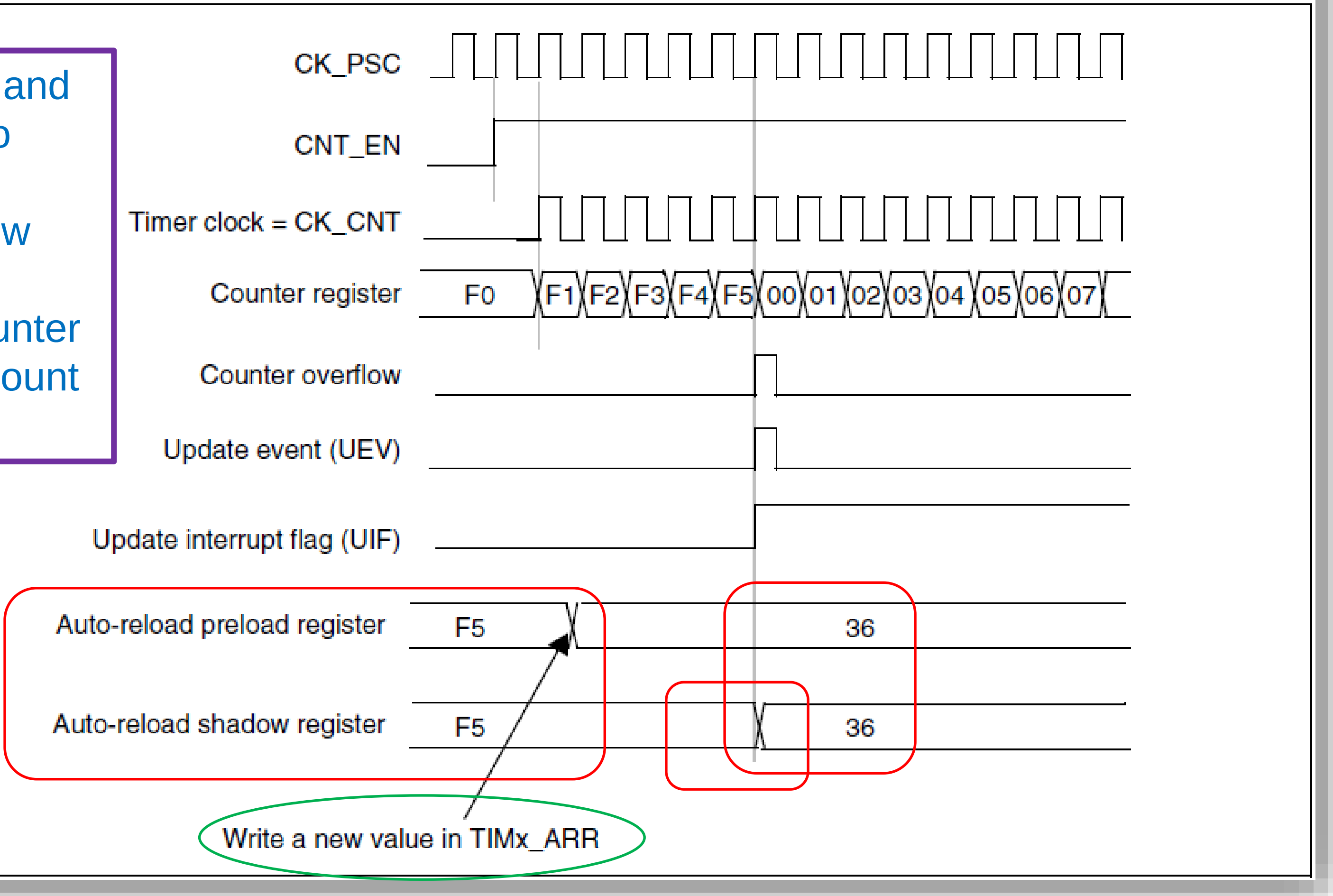
Figure 104. Counter timing diagram, internal clock divided by 2



Contents from RM0008 Ch-15

Upcounting mode

Figure 108. Counter timing diagram, Update event when ARPE=1 (TIMx_ARR preloaded)



TIMx_ARR=0xF5 and is later changed to 0x36.
If ARPE=1, shadow register will be updated when counter reaches the first count value

Contents from RM0008 Ch-15

General-purpose timers - Down-counting mode

Downcounting mode

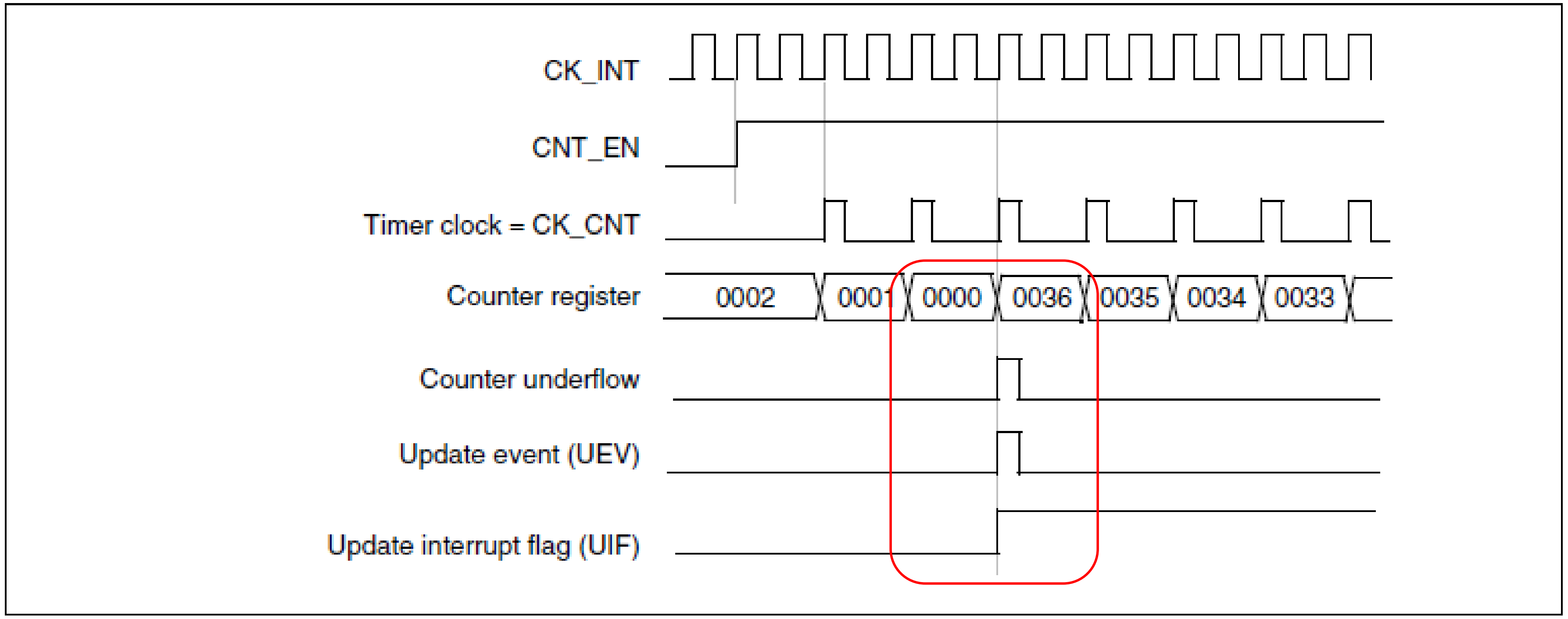
- It is similar to upcounting mode.
- In downcounting mode,
  - the counter counts from the auto-reload value (content of the TIMx_ARR register) down to 0;
  - then restarts from the auto-reload value and generates a counter underflow event.
- An Update event signal can be generated at each counter underflow or by setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller).
- When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set (depending on the URS bit):
  - The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).
  - The auto-reload active register is updated with the preload value (content of the TIMx_ARR register). Note that the auto-reload is updated

Contents from RM0008 Ch-15

Downcounting mode

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

Figure 110. Counter timing diagram, internal clock divided by 2



Contents from RM0008 Ch-15

General-purpose timers — Center-aligned mode (Up / Down counting)

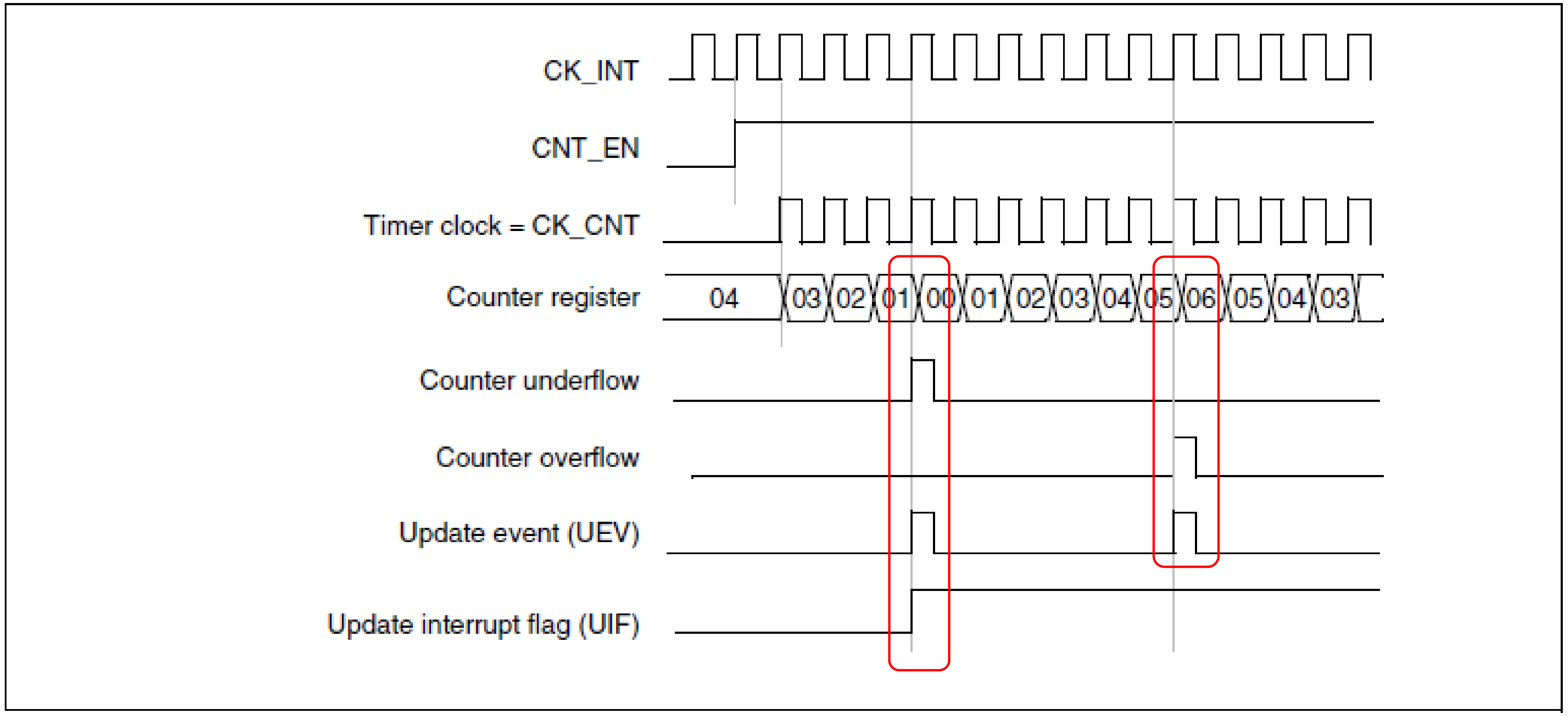
Center-aligned mode (Up/Down counting)

- In center-aligned mode,
  - the counter counts from 0 to the auto-reload value (content of the TIMx_ARR register) – 1;
  - then generates a counter overflow event;
  - then counts from the autoreload value down to 1;
  - then generates a counter underflow event;
  - then it restarts counting from 0.
- Center-aligned mode is active when the CMS bits in TIMx_CR1 register are not equal to '00'.
- In this mode, the direction bit (DIR from TIMx_CR1 register) cannot be written. It is updated by hardware and gives the current direction of the counter.

Contents from RM0008 Ch-15

Center-aligned mode (Up/Down counting)

Figure 114. Counter timing diagram, internal clock divided by 1, TIMx_ARR=0x6



Contents from RM0008 Ch-15

General-purpose timers – Capture/Compare Channels

Input Capture and Output Compare (wiki)

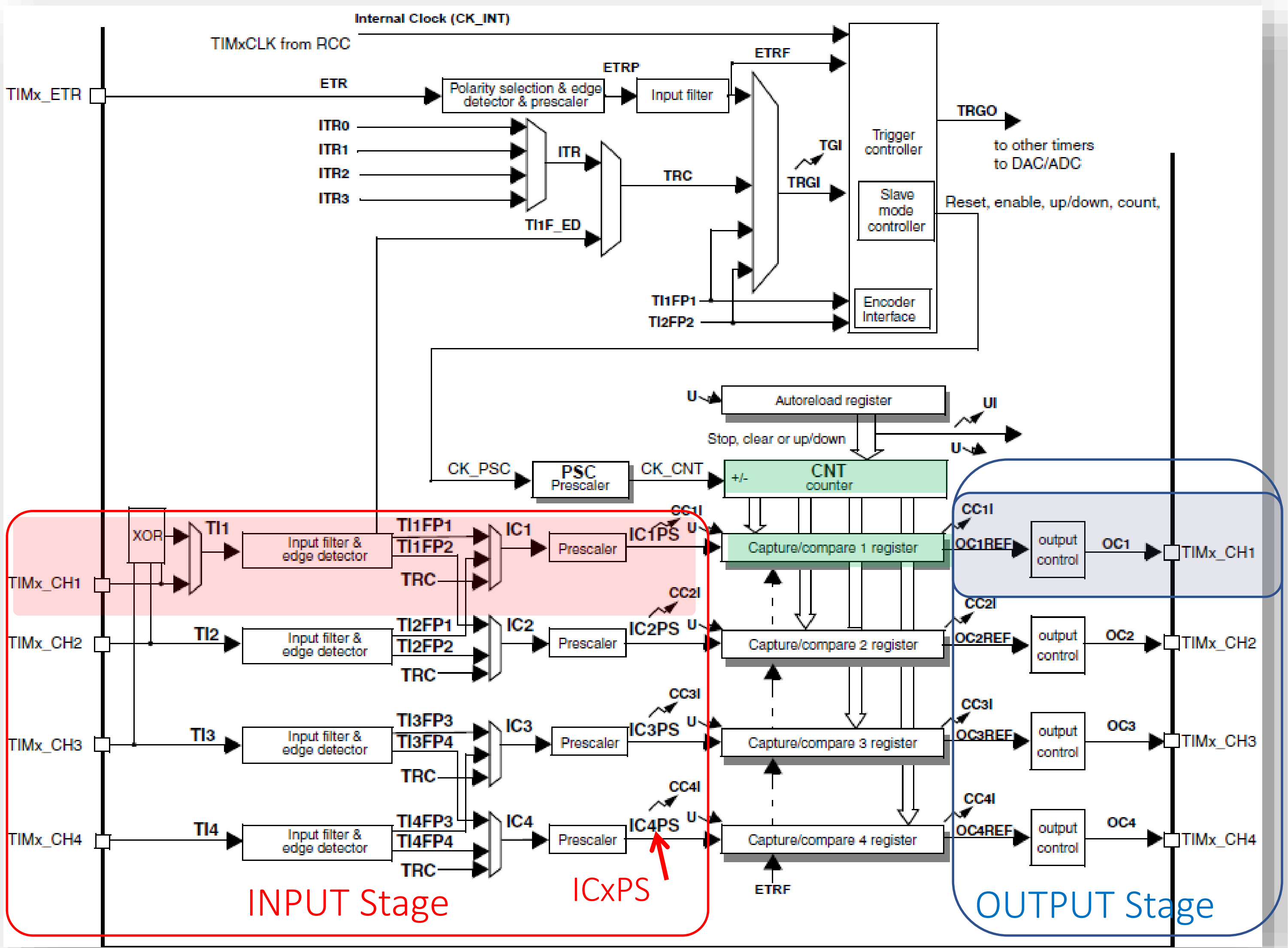
香港中文大學(深圳) The Chinese University of Hong Kong, Shenzhen

- **Input capture**
  - a method used in an embedded system to record a timestamp in memory when an input signal is received.
  - A flag is set when an input is captured. This allows the system to continue executing without interruption while an input is being received while still having the capability to trigger events based on the exact time when the input was received.
  - Input capture of a time can be used to calculate the duty cycle of Pulse width modulation (PWM) signal.
- **Output compare**
  - the corresponding capability to trigger an output at a specified time, based on a timestamp in memory.
- Many programmable interrupt controllers provide dedicated input capture pins and a programmable counter along with it. These pins generate interrupts to the controller, which then executes an interrupt service routine.

http://en.wikipedia.org/wiki/Input_capture

Capture/Compare channels

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen



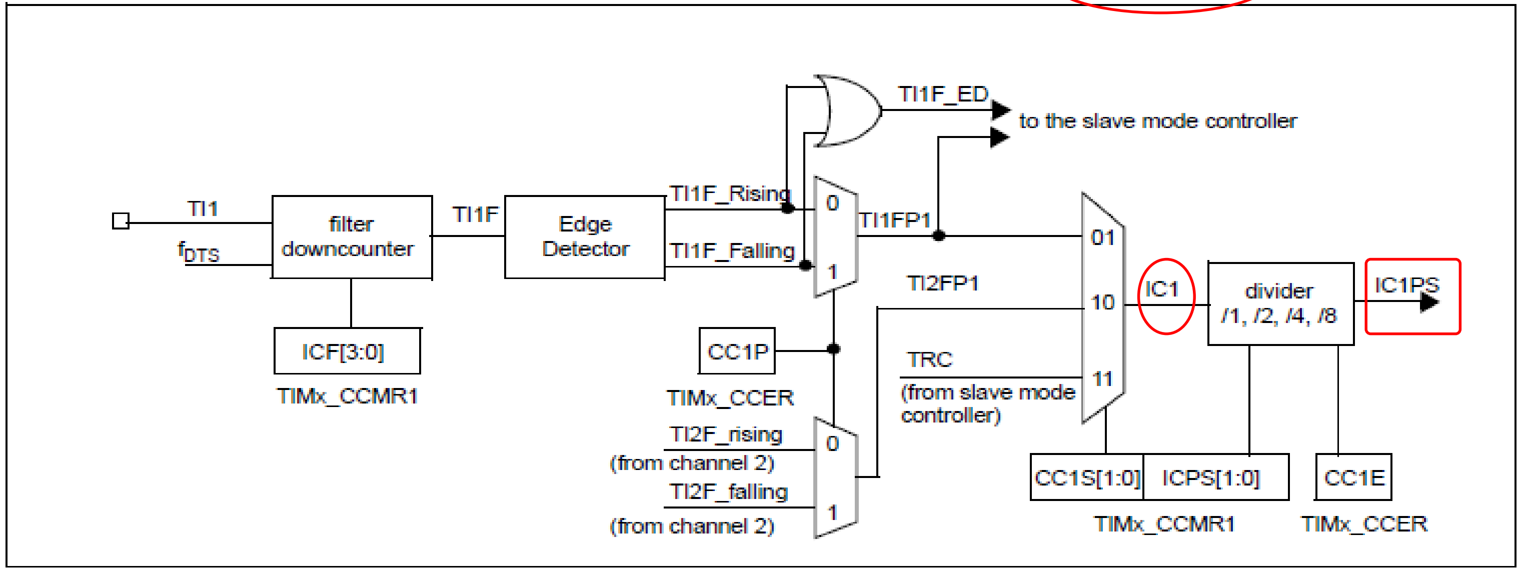
Contents from RM0008 Ch-15

Capture/Compare channels – Input stage

香港中文大學(深圳) The Chinese University of Hong Kong, Shenzhen

- The input stage samples the corresponding TIx input to generate a filtered signal TIxF with jitter removed. Then, an edge detector with polarity selection generates a signal (TIxFPx) that can be used as trigger input by the slave mode controller or as the capture command. It is prescaled before the capture register (ICxPS).

Figure 125. Capture/compare channel (example: channel 1 input stage)



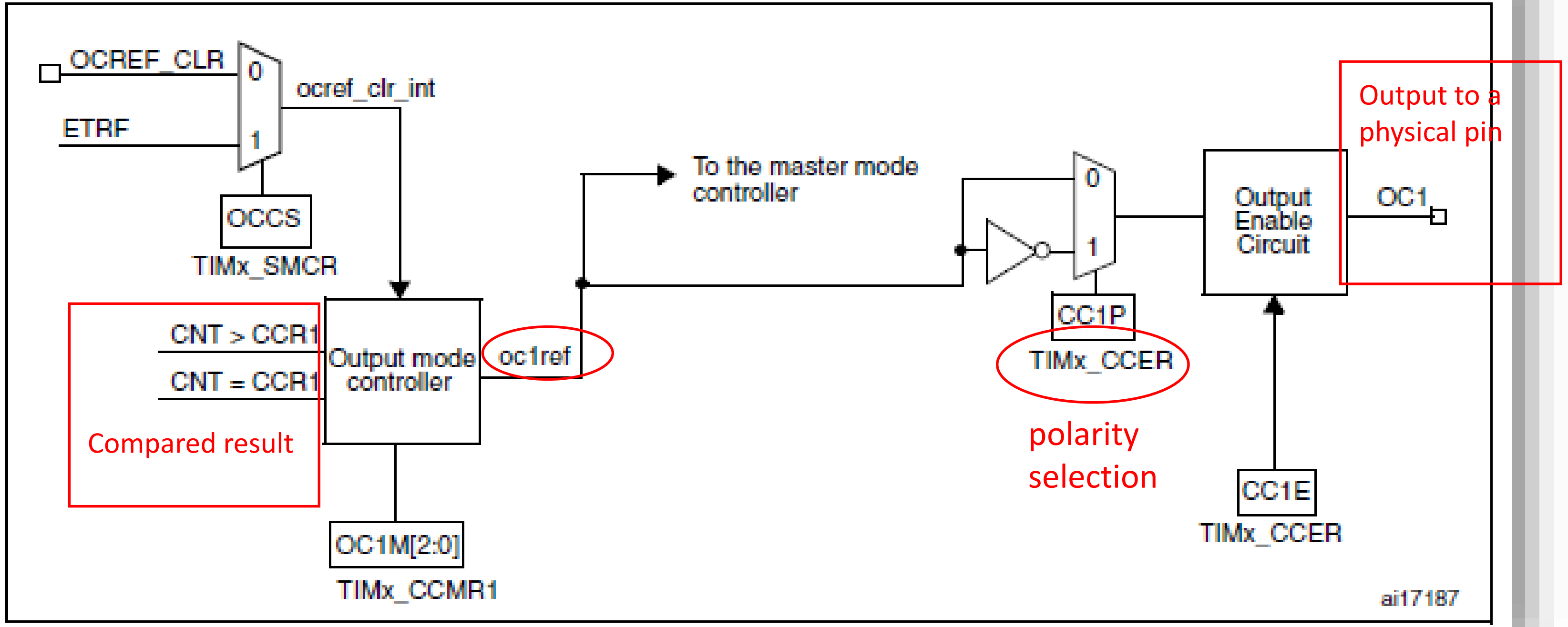
Contents from RM0008 Ch-15

Capture/Compare channels – Output stage

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

- The output stage generates an intermediate waveform, which is then used for reference: OCxRef (active high). The polarity acts at the end of the chain.

Figure 127. Output stage of capture/compare channel (channel 1)

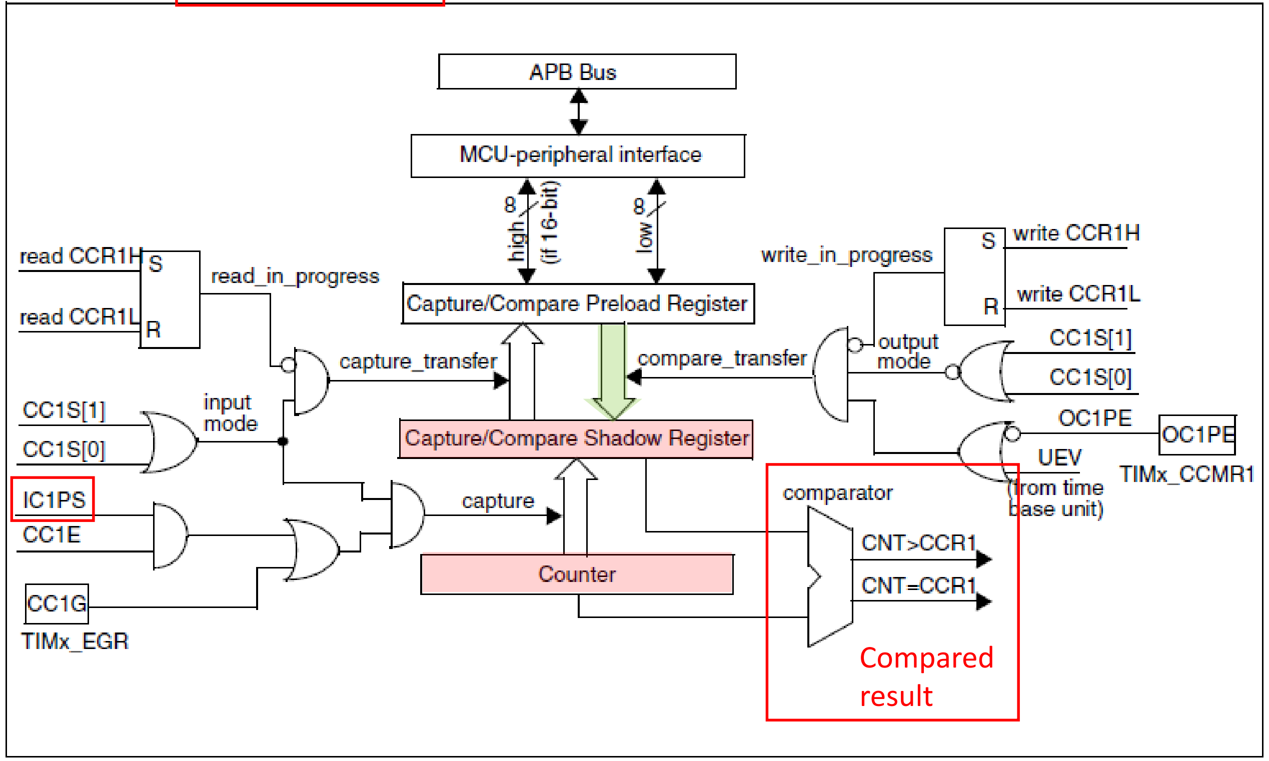


Contents from RM0008 Ch-15

Capture/Compare channels

- The capture/compare block is made of one capture/compare preload register and one shadow register. Write and read always access the preload register.
- In capture mode, captures are actually done in the shadow register, which is copied into the preload register.
- In compare mode, the content of the preload register is copied into the shadow register which is compared to the counter.

Figure 126. Capture/compare channel 1 main circuit



Contents from RM0008 Ch-15

General-purpose timers – Capture/Compare Modes

Input capture mode

- The Capture/Compare Registers (TIMx_CCRx) are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture occurs, the corresponding CCxIF flag (TIMx_SR register) is set and an interrupt or a DMA request can be sent if they are enabled.
- If a capture occurs while the CCxIF flag was already high, then the over-capture flag CCxOF (TIMx_SR register) is set. CCxIF can be cleared by software by writing it to 0 or by reading the captured data stored in the TIMx_CCRx register. CCxOF is cleared when you write it to 0.
- When an input capture occurs:
  - The TIMx_CCR1 register gets the value of the counter on the active transition.
  - CC1IF flag is set (interrupt flag). CC1OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.
  - An interrupt is generated, depending on the CC1IE bit.
  - A DMA request is generated, depending on the CC1DE bit.

Contents from RM0008 Ch-15

PWM Input mode

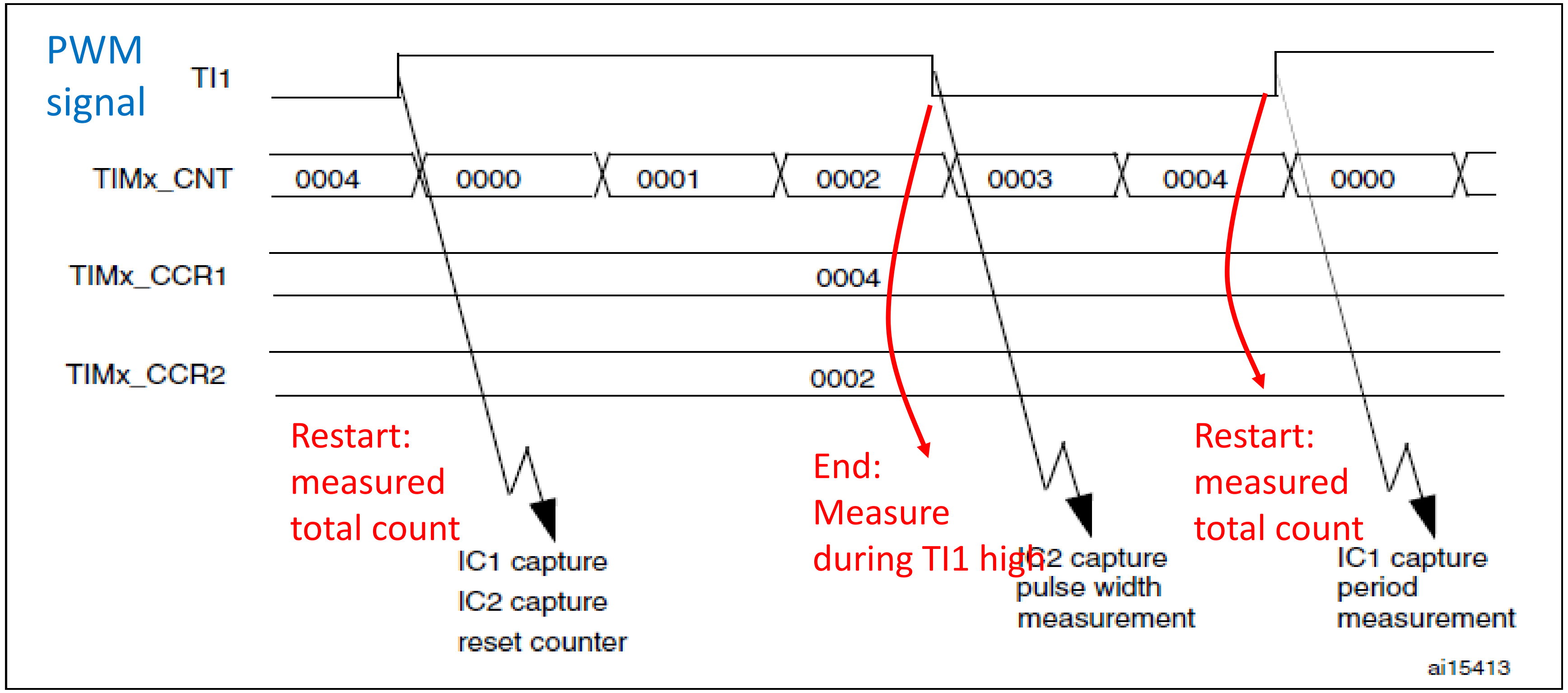
- This mode is one particular case of input capture mode. The procedure is the same except:
  - Two ICx signals are mapped on the same TIx input.
  - These 2 ICx signals are active on edges with opposite polarity.
  - One of the two TIxFP signals is selected as trigger input and the slave mode controller is configured in reset mode.
- For example, you can measure the period (in TIMx_CCR1 register) and the duty cycle (in TIMx_CCR2 register) of the PWM applied on TI1 using the following procedure (details of register setting are on page 378 of RM0008)
  - Selected TI1 for channel 1
  - Set active on rising edge for channel 1
  - Selected TI1 for channel 2
  - Set active on falling edge for channel 2
  - Configure the slave mode controller in reset mode
  - Enable the captures

Contents from RM0008 Ch-15

PWM Input mode

- Can be used to measure PWM cycle time (PWM period) and make time (PWM pulse width).

Figure 128. PWM input mode timing



Pictures from http://hyperphysics.phy-astr.gsu.edu

Forced output mode

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

- In output mode, each output compare signal can be forced to active or inactive level directly by software, independently of any comparison between the output compare register and the counter.
- To force an output compare signal to its active level, you just need to write 101 in the OCxM bits in the corresponding TIMx_CCMRx register. Thus ocxref is forced high (OCxREF is always active high).
- OCx get opposite value to CCxP polarity bit. E.g.: CCxP=0 (OCx active high) => OCx is forced to high level.
- The OCxREF signal can be forced low by writing the OCxM bits to ‘100’ in the TIMx_CCMRx register.

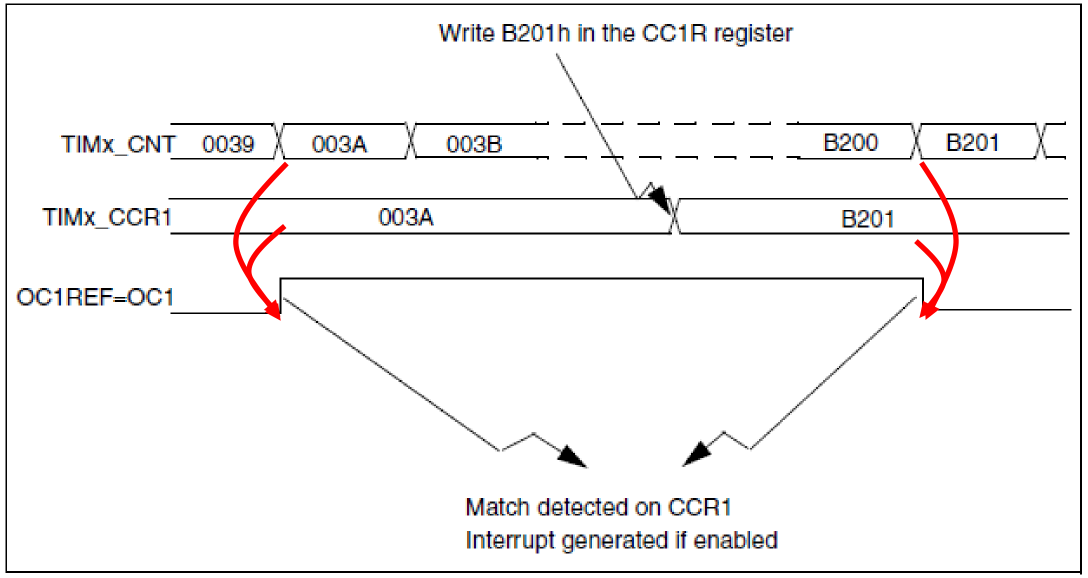
See Fig.127 on page 45 for the output signal.

Contents from RM0008 Ch-15

Output compare mode

- This function is used to control an output waveform or indicating when a period of time has elapsed.
- In output compare mode, the update event UEV has no effect on OCxREF and OCx output.
- The timing resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One-pulse mode).

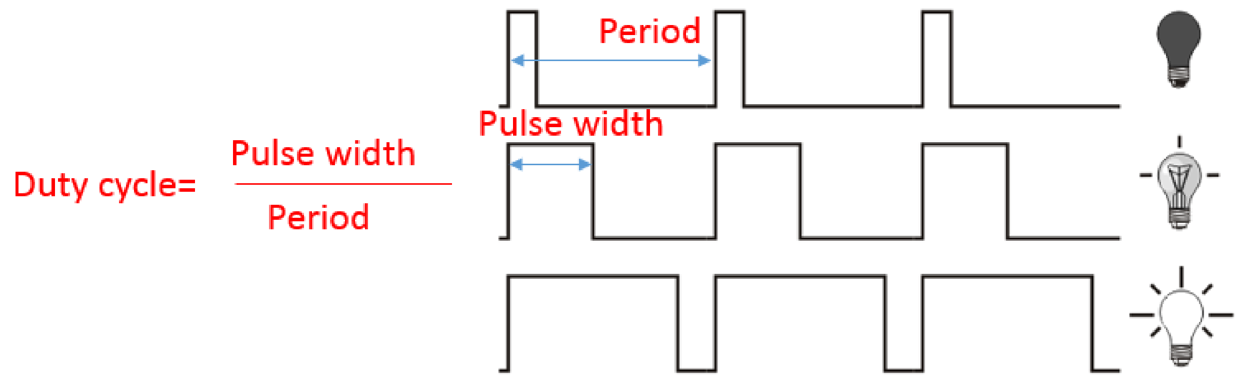
Figure 129. Output compare mode, toggle on OC1.

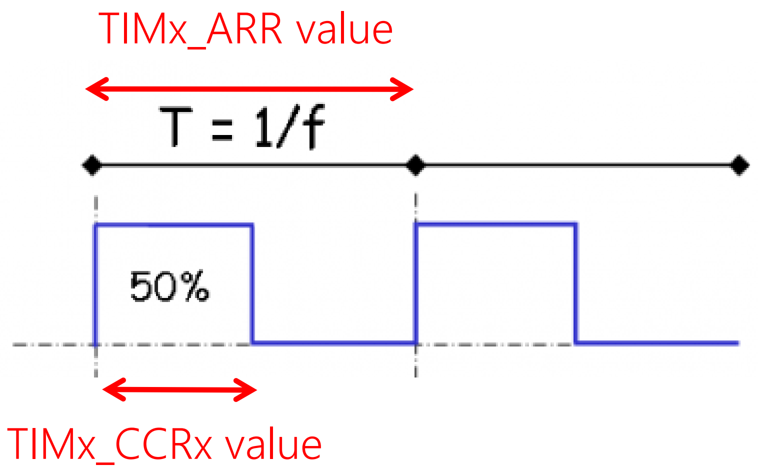


Contents from RM0008 Ch-15

PWM output mode (generation)

- Pulse width modulation mode allows you to generate a signal with a frequency determined by the value of the TIMx_ARR register and a duty cycle determined by the value of the TIMx_CCRx register.
- The PWM mode can be selected independently on each channel. TMIxCH1, TMIxCH2, TMIxCH3 and TIMxCH4.



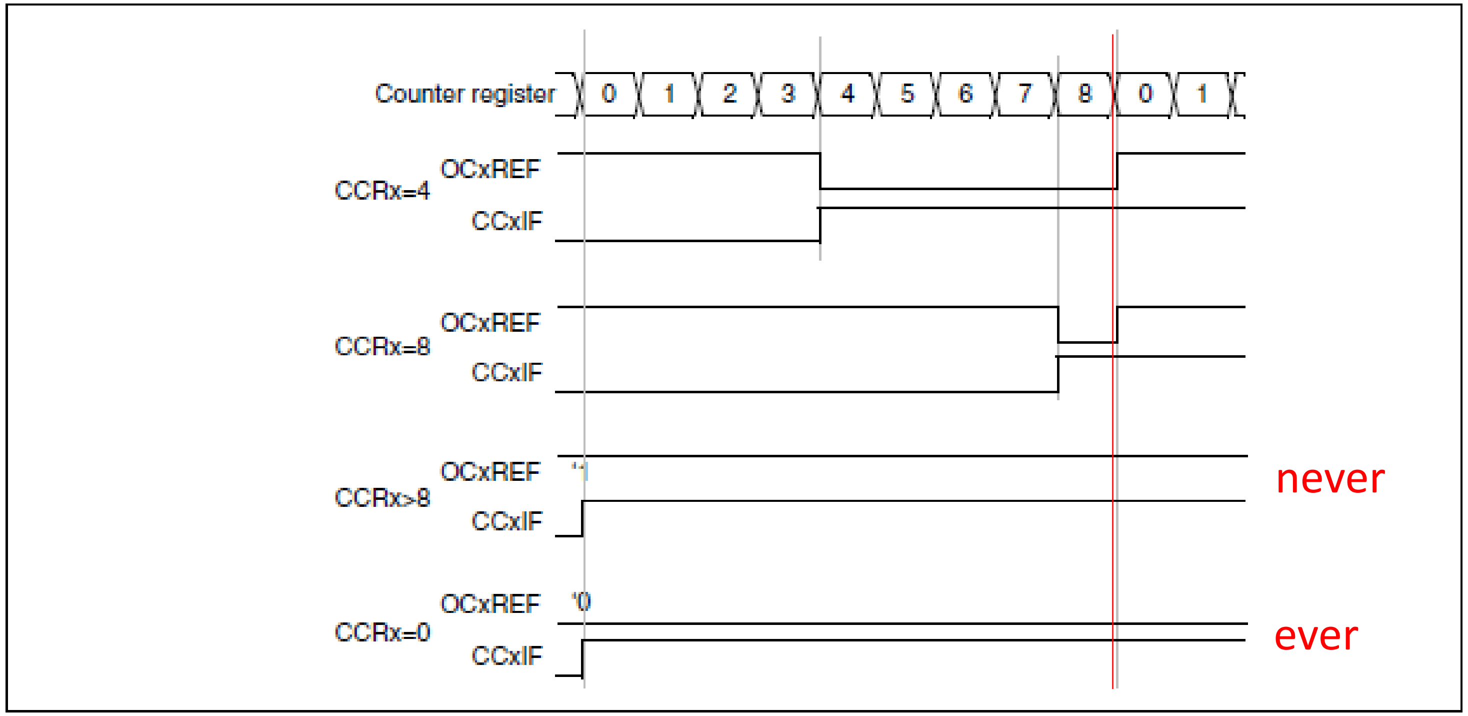


Contents from RM0008 Ch-15

PWM mode (generate)

- It also has edge-aligned mode (Upcounting and downcounting) and center-aligned mode

Figure 130. Edge-aligned PWM waveforms (ARR=8)



Contents from RM0008 Ch-15

General-purpose timers – Encoder Interface mode

Encoder Interface

- The two inputs TI1 and TI2 can be used to interface to an incremental encoder (*Table 85*).
- The counter is clocked by each valid transition on TI1FP1 or TI2FP2 (TI1 and TI2 after input filter and polarity selection, assuming that it is enabled (CEN bit in TIMx_CR1 register written to ‘1).
- The sequence of transitions of the two inputs is evaluated and generates count pulses as well as the direction signal. Depending on the sequence the counter counts up or down, the DIR bit in the TIMx_CR1 register is modified by hardware accordingly. The DIR bit is calculated at each transition on any input (TI1 or TI2), whatever the counter is counting on TI1 only, TI2 only or both TI1 and TI2.

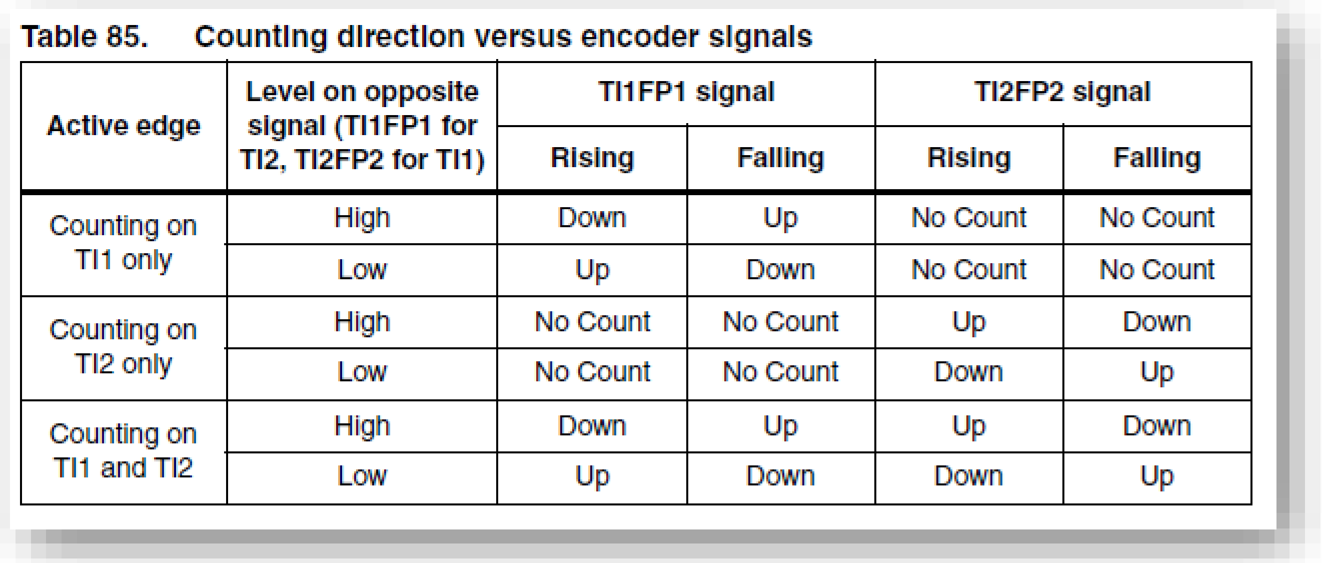


Table 85. Counting direction versus encoder signals

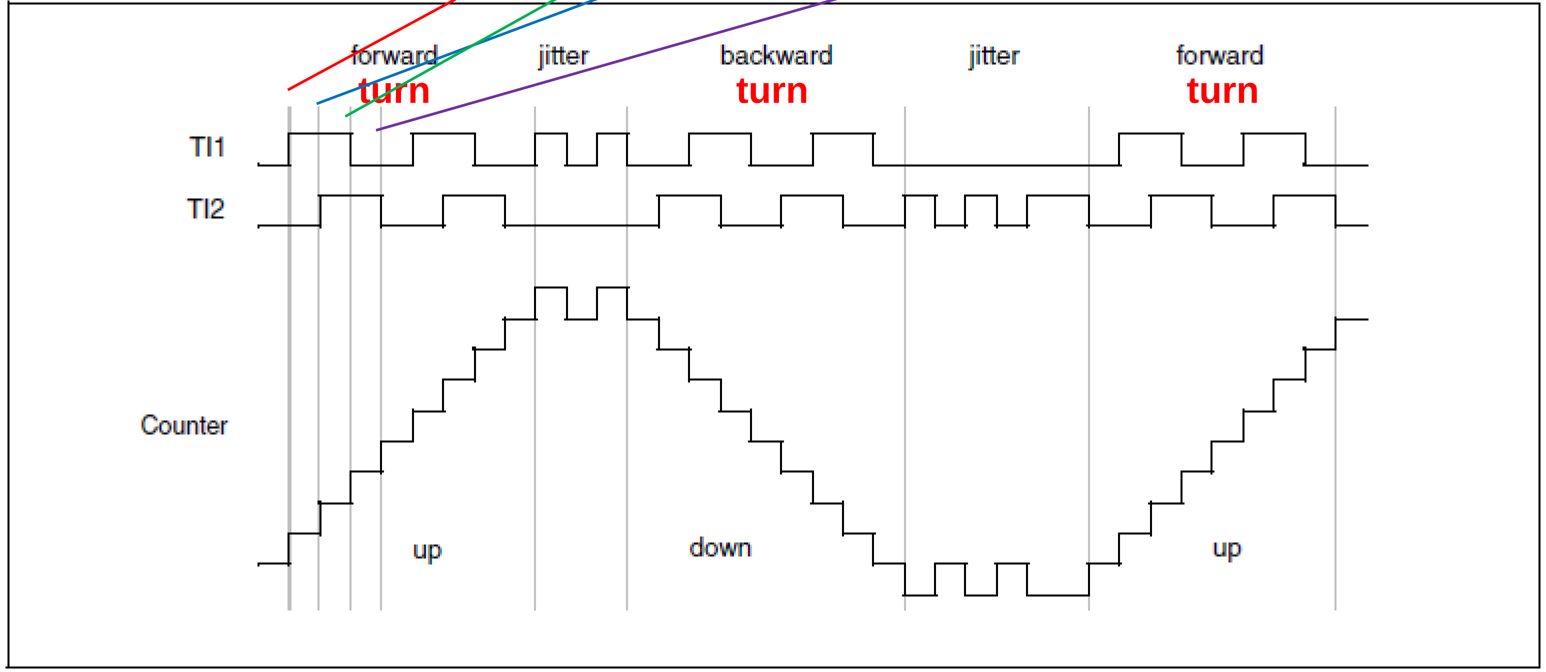
Active edge	Level on opposite signal (TI1FP1 for TI2, TI2FP2 for TI1)	TI1FP1 signal		TI2FP2 signal	
		Rising	Falling	Rising	Falling
Counting on TI1 only	High	Down	Up	No Count	No Count
	Low	Up	Down	No Count	No Count
Counting on TI2 only	High	No Count	No Count	Up	Down
	Low	No Count	No Count	Down	Up
Counting on TI1 and TI2	High	Down	Up	Up	Down
	Low	Up	Down	Down	Up

Contents from RM0008 Ch-15

Encoder Interface

Active edge	Level on opposite signal (TI1FP1 for TI2, TI2FP2 for TI1)	TI1FP1 signal		TI2FP2 signal	
		Rising	Falling	Rising	Falling
Counting on TI1 and TI2	High	Down	Up	Up	Down
	Low	Up	Down	Down	Up

Figure 134. Example of counter operation in encoder interface mode



Contents from RM0008 Ch-15

Encoder Interface

- The timer, when configured in Encoder Interface mode provides information on the sensor's current position.
- You can obtain dynamic information (<u>speed</u>, <u>acceleration</u>, <u>deceleration</u>) by measuring the period between two encoder events using a second timer configured in capture mode.
- The output of the encoder that indicates the mechanical zero can be used for this purpose.
- Depending on the time between two events, the counter can also be read at regular times. You can do this by latching the counter value into a third input capture register if available (then the capture signal must be periodic and can be generated by another timer). When available, it is also possible to read its value through a DMA request generated by a Real-Time clock.

Contents from RM0008 Ch-15

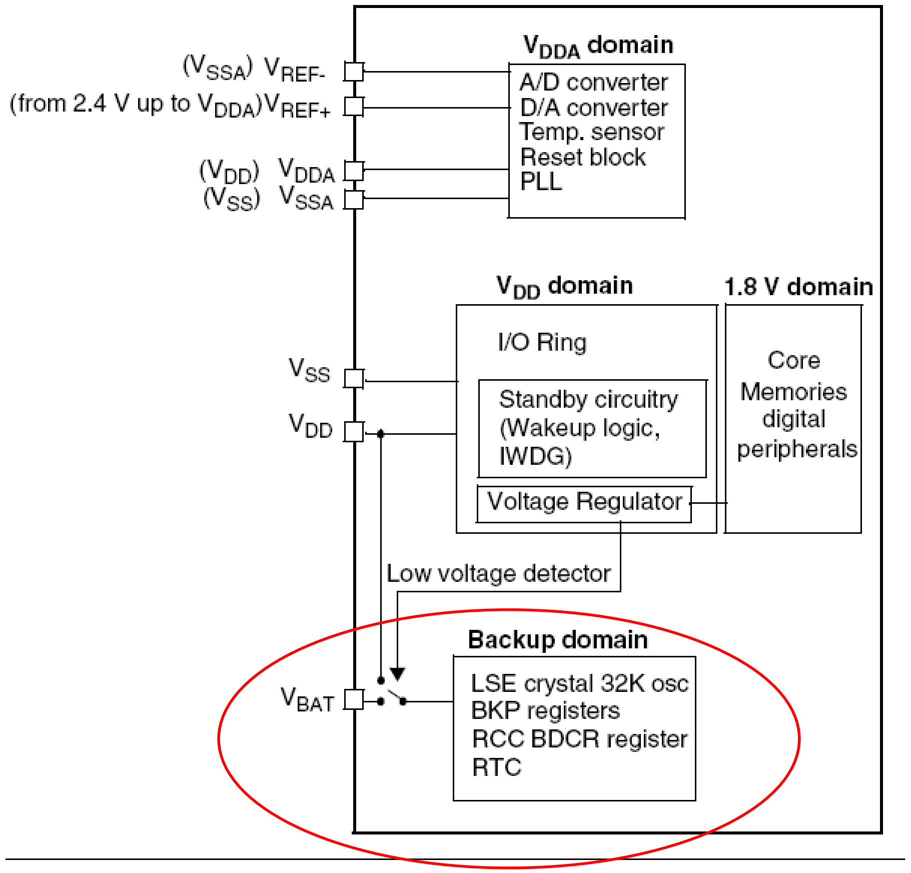
Real-time clock (RTC)

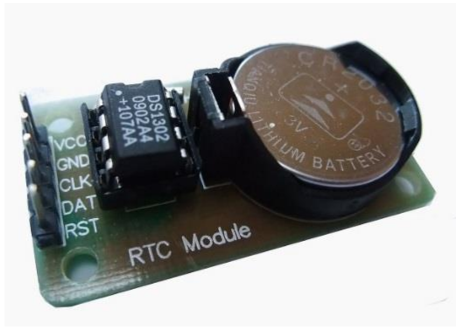
Contents from RM0008 Ch-18

Real-time Clock (RTC)

- The real-time clock is an independent timer. The RTC provides a set of continuously running counters which can be used, with suitable software, to provide a clock-calendar function.
- The counter values can be written to set the current time/date of the system.
- The RTC core and clock configuration (RCC_BDCR register) are in the Backup domain, which means that RTC setting and time are kept even after reset or wakeup from Standby mode.

Figure 4. Power supply overview





Contents from RM0008 Ch-15

Photos from www.futurlec.com & www.hobbytronics.co.uk

RTC features

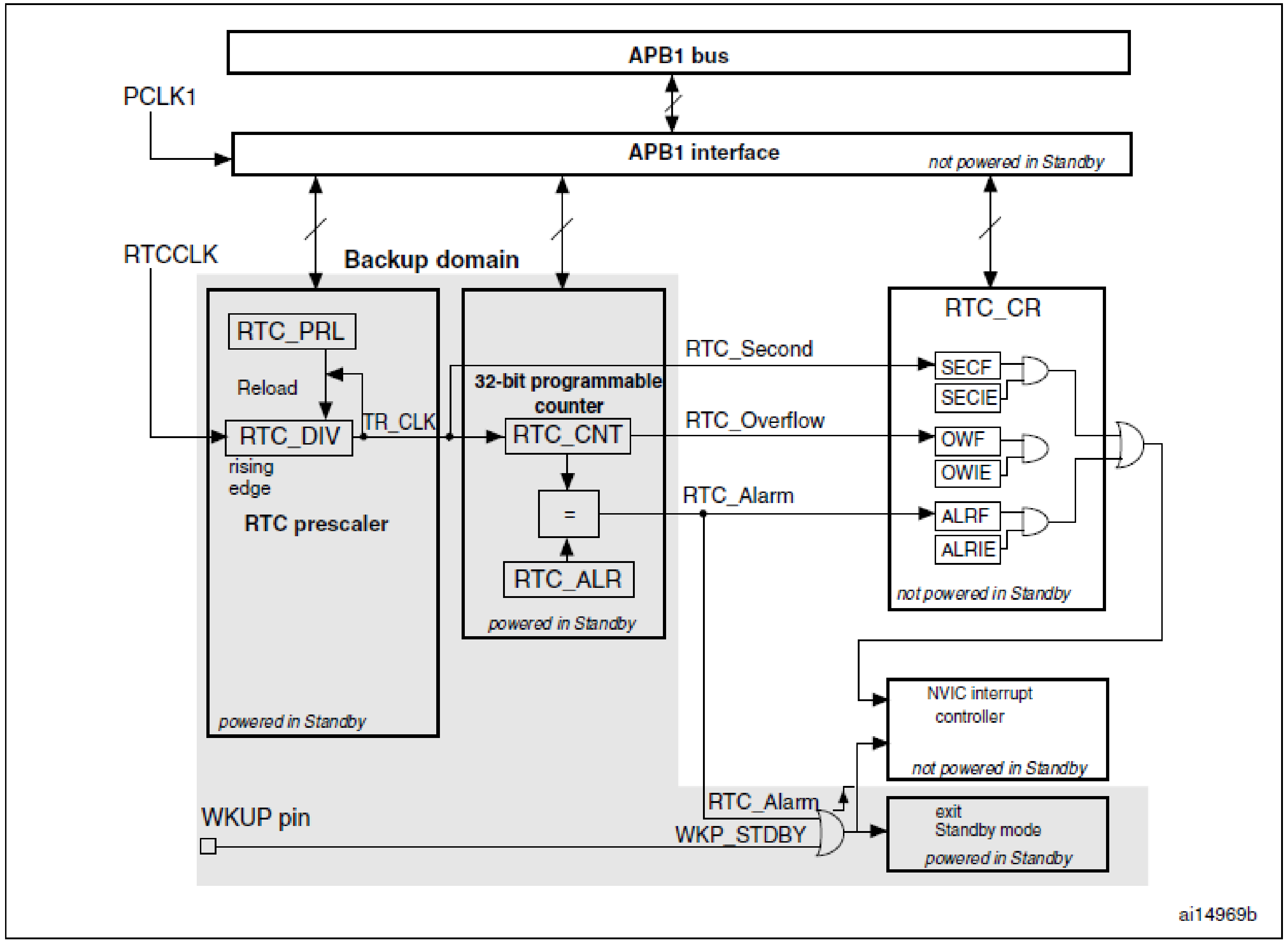
- Programmable prescaler: division factor up to 2^20
- 32-bit programmable counter for long-term measurement
- Two separate clocks: PCLK1 for the APB1 interface and RTC clock (must be at least four times slower than the PCLK1 clock)
- The RTC clock source could be any of the following ones:
  – HSE clock divided by 128
  – LSE oscillator clock
  – LSI oscillator clock (refer to *Section 7.2.8: RTC clock* for details)
- Two separate reset types:
  – The APB1 interface is reset by system reset
  – The RTC Core (Prescaler, Alarm, Counter and Divider) is reset only by a Backup domain reset (see *Section 7.1.3: Backup domain reset* on page 88).
- Three dedicated maskable interrupt lines:
  – Alarm interrupt, for generating a software programmable alarm interrupt.
  – Seconds interrupt, for generating a periodic interrupt signal with a programmable period length (up to 1 second).
  – Overflow interrupt, to detect when the internal programmable counter rolls over to zero.

Contents from RM0008 Ch-15

RTC block diagram

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

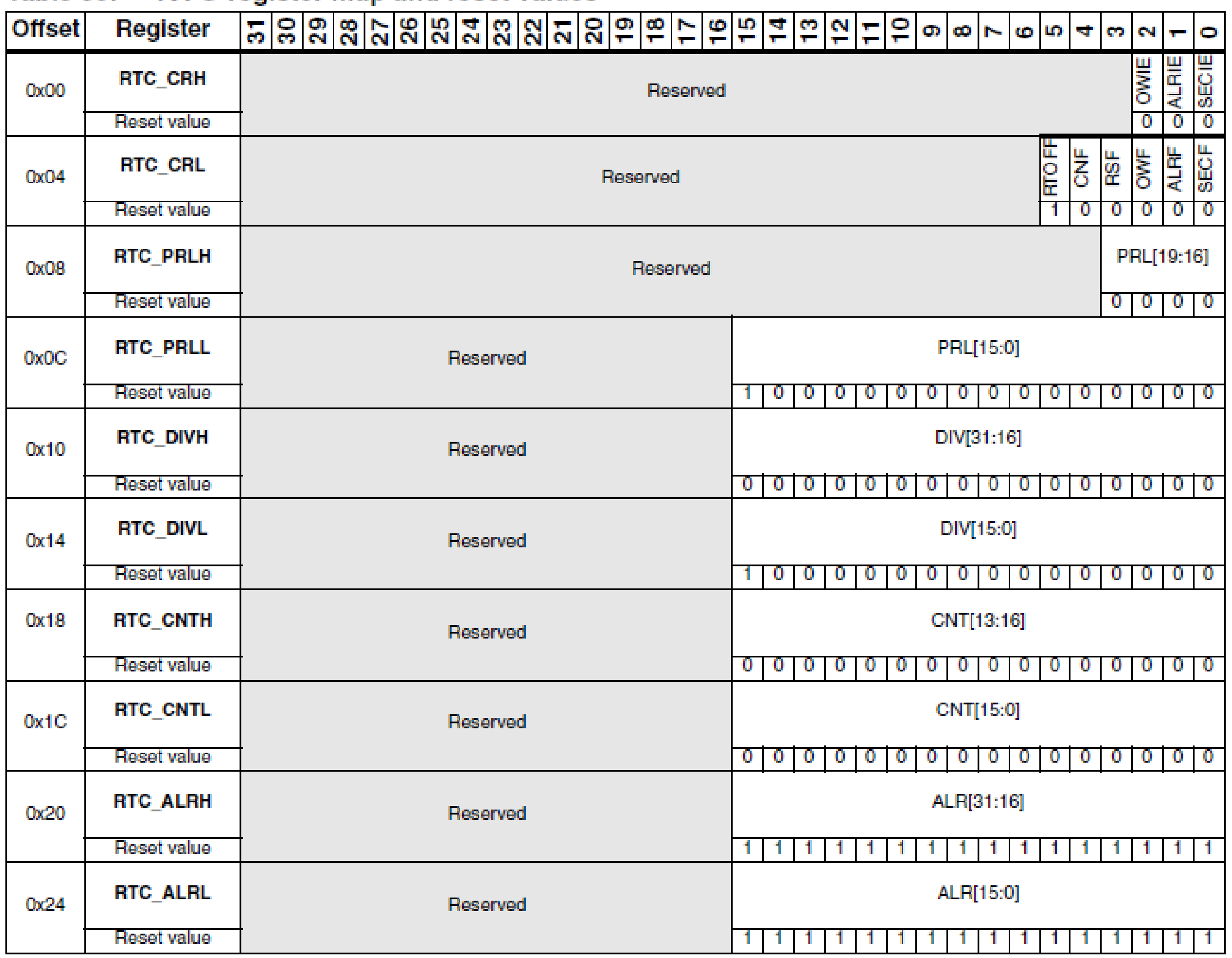
Figure 179. RTC simplified block diagram



Contents from RM0008 Ch-15

RTC registers

Table 95. RTC register map and reset values



Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	RTC_CRH	Reserved																													OWIE	ALRIE	SECIE
	Reset value																														0	0	0
0x04	RTC_CRL	Reserved																										RTOFF	CNF	RSF	OWF	ALRF	SECF
	Reset value																											1	0	0	0	0	0
0x08	RTC_PRLH	Reserved																												PRL[19:16]			
	Reset value																													0	0	0	0
0x0C	RTC_PRLL	Reserved																PRL[15:0]															
	Reset value																	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	RTC_DIVH	Reserved																DIV[31:16]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	RTC_DIVL	Reserved																DIV[15:0]															
	Reset value																	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	RTC_CNTH	Reserved																CNT[13:16]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	RTC_CNTL	Reserved																CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	RTC_ALRH	Reserved																ALR[31:16]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x24	RTC_ALRL	Reserved																ALR[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Contents from RM0008 Ch-15

Watchdog Timers

Watchdog Timers

- The STM32F10xxx have two embedded watchdog peripherals (Independent Watchdog and Window Watchdog) that offer a combination of high safety level, timing accuracy and flexibility of use.
- Both watchdog peripherals serve to detect and resolve malfunctions due to software failure, and to trigger system reset or an interrupt (window watchdog only) when the counter reaches a given timeout value.

Contents from RM0008 Ch-19

IWDG vs. WWDG

- The independent watchdog (IWDG)
  - clocked by its own dedicated low-speed clock (LSI), thus it stays active even if the main clock fails.
  - best suited to applications that require the watchdog to run as a totally independent process outside the main application, but have lower timing accuracy constraints.
- The window watchdog (WWDG)
  - clock is prescaled from the APB1 clock and has a configurable time-window that can be programmed to detect abnormally late or early application behavior.
  - best suited to applications which require the watchdog to react within an accurate timing window.

Contents from RM0008 Ch-19

IWDG main features

- A free-running downcounter;
- Clocked from an independent RC oscillator (can operate in Standby and Stop modes);
- Reset (if watchdog activated) when the downcounter value of 0x000 is reached.
- Simple to use

Contents from RM0008 Ch-19

Independent watchdog counter

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

- Two options to start IWDG:

1. Starts by writing the value 0xCCCC to the Key register (IWDG_KR); then the counter starts counting down from its reset value, 0xFFF.
2. If the "Hardware watchdog" feature is enabled through the device option bits, the watchdog is automatically enabled at power-on.

- A reset signal is generated (IWDG reset) when count=0x000.
- If a key value 0xAAAA is written in the IWDG_KR register, the value in IWDG reload register (IWDG_RLR) is reloaded in the counter and the watchdog reset is prevented.

Contents from RM0008 Ch-19

Independent watchdog

Figure 182. Independent watchdog block diagram

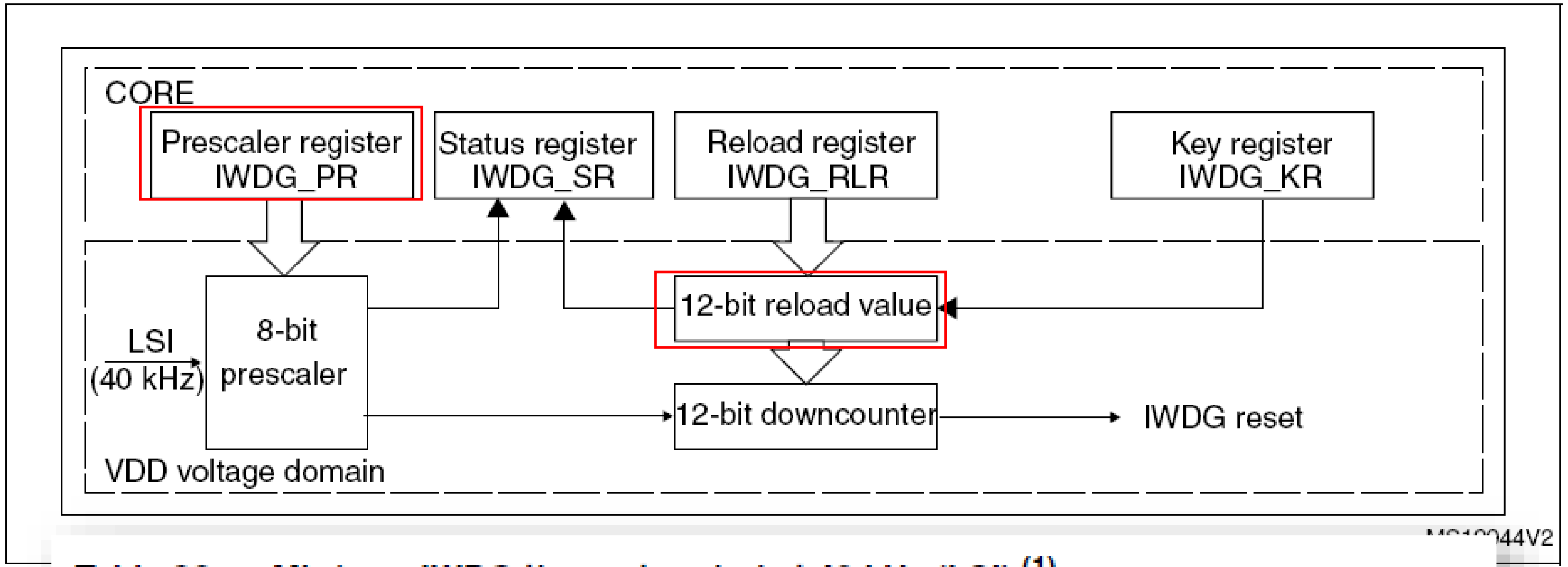


Table 96. Min/max IWDG timeout period at 40 kHz (LSI) (1)

Prescaler divider	PR[2:0] bits	Min timeout (ms) RL[11:0]= 0x000	Max timeout (ms) RL[11:0]= 0xFFF
/4	0	0.1	409.6
/8	1	0.2	819.2
/16	2	0.4	1638.4
/32	3	0.8	3276.8
/64	4	1.6	6553.6

Contents from RM0008 Ch-19

Independent watchdog's registers

Register access protection

- Write access to the IWDG_PR and IWDG_RLR registers is protected. To modify them, you must first write the code 0x5555 in the IWDG_KR register. A write access to this register with a different value will break the sequence and register access will be protected again.
- This implies that it is the case of the reload operation (writing 0xAAAA).
- A status register (IWDG_SR) is available to indicate that an update of the prescaler or the down-counter reload value is on going.

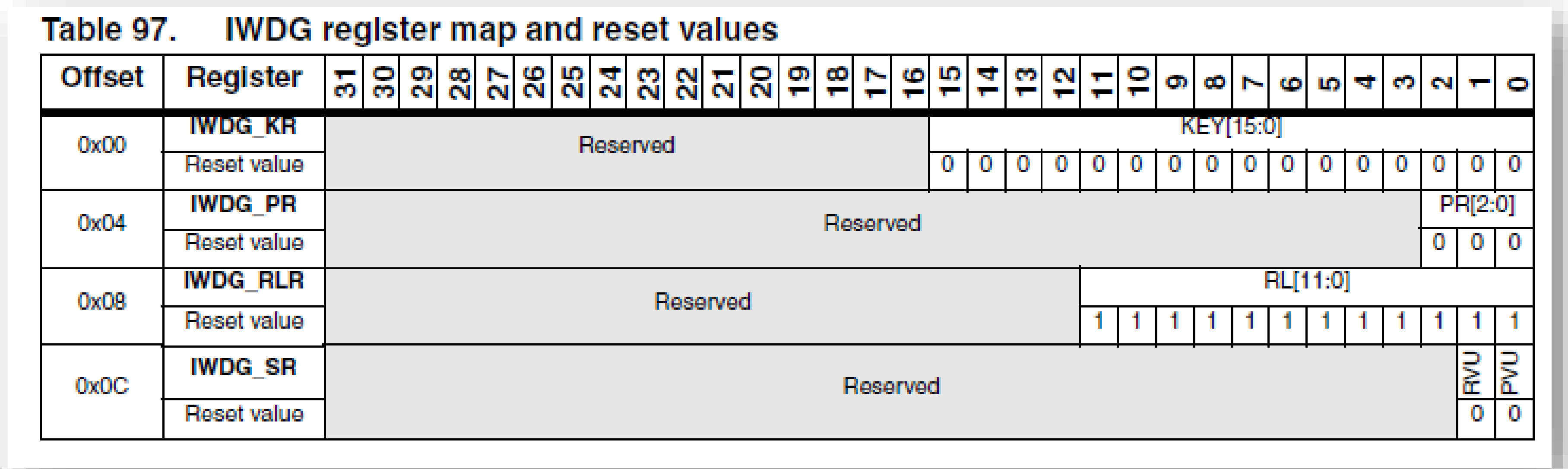


Table 97. IWDG register map and reset values

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	IWDG_KR	Reserved																KEY[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	IWDG_PR	Reserved																													PR[2:0]		
	Reset value																														0	0	0
0x08	IWDG_RLR	Reserved																				RL[11:0]											
	Reset value																					1	1	1	1	1	1	1	1	1	1	1	1
0x0C	IWDG_SR	Reserved																														RVU	PVU
	Reset value																															0	0

Contents from RM0008 Ch-15

Window watchdog (WWDG)

香 港 中 文 大 學 (深 圳)
The Chinese University of Hong Kong, Shenzhen

- The window watchdog is used to detect the occurrence of a software fault (due to external interference or unforeseen logical conditions), which causes the application program to abandon its normal sequence.
- An MCU reset is generated on expiry of a programmed time period, unless the program refreshes the contents of the downcounter before the T6 bit (of the control register, WWDG_CR register) becomes cleared.
- An MCU reset is also generated if the 7-bit downcounter value (in the control register) is refreshed before the downcounter has reached the window register value (in configuration register WWDG_CFR register). This implies that the counter must be refreshed in a limited window.

Contents from RM0008 Ch-20

WWDG main features

- Programmable free-running 7-bit downcounter
- Conditional reset (if watchdog activated)
  - when the downcounter value is < 0x40
  - if the downcounter is reloaded outside the window (window is set in WWDG_CFR register)
- Early wakeup interrupt (EWI): triggered (if enabled and the watchdog activated) when the downcounter is equal to 0x40.

Contents from RM0008 Ch-20

Window watchdog

- If the watchdog is activated (the WDGA bit is set in the WWDG_CR register) and when the 7-bit downcounter (T[6:0] bits) rolls over from 0x40 to 0x3F (T6 becomes cleared), it initiates a reset.
- If the software reloads the counter while the counter is greater than the value stored in the window register (W[6:0]), then a reset is generated.
- To prevent reset: (1) The application program must write in the WWDG_CR register at regular intervals during normal operation to prevent an MCU reset.
  (2) The write operation must occur only when the counter value is lower than the window register value (W[6:0]).

Advanced watchdog interrupt feature

- The Early Wakeup Interrupt (EWI) can be used if specific safety operations or data logging must be performed before the actual reset is generated. The EWI interrupt is enabled by setting the EWI bit in the WWDG_CFR register.
- When the downcounter reaches the value 0x40, an EWI interrupt is generated and the corresponding interrupt service routine (ISR) can be used to trigger specific actions (such as communications or data logging), before resetting the device.
- The last chance to prevent WWDG reset: In some applications, the EWI interrupt can be used to manage a software system check and/or system recovery/graceful degradation, <u>without generating a WWDG reset</u>. In this case, the corresponding interrupt service routine (ISR) should reload the WWDG counter to avoid the WWDG reset, then trigger the required actions. (So the ISR shall happen within the time that counter changes from 0x40 to 0x3F)

Contents from RM0008 Ch-20

SysTick and multi-tasking

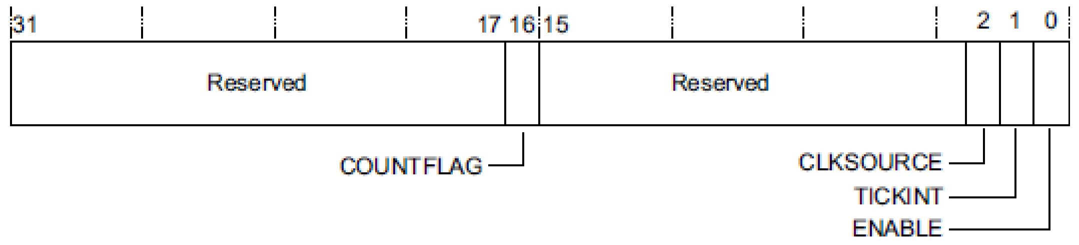
SYSTICK

- The SYSTICK Timer is integrated with the NVIC and can be used to generate a SYSTICK exception (exception type #15).
- In many operating systems, a hardware timer is used to generate interrupts so that the OS can carry out task management—for example, to allow multitasking. And if possible, it should make sure that user applications cannot change the timer behavior.
- The Cortex-M3 processor includes a simple 24-bit down counter. It can use the internal free running processor clock signal on the Cortex-M3 processor or an external reference clock (documented as the STCLK signal on the Cortex-M3 TRM). The actual implementations may vary for different chip manufactures.
- The SYSTICK Timer is controlled by four registers (Tables 8.9–8.12.):
  - SYSTICK CONTROL AND STATUS register
  - SYSTICK RELOAD register
  - SYSTICK CURRENT VALUE register
  - SYSTICK CALIBRATION register

Contents from text book

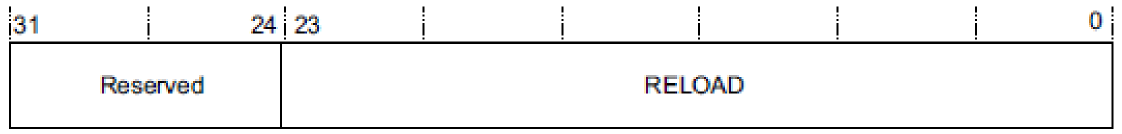
SysTick Register

- SysTick Control and Status Register



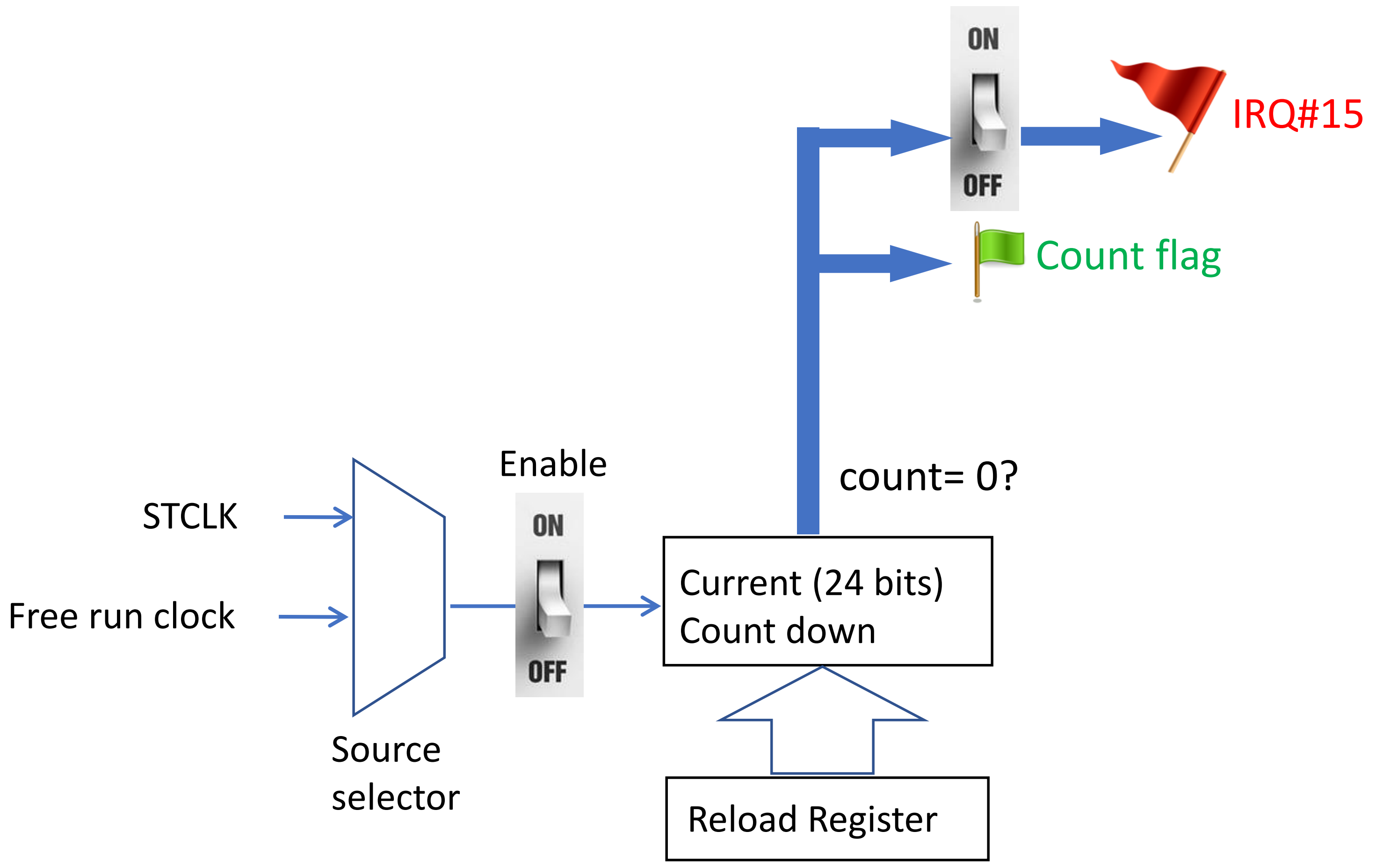
Bits	Field	Function
[31:17]	-	Reserved.
[16]	COUNTFLAG	Returns 1 if timer counted to 0 since last time this was read. Clears on read by application of any part of the SysTick Control and Status Register. If read by the debugger using the DAP, this bit is cleared on read-only if the MasterType bit in the AHB-AP Control Register is set to 0. Otherwise, the COUNTFLAG bit is not changed by the debugger read.
[2]	CLKSOURCE	0 = external reference clock. 1 = core clock. If no reference clock is provided, it is held at 1 and so gives the same time as the core clock. The core clock must be at least 2.5 times faster than the reference clock. If it is not, the count values are Unpredictable.
[1]	TICKINT	1 = counting down to 0 pends the SysTick handler. 0 = counting down to 0 does not pend the SysTick handler. Software can use the COUNTFLAG to determine if ever counted to 0.
[0]	ENABLE	1 = counter operates in a multi-shot way. That is, counter loads with the Reload value and then begins counting down. On reaching 0, it sets the COUNTFLAG to 1 and optionally pends the SysTick handler, based on TICKINT. It then loads the Reload value again, and begins counting. 0 = counter disabled.

- SysTick Reload Value Register



Bits	Field	Function
[31:24]	-	Reserved
[23:0]	RELOAD	Value to load into the SysTick Current Value Register when the counter reaches 0.

Systick architecture



Calibration Value register (0xE000E01C)

- The Calibration Value register provides a solution for applications to generate the same SYSTICK interrupt interval when running on various Cortex-M3 products.
- To use it, just write the value in TENMS to the reload value register. This will give an interrupt interval of about 10 ms. For other interrupt timing intervals, the software code will need to calculate a new suitable value from the calibration value.
- However, the TENMS field might not be available in all Cortex-M3 products
- In addition, the SYSTICK Timer can be used as an alarm timer, for timing measurement, and others.

SYSTICK registers

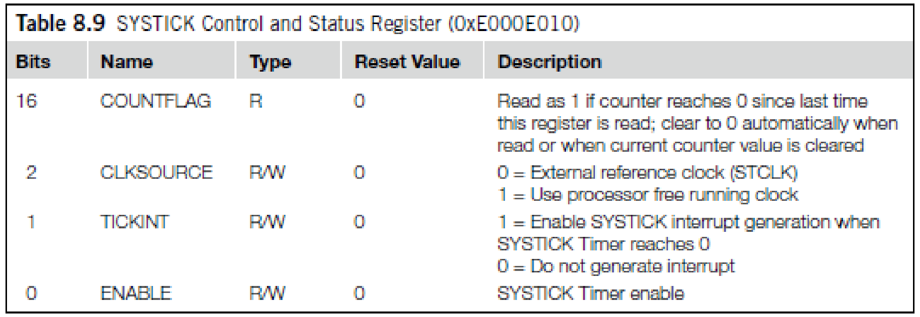


Table 8.9 SYSTICK Control and Status Register (0xE000E010)

Bits	Name	Type	Reset Value	Description
16	COUNTFLAG	R	0	Read as 1 if counter reaches 0 since last time this register is read; clear to 0 automatically when read or when current counter value is cleared
2	CLKSOURCE	R/W	0	0 = External reference clock (STCLK) 1 = Use processor free running clock
1	TICKINT	R/W	0	1 = Enable SYSTICK interrupt generation when SYSTICK Timer reaches 0 0 = Do not generate interrupt
0	ENABLE	R/W	0	SYSTICK Timer enable

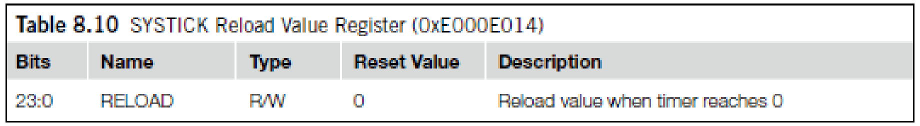


Table 8.10 SYSTICK Reload Value Register (0xE000E014)

Bits	Name	Type	Reset Value	Description
23:0	RELOAD	R/W	0	Reload value when timer reaches 0

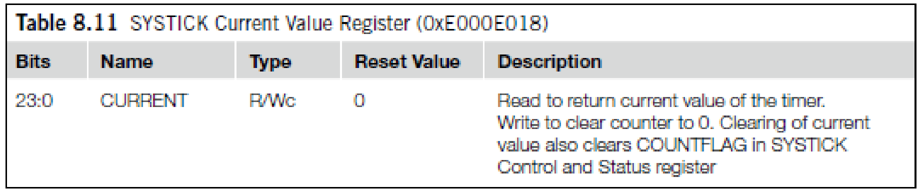


Table 8.11 SYSTICK Current Value Register (0xE000E018)

Bits	Name	Type	Reset Value	Description
23:0	CURRENT	R/Wc	0	Read to return current value of the timer. Write to clear counter to 0. Clearing of current value also clears COUNTFLAG in SYSTICK Control and Status register

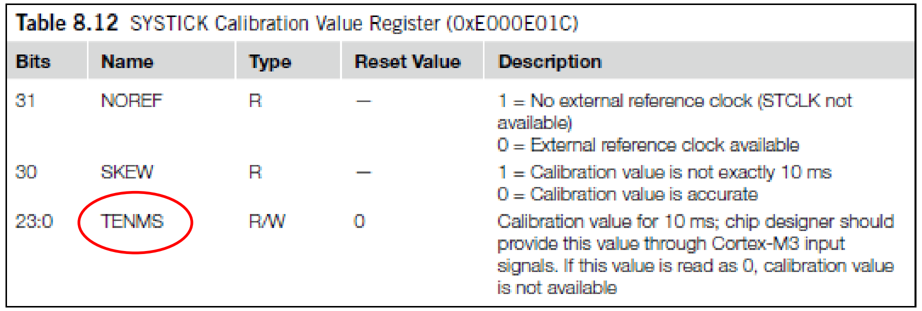


Table 8.12 SYSTICK Calibration Value Register (0xE000E01C)

Bits	Name	Type	Reset Value	Description
31	NOREF	R	—	1 = No external reference clock (STCLK not available) 0 = External reference clock available
30	SKEW	R	—	1 = Calibration value is not exactly 10 ms 0 = Calibration value is accurate
23:0	TENMS	R/W	0	Calibration value for 10 ms; chip designer should provide this value through Cortex-M3 input signals. If this value is read as 0, calibration value is not available

Contents from text book

End