Embedded Systems with ARM Cortex-M Microcontrollers in Assembly Language and C

Chapter 16
General Purpose Timer and PWM

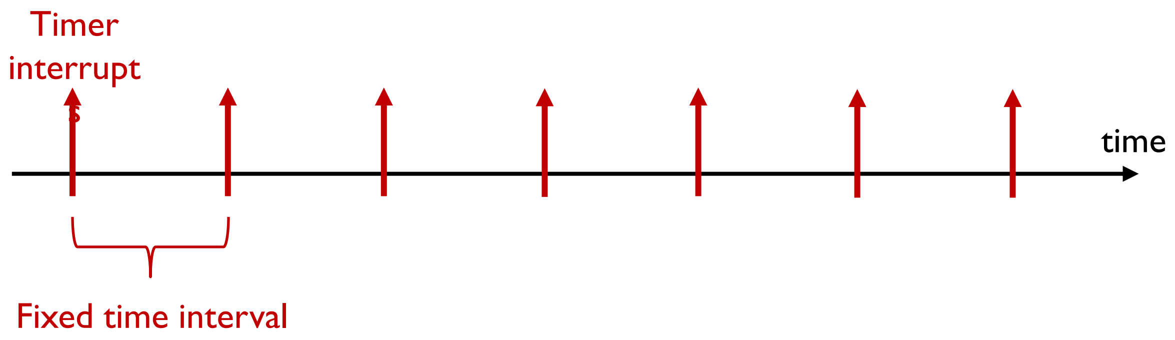
Z. Gu

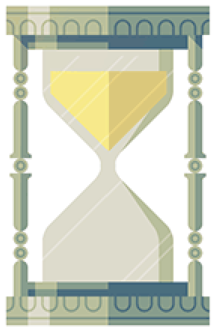
Fall 2025

Acknowledgement: Lecture slides based on Embedded Systems with ARM Cortex-M Microcontrollers in Assembly Language and C, University of Maine https://web.eece.maine.edu/~zhu/book/

Timer

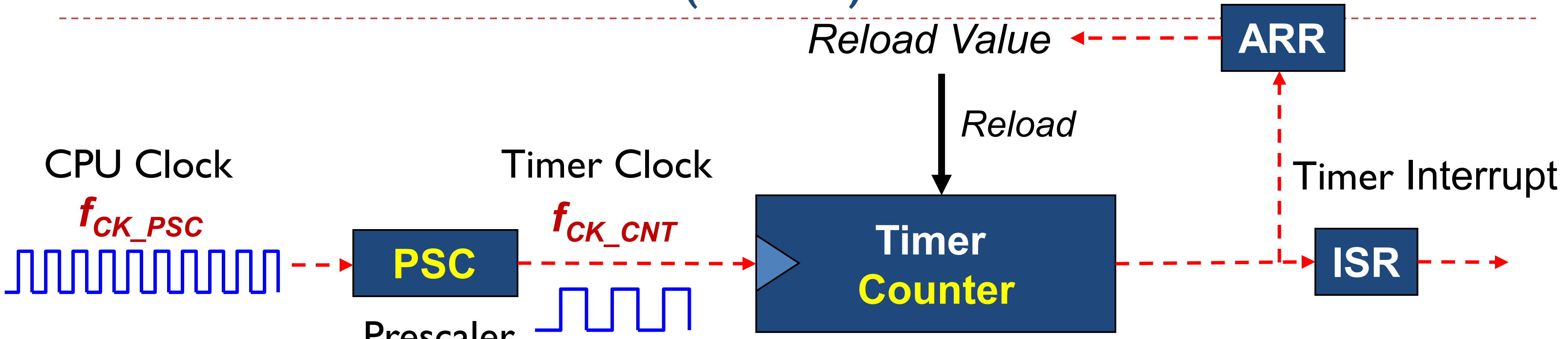
- A timer is a hardware module that generates periodic clock ticks at a fixed time interval





- Example Usages:
  - To measure time elapsed, e.g., to implement a time delay function Delay (100ms).
  - To implement periodic polling to check peripheral device status, or to implement CPU scheduler, which periodically selects a new process, from the ready queue, to run next.

Timer: Prescaler (PSC)



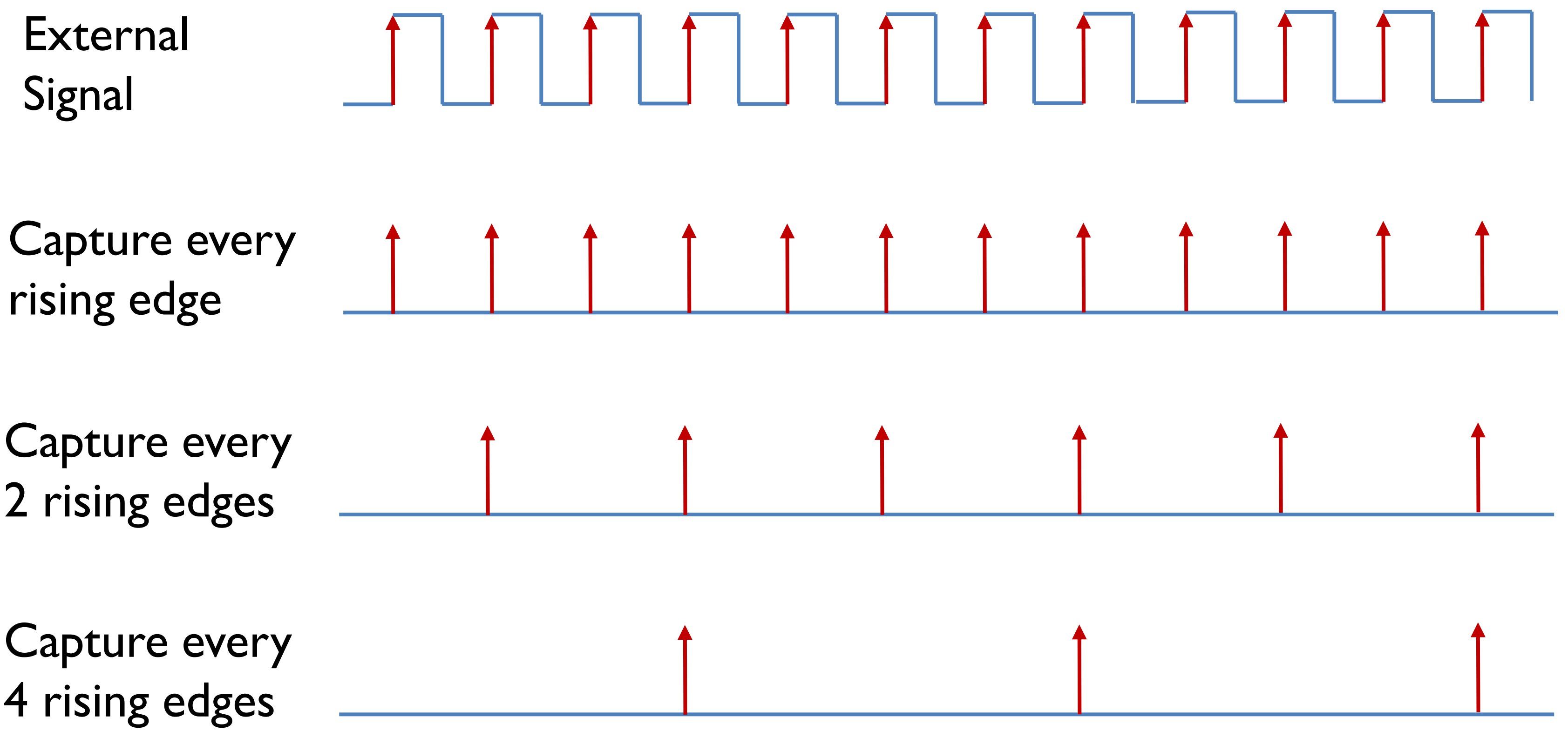
$$f_{CK\_CNT} = \frac{f_{CK\_PSC}}{PSC + 1}$$

$$f_{\text{Timer}} = \frac{f_{CK\_CNT}}{ARR+1} \text{ or } \frac{f_{CK\_CNT}}{2*ARR}$$

- Software can change the prescaler (PSC) to allow the timer clocked at desired rate. PSC serves as a "frequency divider" that enables tradeoffs between timer resolution and timer range. Larger PSC value leads to slower timer clock rate, coarser timer resolution, and larger timer range.
- The 16-bit PSC register holds the frequency pre-scaler value in the range of [0, 2^{16}-1=65535]. The CPU Clock Frequency f_{CK_PSC} is divided by a constant integer, PSC+1, to generate the Timer Clock with frequency f_{CK_CNT} that drives the Timer Counter.
  - e.g., $PSC = 15999, f_{CK\_CNT} = \frac{16\ MHz}{15999+1} = 1\ KHz$
- The 16-bit Auto-Reload Register (ARR) holds the maximum timer counter value.
- CPU clock and timer clock: measured in GHz/MHz; Timer interrupt frequency: measured in Hz, e.g., the Linux OS timer interrupt frequency is 100Hz (period=10ms). Since each timer interrupt triggers an ISR software execution, timer interrupt frequency should not be higher than 1KHz.

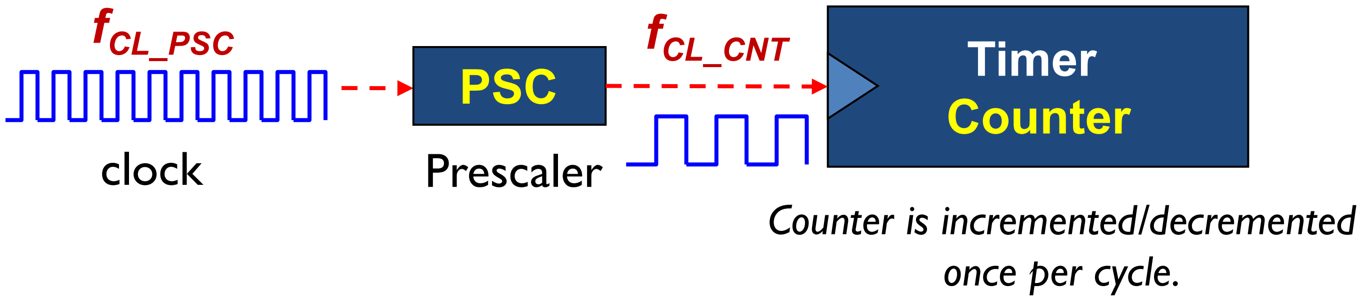
Prescaler

- Assume capture only rising edge


External Signal
Capture every rising edge
Capture every 2 rising edges
Capture every 4 rising edges

Example

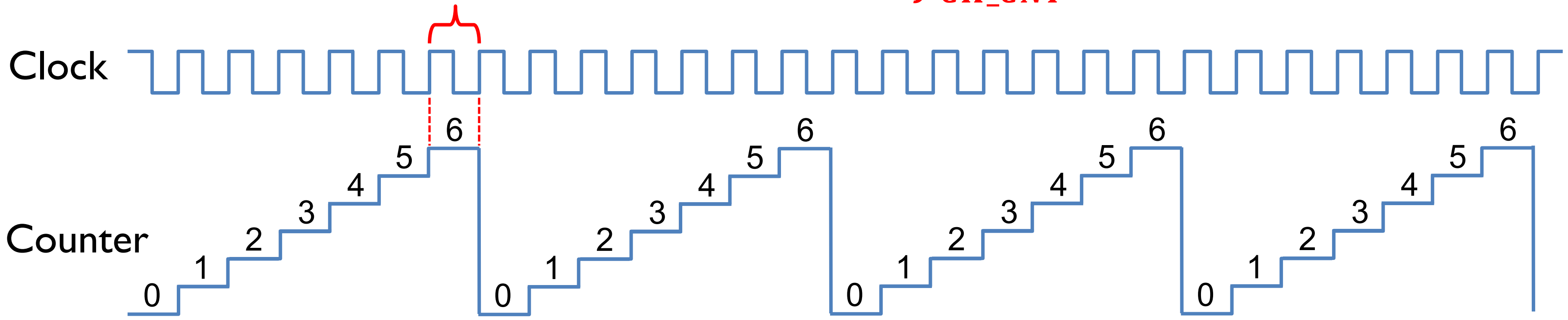
$$f_{CK\_CNT} = \frac{f_{CL\_PSC}}{PSC + 1}$$



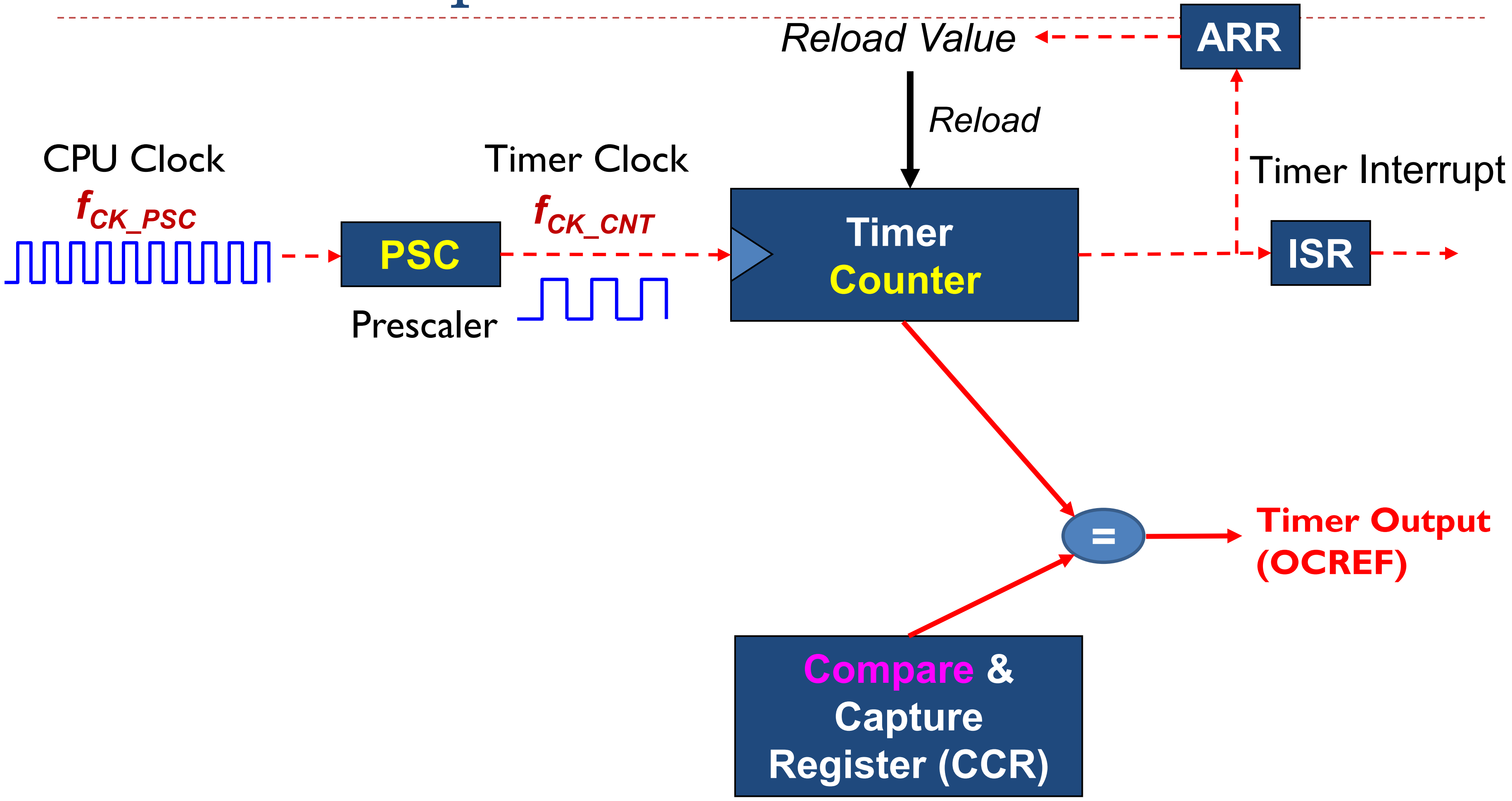
Counter is incremented/decremented once per cycle.

Timer: Upcounting mode, ARR = 6

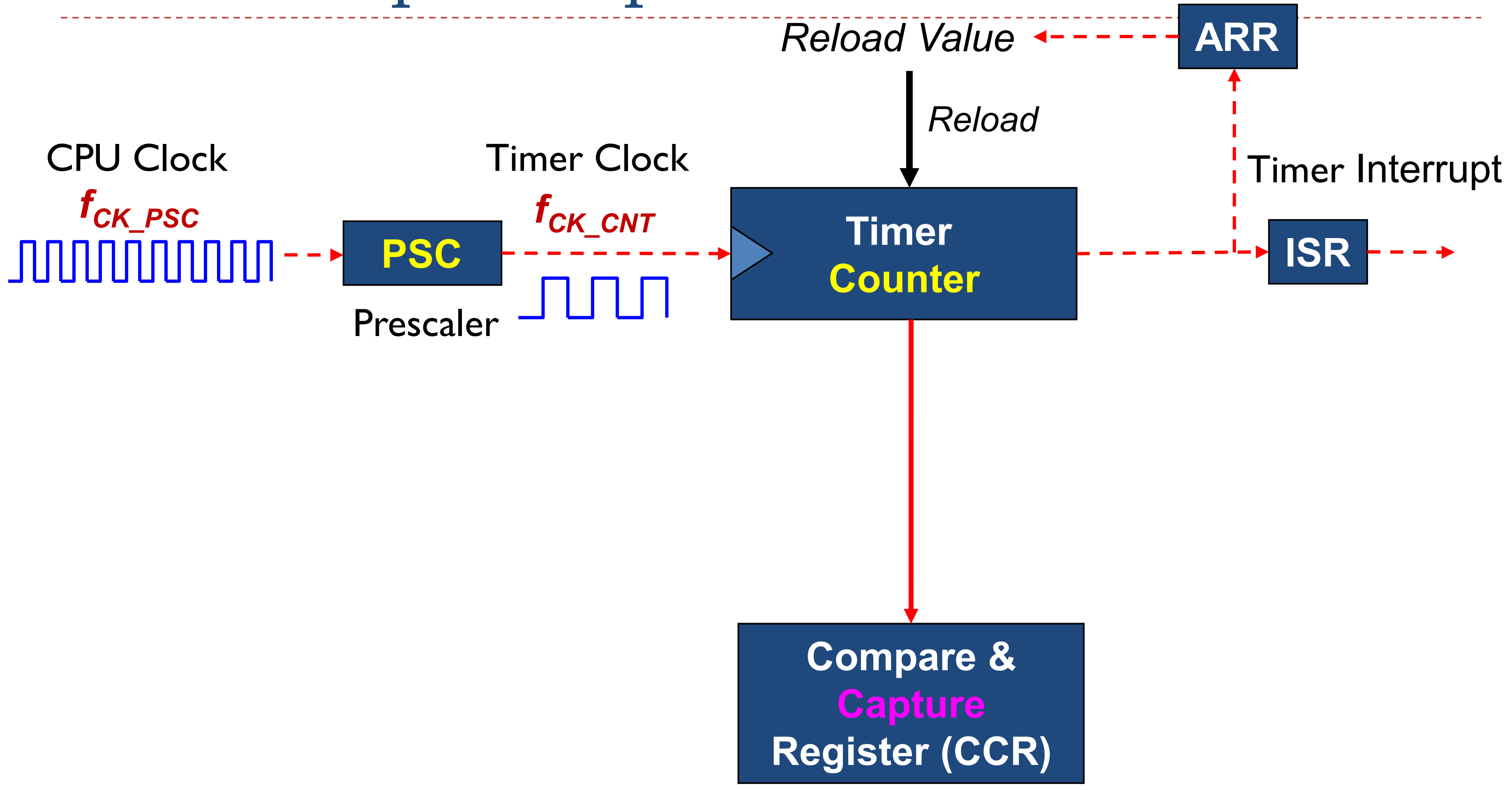
$$\boldsymbol{counter\ clock\ period = \frac{1}{f_{CK\_CNT}}}$$



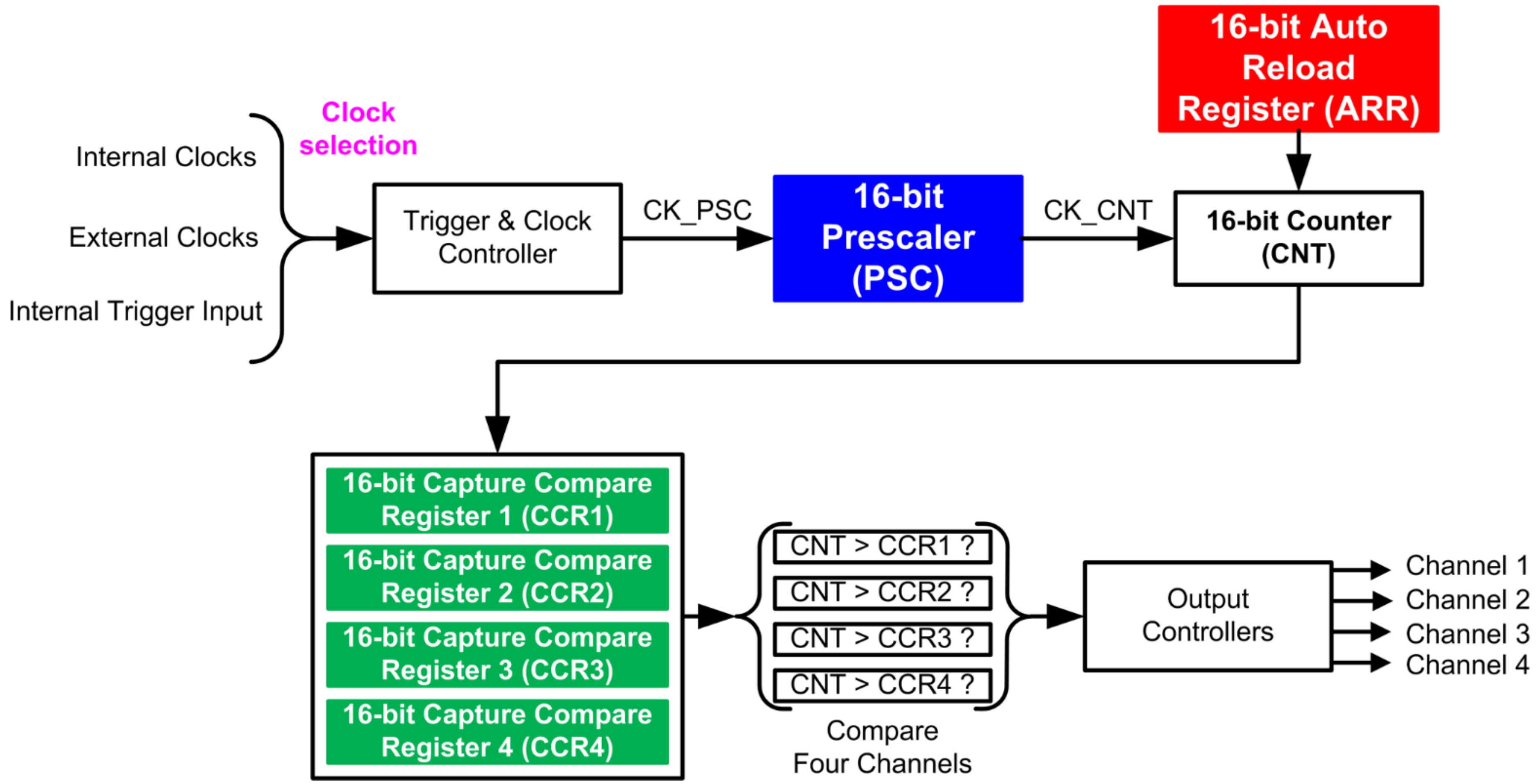
Timer: Output



Timer: Input Capture

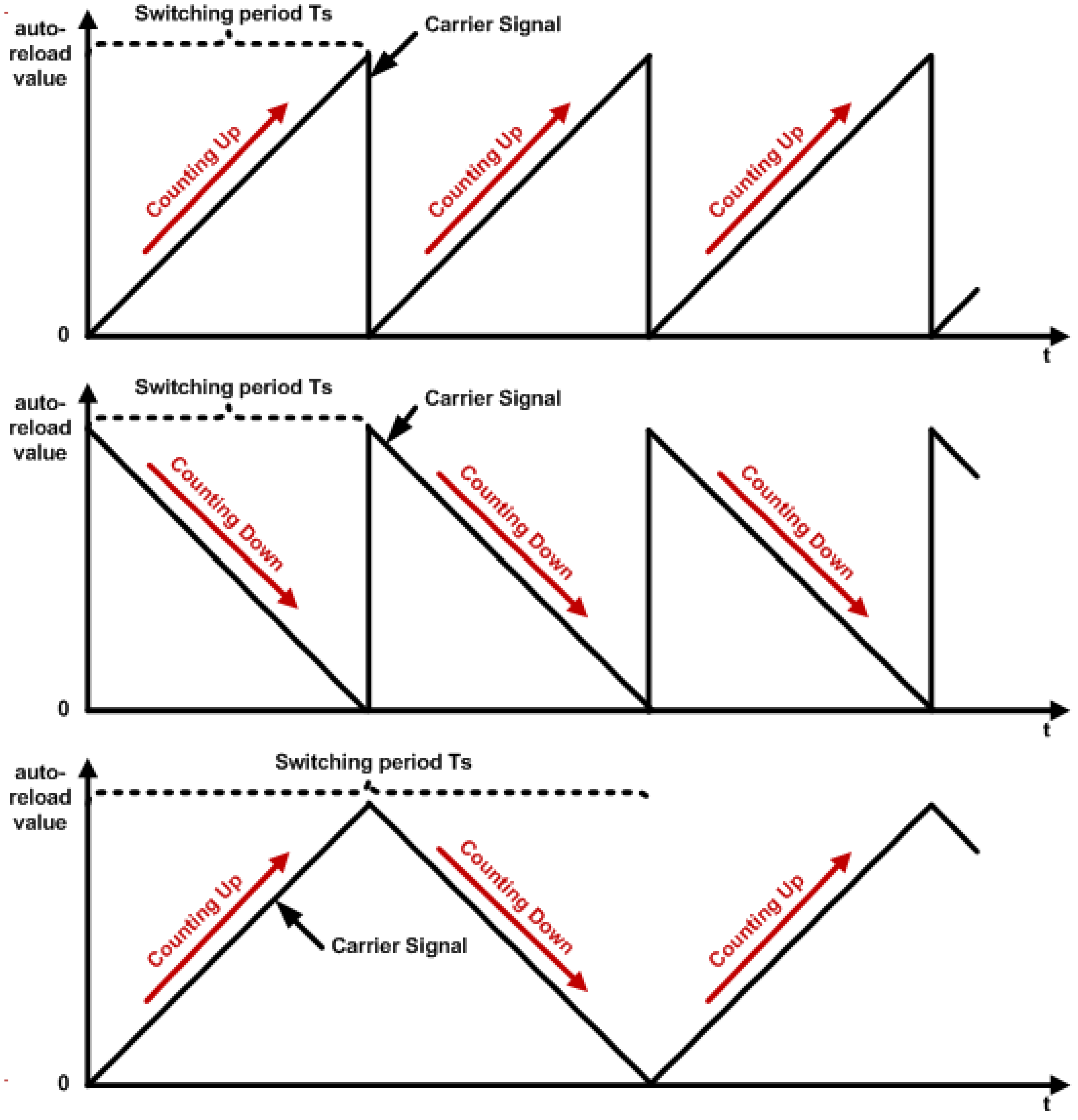


Multi-Channel Outputs



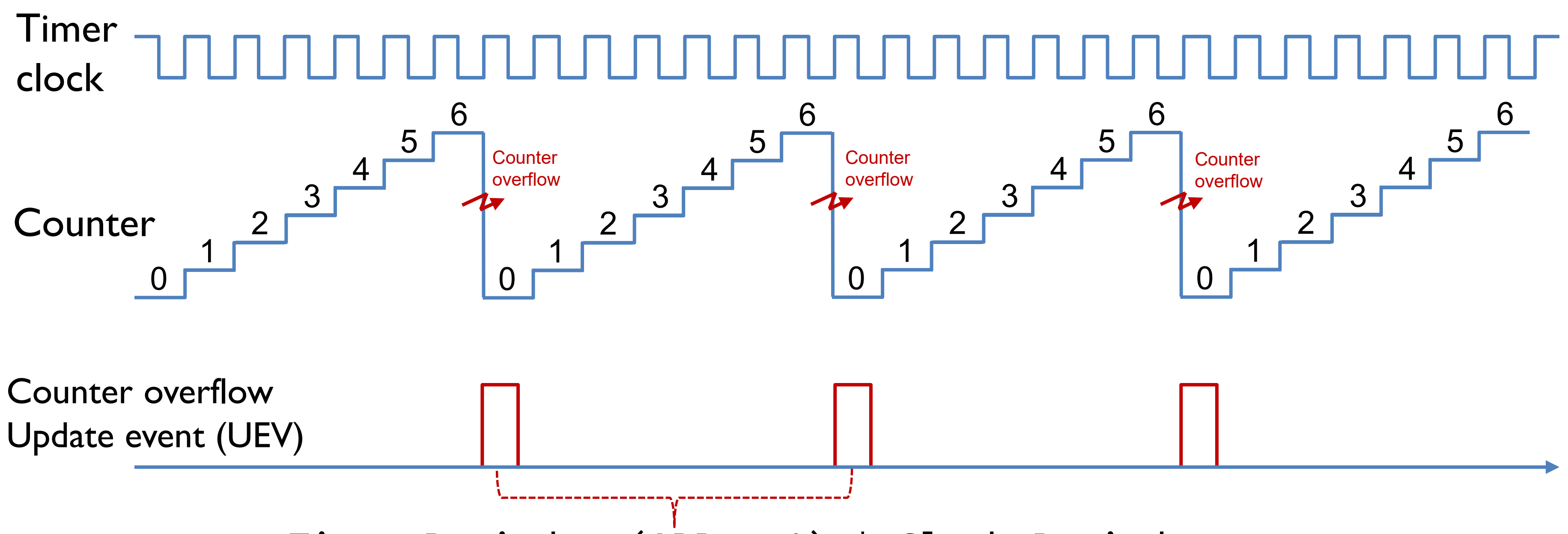
3 Modes

- A general-purpose timer can be configured in 3 modes:
  - Up-counting mode
  - Down-counting mode
  - Center-aligned counting mode (Alternating up-counting and down-counting)
- Related registers include:
  - Counter register (TIMx_CNT)
  - Prescaler register (TIMx_PSC)
  - Auto-reload register (TIMx_ARR)
  - Their values are denoted as CNT, PSC and ARR.
- SysTick timer can be down-counting mode only.



Up-counting Mode

ARR = 6

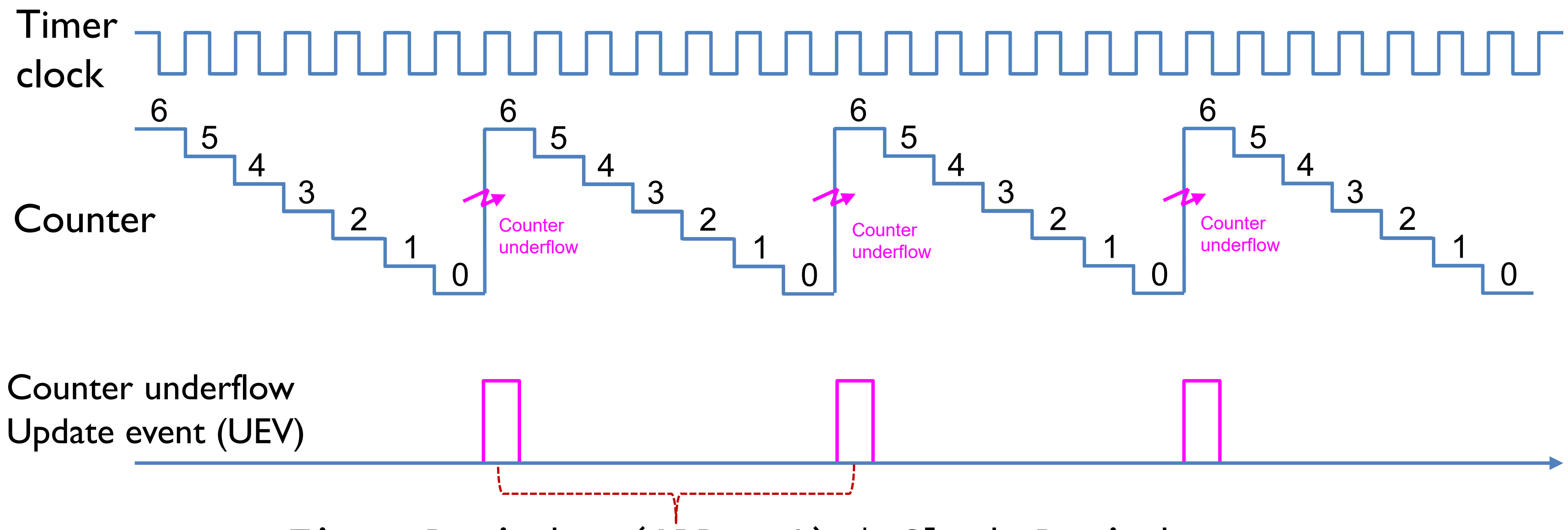


```
Timer Period = (ARR + 1) * Clock Period
             = 7 * Clock Period
```

Auto-Reload Register ARR=6. In up-counting mode, the counter counts up, from 0 to 6. When the counter is reset to 0 from 6, counter overflow occurs, and a Update event (UEV) is generated. On the next clock cycle, the counter counts up again. The Timer Period is 1+ARR clock ticks, with duration of (1+ARR)*Clock Period.

Down-counting Mode

ARR = 6

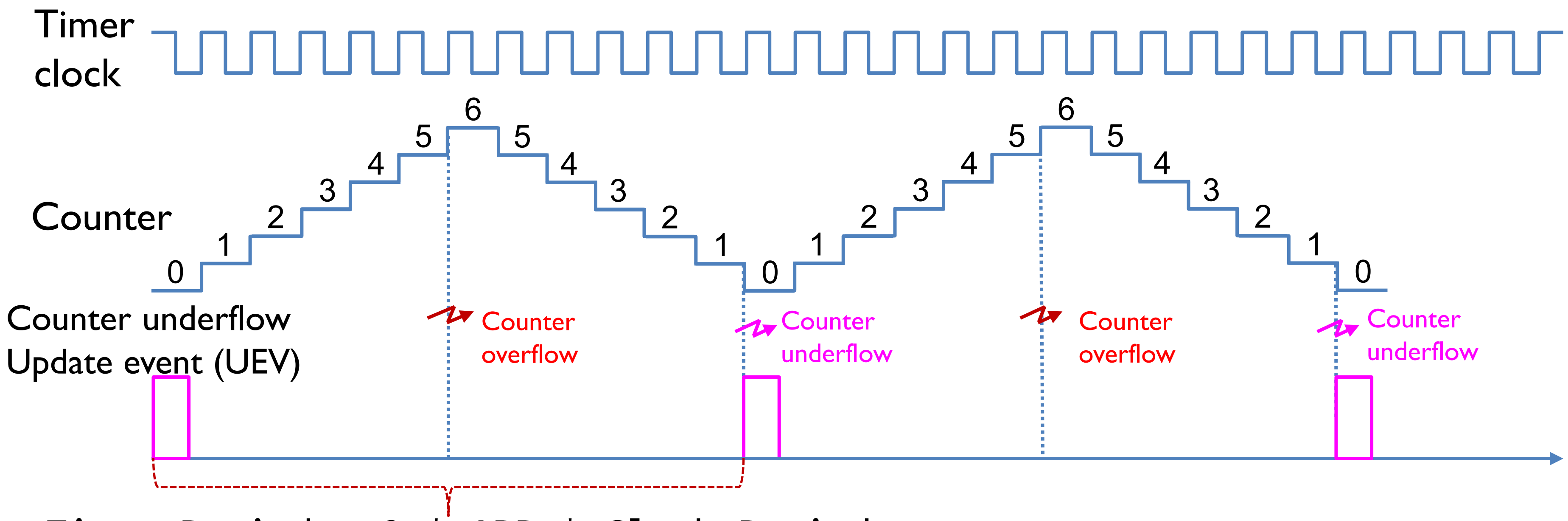


```
Timer Period = (ARR + 1) * Clock Period
             = 7 * Clock Period
```

ARR=6. In down-counting mode, the counter counts down, from 6 to 0. When the counter is reset to 6 from 0, counter underflow occurs, and a UEV is generated. On the next clock cycle, the counts down again. The Timer Period is 1+ARR clock ticks, with duration of (1+ARR)*Clock Period.

Center-aligned Counting Mode

ARR = 6



```
Timer Period = 2 * ARR * Clock Period
             = 12 * Clock Period
```

ARR=6. In center-aligned counting mode, the counter first counts up, from 0 to 6, then counts down, from 6, to 0. When the counter makes transition from 1 to 0, a UEV (Update Event) is generated. On the next clock cycle, the counter repeats the up/down counting process. The Timer Period is 2*ARR clock ticks, with duration of 2*ARR*Clock Period.

Example: Calculating ARR

- Suppose Timer Clock Frequency = 80MHz
- Goal: Timer Frequency = 100Hz
- What should the ARR value be for 1. up-counting mode; 2. down-counting mode; 3. center-aligned counting mode?

For up-counting or down-counting mode:

$$f_{Timer} = \frac{f_{CK\_CNT}}{ARR+1} = \frac{80MHz}{ARR+1} = 100Hz$$

$$ARR = 799999$$

For center-aligned counting mode:

$$f_{Timer} = \frac{f_{CK\_CNT}}{2 * ARR} = \frac{80MHz}{2 * ARR} = 100Hz$$

$$ARR = 400000$$

But a 16-bit ARR register has value range of [0, 65535], so ARR value of 799999 or 400000 is out of range → cannot generate a 10ms timer from a 80MHz Timer Clock!
Solution: use prescaler (PSC) to reduce Timer Clock Frequency

Example: Calculating ARR with Prescaler

- Suppose CPU Clock Frequency = 80MHz， prescaler PSC=79
- Goal: Timer Frequency = 100Hz
- What should the ARR value be for 1. up-counting mode; 2. down-counting mode; 3. center-aligned counting mode?

$$Timer\ Clock\ Freq\ f_{CK\_CNT} = \frac{f_{CK_{PSC}}}{PSC+1} = \frac{80MHz}{80} = 1MHz$$

For up-counting or down-counting mode:

$$f_{Timer} = \frac{f_{CK\_CNT}}{ARR+1} = \frac{1MHz}{ARR+1} = 100Hz$$

$$ARR = 9999$$

For center-aligned counting mode:

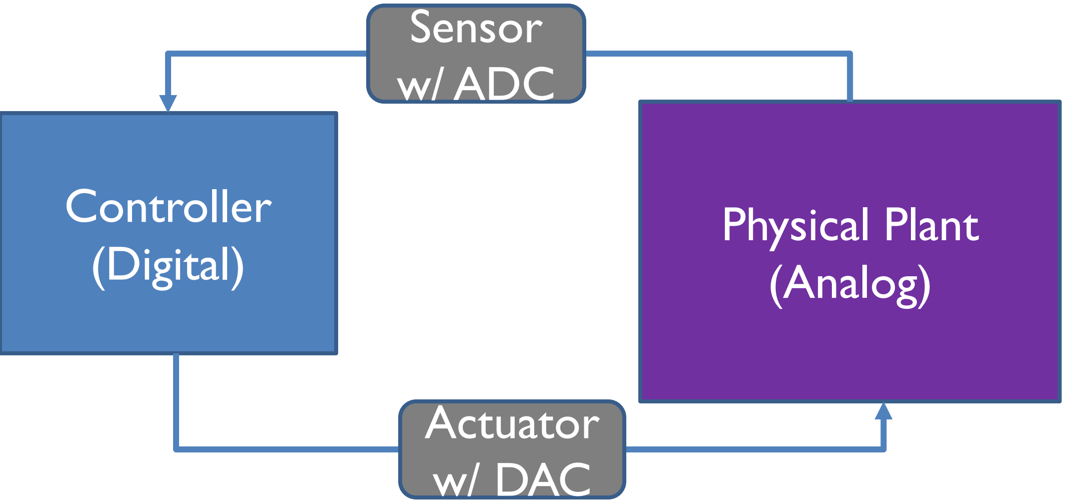
$$f_{Timer} = \frac{f_{CK\_CNT}}{2*ARR} = \frac{1MHz}{2*ARR} = 100Hz$$

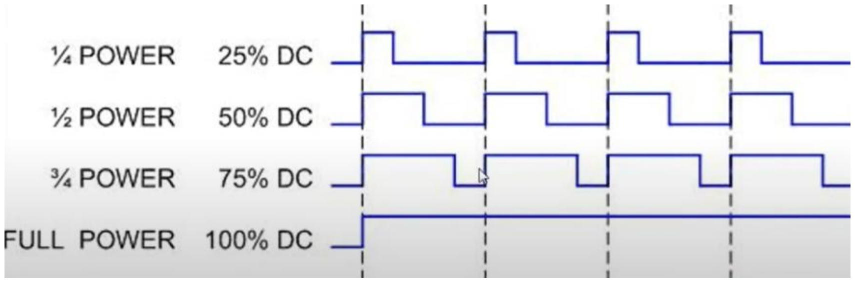
$$ARR = 5000$$

With prescaler, the ARR value of 9999 or 5000 is now within the range of 16-bit ARR register.

Pulse-Width Modulation (PWM) as DAC

- A closed-loop control system consisting of a CPU-based controller (digital) interacting with a physical plant (analog) needs
  - Sensors with ADC (Analog-to-Digital Convertor) functionality to convert analog signals into digital domain, e.g., temperature, humidity…
  - Actuators with DAC (Digital-to-Analog Convertor) functionality to convert digital signals into analog domain, e.g., output voltage to control a motor…(real DAC requires low-pass filtering)
- PWM is a type of DAC. It uses a rectangular waveform to periodically switch on and off a voltage source to produce a desired average voltage output.





PWM for DC motor control

Pulse-Width Modulation (PWM)

- (Analog-to-Digital Convertor) Pulse Width refers to the time interval when the output signal is On; Duty Cycle is the percentage of time that the output signal is On:

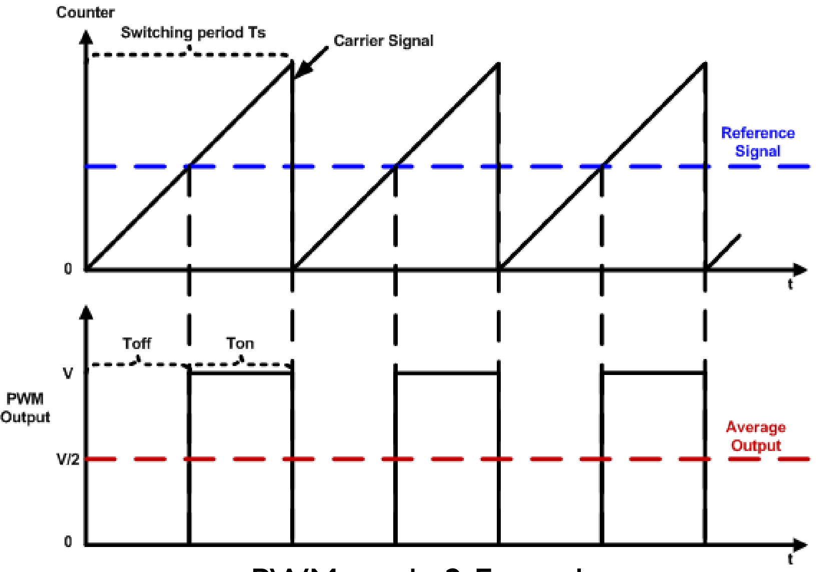
$$\text{Duty Cycle} = \frac{\text{On Time } (T_{on})}{\text{Switching Period } (Ts)} = \frac{T_{on}}{T_{on} + T_{off}}$$

- Higher Duty Cycle means higher average output:

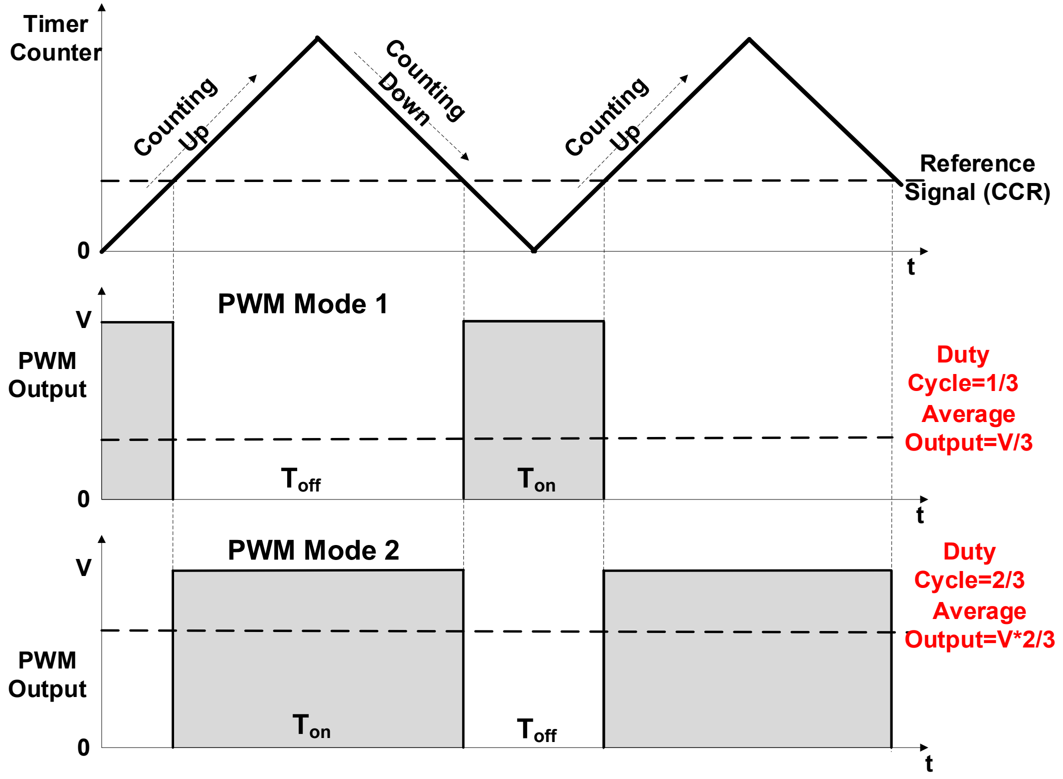
$$\text{Average Output} = \text{Duty Cycle } * \text{ Output in On State}$$

Pulse-Width Modulation (PWM)

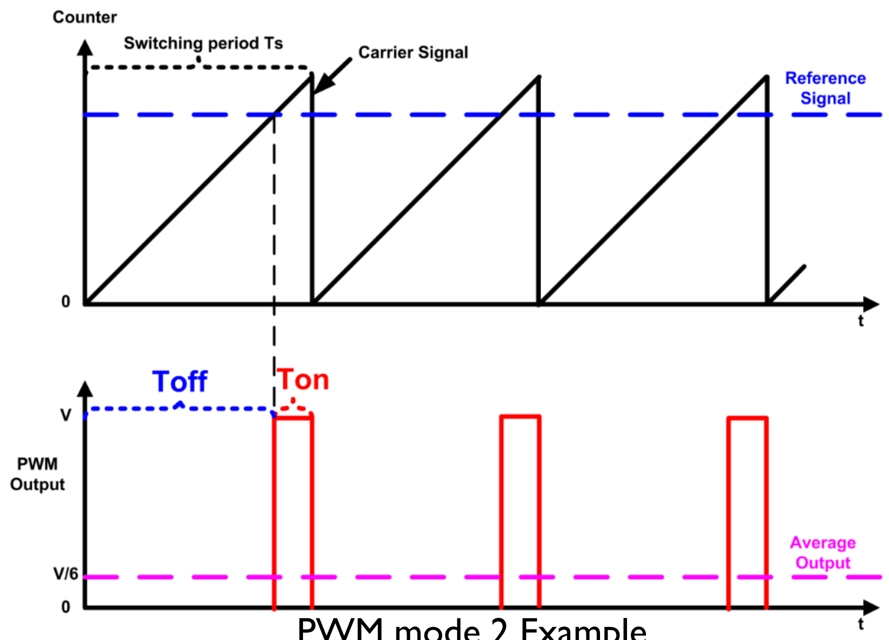
- The PWM output signal is determined by:
  - Reference signal stored in Compare and Capture Register (CCR); and the PWM mode
- PWM mode 1 (Low True): if the timer counter is (< or ≤) reference signal stored in CCR, the PWM output is on; otherwise it is off.
  - Higher CCR → higher duty cycle
- PWM mode 2 (High True): if the timer counter is (> or ≥) reference signal stored in CCR, the PWM output is on; otherwise it is off.
  - Higher CCR → lower duty cycle



PWM mode 2 Example



PWM Mode Summary

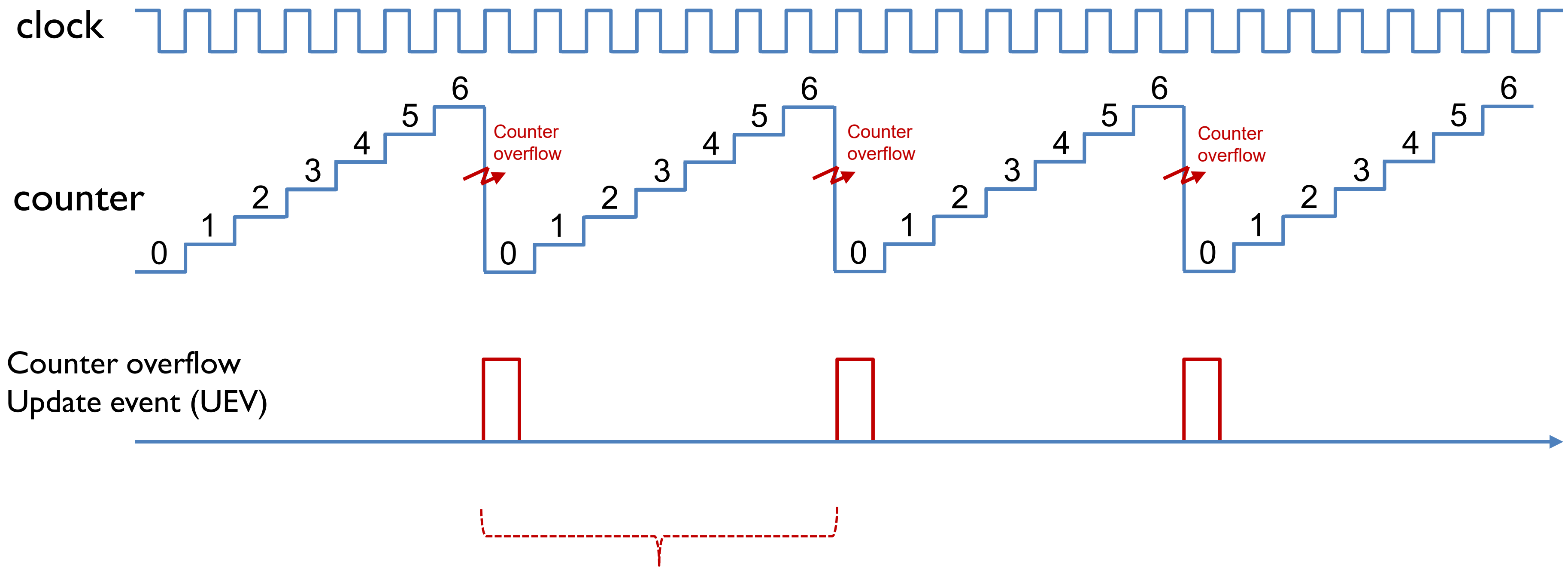


PWM mode 2 Example

Mode	Counter < or ≤ Reference	Counter > or ≥ Reference
PWM mode 1 (Low True)	Active	Inactive
PWM mode 2 (High True)	**Inactive**	**Active**

Edge-aligned Mode (Up-counting)

ARR = 6, RCR = 0

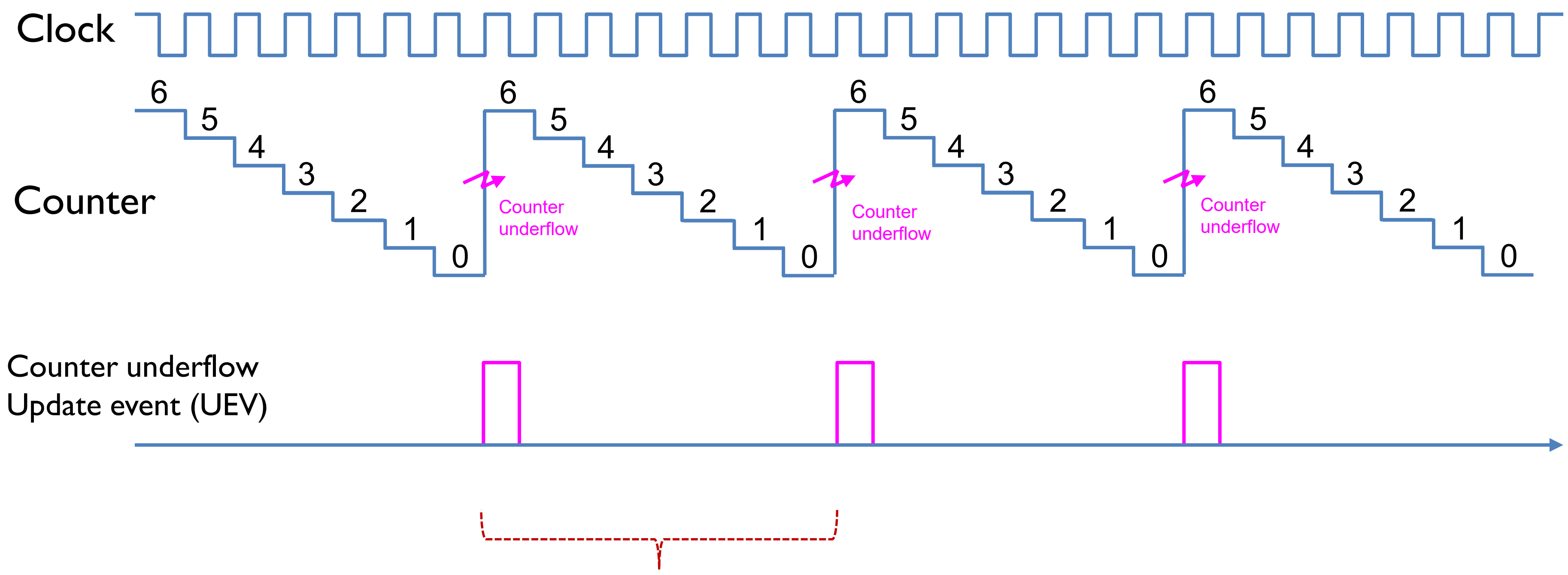


```
Period = (1 + ARR) * Clock Period
       = 7 * Clock Period
```

Clock Period = $1/f_{CNT}$ $f_{CNT} = f_{SOURCE}/(1 + PSC)$

Edge-aligned Mode (down-counting)

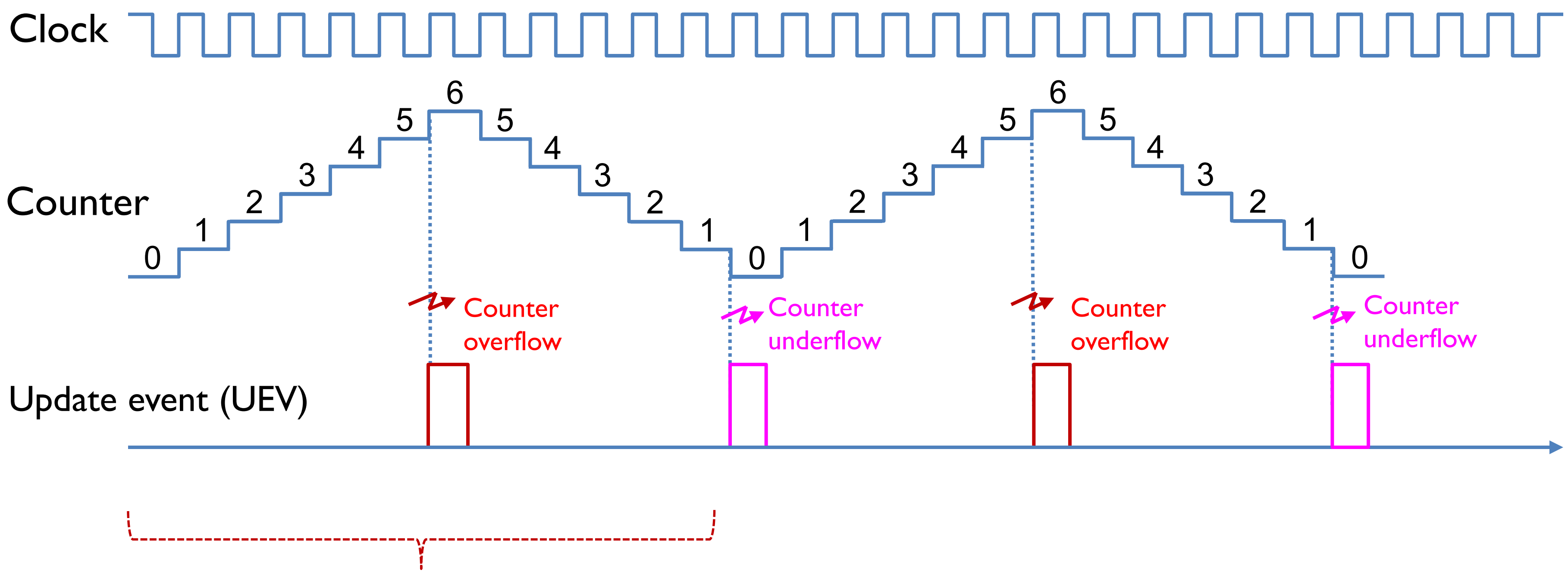
ARR = 6, RCR = 0



```
Period = (1 + ARR) * Clock Period
       = 7 * Clock Period
```

Center-aligned Mode

ARR = 6, RCR = 0



```
Period = 2 * ARR * Clock Period
       = 12 * Clock Period
```

Explanations

- We can configure the timer to repeatedly count up, count down, or, count up and down.
- In up-counting mode, auto-reload register (ARR) holds the maximum counter value, which is also called the auto-reload value. The counter counts from 0 to the auto-reload value, then restarts from 0, and generates a counter overflow event. This figure shows an example of the counter behavior when ARR is 6. In the up-counting mode, the counter rolls over and is reset when it exceeds 6. When the counter resets, it triggers a counter overflow and an update event (UEV). After that, a new period starts. The counting period is determined by the auto-reload value, as well as the clock period.
- In the down-counting mode, the counter counts from the auto-reloaded value down to 0. After the counter has reached 0, hardware automatically reloads the counter with the auto-reloaded value, and generate a counter underflow event and an UEV. The counting period of the down-counting mode is exactly the same as the counting period of the up-counting mode.
- In center-aligned mode, the counter counts from 0 to the auto-reload value minus 1, generates a counter overflow event, then counts from the auto-reload value down to 1 and generates a counter underflow event. Then it restarts counting from 0. In this mode, the counting direction changes automatically on counter overflow and underflow.

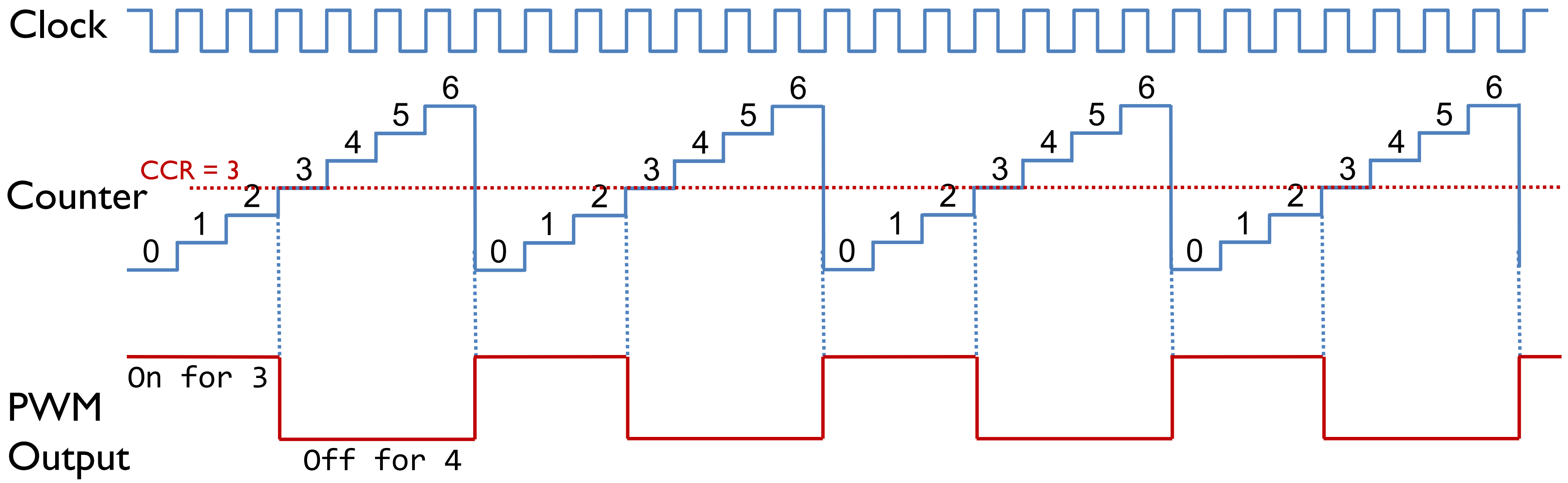
PWM Mode: Rigorous Definition

PWM Mode	Timer Counting Mode	On	Off
Mode 1 (Low True)	Up-counting	CNT < CCR	CNT ≥ CCR
	Down-counting	CNT ≤ CCR	CNT > CCR
Mode 2 (High True)	Up-counting	CNT ≥ CCR	CNT < CCR
	Down-counting	CNT > CCR	CNT ≤ CCR

PWM Mode 1 (Low-True)

PWM	Timer Counting	On	Off
Mode 1 (Low True)	Up-counting	CNT < CCR	CNT ≥ CCR
	Down-counting	CNT ≤ CCR	CNT > CCR
Mode 2 (High True)	Up-counting	CNT ≥ CCR	CNT < CCR
	Down-counting	CNT > CCR	CNT ≤ CCR

Up-counting mode, ARR = 6, CCR = 3



```
Duty Cycle = CCR / (ARR + 1)
           = 3 / 7
```

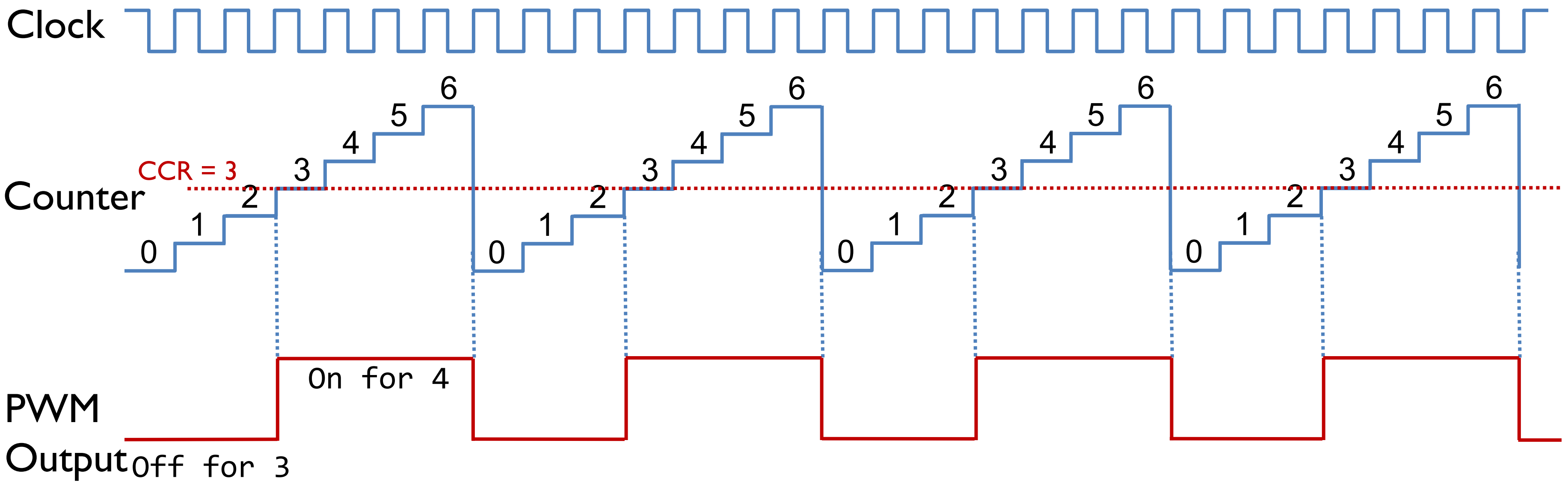
$$\text{Duty Cycle} = \frac{CCR}{ARR + 1} = \frac{3}{7}$$

```
Timer Period = (ARR + 1) * Clock Period
             = 7 * Clock Period
```

PWM Mode 2 (High-True)

PWM	Timer Counting	On	Off
Mode 1 (Low True)	Up-counting	CNT < CCR	CNT ≥ CCR
	Down-counting	CNT ≤ CCR	CNT > CCR
Mode 2 (High True)	Up-counting	CNT ≥ CCR	CNT < CCR
	Down-counting	CNT > CCR	CNT ≤ CCR

Up-counting mode, ARR = 6, CCR = 3



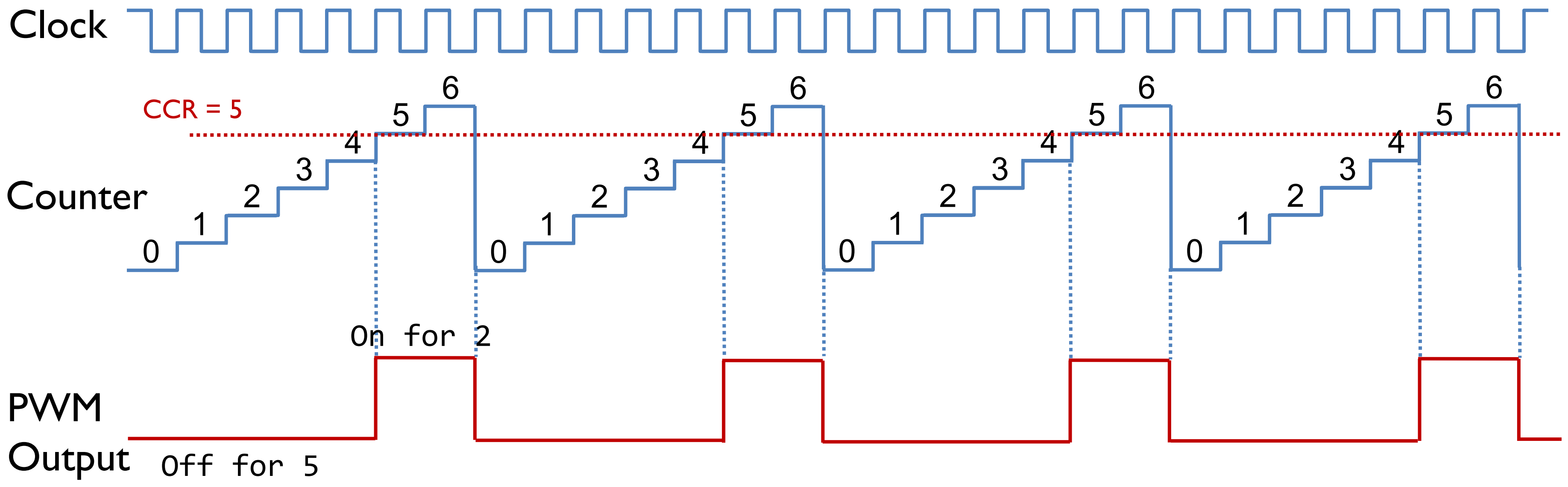
$$\text{Duty Cycle} = 1 - \frac{\text{CCR}}{\text{ARR} + 1} = \frac{4}{7}$$

Timer Period = (ARR + 1) * Clock Period = 7 * Clock Period

PWM Mode 2 (High-True)

PWM	Timer Counting	On	Off
Mode 1 (Low True)	Up-counting	CNT < CCR	CNT ≥ CCR
	Down-counting	CNT ≤ CCR	CNT > CCR
Mode 2 (High True)	Up-counting	CNT ≥ CCR	CNT < CCR
	Down-counting	CNT > CCR	CNT ≤ CCR

Up-counting mode, ARR = 6, CCR = 5



```
Duty Cycle = 1 - CCR / (ARR + 1)
           = 2 / 7
```

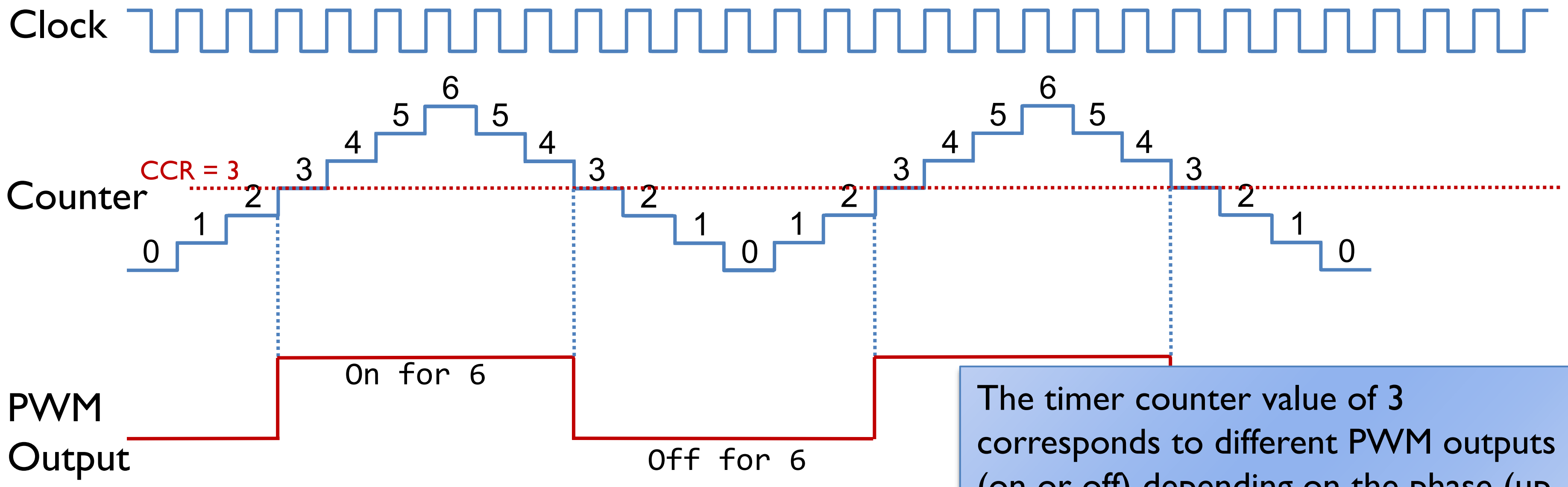
$$\text{Duty Cycle} = 1 - \frac{\text{CCR}}{\text{ARR} + 1} = \frac{2}{7}$$

```
Timer Period = (ARR + 1) * Clock Period
             = 7 * Clock Period
```

PWM Mode 2 (High-True)

PWM	Timer Counting	On	Off
Mode 1 (Low True)	Up-counting	CNT < CCR	CNT ≥ CCR
	Down-counting	CNT ≤ CCR	CNT > CCR
Mode 2 (High True)	Up-counting	CNT ≥ CCR	CNT < CCR
	Down-counting	CNT > CCR	CNT ≤ CCR

Center-aligned mode, ARR = 6, CCR = 3



The timer counter value of 3 corresponds to different PWM outputs (on or off) depending on the phase (up-counting or down-counting)

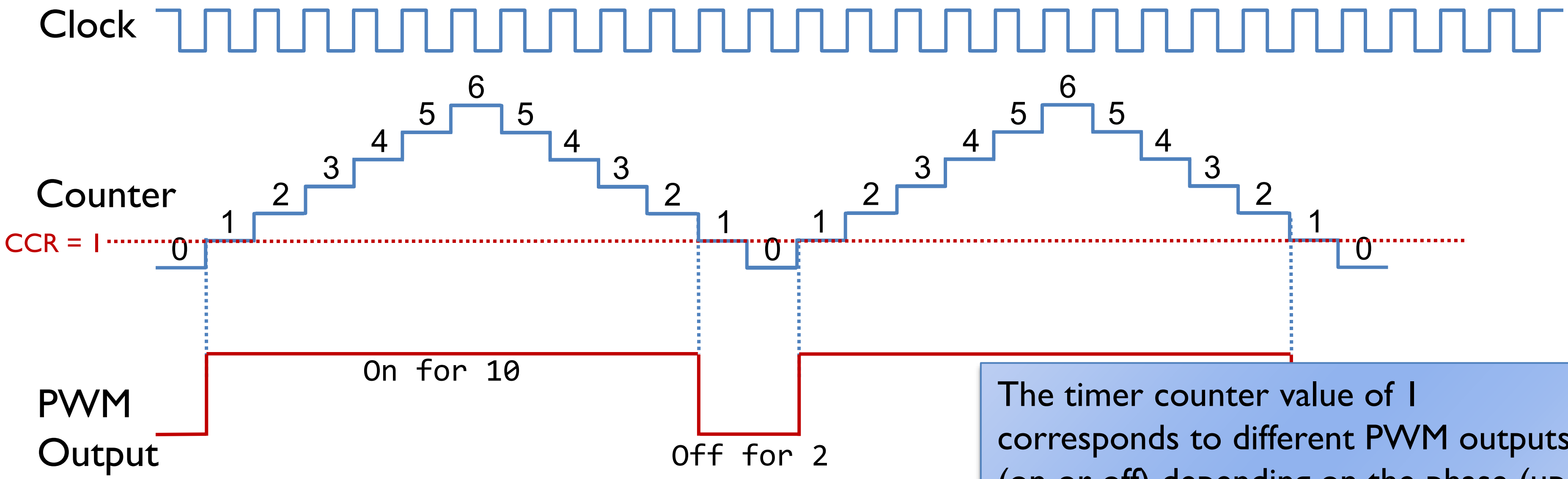
$$\text{Duty Cycle} = 1 - \frac{CCR}{ARR} = \frac{1}{2}$$

Timer Period = 2 * ARR * Clock Period = 12 * Clock Period

PWM Mode 2 (High-True)

PWM	Timer Counting	On	Off
Mode 1 (Low True)	Up-counting	CNT < CCR	CNT ≥ CCR
	Down-counting	CNT ≤ CCR	CNT > CCR
Mode 2 (High True)	Up-counting	CNT ≥ CCR	CNT < CCR
	Down-counting	CNT > CCR	CNT ≤ CCR

Center-aligned mode, ARR = 6, CCR = 1



The timer counter value of 1 corresponds to different PWM outputs (on or off) depending on the phase (up-counting or down-counting)

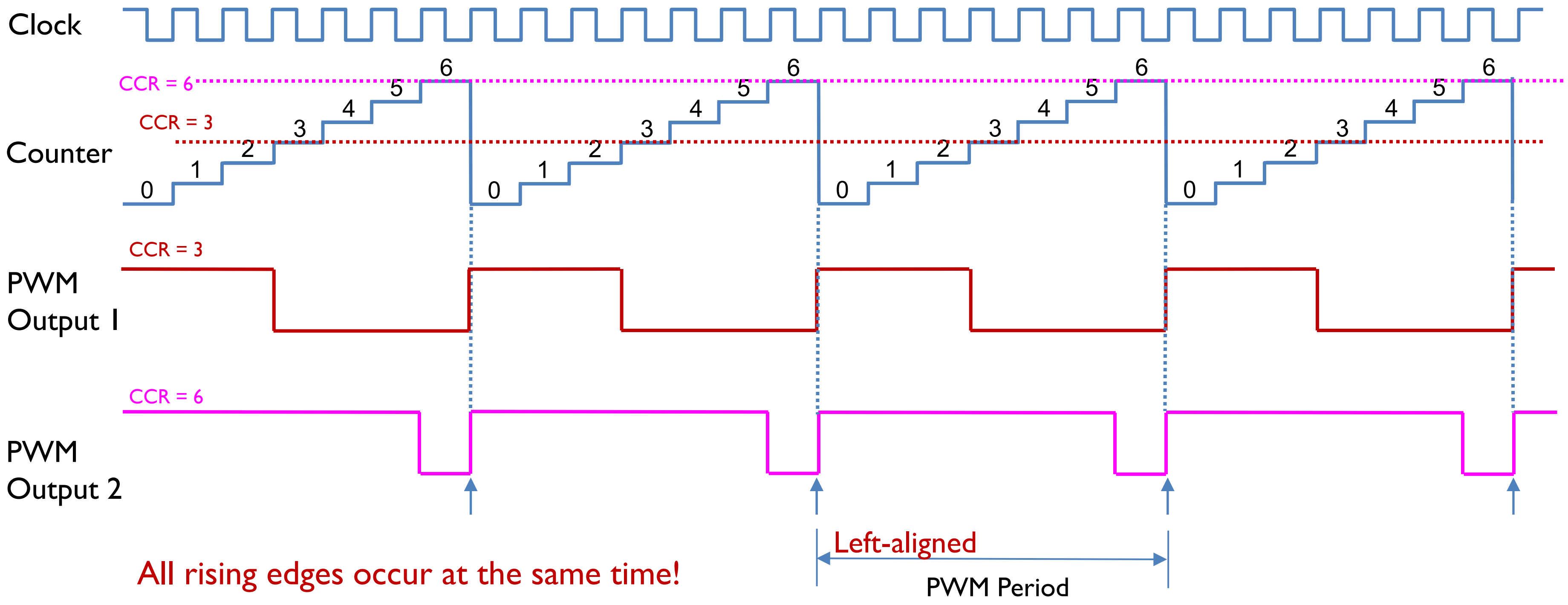
$$\text{Duty Cycle} = 1 - \frac{\text{CCR}}{\text{ARR}} = \frac{5}{6}$$

```
Timer Period = 2 * ARR * Clock Period
             = 12 * Clock Period
```

PWM Mode 1: Left Edge-aligned

PWM	Timer Counting	On	Off
Mode 1 (Low True)	Up-counting	CNT < CCR	CNT ≥ CCR
	Down-counting	CNT ≤ CCR	CNT > CCR
Mode 2 (High True)	Up-counting	CNT ≥ CCR	CNT < CCR
	Down-counting	CNT > CCR	CNT ≤ CCR

Up-counting mode, ARR = 6, CCR = 3



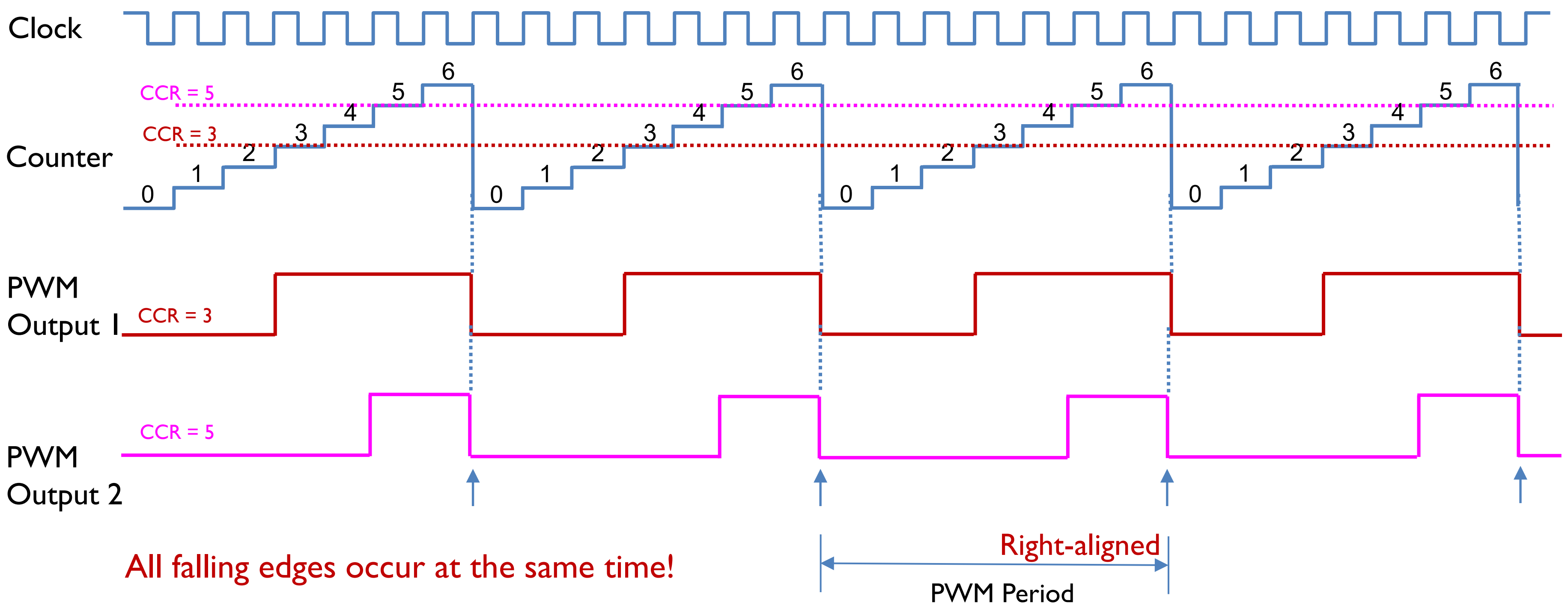
All rising edges occur at the same time!

In the up-counting mode, when multiple PWM signals are generated by the same timer, the PWM pulses are left edge aligned, because all rising edges are aligned to the left side of the PWM period.

PWM Mode 2: Right Edge-aligned

PWM	Timer Counting	On	Off
Mode 1 (Low True)	Up-counting	CNT < CCR	CNT ≥ CCR
	Down-counting	CNT ≤ CCR	CNT > CCR
Mode 2 (High True)	Up-counting	CNT ≥ CCR	CNT < CCR
	Down-counting	CNT > CCR	CNT ≤ CCR

Up-counting mode, ARR = 6, CCR = 3



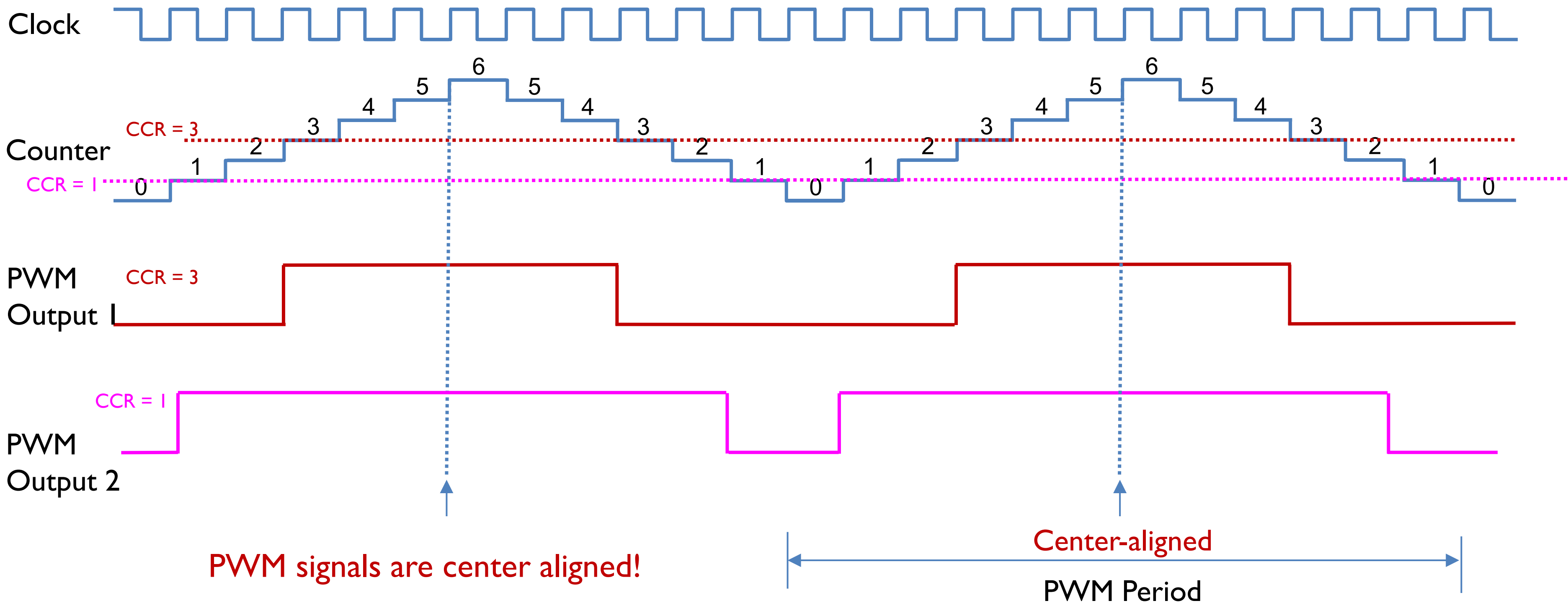
All falling edges occur at the same time!

In the down-counting mode, when multiple PWM signals are generated by the same timer, the PWM pulses are right edge aligned, because all falling edges are aligned to the right side of the PWM period.

PWM	Timer Counting	On	Off
Mode 1 (Low True)	Up-counting	CNT < CCR	CNT ≥ CCR
	Down-counting	CNT ≤ CCR	CNT > CCR
Mode 2 (High True)	Up-counting	CNT ≥ CCR	CNT < CCR
	Down-counting	CNT > CCR	CNT ≤ CCR

PWM Mode 2: Center aligned

Center-aligned counting mode, ARR = 6, CCR = 3



In the center-aligned counting mode, when multiple PWM signals are generated by the same timer, the PWM pulses are center aligned, because their centers are aligned with peaks of the timer counter.

PWM Pulse Alignment

PWM Mode	Timer Counting Mode	Pulse Alignment
Mode 1 (Low True)	Up-counting	Left edge
	Down-counting	Right edge
	Center-aligned Counting	Center aligned
Mode 2 (High True)	Up-counting	Right edge
	Down-counting	Left edge
	Center-aligned Counting	Center aligned

PWM Output Polarity

- Each timer has two PWM modes, Mode 1 and Mode 2. These two modes are complementary to each other. However, software can select the output polarity by writing the CCxP bit in the CCER register.
- Software can select either active high or active low. If it is active high, the output is high voltage for the active state, and low voltage for the inactive state. On the other hand, for active low, the output is low voltage for the active state, and high voltage for the inactive state.

Mode	Counter < CCR	Counter ≥ CCR
PWM mode 1 (Low True)	Active	Inactive
PWM mode 2 (High True)	Inactive	Active

Output Polarity	Active	Inactive
Active High	High Voltage	Low Voltage
Active Low	Low Voltage	High Voltage

Summary of Equations

- Timer clock frequency f_{CK_CNT} vs. CPU Clock Frequency f_{SOURCE}

$$f_{CK\_CNT} = \frac{f_{SOURCE}}{PSC + 1}$$

- Timer interrupt frequency f_{Timer} with up-counting or down-counting mode:

$$f_{Timer} = \frac{f_{CK\_CNT}}{ARR + 1}; \; Timer\ Period = \frac{ARR + 1}{f_{CK\_CNT}} = (ARR + 1) * Clock\ Period$$

- Timer interrupt frequency f_{Timer} with center-aligned counting mode: $f_{Timer} = \frac{f_{CK\_CNT}}{2 * ARR}; Timer\ Period = (2 * ARR) * Clock\ Period$

- PWM duty cycle for Mode 1 (Low-True):

$$Duty\ Cycle = \frac{\mathrm{CCR}}{\mathrm{ARR} + 1}$$

- PWM duty cycle for Mode 2 (High-True):

$$Duty\ Cycle = 1 - \frac{\mathrm{CCR}}{\mathrm{ARR} + 1}$$

References

- Lecture 13: Timer PWM Output
  - https://www.youtube.com/watch?v=zkrVHIcLGww&list=PLRJhV4hUhIymmp5CCeIFPyxbknsdcXCc8&index=13