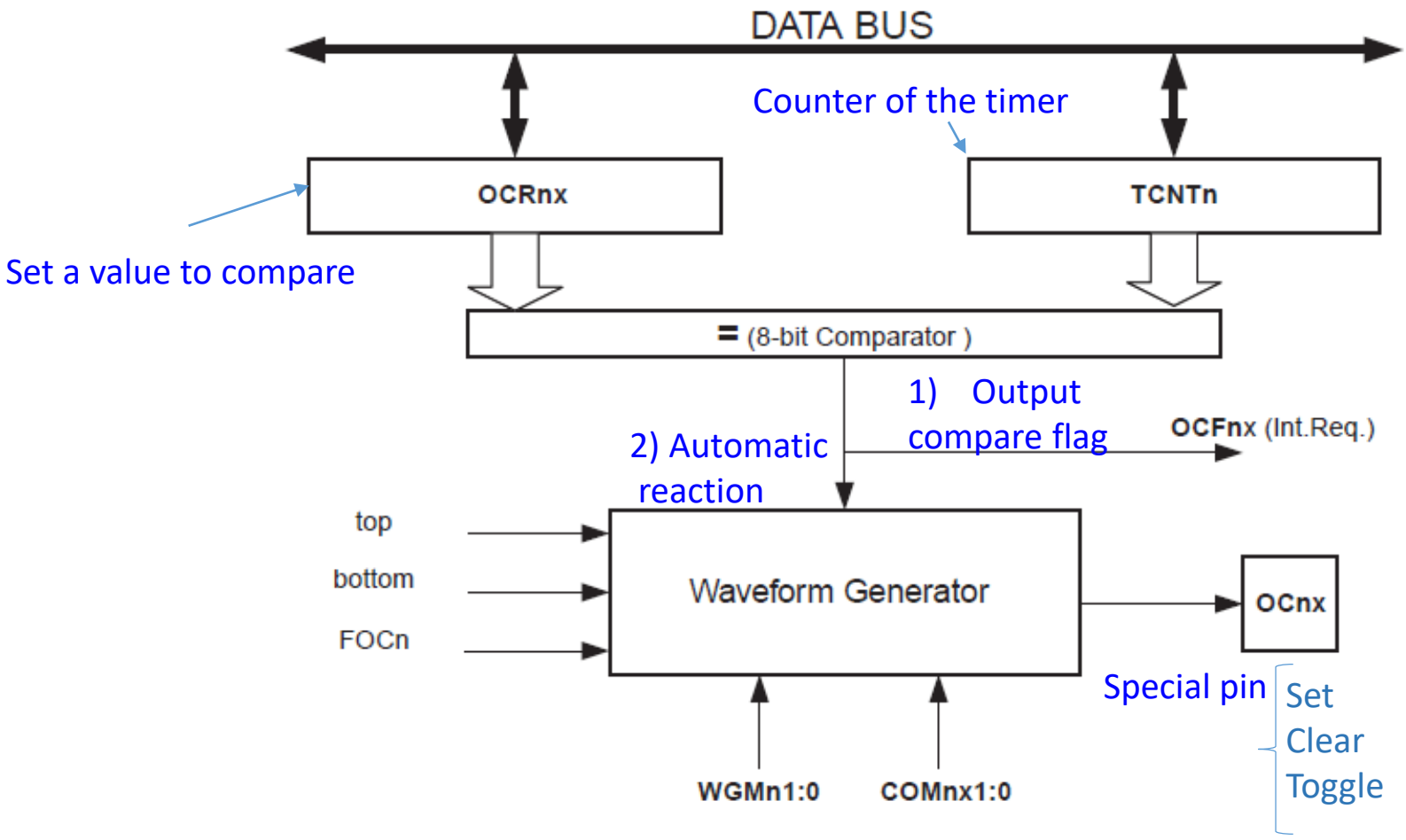
ATMega328 Timers: Clear Timer on Match Compare (CTC) Mode

Timer0 : Clear Timer on Match Compare (CTC)

Figure 15-3. Output Compare Unit, Block Diagram



Source: Microchip

Timer0 : Clear Timer on Match Compare Mode (CTC)

- Counts for every tick until reaches the value in the compare register (OCR0A)
- ATMEGA328P offers two compare registers.
- After match (OCR0A) the count (TCNT0) is cleared and starts again at zero.

OCR0A – Output Compare Register A

Bit	7	6	5	4	3	2	1	0	
0x27 (0x47)	OCR0A[7:0]								OCR0A
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

OCR0B – Output Compare Register B

Bit	7	6	5	4	3	2	1	0	
0x28 (0x48)	OCR0B[7:0]								OCR0B
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Timer0 : Clear Timer on Match Compare Mode (CTC)

➢ To enable the CTC mode:

TCCR0A – Timer/Counter Control Register A

Bit	7	6	5	4	3	2	1	0
0x24 (0x44)	COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

TCCR0B – Timer/Counter Control Register B

Bit	7	6	5	4	3	2	1	0
0x25 (0x45)	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00
Read/Write	W	W	R	R	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Table 15-8. Waveform Generation Mode Bit Description

Mode	WGM02	WGM01	WGM00	Timer/Counter Mode of Operation	TOP	Update of OCRx at	TOV Flag Set on(1)(2)
0	0	0	0	Normal	0xFF	Immediate	MAX
1	0	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	0	1	0	CTC	OCRA	Immediate	MAX
3	0	1	1	Fast PWM	0xFF	BOTTOM	MAX
4	1	0	0	Reserved	–	–	–
5	1	0	1	PWM, Phase Correct	OCRA	TOP	BOTTOM
6	1	1	0	Reserved	–	–	–
7	1	1	1	Fast PWM	OCRA	BOTTOM	TOP

Normal

CTC

Notes: 1. MAX = 0xFF
2. BOTTOM = 0x00

Source: Microchip

Timer0 : Clear Timer on Match Compare (CTC)

- Two choices when a match compare occurs:

1) Interrupt Notification: Output Compare Flag → '1'

TIFR0 – Timer/Counter 0 Interrupt Flag Register

Flags for polling or…

Bit	7	6	5	4	3	2	1	0	
0x15 (0x35)	–	–	–	–	–	OCF0B	OCF0A	TOV0	TIFR0
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

TIMSK0 – Timer/Counter Interrupt Mask Register

Enable interrupt

Bit	7	6	5	4	3	2	1	0	
(0x6E)	–	–	–	–	–	OCIE0B	OCIE0A	TOIE0	TIMSK0
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

2) Automatic Reaction on Events (Hardware CTC): Related output pins can be configured to be set, cleared, or toggled automatically on a compare match.

Timer0 : Clear Timer on Match Compare (CTC)

2) Automatic Reaction on Events (Hardware CTC):

- Non-PWM mode: Toggle, clear and set on pins OC0A or OC0B
- PWM Mode:
  - Fast PWM
  - Phase correct PWM

TCCR0A – Timer/Counter Control Register A

Bit	7	6	5	4	3	2	1	0
0x24 (0x44)	COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

A similar table to compare with OCR0B

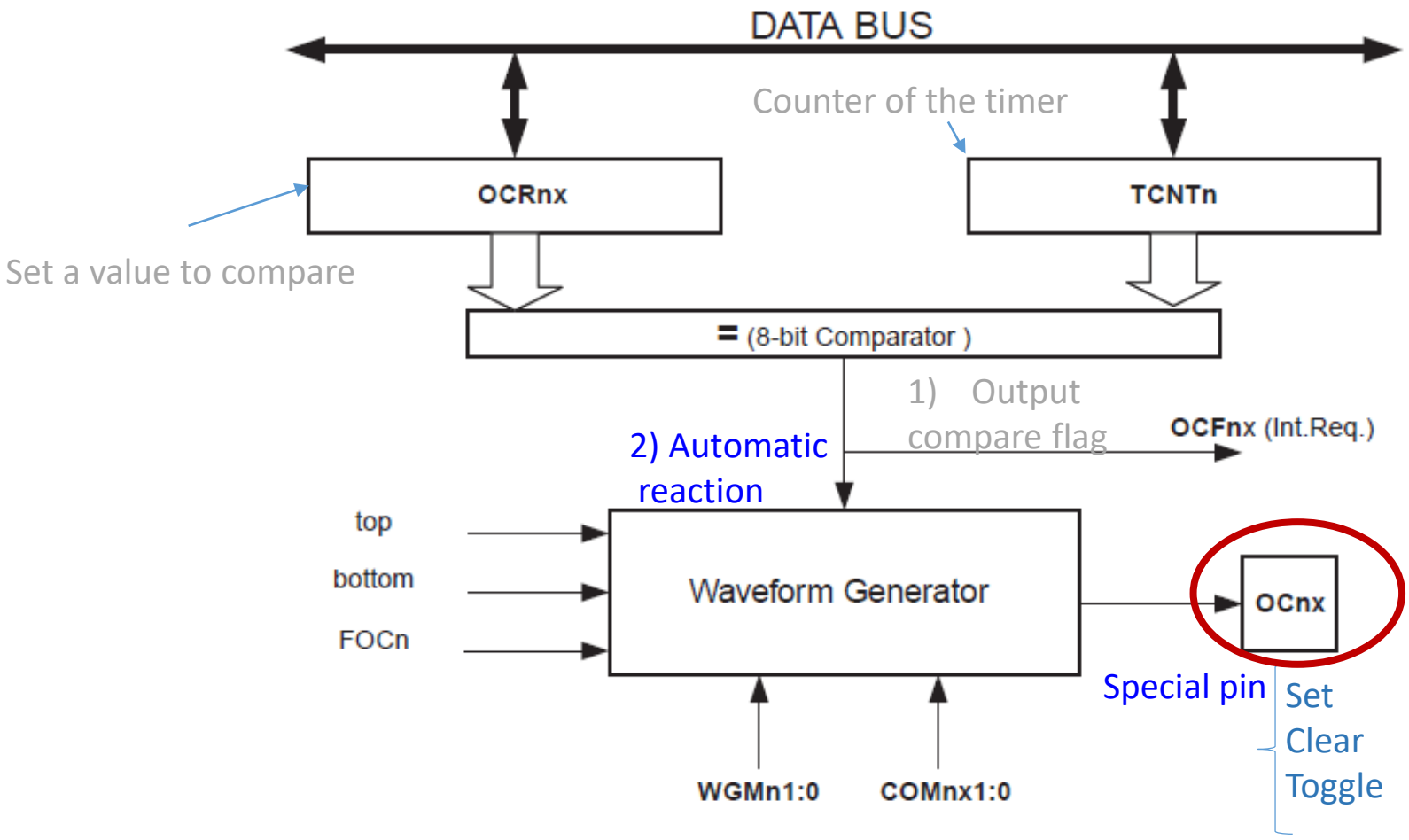
Table 15-2. Compare Output Mode, non-PWM Mode

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected.
0	1	Toggle OC0A on Compare Match
1	0	Clear OC0A on Compare Match
1	1	Set OC0A on Compare Match

Set pin OC0A as output

Timer0 : Clear Timer on Match Compare (CTC)

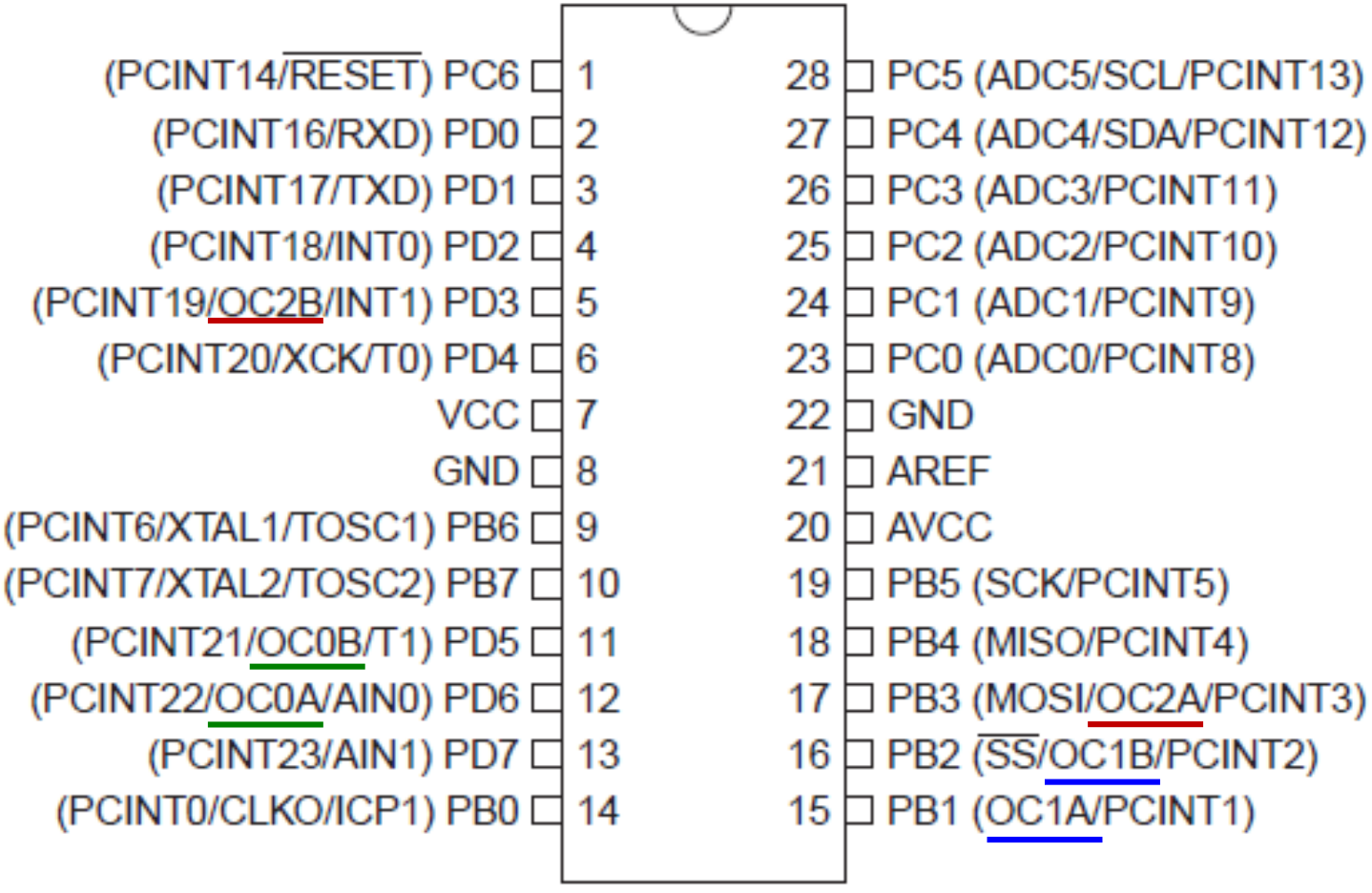
Figure 15-3. Output Compare Unit, Block Diagram



Source: Microchip

Timer0 : Clear Timer on Match Compare (CTC)

OUTPUT COMPARE PINs → In Arduino board marked with " ~ "



TIMER0 TIMER1 TIMER2

Timer0: Hardware CTC Mode – Blinking LED – 12.8 ms

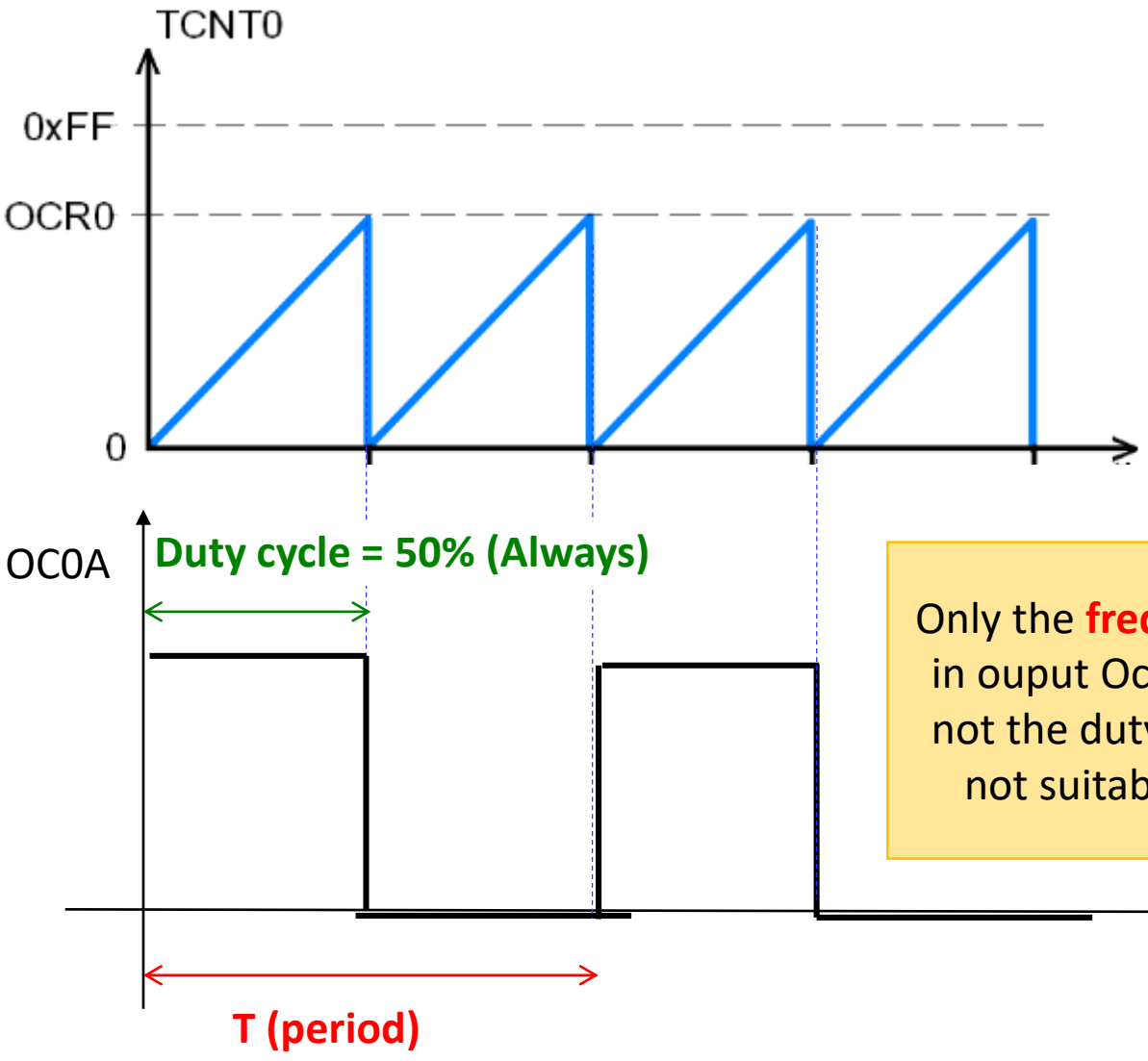
```
#include <avr/io.h>

int main(void) {
//set PD6 as output. PD6/OC0A (Hardware waveform)
 DDRD |= (1 << DDD6);

TCNT0 = 0;      // Initialize counter (first time)
OCR0A = 200;  // Set number of ticks for comparing
TCCR0A |= (1 << WGM01); // Mode CTC on OCR0A

TCCR0A |= (1 << COM0A0);   // Set Toggle for pin OC0A when
                           // Compare Match.

// set prescaler to 1024 and start the timer
TCCR0B |= (1 << CS02) | (1 << CS00);

while (1) {
// do nothing
}
}
```

200 ticks * (1/ (16MHz/1024)) = 12.800 us = 12,8 ms

Prescaler

That's all we need to do! No need to check any flag bit, no need to attend to any interrupts, nothing…

Timer0: Hardware CTC Mode – Output waveform (OC0A)



Only the **frequency** of the waveform in ouput Ocnx can be modified but not the duty. For this reason CTC is not suitable to provide a PWM.