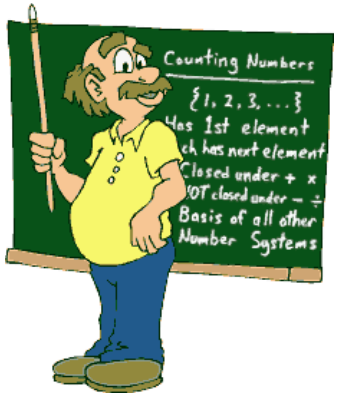
IECA

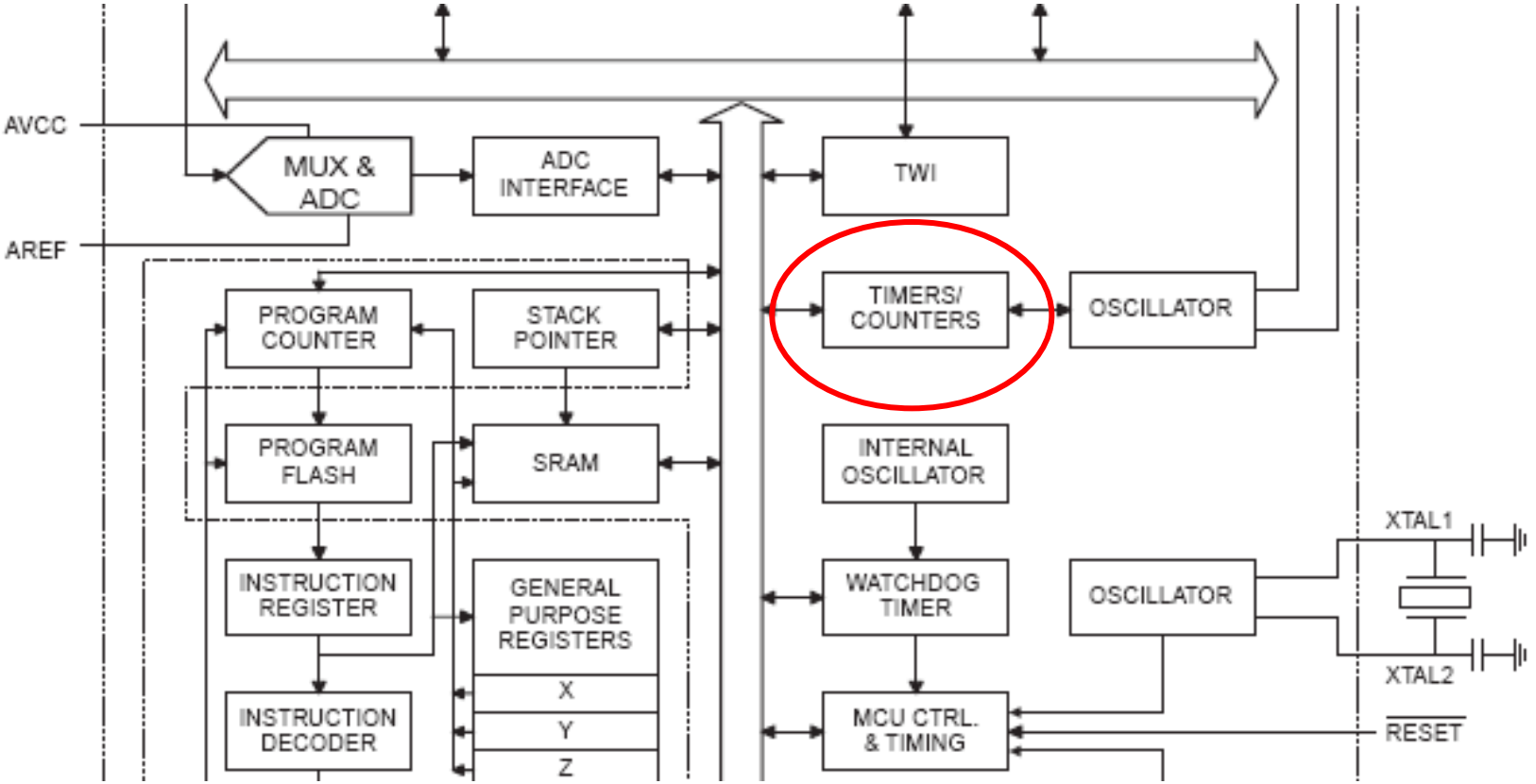
Embedded Computer Architecture

Lesson 13: Timers in Normal Mode



Version: 9-11-2015, Henning Hargaard

The 3 timers of Mega32



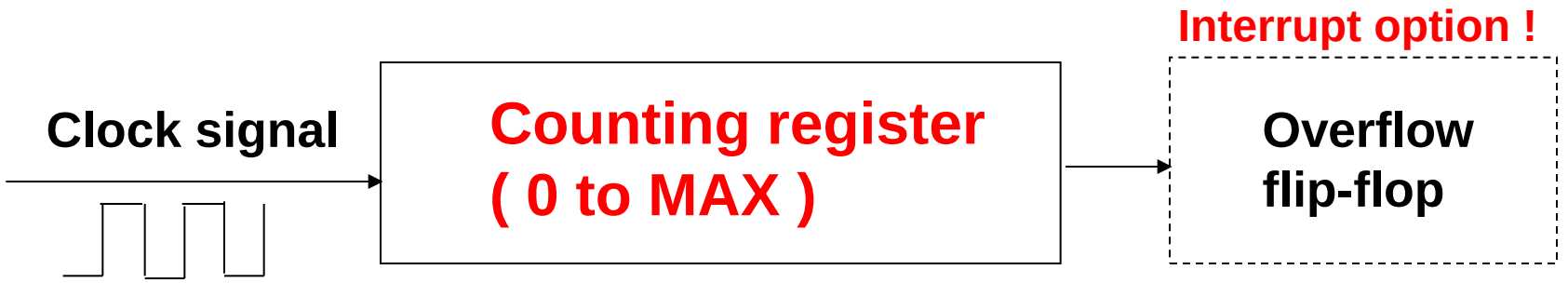
The Mega32 timers

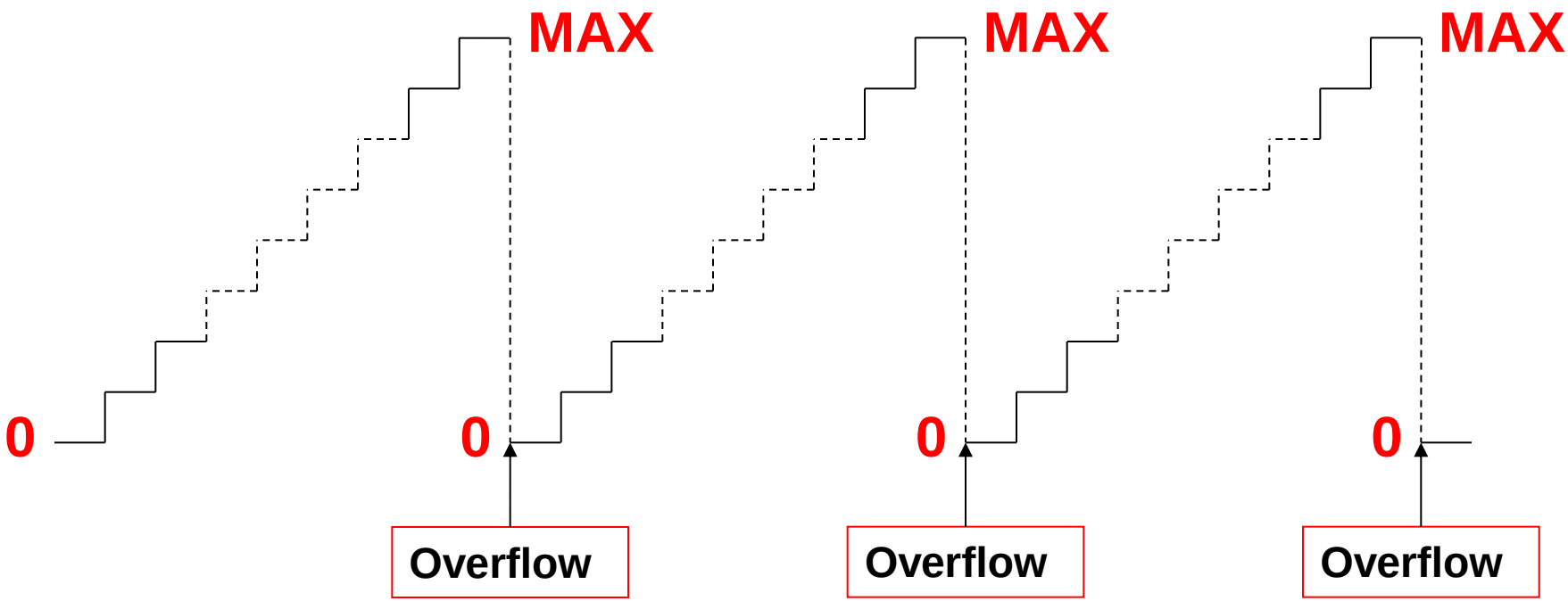
- **Timer 0 :**
  8 bit (MAX = 255).
  Normal, CTC and PWM modes.

- **Timer 1 :**
  16 bit (MAX = 65535).
  Normal, CTC, many PWM modes.
  Input Capture.

- **Timer 2 :**
  8 bit (MAX = 255).
  Normal, CTC and PWM modes.
  Asynchronous mode (Real Time Clock).

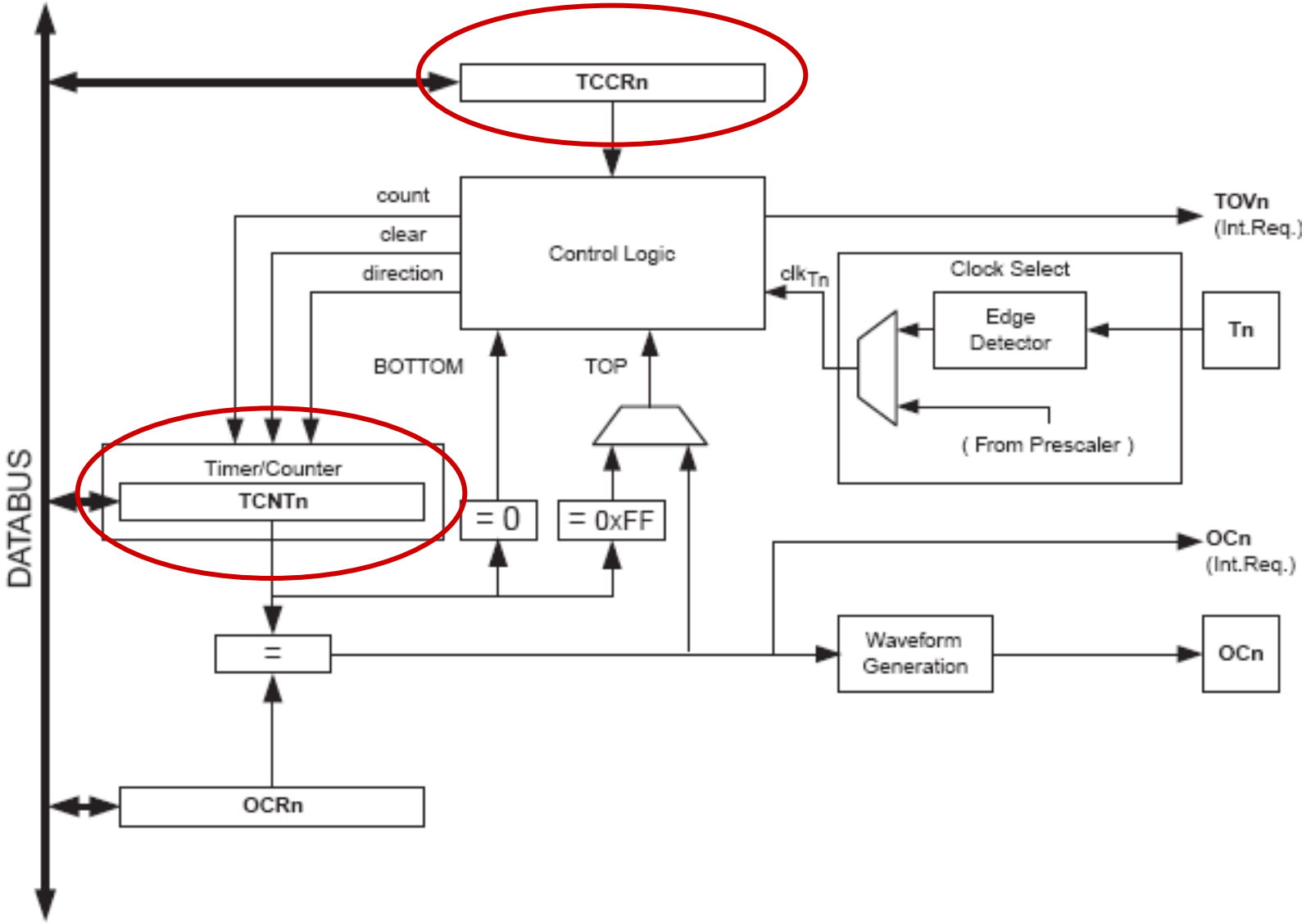
Timer in ”normal mode”





TIMER 0

Timer 0 (8 bit)

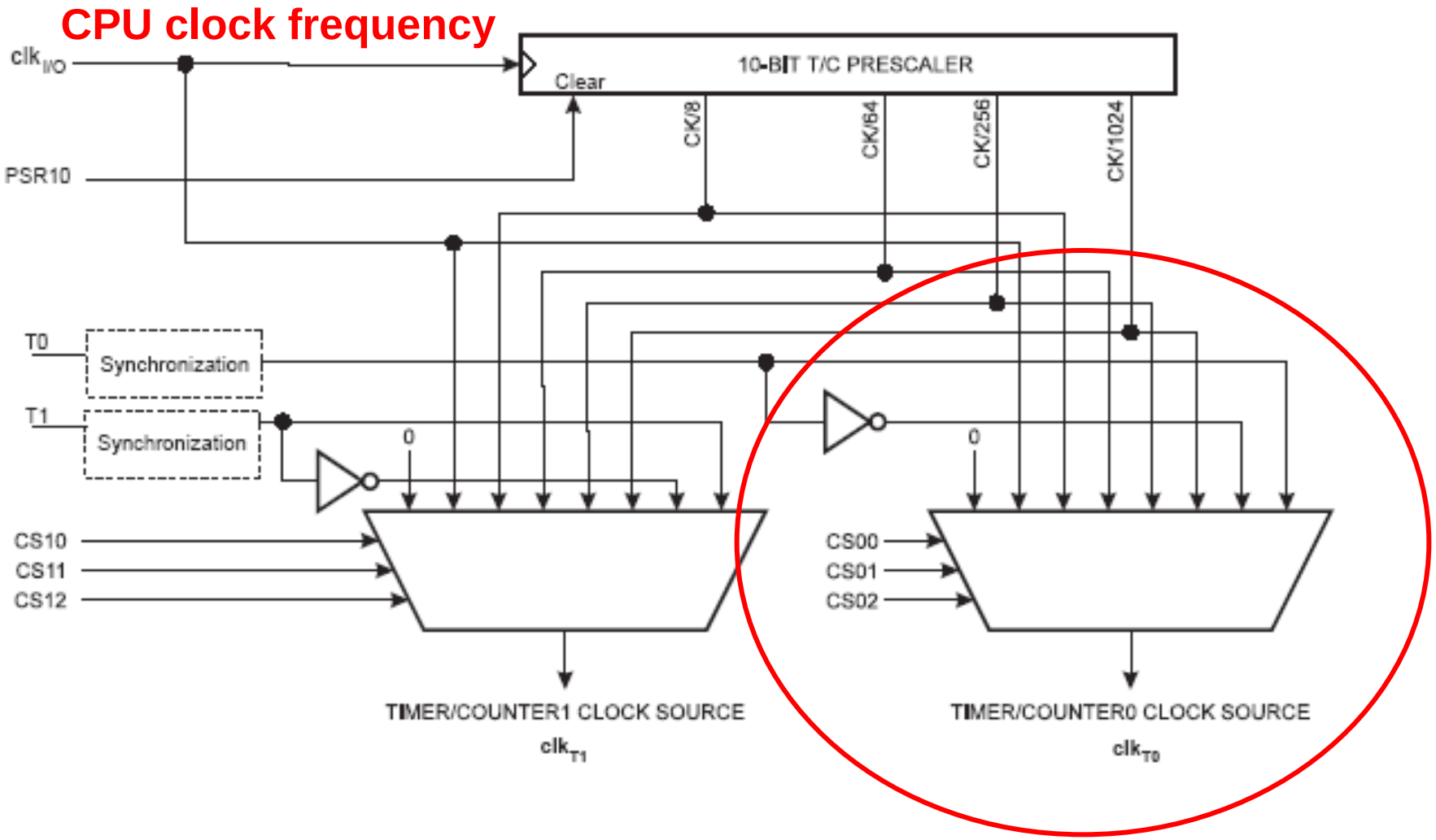


Timer 0: Counting register

Bit	7	6	5	4	3	2	1	0	
	TCNT0[7:0]								TCNT0
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	

- TCNT0 is the counting registeret for timer 0.
- Automatically upcounted by the timer 0 clock signal.
- Notice that the register both can be read and written (from the program).

Prescaler for timer 0



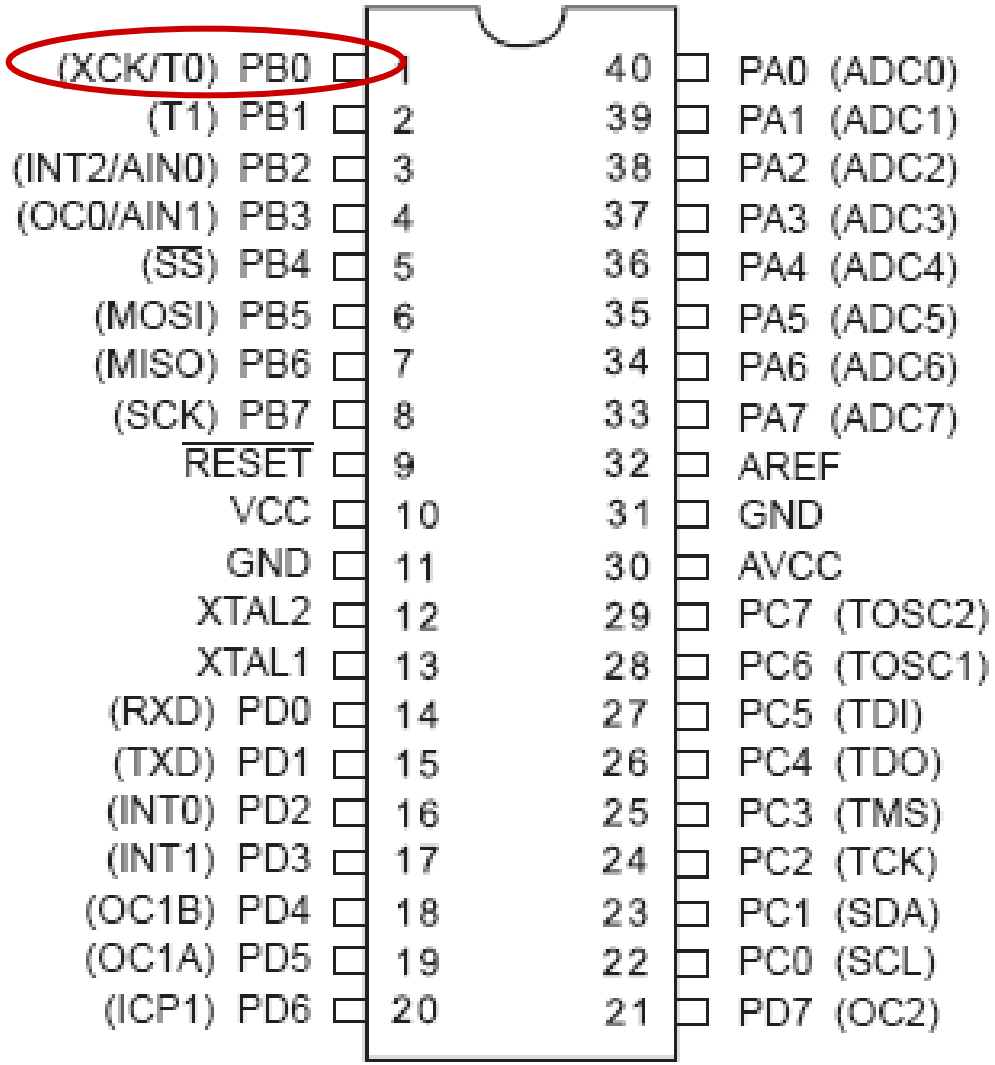
Timer 0: Clock selection

Bit	7	6	5	4	3	2	1	0	
	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00	TCCR0
Read/Write	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	

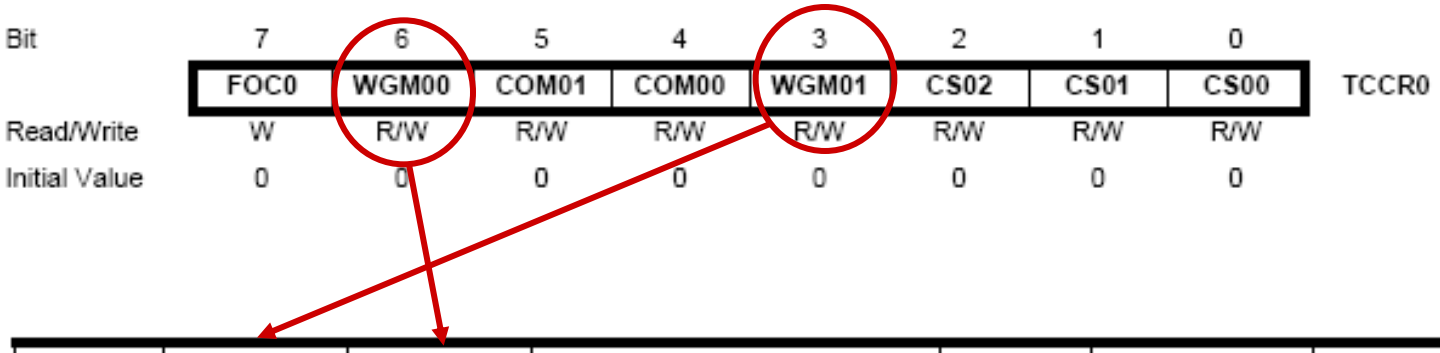
CS02	CS01	CS00	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	$clk_{I/O}$/(No prescaling)
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/64 (From prescaler)
1	0	0	$clk_{I/O}$/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)
1	1	0	External clock source on T0 pin. Clock on falling edge.
1	1	1	External clock source on T0 pin. Clock on rising edge.

Timer 0: External clock (the pin T0)

Timer 0



Timer 0: Mode selection



Bit	7	6	5	4	3	2	1	0	
	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00	TCCR0
Read/Write	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	

Mode	WGM01 (CTC0)	WGM00 (PWM0)	Timer/Counter Mode of Operation	TOP	Update of OCR0	TOV0 Flag Set-on
0	0	0	Normal	0xFF	Immediate	MAX
1	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	1	0	CTC	OCR0	Immediate	MAX
3	1	1	Fast PWM	0xFF	TOP	MAX

Timer 0 overflow flag (register TIFR)

Bit	7	6	5	4	3	2	1	0	
	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	OCF0	TOV0	TIFR
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	

- **Bit 0 in the register TIFR will be set to 1, each time Timer 0 overflows.**
- **Can be cleared again, if we *write a 1* to it (unless we use interrupts).**
- **C code:**
  TIFR = 0b00000001;
  or:
  TIFR = 1<<TOV0;

(Timer 0: Overflow interrupt)

Vector No.	Program Address(2)	Source	Interrupt Definition
1	$000(1)	RESET	External Pin, Power-on Reset, Brown-out Reset, Watchdog Reset, and JTAG AVR Reset
2	$002	INT0	External Interrupt Request 0
3	$004	INT1	External Interrupt Request 1
4	$006	TIMER2 COMP	Timer/Counter2 Compare Match
5	$008	TIMER2 OVF	Timer/Counter2 Overflow
6	$00A	TIMER1 CAPT	Timer/Counter1 Capture Event
7	$00C	TIMER1 COMPA	Timer/Counter1 Compare Match A
8	$00E	TIMER1 COMPB	Timer/Counter1 Compare Match B
9	$010	TIMER1 OVF	Timer/Counter1 Overflow
10	$012	TIMER0 OVF	Timer/Counter0 Overflow
11	$014	SPI, STC	Serial Transfer Complete
12	$016	USART, RXC	USART, Rx Complete
13	$018	USART, UDRE	USART Data Register Empty
14	$01A	USART, TXC	USART, Tx Complete
15	$01C	ADC	ADC Conversion Complete
16	$01E	EE_RDY	EEPROM Ready
17	$020	ANA_COMP	Analog Comparator
18	$022	TWI	Two-wire Serial Interface
19	$024	INT2	External Interrupt Request 2
20	$026	TIMER0 COMP	Timer/Counter0 Compare Match
21	$028	SPM_RDY	Store Program Memory Ready

Timer 0 overflow interrupt

(Timer 0: Overflow interrupt enable)

Bit	7	6	5	4	3	2	1	0	
	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0	TIMSK
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	

- Bit TOIE0:

  0 => Disable o**verflow interrupt**.

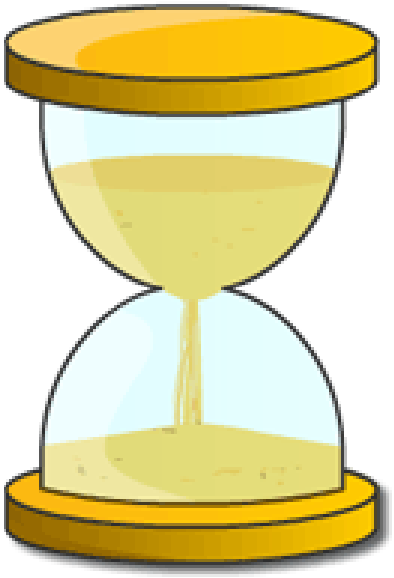
  1 => Enable o**verflow interrupt**.

Test (”socrative.com”: Room = MSYS)

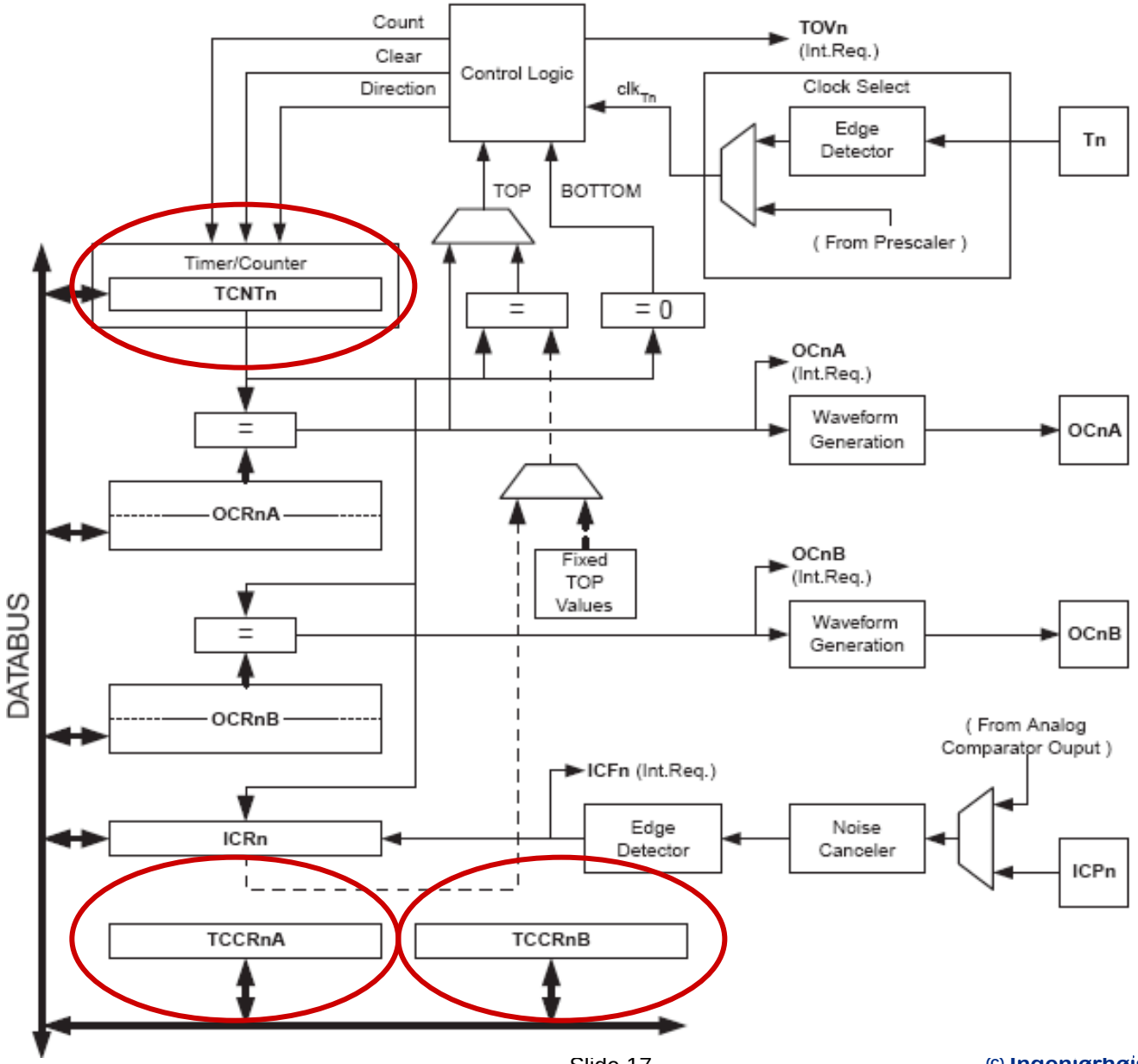
- Assume we are using a CPU clock frequency of 3,6864 MHz, and Timer 0 is initialized to Normal Mode.
  The Timer 0 clock prescaler is set to 64.
  How often will Timer 0 overflow ?

- A: 225 times per second.
- B: 256 times per second.
- C: 57600 times per second.
- D: 64 times per second.

TIMER 1



Timer 1 (16 bit)



Timer 1: Counting register (16 bit)

Bit	7	6	5	4	3	2	1	0	
	TCNT1[15:8]								TCNT1H
	TCNT1[7:0]								TCNT1L
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	

- TCNT1 is the counting registeret for timer 1.
- Automatically upcounted by the timer 1 clock signal.
- Notice that the register both can be read and written (from the program).

Timer 1: Counting register (16 bit)

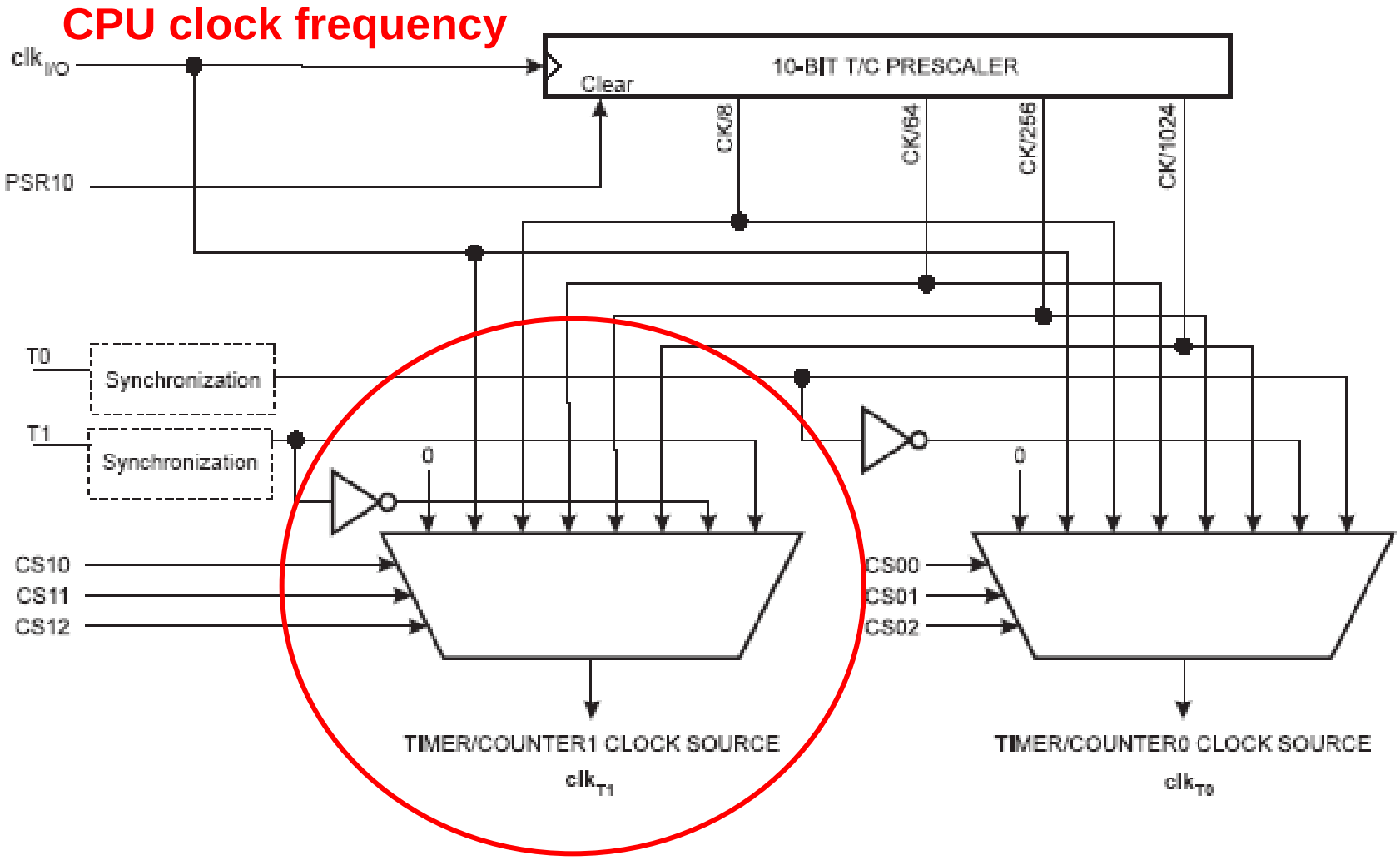
Bit	7	6	5	4	3	2	1	0	
	TCNT1[15:8]								TCNT1H
	TCNT1[7:0]								TCNT1L
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Sequence by writing: TCNT1H, then TCNT1L.**
- **Sequence by reading: TCNT1L, then TCNT1H.**

```
AVR GCC:
#include <avr/io.h>

// Hereafter 16 bit access is possible:
TCNT1 = 12345;
```

Prescaler for timer 1

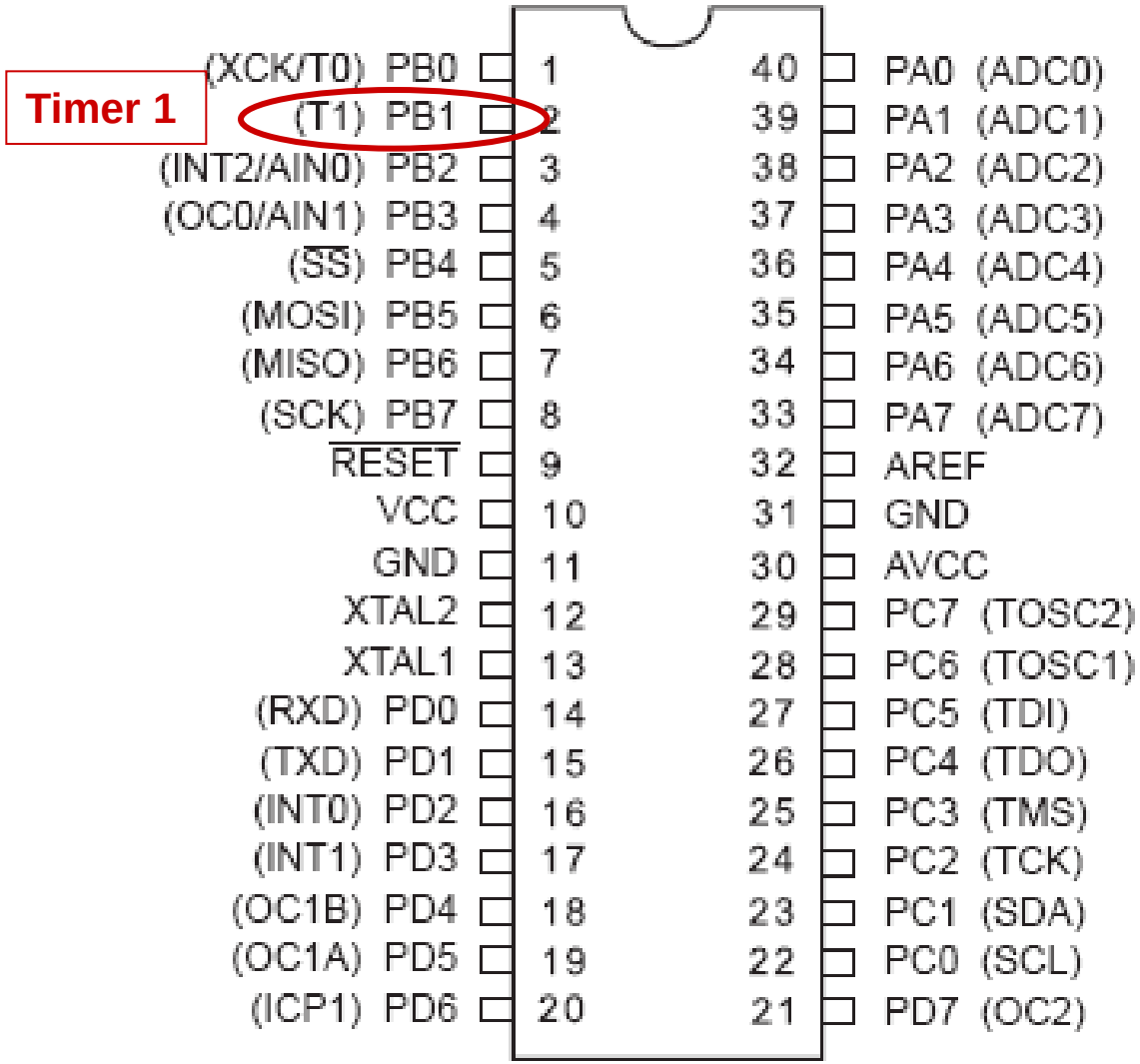


Timer 1: Clock selection

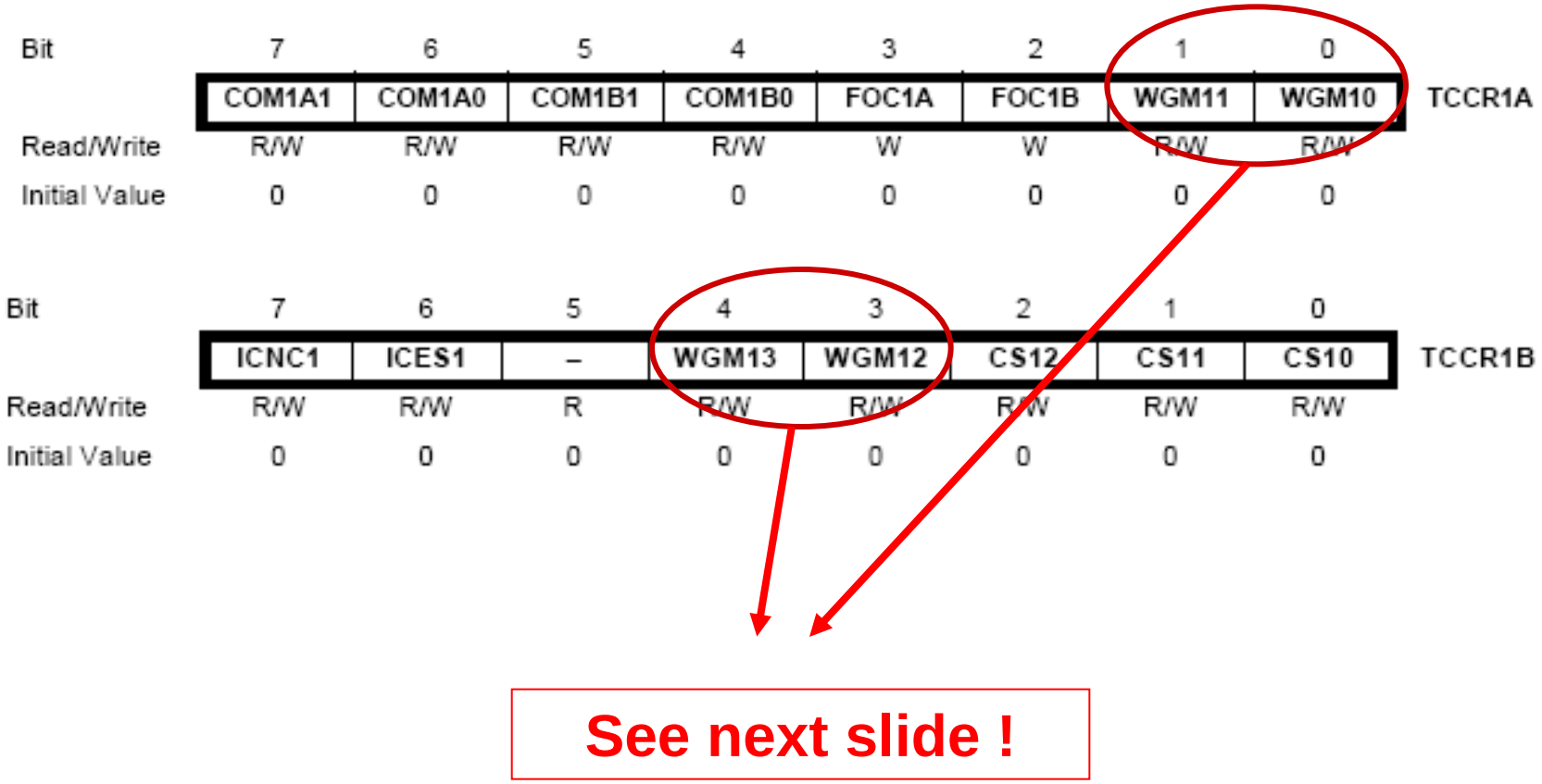
Bit	7	6	5	4	3	2	1	0	
	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	TCCR1B
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

CS12	CS11	CS10	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	$clk_{I/O}$/1 (No prescaling)
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/64 (From prescaler)
1	0	0	$clk_{I/O}$/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)
1	1	0	External clock source on T1 pin. Clock on falling edge.
1	1	1	External clock source on T1 pin. Clock on rising edge.

Timer 1: External clock (the pin T1)



Timer 1: Mode selection



See next slide !

Timer 1: Mode selection

Mode	WGM13	WGM12 (CTC1)	WGM11 (PWM11)	WGM10 (PWM10)	Timer/Counter Mode of Operation	TOP	Update of OCR1X	TOV1 Flag Set on
0	0	0	0	0	Normal	0xFFFF	Immediate	MAX
1	0	0	0	1	PWM, Phase Correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, Phase Correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, Phase Correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCR1A	Immediate	MAX
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	TOP	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	TOP	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	TOP	TOP
8	1	0	0	0	PWM, Phase and Frequency Correct	ICR1	BOTTOM	BOTTOM
9	1	0	0	1	PWM, Phase and Frequency Correct	OCR1A	BOTTOM	BOTTOM
10	1	0	1	0	PWM, Phase Correct	ICR1	TOP	BOTTOM
11	1	0	1	1	PWM, Phase Correct	OCR1A	TOP	BOTTOM
12	1	1	0	0	CTC	ICR1	Immediate	MAX
13	1	1	0	1	Reserved	–	–	–
14	1	1	1	0	Fast PWM	ICR1	TOP	TOP
15	1	1	1	1	Fast PWM	OCR1A	TOP	TOP

Timer 1 overflow flag (register TIFR)

Bit	7	6	5	4	3	2	1	0	
	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	OCF0	TOV0	TIFR
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	

- **Bit 2 in the register TIFR will be set to 1, each time Timer 1 overflows.**
- **Can be cleared again, if we *write a 1* to it (unless we use interrupts).**
- **C code:**

```
TIFR = 0b00000100;
```

or:

```
TIFR = 1<<2;
```

or:

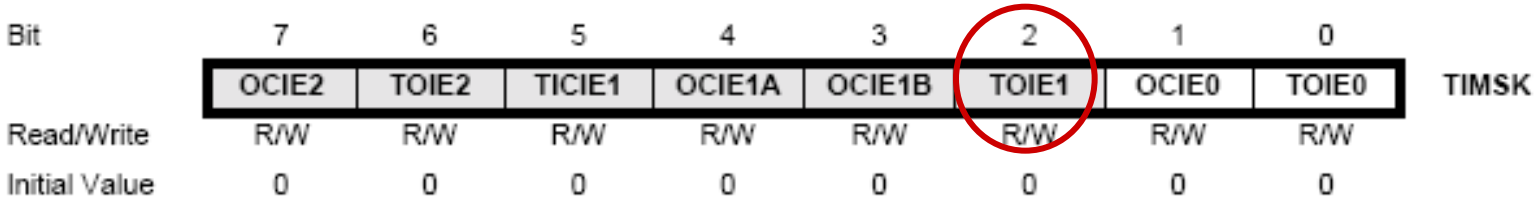
```
TIFR = 1<<TOV1;
```

(Timer 1: Overflow interrupt)

Vector No.	Program Address(2)	Source	Interrupt Definition
1	$000(1)	RESET	External Pin, Power-on Reset, Brown-out Reset, Watchdog Reset, and JTAG AVR Reset
2	$002	INT0	External Interrupt Request 0
3	$004	INT1	External Interrupt Request 1
4	$006	TIMER2 COMP	Timer/Counter2 Compare Match
5	$008	TIMER2 OVF	Timer/Counter2 Overflow
6	$00A	TIMER1 CAPT	Timer/Counter1 Capture Event
7	$00C	TIMER1 COMPA	Timer/Counter1 Compare Match A
8	$00E	TIMER1 COMPB	Timer/Counter1 Compare Match B
9	$010	TIMER1 OVF	Timer/Counter1 Overflow
10	$012	TIMER0 OVF	Timer/Counter0 Overflow
11	$014	SPI, STC	Serial Transfer Complete
12	$016	USART, RXC	USART, Rx Complete
13	$018	USART, UDRE	USART Data Register Empty
14	$01A	USART, TXC	USART, Tx Complete
15	$01C	ADC	ADC Conversion Complete
16	$01E	EE_RDY	EEPROM Ready
17	$020	ANA_COMP	Analog Comparator
18	$022	TWI	Two-wire Serial Interface
19	$024	INT2	External Interrupt Request 2
20	$026	TIMER0 COMP	Timer/Counter0 Compare Match
21	$028	SPM_RDY	Store Program Memory Ready

Timer 1 overflow interrupt

(Timer 1: Overflow interrupt enable)



Bit	7	6	5	4	3	2	1	0	
	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0	TIMSK
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	

- Bit TOIE1:

  0 => Disable o**verflow interrupt**.

  1 => Enable o**verflow interrupt**.

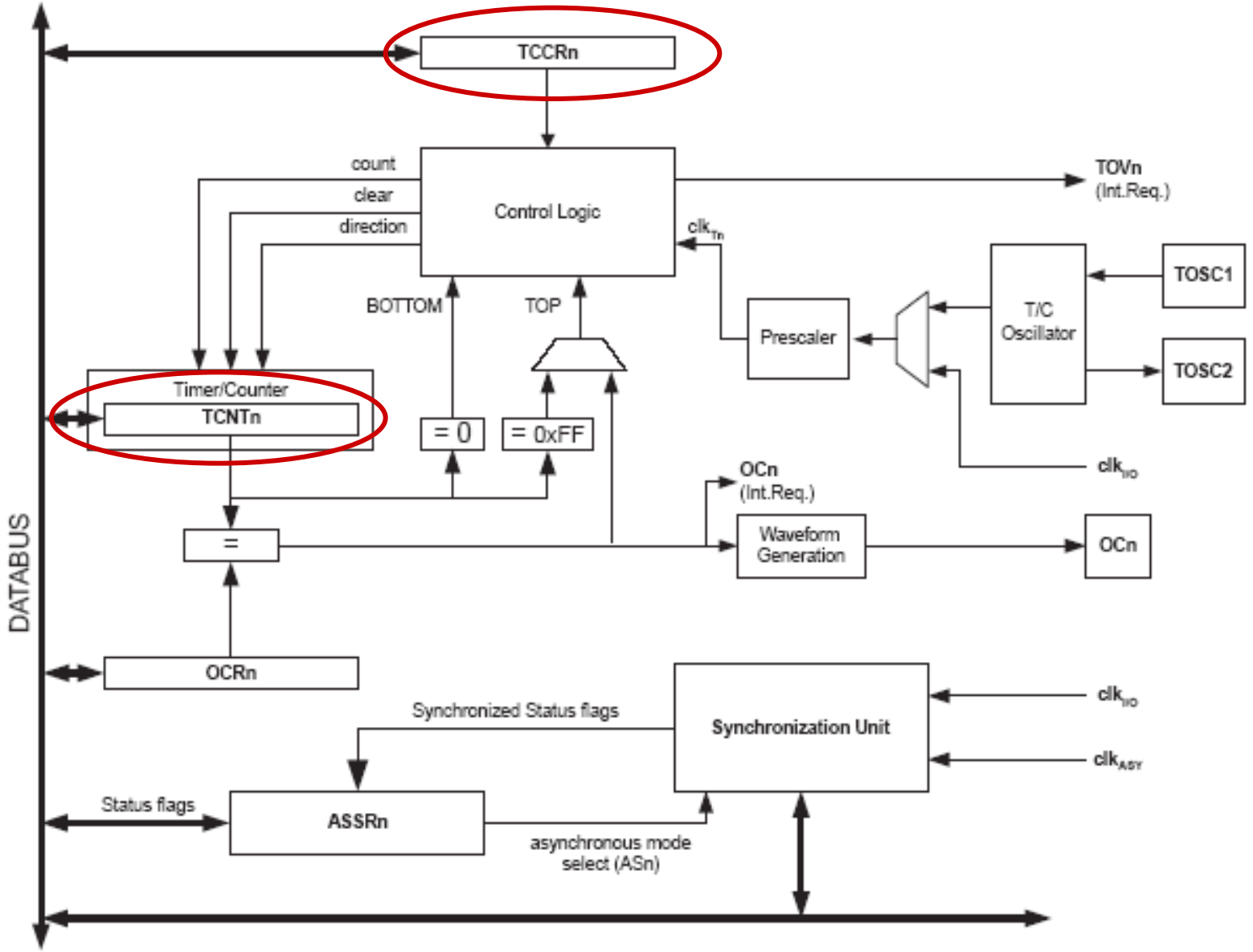
Test (”socrative.com”: Room = MSYS)

- Timer 1 is in Normal Mode and the CPU clock frequency is 16 MHz.
  What is the longest time we can have between each Timer 1 overflow ?

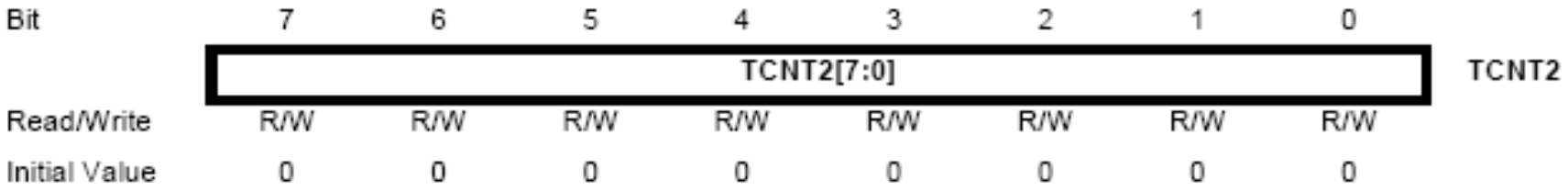
- A: About 244 seconds.
- B: 15625 seconds.
- C: About 240 ms.
- D: About 4,2 seconds.

TIMER 2

Timer 2 (8 bit)

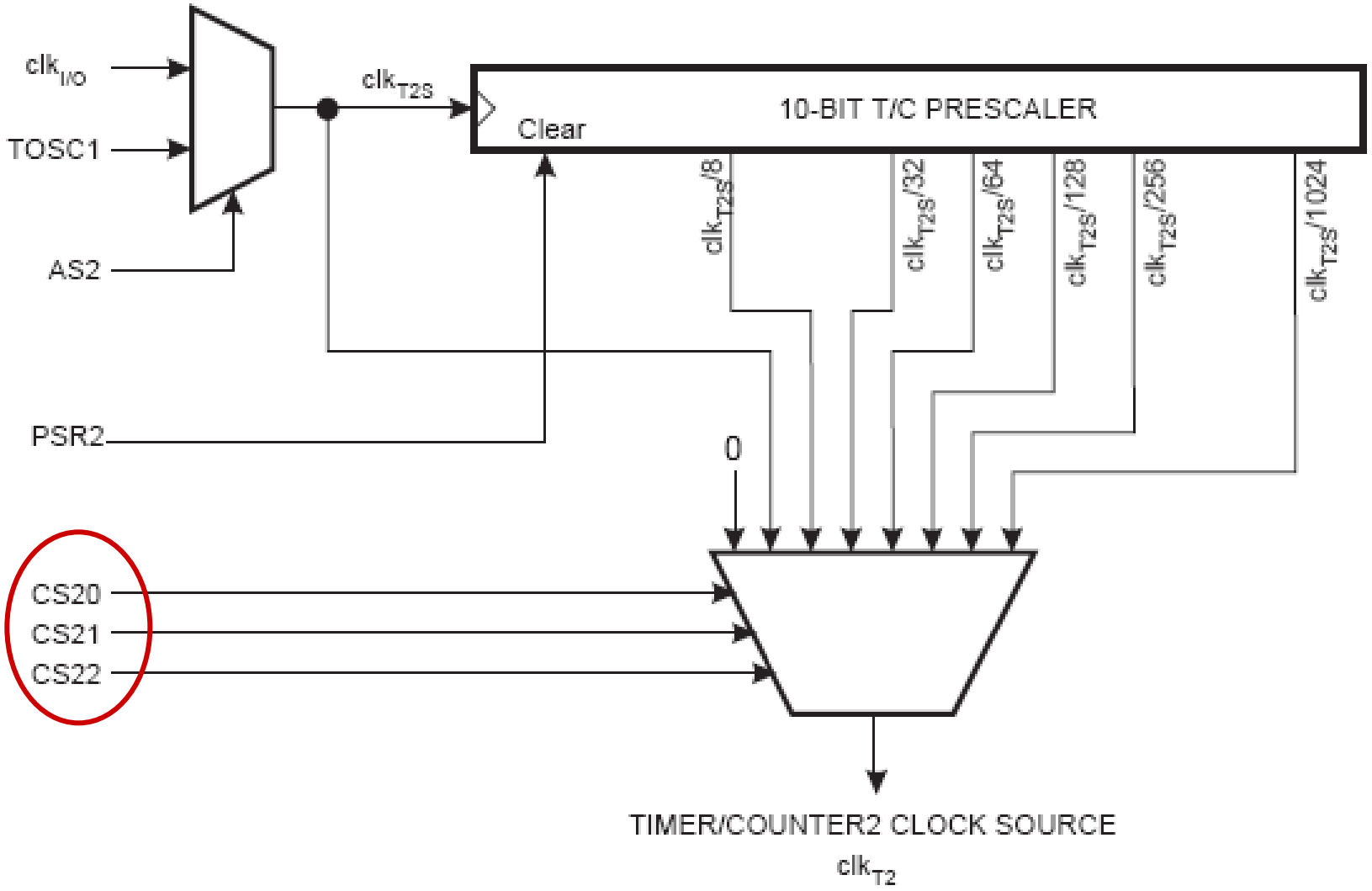


Timer 2: Counting register



- TCNT2 is the counting registeret for timer 2.
- Automatically upcounted by the timer 2 clock signal.
- Notice that the register both can be read and written (from the program).

Prescaler for timer 2

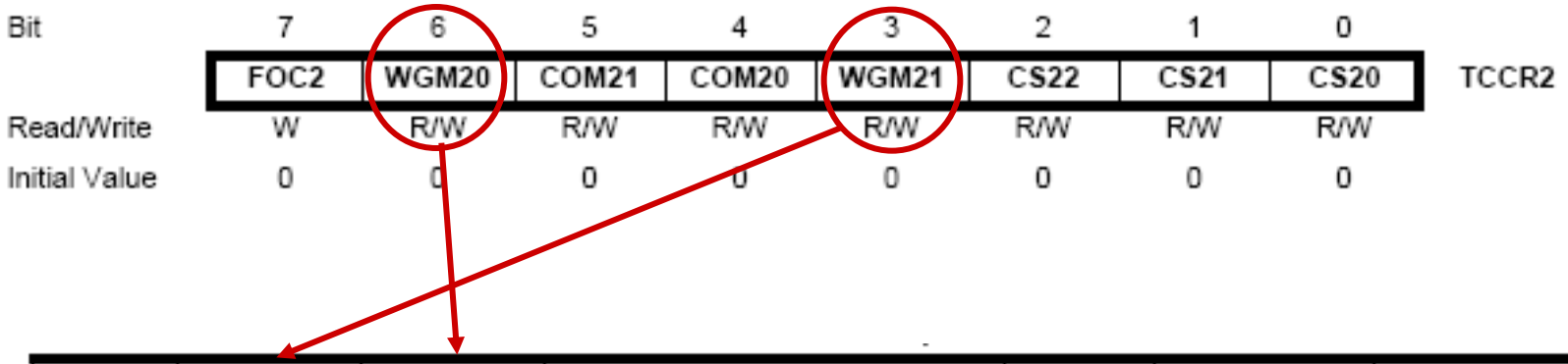


Timer 2: Clock selection

Bit	7	6	5	4	3	2	1	0	
	FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20	TCCR2
Read/Write	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	

CS22	CS21	CS20	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	clk_{T2S}/(No prescaling)
0	1	0	clk_{T2S}/8 (From prescaler)
0	1	1	clk_{T2S}/32 (From prescaler)
1	0	0	clk_{T2S}/64 (From prescaler)
1	0	1	clk_{T2S}/128 (From prescaler)
1	1	0	clk_{T2S}/256 (From prescaler)
1	1	1	clk_{T2S}/1024 (From prescaler)

Timer 2 : Mode selection



Bit	7	6	5	4	3	2	1	0	
	FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20	TCCR2
Read/Write	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	

Mode	WGM21 (CTC2)	WGM20 (PWM2)	Timer/Counter Mode of Operation	TOP	Update of OCR2	TOV2 Flag Set on
0	0	0	Normal	0xFF	Immediate	MAX
1	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	1	0	CTC	OCR2	Immediate	MAX
3	1	1	Fast PWM	0xFF	TOP	MAX

Timer 2 overflow flag (register TIFR)

Bit	7	6	5	4	3	2	1	0	
	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	OCF0	TOV0	TIFR
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	

- **Bit 6 in the register TIFR will be set to 1, each time Timer 2 overflows.**
- **Can be cleared again, if we *write a 1* to it (unless we use interrupts).**
- **C code:**

```
TIFR = 0b01000000;
```

or:

```
TIFR = 1<<6;
```

or:

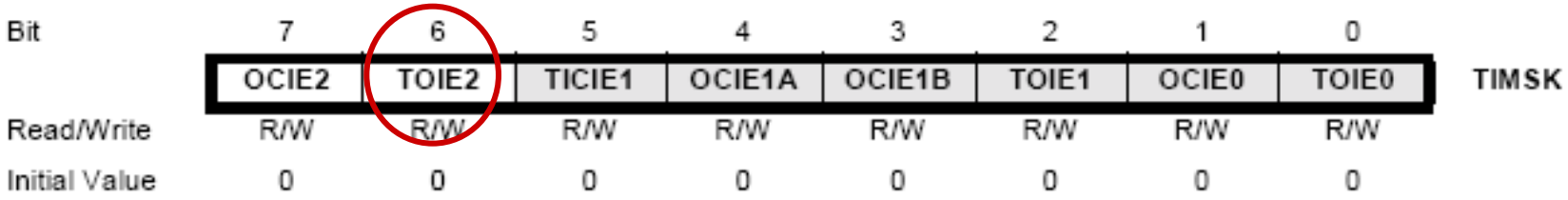
```
TIFR = 1<<TOV2;
```

(Timer 2: Overflow interrupt)

Timer 2 overflow interrupt

Vector No.	Program Address(2)	Source	Interrupt Definition
1	$000(1)	RESET	External Pin, Power-on Reset, Brown-out Reset, Watchdog Reset, and JTAG AVR Reset
2	$002	INT0	External Interrupt Request 0
3	$004	INT1	External Interrupt Request 1
4	$006	TIMER2 COMP	Timer/Counter2 Compare Match
5	$008	TIMER2 OVF	Timer/Counter2 Overflow
6	$00A	TIMER1 CAPT	Timer/Counter1 Capture Event
7	$00C	TIMER1 COMPA	Timer/Counter1 Compare Match A
8	$00E	TIMER1 COMPB	Timer/Counter1 Compare Match B
9	$010	TIMER1 OVF	Timer/Counter1 Overflow
10	$012	TIMER0 OVF	Timer/Counter0 Overflow
11	$014	SPI, STC	Serial Transfer Complete
12	$016	USART, RXC	USART, Rx Complete
13	$018	USART, UDRE	USART Data Register Empty
14	$01A	USART, TXC	USART, Tx Complete
15	$01C	ADC	ADC Conversion Complete
16	$01E	EE_RDY	EEPROM Ready
17	$020	ANA_COMP	Analog Comparator
18	$022	TWI	Two-wire Serial Interface
19	$024	INT2	External Interrupt Request 2
20	$026	TIMER0 COMP	Timer/Counter0 Compare Match
21	$028	SPM_RDY	Store Program Memory Ready

(Timer 2: Overflow interrupt enable)



Bit	7	6	5	4	3	2	1	0	
	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0	TIMSK
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	

- Bit TOIE2:

  0 => Disable o**verflow interrupt**.

  1 => Enable o**verflow interrupt**.

Test (”socrative.com”: Room = MSYS)

- A Mega32 CPU clock frequency is 3,6864 MHz. How do we initialize Timer 2 to normal mode, so that we will have a Timer 2 overflow interrupt 14400 times per second ?

- A: TCCR2 = 0b00000001;
- B: TCCR2 = 0b00000110;
- C: TCCR2 = 0b00000100;
- D: TCCR2 = 0b00000000;

Example: Timer 0

Example 9-39

Write a C program to toggle all the bits of PORTB continuously with some delay. Use Timer0, Normal mode, and no prescaler options to generate the delay.

Solution:

```
#include "avr/io.h"
void T0Delay ( );
int main ( )
{
      DDRB = 0xFF;       //PORTB output port

      while (1)
      {
            PORTB = 0x55;      //repeat forever
            T0Delay ( );       //delay size unknown
            PORTB = 0xAA;      //repeat forever
            T0Delay ( );
      }
}

void T0Delay ( )
{
      TCNT0 = 0x20;            //load TCNT0
      TCCR0 = 0x01;            //Timer0, Normal mode, no prescaler
      while ((TIFR&0x1)==0);   //wait for TF0 to roll over
      TCCR0 = 0;
      TIFR = 0x1;              //clear TF0
}
```

Example: Timer 1

Example 9-42 (C version of Example 9-32)

Write a C program to toggle only the PORTB.4 bit continuously every second. Use Timer1, Normal mode, and 1:256 prescaler to create the delay. Assume XTAL = 8 MHz.

Solution:

XTAL = 8 MHz ➔ $T_{machine\ cycle}$ = 1/8 MHz = 0.125 µs = T_{clock}

Prescaler = 1:256 ➔ T_{clock} = 256 × 0.125 µs = 32 µs

1 s/32 µs = 31,250 clocks = 0x7A12 clocks ➔ 1 + 0xFFFF – 0x7A12 = **0x85EE**

```
#include "avr/io.h"

void T1Delay ( );

int main ( )
{
      DDRB = 0xFF;       //PORTB output port

      while (1)
      {
            PORTB = PORTB ^ (1<<PB4); //toggle PB4
            T1Delay ( );       //delay size unknown
      }
}

void T1Delay ( )
{
      TCNT1H = 0x85;     //TEMP = 0x85
      TCNT1L = 0xEE;

      TCCR1A = 0x00;     //Normal mode
      TCCR1B = 0x04;     //Normal mode, 1:256 prescaler

      while ((TIFR&(0x1<<TOV1))==0);      //wait for Tf0 to roll over

      TCCR1B = 0;
      TIFR = 0x1<<TOV1;        //clear TOV1
}
```

End of lesson 13