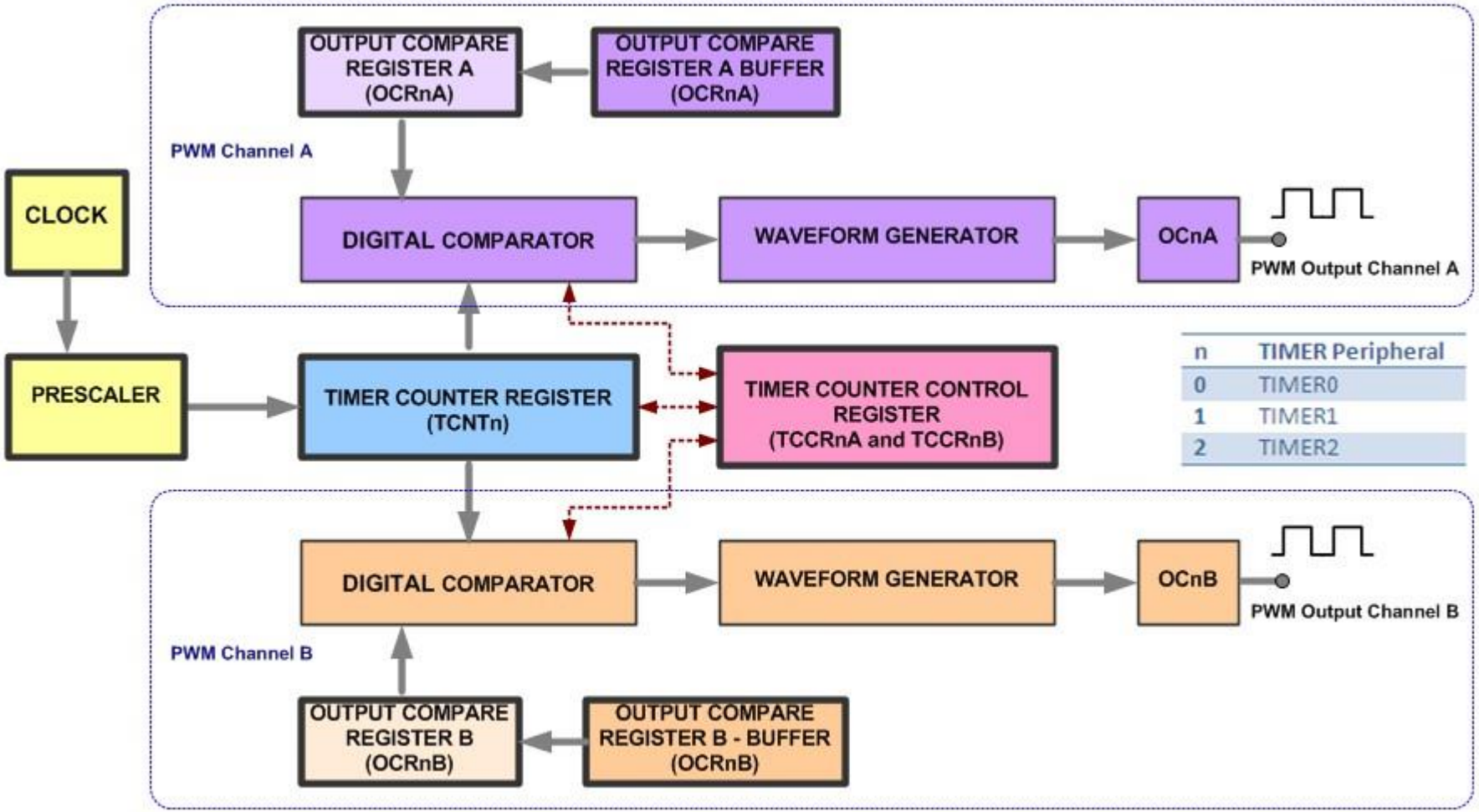
Вбудовані системи

Таймери лічильники

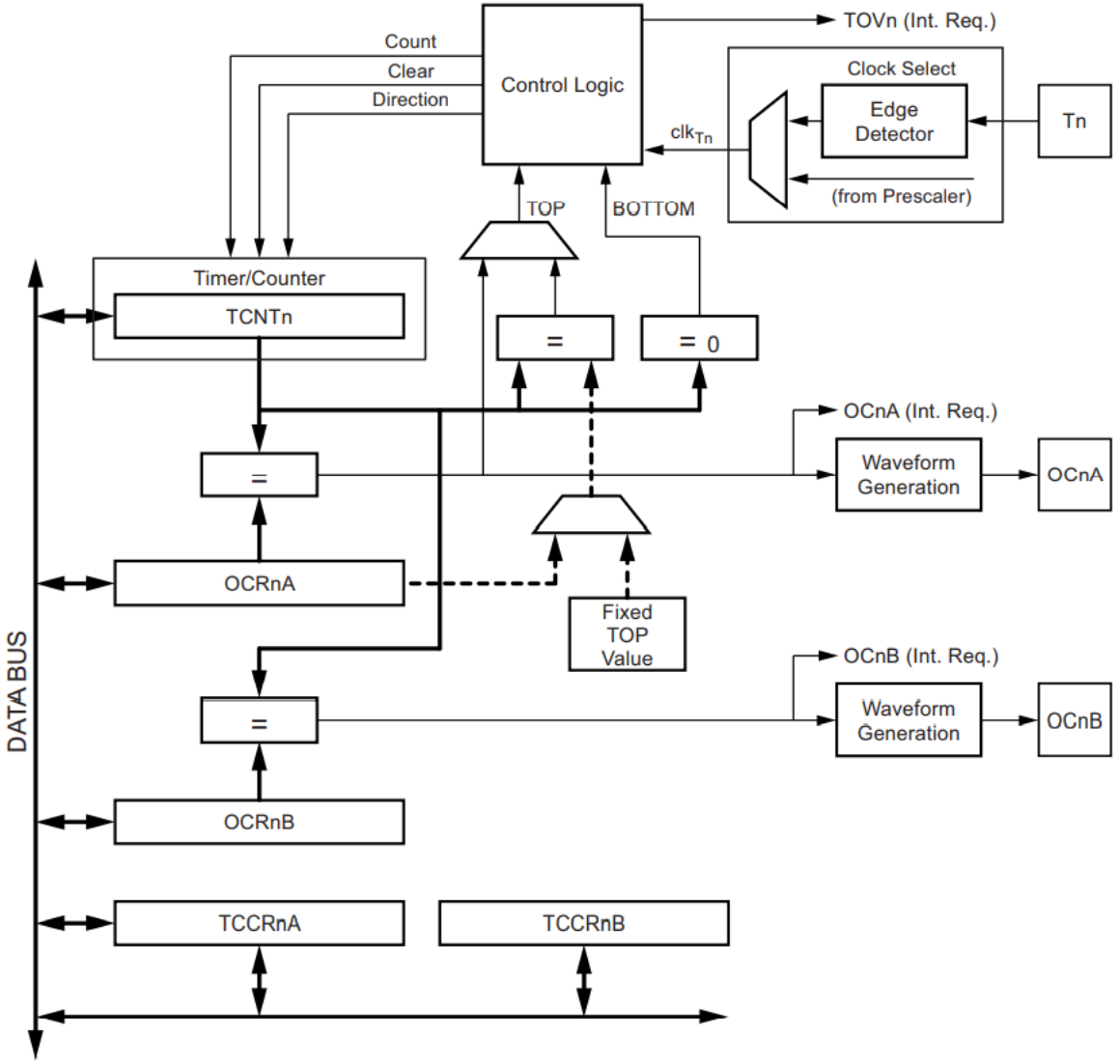
Блок-схема таймера/лічильника з схемою широтно-імпульсної модуляції



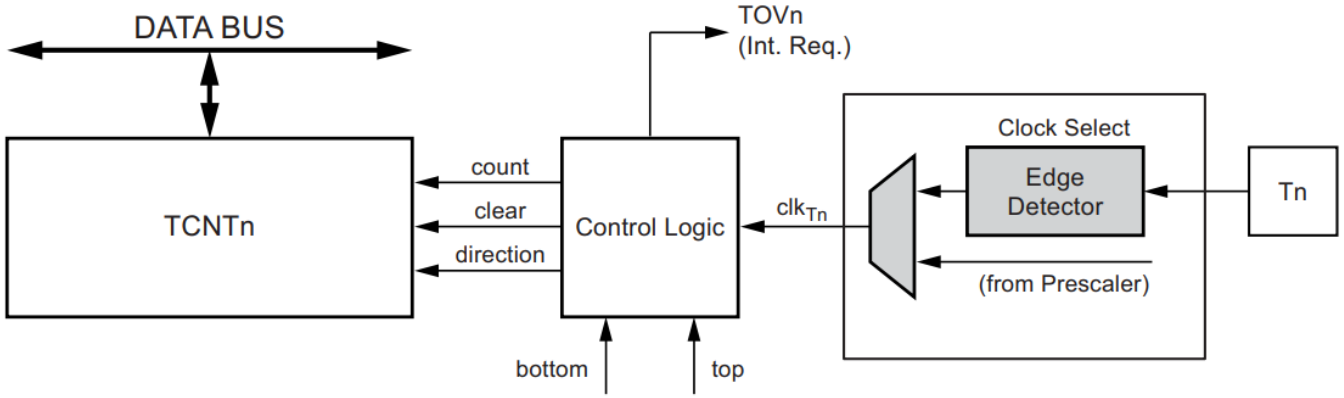
Simplified Atmel AVR ATMega48/88/168/328 Microcontroller PWM Peripheral

Таймер/лічильник 0 мікроконтролера ATMega328

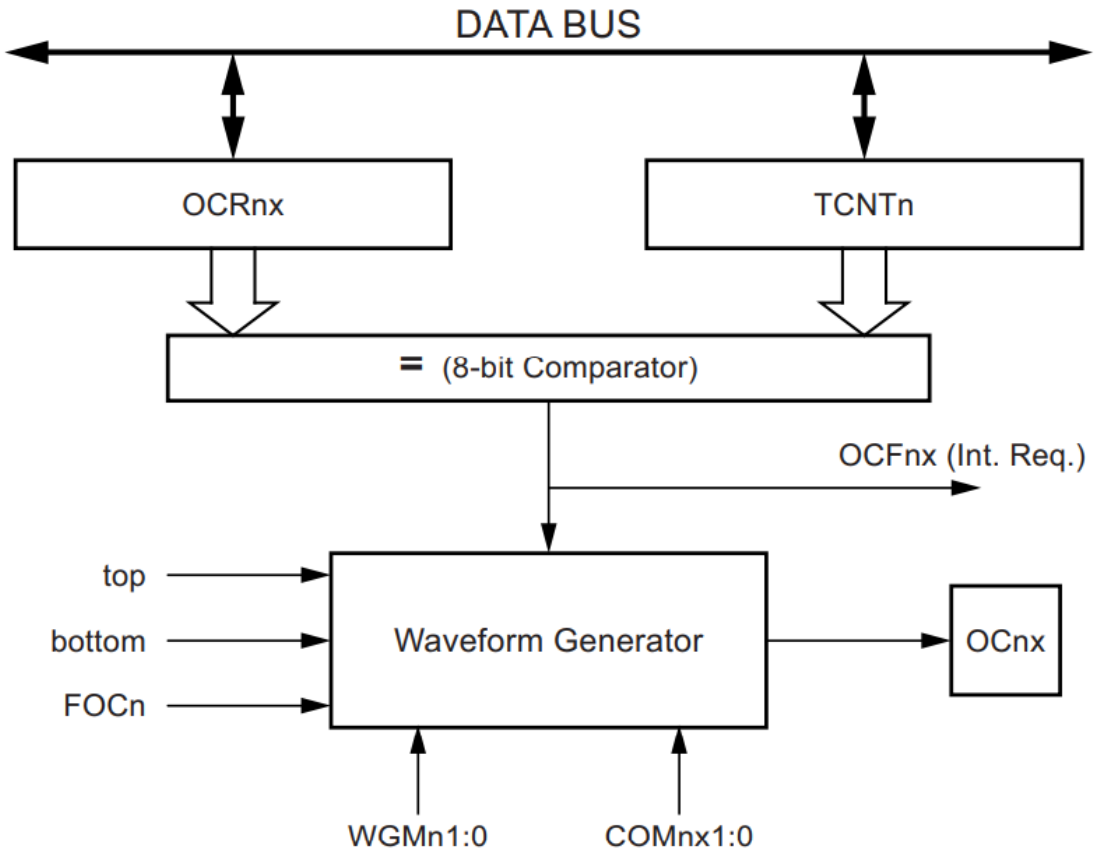
Блок-діаграма таймера/лічильника 0 (8-bit Timer/Counter0)



Блок-діаграма модуля лічильника таймера 0



Блок-діаграма модуля компаратора таймера 0



Регістри управління таймером/лічильником 0

Bit	7	6	5	4	3	2	1	0	
0x24 (0x44)	COM0A**1**	COM0A**0**	COM0B**1**	COM0B**0**	–	–	**WGM01**	**WGM00**	TCCR0A
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	

Bit	7	6	5	4	3	2	1	0	
0x25 (0x45)	FOC0A	FOC0B	–	–	**WGM02**	**CS02**	**CS01**	**CS00**	TCCR0B
Read/Write	W	W	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

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	

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	

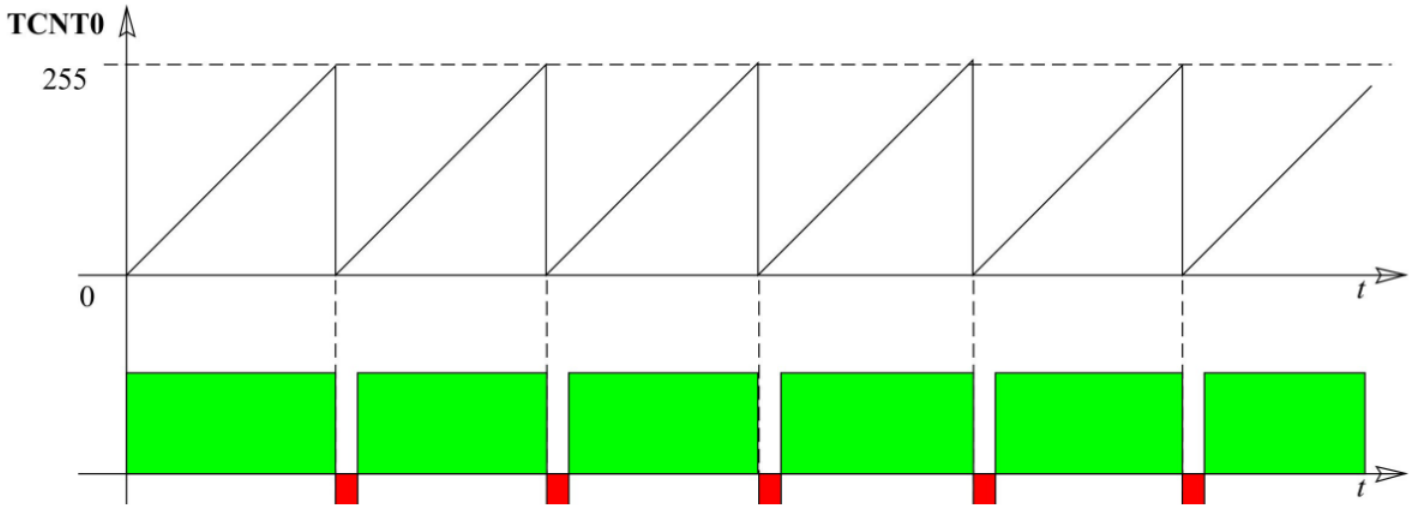
Режими роботи таймера/лічильника

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

MAX = 0xFF BOTTOM = 0x00

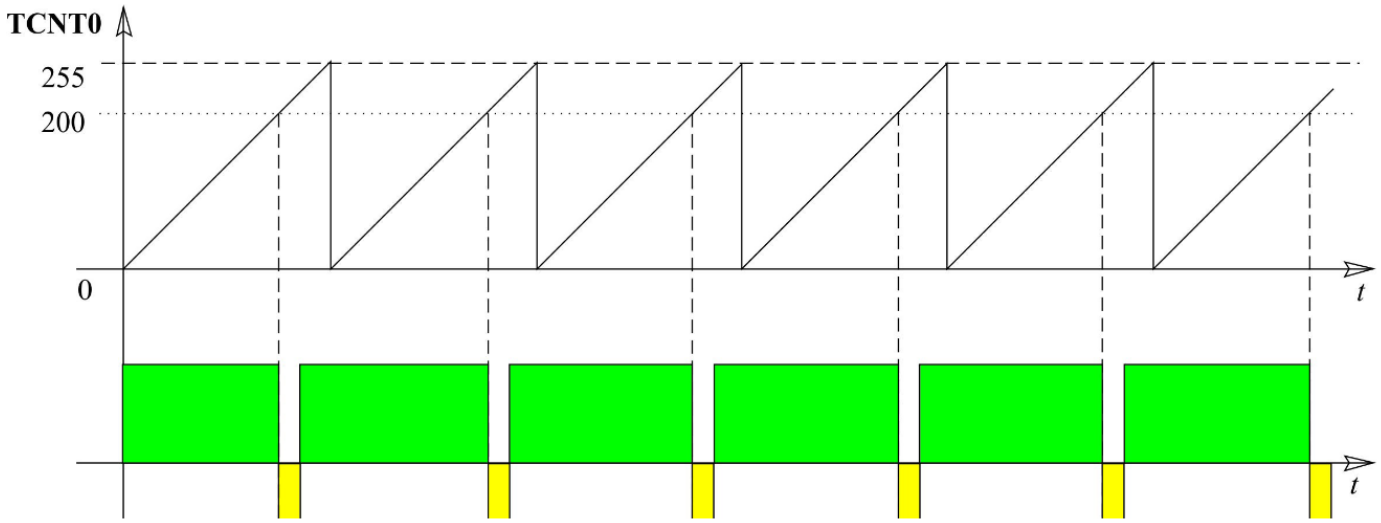
CS02	CS01	CS00	Description
0	0	0	No clock source (Timer/Counter stopped)
0	0	1	$clk_{I/O}$/(no prescaling)
0	1	0	$clk_{I/O}$/8 (from prescaler)
0	1	1	$clk_{I/O}$/64 (from prescaler)
1	0	0	$clk_{I/O}$/256 (from prescaler)
1	0	1	$clk_{I/O}$/1024 (from prescaler)
1	1	0	External clock source on T0 pin. Clock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

Біти вибору джерела тактових імпульсів та подільника частоти



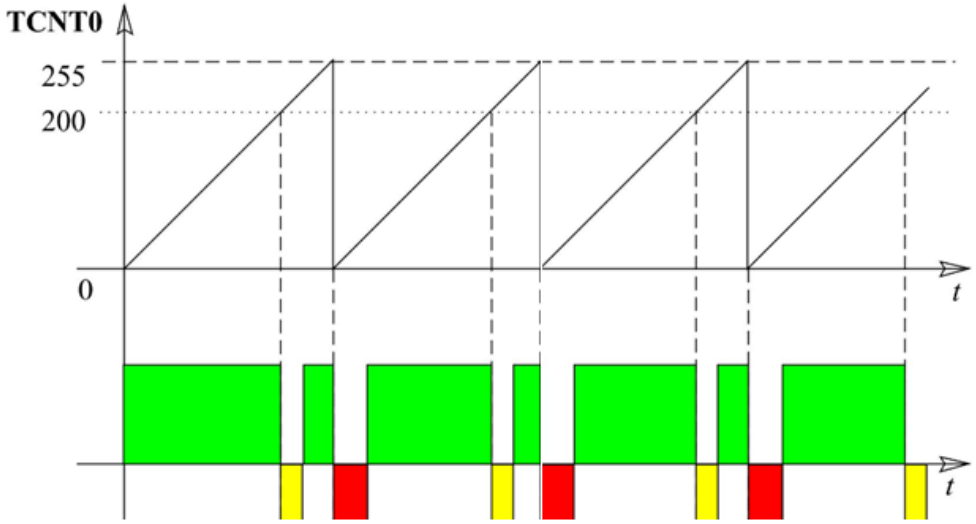
Виконання основної програми

Виконання процедури обробки переривання по переповненню таймера



Виконання основної програми

Виконання процедури обробки переривання таймера по співпадінню (OCR=200)



Використання таймера 0 для реалізації функції millis()

```
#define MICSEC_T0_OVF ((64 * 256)/(F_CPU/1000000L))
#define MILLIS_INC (MICSEC_T0_OVF / 1000)
#define FRACT_INC ((MICSEC_T0_OVF % 1000) >> 3)
#define FRACT_MAX (1000 >> 3)

volatile uint32_t timer0_millis = 0;
static uint8_t timer0_fract = 0;

ISR(TIMER0_OVF_vect)
{
  uint32_t m = timer0_millis;
  uint8_t f = timer0_fract;
    m += MILLIS_INC;
    f += FRACT_INC;
    if(f >= FRACT_MAX) {
        f -= FRACT_MAX;
        m += 1;
    }
    timer0_fract = f;
    timer0_millis = m;
}
```

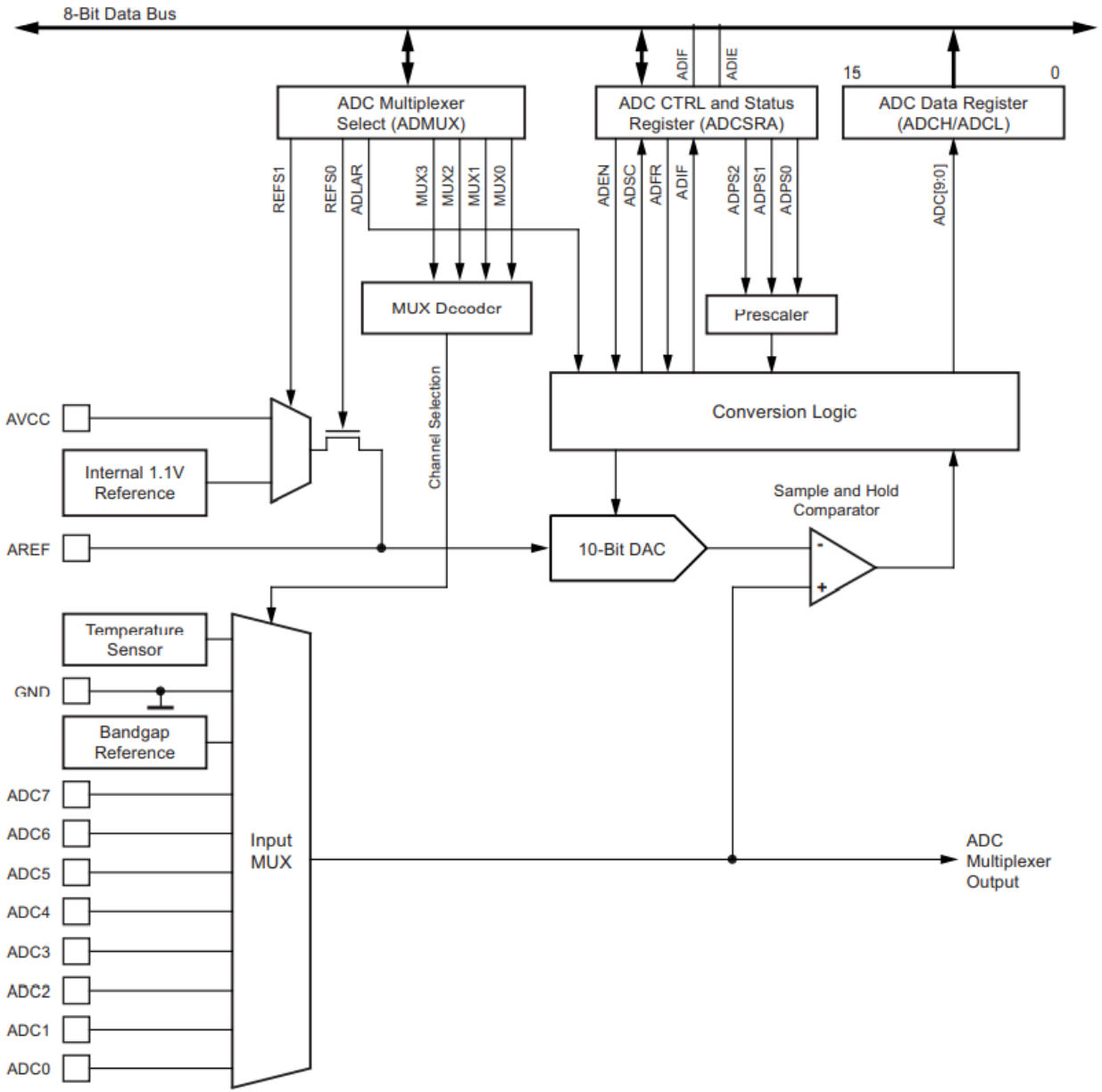
```
//продовження

uint32_t millis(void)
{
  uint32_t m;
  uint8_t oldSREG = SREG;
    cli();
    m = timer0_millis;
    SREG = oldSREG;
  return m;
}

void InitTimer0(void)
{
  // fast pwm timer 0
  TCCR0A = (1<<WGM01)|(1<<WGM00);
  TCCR0B = (1<<CS01)|(1<<CS00);
  TIMSK0 = 1<<TOIE0;
  sei();
}
```

Приклад використання millis

```
#include <avr/io.h>
// підключаємо бібліотеку з функцією millis()
#include "my_millis.h"

int main(void)
{
  uint32_t tmp, delay1_millis;
    InitTimer0();
    delay1_millis = millis();
    DDRB |= 1<<PB5;

    for(;;)
    {
        tmp = millis();
        if((uint32_t)(tmp - delay1_millis) >= 1000) {
            delay1_millis = tmp;
            PORTB ^= 1<<PB5;
        }
       // ...
    }
  return 0;
}
```

Аналогово-цифровий перетворювач (ADC)

Аналогово-цифровий перетворювач мікроконтролера ATMega328



Основні параметри АЦП:

- Роздільна здатність 10-біт
- Інтегральна нелінійність 0.5 LSB
- Абсолютна точність ± 2 LSB
- Час перетворення від 65 до 260 мкс
- 8 мультиплексованих вхідних каналів
- Вхідний канал вбудованого датчика температури
- Вибір опорної напруги АЦП 1.1 В
- Постійний режим роботи або режим одиночного перетворення
- Переривання АЦП при завершені перетворення
- Пониження шуму в режимі сну

$$ADC = \frac{V_{IN} \times 1024}{V_{REF}}$$

Регістри керування АЦП

Регістр вибору входу мультиплексора

Bit	7	6	5	4	3	2	1	0	
(0x7C)	REFS1	REFS0	ADLAR	–	MUX3	MUX2	MUX1	MUX0	ADMUX
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Біти вибору джерела опорної напруги

REFS1	REFS0	Voltage Reference Selection
0	0	AREF, internal V_{REF} turned off
0	1	AV_{CC} with external capacitor at AREF pin
1	0	Reserved
1	1	Internal 1.1V voltage reference with external capacitor at AREF pin

Регістр вибору джерела запуску АЦП перетворення

Bit	7	6	5	4	3	2	1	0	
(0x7B)	–	ACME	–	–	–	ADTS2	ADTS1	ADTS0	ADCSRB
Read/Write	R	R/W	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

MUX3..0	Single Ended Input
0000	ADC0
0001	ADC1
0010	ADC2
0011	ADC3
0100	ADC4
0101	ADC5
0110	ADC6
0111	ADC7
1000	ADC8(1)
1001	(reserved)
1010	(reserved)
1011	(reserved)
1100	(reserved)
1101	(reserved)
1110	1.1V (V_{BG})
1111	0V (GND)

Регістр управління АЦП A (ADCSRA)

Bit	7	6	5	4	3	2	1	0	
(0x7A)	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
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	

ADEN (7) - вмикає або вимикає АЦП (1-включений).

ADSC (6) - запускає перетворення якщо в нього записати 1 (автоматично скидається після завершення перетворення).

ADATE (5) - дозволяє запускати перетворення по перериванню від периферійних пристроїв мікроконтролера якщо встановити в 1.

ADIF (4) - прапор переривання від АЦП.

ADIE (3) - дозвіл переривання від АЦП якщо встановлений в 1.

Біти **ADPS2, ADPS1, ADPS0** (2 - 0) вибирають режим роботи подільника тактової частоти (робоча частота АЦП від 50kHz до 200kHz):

000 - CLK / 2
001 - CLK / 2
010 - CLK / 4
011 - CLK / 8
100 - CLK / 16
101 - CLK / 32
110 - CLK / 64
111 - CLK / 128

ADTS2	ADTS1	ADTS0	Trigger Source
0	0	0	Free running mode
0	0	1	Analog comparator
0	1	0	External interrupt request 0
0	1	1	Timer/Counter0 compare match A
1	0	0	Timer/Counter0 overflow
1	0	1	Timer/Counter1 compare match B
1	1	0	Timer/Counter1 overflow
1	1	1	Timer/Counter1 capture event

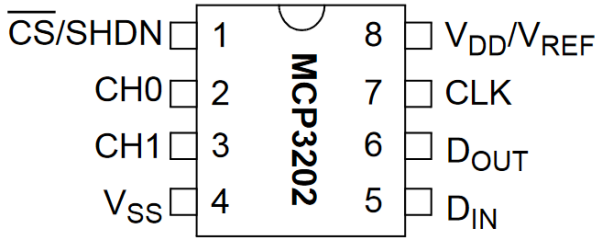
Вибір джерела запуску АЦП перетворення (регістр **ADCSRB**)

Прилад використання АЦП

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

int main(void)
{
  uint32_t data;

    // Задаємо джерело опорної напруги AVCC =3.3V та вибираємо вхід ADC1
    ADMUX = (1<<REFS0)|(1<<MUX0);
    // Включаємо АЦП та задаємо подільник частоти CLK/128
    ADCSRA= (1<<ADEN)|(1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0);

    for(;;)
    {
        // Запускаємо перетворення АЦП
        ADCSRA |= (1<<ADSC);
        // Чекаємо закінчення перетворення АЦП
        while (ADCSRA & (1<<ADSC));
        // Зчитуємо результат перетворення (ADCL | (ADCH << 8));
        data = (ADCW * 330) / 1024;
        // printf("V= %d.%d", data/100, data%100); // Вивід даних
       _delay_ms(1000);
    }
  return 0;
}
```

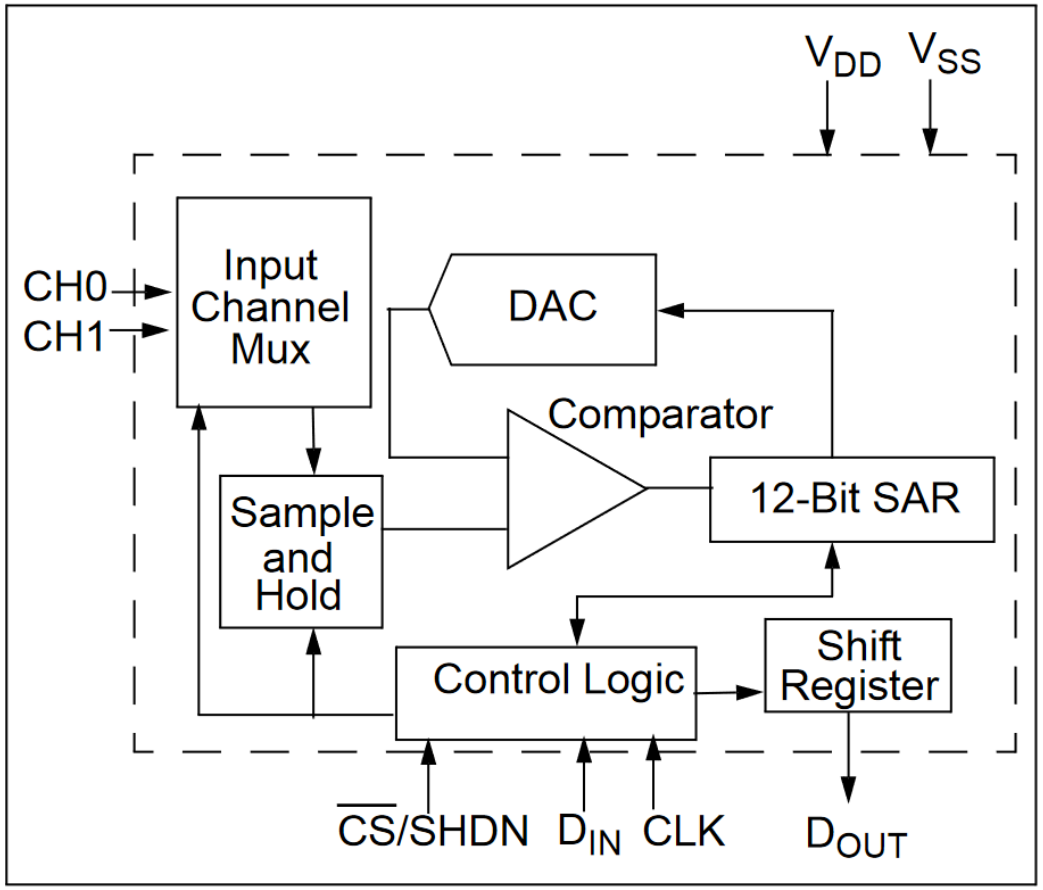
MCP3202 - 12 bit ADC

$$Digital\ Output\ Code = \frac{4096 \cdot V_{IN}}{V_{DD}}$$



	Config Bits		Channel Selection	
	C1	C2	0	1
Single Ended Mode	1	0	+	—
	1	1	—	+
Pseudo-Differential Mode	0	0	IN+	IN-
	0	1	IN-	IN+

Функціональна схема мікросхеми



Часова діаграма протоколу передачі даних