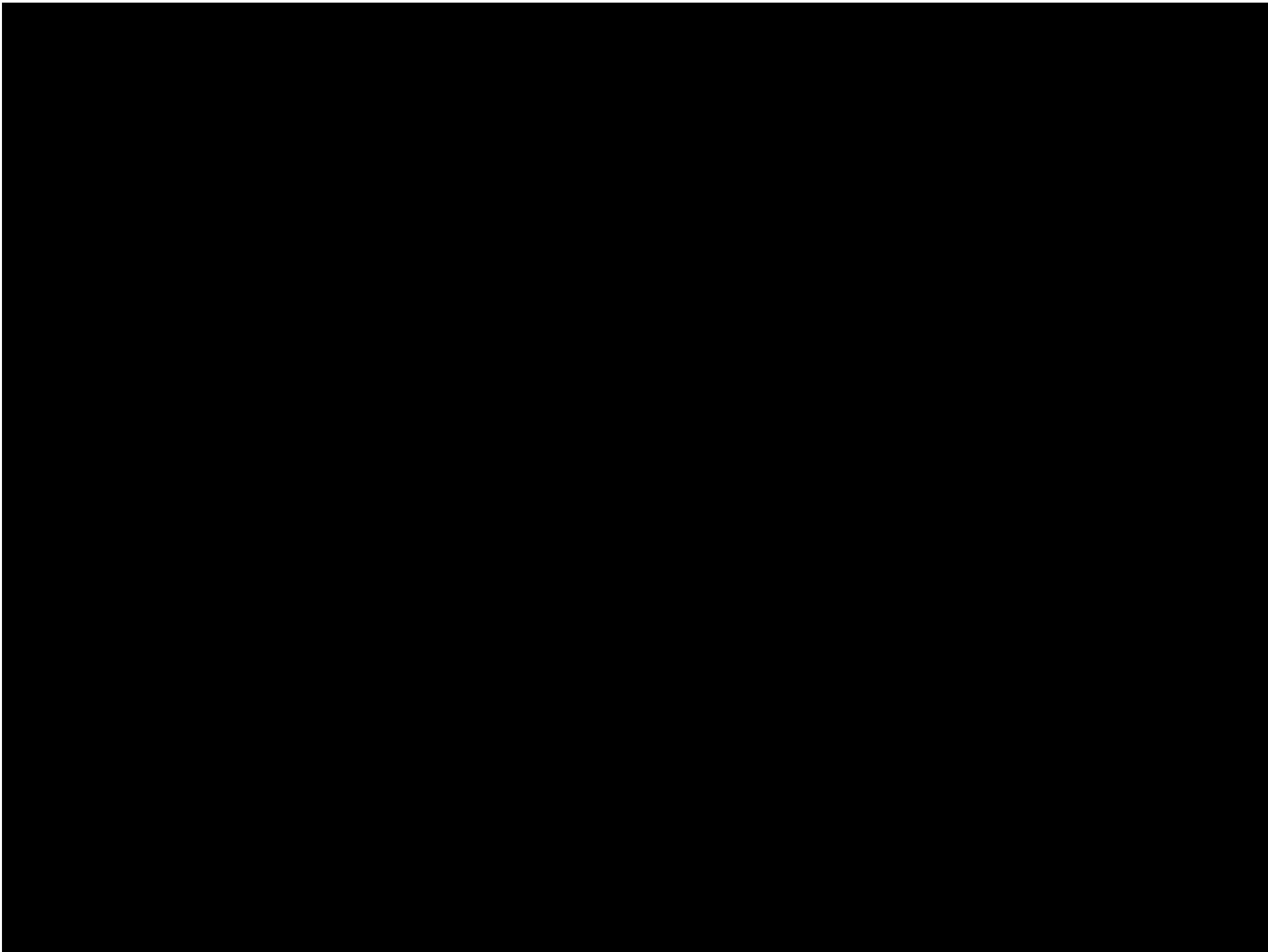
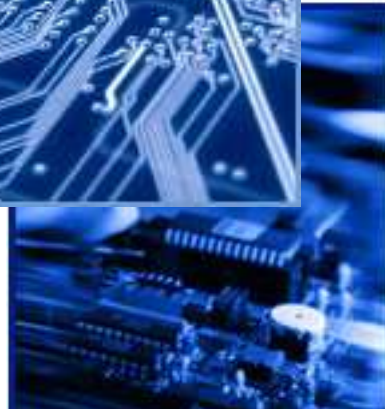
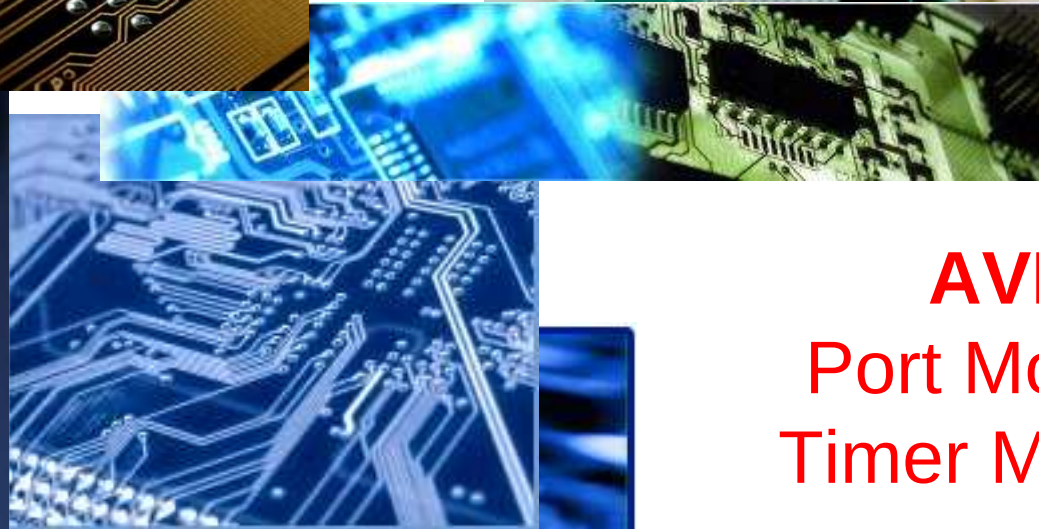
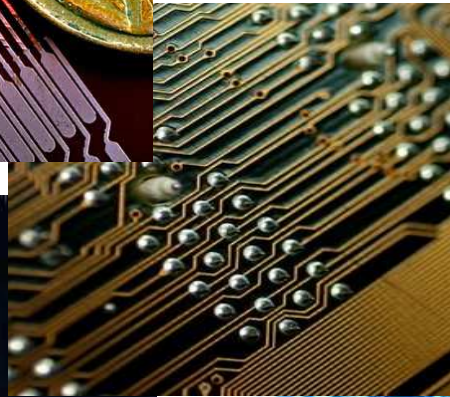
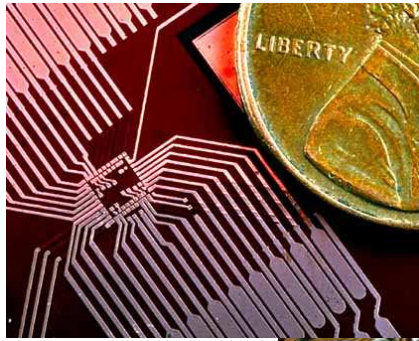






AVR **Port Module** **Timer Module**

Muhammad Kamal





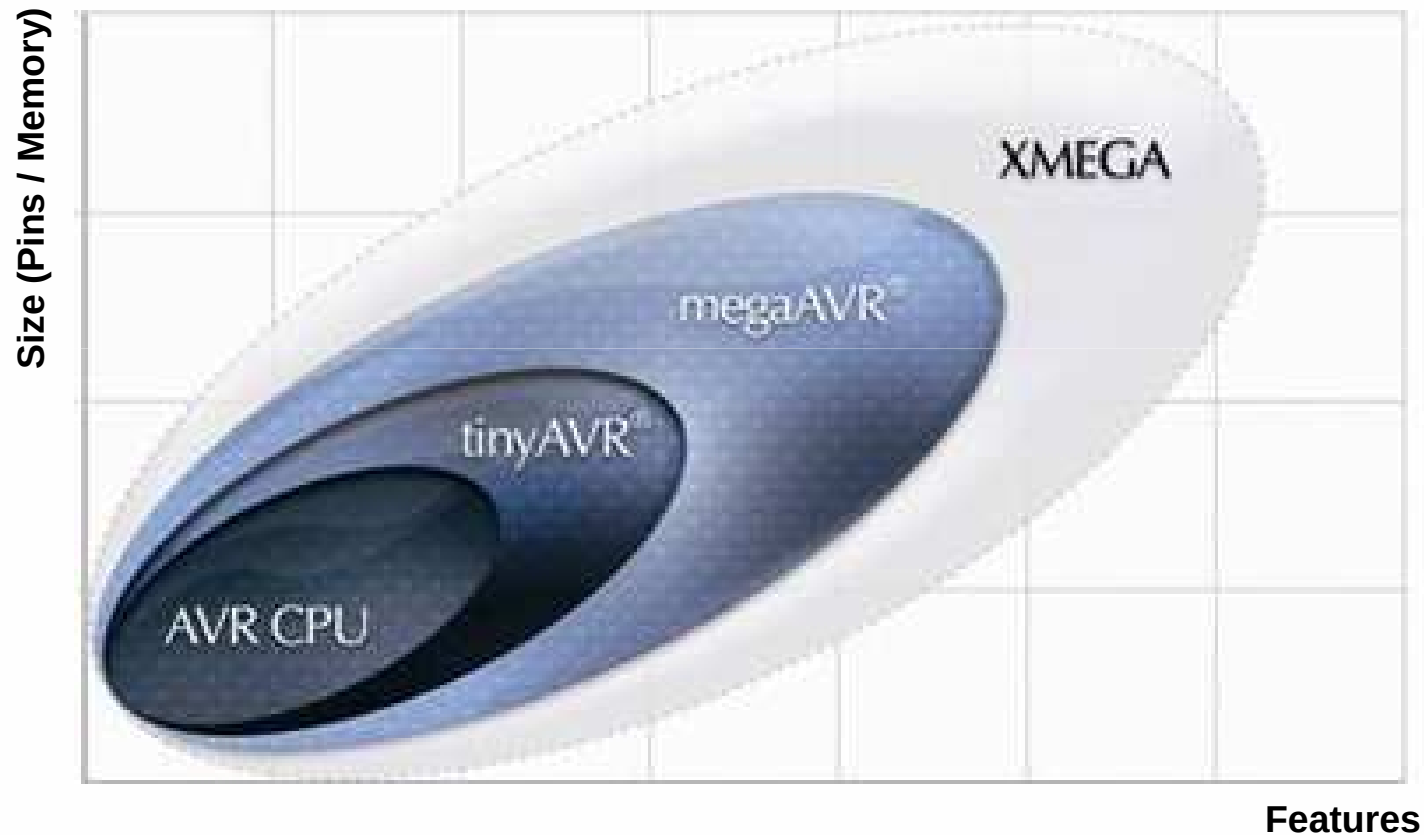
Objectives

- To be familiar with “AVR studio” IDE
- Understand and program the AVR peripherals as DIO, Timer, and Interrupt circuitry
- Know how to download your program on the AVR controller

Agenda

- DIO Module
- How to work with the DIO module ?
- AVR Studio IDE & WinAVR
- Interrupts
- Timer Module
- How to work with the Timer module ?
- Download your program
- Lab

AVR Technology

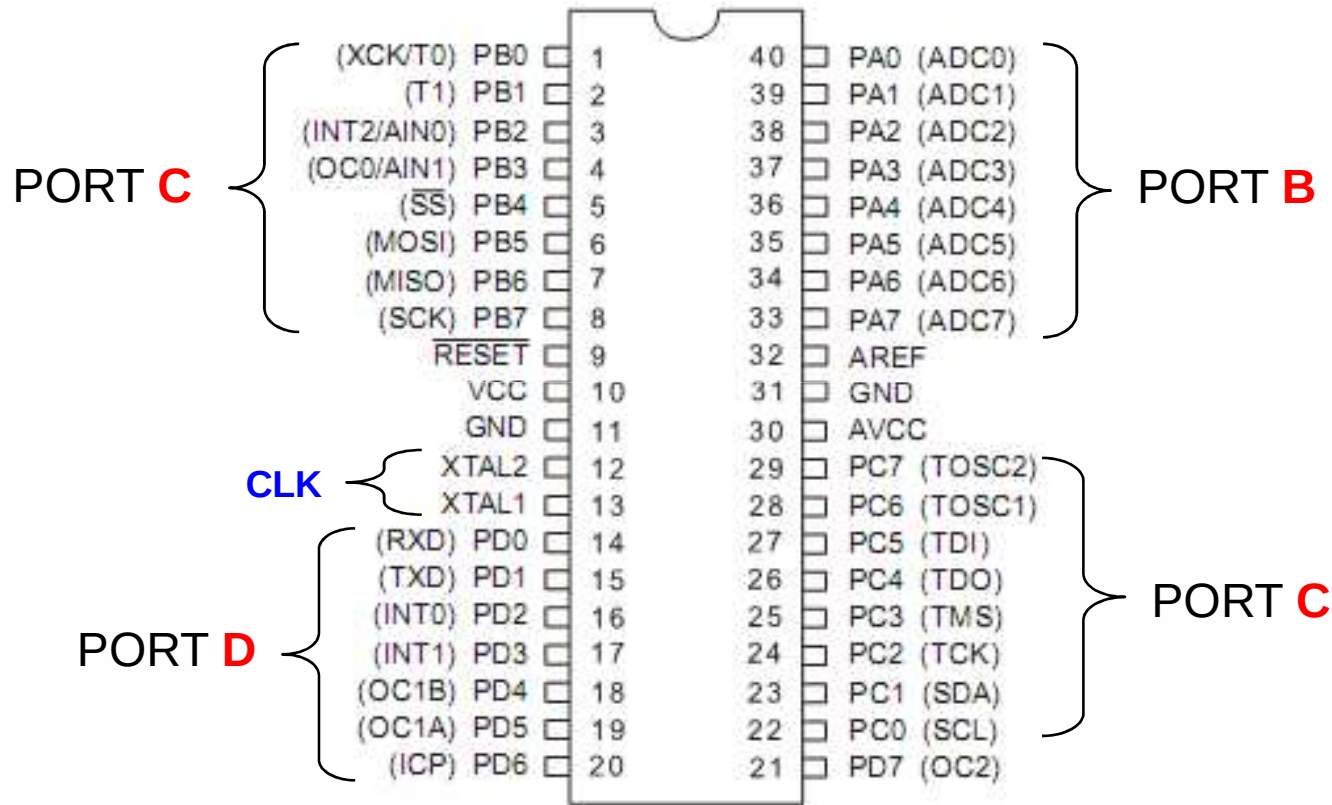


DIO Module

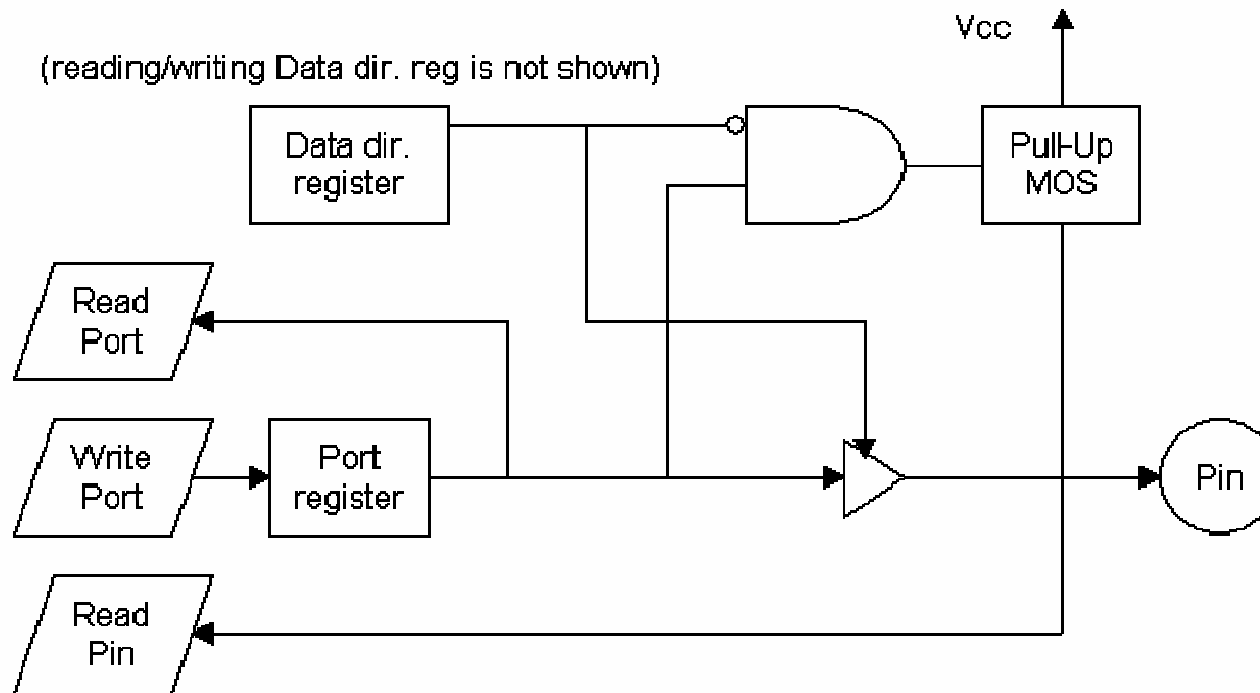
DIO

- ATmega32 have **32** Programmable I/O Lines.
- These lines are divided into **4** Ports:
 - PORT **A** (**PA7..PA0**)
 - PORT **B** (**PB7..PB0**)
 - PORT **C** (**PC7..PC0**)
 - PORT **D** (**PD7..PD0**)
- Each port consist of **3** registers:
 - DDRx (Data Direction Register)
 - PORTx (Output Register)
 - PINx (Input Register)
- Most port pins have alternate functions in addition to being general digital I/Os

ATmega32 Pin out



I/O Pin equivalent Diagram



Register Description for I/O Port

Port A Data Register – PORTA

Bit	7	6	5	4	3	2	1	0	
	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	PORTA
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	

Port A Data Direction Register – DDRA

Bit	7	6	5	4	3	2	1	0	
	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	DDRA
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	

Port A Input Pins Address – PINA

Bit	7	6	5	4	3	2	1	0	
	PINA7	PINA6	PINA5	PINA4	PINA3	PINA2	PINA1	PINA0	PINA
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

Programming DIO

1. Decide number of input lines and output lines that will be used
2. Configure the ports direction in the register DDRx
3. Use register PORTx in case of writing (O)
4. Use register PINx in case of reading (I)



AVR Studio IDE



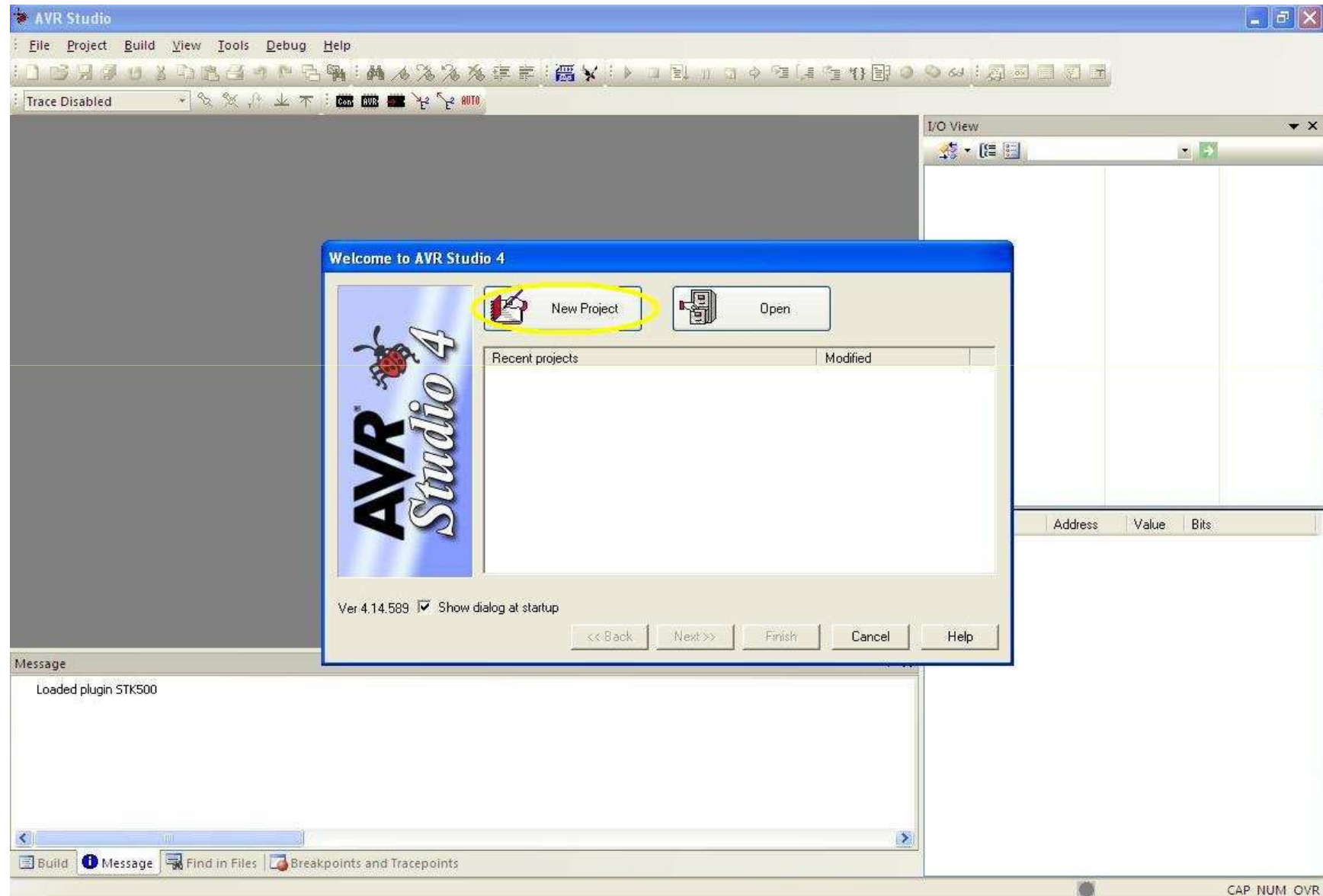
AVR Studio



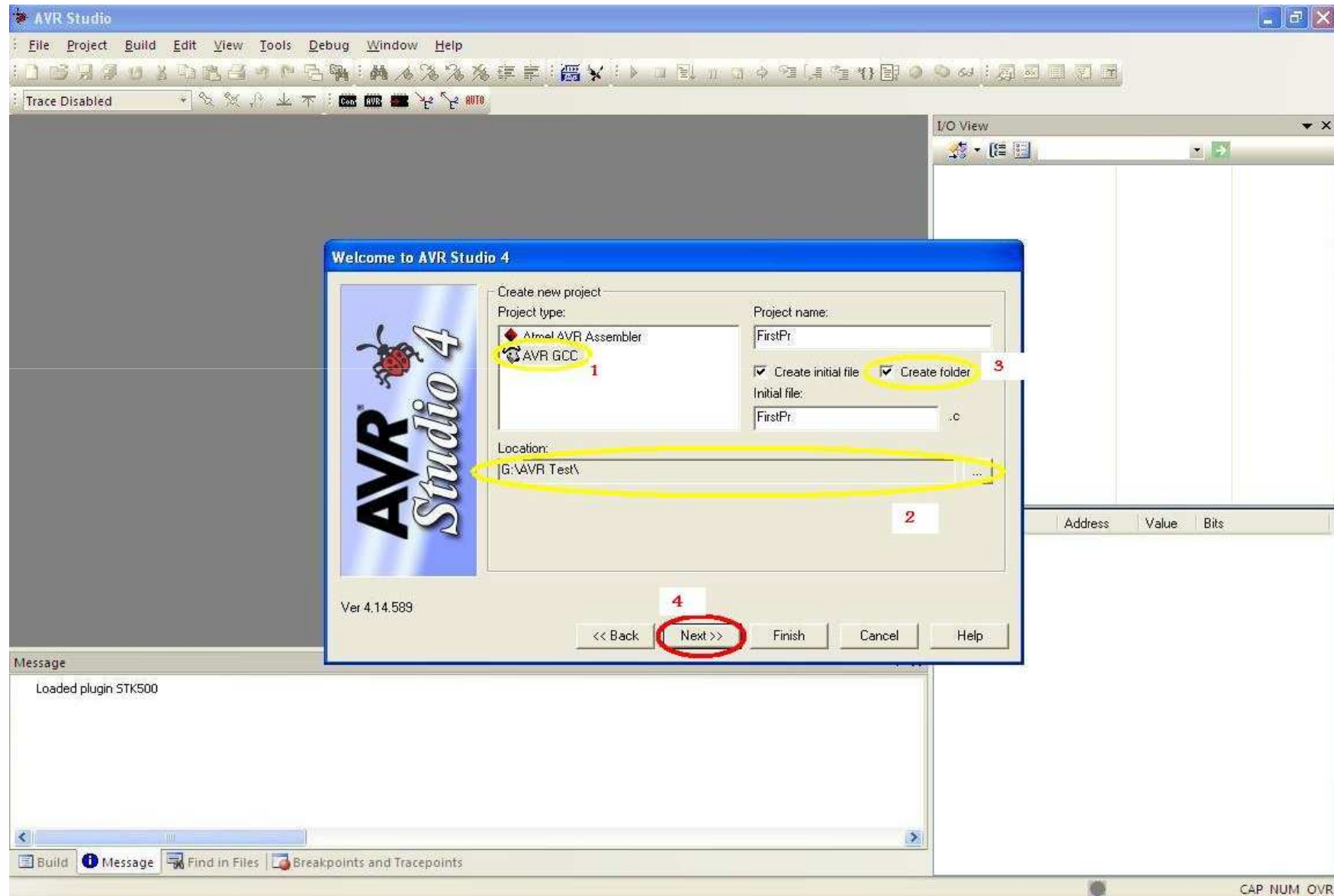
- AVR Studio is an Integrated Development Environment (IDE) for writing and debugging AVR applications in Windows environment.
- AVR Studio 4 includes an assembler and a simulator ONLY
- AVR Studio is FREE IDE from atmel
- AVR Studio does not contain C compiler

WINAVR

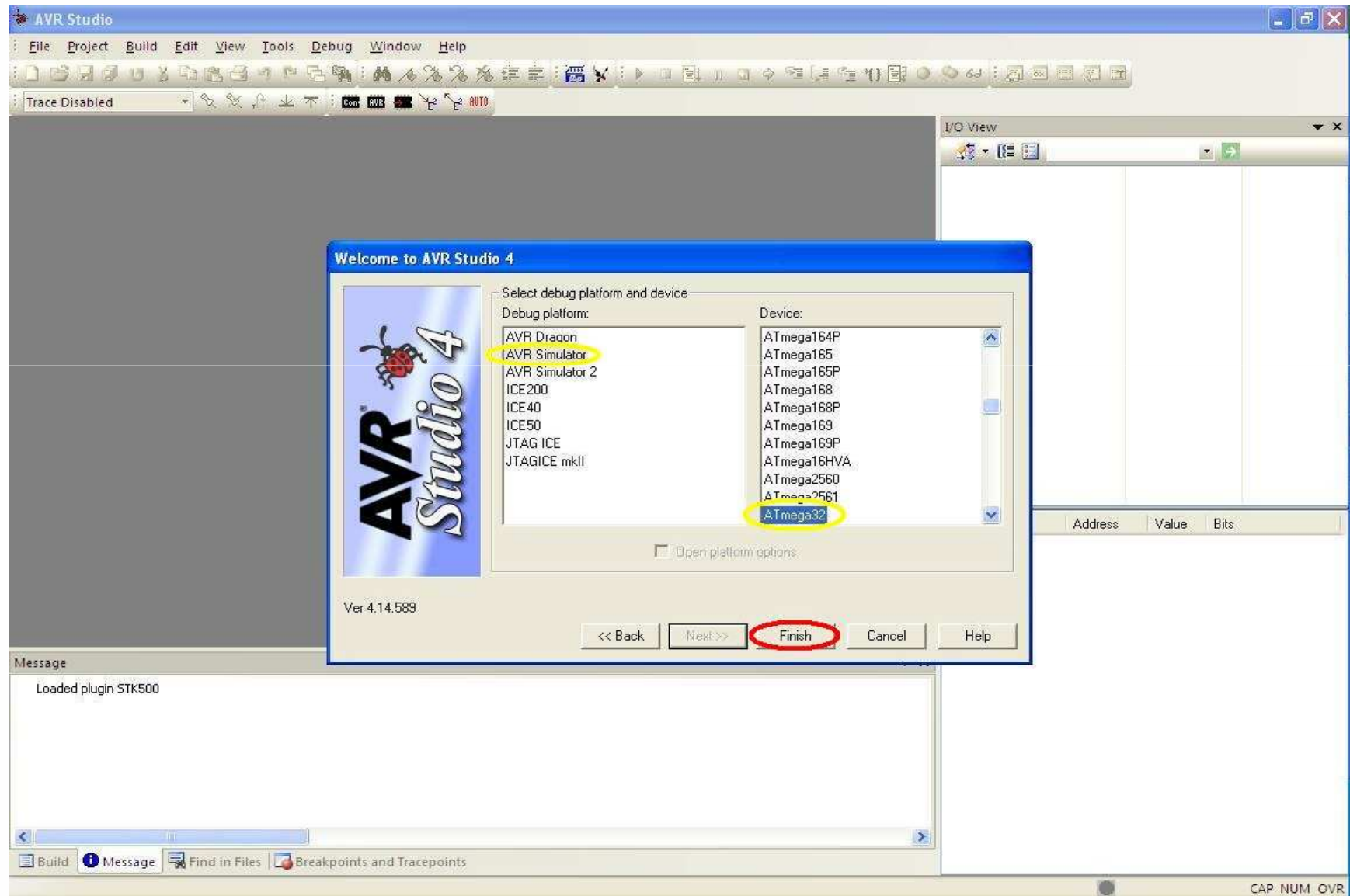
AVR Studio



AVR Studio

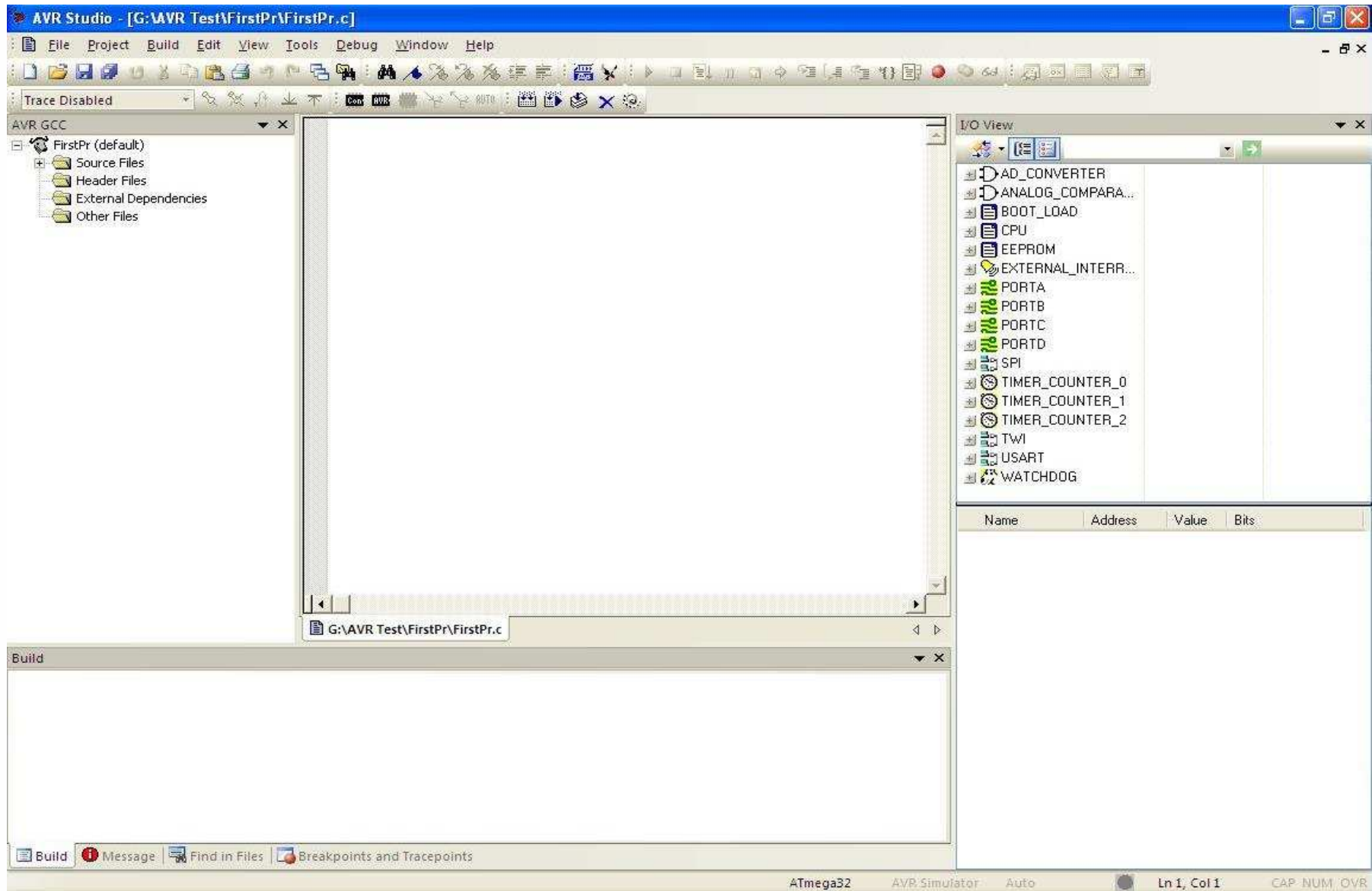


AVR Studio





AVR Studio



Interrupt

Status Register (SREG)

Bit	7	6	5	4	3	2	1	0	
	I	T	H	S	V	N	Z	C	SREG
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	

- The status register contains:
 - Six bits status indicators (**Z,C,H,V,N,S**)
 - One bit for global interrupt enable (**I**)
 - One bit for bit copy storage (**T**)
- The status bits reflect the results of CPU operation as it executes instructions

Stack Pointer (SP)

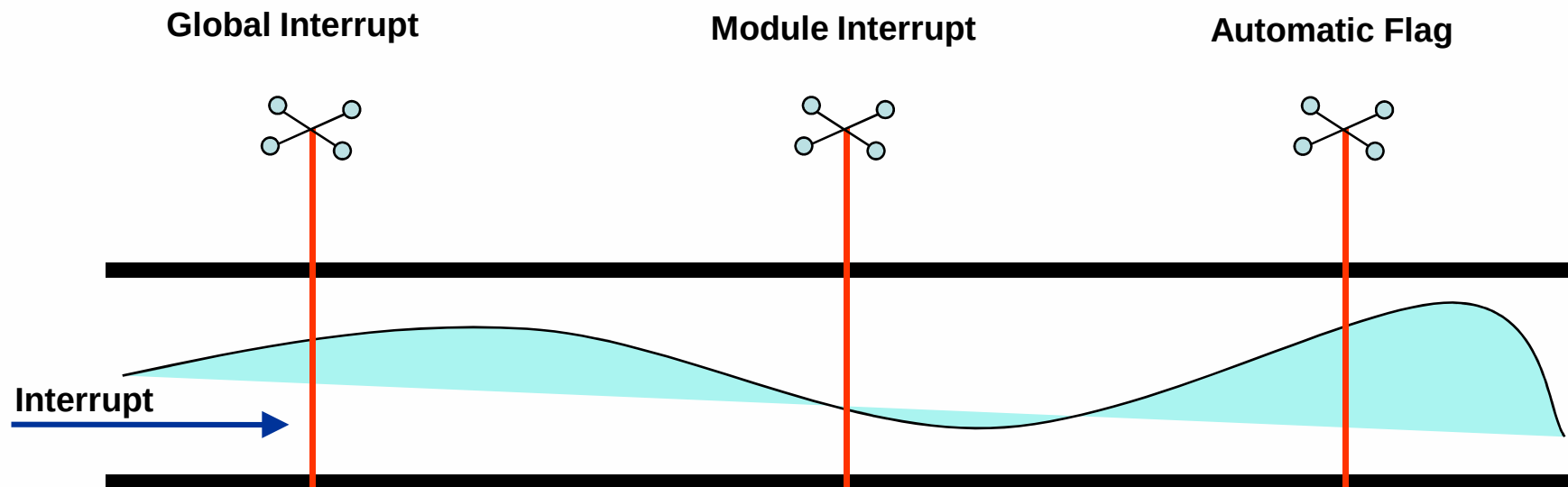
Bit	15	14	13	12	11	10	9	8	
	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	SPH
	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	SPL
	7	6	5	4	3	2	1	0	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

- The stack is used to save system context during **subroutine calls** and **interrupts** and can also be used for **temporary data storage**.
- The stack pointer (SP) holds the 16-bit address of the last stack location used.
- A Push to the Stack Automatically decrement SP.
- A Pull from the Stack Automatically increment SP.

ATmega32 Interrupt Vector

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	INT2	External Interrupt Request 2
5	\$008	TIMER2 COMP	Timer/Counter2 Compare Match
6	\$00A	TIMER2 OVF	Timer/Counter2 Overflow
7	\$00C	TIMER1 CAPT	Timer/Counter1 Capture Event
8	\$00E	TIMER1 COMPA	Timer/Counter1 Compare Match A
9	\$010	TIMER1 COMPB	Timer/Counter1 Compare Match B
10	\$012	TIMER1 OVF	Timer/Counter1 Overflow
11	\$014	TIMER0 COMP	Timer/Counter0 Compare Match
12	\$016	TIMER0 OVF	Timer/Counter0 Overflow
13	\$018	SPI, STC	Serial Transfer Complete
14	\$01A	USART, RXC	USART, Rx Complete
15	\$01C	USART, UDRE	USART Data Register Empty
16	\$01E	USART, TXC	USART, Tx Complete
17	\$020	ADC	ADC Conversion Complete
18	\$022	EE_RDY	EEPROM Ready
19	\$024	ANA_COMP	Analog Comparator
20	\$026	TWI	Two-wire Serial Interface
21	\$028	SPM_RDY	Store Program Memory Ready

Interrupt

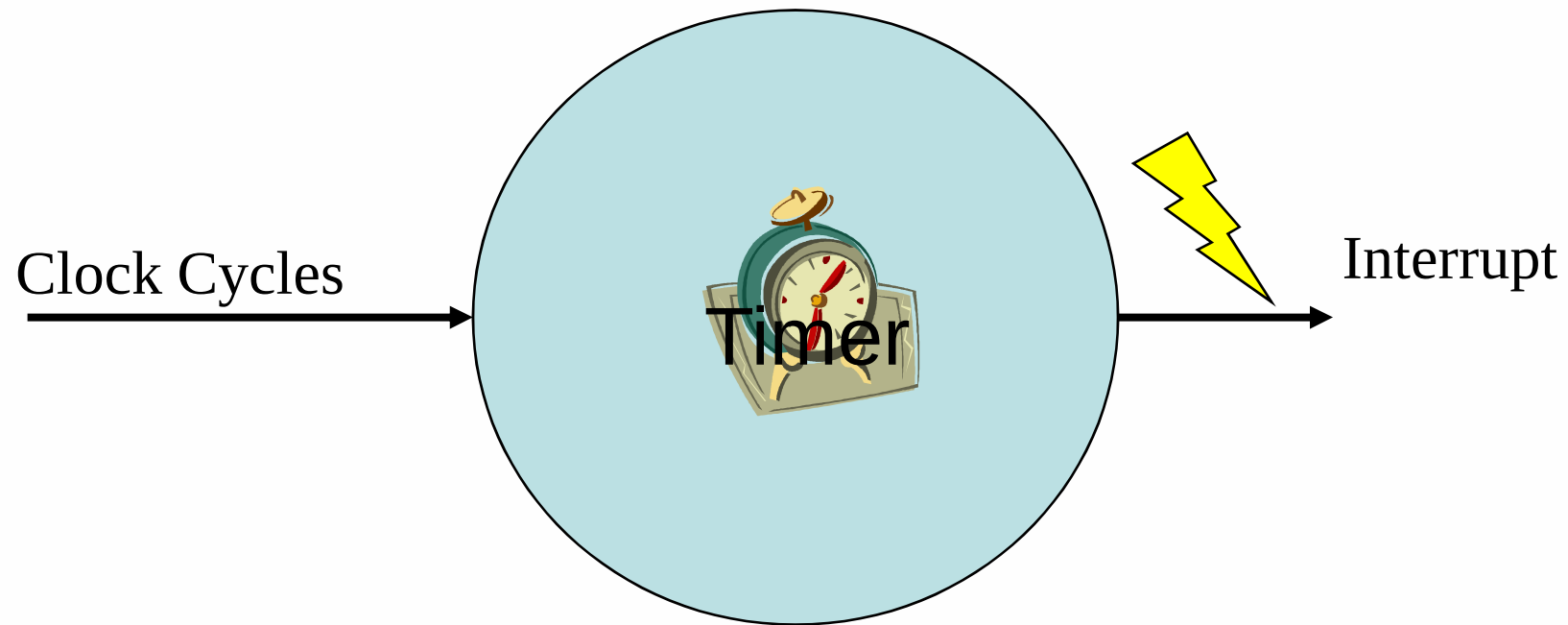


Interrupt Action

- What happens when an interrupt occurs?
 - The current instruction finishes execution
 - The address of the next instruction is calculated and pushed onto the stack
 - All the CPU registers are pushed onto the stack
 - The program counter is loaded with the address pointed to by the interrupt vector and execution continues

Timer Module

Timer ?



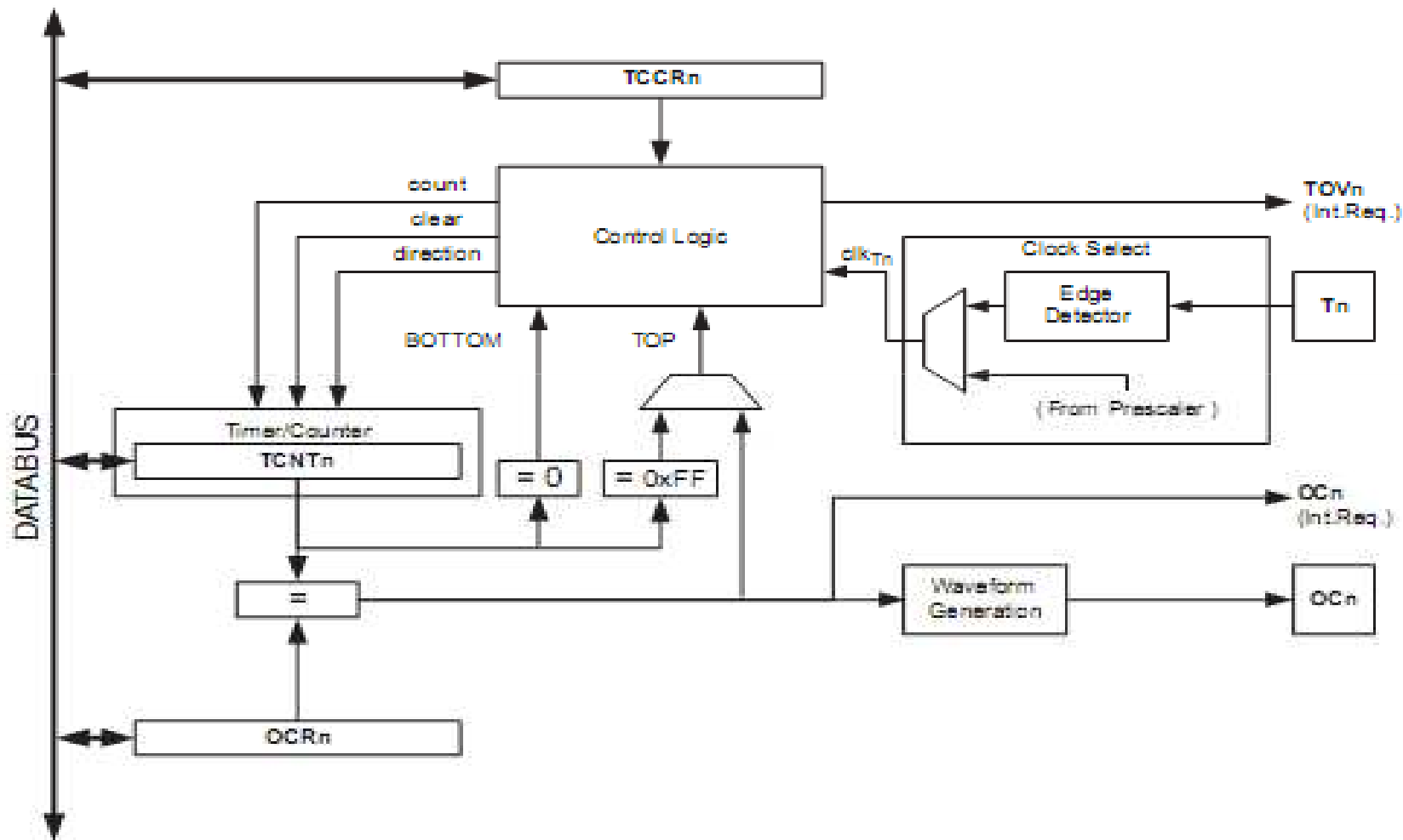
AVR Timers

- **ATmega32** have **3** Timer/counter
 - Two 8-bit Timer/Counters with Separate Prescalers
 - One 16-bit Timer/Counter with Separate Prescaler
- The 3 timers are divided as:
 - 8-bit Timer/Counter**0** with PWM
 - 16-bit Timer/Counter**1**
 - 8-bit Timer/Counter**2** with PWM and Asynchronous Operation

8-bit Timer

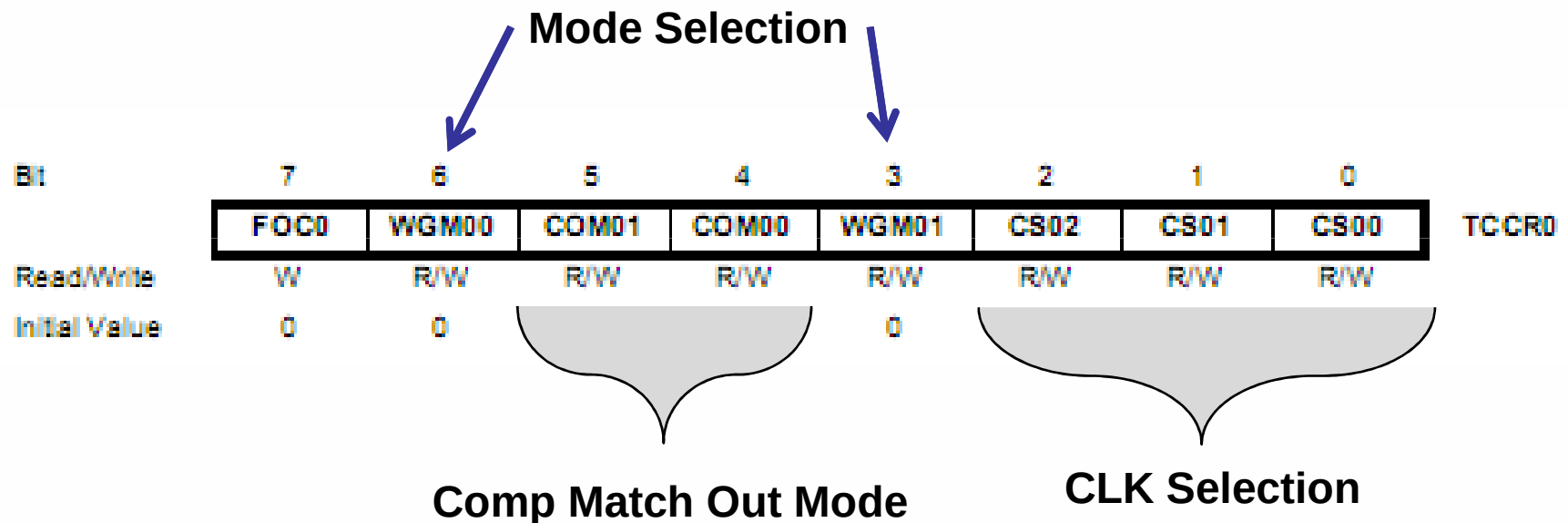
- 8-bit timer module contains **5** registers:
 - Timer/Counter Register – **TCNT0**
 - Timer/Counter Control Register – **TCCR0**
 - Output Compare Register – **OCR0**
 - Timer/Counter Interrupt Mask Register – **TIMSK**
 - Timer/Counter Interrupt Flag Register – **TIFR**
- 8-bit timer have **4** modes of operation:
 - Normal Mode
 - Clear Timer on Compare Match (CTC) Mode
 - Fast PWM Mode
 - Phase Correct PWM Mode

8-bit Timer/Counter Block Diagram



8-bit Timer/Counter Registers

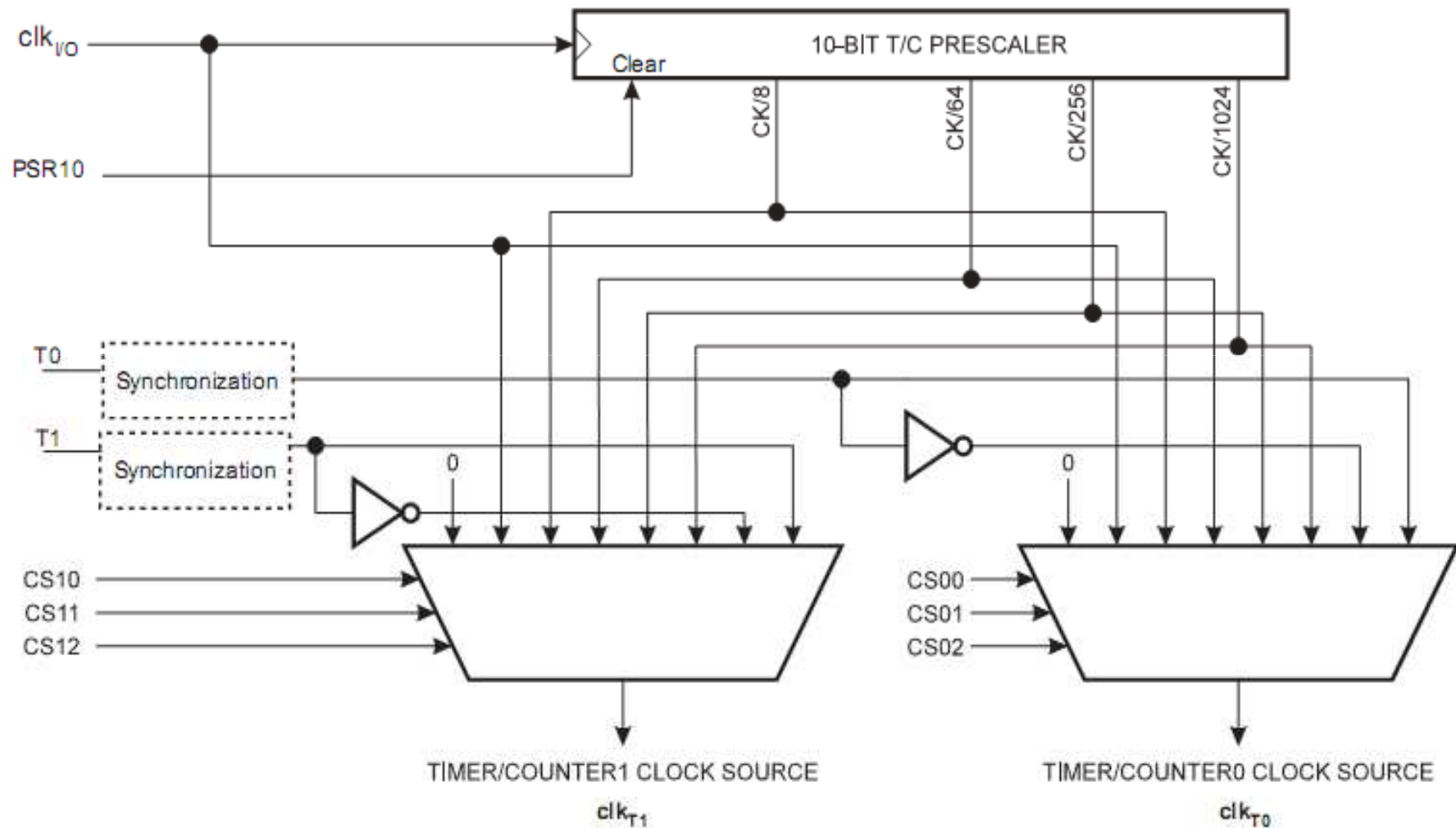
TCCR0



Clock Selection

CS02	CS01	CS00	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	clk_{IO} /(No prescaling)
0	1	0	$\text{clk}_{\text{IO}}/8$ (From prescaler)
0	1	1	$\text{clk}_{\text{IO}}/64$ (From prescaler)
1	0	0	$\text{clk}_{\text{IO}}/256$ (From prescaler)
1	0	1	$\text{clk}_{\text{IO}}/1024$ (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.

Prescaler ?



8-bit Timer/Counter Registers

TCNT0

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	

8-bit Timer/Counter Registers

OCR0

Bit	7	6	5	4	3	2	1	0	
	OCR0[7:0]								OCR0
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	

8-bit Timer/Counter Registers

TIMSK

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	

This register is shared between the **three** timers

8-bit Timer/Counter Registers

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	

This register is shared between the **three** timers

Timer Module

- Initialize timer
- Enable global interrupt
- Start counting
- Define interrupt vector used
- Write the ISR

How to download your program?

LAB

References

- www.atmel.com
- www.avrbeginners.net
- ATmega32 Datasheet



info@escommittee.com

education@escommittee.com