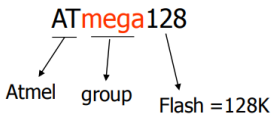
- Microprocessor-> No RAM, ROM, I/O Ports on chip itself
- Microcontroller -> Microprocessor, RAM, ROM, timers, I/O Ports on a single chip.
- Embedded system is controlled by its own internal microprocessor (or microcontroller) as opposed to an external controller.
- The AVR is an 8-bit RISC single-chip microcontroller with **Harvard architecture.**

CISC (Complex Instruction Set Computer)	RISC (Reduced Instruction Set Computer)
A large number of instructions, typically from 100 to 250 instructions	Relatively few instructions
A large variety of addressing modes, typically from 5 to 20 different modes	Relatively few addressing modes
Variable-length instruction formats	Fixed-length, easily decoded instruction format

- von Neumann (Princeton) architecture:
  - The same bus is used for accessing both the code and data
  - Pentium Processor is based on von Neumann Architecture
- Harvard architecture:
  - Separate buses are used for accessing the code and data memory.
  - 4 set of buses:
    1. A set of data buses for carrying data into and out of the CPU
    2. A set of address buses for accessing the data
    3. A set of data buses for carrying code into the CPU
    4. An address bus for accessing the code.
  - implement inside an IC chip such as a microcontroller where both **ROM code** and **data RAM** are internal (on-chip) and distances are on the micron and millimeter scale
- RAM, ROM, EEPROM memories are in bytes.
- Data RAM is the amount of RAM available for data manipulation and register space.



- One single instruction executed in one clock cycle

PINS

- PB0 -> T0
- PB1 -> T1
- PB2 -> INT2

- PD2 -> INT0
- PD3 -> INT1

Timers

- TCCR0:
  - FOC0 -> writing 1 to it forces the wave generator to act as if compare match has occurred
  - WGM00(**D6**) ,WGM01(**D3**)-> Timer0 selector mode (Normal /CTC/PWM/fast PWM)
  - COM01(**D5**) ,COM00(**D4**) -> Compare output mode(normal port/toggle OC0/set OC0 /clear OC0)
  - CS02:00 -> Timer0 clock selector (no clock, no prescale, clk/8, clk/64 , clk/256 , clk/1024,falling edge ,rising edge)
- TIFR:
  - TOV0 -> Timer0 overflow flag
  - OCF0 -> Timer0 Compare flag
  - TOV1 -> Timer1 overflow flag
  - OCF1B -> Timer1 Compare B flag
  - OCF1A -> Timer1 Compare A flag
  - ICF1 -> Input capture flag
  - TOV2 -> Timer2 overflow flag
  - OCF2 -> Timer2
- Delay length depends on crystal frequency, prescaler factory and C compiler
- AVR Timer use to generate time delay, count, detect, and measure the time of events occur outside the AVR
- When the timer is used as timer, the AVR's crystal is used as the source of the frequency
- When the timer is used as counter, increments the TCNTx register

Interrupt

- **The interrupt service routine (ISR) -> the program associated with the interrupt.**
- Interrupt vector table (**ROM**) -> the group of fixed memory locations set aside to hold the addresses of ISRs
- Source of interrupt:

    - There at least 2 interrupt set aside for each timer (overflow, compare match)
    - External hardware interrupt (INT0 ,INT1,INT2)
    - Serial communication's USART has 3 interrupt(1-> receive , 2-> transmit)
    - SPI interrupt
    - ADC
- SREG:
    - C ->Carry flag
    - Z -> Zero flag
    - N ->Negative flag
    - V -> Overflow flag
    - S ->Sign flag
    - H -> Half carry
    - T -> Bit copy storage
    - I -> Global interrupt enable
- TIMSK:
    - TOIE0 -> Timer0 overflow interrupt enable
    - OCIE0 -> Timer0 output compare match interrupt enable
    - TOIE1 -> Timer1 overflow interrupt enable
    - OCIE1B -> Timer1 output compare B match interrupt enable
    - OCIE1A -> Timer0 output compare A match interrupt enable
    - TICIE1 -> Timer1 input capture enable
    - TOIE2 -> Timer2 overflow interrupt enable
    - OCIE2 -> Timer2 output compare match interrupt enable
- GICR:
    - INT0
    - INT1
    - INT2
- Types of activation for external hardware interrupts:
    - Level-triggered
    - Edge-triggered
- **INT0, INT1 have 2 types, but INT2 has only edge-triggered**
- MCUCR:
    - ISC01, ISC00 -> define the level or edge on external INT0 pin (low, any logical changes, falling, rising)

- ISC11, ISC01 -> define the level or edge on external INT1 pin (low, any logical changes, falling, rising
- If 2 interrupts are activated at the same time, interrupt with higher priority is served first.
- The priority of each interrupt is related to the address of interrupt in interrupt vector.
- The interrupt that has lower address, has a higher priority (INT0 then INT1 then INT2).
- cli(), sei() -> instructions to clear/set I bit in SREG register
- لومش مدياني clock هفرضها ب 1MHZ
- **Context saving -> the C compiler automatically adds instructions to the beginning of the ISRs, which save the contents of all general purpose registers and the SREG register on the stack. Some instructions are added to the end of the ISRs to reload the registers.**

Serial port programming

- In RS232, 1 is represented by -3, -25v, while 0 is represented by +3, +25v, making -3, +3v undefined, so to connect any RS232 to a microcontroller system we must use voltage converters to convert the TTL logic levels to the RS232 voltage levels
- The MAX232 convert from RS232 voltages levels to TTL voltage levels and vice versa.
- RS232 operates in full duplex mode

Desired Baud Rate = Fosc/ (16(X + 1))

where X is the value we load into the UBRR register. To get the X value for different baud rates we can solve the equation as follows:

X = (Fosc/ (16(Desired Baud Rate))) – 1

Assuming that Fosc = 8 MHz, we have the following:

Desired Baud Rate = Fosc/ (16(X + 1)) = 8 MHz/16(X + 1) = 500 kHz/(X + 1)

X = (500 kHz/ Desired Baud Rate) – 1

- UCSRA:
  - RXC -> set when there are new data in receive buffer that are not read yet / generate a receive complete interrupt (cleared when receive buffer is empty).
  - TXC -> set when the entire frame in transmit shift register has been transmitted and there are now no data available in (TXB) / generate a transmit complete interrupt (cleared by writing a one

bit to its bit location / automatically when transmit complete interrupt is executed).
    - UDRE -> set when transmit data buffer is empty and it is ready to receive new data / generate a data register empty interrupt
    - FE -> set if frame error has occurred in receiving the next character in receive buffer. It is detected when the first stop bit of the next character in receive buffer is 0 .
    - DOR -> set when the receive buffer and receive shift register are full, and new start bit is detected.
    - PE -> set if parity checking was enabled(UPM1 =1) and the next character in receive buffer had a parity error when received.
    - U2X -> setting this bit will double the transfer rate asynchronous communication.
- UCSRB:
    - RXCIE -> to enable interrupt on the RXC flag in USCRA
    - TXCIE -> to enable interrupt on the TXC flag in USCRA
    - UDRIE -> to enable interrupt on the UDRE flag in USCRA
    - RXEN -> to enable the USART receiver.
    - TXEN -> to enable the USART transmitter
    - UCSZ2 -> combined with UCSZ1:0 in USCRC set character size in a frame.
    - RXB8
    - TXB8
- USCRC:
    - URSEL -> select to access either USCRC or UBRRH.
    - UMSEL -> select operate in either the asynchronous (0) or synchronous mode (1)
    - UPM1:0 -> disable or enable and set the type of parity generation and check (disabled, reserved, even parity, odd parity)
    - USBS -> select number of stop bits to be transmitted
    - UCSZ1:0 -> combined with UCSZ2 in USCRB set character size in a frame.
- The character byte to be transmitted serially is written into the UDR register.

- Modulation يعني بحول من Volt ل bit
- Demodulation يعني بحول من bit ل volt

Note: For Fosc = 8 MHz we have UBRR = (500000/BaudRate) – 1

ADC

- $D_{out} = v_{in}$ / step size

For an 8-bit ADC, we have V_{ref} = 2.56 V. Calculate the D0–D7 output if the analog input is: (a) 1.7 V, and (b) 2.1 V.
Solution:

Because the step size is 2.56/256 = 10 mV, we have the following:
(a) D_{out} = 1.7 V/10 mV=170 in decimal, which gives us 10101010 in binary for D7–D0.
(b) D_{out} = 2.1 V/10 mV = 210 in decimal, which gives us 11010010 in binary for D7–D0.

- ADC Feature:
  - 10-bit ADC
  - 8 analog input channels
  - Convert output binary data is held by ADCL (A/D Result low), ADCH (A/D Result high)
  - ADCL: ADCH registers give us 16 bits, 6 bits are unused (either the upper 6 bits or the lower 6 bits), ADC data out is only 10 bits.
  - V_{ref}:
    - Connected to AVCC(Analog v_{cc})
    - Internal 2.56V reference
    - External AREF pin
  - The conversion time is dictated by crystal frequency connected to XTAL pins (Fosc) and ADPS0:2 bits
- AVCC pin provides the supply for analog ADC circuitry
- ADMUX:
  - REFS1:0 -> select reference voltage for the ADC (AREF, AVCC, Reserved, Internal 2.56v)
  - ADLAR -> dictate either the left bits or right bits of the result registers ADCL: ADCH that are used to store the result (1 -> left, 0 -> right)
  - MUX4:0 -> select gain for differential channels and which combination of analog inputs are connected to the ADC
- ADCSRA:
  - ADEN -> enable or disable ADC
  - ADSC -> to start conversion you have to set this bit to 1
  - ADATE -> enable auto triggering of ADC
  - ADIF -> set when an ADC conversion complete and data registers are updated

    - ADIE -> enable ADC conversion complete interrupt
    - ADP2:0 -> determine the division factor between the XTAL and the input clock to the ADC
- You have to read ADCL before ADCH
- ADC requires an input clock frequency less than 200khz