MICROCONTROLLERS

Maurício Féo

m.feo@cern.ch

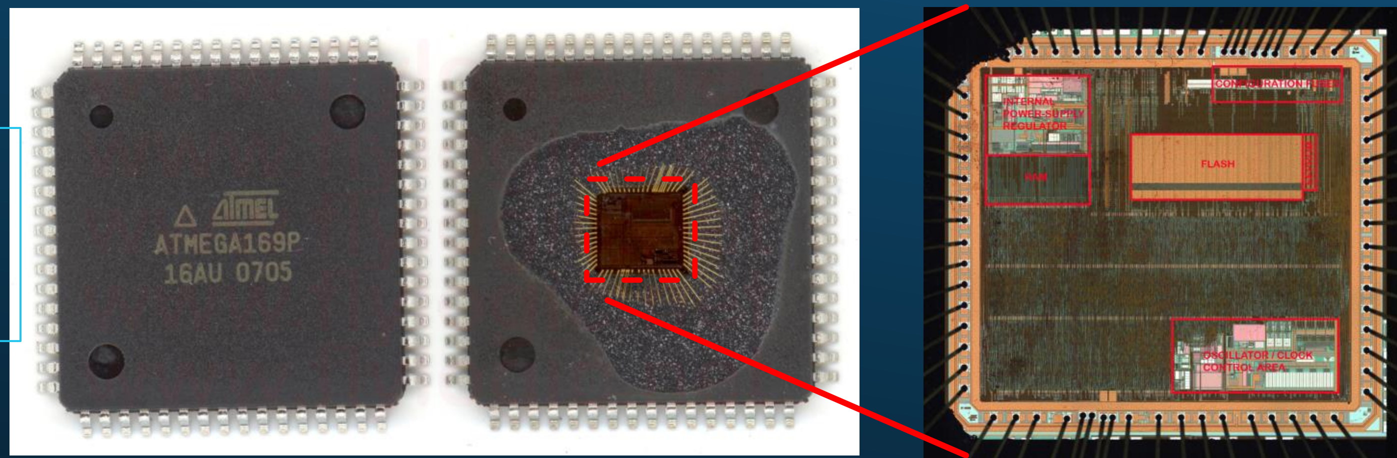
CERN

OBJECTIVES

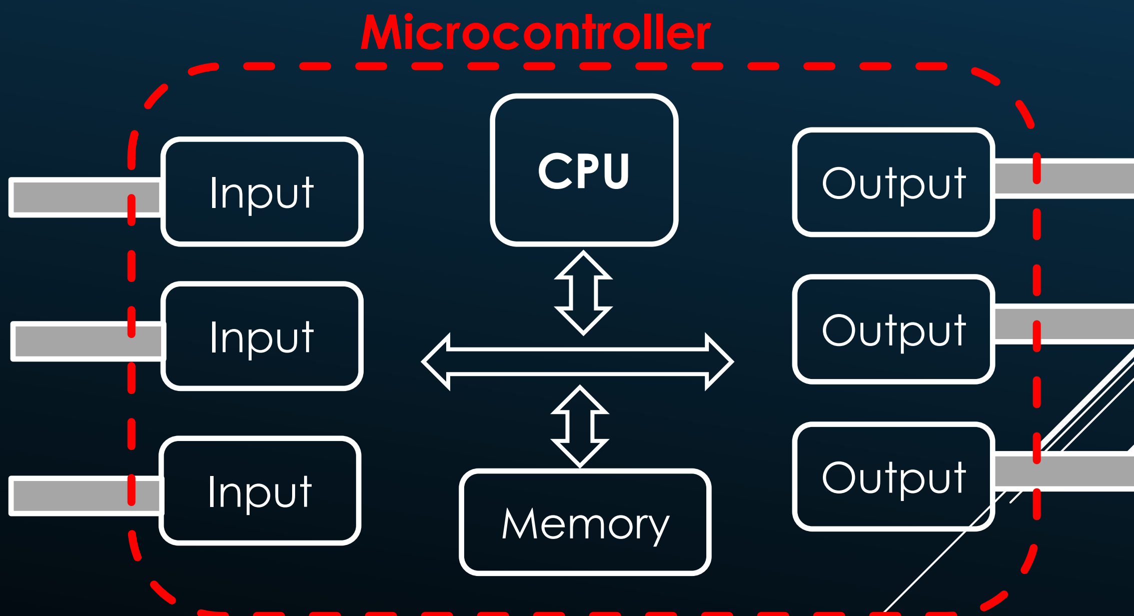
- Understand what are microcontrollers.
  - What are they?
  - What are they used for?
  - How do they work?
  - How can I use them?
  - Are they suitable for my project?
- Example of applications.
- Have an overview of the lab 10.

WHAT IS A MICROCONTROLLER?

Tiny computer integrated in the same chip.

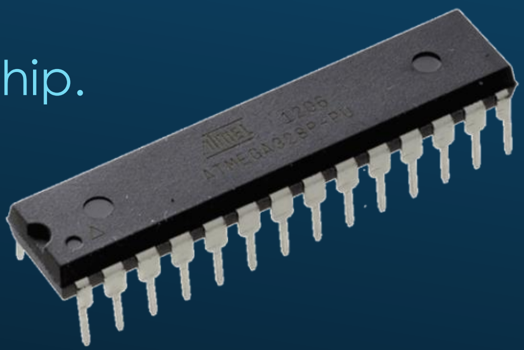


- CPU
- Memories
- I/O Interfaces
- Etc.



WHAT IS A MICROCONTROLLER?

- **Tiny computers integrated in a single chip**
  - CPU, Memories and Peripherals in the same chip.



Main differences w.r.t. a computer:

- Suitable for embedded applications.
- Low cost (ATtiny: **~$0.27**)
- Low power consumption (ATtiny43U: **0.15uA** in Sleep Mode)
- Reduced clock frequency (~ dozens of MHz)
- Stand-alone devices (Some require only power to work)
- Low level control of your application.

WHAT ARE THEY USED FOR?

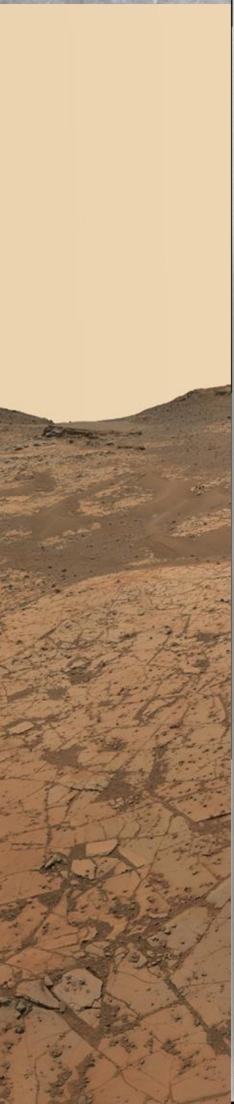
- Monitoring
- Data Acqusition
- Control

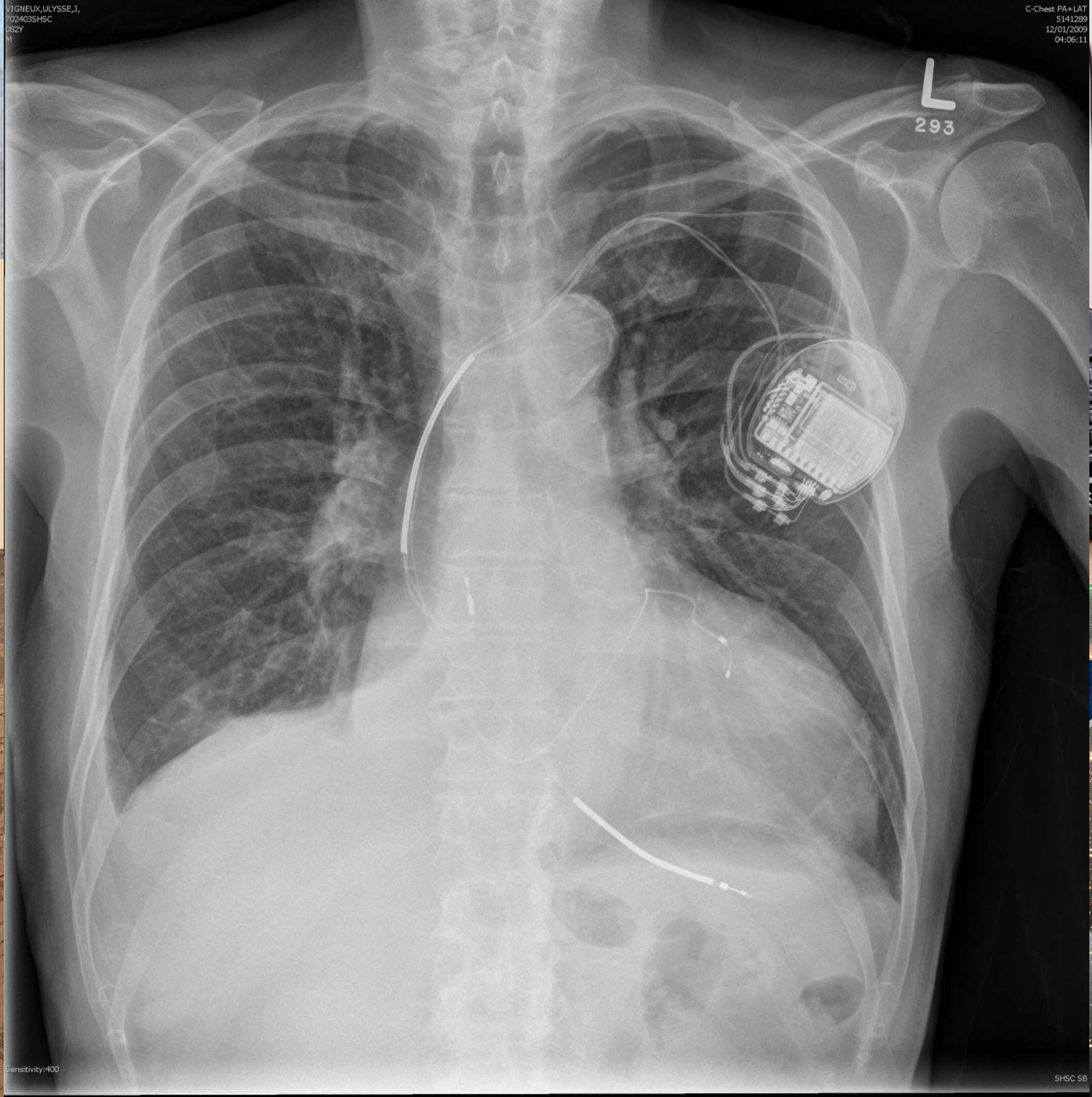
- Applications where:
  - High performance is not required.
  - Other devices are inadequate (overkill) due to:
    - High power consumption
    - need of external memories and peripherals
    - cost
    - etc.

WHERE ARE THEY USED?

- Everywhere!
- Consumer electronics, home appliances, toys, vehicles, computers, hobbyist projects, etc.

- According to ~~wikipedia~~ trusted sources, a typical mid-range automobile has as many as 30 or more microcontrollers.
- According to ~~me~~ an even more trusted source, you have at least one in your pocket right now.




VIGNEUX,ULYSSE,J,
702403SHSC
082Y
M
C-Chest PA+LAT
5141289
12/01/2009
04:06:11
L
293
Sensitivity:400
SHSC SB

Windows

An error has occurred. To continue:

Press Enter to return to Windows, or

Press CTRL+ALT+DEL to restart your computer. If you do this,
you will lose any unsaved information in all open applications.

Error: 0E : 016F : BFF9B3D4

Press any key to continue _

CERN CENTOS 7

~~Windows~~

An error has occurred. To continue:

Press Enter to return to Windows, or

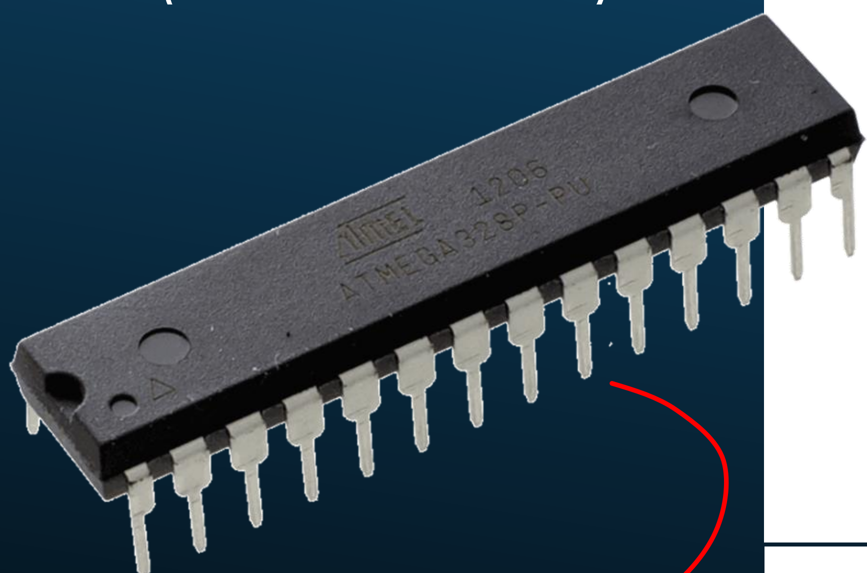
Press CTRL+ALT+DEL to restart your computer. If you do this,
you will lose any unsaved information in all open applications.

Error: 0E : 016F : BFF9B3D4

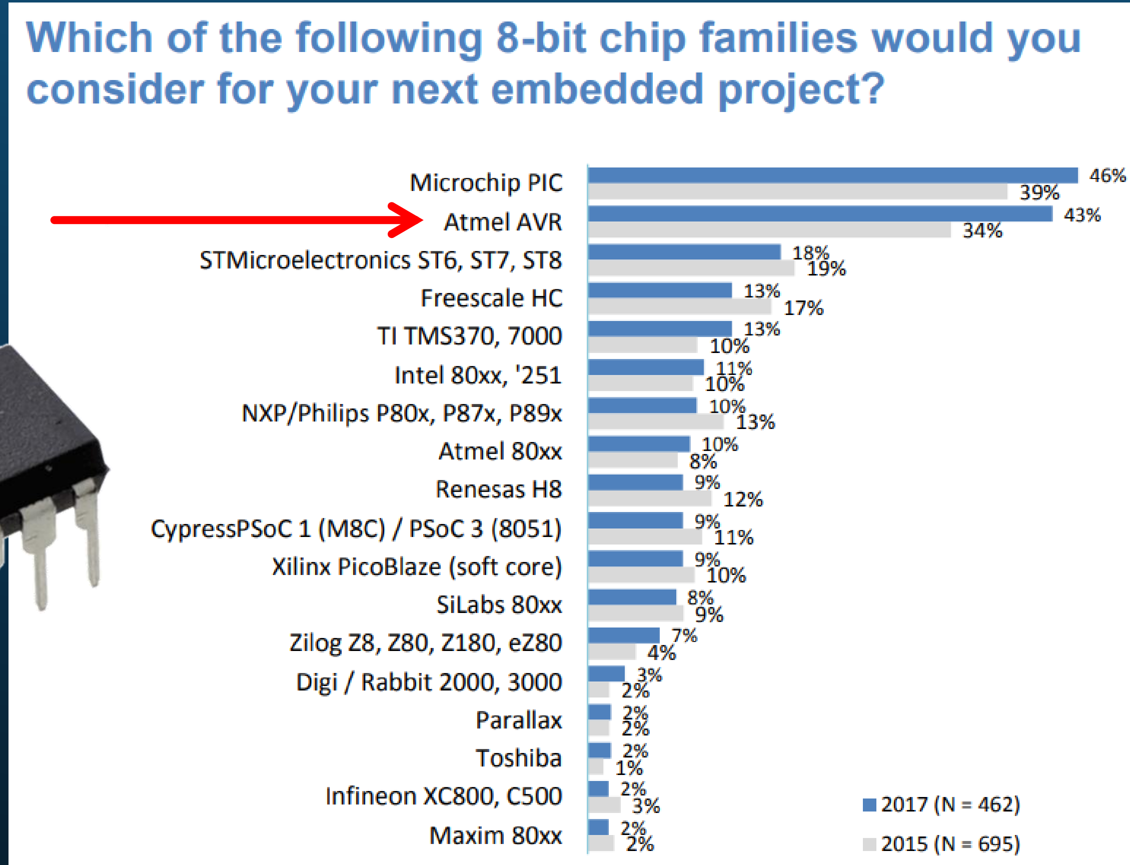
Press ~~any key~~ to continue _

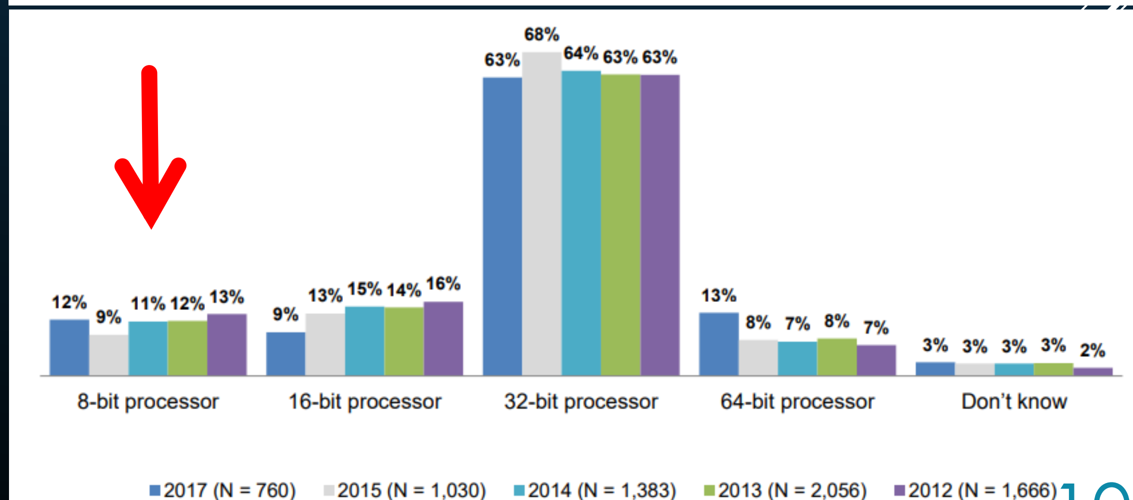
rm –rf $ROOTSYS

AVR ARCHITECTURE (ATMEGA328P)

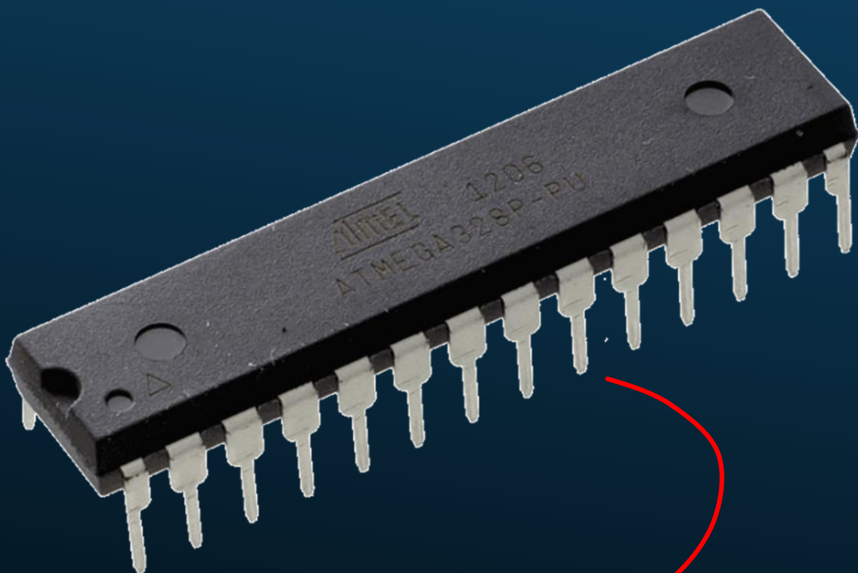


The one used on the lab.

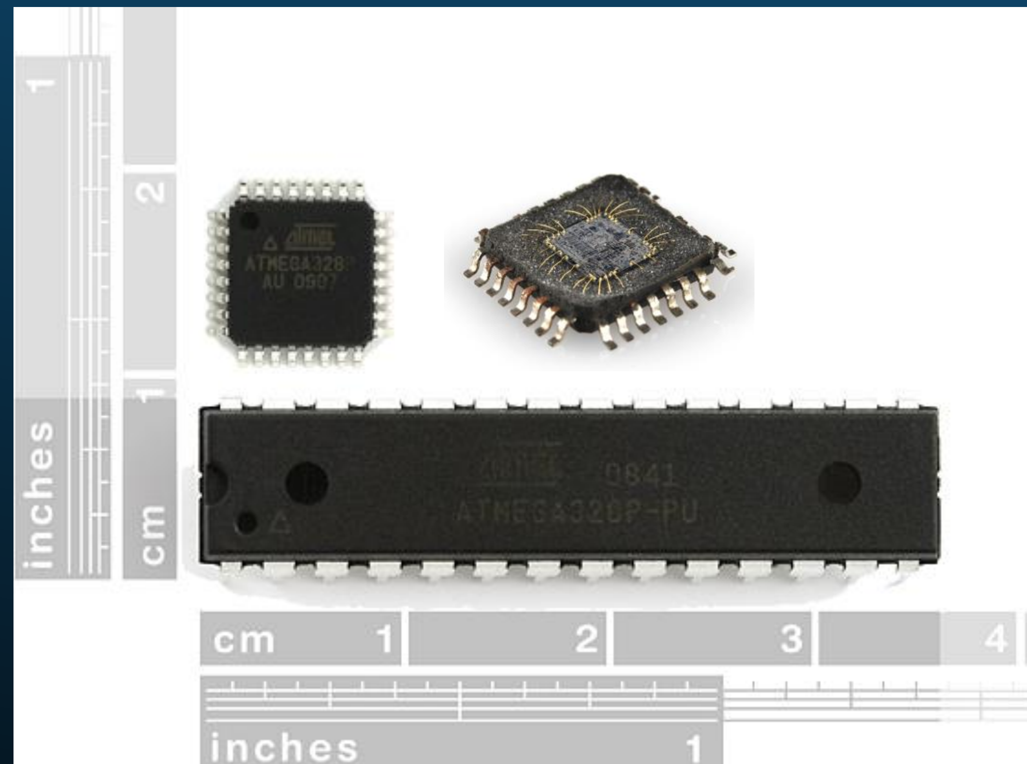




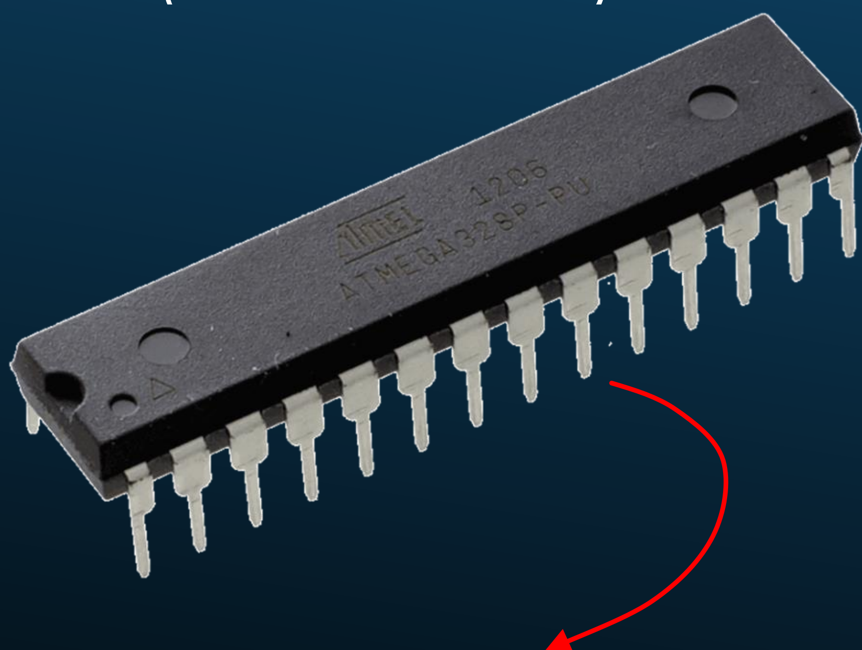
AVR ARCHITECTURE (ATMEGA328P)



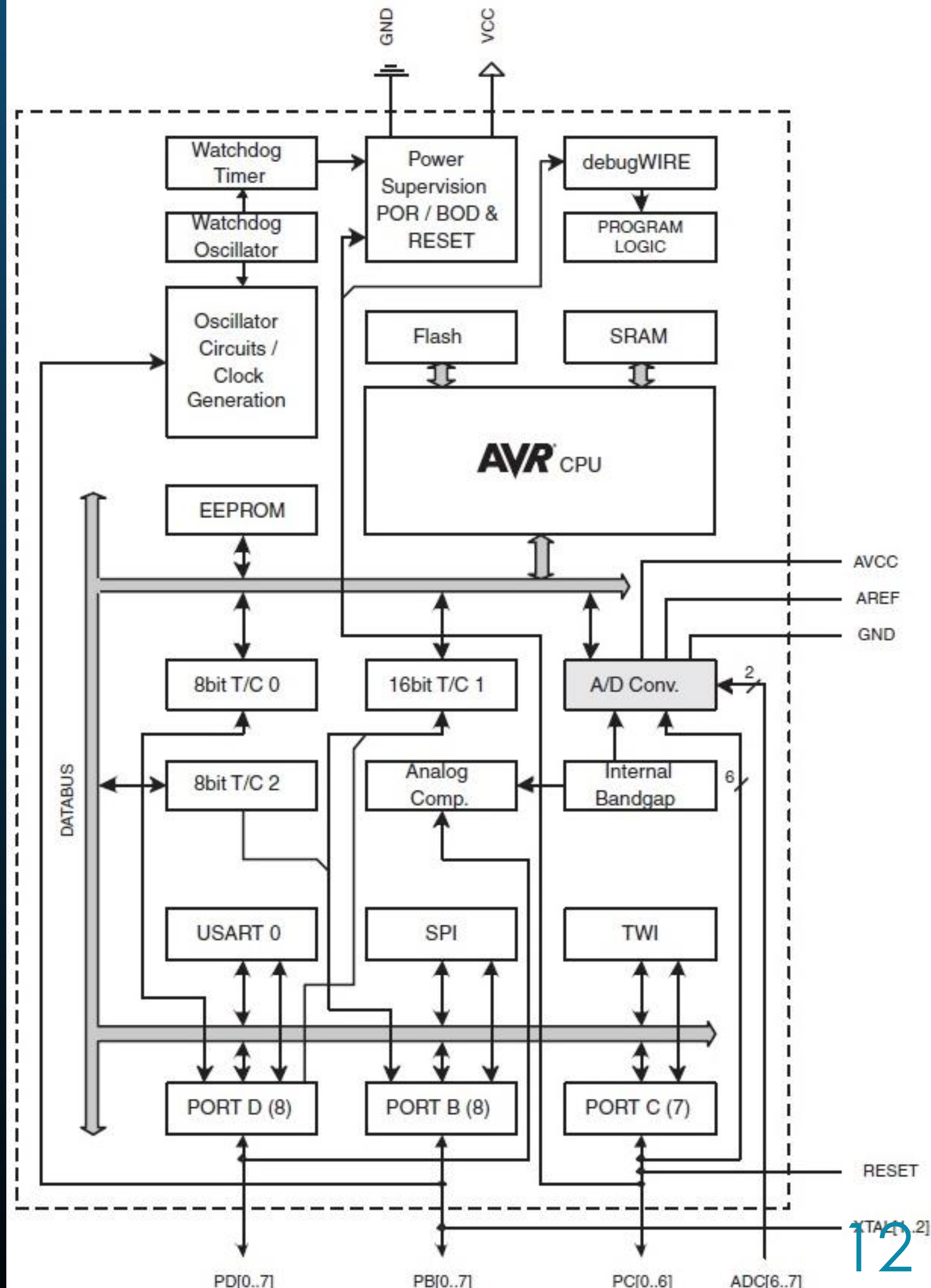
The one used on the lab.



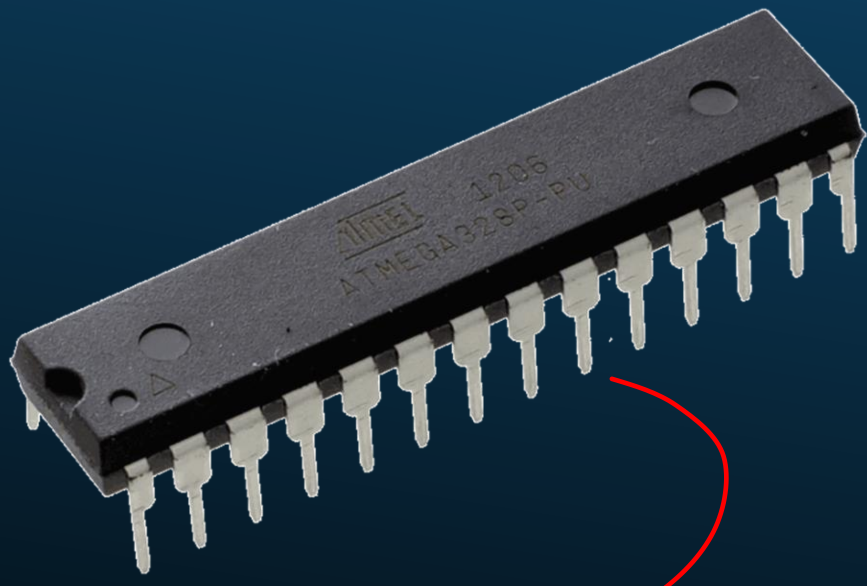
AVR
ARCHITECTURE
(ATMEGA328P)



The one used on the lab.



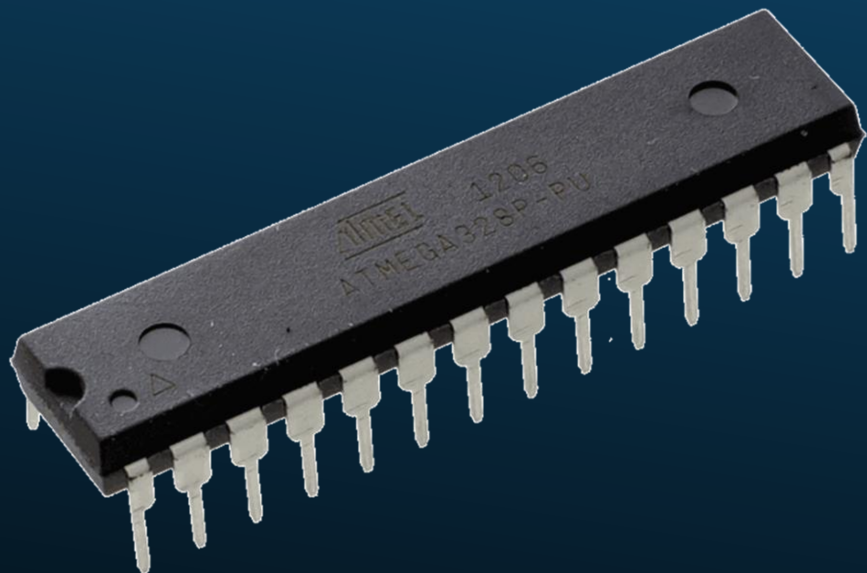
AVR ARCHITECTURE (ATMEGA328P)

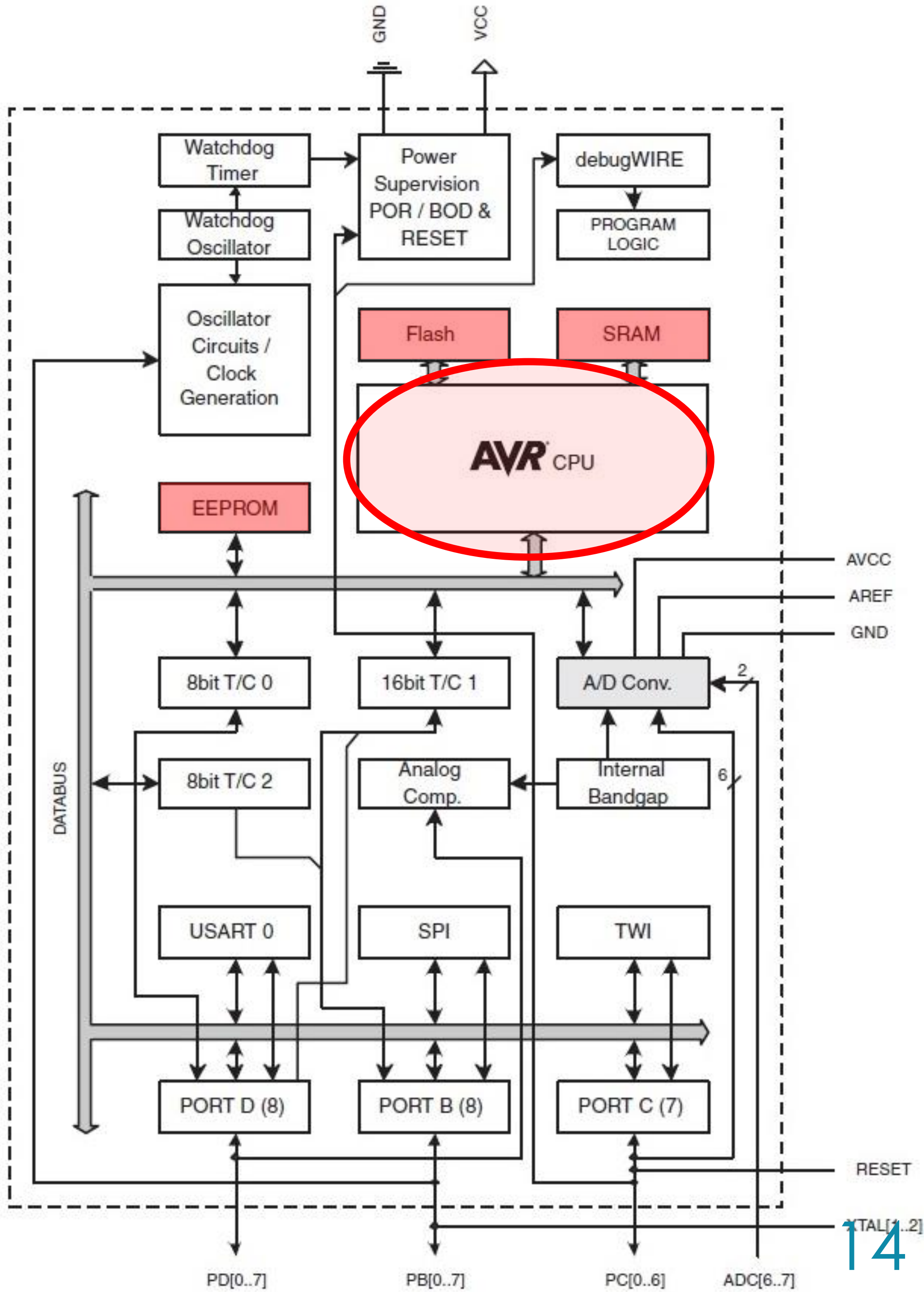


The one used on the lab.

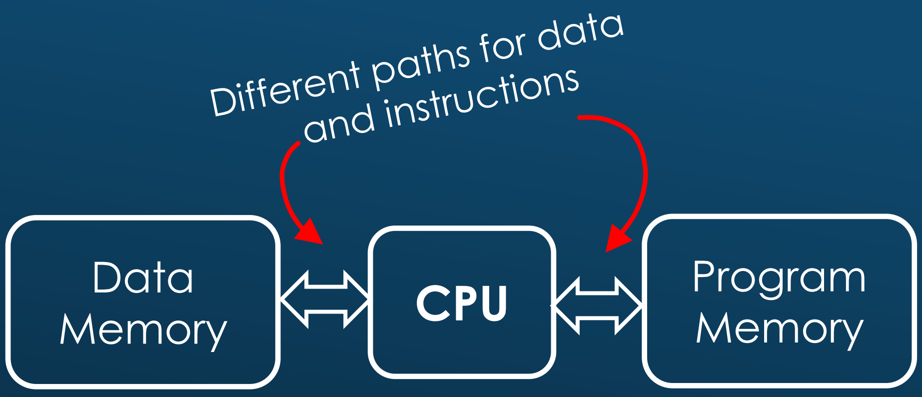
- 8 bits architecture
- 32kB Flash program memory
- 2kB RAM
- 2kB EEPROM
- Max 20MHz
- 6 x PWM
- 6 x ADC channels (10bits)
- 23 I/O pins
- 3 timers (2x8 bits 1x16 bits)
- 1x USART
- 1x SPI
- 1x TWI (I^2C)
- 0.6mA/MHz

AVR ARCHITECTURE (ATMEGA328P)





AVR CPU



- Harvard Architecture
- 8 bits architecture (with 16bits for instructions)
  - Instructions executed 8 bits by 8 bits
- Reduced Instruction Set Computing (RISC) (~130 instructions)
- Up to 20 MIPS at 20 MHz (1 instruction / clock cycle)

Features

- High performance, low power AVR® 8-bit microcontroller
- Advanced RISC architecture
  - 131 powerful instructions – most single clock cycle execution
  - 32 × 8 general purpose working registers
  - Fully static operation
  - Up to 16MIPS throughput at 16MHz
  - On-chip 2-cycle multiplier

6.6 Instruction Execution Timing

This section describes the general access timing concepts for instruction execution. The AVR® CPU is driven by the CPU clock clk_{CPU}, directly generated from the selected clock source for the chip. No internal clock division is used.

Figure 6-4 shows the parallel instruction fetches and instruction executions enabled by the harvard architecture and the fast-access register file concept. This is the basic pipelining concept to obtain up to 1MIPS per MHz with the corresponding unique results for functions per cost, functions per clocks, and functions per power-unit.

Figure 6-4. The Parallel Instruction Fetches and Instruction Executions

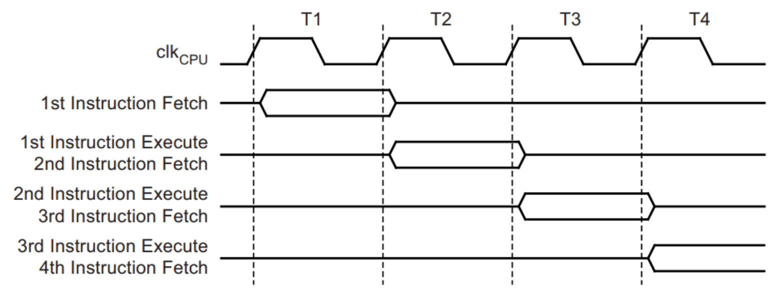
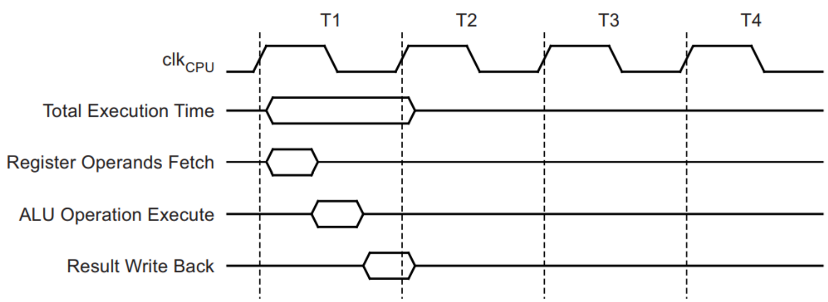
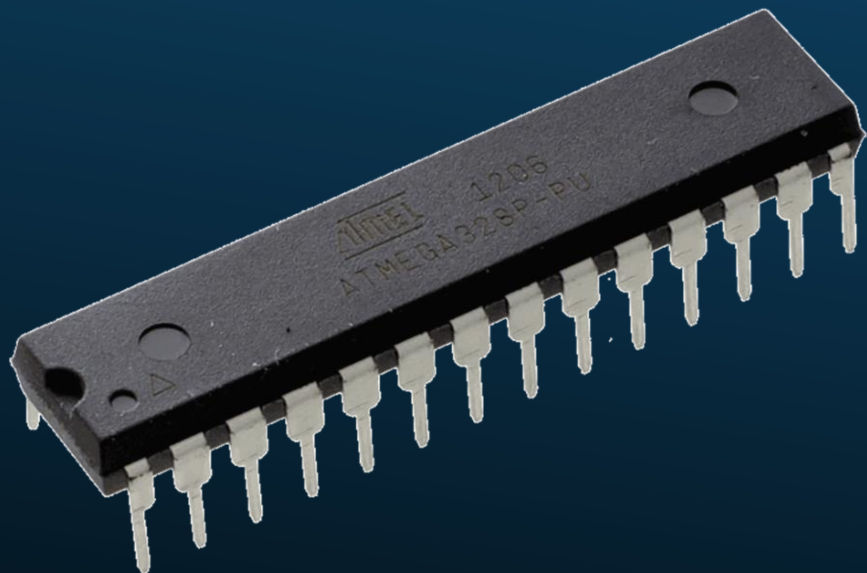


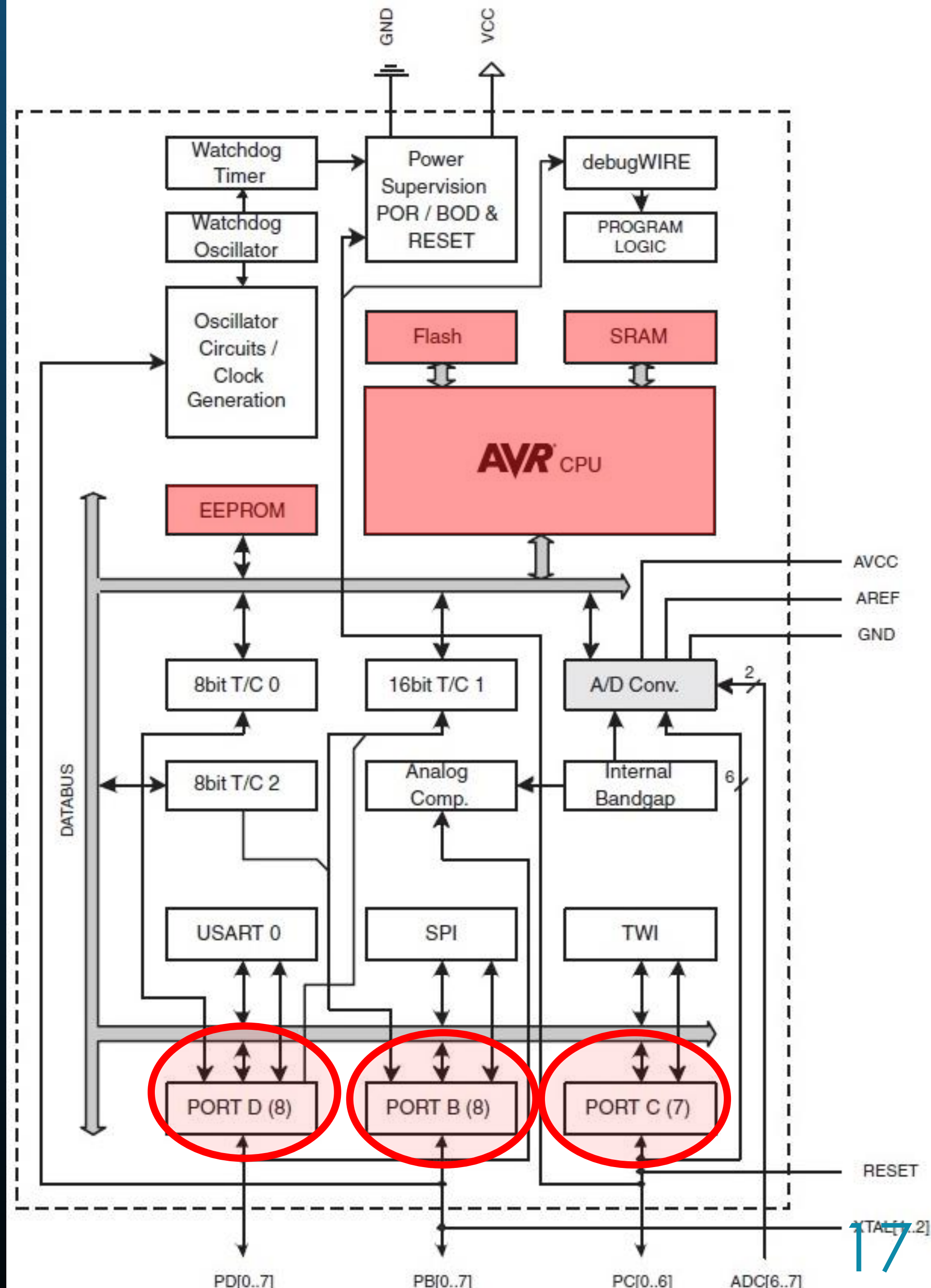
Figure 6-5 shows the internal timing concept for the register file. In a single clock cycle an ALU operation using two register operands is executed, and the result is stored back to the destination register.

Figure 6-5. Single Cycle ALU Operation



AVR
ARCHITECTURE
(ATMEGA328P)

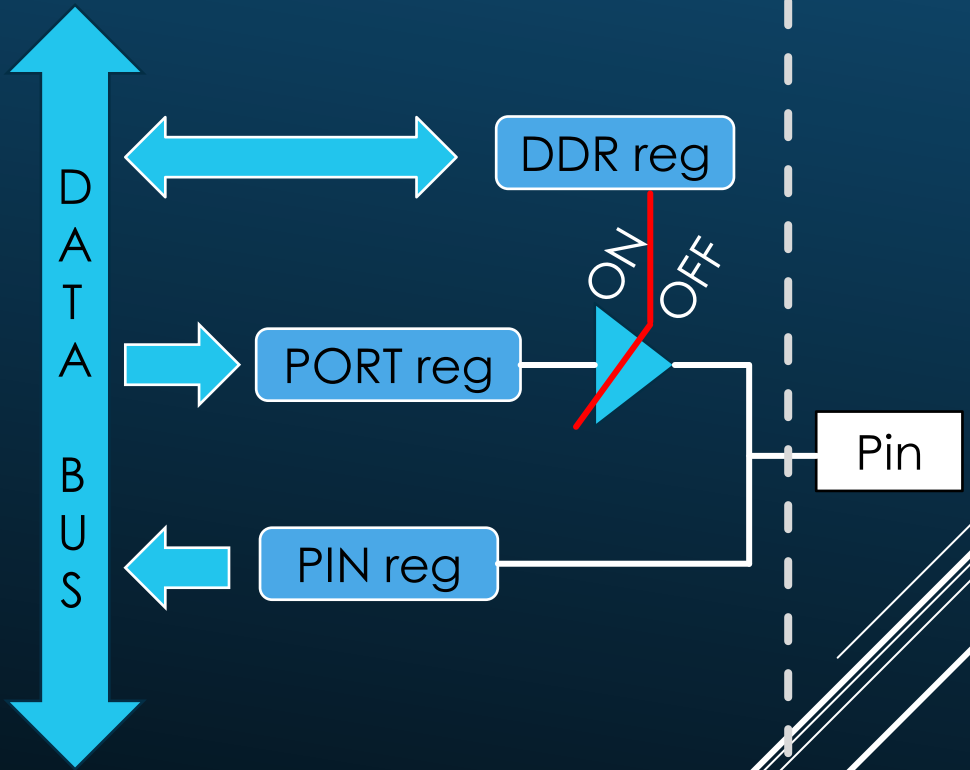




GENERAL PURPOSE INPUT/OUTPUT (GPIO)

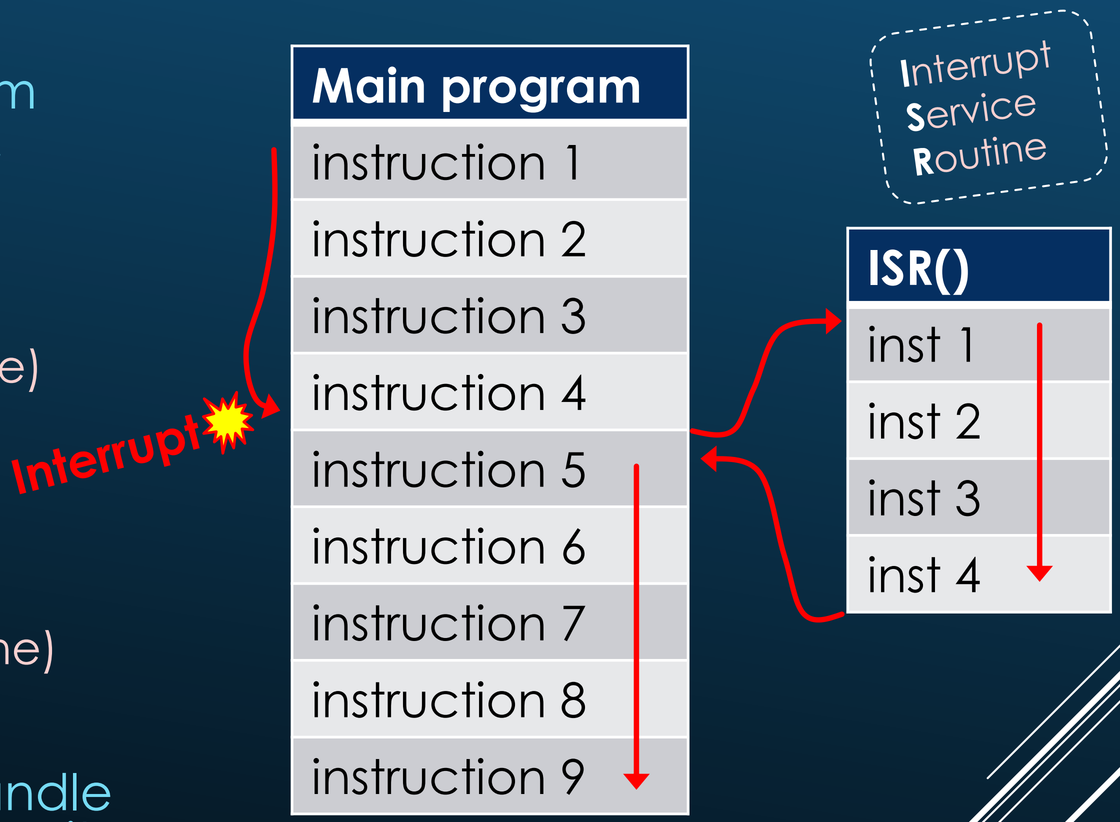
- Pins programmable as Input and Output
- Read / Write digital signals
- '0' = 0V (Gnd), '1' = 5V (Vcc)

- Controlled by 3 registers:
- DDR (Data Direction Register)
- PORT (Where you write when it's output)
- PIN (Where you read when it's input)

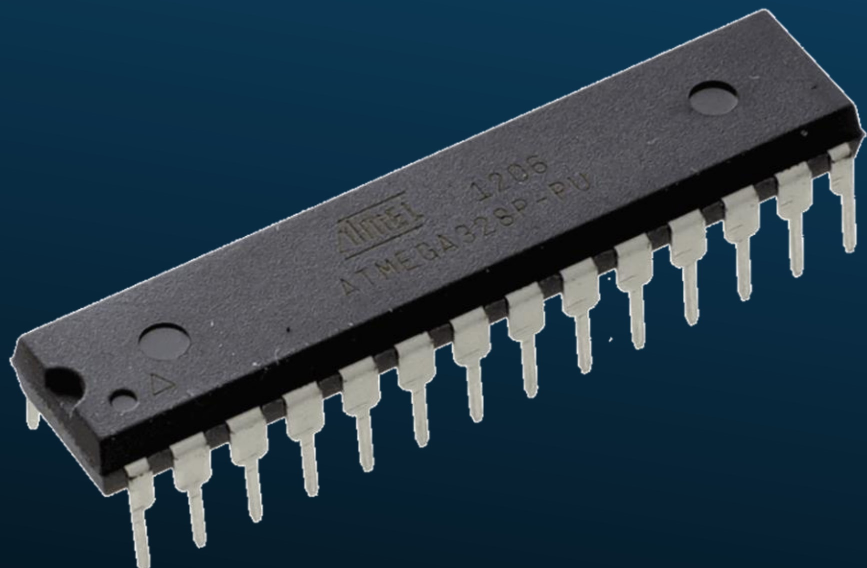


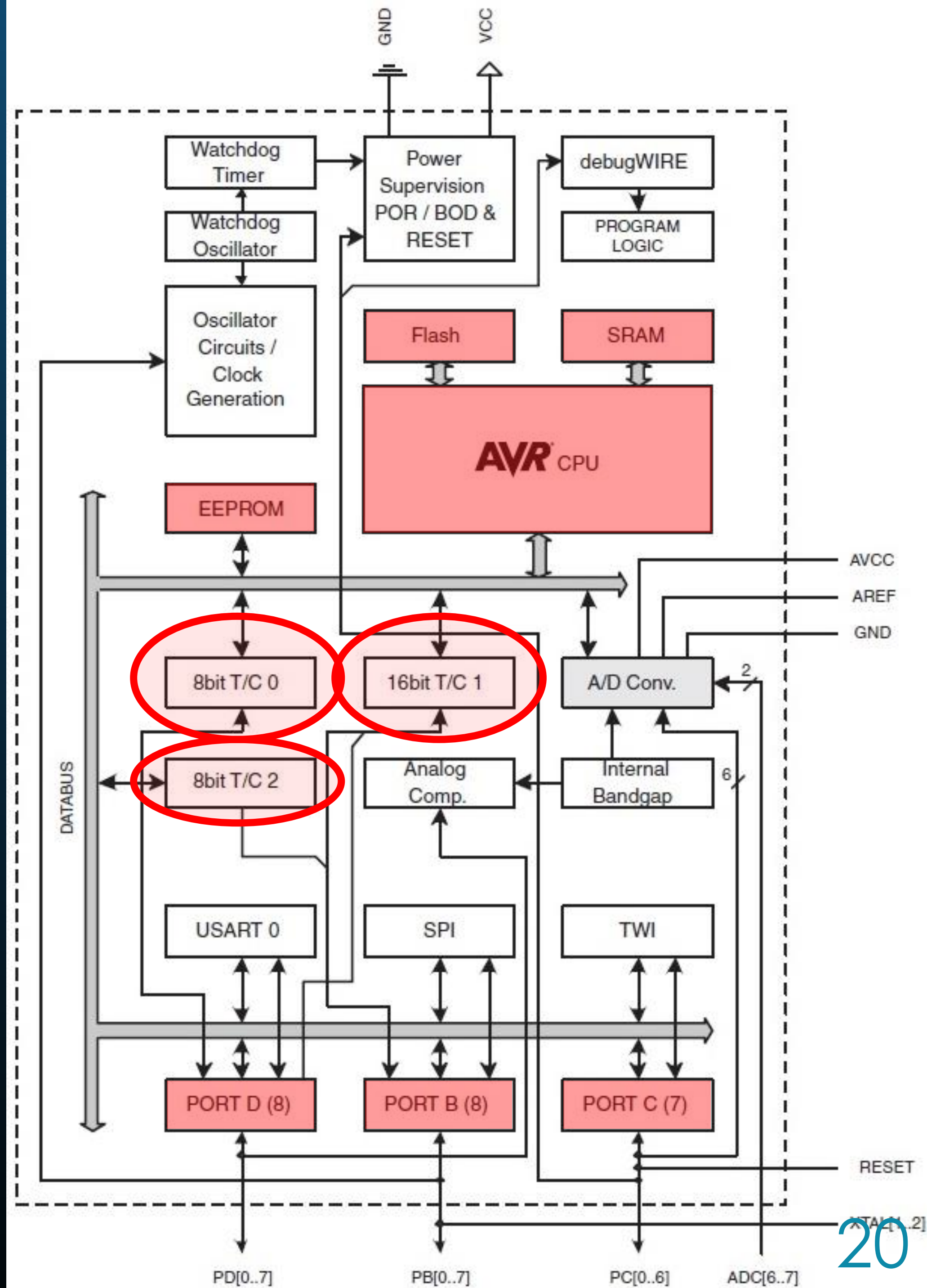
INTERRUPT

- Interrupts break the program flow to handle some event.
- It may be triggered by:
  - Pin change (rise/fall/toggle)
  - Timers / Counters
  - Analog Comparator
  - ADC reading done
  - Serial interfaces (Rx/Tx done)
- It allows the program to handle an event "right after" its occurrence, regardless of where the program is and without the need of polling constantly.



AVR ARCHITECTURE (ATMEGA328P)





TIMERS / COUNTERS

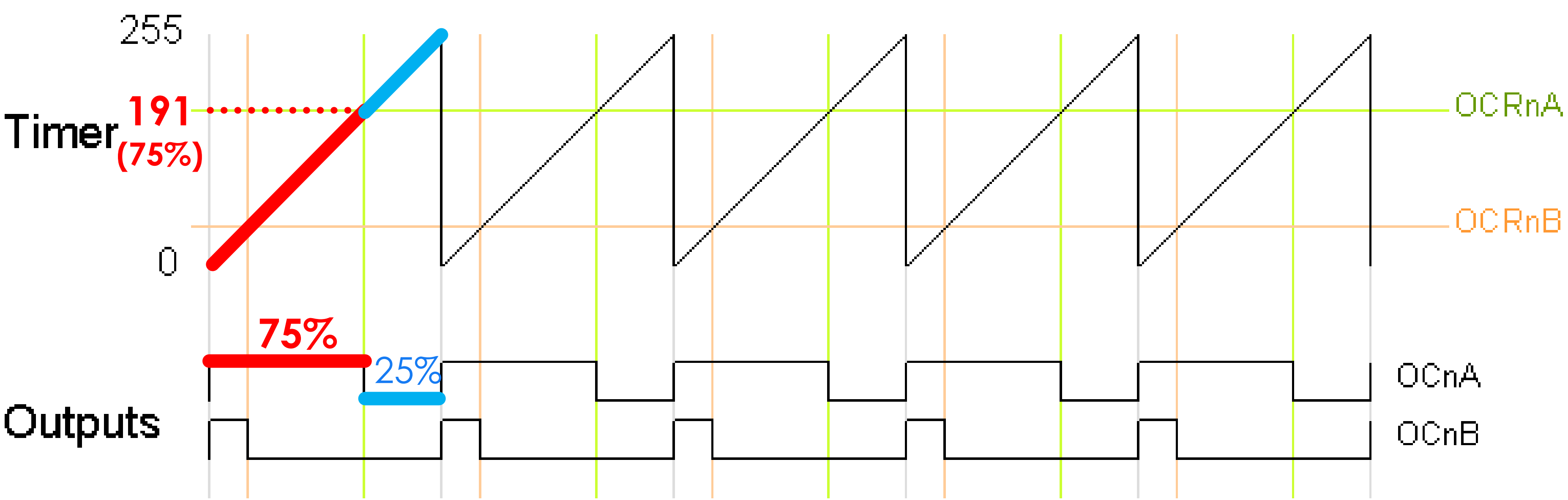
- Internal registers that increment triggered by:
  - A clock source: **Timer**
  - An external event: **Counter**
- May be used to:
  - Measure time
  - Raise interruption on:
    - Overflow
    - Reach a certain value (OCR)
  - Create waveform
    - PWM

Output **C**ompare **R**egister

Ultrasonic distance sensor
Measures distance based on the time to echo of an ultrasonic pulse.

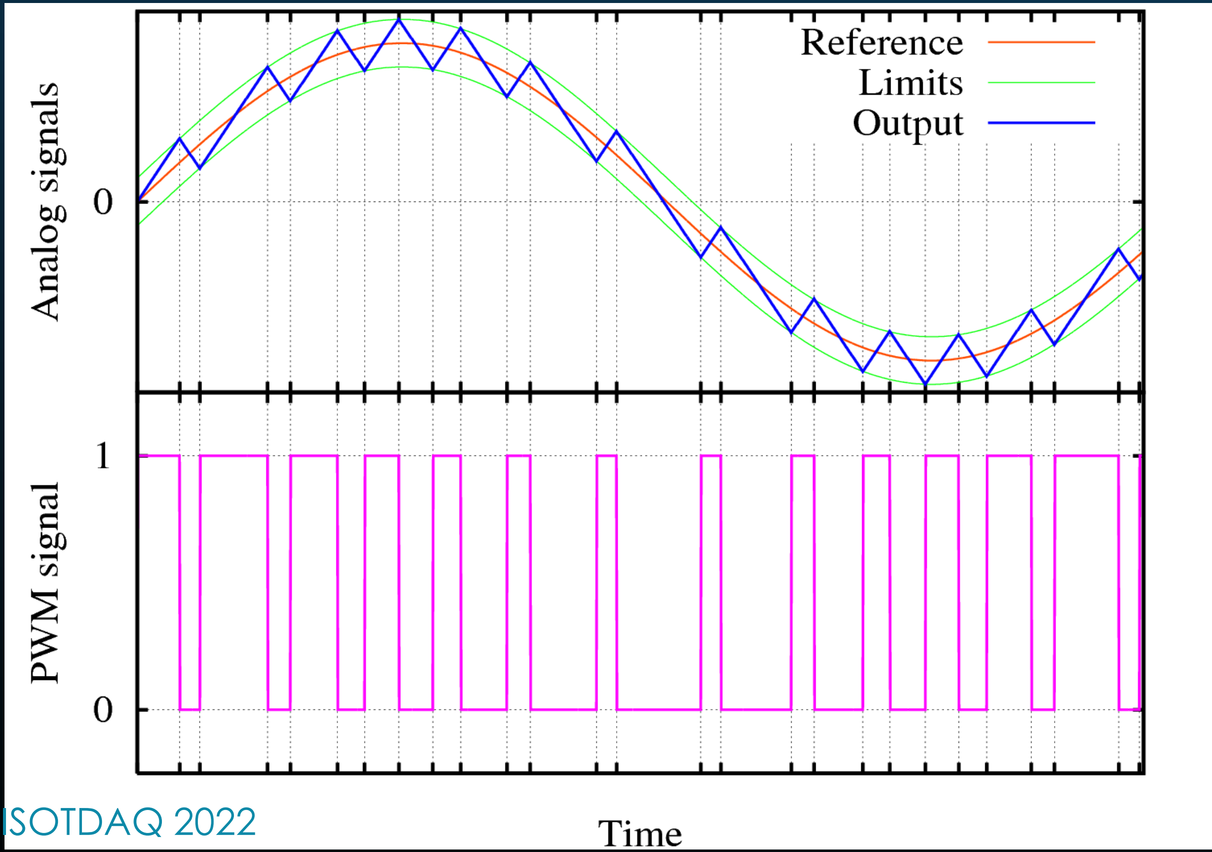
PULSE WIDTH MODULATION (PWM)

- You can create an output signal which value depends on the status of the timer.
- Outputs a train of periodic digital pulses with controlled width.
  - (Can be used to "mimic" an analog signal)

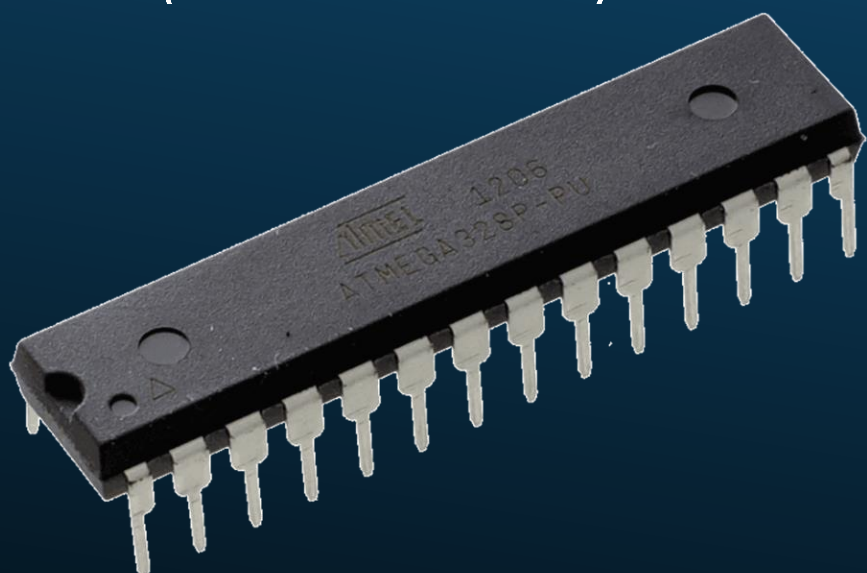


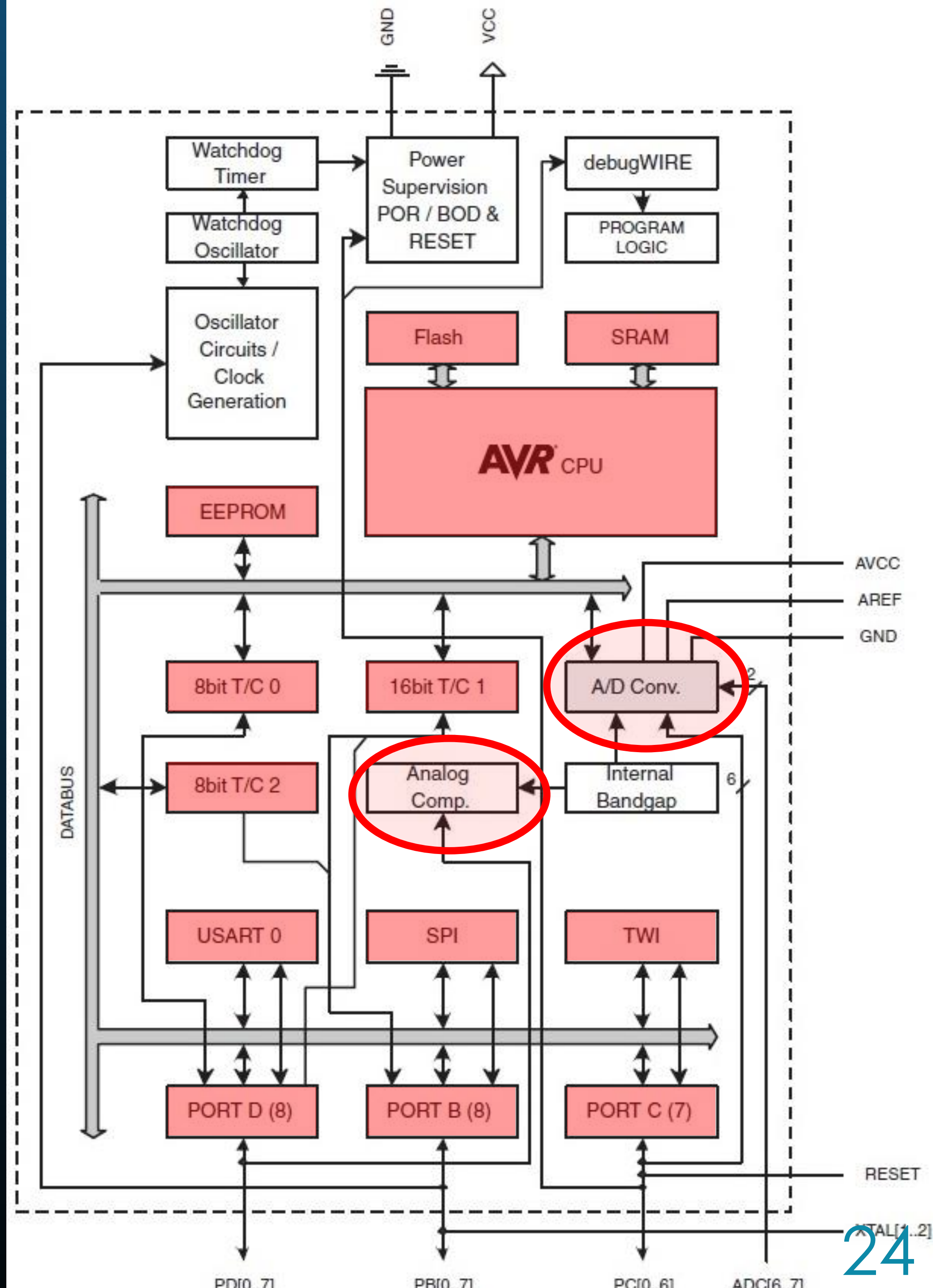
PULSE WIDTH MODULATION (PWM)

- You can create an output signal which value depends on the status of the timer.
- Outputs a train of periodic digital pulses with controlled width.
  - (Can be used to "mimic" an analog signal)



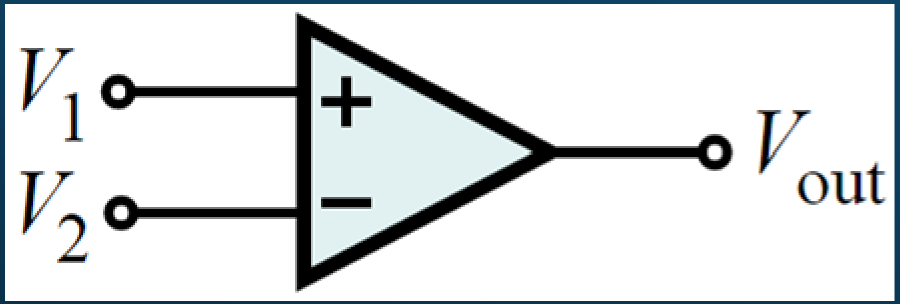
AVR ARCHITECTURE (ATMEGA328P)





ANALOG COMPARATOR

- Tells whether positive pin AIN0 voltage is higher than negative pin AIN1.
- Output is the internal bit ACO* of reg ACSR*.
- Can be used to:
  - Compare two analog signals
  - Trigger a Timer/Counter
  - Trigger an interrupt (rise, fall or toggle)



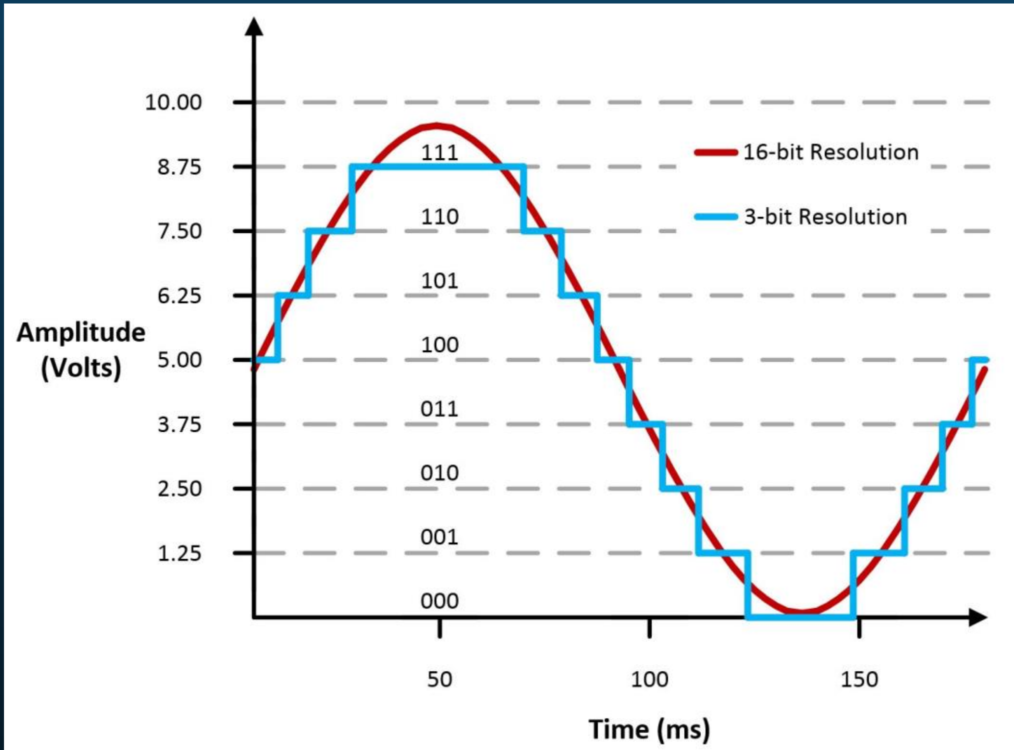
$V_1 > V_2 \longrightarrow V_{out} = 1$

$V_1 < V_2 \longrightarrow V_{out} = 0$

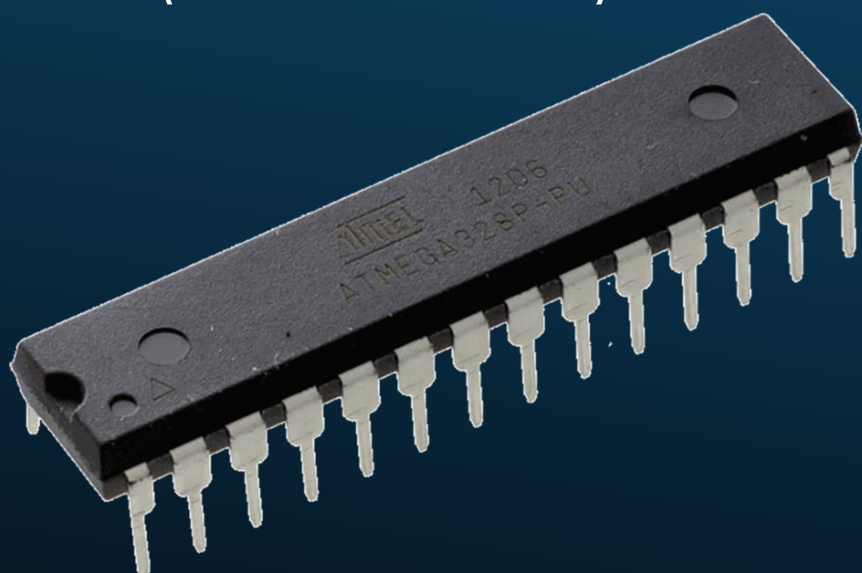
* **ACO** = Analog Comparator Output
ACSR = Analog Comp. Control and Status Register

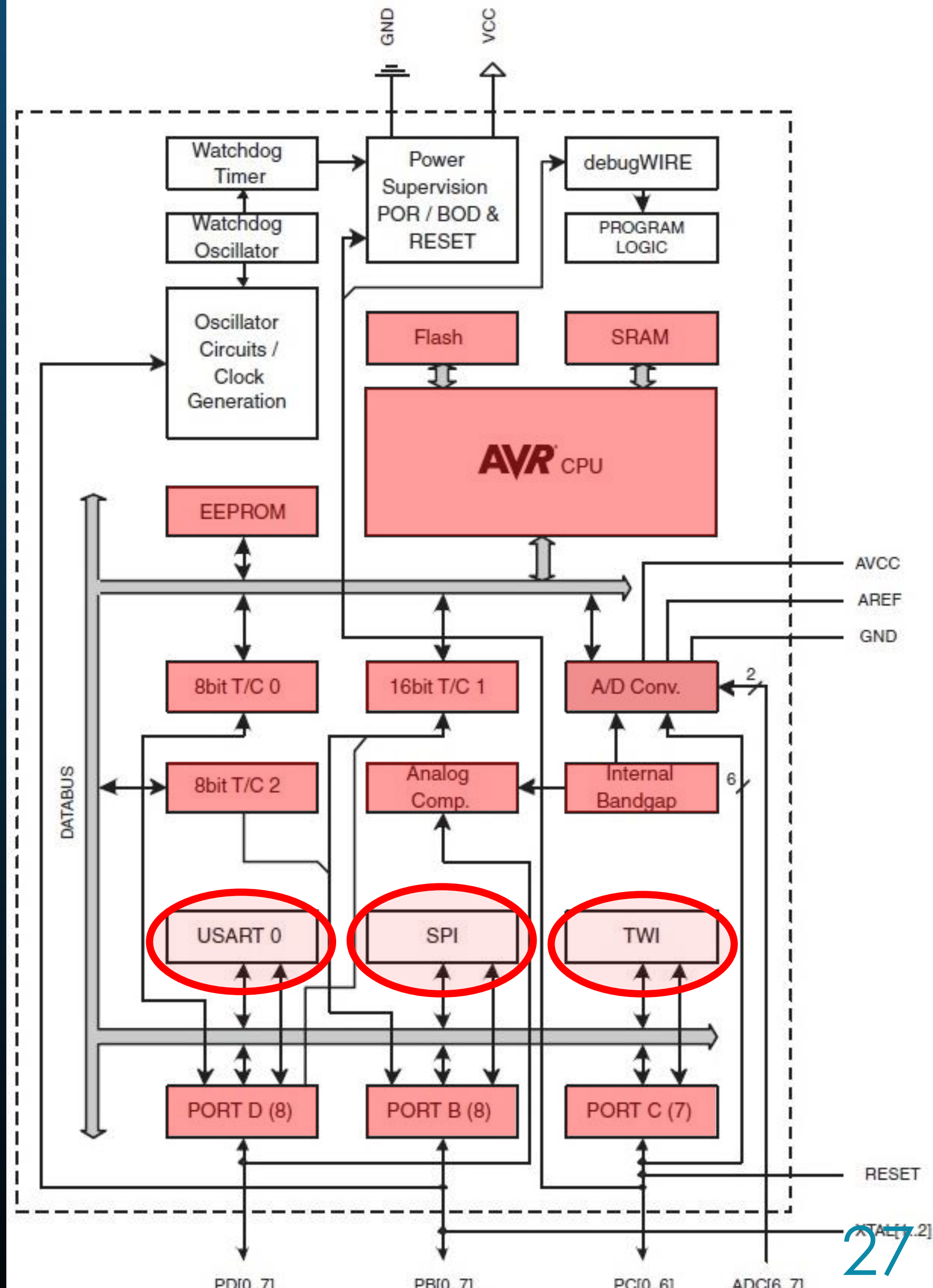
ANALOG TO DIGITAL CONVERTER (ADC)

- 10-bit resolution
  - 0V-Vref → 0-1023
- Vref can be:
  - Vcc (Power source)
  - 1.1V internal ref.
    (from bandgap)
  - External ref. on pin 'AREF'
- Successive approximation
  - 13-260 us Conversion time
- Interrupt on complete
- 6 multiplexed channels on DIP package
  - (internal Temp sensor on ch8)



AVR ARCHITECTURE (ATMEGA328P)

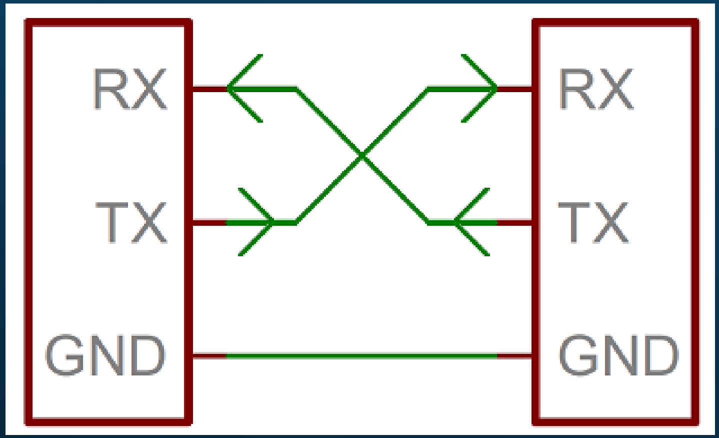


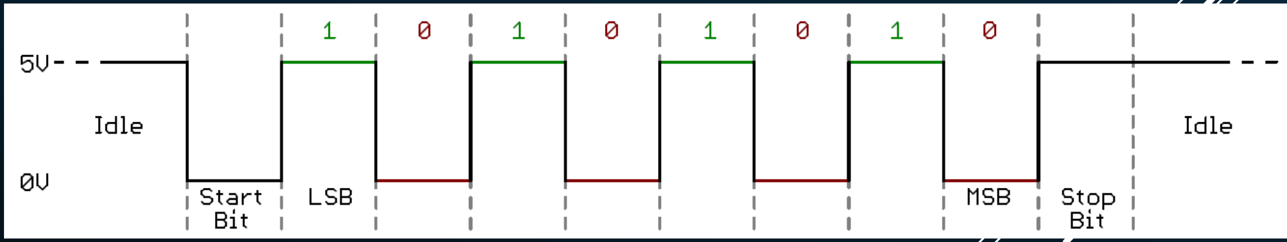


SERIAL INTERFACES: USART

UNIVERSAL SYNCHRONOUS-ASYNCHRONOUS RECEIVER TRANSMITTER

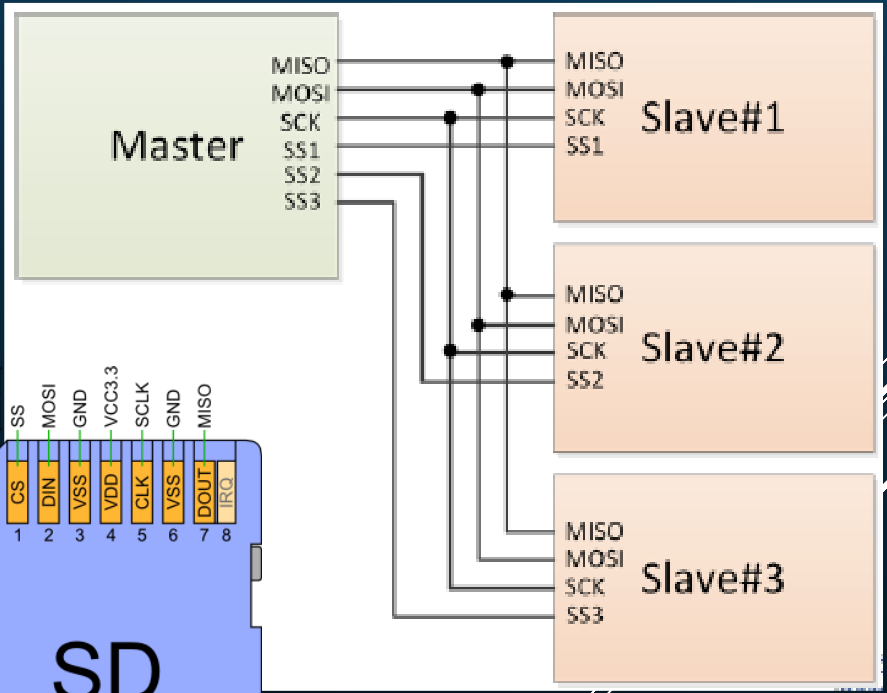
- A simple protocol
- Widely used to communicate with PCs due to compability with RS232 protocol. (RS232 is not used anymore in most PCs but it's still very easy to find USB-Serial converters)
- Up to 250kbps
- May trigger interrupts:
  - Tx complete
  - Rx complete
  - Data reg empty

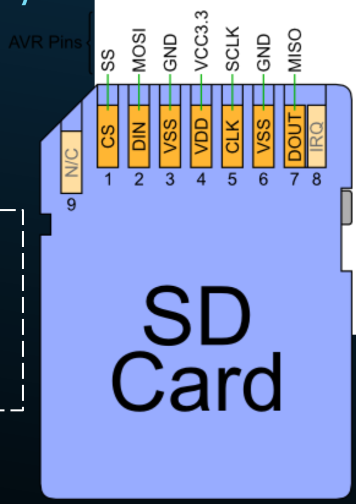




SERIAL INTERFACES: SPI
SERIAL PERIPHERAL INTERFACE

- Differently from the USART, SPI can talk to multiple devices on the same bus, but needs a Slave Select signal per Slave Device
- Up to 10Mbps! (clk/2)
- Slaves do not "talk" autonomously.
  - Must be querried (and clocked) by master.



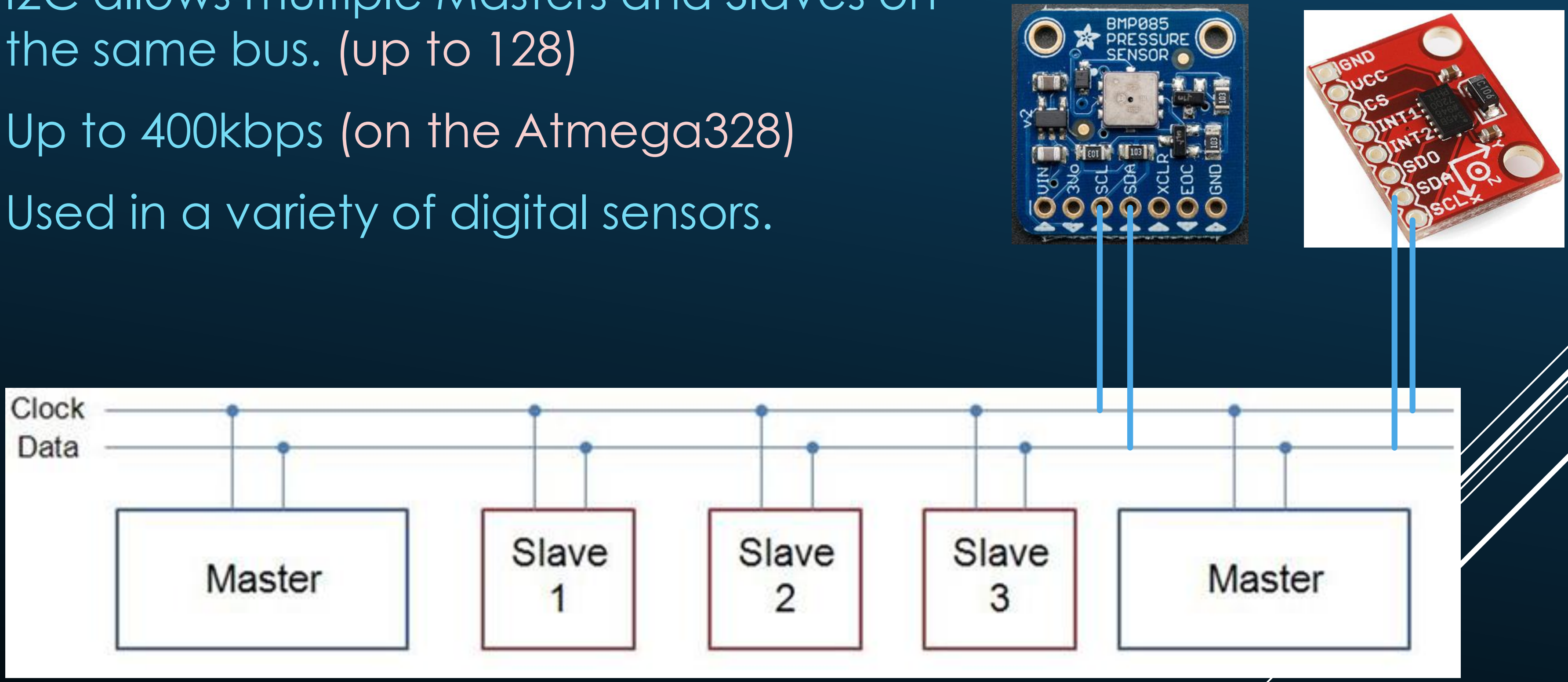


All SD cards
Uses the SPI serial interface and can be easily accessed from a uC.

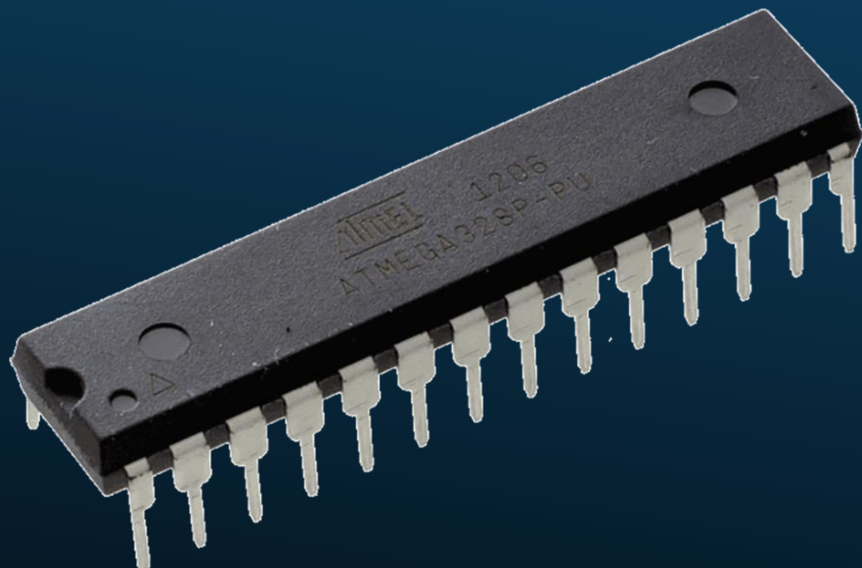
SERIAL INTERFACES: TWI (I^2C)

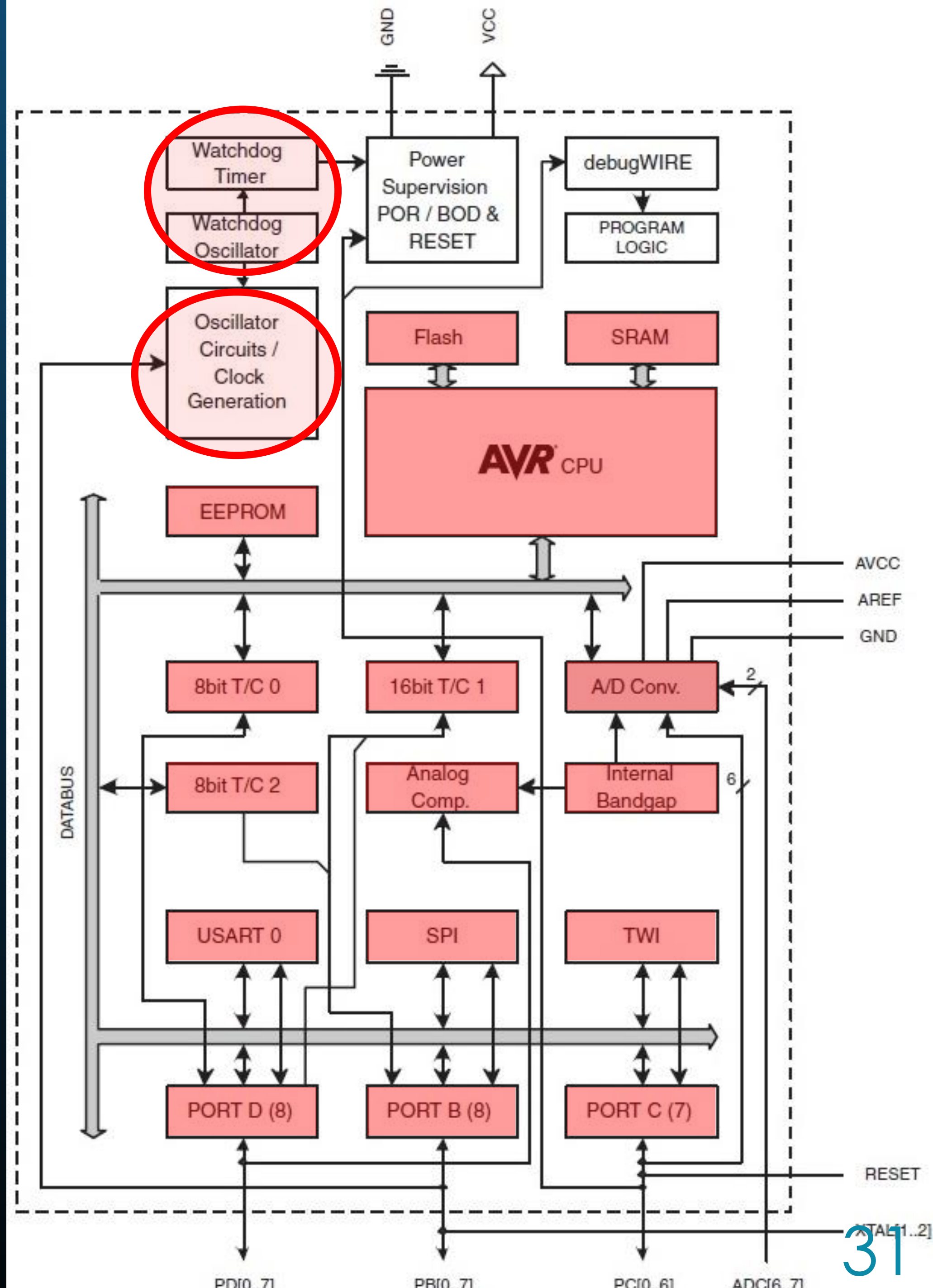
TWO WIRE INTERFACE (INTER-INTEGRATED CIRCUIT)

- I2C allows multiple Masters and Slaves on the same bus. (up to 128)
- Up to 400kbps (on the Atmega328)
- Used in a variety of digital sensors.



AVR ARCHITECTURE (ATMEGA328P)





WATCHDOG TIMER (WDT)

- A Watchdog Timer is a timer clocked by an on-chip oscillator
- Once the counter reaches a certain value, the microcontroller may:
  - Trigger an interrupt
  - Reset the microcontroller
- Used to prevent your program from getting stuck in any part of the code.
- You use it by enabling the WDT and spreading WDT reset instructions on particular places of your code.
  - If it gets stuck in an infinite loop, for ex., the counter won’t be reset and the microcontroller will be reset.

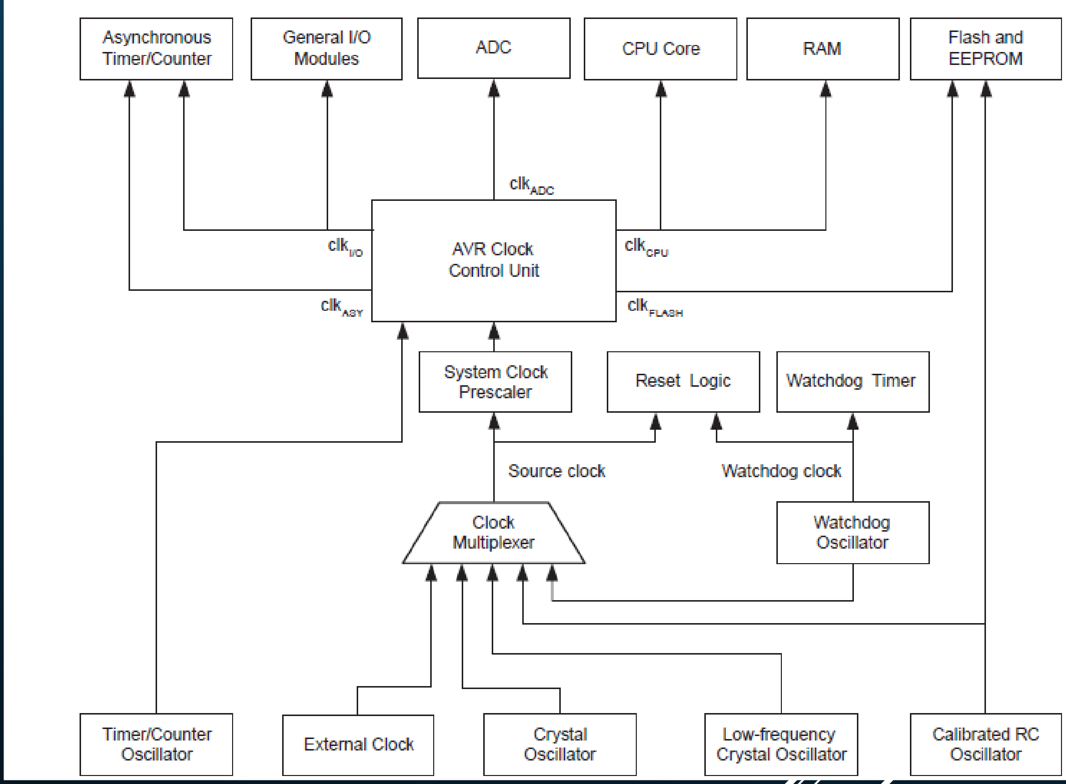
CLOCK CIRCUIT

- Up to 20MHz from:
  - External clock from a pin
  - External crystal oscillator
  - Internal RC oscillator
    - 7.3-8.1 MHz
  - 128kHz Internal oscillator
    - 128 kHz

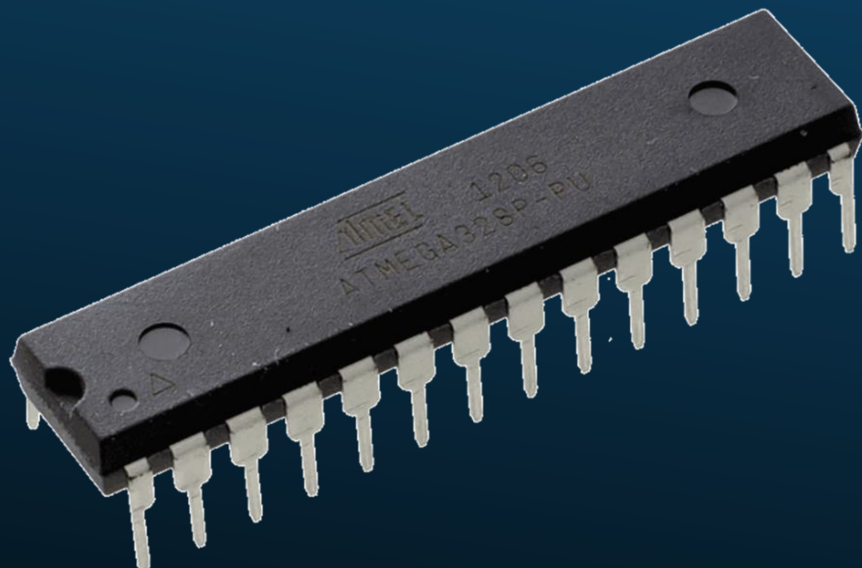
- System Clock Prescaler
  - Divides the clock if needed
- Keep in mind:

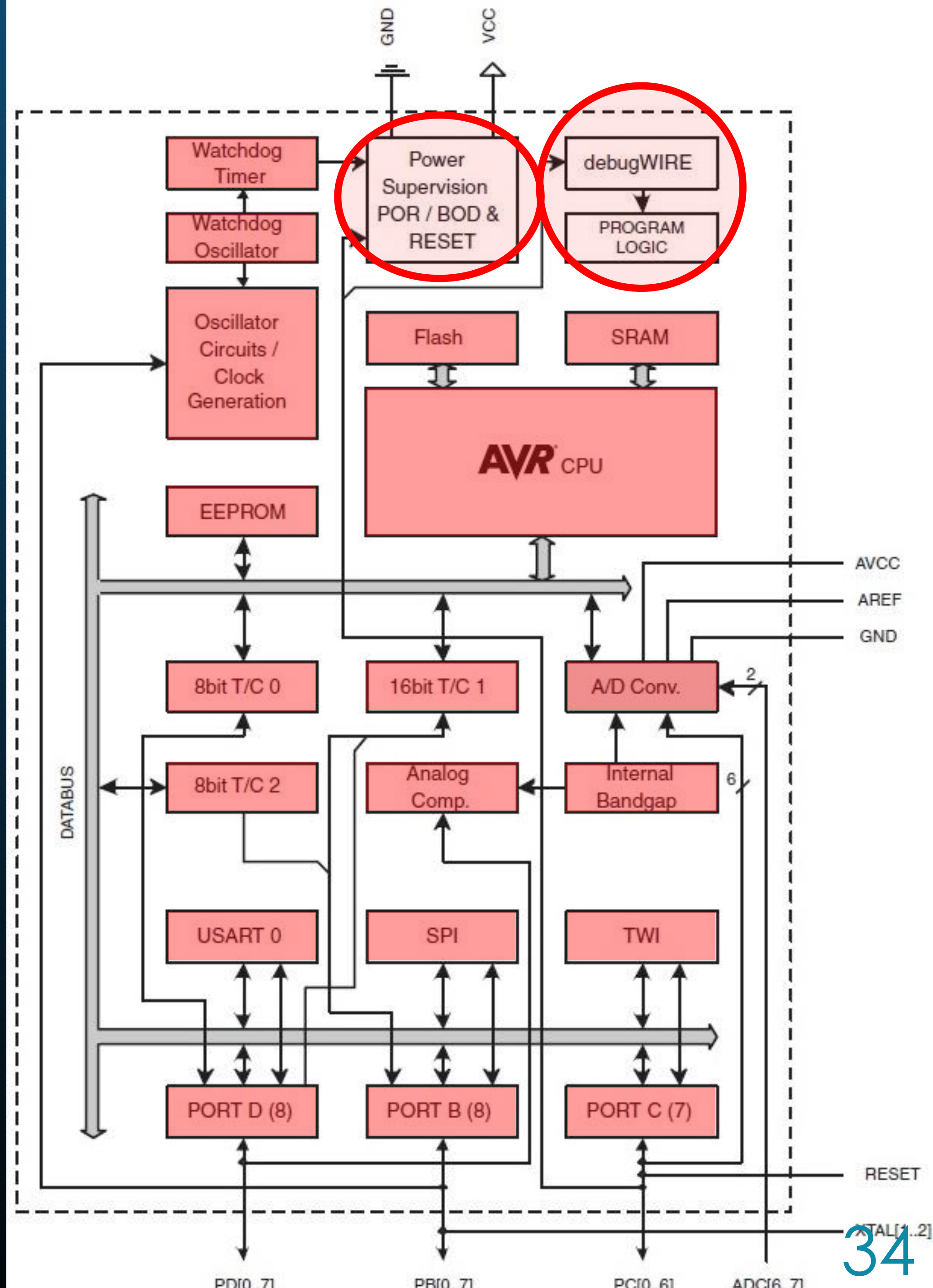
  Power consumption is proportional to clock frequency.

Figure 8-1. Clock Distribution



AVR
ARCHITECTURE
(ATMEGA328P)





SLEEP MODES

- There are multiples Sleep Modes available. Each turns off certain parts of the microcontroller to save power and can only be waken up by certain sources.

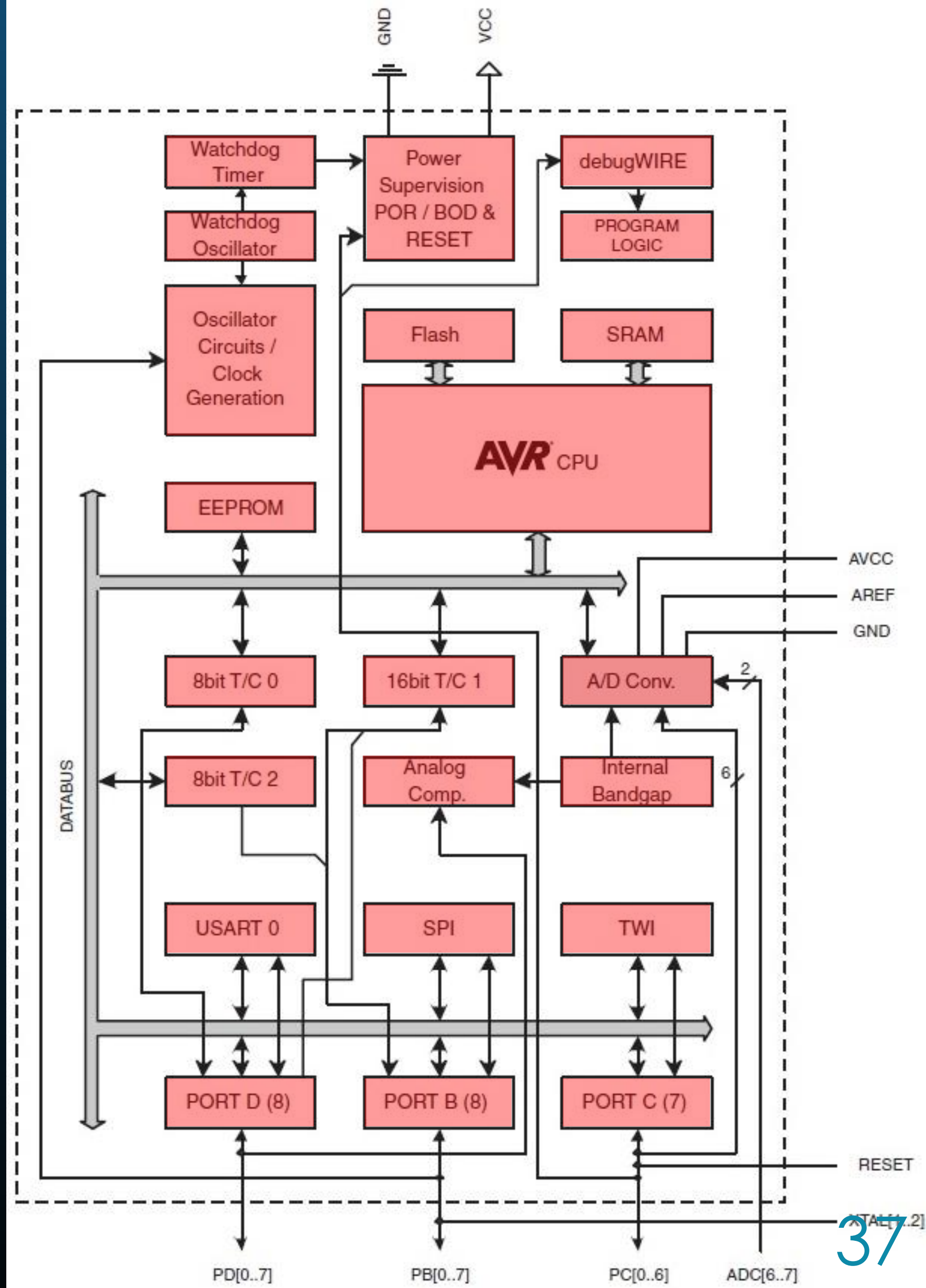
Symbol	Parameter	Condition	Min.	Typ.[2]	Max.	Units
	Power-down mode[3]	WDT enabled, V_{CC} = 3V		4.2	8	µA
		WDT disabled, V_{CC} = 3V		0.1	2	µA

	Active Clock Domains					Oscillators		Wake-up Sources							
Sleep Mode	clk_{CPU}	clk_{FLASH}	clk_{IO}	clk_{ADC}	clk_{ASY}	Main Clock Source Enabled	Timer Oscillator Enabled	INT1, INT0 and Pin Change	TWI Address Match	Timer2	SPM/EEPROM Ready	ADC	WDT	Other I/O	Software BOD Disable
Idle			X	X	X	X	$X^{(2)}$	X	X	X	X	X	X	X	
ADC Noise Reduction				X	X	X	$X^{(2)}$	$X^{(3)}$	X	$X^{(2)}$	X	X	X		
Power-down								$X^{(3)}$	X				X		X
Power-save					X		$X^{(2)}$	$X^{(3)}$	X	X			X		X
Standby[1]						X		$X^{(3)}$	X				X		X
Extended Standby					$X^{(2)}$	X	$X^{(2)}$	$X^{(3)}$	X	X			X		X

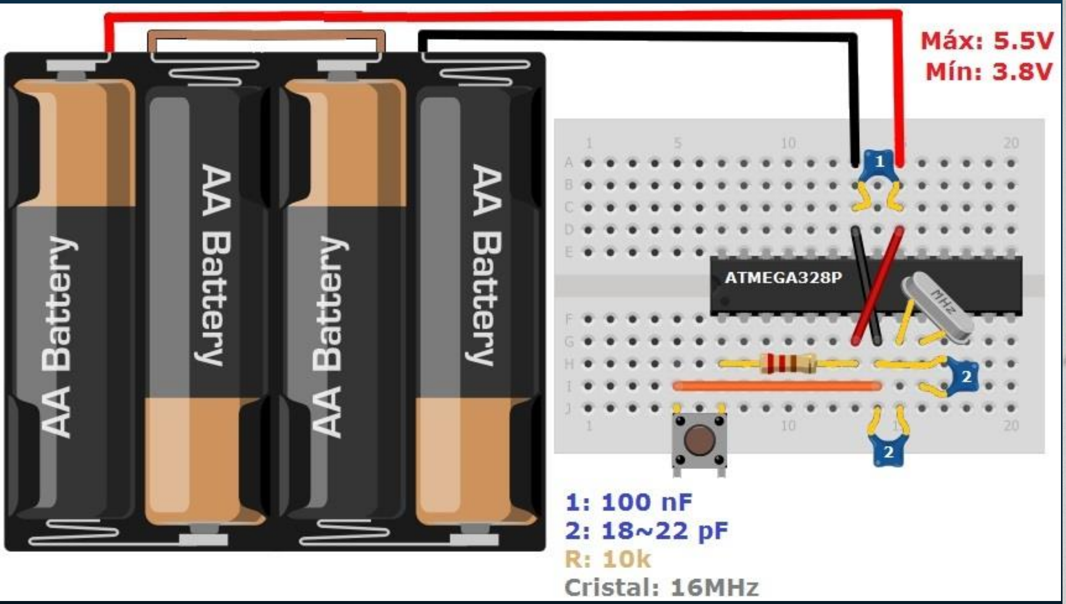
POWER AND DEBUG

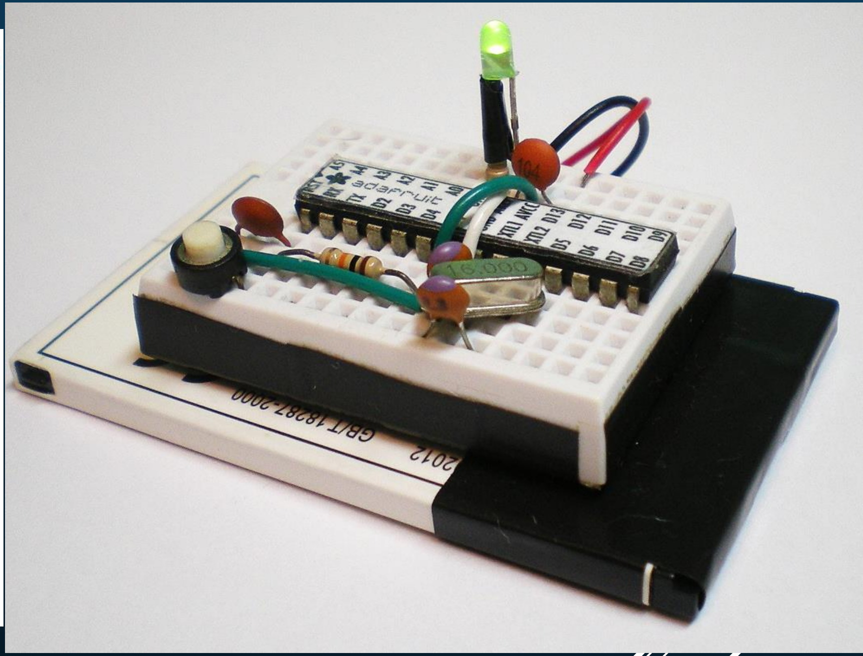
- Brown-Out Detector (BOD)
  - Resets the device whenever Vcc is below a certain threshold.
- Power-on Reset (POR)
  - Ensures the device is reset from Power On.
- DebugWIRE
  - On-chip debug tool from AVR.

REVIEW


GND
VCC
Watchdog Timer
Watchdog Oscillator
Oscillator Circuits / Clock Generation
Power Supervision POR / BOD & RESET
debugWIRE
PROGRAM LOGIC
Flash
SRAM
AVR CPU
EEPROM
AVCC
AREF
GND
DATABUS
8bit T/C 0
16bit T/C 1
A/D Conv.
2
8bit T/C 2
Analog Comp.
Internal Bandgap
6
USART 0
SPI
TWI
PORT D (8)
PORT B (8)
PORT C (7)
RESET
XTAL[1..2]
PD[0..7]
PB[0..7]
PC[0..6]
ADC[6..7]

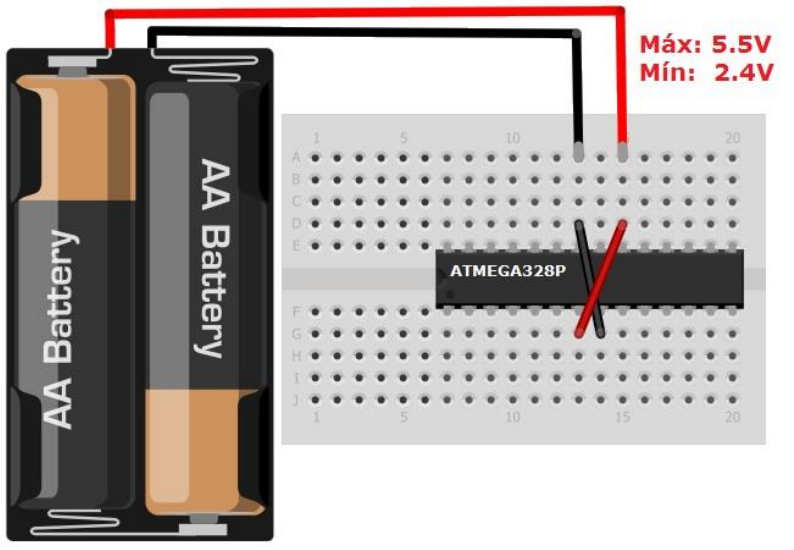
ATMEGA328 MINIMUM REQUIRED CIRCUIT
NEEDED CIRCUITRY: USING AN EXTERNAL CRYSTAL

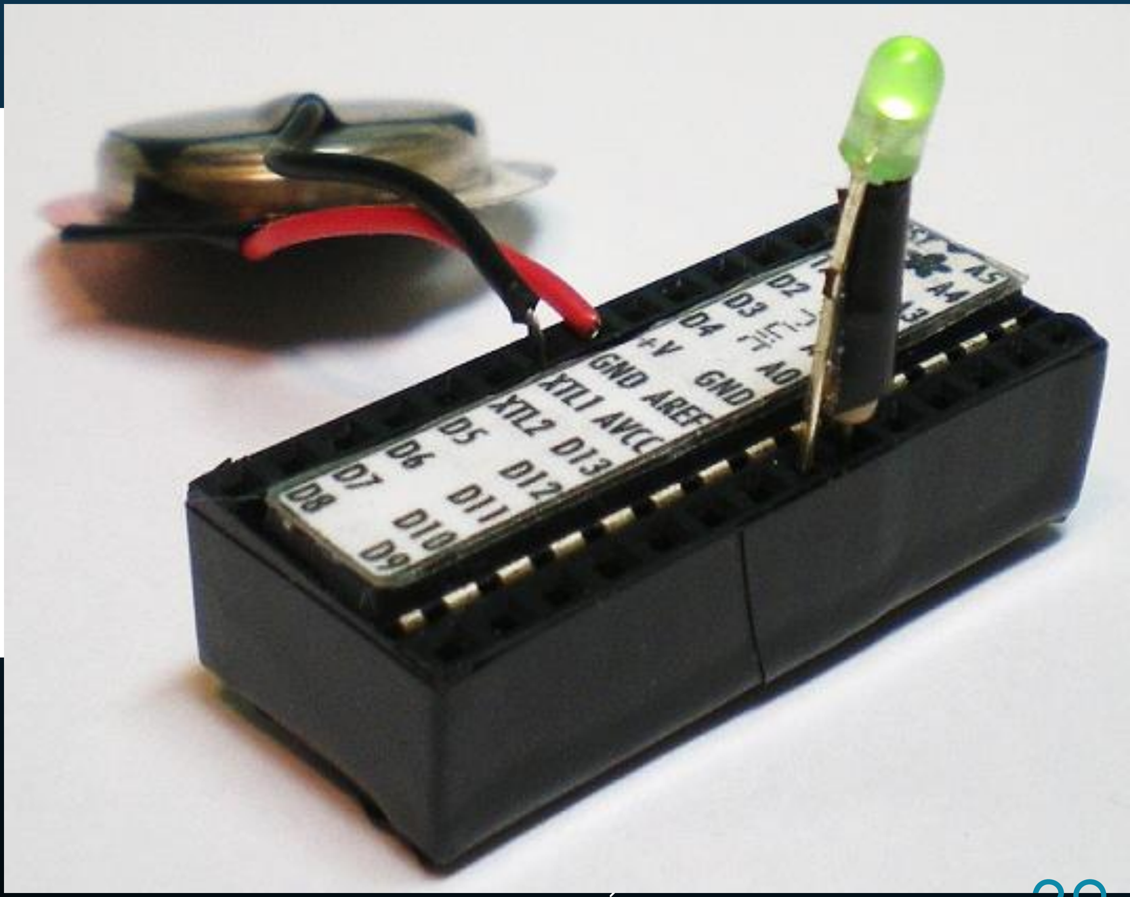




ATMEGA328 MINIMUM REQUIRED CIRCUIT
NEEDED CIRCUITRY: USING THE INTERNAL OSCILLATOR

- Atmega328 comes with an internal 8MHz oscillator that can minimize the required circuit to a single battery.

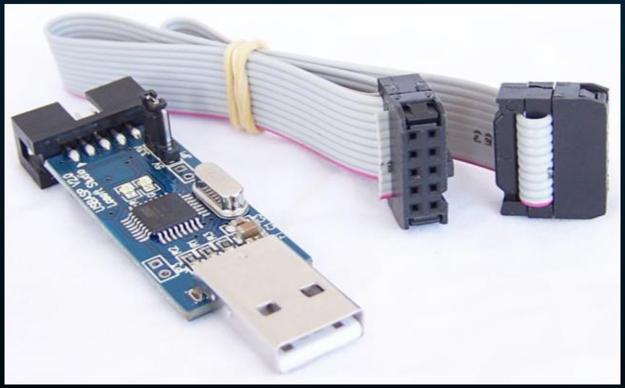




USAGE OF MICROCONTROLLERS

DEVELOPMENT CYCLE

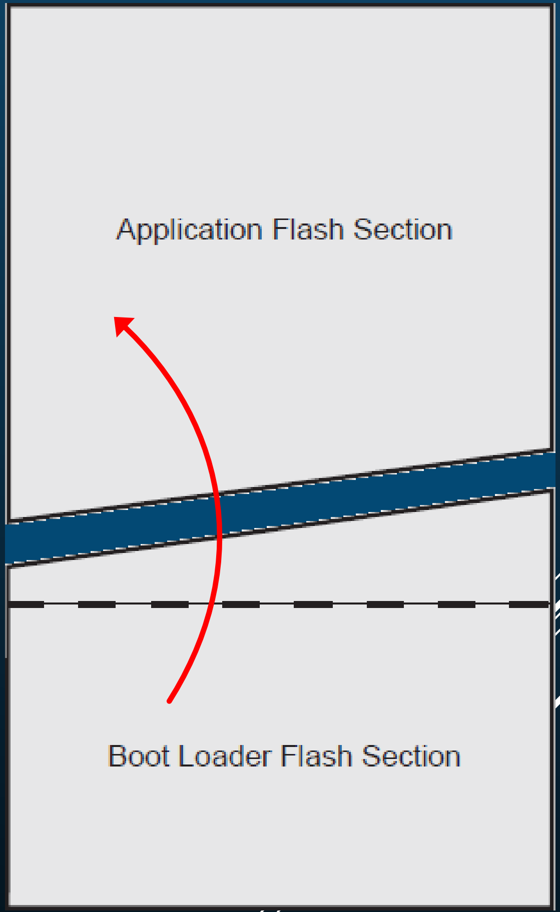
- Write your code. From Assembly to C. Or even higher level:
- Compile it. (debug)
- Upload to the uC memory (On Chip debug)
  - Parallel programming
  - Serial downloading (SPI)
  - Self-programming (With bootloader)
- (Burn the fuses)



SELF-PROGRAMMING: BOOTLOADER

- The AVR core can write to it's own program memory.
- The Bootloader Section can be locked from rewritting.
- This way developers can allow users to write their own programs without compromising the bootloader section.
- This can also be used to ease the programming of the memory, eliminating the need for an external programmer.

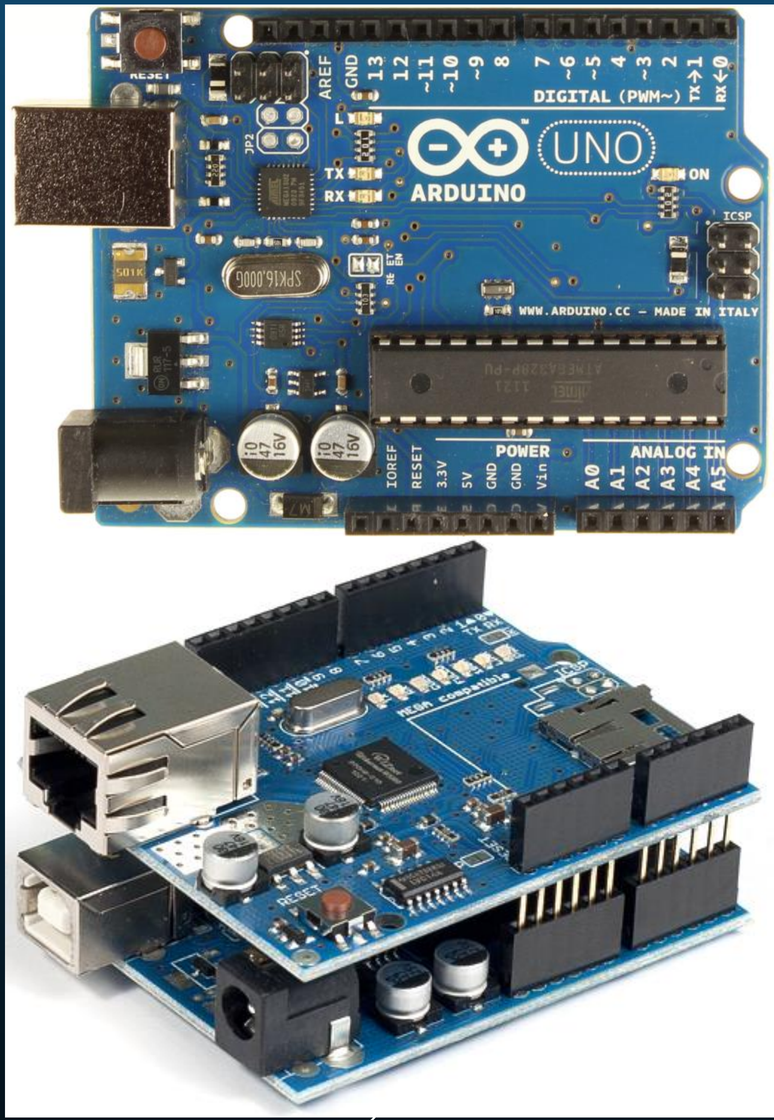
USB-Serial



ARDUINO

ARDUINO

- Open-source platform created for "makers" who have no knowledge in electronics but still want their creations to interact with the environment.
- Custom IDE + libraries
- Inexpensive and really easy to use
  - (Almost plug and play)
- Huge community all over the world creating libraries, compatible hardware and sharing projects
- Stackable addons called shields



SETTING PWM TO 25% DUTY CYCLE

THE ~~MEDIEVAL~~ ORIGINAL WAY

Registers:

DDR = Data Direction Register
TTCR0A = Timer/Counter Control Register A
TTCR0B = Timer/Counter Control Register B
OCR0A = Output Compare Register 0 A


```
DDRD  |= 0x40;

TCCR0A  |= (1<<WGM00)  |(1<<WGM01);

TCCR0A  |= (1<<COM0A1)|(1<<COM0A0);

OCR0A  = 0x3F;

TCCR0B  |= (1<<CS00);
```

Set direction of pin 6 from PORTD to output.

Set WGM to Fast PWM.

Set COM to «Clear On Compare».

Set OCR to 25% (0x3F of 0xFF)

Set the clock source to timer.

Bits:

WGM0[1..0] = Waveform Generator Mode 0
COM0A[1..0] = Compare Output Mode 0 A
CS0[2..0] = Clock Select 0

SETTING PWM TO 25% DUTY CYCLE

THE ~~CHEATING~~ ARDUINO WAY

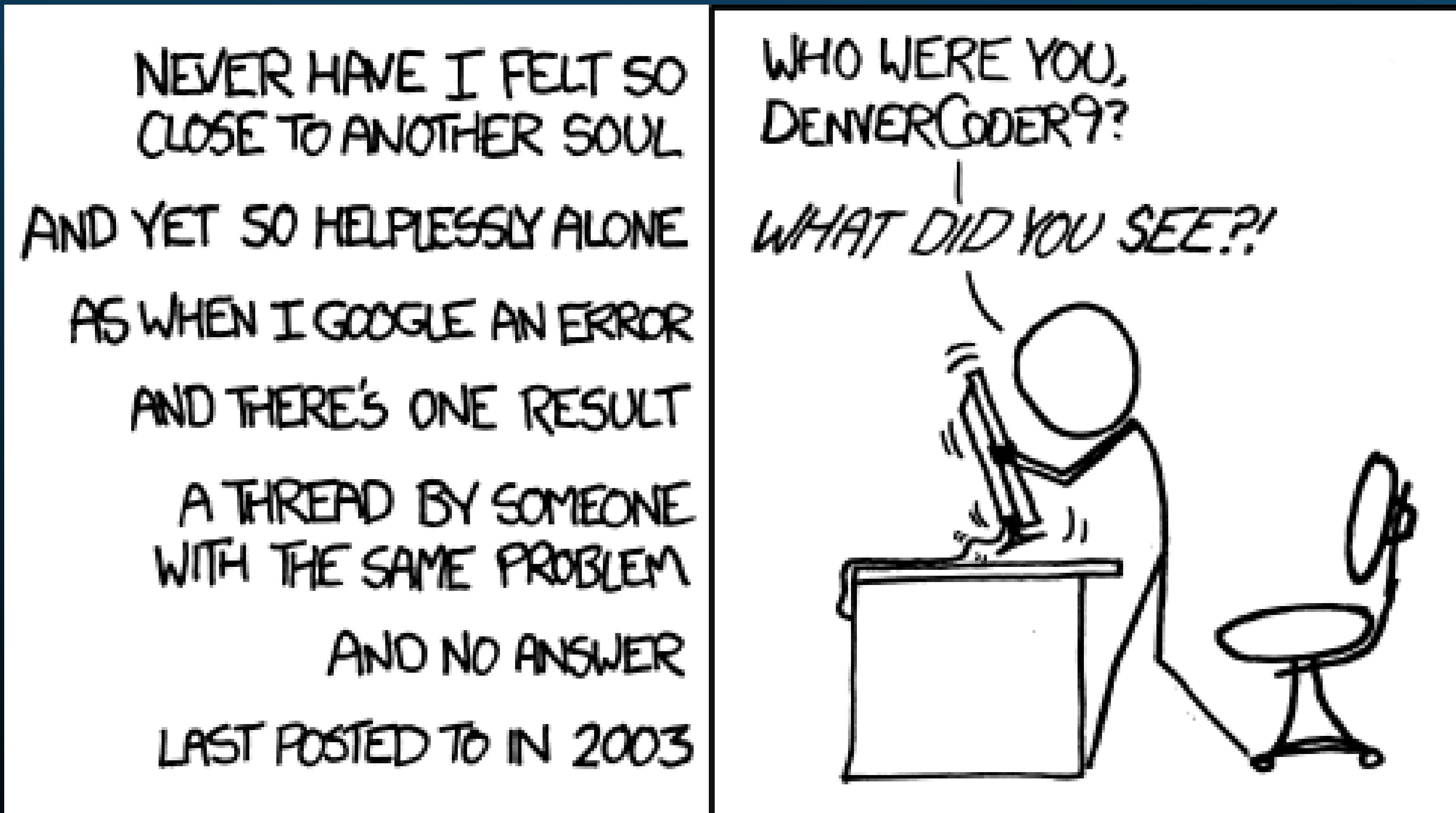
Look at the board which pin you want to use.

```
analogWrite(6, 63);
```

$$\frac{25}{100} = \frac{x}{255}$$

Find x.

ADVANTAGES OF PROTOTYPING PLATFORMS



http://xkcd.com/979/

ADVANTAGES OF PROTOTYPING PLATFORMS

PRO	CON
Community / Support / StackOverflow	Performance is generally not good
Much easier to learn	No full control over the code
Fast development and Prototyping	Cost is higher
Portable code between supported devices	TRU ENGINEERS gon' make fun of u cos ur not BRAVE ENOUGH to handle raw bits.

USAGE OF MICROCONTROLLERS

FIRST OF ALL

- Is a microcontroller suitable for my application?
  - Cost
  - Development time
  - Power consumption
  - Processing power
  - Timing requirements
  - Etc.

USAGE OF MICROCONTROLLERS

CHOOSING A MICROCONTROLLER

- What kind of problem do I have?
  - Processing intensive? Power limitation? Embedded?
- Which kind of sensors/actuators will be used?
  - Digital/Analog? Voltage levels?
- What are the required peripherals?
  - USB? I2C? ADCs? Timers?
- What is the enviroment?
  - Space? Right next to the LHC beam?
    (Temperature, radiation, etc.)

Alternatives?

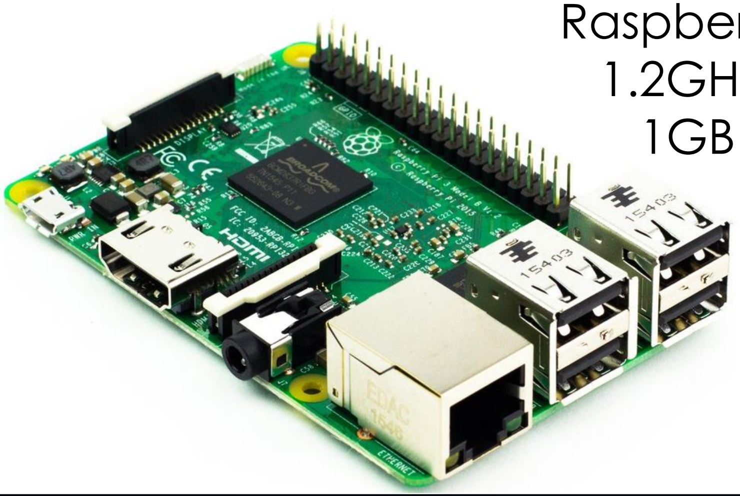
EMBEDDED SYSTEMS

- Microcontroller
- FPGA – Field Programmable Gate Array
- DSP – Digital Signal Processor
- SoC – System On a Chip
- Single Board Computer
- ASIC – Application Specific Integrated Circuit

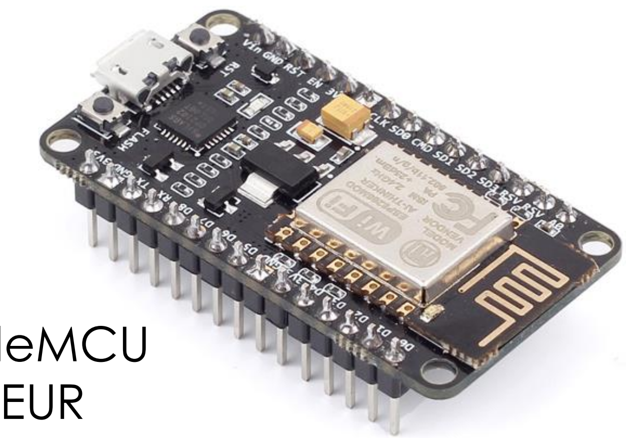
EMBEDDED SYSTEMS

FPGA Xilinx Spartan 6
Ethernet Controller + USB

Raspberry Pi Zero
SoC (no FPGA)
$5 USD
1GHz
512MB RAM



Raspberry Pi 3
1.2GHz 64-bit quad-core
1GB RAM
Wifi + Bluetooth

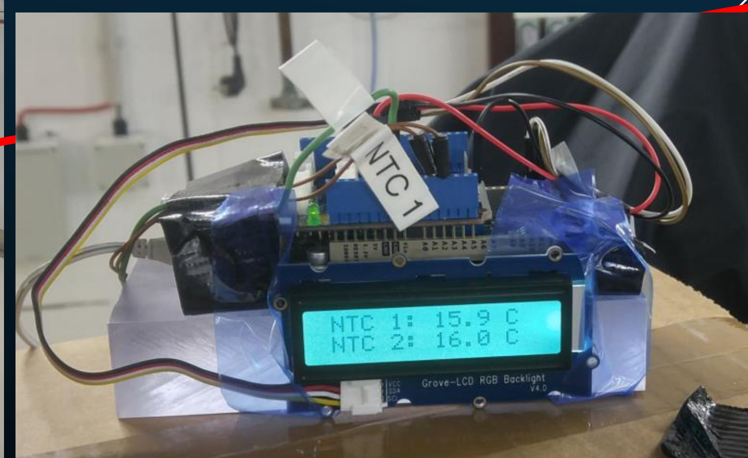


NodeMCU
~$7 EUR
Wifi Microcontroller

APPLICATIONS IN HEP ENVIRONMENTS

Arduino as a delay control unit,

as a temperature monitor,

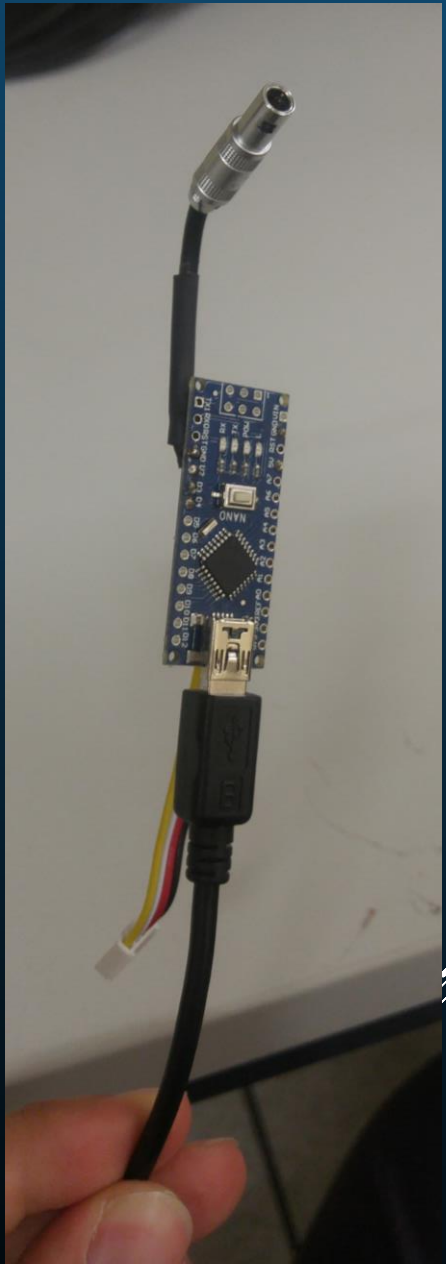


APPLICATIONS IN HEP ENVIRONMENTS

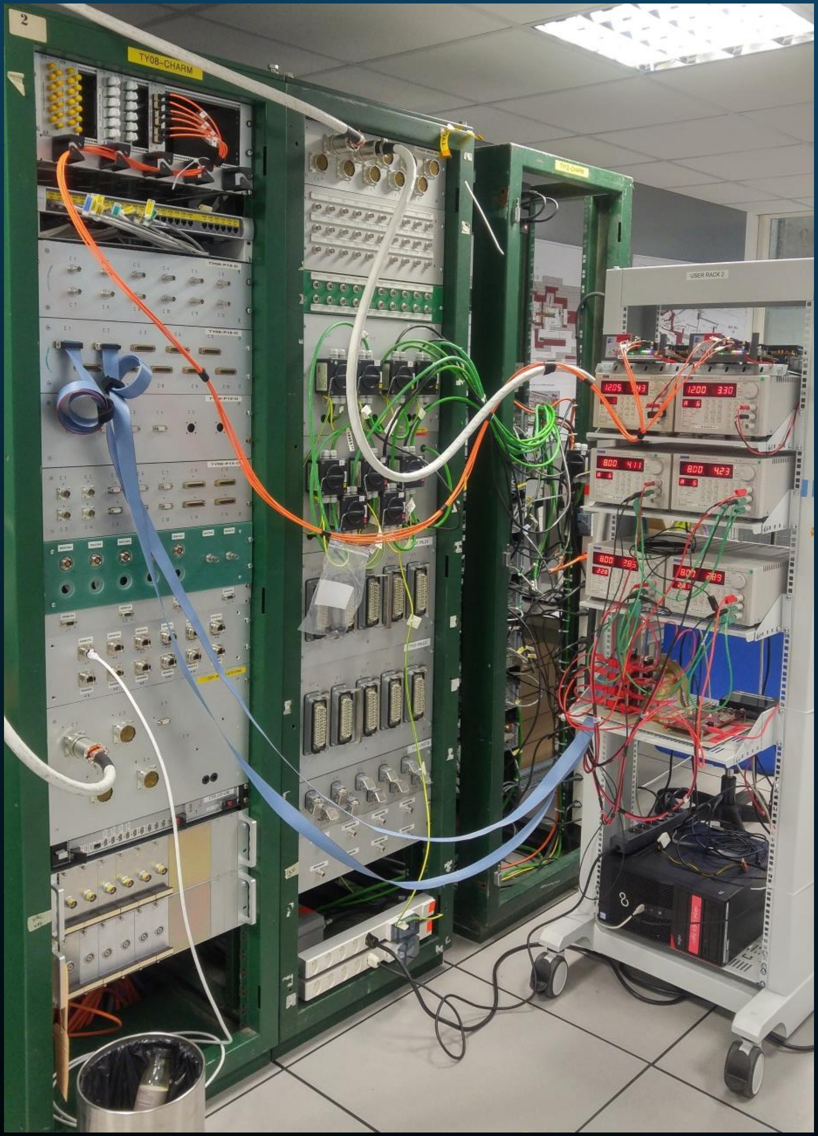
Arduino as a delay control unit,

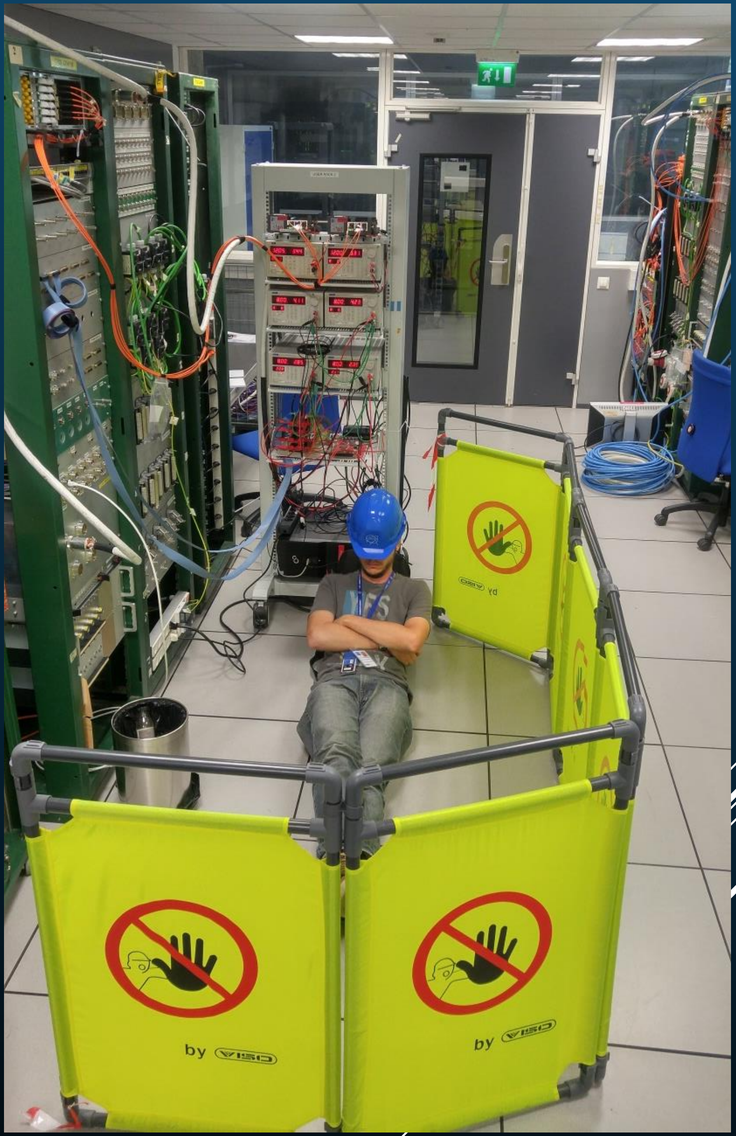
as a temperature monitor,

as an USB TTL/NIM generator.

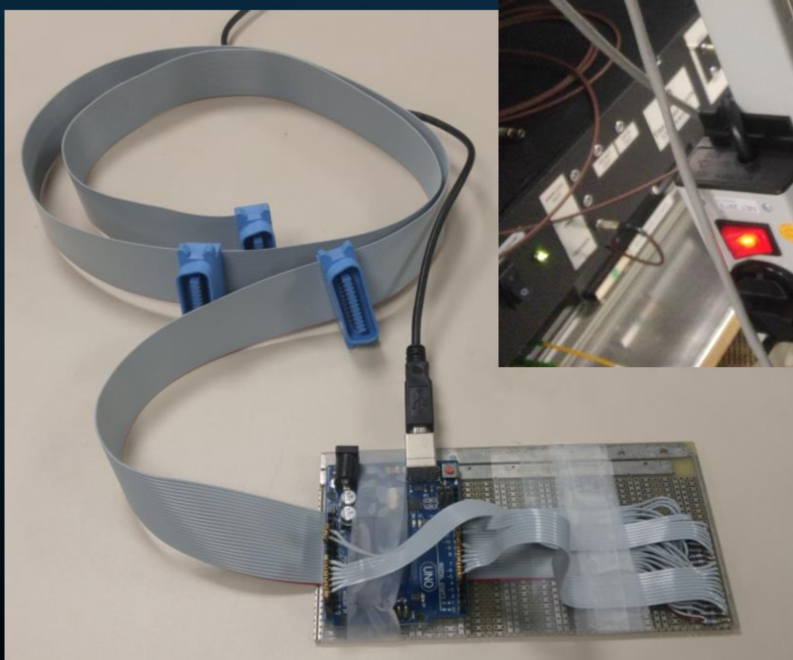


APPLICATIONS IN HEP ENVIRONMENTS





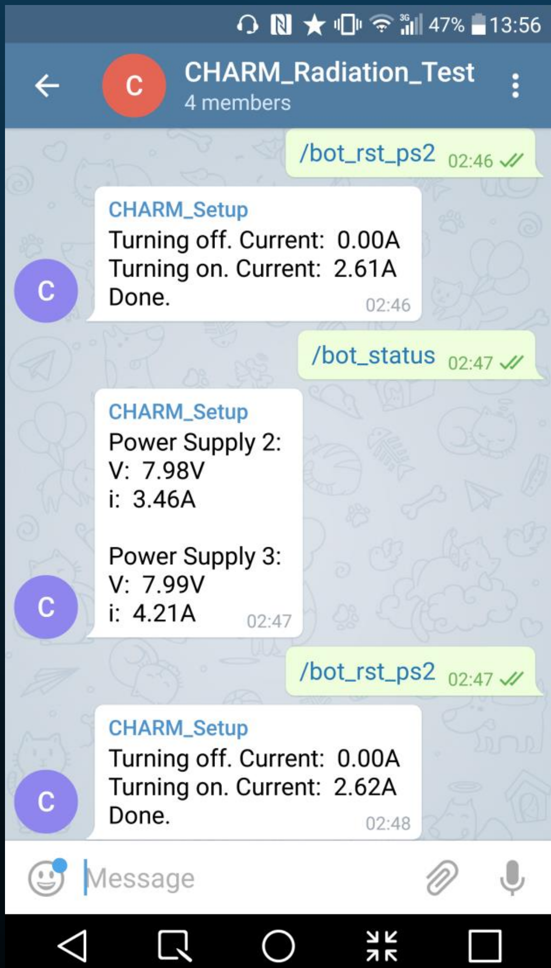
APPLICATIONS IN HEP ENVIRONMENTS



Arduino as a remote GPIB controller.

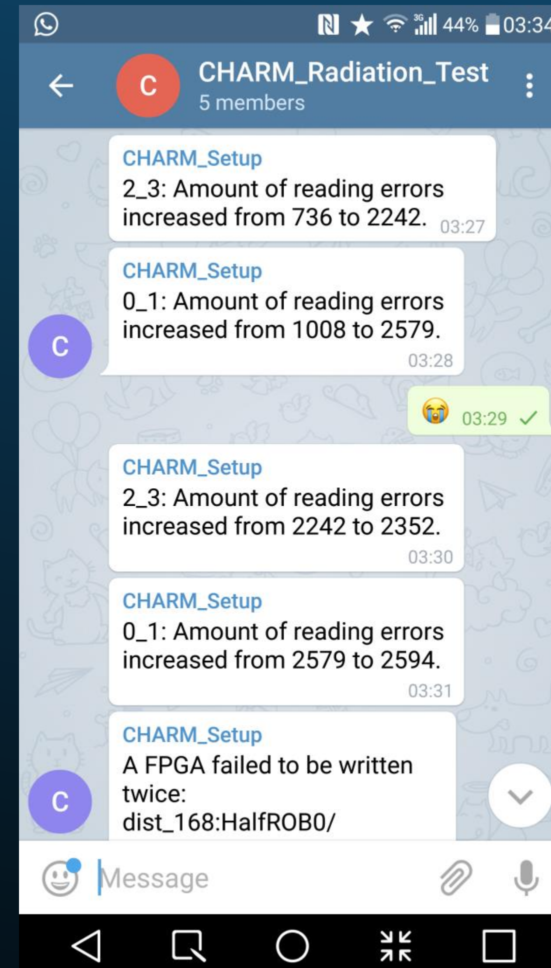
APPLICATIONS IN HEP ENVIRONMENTS

Telegram BOT for Testbeam monitoring.



What's your purpose?

-To report status and wake up Mauricio at 3am.



APPLICATIONS IN HEP ENVIRONMENTS

Is it possible to build a complete particle detector and data acquisition system using Arduino microcontroller and Arduino Language ?

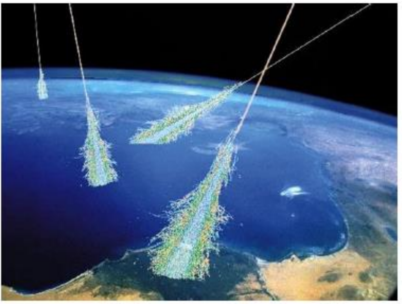
ArduSiPM a low cost particle detector

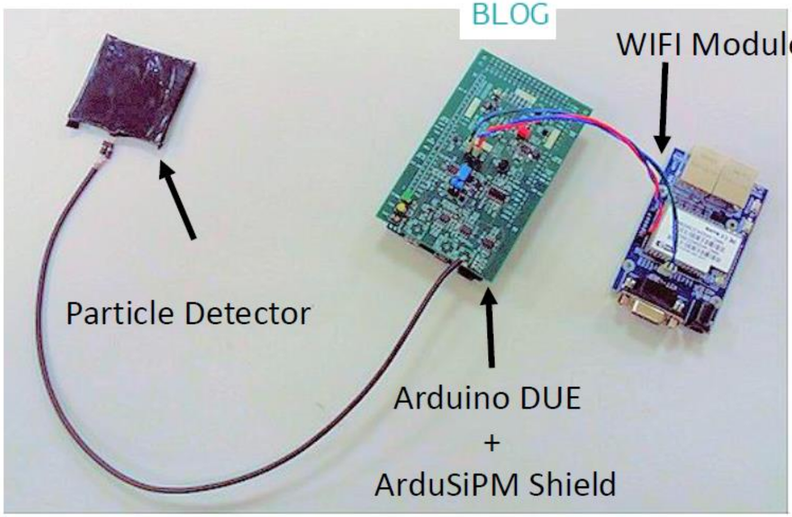
ARDUINO BLOG

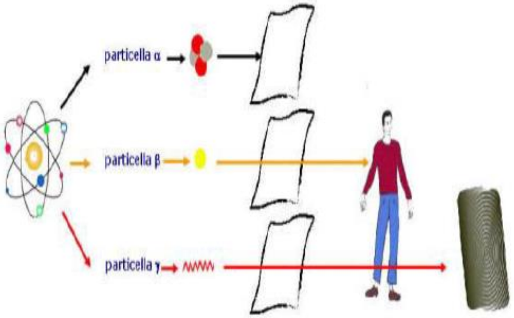
http://www.arduino.org/blog/ardusipm-solution

"The ambit of data acquisition for particle detection is a field apparently limited to top scientists from CERN in Geneva and Fermilab in Chicago. Cosmic ray and radiation detection can be a great exploration for teachers, students and science enthusiasts, and ArduSiPM was created to make it accessible."

Cosmic Ray detector







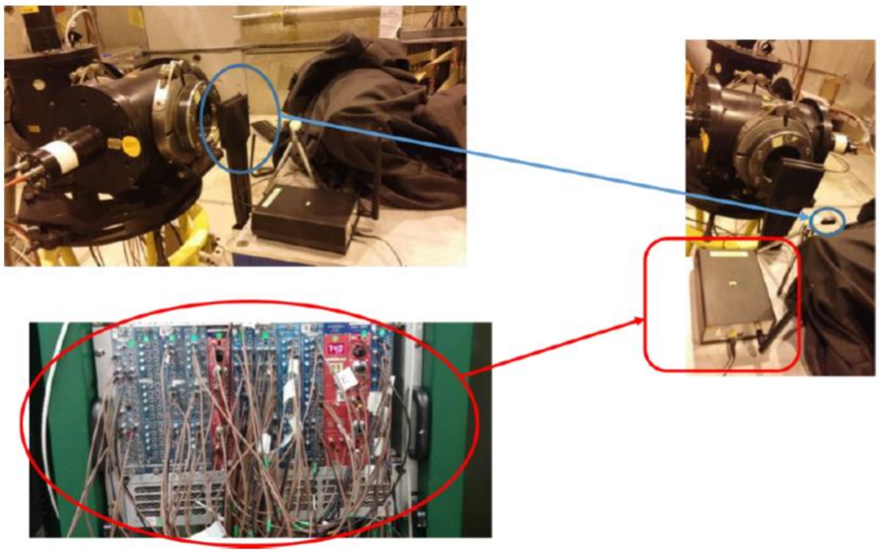
Dr .Valerio Bocci Nov 2°, 2016 INFN
INTERNATIONAL COSMIC DAY

APPLICATIONS IN HEP ENVIRONMENTS

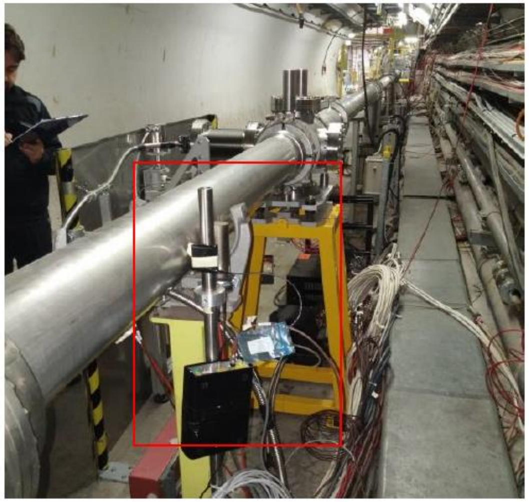
Application Example 2:

Use of ArduSiPM in the CERN UA9 and CRYSBEAM activity

(substitute old Scintillator and electronics for PM)



- As beam trigger @ extracted beam line H8 (CERN)



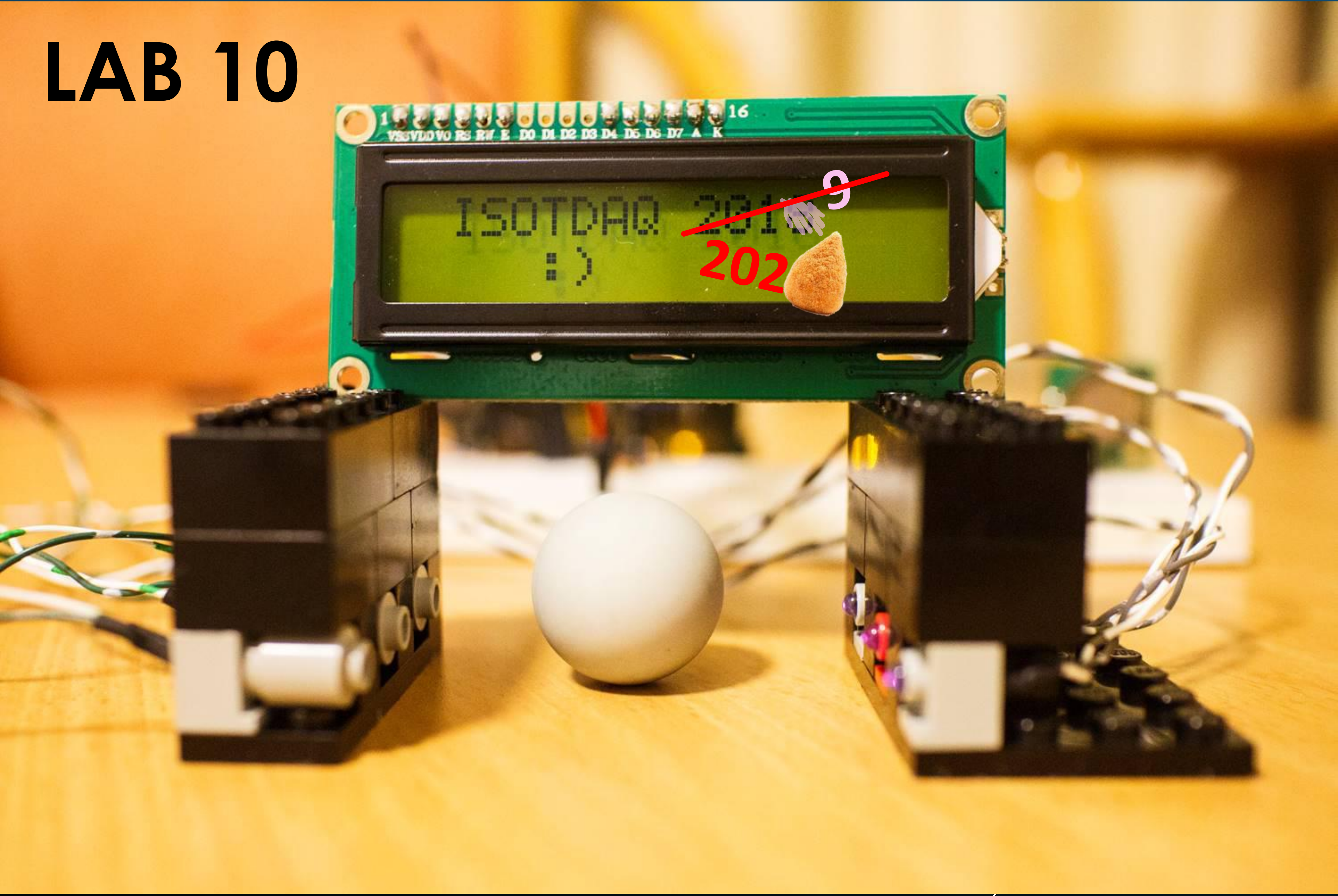
- As beam losses counter @ SPS

- *This work has been supported by the ERC Ideas Consolidator Grant*
- *No.615089 "CRYSBEAM".*

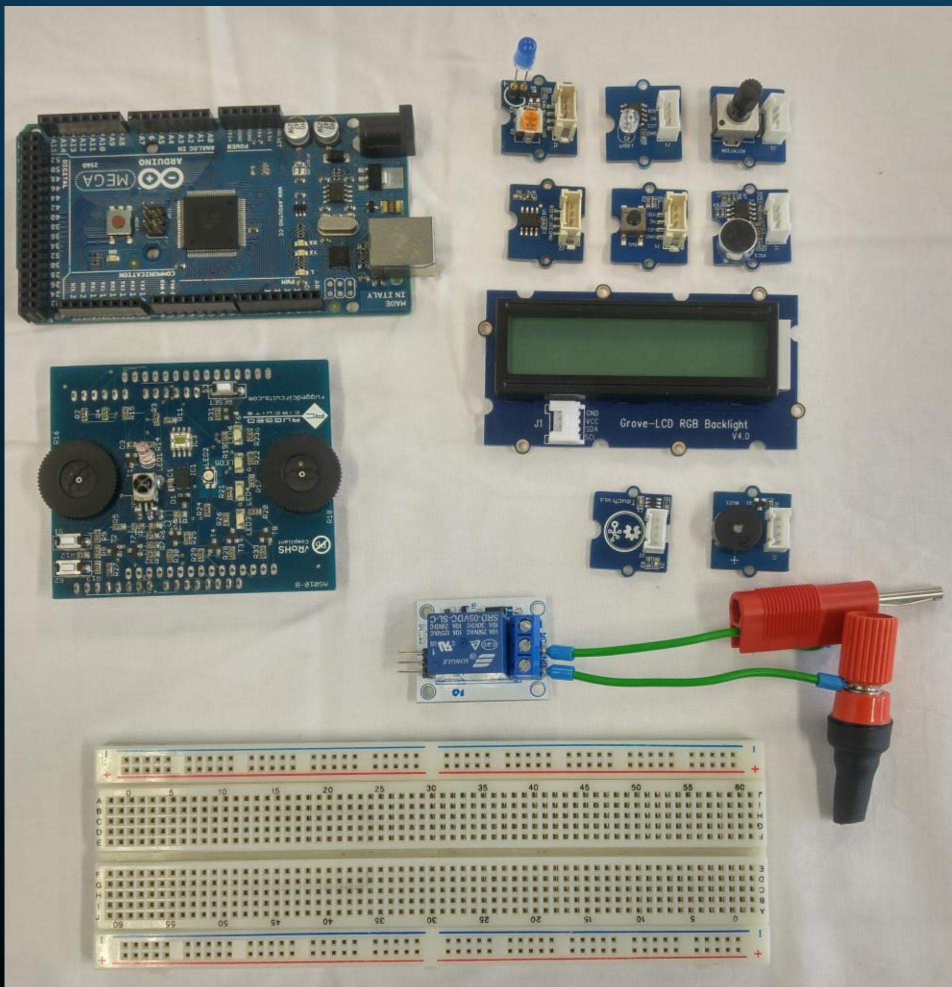
Dr .Valerio Bocci Nov 2°, 2016

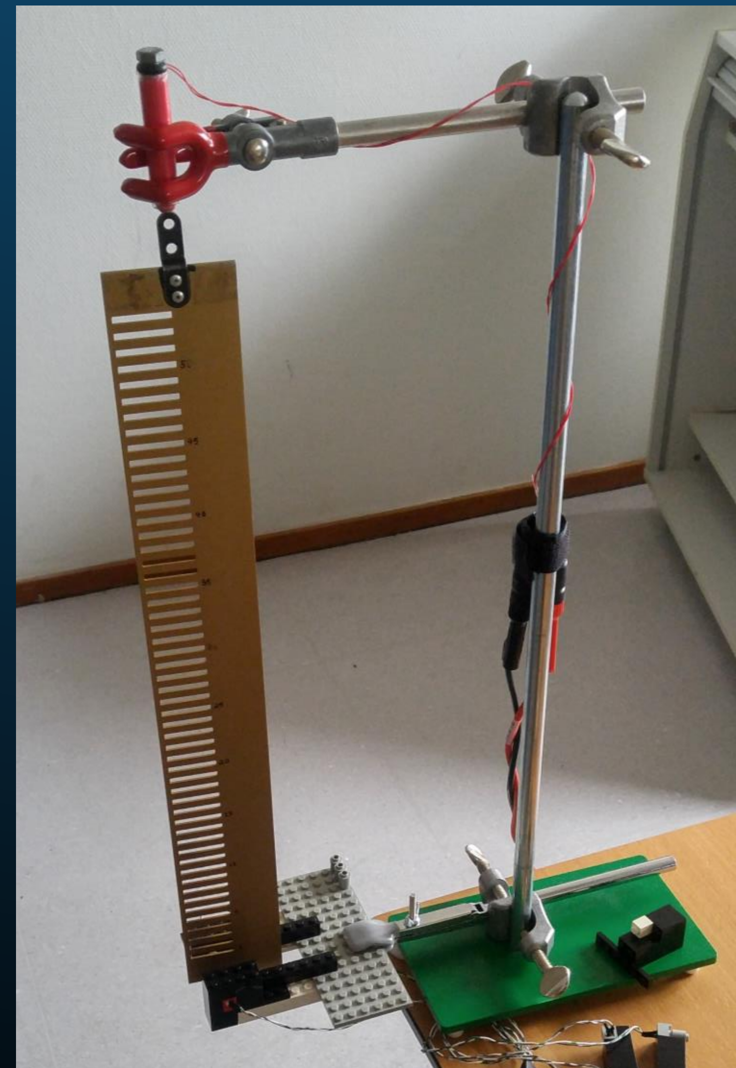
INTERNATIONAL COSMIC DAY

LAB 10



LAB 10 - MICROCONTROLLERS





THAT'S IT.

OBRIGADO!

Maurício Féo

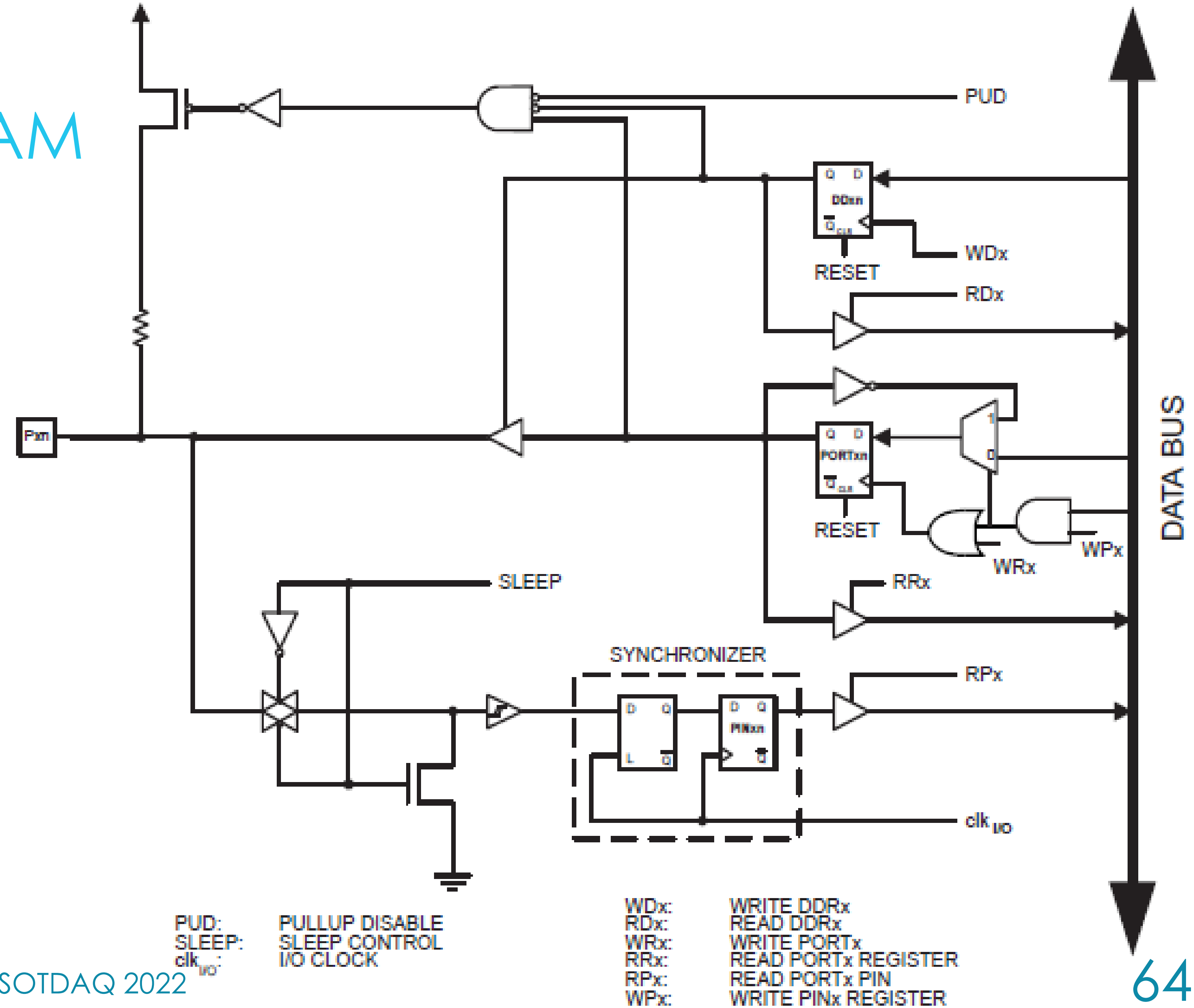
m.feo@cern.ch


16
V0 RS RW E D0 D1 D2 D3 D4 D5 D6 D7 A K
ISOTDAQ 2017
:)

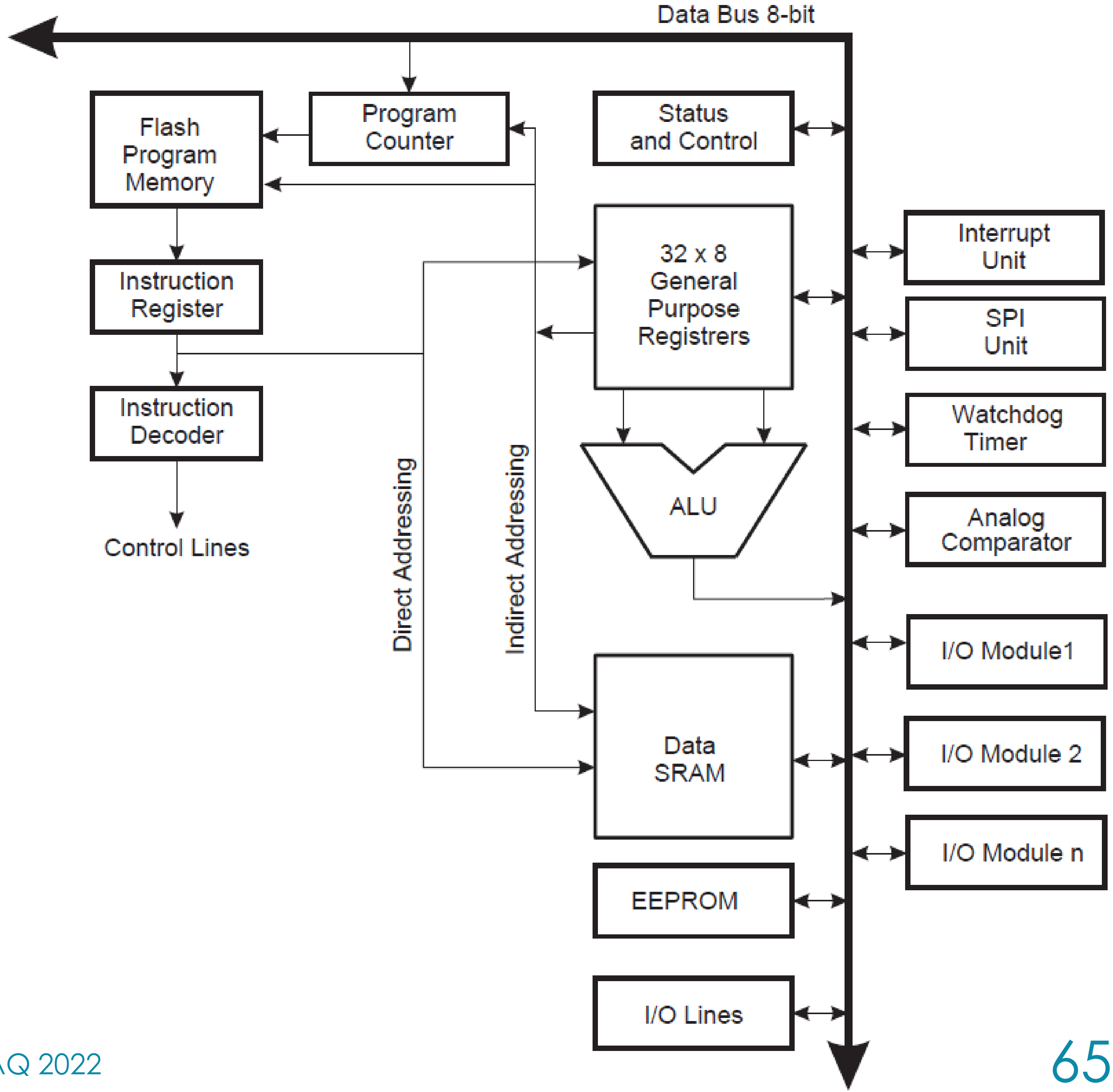

16
V0 RS RW E D0 D1 D2 D3 D4 D5 D6 D7 A K
ISOTDAQ 2018
:)

GPIO DIAGRAM

Figure 13-2. General Digital I/O[1]

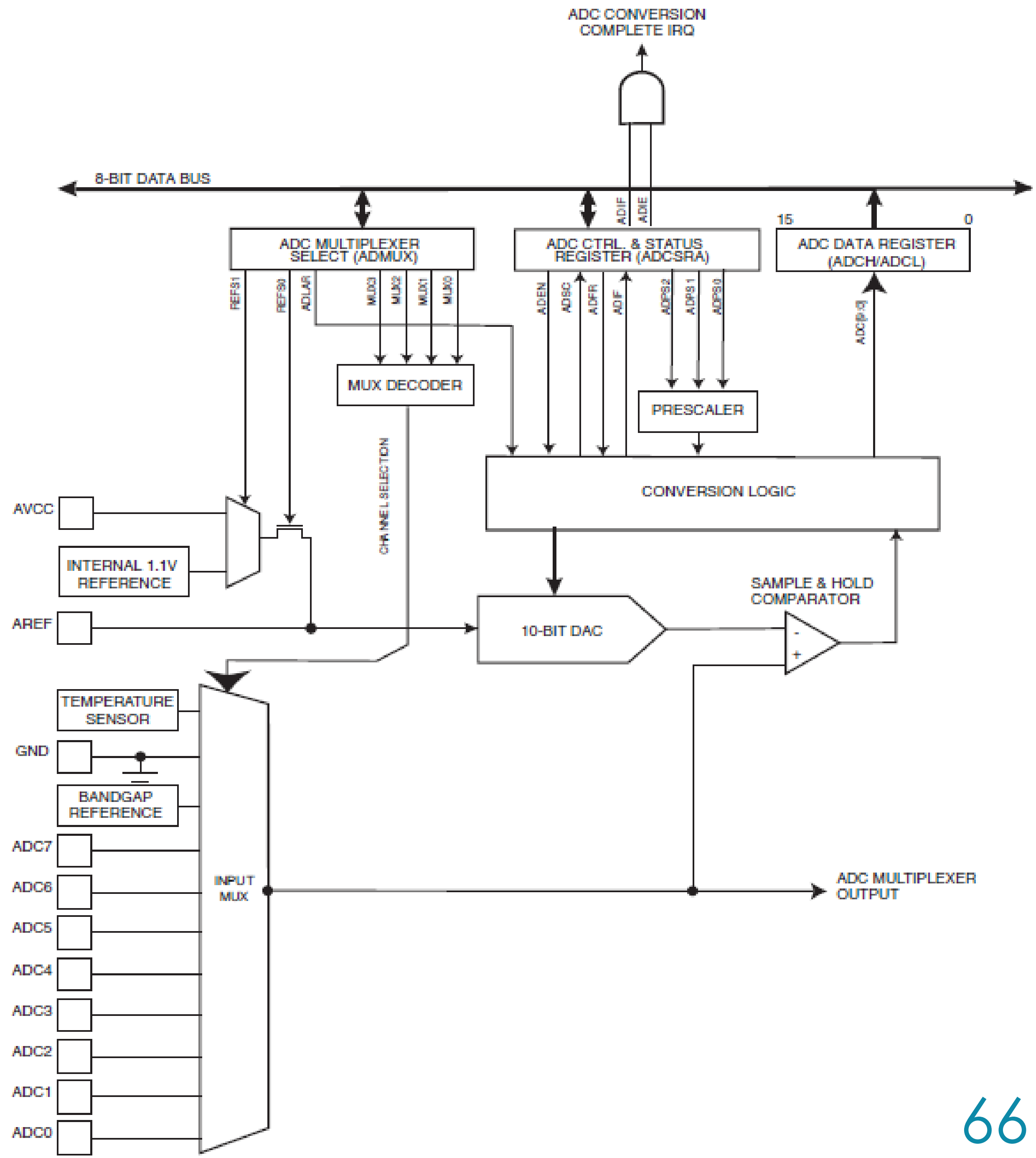


CPU DIAGRAM

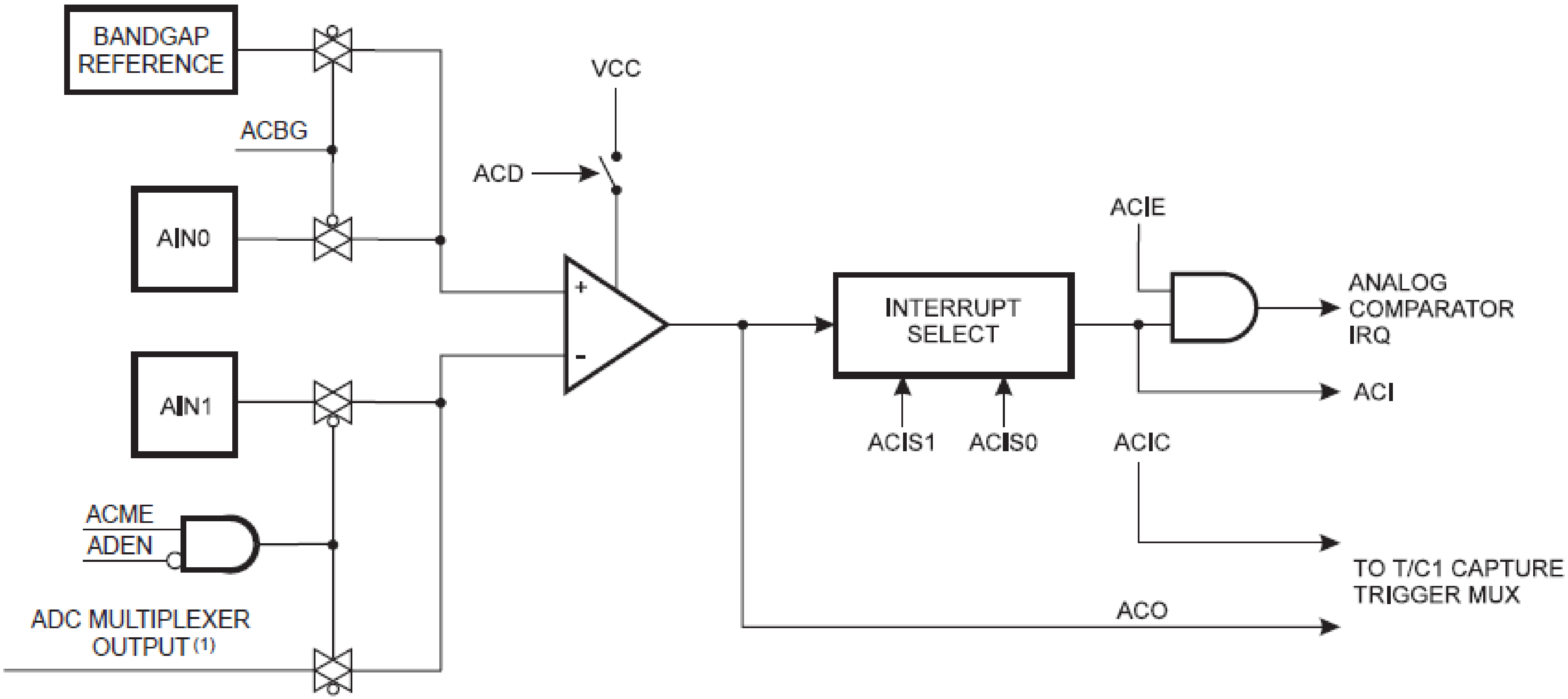

Data Bus 8-bit
Flash
Program
Memory
Program
Counter
Status
and Control
Instruction
Register
32 x 8
General
Purpose
Registrers
Interrupt
Unit
SPI
Unit
Instruction
Decoder
Watchdog
Timer
ALU
Analog
Comparator
Control Lines
Direct Addressing
Indirect Addressing
I/O Module1
Data
SRAM
I/O Module 2
I/O Module n
EEPROM
I/O Lines

ADC DIAGRAM

Figure 23-1. Analog to Digital Converter Block Schematic Operation,



ANALOG COMPARATOR



USART

Figure 19-1. USART Block Diagram[1]