

MICROCONTROLLERS

Maurício Féo Rivello

m.feo@cern.ch

Brazilian Center for Physics Research

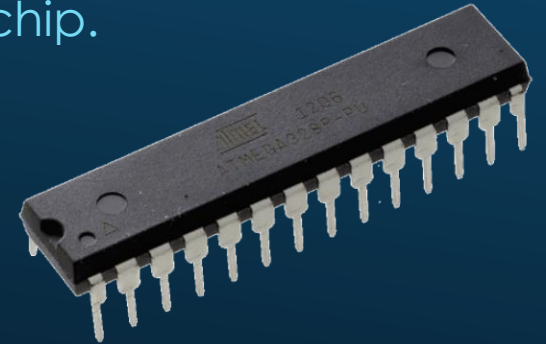
CBPF

OBJECTIVES

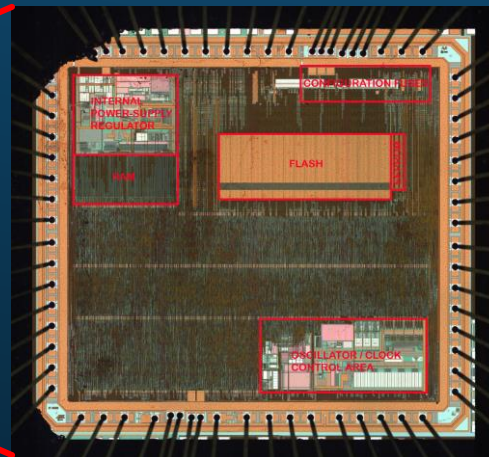
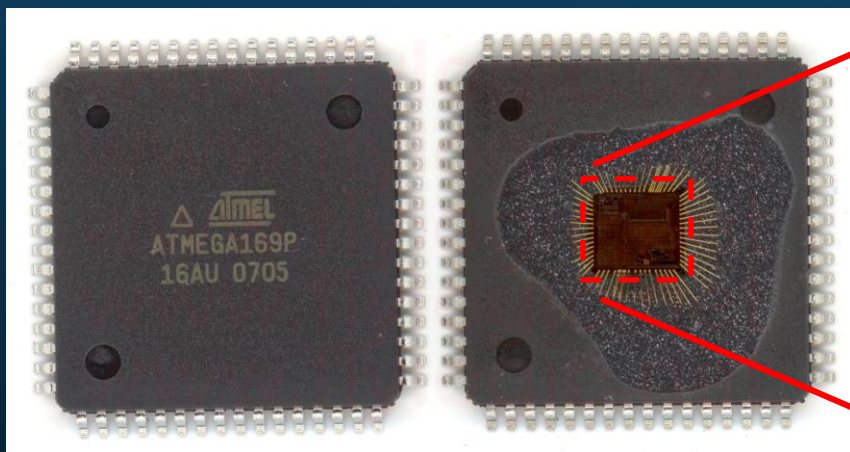
- ▶ Understand what are microcontrollers.
 - ▶ What are they?
 - ▶ What are they used for?
 - ▶ How do they work?
 - ▶ Are they suitable for my project?
 - ▶ How can I use them?
- ▶ Have an overview of the lab 10.

WHAT IS A MICROCONTROLLER?

- ▶ **Tiny computers integrated in a single chip**
 - ▶ CPU, Memories and Peripherals in the same chip.
- ▶ Suitable for embedded applications.
- ▶ Low cost (PIC10F200T: \$0.32)
- ▶ Low power consumption (ATtiny43U: 0.15uA in Sleep Mode)
- ▶ Reduced clock frequency (~ dozens of MHz)
- ▶ Stand-alone devices (Some require only power to work)

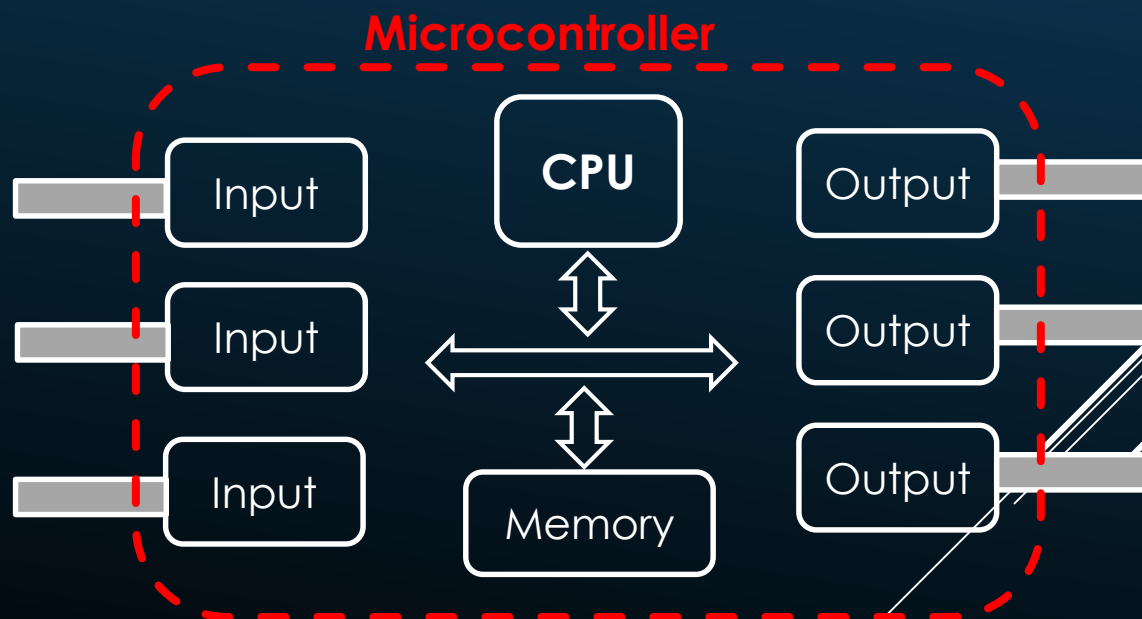


WHAT IS A MICROCONTROLLER?



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

Integrated in the same chip.



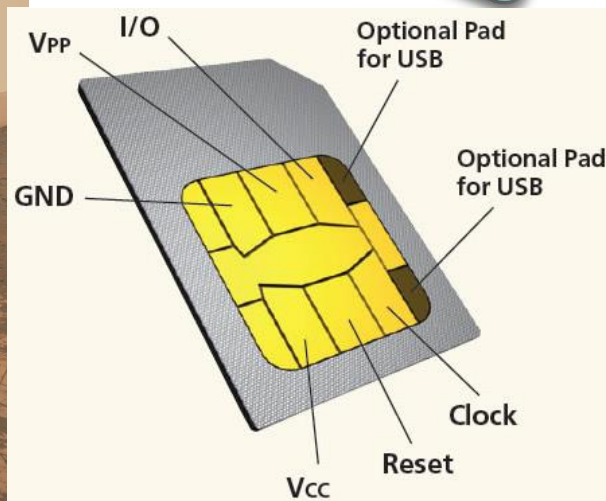
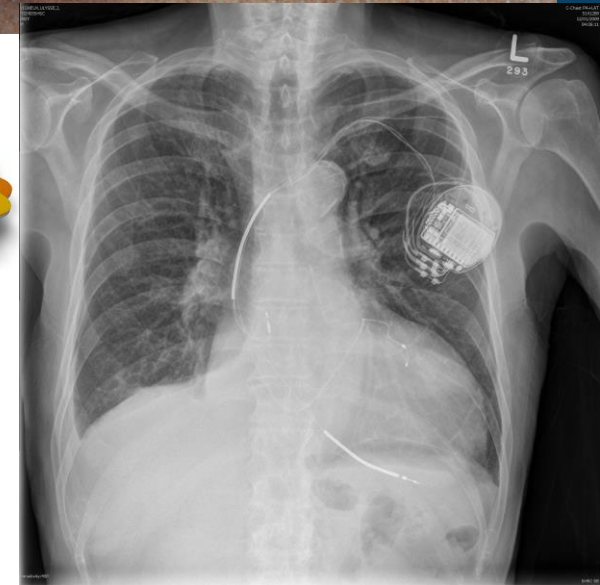
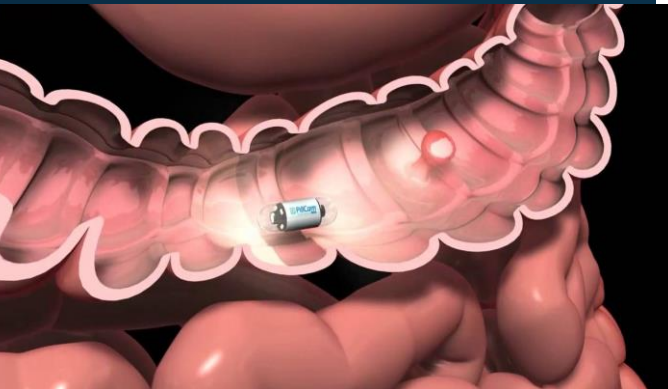
WHAT ARE THEY USED FOR?

- ▶ Monitoring
 - ▶ Data Acquisition
 - ▶ Control
-
- ▶ Applications where high processing performance is not required and general purpose microprocessors are considered inadequate due to high power consumption, high pin count packages and the need of external memories and peripherals.

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.

WHERE ARE THEY USED?



Windows

An error has occurred. To continue:

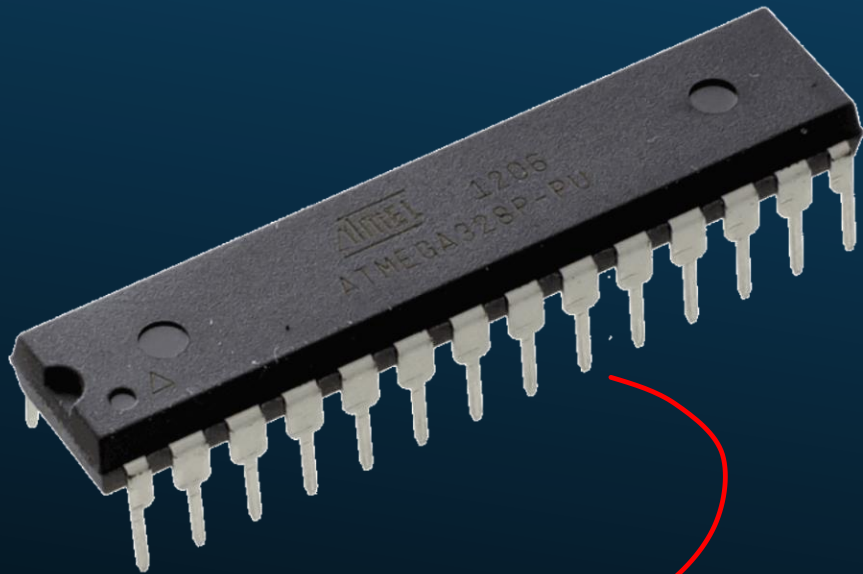
Press Enter to return to Windows, or

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

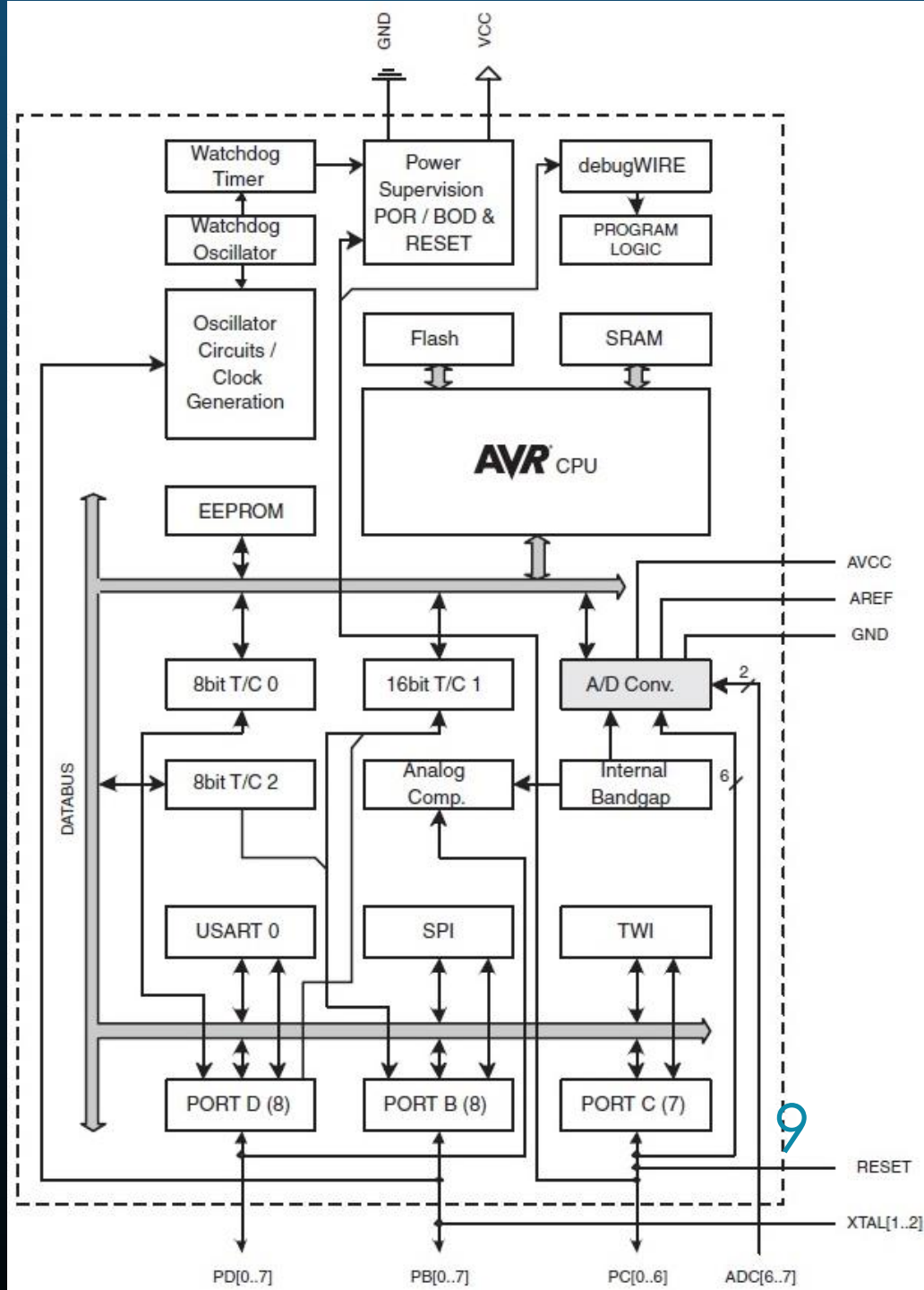
Press any key to continue _

AVR ARCHITECTURE (ATMEGA328P)



The one used on the lab.

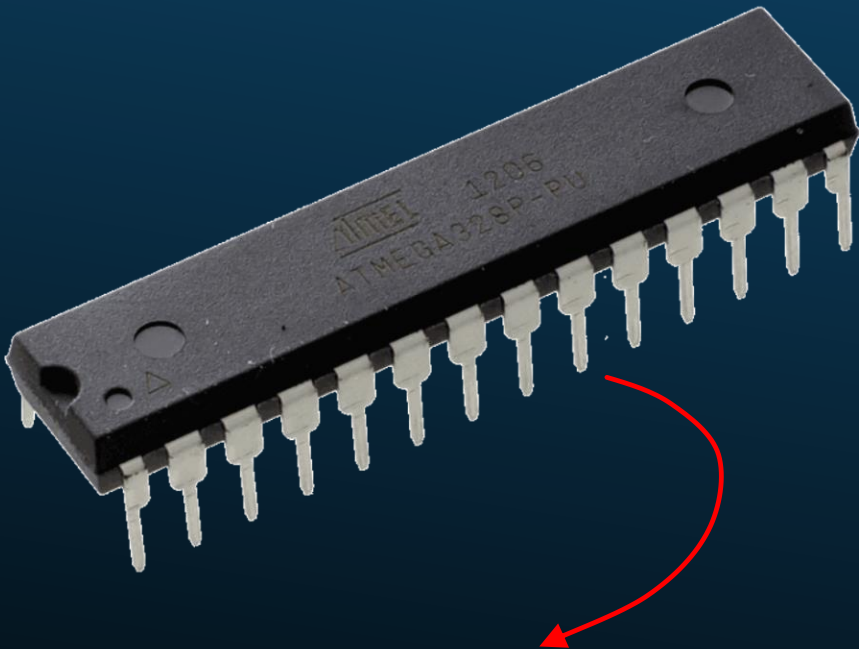
m.feo@cern.ch – ISOTDAQ 2016



AVR

ARCHITECTURE

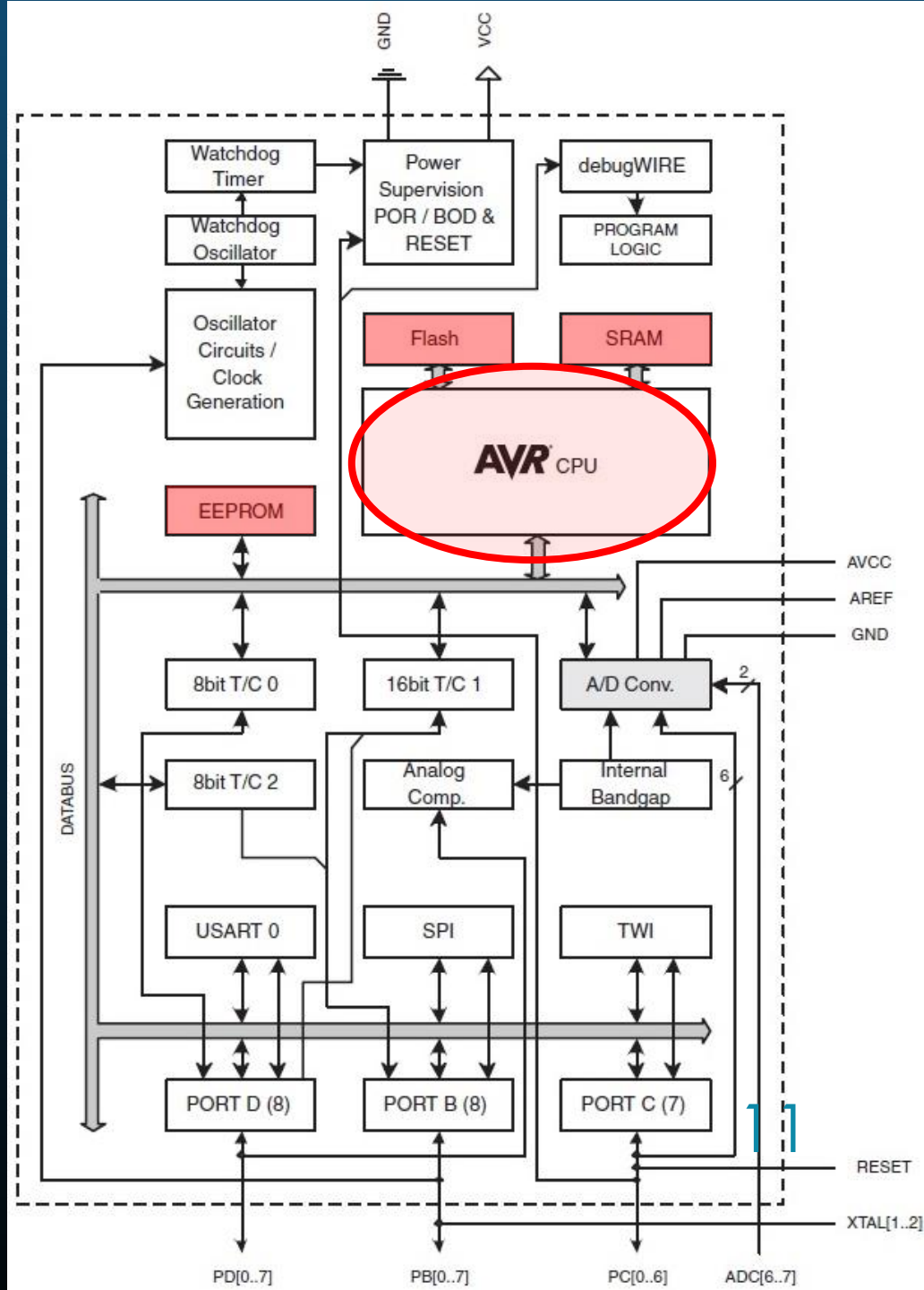
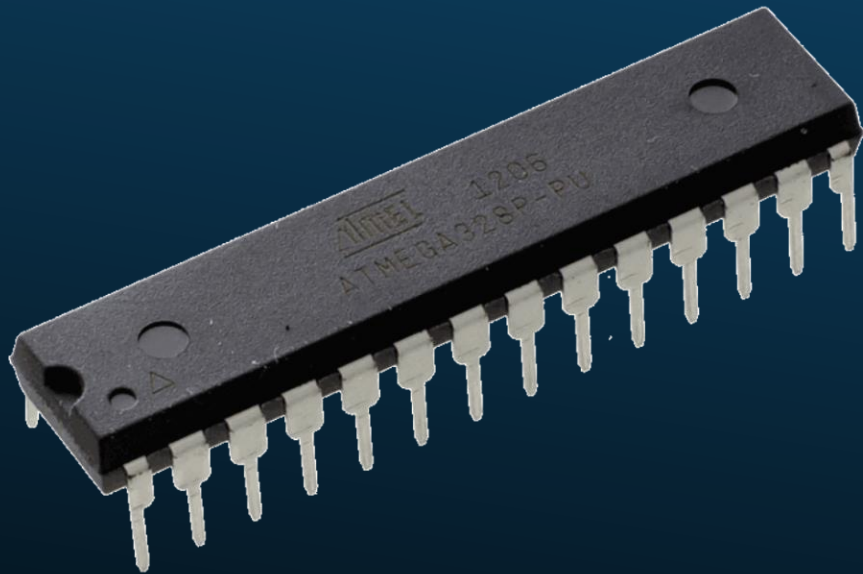
(ATMEGA328P)



The one used on the lab.

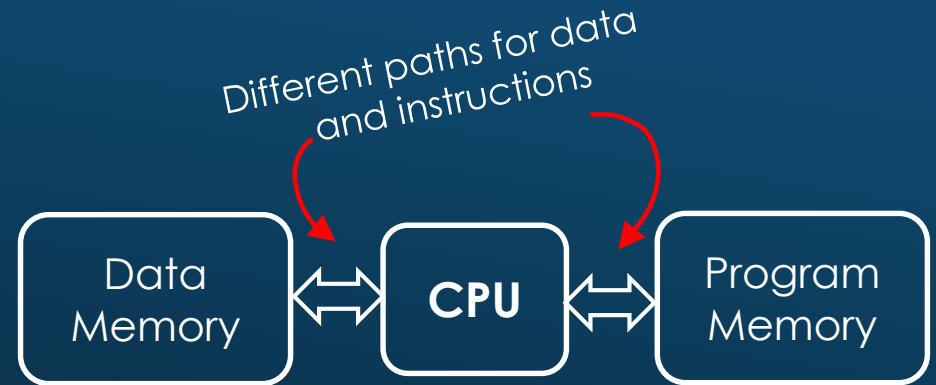
- ▶ 8 bit microcontroller
- ▶ 32kB Flash program memory
- ▶ 2kB RAM
- ▶ 2kB EEPROM
- ▶ Max 20MHz
- ▶ 6 x PWM
- ▶ 6 x ADC channels (10bits)
- ▶ 23 I/O pins
- ▶ 3 timers (2x8 bits 1x16 bits)
- ▶ 1x USART
- ▶ 1x SPI
- ▶ 1x TWI (I²C)
- ▶ 0.6mA/MHz

AVR ARCHITECTURE (ATMEGA328P)



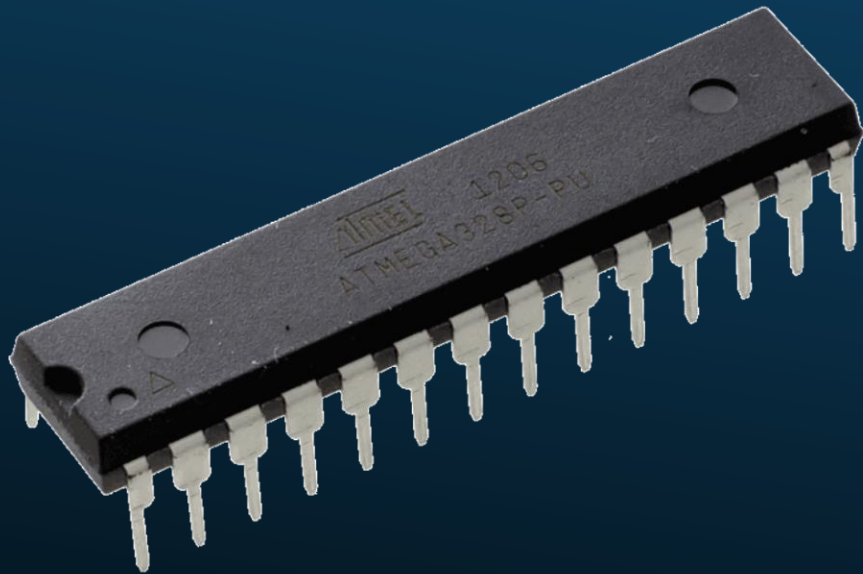
AVR CPU

- ▶ Harvard Architecture

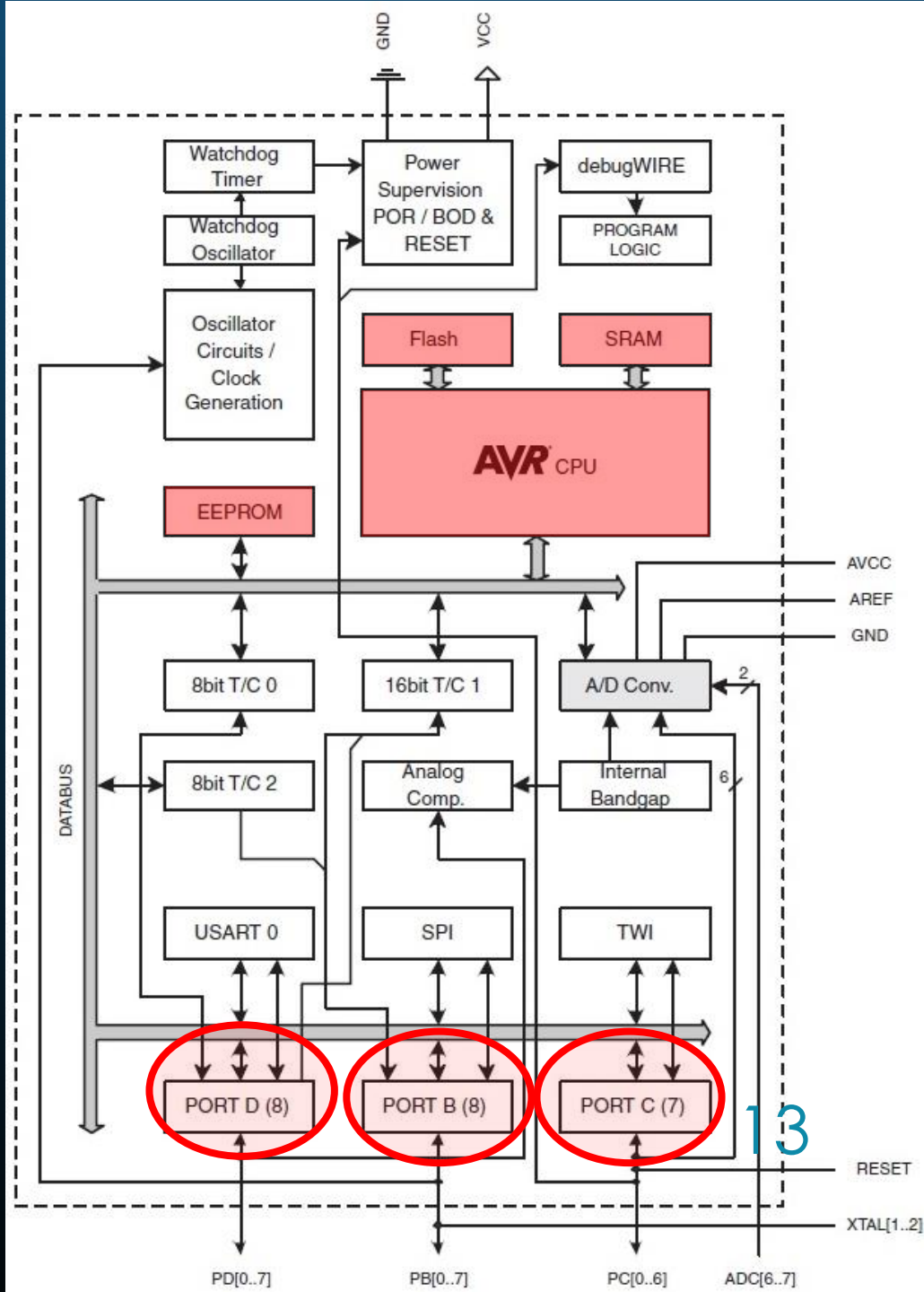


- ▶ 8 bits architecture (with 16bits for instructions)
- ▶ Reduced Instruction Set Computing (RISC) (~130 instructions)
- ▶ Up to 20 MIPS at 20 MHz (1 instruction / clock cycle)

AVR ARCHITECTURE (ATMEGA328P)

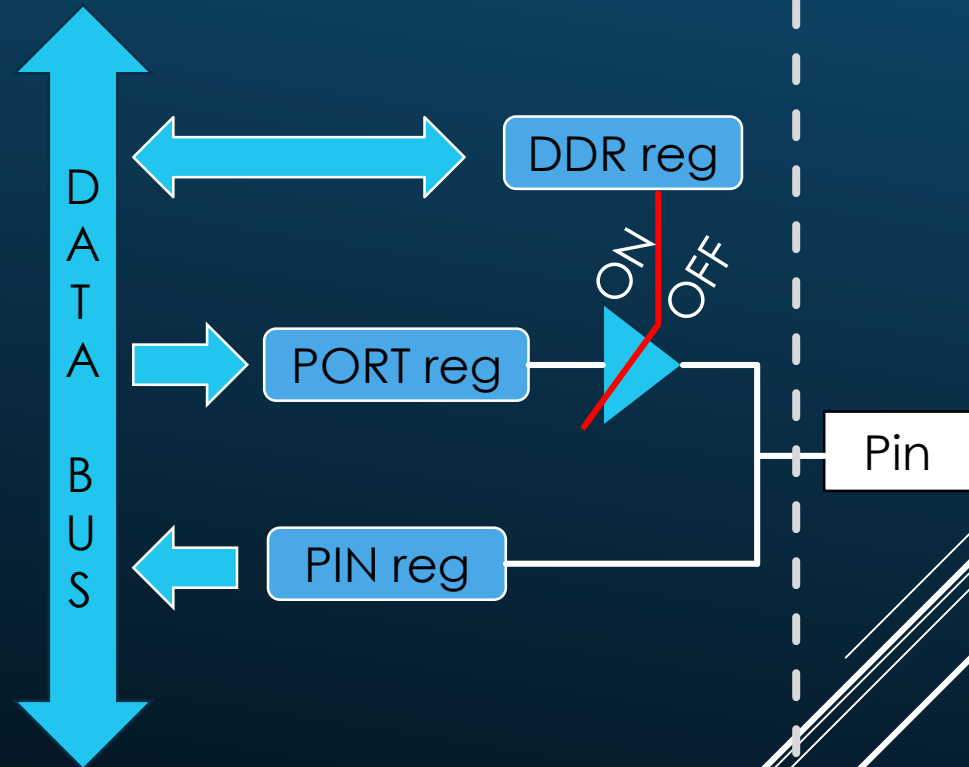


m.feo@cern.ch – ISOTDAQ 2016



GENERAL PURPOSE INPUT/OUTPUT (GPIO)

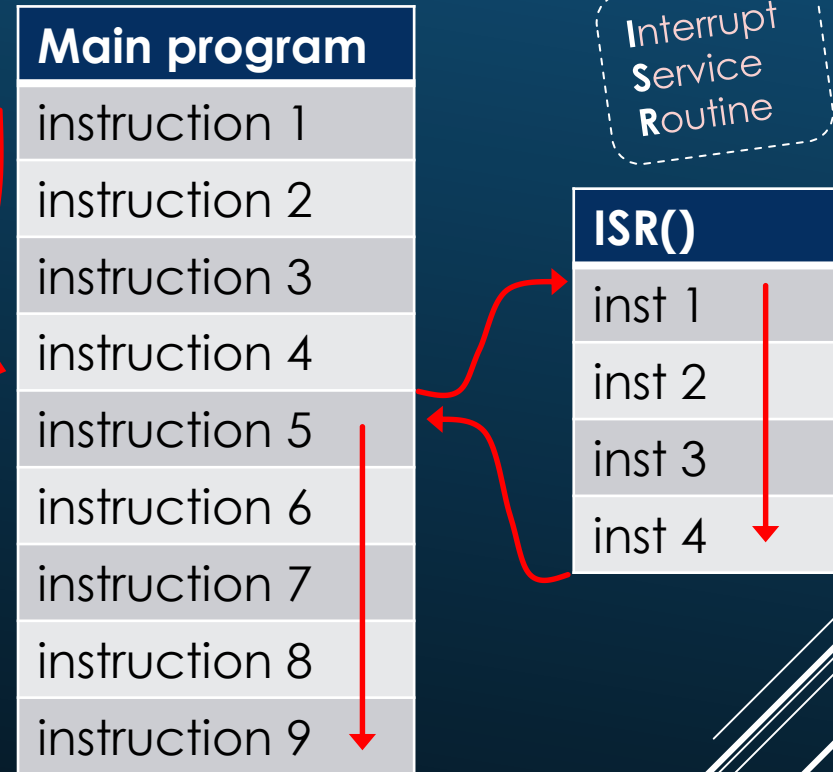
- ▶ Pins programmable as Input and Output
- ▶ Read / Write digital signals
- ▶ '0' = 0V (Gnd), '1' = 5V (Vcc)
- ▶ Controlled by 3 registers:
 - ▶ DDR (Data Direction Register)
 - ▶ PORT (Where you write when it's output)
 - ▶ PIN (Where you read when it's input)



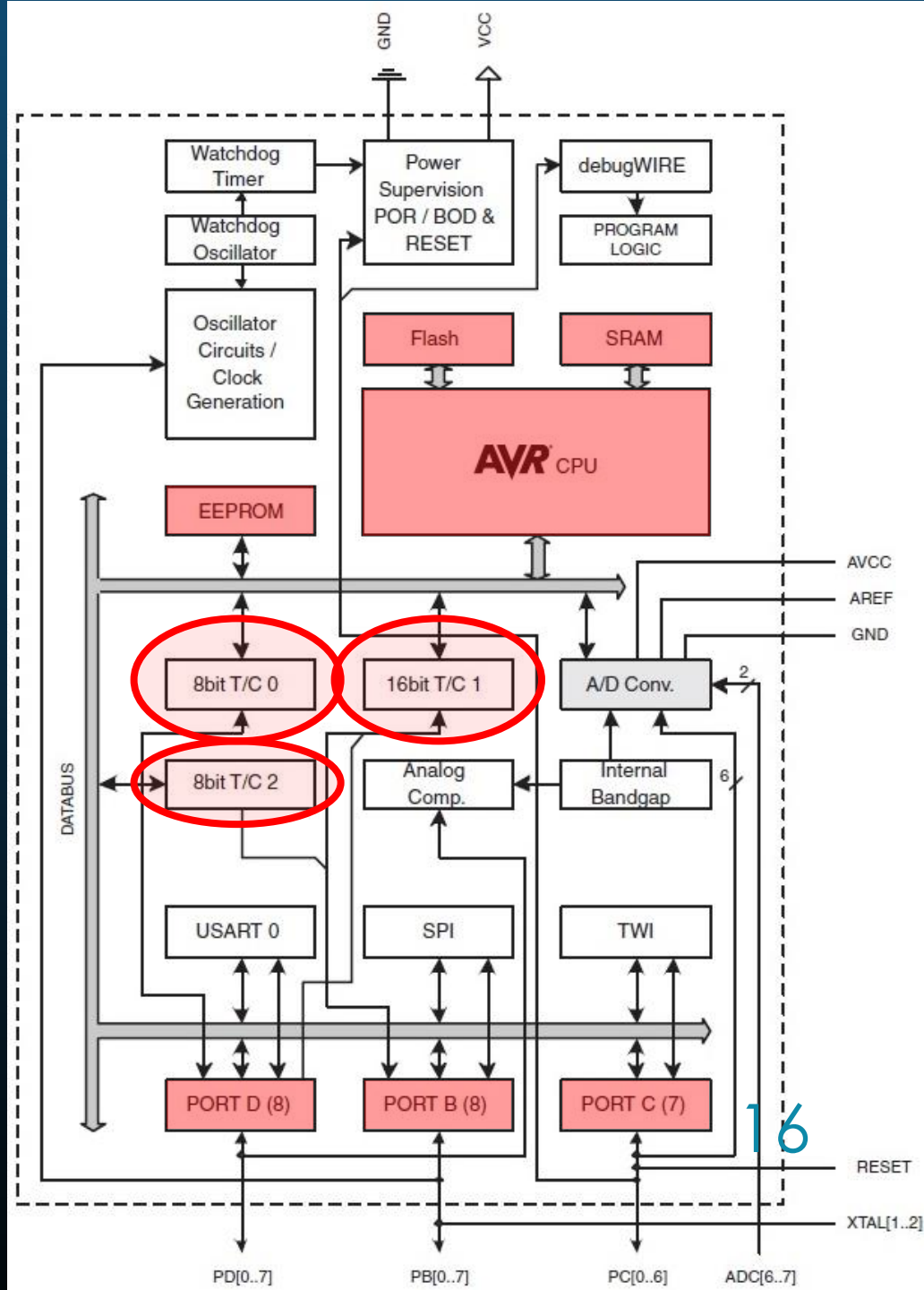
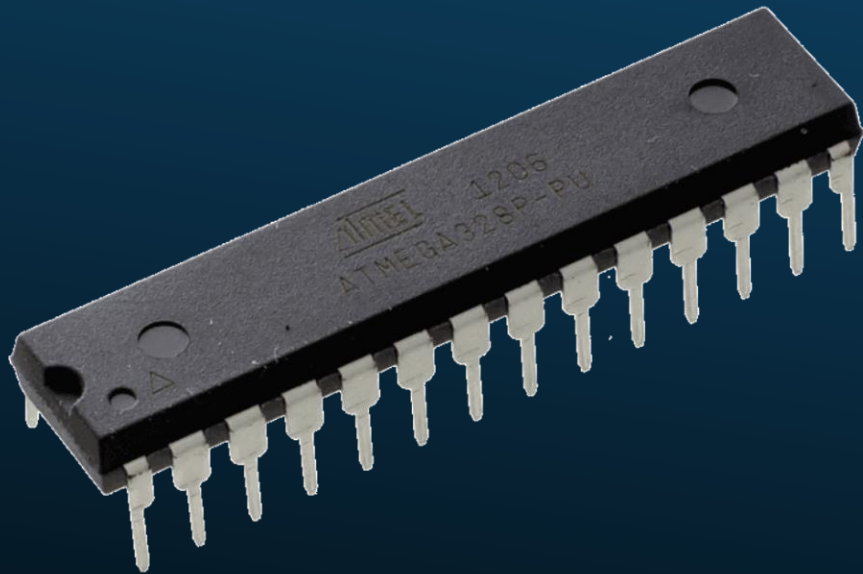
INTERRUPT

- ▶ 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.

Interrupt 



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

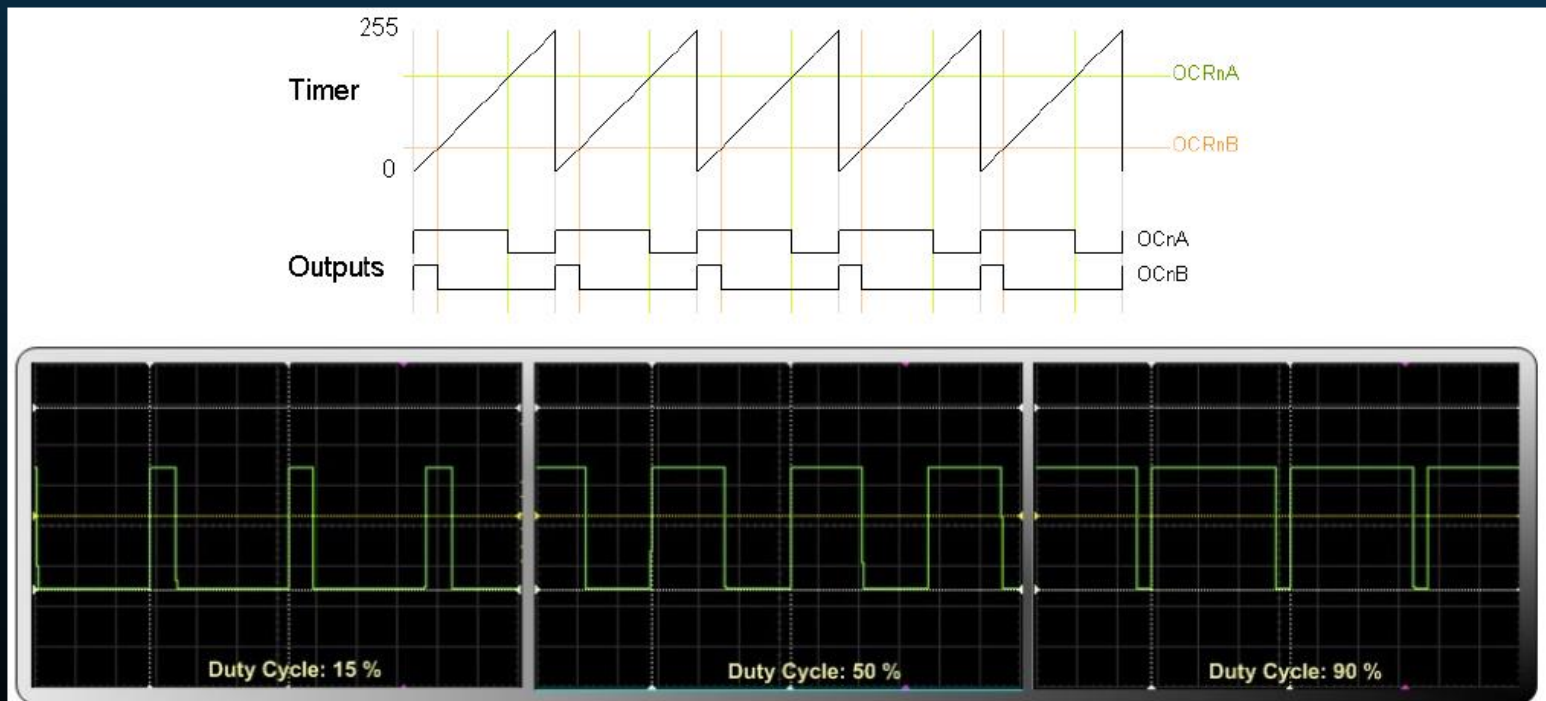
Output
Compare
Register



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

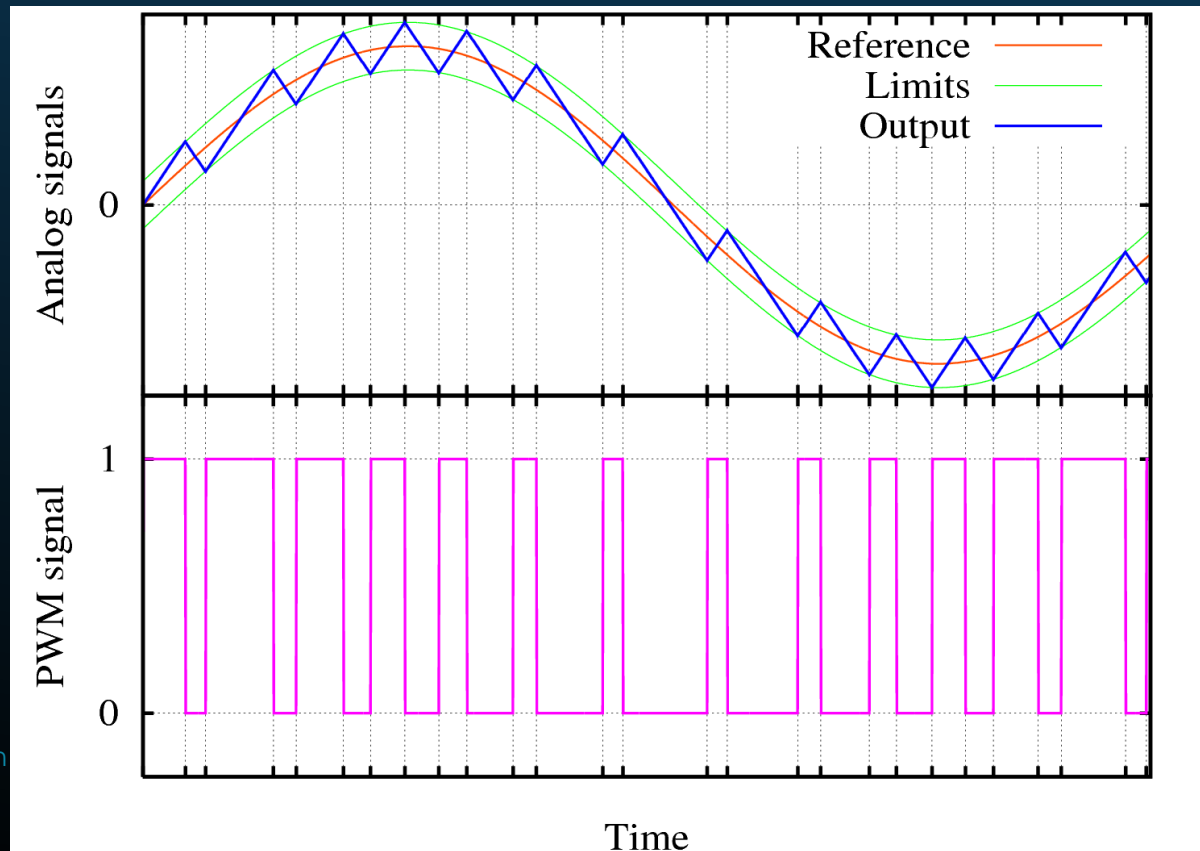
PULSE WIDTH MODULATION (PWM)

- ▶ 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)

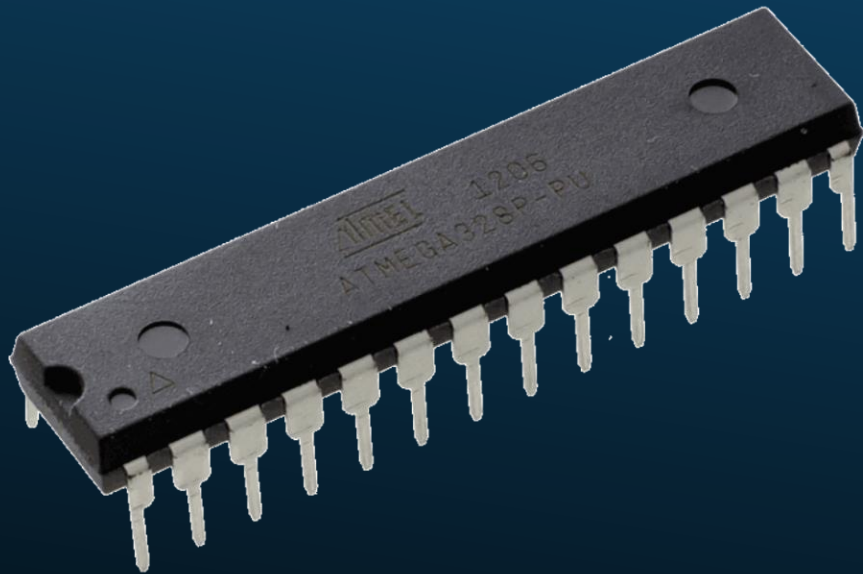


PULSE WIDTH MODULATION (PWM)

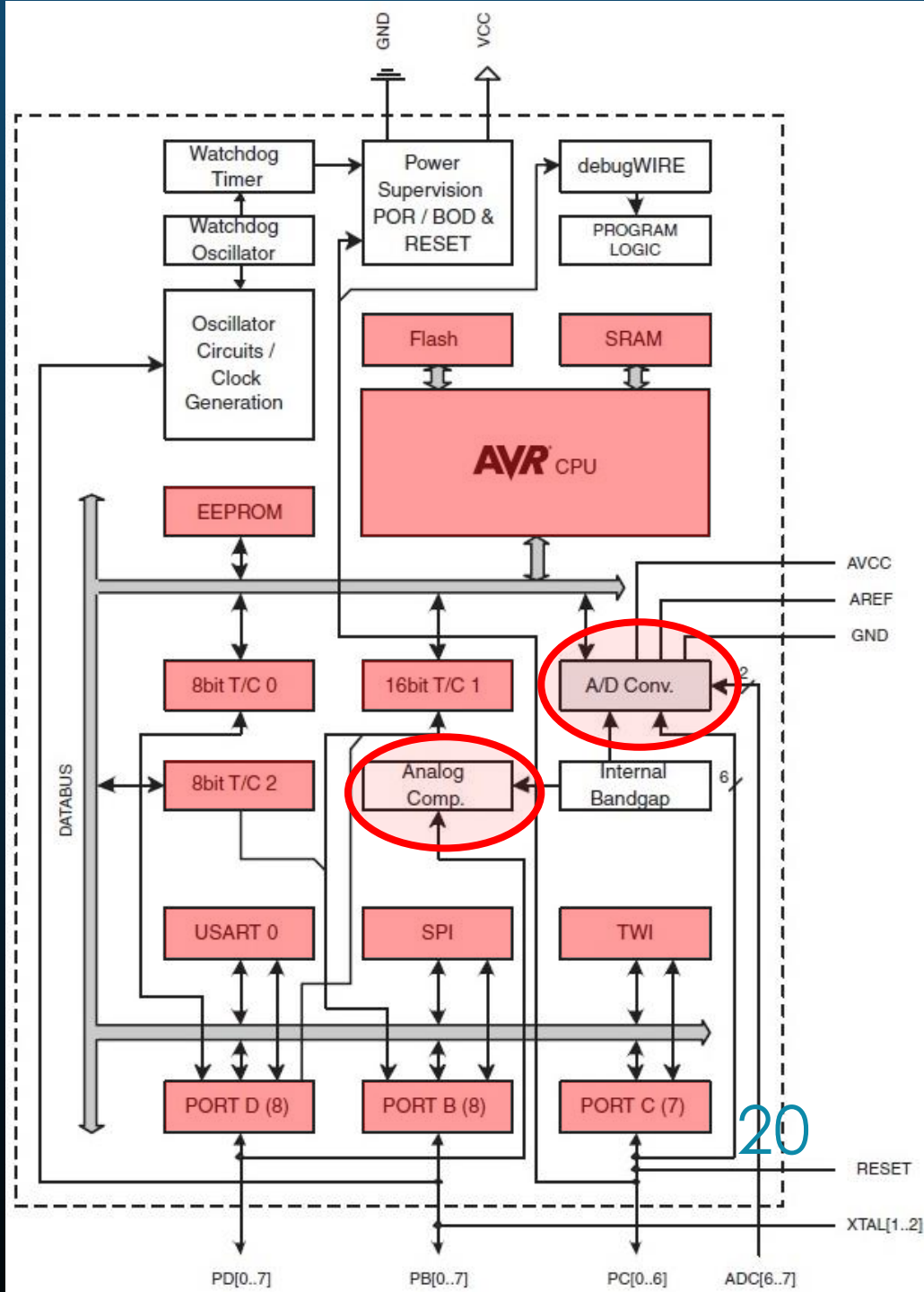
- ▶ You can create an output signal which value depends on the status of the timer.
- ▶ Outputs a train of periodic digital pulses with controlled width.
 - ▶ (Can be used to "mimic" an analog signal)



AVR ARCHITECTURE (ATMEGA328P)

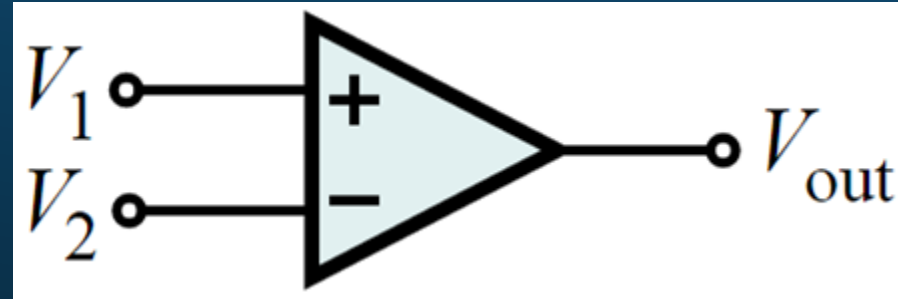


m.feo@cern.ch – ISOTDAQ 2016



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)

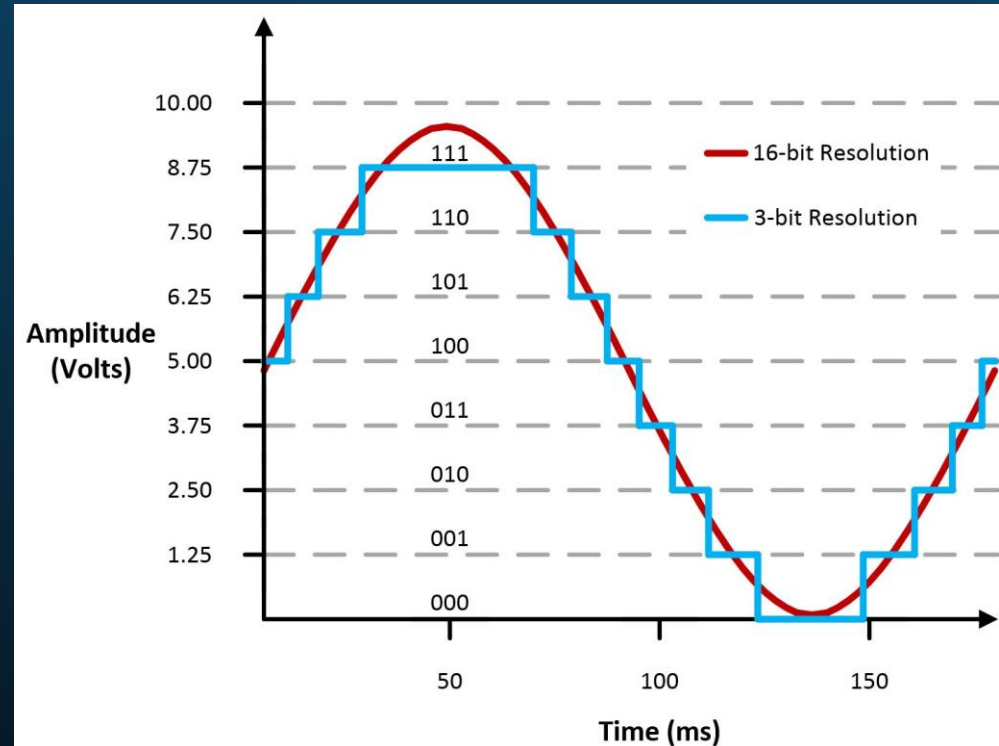


$$\begin{array}{l} V_1 > V_2 \rightarrow V_{\text{out}} = 1 \\ V_1 < V_2 \rightarrow V_{\text{out}} = 0 \end{array}$$

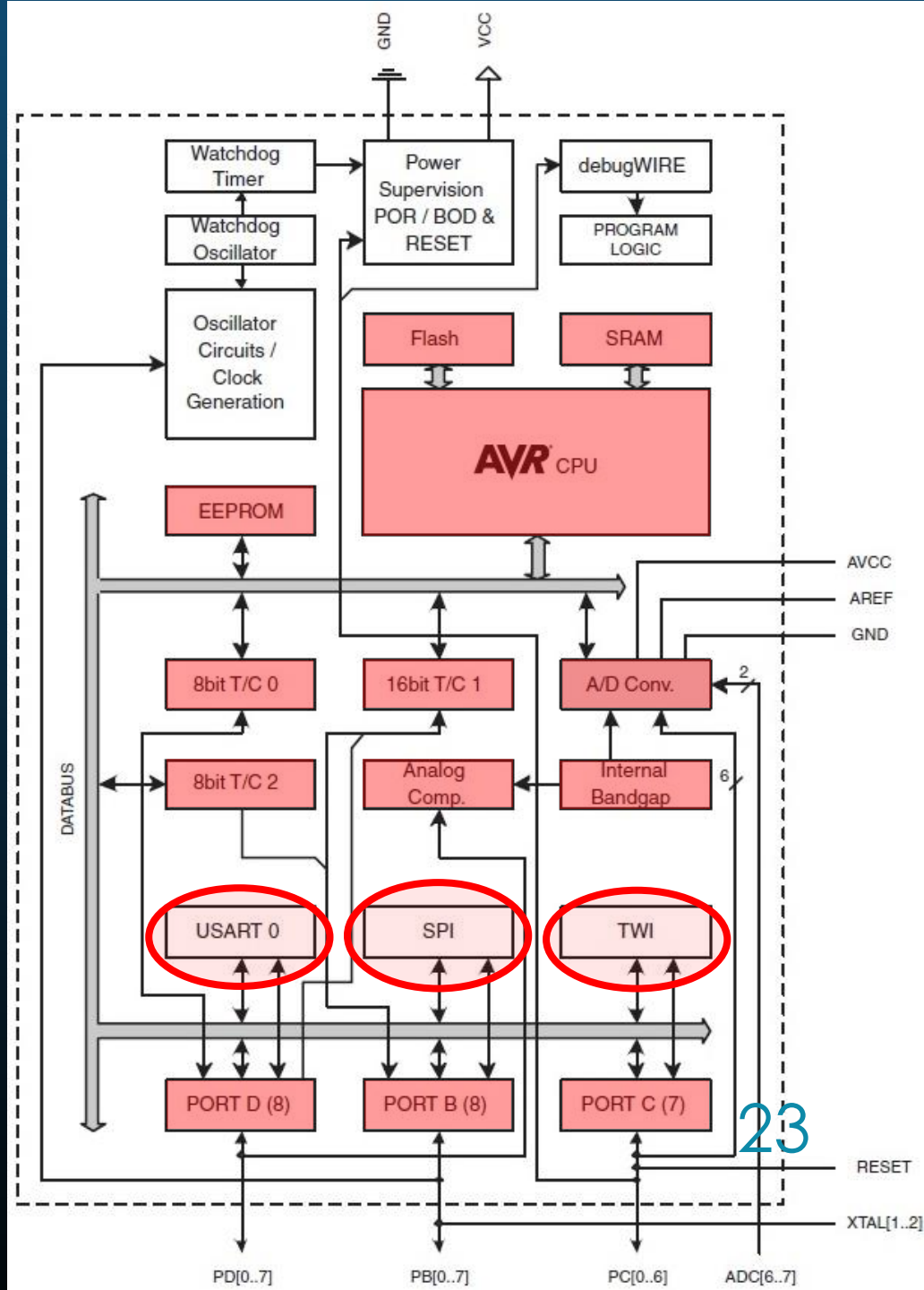
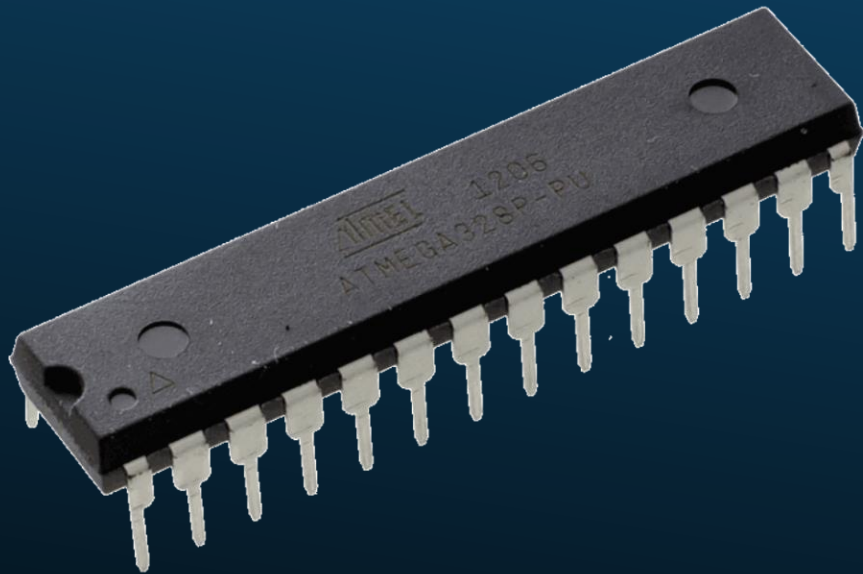
* **ACO** = Analog Comparator Output
* **ACSR** = Analog Comp. Control and Status Register

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 μ s Conversion time
- ▶ Interrupt on complete
- ▶ 6 multiplexed channels on DIP package
 - ▶ (internal Temp sensor on ch8)



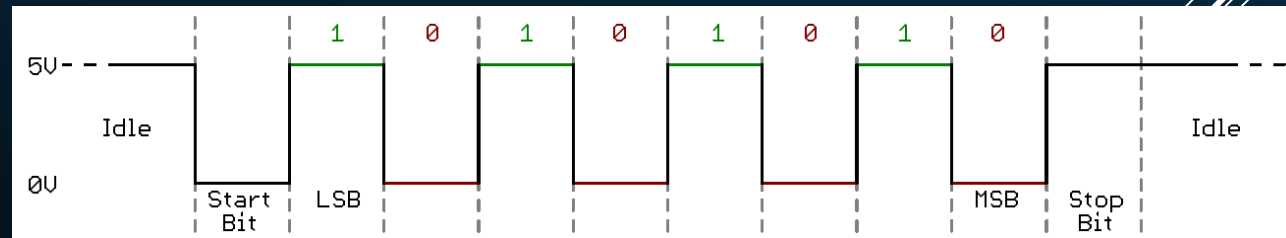
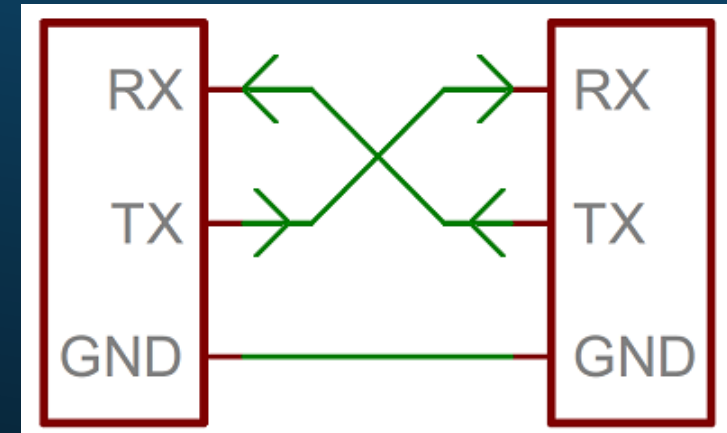
AVR ARCHITECTURE (ATMEGA328P)



SERIAL INTERFACES: USART

UNIVERSAL SYNCHRONOUS-ASYNCHRONOUS RECEIVER TRANSMITTER

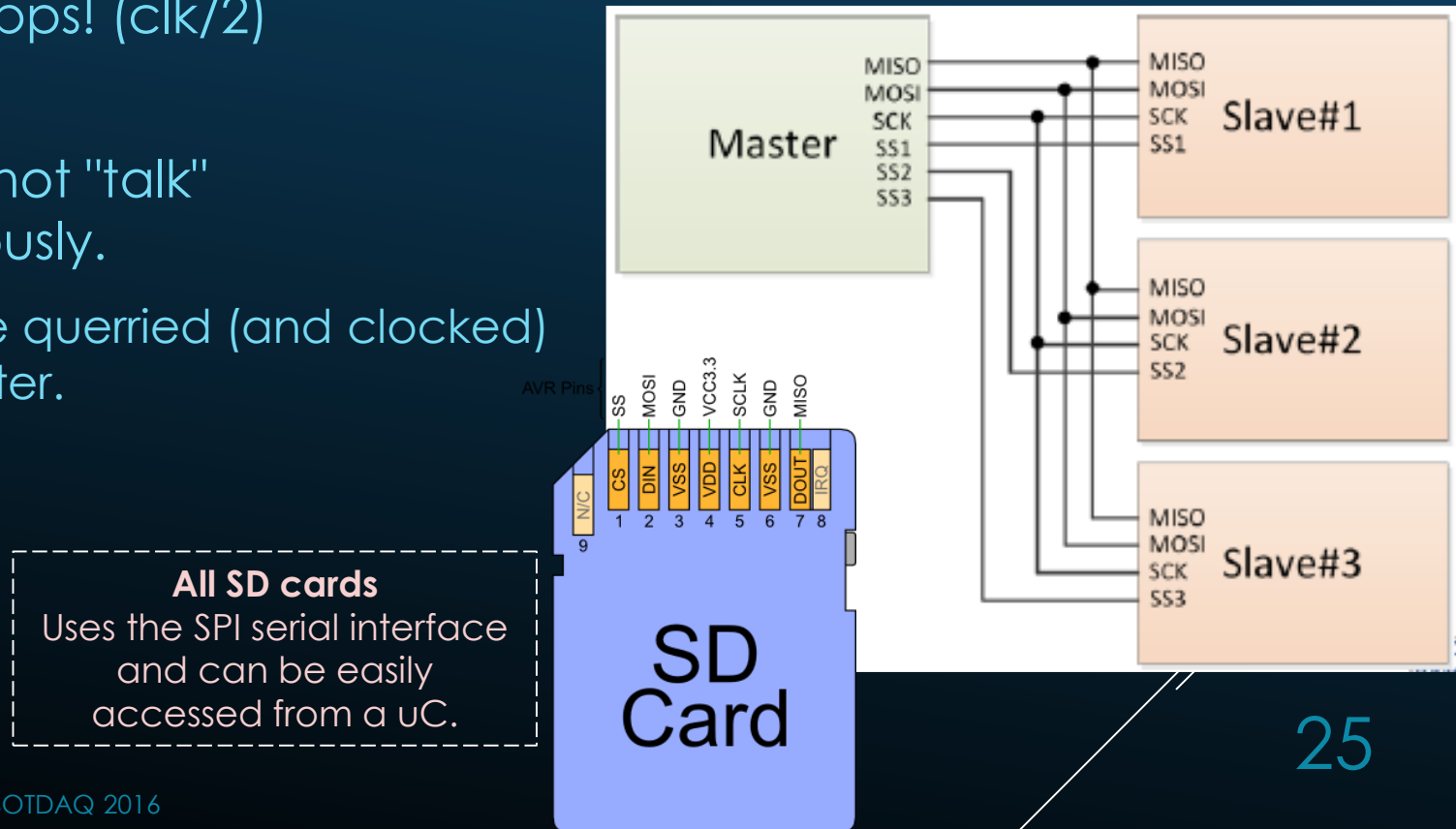
- ▶ A simple protocol
- ▶ Widely used to communicate with PCs due to compability with RS232 protocol. (RS232 is not used anymore in most PCs but it's still very easy to find USB-Serial converters)
- ▶ Up to 250kbps
- ▶ May trigger interrupts:
 - ▶ Tx complete
 - ▶ Rx complete
 - ▶ Data reg empty



SERIAL INTERFACES: SPI

SERIAL PERIPHERAL INTERFACE

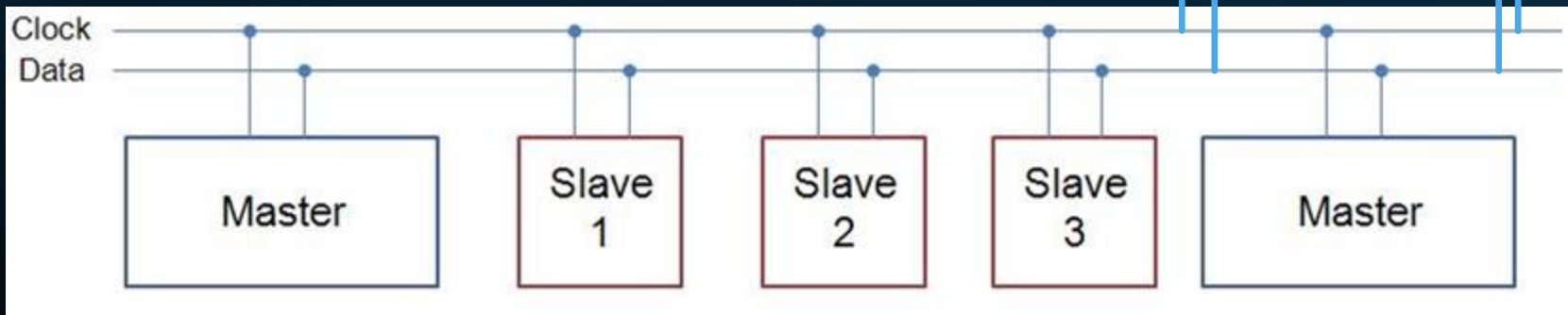
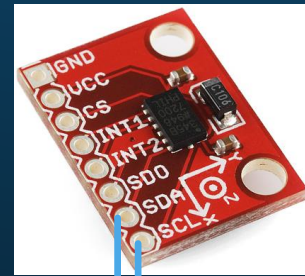
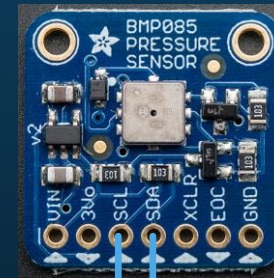
- ▶ Differently from the USART, SPI can talk to multiple devices on the same bus, but needs a Slave Select signal per Slave Device
- ▶ Up to 10Mbps! (clk/2)
- ▶ Slaves do not "talk" autonomously.
 - ▶ Must be queried (and clocked) by master.



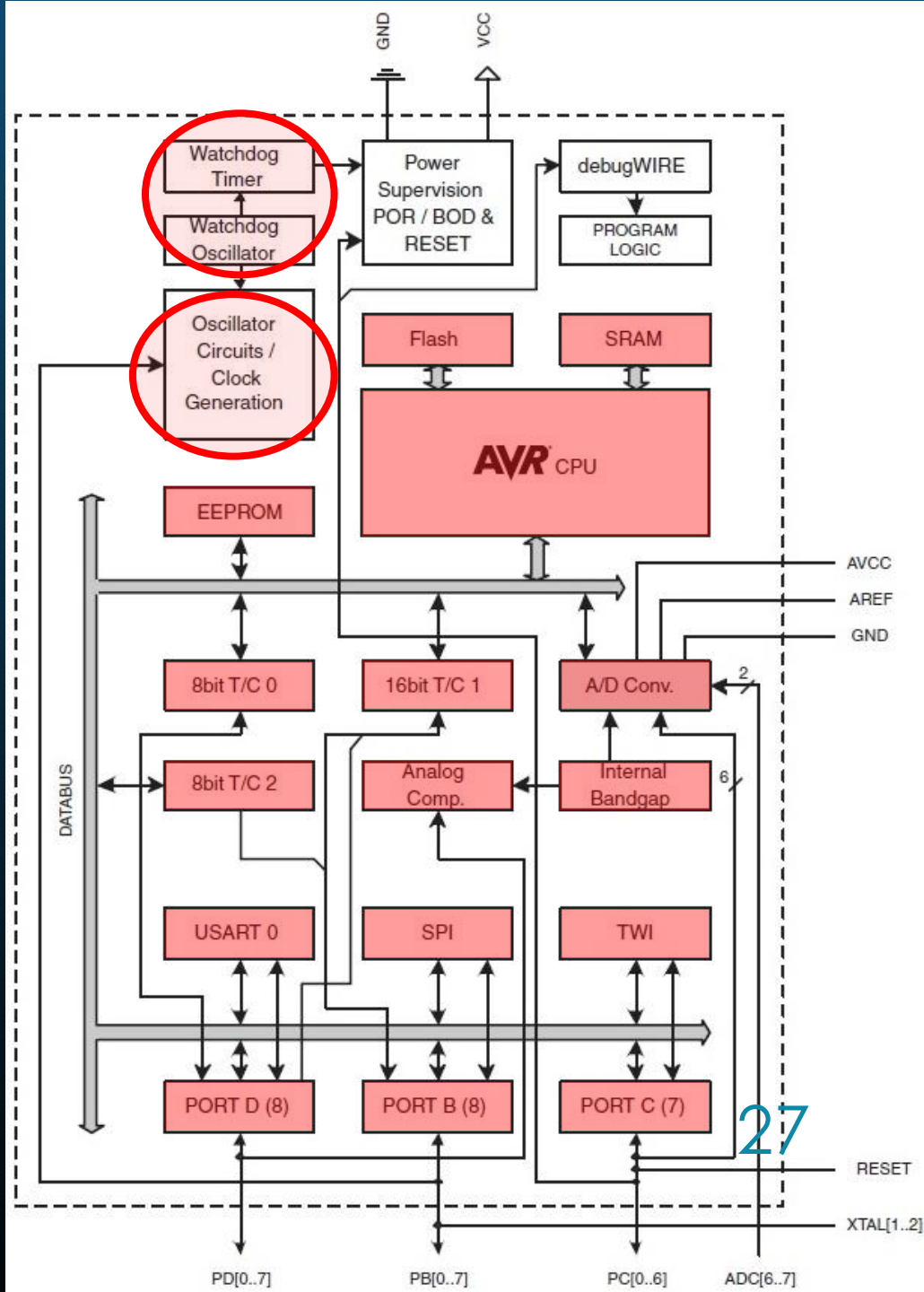
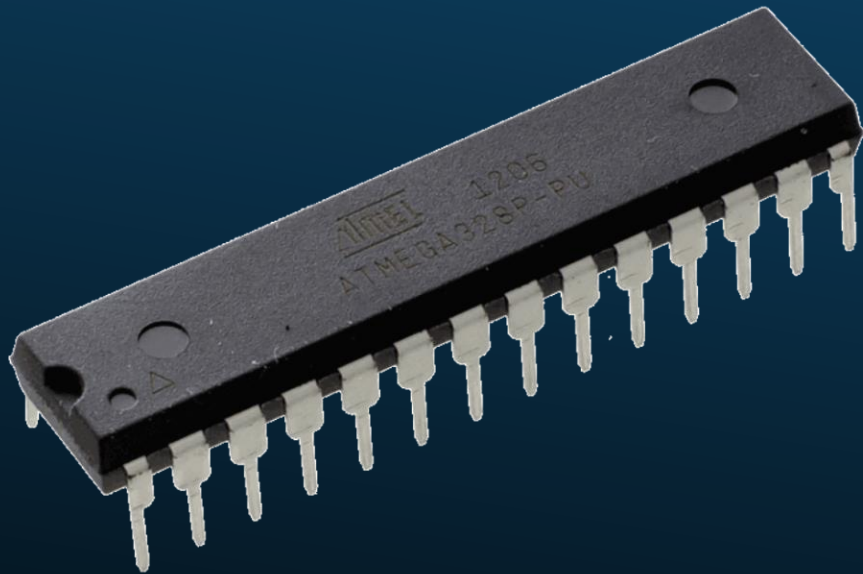
SERIAL INTERFACES: TWI (I²C)

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

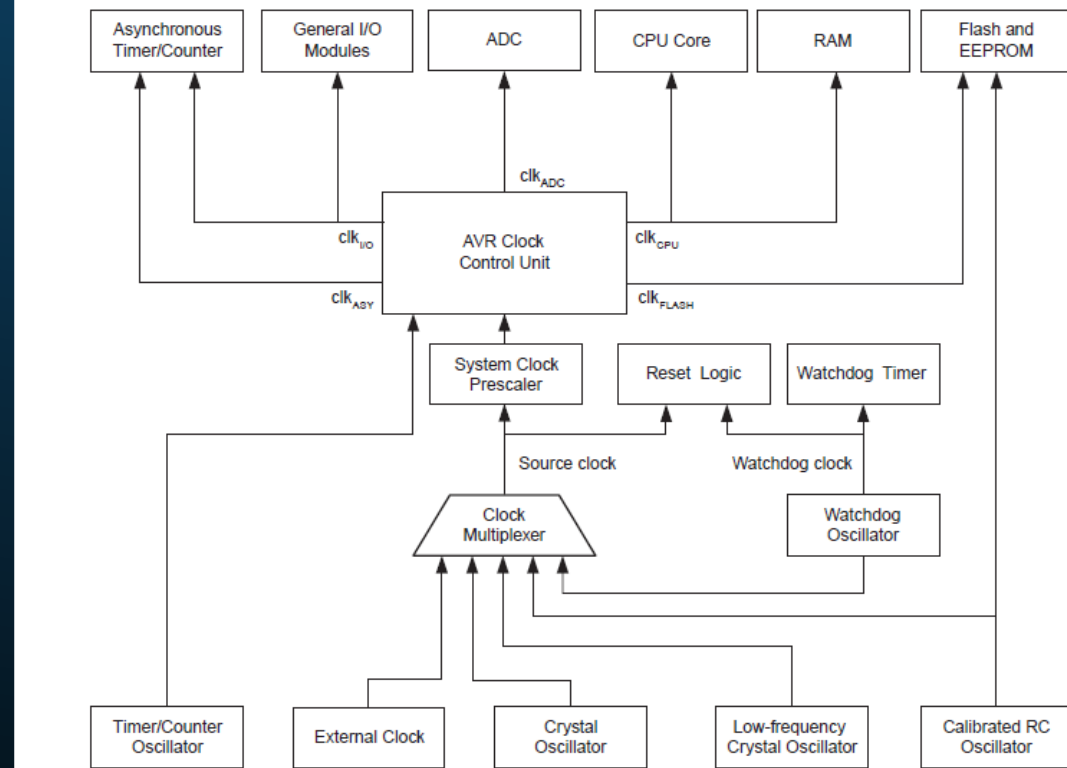
- ▶ Up to 20MHz from:
 - ▶ External clock from a pin
 - ▶ External crystal oscillator
 - ▶ Internal RC oscillator
 - ▶ 7.3-8.1 MHz
 - ▶ 128kHz Internal oscillator
 - ▶ 128 kHz

- ▶ System Clock Prescaler
 - ▶ Divides the clock if needed

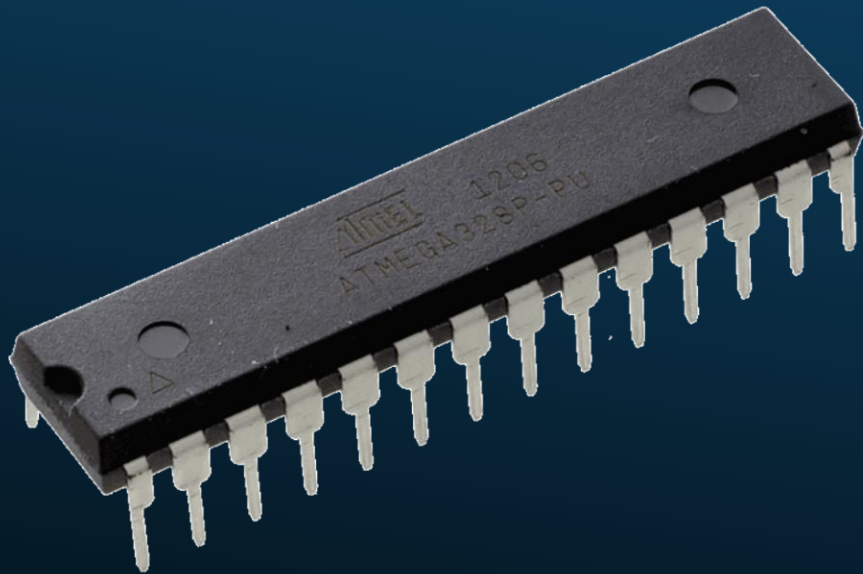
- ▶ Keep in mind:

Power consumption is proportional to clock frequency.

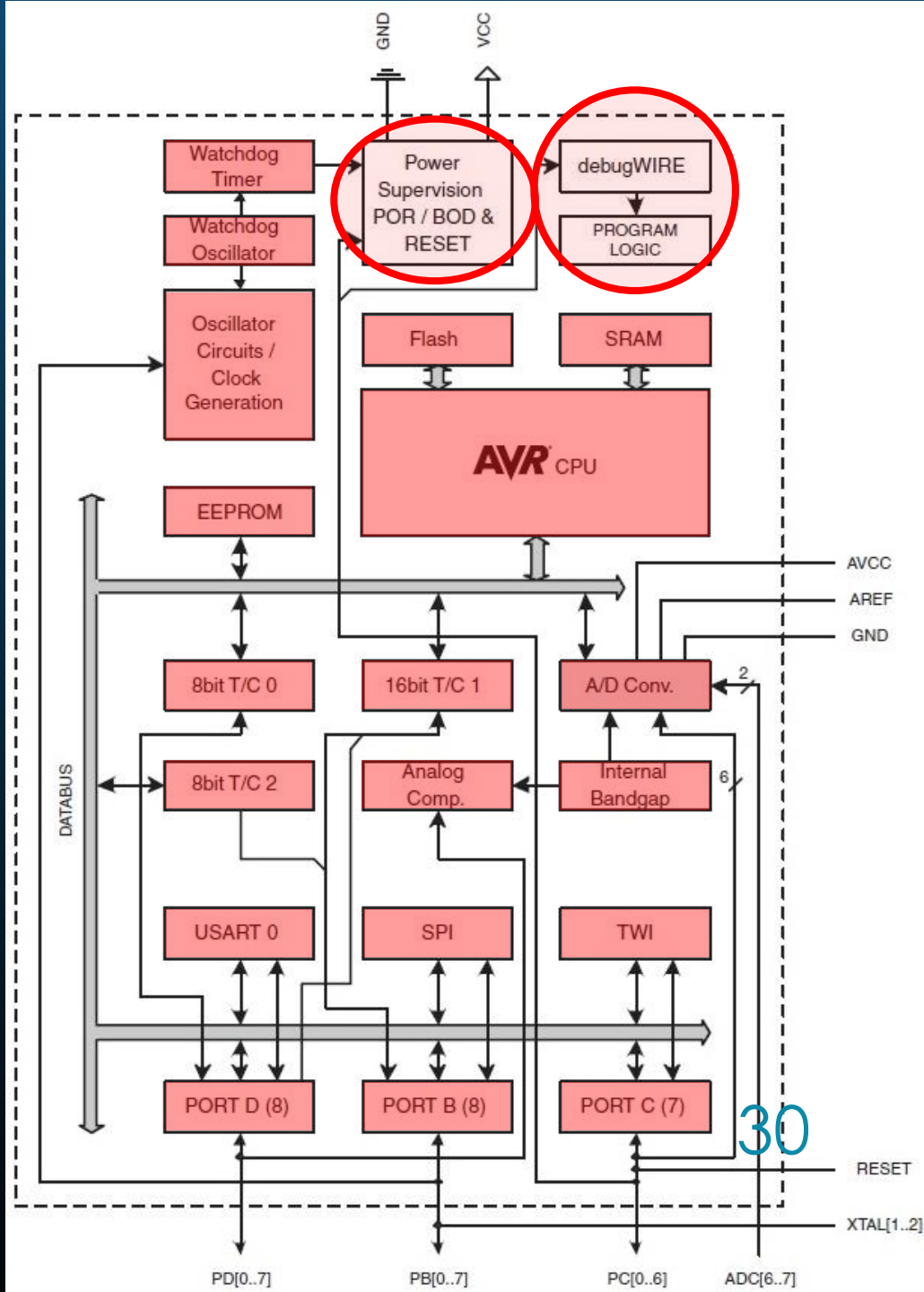
Figure 8-1. Clock Distribution



AVR ARCHITECTURE (ATMEGA328P)



m.feo@cern.ch – ISOTDAQ 2016



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
	<u>Power-down mode⁽³⁾</u>	WDT enabled, $V_{CC} = 3V$		<u>4.2</u>	8	μA
		WDT disabled, $V_{CC} = 3V$		<u>0.1</u>	2	μA

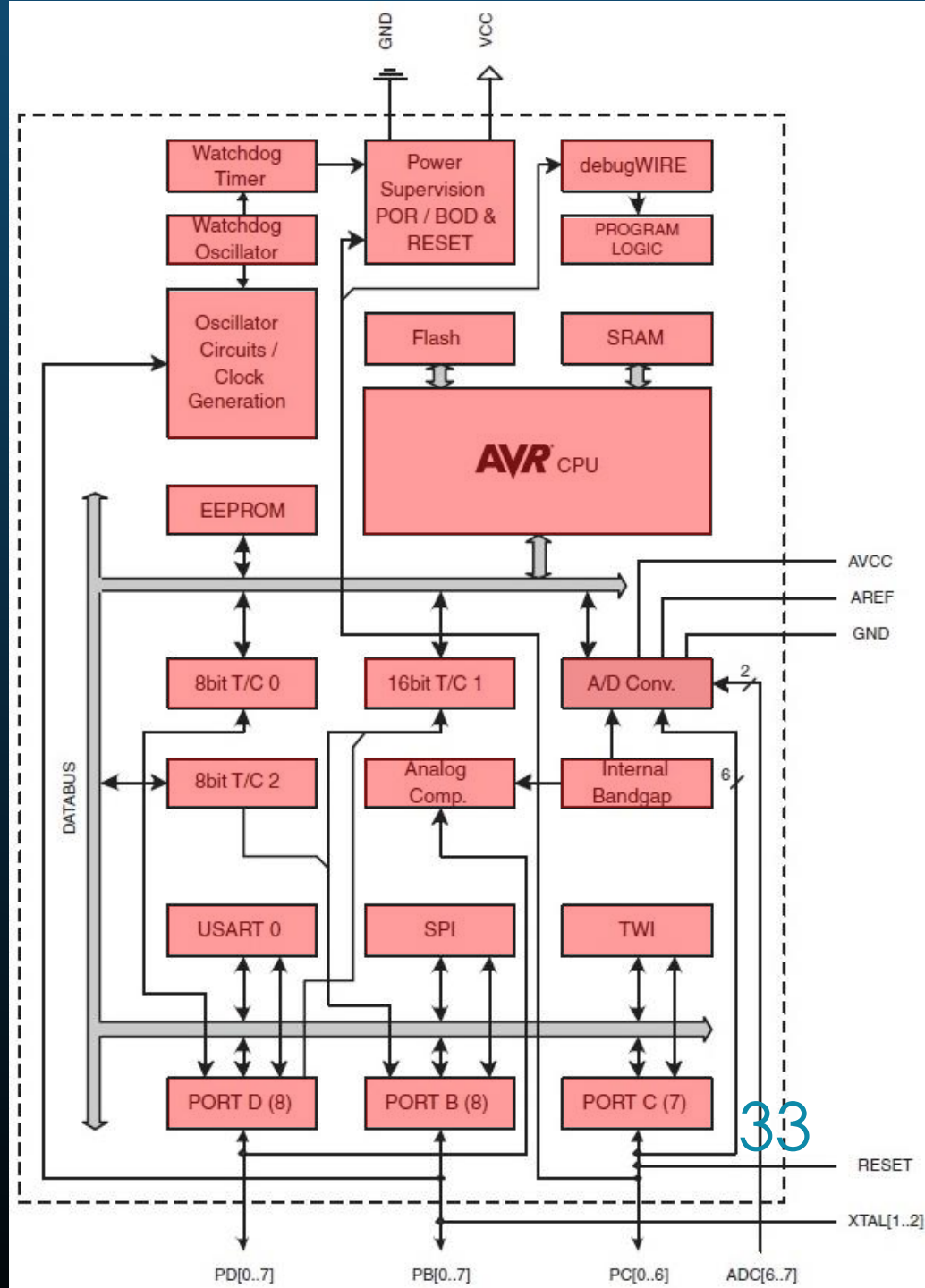
Sleep Mode	Active Clock Domains					Oscillators		Wake-up Sources							Software BOD Disable
	clk _{CPU}	clk _{FLASH}	clk _{I/O}	clk _{ADC}	clk _{ASY}	Main Clock Source Enabled	Timer Oscillator Enabled	INT1, INT0 and Pin Change	TWI Address Match	Timer2	SPM/EEPROM Ready	ADC	WDT	Other I/O	
Idle			X	X	X	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		
<u>Power-down</u>								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 V_{cc} 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

Cool,
but how do I use it after all?



USAGE OF MICROCONTROLLERS

FIRST OF ALL

- ▶ Is a microcontroller suitable for my application?
 - ▶ Cost, development time, power consumption, processing power, timing requirements, etc.
- ▶ Alternatives:
 - ▶ FPGA
 - ▶ Processor / computer
 - ▶ DSP
 - ▶ ASIC
 - ▶ FPGA with embedded CPU
 - ▶ Etc.

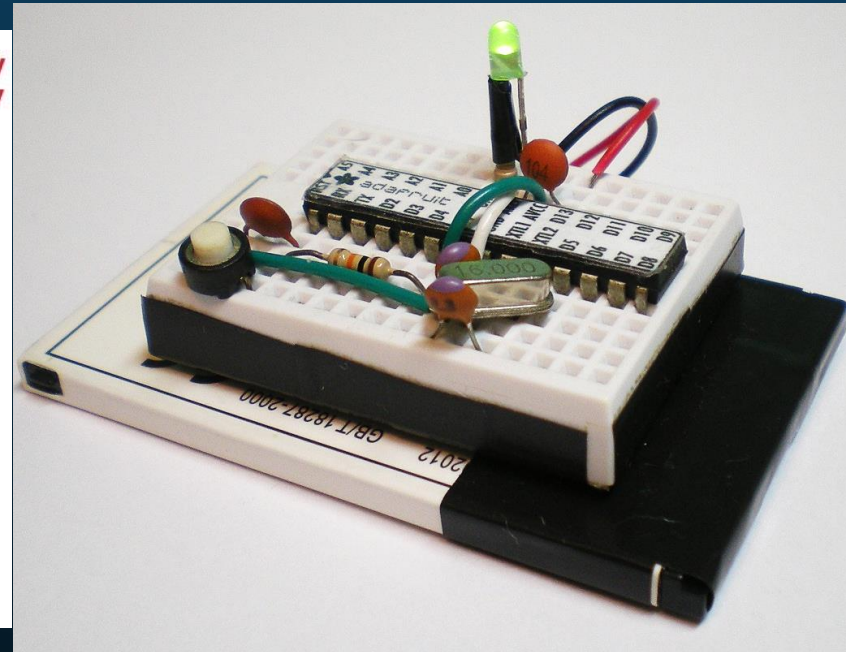
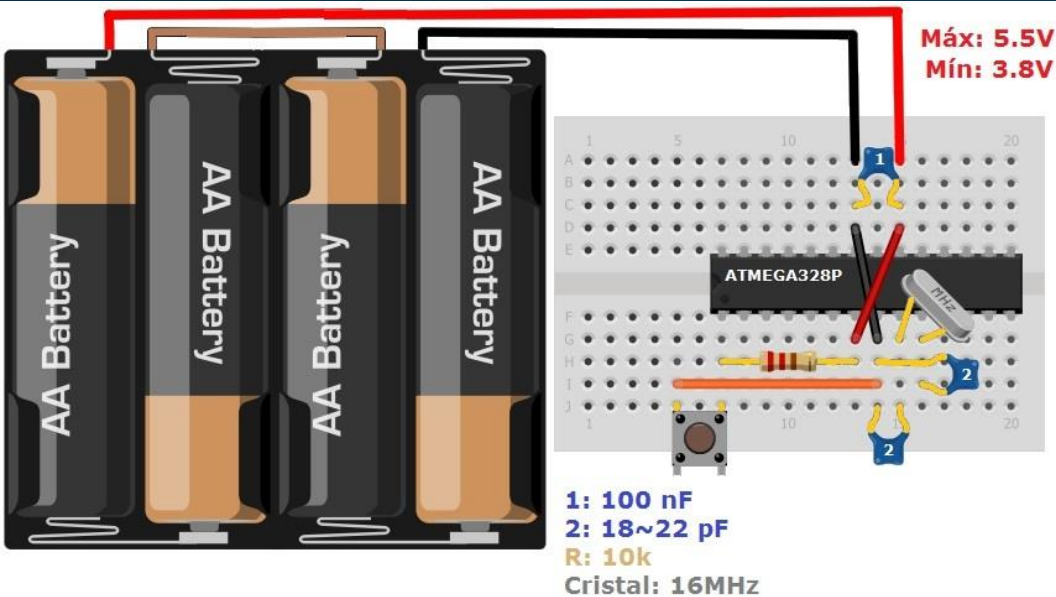
USAGE OF MICROCONTROLLERS

CHOOSING A MICROCONTROLLER

- ▶ What kind of problem do I have?
- ▶ Which kind of sensors/actuators will be used?
- ▶ What are the required peripherals?
- ▶ What is the environment? (Space? Right next to the LHC beam?)
- ▶ The best solution is the one that meets your requirements with the least amount of time and money invested.

ATMEGA328 MINIMUM REQUIRED CIRCUIT

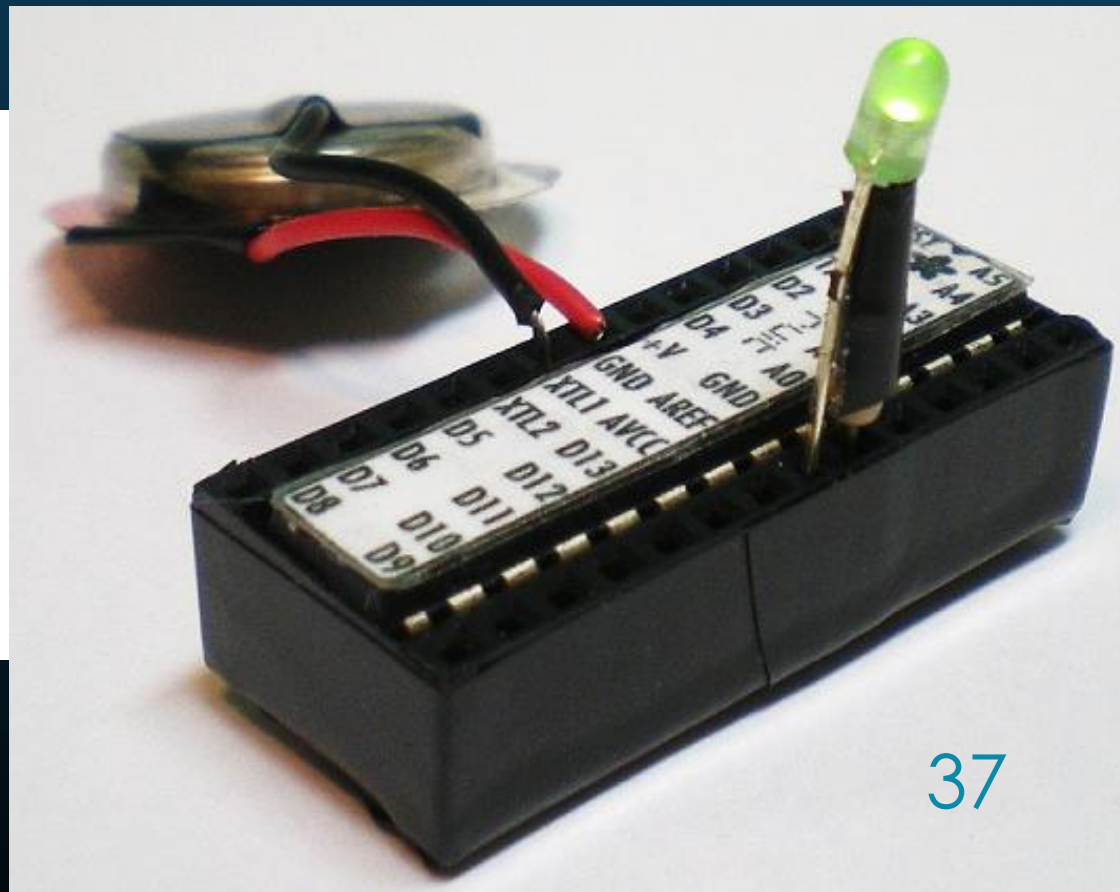
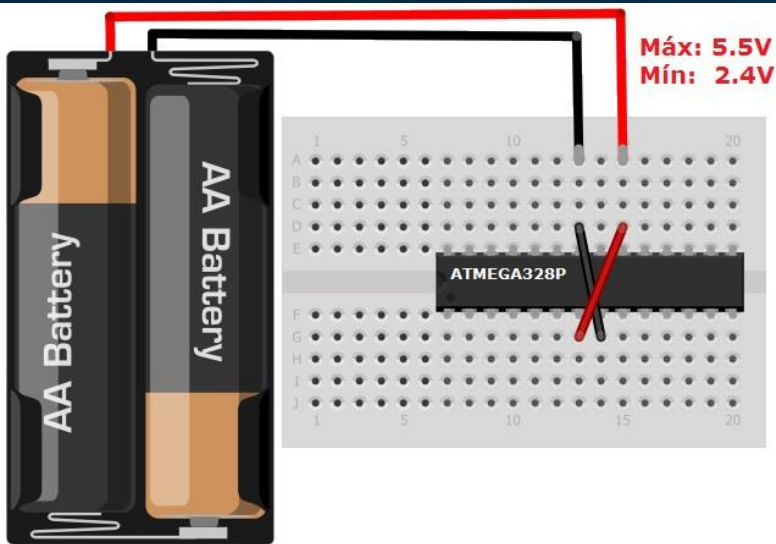
NEEDED CIRCUITRY: USING AN EXTERNAL CRYSTAL



ATMEGA328 MINIMUM REQUIRED CIRCUIT

NEEDED CIRCUITRY: USING THE INTERNAL OSCILLATOR

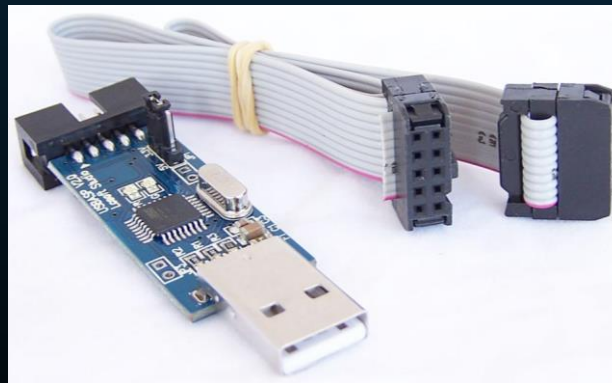
- ▶ Atmega328 comes with an internal 8MHz oscillator that can minimize the required circuit to a single battery.



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)

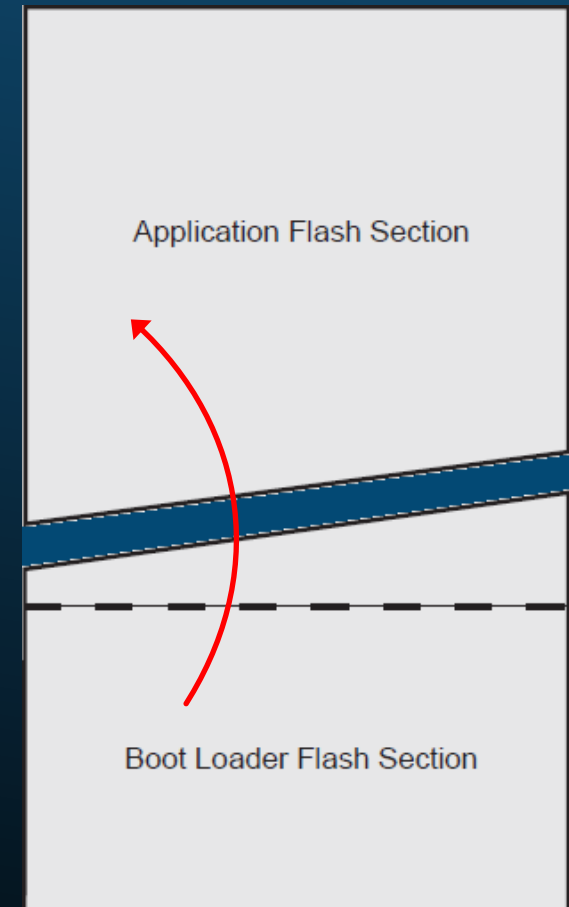
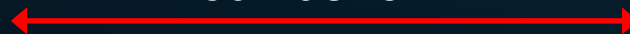


SELF-PROGRAMMING: BOOTLOADER

- ▶ The AVR core can write to it's own program memory.
- ▶ The Bootloader Section can be locked from rewriting.
- ▶ 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

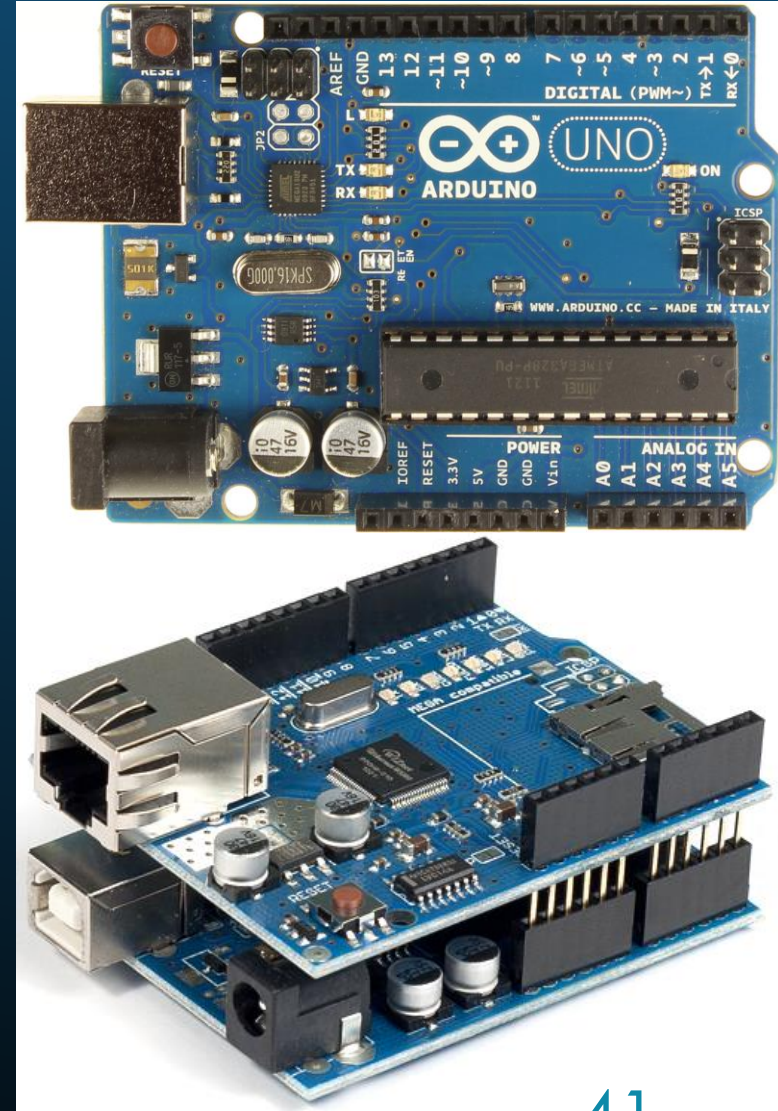


ARDUINO



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;
```

Set direction of pin 6 from PORTD to output.

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

Set WGM to Fast PWM.

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

Set COM to «Clear On Compare».

```
OCR0A = 0x3F;
```

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

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

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);
```

$$\frac{25}{100} = \frac{x}{255} \quad \text{Find } x.$$



eLua

From Assembly to the Moon





eLua

From Assembly to the Moon

(and Mars lately)



- ▶  Lua for uCs (whole Lua, no cuts) 
- ▶ ~6 kb RAM needed (depending on application)
- ▶ Portable between different devices!
- ▶ Free and open-source



SETTING PWM TO 25% DUTY CYCLE

THE ELUA'S WAY



Set frequency used by PWM timer

```
pwm.setclock( 6, 16000000 )  
pwm.setup(6, 16000000, 25)  
pwm.start(6)
```

Set PWM frequency + duty cycle in %

Start PWM!

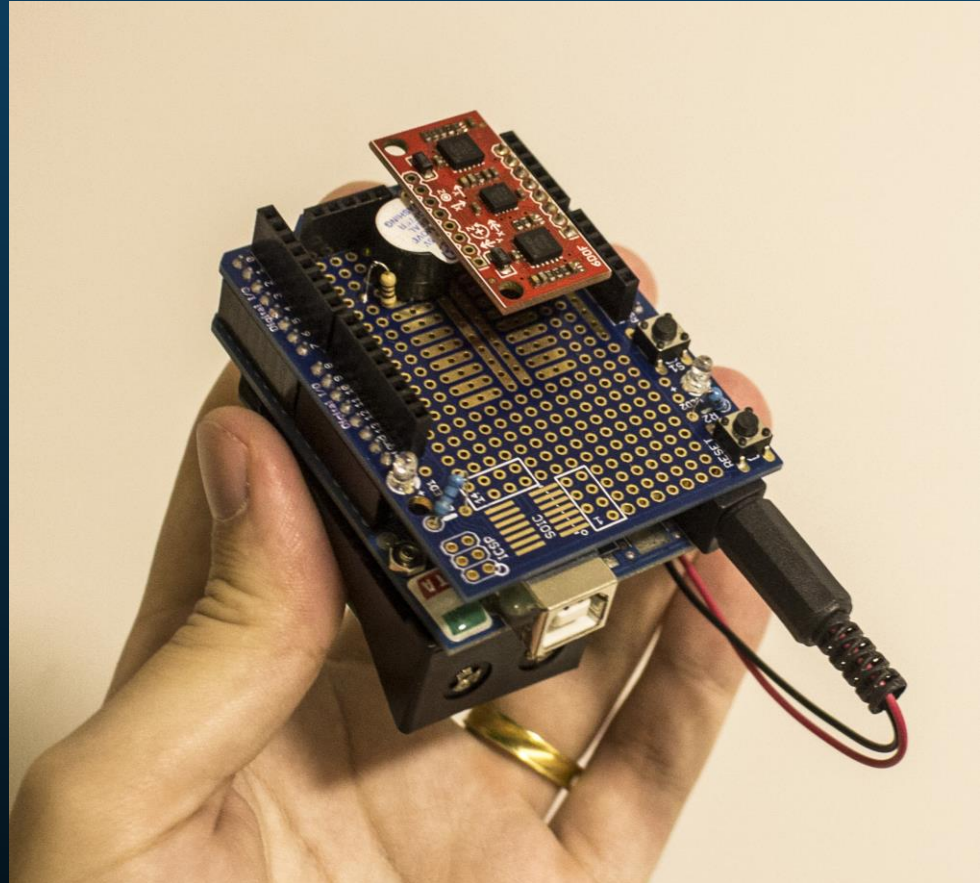
LAB 10



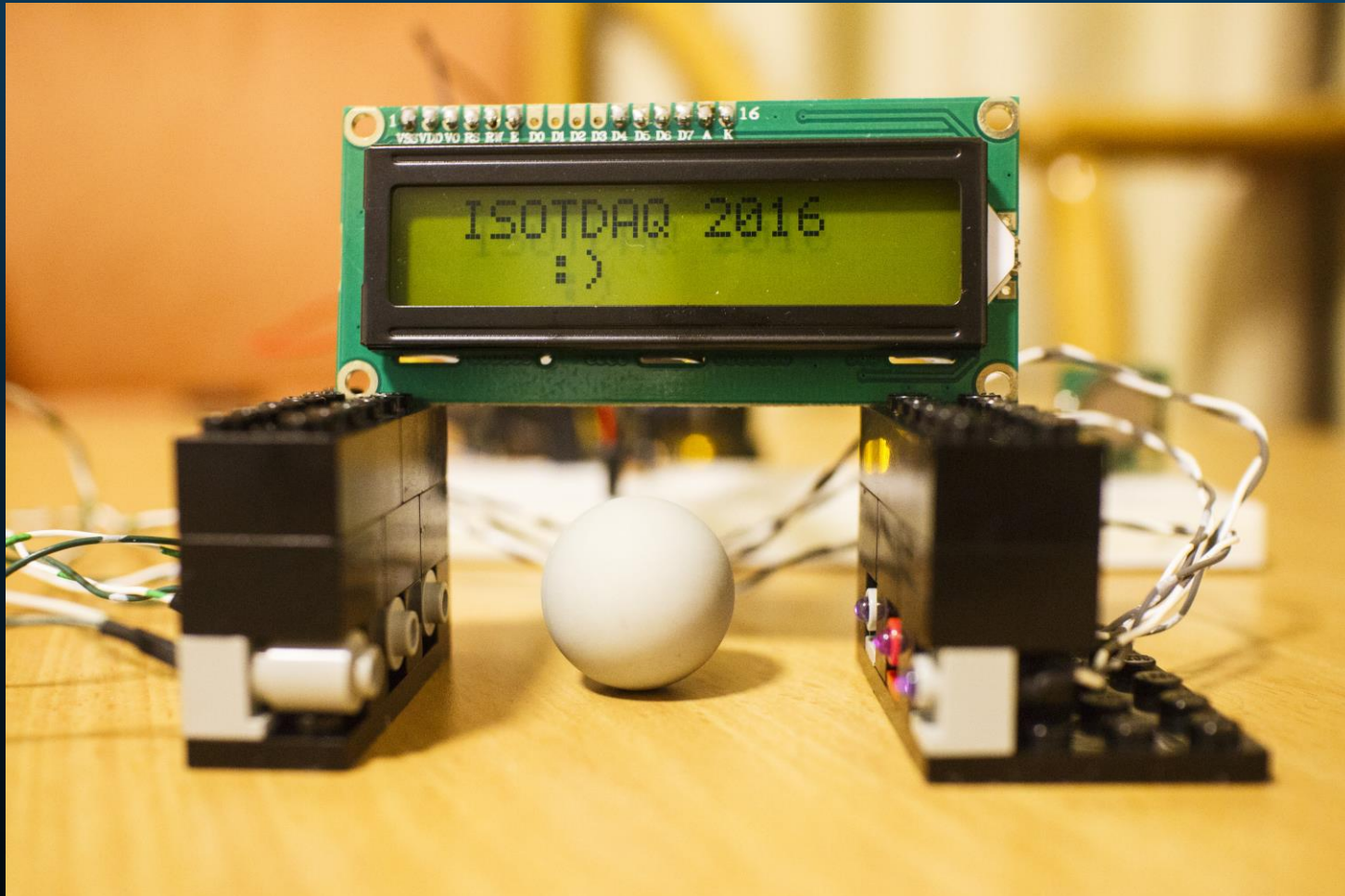
LAB 10



FREE-FALL DETECTION



LEGO SPEED LIMIT ENFORCEMENT

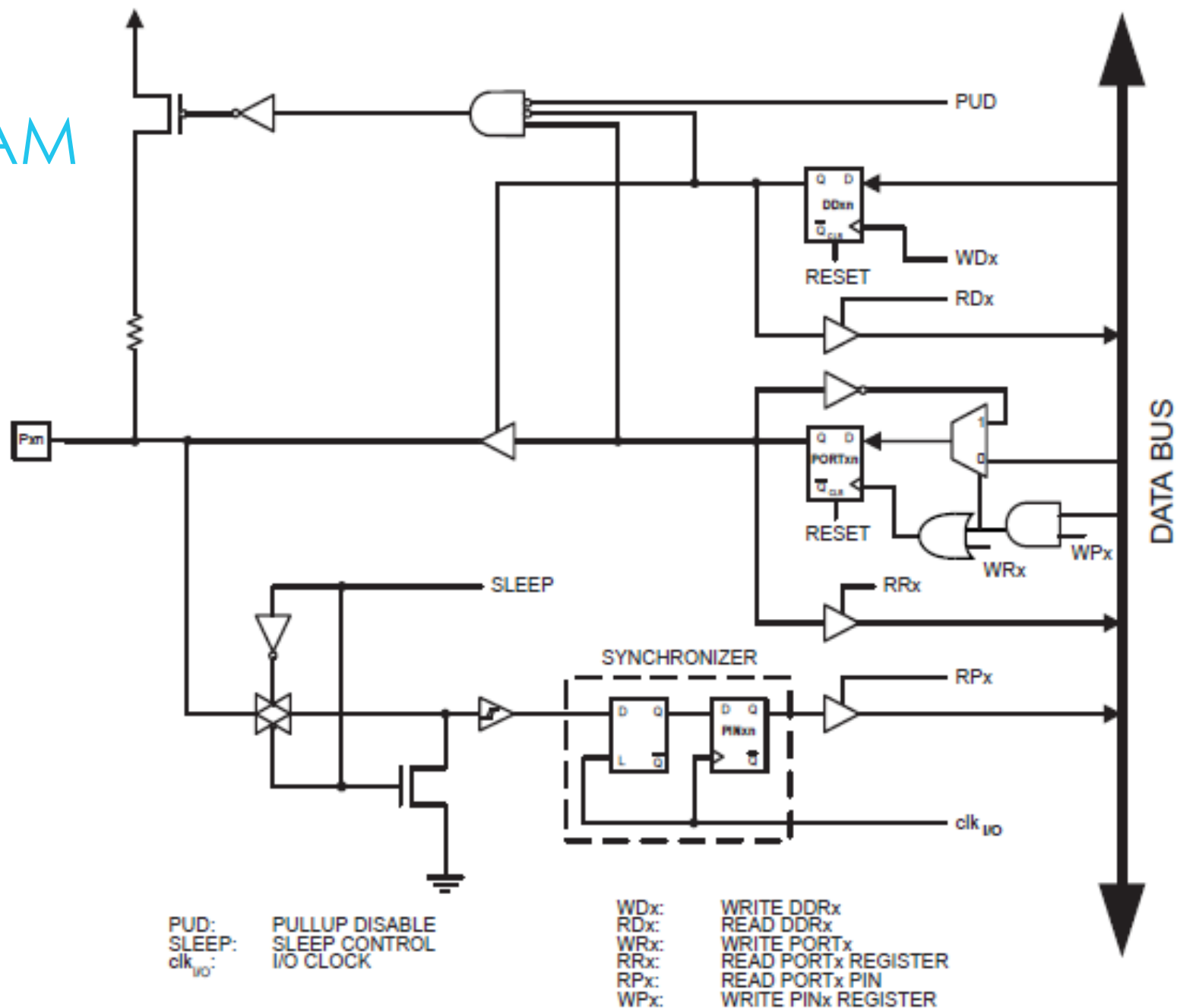


THAT'S IT.
OBRIGADO!

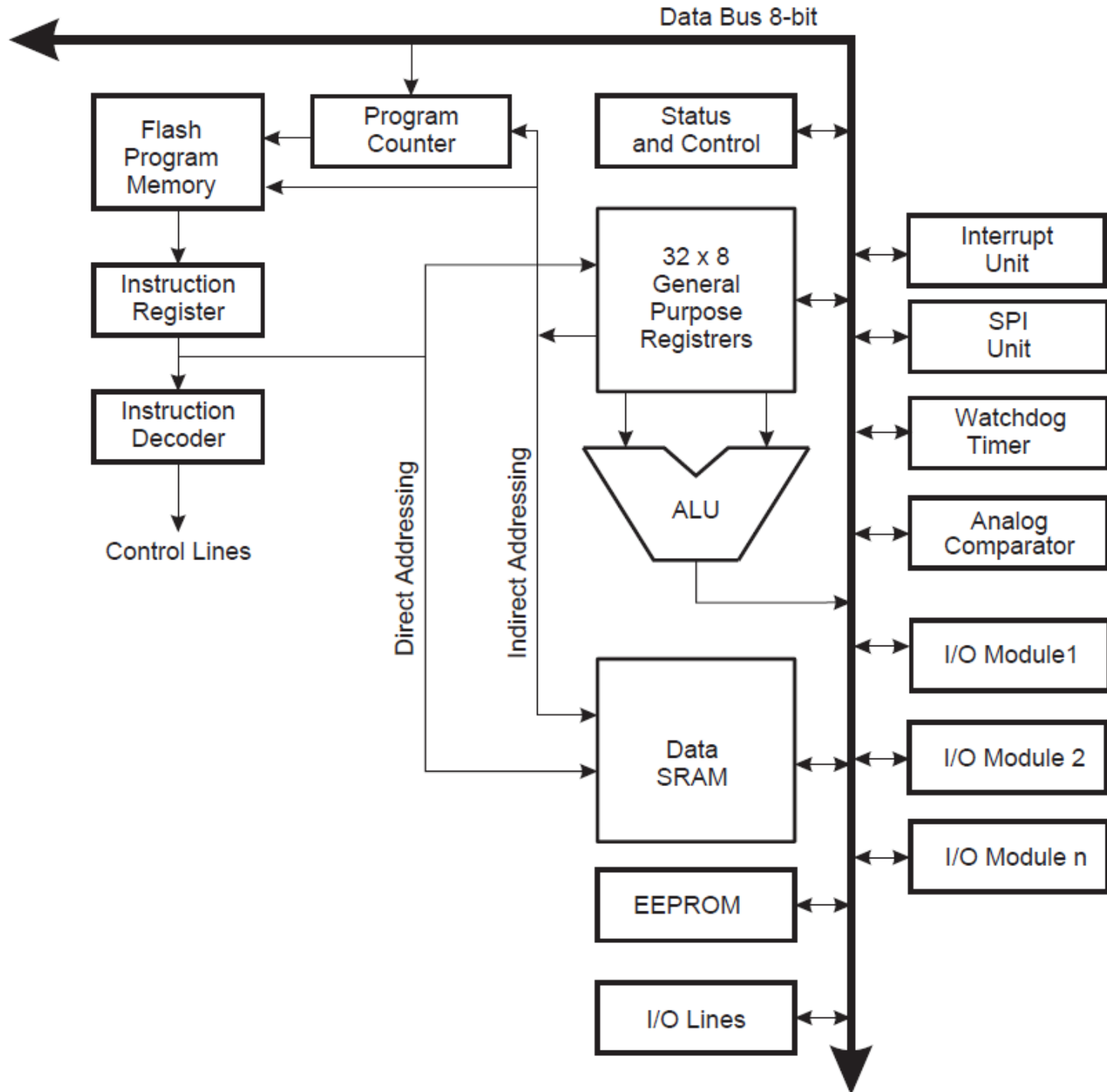
Maurício Féo Rivello
m.feo@cern.ch

Figure 13-2. General Digital I/O⁽¹⁾

GPIO DIAGRAM

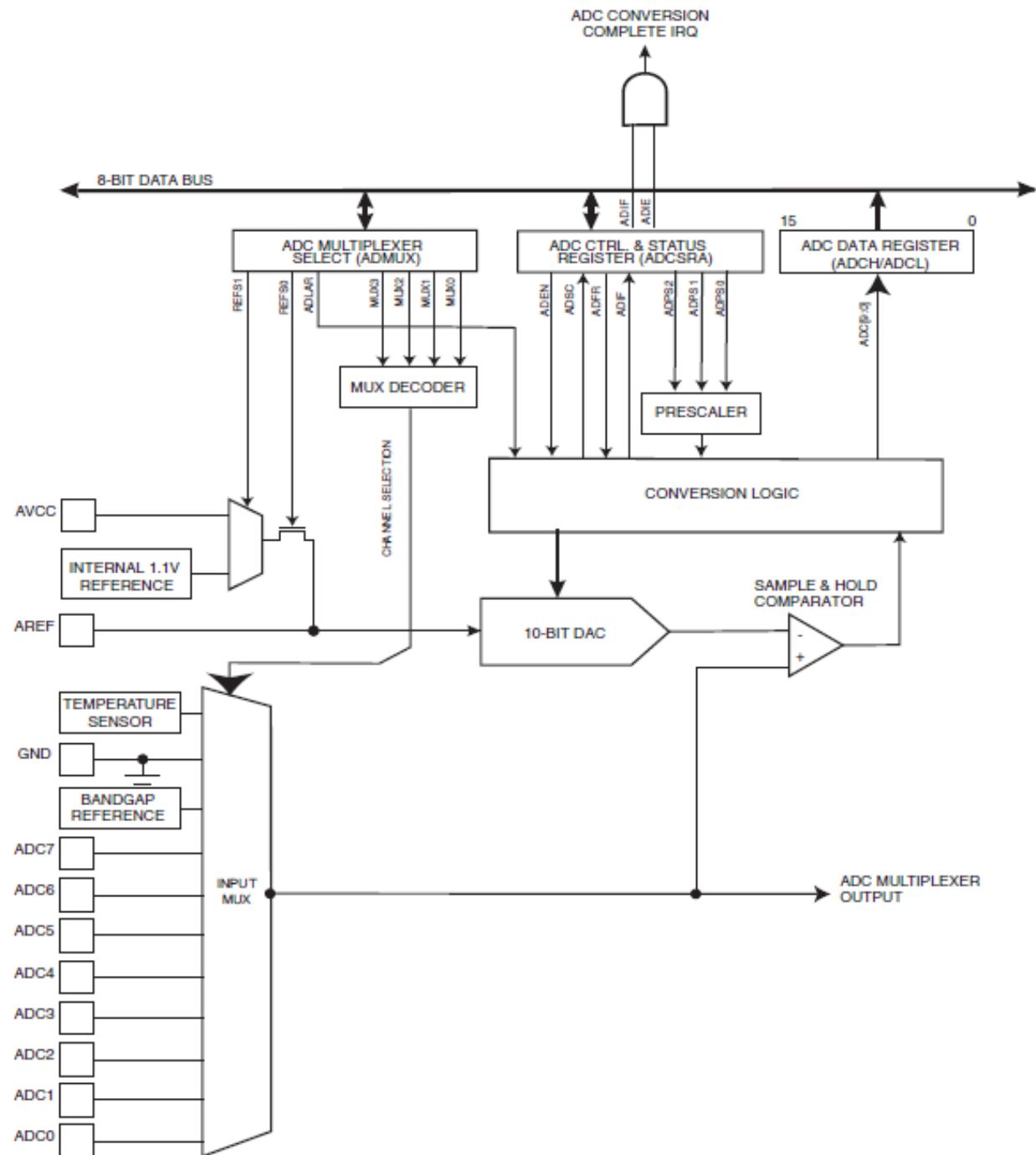


CPU DIAGRAM

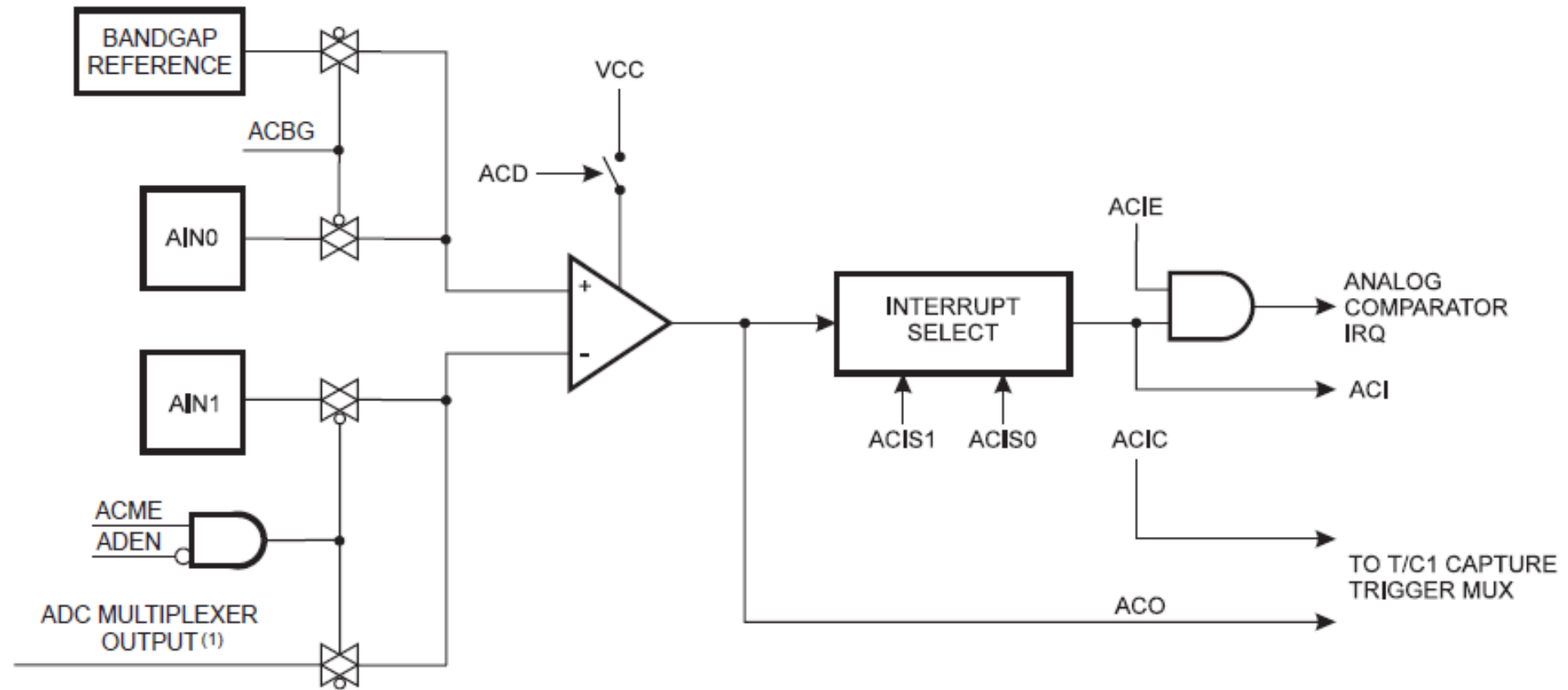


ADC DIAGRAM

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



ANALOG COMPARATOR



USART

Figure 19-1. USART Block Diagram⁽¹⁾

