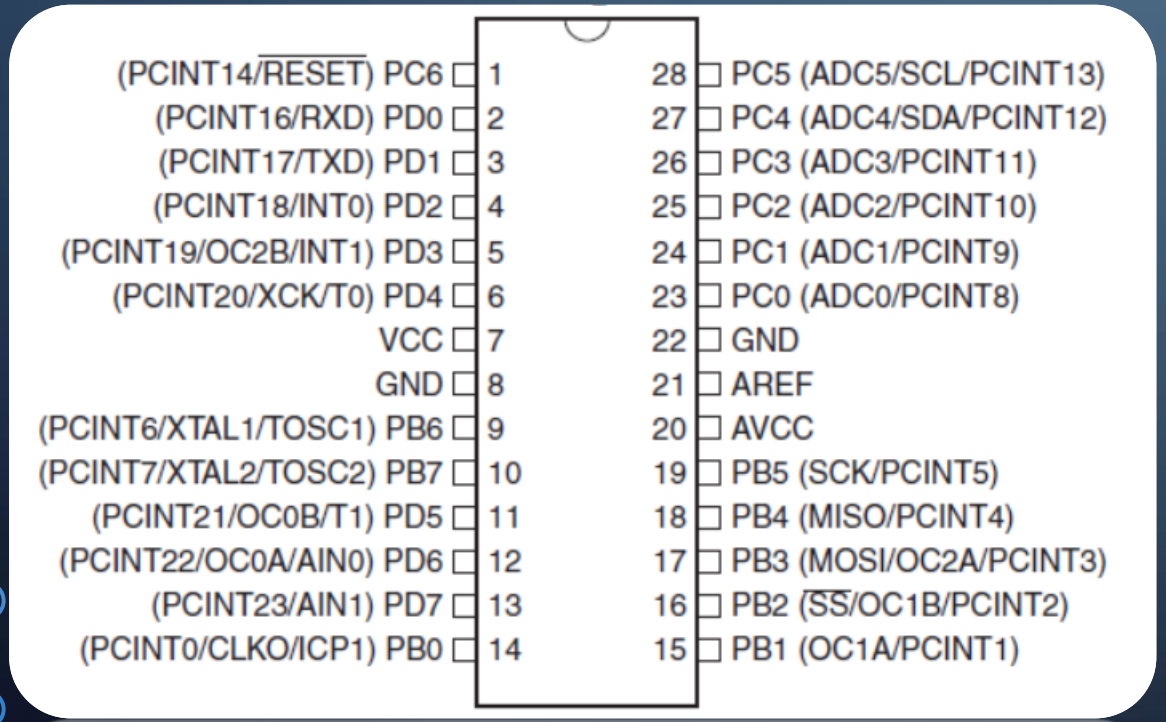
ATMEGA328 BASE

PRIMO APPROCCIO CON IL MICROCONTROLLORE

FONTI

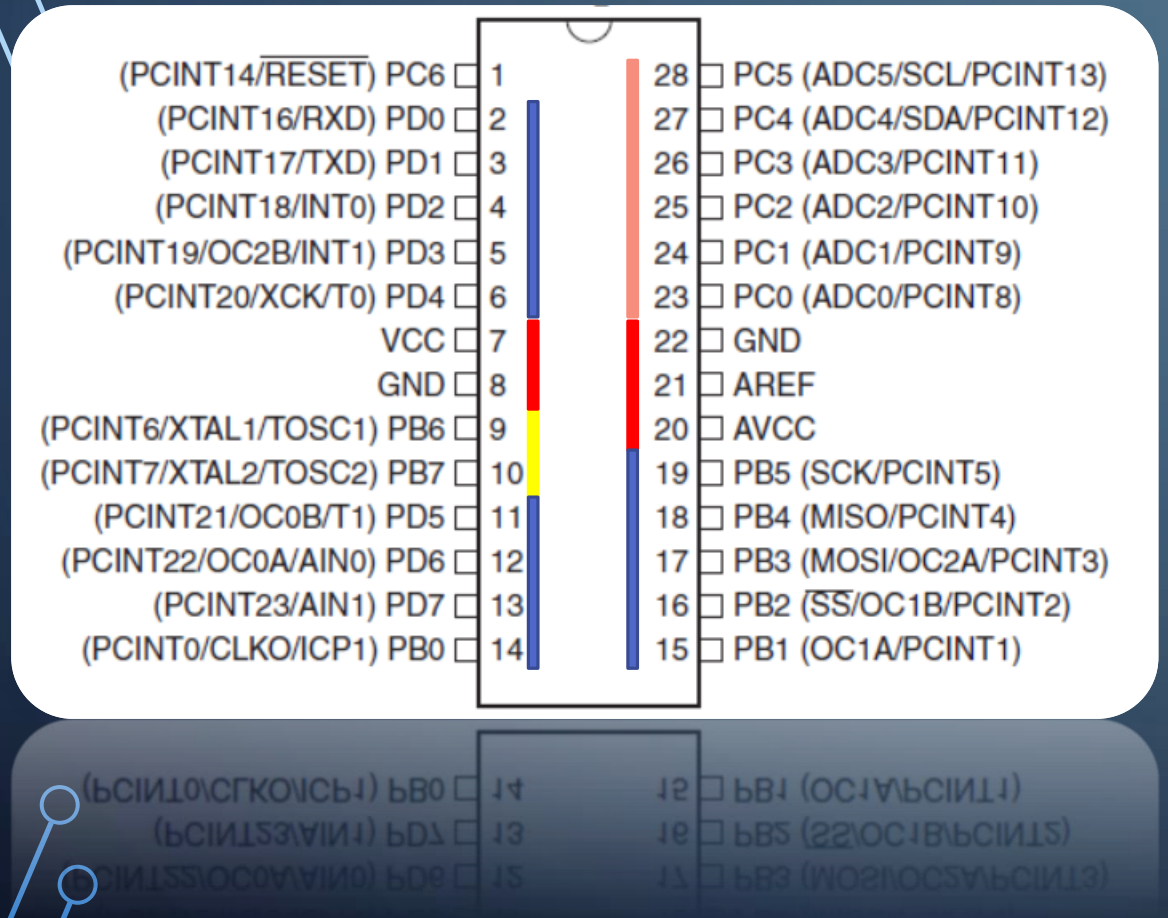
- ATmega328-328P_Datasheet
- https://www.theengineeringprojects.com/2017/08/introduction-to-atmega328.html
- https://courses.cs.washington.edu/courses/csep567/10wi/lectures/Lecture6.pdf

ATMEGA328 PINOUT



1	PC6	Reset
2	PD0	Digital Pin (RX)
3	PD1	Digital Pin (TX)
4	PD2	Digital Pin
5	PD3	Digital Pin (PWM)
6	PD4	Digital Pin
7	Vcc	Positive Voltage (Power)
8	GND	Ground
9	XTAL 1	Crystal Oscillator
10	XTAL 2	Crystal Oscillator
11	PD5	Digital Pin (PWM)
12	PD6	Digital Pin (PWM)
13	PD7	Digital Pin
14	PB0	Digital Pin
15	PB1	Digital Pin (PWM)
16	PB2	Digital Pin (PWM)
17	PB3	Digital Pin (PWM)
18	PB4	Digital Pin
19	PB5	Digital Pin
20	AVCC	Positive voltage for ADC (power)
21	AREF	Reference Voltage
22	GND	Ground
23	PC0	Analog Input
24	PC1	Analog Input
25	PC2	Analog Input
26	PC3	Analog Input
27	PC4	Analog Input
28	PC5	Analog Input

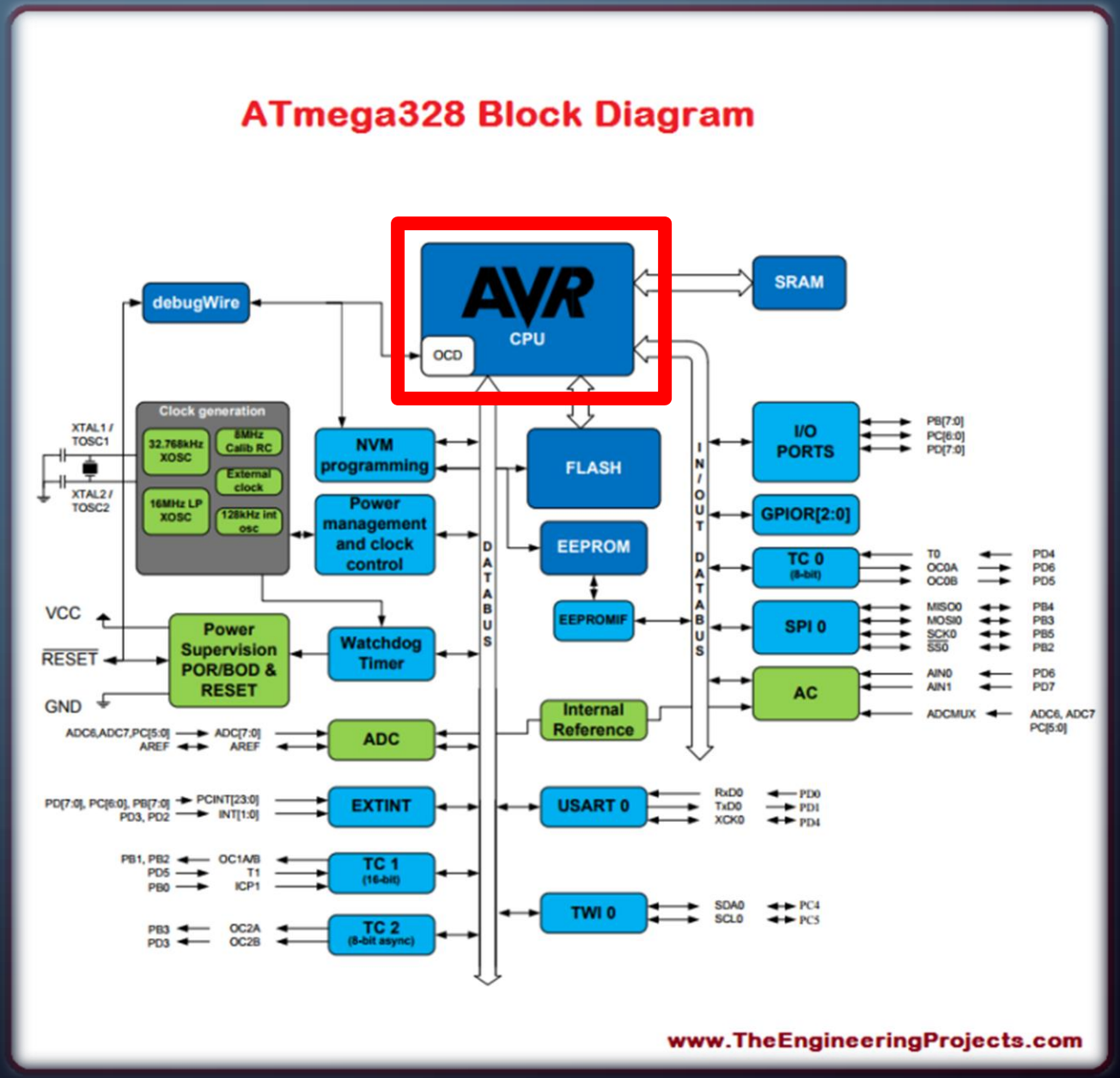
ATMEGA 328 PINOUT



- 2 pin sono l'ingresso dell'oscillatore a cristallo per il clock di sistema (TOSC1, TOSC2)
- 2 pin sono per l'alimentazione (VCC e GND)
- 3 pin alimentano l'ADC (AVCC, GND, AREF)
  - AREF è la tensione di riferimento utilizzata dall'ADC per convertire un segnale analogico nel corrispondente valore digitale
- 6 pin sono ingressi analogici (ADC0 – ADC5)
- 14 pin sono I/O digitali
  - 6 possono funzionare per fornire un'uscita PWM

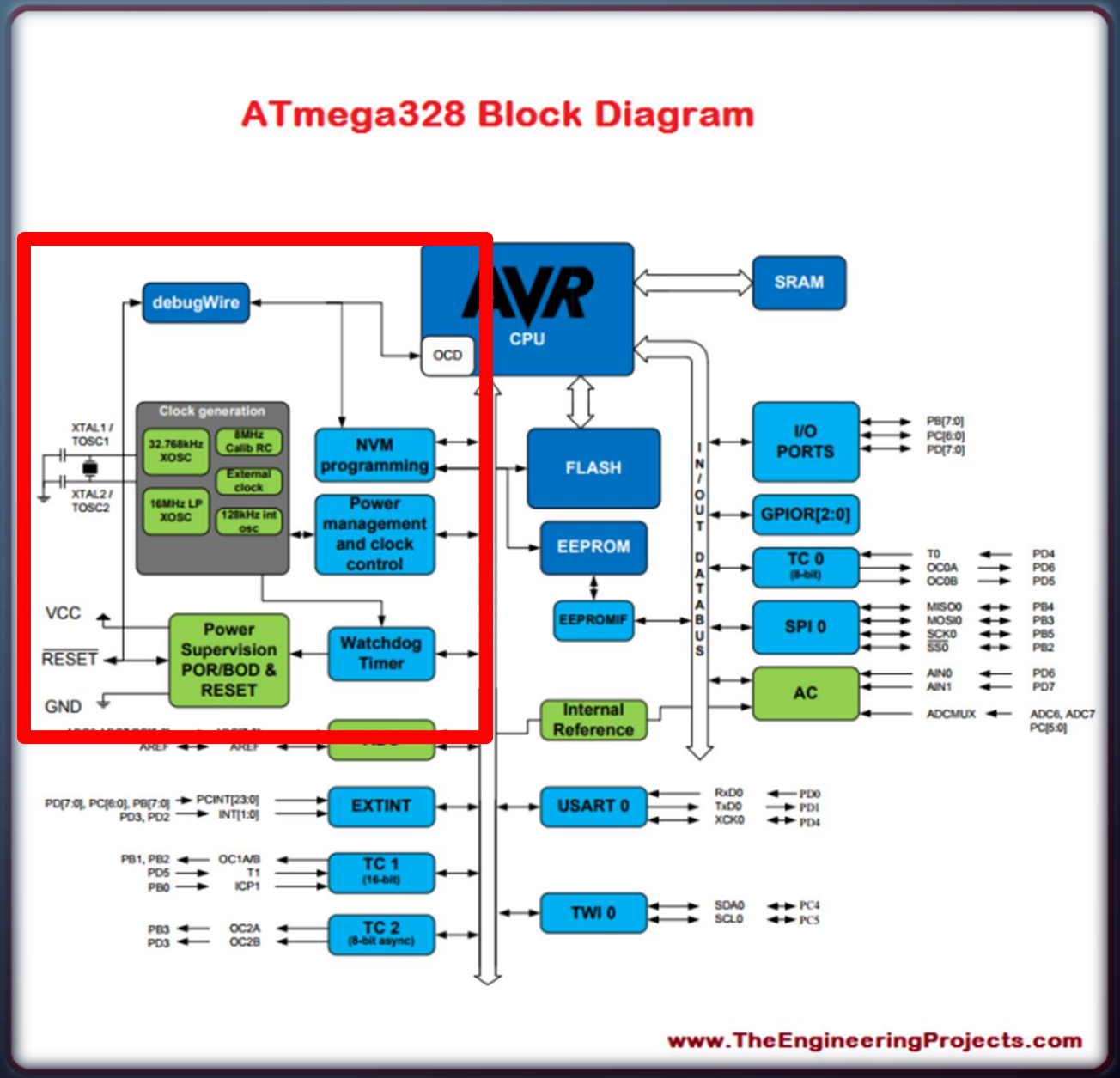
CPU

- AVR
  - microcontrollore RISC (set di istruzioni ridotto)
  - architettura Harvard (bus istruzioni e bus dati separati)
  - sviluppati dalla Atmel a partire dal 1996
  - 8 bit bus dati
  - 16 bit bus istruzioni
- Analizzata più in profondità successivamente



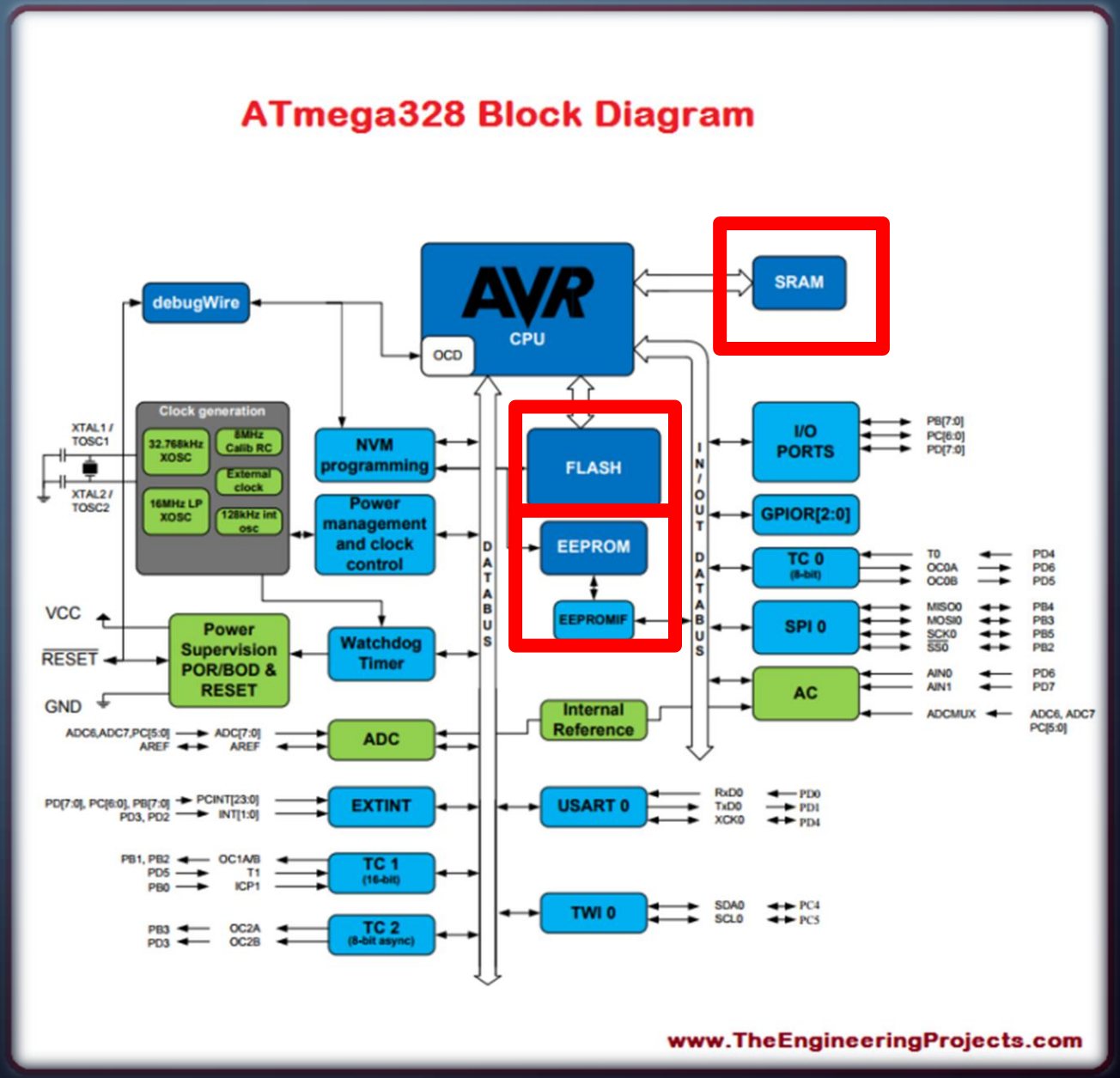
ALIMENTAZIONE

- Controllo dell'alimentazione
  - il microcontrollore si resetta se il valore di tensione scende troppo
- Clock
  - fmax = 20 MHz (Arduino 16 MHz)
- Watchdog timer
  - clock indipendente
  - resetta il microcontrollore se non viene resettato a intervalli regolari (evita blocchi della CPU)



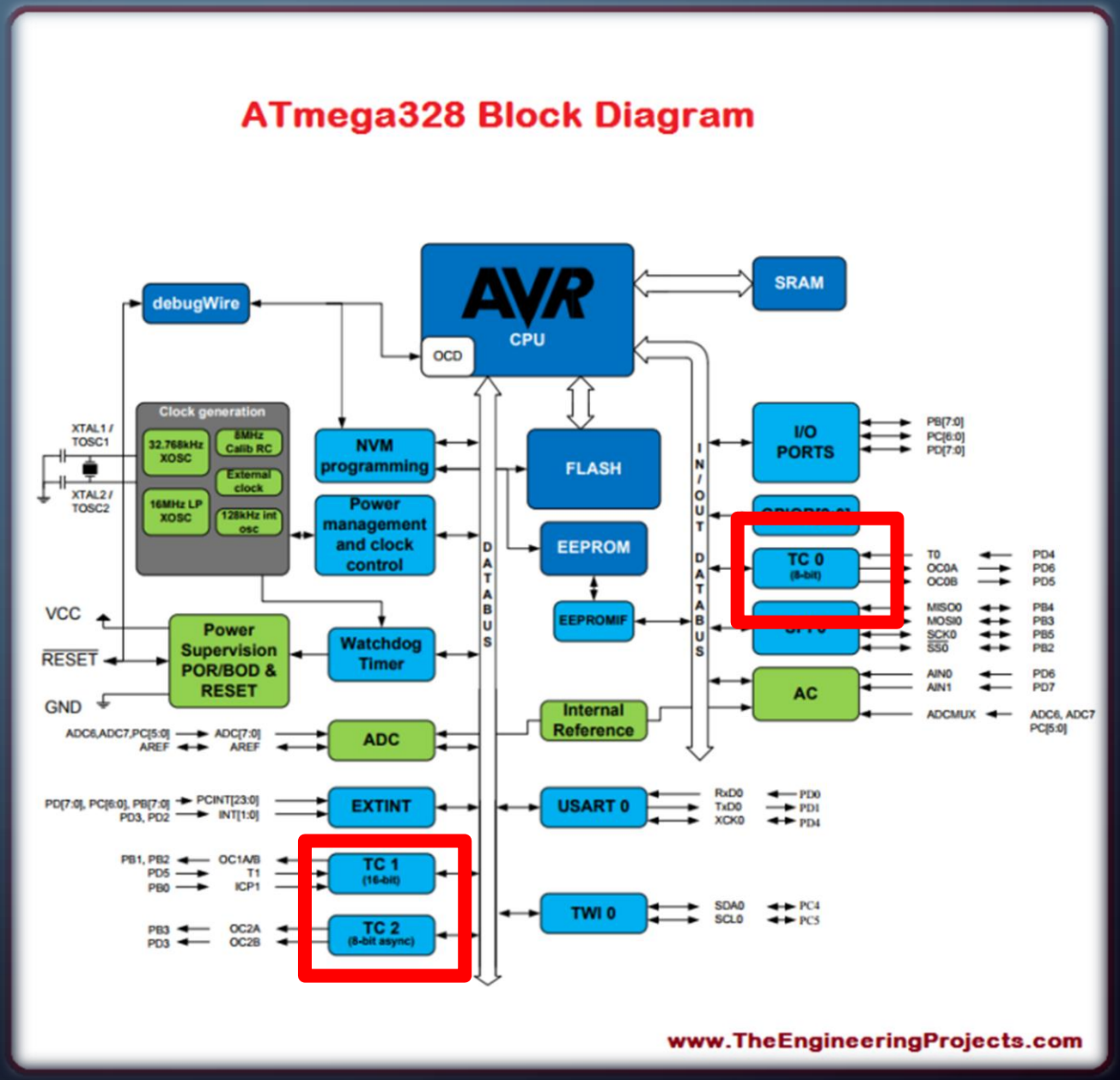
MEMORIA

- Flash
  - usata come memoria programma
  - 32 Kbyte
- SRAM
  - usata come memoria dei dati
  - 2 Kbyte
- EEPROM
  - per memorizzare dati a lungo termine (per esempio valori di taratura)
  - collegata anche al bus di I/O
  - 1 Kbyte



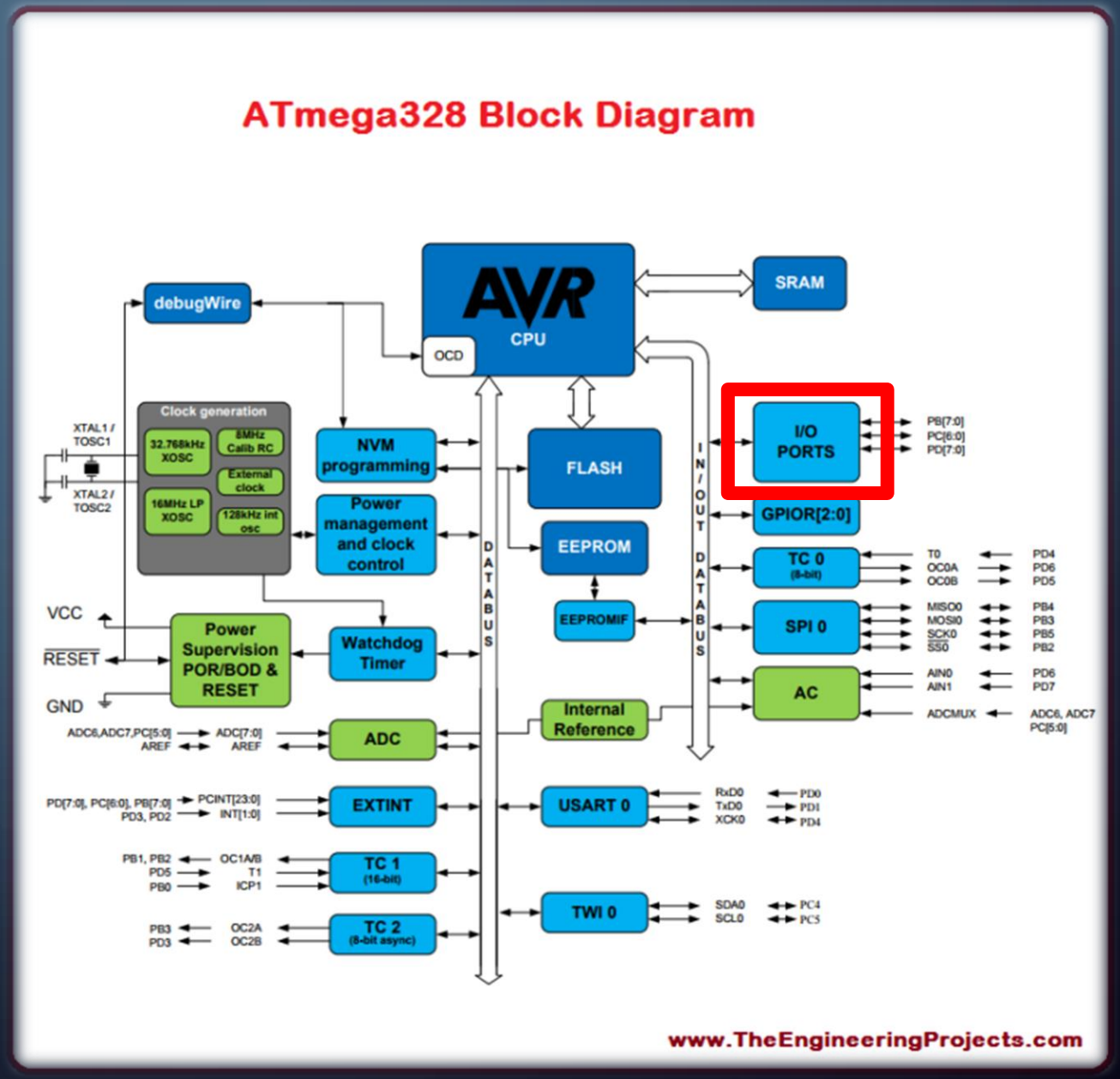
TIMER

- 3 timer nei quali si può:
  - Scegliere la velocità del clock
  - Scegliere fino a che valore contare (roll-over value)
  - Generare degli interrupt
  - Generare un segnale PWM



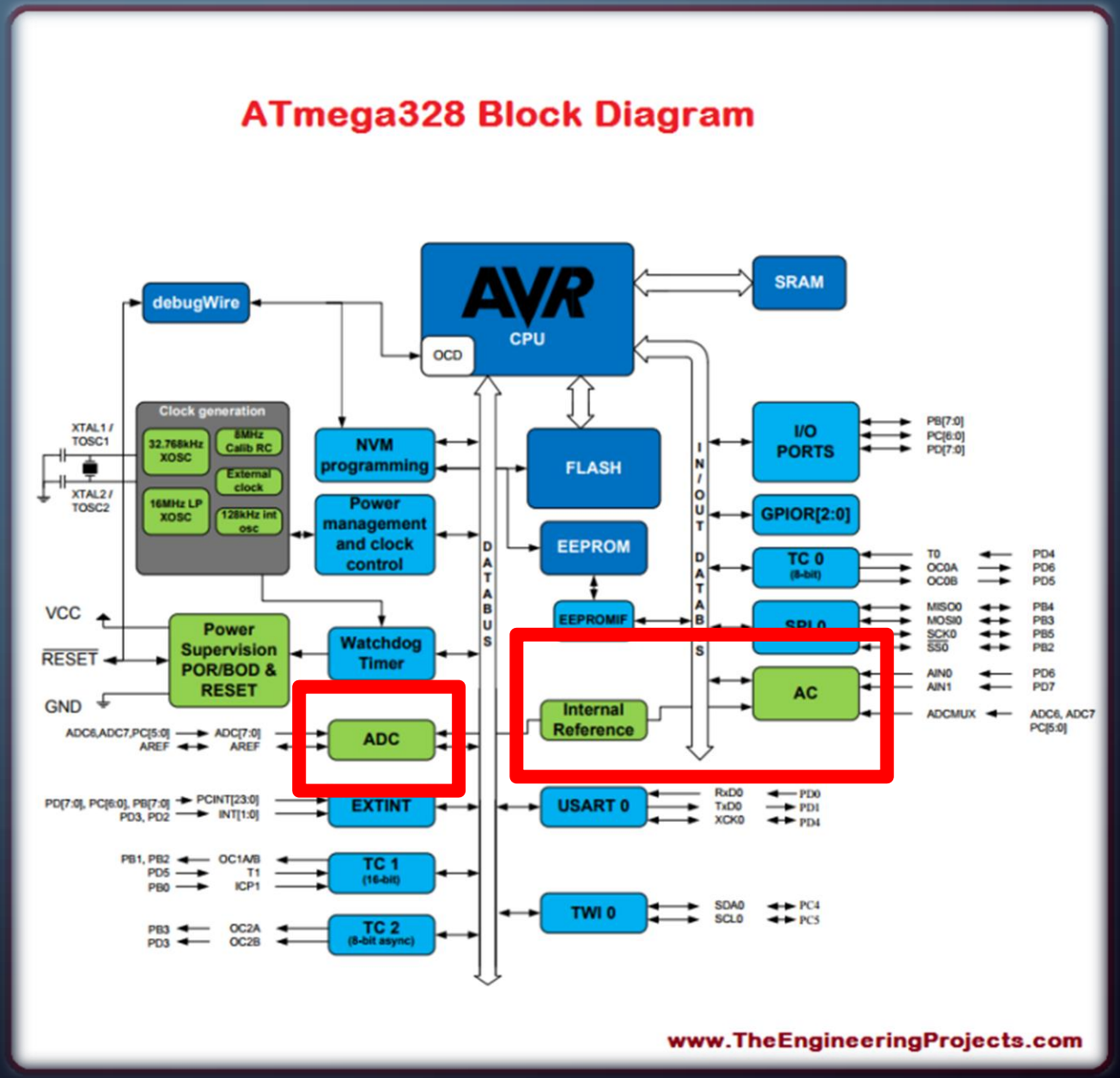
I/O DIGITALI

- 3 banchi (port) di I/O
  - ogni pin è direttamente programmabile come input o output
  - in ogni pin è possibile leggere o scrivere singolarmente un valore
  - è possibile inserire via software una resistenza di pull-up
- I pin possono essere condivisi con altre funzionalità:
  - ingressi analogici
  - clock
  - reset



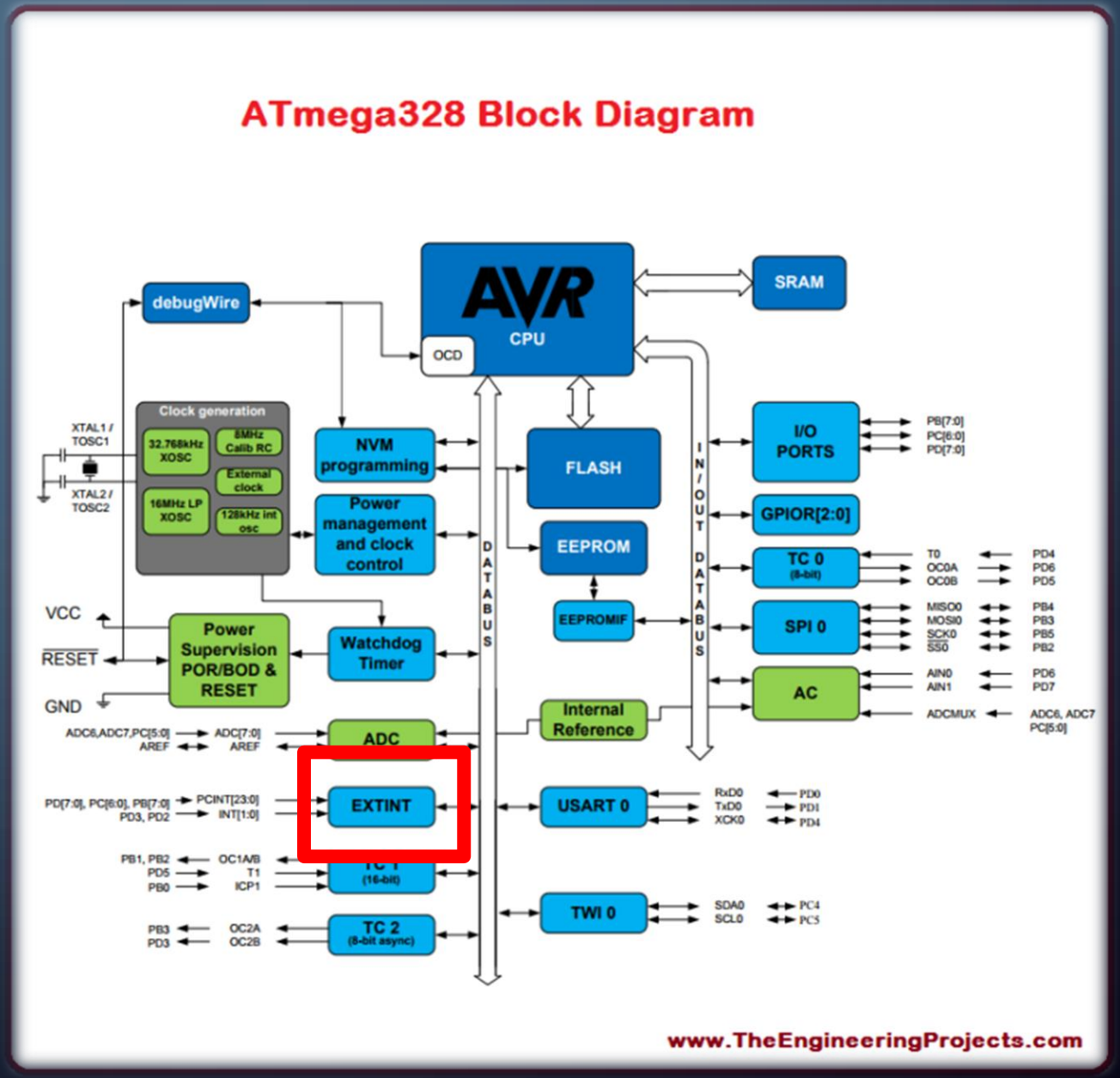
INPUT ANALOGICI

- Comparatore analogico
  - compara due segnali su sue pin
- Convertitore analogico-digitale (ADC)
  - 7 ingressi analogici
  - 10 bit di risoluzione (0-1023)
  - converte i segnali con ampiezza tra 0 e Aref (5 V se il piedino è scollegato)



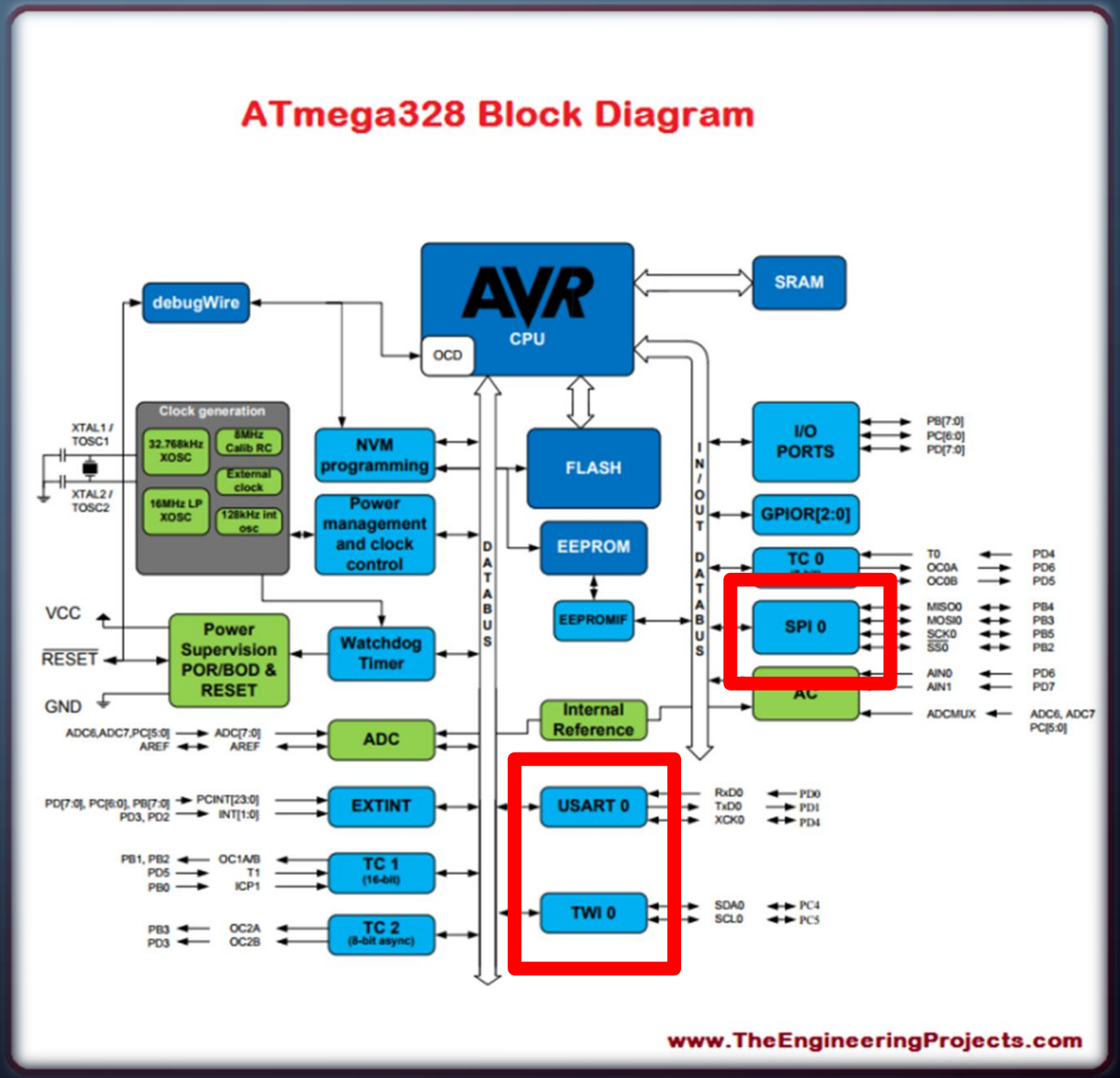
INTERRUPT ESTERNI

- Interrupt esterni
  - è possibile intervenire sul funzionamento del microprocessore tramite interrupt esterno (per applicazioni real-time)

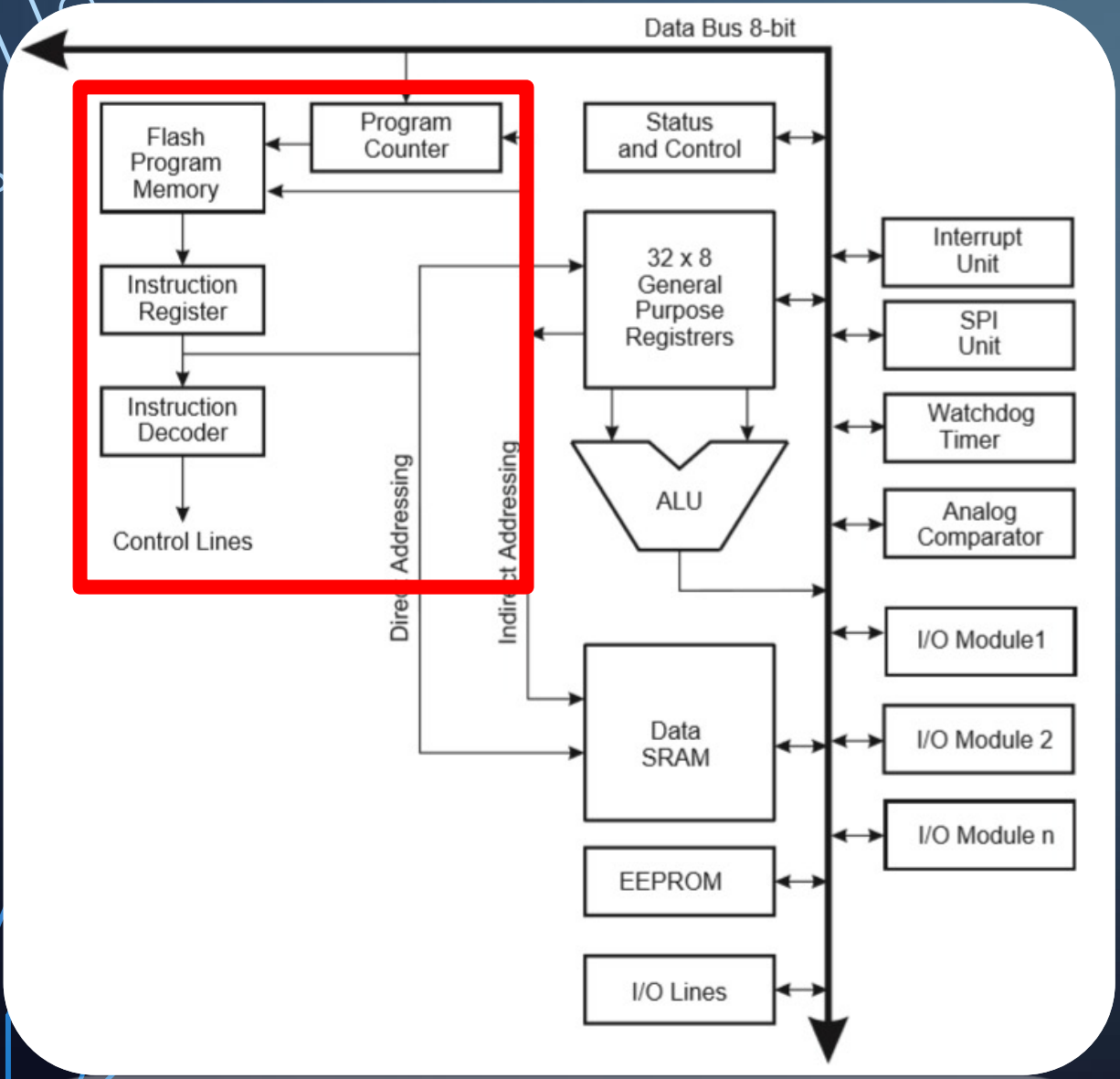


INTERFACCE DI COMUNICAZIONE

- Universal serial bus (USART)
  - sincrona
  - asincrona
- Bus seriale sincrono (SPI)
- 2-wire serial interface (TWI)

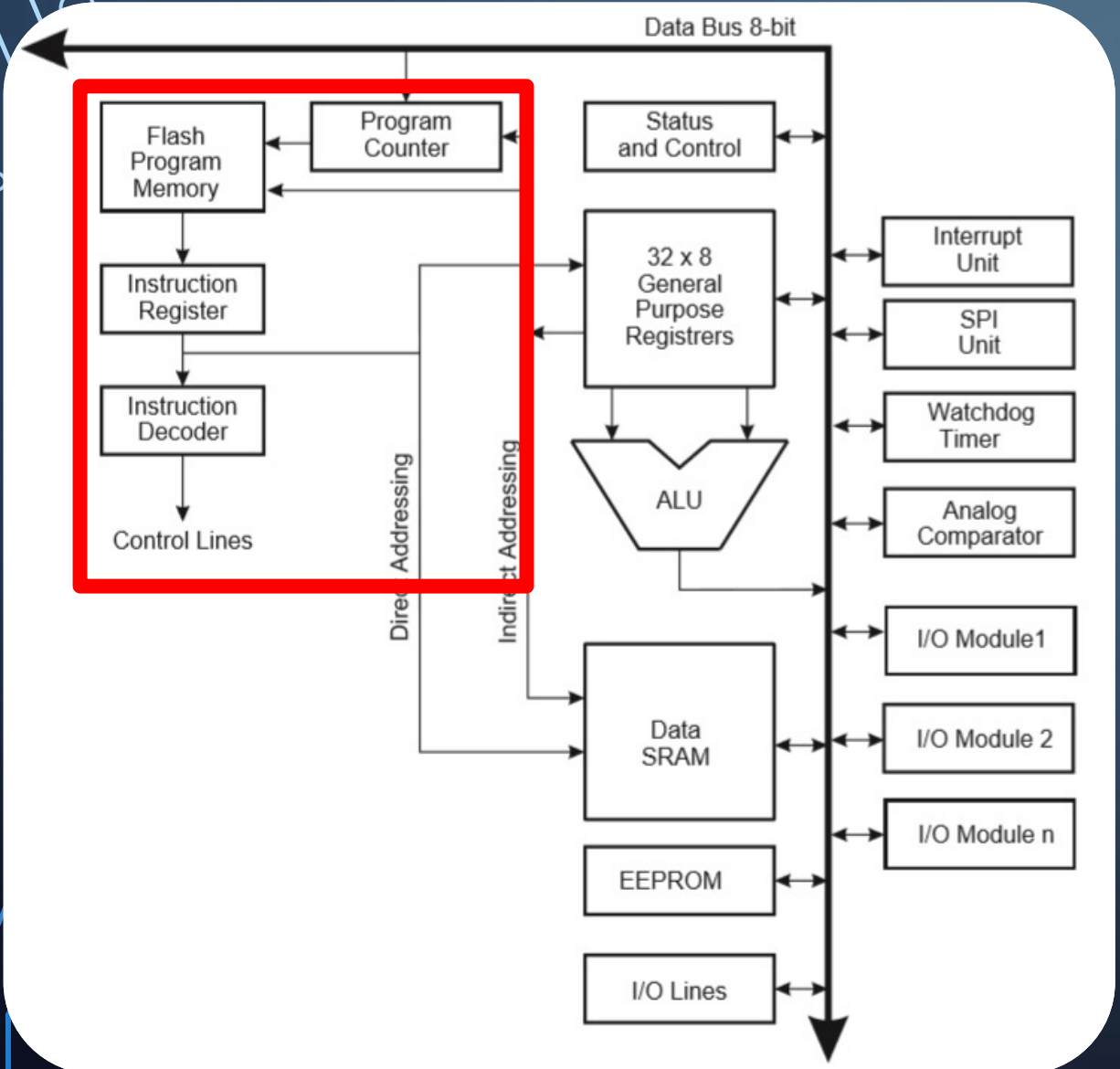


ESECUZIONE DELLE ISTRUZIONI



- In ogni CPU le istruzioni vengono eseguite in 3 passaggi:
  - Fetch
  - Decode
  - Excute
- Nella fase di fetch viene prelevata un'istruzione dalla memoria FLASH
- Nella fase di decode viene decodificata e quindi viene interpretato «cosa fare»
- Nella fase di execute l'istruzione viene eseguita coinvolgendo i dati relativi all'istruzione stessa (che possono essere in un registro o in memoria FLASH o nella RAM) e i vari componenti della CPU

ESECUZIONE DELLE ISTRUZIONI



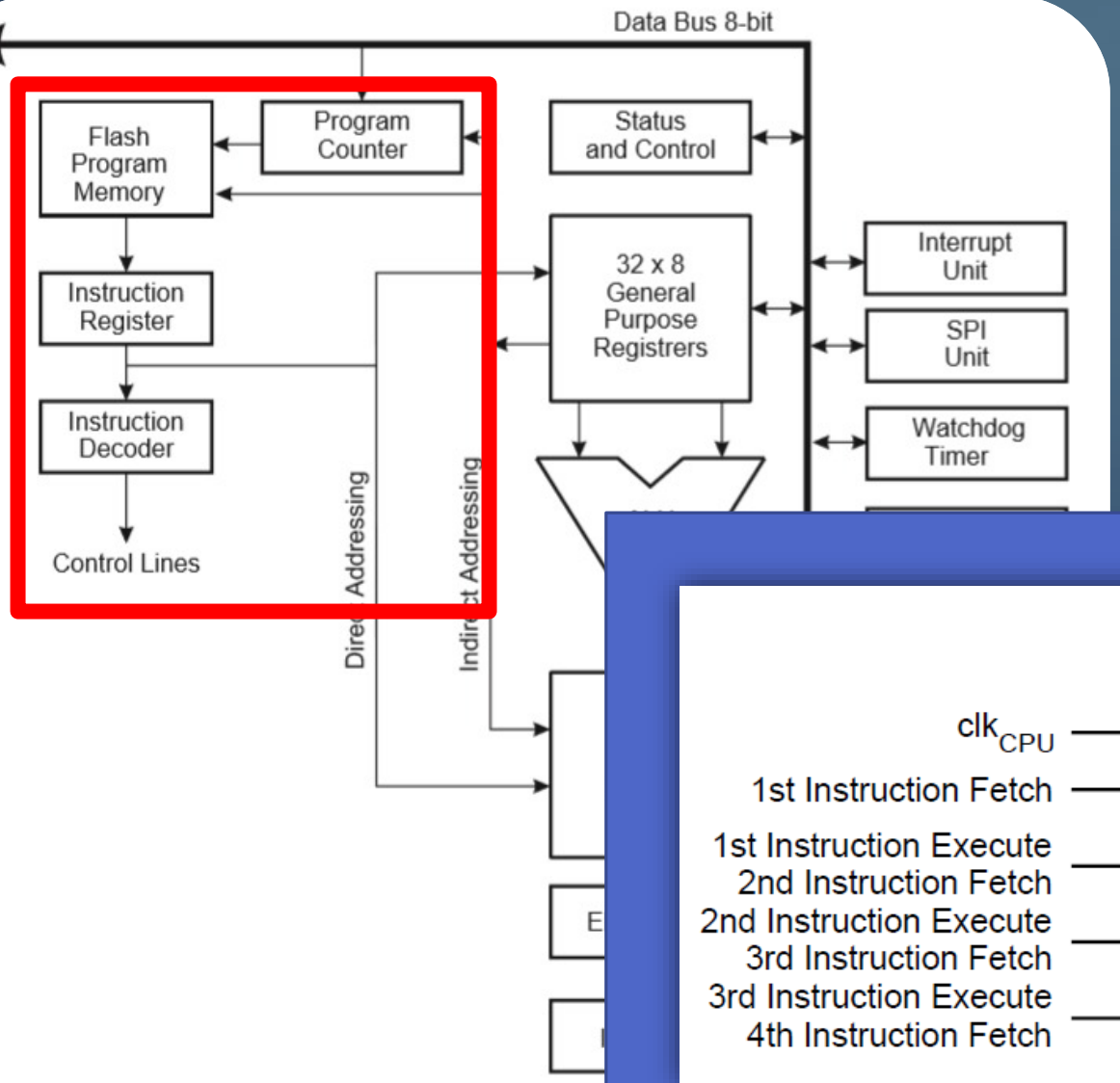
- L'istruzione che deve essere eseguita è indicata dal PROGRAM COUNTER
  - Durante ogni fase di fetch il PROGRAM COUNTER viene incrementato di 1 in modo da puntare alla successiva istruzione da eseguire
  - L'effettivo caricamento dell'istruzione successiva avviene però dopo la fase di execute

Location	Instruction
0	LOAD R0 1
1	ADD R0 1
2	WRITE 0 R0
3	READ R1 0
4	LOAD R2 4
5	COMPARE R0 R2
6	JUMPLT 2
7	EXIT
8	LOAD R1 4

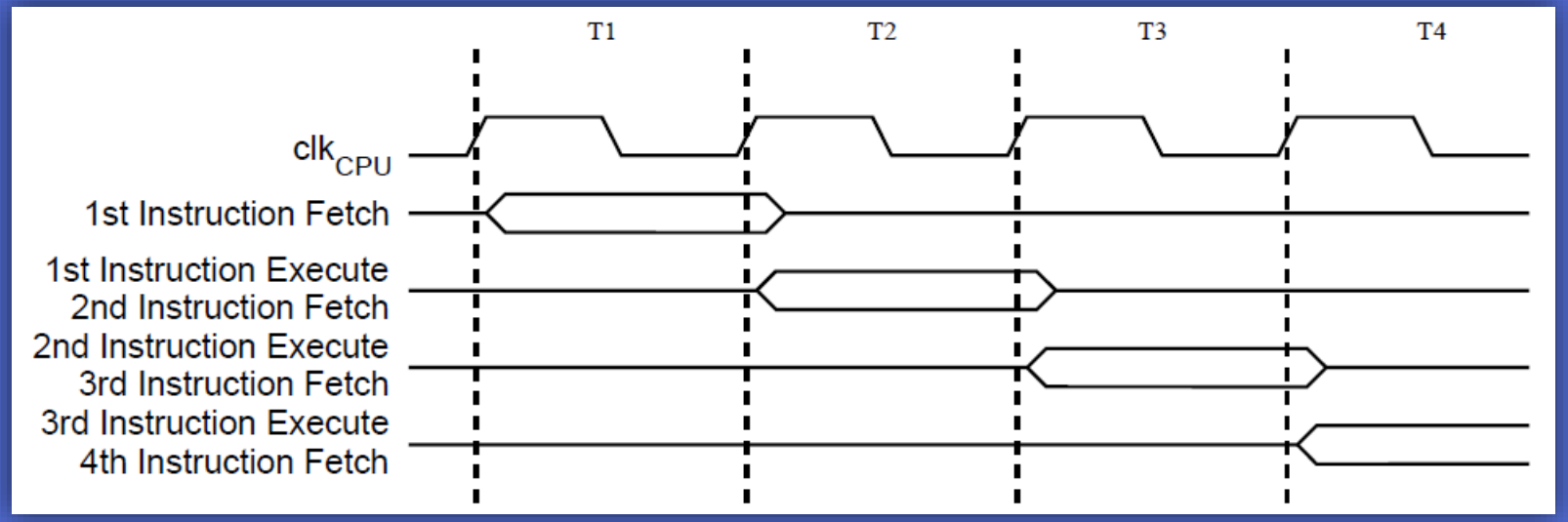
Program Counter	3

Valore del Program Counter all'INIZIO della fase di fetch dell'istruzione READ R1 0

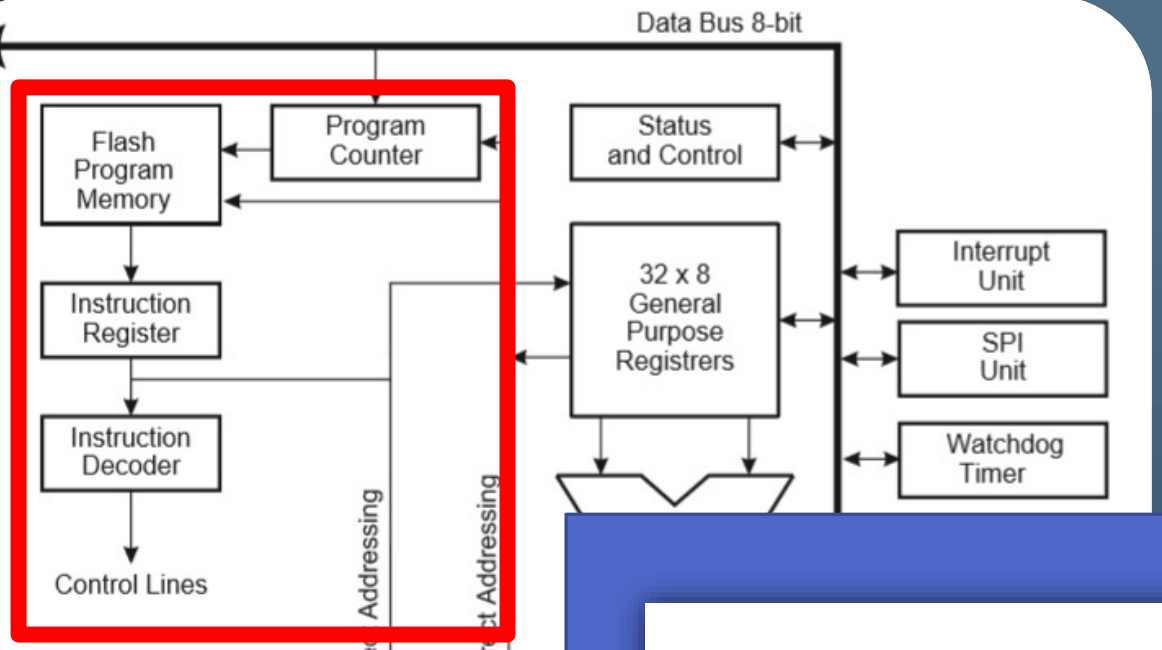
ESECUZIONE DELLE ISTRUZIONI



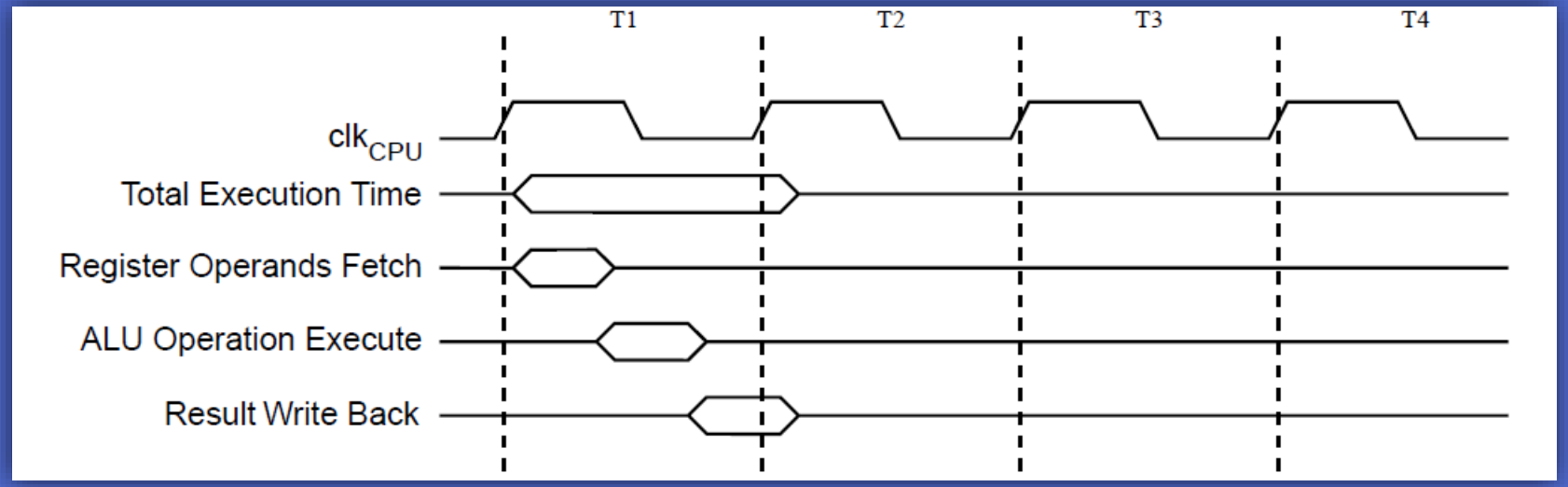
- La particolarità delle architetture Harvard con memoria programma e memoria dati separate è l'efficienza. Si ha infatti che la fase di execute di un'istruzione quella di fetch della successiva possono essere fatte insieme



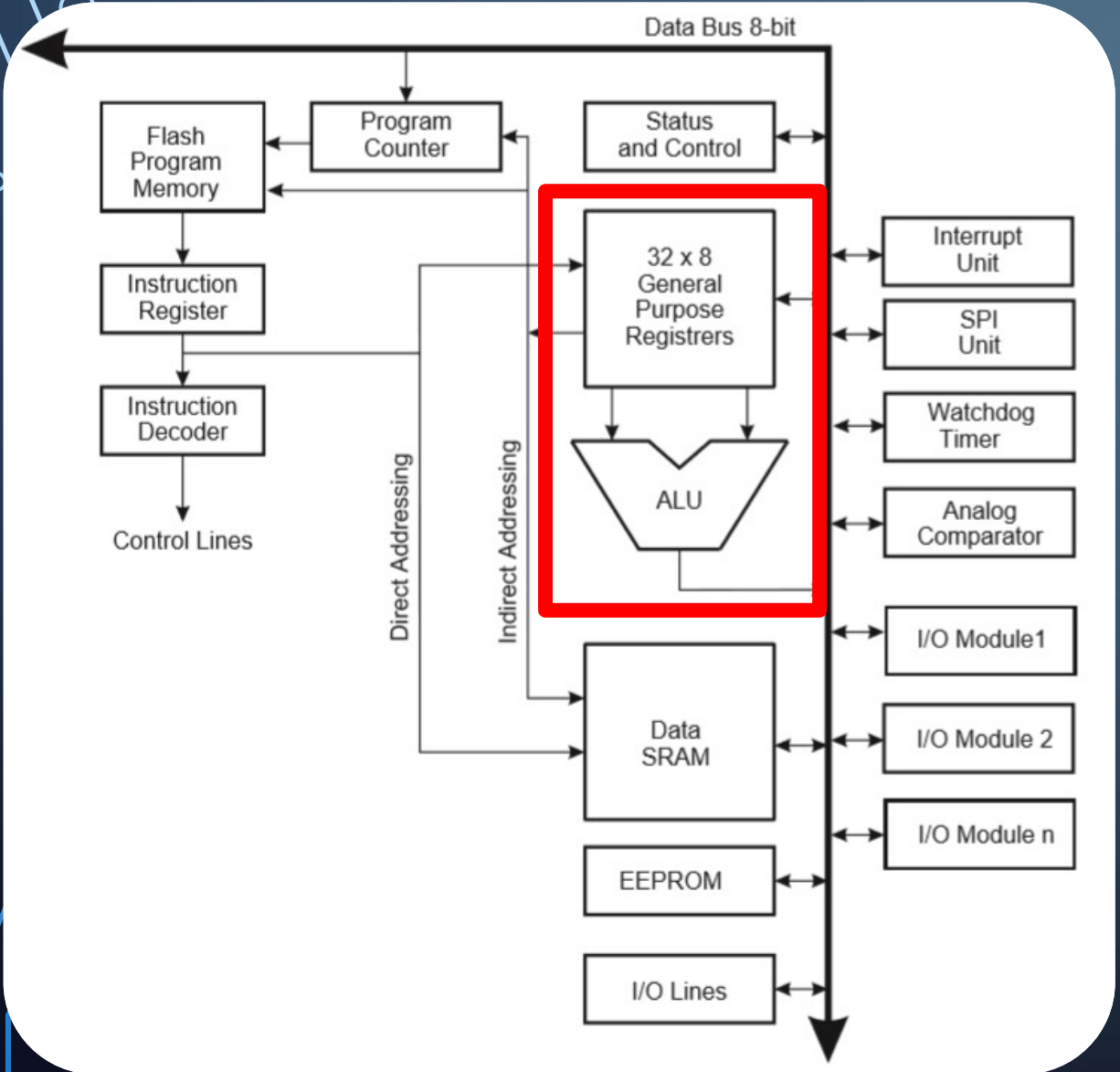
ESECUZIONE DELLE ISTRUZIONI



- In un singolo ciclo di clock si completa in genere anche la fase di decode ed execute delle operazioni che coinvolgono l'Unità Aritmetico Logica
  - Operazioni più complesse come quelle di salto impiegano invece più cicli di clock

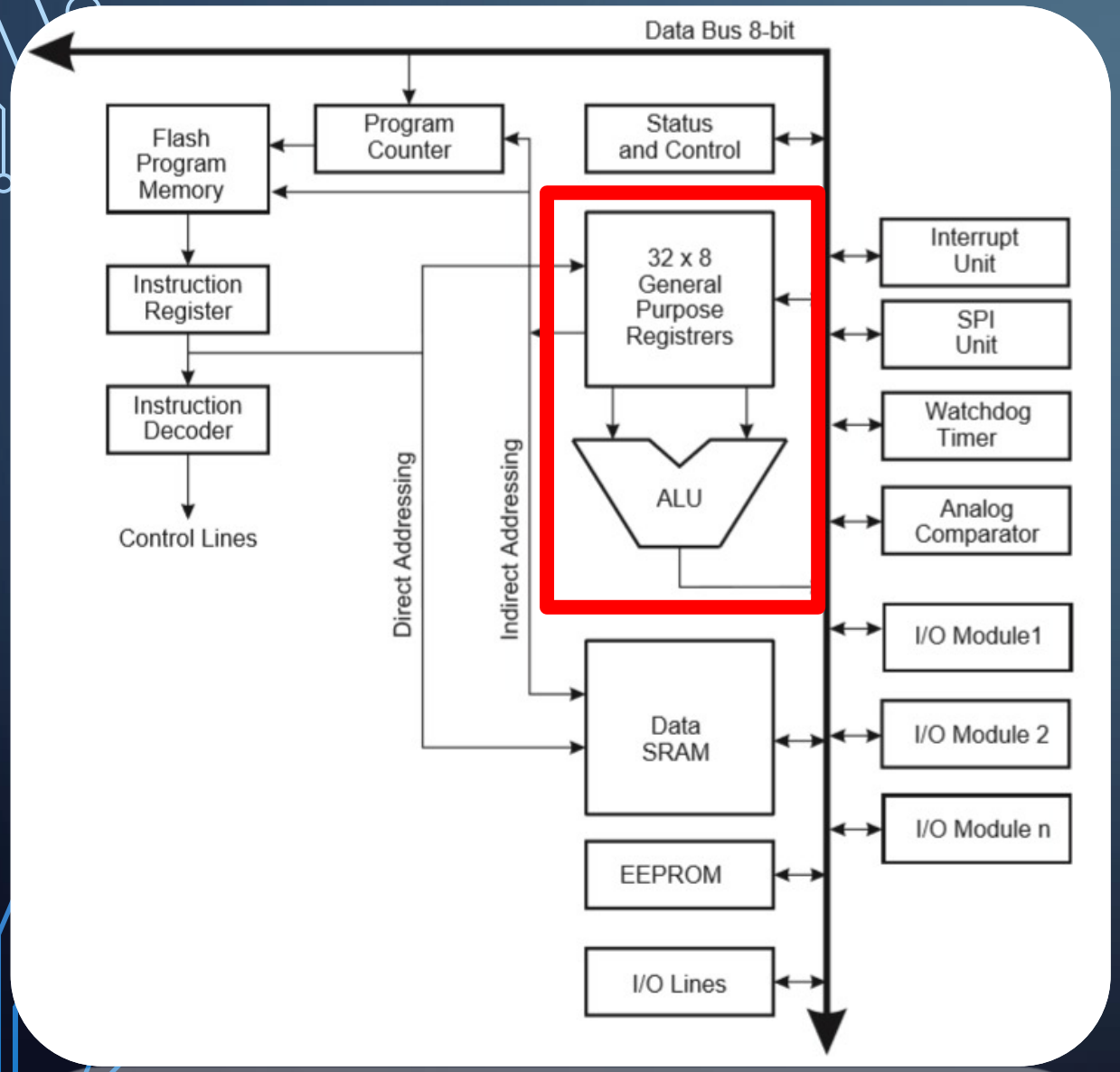


ALU E REGISTRI



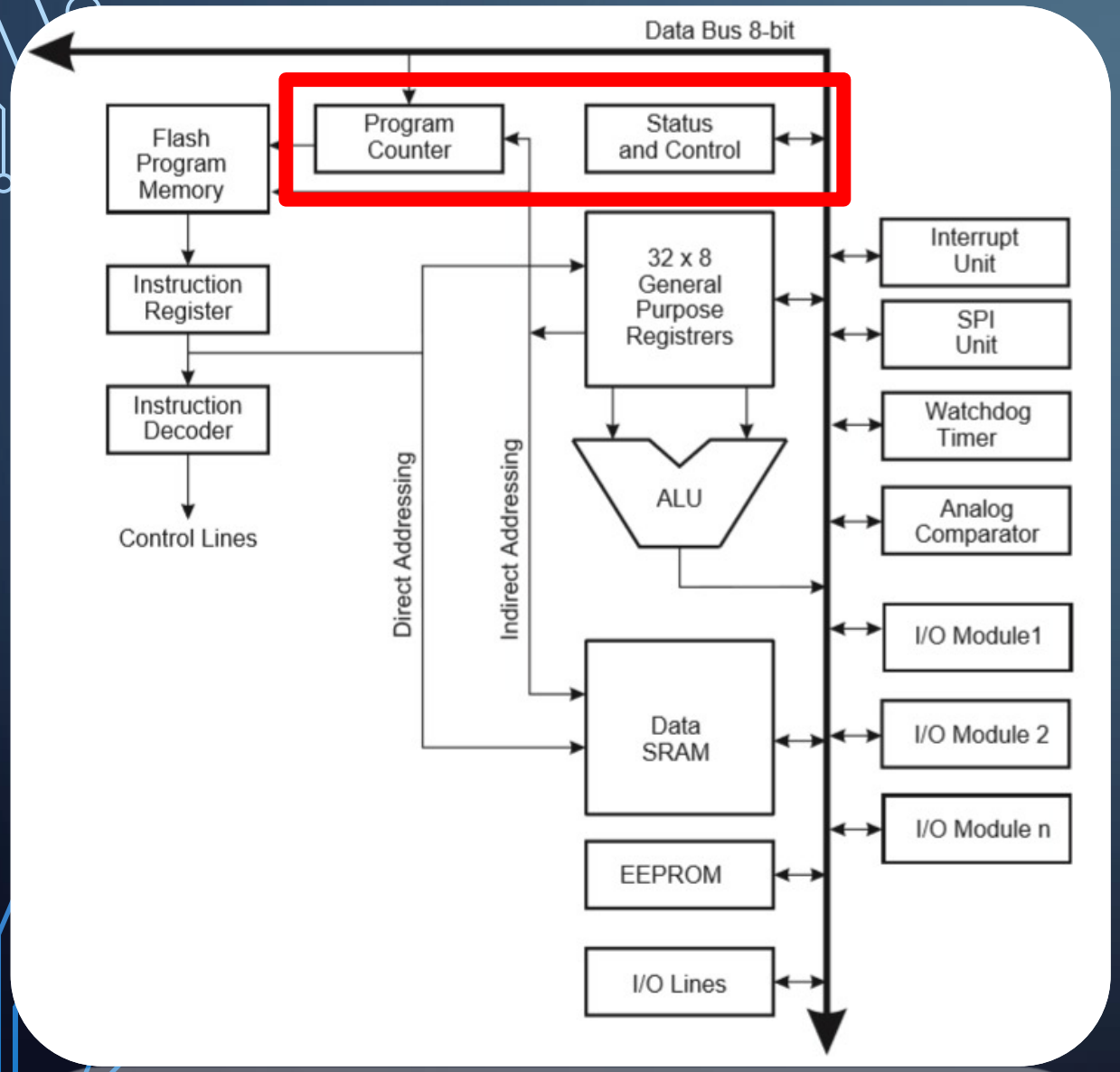
- Le istruzioni eseguite da un microprocessore sono di tipo
  - aritmetico (per esempio somma e moltiplicazione)
  - logico (per esempio AND, OR, XOR, NAND, NOR)
  - di confronto (per esempio maggiore, minore, uguale, diverso, maggiore-uguale, minore-uguale)
  - di lettura e scrittura di valori in memoria RAM
  - di I/O (input/output) di valori nelle periferiche
  - di salto in seguito all'occorrenza di un evento
- Le prime 3 tipologie di istruzione fanno uso dell'Unità Aritmetico Logica (Arithmetic Logic Unit o ALU) che prende in ingresso 1 o 2 operandi dai REGISTRI, svolge le operazioni necessarie e scrive i risultati in un REGISTRO

ALU E REGISTRI



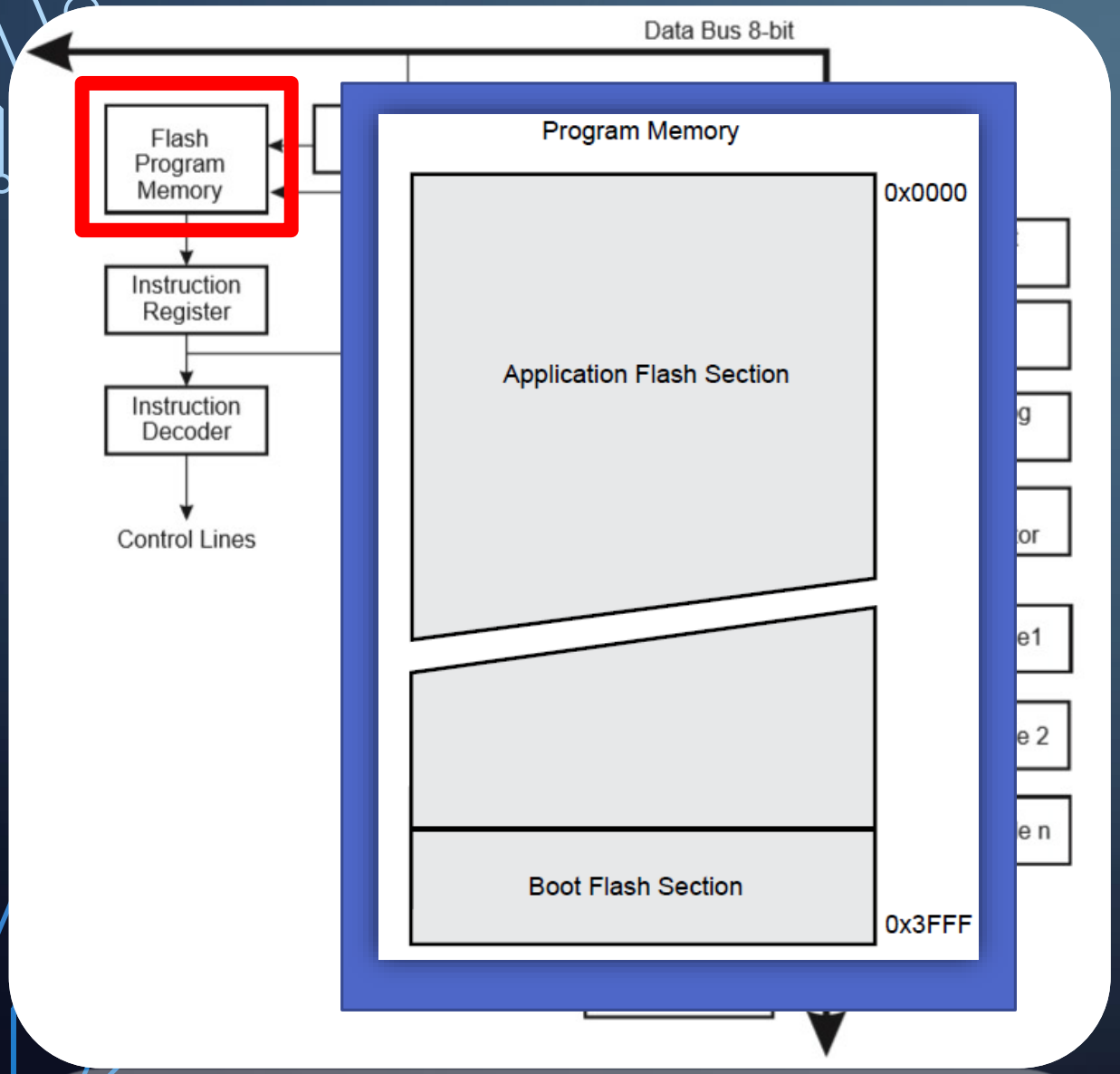
- L'ATmega328 dispone di 32 REGISTRI a 8 bit di USO GENERALE
  - un registro è una locazione di memoria ad alta velocità interna alla CPU
  - i dati per poter essere elaborati dall'ALU devono necessariamente essere in un registro
  - i risultati delle elaborazioni sono sempre salvate in un registro
  - se un valore non risiede in un registro (per esempio è in memoria, in un timer, in una periferica di I/O) deve essere prima caricato in un registro per poter essere elaborato tramite un'istruzione

REGISTRI SPECIALI



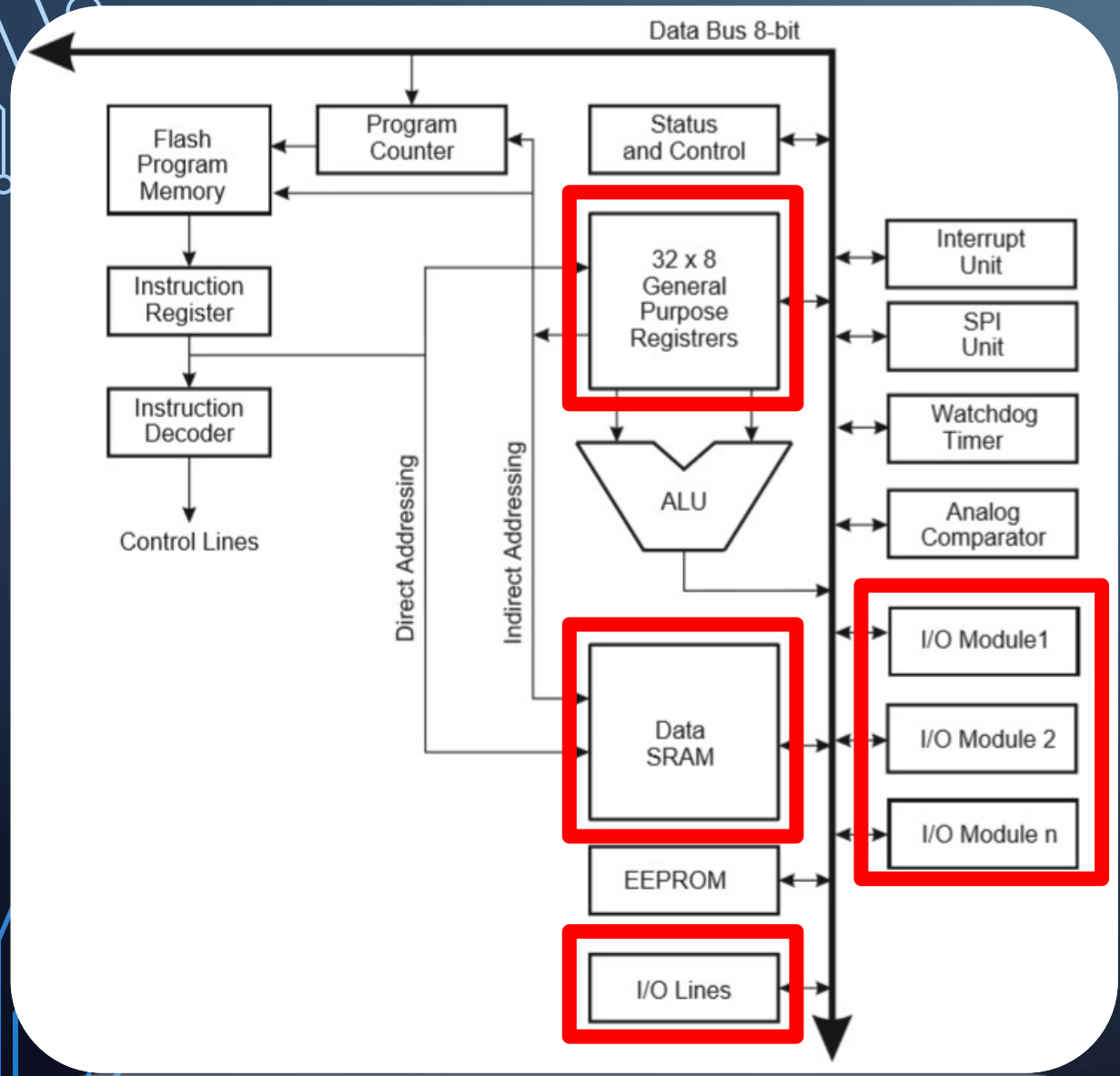
- L'ATmega328 dispone anche di REGISTRI SPECIALI necessari al suo funzionamento
  - il PROGRAM COUNTER (PC) è un registro speciale
- altri registri speciali sono
  - lo STATUS REGISTER (SR) che viene aggiornato alla fine di ogni operazione
  - lo STACK POINTER (SP) che consente di realizzare il meccanismo di chiamata alle subroutine

MEMORIA FLASH

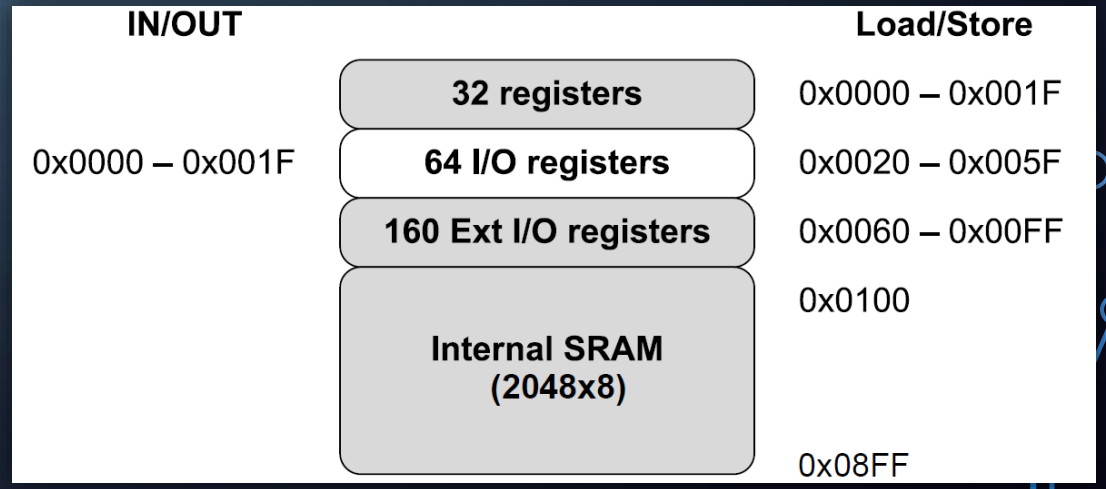


- Le istruzioni dell'ATmega328 sono a 16 o 32 bit, quindi la memoria flash è organizzata in banchi da 16 bit
- Il numero dei banchi è 16K (16000 banchi)
- Per motivi di sicurezza del software la memoria flash è divisa in due sezioni: la sezione destinata al **bootloader** e la sezione destinata alla **memoria programma** nel dispositivo
  - In fase di avvio del dispositivo il bootloader predispone il microcontrollore al funzionamento e carica nel program counter l'indirizzo della prima istruzione del programma caricato nella memoria programma
- Il program counter (PC) dell'ATmega328 è largo 14 bit, in modo da indirizzare le 16K locazioni di memoria programma

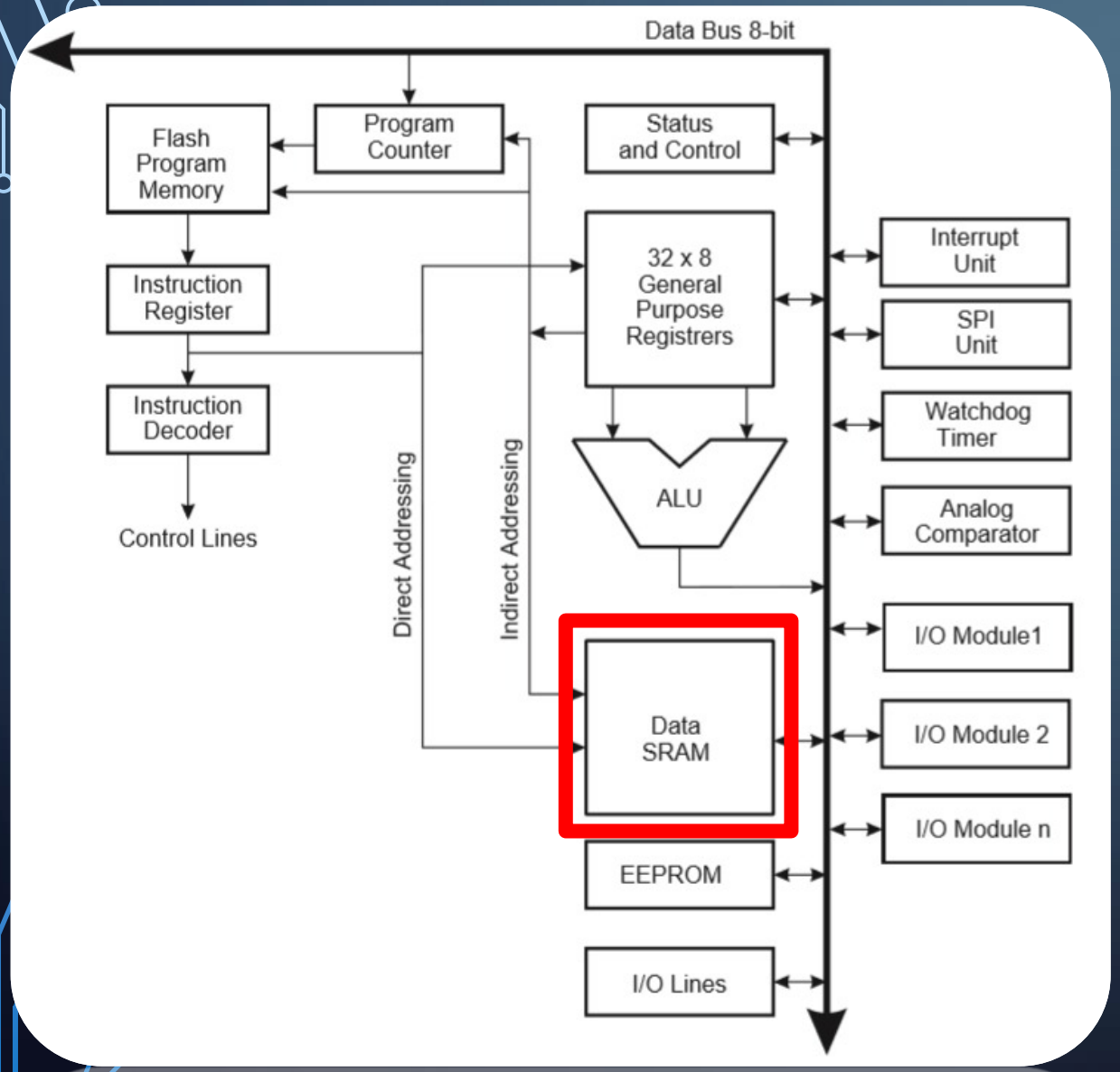
MEMORIA RAM



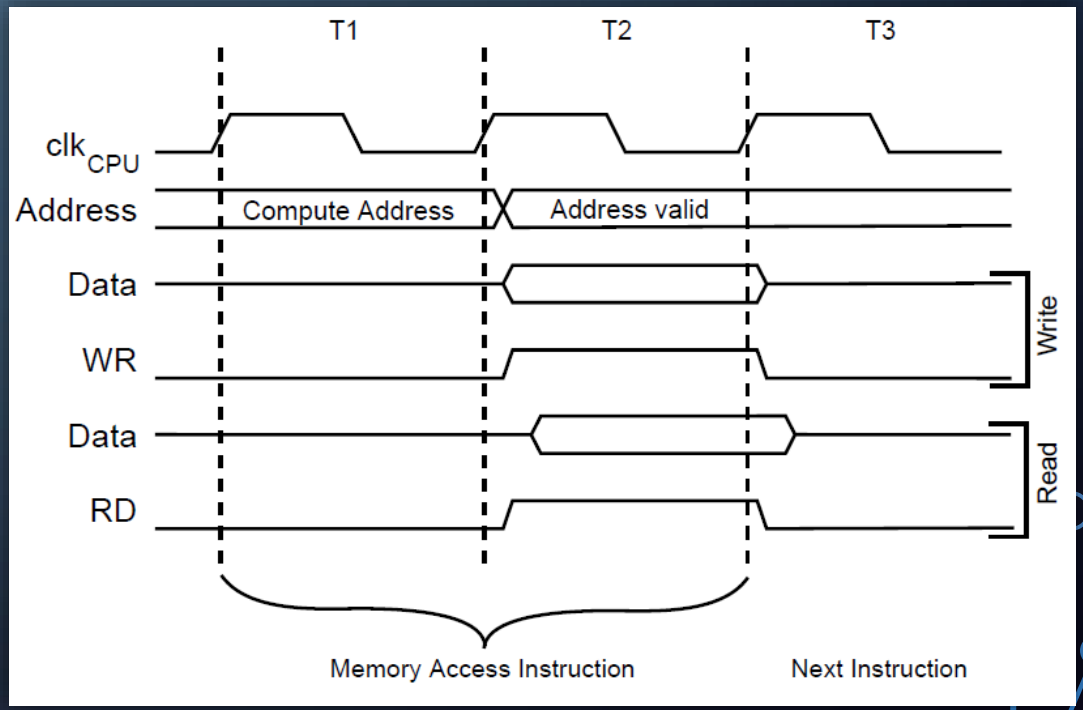
- La memoria RAM è organizzata in blocchi con indirizzi successivi
  - nel primo blocco è possibile trovare i 32 registri di uso generico
  - nel secondo blocco sono mappati 64 registri di I/O (non tutti sono utilizzati)
  - nel terzo blocco ci sono 160 ulteriori registri di I/O
  - nel quarto blocco sono mappati i 2048 byte di memoria RAM statica (SRAM)



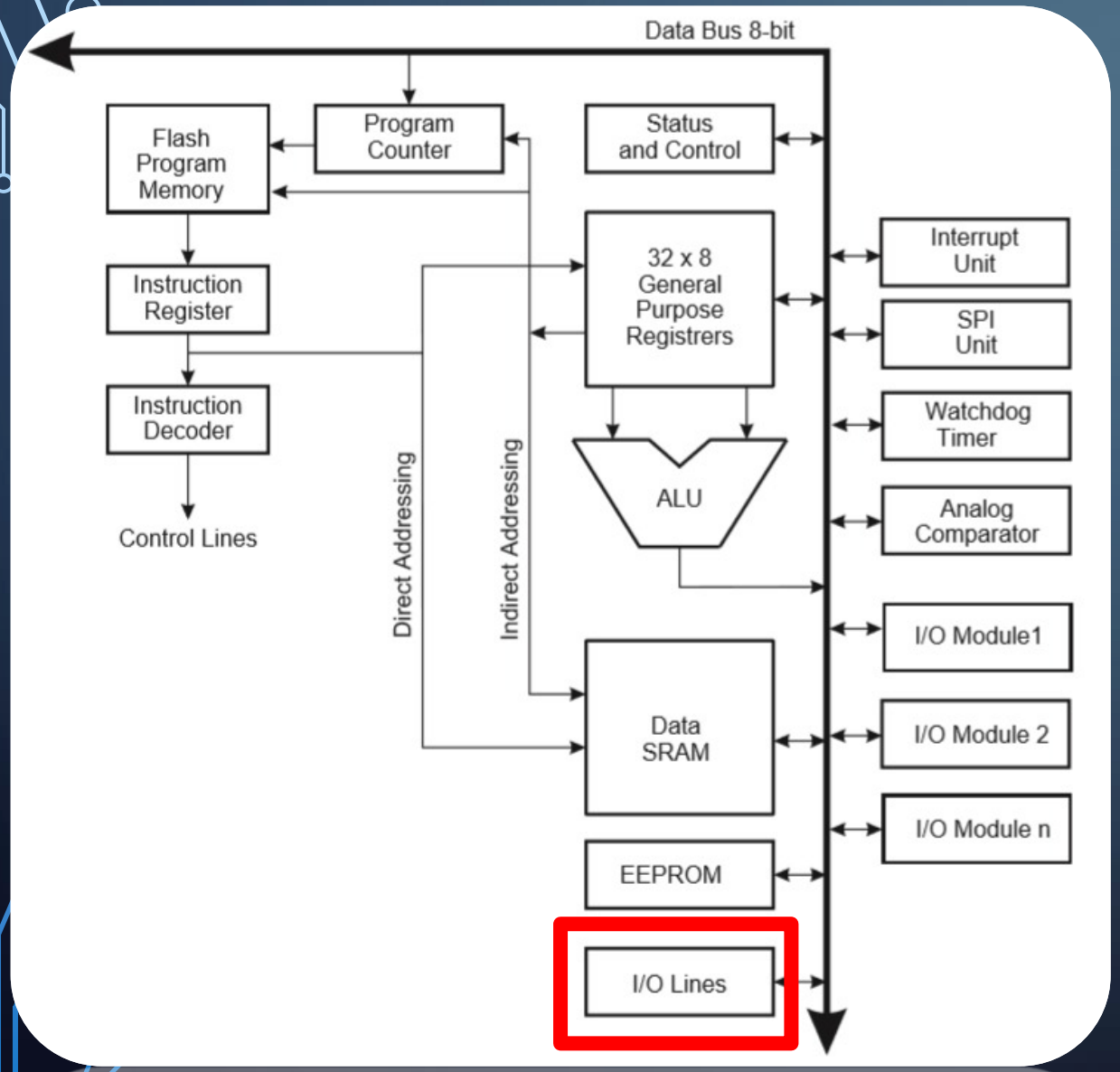
ACCESSO ALLA MEMORIA SRAM



- Mentre l'accesso ai registri è fatto in un solo ciclo di clock, l'accesso alla SRAM avviene con 2 cicli di clock



INGRESSI E USCITE DIGITALI (I/O)

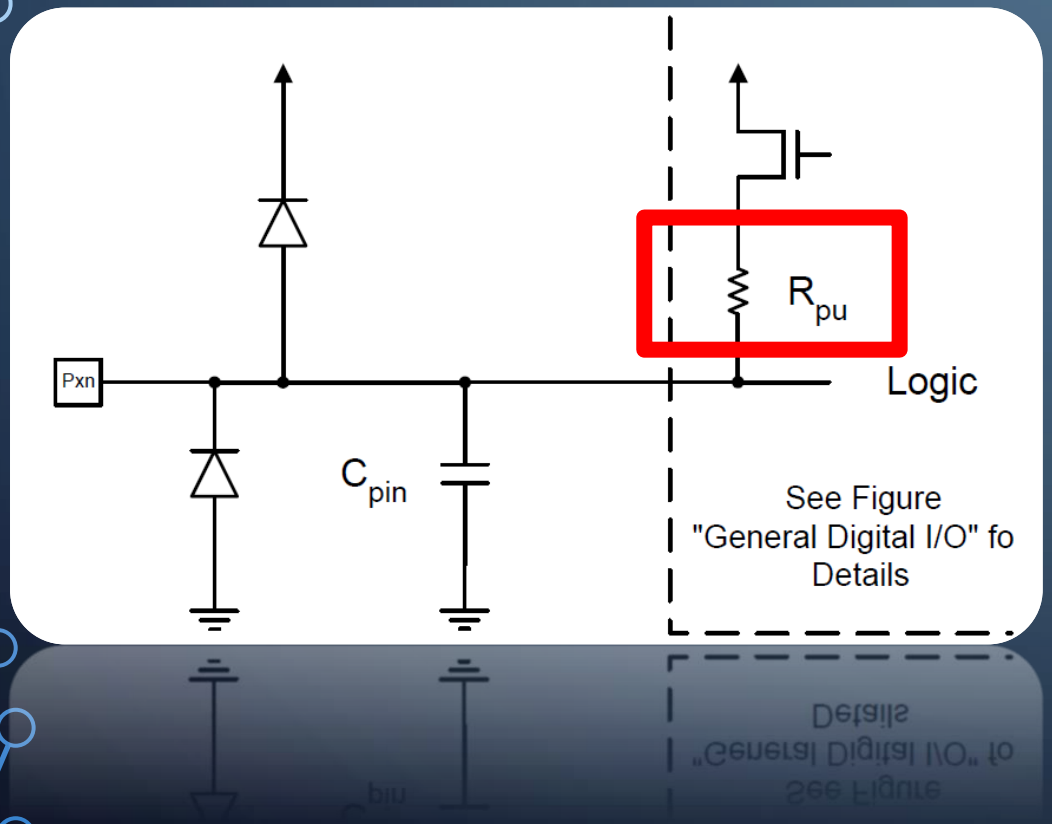


- Gli I/O digitali sono organizzati in gruppi chiamati PORT (porte) da 8 bit
  - le porte sono PORTA PORTB PORTC e PORTD
- Ogni bit corrisponde ad un I/O
  - a seconda del package poi gli I/O possono essere o non essere disponibili in un piedino esterno

In fase di setup del dispositivo ad ogni piedino di I/O deve essere assegnato uno dei due stati

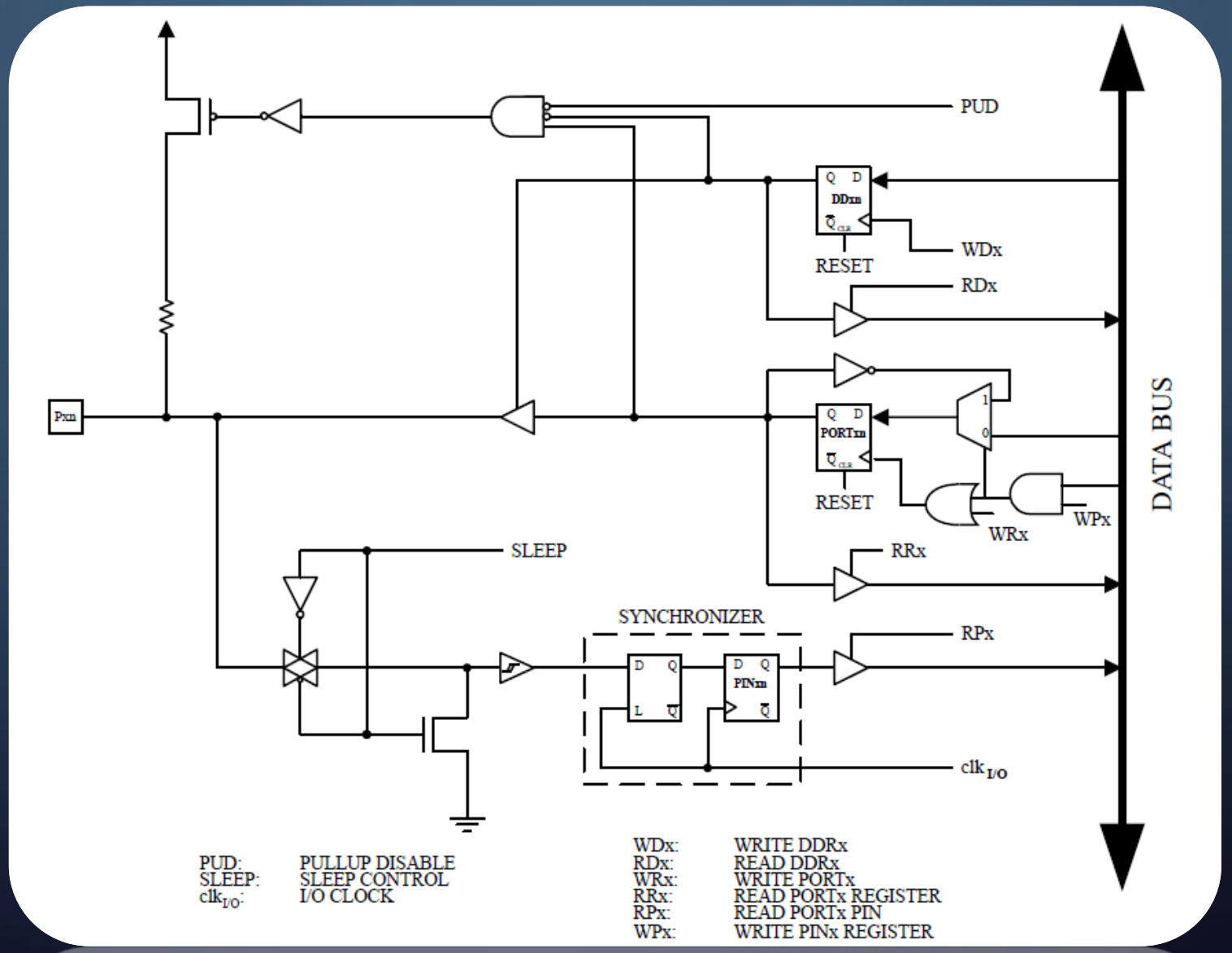
- ingresso
- uscita

INGRESSI E USCITE DIGITALI (I/O)

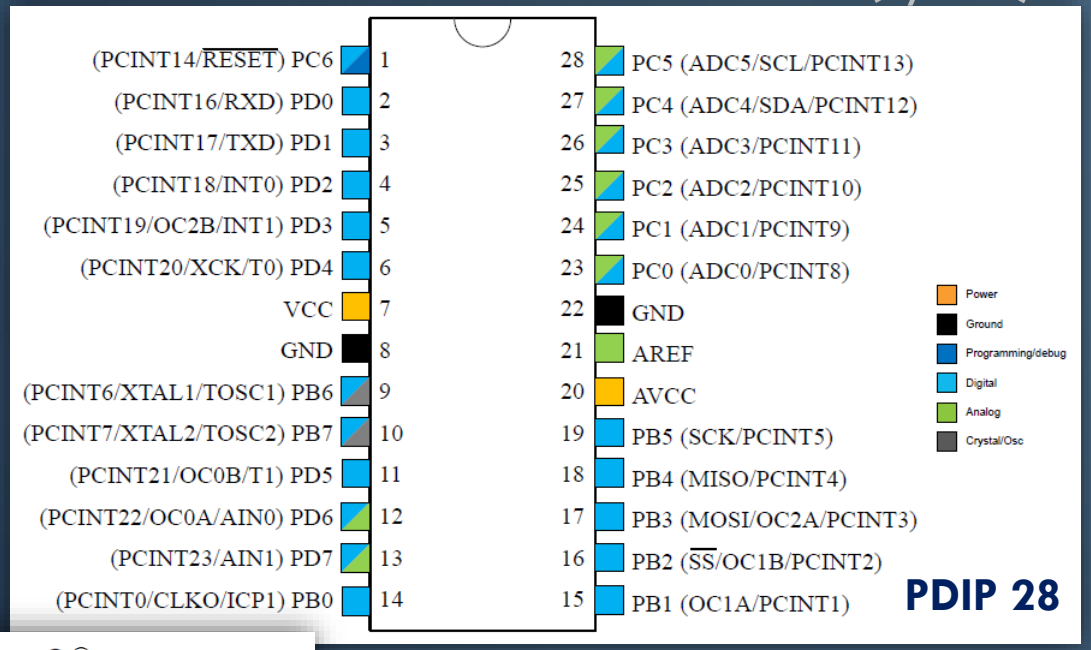


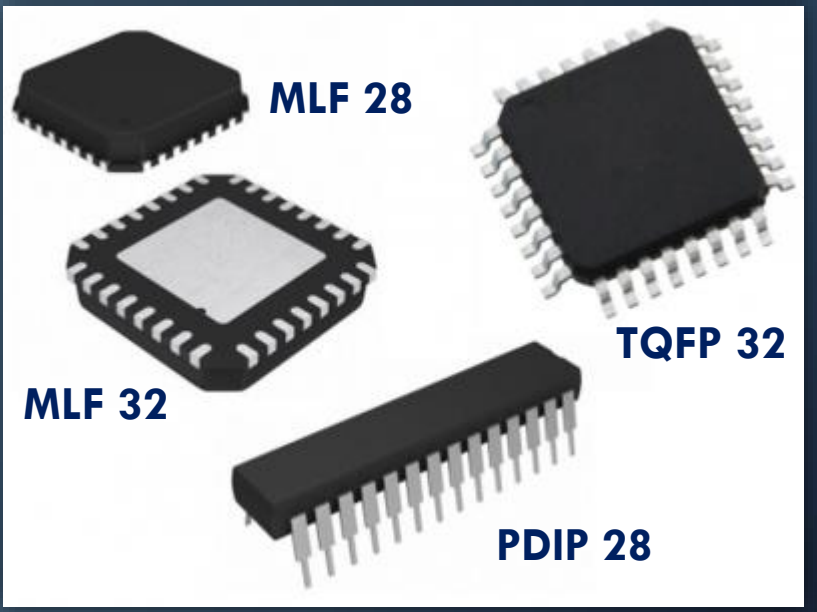
- Se gli I/O sono fisicamente collegati ai piedini di uscita lo schema elettrico equivalente è riportato a lato
- La massima corrente erogabile da ogni pin è di 40 mA
  - valori superiori possono danneggiare irreparabilmente il microcontrollore
- Vi è la possibilità di attivare via software un resistore di pull-up che mantiene il livello logico alto nel caso il piedino sia di input

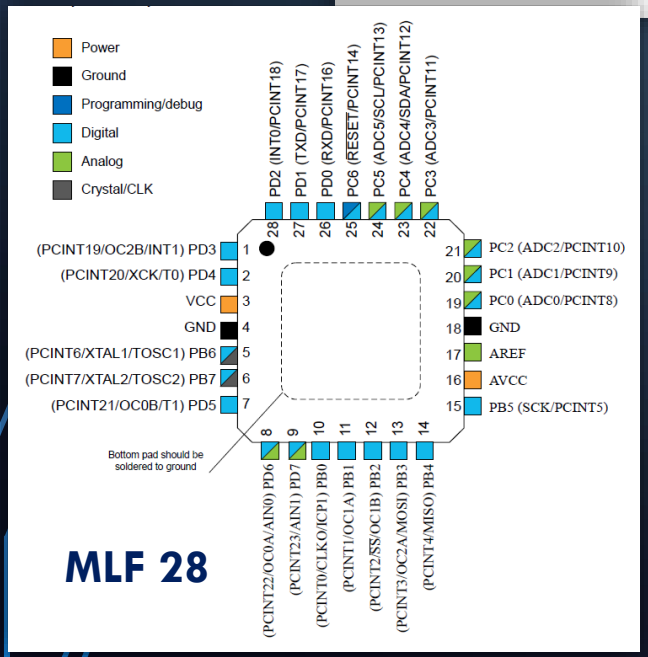
SCHEMA ELETTRICO DI UN I/O



INGRESSI E USCITE DIGITALI (I/O)

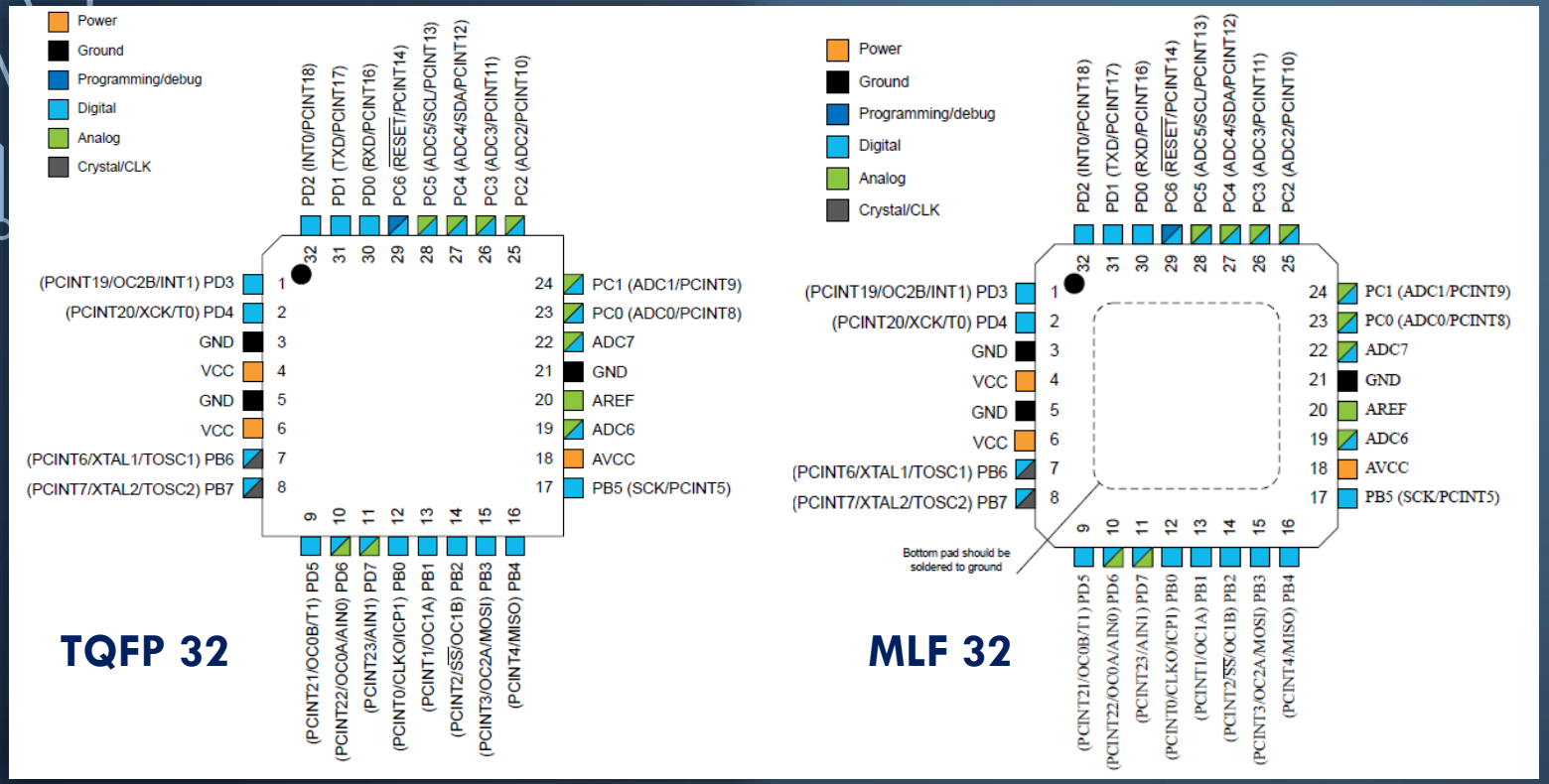


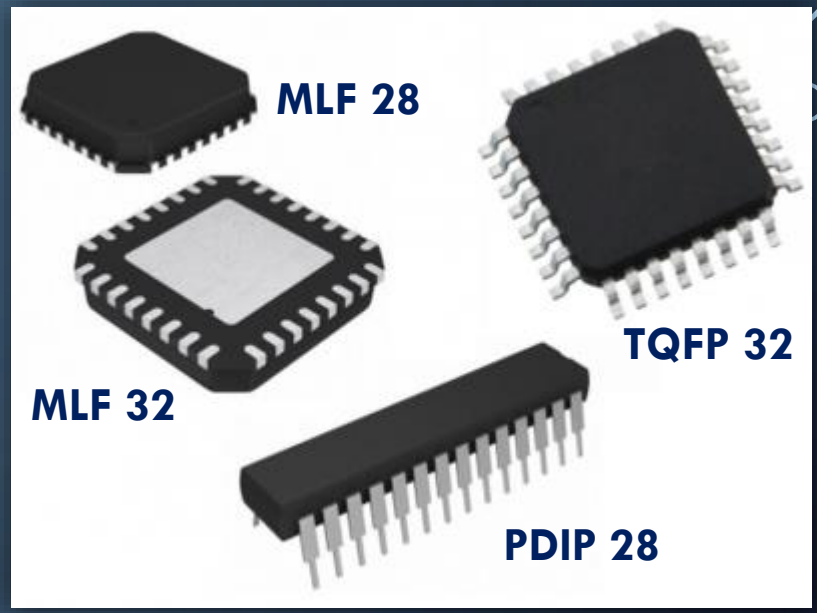




- Nel caso il package sia a 28 pin (PDIP 28 o MLF 28) gli I/O disponibili sono:
  - PORT D [0:7] (nel PDIP 28 pin 2-6,11-13)
  - PORT B [0:5] (nel PDIP 28 pin 14-19)
  - PORT C [0:5] (nel PDIP 28 pin 1, 23-28)
- I pin non sono contigui
- Cambiando il package la disposizione dei piedini cambia

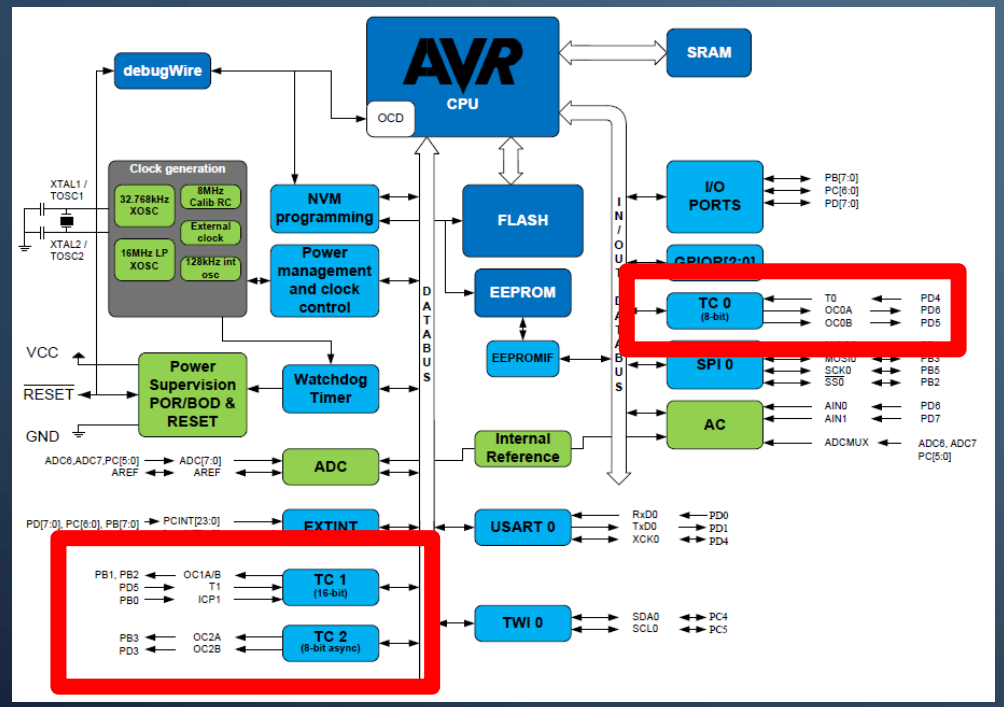
INGRESSI E USCITE DIGITALI (I/O)





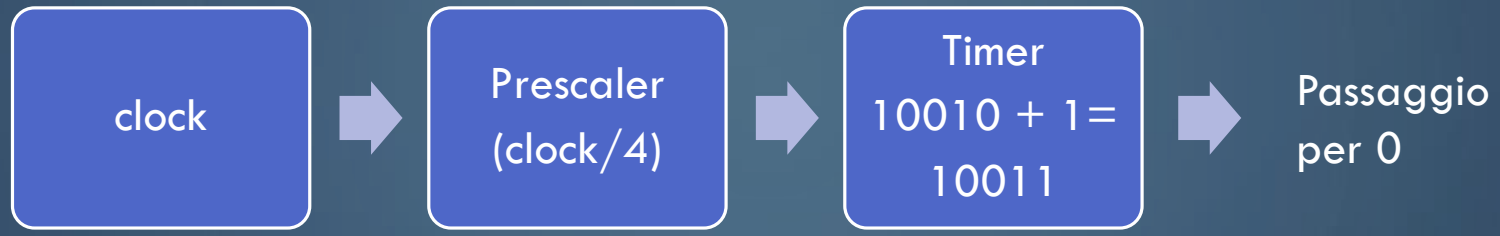
- Nel caso il package sia a 32 pin (TQFP 32 o MLF 32) gli I/O disponibili sono gli stessi
- La disposizione passando tra i 2 package a 32 pin resta la stessa

I TIMER/COUNTER



- L'ATmega328 dispone di 3 timer/counter
  - Timer 0: TC 0 a 8 bit
    - viene utilizzato da Arduino per le funzioni di timing come delay o millis
    - quando si usa l'ambiente Arduino è bene non utilizzare il Timer 0
  - Timer 1: TC 1 a 16 bit
    - può contare fino ad un massimo di 65535
    - la libreria servo di Arduino usa questo timer
  - Timer 2: TC 2 a 8 bit
    - viene utilizzato dal comando tone

I TIMER/COUNTER



- Un timer funziona incrementando una variabile di conteggio contenuta in un registro di conteggio
- Il registro può contare fino ad un certo valore in base alla sua dimensione, dopo di che viene resettato a zero e ricomincia il conteggio
- Passando per lo zero il timer di solito setta un bit di flag per far capire che è avvenuto un overflow. Questo flag può essere controllato manualmente oppure può essere innescata una interruzione appena il flag è settato
    - come ogni altra interruzione è possibile impostare la Interrupt Service Routine (ISR) per avviare il codice voluto quando il flag è stato settato. La ISR resetta il flag automaticamente
- Per incrementare il contatore c'è un **segnale di clock**
    - il segnale di clock può essere quello di sistema opportunamente scalato con un prescaler oppure può essere prelevato dall'esterno

I TIMER/COUNTER

- Un counter si differenzia da un timer per il fatto che in genere conta all'indietro
- Il registro del counter viene preventivamente caricato con un valore
- Ad ogni impulso in arrivo tipicamente da un piedino esterno il valore del registro viene decrementato
  - non si usa un prescaler
- Quando arriva a zero il counter tipicamente setta un bit di flag
  - nuovamente questo flag può essere controllato manualmente oppure può essere innescata una interruzione appena il flag è settato