Corso di Percezione Robotica

-

Prof.ssa Cecilia Laschi

10/04/2008

Introduzione ai Microcontroller

Dr. Virgilio Mattoli
(mattoli@crim.sssup.it)

Processori Embedded

✓ I microprocessori embedded sono contenuti in tutto ciò che ci circonda.

✓ I primi microprocessori sono apparsi negli anni ’70 → Intel 8080

✓ Oggi sono integrati praticamente in ogni apparecchio elettronico presente sule mercato: lavatrici, forni a microonde, telefonini, autovetture, …

✓ Ogni anno vengono venduti nel mondo **miliardi** di processori embedded

Processori Embedded

Mediamente un microprocessore per essere definito embedded deve avere le seguenti caratteristiche:

→Deve essere dedicato al controllo real-time di uno specifico dispositivo o funzione.

→ Deve contenere il proprio programma operativo in qualche tipo di memoria non volatile

→ Deve essere trasparente all'utente (deve funzionare come un hardware dedicato)

Processori Embedded

Un *sistema embedded* deve contenere solitamente le seguenti componenti:

- ✓ Un microprocessore
- ✓ Memoria RAM (random access memory)
- ✓ Memoria non-volatile : ROM, EEPROM,, FLASH, …
- ✓ I/O (interfaccia con l'ambiente)

In cosa un sistema embedded è diverso da un computer (PC)? **Risorse**

	PC	Embedded System
RAM	GB	Centinaia di GByte
ROM	Centinaia di GByte	KByte.

Processori Embedded

In cosa un sistema embedded è diverso da un computer (PC)? **Applicazione!**

→ Personal Computer devono poter svolgere una varietà virtualmente illimitata di funzioni e programmi e applicazioni;

→ I sistemi embedded devono svolgere un limitato numero di task (as. controllo della temperatura di un forno, controllo del tempo di lavaggio di una lavatrice, ….)

Processori Embedded

Perche usare un sistema a microprocessore embedded?

Costo. – Microprocessore embedded è molto vantaggioso rispetto all'implementazione hardware in componenti discreti

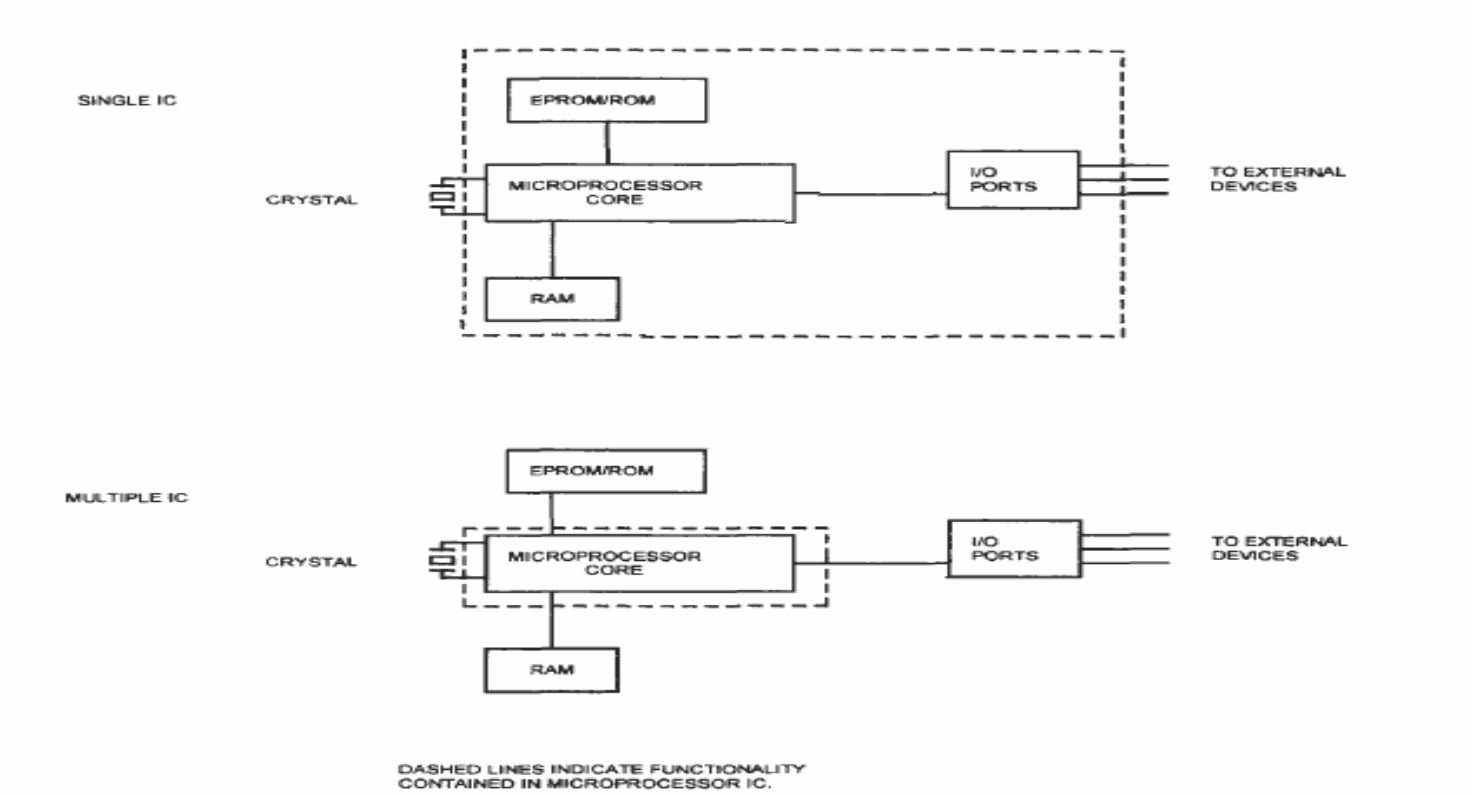
Programmabilità – La stressa piattaforma hardware permette di implementare differenti applicazioni

Flessibilità - Le funzionalità possono essere semplicemente ri-programmate in firmware

Adattabilità – Implementazione di sistemi intelligenti ("smart") con capacità di adattarsi all'ambiente

Cosa è un microcontroller?

Microcontroller = Microprocessore Embedded in un singolo chip



Un microcontroller è progettato per minimizzare il numero dei componenti richiesti per la realizzazione di un sistema embedded, incorporando memoria e I/O.

Spesso sono specializzati per una certa applicazione (a scapito della flessibilità).

Cosa è un microcontroller?

Dimensioni e Packaging

P: 40-pin PDIP
(52.27 x 15.24 x 3.81 mm)

PF: 64-pin TQFP
(14 x 14 x 1 mm)

PT: 64-pin TQFP
(10 x 10 x 1 mm)

SO: 18-pin SOIC
(11.53 x 10.34 x 2.31 mm)

P: 18-pin PDIP
(22.81 x 7.95 x 3.3 mm)

PF: 80-pin TQFP
(14 x 14 x 1 mm)

PF: 100-pin TQFP
(14 x 14 x 1 mm)

S: 20-pin SSOP
2 x 7.85 x 1.85 mm)

SO: 20-pin SOIC
(12.80 x 10.34 x 2.31 mm)

PT: 44-pin TQFP
(10 x 10 x 1 mm)

PT: 80-pin TQFP
(12 x 12 x 1 mm)

PT: 100-pin TQFP
(12 x 12 x 1 mm)

P: 20-pin PDIP
(26.24 x 7.87 x 3.3 mm)

ML: 28-pin QFN
(6 x 6 x 0.9 mm)

SO: 28-pin SOIC
(17.88 x 10.34 x 2.31 mm)

ML: 44-pin QFN
(8 x 8 x 0.9 mm)

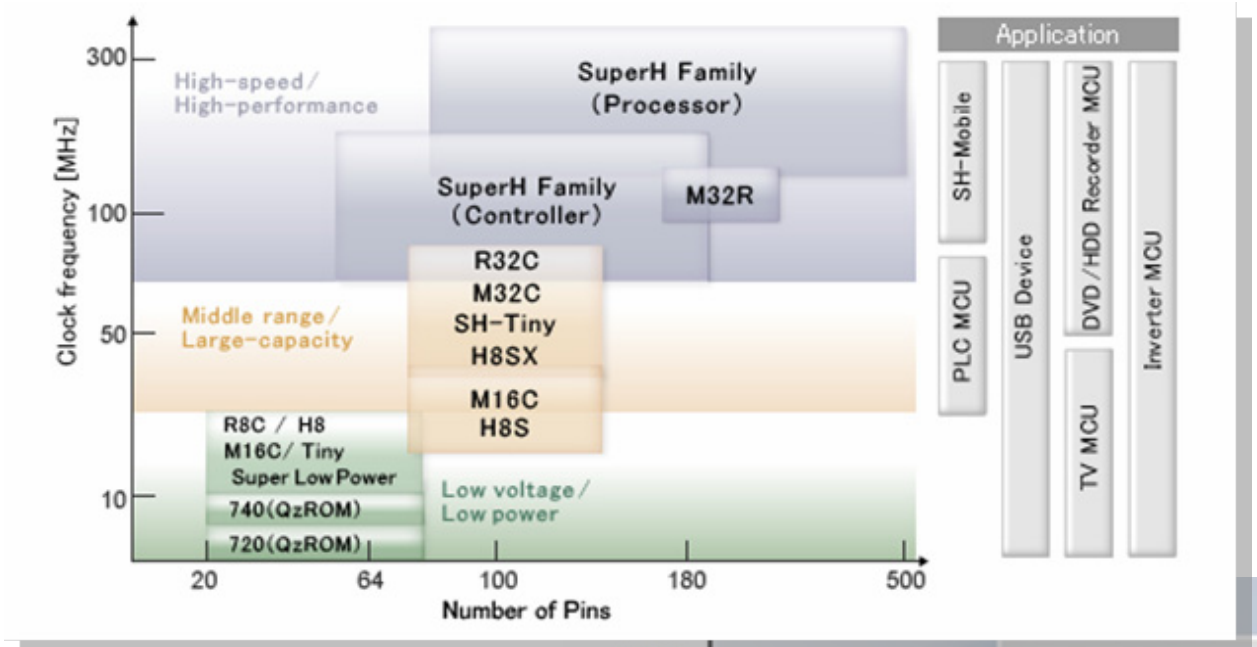
MM: 28-pin QFN
(6 x 6 x 0.9 mm)

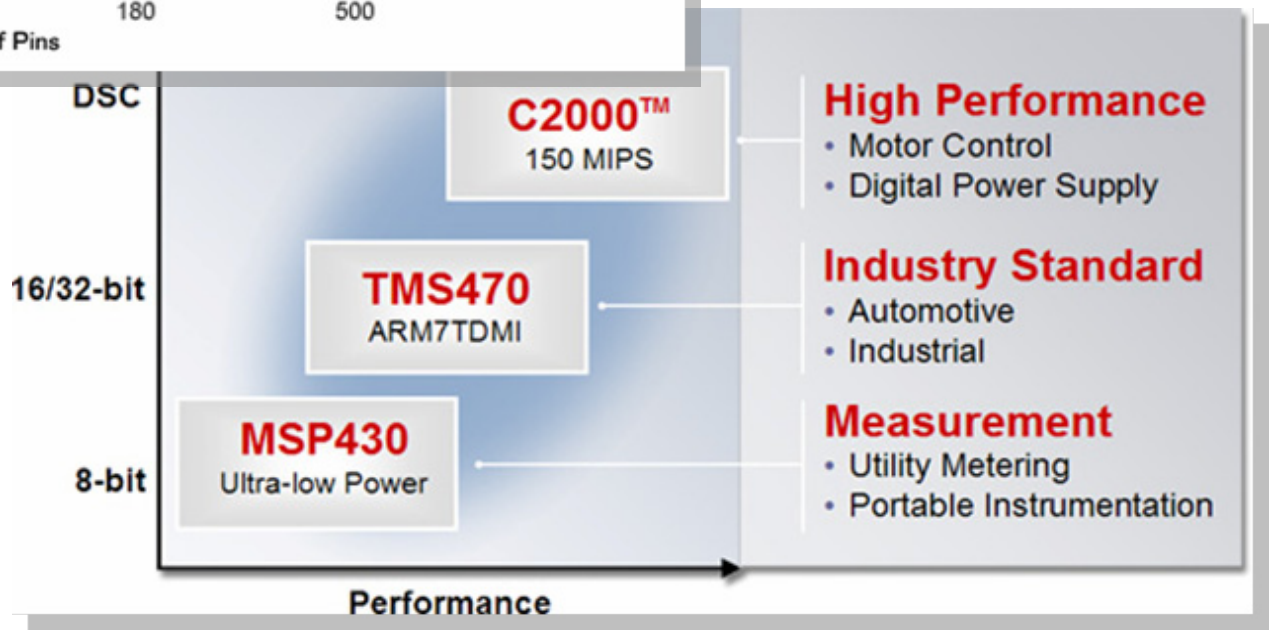
SP: 28-pin SPDIP
(34.67 x 7.87 x 3.3 mm)

SS: 28-pin SSOP

Cosa è un microcontroller?

8, 16 o 32 bit





Cosa è un microcontroller?

Principali Produttori

http://www.microchip.com

http://www.ti.com/

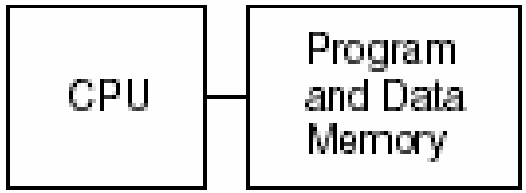
http://www.atmel.com/

http://eu.renesas.com/

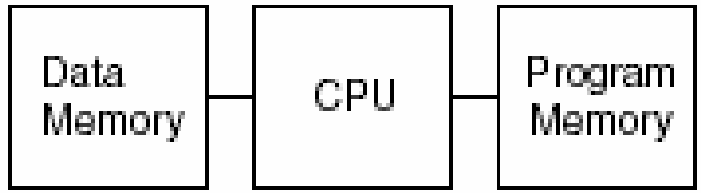
Architettura di un microcontroller

I sistemi a microprocessore hanno solitamente un'architettura di tipo von Neumann con una singola memoria per il programma e i dati che permette la massima flessibilità di allocazione; i microcontroller hanno invece tipicamente un'architettura di tipo Harvard in cui la meria di programma è separata da quella per i dati.

von Neumann architecture



Harvard architecture



Il vantaggio dell'architettura Harvard per applicazioni embedded è dovuta alla possibilità di usare due tipi diversi di memoria per i dati e il programma

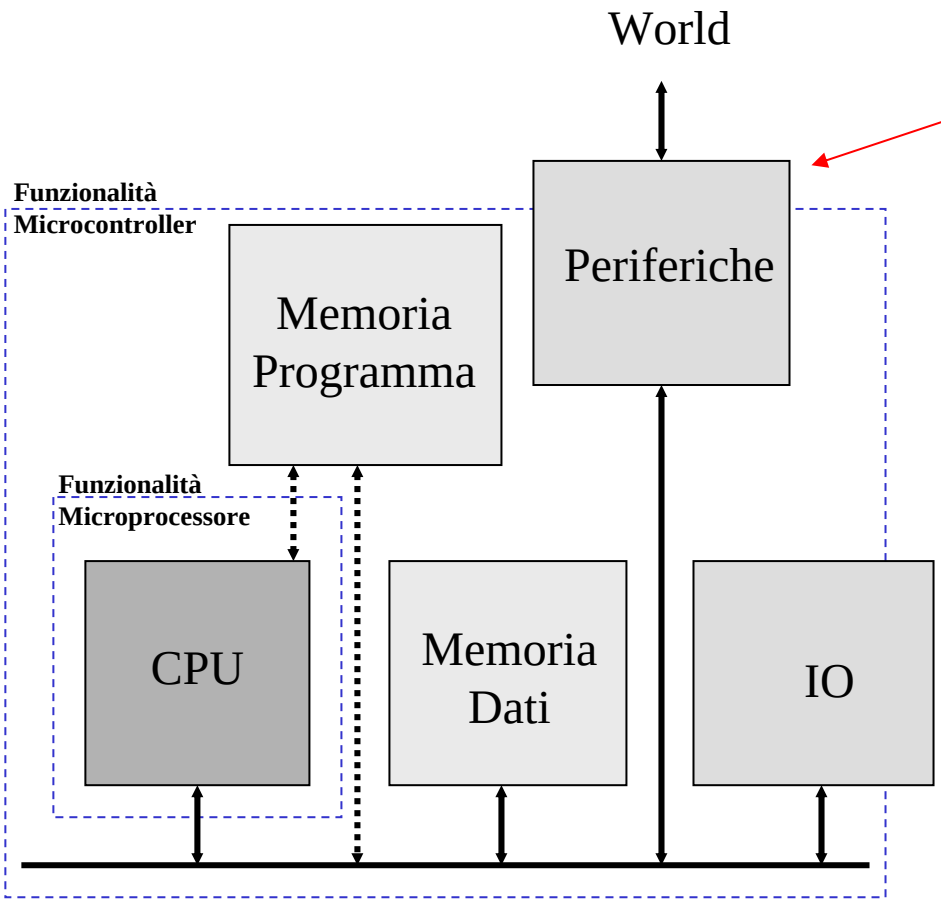
Programma → memoria non-volatile (ROM, programma non si perde allo spenimento)

Dati variabili → RAM volatile

Un altro potenziale vantaggio dell'architettura Harvard è dato dal fatto che il trasferimento dei dati e delle istruzioni di programma avviene in parallelo (velocità doppia).

Architettura generica di un microcontroller

Tipica architettura di un microcontroller: CPU, memorie, I/O e periferiche per l'interfaccia con l'esterno. Componenti collegate da un Bus comune!



Periferiche: timers, counters, porte scambio dati (seriali parallele) convertitori Analogico-Digitale (DAC) e Digitale-Analogico (DAC) integrati.

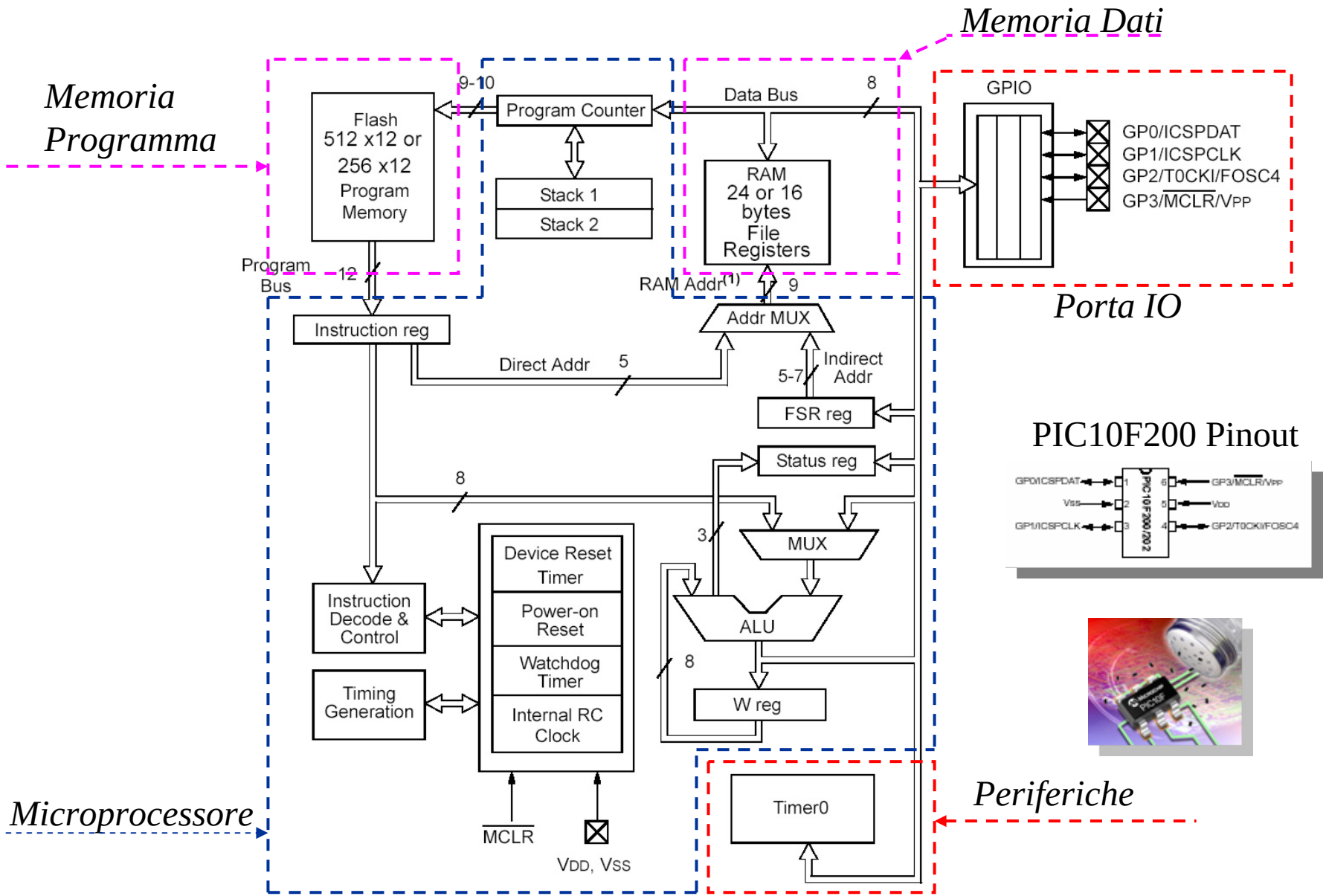
Vantaggi dell'integrazione

- Molteplici funzionalità in chip singolo
- Costi e dimensioni minori
- Minor consumo energetico
- Minor numero di connessioni esterne
- Più pin disponibili per I/O
- Maggior affidabilità del sistema (minor numero di componenti)

Svantaggi dell'integrazione

- Flessibilità delle periferiche ridotta
- Limitata espandibilità (memoria e IO)
- Performance minori di periferiche e IO

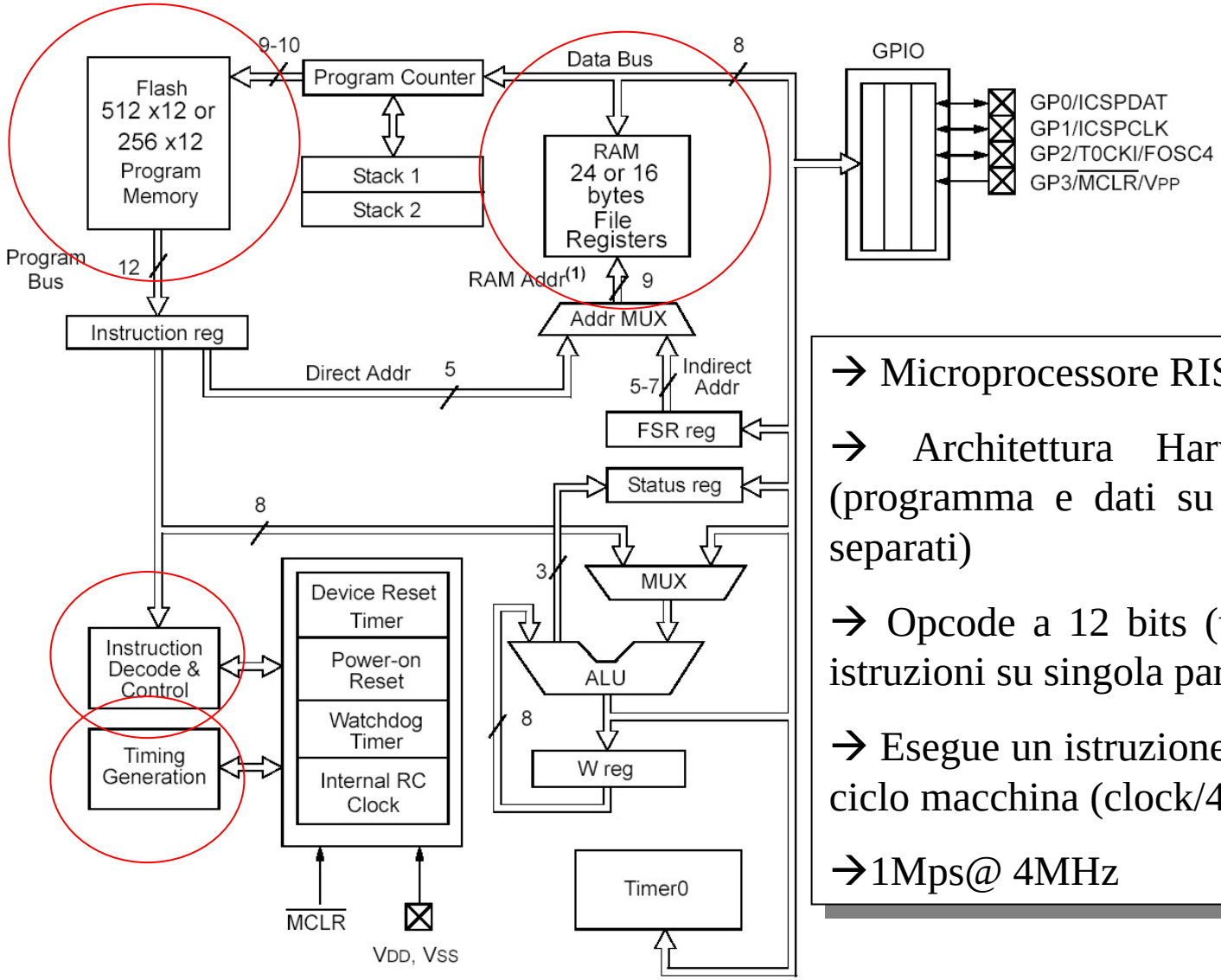
Architettura di un semplice microcontrollore - PIC10F200 (Microchip)



PIC10F200 Pinout

Architettura di un semplice microcontrollore - PIC10F200

CPU



→ Microprocessore RISC

→ Architettura Harvard (programma e dati su bus separati)

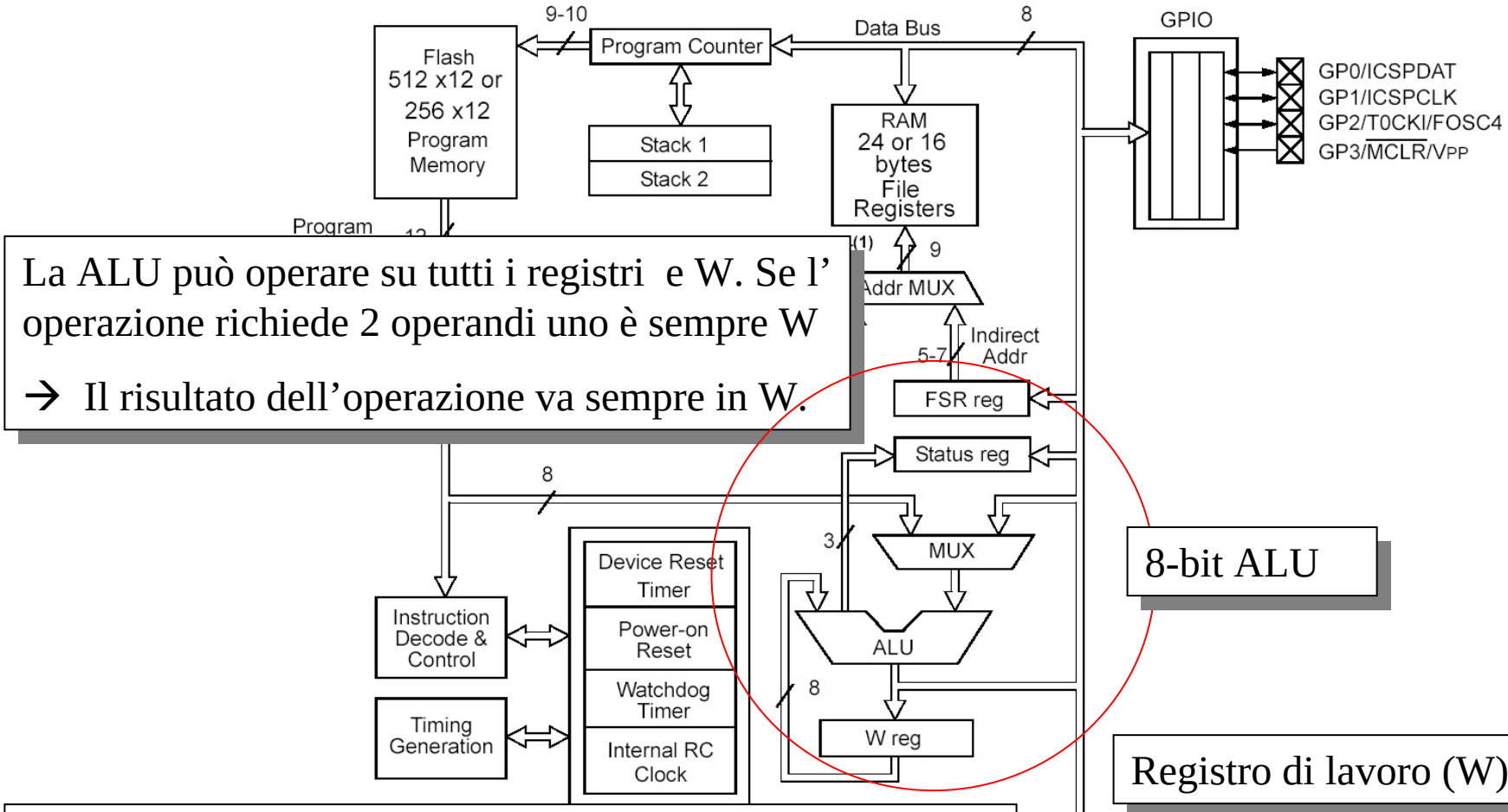
→ Opcode a 12 bits (tutte istruzioni su singola parola)

→ Esegue un istruzione per ciclo macchina (clock/4)

→1Mps@ 4MHz

Architettura di un semplice microcontrollore - PIC10F200

CPU



La ALU può operare su tutti i registri e W. Se l' operazione richiede 2 operandi uno è sempre W

→ Il risultato dell'operazione va sempre in W.

8-bit ALU

Registro di lavoro (W)

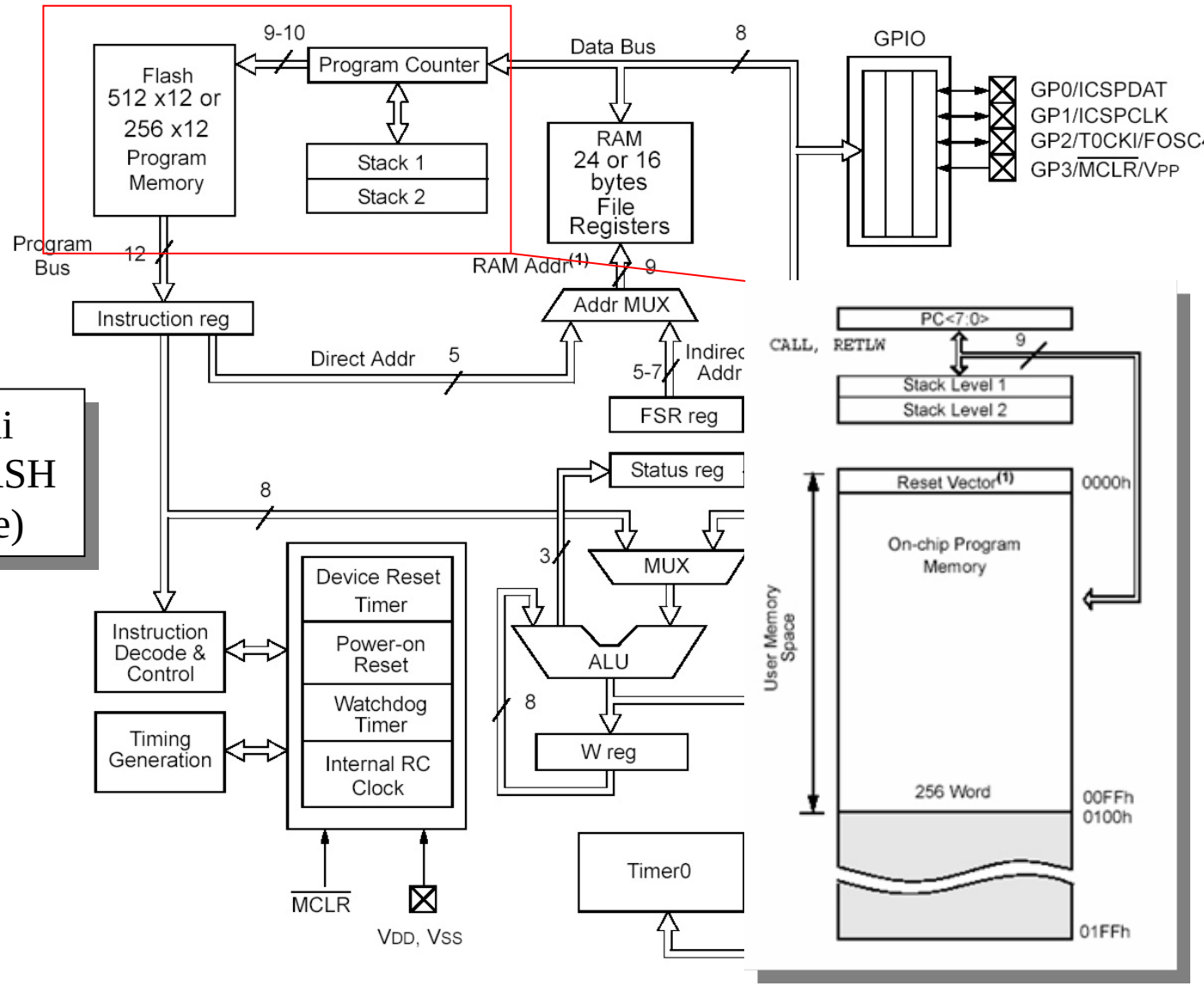
Operazioni supportate: +, -, AND, OR, XOR, NOT, shift

Nello "Status register" si trovano i bit di Carry (C,) e Zero (Z) che vengono influenzati dalle operazioni della ALU

Architettura di un semplice microcontrollore - PIC10F200

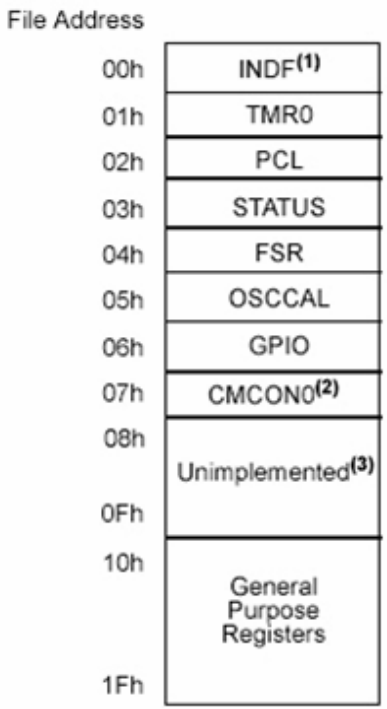
Memoria di Programma

256 bytes di memoria FLASH (non volatile)

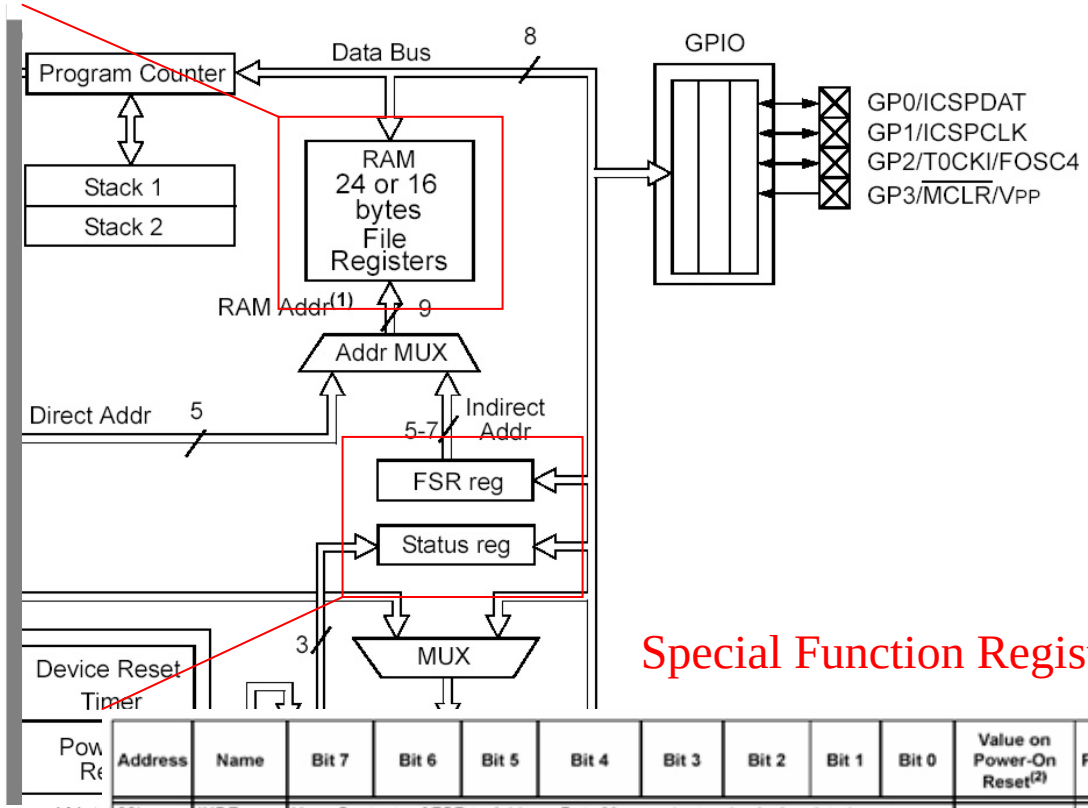


Architettura di un semplice microcontrollore - PIC10F200

Memoria Dati



16 bytes di memoria RAM statica (volatile)

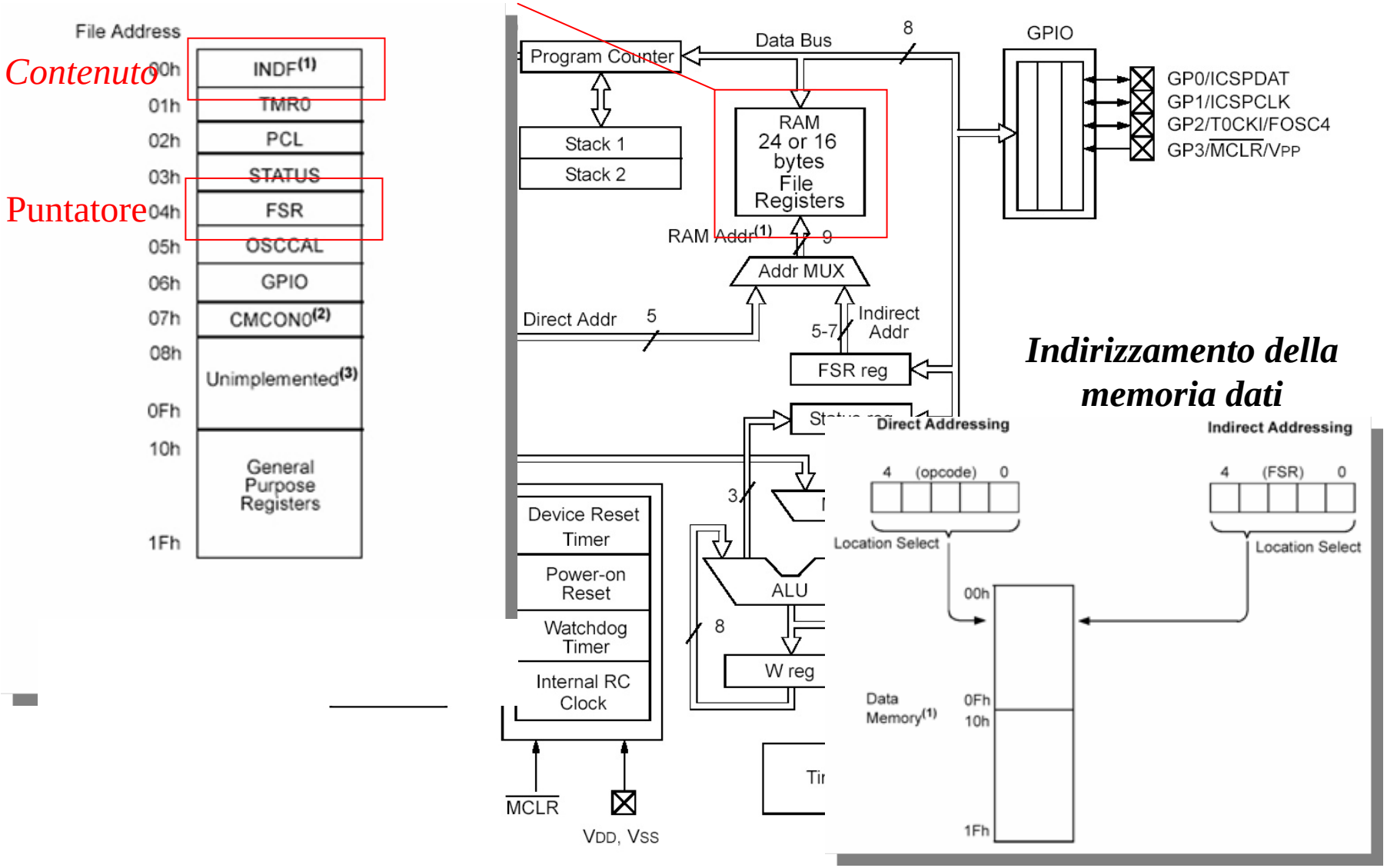


Special Function Registers

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-On Reset(2)	Page #
00h	INDF	Uses Contents of FSR to Address Data Memory (not a physical register)								xxxx xxxx	23
01h	TMR0	8-bit Real-Time Clock/Counter								xxxx xxxx	29, 33
02h(1)	PCL	Low-order 8 bits of PC								1111 1111	22
03h	STATUS	GPWUF	CWUF(5)	—	$\overline{TO}$	$\overline{PD}$	Z	DC	C	00-1 1xxx(3)	19
04h	FSR	Indirect Data Memory Address Pointer								111x xxxx	23
05h	OSCCAL	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0	FOSC4	1111 1110	21
06h	GPIO	—	—	—	—	GP3	GP2	GP1	GP0	---- xxxx	25
07h(4)	CMCON0	CMPOUT	$\overline{COUTEN}$	POL	$\overline{CMPTOCS}$	CMPON	CNREF	CPREF	$\overline{CWU}$	1111 1111	34
N/A	TRISGPIO	—	—	—	—	I/O Control Register				---- 1111	37
N/A	OPTION	$\overline{GPWU}$	$\overline{GPPU}$	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	20

Architettura di un semplice microcontrollore - PIC10F200

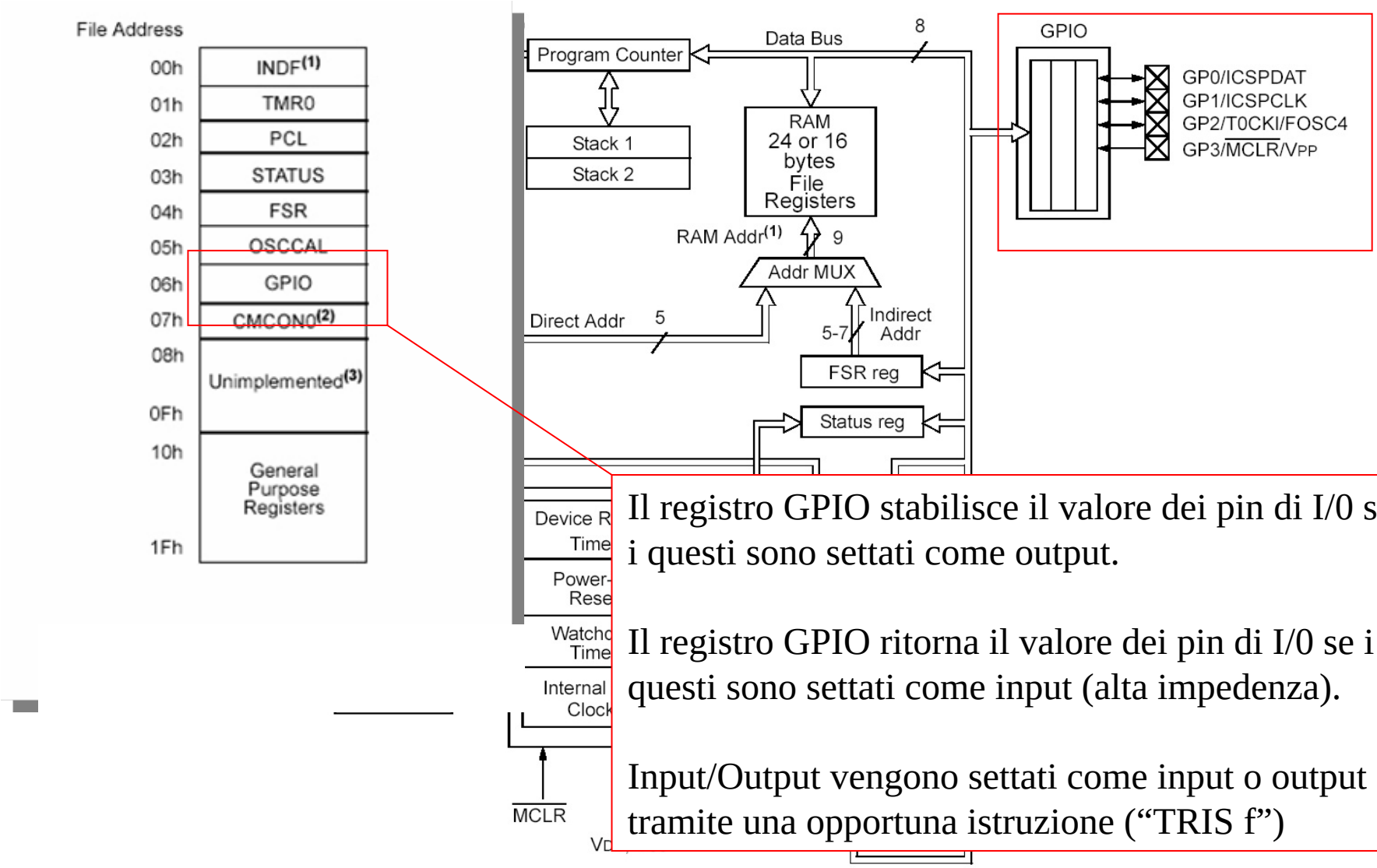
Memoria Dati



Indirizzamento della memoria dati

Architettura di un semplice microcontrollore - PIC10F200

Pin di Input/Output (Porta bidirezionale)



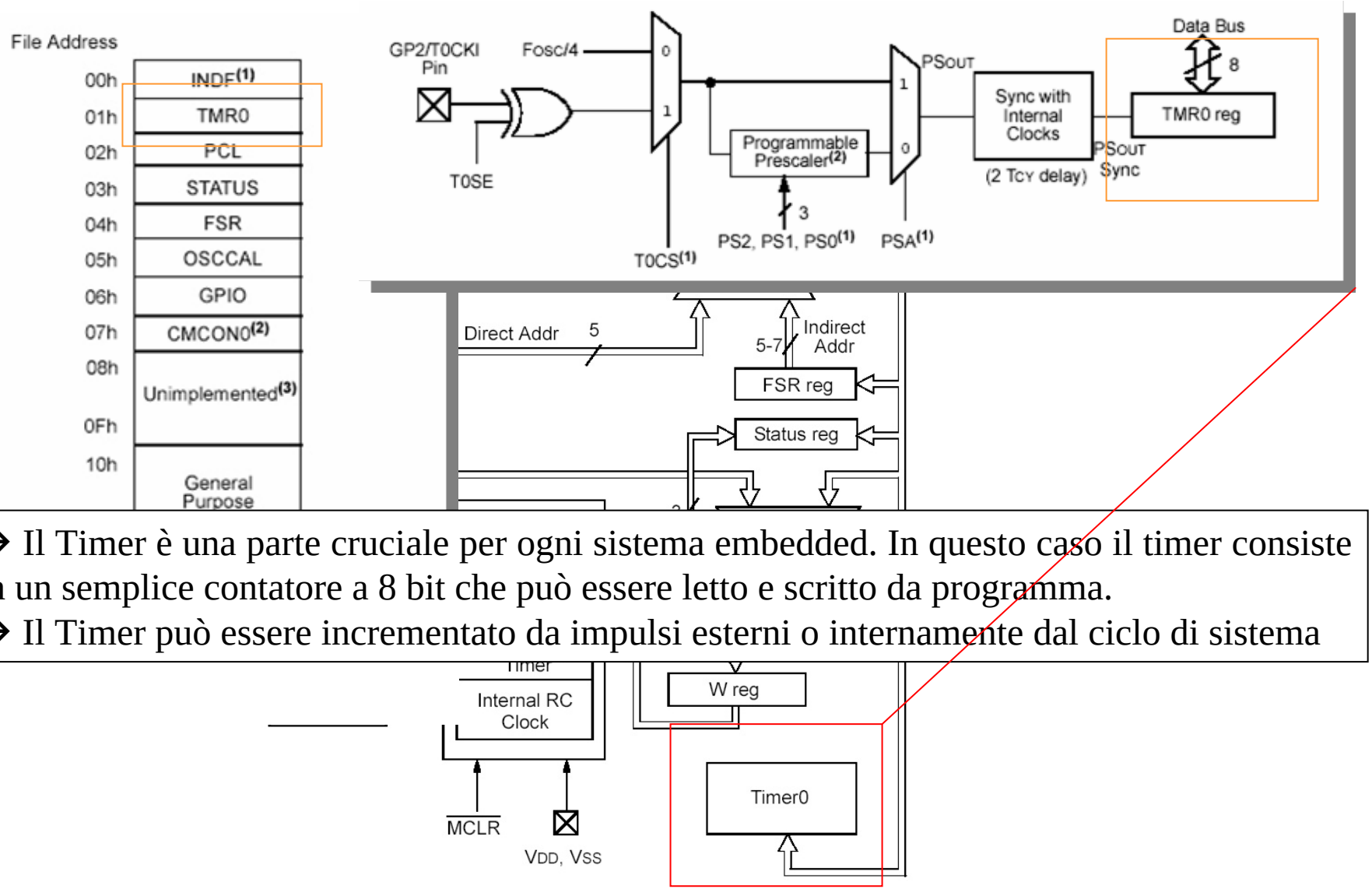
Il registro GPIO stabilisce il valore dei pin di I/0 se i questi sono settati come output.

Il registro GPIO ritorna il valore dei pin di I/0 se i questi sono settati come input (alta impedenza).

Input/Output vengono settati come input o output tramite una opportuna istruzione (“TRIS f”)

Architettura di un semplice microcontrollore - PIC10F200

Pin di Input/Output



→ Il Timer è una parte cruciale per ogni sistema embedded. In questo caso il timer consiste in un semplice contatore a 8 bit che può essere letto e scritto da programma.

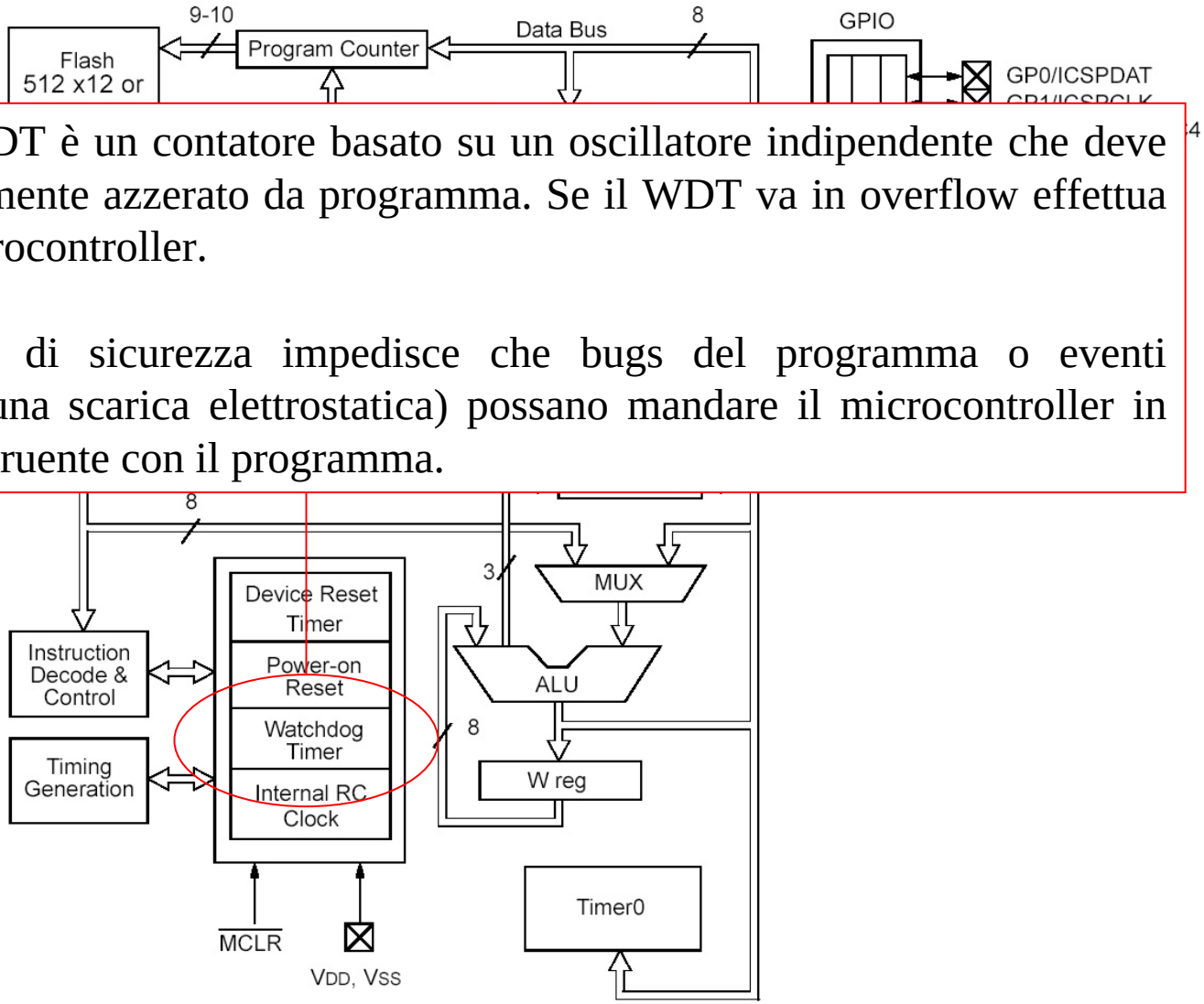
→ Il Timer può essere incrementato da impulsi esterni o internamente dal ciclo di sistema

Architettura di un semplice microcontrollore - PIC10F200

Wathcdog Timer

Il Watchdog WDT è un contatore basato su un oscillatore indipendente che deve essere continuamente azzerato da programma. Se il WDT va in overflow effettua un reset del microcontroller.

Questo sistema di sicurezza impedisce che bugs del programma o eventi imprevisti (es. una scarica elettrostatica) possano mandare il microcontroller in uno stato incongruente con il programma.



Architettura di un semplice microcontrollore - PIC10F200

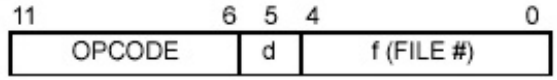
Il set di istruzioni

Il set di istruzioni per il PIC10F200 è composto da 33 istruzioni divise in tre categorie:

- Operazioni Byte-oriented
- Operazioni Bit-oriented
- Operazioni con costanti e di controllo

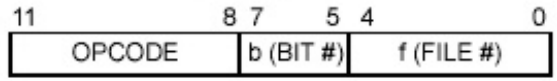
Ogni istruzione è una parola **12-bit** che contiene l'**opcode** (che specifica il tipo di istruzione) e uno o più **operandi**

Byte-oriented file register operations



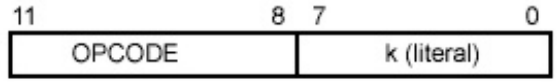
d = 0 for destination W
d = 1 for destination f
f = 5-bit file register address

Bit-oriented file register operations



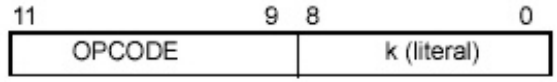
b = 3-bit address
f = 5-bit file register address

Literal and control operations (except GOTO)



k = 8-bit immediate value

Literal and control operations – GOTO instruction



k = 9-bit immediate value

Architettura di un semplice microcontrollore - PIC10F200

Il set di istruzioni

Byte-oriented

Mnemonic, Operands		Description	Cycles	12-Bit Opcode MSb		LSb	Status Affected	Notes
ADDWF	f, d	Add W and f	1	0001	11df	ffff	C, DC, Z	1, 2, 4
ANDWF	f, d	AND W with f	1	0001	01df	ffff	Z	2, 4
CLRF	f	Clear f	1	0000	011f	ffff	Z	4
CLRW	–	Clear W	1	0000	0100	0000	Z	
COMF	f, d	Complement f	1	0010	01df	ffff	Z	
DECF	f, d	Decrement f	1	0000	11df	ffff	Z	2, 4
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	0010	11df	ffff	None	2, 4
INCF	f, d	Increment f	1	0010	10df	ffff	Z	2, 4
INCFSZ	f, d	Increment f, Skip if 0	1(2)	0011	11df	ffff	None	2, 4
IORWF	f, d	Inclusive OR W with f	1	0001	00df	ffff	Z	2, 4
MOVF	f, d	Move f	1	0010	00df	ffff	Z	2, 4
MOVWF	f	Move W to f	1	0000	001f	ffff	None	1, 4
NOP	–	No Operation	1	0000	0000	0000	None	
RLF	f, d	Rotate left f through Carry	1	0011	01df	ffff	C	2, 4
RRF	f, d	Rotate right f through Carry	1	0011	00df	ffff	C	2, 4
SUBWF	f, d	Subtract W from f	1	0000	10df	ffff	C, DC, Z	1, 2, 4
SWAPF	f, d	Swap f	1	0011	10df	ffff	None	2, 4
XORWF	f, d	Exclusive OR W with f	1	0001	10df	ffff	Z	2, 4

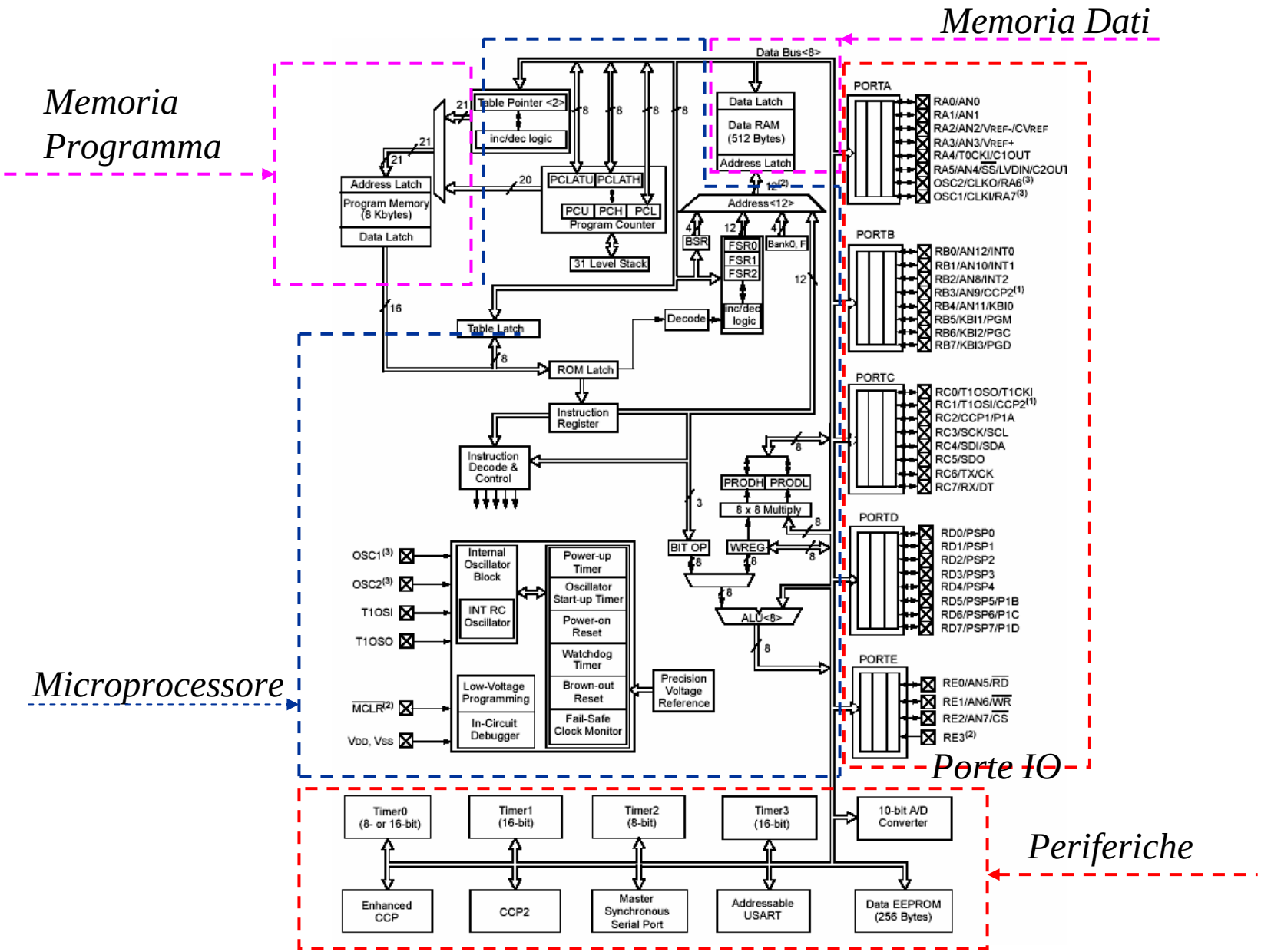
Bit-oriented

BCF	f, b	Bit Clear f	1	0100	bbbf	ffff	None	2, 4
BSF	f, b	Bit Set f	1	0101	bbbf	ffff	None	2, 4
BTFSC	f, b	Bit Test f, Skip if Clear	1(2)	0110	bbbf	ffff	None	
BTFSS	f, b	Bit Test f, Skip if Set	1(2)	0111	bbbf	ffff	None	

Costanti e controllo

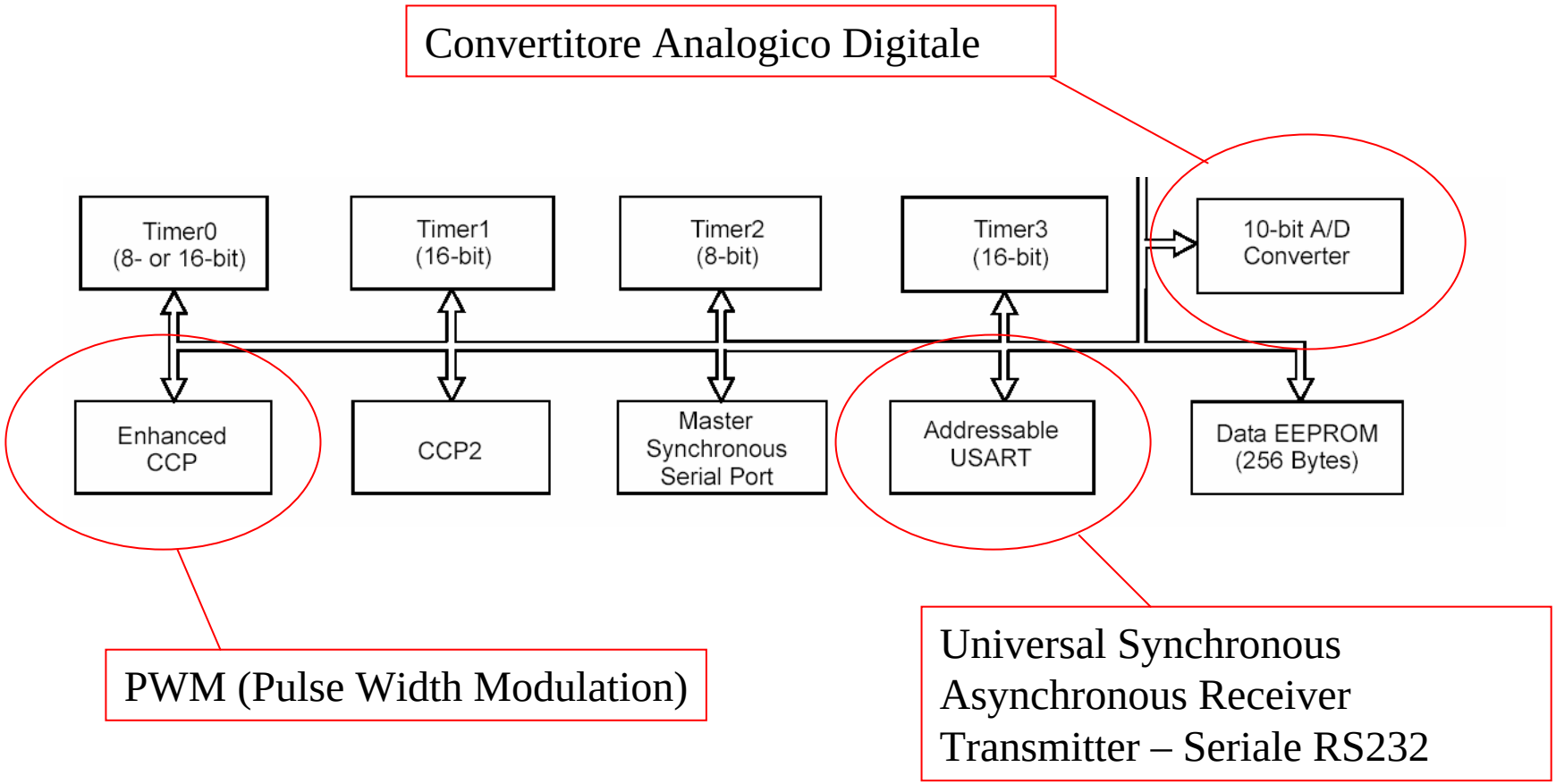
ANDLW	k	AND literal with W	1	1110	kkkk	kkkk	Z	
CALL	k	Call Subroutine	2	1001	kkkk	kkkk	None	1
CLRWDT		Clear Watchdog Timer	1	0000	0000	0100	$\overline{TO}$, $\overline{PD}$	
GOTO	k	Unconditional branch	2	101k	kkkk	kkkk	None	
IORLW	k	Inclusive OR literal with W	1	1101	kkkk	kkkk	Z	
MOVLW	k	Move literal to W	1	1100	kkkk	kkkk	None	
OPTION	–	Load Option register	1	0000	0000	0010	None	
RETLW	k	Return, place Literal in W	2	1000	kkkk	kkkk	None	
SLEEP	–	Go into Standby mode	1	0000	0000	0011	$\overline{TO}$, $\overline{PD}$	
TRIS	f	Load TRIS register	1	0000	0000	0fff	None	3
XORLW	k	Exclusive OR literal to W	1	1111	kkkk	kkkk	Z	

Architettura di un microcontrollore più complesso - PIC18F4320 (Microchip)



Architettura di un microcontrollore più complesso - PIC18F4320 (Microchip)

Altre Periferiche



Architecture Software

Il software per sistemi embedded è di solito implementato sulla base di una delle seguenti architetture:

Singolo ciclo di polling - Singolo pezzo di codice che testa continuamente un certo imput ed esegue di conseguenza le routines necessarie

Macchina a stati - Compartimentazione delle funzionalità, richiede duplicazione del codice, adatto a applicazioni che effettuano singole funzioni

Macchina a stati multipla/Ciclo di polling - un ciclo di pollig per ogni processo, ogni ciclo di polling esegue le routines relative allo stato attuale, finito un ciclo si passa al successivo

RTOS - Sistema Operativo Real-time : permette l'esecuzione di più task in parallelo

The Development Language

Linguaggio di alto livello vs. Assembler

- → Permette applicazioni più complesse
- → Riusabilità
- → Portabilità
- → Richiede maggiori risorse

- → Velocità
- → Miglior controllo
- → Strumenti di sviluppo gratuiti
- → Specifico per il processore

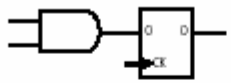
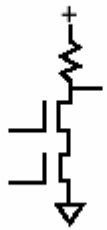
High Level	Sum := Sum + 1
Assembly	MOV BX,SUM INC (BX)
Machine	1101010100001100 0010001101110101 1111100011001101
Register Transfer	Fetch Instruction, Increment PC, Load ALU with SUM ...
Gate	
Circuit	

Figure 2-1: "Layers" of a computer system.

La scelta del microcontroller

Solitamente più scelte diverse sono possibili: nello specifico vanno tenuti in considerazione i seguenti parametri sulla base dell'applicazione finale.

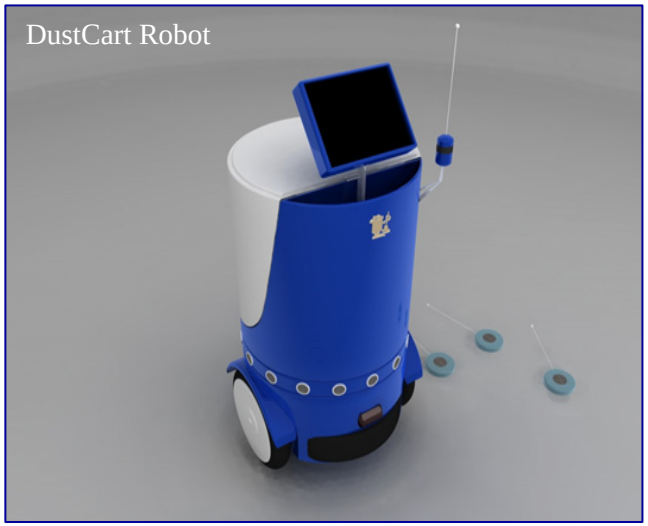
- ✓ Numero dei pin di I/O
- ✓ Interfacce
- ✓ Memoria RAM (quantità e tipo)
- ✓ Numero e tipo di interrupts
- ✓ Velocità del processore
- ✓ Consumi energetici
- ✓ Memoria di programma (quantità e tipo)
- ✓ Ambiente di sviluppo

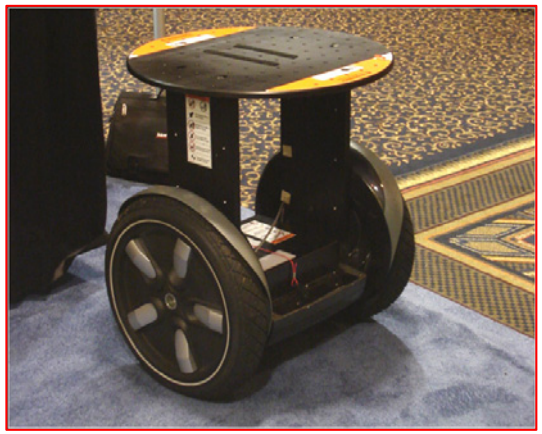
Microprocessori in robotica

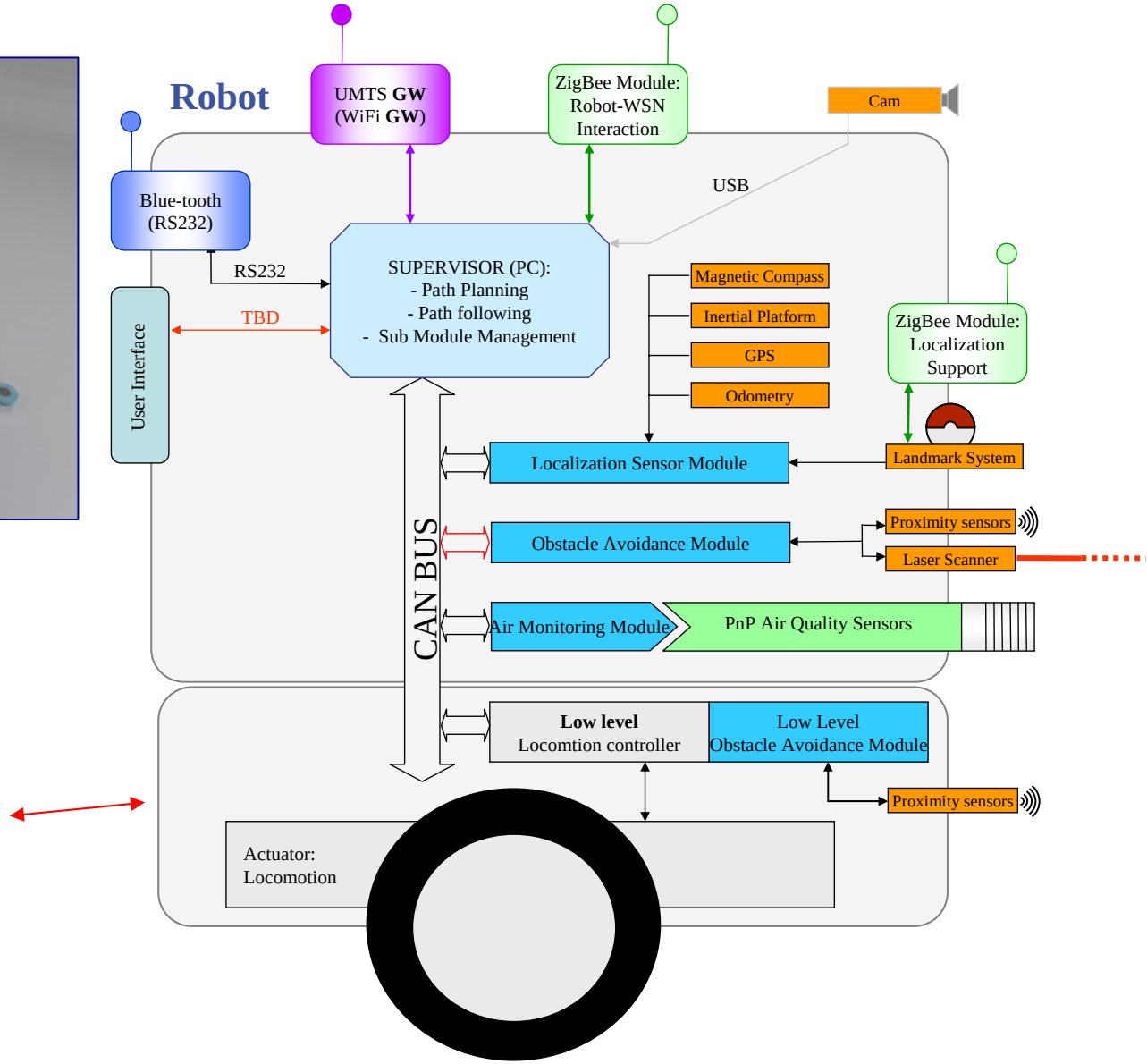
- ✓ Controllo dei motori/attuatori (basso livello)
- ✓ Controllo movimento (alto livello)
- ✓ Power management (carica/scarica batterie)
- ✓ Acquisizione/elaborazione dei segnali dei sensori (prossimità, encoders, odometria, contatto, visione)
- ✓ Interfacce

Microprocessori in robotica

Architettura di un sistema robotico - Esempio 1: DustBot

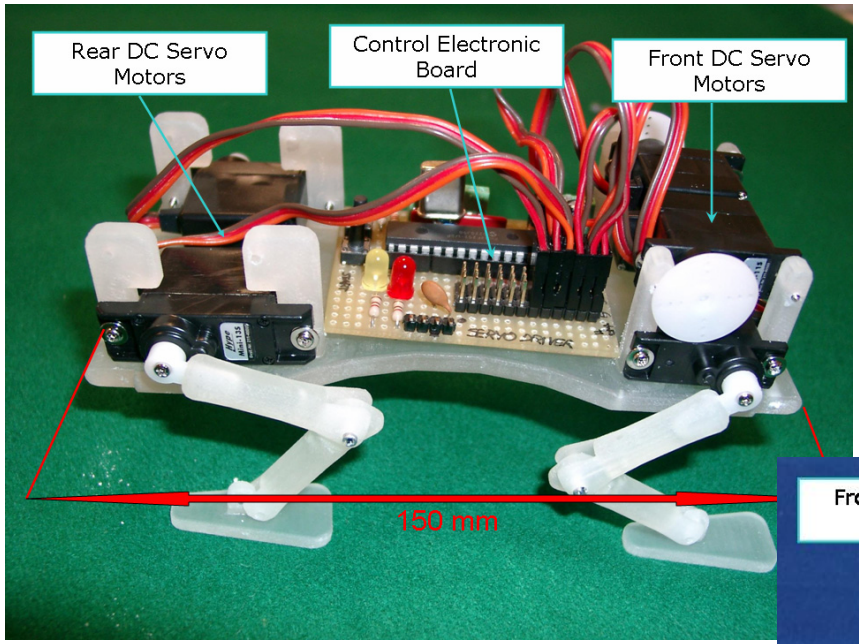


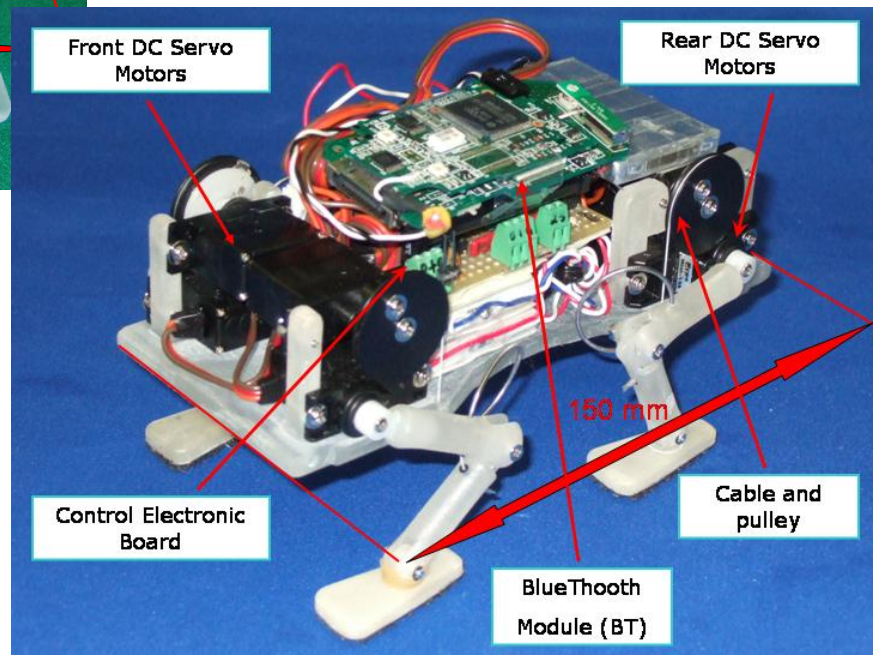




Microprocessori in robotica

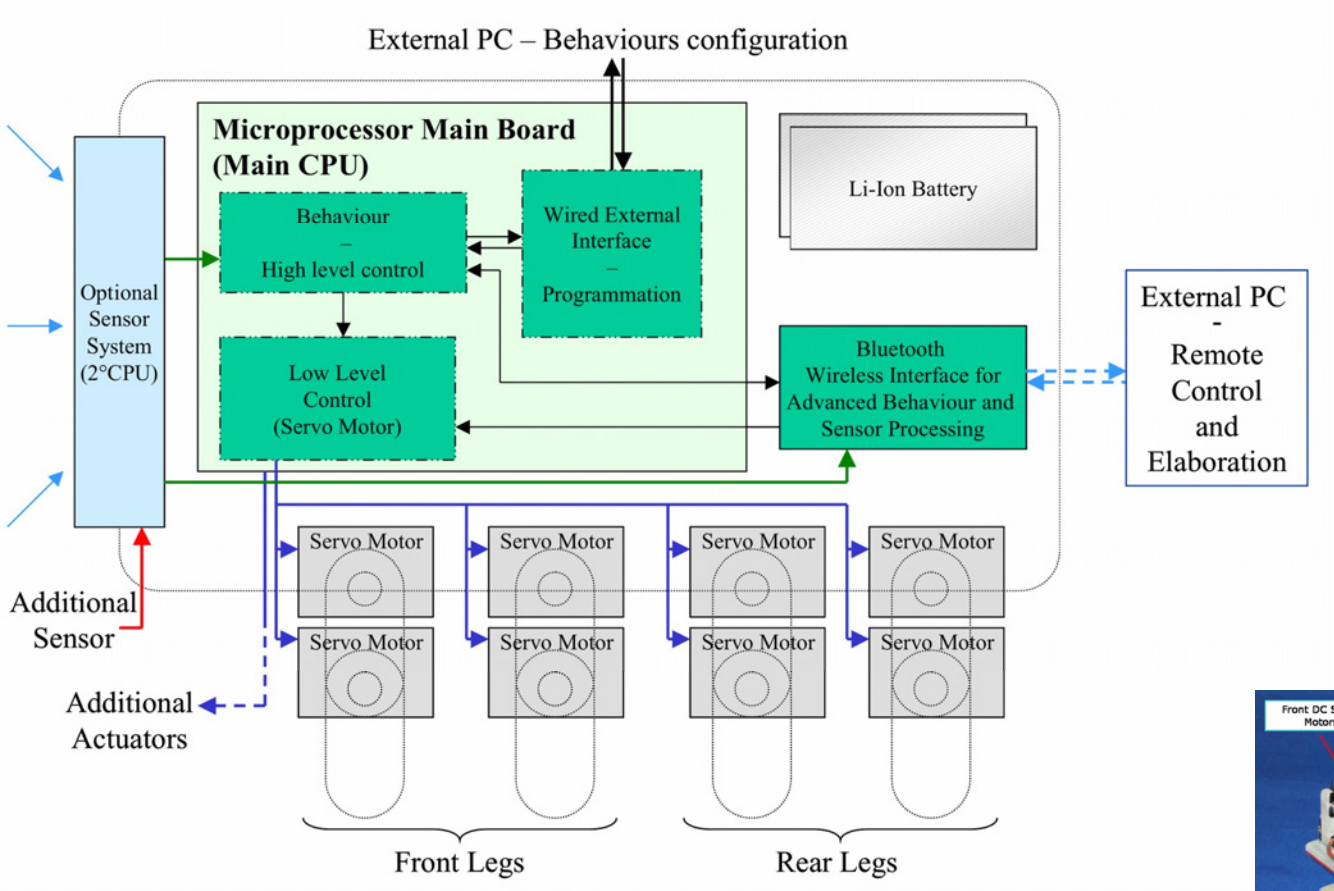
Architettura di un sistema robotico - Esempio 2: Ropbot biomimetico a 4 zampe

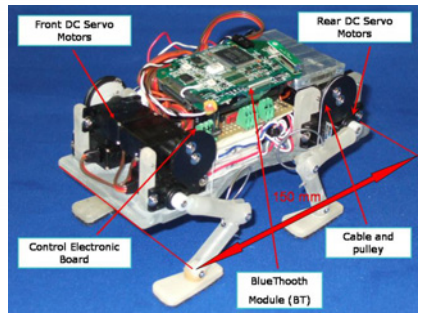




Microprocessori in robotica

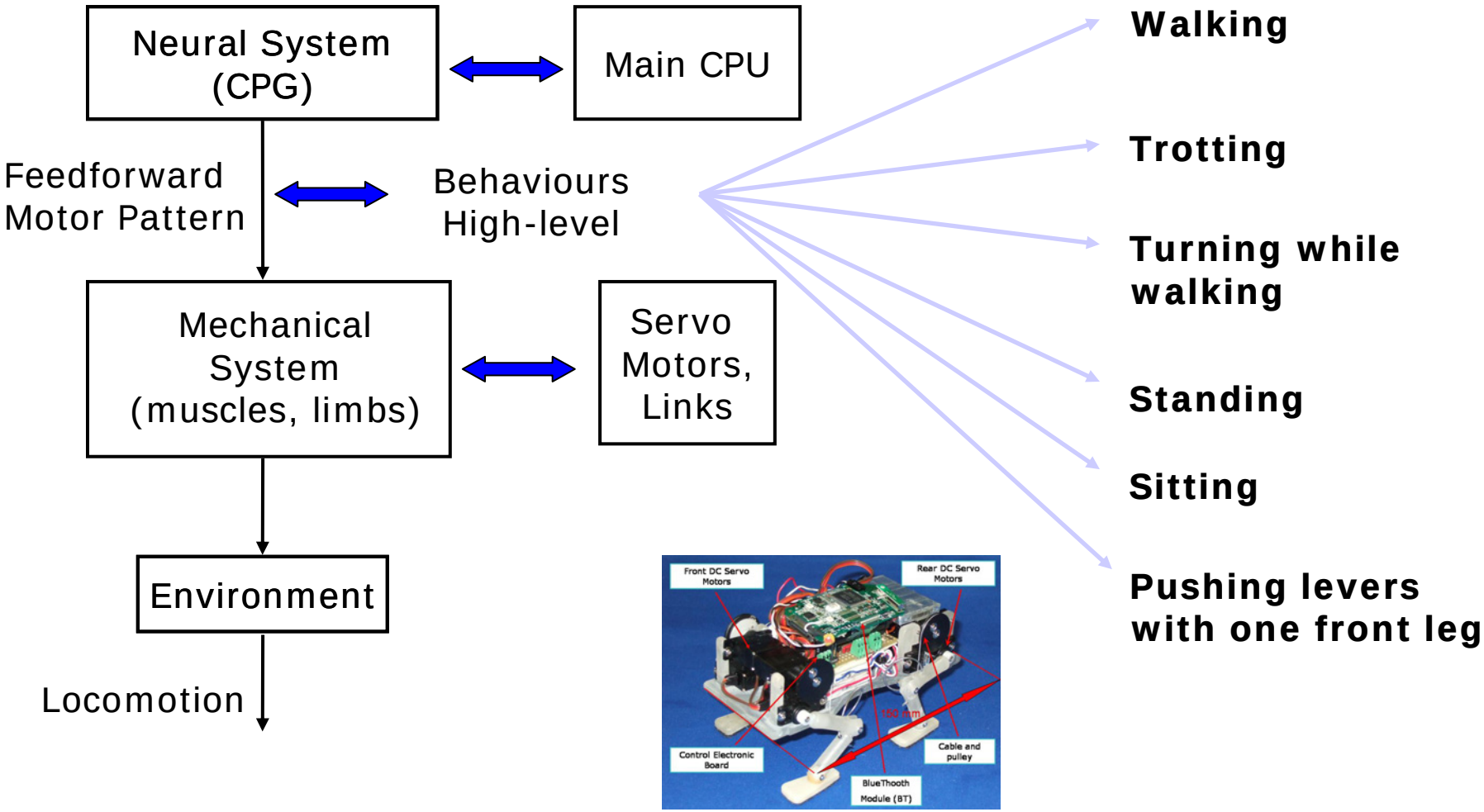
Architettura di un sistema robotico - Esempio 2: Ropbot biomimetico a 4 zampe





Microprocessori in robotica

Architettura di un sistema robotico - Esempio 2: Ropbot biomimetico a 4 zampe



Esercitazione
“ServoMotor” Driver

ServoMotor Driver Controller (Servo Driver)

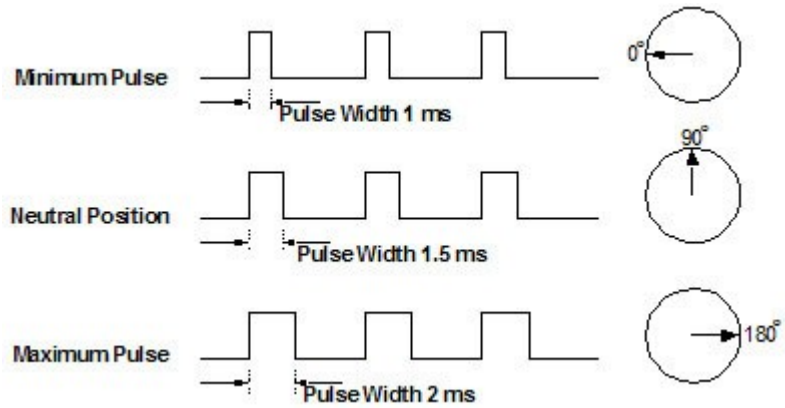
IDEA:

Provide an easy to use configurable driver/controller for the actuation of rapid prototyped robot by using RC-servos.

RC Servo

Why servomotors for robot prototypes?:

- Low weight
- Relatively High Power
- Low cost
- Easy Control (Position Control)



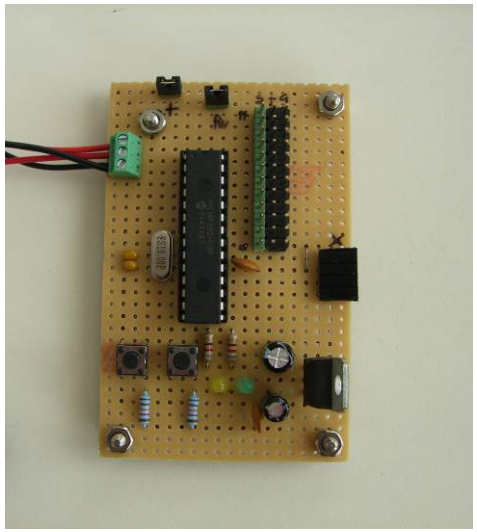
50 Hz PWM

ServoMotor Driver Controller

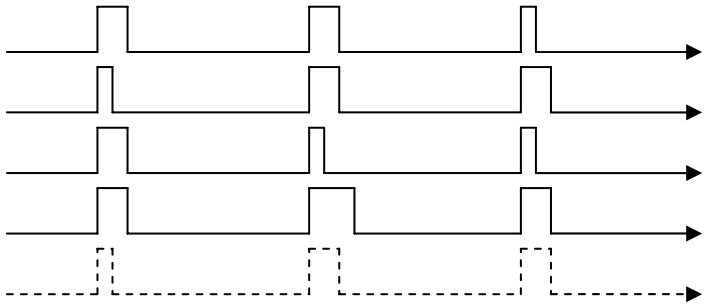
First version of Driver-Controller – Open Loop Control

Based on Microchip PIC18F252 (assembler firmware)
Interfaces with analog RC servos (12 Servos)
Max servo resolution is 0.5° - Servo span is 128°
Max update rate is 50Hz (20ms)

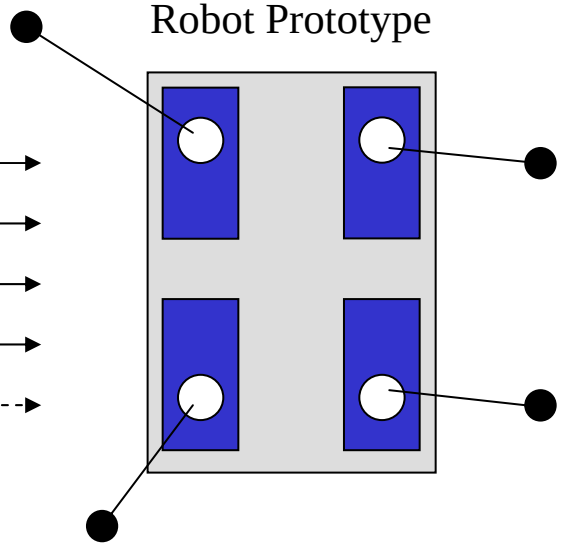
Servo Driver board

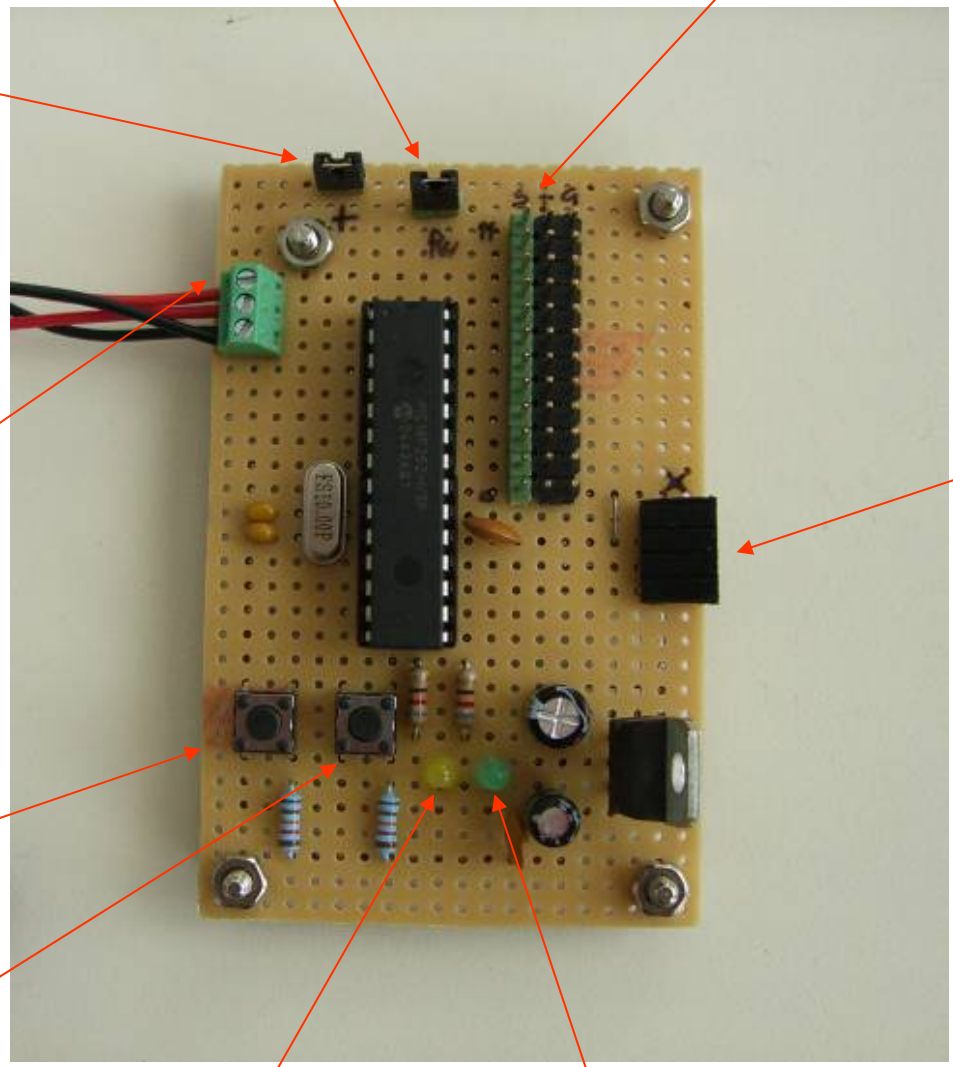


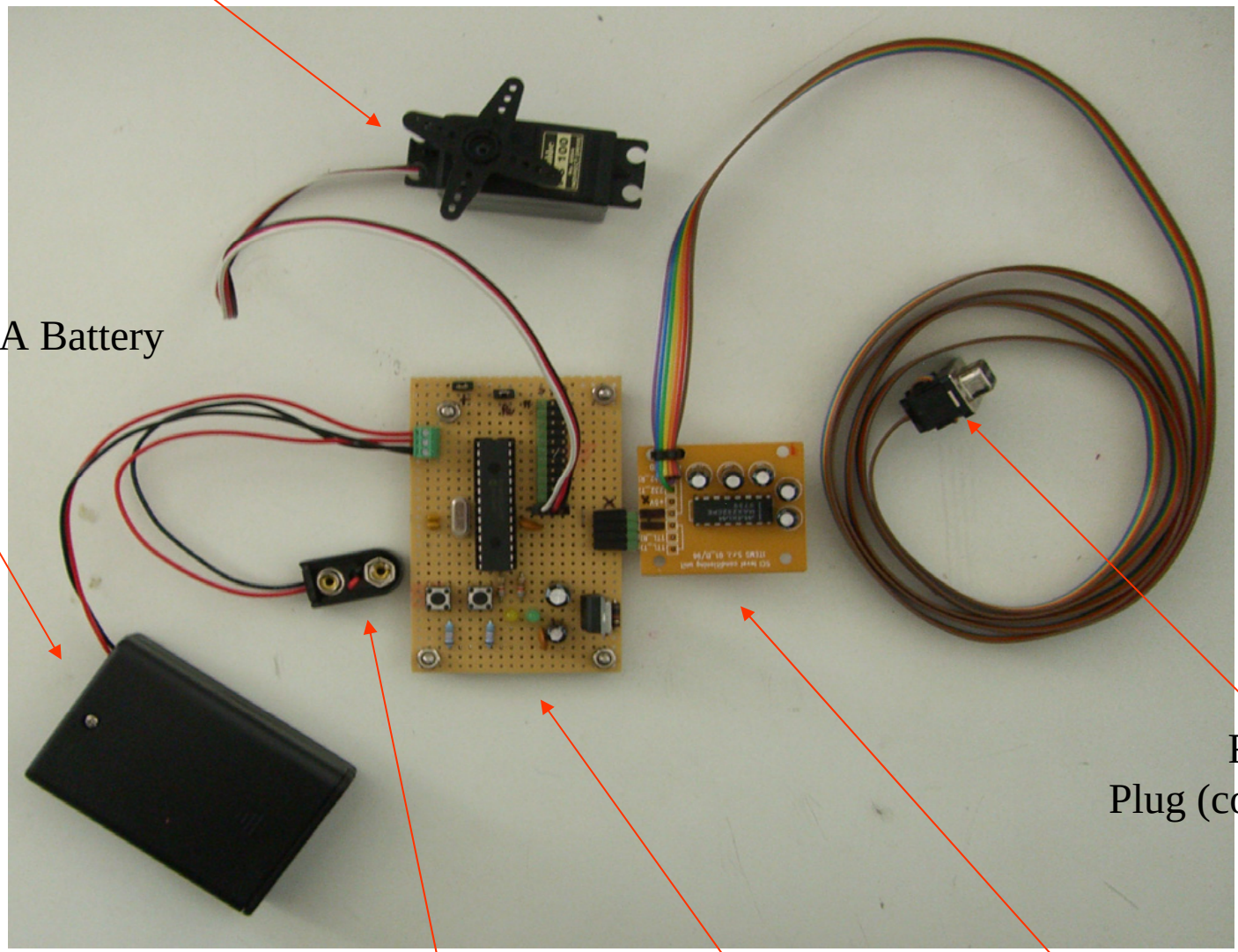
RC servos signals



Robot Prototype

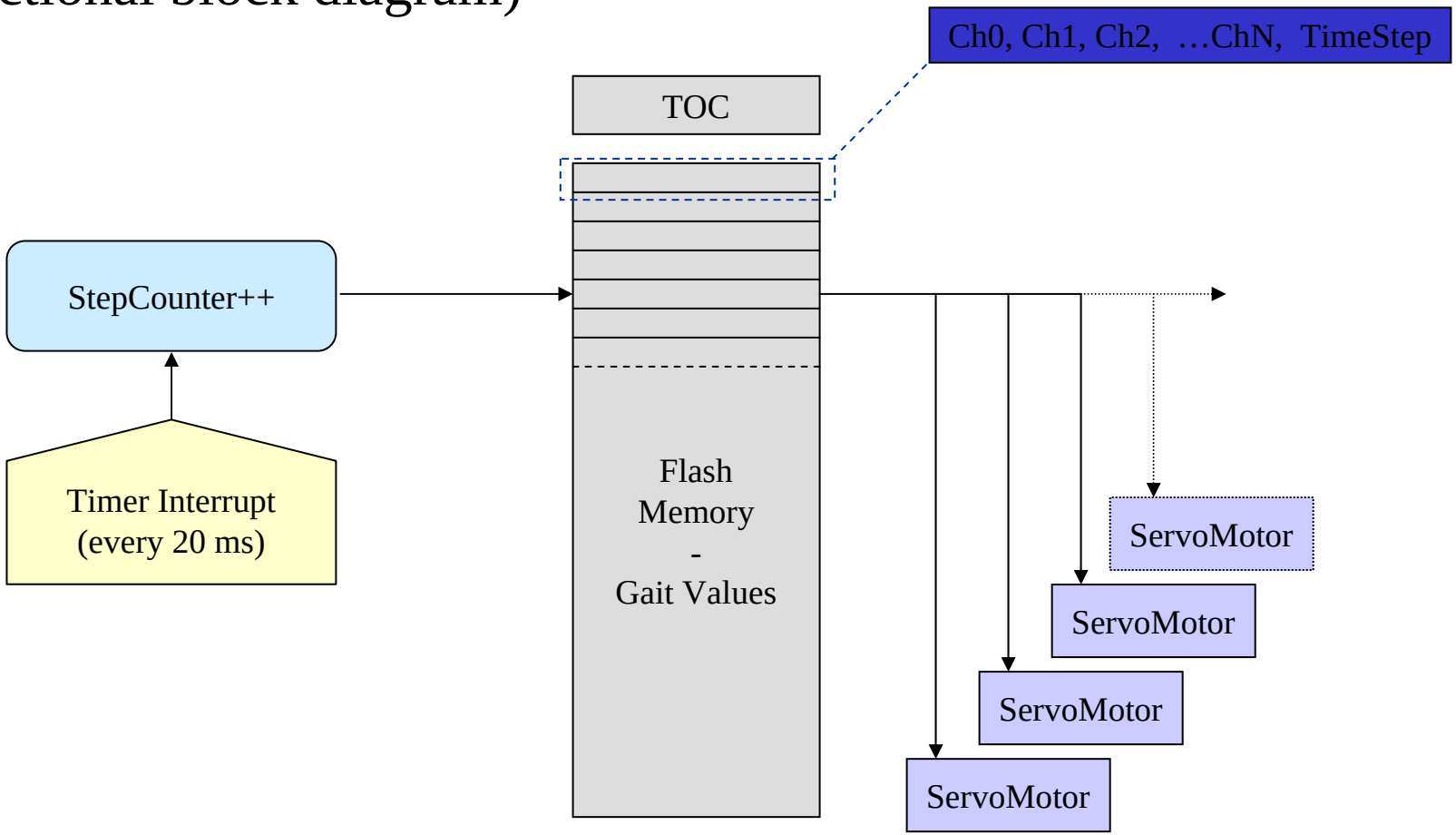



ServoMotor Power Enabling Jumper
ServoMotor Connector
Microcontroller Power Enabling Jumper
Power Connector
RS232 Connector
Key1 (START)
Key2 (STOP)
Yellow LED
Green LED


ServoMotor
3 x 1.5V AA Battery
RS232
Plug (connect to PC
1 x 9V Battery
Servo Driver board
RS232 adaptor

ServoMotor Driver Controller (FW)

Open Loop Control – Direct Gait Generator
(Functional block diagram)

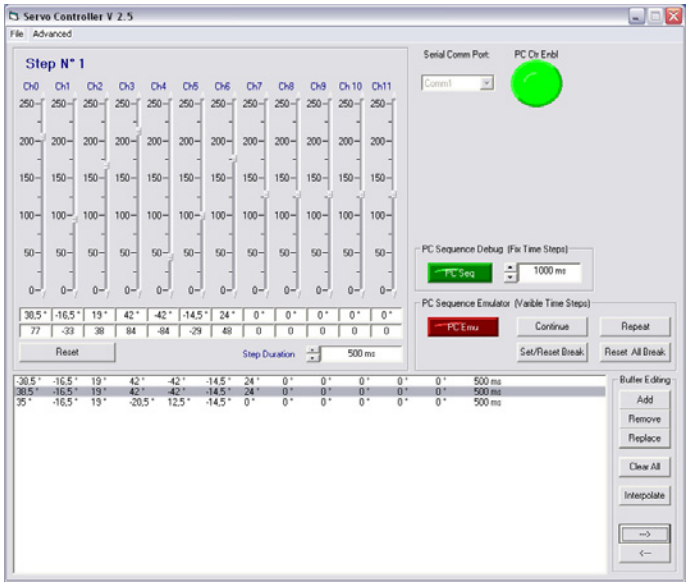


Max sequence length - 1024 steps

ServoMotor Driver Controller

Sequence Design e Memory Programming - Software tools

Software for Sequence Management
Editing / Simulation / Programming



Robot Prototype

Driver-Controller

Flash Memory
Gait Values

ServoMotor

ServoMotor

ServoMotor

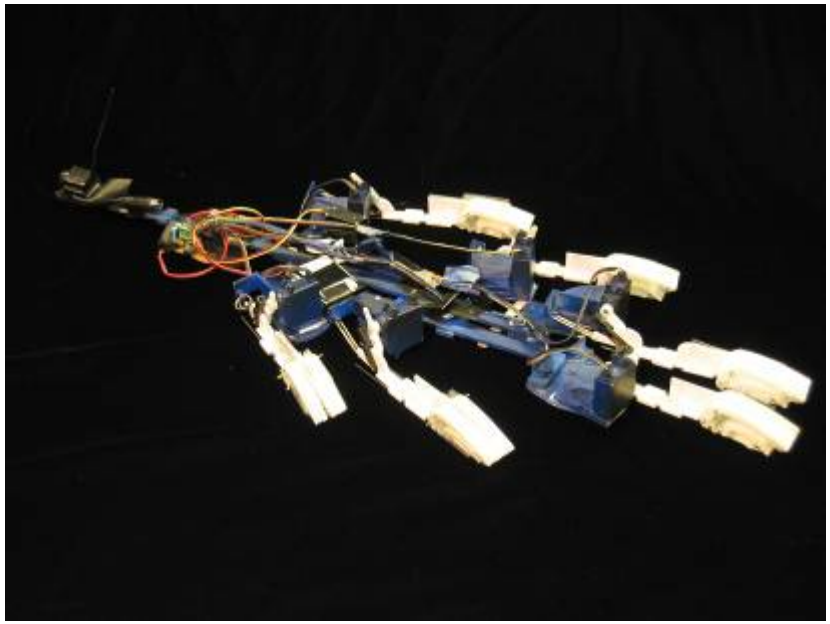
RS232

Gait Development – Based on cinematic

ServoMotor Driver Controller

Used at Stanford For:

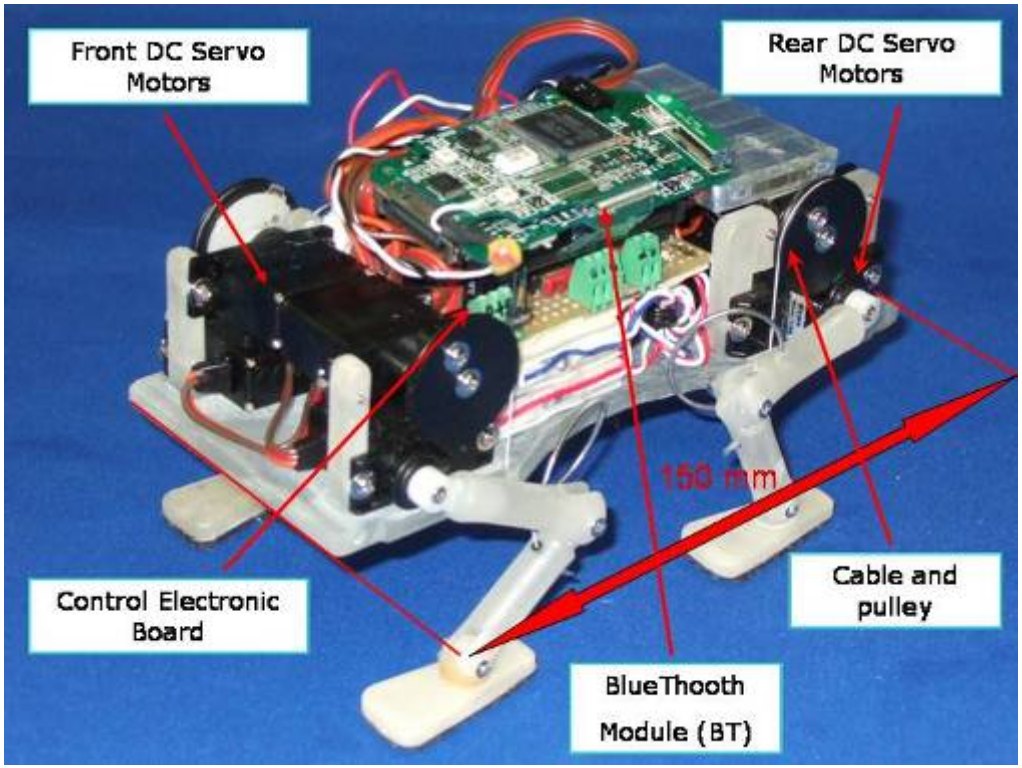
SpinyBot



StickyBot

ServoMotor Driver Controller - New Version

Used at Robocasa-Waseda for Rat-Robot prototype:



High-level Behaviors

Walking

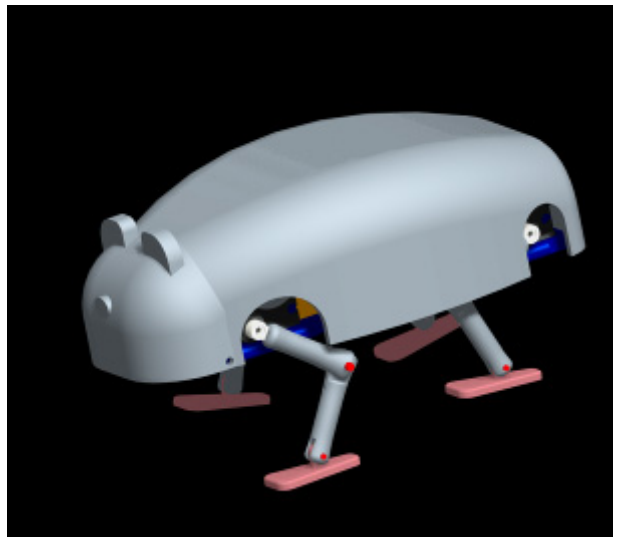
Trotting

Turning while walking

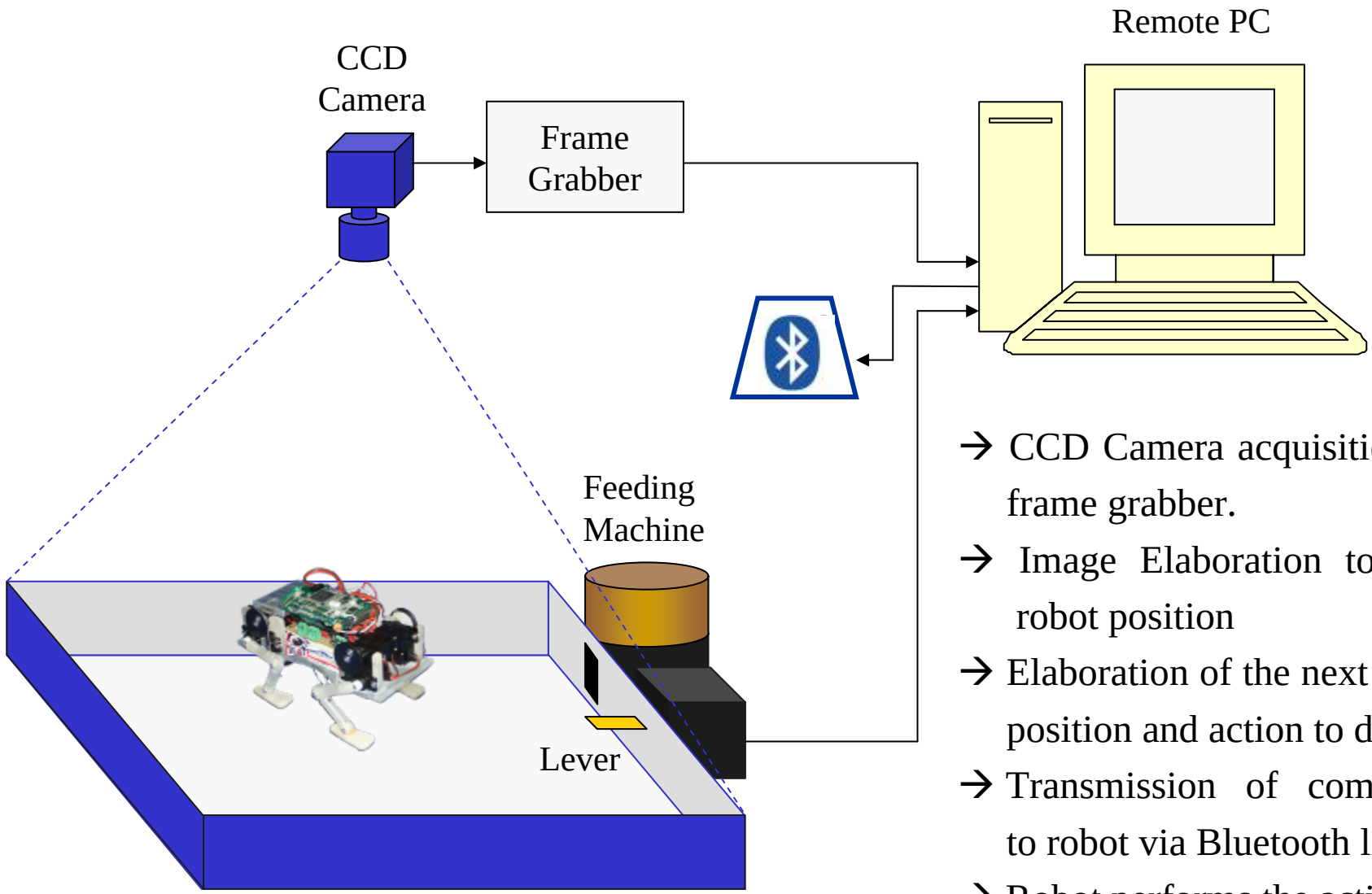
Standing

Sitting

Pushing levers with one front leg



Overview of the experimental Rat-Robot system for animal psicology experiments



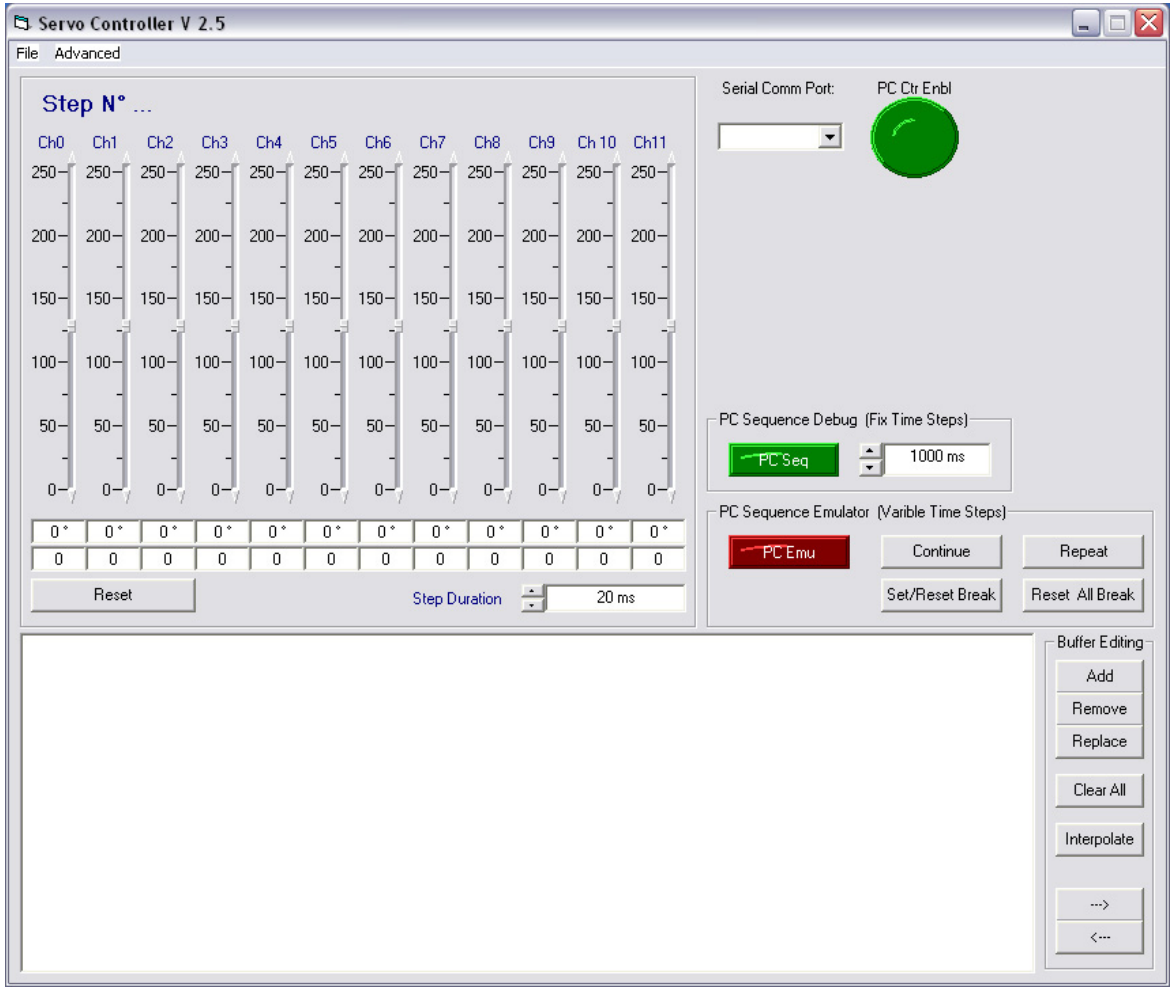
- → CCD Camera acquisition by frame grabber.
- → Image Elaboration to find robot position
- → Elaboration of the next robot position and action to do
- → Transmission of command to robot via Bluetooth link
- → Robot performs the action

Closed loop - Discrete Reactive Control System

“ServoController” software

“ServoController” software

Dedicated PC software to support the implementation of movement sequences in servomotors-based robot prototypes.



<u>The software allows:</u>

→ Real-time control of servos connecting the Servo Driver board through a RS232 connection

→ Advanced sequences editing

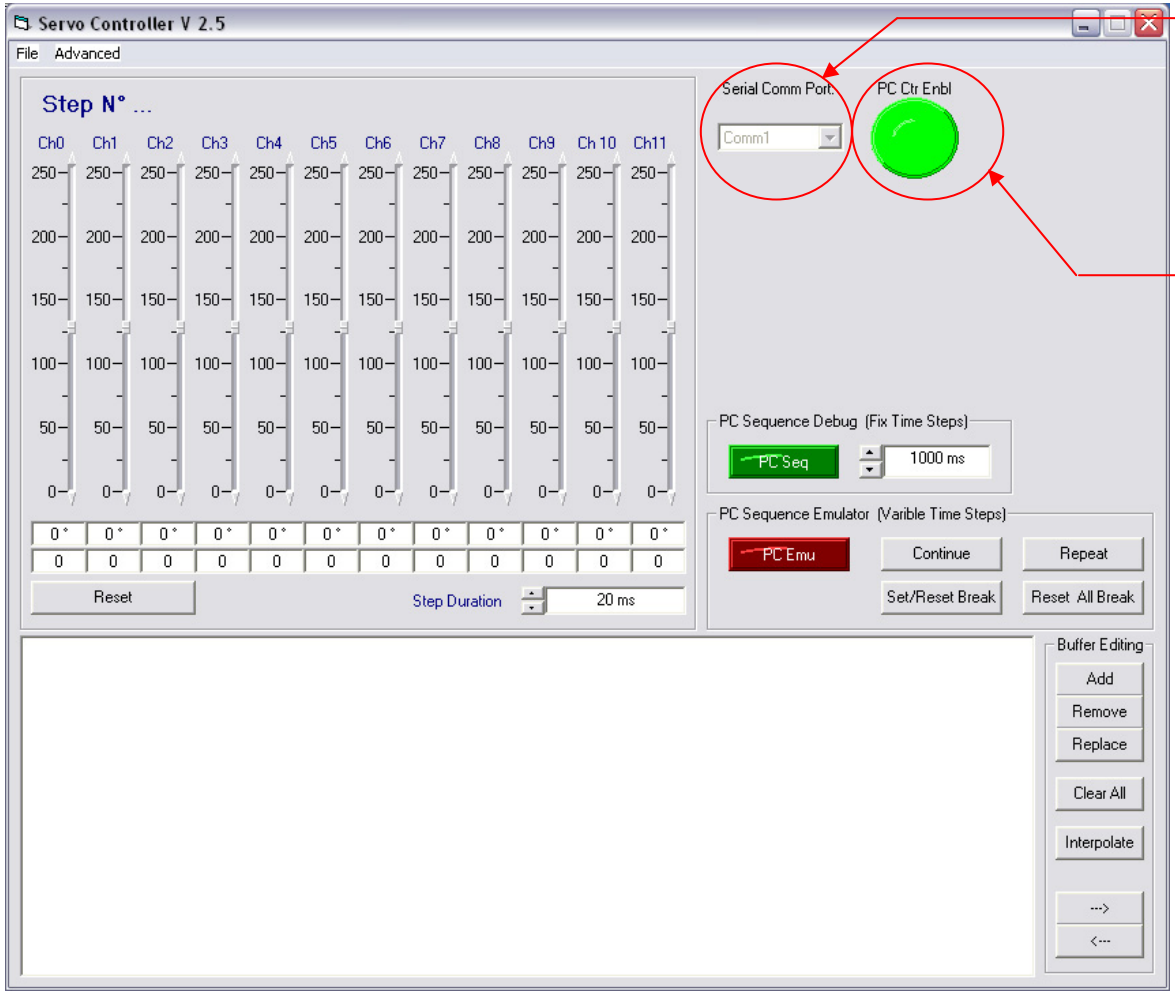
→ Sequences import/export

→ Program/read sequences on the ServoDriver board

→ Emulation of sequences with multiple breakpoints

“ServoController” software

First step: setting of RS232 comm port for board interface
Second step: Enable PC control of ServoDriver board

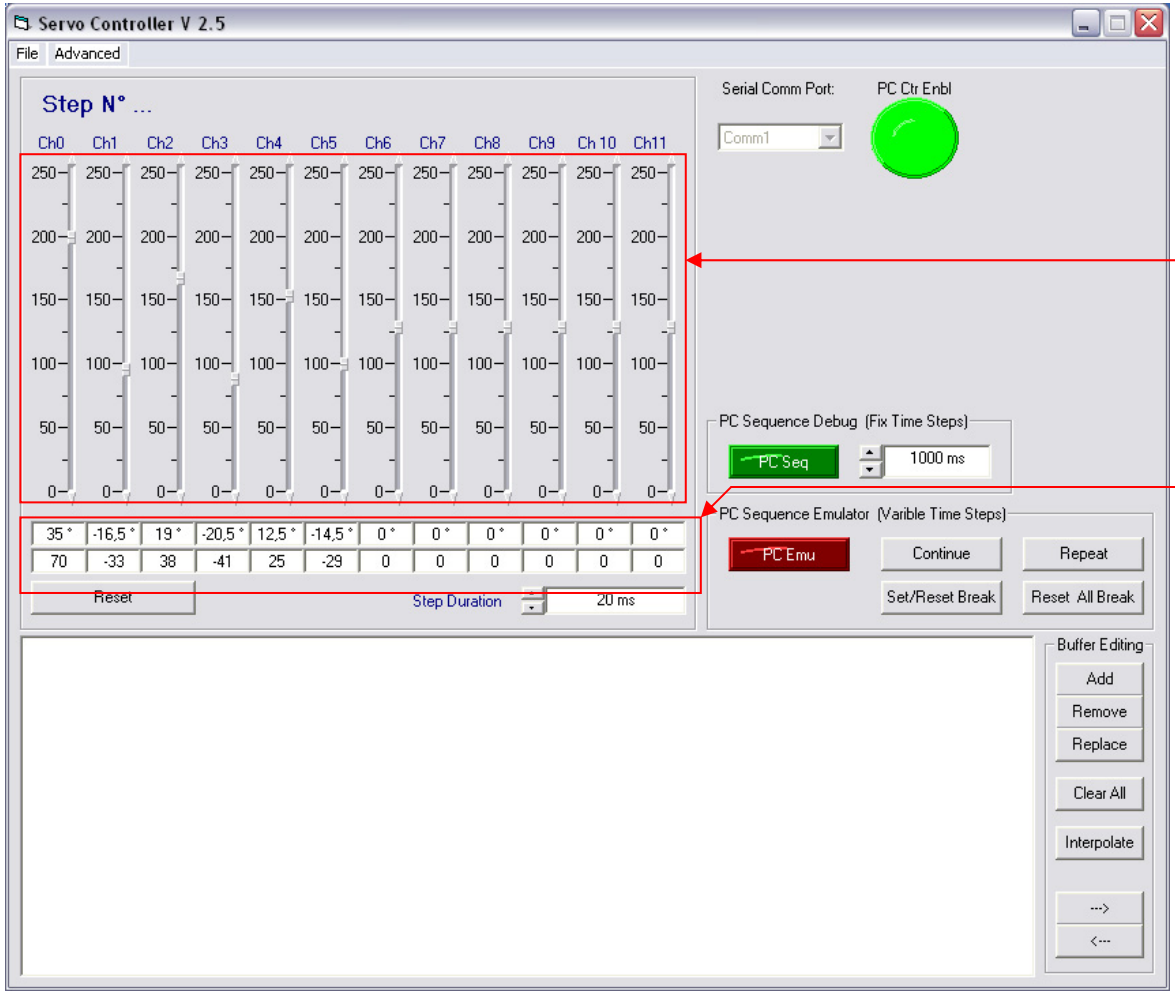


RS232 Comm Port Setting

Servo Driver board control enabled: servomotor are powered-on and can be controlled on line; the green LED of the Servo Driver board is ON when on line control is enable

“ServoController” software

Servomotor can be controlled to find positions and gaits

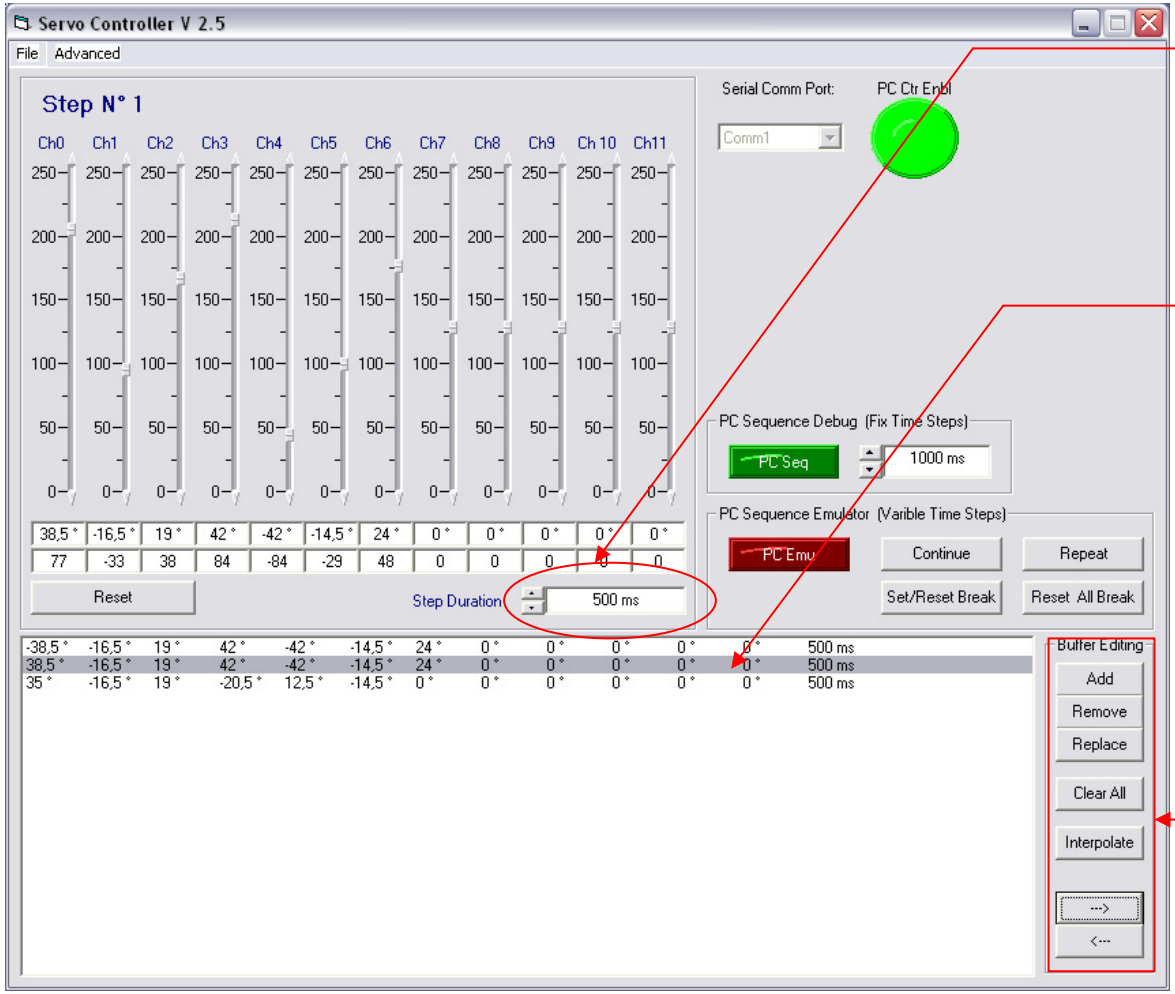


Each slider controls a servo-motor

Here are reported:
- the angles set on servomotors (upper part, only readable) and
- the relative binary values (lower part, readable and writable)

“ServoController” software

Sequences can be edited and memorised on the buffer step by step



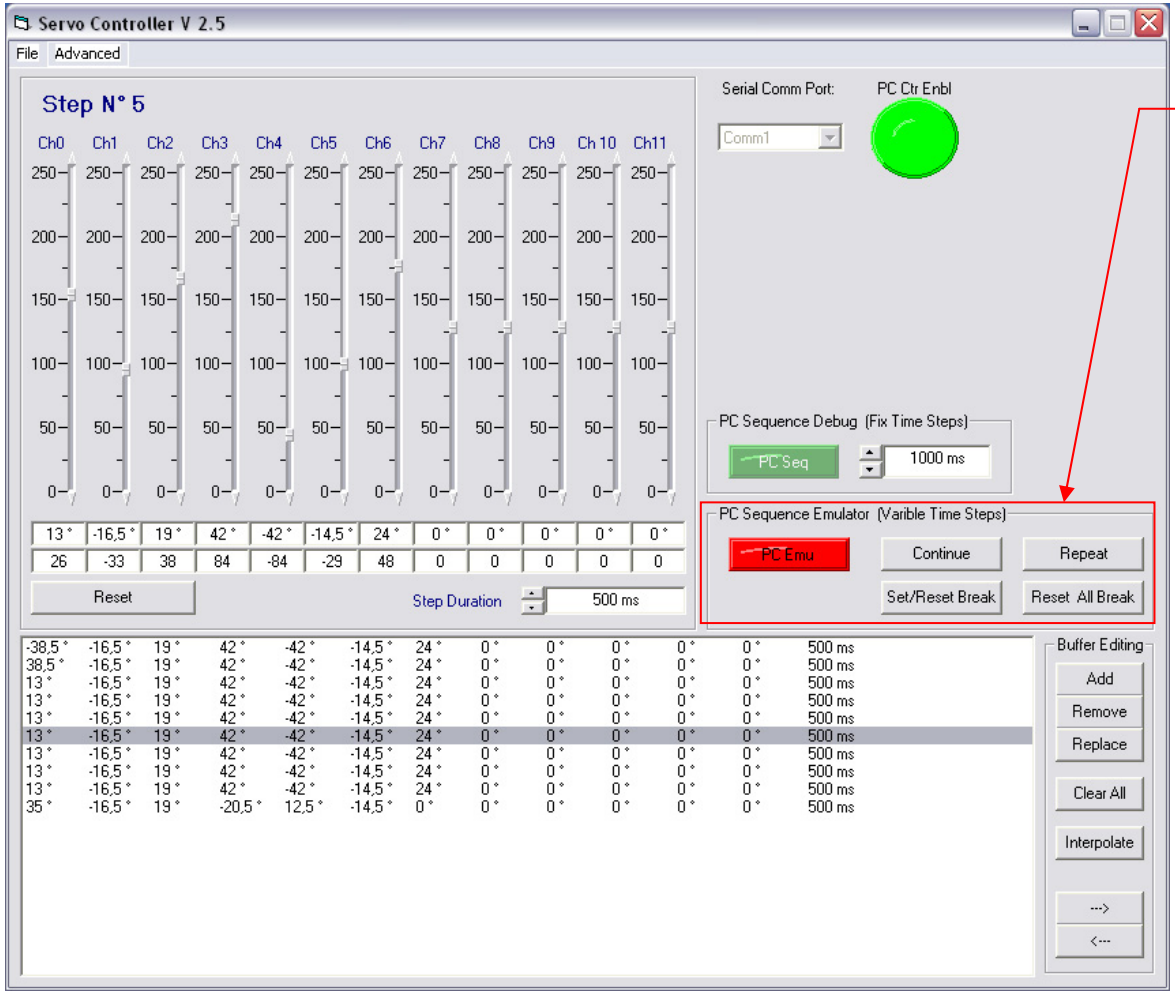
Step Time: this indicate the duration of selected/added step

Sequence buffer: the position of gaits can be exchanged by drag and drop; double click to select (execute) a step

Editing panel

“ServoController” software

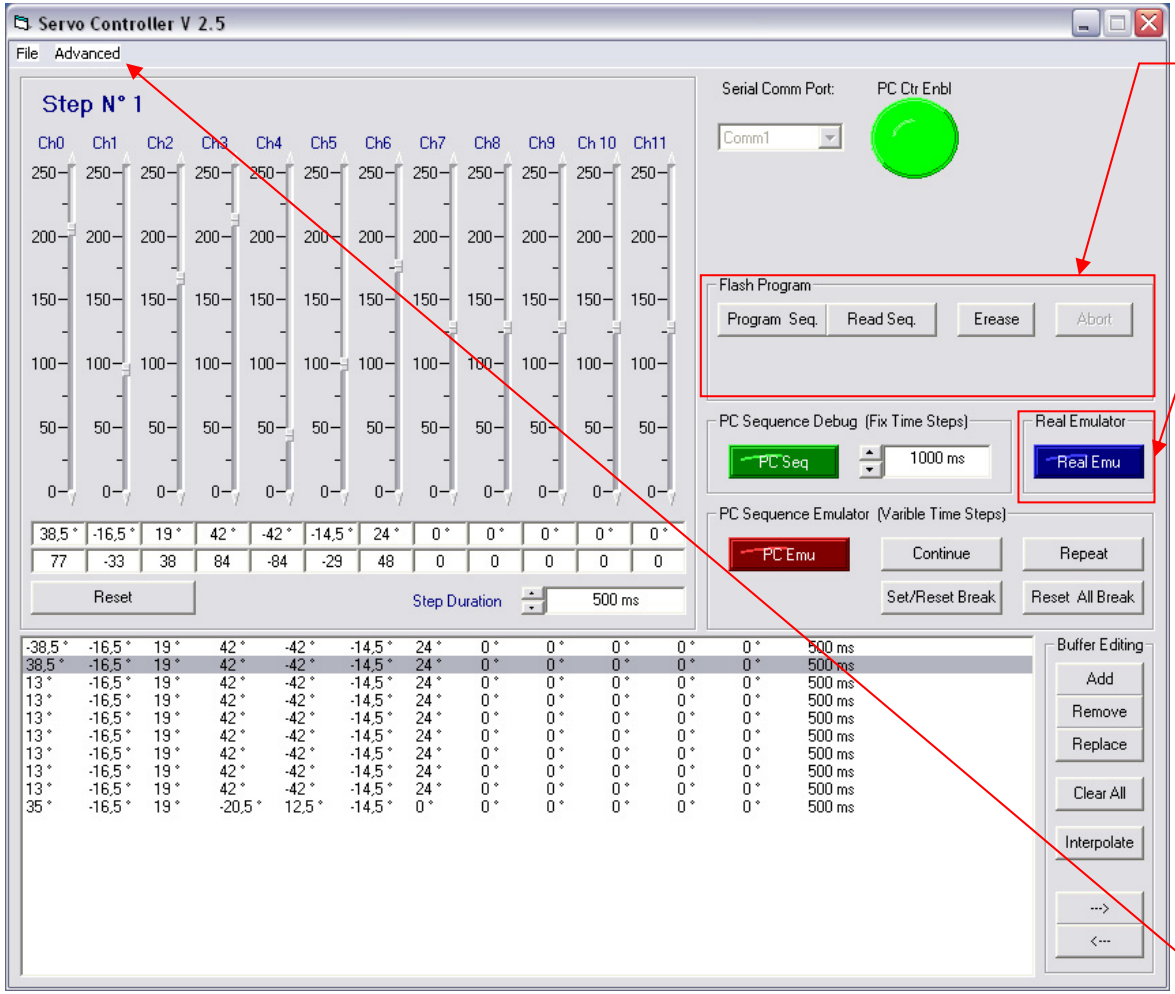
Sequences can be emulated (step are set at the specified times intervals)



PC Emulator panel

“ServoController” software

Once the sequence is ready, it can be loaded (programmed) on the ServoDriver board for stand alone working



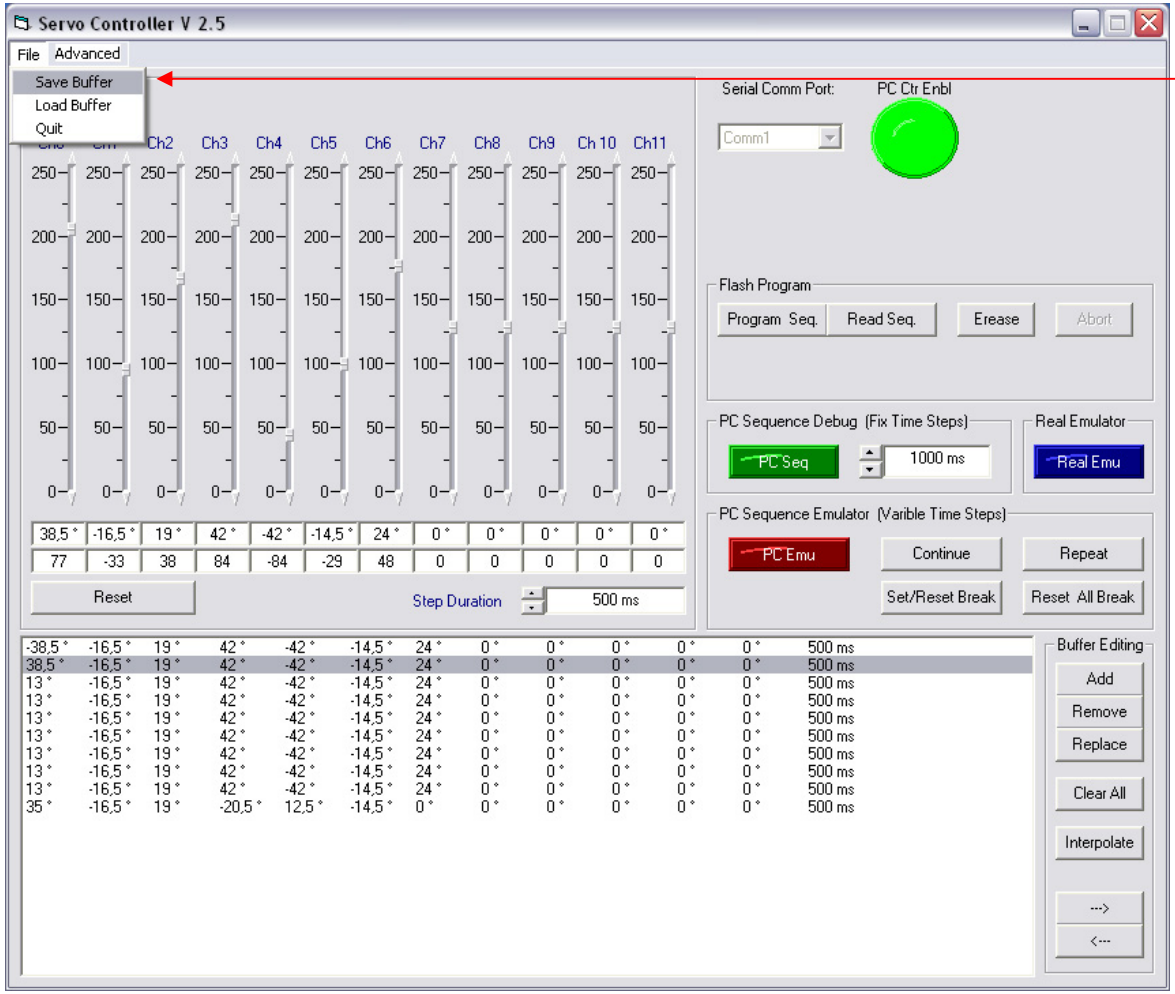
Flash Programming panel: it used for sequence downloading on Servo Driver Board

Real emulator: the sequence is run by the Servo Driver board (not by the PC program as in “PC emulation”).
The behaviour is the same of stand–alone operations (step are interpolated automatically for a more fluid movement; in “PC Emulation” steps are not interpolated).

Here Flash and Emulator Panels are selected

“ServoController” software

Sequence buffers can be saved and loaded on the PC



[Save / Load commands]

More information on

ServoMotor Driver Controller HW V2

&

ServoController PC SW V2.5

at

http://bdml.stanford.edu/twiki/bin/view/Main/ServoDriverController