Embedded System

L1

Sistemi elettronici di elaborazione digitale a microprocessore progettati appositamente per una determinata applicazione

Generalmente non riprogrammabili dall'utente per altri scopi,

spesso con una piattaforma hardware ad hoc,

- Minore potenza di elaborazione rispetto a GP
- Talvolta maggiore in applicazioni specifiche
- Minore consumo di potenza

- ASIC vs. Programmable:
 - ASIC: Application Specific Integrated Circuit, hardware dedicato, performance e costi elevati.
 - Programmable: (ARM, x86) maggiore flessibilità
 - Alternativa intermedia: FPGA
- Trend attuale: System/on/Chip (SoC)

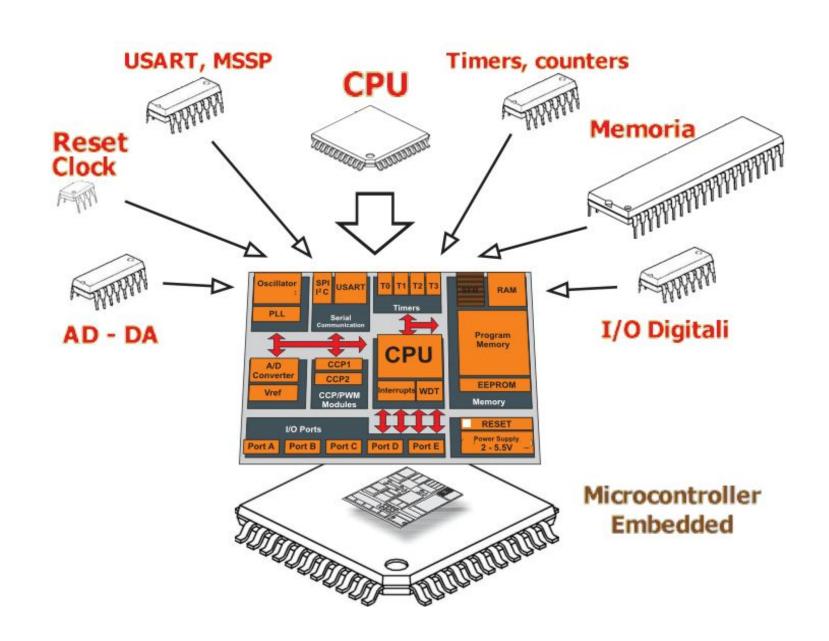
Microcontrollore

I microcontrollori sono microprocessori specializzati nelle applicazioni di controllo elettronico e sono presenti in tutti gli elettrodomestici e le apparecchiature moderne.

A differenza dei microprocessori di impiego generale, hanno al loro interno tutto quello che serve all'interfacciamento digitale ed analogico, cioè, ad esempio:

I/O, convertitori ADC e DAC, comparatori, interfacce di comunicazione RS485, RS232, USB.

microprocessore	microcontroller
il microprocessore ha bisogno di tutti i numerosi componenti esterni aggiuntivi per poter funzionare (memoria, oscillatore di clock, periferiche di ingresso/uscita ecc), richiedendo una certa superficie per la realizzazione di un sistema anche semplice ed un costo sensibile	Il microcontroller comprende in un solo elemento tutto quello che serve e può virtualmente funzionare praticamente senza elementi esterni; questo richiede il minimo di spazio e costo
il microprocessore si può espandere sui bus in maniera virtualmente illimitata, permettendo di realizzare sistemi di alta complessità	il microcontroller non ha veri e propri bus esterni per espandersi, in quanto la sua funzione è il controllo di di sistemi relativamente poco complessi
nei sistemi a microprocessore l'espandibilità consente di aggiungere memoria e periferiche a seconda delle necessità del sistema	il microcontroller non prevede, in genere, la possibilità di espandere memoria o periferiche ed il numero degli I/O e delle funzioni di controllo sul mondo esterno è forzatamente limitato al numero dei pin accessibili.



I microcontrollori PIC adottano
l'architettura Harvard,
proposta da Howard Aiken per lo
sviluppo dei computer Mark I, II, III e IV,
appunto all'università di Harvard:

usa memorie differenti per depositare dati e istruzioni.

Al contrario, l'architettura **Von Neumann,** proposta per sviluppare l'ENIAC (Electronic Numerical Integrator And Calculator) all'università della Pennsylvania durante la seconda guerra mondiale, usa la stessa memoria per dati e programmi.

Questa architettura usa meno linee ed è più economica, ma non sfrutta il parallelismo ed è quindi meno efficiente e veloce.

Harvard von-Neumann

Data
Memory 8 CPU 14 Program
Memory CPU 8 Program
And
Data
Memory Memory

Figure 4-1: Harvard vs. von Neumann Block Architectures

I PIC sono microcontrollori RISC (Reduced Instruction Set Computer)

hanno un set di poche istruzioni, fra 33 e 77, e lunghe fra 12 e 16 bit.

Altri microcontrollori sono invece CISC (Complex Instruction Set Computer)

Ci sono vantaggi e svantaggi per i microprocessori RISC e CISC di uso generale (CPU per Personal Computer, ecc.)

Tuttavia, si ritiene comunemente che i microcontrollori RISC siano più efficienti e veloci, anche se la programmazione è un po' più difficoltosa, pur se le istruzioni sono più semplici.

I PIC, come tutti i moderni microcontrollori, dispongono al loro interno di watchdog che provvede a un reset automatico quando un contatore interno di guardia (indipendente dal Program Counter) raggiunge la fine.

Se un programma funziona correttamente impedisce al contatore di raggiungere il massimo, azzerandolo periodicamente;

se invece il programma va in stallo oppure non viene attivato per un certo tempo, il watchdog attiva il reset.

Nei PIC tutte le istruzioni

non di salto sono eseguite in un ciclo macchina, pari a 4 periodi di clock;

quelle di salto in 2.

Se quindi il clock è a 4 MHz un'istruzione dura 1 microsecondo

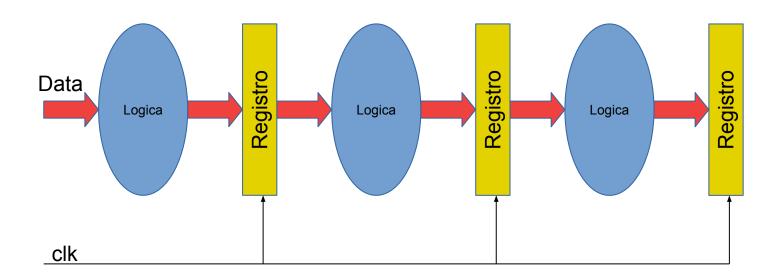
Se il clock è 40 MHz si eseguono ben 10 milioni di istruzioni al secondo.

I PIC sfruttano la tecnica del pipelining

mentre il microcontrollore interpreta un'istruzione (fase di fetch = interpretazione),

contemporaneamente ne esegue un'altra che era stata prima interpretata

I PIC sfruttano la tecnica del pipelining

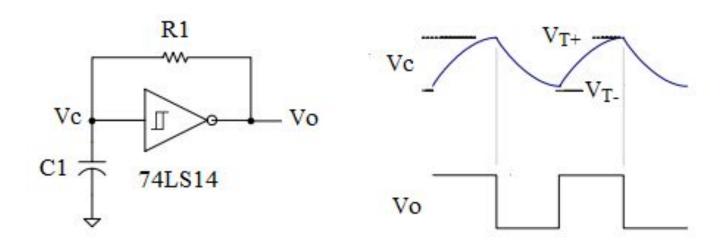


L'oscillatore che sincronizza tutte le operazioni, cioè il clock, può essere un circuito RC

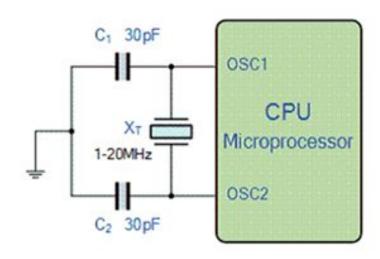
(si collegano ai 2 piedini una resistenza e un condensatore) o meglio un circuito al quarzo

(XTAL e due condensatori dell'ordine dei pico Farad) più stabile e preciso, o, infine, un clock esterno.

L'oscillatore che sincronizza tutte le operazioni, cioè il clock, può essere un circuito RC



un **circuito al quarzo**(XTAL e due condensatori dell'ordine dei pico Farad) più stabile e preciso



Il PIC memorizza in memoria interna non volatile EEPROM alcuni bit di configurazione:

Tipo di oscillatore, Watchdog attivato o disattivato, Protezione memoria programma e dati, Specifiche per il reset e l'alimentazione.

REGISTER 3-1: CONFIGURATION WORD FOR PIC16F627A/PIC16F628A/PIC16F648A (ADDRESS: 2007h)

R/P-1	U-1	U-1	U-1	U-1	R/P-1								
CP			F <u>=11</u>	_	CPD	LVP	BOREN	MCLRE	FOSC2	PWRTE	WDTE	FOSC1	FOSC0
bit 13	•			•	•								bit 0

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '1' P = Programmable
-n = Value at POR '1' = Bit is set '0' = Bit is cleared x = Bit is unknown

bit 13 CP: FLASH Program Memory Code Protection bit (PIC16F648A)

1 = Code protection off

1 = Code protection on

0 = 0000h to 0FFFh code-protected

(PIC16F628A)

1 = Code protection off

0 = 0000h to 07FFh code-protected

(PIC16F627A)

1 = Code protection off

o = 0000h to 03FFh code-protected

bit 12-9 Unimplemented: Read as '1'

bit 8 CPD: Data Code Protection bit(2)

1 = Data memory code protection off

o = Data memory code-protected

bit 7 LVP: Low Voltage Programming Enable bit

1 = RB4/PGM pin has PGM function, low-voltage programming enabled

0 = RB4/PGM is digital I/O, HV on MCLR must be used for programming

bit 6 BOREN: Brown-out Reset Enable bit(1)

1 = BOR enabled

0 = BOR disabled

bit 5 MCLRE: RA5/MCLR Pin Function Select bit

1 = RA5/MCLR pin function is MCLR

0 = RA5/MCLR pin function is digital I/O, MCLR internally tied to VDD

bit 3 PWRTE: Power-up Timer Enable bit(1)

1 = PWRT disabled

0 = PWRT enabled

bit 2 WDTE: Watchdog Timer Enable bit

1 = WDT enabled

o = WDT disabled

bit 4, 1-0 FOSC<2:0>: Oscillator Selection bits(3)

111 = RC oscillator: CLKOUT function on RA6/OSC2/CLKOUT pin, Resistor & Capacitor on RA7/OSC1/CLKIN

110 = RC oscillator: I/O function on RA6/OSC2/CLKOUT pin, Resistor & Capacitor on RA7/OSC1/CLKIN

101 = INTOSC internal oscillator: CLKOUT function on RA6/OSC2/CLKOUT pin, I/O function on RA7/OSC1/CLKIN

100 = INTOSC internal oscillator: I/O function on RA6/OSC2/CLKOUT pin, I/O function on RA7/OSC1/CLKIN

011 = EXTCLK: I/O function on RA6/OSC2/CLKOUT pin, I/O function on RA7/OSC1/CLKIN

010 = HS oscillator: High speed crystal/resonator on RA6/OSC2/CLKOUT and RA7/OSC1/CLKIN

= HS oscillator: High speed crystal/resonator on RA6/OSC2/CLKOUT and RA7/OSC1/CLKIN
 = XT oscillator: Crystal/resonator on RA6/OSC2/CLKOUT and RA7/OSC1/CLKIN

000 = LP oscillator: Low power crystal on RA6/OSC2/CLKOUT and RA7/OSC1/CLKIN

Note 1: Enabling Brown-out Reset does not automatically enable the Power-up Timer (PWRT).

2: Only a Bulk Erase will reset the Configuration Word, including the CP bits.

While MCLR is asserted in INTOSC mode, the internal clock oscillator is disabled.

REGISTER 3-1: CONFIGURATION WORD FOR PIC16F627A/PIC16F628A/PIC16F648A (ADDRESS: 2007h)

R/P-1	U-1	U-1	U-1	U-1	R/P-1								
CP			-	_	CPD	LVP	BOREN	MCLRE	FOSC2	PWRTE	WDTE	FOSC1	FOSC0
bit 13					-					M11.			bit 0

Legend:				
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '1'	P = Programmable	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown	

(PIC16F6: 1 = Code 0 = 0000h (PIC16F6: 1 = Code 0 = 0000h (PIC16F6: 1 = Code	CP: FLASH Program Memory Code Protection bit (PIC16F648A)	bit 5	MCLRE: RA5/MCLR Pin Function Select bit 1 = RA5/MCLR pin function is MCLR 0 = RA5/MCLR pin function is digital I/O, MCLR internally tied to VDD				
	1 = Code protection off 0 = 0000h to 0FFFh code-protected (PIC16F628A)	bit 3	PWRTE: Power-up Timer Enable bit ⁽¹⁾ 1 = PWRT disabled 0 = PWRT enabled				
	1 = Code protection off 0 = 0000h to 07FFh code-protected	bit 2	WDTE: Watchdog Timer Enable bit 1 = WDT enabled 0 = WDT disabled				
	(PIC16F627A) 1 = Code protection off 0 = 0000h to 03FFh code-protected	bit 4, 1-0	FOSC<2:0>: Oscillator Selection bits ⁽³⁾ 111 = RC oscillator: CLKOUT function on RA6/OSC2/CLKOUT pin, Resistor & Capacitor on RA7/OSC1/CLKIN 110 = RC oscillator: I/O function on RA6/OSC2/CLKOUT pin, Resistor & Capacitor on RA7/OSC1/CLKIN 101 = INTOSC internal oscillator: CLKOUT function on RA6/OSC2/CLKOUT pin, I/O function on RA7/OSC1/CLKIN 100 = INTOSC internal oscillator: I/O function on RA6/OSC2/CLKOUT pin, I/O function on RA7/OSC1/CLKIN				
bit 12-9	Unimplemented: Read as '1'		011 = EXTCLK: I/O function on RA6/OSC2/CLKOUT pin, I/O function on RA7/OSC1/CLKIN 010 = HS oscillator: High speed crystal/resonator on RA6/OSC2/CLKOUT and RA7/OSC1/CLKIN				
	CPD: Data Code Protection bit ⁽²⁾		001 = XT oscillator: Crystal/resonator on RA6/OSC2/CLKOUT and RA7/OSC1/CLKIN 000 = LP oscillator: Low power crystal on RA6/OSC2/CLKOUT and RA7/OSC1/CLKIN				
	1 = Data memory code protection off 0 = Data memory code-protected		Only a Bulk Erase will reset the Configuration Word, including the CP bits.				
bit 7	LVP: Low Voltage Programming Enable bit 1 = RB4/PGM pin has PGM function, low-voltage pro 0 = RB4/PGM is digital I/O, HV on MCLR must be used.	ogramm	ing enabled				
bit 6	BOREN: Brown-out Reset Enable bit ⁽¹⁾ 1 = BOR enabled						

0 = BOR disabled

Feature	SYMBOLS			
	_RC_OSC			
	EXTRC OSC			
	EXTRC_OSC_CLKOUT			
	EXTRC OSC NOCLKOUT			
On villators	_INTRC_OSC			
Oscillators	_INTRC_OSC_CLKOUT			
	INTRC_OSC_NOCLKOUT			
	_LP_OSC			
	_XT_OSC			
	_HS_OSC			
W.L.B. T	_WDT_ON			
Watch Dog Timer	WDT_OFF			
Power-up Timer	PWRTE ON			
	_PWRTE_OFF			
D	BODEN ON			
Brown-out Reset	BODEN OFF			
Master Class Freshla	_MCLRE_ON			
Master Clear Enable	_MCLRE_OFF			
	_CP_ALL			
	_CP_ON			
Code Protect	_CP_75			
	_CP_50			
	_CP_OFF			
Code Destart Data EEDDOM	_DP_ON			
Code Protect Data EEPROM	_DP_OFF			
Code Protect Calibration Succession	_CPC_ON			
Code Protect Calibration Space	_CPC_OFF			

Note 1: Not all configuration bit symbols may be available on any one device. Please refer to the MIcrochip include file of that device for available symbols.

I PIC possono essere collocati in modalità dormiente (sleep mode) con risparmio energetico, ed essere risvegliati all'occorrenza.

I PIC sono caratterizzati anche dall'uso distinto della

memoria Dati (RAM volatile e/o EEPROM)

e memoria programma (per lo più FLASH).

(Harvard)

I PIC si possono collocare in 3 famiglie:

PIC a basso, medio e alto livello a 8 bit (il livello è dato dal numero di istruzioni);

PIC24 a 16 bit e

PIC32 a 32 bit.

I PIC a 8 bit trattano direttamente dati lunghi 8 bit.

Quelli a 16 trattano dati a 16 bit, ecc.

PIC a basso livello

Hanno 33 istruzioni lunghe 12 bit ciascuna. La memoria programma arriva fino a 2 K parole di 12 bit.

Sono dispositivi con 6, 8, 10, 20, 28 piedini.

PIC a medio livello

Hanno 35 istruzioni ognuna lunga 14 bit.

La memoria programmi arriva a 8 K, organizzata in pagine di 2 K parole.

La memoria dati è organizzata in banchi di 120 registri a 8 bit.

Gestiscono interrupt interni fissi e un interrupt esterno.

Hanno diverse porte GPIO, fino a 3 timer, porte seriali, parecchi convertitori A/D, comparatori e altre interfacce.

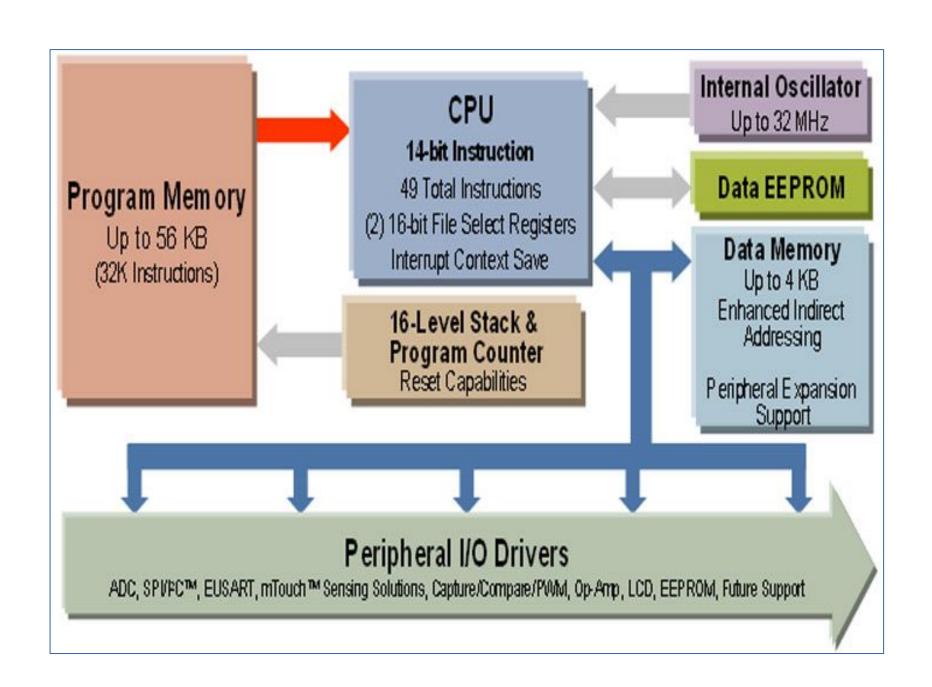
PIC a medio livello

Il Program Counter (PC) ha 13 bit e quindi può indirizzare fino a 8k word di memoria Programma.

Poiché la memoria è suddivisa in pagine di 2k, il bit 12 e il bit 11 del PC indicano una delle 4 pagine, mentre i bit da 0 a 10 danno l'indirizzo all'interno della pagina.

PIC a medio livello

Pagina 0	Pagina 1	Pagina 2	Pagina 3
0h	800h	1000h	1800h
7FFh	FFFh	17FFh	1FFFh



High-Performance RISC CPU:

- Operating speeds from DC 20 MHz
- Interrupt capability
- 8-level deep hardware stack
- 35 single-word instructions
- 3.5KB program memory
- 224 Byte RAM

Special Microcontroller Features:

- Internal and external oscillator options:
 - Precision internal 4MHz oscillator factory calibrated to ±1%
 - Low-power internal 48kHz oscillator
 - External Oscillator support for crystals and resonators

Special Microcontroller Features:

- Power-saving Sleep mode
- Programmable weak pull-ups on PORTB
- Watchdog Timer with independent oscillator
- In-Circuit Serial Programming
- Programmable code protection

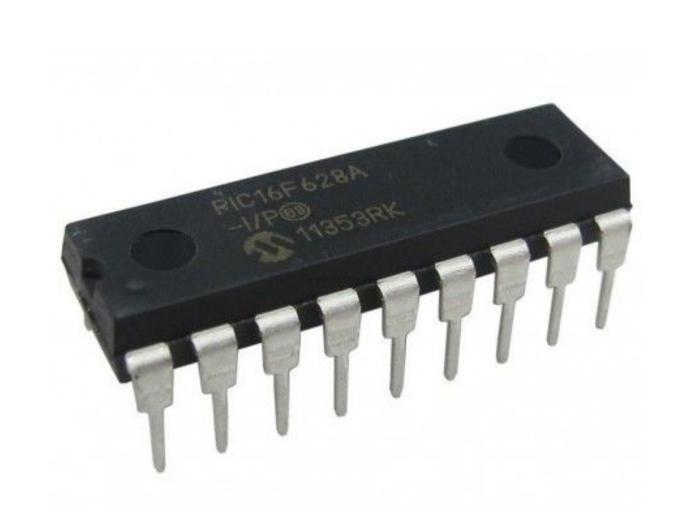
Special Microcontroller Features:

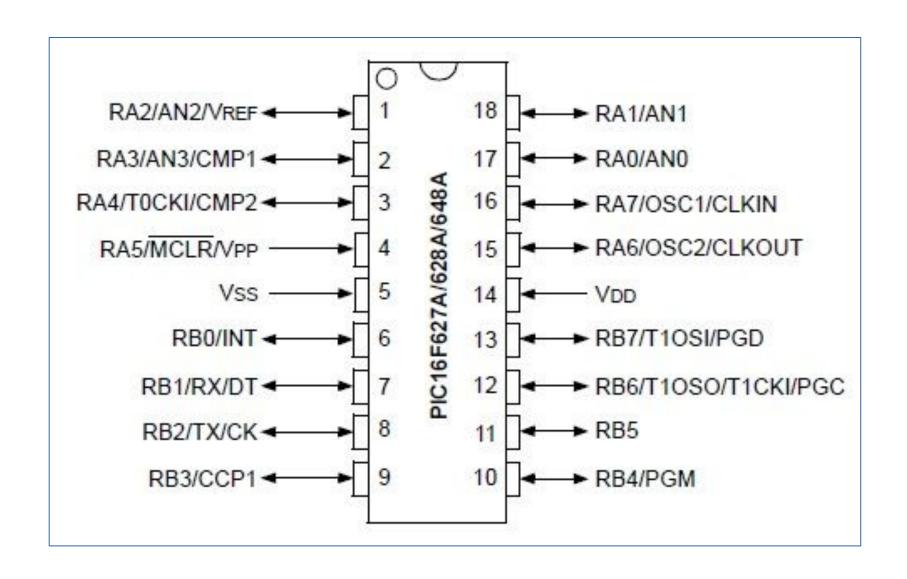
High-Endurance Flash/EEPROM cell:

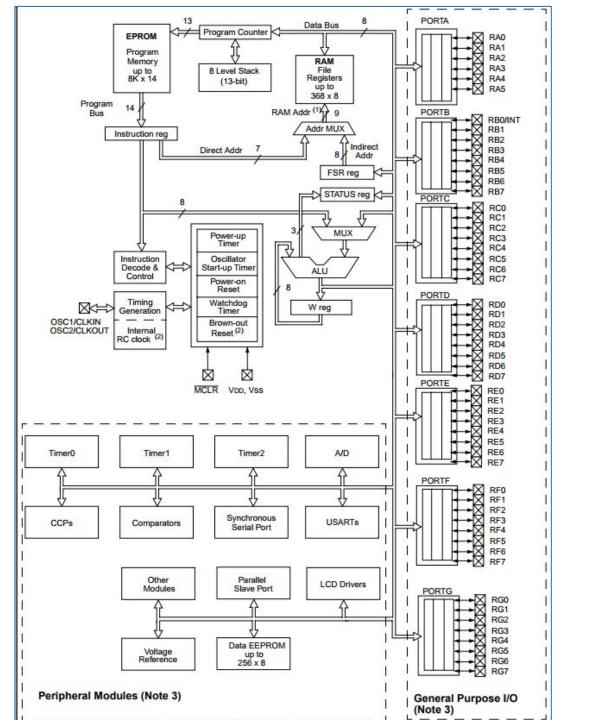
- 100,000 write Flash endurance
- 1,000,000 write EEPROM endurance
- 40 year data retention

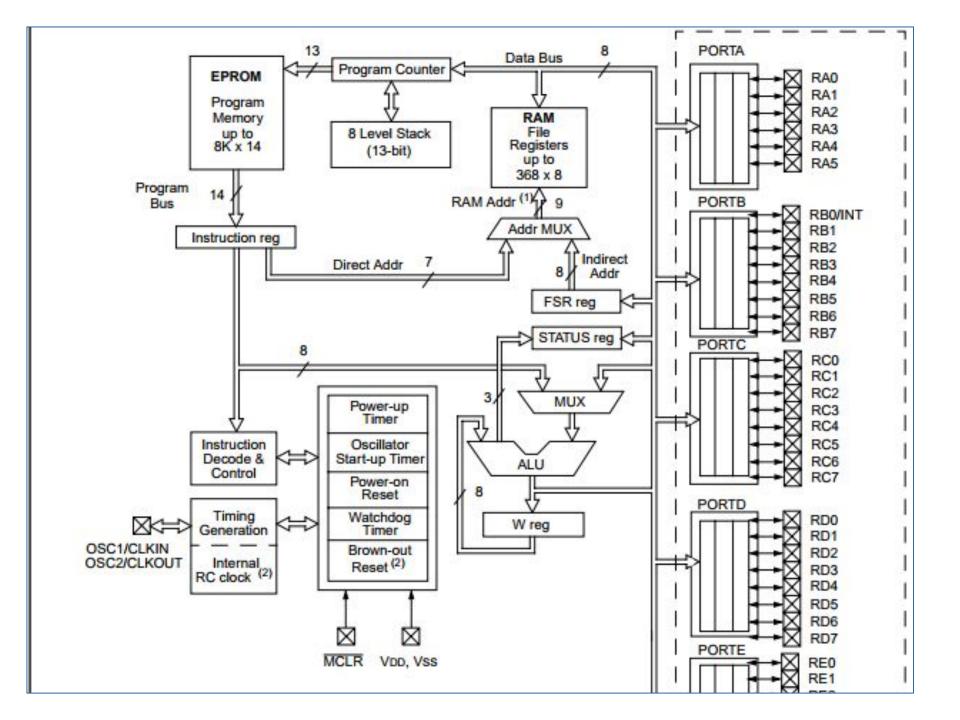
Peripheral Features:

- 16 I/O pins with individual direction control
- Analog comparator module with:
 - Two analog comparators
 - Programmable on-chip voltage reference (VREF) module
 - Selectable internal or external reference
 - Comparator outputs are externally accessible









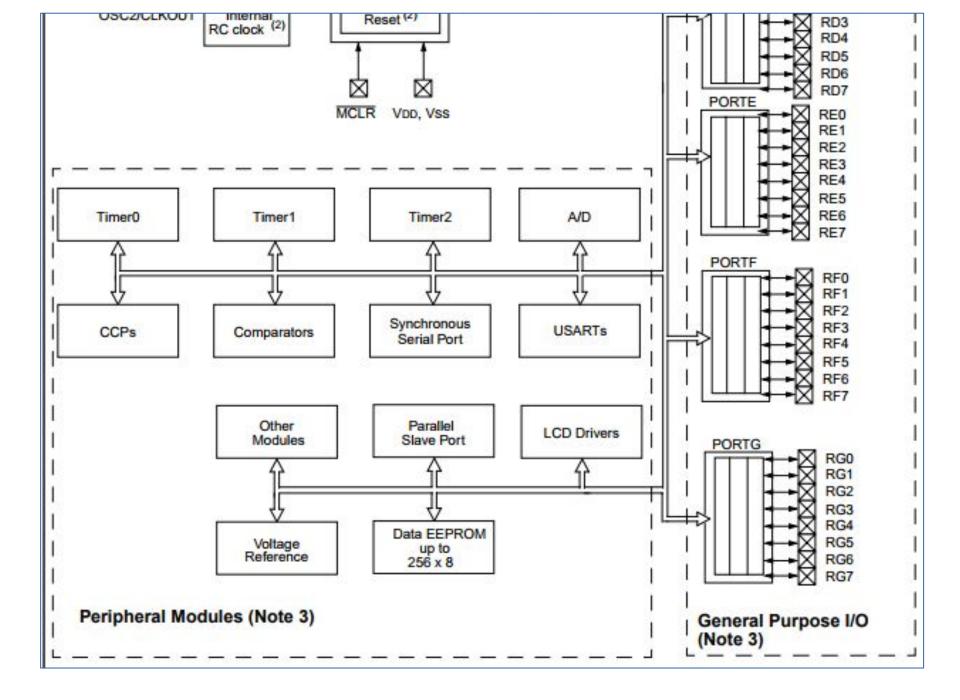


Table 5-1: Mid-Range MCU Instruction Set

Mnemonic, Operands				14	14-Bit Instruction Word							
		Description	Cycles	MSb			LSb	Bits Affected	Note			
BYTE-ORIENTED FILE REGISTER OPERATIONS												
ADDWF	f, d	Add W and f	1	00	0111	dfff	ffff	C,DC,Z	1,2			
ANDWF	f, d	AND W with f	1	00	0101	dfff	ffff	Z	1,2			
CLRF	f	Clear f	1	00	0001	lfff	ffff	Z	2			
CLRW	-	Clear W	1	00	0001	0xxx	XXXX	Z	570,5963.2			
COMF	f, d	Complement f	1	00	1001	dfff	ffff	Z	1,2			
DECF	f, d	Decrement f	1	00	0011	dfff	ffff	Z	1,2			
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	00	1011	dfff	ffff		1,2,3			
INCF	f, d	Increment f	1	00	1010	dfff	ffff	Z	1,2			
INCFSZ	f, d	Increment f, Skip if 0	1(2)	00	1111	dfff	ffff		1,2,3			
IORWF	f, d	Inclusive OR W with f	1	00	0100	dfff	ffff	Z	1,2			
MOVF	f, d	Move f	1	00	1000	dfff	ffff	Z	1,2			
MOVWF	f	Move W to f	1	00	0000	lfff	ffff		10.0000000			
NOP	-	No Operation	1	00	0000	0xx0	0000					
RLF	f, d	Rotate Left f through Carry	1	00	1101	dfff	ffff	C	1,2			
RRF	f, d	Rotate Right f through Carry	1	00	1100	dfff	ffff	C	1,2			
SUBWF	f, d	Subtract W from f	1	00	0010	dfff	ffff	C,DC,Z	1,2			
SWAPF	f, d	Swap nibbles in f	1	00	1110	dfff	ffff	sort you to distin	1,2			
XORWF	f, d	Exclusive OR W with f	1	00	0110	dfff	ffff	Z	1,2			
BIT-ORIENT	ED FILE	REGISTER OPERATIONS										
BCF	f, b	Bit Clear f	1	01	00bb	bfff	ffff		1,2			
BSF	f, b	Bit Set f	1	01	01bb	bfff	ffff		1,2			
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	01	10bb	bfff-	ffff		3			
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	01	11bb	bfff	ffff		3			
LITERAL A	ND CON	TROL OPERATIONS		-								
ADDLW	k	Add literal and W	1	11	111x	kkkk	kkkk	C,DC,Z				
ANDLW	k	AND literal with W	1	11	1001	kkkk	kkkk	Z				
CALL	k	Call subroutine	2	10	0kkk	kkkk	kkkk					
CLRWDT		Clear Watchdog Timer	1	00	0000	0110	0100	TO,PD				
GOTO	k	Go to address	2	10	1kkk	kkkk	kkkk					
IORLW	k	Inclusive OR literal with W	1	11	1000	kkkk	kkkk	Z				
MOVLW	k	Move literal to W	1	11	00xx	kkkk	kkkk	est.				
RETFIE		Return from interrupt	2	00	0000	0000	1001					
RETLW	k	Return with literal in W	2	11	01xx	kkkk	kkkk					
RETURN	-	Return from Subroutine	2	00	0000	0000	1000					
SLEEP	-	Go into standby mode	1	00	0000	0110	0011	TO.PD				
SUBLW	k	Subtract W from literal	1	11	110x	kkkk	kkkk	C,DC,Z				
XORLW	k	Exclusive OR literal with W	1	11	1010	kkkk	kkkk	7				

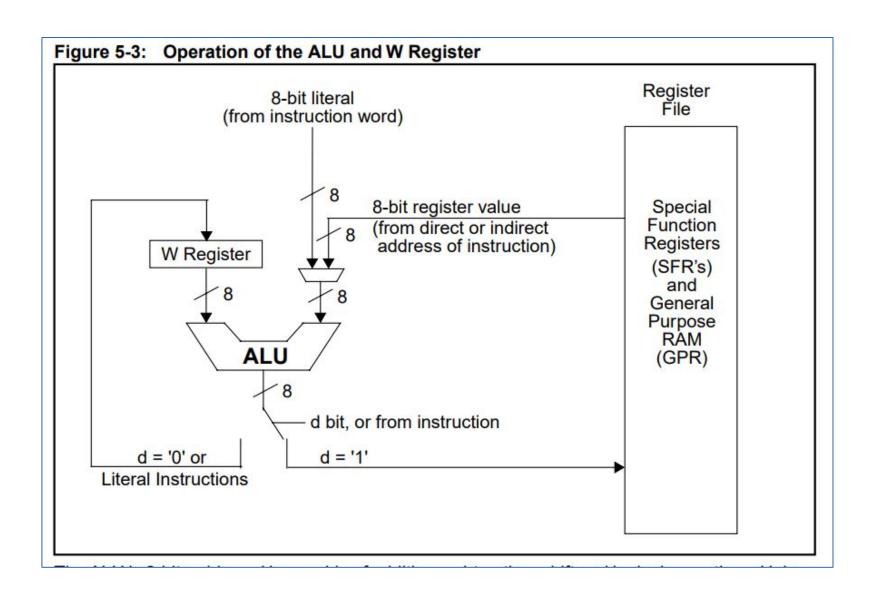
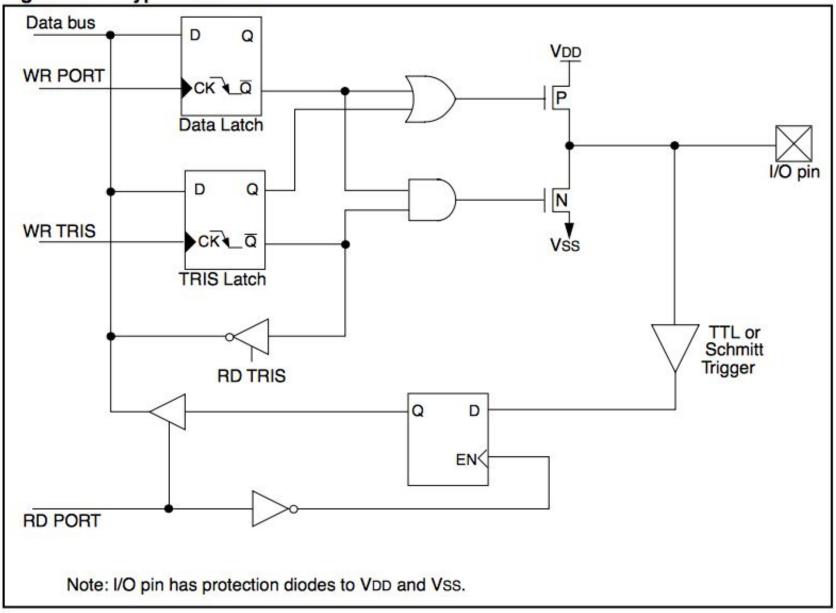


Figure 9-1: Typical I/O Port



Ambiente di lavoro

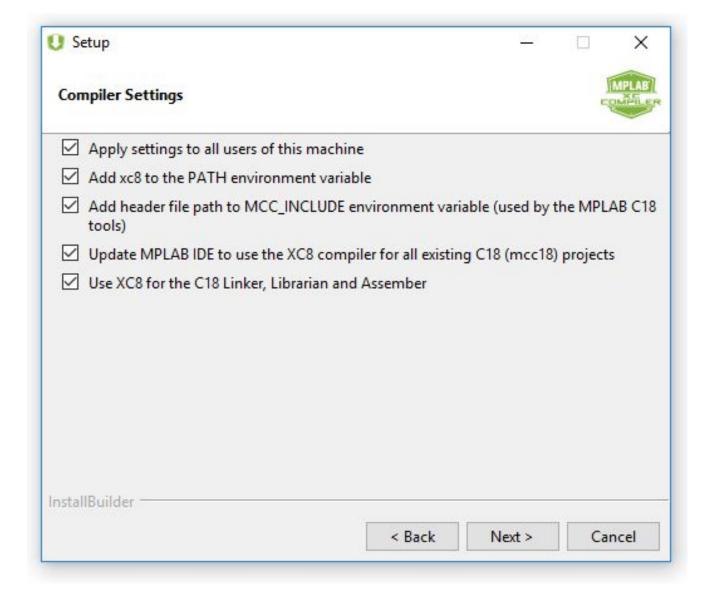
https://www.microchip.com/wwwproducts/en/PIC16F628A

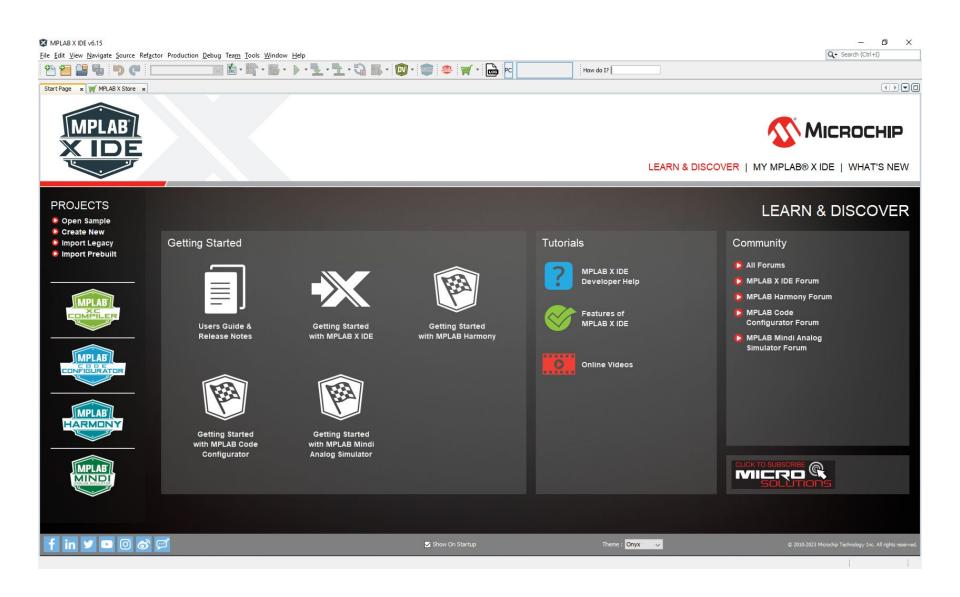
http://www.microchip.com/mplab/mplab-x-ide
Versione attuale 6.15

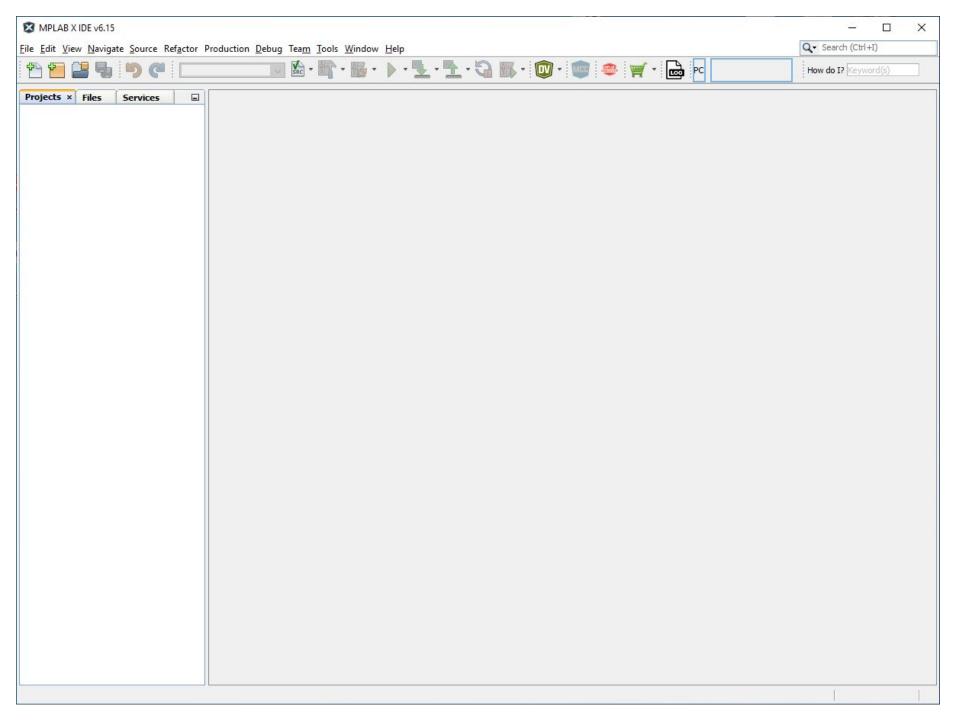
http://www.microchip.com/mplab/compilers
Scaricare MPLAB® XC8 Compiler v2.45

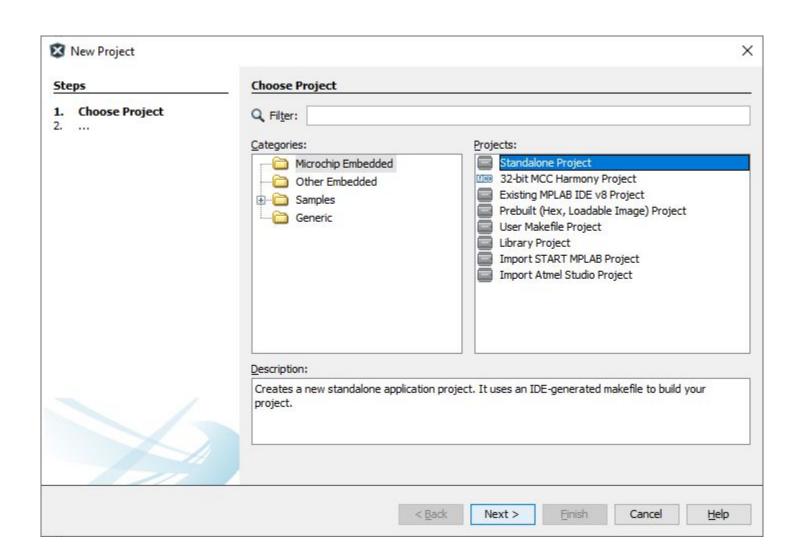
https://sourceforge.net/projects/picsim/ Versione 0.9.00

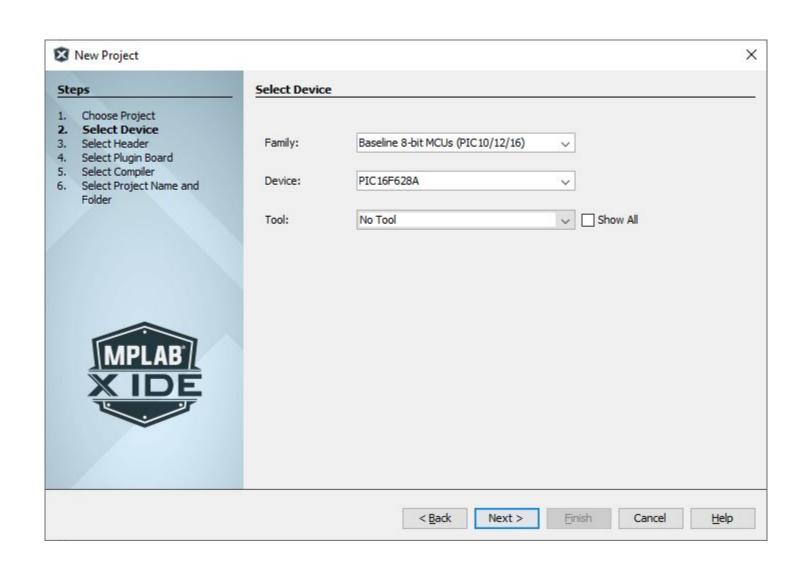
Durante installazione XC8

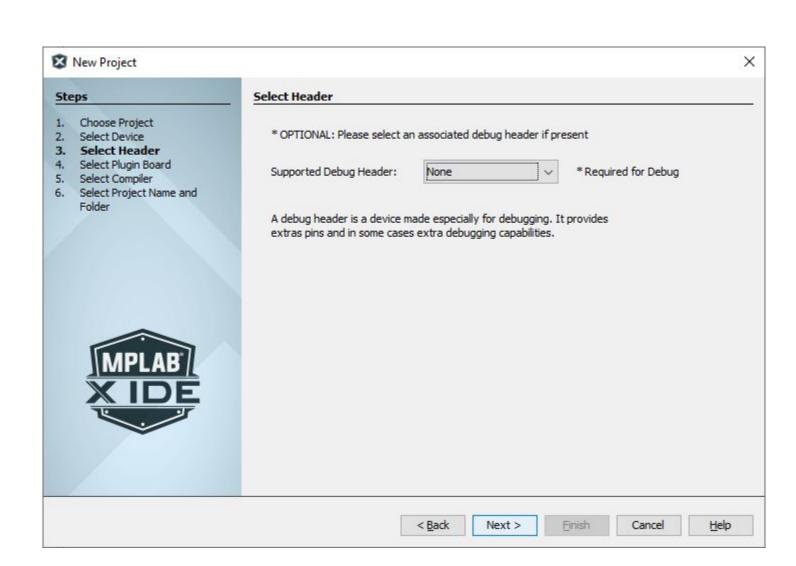


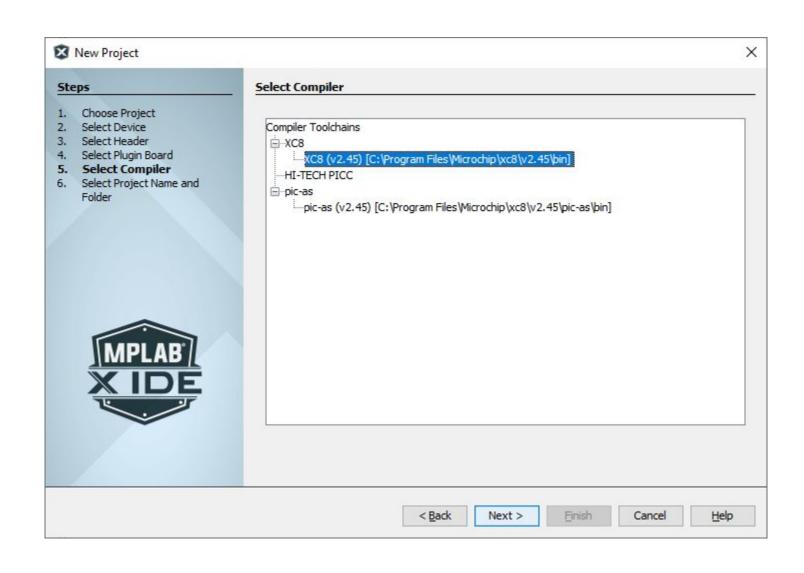


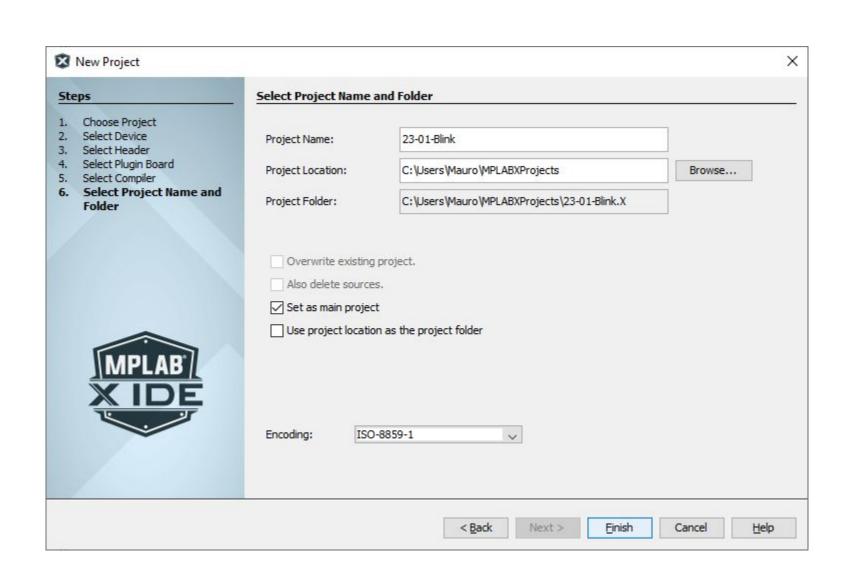


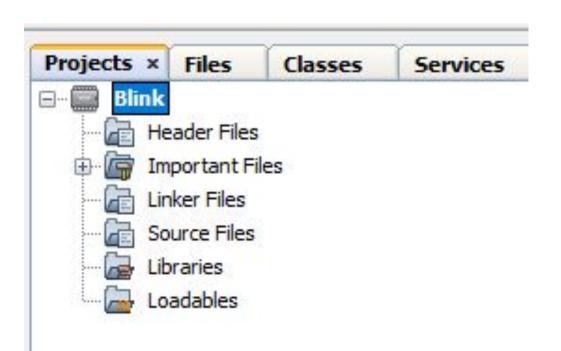


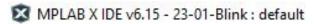




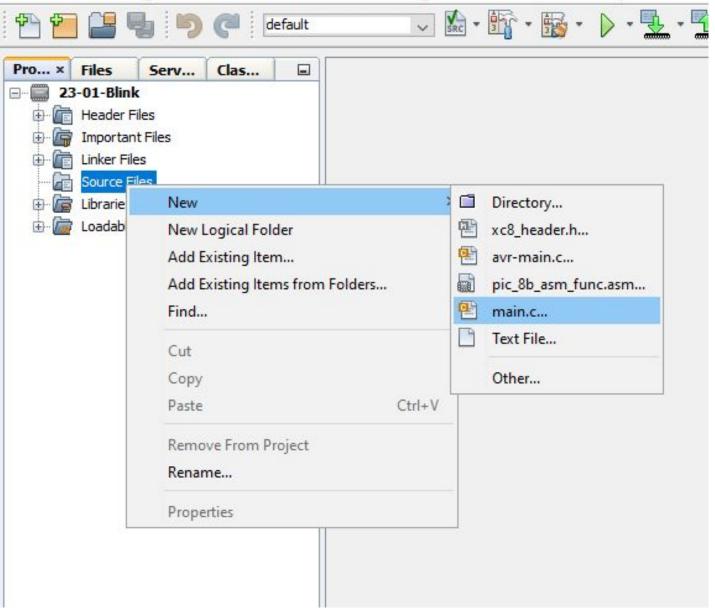


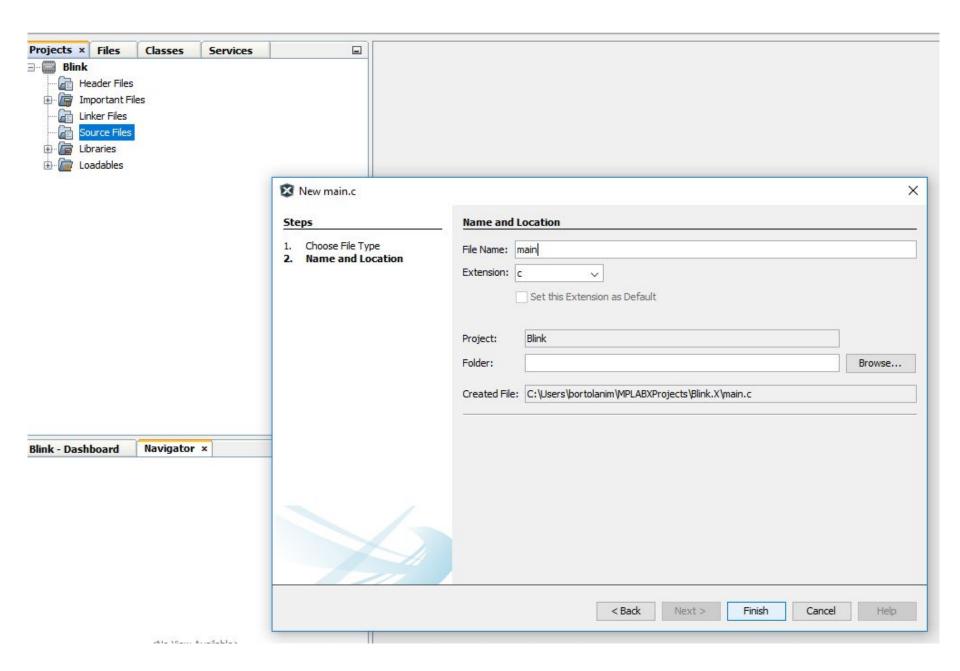






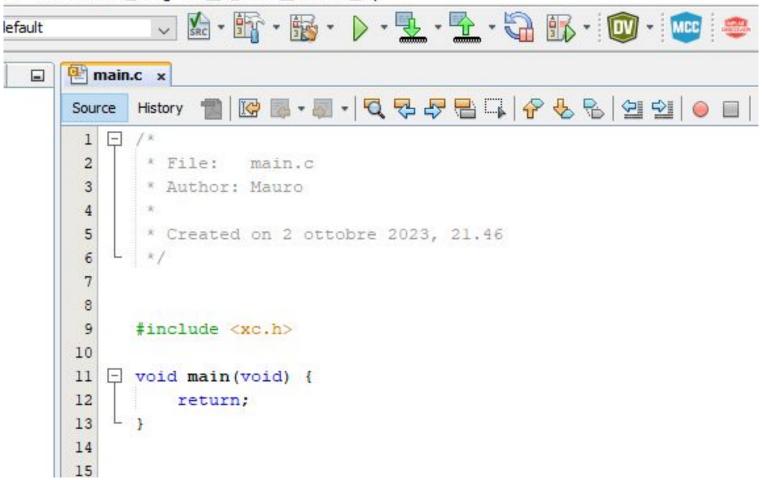
File Edit View Navigate Source Refactor Production Debug Team Tools Window Help





ault

tor Production Debug Team Tools Window Help

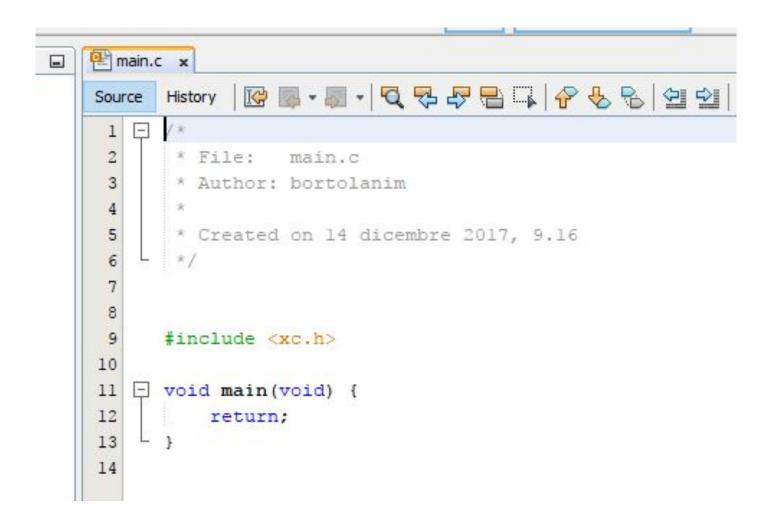


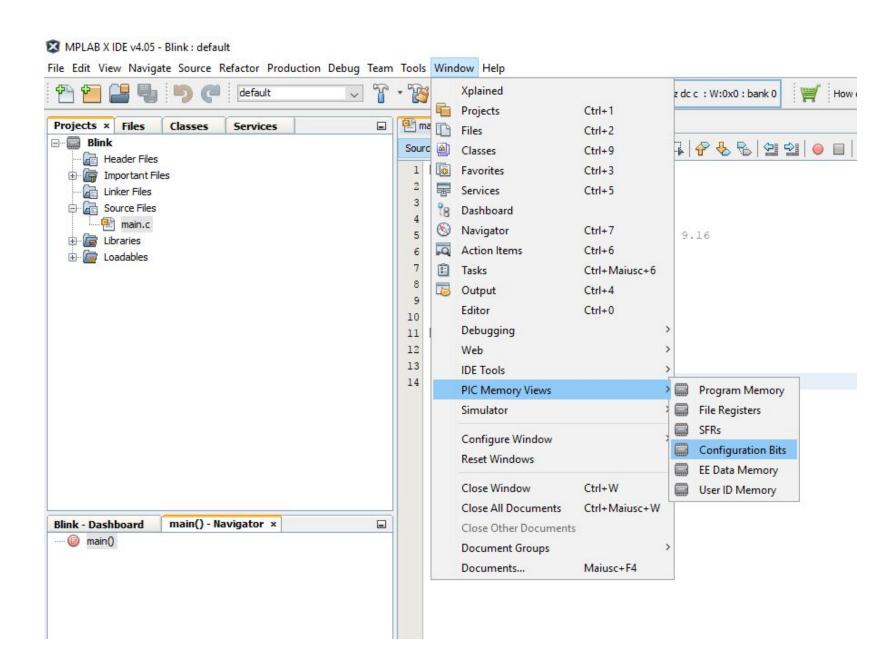
In grassetto -> "non salvato" ault tor Production Debut Team Tools Window Help lefault main.c x Source -2 * File: main.c 3 * Author: Mauro 4 * Created on 2 ottobre 2023, 21.46 5 6 7 8 9 #include <xc.h> 10 11 void main(void) { 12 return; 13 14 15

Compila solo i file modificati (Build Main Project) Ri-Compila Tutto main.c History Source main.c

* Author: bortolanim

3





IL - DIIIK (DUI	iu, Lodu j	COI	iigurauvi	I DILS A	
Address	Name	Value	Field	Option	Category
2007	CONFIG	FF62	FOSC	HS V	Oscillator Selection bits
			WDTE	OFF	Watchdog Timer Enable bit
			PWRTE	ON	Power-up Timer Enable bit
			MCLRE	ON	RA5/MCLR/VPP Pin Function Select bit
			BOREN	ON	Brown-out Detect Enable bit
			LVP	OFF	Low-Voltage Programming Enable bit
			CPD	OFF	Data EE Memory Code Protection bit
			CP	OFF	Flash Program Memory Code Protection bit



Output × Configuration Bits

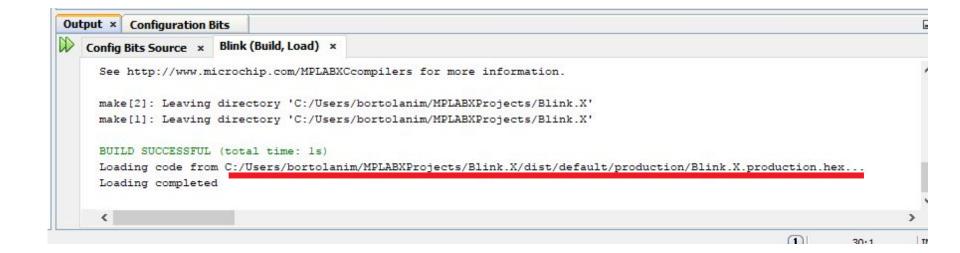
```
Blink (Build, Load) x Config Bits Source x
```

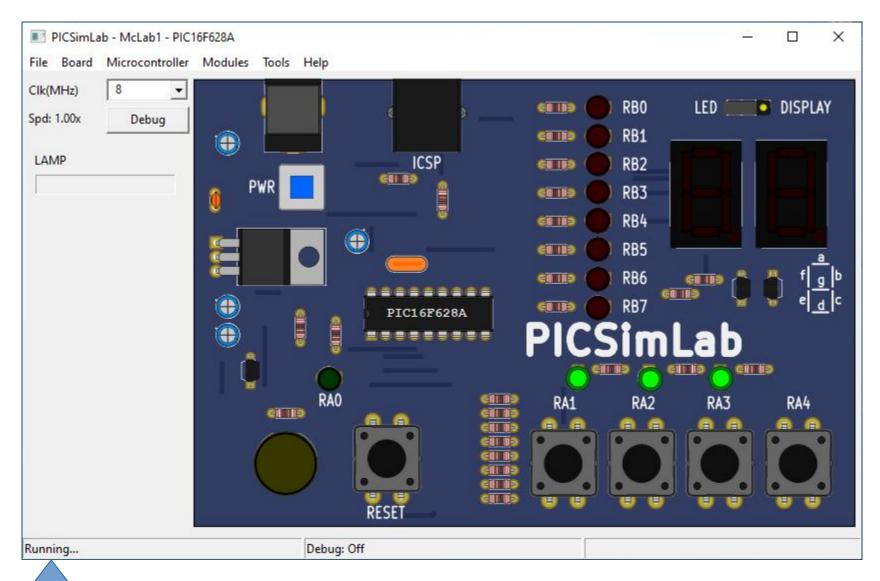
```
// PIC16F628A Configuration Bit Settings
// 'C' source line config statements
// CONFIG
#pragma config FOSC = HS
                              // Oscillator Selection bits (HS oscillator: High-spe
                              // Watchdog Timer Enable bit (WDT disabled)
#pragma config WDTE = OFF
#pragma config PWRTE = ON
                              // Power-up Timer Enable bit (PWRT enabled)
#pragma config MCLRE = ON
                              // RA5/MCLR/VPP Pin Function Select bit (RA5/MCLR/VP)
#pragma config BOREN = ON
                              // Brown-out Detect Enable bit (BOD enabled)
#pragma config LVP = OFF
                              // Low-Voltage Programming Enable bit (RB4/PGM pin ha
#pragma config CPD = OFF
                              // Data EE Memory Code Protection bit (Data memory co
#pragma config CP = OFF
                             // Flash Program Memory Code Protection bit (Code pro
// #pragma config statements should precede project file includes.
// Use project enums instead of #define for ON and OFF.
#include <xc.h>
```

```
Source History | 😭 👨 + 🐻 + 🔩 🐶 😓 📮 📮 🖟 🐶 😓 | 🖭 😂 | 🍅 🔲 | 👑 🚅 🚱
1 - /*
      * File: main.c
      * Author: bortolanim
 3
 4
      * Created on 14 dicembre 2017, 9.16
 5
 6
  - // PIC16F628A Configuration Bit Settings
9
    // 'C' source line config statements
10
11
    // CONFIG
12
     #pragma config FOSC = HS
                                // Oscillator Selection bits (HS oscillator: Hi
13
     #pragma config WDTE = OFF
                                // Watchdog Timer Enable bit (WDT disabled)
14
     #pragma config PWRTE = ON
                                 // Power-up Timer Enable bit (PWRT enabled)
15
                                 // RA5/MCLR/VPP Pin Function Select bit (RA5/MC
     #pragma config MCLRE = ON
16
     17
    #pragma config LVP = OFF // Low-Voltage Programming Enable bit (RB4/PGM :
18
     #pragma config CPD = OFF // Data EE Memory Code Protection bit (Data mem
19
     #pragma config CP = OFF
                                 // Flash Program Memory Code Protection bit (Co
20
21
22
  #pragma config statements should precede project file includes.
    // Use project enums instead of #define for ON and OFF.
23
24
25
     #include <xc.h>
26
  void main(void) {
27
28
        return;
29
30
```



A noi serve il file .hex che è stato generato dal compilatore





File → Load Hex

PORTB è un registro che accede direttamente alla porta di uscita del microcontrollore. È un registro a 8 bit, ogni singolo bit indicizza direttamente un filo di uscita

RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
0x80	0x40	0x20	0x10	0x08	0x04	0x02	0x01

TRISB è un registro che configura il funzionamento dei singoli bit di PORTB.

1 -> Input 0 -> Output

TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0
0x80	0x40	0x20	0x10	0x08	0x04	0x02	0x01

PORTA è un registro che accede direttamente alla porta di uscita del microcontrollore. È un registro a 8 bit, ogni singolo bit indicizza direttamente un filo di uscita

RA5, RA6, RA7 normalmente non disponibili

RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0
X	X	X					
0x80	0x40	0x20	0x10	0x08	0x04	0x02	0x01

TRISA è un registro che configura il funzionamento dei singoli bit di PORTA.

1 -> Input 0 -> Output

TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0
0x80	0x40	0x20	0x10	0x08	0x04	0x02	0x01

Esempio:

TRISB = 0x7C; // 7C in esadecimale equivale alla configurazione binaria 01111100

TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0
0	1	1	1	1	1	0	0
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
	1	1	1	1	1		
Ο	ı	1	1	I	I	0	Ο

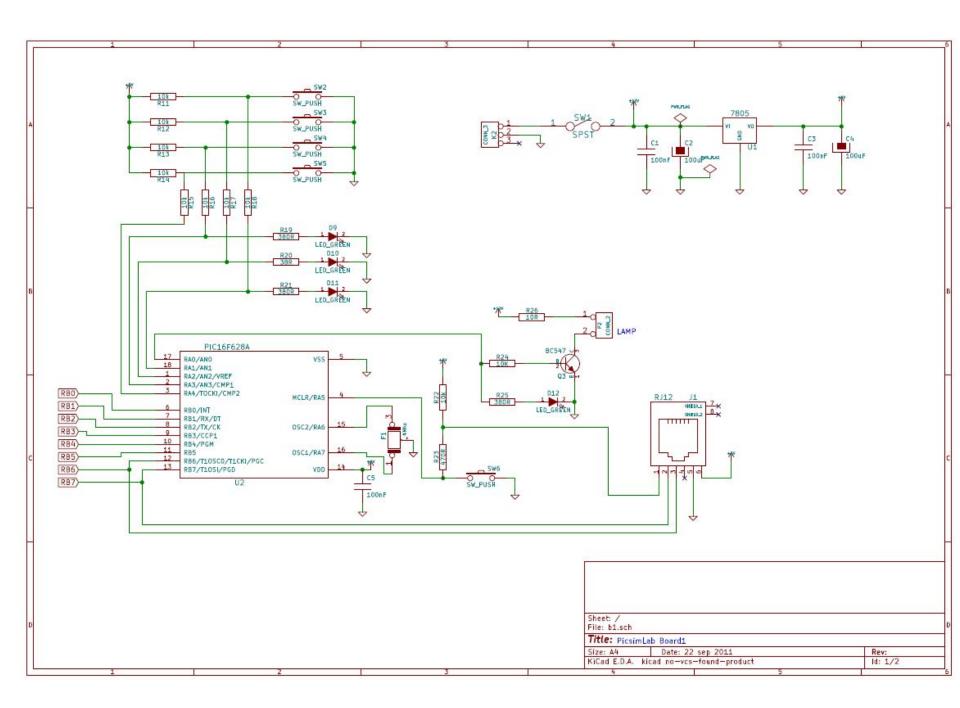
Il che equivale a configurare per PORTB i PIN 0, 1 e 7 come uscite mentre i bit 2, 3, 4, 5, 6 come Input

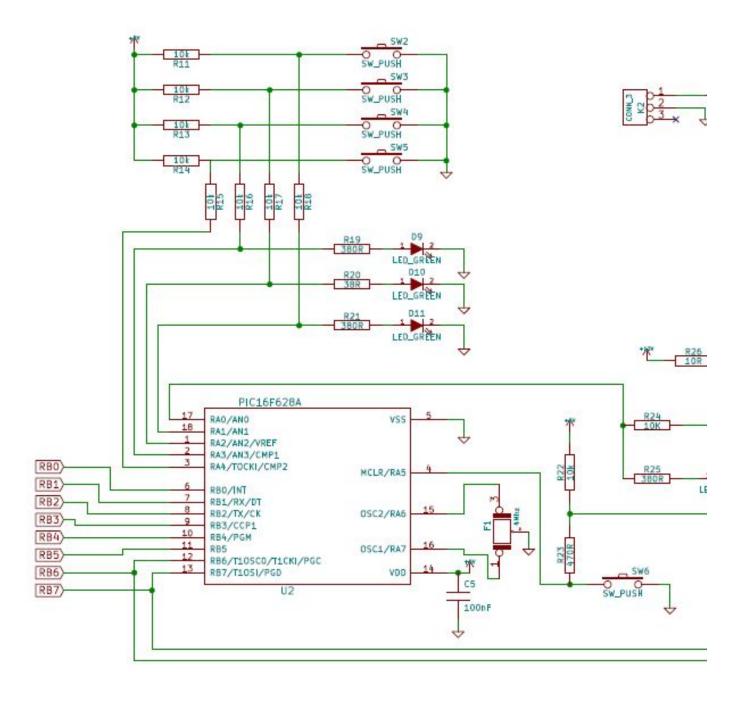
PORTB = 0x51; // 51 in esadecimale equivale alla configurazione binaria 01010001

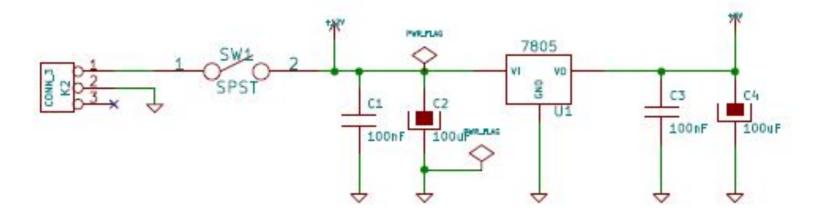
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	
0	1	0	1	0	0	0	1	

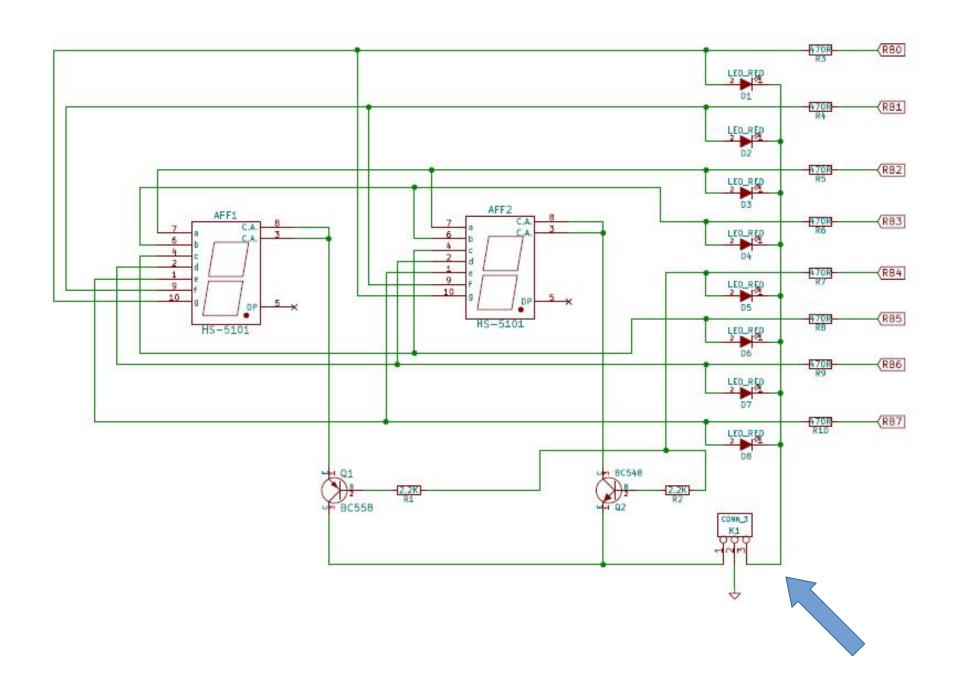
Il che equivale ad imporre su PORTB i PIN 0, 4 e 6 una tensione di 5V, mentre nei bit 1, 2, 3, 5 e 7 una tensione di 0V.

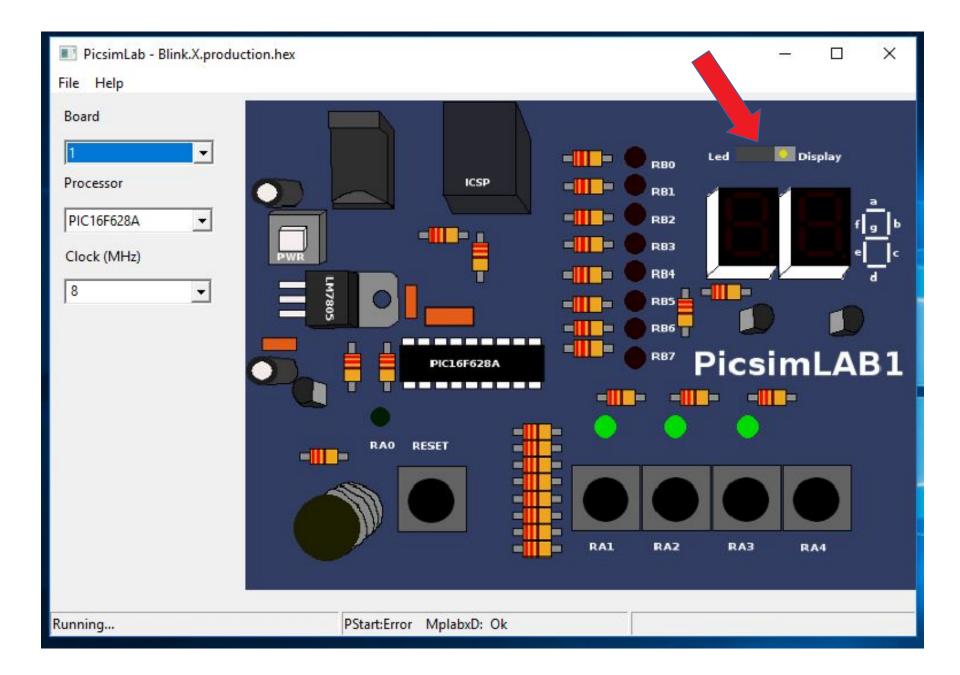
```
// uC configuration
#pragma config FOSC = HS
#pragma config WDTE = OFF
#pragma config PWRTE = OFF
#pragma config MCLRE = ON
#pragma config BOREN = ON
#pragma config LVP = OFF
#pragma config CPD = OFF
#pragma config CP = OFF
#include <xc.h> // include library
void main()
   TRISB = 0 \times 00; // SET all bits of PORTB as OUTPUT
   PORTB = 0xFF; // Set all bits of PORTB to 5V
   return;
```











```
🎮 main.c 🔕
                   Source
        History
   □ // PIC16F628A Configuration Bit Settings
 2
 3
     // 'C' source line config statements
 4
 5
     // CONFIG
 6
     #pragma config FOSC = HS
                                    // Oscillator Selection bits (HS oscillator: High
 7
     #pragma config WDTE = OFF
                                    // Watchdog Timer Enable bit (WDT disabled)
     #pragma config PWRTE = OFF
                                    // Power-up Timer Enable bit (PWRT disabled)
 9
     #pragma config MCLRE = ON
                                    // RA5/MCLR/VPP Pin Function Select bit (RA5/MCLR
     #pragma config BOREN = ON
10
                                    // Brown-out Detect Enable bit (BOD enabled)
11
     #pragma config LVP = ON
                                    // Low-Voltage Programming Enable bit (RB4/PGM pi
12
     #pragma config CPD = OFF
                                    // Data EE Memory Code Protection bit (Data memor
13
     #pragma config CP = OFF
                                    // Flash Program Memory Code Protection bit (Code
14
15
     #define XTAL FREQ 8000000
16
     #include <xc.h>
17
18
     void main(void) {
19
         // configuro la porta B come uscita
20
         TRISB = 0 \times 00;
21
22
         while(1)
23
24
             PORTB = 0x00; // spengo tutte le uscite
25
             delay ms(500);
                                // attendo
             PORTB = 0xFF; // accendo tutte le uscite
26
27
             __delay_ms(500); // attendo
28
29
         return;
30
31
```