TEP 2014

Guido Piangatello
(guido@piangatello.it)

Tecnologie **E** **P**rogettazione **S**istemi **E**lettrici ed **E**lettronici

1	**Giunzione pn**		
1.1	Cond. di eq. della g.	Fisica della giunzione PN	
1.2	Caratt. I-V e infl. di T	Cosa da sapere sui diodi	
1.3	Comportam. dinam.		
1.4	Giunz. Metallo-semic.		
1.5	Drogaggio e conten.		

2	**Diodi** a semiconduttore		
		LED: tensioni, correnti, protez.	7’
		Progetto alimentatore non stab.	9’
		Alimentatore duale 1.2-24V1A	9’ S
		Alimentatore 3-30V 3A	
2.9	M. di fabbricaz. diodi		

3	**BJT**		
		Introduzione al transistor BJT	8’
		Domande/ris. sul BJT con LED	5’
		Carica batterie minimale	
		Carica batterie con scarica	
 Robot che segue la luce		Robot luce: arresto e marcia	4’
		Robot luce: sterzata	3’
3.6	M. di fabbricaz. BJT		

4	**jFET e MOS**
5	**Memorie**

6	**Circuiti stampati**
7	**Saldatura**
8	**Progetto c. stampati**
9	**OrCAD**
9.1	Lista di connessione
9.2	Piattaforma di lavoro
9.3	Disegni di fabbricaz.
9.4	Librerie di layout

10	**CIRCAD**

11	**CAD Eagle**

12	**Operazionali**	L’amplificatore operazionale	8’
		AO in configurazione invertente	12’
		AO non invertente e inseguitore	6’
		AO con limiti di banda	8’
		AO a singola alimentazione	7’

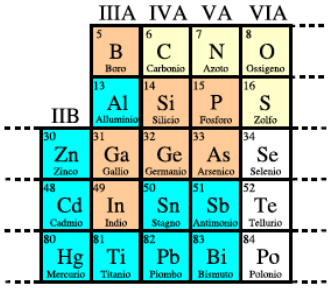
13	**Struttura PLC**
14	**Programmaz. PLC**
15	**Automaz. con PAC**

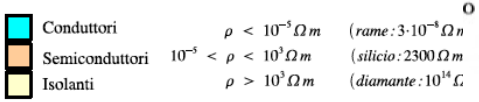
16	**Domotica**	Funzioni di un imp. domotico	10’
		Impianto LonWorks minimale	10’
		Impianto LonWorks completo	7’
		Impianto KNX con GWBUS	6’
		Struttura sist. LonWorks	
		Struttura sistema KNX	
		3+6 sist. dom. a confronto	8’
		Introduzione a KONNEX	4’
		Introduzione a LonWorks	4’
		Introduzione a MY HOME	4’
		Sulla diffusione domotica	6’
		Controller luci DALI	2’

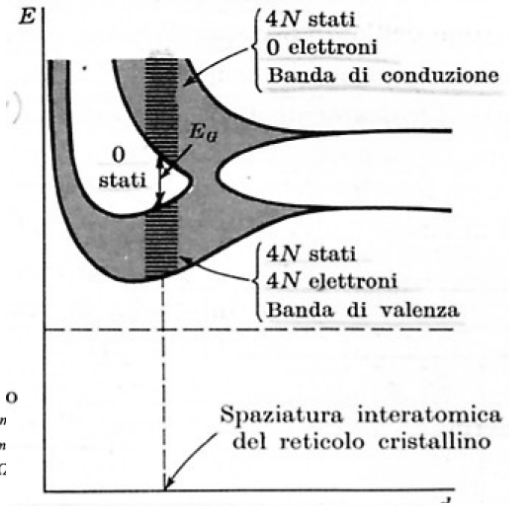
17	**Microprocessori**	Una CPU in azione	14’
18	**Process. seg. digitali**		
19	**Sistemi Scada**		
20	**Personal computer**		
21	**Microcontrollore PIC**	Struttura di un microcontrollore	6’
		PIC 16F84: caratteristiche e pin	13’
		I registri del PIC 16F84	11’
		Etichette, direttive, ritardi e ast.	20’

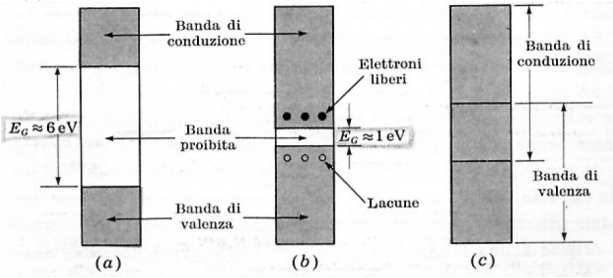
22	**Microcontrollore st62**	Arduino uno R3	10’
		Ingressi e uscite	8’
		Buzzer + fotoresistenza	9’
		Led RGB con cursore	9’
		Com. LCD 16 col x 2 righe	9’
		On-off con telecom. TV IR	4’
		Servom. che copia un potenz.	11’
		Motore stepper (passo-p.)	15’

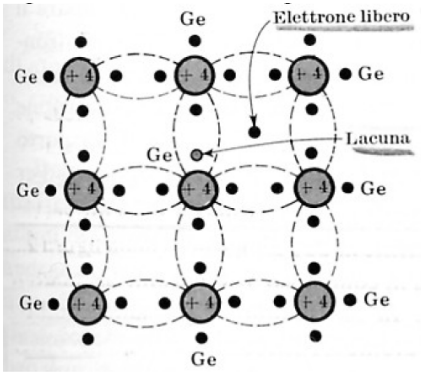
23	**Manutenzione e qualità**
24	**Sicurezza**

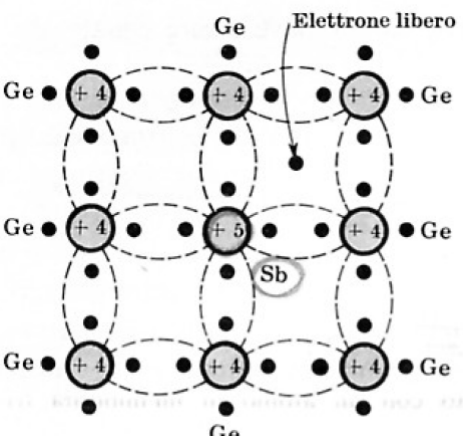


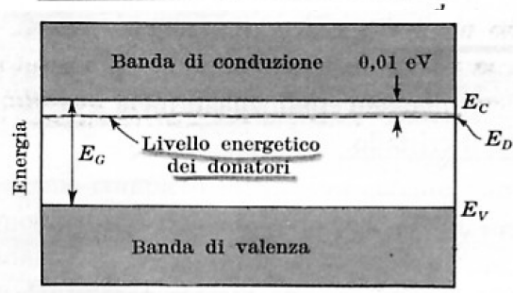


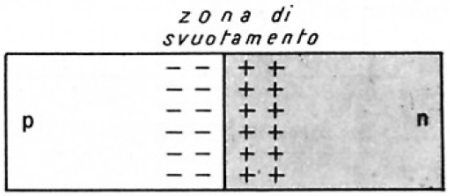


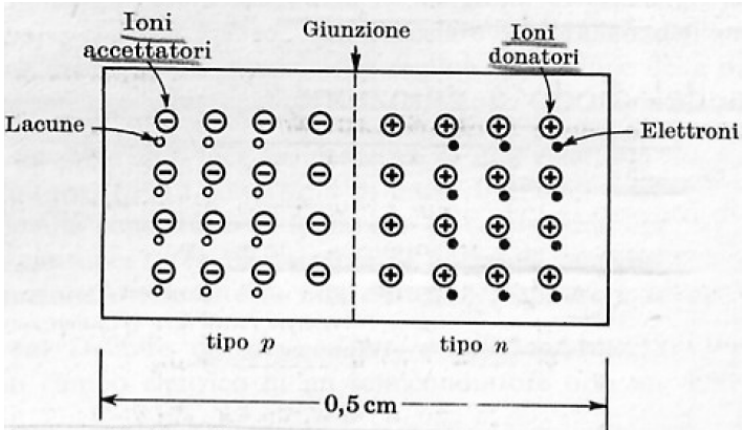


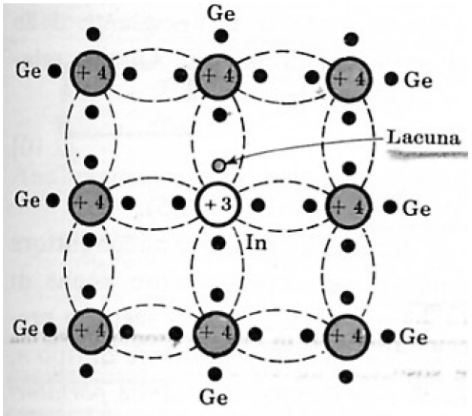


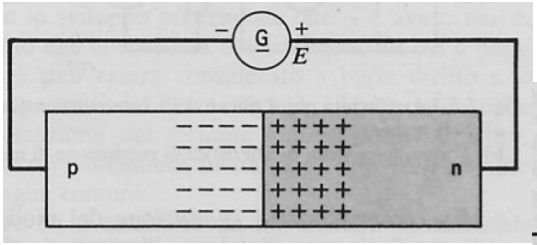


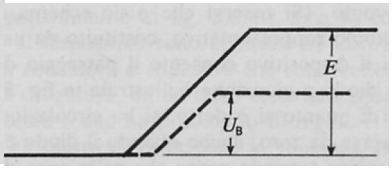








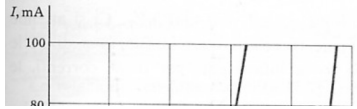




ottenere densità di corrente dell'ordine di
e di 1 A/mm^2 nei diodi al silicio. Le te
raggiungere alcune centinaia di volt nei p
secondi.

La figura 18 mostra la caratteristica dire
un diodo al germanio ed uno al silicio.

Figura 18 – Caratteristica tensione-corrente di
diodo al silicio a 25° C.



Giunzione PN, fotodiodi, varicap e led

(?'')

I semiconduttori sono conduttori o isolanti?

1) Sono isolanti nei quali ci vuole poca energia per creare alcuni elettroni liberi.

2) a Tamb hanno un numero limitato di elettroni liberi, per cui la corrente dovuta ad essi aumenta con la tensione finché non raggiunge un valore massimo (che chiameremo corrente inversa di saturazione I_0), poi la corrente resta la stessa anche aumentando la tensione.

Questa corrente dovuta alle cariche che si sono liberate da sole aumenta se si aumenta cosa?

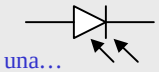
1) se si aumenta la temperatura T del semiconduttore;

2) se si aumenta la luce incidente sul semiconduttore.

Cosa è il drogaggio?

L'aggiunta di pochi atomi pentavalenti-donatori (drogaggio n) o trivalenti-accettori (drog. p)

Fotodiodi

Simbolo fotodiodo 

Usato come sensore è una…

R che diminuisce aumentando la luce

Lavora in polarizzazione diretta o inversa? Inversa, in modo che la sua (piccola) corrente sia dovuta alle cariche minoritarie, che sono di più se la luce (e/o la T) è maggiore.

Perché la zona di svuotamento si chiama così?

Perché in questa zona (spessore sui 0.5 μm) non ci sono cariche libere.

Perché una giunzione pn ha una capacità C ?

Perché ai lati è conduttrice mentre sulla giunzione è isolante.

Perché la zona di svuotamento si chiama anche zona di carica spaziale? Chi blocca la corrente di diffusione, ad un certo punto?

Perché un elettrone che migra crea uno ione + e fermandosi in una buca crea uno ione -. Questa zona ionizzata si oppone alla diffusione, azzerandola quando la zona di svuotamento è abbastanza estesa.

Dal punto di vista dell'energia, cosa succede quando un elettrone "cade" in una buca?

L'elettrone perde energia e l'energia ceduta può diventare energia termica o energia luminosa.

Diodi varicap

Simbolo del diodo varicap 

Che capacità C può avere? 1 ÷ 100 pF

Lavora in polarizzazione diretta o inversa?

Inversa, se no conduce e un C non deve condurre.

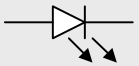
Come si può variare la sua C?

Variando la tensione inversa, cosa che fa variare lo spessore della zona di svuotamento.

Come si può ottenere luce durante una ricombinazione elettrone-lacuna?

Usando determinati droganti

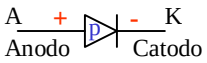
Diodi Emettitori di Luce (LED)

Simbolo del LED 

$V_{diretta}$ per farlo condurre	1,6 ÷ 2,2 V
I per bassa – alta luminosità	5 ÷ 20 mA
$V_{inversa}$ massima	3 ÷ 5 V

Polarizzare un diodo è… applicargli una V

La polarizzazione è diretta se…

la V ha questi segni: 

Una $V_{diretta}$ allarga o restringe la zona di svuotamento e con che effetti?

La restringe, rendendola insufficiente per fermare la corrente di diffusione e provocando quindi lo scorrere di una corrente grande perché dovuta alle cariche maggioritarie.

Una $V_{inversa}$ allarga o restringe la zona di svuotamento e con che effetti?

La allarga, per cui può scorrere solo la corrente dovuta alle cariche minoritarie (frazioni di μA)

Caratteristica del diodo

La tensione di soglia è… la tensione diretta sotto la quale la corrente è trascurabile.

Vale… 0,5 V per il Si e 0,1 V per il Ge

Perché I_0 dicesi di saturazione?… Perché non aumenta con Vinversa ma è costante

Valori di I_0: Frazioni di μA

La tensione di rottura (breakdown) inversa è… la tensione inversa da non raggiungere se non si vuole danneggiarlo irreparabilmente.

Vale… decine di V (es.: 80 V) o centinaia di V (es. 400 V per reggere la Vrete di 311V)

$$I = I_0 e^{\frac{V}{2\frac{T}{11600}}} - I_0$$

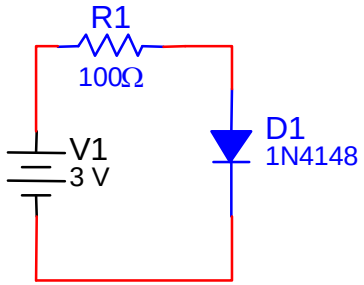
1 per alte correnti

Variaz. di I_0 con T: raddoppia ogni 10 ^{0}C di aumento di T

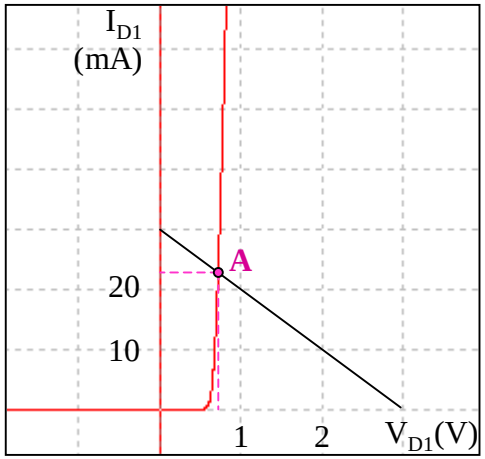
Se T aumenta do x gradi, di quanto deve diminuire V per far scorrere la stessa corrente? di x * 2.5 mV

T/11600 = V_T = 26 mv a 300 ^{0}K

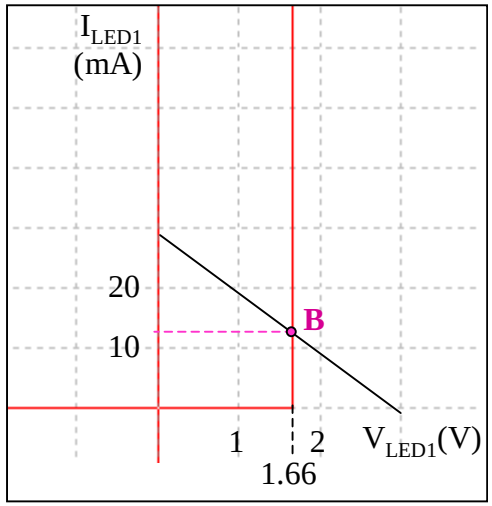
Il diodo come elemento circuitale



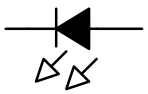
Punto di lavoro del diodo:
A (713 mV ; 23 mA)



Punto di lavoro del LED:
B (1.66 V ; 13 mA)



LED: tensioni, correnti, protezione (7')



Un **LED** (Light Emitting Diode) è un diodo che emette luce quando è attraversato dalla corrente.

Le differenze rispetto al diodo sono 2:

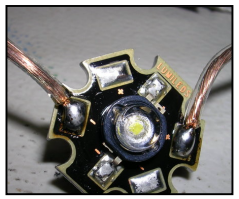
1) Serve più tensione per condurre:

a) **tensione di soglia** maggiore;

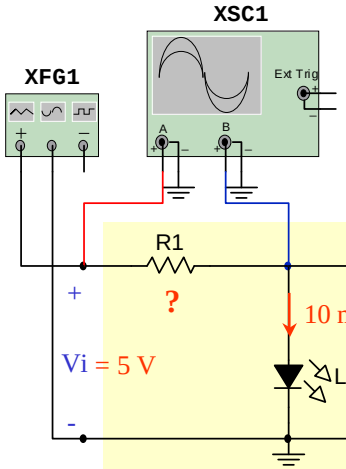
b) **quando conduce** ai suoi capi non ci sono all'incirca 0,7 V, ma:

Vsoglia = 0,5V
Vsoglia > 1.5V

Vconduzione 0,7V
Vconduzione > 1,7 V



⇦ 1,7 V (led rossi) o 2V (led verdi)
o 3V (led bianchi)
o 4,5V (led ultravioletti)
o 1,3V (infrarossi)



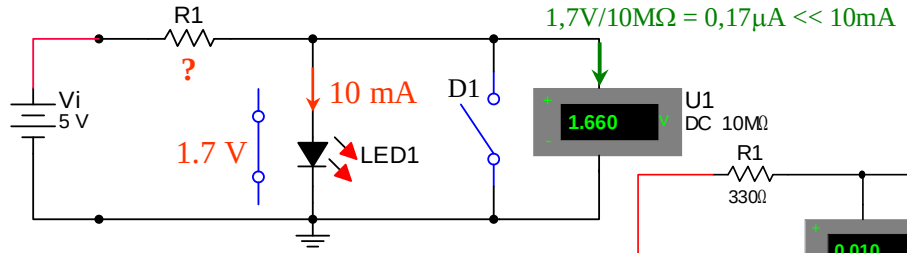
Visualizzatore a LED di tensione Vi alta o bassa

protezione del LED contro le inversioni in ingresso

Vrottura > 70 V

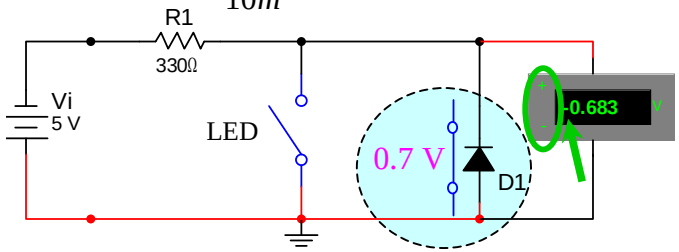
2) la **tensione inversa di rottura** non è 60/80V come nei diodi normali (400V nei diodi raddrizzatori) **ma di soli 3-10V**

Vrottura= 3-10V nei LED

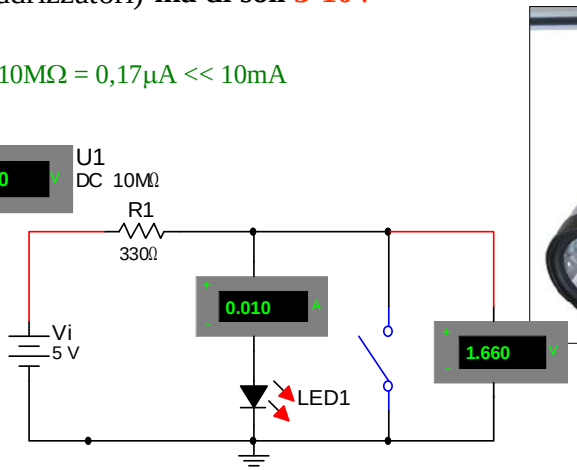


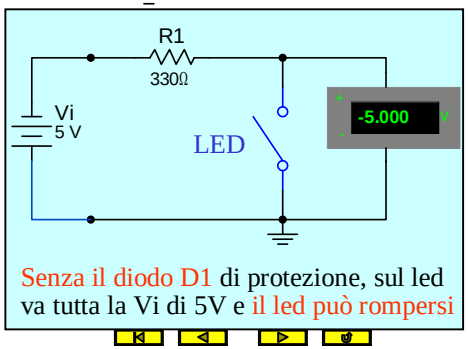
$$I_{LED1} = I_{R1} = \frac{V_i}{R1} = \frac{V_i - 1{,}7}{R1} = \frac{3{,}3}{R1} = 10m$$

$$\Rightarrow \quad R1 = \frac{3{,}3}{10m} = 0{,}33k = 330\Omega$$



D1 limita a circa 1V la tensione *inversa* (quella quando non conduce) sul LED





Senza il diodo D1 di protezione, sul led va tutta la Vi di 5V e il led può rompersi

In un **LED normale** la corrente varia da 5mA (si vede acceso, ma la luminosità è bassa) a 20mA (alta luminosità).

Nei **LED a basso consumo** bastano 3mA (bassa luminosità) e 10mA (alta luminosità).

Nei **LED di potenza** (quelli usati per l'illuminazione) si va da 100mA a 20A

(es. un LED da **1W** viene alimentato con una corrente costante di **350 mA**; alimentarli a tensione costante pregiudica o la durata o la luminosità e non si fa)

I LED di potenza costano molto di più dei led a bassa corrente visti i **230 euro** che costava (prezzo 2010) questo faretto a led da **27W**…

che emette la stessa luce di un neon da 27 W (2200 lumen) e quindi ha la stessa efficienza del neon (**82 lumen/W**)…

offrendo come vantaggio rispetto al neon solo una **durata di circa 10 volte maggiore**, **assenza di sfarfallamento** e **accensione istantanea**, comportandosi quanto a costanza della luce come un'alogena che però consuma 4/5 volte di più

Progetto alimentatore 20V, 1 A, 5% di ripple

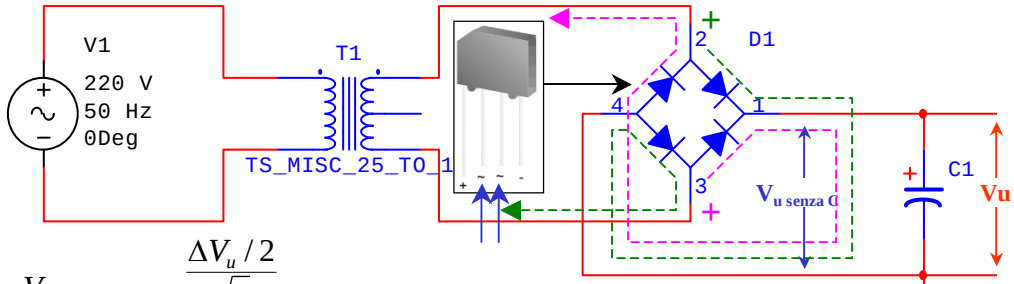
(9')

Progettare un alimentatore non stabilizzato con raddrizzamento a doppia semionda e filtro capacitivo con le seguenti specifiche:

Vu = 20 V - Iumax = 1 A - ripple r < 5%

Lo schema è questo e progettarlo significa scegliere i componenti perché abbia le prestazioni (specifiche) previste dal progetto.

È il **condensatore** che eroga la corrente di di uscita Iu, con un abbassamento di tensione ΔVu in un tempo Δt, e il legame tra Iu e ΔVu/Δt è proprio la capacità C del condensatore:

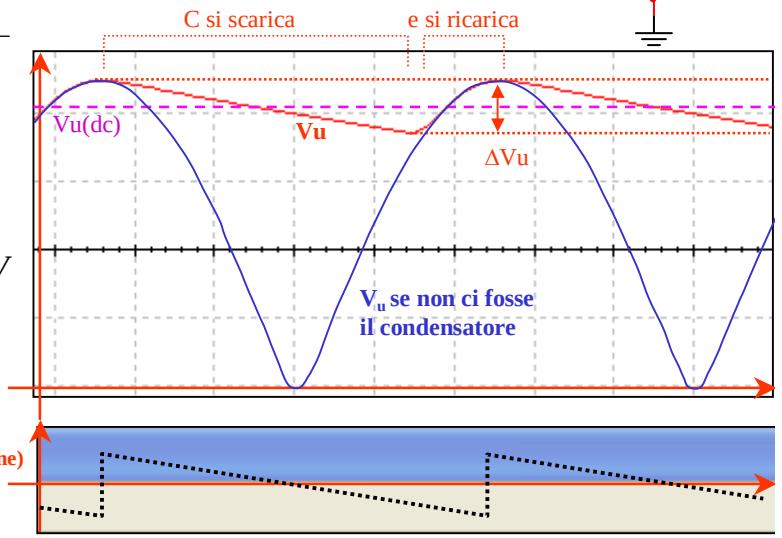


$$I_u = C\frac{\Delta V_u}{\Delta t} \cong C\frac{\Delta V_u}{T} \cong C\frac{\Delta V_u}{\frac{1}{f}} = Cf\Delta V_u \Rightarrow C = \frac{I_u}{f\Delta V_u} \Rightarrow C = \frac{1}{100 * \Delta V_u} = 2890\mu F$$

$$\Delta V_u = 3.46V$$

$$r = \frac{V_{ripple(eff)}}{V_{u(dc)}} = \frac{\frac{\Delta V_u / 2}{\sqrt{3}}}{V_{u(dc)}}$$

$$r = \frac{\frac{\Delta V_u / 2}{\sqrt{3}}}{V_{u(dc)}} = 0.05 = \frac{5\%}{100} \Rightarrow \frac{\Delta V_u / 2}{\sqrt{3}} = 0.05 * V_{u(dc)} = 0.05 * 20 = 1V \Rightarrow \Delta V_u / 2 = \sqrt{3} * 1V$$



I condensatori seguono la serie E12 delle resistenze. Scegliamo il valore standard di C più basso, 2700 μF, perché questa formula fornisce un C più grande di quello che serve per avere quel ripple r.

E12 (10%)
10
12
15
18
22
27
33
39
47
56
68
82

Il C lavora alla tensione Vu di 20 V, ma la tensione di rete può variare ±10% e un po' di margine non guasta per cui prendiamo un 35V

$$C1 = 2700\mu F - 35V$$

Il **trasformatore** deve fornire la tensione Vu + ΔVu/2 (tensione di picco della Vu) nonostante le cadute di tensione su due diodi, per cui:

$$V_{s(max)} = V_u + \frac{\Delta V_u}{2} + 2V_{diodo} = 20 + \frac{3.46}{2} + 2 * 0.8 = 23.3V$$

Maggioriamo del 10% perché erogando corrente la V fornita dal trasformatore s'abbassa (meno del 10% nei grossi trasformatori e più del 10% in quelli piccoli)

$$V_{s(max)} = 1.1 * 23.3 = 25.63V \qquad V_{s(eff)} = \frac{Vs(max)}{\sqrt{2}} = 18.1V$$

Per la corrente usiamo la formula pratica: $I_{s(eff)} = 1.8I_u = 1.8I_u = 1.8A$

Il trasformatore deve avere una $P = V_{s(eff)}I_{s(eff)} = 1.8 * 18.1 = 32.5VA$

$$Trasformatore: 18V - 36VA$$

Vu	**V**	**20**
Iu	**A**	**1**
r	**%**	**5**
Delta Vu eff	V	1
delta Vu	V	3,46
C	**uF**	**2887**
Vc	**V**	**40**
Vdiodo	V	0,8
Vsecmax	V	23,33
maggioraz	%	10
VsecmaxOK		25,67
Vseff	**V**	**18,15**
Iseff	A	1,8
P	**VA**	**32,67**
Idiodi	A	0,5
IdiodiSicura	A	1
C	**mA**	**1000**
IpiccoRipet	A	10
Vr	V	25,67
B	**V**	**30**

B sta per V (tensione inversa) →
C sta per corrente →

		B40 C1000	B80 C1000	B125 C1000
V_{RRM}	Peak Recurrent Reverse Voltage (V)	100	200	300
V_{RMS}	Maximum RMS Voltage (V)	70	140	210
V_R	Recommended Input Voltage (V)	40	80	125
$I_{F(AV)}$	Forward current at Tamb = 25 °C R load			1.2 A
	C load			1.0 A
I_{FRM}	Recurrent peak forward current			10 A
I_{FSM}	10 ms. peak forward surge current			40 A

Per dimensionare i 4 **diodi**, o il ponte a diodi osserviamo che quando in una coppia di diodi passa la corrente Iu nell'altra coppia non passa corrente. Pertanto la **corrente media (average) diretta (forward)** è Iu /2 = 0.5 A

La tensione inversa = Vsec(max) = 26 V.

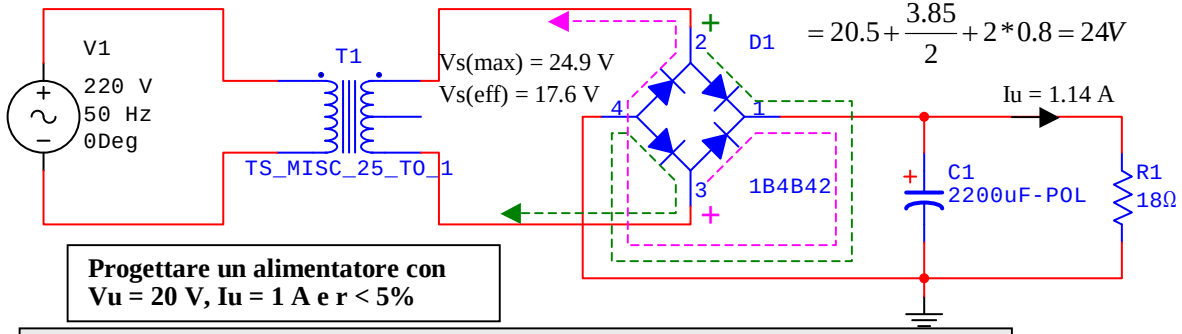
$$Ponte: B40 - C1000$$

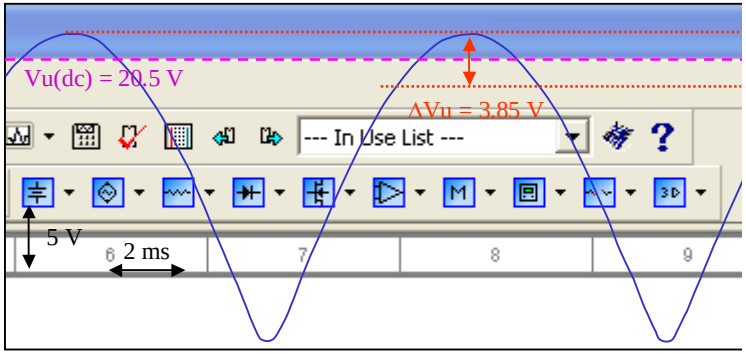
Progetto alimentatore 20V, 1 A, 5% di ripple

$$V_{s(\max)} = V_u + \frac{\Delta V_u}{2} + 2V_{diodo} =$$

$$= 20.5 + \frac{3.85}{2} + 2*0.8 = 24V$$

$$r = \frac{\Delta V_{u(eff)}}{V_{u(dc)}} = \frac{\Delta V_2 / 2}{\sqrt{3}} = \frac{3.85/2}{\sqrt{3}} = \frac{1.11}{20.5} = 0.05 = 5\%$$





Progettare un alimentatore con Vu = 20 V, Iu = 1 A e r < 5%

Condensatore C1

È C1 che eroga la corrente di 1 A, con un abbassamento di tensione ΔVu in un tempo Δt che, se trascuriamo il tempo di ricarica, è il T di un segnale a f = 100 Hz

$$I_u = C\frac{\Delta V_u}{\Delta t} \cong C\frac{\Delta V_u}{T} = Cf\Delta V_u \Rightarrow \boxed{C = \frac{I_u}{f\Delta V_u}}$$

Ci serve ΔVu

$$\Delta V_{u(eff)} = \frac{\Delta V_u}{2\sqrt{3}} = rV_{u(dc)} = 0.05*20 = 1V \Rightarrow \Delta V_u = 2\sqrt{3}*1 = 3.46V$$

$$C = \frac{1}{100*3.46} = 2890\mu F$$

I condensatori seguono la serie E12 delle resistenze. Scegliamo il valore standard di C più basso, 2700 µF, perché questa formula fornisce un C più gran-de di quello che serve per avere quel ripple r (nel circuito un C da 2200 µF basta per avere quel r)

$$\boxed{C1 = 2700\mu F - 50V}$$

Il C lavora alla tensione Vu di 20 V, quindi basterebbe un C da 25 V di tensione massima, ma per sicurezza prendiamo uno che sopporta una tensione doppia: 50 V

Se C sta per corrente, B non può che stare per tensione

		B40 C1000	B80 C1000	B125 C1000	B250 C1000	B380 C1000
V_{RRM}	Peak Recurrent Reverse Voltage (V)	100	200	300	600	900
V_{RMS}	Maximum RMS Voltage (V)	70	140	210	420	630
V_R	Recommended Input Voltage (V)	40	80	125	250	380
$I_{F(AV)}$	Forward current at Tamb = 25 °C R load / C load	1.2 A / 1.0 A				
I_{FRM}	Recurrent peak forward current	10 A				
I_{FSM}	10 ms. peak forward surge current	40 A				

Maggiorando del 10% perché erogando corrente la V fornita si abbassa ⇩

Absolute Maximum Ratings		T_C = 25 °C unless otherwise specified	
Symbol	Conditions	Values	Units
I^2t	Rating for fusing, t < 10 ms	110	A^2s
I_{FRM}	Repetitive peak forward current > 15 Hz	30	A
I_{FSM}	Peak forward surge current, 50 Hz half sine-wave	150	A
t_{rr}	Reverse recovery time (I_F = A)	/	ns
I_{FAV}	Max. averaged fwd. current, R-load, T_A = 50 °C 1)	2,7	A
I_{FAV}	Max. averaged fwd. current, C-load, T_A = 50 °C 1)	2,2	A
I_{FAV}	Max. current with cooling fin, R-load, T_C = 100 °C 2)	4,8	A
I_{FAV}	Max. current with cooling fin, C-load, T_C = 100 °C 2)	3,7	A

Trasformatore

$$V_{s(\max)} = V_u + \frac{\Delta V_u}{2} + 2V_{diodo} = 20 + \frac{3.46}{2} + 2*0.8 = 23.3V \qquad V_{s(\max)} = 1.1*23.3 = 25.63V$$

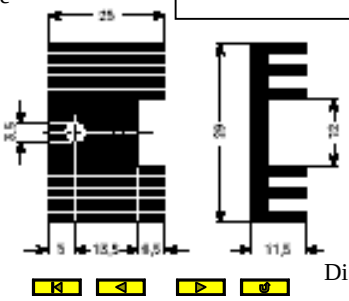
Formula pratica $\boxed{I_{s(eff)} = 1.8I_u}$ $\quad I_{s(eff)} = 1.8A \quad P = V_{s(eff)}I_{s(eff)} = 1.8*18.1 = 32.5VA \quad V_{s(eff)} = \frac{Vs(\max)}{\sqrt{2}} = 18.1V$

$$\boxed{Trasformatore: 18V - 36VA}$$

Ponte a diodi Quando in una coppia di diodi passa la corrente Iu = 1, nell'altra non passa corrente. La corrente media (average) diretta (forward) è Iu /2 = 0.5 A

$$I_{F(AV)diodiPonte} = \frac{I_u}{2}$$

Ci basterebbe un ponte C500 (C sta per corrente e questa è in mA), ma non è bene far lavorare i diodi al limite e quindi la raddoppiamo prendendo un C1000.
In realtà nel diodo passa una corrente molto maggiore (3-5 volte maggiore) per un tempo molto minore del periodo, ma un C1000 ha una corrente di picco ripetitiva (I_{FRM} dove M = max, R = picco ripetitivo e F = forward = diretta) 10 volte maggiore di quella average (10 A) e ben 40 volte maggiore se il picco è una tantum (picco non ripetitivo) e non dura più di 10 ms.
(In un ponte C3700-2200 la corrente maggiore è con raffreddatore e vale la minore se il ponte non è montato sopra un raffreddatore alettato di alluminio)
La tensione inversa = Vsec(max) = 26 V. Ci basterebbe un ponte con V_{RRM} (max ripetitiva reverse tensione) di 40 V ma prendiamo un B40 che ha di 40 V la V_{RRMS} (RMS = root mean square = radice media quadrati = efficace) e quindi una V_{RRM} di 80V o anche di più



C = corrente
B = tensione

Type	Alternating Input voltage V_{VRMS} V	Repetitive peak reverse voltage V_{RRM} V	Max. admissable load capacitor C_L µF	Min. required protective resistor R_t Ω
B40 C3700-2200	40	80	5000	0,5
B80 C3700-2200	80	160	2500	1
B125 C3700-2200	125	250	1500	2
B250 C3700-2200	250	600	800	4
B380 C3700-2200	380	800	600	5
B500 C3700-2200	500	1000	400	6,5

Dissipatore da 17 ^{0}C/W

$$\boxed{Ponte: B40 - C1000}$$

Alimentatore duale 1.2-24V 1A

Trasformatore (non disegnato): riduce la tensione e isola l'utente dalla 220V; deve essere a presa centrale per fare un alimentatore duale.

D1-D2-D3-D4: ponte (realizzato con componenti separati) per raddrizzamento a doppia semionda

C1: condensatore di filtro dell'alimentatore non stabilizzato (nel calcolo per far tornare un condensatore di filtro di 470 µF ho dovuto mettere come Imax 0,8A e come ripple il 25%, alto ma il regolatore lo riduce)

C3: 0, 1 µF in // a 470 µF sembra ininfluente, ma gli elettrolitici non si comportano bene in alta f e il C piccolo si mette per eliminare i disturbi a radio-f

VR1 (LM317): regolatore (ovvero stabilizzatore di tensione) a tre piedini (adatto per correnti fino a 1A in contenitore TO220 come i nostri e fino a 2A in contenitore T03, in ogni caso serve un dissipatore adeguato) di tipo variabile (dal terzo piedino - chiamato Adjust e non Ground - esce una corrente molto più piccola di quella dei regolatori fissi) e di tipo positivo

VR2 (LM337): versione negativa dell'LM317:(la differenza è che il transistore interno usato per variare la tensione di uscita è girato per far passare una corrente che entra dall'uscita invece che andare verso l'uscita)

R1 trasforma la tensione fissa di uscita (1,2V tra Output e Adjust) in una corrente fissa (di 10 mA, con R=120Ω) che serve a RV1 per variare l'uscita

RV1 varia la tensione di uscita: se RV1 = 0 allora l'uscita SK3+ sarà di 1,2V (Vu minima). Se invece RV1=22k su di essa i 10mA creano una V di 22V col risultato che in uscita ci sarà 22 + 1,2 = 23,2 V (Vu massima)

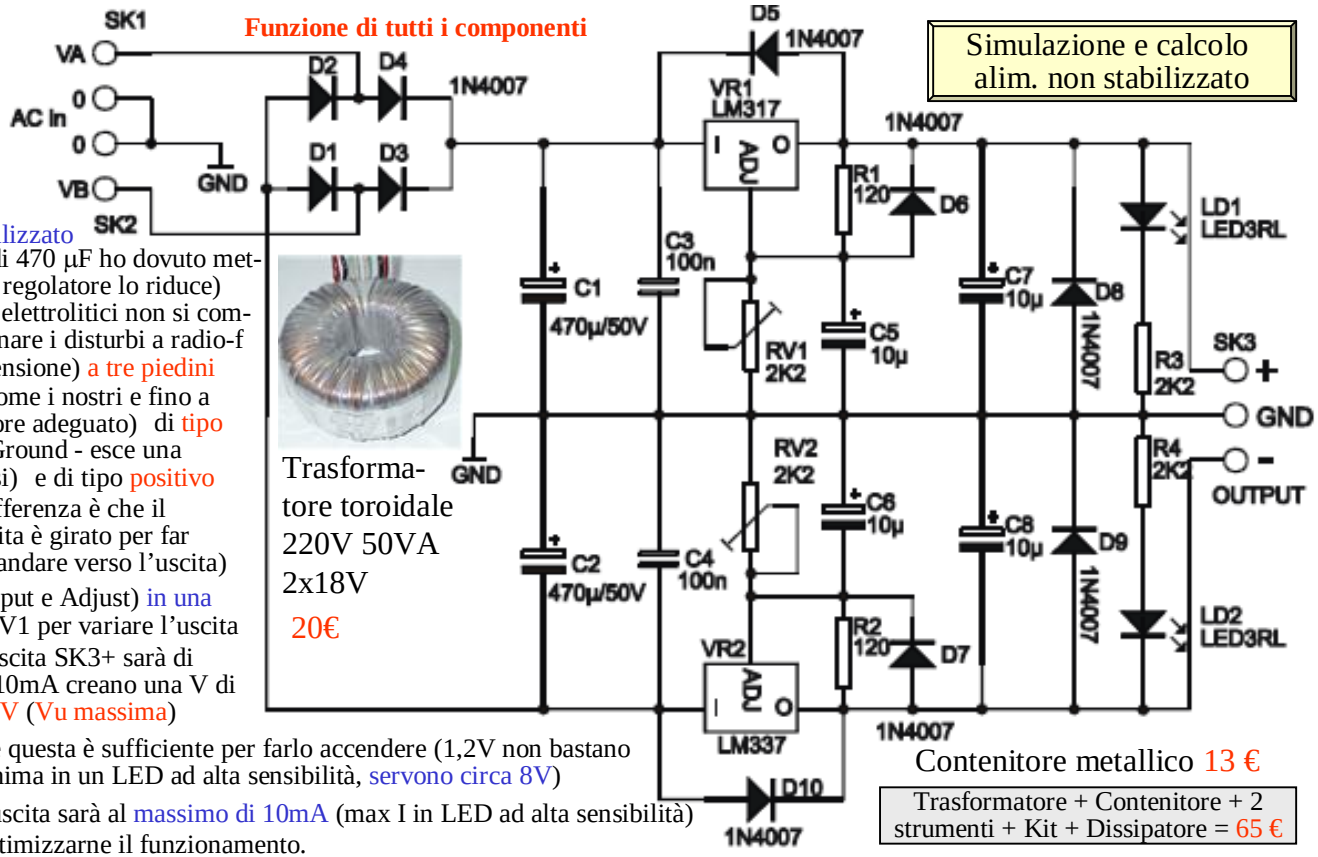
LD1: segnala la presenza di una tensione in uscita, se questa è sufficiente per farlo accendere (1,2V non bastano di sicuro e per far scorrere almeno 3mA, corrente minima in un LED ad alta sensibilità, servono circa 8V)

R3: limita la corrente nel led, che anche con 24V in uscita sarà al massimo di 10mA (max I in LED ad alta sensibilità)

C5-C7: consigliati dal costruttore dell'LM317 per ottimizzarne il funzionamento.

D5: scarica C7 che danneggerebbe l'LM317 se si cortocircuitasse l'ingresso **D6**: Insieme a D5 scarica C5 che danneggerebbe l'LM317 se si cortocircuitasse l'ingresso

D8: protegge VR1 dall'eventualità che all'accensione VR2 fornisca una tensione negativa prima che VR1 dia in uscita la sua tensione positiva, rendendo negativa l'uscita di VR1 se sono inseriti i carichi



Trasformatore toroidale 220V 50VA 2x18V 20€

Simulazione e calcolo alim. non stabilizzato

Contenitore metallico 13 €

Trasformatore + Contenitore + 2 strumenti + Kit + Dissipatore = 65 €

$$P_{Da\ dissip.} = (V_I - V_O)I$$

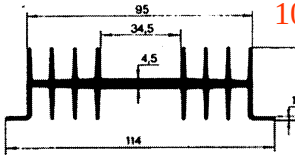
$$P_{Dissipata} = \frac{T_j - T_a}{\theta_{ja}}$$

Junzione Case Dissipatore Ambiente

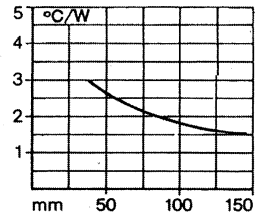
$$\theta_{jc} + \theta_{cd} + \theta_{da}$$

TO 220 $\theta_{jc} = 4\ ^0C/W$
TO 3 $\theta_{jc} = 2\ ^0C/W$

θ_{cs} (°C/W)	Contatto diretto	Con silicone	Con silicone e mica
TO3	0,5 ÷ 0,7	0,3 ÷ 0,5	0,4 ÷ 0,6
TO220	1,0 ÷ 1,3	0,6 ÷ 0,8	0,8 ÷ 1,1



10 €



6 €

Voltmetro digitale LCD da incasso a 3 1/2 cifre. Alimentazione: 9Vdc (7 - 12Vdc). Lettura fondo scala 200 mV.

In Kit **11 €**

$P_{con\ Vo\ min\ di\ 5V} = (20\text{-}5)1 = 15W$

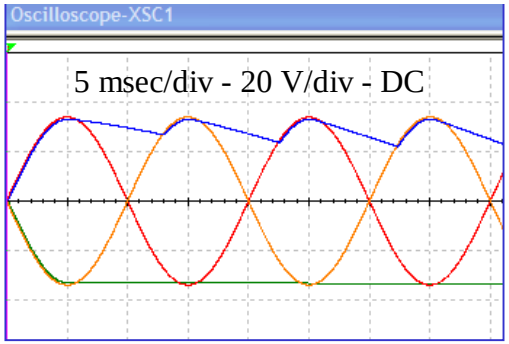
$$\theta_{tot} = \frac{T_{jmax} - T_{amax}}{P} = \frac{125 - 60}{15} = 4.3°C/W$$

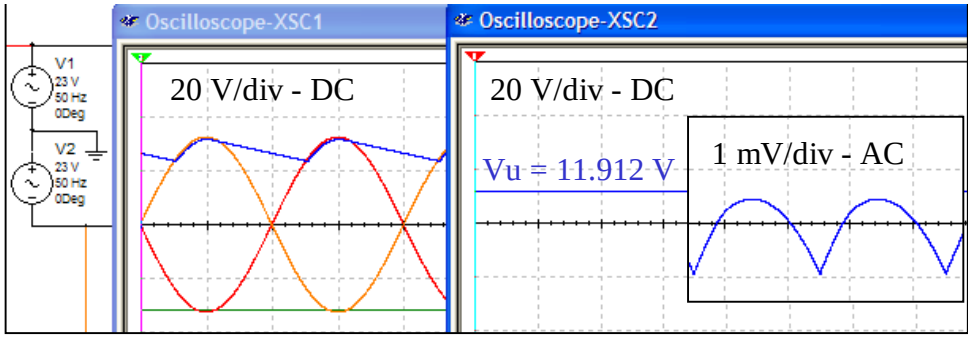
Con **TO3** e silicone 4.3 – 2 – 0.8 = 1.5 $^0C/W$ => 15 cm

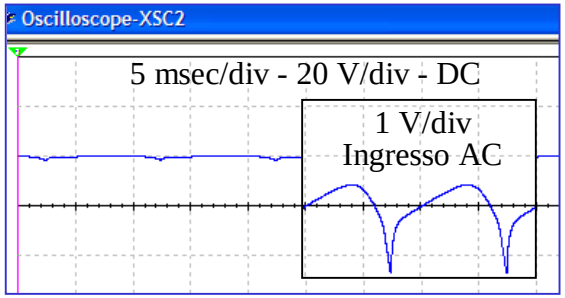
Per il **TO220** I_{max} = 0,7A (=> θ_{tot} = 6.2 $^0C/W$)

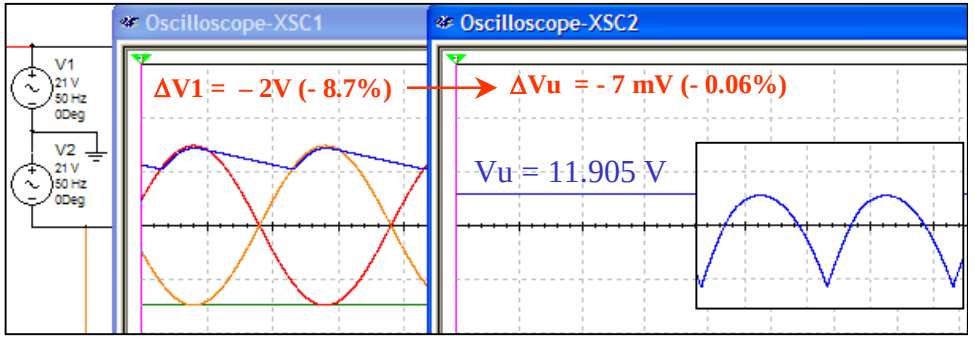
Simulazione alimentatore duale

Vu	V	20
Iu	A	0,8
r	%	25
Delta Vu eff	V	5
delta Vu	V	17,32
C	**uF**	**462**
Vc	**V**	**40**
Vdiodo	V	0,8
Vsecmax	V	30,26
maggioraz	%	10
VsecmaxOK		33,29
Vseff	**V**	**23,54**
Iseff	A	1,44
P	**VA**	**33,89**
Idiodi	A	0,4
IdiodiSicura	A	0,8
C	**mA**	**800**
IpiccoRipet	A	8
Vr	V	33,29
B	**V**	**30**

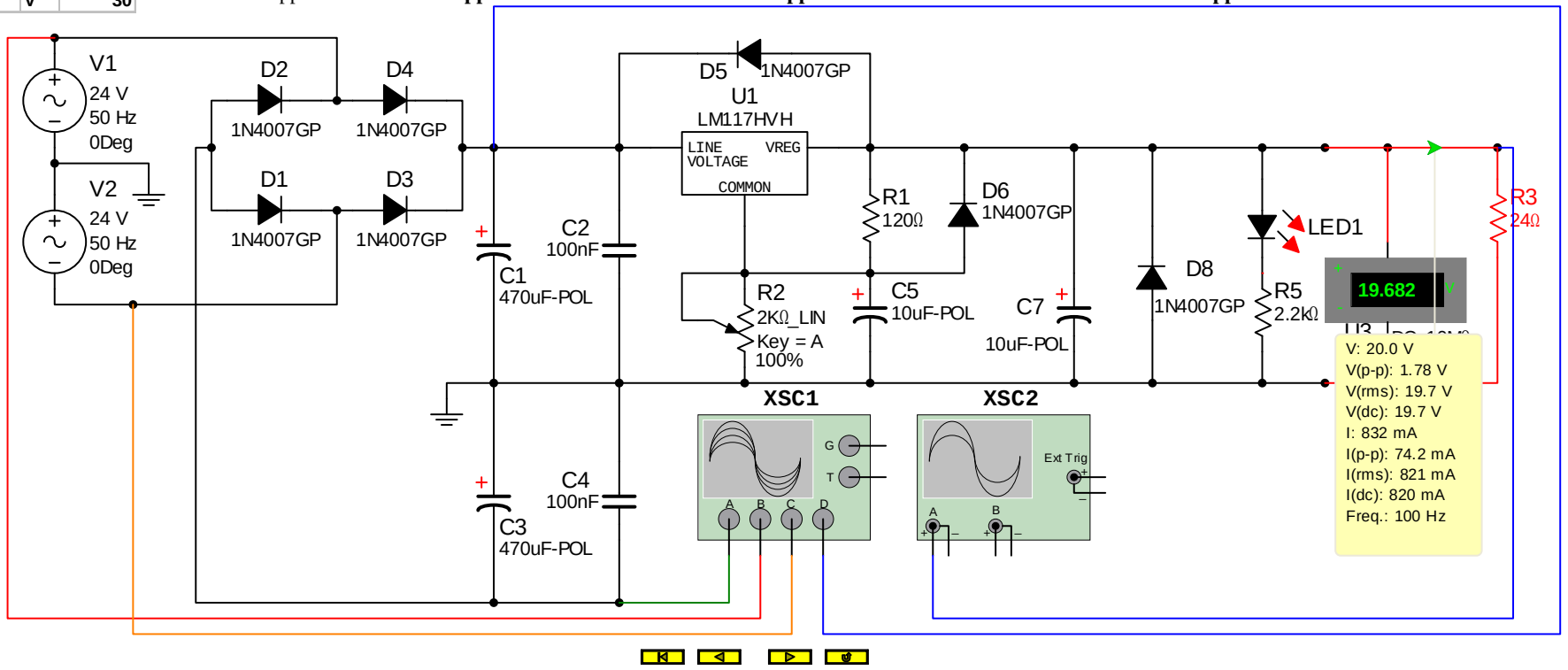








Il ripple è **solo di 1.3 mVpp** a 12 V in uscita – diventa di **6 mVpp** a 15 V in uscita – a 20 V di uscita è di ben **2 Vpp** !

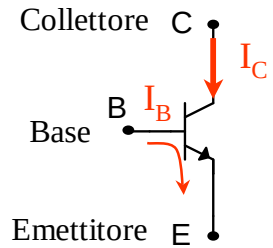


Introduzione al transistor BJT

(8')

Transistor di tipo **BJT** (**T**ransistor a **J**unction **B**ipolare)

(Ci sono altre due tipi di transistor: transistor **FET** e **MOS**)



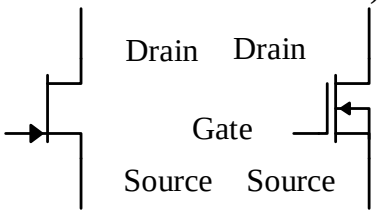
Per capire il funzionamento del BJT in questo circuito basta sapere due cose:

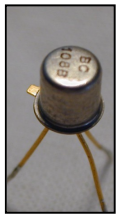
1) che tra B ed E c'è un diodo;

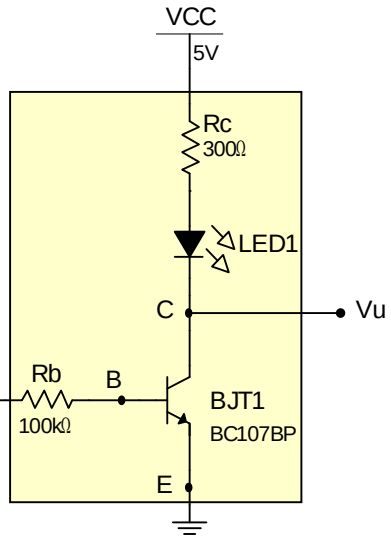
Se Vbb ≤ 0,5V => $I_B = 0$

Se Vbb > 0,5V =>

$$I_B = \frac{V_{BB} - 0{,}7}{R_B}$$



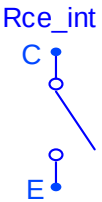


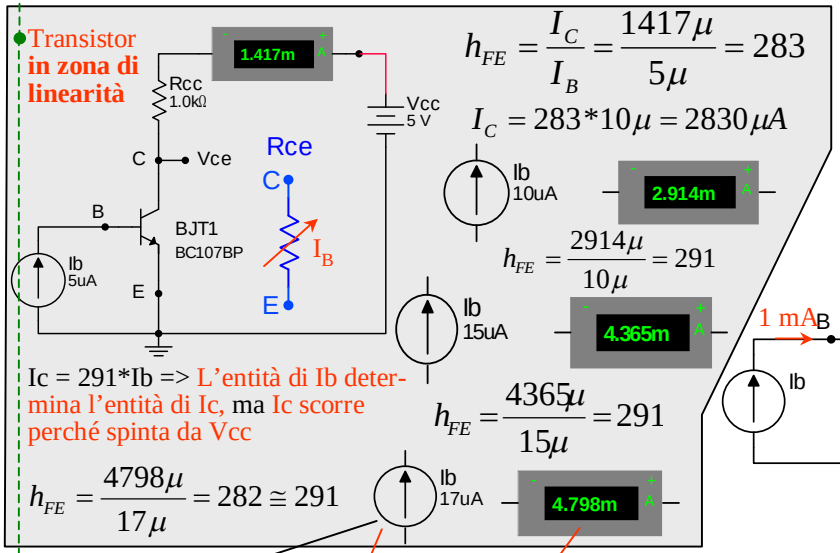


I 3 possibili stati del BJT

2) che I_C è proporzionale ad I_B ovvero che $I_C = k\ I_B$

$I_B = 0$ => $I_C = h_{FE} * 0 = 0$ Se $I_B = I_C = 0$ il BJT si dice **interdetto**





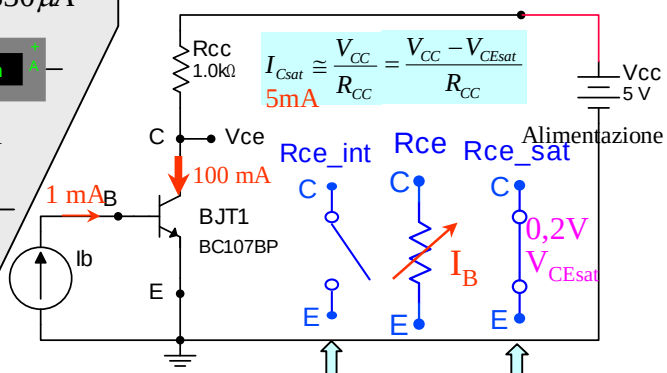
$$h_{FE} = \frac{I_C}{I_B} = \frac{1417\mu}{5\mu} = 283$$

$$I_C = 283 * 10\mu = 2830\mu A$$

$$h_{FE} = \frac{2914\mu}{10\mu} = 291$$

$$h_{FE} = \frac{4365\mu}{15\mu} = 291$$

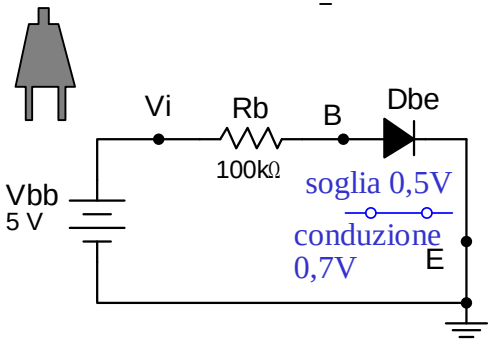
$$h_{FE} = \frac{4798\mu}{17\mu} = 282 \cong 291$$



$$I_{Csat} \cong \frac{V_{CC}}{R_{CC}} = \frac{V_{CC} - V_{CEsat}}{R_{CC}}$$

5mA

Alimentazione



$I_B = 0$

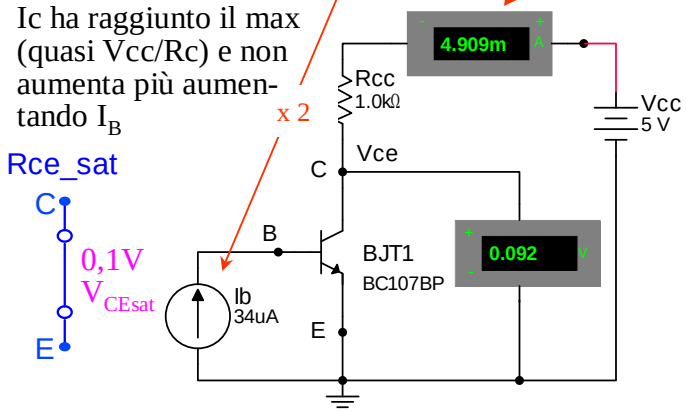
$$I_B \geq I_{Bsat} = \frac{I_{Csat}}{h_{FE\ min}}$$

$$I_{Bsat} = \frac{5000\mu}{291} = 17\mu A$$

Transistor **saturato:**
Ic ha raggiunto il max (quasi Vcc/Rc) e non aumenta più aumentando I_B

= (da 4.8 a 4.9mA)

x 2

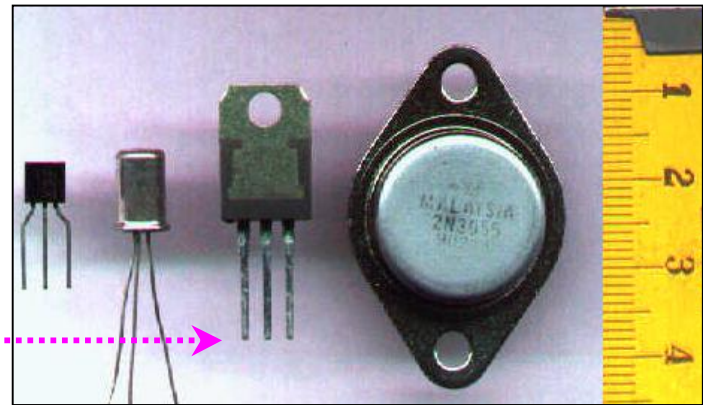


$I_B = h_{FE} * I_C$ (barrato)

$I_C = h_{FE} * I_B$

guadagno di corrente in continua del BJT

Valori di h_{FE} da 30 (nei **BJT di potenza**) a 500



Domande/risposte sul BJT con led

(5')

Collettore

Base

Emettitore

Per capire il funzionamento del BJT in questo circuito basta sapere due cose:

1) che tra B ed E c'è un diodo;

Se Vbb ≤ 0,5V => $I_B = 0$

Se Vbb > 0,5V => $I_B = \frac{V_{BB} - 0{,}7}{R_B}$

2) che I_C è proporzionale ad I_B ovvero che $I_C = k\ I_B$

I 3 possibili stati del BJT

$I_B = 0 \Rightarrow I_C = h_{FE} * 0 = 0$ Se $I_B = I_C = 0$ il BJT si dice **interdetto**

Transistor **in zona di linearità**

$h_{FE} = \frac{I_C}{I_B} = \frac{1417\mu}{5\mu} = 283$

$I_C = 283*10\mu = 2830\mu A$

$h_{FE} = \frac{2914\mu}{10\mu} = 291$

$h_{FE} = \frac{4365\mu}{15\mu} = 291$

Ic = 291*Ib => L'entità di Ib determina l'entità di Ic, ma Ic scorre perché spinta da Vcc

$h_{FE} = \frac{4798\mu}{17\mu} = 282 \cong 291$

Transistor **saturato** (I_C ha raggiunto il max - Vcc/Rc – e non aumenta più aumentando I_B

= (da 4.8 a 4.9mA)

x 2

Rce_sat: 0,1V V_{CEsat}

$I_{Csat} \cong \frac{V_{CC}}{R_{CC}} = \frac{V_{CC} - V_{CEsat}}{R_{CC}}$ 5mA

100 mA

1 mA

$I_B = 0$

$I_C = h_{FE} * I_B$

guadagno di corrente in continua del BJT

$I_B \geq I_{Bsat} = \frac{I_{Csat}}{h_{FE\min}}$

$I_{Bsat} = \frac{5000\mu}{291} = 17\mu A$

1) Siamo in grado di calcolare quanto vale I_{LED1} se BJT1 non è saturato?

No!

$I_{LED1} = I_{Rc} = \frac{V_{CC}}{R_C} = \frac{V_{Rc}}{R_C} = \frac{V_{Rc}}{300} = \frac{5 - 1{,}7 - V_{CE}}{300}$

2) Se BJT1 è saturato quanto vale I_{LED1} ?

$I_{LED1} = \frac{5 - 1{,}7 - V_{CEsat}}{300} = \frac{5 - 1.7 - 0.2}{300} = 10.3mA$

$I_{LED1} \cong \frac{5 - 1{,}7}{300} = 11mA$

3) Come stabilire se il BJT1 è saturato o no?

Se BJT1 è saturato

$I_{C\max} = \frac{5 - 1{,}7}{300} = 11mA = I_{Csat}$

$I_{Bsat} = \frac{I_{Csat}}{h_{FE}} = \frac{11000\mu}{291} = 38\mu A$

Se $I_B \geq I_{Bsat}$ il transistor è saturato

$I_B = \frac{V_{bb} - V_{be}}{R_B} = \frac{5 - 0.7}{100k} = 43\mu A$

43 µA > 38 µA => Transistor saturato

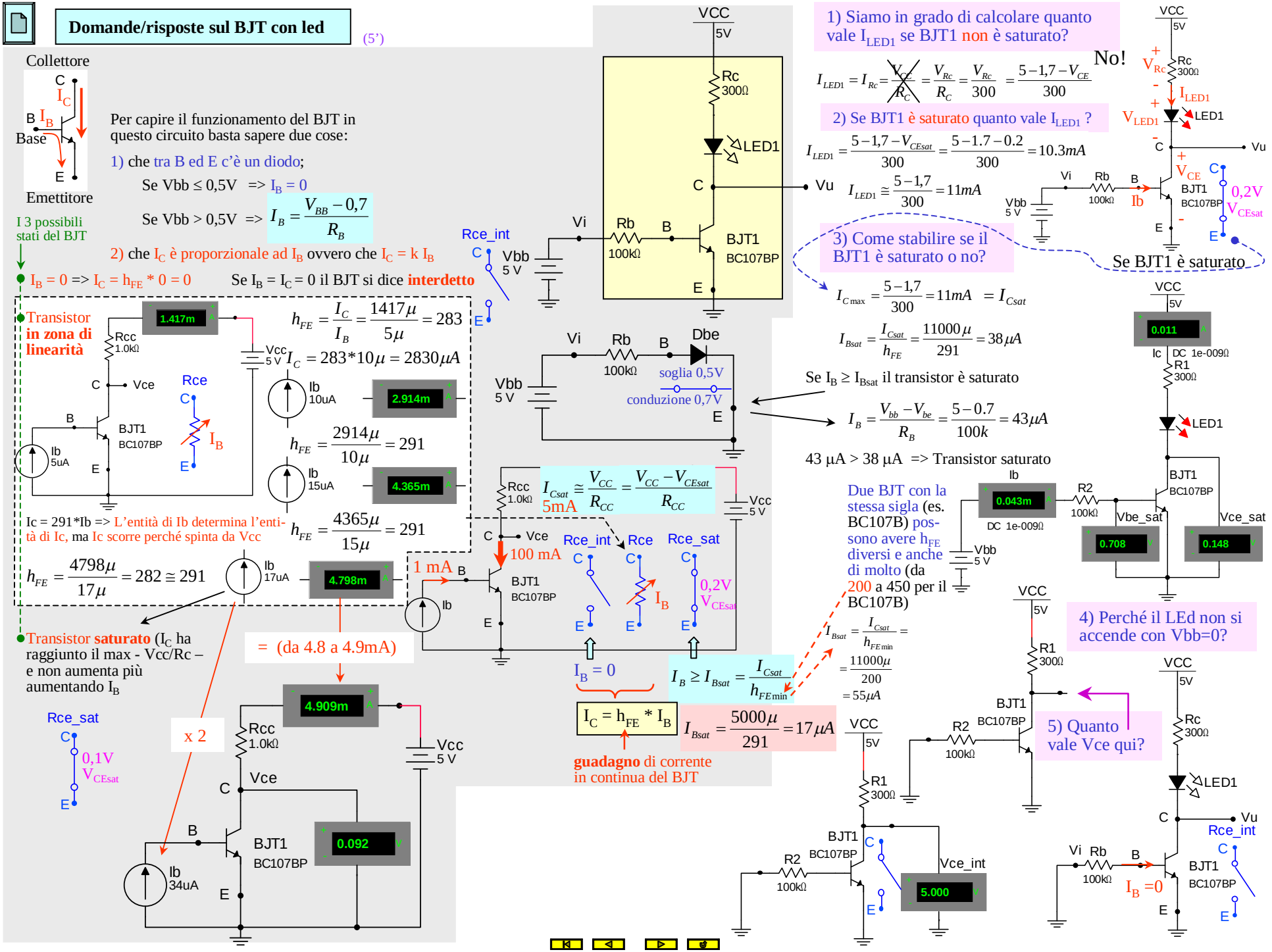
Due BJT con la stessa sigla (es. BC107B) possono avere h_{FE} diversi e anche di molto (da 200 a 450 per il BC107B)

$I_{Bsat} = \frac{I_{Csat}}{h_{FE\min}} = \frac{11000\mu}{200} = 55\mu A$

4) Perché il LEd non si accende con Vbb=0?

5) Quanto vale Vce qui?

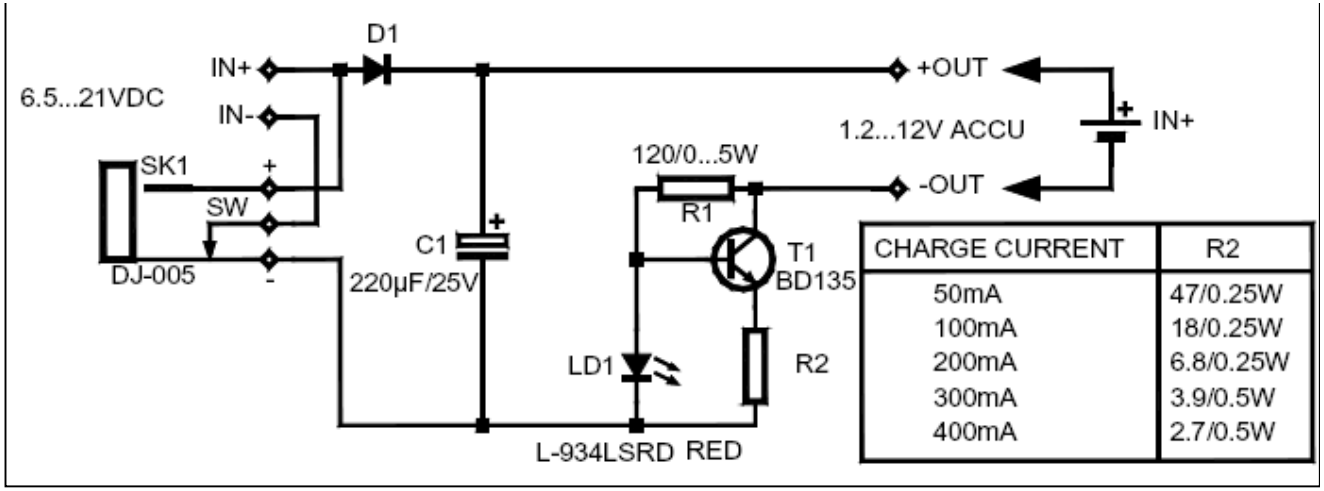
$I_B = 0$



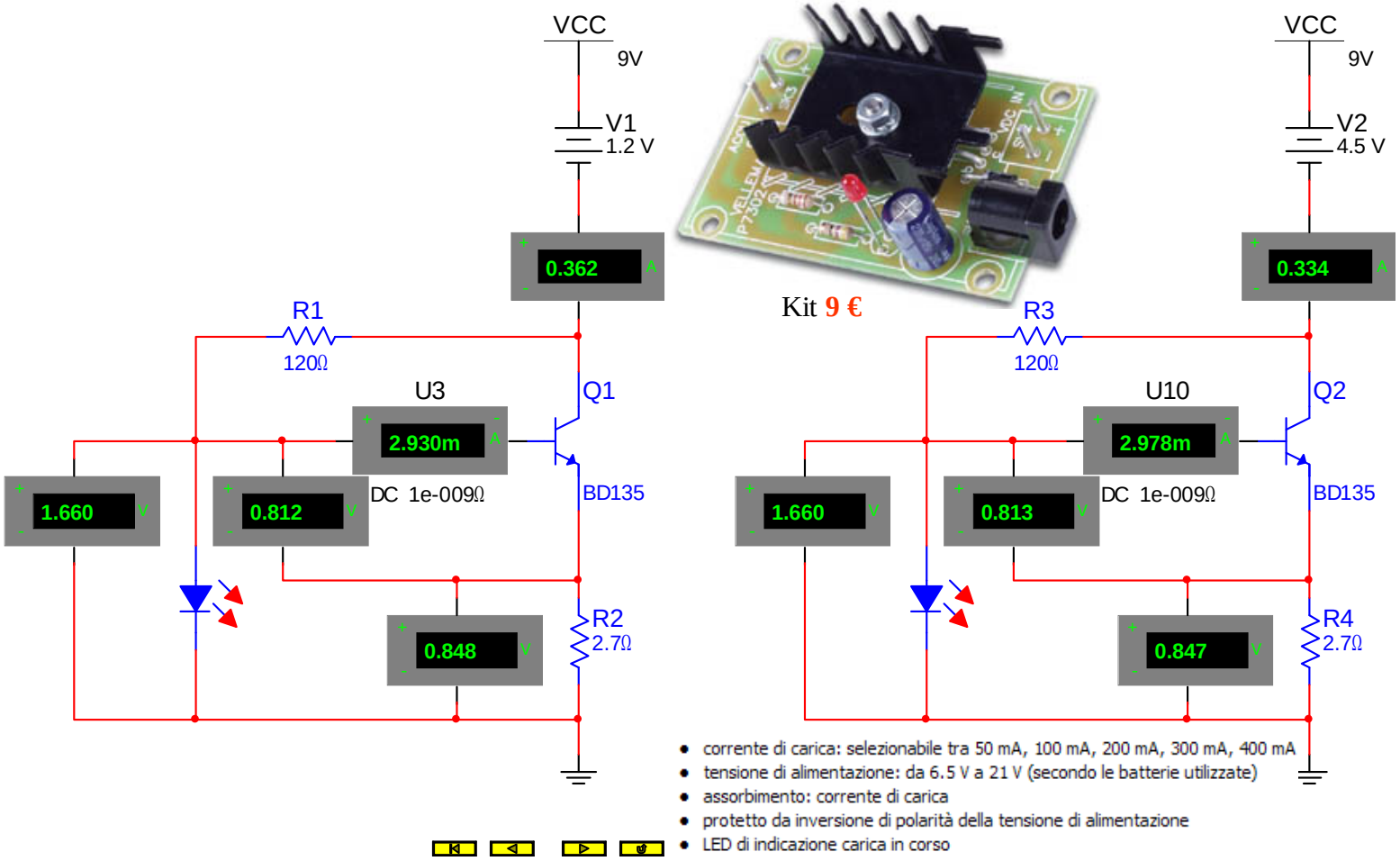
Carica batterie minimale

Funzione di tutti i componenti :

Prima bisogna capire come mai con batterie in carica di diversa tensione (qui V1 = 1,2V e V2 = 4,5V) passa quasi la stessa corrente (qui 350 mA perché R2 = R4 = 2.7 Ω)



CHARGE CURRENT	R2
50mA	47/0.25W
100mA	18/0.25W
200mA	6.8/0.25W
300mA	3.9/0.5W
400mA	2.7/0.5W



Kit **9 €**

- corrente di carica: selezionabile tra 50 mA, 100 mA, 200 mA, 300 mA, 400 mA
- tensione di alimentazione: da 6.5 V a 21 V (secondo le batterie utilizzate)
- assorbimento: corrente di carica
- protetto da inversione di polarità della tensione di alimentazione
- LED di indicazione carica in corso

Arresto e marcia

Sterzata

Robot che segue la luce

Kit Velleman MK127 **14.5 €**

Il robot è sostanzialmente un oggetto messo in movimento da due piccoli motori in continua funzionanti a 3 volt, posti inclinati in prossimità degli angoli anteriori; posteriormente, un piccolo rullo permette di far scorrere la coda. I motori sono disposti in modo che i loro perni tocchino il suolo stando inclinati di circa 45°, cosicché ruotando facciano muovere in avanti il tutto; allo scopo il perno di ciascuno ruota nel verso contrario rispetto a quello dell'altro.

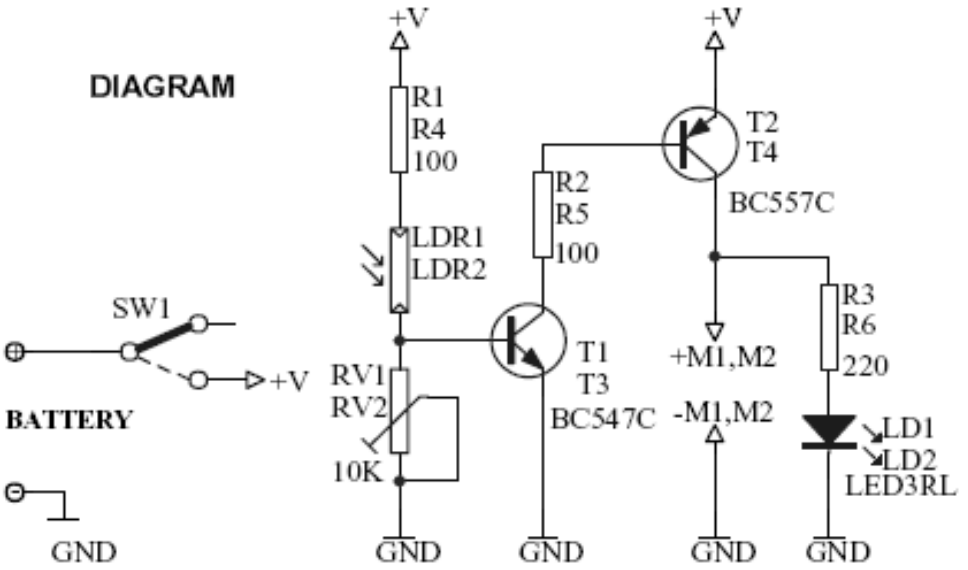
Siccome ciascuno di essi viene alimentato separatamente da un circuito a sé, non è detto che entrambi diano la stessa velocità: questo è alla base del cambiamento di direzione; infatti il piccolo automa gira da una parte o dall'altra a seconda di quale motore ruota più velocemente. Insomma, la cosa funziona come i cingolati, che sterzano rallentando o fermando uno dei cingoli.

La velocità di ciascun motore dipende essenzialmente da due fattori: la regolazione fatta mediante due trimmer e la quantità di luce che investe le fotoresistenze, che fungono da occhi. Il particolare movimento, il fatto che il robot insegua la luce e si fermi al buio, dipende dal collegamento dei motori, ciascuno dei quali è alimentato tramite un transistor, la cui corrente di base dipende dall'intensità della luce che investe la rispettiva fotoresistenza; notate che i circuiti di comando dei motori sono identici. Il motore di sinistra viene alimentato in base alla luce vista dall'occhio di destra e viceversa. In condizioni di oscurità ogni fotoresistore presenta un valore resistivo molto più alto di quello del relativo trimmer RV1 (o RV2) tanto che il transistor T1 (T3) non riesce a condurre e resta interdetto; così nemmeno T2 (T4) può condurre e il motore non viene alimentato, quindi rimane fermo. Il led LD1 (LD2), che fa da monitor dell'attività, è spento.

Quando la luce investe LDR1 (LDR2) con sufficiente intensità, la differenza di potenziale ai capi del trimmer RV1 (RV2) diviene abbastanza elevata da mandare T1 (T3) in saturazione, cosicché la corrente di collettore di quest'ultimo polarizza la base del T2 (T4) facendo condurre anche questi. Il collettore di T2 (T4) alimenta dunque il motore e, tramite la resistenza R3 (R6), il led LD1 (LD2). La velocità di rotazione del perno del motore dipende sia dall'intensità della luce incidente sulla superficie sensibile del fotoresistore, sia dalla resistenza assunta dal trimmer: maggiore è quest'ultima, più il motore gira alla svelta. Lo stesso dicasi per l'intensità luminosa.

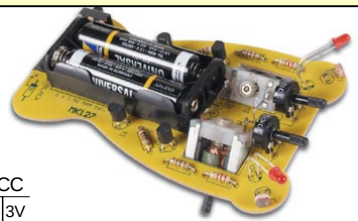
Ora riprendiamo il discorso lasciato in sospeso, quello sulla collocazione dei motori: LDR1 è posta dal lato opposto a quello dove si trova M1 (il motore controllato da T2); invece il led LD1, quello che monitorizza l'accensione di tale motore, è dalla stessa parte di LDR1. Lo stesso vale per LDR2, che sta dalla parte opposta a quello dove si trova M2, il rispettivo motore. Questa collocazione permette di far girare più rapidamente il perno del motore posto dal lato opposto a quello in cui viene rilevata più luce: ne deriva che il robot si sposta proprio nella direzione della fotoresistenza più illuminata. In condizioni normali, cioè in un locale ben illuminato, i motori devono avere la stessa velocità e il robot deve procedere diritto; allo scopo, in fase di messa a punto occorre registrare adeguatamente i trimmer.

L'intero oggetto funziona con due pile ministilo da 3 volt ed ha un interruttore con il quale può essere spento.



Robot che segue la luce: arresto e marcia

(4')



$I_C = h_{FE} * I_B$

Collettore

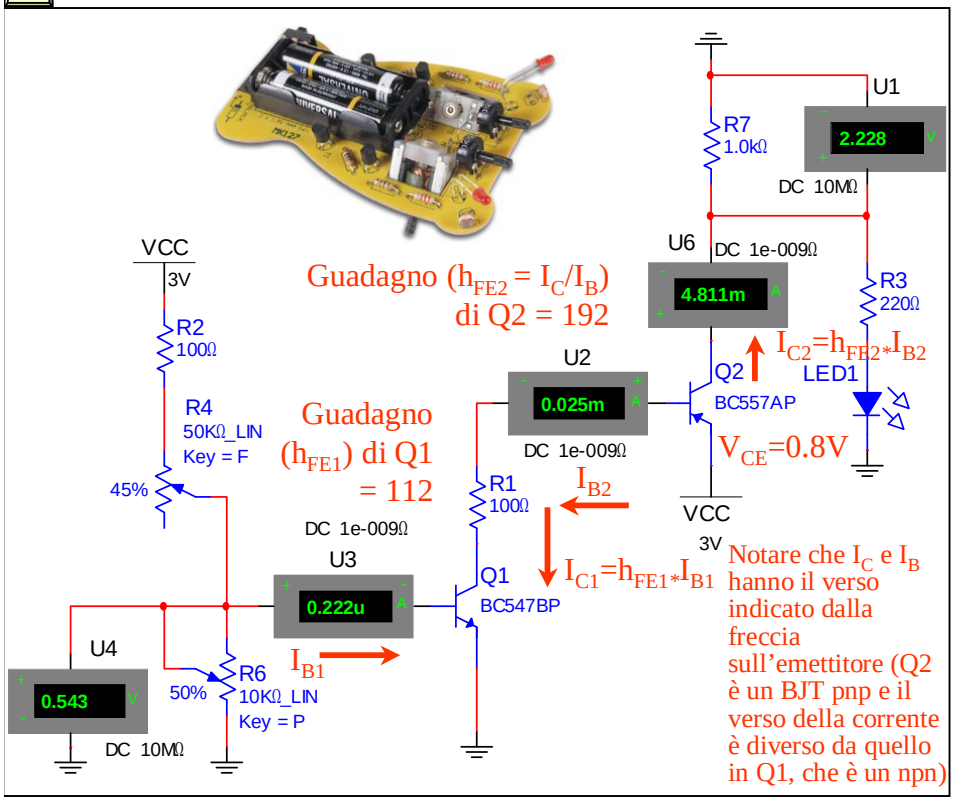
Base

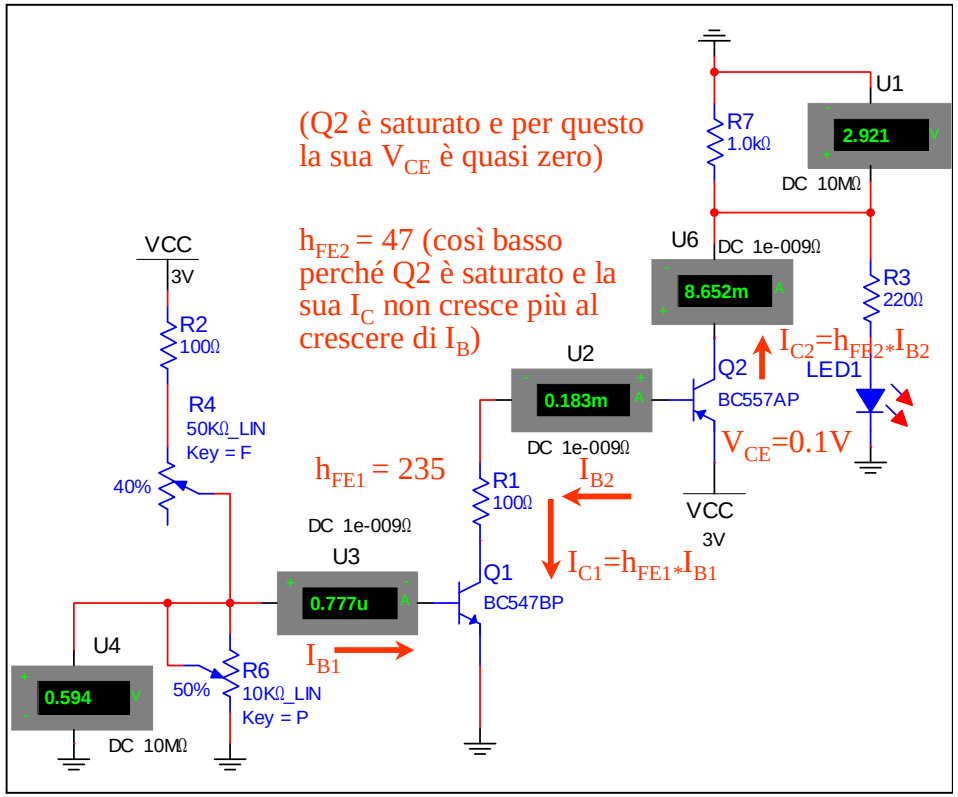
Emettitore

In assenza di luce (R4 grande), Q1 è interdetto (tasto aperto tra C ed E) e interdice Q2, che è aperto pure lui e non fa arrivare tensione al motore

In presenza di luce (R4 piccola), il motore gira perché la forte conduzione del transistor Q2 (saturazione di Q2) lo rende un tasto chiuso tra C ed E, per cui i 3 V arrivano al motore

Robot che segue la luce: sterzata (3')





Sensore (fotodiodo) con meno luce (R4 = 45% di 50K = 22.5K) | **Sensore (fotodiodo) con più luce (R4 = 40% di 50K = 20K)**

$V_{BE\ di\ Q1}$ = 543 mV < $V_{BE\ di\ Q1}$ = 594 mV

I_{B1} = 222 nA < I_{B1} = 777 nA

I_{C1} = Guadagno1 * I_{B1} = 112 (a I_B basse guadagna meno) * 222n = 25µA < I_{C1} = Guadagno1 * I_{B1} = 235 (qui guadagna di più) * 777n = 183µA

I_{C2} = Guadagno2 * I_{B2} (= I_{C1}) = 192 * 25µ = 4.8mA < I_{C2} = Guadagno2 * I_{B2} (= I_{C1}) = 47 (Q2 è saturato) * 183µ = 8.6mA

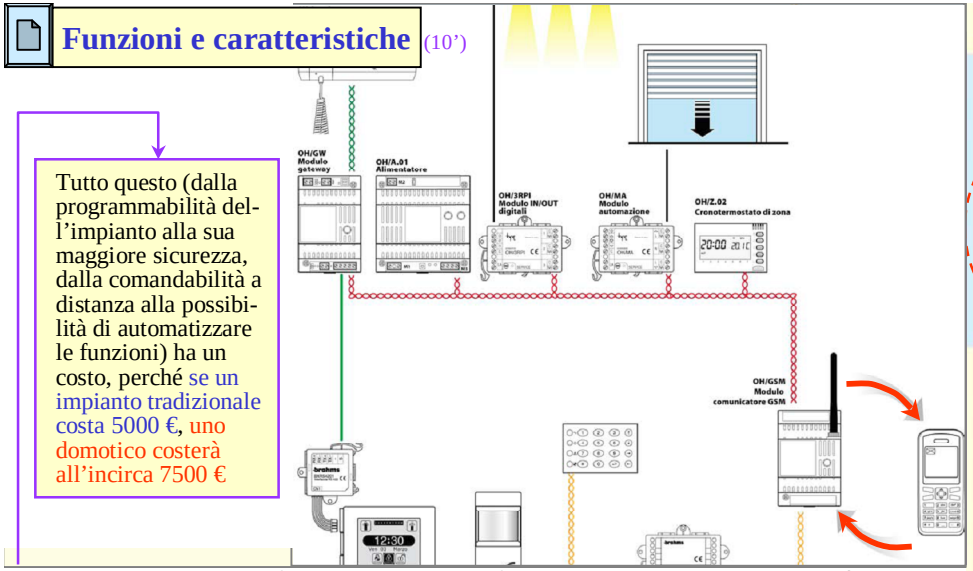
Il LED1 non ce la fa ad accendersi (servono almeno 5 mA) | Il LED1 si accende

I_C grande => V_{CE} piccola e V_{CE2} = 0.8 V > V_{CE2} = 0.1 V

V_{R7} ($V_{MOTORE1}$) = 2.2 V < V_{R7} ($V_{MOTORE2}$) = 2.9 V (quasi tutta l'alimentazione, che è di 3V)

Il MOTORE 1 (a sinistra) gira più piano (N_{giri} è proporzionale alla V_{MOTORE}) **Il MOTORE 2 va più forte e fa sterzare dalla sua parte il robot**

Funzioni e caratteristiche (10')



Tutto questo (dalla programmabilità dell'impianto alla sua maggiore sicurezza, dalla comandabilità a distanza alla possibilità di automatizzare le funzioni) ha un costo, perché se un impianto tradizionale costa 5000 €, uno domotico costerà all'incirca 7500 €

La **DOMOTICA** (dal francese *domotique*, fusione tra *domus* 'casa' e *informatique* 'informatica') si propone di ottimizzare la gestione degli impianti presenti in una casa (*Home Automation*) o in un insieme di edifici dove vivono o lavorano molte persone (*Building Automation*). Essa può essere applicata (e lo sarà sempre di più perché migliora la vita):

- all'**illuminazione**;
- alle **aperture motorizzate** (come le tapparelle motorizzate);
- al **riscaldamento/raffreddamento**;
- alle **temporizzazioni** (ad es. attivazione notturna di un sistema);
- alla gestione di **scenari**, periodici (situazioni che si ripetono ogni giorno) o legati a eventi;
- alla gestione del **videocitofono**;
- all'**irrigazione**;
- al **controllo dei carichi elettrici**, per evitare il sovraccarico o per risparmiare sui consumi;
- ai **sistemi anti-intrusione**;
- al **controllo a distanza** (per es. via cellulare).

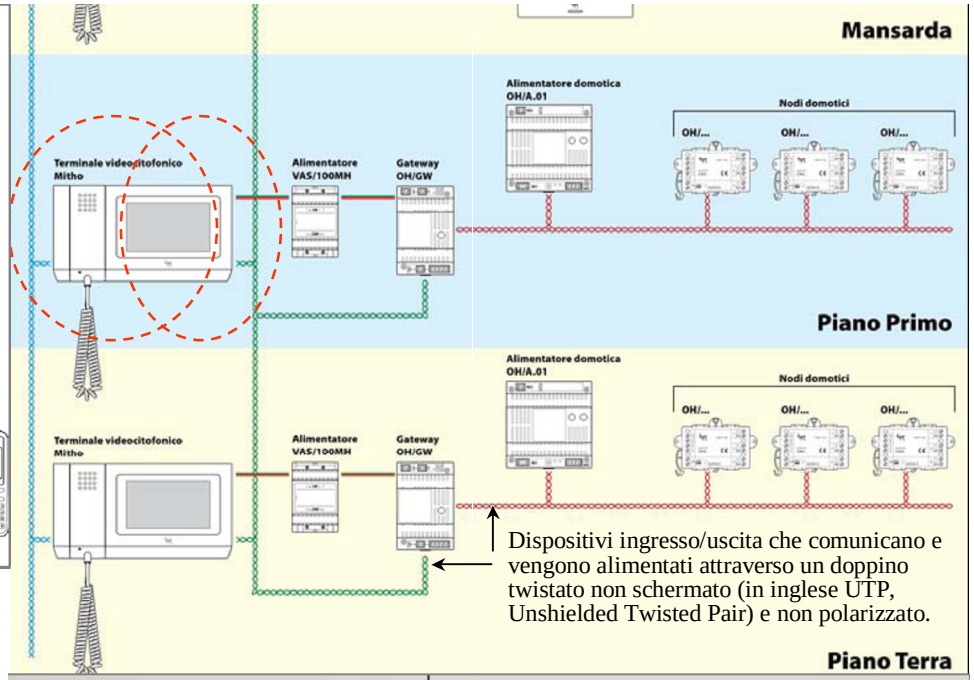
BPT utilizza il protocollo definito dallo **standard ISO/IEC – 14908** "Open Data Communication in Building Automation, Controls and Building Management" (protocollo chiamato prima LonTalk e oggi LonWorks), il più diffuso nel mondo per applicazioni di domotica. «È infatti stimato che circa il 65% dei sistemi di automazione di edifici in Europa sia basato su questa tecnologia» (wikipedia). Standard europeo per la building automation dal 2005; standard nazionale cinese dal 2006; Rico-nosciuto dall'ente europeo dei costruttori di elettrodomestici dal 2007.

Impianto programmabile e senza tensione di rete sui comandi

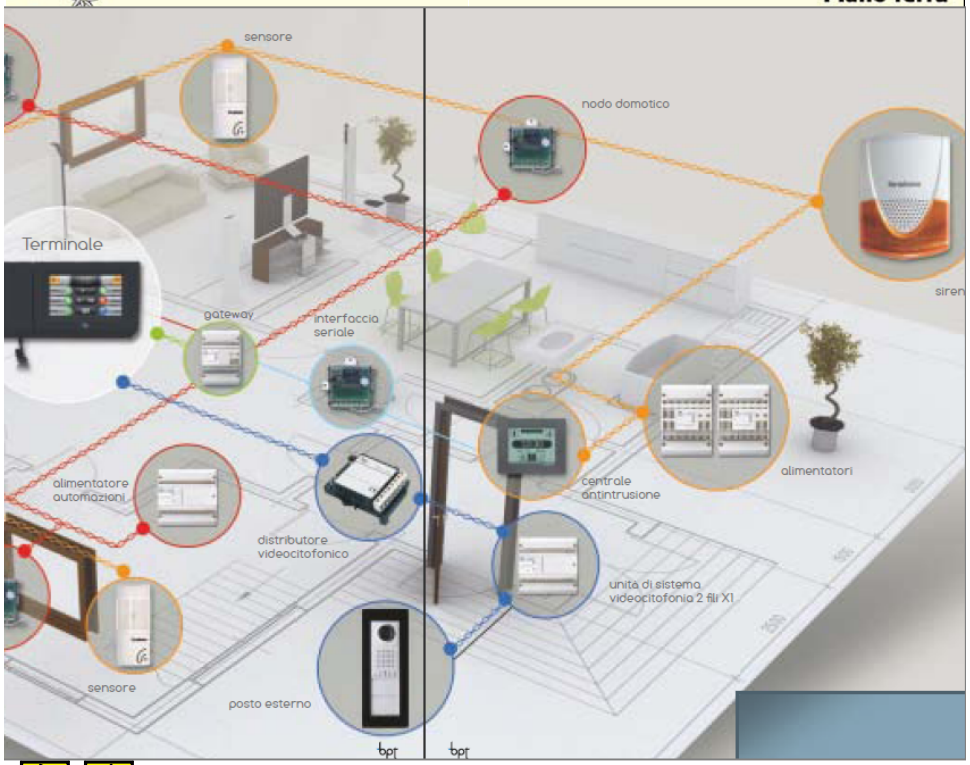
1) Un Tasto1 oggi adibito al comando della Luce1 **può in qualunque momento essere adibito al comando di qualunque altra funzione comandabile dall'impianto** perché non c'è un filo che va dalla posizione del Tasto1 alla posizione della Luce1.

Il Tasto1 è collegato al bus e l'evento Tasto1_premuto via bus arriva alla centrale di controllo. È la **programmazione** della centrale di controllo che specifica cosa deve essere fatto quando il Tasto1 viene premuto, non importa in quale angolo della casa perché ogni attuatore è collegato al bus e può eseguire l'azione prevista dalla programmazione per quell'evento

La tensione di rete arriva agli attuatori posizionati in prossimità dei carichi ma non arriva a nessun Tasto di comando, con un aumento importante della **sicurezza**



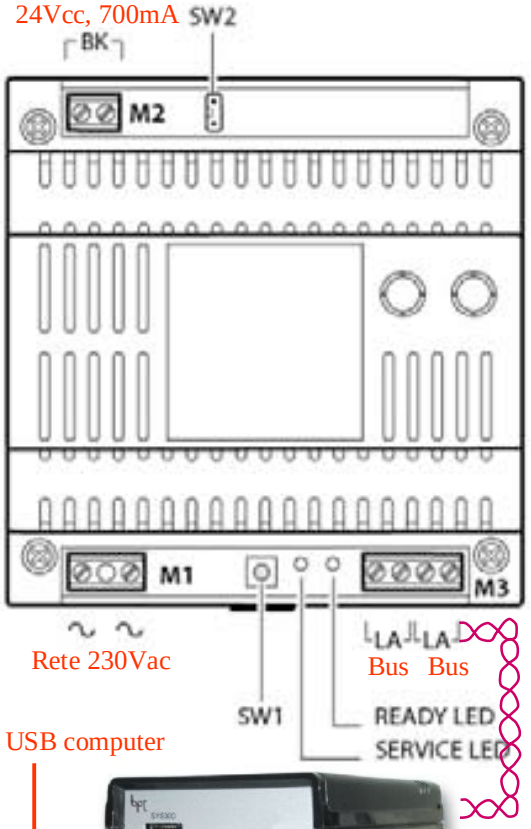
Dispositivi ingresso/uscita che comunicano e vengono alimentati attraverso un doppino twistato non schermato (in inglese UTP, Unshielded Twisted Pair) e non polarizzato.

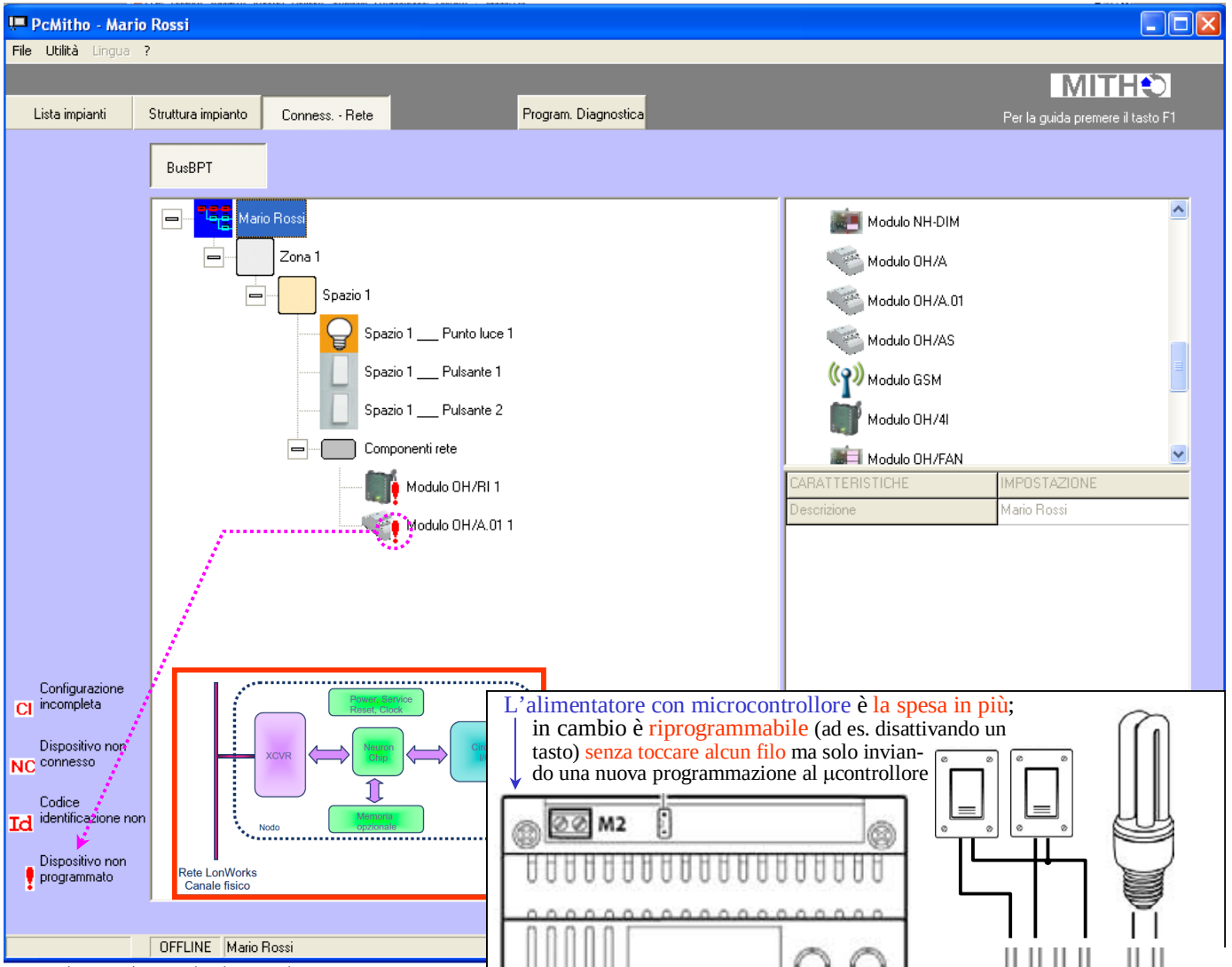


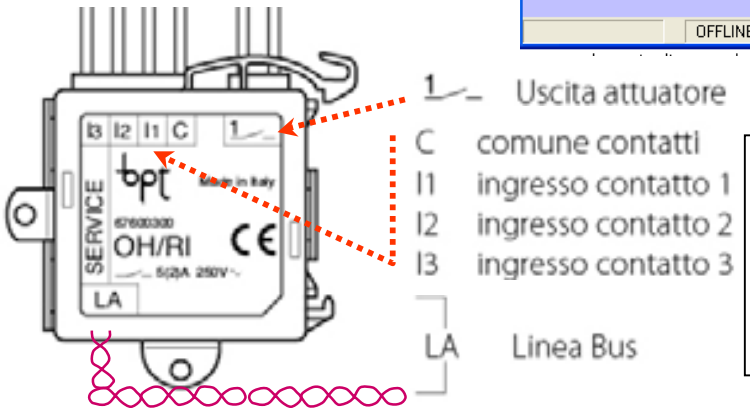
Creazione del 1⁰ impianto

(10')

Alim. di soccorso
24Vcc, 700mA

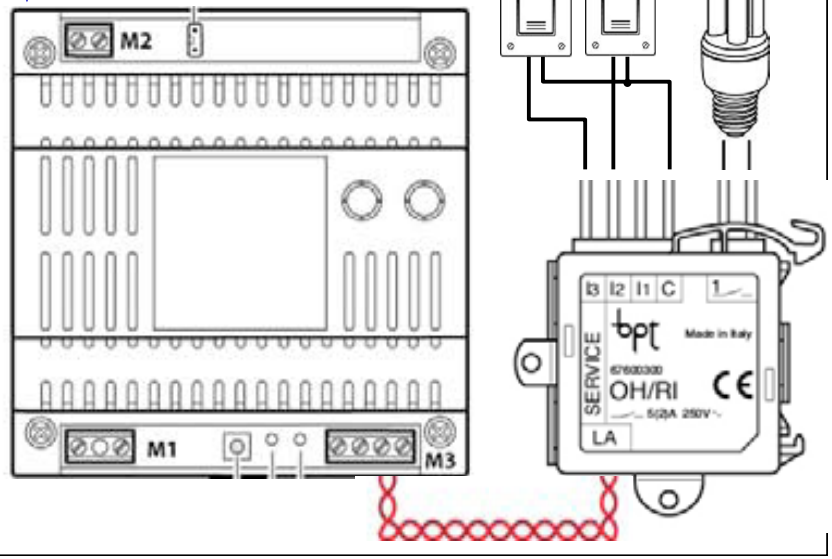




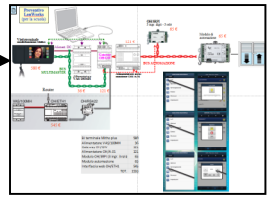


Un **impianto complesso** è composto da *molte* sottoparti come quella che abbiamo visto qui, ma per programmarlo basta ripetere per ognuna di esse i passaggi visti qui.

L'alimentatore con microcontrollore è la spesa in più; in cambio è riprogrammabile (ad es. disattivando un tasto) senza toccare alcun filo ma solo inviando una nuova programmazione al µcontrollore

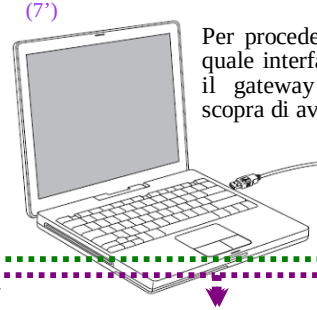


1⁰ impianto completo



Prezzi 2012 di un sistema minimo Bpt per usi didattici

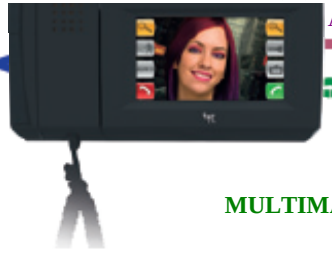
(7')



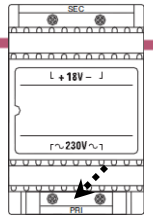
Per procedere adesso devo sapere a quale interfaccia seriale ho collegato il gateway e supporremo che io scopra di aver usato la COM13

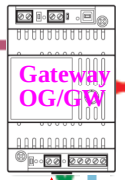
Cavo USB da stampante

Viedoterminale multifunzione Mitho

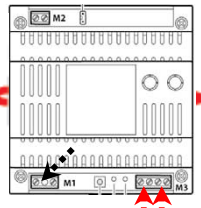


Aliment. DC





Gateway OG/GW

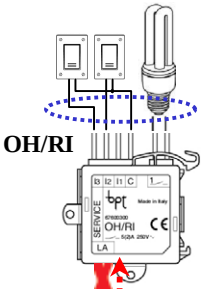


Alimentatore automazione OH/A.01

Alimentatore VAS/100MH

BUS MULTIMASTER

BUS VIDEOCITOFONIA



OH/RI

OH/6I
6 ingr. digitali

OH/3RPI
3 ingr. digit - 3 relè

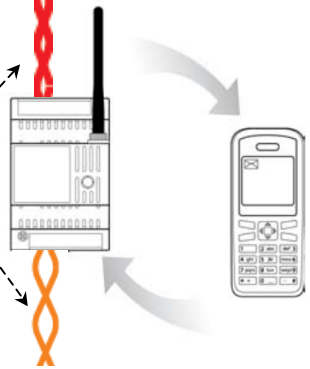
OH/Z.02

Crono-termostato

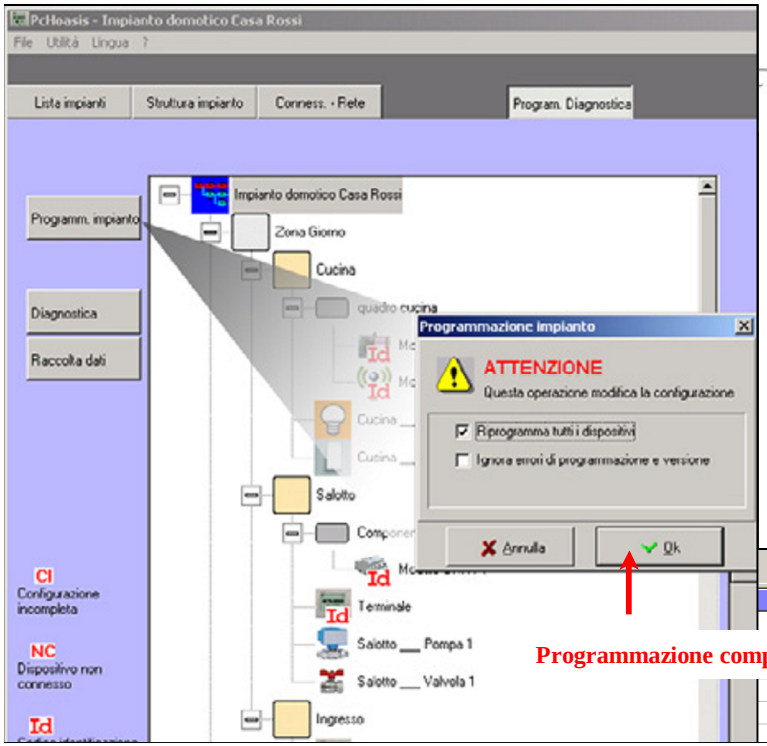
50÷100 €
ogni nodo

BUS AUTOMAZIONE

OH/GSM Per ricevere SMS o per farli inviare dal bus automazione o dal bus antintrusione



Circa 500 €

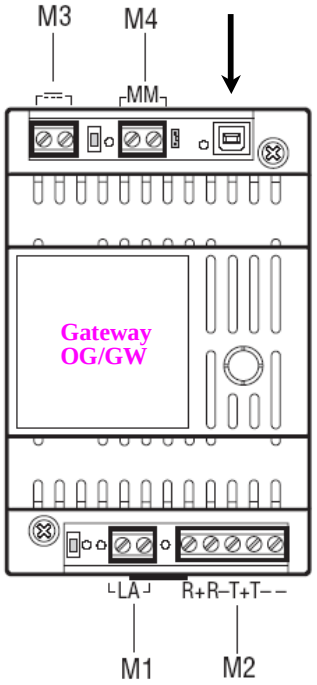


Programmazione completata

INTERFACCIA SERIALE BXRS4201

UNITÀ CENTRALE B2UC0002

BUS ANTIINTRUSIONE



MM Bus MultiMaster

Morsettiera M2

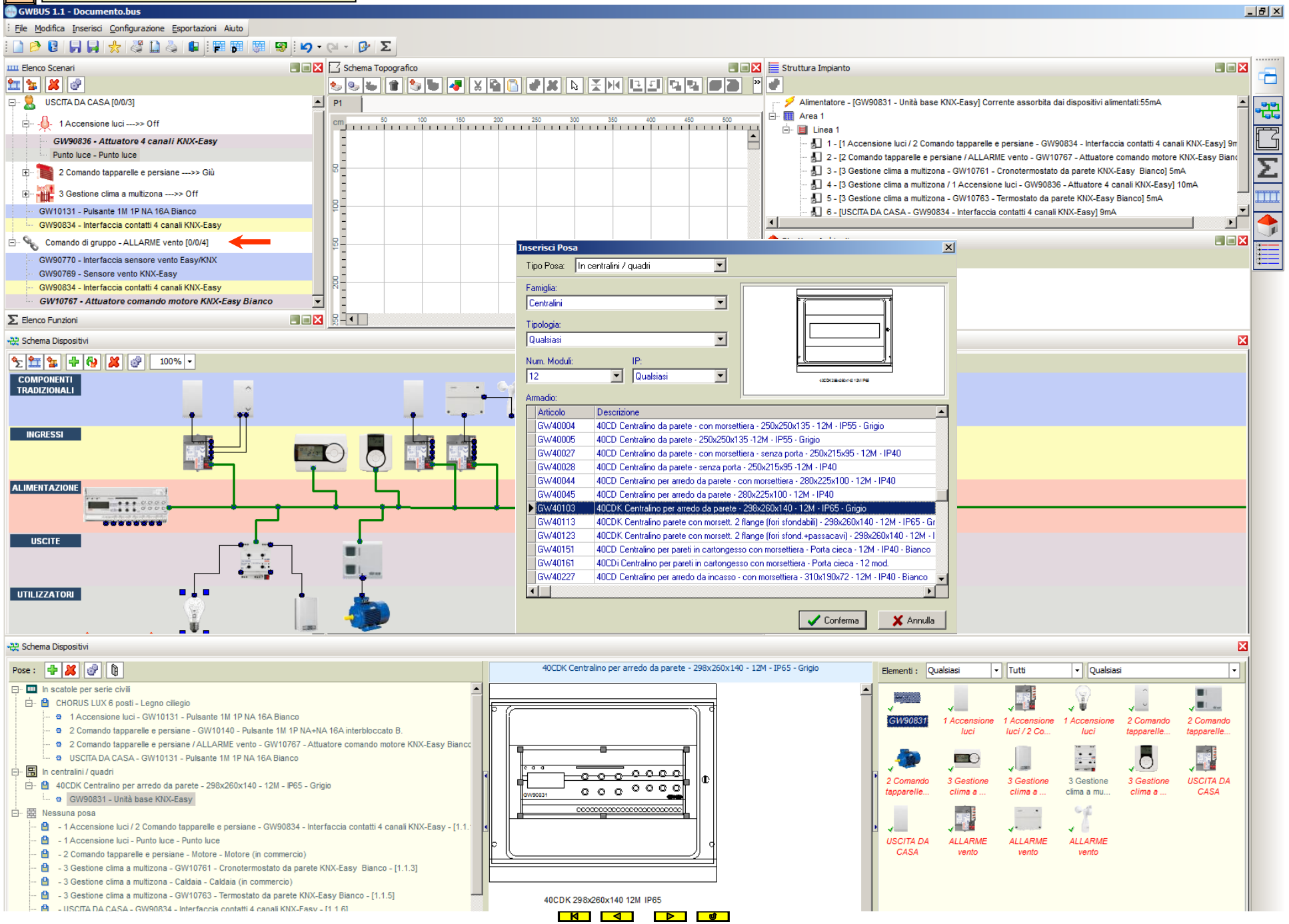
R+ Collegamento al
R– bus RS422 per
T+ impianti antintrusione B2
T–
– Massa comune

⎓ Alimentazione 12÷24 V DC

LA Linea Bus automazione elettrica

Impianto KNX con GWBUS 1.1

(6')



Dalla tesi di laurea "Microcontrollori in Rete per la Domotica" dello studente di Ingeneria informatica Domenico Della Ratta, AA 2001-2002 (http://www.ing.unisannio.it/fiengo/Tesi/Domenico%20Della%20Ratta/Tesi_Mimmo.doc)

Considerazioni sulla diffusione della domotica

Anche se si parla di automazione domestica già da tempo, si può affermare che essa ha realmente stimolato l'interesse dei produttori e degli utenti solo negli ultimi anni, diventando una nuova industria di prodotti per il consumatore.

I requisiti che un buon sistema di home automation dovrebbe avere sono:

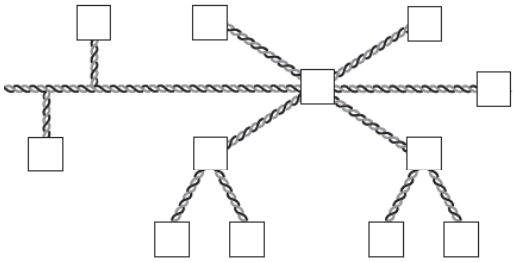
1) Basso costo, inteso come economicità delle periferiche e della rete di interconnessione, unitamente alla semplicità di installazione della stessa
[*il sistema presentato qui fa lievitare il costo da 5000 => a 7500 €*].

2) Flessibilità intesa come modularità. La vita media di un impianto elettrico domestico è molto lunga: per questo motivo si richiede che nel tempo possono essere effettuate delle modifiche introducendo nuove periferiche o eliminandone delle vecchie in modo indolore per l'utente. [*Il sistema presentato qui permette di programmare i componenti aggiunti anche se non si ha più il programma installato originariamente non spuntando la casella "Riprogramma tutti i dispositivi"*]

3) Integrazione di un vasto numero di applicazioni.

- all'illuminazione;
- alle aperture motorizzate (come le tapparelle motorizzate);
- al riscaldamento/raffreddamento;
- alle temporizzazioni (ad es. attivazione notturna di un sistema);
- alla gestione di scenari, periodici (situazioni che si ripetono ogni giorno) o legati a eventi;
- alla gestione del videocitofono;
- all'irrigazione;
- al controllo dei carichi elettrici, per evitare il sovraccarico o per risparmiare sui consumi;
- ai sistemi anti-intrusione;
- al controllo a distanza (per es. via cellulare o via internet).

4) Capacità di supportare diversi mezzi di comunicazione come i raggi infrarossi, la radio frequenza, il doppino intrecciato, e la linea di alimentazione.



[*Il bus trasmissivo in Lonworks può essere il doppino telefonico, la radiofrequenza, le onde convogliate. La rete può avere parti a bus, parti a stella e parti ad albero*]

5) Facilità di utilizzo. Qualsiasi utente deve essere in grado di utilizzare al massimo le potenzialità dell'impianto con il minimo sforzo. [*10 + 7 minuti di audio non sono molti*]

6) Possibilità di interagire con apparecchi di marca diversa.
[*Il sistema domotico BPT è compatibile con impianti elettrici di qualsiasi marca*]

7) Plug and Play. Se possibile, infatti, si vorrebbe evitare l'intervento di un tecnico specializzato ogni volta che devono essere apportate delle modifiche all'impianto.

Molti di questi requisiti sono purtroppo in contrasto tra loro e questo è il motivo che ha portato i produttori a sviluppare delle soluzioni spesso estremamente diverse tra di loro.

Alcuni hanno fatto dell'economicità il requisito fondamentale del loro sistema, spesso rinunciando all'implementazione di caratteristiche che avrebbero aumentato la soddisfazione dell'utente finale. Altri, invece, hanno sviluppato sistemi capaci di integrare un numero molto elevato di applicazioni, mettendo in secondo piano l'aspetto economico.
La conseguenza di ciò è che il prodotto di un'azienda spesso è *incompatibile* con quello di una concorrente, dando vita ad uno scenario di mercato estremamente eterogeneo. Inoltre, questa differenza di vedute è probabilmente la causa fondamentale del **mancato affermarsi di uno standard nel settore**, nonostante si parli di automazione domestica già da circa venti anni.

Già dal 1984, difatti, le organizzazioni americane e internazionali hanno realizzato standard di comunicazione per automazione domestica. Ci si aspettava che questi standard fornissero un'infrastruttura che potesse stimolare lo sviluppo dei prodotti e del mercato. Tuttavia, tale sviluppo fu molto lento e questo rallentò il lavoro di completamento degli standard.

Nei primi anni '90 gli standard cominciarono a diventare popolari. Allo stesso tempo Internet divenne popolare ed alcune compagnie decisero di creare dei nuovi standard privati per sviluppare applicazioni basate su TCP/IP. Altre compagnie formarono dei consorzi privati per realizzare tecnologie di nicchia, come la trasmissione dei dati sui cavi telefonici esistenti. Tutte queste organizzazioni acquisiscono ed abbandonano in continuazione partecipanti col risultato che **lo sviluppo dei prodotti è confuso** e non è concentrato su un'unica stabile infrastruttura.

Il problema fondamentale dello sviluppo dell'automazione domestica è convincere il consumatore stesso dell'effettiva necessità del prodotto: un tempo i telecomandi per i televisori erano considerati un optional, oggi è impossibile vendere un apparecchio che non ne sia dotato. Invece, lo sviluppo di sistemi per l'automazione domestica è estremamente complicato a causa del fatto che l'utente è scarsamente informato dei benefici e delle possibilità del sistema. La domanda di sistemi domestici è quindi latente.

Potenziali acquirenti potrebbero essere:

1) Persone adulte che vogliono risparmiare tempo […].

2) Persone anziane che desiderano assistenza per sopperire dei limiti fisici. Negli ultimi anni in America la popolazione oltre i 65 anni sta crescendo rapidamente, raggiungendo circa i 35 milioni di persone, e si prevede arrivi ai 53 milioni entro il 2020. Le persone che appartengono a questa fascia di età spesso hanno una o più disfunzione che limita le loro azioni. Nonostante ciò oltre il 95% di essi desidera continuare a vivere in casa propria.

3) Utenti che passano molto tempo fuori casa, e che vogliono comunicare e controllare l'ambiente domestico dal posto di lavoro.

Tuttavia, considerando l'ampia gamma e le diversità dei desideri degli utenti risulta difficile sviluppare un sistema capace di ottimizzare i bisogni di tutti.

La chiave dello sviluppo di un prodotto vincente [7] è fornire un prodotto che sia facilmente adattabile ai requisiti futuri, competitivo nei costi e [… con] una complessità interna invisibile all'utente che deve poter realizzare semplici procedure con un minimo addestramento.

3+6 sistemi domotici a confronto

(8')

http://www.clichome.it/domotica_confronto.php

	Connex (KNX)	LonWorks	My Home						
Luci e automazioni	**5** - Gestisce direttamente e completamente tutti i principali tipi di dimmer e i principali sistemi di gestione luci come DMX e DALI.	**4** – Ha attuatori per quasi ogni esigenza di luci e gateway per il bus DALI, ma non c'è nessuna tipologia di controllo per DMX	**3** – Molti i componenti a catalogo. Interfacce 1-10V e DALI ma non DMX512 (essenz. per contr led RGB)						
Clima	**4** - Gestisce direttamente valvole a posizionamento o PWM, pompe, sist. di ricircolo. Esistono sonde di ogni tipo e interfacce x i sistemi più utilizzati come Daikin, Mitsubishi, Toshiba, McQuay. Interfacce x tutti gli altri protocolli	**5** – Molti HVAC (*Heating, Ventilation and Air Conditioning*) nascono x LonWorks e comunque ogni produttore di climatizzatori ha un'interfaccia Lon. Inoltre può controllare valvole proporzionali e fan-coil.	**2** – Controlla elettrovalvole e fan-coil ma non si integra con i canalizzati o coi sist. a espansione diretta e la recente interfaccia IR migliora di poco la situazione xché l'IR è unid.						
Supervisione e controllo	**5** – Ampia scelta di supervisori da 2,3'' a 22'' e oltre. Molti SW, anche su misura. Molte App x iPone e x WebServer	**4** – I SW sono potenti perché nascono x l'industria e il terziario avanzato, ma l'estetica non è il max e scarseggiano le interfacce per Apple	**4** – Touch da 15'' libero nella programmazione e web-server blindato ma completo. 1 ottimo con. iPhone						
Sist. antifurto/sicurezza	**4** – Sono supportati i principali sistemi antifurto	**3** – Molte delle migliori centrali anti-intrusione e anti-incendio sono Lon ma se non lo sono può essere impossibile integr	**2** – C'è l'anti-intrusione MyHome, ma altri sistemi si integrano in modo assai limitato						
Audiovideo multiroom	**3** – C'è un sistema multiroom gestibile da ogni centralina Knx ma l'integrazione è limitata all'IR o ad un supervisore comune via seriale	**1** – Non ci sono componenti audio-video su LonWorks e l'integrazione può essere realizzata solo a livello di supervisione	**2** – Il multiroom MyHome è scadente come audio e limitato. Altri non sono integrabili nemmeno via IR						
Home Cinema	**2** – È limitata all'uso di un unico supervisore	**1** – Non c'è nessuna integrazione Home Cinema	**1** – Nessuna possibilità di controllare componenti HomeC						
Sensori evoluti	**5** – C'è tutto per tutto, dalla composiz. dell'acqua all'analisi dell'aria e alla T del solaio.	**4** – C'è quasi tutto per tutto. Eccellente il sist. di misura e controllo dei consumi energetici	**3** – La maggior parte dei sensori evoluti utili in casa ci sono. Nessun iA						
Controllo accessi	**4** – Ci sono diversi sistemi di controllo degli accessi	**4** Ci sono sistemi di controllo accessi integrati, ma non sono integrabili i sistemi + diffusi	**1** – Non è previsto nessun controllo di accesso						
Complessivamente	**4 (8 su 10)**	**3,25 (7 su 10)**	**2,25 (4½ su 10)**						
Costo	**medio** Prezzi scesi molto; ora sono molto competitivi e con una qualità/prezzo molto buona.	**medio** Costo allineato alla concorrenza	**contenuto** con ottimo rapporto qualità/prezzo						
Affidabilità	**5** - Qualità e affidabilità molto elevata.	**5** – I prodotti Echelon LonWorks sono affidabili nelle cond. + estreme	**4** – Alta x i comp. di qualità e l'intelligenza distribuita						
Flessibilità	**5** – É possibili realizzare davvero qualsiasi funzione.	**5** – Programmabile senza limiti e in grado di soddisfare qualsiasi esigenza	**2** – Programmazione molto limitata.						
Espandibilità futura	**4** – Ottima, ma penalizzata dall'bbandono delle soluzioni senza fili	**5** – Espansione sempre possibile e senza stendere nuovi cavi	**2** – Espand. futura è limitata e può servire un nuovo bus						
Ambiti applicativi elettivi	Residenziale, Terziario, Alberghiero, Yachting, Home e Buildibg Autmation in genere	Terziario ed Industriale	Automazione domestica (ottimo come sistema entry level)						

Una CPU in azione

(14')

Poiché la **CPU** (**Central Processing Unit**) è un integrato chiamato microprocessore in grado di eseguire un programma (memorizzato nella **memoria centrale** o **RAM** che è una memoria volatile che si cancella allo spegnimento ma veloce quanto la Cpu), vediamo le operazioni necessarie per eseguire il **programma** composto dalle seguenti tre istruzioni: X = X + Y ; stampa X ; end.

Il **program counter** (**PC**) è un **registro**, ovvero una memoria interna alla CPU, e contiene l'indirizzo della prossima istruzione da eseguire.

L'indirizzo a cui vuole operare la CPU viene comunicato all'esterno scrivendolo nel registro **MAR** (**memory address register**)

ALU (Arithmetic Logic Unit) è la parte della CPU che esegue le operazioni logiche (and, or, not…) e le 4 operazioni aritmetiche.

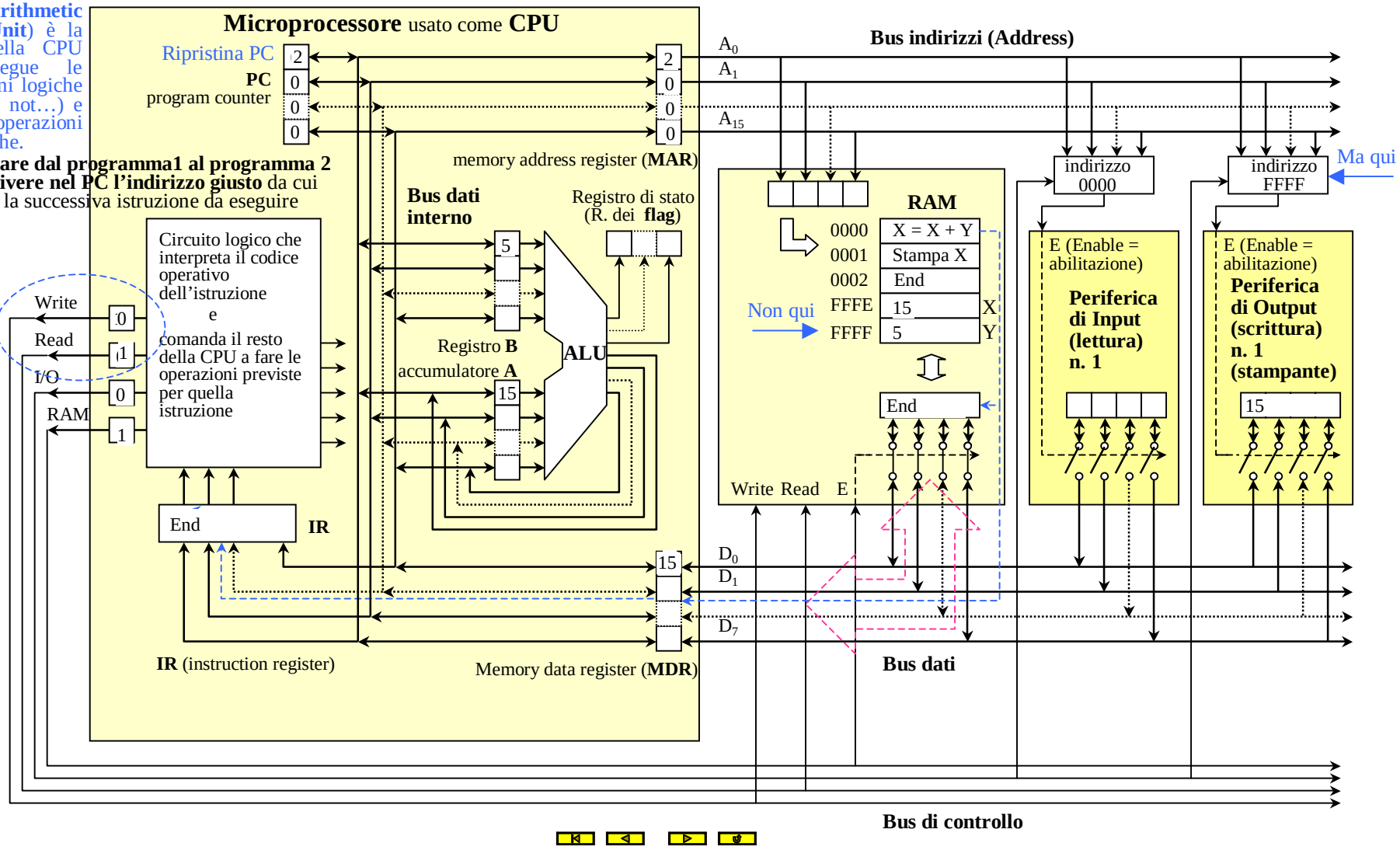
Per passare dal programma1 al programma 2 basta **scrivere nel PC l'indirizzo giusto** da cui prendere la successiva istruzione da eseguire

Il **bus indirizzi** è una serie di fili che portano l'indirizzo alla memoria e anche alle altre periferiche che possono essere di input (ovvero in grado di dare un numero alla CPU) o di autput (in grado di prendere un numero dalla CPU).

L'attivazione del piedino di **abilitazione** (**Enable**) fa collegare effettivamente un componente al resto del computer e in particolare al bus dati

Il **bus dati** è un insieme di fili che possono essere essere usati da tutte le periferiche per inviare un dato alla CPU o per riceverlo da essa, ma una per volta, per cui se usa il bus dati la RAM tutte le altre periferiche devono risultare sconnesse dal bus dati

La fase di **fetch** è quando la CPU preleva l'istruzione dalla RAM portandola nel registro istruzioni e la fase di **esecuzione** è quando l'istruzione presente in questo registro viene effettivamente eseguita



Struttura di un microntrollore

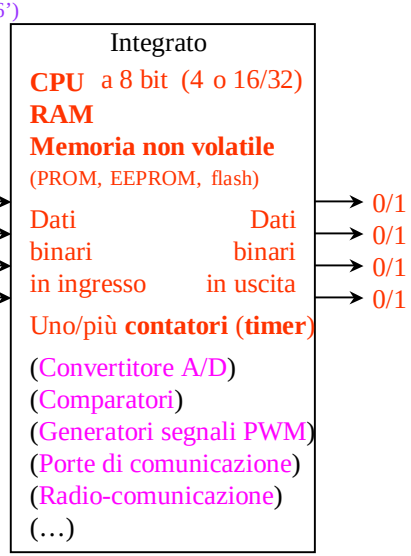
(6')

Un **microcontrollore** (µC) è *un computer contenuto in un integrato* ovvero un integrato che contiene al suo interno:

- una CPU
- una RAM
- una memoria non volatile (= che si mantiene anche senza alimentazione) (e che sta al posto dell'hard disk di un computer tradizionale)

Se essa è una PROM il microntrollore è programmabile una sola volta (OTP, one time programmable)

Come memoria non volatile di un microcontrollore si potrebbe usare la EEPROM (a volte si scrive E^2PROM), che potendo essere riscritta senza toglierla per metterla nel cancellatore diventa una RAM con conservazione dei dati allo spegnimento, ma essa è costosa e lenta (da 1 a 20 msec).



La soluzione si chiama FLASH, una EEPROM insieme più economica e più veloce ottenuta rinunciando alla cancellazione del singolo bit a favore di una cancellazione a blocchi.

- Linee di ingresso per far entrare i dati binari e linee di uscita per farli uscire.
- Serve poi almeno un contatore, usabile come tale o come timer (temporizzatore)

Un µC può contenere al suo interno molto altro ancora, a cominciare da un convertitore A/D (che permettondio di far entrare una V invece che un numero a 8/10/12 bit, fa risparmiare le preziose linee di ingresso) e dai gestori dei principali protocolli di comunicazione seriale o parallela su fili o anche via radio.
Ci possono essere comparatori in grado di rilevare quando una tensione raggiunge un certo valore ovvero quando si verifica un dato evento, generatori di segnali PWM (onde quadre con Duty Cycle proporzionale a qualcosa), contatori supplementari che fanno sempre comodo e altro ancora.

Generalmente il bus dati è a 8 bit (come i primi personal computer e come lo Z80) ma ci sono µC più semplici/lenti a 4 bit e µC più veloci a 16 o a 32 bit

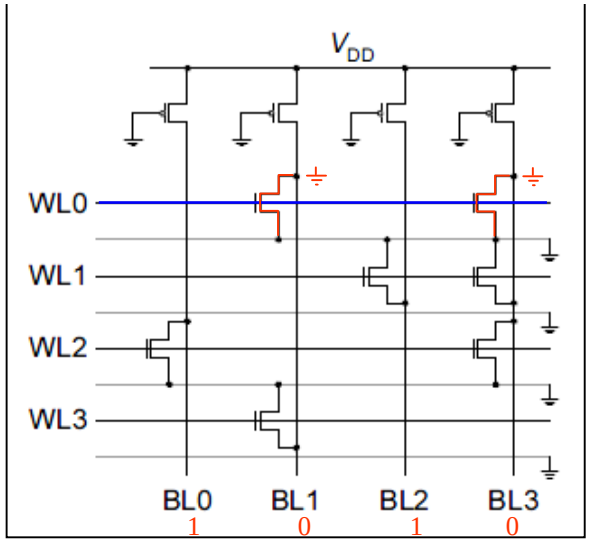
La diffusione dei µcontrollori ha spinto molti produttori a immettere sul mercato le loro famiglie.

Tra i più diffusi 8 bit ci sono:

- i **PIC** (dal PIC10 al PIC18) della Microchip Tecnology
- gli **AVR** (AVR8) della Atmel, che ha un completo sistema di sviluppo sotto Linux

Tra i 32 bit:

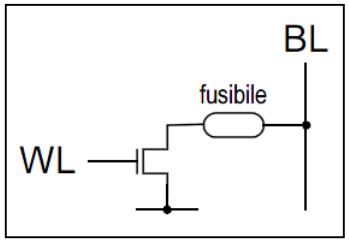
- il più diffuso è l'**ARM** (ARM7, 9, 11, Cortex) prodotto da molte case
- ma ci sono anche i PIC32 o gli AVR32



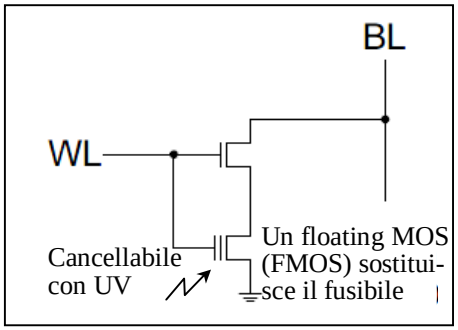
ROM (Read Only Memory)

L'indirizzo 0 si legge mettendo tensione su WL0. Tale tensione satura i due MOS che mettono 0 su BL1 e BL3 collegandoli a massa, mentre resta 1 su BL0 e BL2 che sono collegati al positivo da un resistore (MOS in alto) non percorso da corrente

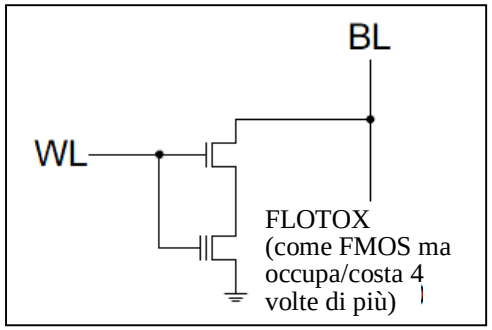
Tutti gli incroci hanno il MOS, ma bruciando il fusibile lo si scollega dove non lo si vuole



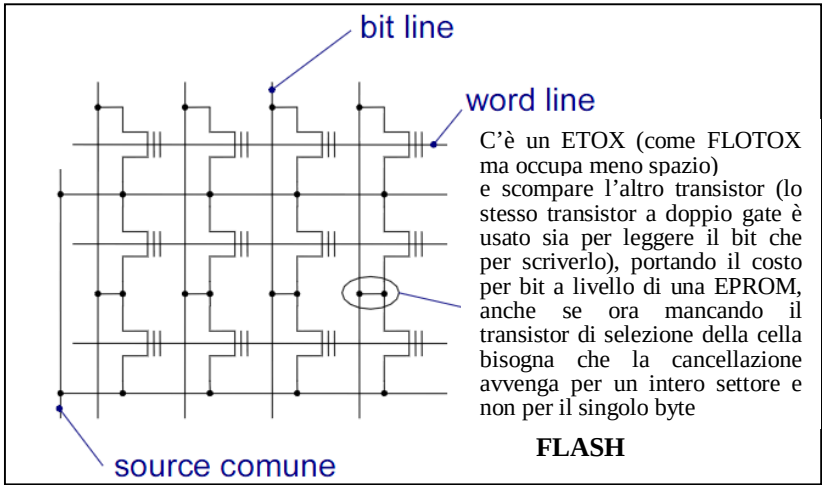
PROM (Programmable ROM)



EPROM (Erasable PROM)



EEPROM (Elettricaly EPROM)



C'è un ETOX (come FLOTOX ma occupa meno spazio) e scompare l'altro transistor (lo stesso transistor a doppio gate è usato sia per leggere il bit che per scriverlo), portando il costo per bit a livello di una EPROM, anche se ora mancando il transistor di selezione della cella bisogna che la cancellazione avvenga per un intero settore e non per il singolo byte

FLASH

PIC 16F84A: caratteristiche e pin

(13')

Il piedino di **reset** è chiamato master clear e attivo basso. Visto che il reset all'accensione è automatico, possiamo collegarlo a +Vdd e non usarlo.

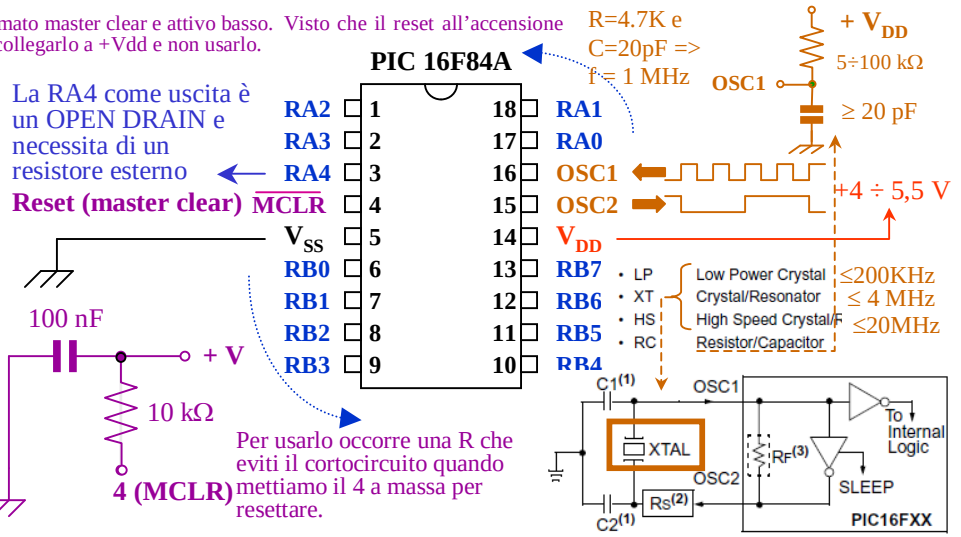
	16F84A (DIP a 18 pin)	**16F876A** (DIP a 28 pin)
Alimentazione	4 ÷ 5,5 V (2 ÷ 5,5 in vers. speciali) CMOS Idd<2mA (trascurabile in sleep)	2 ÷ 5,5 V a secondo della versione
Frequenza di lavoro	0 ÷ 20 MHz	
Parallelismo	8 bit	
Linee di I/O	13: A (5bit) B (8bit) fino a 25mA	A (6bit), B e C (8bit) – 25mA (16F877A in DIP 40 ha anche D ed E)
CPU	tipo RISC con solo 35 istruzioni di lunghezza 14 bit uguali per entrambi	
Memoria programma	1 K (1024) x 14 bit di FLASH	8 K (8 x 1024) di FLASH
RAM (volatile) +	68 Byte (68 registri da 8 bit) +	368 Byte (368 registri da 8 bit) +
EEPROM (permanente)	64 Byte (= 132 Byte)	256 Byte (= 624 Byte)
Contatori (timer)	1 Contatore a 8 bit	3 Contatori (uno è da 16 bit)
Interrupt	6 sorgenti di interrupt	14 sorgenti di interrupt
Stack	8 livelli (consentendo fino a 8 sottoprogrammi annidati)	

Convertitore A/D	0	A 10 bit da 1 dei 5 ingressi analogici (8 ingressi analogici nel 16F877A)
Comparatori analogici	0	2
Moduli CatturaConfrontaP(genera PWM)	0	2
USART (comunicazione seriale sincrona o asincrona)	0	1
MSSP (comunicazione seriale sincrona I²C o SPI)	0	1
PSP (comunicazione parallela)	0	0 (1 per il 16F877A)

La memoria per i dati è separata dalla memoria programma nell'**architettura Harward** usata in questo PIC (mentre quando la memoria è unica per dati e programmi è **architettura Von Neumann**)
La doppia memoria serve a velocizzare perché mentre il bus dati è impegnato coi dati che si elabora nell'istruzione N, il bus program è libero e può trasferire nel registro istruzioni l'istruzione N+1, che sarà elaborata appena finito di elaborare l'istruzione N essendo già stato fatto il FETCH dell'istruzione successiva

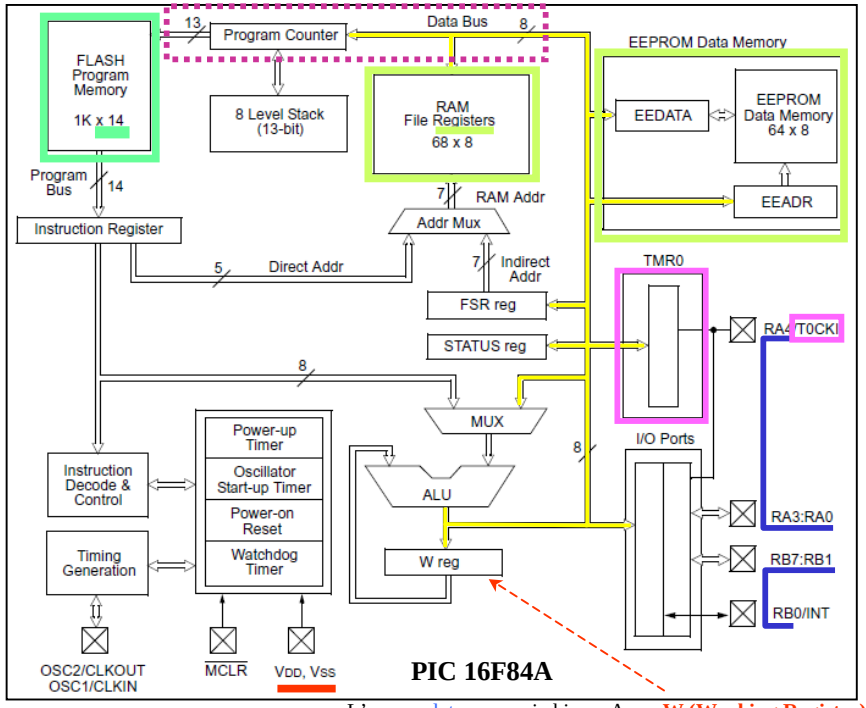
Un **interrupt** è una interruzione del programma principale per andare ad eseguire il **sottoprogramma** preparato dal programmatore per l'interruzione proveniente da quella sorgente e tornando al programma principale quando si è finito col sottoprogramma.

Quando si lascia un'esecuzione per riprenderla più tardi bisogna salvare i dati attuali in uno **stack** (**pila** in italiano, per ricordare che l'ultimo dato immesso sarà il primo ad essere ripreso) e visto che entrambi i microntrollori hanno uno stack con 8 posti per salvare i dati correnti essi consentono fino a un massimo di 8 sottoprogrammi annidati uno all'interno dell'altro



Mode	Freq	OSC1/C1	OSC2/C2
XT	455 kHz	47 - 100 pF	47 - 100 pF
	2.0 MHz	15 - 33 pF	15 - 33 pF
	4.0 MHz	15 - 33 pF	15 - 33 pF
HS	8.0 MHz	15 - 33 pF	15 - 33 pF
	10.0 MHz	15 - 33 pF	15 - 33 pF

Il program counter è di 13 bit ma gli arriva un bus dati di 8 bit, in grado di modificare solo gli 8 bit bassi (il PCL, Program Counter Low)



L'accumulatore non si chiama A ma **W (Working Register)**

I registri del PIC 16F84A

(11')

Programmare col C è molto più veloce e comodo ma per ottimizzare la velocità di esecuzione di un programma su una CPU bisogna scriverlo usando l'**assembly** di quella CPU (ovvero le istruzioni che quella CPU è in grado di eseguire scritte in formato simbolico, tanto a tradurle in zeri e uni producendo il programma in **linguaggio macchina** ci penserà l'**assemblatore**).

Per programmare nell'assembly di una CPU bisogna sapere che registri ha, per cui andiamo a vedere i registri del 16F84. Nel PIC sono chiamati registri anche le 68 locazioni di RAM e per distinguere le due categorie di registri, teniamo presente che i registri-registri sono chiamati **registri speciali**, mentre i registri-RAM sono chiamati **registri dati** (e nel loro insieme costituiscono il **file register**, abbreviato con F nelle istruzioni che vanno ad operare sulla RAM; Es.: MOVWF *12 = muovi in RAM, appunto in F, alla locazione di indirizzo 12 ciò che è in W, che è l'accumulatore*).

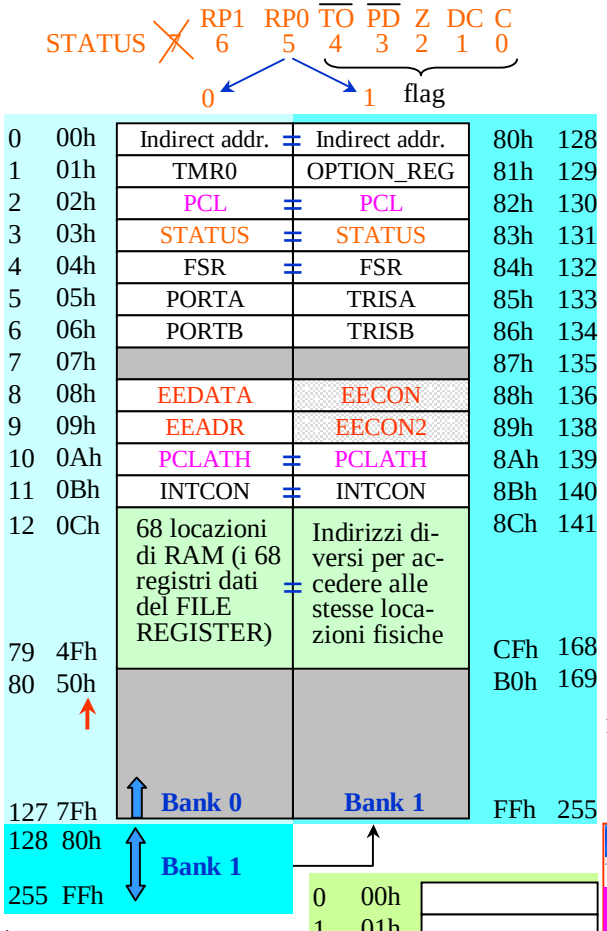
Prima di esaminare i 15 registri speciali disponibili vediamo come sono usate le 128 locazioni indirizzabili con 8 bit (da 0 a 127), che sono chiamate locazioni in **pagina 0**. Le successive 128 locazioni sono locazioni in **pagina 1**. Esse sono disegnate a fianco della pagina 0 perché come vedremo molte locazioni di pagina 1 sono copie delle corrispondenti locazioni in pagina 0.

Le prime 12 posizioni della pagina 0 (da 0 a 11) sono registri speciali (11 registri perché la 7^0 posizione non è usata).

Le successive 68 posizioni sono le **68** locazioni di RAM utilizzabili dall'utente. Le restanti locazioni della pagina 0 non sono utilizzabili e lette danno sempre 0.

Anche le prime 12 posizioni della pagina 1 (da 128 a 140) sono riservate a registri speciali (sono solo 9 posizioni utili perché la 7^0 non è usata, e l'8^0 e la 9^0 sono particolari). Di queste 9 posizioni, 4 sono usate per i registri speciali che mancano per il totale di 15, mentre 5 posizioni sono **duplicati** (ovvero *indirizzando 02h oppure 82h si scrive/legge sempre nella stessa locazione fisica*)

Anche le successive 68 posizioni sono duplicati in pagina 1 della RAM utente in pagina 0: due indirizzi diversi per accedere alla stessa locazione fisica. Gli indirizzi restanti di pag 1 sono inutilizzati.

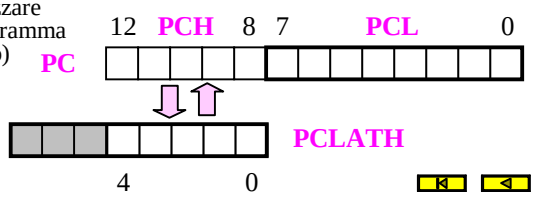


I **64** bytes dalla **EEPROM** hanno indirizzi da 00h a 3Fh. Il dato da scrivere/leggere va in **EEDATA**, mentre l'indirizzo va in **EEADR**. L'esecuzione di una operazione su EEPROM coinvolge due registri ausiliari, **EECON** ed **EECON2**, gestiti dal processore e non dal programmatore, che li può ignorare.

Per leggere/scrivere in pagina 0 occorre che il 5^0 bit del registro **STATUS** sia 0. Per operare in pagina 1, occorre che tale bit sia a 1 (la coppia che decide la pagina è RP0 e RP1, il 5^0 e il 6^0 bit, ma avendo solo 2 pagine basta agire su RP0).

I prime 5 bit del registro **STATUS** sono **flag**: il bit 0 (C) indica il carry/borrow (riporto/prestito); il bit 1 (DC) indica il carry del digit (del semibyte); il bit 2 (Z) va ad 1 se l'operazione ha dato come risultato 0; il bit 3 ($\overline{PD}$) è 0 su un power down, stato di sleep; il bit 4 ($\overline{TO}$) è 0 su time out del watchdog (TMR0 che conta all'indietro senza arrivare a 0 perché rimesso alto) contro i blocchi

Il **program counter** è a 13 bit per indirizzare 2^{13} = 8 k locazioni della memoria di programma (anche se il 16F84A ha solo 1 k di spazio) I primi 8 bit sono nel registro **PCL** (PC low). Gli altri 5 sono inaccessibili, ma vengono copiati dal PCH in **PCLATH** o viceversa nelle istruzioni di salto in cui si deve impostare il PC.



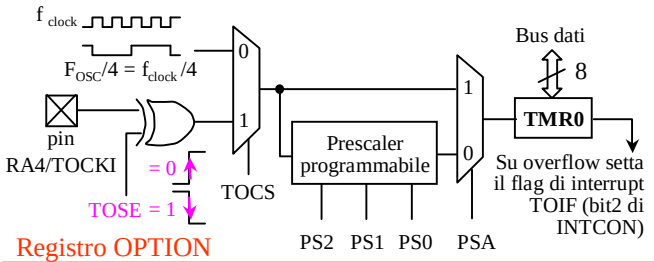
Il registro **INTCON** lo vedremo parlando di interrupt

Il registro **FSR** è il registro **indice** (quello che contiene l'indirizzo su cui operare) nel caso di un indirizzamento indiretto.
Non ci sono istruzioni a indirizzamento indiretto ed esso avviene quando s'indirizza il **registro fittizio 00h** (se, ad es., si scrive all'indirizzo 00h, si scrive in realtà all'indirizzo specificato in FSR).
La prima cosa da decidere nella programmazione è cosa usare come ingresso e cosa è usare come uscita delle RB scrivendo in **TRISB** (dopo aver messo a 1 il bit RP0 di Status) 1 per settare Input e 0 per settare Output.
Per settare le RA invece si scrive su **TRISA**

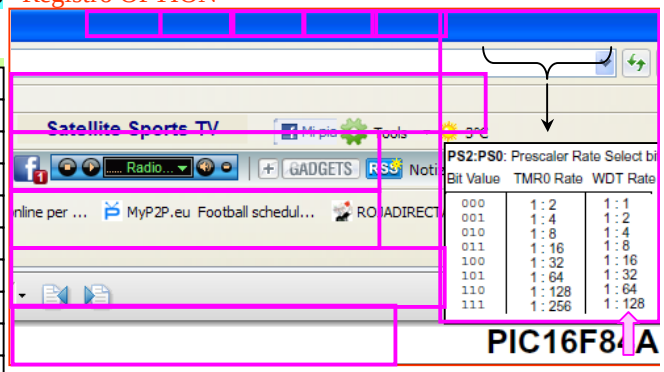
PORTA contiene i dati delle RA, messi dal PIC se sono uscite o dall'esterno se sono ingressi.
PORTB contiene i dati delle RB e può essere letto dal PIC (lettura all'indirizzo 6) o scritto dal PIC (scrittura all'indir. 6)

Il registro **TMR0** contiene il risultato del conteggio di un contatore che, partendo dal valore impostato, conta i fronti del clock interno/4 o i fronti un segnale esterno applicato su RA4

Il registro **OPTION** contiene opzioni che riguardano il contatore



Registro OPTION



Etichette, direttive, sottoprogrammi di ritardo e astabile

(20')

Questo è il primo programma che studieremo scritto in **assembly** PIC16 (cioè usando le istruzioni riconosciute dal PIC16 scritte in formato simbolico).

E questa è la sua traduzione in **linguaggio macchina** fatta dall'**assemblatore**

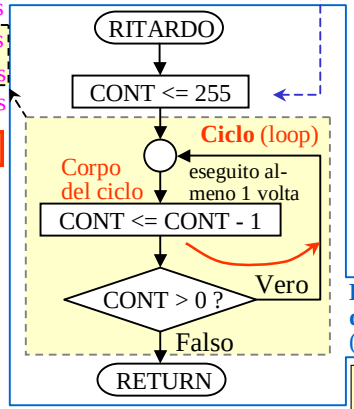
Nessuna di queste 3 linee ha prodotto un'istruzione eseguibile.

```
; --- ritardo 765us ---
CONT EQU 0C
; --- sub RITARDO ---
RITARDO MOVLW D'255'
        MOVWF CONT
CICLO DECFSZ CONT,1
      GOTO CICLO
      RETURN
      END
```

Commento che dice cosa fa il programma: produce un ritardo di 765 microsecondi.
Direttiva al preprocessore che gli comanda di sostituire CONT con 0xC ogni volta che lo incontra
Commento che qualifica il programma come sub-routine richiamabile con CALL RITARDO
Istruzione muovi la costante (literal) decimale 255 in W (nel registro di working = nell'accumulatore)
Istruzione muovi il contenuto di W nel file register all'indirizzo CONT (0Ch) "Metti 255 all'indirizzo 0C"
Decrementa (cala di 1) il file register di ind. CONT e skip (salta) l'istr. successiva se il risultato del dec è zero
Go to (vai a) eseguire l'istruzione con etichetta CICLO
Ritorna al programma chiamante eseguendo l'istruzione seguente alla CALL RITARDO che ha chiamato il sottoprogramma

Address	Opcode		$t_{esecuzione}$
000	30FF	MOVLW 0xff	1 Ciclo (4 Ck) 1 µs
001	008C	MOVWF 0xc	1 Ciclo 1 µs
002	0B8C	DECFSZ 0xc, F	1 Ciclo 1 µs / 2 Cicli 2 µs
003	2802	GOTO 0x2	2 Cicli 2 µs
004	0008	RETURN	

$t_{sotto} = t_{loop} + 4\mu s = 769\ \mu s$ $t_{loop} = 255 \times 3\ \mu s = 765\ \mu s$



Una **direttiva** è una pseudoistruzione, ovvero una linea di programma che non darà origine a nessuna istruzione macchina. Una direttiva **EQU** comanda all'assemblatore di sostituire una data etichetta con un dato numero ogni volta che incontra quell'etichetta (ad es. PORTA EQU 05 comanda di sostituire l'esadecimale 05 all'etichetta PORTA ogni volta che la incontra e serve a sollevare il programmatore dal ricordarsi quel numero, che è l'indirizzo del registro PORTA). Una direttiva **ORG** comanda all'assemblatore di caricare il programma che segue a partire dalla locazione indicata dalla ORG (es. ORG 0C comanda di caricare il programma a partire dalla locazione 0C). Una direttiva **END** comanda all'assemblatore di ignorare il testo che segue (in pratica dice dove finisce il programma) e se per errore ci sono istruzioni messe dopo la END, esse saranno ignorate)

Un'**etichetta** (label) è un nome scelto a piacere dal programmatore e scritto a partire dalla prima colonna. Le etichette RITARDO e CICLO individuano una linea di programma permettendo di sostituire un'istruzione del tipo "vai ad eseguire l'istruzione che si trova nella locazione 002 di memoria" con un'istruzione del tipo "vai ad eseguire l'istruzione con l'etichetta CICLO" (che lascia all'assemblatore il compito di calcolare in quale locazione si trova quell'istruzione e di sostituire il salto a CICLO col salto alla locazione 002)

Col clock a 4 Mhz il **ritardo minimo** è di 1 µsec e anche portando il clock al massimo, ovvero a 20 Mhz, non si può scendere sotto a 1/5 di µsec (**200 nsec**) visto che per eseguire un'istruzione che cambia lo stato di un'uscita ci vogliono 4 clock ovvero 200 nsec.
Con un PIC si possono sostituire molti altri integrati, ma la sua velocità massima è limitata.
Non ci sono problemi invece ad **aumentare il ritardo** oltre ai 765 µsec (1 msec scarso) consentiti dal decremento di un solo contatore, usando un secondo contatore da azzerare prima di passare al prossimo decremento del primo contatore

Quanti cicli servono per un ritardo di 1 sec? 1sec / 3µsec = 333 333

Quanti contatori servono per fare 333 333 cicli prima di finire il loop?
Essendo 333 333 > 255 un CONT1 non basta e metto un CONT2. Essendo 333 333 / 255 = 1307,2 > 255 il CONT2 non basta e metto un CONT3. Essendo 1307 / 255 < 5,25 bastano 3 CONT

```
; --- ritardo 1 s ---
cont1 equ 0C
cont2 equ 0D
cont3 equ 0E
; sub ritardo 1s ---
rit_1s MOVLW D'5'
       MOVWF cont3
ciclo3 MOVLW D'255'
       MOVWF cont2
ciclo2 MOVLW D'255'
       MOVWF cont1
ciclo1 DECFSZ cont1,1
       GOTO ciclo1
       DECFSZ cont2,1
       GOTO ciclo2
       DECFSZ cont3,1
       GOTO ciclo3
       return
```

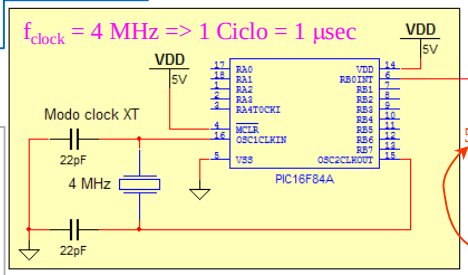
Ciò che è scritto dopo un ";" è un **commento** ed è importante per rendere più leggibile il programma ma viene ignorato dall'assemblatore.

Convenzioni sui dati numerici:

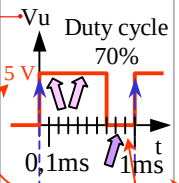
Un numero senza indicazioni è considerato esadecimale: es. MOVLW 10 mette in W il byte 0010000 = 16d)
Per farlo considerare decimale occorre una D e gli apici: es. MOVLW D'16' mette in W il byte 0010000=16d
Per farlo considerare binario occorre una B e gli apici: es. MOVLW B'0010000' mette in W il byte 0010000
MOVLW FF non trasferisce FFh in W perché il dato, non cominciando con un numero, è preso per etichetta
MOVLW 0xFF mette in W il byte 11111111 perché 0x specifica che FF è un esadecimale (FFh)
MOVLW 0FF mette in W il byte 11111111 perché anche solo 0 specifica che FF è un esadecimale (FFh)
MOVLW H'FF' mette in W il byte 11111111 perché anche la H col dato tra apici lo qualifica esadecimale

Diagramma di flusso (flow-chart)

Come si vede dal diagramma di flusso, il sottoprogramma RITARDO non compie azioni visibili, limitandosi a far passare il tempo necessario a decrementare CONT 255 volte, dopo di che termina.



Iumax = 25mA



astabile (generatore di clock) col PIC (invece che col 555)

Per avere quest'uscita bisognerà 1) mettere alta una linea di uscita e poi chiamare una routine di ritardo che faccia passare 0,7 msec; poi 2) metterla bassa e chiamare una seconda routine di ritardo che faccia passare 0,3 msec.

Con che numero va caricato il contatore CONT per impiegare 0,7 msec (700 µsec) per azzerarlo? Visto che ogni giro impiega 3 µsec, servono 700/3 = 233,3 giri, per cui N = 233.
E per impiegare 0,3 msec (300 µsec)? 300/3 = 100 giri

SECONDA SOLUZIONE: potremmo anche usare una sola routine di ritardo che faccia passare 0,1 msec (100 µsec)
e mettere la sua chiamata dentro ad un ciclo1 che si ripete 7 volte per produrre uno stato alto di 0,7 msec
e mettere la sua chiamata dentro ad un ciclo2 che si ripete 3 volte per produrre uno stato basso di 0,3 msec.

```
;- astabile 1kHz 70% DC -
;---- dichiarazioni -----
PORTA equ 05
PORTB equ 06
CONT  equ 0C
N_RIP equ 0D
;--- inizializzazione ---
      MOVLW B'00000'
      TRIS PORTA
      MOVLW B'00000000'
      TRIS PORTBTRISB<=W
      BCF  PORTB,0
;-------- main ----------
START BSF PORTB,0
      MOVLW D'7'
      MOVWF N_RIP
ciclo1 CALL RIT_01
       DECFSZ N_RIP
       GOTO ciclo1
      BCF PORTB,0
      MOVLW D'3'
      MOVWF N_RIP
ciclo2 CALL RIT_01
       DECFSZ N_RIP
       GOTO ciclo2
       GOTO START
;- sub RITARDO 0,1msec -
RIT_01 MOVLW D'33'
       MOVWF CONT
ciclo3 DECFSZ CONT
       GOTO ciclo3
       RETURN
```

Arduino UNO R3

10’

Sistema di sviluppo = HW da usare per costruire il circuito che sta intorno al µControllore + SW che faciliti la programmazione del µC

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

I2C Master (SDA, SCL) — I2C Slave (SDA, SCL)

SCL SDA

USB (Vcc = 5V)

µC ATmega16U2 **convertore seriale/USB**(in contenitore QSP a montaggio superficiale)

ICSP = In-Circuit Serial Programming

Quarzo per il clock a **16 MHz**

Connettori strip line femmina

Jack di potenza =>

=> 7÷12V

≥ 7V

≥ 3.5V

0 V

5V

+5V

Usate dalle estensioni HW (**shield**) per adattare la V a quella di Arduino

7÷12V

NC

SPI Master: SCLK, MOSI, MISO, SS — SPI Slave: SCLK, MOSI, MISO, SS

Comunicazione seriale SPI, molto più veloce della I2C, non limitata a parole di 8 bit e con meno consumi

SCK
MISO
MOSI
SS for Ethernet controller
SS for SD card

UNO Rev3

5V - 500 mA

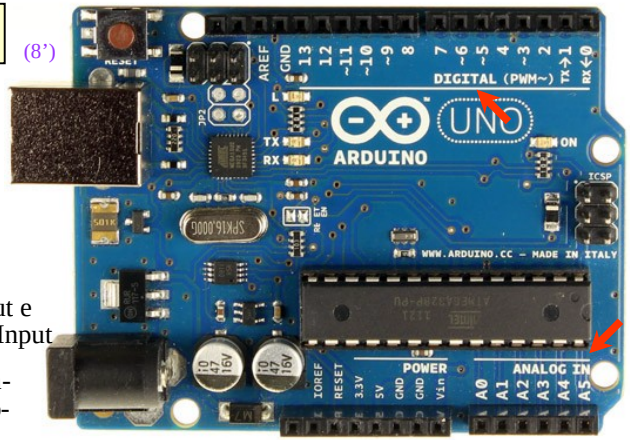
ATmega16U2 (nella precedente versione della UNO era un 8U2; nella 2009 era un FTDI232)

ATmega328P

AVR della ATMEL a 8 bit (thiny fino a 32 pin e mega da 28 a 100 pin) a 16 bit (Xmega) e a 32 bit

Ingressi e uscite

(8')



Arduino uno ha 14 pin di I/O digitali (pin 0 ÷ 13 sulla strip DIGITAL) e 6 ingressi analogici (pin A0 ÷ A5 sulla strip ANALOG)

I/O DIGITALI

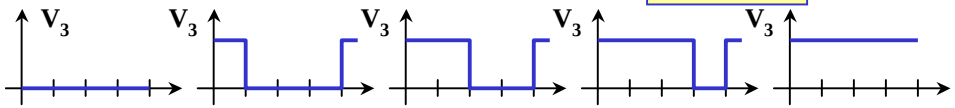
pinMode(13,OUTPUT) imposta il pin 13 come Output e **pinMode(13,INPUT)** come Input

Se il cui numero è nella variabile intera **pin** è impostato come Input si può andare a **leggere** su di esso, mettendo il suo valore H/L (1/0) nella variabile intera **val** con **val = digitalRead(pin)**

Se invece **pin** è impostato come Output si può **scrivere** su di esso alto con **digitalWrite(pin,HIGH)** O ci si può scrivere basso con **digitalWrite(pin,LOW)**

La tensione HIGH è di **5 V** e quella LOW è di **0 V**.
La corrente max, erogata se H e assorbita se L, è **40 mA**

Se però il pin di Output è uno di quelli col trattino (3, 5, 6, 9, 10, 11) in grado di fornire **uscite PWM** , ovvero onde quadre a duty cycle variabile, allora non bisogna specificare H o L ma un valore compreso tra **0** (DC = 0 ovvero sempre L) e **255** (DC = 100% ovvero sempre H)
Trattandosi di un'uscita analogica anche se di tipo quadro, si ricordera che l'istruzione da usare non è digitalWrite(…) ma **analogWrite(pin, valore)**



analogWrite(3, 0) **analogWrite(3, 63)** **analogWrite(3, 127)** **analogWrite(3, 191)** **analogWrite(3, 255)**

DC = 0 / 255 = 0% 63 / 255 = 24.7% 127 / 255 = 49,8% 191 / 255 = 74,9% 255 / 255 = 100%

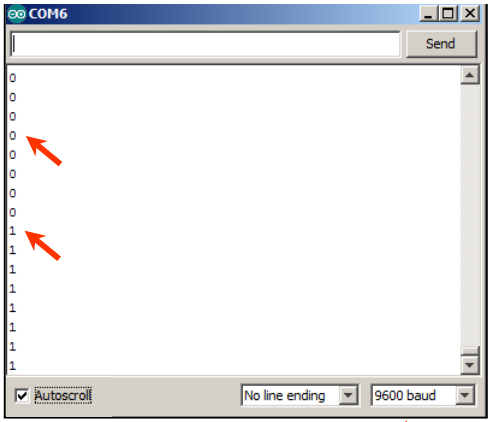
I/O ANALOGICI

Su ognuno dei 6 ingressi analogici (A0 ÷ A5) si può mettere una V compresa tra 0 e 5 V, che verrà convertita da un A/D con risoluzione da 10 bit in un numero intero compreso tra 0 e 1023

Se **pin** è un intero compreso tra 0 e 5 l'istruzione **val = analogRead(pin)** mette nella variabile intera **val** un numero da 0 a 1023 (se V=2,5V sarà 1023/2 = 511,5 = 511; se V=1V sarà 1023/5 = 204,6 = 205)

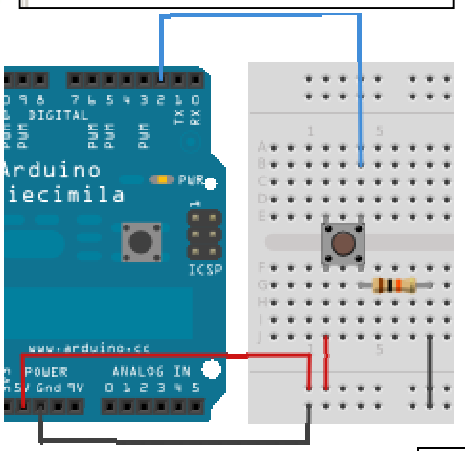
I pin 2 e 3 dell'ATmega328 sono inviate ai pin 0 (RX) e 1 (TX) ma anche all'USB via ATmega 16

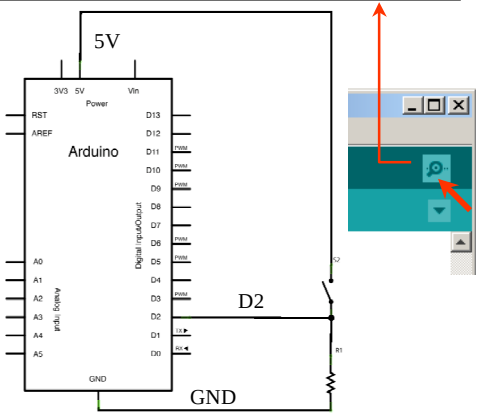
```
DigitalReadSerial §

void setup() {
  Serial.begin(9600);
  pinMode(2, INPUT);
}

void loop() {
  int sensorValue = digitalRead(2);
  Serial.println(sensorValue);
}
```

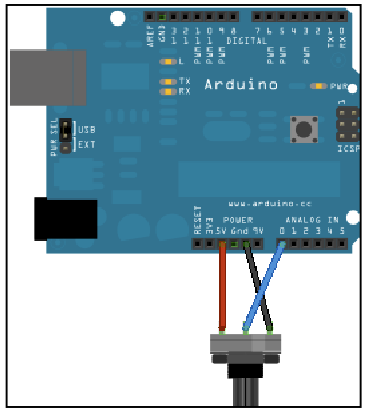


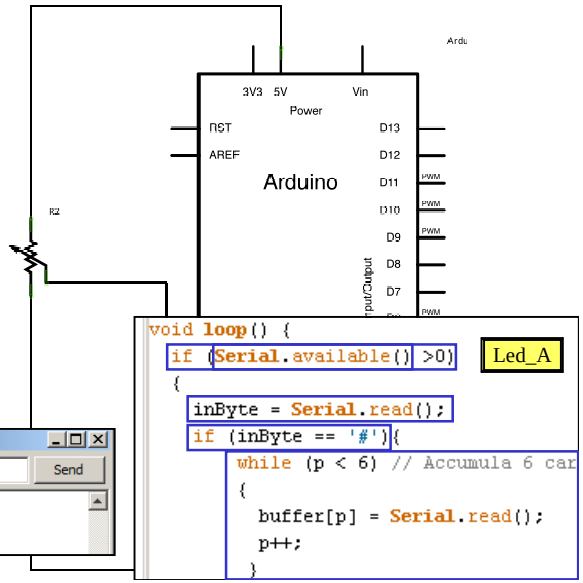

```
Blink | Arduino 1.0
File Edit Sketch Tools Help
Blink §

void setup() {
  pinMode(13, OUTPUT);
}

void loop() {
  digitalWrite(13, HIGH);
  delay(1000);
  digitalWrite(13, LOW);
  delay(1000);
}
```

f = 490 Hz










```
AnalogReadSerial §

void setup() {
  Serial.begin(9600);
}

void loop() {
  int sensorValue = analogRead(A0);
  Serial.println(sensorValue);
}
```


```
void loop() {
  if (Serial.available() >0)
  {
    inByte = Serial.read();
    if (inByte == '#'){
      while (p < 6) // Accumula 6 car
      {
        buffer[p] = Serial.read();
        p++;
      }
```

Led_A

Buzzer + fotoresistenza (1)

(9')

Vogliamo produrre una nota musicale, la cui frequenza dipenda dalla luce che colpisce una fotoresistenza, usando la funzione **tone()**

Buzzer_Fotoresistenza §

```
int Buzzer=9;    // Il + del buzzer magnetico (50mA a 5V) è collegato a D9
int Fotores=0;   // L'uscita del partitore con la fotoResistenza va ad A0
int Nota;        // Variabile che conterrà la f in Hz della funzione nota()

void setup() {
  pinMode (Buzzer, OUTPUT);          // Setta Buzzer come uscita
}

void loop() {                        // Da qui tutto si ripete all'infinito
  Nota=3*(analogRead(Fotores))+500;  // f=500Hz se V=0 e 3500Hz se V=1000
  tone(Buzzer,Nota);                 // Vquadra 5V 50% a f=Nota su Buzzer
  delay(10);                         // Aspetta 10ms per la conversione A/D
}
```

=> Variabili e scopi

Il tempo di conversione dell'A/D è di 100 µs, ma in quest'applicazione servono 5-10ms di ritardo

Buzzer_Fotoresistenza_3_ino

1) Una nota fissa viene facilmente a noia, per cui vogliamo modificare il programma in modo che il suono cominci solo oscurando la fotoresistenza.

2) Inoltre invece di una sola nota vogliamo emettere 7 note di seguito aumentando la frequenza ogni volta di 40 Hz

```
void loop() {
  Nota=2*(analogRead(Fotores))+500; // f=500Hz se V=0 e 2500Hz se V=1000
  if (Nota < 1590) {
    // Non suona se la luce è alta (Vpartitore bassa < 45/1023)*5= 219mV)
  }
  else {
    for (int i=0; i<7; i++) {        // Ciclo di 7 note (da i=0 fino a i=6
      tone(Buzzer, (Nota+(40*i)));   // f = Nota, Nota+40, Nota+80, ...
      delay(70);                     // Aspetta 70ms prima della nuova n
    }
    noTone(Buzzer);                  // Stop nota
  }
}
```

=> I cicli

tone()

Description

Generates a square wave of the specified frequency (and 50% duty cycle) on a pin. A duration can be specified, otherwise the wave continues until a call to noTone(). The pin can be connected to a piezo buzzer or other speaker to play tones.

Only one tone can be generated at a time. If a tone is already playing on a different pin, the call to tone() will have no effect. If the tone is playing on the same pin, the call will set its frequency.

Use of the tone() function will interfere with PWM output on pins 3 and 11 (on boards other than the Mega).

NOTE: if you want to play different pitches on multiple pins, you need to call noTone() on one pin before calling tone() on the next pin.

Syntax

tone(pin, frequency)
tone(pin, frequency, duration)

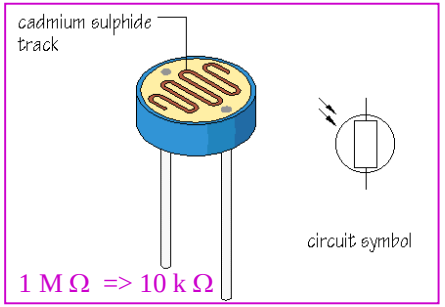
=> Le funzioni

Parameters

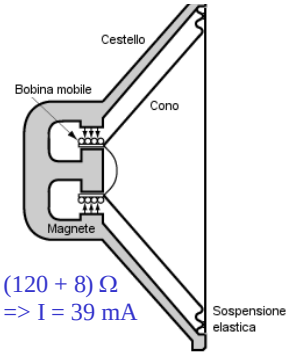
pin: the pin on which to generate the tone

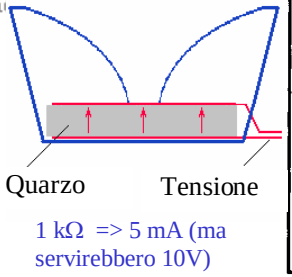
frequency: the frequency of the tone in hertz - *unsigned int* (fino a oltre 65 kHz)

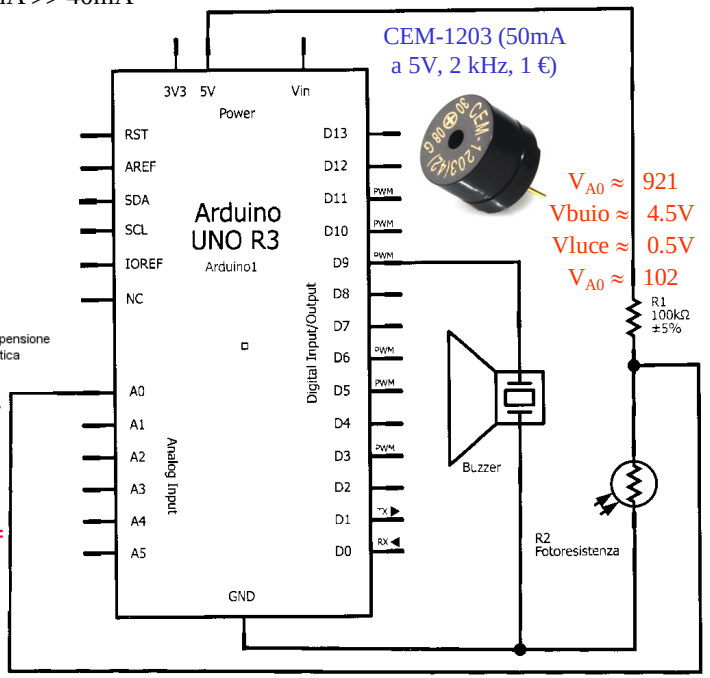
duration: the duration of the tone in milliseconds (optional) - *unsigned long* (fino a 1.193 ore)



$$I_{max} = \frac{V_{max}}{Z} = \frac{5}{8} = 625mA >> 40mA$$



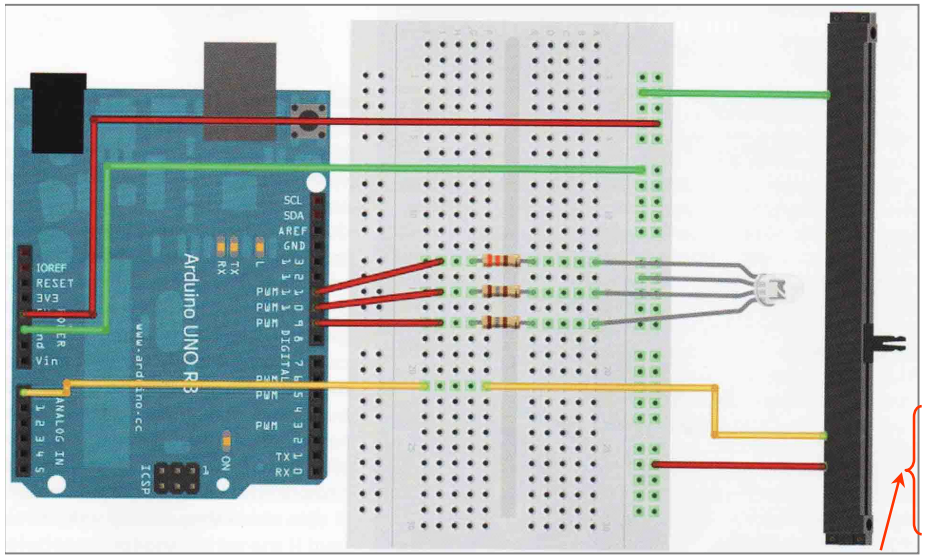




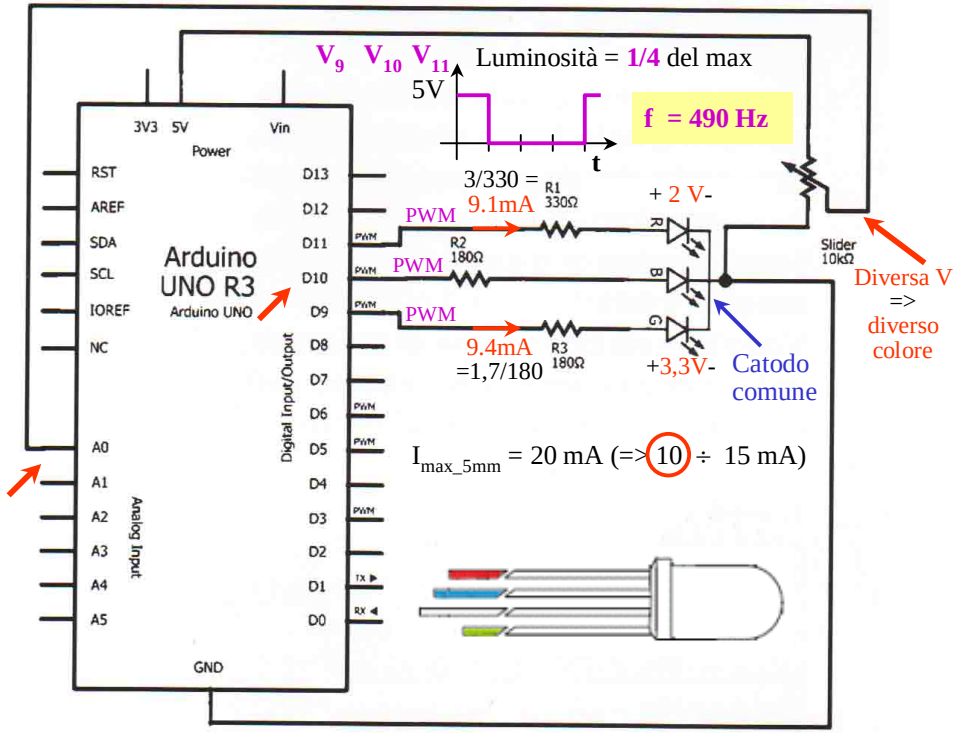
(1) Dal libro "*Primi passi con Arduino*" a cura di Simone Majocchi, speciale di Elettronica In

LED RGB con cursore (slider) (1)

(9')



Varia l'intensità luminosa delle luce bianca (rosso=blu=verde) <= **Dimmer** (VariaLuce)




```
int redpin = 11;  // abbiniamo i pin ai colori
int greenpin = 9; // da verificare con le specifiche
int blupin = 10;  // del led RGB
int Slider=0;     // il pin analogico per il potenziometro
int Lettura=0;    // variabile per il valore del potenziometro
int Gruppo;       // da 0 a 3 per individuare la routine
int Lumi;         // il valore da usare nella coppia di colori

void setup() {
 pinMode(redpin,   OUTPUT);   // I tre pin in OUTPUT
 pinMode(greenpin, OUTPUT);
 pinMode(blupin,   OUTPUT);
 analogReference(DEFAULT);    // imposta a 5V la soglia max A/D
}                             (INTERNAL) => Vmax A/D = 1,1 V
                              (EXTERNAL) => Vmax A/D = V aref
void loop() {
 Lettura = analogRead(Slider); // acquisiamo il valore del potenziometr
 Gruppo = (Lettura & 768)/256;  // estraiamo i primi 2 bit di sinistra
 Lumi = (Lettura & 255);        // mascheriamo gli 8 bit di destra
 switch (Gruppo) {    => switch in Panoramica istruzioni
  case 0:    // siamo nell'intervallo da 0 a 255 - bianco
   analogWrite(greenpin,Lumi);
   analogWrite(redpin,Lumi);
   analogWrite(blupin,Lumi);
   break;     => break in Cicli
  case 1:    // primo blocco colore da verde a blu
   analogWrite(greenpin,255-Lumi);
   analogWrite(blupin,Lumi);
   analogWrite(redpin,0);
   break;
  case 2:    // secondo blocco colore da blu a rosso
   analogWrite(blupin,255-Lumi);
   analogWrite(redpin,Lumi);
   analogWrite(greenpin,0);
   break;
  case 3:    // terzo blocco colore da rosso a verde
   analogWrite(redpin,255-Lumi);
   analogWrite(greenpin,Lumi);
   analogWrite(blupin,0);
   break;
 }
delay (10);
}
```

768 = 1100000000
Lettura = 0100111101
Lettura & 768 = 0100000000
/2^8 = 256 = spostare a destra di 8 posizioni

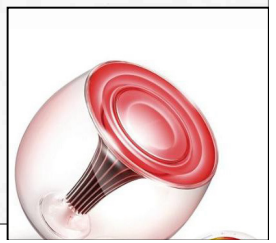
& = AND **bit a bit** (=> Elementi lessicali del C++)

& = AND **bit a bit**

255 = 0011111111
Lettura = 0100111101
Lettura & 768 = 0000111101

00+lumi
01+lumi
10+lumi
11+lumi

3
2
1
0

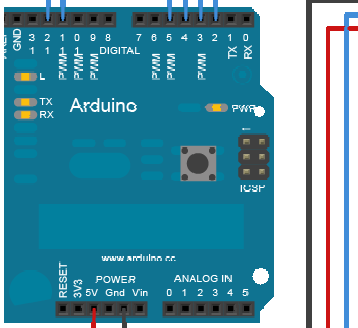


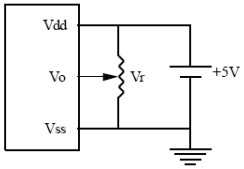
(1) Dal libro "*Primi passi con Arduino*" a cura di Simone Majocchi, speciale di Elettronica In

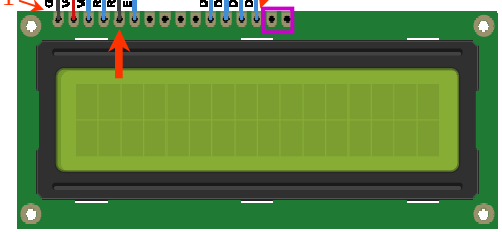
Comando LCD 16x2

(9')

Liquid **C**rystal **D**isplay





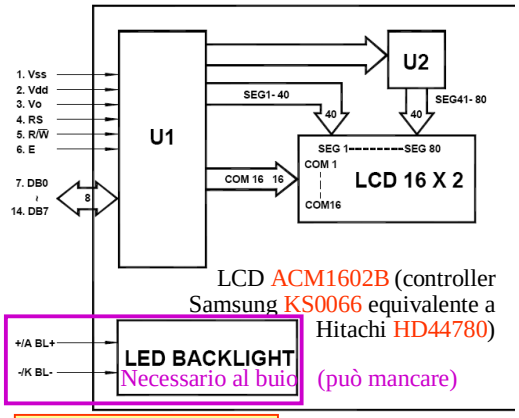


This library allows an Arduino board to control LiquidCrystal displays (LCDs) based on the Hitachi HD44780 (or a compatible) chipset, which is found on most text-based LCDs. The library works with in either 4- or 8-bit mode (i.e. using 4 or 8 data lines in addition to the rs, enable, and, optionally, the rw control lines).

Function

- LiquidCrystal()
- begin()
- clear()
- home()
- setCursor()
- write()
- print()
- cursor()
- noCursor()
- blink()
- noBlink()
- display()
- noDisplay()
- scrollDisplayLeft()
- scrollDisplayRight()
- autoscroll()
- noAutoscroll()
- leftToRight()
- rightToLeft()
- createChar()

Pin No.	Symbol	Function
1	Vss	Ground
2	Vdd	+5V
3	Vo	LCD contrast adjust
4	RS	Register select
5	R/W	Read / write
6	E	Enable
7	DB0	Data bit 0
8	DB1	Data bit 1
9	DB2	Data bit 2
10	DB3	Data bit 3
11	DB4	Data bit 4
12	DB5	Data bit 5
13	DB6	Data bit 6
14	DB7	Data bit 7
-	BL-	Power Supply for BL-
+	BL+	Power Supply for BL+



LCD ACM1602B (controller Samsung KS0066 equivalente a Hitachi HD44780)

(può mancare)

(Funzioni <=) Prototipo (dichiarazione) della funzione lcd(), una variabile di tipo LiquidCrystal ma avendo le () è una funzione e tra parentesi ci sono le uscite di Arduino usate per comandarla

LiquidCrystal(rs, enable, d4, d5, d6, d7) nome di funz.

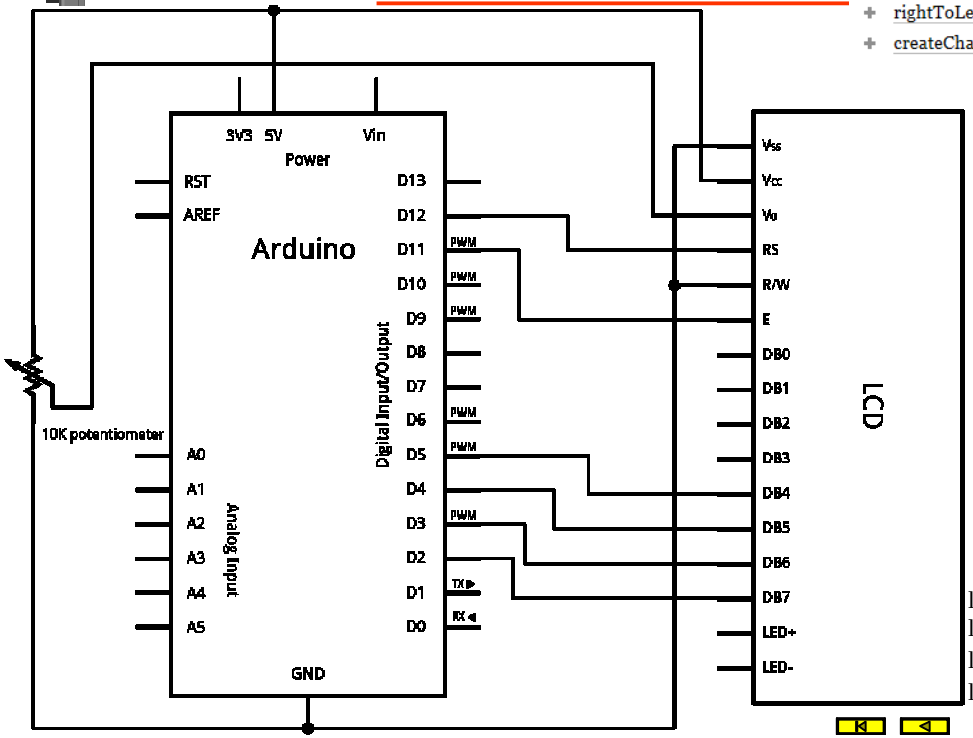
LiquidCrystal(rs, rw, enable, d4, d5, d6, d7) sovraccarico

LiquidCrystal(rs, enable, d0, d1, d2, d3, d4, d5, d6, d7)

LiquidCrystal(rs, rw, enable, d0, d1, d2, d3, d4, d5, d6, d7)

=> Direttiva include

```
#include <LiquidCrystal.h>
// Direttiva che include il programma LiquidCrystal.h
LiquidCrystal lcd(12,11,5,4,3,2);//Prototipo (dichiaraz.) di lcd()
void setup() {
lcd.begin(16, 2); // Dice che lcd ha 16 caratteri su 2 righe
lcd.print("Ciao, mondo!"); // Stampa una volta Ciao, mondo!
}

void loop() {
lcd.setCursor(0, 1); // Posizionati a colonna 0 riga 1 (la II riga)
lcd.print(millis()/1000); // Stampa il tempo dall'ultimo reset
}
```



lcd.clear
Cancella il display e torna a colonna 0 riga 0

lcd.home
Torna a colonna 0 riga 0 (senza cancellare)

lcd.write (dato)
Scrive un carattere

lcd.print (dato)
Scrive un carattere, una stringa o un numero)

lcd.print (dato, DEC)
lcd.print (dato, BIN)
lcd.print (dato, HEX)
lcd.print (dato, OCT)

Scritta "Ciao, mondo!" scorrevole

```
#include <LiquidCrystal.h>
String textMsg="hello, world!";
int cols = 16; int rows = 2;
int strLength = textMsg.length();
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
lcd.begin(cols, rows); // Print a message to the LCD.
lcd.print(textMsg);
delay(1000);}

void loop() {
  for (int pCont = 0; pCont < strLength; pCont++) {
  lcd.scrollDisplayLeft();     // wait a bit:
  delay(150);
  }
  delay(1000);}
```

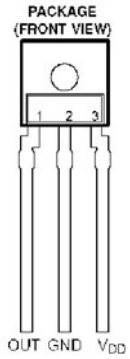
On-off con telecomando TV a IR

Sensore IR (infrarossi) con **preamplificatore, demodulatore** e **filtro** tarato sulla frequenza di 38 kHz – 600 µs con **amplificatore/squadratore** incorporato

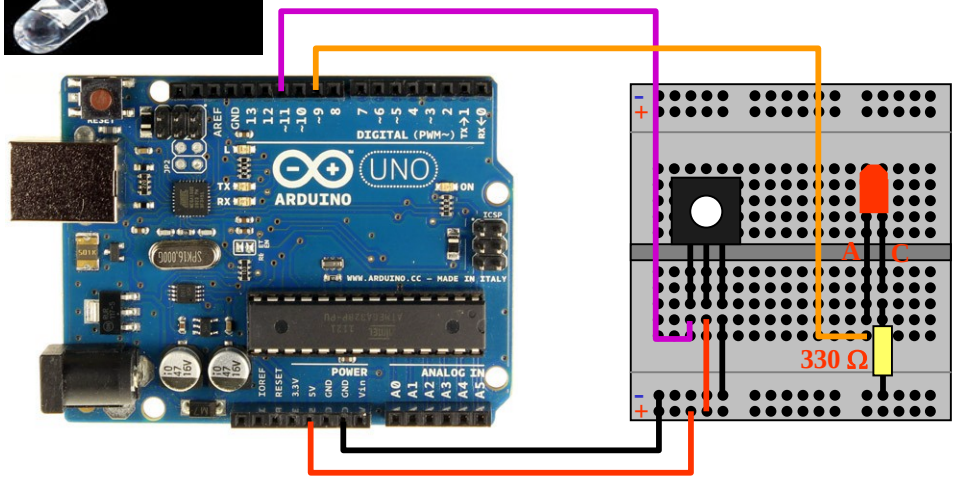
- Uscita: attiva a livello basso (0,2 V)
- Alimentazione: 4,5 V ~ 5,5 V
- Consumo: 0,6 mA
- Lunghezza d'onda: 940 nm
- BPF center frequency: 38 kHz
- Temperatura di lavoro: -10°C ~ +60°C
- signal range: max. 10 m
- Dimensioni: 7,3 x 7,6 x 5,2 mm
- Peso: 0,68 g

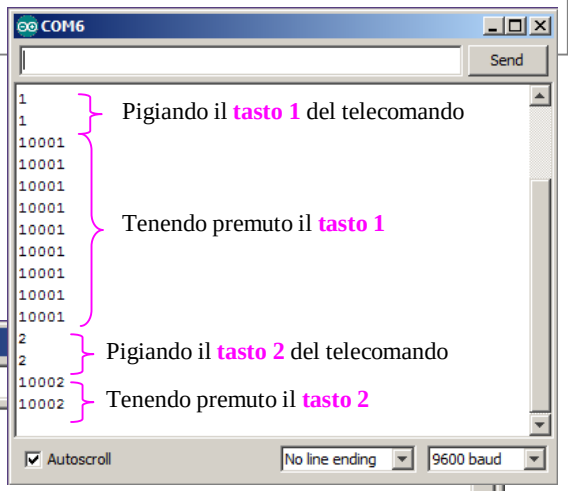
V_{DD} = + 5 Vdc
GND = massa
OUT = uscita segnale TTL

(4')

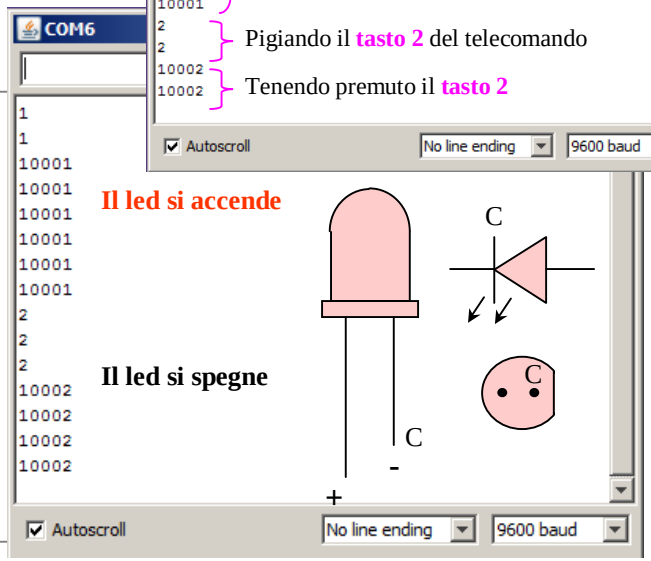



```
#include <IRremote.h>    // Include la libreria IRremote
// Per Arduino 1.0 in IRRemoteInt.h bisogna sostituire
// #include<WProgram.h> con #include<Arduino.h>
int receiver = 11;       // Pin 1 del ricevitore IR a D11
IRrecv irrecv(receiver);// Crea una istanza di 'irrecv'
decode_results results;

void setup() {
  Serial.begin(9600);  // Attiva la seriale a 9600 baud
  irrecv.enableIRIn(); // Attiva ricezione dati dal sensore IR
  pinMode(9, OUTPUT);  // Setta D9 come output
}

void loop() {
  if (irrecv.decode(&results))    // Se hai ricevuto un segnale IR...
    { Serial.println(results.value, HEX); // Mostralo in esadecimale
      irrecv.resume();            // Ricevi il prossimo valore
    }
}
```






```
void loop() {
  if (irrecv.decode(&results))          // Se hai ricevuto un segnale IR...
  { Serial.println(results.value, HEX); // Mostralo in esadecimale
    irrecv.resume();                    // Ricevi il prossimo valore
  }
  if ( results.value == 0x1 || results.value == 0x10001 ){ // Se tasto 1 telecomando
  digitalWrite(9, HIGH);                                   // accendi il LED
  }
  if ( results.value == 0x2 || results.value == 0x10002 ){ // Se tasto 2 telecomando
  digitalWrite(9, LOW);                                    // spegni il LED
  }
}
```

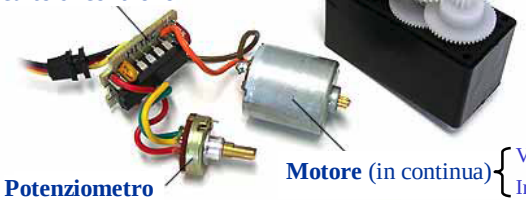


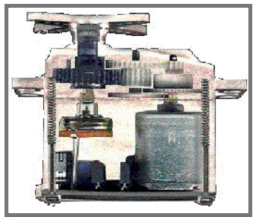
Servomotore che copia un potenziometro (1)

(11')

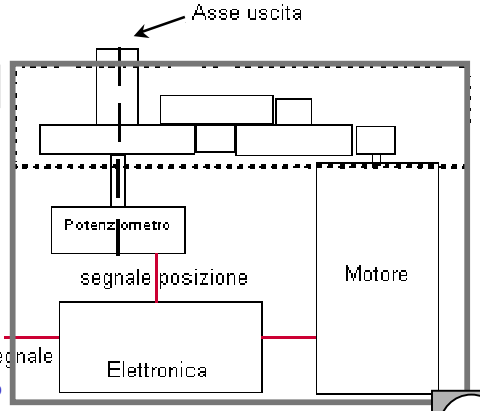
Servo digitale
Posizionamento digitale (+ preciso, con opzioni, programmabile, +forza)
Spesso intercambiabili

Asse di uscita — **Ingranaggi di trasmissione** — **Contenitore** — **Circuito di controllo**



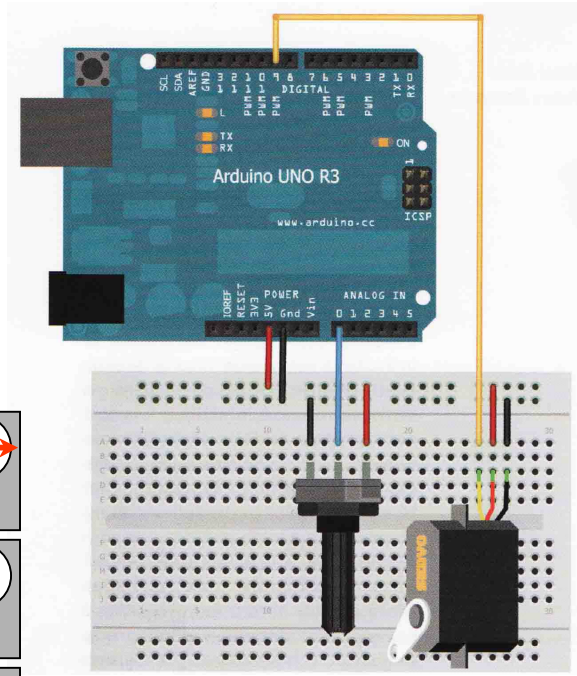


Servo analogico



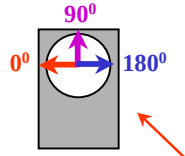
Motore (in continua) { Velocità = k V ; Invers. rotaz. se V cambia segno

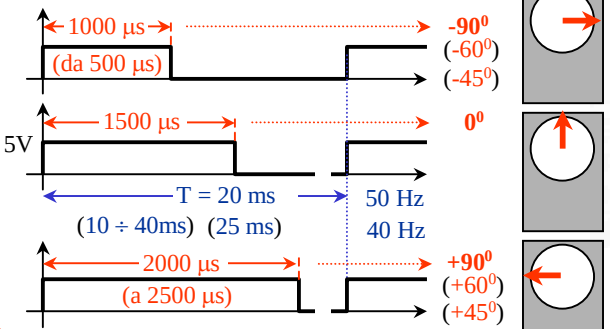
Potenziometro
(coassiale con l'uscita, usato come sensore di posizione angolare)



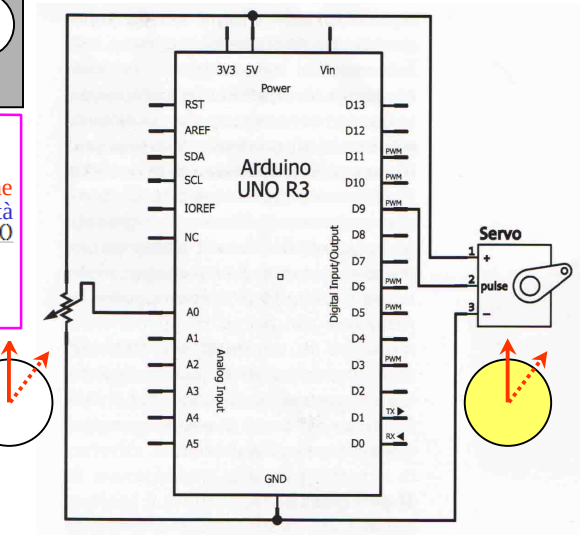

```
#include <Servo.h>   // Direttiva che include la libreria Servo.h
Servo Servo1;        // Dichiaro Servo1 variabile di tipo Servo
int Pot = 0;         // Centrale del potenziometro sul pin A0
int Posizione;       // Posizione del centrale del potenziometro
int Gradi;           // Posizione del servomotore

void setup() {
  Servo1.attach(9);
}

void loop() {
  Posizione = analogRead(Pot); // Leggo il potenziometro
  Gradi = map(Posizione, 0, 1023, 0, 180); // Da una scala all'altra
  Servo1.write(Gradi);  // Sposta il motore nella posizione Gradi
  delay(15);            // Aspetta 15 ms
}
```

Write(90)
Write(0)
Write(180)
Va a 270^0
(o a 120^0)





Non è PWM perché se vario T ma non la durata di T_H, => varia il DC ma non varia la posizione

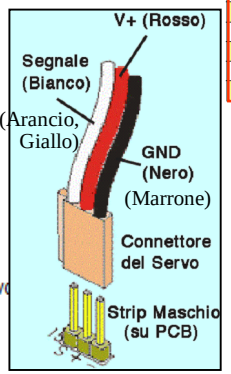


Functions
- attach()
- write() { Posizione o velocità
- writeMicroseconds()
- read()
- attached()
- detach()

Impulso di Controllo					Cavi di Collegamento		
Costruttore	Min.	Centro.	Max	Hz	+V Servo	GND	Controllo
Futaba 116	0.9	1.5	2.1	50	Rosso	Nero	Bianco
Hitech 115	0.9	1.5	2.1	50	Rosso	Nero	Giallo
Graupner/Jr 255	0.8	1.5	2.2	50	Rosso	Marrone	Arancio
Multiplex 34	1.05	1.6	2.15	40	Rosso	Nero	Giallo
Robbe 24	0.65	1.3	1.95	50	Rosso	Nero	Bianco

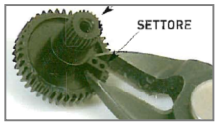
http://www.servodatabase.com/

Servomotore R/C



iFLIGHT(iS900-9G) 9g Micro Servo
Technical Data:
Operating Voltage: 4.8v ~ 6.0v
Operating Speed: 0.10sec/60°
Stall Torque: ≥1.5kg
Dimentions:
Size: 22x12x29mm
Weight: 9g
No Ball bearing,MOS-FET Drive

± 45^O ± 90^O e poi si fermano.

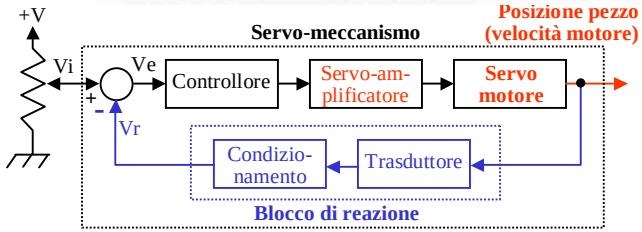


Servo1.attach (pin, min, max); // min e max sono optional e per //default valgono min = 544 e max = 2400 µs

Servo1.detach (pin, min, max); // Si può tornare a fare una // analogWrite su 9 e/o su 10 solo se tutti // i pin prima attaccati sono stati distaccati

Servo1.writeMicroseconds (uS); // Comandato coi µsec invece che coi gradi (risoluzione maggiore perché 1000µsec = 5,5*180^0)

Servo1.read (); // Restituisce l'ultimo valore passato con Servo1.write()



(1) Dal libro "*Primi passi con Arduino*" a cura di Simone Majocchi, speciale di Elettronica In

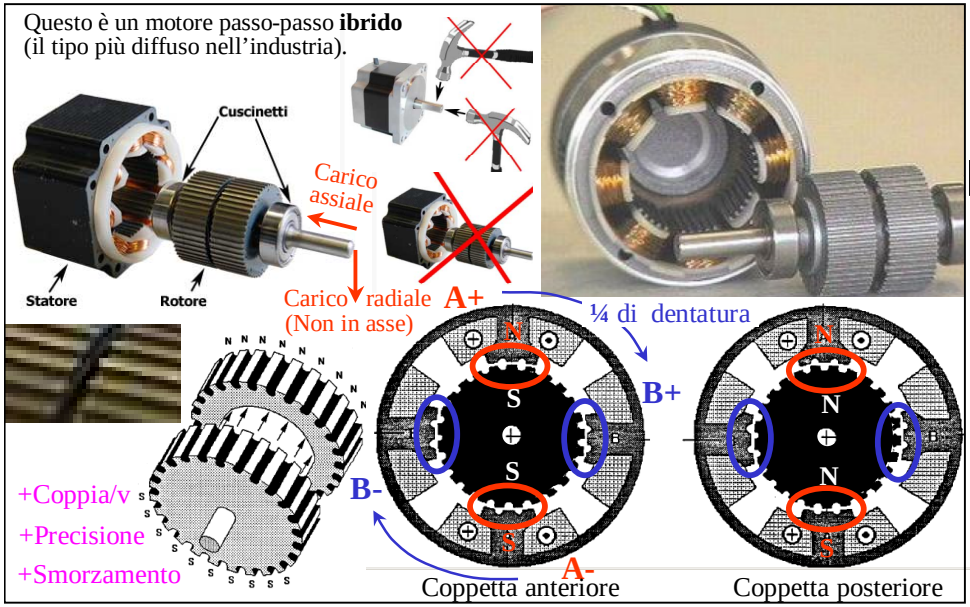
Motori passo-passo (motor stepper)

(15')

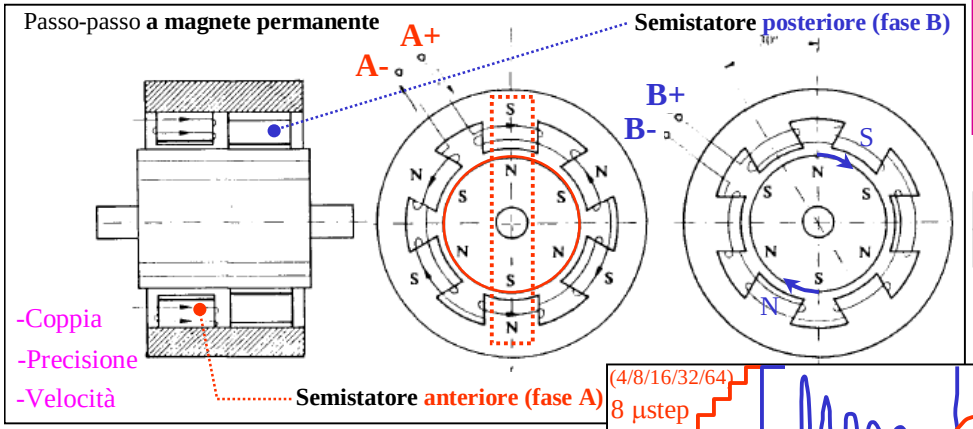
Da www.motoripassopasso.it/ e http://webm.dsea.unipi.it/taponecco/mec catronica.html

I **motori passo-passo** sono motori in continua che si muovono a scatti, assumendo posizioni ben precise senza bisogno di retroazione per verificare la posizione assunta.
Anche la velocità è precisa e quindi non ha bisogna di controllo, perché non cambiano velocità applicando carichi diversi e se non ce la fanno a seguire gli impulsi di comando si fermano.
A parità di volume la coppia erogata da un motore passo-passo è molto più alta di quella di un **motore brushless** (= senza spazzole) o di un tradizionale **motore in continua**. La costruzione semplice e robusta e il fatto che i brushless costano di più hanno favorito la diffusione del motore passo-passo in ogni settore. In breve: un passo-passo è **un brushless economico** in grado di **funzionare senza feedback** (quindi senza sensori e senza i rischi dei sistemi retroazionati)

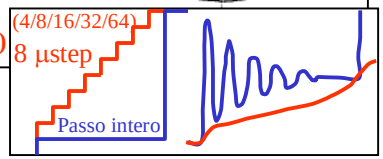
Questo è un motore passo-passo **ibrido** (il tipo più diffuso nell'industria).



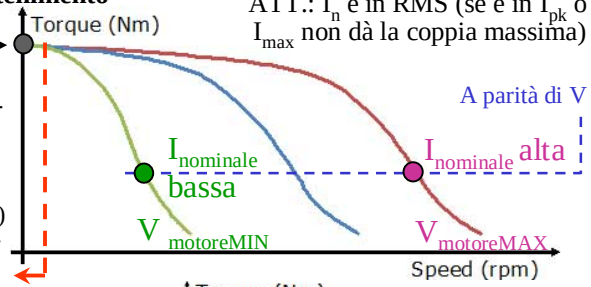
Passo-passo **a magnete permanente**



Il motore passo-passo **a riluttanza variabile** è poco usato in quanto presenta: alti costi, coppie limitate e pessimi smorzamenti



Coppia statica (o **di mantenimento** o **di tenuta**) (**holding torque**) a $I_{nominale}$



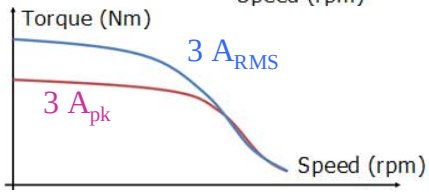
ATT.: I_n è in RMS (se è in I_{pk} o I_{max} non dà la coppia massima)

La **CURVA DI COPPIA** è la **coppia** (**torque**) in Newton per metro, in funzione della **velocità** (**speed**) in giri (revolutions) per minuto.

La **coppia residua** (**detent torque**) è la coppia quando non è alimentato (uno dei pregio di tale motore)

La **corrente nominale** (**rated current**) è la corrente efficace da far passare nelle fasi per avere la coppia statica. Essa va impostata nell'azionamento, ma…

Se lo stesso motore viene fornito con diverse correnti nominali, è da preferire quello ad alta corrente per v alte (e quello a bassa corrente solo per v basse)



48/100/**200**/400 passi/giro

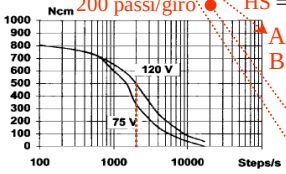
Φ4,2'' L7,0''

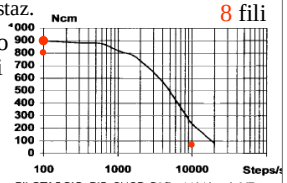
Nema17: 42x42 mm
Nema**23**: 57x57 mm *
Nema**34**: 86x86 mm
Nema**42**: 108x108mm

Flangia	Coppia statica	Prezzo
NEMA23	1 Nm	€ 20,00
NEMA23	2 Nm	€ 30,00
NEMA23	3 Nm	€ 38,00
NEMA34	4 Nm	€ 50,00
NEMA34	8 Nm	€ 70,00
NEMA34	12 Nm	€ 90,00
NEMA42	15 Nm	€ 120,00
NEMA42	22 Nm	€ 180,00

Costa molto meno di un brushless con la stessa C

HY = Hybrid
HS = H alte prestaz.
200 passi/giro
8 fili
A = 1 albero
B = 2 alberi





Specifiche / *Specifications*

		340 A8 340 B8	710 A8 710 B8
- angolo di passo - *step angle*		1,8°	1,8°
- precisione dell'angolo di passo - *step angle accuracy*	%	5	5
- corrente di fase nominale - *rated phase current*	A	3,4	7,1
- resistenza di fase - *phase resistance*	Ω	1,1	0,30
- induttanza di fase - *phase inductance*	mH	6,3	2,0
- coppia di mantenimento unipolare* - *holding torque unipolar**	Ncm	798	830
- coppia di mantenimento bipolare* - *holding torque bipolar**	Ncm	990	1030
- coppia residua - *detent torque*	Ncm	70	70
- inerzia del rotore - *rotor inertia*	g cm²	5500	5500
- massa - *mass*	Kg	7,3	7,3
- tensione massima applicabile - *maximum applicable voltage*	V	140	140
- classe di isolamento - *insulation class*		C: 75°C ±5%	B: 80°C ±10% F: 100°C ±15% H: 125°C

$I_{rotore} > 10\ I_{carico}$

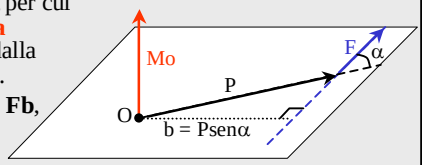
$g*cm^2 = oz*in^2*182,9$

N*m = Kg*m*9,8 (perché F=m*a= Kg_{massa}*9,8 Newton) = **oz*in/141,6** (dove oz=ounce=oncia)

La **coppia** (motrice) è il **momento** applicato dal motore al corpo che fa ruotare

Un vettore forza **F** può produrre rotazione oppure no, per cui dobbiamo introdurre un vettore **momento della forza rispetto al punto O** (**Mo**), che dipende sia da **F** sia dalla **distanza Psenα** della sua retta di azione dal punto O.

Mo = prodotto vettoriale **FxP** e ha modulo **FPsenα = Fb**, **direzione ortogonale al piano F-O**, verso come **l'avanzamento di una vite destrorsa**



Il **momento di inerzia di una massa** si indica con **I**, esprime l'inerzia alla rotazione (**I = M/α**) e dipende dalla massa e dalla sua distanza dal centro di rotazione (**$I = m\ d^2$**)