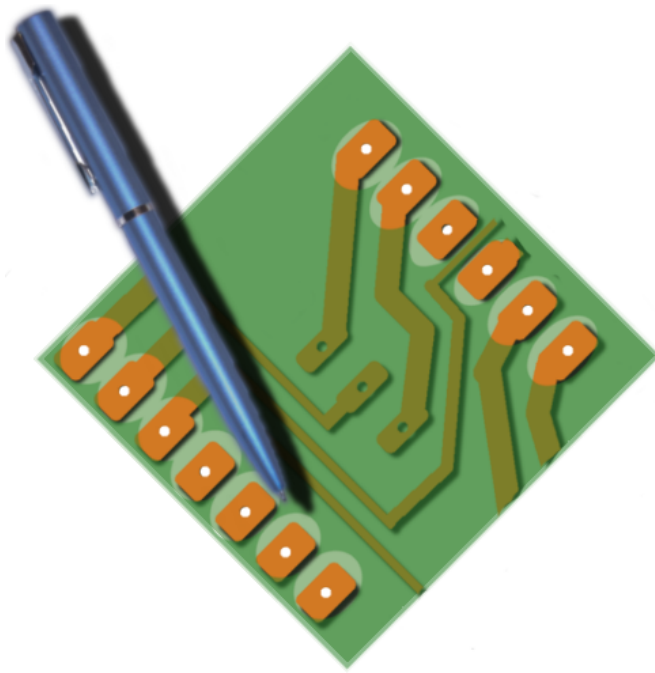


# **FidoCadJ 0.24.3**

## **user manual**

Davide Bucci



26 ottobre 2013



Quest'opera è protetta dalla Creative Commons Public License, versione 2.5 o superiore. Il testo della licenza è disponibile all'indirizzo (in lingua Inglese)

<http://creativecommons.org/licenses/by-nc-nd/3.0/>.

E' possibile riprodurre, diffondere, trasmettere o esporre in pubblico, illustrare, eseguire o rappresentare quest'opera alle condizioni seguenti:

**Attribution** E' necessario attribuire l'opera all'autore o al conceditore nel modo specificato dagli stessi (e in nessun modo che possa portare a credere che gli autori abbiano ceduto l'uso dell'opera.)

**Noncommercial** L'opera non è utilizzabile a fini commerciali.

**No Derivative Works** Non è possibile modificare, trasformare o espandere l'opera.

Qualunque variazione alle condizioni riportate deve essere espressamente approvata dal detentore del copyright (Davide Bucci).

Tutti i nomi commerciali, i logo, i marchi depositati citati in quest'opera sono registrati dai loro proprietari.

# Introduzione

Questo documento è il manuale ufficiale di FidoCadJ. Dopo una breve introduzione sulla storia e sulla nascita di questo software, viene descritto l'uso di base di FidoCadJ. L'obiettivo è di apprendere il disegno di semplici schemi elettrici con i loro relativi circuiti stampati. Il manuale termina con una descrizione dettagliata del formato interno di FidoCadJ (e di FidoCAD) e con alcuni consigli per la corretta installazione sui più comuni sistemi operativi: Linux, MacOSX e Windows.

# Ringraziamenti

Un certo numero di persone ha usato questo programma sin dalla sua prima versione e mi ha aiutato fornendomi preziosi consigli. In tal senso vorrei ringraziare i partecipanti del newsgroup [it.hobby.elettronica](mailto:it.hobby.elettronica) per le loro fruttuose discussioni.

Questo programma è stato testato a fondo su Linux da Stefano Martini. Egli è un alfa e beta tester molto scrupoloso! Vorrei anche ringraziare Olaf Marzocchi e Emanuele Baggetta per i test condotti su MacOSX.

Ringrazio “F. Bertolazzi” per i preziosi consigli circa l'utilizzabilità del programma e per aver costruito una compatibility library per CadSoft Eagle, utile per esportare i disegni FidoCadJ su Eagle.

Ringrazio “Celsius”, che ha testato la parte del programma relativa alla realizzazione dei PCB, comprese le librerie.

Ringrazio Andrea D'Amore, per i suoi consigli riguardanti l'aspetto grafico di FidoCAD su Apple Macintosh, dalla versione 0.21.1

Ringrazio Roby IZ1CYN per le interessanti discussioni riguardanti le librerie e per aver scritto la sezione [A.2](#) di questo manuale, riguardante la installazione su Linux.

Gli utenti Macintosh possono fruire di una versione di FidoCadJ ben integrata al loro sistema operativo grazie a Quaqua di Werner Randelshofer.

Werner ha dato utili consigli circa l'interfaccia utente di FidoCadJ: grazie!

Questo manuale è stato tradotto in inglese da “Pasu”, che vorrei ringraziare per il lavoro svolto. Vorrei anche ringraziare “qhg007” per l'opera di controllo svolta e per le utilissime osservazioni.

Da Aprile 2010, FidoCadJ è stato integrato nel ben noto sito Italiano [www.electroyou.it](http://www.electroyou.it), sul quale gira silenziosamente, convertendo magicamente i disegni postati sul forum dai suoi utenti. Vorrei esprimere la mia gratitudine all'amministratore Zeno Martini e al webmaster Nicolò Martini e a tutti gli altri utenti di quel sito per avermi dato moltissime idee per il controllo batch di FidoCadJ: è stata tracciata una via molto promettente che merita di essere esplorata a fondo. Più recentemente è nata la classe FidoReadPHP, che legge e interpreta i disegni FidoCadJ su un server PHP. E' stata implementata dal noto sito [www.grix.it](http://www.grix.it) grazie ad Arniek e Sstrix. Gloria a voi!

# Licenza FidoCadJ

Copyright © 2007-2012 Davide Bucci [davbucci@tiscali.it](mailto:davbucci@tiscali.it)

Questo programma è gratuito: è possibile distribuirlo e/o modificarlo entro i termini della GNU General Public License emessa dalla Free Software Foundation, versione 3.

Questo programma è distribuito nella speranza che possa essere utile, ma SENZA ALCUNA GARANZIA; senza neppure la garanzia implicita di COMMERCIALIZZABILITA' o IDONEITA' PER UN PARTICOLARE SCOPO. Riferirsi alla GNU General Public License per ulteriori dettagli.

L'utente dovrebbe aver ricevuto una copia della GNU General Public License. Qualora non sia così è possibile reperirla presso <http://www.gnu.org/licenses/>.

# Indice

<b>1. Introduzione</b>	<b>1</b>
1.1. Filosofia FidoCadJ . . . . .	1
1.2. Storia di questo programma . . . . .	2
1.3. FidoCadJ e il futuro . . . . .	3
<b>2. Disegnare con FidoCadJ</b>	<b>6</b>
2.1. Strumenti di disegno . . . . .	6
2.2. un semplice schema elettrico . . . . .	9
2.3. I livelli . . . . .	15
2.4. La griglia . . . . .	15
2.5. un semplice PCB . . . . .	16
2.6. Usare il righello . . . . .	22
2.7. Frecce e stili di tratteggio . . . . .	23
2.8. Esportazione . . . . .	25
2.9. Opzioni da linea di comando . . . . .	25
2.10. Gestione delle librerie . . . . .	29
2.10.1. Usare i file di libreria . . . . .	29
2.10.2. Definire nuovi simboli . . . . .	30
2.10.3. Modificare un simbolo esistente . . . . .	33
<b>3. Formato di disegno, macro e librerie FidoCAD</b>	<b>34</b>
3.1. Descrizione dell'intestazione . . . . .	34
3.2. Sistema di coordinate . . . . .	34
3.3. Elementi di disegno . . . . .	35
3.4. Estensioni FidoCadJ . . . . .	41
3.4.1. Impostazione dei livelli . . . . .	42
3.4.2. Impostazioni delle connessioni elettriche . . . . .	42
3.4.3. Larghezza del tratto . . . . .	43
3.5. Tolleranza agli errori di sintassi . . . . .	43
3.6. Formati delle librerie . . . . .	43
3.7. Librerie standard . . . . .	44
<b>4. Conclusioni</b>	<b>45</b>

## *Indice*

<b>A. Informazioni relative alla piattaforma specifica</b>	<b>46</b>
A.1. MacOSX . . . . .	46
A.1.1. Come scaricare ed eseguire FidoCadJ su MacOSX . . . . .	46
A.2. Linux . . . . .	47
A.2.1. Usare una piattaforma qualunque, da terminale . . . . .	47
A.2.2. Su di un sistema grafico . . . . .	48
A.3. Windows . . . . .	49
A.3.1. come scaricare ed eseguire FidoCadJ . . . . .	49
<b>B. FidoCadJ art</b>	<b>51</b>



# Elenco delle figure

1.1. Parte di uno dei miei post su <a href="http://www.electroyou.it">www.electroyou.it</a> . Con un semplice click è possibile fare lo zoom sullo schema. Con un secondo è possibile ottenere il codice sorgente, che è possibile incollare su FidoCadJ per apportare modifiche. . . . .	4
1.2. Un esempio di un disegno FidoCadJ integrato in un post sul forum <a href="http://www.grix.it">www.grix.it</a> . E' possibile ottenere il sorgente del codice con un semplice click. . . . .	5
2.1. Una tipica sessione FidoCadJ che gira su MacOSX Tiger. Appendix ?? descrive le peculiarità della versione specifica per Macintosh. . . . .	7
2.2. FidoCadJ con Look e Feel Metal. . . . .	7
2.3. La funzione di ricerca per le librerie installate. . . . .	9
2.4. Finestra di dialogo per i parametri testuali di un disegno FidoCadJ. . .	10
2.5. Lo schema elettrico di riferimento: uno specchio di corrente realizzato con transistori NPN. . . . .	10
2.6. Cominciamo a disegnare la coppia di transistori. . . . .	11
2.7. Selezionare e specchiare con il tasto <b>S</b> il transistor sulla sinistra. . . .	12
2.8. Siamo troppo vicini al confine superiore del foglio di disegno, selezioniamo quindi l'intero disegno e spostiamolo verso il centro . . . . .	12
2.9. Il circuito quasi completato. . . . .	13
2.10. Il circuito finale. . . . .	14
2.11. Un semplice amplificatore che fa uso di un transistor NPN connesso in configurazione ad emettitore comune. . . . .	17
2.12. I dispositivi più importanti sono piazzati sulla scheda. . . . .	17
2.13. Sono state aggiunte le linee di alimentazione tramite una linea poligonale. .	18
2.14. Sono state aggiunte le connessioni rimanenti. . . . .	19
2.15. Il PCB quasi completato. . . . .	20
2.16. Il lavoro finito con il silk-screen. . . . .	20
2.17. Il PCB, come appare una volta stampato (specchiato) su un foglio ISO-UNI A4. . . . .	21
2.18. Fare click col tasto destro per attivare il righello FidoCadJ. . . . .	23
2.19. Uno schema elettrico (un GIC di Antoniou) nel quale sono state utilizzate alcune estensioni di FidoCadJ. . . . .	24
2.20. Finestra di dialogo per la curva Bézier utilizzata per lo schema elettrico di figura 2.19 (In francese). . . . .	24
2.21. Come appare il programma su MacOSX, usando il look & feel Motif. . .	29

2.22. Il menu popup che appare cliccando con il tasto destro permette di trasformare un disegno in un simbolo . . . . .	31
2.23. La finestra di dialogo per il nuovo simbolo, tramite la quale è possibile impostare tutte le caratteristiche del simbolo. Si noti che l'origine è definita dai due assi rossi. . . . .	31
2.24. Il simbolo appena creato, mostrato nella lista dei simboli e nel disegno. Sulla sinistra è ancora presente il disegno utilizzato per creare il simbolo. Notare che il punto di controllo è presente nel disegno, per il nuovo simbolo. . . . .	32
2.25. Il menu popup utilizzato per modificare le proprietà dei simboli nella libreria utente. . . . .	33
3.1. La figura 2.19 come apparirebbe su FidoCAD per Windows. . . . .	42
A.1. L'impostazione dei permessi del file, su Ubuntu 8.04. . . . .	49
A.2. Impostazione dei diritti di esecuzione per la Java virtual machine, su Ubuntu 8.04. . . . .	50

## Elenco delle tabelle

2.1. Sommario dei comandi di disegno disponibili in FidoCadJ. I tasti mostrati sulla colonna sinistra permettono la loro rapida selezione attraverso la tastiera. Un click col tasto destro in una delle modalità di inserimento delle primitive permette di accedere alla finestra delle proprietà. . . . .	8
2.2. Lista di tutti i formati esportabili disponibili in FidoCadJ. . . . .	26
3.1. Significato del parametro $a$ per la presenza di una freccia alle estremità di una linea o di una curva Bézier. . . . .	36
3.2. Significato del parametro $b$ per lo stile della freccia. . . . .	36
3.3. Funzione dei bit nel campo dello stile di testo. . . . .	37

# 1. Introduzione

In questo capitolo introdurremo brevemente FidoCadJ. In particolare, analizzeremo la filosofia alla base del programma, come anche una breve descrizione storica del suo sviluppo.

## 1.1. Filosofia FidoCadJ

FidoCAD (senza la J alla fine) era un programma di disegno vettoriale particolarmente adatto al disegno di schemi elettrici e di circuiti stampati. Era particolarmente diffuso nella comunità Usenet italiana degli anni novanta.

E' possibile scaricarlo gratuitamente una versione Windows in lingua Italiana dalla pagina web di Lorenzo Lutti:

<http://www.enetsystems.com/~lorenzo/fidocad.asp>

I file generati da questo programma sono testuali e compatti. Questa caratteristica rende possibile includere disegni in messaggi di testo, come quelli usati nei gruppi non binari Usenet.

Sfortunatamente, FidoCAD esiste solo in versione Windows. Gli utenti Linux hanno la possibilità di farlo girare usando WInE, ma per coloro che utilizzano altre piattaforme (io uso MacOSX) questo non è possibile. Ho quindi deciso di portare il mio piccolo contributo alla comunità Usenet scrivendo FidoCadJ (con, stavolta, la J finale). Questo editor è scritto completamente in Java ed è completamente multipiattaforma. FidoCadJ permette di visualizzare e di modificare un disegno utilizzando il formato FidoCAD.

I vecchi utenti FidoCAD imparano ad usare FidoCadJ rapidamente poichè molti comandi e procedure sono piuttosto simili alla applicazione originale. Al momento della stesura di questo manuale, allo stato dei fatti a mia conoscenza, FidoCadJ è pienamente compatibile con l'originale FidoCAD tranne alcuni dettagli. L'obiettivo della piena compatibilità è stato ricercato il più possibile, ma, recentemente, le necessità di alcuni nuovi utenti hanno spinto verso l'implementazione di nuove estensioni al formato originale.

Alcune delle funzioni offerte da FidoCadJ che non sono presenti nel programma originale sono le capacità di esportare formati. Poichè io sono un utente  $\text{\LaTeX}$ , ho deciso di includere una funzione di esportazione verso un certo numero di formati vettoriali, incluso l'Encapsulated PostScript (EPS). Certamente FidoCadJ può esportare anche verso il ben noto formato PDF. Un altro formato di file utile per gli schemi elettrici è CadSoft Eagle script, che è disponibile dalla versione 0.21 di FidoCadJ. In questo modo, uno schema elettrico disegnato con FidoCadJ può essere esportato su Eagle.

## 1. Introduzione

L'appendice A descrive brevemente come installare FidoCadJ sui più comuni sistemi operativi.

## 1.2. Storia di questo programma

Mi interessavo da tempo di circuiti elettronici. Da quando ho cominciato a seguire diversi gruppi Usenet, ho notato che molti schemi elettrici venivano distribuiti usando il formato FidoCAD per Windows. Questo permetteva di evitare gli scomodi disegni ASCII. Da qualche anno non uso più Windows, quindi è stato praticamente impossibile per me riuscire a decifrarli, così ho voluto provare a risolvere il problema.

*A proposito, credo che sia meglio lavorare per trovare una soluzione piuttosto che affermare che un sistema operativo differente da Windows sia carente di software.*

La prima cosa che ho fatto è stata quella di studiare il formato di file utilizzato da FidoCAD e scrivere una applet Java chiamata FidoReadJ, capace di analizzare un circuito e di mostrarlo in una pagina web. Ho cominciato cercando ovunque (vecchi post, pagine web) facendo molto reverse engineering sui file FidoCAD esistenti. Ho scaricato i sorgenti FidoCAD, scritti in un C++ molto pulito da Lorenzo Lutti. Ho fatto questo lavoro intorno a Marzo 2007. Qualche mese dopo, la applet era online ed era già stata testata da una parte della comunità di it.hobby.elettronica e di it.hobby.fai-da-te.<sup>1</sup>

*a dire il vero, il primo tentativo di fare un sistema di disegno vettoriale risale al 1993.*

Fin da quando ho un interprete per il formato FidoCAD, mi sono interessato a continuare il lavoro in modo da ottenere un editor completo. La maggior parte del lavoro è stato fatto in diversi passi, tra Gennaio e Luglio 2008. FidoCadJ non è un adattamento o un porting di FidoCAD per Windows, ma è un programma completamente riscritto.

La scelta di usare Java è dovuta al fatto che negli ultimi anni ho cambiato diversi sistemi operativi. Impegnare del tempo su qualcosa di non completamente portabile non mi sembrava molto interessante. Lo sforzo profuso all'apprendimento dell'ambiente Cocoa avrebbe dato dei risultati migliori su MacOSX, ma avrebbe reso FidoCadJ completamente non portabile. Non mi dedico interamente al computer e il tempo speso a programmare è tempo portato via ai miei interessi verso l'elettronica.

Infatti, una semplice analisi al codice sorgente di FidoCadJ dimostra che io non sia un vero programmatore purista Java e orientato agli oggetti e sono sicuro che molte soluzioni adottate siano più pratiche che eleganti.

In definitiva la cosa importante è l'usabilità del programma da parte dell'utente, più che una scelta specifica di un linguaggio di programmazione. Per questa ragione sono sempre attento ai suggerimenti, in modo da comprendere come proseguire lo sviluppo del progetto. Ricapitolando, so che Java non è la scelta perfetta o la soluzione ad ogni problema. Comunque sono sicuro che la sua cattiva nomea derivi da applicazioni mal scritte, le quali non sono ben integrate con l'ambiente dell'utente.

Senza voler arrivare alla perfezione, conoscendo i miei limiti di programmatore, il mio intento è stato quello di voler fare in modo che FidoCadJ NON fosse un'altra applicazione di scarsa qualità. Per questa ragione ogni segnalazione relativa a qualche bug o all'usabilità del programma sarà molto apprezzata.

<sup>1</sup>FidoReadJ è ancora disponibile all'indirizzo:

<http://davbucci.cher-alice.fr/index.php?argument=elettronica/fidoreadj/fidoreadj.inc>.

Da Novembre 2009 ho aperto un progetto SourceForge dedicato a FidoCadJ. Da quella pagina potrete scaricare qualunque eseguibile, manuale e anche il codice sorgente. Potrete partecipare attivamente allo sviluppo di FidoCadJ lavorando sul suo codice sorgente usando Subversion (SVN) oppure il SVN browser predisposto da SorceForge:

<http://fidocadj.svn.sourceforge.net/viewvc/fidocadj/>

Se volete contribuire alla comunità FidoCadJ, non dovete preoccuparvi: non dovete essere esperti nella programmazione Java. Potrete, per esempio, tradurre l'interfaccia utente o il manuale in un'altra lingua, controllare il manuale esistente per trovare incongruenze, refusi o errori. FidoCadJ è disponibile, attualmente, in italiano, francese, inglese, spagnolo, tedesco, cinese, giapponese e olandese.

E' piuttosto semplice tradurre i menu e i comandi, quindi, se vi facesse piacere vedere FidoCadJ nella vostra madrelingua, fatemelo sapere!

C'è anche del lavoro da fare per tradurre le librerie (almeno quella standard) e, ovviamente, il manuale.

Potete anche lavorare alla libreria standard...

C'è da considerare che solo metà del mio tempo libero che dedico a FidoCadJ viene usato per programmare. Il resto viene speso per rispondere a domande degli utenti, scrivere e migliorare la documentazione e così via. Su SourceForge, potete partecipare ai forum, scrivere delle recensioni sul programma e anche suggerire miglioramenti o segnalare dei bug:

<http://sourceforge.net/projects/FidoCadJ/>

## 1.3. FidoCadJ e il futuro

Ad Aprile 2010, sono capitato casualmente in una discussione sul ben noto sito Italiano [www.electroyou.it](http://www.electroyou.it). Un utente, Piercarlo Boletti, proponeva di integrare uno strumento per il disegno elettrico direttamente dentro il forum; più o meno come era già stato fatto per le equazioni  $\text{\LaTeX}$ .

Poichè FidoCadJ era stato citato nel suo intervento, mi sono iscritto al forum e mi sono offerto di collaborare. Ho interagito con il gentilissimo webmaster per qualche giorno e voilà, il sistema è stato reso operativo: un semplice copia-incolla nel post del codice che descrive il disegno, entro un paio di tag, e il software che fa girare FidoCadJ sul forum restituisce una immagine, direttamente dal codice. Questa immagine viene mostrata nel post sul forum, ma il suo codice sorgente rimane comunque disponibile. In questo modo il lettore della discussione non deve avere alcunchè installato sul proprio computer per visualizzarla, ma, qualora desideri apportare delle modifiche, può ottenere il codice con pochi click.

Figure 1.1 mostra parte di un mio post scritto su [www.electroyou.it](http://www.electroyou.it). Questo sistema è così potente e flessibile che io stesso sono stato il primo ad essere sorpreso dall'entusiasmo dimostrato dagli utenti! <sup>2</sup> FidoCadJ ha dimostrato di essere utile anche per disegni non direttamente collegati all'elettronica (vai all'appendice B per capire cosa intendo).

---

<sup>2</sup>qui si trova l'articolo che scrissi, in Italiano:

<http://www.electroyou.it/darwinne/wiki/FidoCadJ>

## 1. Introduzione

[12] Re: Orologi e orologiai  
di **DarwinNE** il 25 mag 2011, 1:36

**TardoFreak** ha scritto:  
*Beh ... ecco ... timidamente potrei offrirti di sviluppare il firmware (a disposizione di tutti) per fare l'orologio di EY. 😊*

Penso proprio che i ragazzi abbiano intenzione di lasciare fuori i microcontrollori, per fare un ottimo esercizio di stile. Perché non utilizzare un contatore analogico? Anni fa ne avevo realizzato uno per un tracciacurve, è un circuito un bel po' critico, ma abbastanza affascinante. Penso che parte della criticità venga dal fatto che ho simulato degli unijunzione programmabili con una coppia di transistor NPN+PNP (Q2 e Q3, Q7 e Q8). Inoltre, i generatori di corrente che ho adottato sono probabilmente migliorabili.

Sorgente FidoCadJ | Ingrandisci | Cos'è FidoCad

Figura 1.1.: Parte di uno dei miei post su [www.electroyou.it](http://www.electroyou.it). Con un semplice click è possibile fare lo zoom sullo schema. Con un secondo è possibile ottenere il codice sorgente, che è possibile incollare su FidoCadJ per apportare modifiche.

### 1.3. FidoCadJ e il futuro

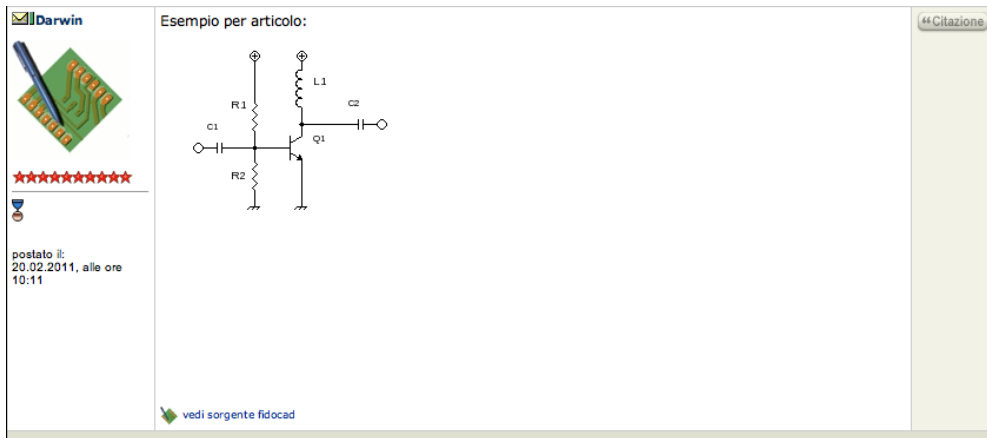


Figura 1.2.: Un esempio di un disegno FidoCadJ integrato in un post sul forum [www.grix.it](http://www.grix.it). E' possibile ottenere il sorgente del codice con un semplice click.

Il successo che ha avuto FidoCadJ su [www.electroyou.it](http://www.electroyou.it) ha stimolato altre richieste di implementazione su altre piattaforme, in particolare su [www.grix.it](http://www.grix.it), un altro ben noto sito Italiano dedicato all'elettronica. FidoCadJ è anche usato dalla comunità di [Matematicamente](http://Matematicamente.it). Relativamente a [www.grix.it](http://www.grix.it), il server non può far girare un programma Java, così è stata scritta la classe FidoReadPHP che può essere fatta girare da un interprete PHP, per ottenere le immagini dal codice sorgente FidoCadJ. Il progetto FidoReadPHP è open source ed è disponibile su SourceForge:

<https://sourceforge.net/projects/fidoreadphp/>

Il risultato è piuttosto simile a quello ottenuto su [www.electroyou.it](http://www.electroyou.it), anche se le capacità del PHP sono piuttosto limitate rispetto al Java.

<sup>3</sup> Figure 1.2 mostra qualche esempio di disegno ottenuto con FidoReadPHP.

Io credo che il futuro di FidoCadJ non sarà più complesso di così, come CAD completo per elettronica. Probabilmente, l'integrazione con i gruppi di discussione e i forum potrebbero essere sviluppati. Il cammino intercorso ha mostrato che questo è possibile e che l'entusiasmo degli utenti è stato stimolante.

---

<sup>3</sup>Ecco l'articolo in Italiano:

<http://www.grix.it/viewer.php?page=9335>

## 2. Disegnare con FidoCadJ

*Gli utenti Mac esperti  
noteranno che tutti i menu  
sono al loro posto!*

L'uso di FidoCadJ dovrebbe essere intuitivo per coloro che hanno già utilizzato una applicazione di disegno vettoriale. Una videata del programma su MacOSX è riportata in in Fig. 2.1; Alcuni dettagli potrebbero essere diversi quando il programma gira su altri sistemi operativi (Per esempio Fig. 2.2 mostra il risultato con Look and Feel Metal su Sun/Oracle), ma la filosofia è sempre la stessa. Vedremo quali sono le caratteristiche del programma e i suoi elementi di base (le primitive) che formano un disegno FidoCadJ.

### 2.1. Strumenti di disegno

*Questo modo di fare è  
ispirato alle vecchie radio a  
valvole termoelettroniche e ai  
loro selettori moda anni 70*

Nella barra degli strumenti (in cima alla finestra), possiamo trovare gli strumenti che permettono la creazione e la modifica di un disegno. Table 2.1 mostra un breve riassunto delle funzionalità e dei comandi e ne descrive l'uso possibile. Si può notare che, una volta premuto il tasto, esso rimarrà selezionato finché non si selezionerà un'altra funzione dalla barra degli strumenti. Dalla barra degli strumenti è possibile selezionare quale primitiva di disegno usare.<sup>1</sup> Sulla destra, un menu a discesa mostrerà il livello corrente (vedere 2.3 per ulteriori informazioni).

La barra di comando può essere parzialmente personalizzata. In particolare è possibile scegliere dove posizionare l'icona per ogni tasto o l'icona e la sua descrizione testuale. Le icone sono selezionabili in due diversi formati. Per cambiare queste impostazioni vedere il menu "File/Options".

<sup>2</sup> Qualunque cambiamento nelle impostazioni verrà applicato al riavvio dell'applicazione, poichè è poco probabile che si apportino cambiamenti tutti i giorni. Le figure 2.1 e 2.2 mostrano la barra dei comandi (a destra sotto la barra del titolo), configurata per mostrare testo e icone nella loro versione più piccola. Sulla seconda riga, partendo da sinistra possiamo vedere le impostazioni per lo zoom e i pulsanti "Fit", "Show grid" e "Snap to grid". Il primo permette di selezionare automaticamente il miglior zoom per mostrare tutto il disegno sullo schermo. Il secondo permette di commutare la visibilità della griglia, mentre il terzo permette di selezionare se l'elemento prescelto debba essere posizionato sul punto di griglia più vicino o no. Nel caso in cui ci sia il bisogno di posizionare con precisione gli elementi, è possibile premere il tasto Alt e utilizzare i tasti cursore.

<sup>1</sup>Per altre informazioni sugli elementi di disegno, vedere 3.3.

<sup>2</sup>Ad eccezione di MacOSX, dove queste impostazioni si trovano nel menù di FidoCadJ e sono chiamate "Preferences".



## 2.1. Strumenti di disegno

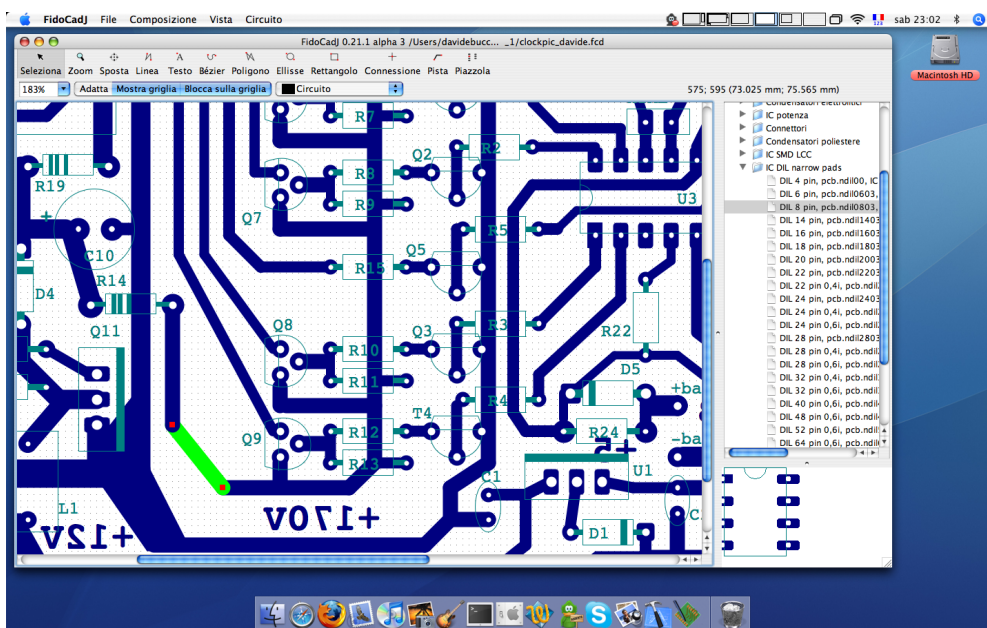


Figura 2.1.: Una tipica sessione FidoCadJ che gira su MacOSX Tiger. Appendix ?? descrive le peculiarità della versione specifica per Macintosh.

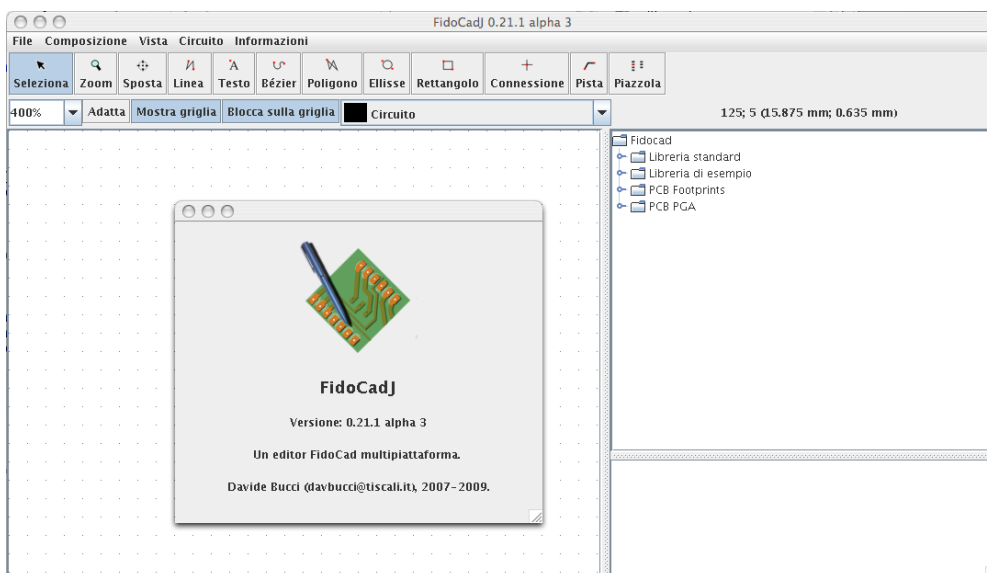


Figura 2.2.: FidoCadJ con Look e Feel Metal.

## 2. Disegnare con FidoCadJ














Tasto	Comando	Uso
<span>A</span> o Barra Spaziatrice	 SELECT	Seleziona uno o più elementi grafici. Premere <span>Control</span> ( <span>Command</span> solo su MacO-SX) per selezioni multiple o per deselezionare un elemento. Cliccare e trascinare per selezionare diversi elementi in un'area. Premere <span>R</span> per ruotare un elemento selezionato. Premere <span>S</span> per specchiare un elemento selezionato. Fare doppio click su un elemento per modificare le sue proprietà.
	 ZOOM	Cliccare col tasto sinistro per incrementare lo zoom. Cliccare col destro per decrementarlo.
	 MOVE	Cliccare sul disegno e muovere il mouse per muovere il disegno.
<span>L</span>	 LINE	Inserisce una linea o una serie di linee. Premere <span>Esc</span> o fare doppio click per terminare l'inserimento
<span>T</span>	 TEXT	Inserisce una stringa testuale.
<span>B</span>	 BÉZIER	Disegna una curva Bézier
<span>P</span>	 POLYLINE	Disegna un poligono. Un doppio click o una pressione del tasto <span>Esc</span> fanno terminare l'inserimento di nuovi vertici.
<span>K</span>	 CURVE	Curva spline aperta o chiusa. Un doppio click o una pressione del tasto <span>Esc</span> fanno terminare l'inserimento di nuovi vertici.
<span>E</span>	 ELLIPSE	Disegna una ellisse, piena o vuota (Tenere premuto il tasto <span>Control</span> per disegnare un cerchio).
<span>G</span>	 RECT.	Disegna un rettangolo pieno o vuoto. (Tenere premuto il tasto <span>Control</span> per disegnare un quadrato).
<span>C</span>	 JUNCTION	Inserisce una connessione elettrica.
<span>I</span>	 PCB TRACK	Disegna una traccia PCB. La larghezza preimpostata può essere modificata accedendo al menu "File/Options".
<span>Z</span>	 PCB PAD	Disegna una piazzola PCB. La dimensione preimpostata può essere modificata accedendo al menu "File/Options".

Tabella 2.1.: Sommario dei comandi di disegno disponibili in FidoCadJ. I tasti mostrati sulla colonna sinistra permettono la loro rapida selezione attraverso la tastiera. Un click col tasto destro in una delle modalità di inserimento delle primitive permette di accedere alla finestra delle proprietà.

## 2.2. un semplice schema elettrico

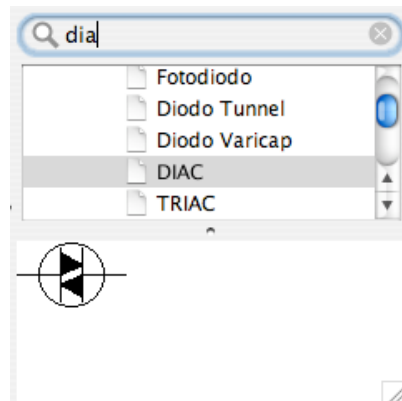


Figura 2.3.: La funzione di ricerca per le librerie installate.

Sulla destra vengono mostrate in una lista ad albero i simboli (anche chiamate macro) delle librerie caricate. Per inserire nel disegno un elemento da una libreria, è sufficiente selezionarlo dalla lista e posizionarlo, cliccando, sul disegno. Le librerie FidoCAD includono tutti i simboli standard utilizzati negli schemi elettrici e una larga scelta di footprint per il disegno di PCB.

E' possibile anche effettuare ricerche rapide nelle librerie caricate in FidoCadJ. E' sufficiente digitare qualcosa nella area di testo che compare sopra la lista ad albero che rappresenta le librerie installate (riferirsi alla figura 2.3) Con i tasti cursore è possibile spostarsi all'interno dei risultati trovati.

La figura 2.4 mostra un esempio di cosa può essere ottenuto, facendo un doppio-click, nel modo seleziona, su un elemento del disegno (in questo caso una stringa di testo). Attraverso questa finestra è possibile modificare tutti i parametri (coordinate, rotazioni...) di ogni elemento del disegno. L'aspetto di questa finestra è variabile perchè le proprietà che possono essere modificate dipendono dall'elemento selezionato.

## 2.2. un semplice schema elettrico

Volendo fare un esempio d'uso di questa applicazione, mostreremo come disegnare il semplice schema elettrico di figura 2.5.

Avendo FidoCadJ aperto, apriamo un nuovo disegno utilizzando il menu "File/-New".

Cominciamo disponendo nell'area di disegno i simboli per i due transistori, attorno ai quali è costruito il nostro schema elettrico. Per fare ciò possiamo utilizzare le macro contenute nella libreria standard, la quale viene caricata automaticamente all'avvio e si trova sulla parte destra dello schermo. La macro di cui abbiamo bisogno è chiamata "NPN transistor" ed è inclusa nella categoria "Diodes and transistors" nella "Standard library". Cliccando sul nome dell'elemento da selezionare è possibile piazzare la macro desiderata in una coordinata qualunque del disegno (FidoCadJ visualizzerà

## 2. Disegnare con FidoCadJ

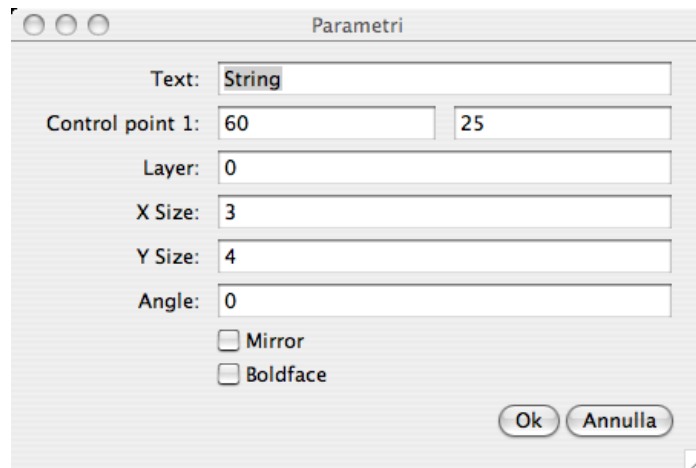


Figura 2.4.: Finestra di dialogo per i parametri testuali di un disegno FidoCadJ.

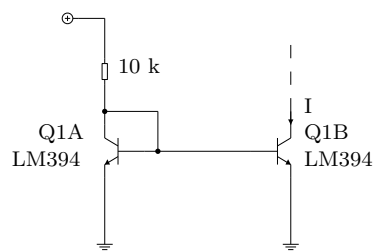


Figura 2.5.: Lo schema elettrico di riferimento: uno specchio di corrente realizzato con transistori NPN.

## 2.2. un semplice schema elettrico

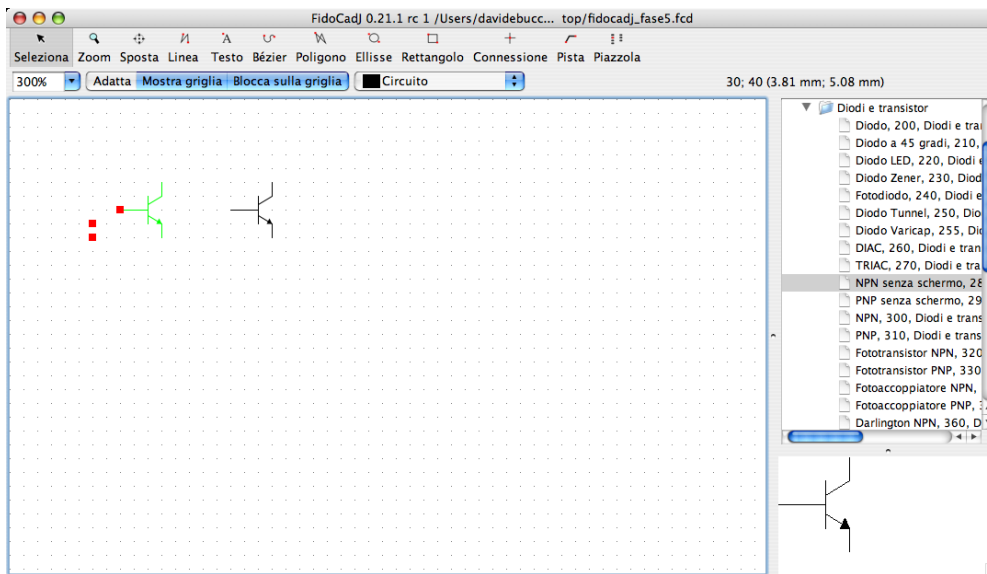


Figura 2.6.: Cominciamo a disegnare la coppia di transistori.

un'anteprima), cliccando una seconda volta si opererà il piazzamento alla coordinata desiderata. A questo punto il disegno dovrebbe essere simile a quello mostrato in figura 2.6.

Possiamo osservare che il transistore bipolare sulla sinistra non è orientato correttamente. Per risolvere questo problema è sufficiente cliccare su “Select”, dalla barra degli strumenti, selezionare il transistore (che verrà evidenziato in verde. Verranno visualizzati anche i suoi tre punti di controllo, identificati da tre piccoli quadrati rossi) e premere il tasto **[S]** per ottenerne una versione specchiata. E' anche possibile premere il tasto **[S]** subito dopo aver selezionato il simbolo dalla libreria, al momento del posizionamento. Si otterrà un risultato simile a quello mostrato in figura 2.7.

Usando lo strumento “Line” dalla barra degli strumenti è possibile disegnare le connessioni elettriche, almeno fino a quando non si arriva troppo vicino ai bordi dell'area di disegno. Il problema può essere facilmente risolto selezionando l'intero disegno: usando il modo “Select” si può cliccare sul bordo superiore sinistro del disegno e, tenendo premuto il tasto sinistro del mouse, trascinare il cursore fino al bordo inferiore destro. Apparirà un rettangolo dal perimetro verde per indicare che stiamo selezionando tutti gli elementi in esso inclusi. Per poter spostare tutto quanto abbiamo disegnato finora, come prima cosa, è necessario selezionarlo (see figure 2.8). Successivamente sarà possibile, cliccando su un elemento selezionato qualunque, trascinare la selezione nella posizione desiderata.

E' ora possibile continuare a piazzare le altre parti del circuito, in particolare un resistore (Standard Library/Discrete devices/Resistor) e il terminale per l'alimentazione positiva (Standard Library/Basic symbols/Terminal +). E' necessario quindi ruotarlo

## 2. Disegnare con FidoCadJ

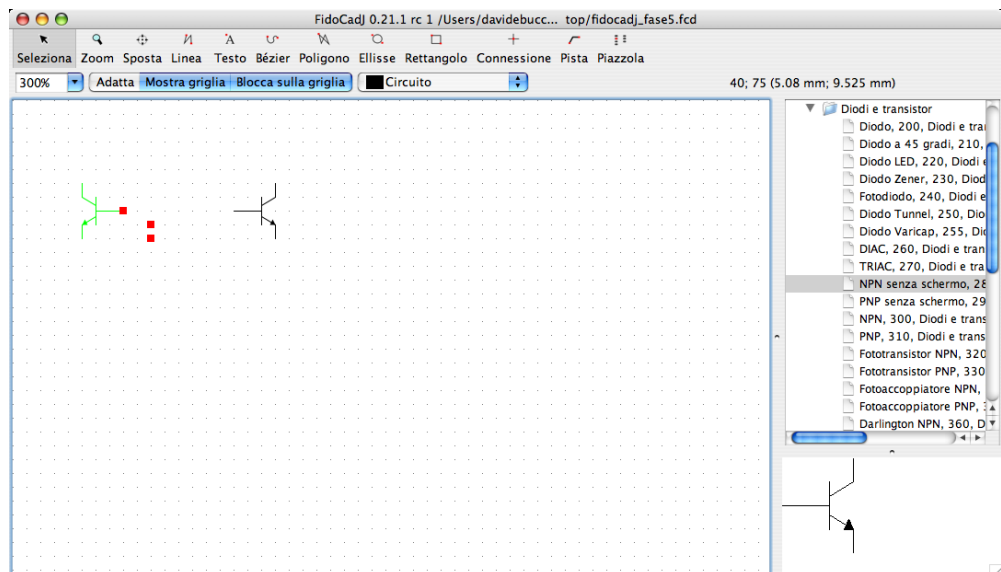


Figura 2.7.: Selezionare e specchiare con il tasto **S** il transistor sulla sinistra.

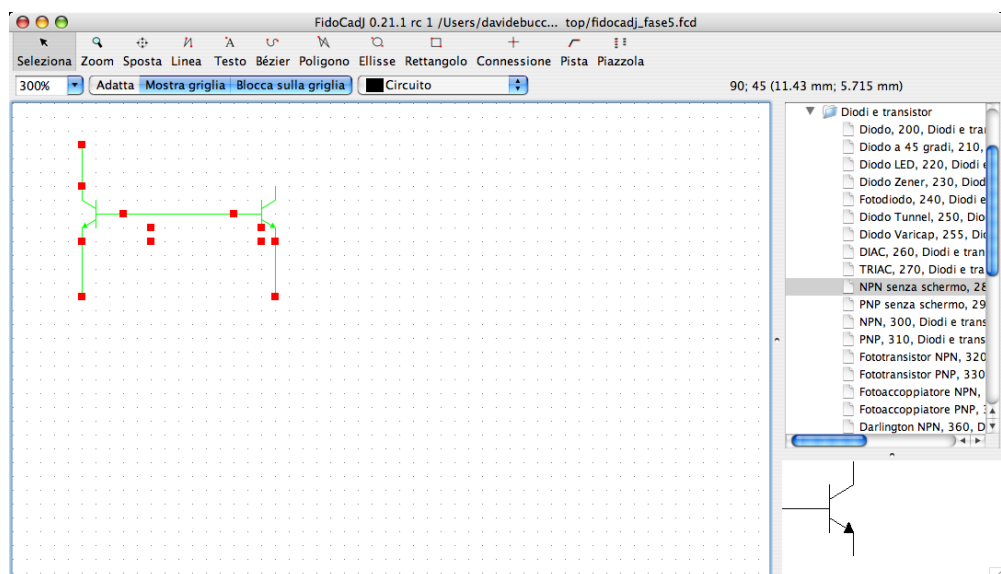


Figura 2.8.: Siamo troppo vicini al confine superiore del foglio di disegno, selezioniamo quindi l'intero disegno e spostiamolo verso il centro

## 2.2. un semplice schema elettrico

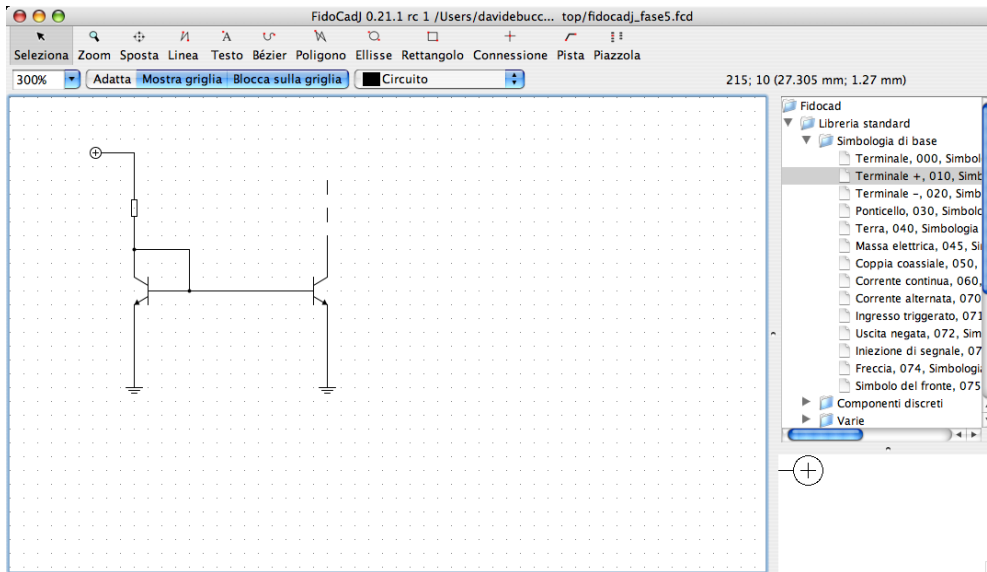


Figura 2.9.: Il circuito quasi completato.

in modo da piazzarlo nella posizione desiderata. Nuovamente è possibile selezionarlo e quindi premere il tasto **[R]** fino a che si è ottenuto il risultato desiderato. Dovremmo ora avere ottenuto una schermata simile a quella di figura 2.9.

Per completare lo schema elettrico è sufficiente aggiungere una stringa testuale e le frecce per indicare la direzione della corrente. Per quest'ultima è disponibile una macro chiamata "Arrow", contenuta in "Standard library/Basic symbols". Per piazzare il testo, è possibile premere il tasto "Text" dalla barra degli strumenti e cliccare nell'area di disegno nella posizione desiderata. Verrà stampato il testo predefinito "String" che è possibile modificare, in modalità selezione, facendoci doppio click sopra (si veda la figura 2.4). La sigla del transistor e il suo nome di riferimento utilizzati (nel nostro esempio si fa riferimento ad una coppia di transistor appaiata) vengono specificati nei campi "Name" e "Value" a cui si può accedere tramite la modalità selezione facendo doppio click sulla macro.

La dimensione suggerita per lavorare con i circuiti elettrici è di 4 unità in verticale e di 3 unità in orizzontale. Il circuito completo è mostrato in figura 2.10.

A titolo di curiosità viene qui riportato il codice che descrive il circuito del nostro esempio. Per accedere al codice è sufficiente selezionare "Insert Circuit" dal menu "Circuit". E' ora possibile copiare ed incollare il circuito in un messaggio e-mail, in un newsgroup, o in un forum.

<sup>3</sup>L'opzione che permette di aggiungere un nome ed un valore ad una macro o ad un simbolo di un componente è una estensione introdotta con FidoCadJ non presenti nel FidoCAD originale. Confrontare la sezione ?? per altre informazioni riguardanti la compatibilità.

## 2. Disegnare con FidoCadJ

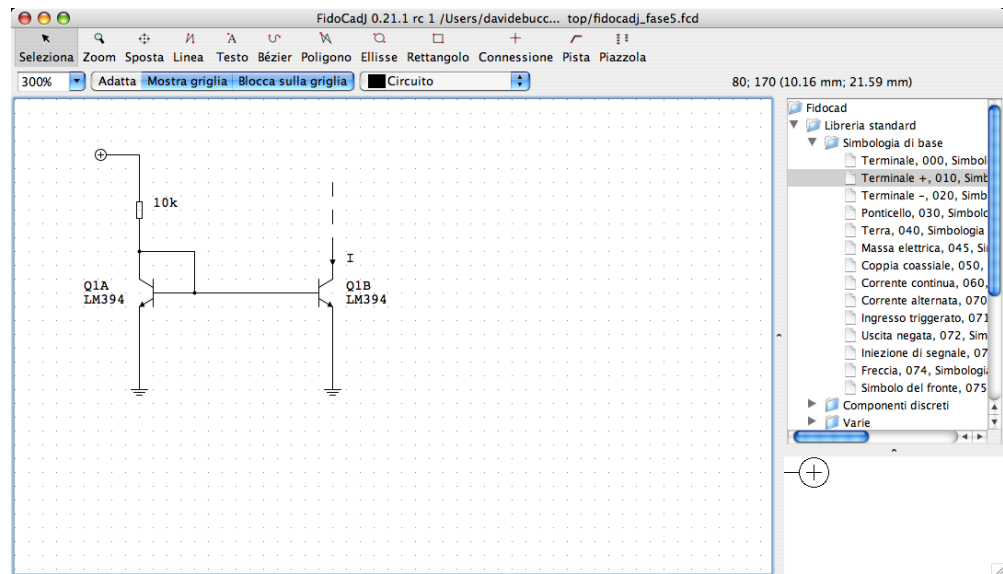


Figura 2.10.: Il circuito finale.

```
[FidoCAD]
MC 95 65 0 0 280
FCJ
TY 115 60 4 3 0 0 0 * Q1B
TY 115 65 4 3 0 0 0 * LM394
MC 55 65 0 1 280
FCJ
TY 20 60 4 3 0 0 0 * Q1A
TY 20 65 4 3 0 0 0 * LM394
LI 55 65 95 65 0
LI 40 75 40 95 0
LI 110 75 110 95 0
LI 40 40 40 55 0
MC 40 30 0 0 115
LI 40 15 40 30 0
LI 30 15 40 15 0
MC 30 15 2 0 010
LI 40 50 60 50 0
LI 60 50 60 65 0
SA 60 65 0
SA 40 50 0
LI 110 45 110 55 0
LI 110 35 110 40 0
LI 110 25 110 30 0
```



```
MC 40 95 0 0 040
MC 110 95 0 0 040
TY 45 30 4 3 0 0 0 * 10 k
TY 115 50 4 3 0 0 0 * I
MC 110 50 1 0 074
```

Se si è interessati al formato di esportazione utilizzato da FidoCAD si può consultare il capitolo 3.

In realtà non è necessario usare il comando “Insert circuit”; è possibile semplicemente selezionare l'intero disegno con `Ctrl+A`, copiarlo (selezionando “edit/copy” o premendo `Ctrl+C`) e incollarlo nel messaggio in fase di scrittura per fare in modo che il codice sia aggiunto automaticamente.

In funzione delle impostazioni programmate, FidoCadJ potrebbe applicare uno spostamento diagonale agli elementi copiati e incollati. Le distanze  $x$  e  $y$  sono uguali alla distanza di un punto di griglia. Il comportamento predefinito è quello di procedere in questo modo, per differenziare gli elementi incollati da quelli originali. In alcune situazioni questo potrebbe essere inutile e problematico, ma queste impostazioni possono essere cambiate tramite il menu “Options”.

## 2.3. I livelli

Un modo per immaginare i livelli è quello di immaginare un disegno stampato su un foglio acetato. Il disegno definitivo sarà dato dalla combinazione di tutti i livelli, che saranno sovrapposti come i fogli di acetato. Ogni livello è caratterizzato da un diverso colore e può essere visibile o invisibile. Questo approccio è comune a molti sistemi CAD, poichè permette una rappresentazione e una gestione semplice di diverse parti di un disegno che possono essere sovrapposte, come per esempio in un progetto di un PCB.

FidoCadJ supporta fino a 16 livelli, numerati da 0 a 15. Convenzionalmente, alcuni livelli hanno uno scopo specifico. In particolare, il livello zero è usato per gli schemi elettrici, il livello uno per il rame lato saldature, il livello due per il rame lato componenti e il lato tre per il silk-screen. I livelli rimanenti non hanno uno scopo prefissato e possono essere utilizzati liberamente.

Il nome e il colore di ogni livello può essere specificato tramite il menu “View/-Layer”. Dallo stesso menu è possibile selezionare i livelli da mostrare sullo schermo o da stampare.

L'ordine dei livelli è importante, poichè un livello con un numero inferiore verrà disegnato per primo. I livelli successivi copriranno i livelli di ordine più basso.

## 2.4. La griglia

L'unità logica in FidoCadJ è 5 mil (127 micron) e le “mezze unità” non sono permesse, poichè le coordinate di ogni elemento grafico devono essere numeri interi. Questo permette di ottenere una risoluzione sufficientemente fine per il disegno di uno schema

## 2. Disegnare con FidoCadJ

elettrico e della stragrande maggioranza dei PCB. In ogni caso, per facilitare il disegno, l'applicazione permette di selezionare una griglia meno fine, che forza il mouse ad allinearsi con il punto più vicino alla griglia stessa.

Per abilitare questa funzionalità sono presenti due tasti, “Show Grid” e “Snap”, che permettono di commutare tra griglia visibile/invisibile e di forzare il cursore del mouse a spostarsi sulla griglia o a lasciarlo libero di muoversi, rispettivamente. La granularità della griglia può essere impostata tramite una finestra di dialogo alla quale è possibile accedere dal menu “File/Options”.

## 2.5. un semplice PCB

*E' il lettore che trova la proprio modus operandi: qualche scarabocchio su un foglio e una matita (e un sacco di cancellature) faranno risparmiare del tempo perchè daranno una idea chiara di come procedere usando il computer.*

Per fare pratica con quanto finora appreso, vedremo come progettare un semplice PCB. Diversamente da altri software per il CAD elettronico, i quali sono molto potenti e talvolta difficili da gestire, FidoCadJ è la versione elettronica dei buoni vecchi trasferibili R41. Chiaramente avendo a disposizione un calcolatore avremo a disposizione tutta la flessibilità offerta dalla macchina.

Degno di nota è che il progetto di un PCB, in particolar modo se complesso, non è un compito semplice. Il piazzamento automatico e lo sbroglio automatico promettono miracoli, almeno così si legge sulle pubblicità fatte dalle più grandi compagnie produttrici di CAD. Non c'è dubbio, invece, che l'esperienza del progettista, in questo campo, giochi un ruolo molto importante. FidoCadJ rappresenta uno strumento molto immediato e veloce per il progetto di piccoli PCB per il fai da te. Vedremo qui come disegnare un semplice, ma completo, PCB.

Mi permetto di suggerire di cominciare avendo le idee chiare di dove piazzare ogni componente e di come disegnare le piste, in modo da minimizzare gli incroci il più possibile.

In questo caso bareremo un po', cominciando dal risultato che si vuole ottenere, come mostrato in figura 2.11. Si tratta di un semplice amplificatore a emettitore comune costruito attorno ad un transistor NPN, di tipo BC547 o simili. Giova immaginarsi la scheda come se fosse trasparente, guardandola dal lato componenti. Per questo motivo il silk-screen e i componenti disegnati sulla scheda sono un valido riferimento, anche se probabilmente non li si vorrà stampare sulla scheda, per un progetto amatoriale.

Il primo consiglio che posso suggerire è quello di piazzare tutti i componenti al meglio. Nell'esempio proposto essi sono: il transistor (dalla libreria “PCB footprints/3 terminals semiconductors/TO92”), il resistore (“PCB footprints/Resistors/Resistor 1/4 W 0,4 i”), e il condensatore elettrolitico (“PCB footprints/Electrolytic capacitor/Vert. diam. 5 mm 2.5 mm pitch”). Per delimitare l'area della scheda può essere utile tracciare un rettangolo vuoto sul livello silk-screen (layer No. 3). Per fare ciò è possibile utilizzare la primitiva *rectangle*, assicurandosi di aver selezionato precedentemente il livello appropriato. Il risultato ottenuto dovrebbe essere simile a quello mostrato in figura 2.12.

Possiamo ora passare a disegnare le aree di rame che fungeranno da alimentazione positiva e negativa. Esse possono essere tracciate disegnando un poligono (usando la primitiva *poly line*). Facendo doppio click sul bordo del poligono, in modalità

## 2.5. un semplice PCB

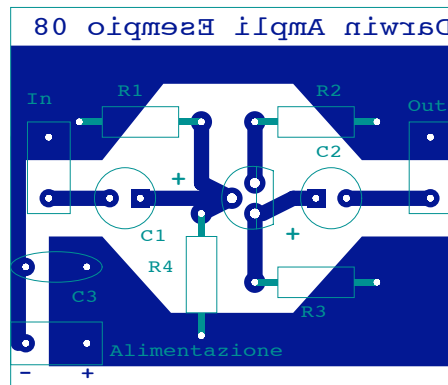


Figura 2.11.: Un semplice amplificatore che fa uso di un transistor NPN connesso in configurazione ad emettitore comune.

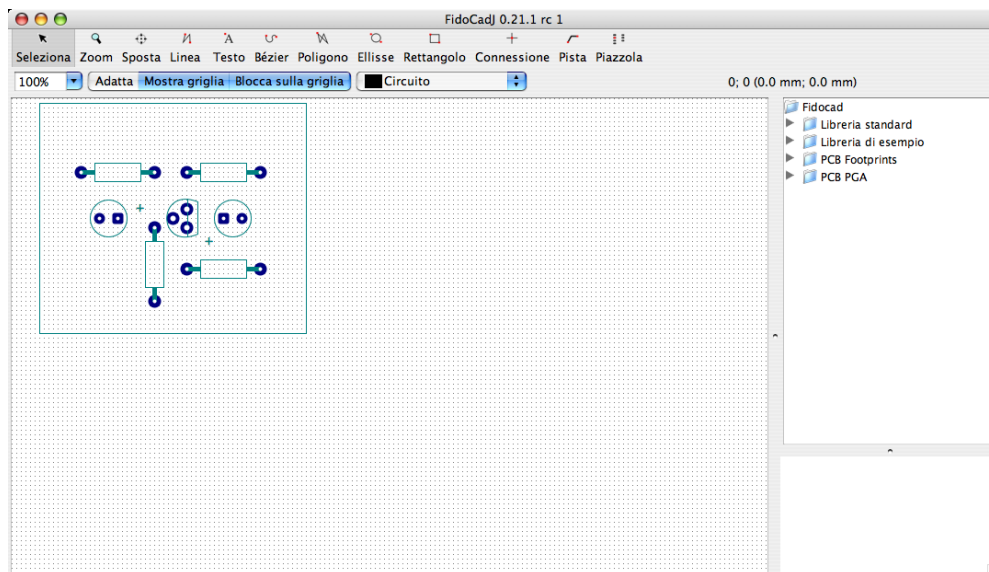


Figura 2.12.: I dispositivi più importanti sono piazzati sulla scheda.

## 2. Disegnare con FidoCadJ

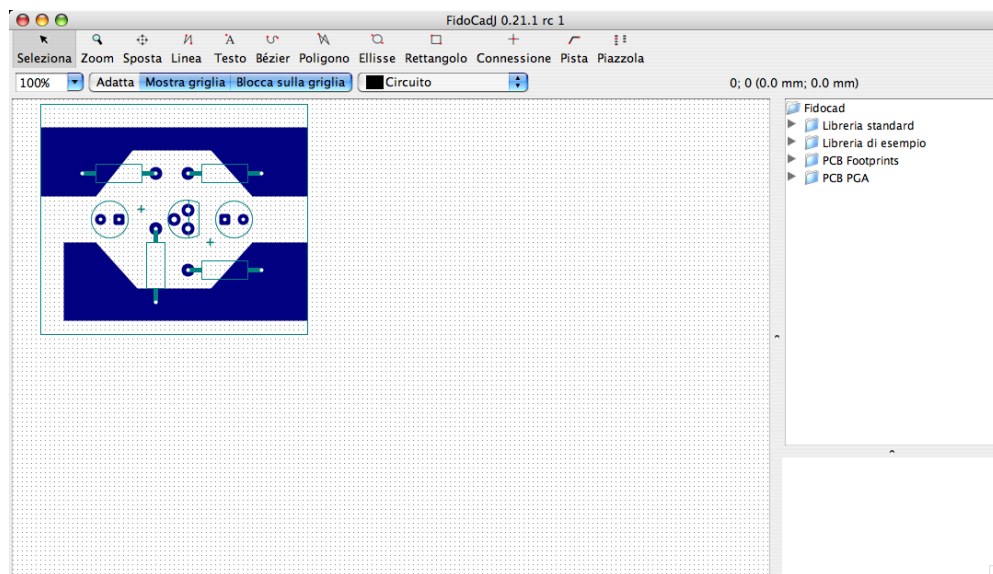


Figura 2.13.: Sono state aggiunte le linee di alimentazione tramite una linea poligonale.

selezione è possibile specificare a FidoCadJ (attraverso la finestra di dialogo) che il poligono voluto deve essere riempito. Prima di piazzare il poligono assicurarsi che il livello corrente sia quello dove intendiamo piazzare l'area di rame (livello N 1, rame lato saldature).

L'uso di un piano per connettere le linee di alimentazione è utile per assicurarsi che le connessioni abbiano una bassa induttanza parassita. Il risultato fin qui ottenuto dovrebbe quindi essere simile a quello riportato in figura 2.13.

Per completare le connessioni possiamo utilizzare la primitiva *PCB line*. Io ho scelto di tracciare una pista di spessore pari a 10 unità (1.27 mm), che sarà utile durante il processo di saldatura.

E' chiaro che la scheda debba avere a bordo qualche connettore: uno per l'ingresso, uno per l'uscita e uno per l'alimentazione. E' possibile utilizzare, a tale scopo, il footprint di un condensatore al poliestere, che avrà probabilmente le dimensioni giuste. Non dobbiamo dimenticare, infatti, che FidoCadJ è pensato per essere il successore dei trasferibili R41...

Possiamo anche piazzare un "+" e un "-" sul lato rame e piazzare un condensatore ceramico in parallelo all'alimentazione. Mettiamo ora le scritte anche sulla scheda stessa. Per scrivere sul lato rame, dopo avere selezionato il livello giusto, è necessario specchiare tutto ciò che scriveremo. Questo può essere realizzato facilmente tramite le ormai note proprietà accessibili facendo doppio click sulla scritta da modificare mentre si è in selection mode. Servirà fare qualche prova prima di arrivare ad ottenere le dimensioni giuste dei caratteri. Per avere qualche riferimento si può pensare di avere

*Fate attenzione con la larghezza delle piste: quello che appare grande come un'autostrada sullo schermo potrebbe essere una traccia così fine che si staccherà dalla scheda durante il processo di saldatura.*

## 2.5. un semplice PCB

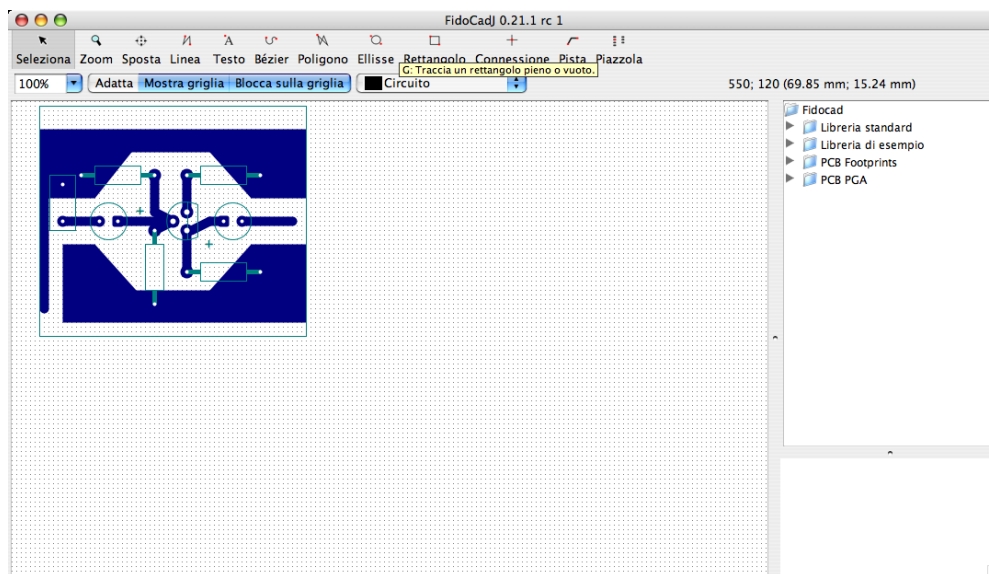


Figura 2.14.: Sono state aggiunte le connessioni rimanenti.

un rapporto di 3/4 tra la dimensione orizzontale e quella verticale dei caratteri. La figura 2.15 mostra il risultato ottenuto usando 11 unità per la dimensione orizzontale e 18 unità per quella verticale del testo.

A questo punto l'unica cosa mancante è il testo che riporta il nome di ogni componente, che può essere piazzato sul livello 3 (silk-screen). Il PCB completo viene mostrato in figura 2.16.

Una volta che il PCB sarà stato completato, si vorrà probabilmente stamparlo su un foglio acetato per utilizzarlo come maschera su un bromografo oppure per utilizzarlo con altri metodi, come per esempio il “Press&Peel”. Per fare ciò è necessario rendere invisibili tutti i livelli che non si intendono stampare. Fare ciò è possibile tramite la finestra di dialogo accessibile dal menu “Vista/Layer”. Nel nostro caso è sufficiente nascondere il livello 3 contenente il silk-screen. Il programma mostrerà quindi solo il lato rame. Stamperemo quindi tutto ciò che viene mostrato sullo schermo (ovviamente NON bisogna adattare la stampa alla dimensione del foglio, poichè siamo intenzionati a stampare il disegno con le sue dimensioni originali), avendo cura di impostare la stampa in bianco e nero così da assicurare il contrasto massimo. Potrebbe essere utile specchiare l'intero disegno, dipendentemente dalla tecnica utilizzata per la produzione del PCB. A causa del fatto che il PCB da noi realizzato è piuttosto piccolo, le sue dimensioni saranno tali da occupare solo un piccolo angolo del foglio (assumendo di utilizzare un foglio standard ISO-UNI A4), come possiamo vedere in Figura 2.17.

Per completezza, viene qui riportato il codice che genera il PCB dell'esempio (fate attenzione alle linee lunghe!):

```
[FidoCAD]
```

## 2. Disegnare con FidoCadJ

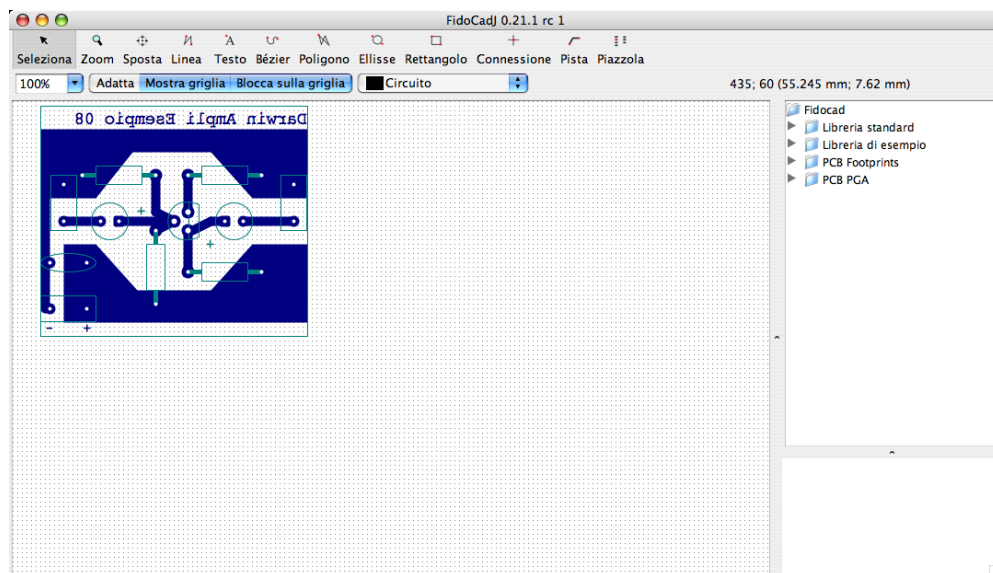


Figura 2.15.: Il PCB quasi completato.

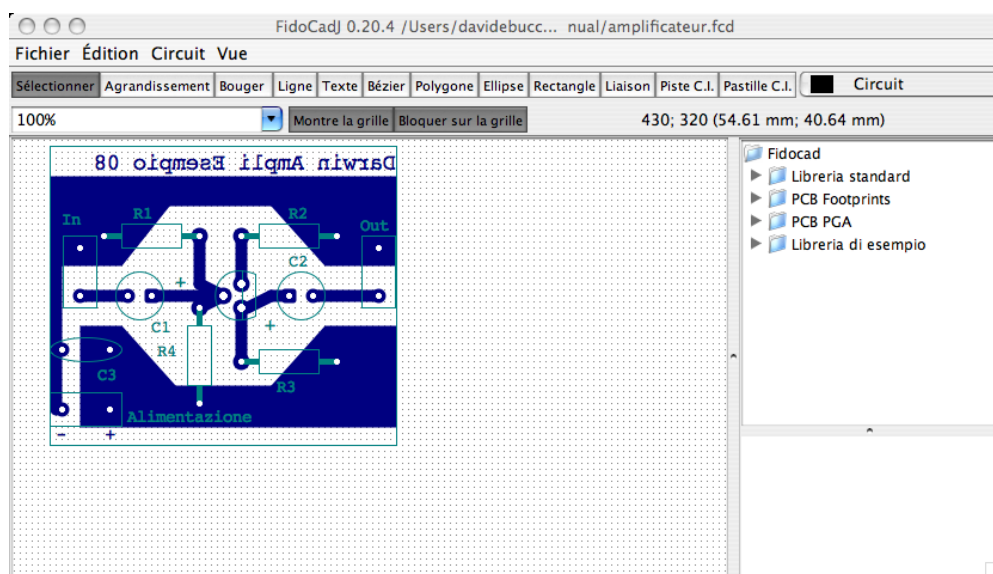


Figura 2.16.: Il lavoro finito con il silk-screen.

## 2.5. un semplice PCB

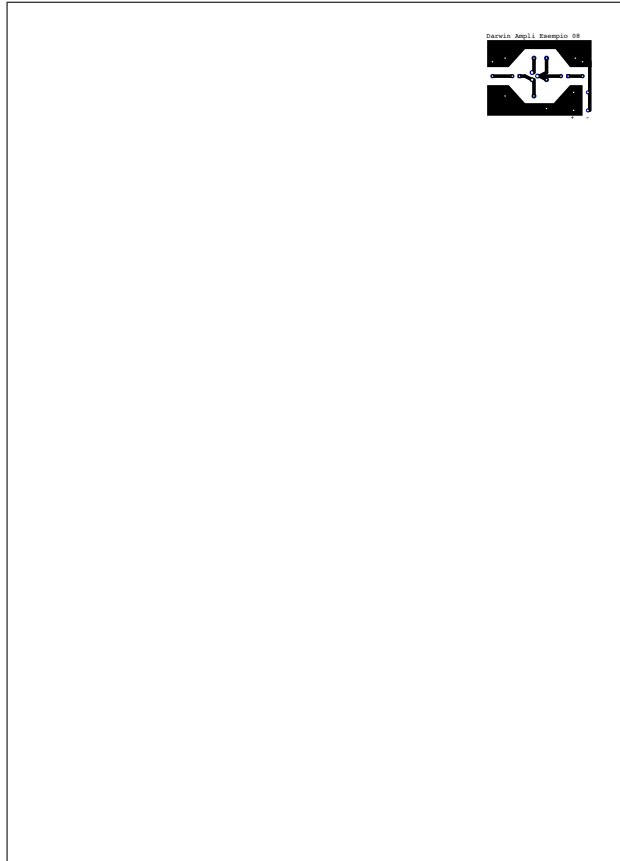


Figura 2.17.: Il PCB, come appare una volta stampato (specchiato) su un foglio ISO-UNI A4.

## 2. Disegnare con FidoCadJ

```
TY 320 10 18 11 0 4 1 * Darwin Ampli Esempio 08
TY 85 240 12 8 0 5 1 * +
TY 44 239 12 8 0 5 1 * -
PL 35 90 35 225 10 1
PL 55 130 95 130 10 1
PL 250 130 305 130 10 1
PL 215 130 230 130 10 1
PL 195 140 215 130 10 1
PL 115 130 175 130 10 1
MC 155 220 3 0 PCB.R01
MC 75 80 0 0 PCB.R01
MC 270 185 2 0 PCB.R01
MC 270 80 2 0 PCB.R01
MC 230 130 3 0 PCB.CE00
MC 115 130 1 0 PCB.CE00
MC 40 175 0 0 PCB.CC50
PL 190 80 190 120 10 1
PL 190 140 190 185 10 1
PL 155 80 155 120 10 1
PL 155 120 175 130 10 1
PL 155 140 175 130 10 1
PP 30 30 30 105 90 105 130 55 215 55 260 105 320 105 320 30 1
PP 320 240 320 155 260 155 215 205 135 205 90 155 55 155 55 240 1
MC 190 120 0 0 PCB.T092
MC 305 90 1 0 PCB.CPBX352
MC 55 90 1 0 PCB.CPBX352
MC 80 225 2 0 PCB.CPBX352
TY 290 65 12 8 0 0 3 * Out
TY 40 60 12 8 0 0 3 * In
TY 95 225 12 8 0 0 3 * Alimentazione
TY 70 190 12 8 0 0 3 * C3
TY 230 95 12 8 0 0 3 * C2
TY 115 150 12 8 0 0 3 * C1
TY 120 170 12 8 0 0 3 * R4
TY 220 200 12 8 0 0 3 * R3
TY 230 55 12 8 0 0 3 * R2
TY 100 55 12 8 0 0 3 * R1
RV 30 5 320 255 3
```

## 2.6. Usare il righello

Quando si disegna un PCB, è spesso utile misurare delle distanze sull'area di lavoro. Per esempio è talvolta necessario controllare lo spessore di una posta, la distanza tra due piste o la dimensione totale della scheda. FidoCadJ offre (dalla versione 0.23.2) un righello che permette di fare ciò in modo semplice: è sufficiente fare click con il tasto destro e trascinare. Si ottiene un righello verde, come quello mostrato in figura 2.18.



## 2.7. Frecce e stili di tratteggio

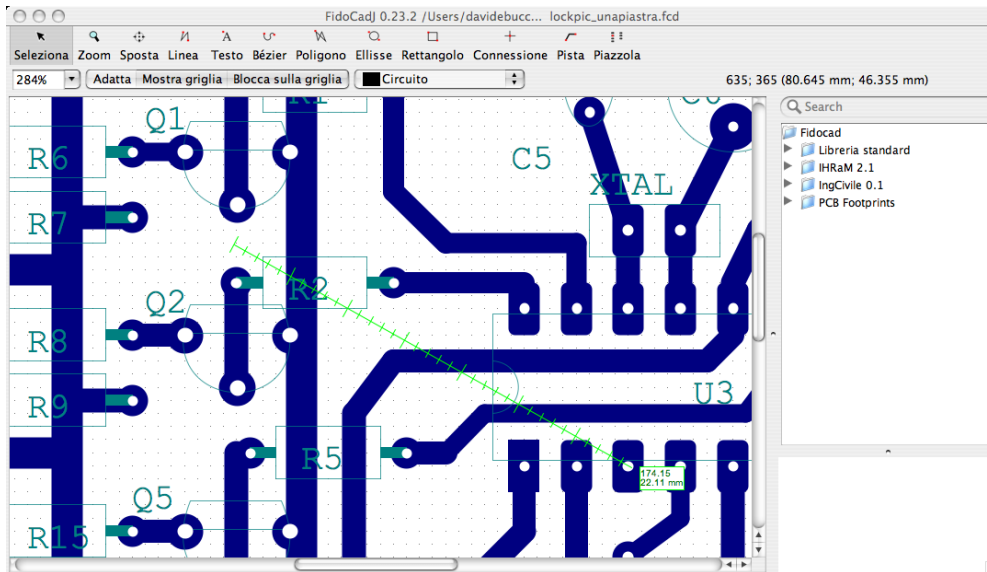


Figura 2.18.: Fare click col tasto destro per attivare il righello FidoCadJ.

Se il tasto destro e il trascinamento non sono utilizzati nel sistema operativo in uso è possibile fare click con il tasto sinistro e trascinare mentre si preme il tasto **[Shift]**.

La lunghezza misurata è mostrata in unità logiche FidoCadJ e anche in millimetri. Questo è utile quando il disegno è stampato 1:1 (come anche i PCB).

## 2.7. Frecce e stili di tratteggio

FidoCadJ permette di disegnare delle frecce all’inizio e alla fine delle linee, curve Bézier e spline cubiche. Permette anche di specificare dove disegnare le frecce, se all’inizio o alla fine dell’elemento (o su entrambi i lati) e offre anche diversi stili di tratteggio.

Qualche stile di tratteggio è reso disponibile per il disegno tecnico o meccanico. La figura 2.19 mostra un esempio nel quale un circuito elettronico (un GIC) è stato rinchiuso in un rettangolo tratteggiato. E’ stata utilizzata anche una freccia alla fine di una curva Bézier. Facendo un doppio click su questo elemento, FidoCadJ mostra una finestra di dialogo che è simile a quella mostrata in figura 2.20. E’ possibile notare che è stato selezionata l’opzione “Arrow at start”. FidoCadJ disegnerà quindi una freccia avendo cura di impostarne la corretta orientazione. Sono disponibili diversi stili di disegno per le frecce come anche per i tratteggi. Qualche prova permetterà al lettore di apprezzarne le differenze.

La possibilità di inserire linee tratteggiate e frecce non era contemplata nel formato originale FidoCAD. Questo sfortunatamente significa che i disegni FidoCadJ che usano queste funzionalità non sono completamente retrocompatibili con FidoCAD per Win-

## 2. Disegnare con FidoCadJ

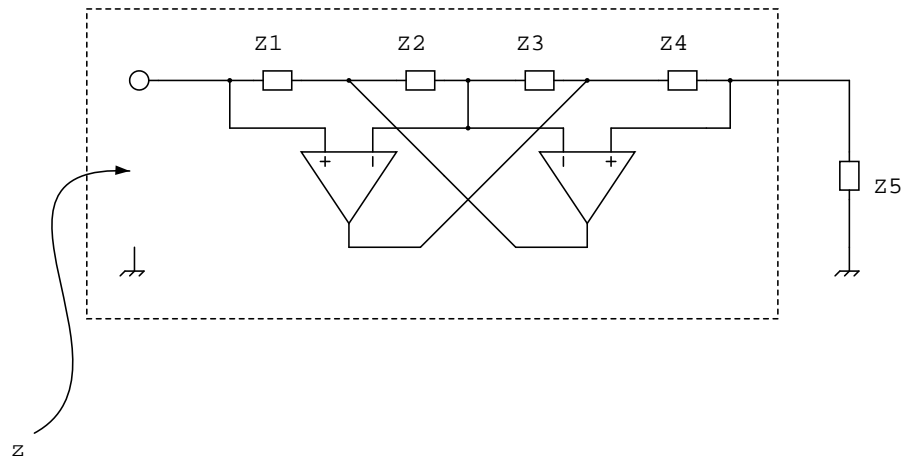


Figura 2.19.: Uno schema elettrico (un GIC di Antoniou) nel quale sono state utilizzate alcune estensioni di FidoCadJ.

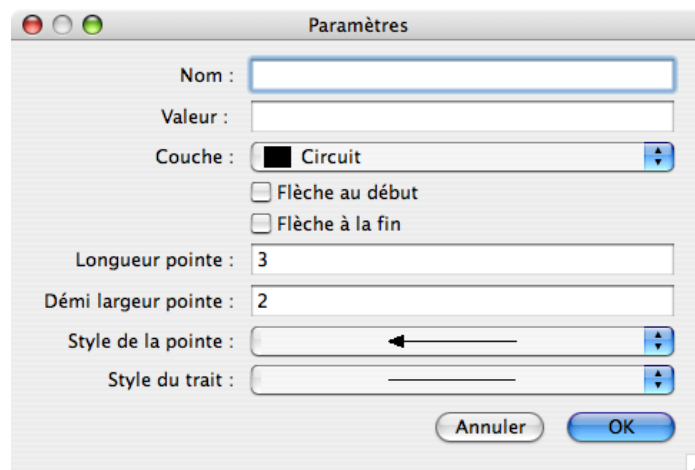


Figura 2.20.: Finestra di dialogo per la curva Bézier utilizzata per lo schema elettrico di figura 2.19 (In francese).

dows. Quando si opera la scelta di utilizzo di una estensione FidoCadJ è necessario conoscere che cosa si stia facendo. Se la compatibilità FidoCAD è assolutamente necessaria, è possibile selezionare l'opzione "Strict FidoCAD compatibility" in "FidoCadJ extensions" della finestra "FidoCadJ preferences". In questo modo il programma non permetterà di introdurre elementi grafici che interromperebbero la compatibilità con FidoCAD. Per ulteriori dettagli riguardanti la compatibilità dei nuovi elementi grafici con FidoCAD, riferirsi alla sezione ??.

## 2.8. Esportazione

Per me una delle caratteristiche più importanti di FidoCadJ è la possibilità di creare semplici schemi elettrici per uso tipografico. Per questa ragione ho introdotto la possibilità di esportare i disegni in diversi formati di file.

Per esportare il disegno corrente, selezionare il comando "Export" dal menu "File". La tabella 2.2 mostra la lista di formati grafici disponibili. Per ogni formato di file, la tabella (ma anche la finestra di dialogo in FidoCadJ) specifica se il formato è vettoriale o bitmap. Ovunque sia possibile è meglio scegliere il formato vettoriale per ottenere il miglior risultato <sup>4</sup>.

Per il formato di file bitmap può essere utile abilitare l'opzione "Anti aliasing", per ridurre l'effetto irritante della quantizzazione, visibile specialmente sulle linee diagonali. L'opzione "Anti aliasing" non è usata quando si esporta un formato di file vettoriale. In questo caso è possibile specificare un fattore di scala.

L'opzione "Black&White" permette di stampare ogni livello visibile in nero. Questo è importante per la preparazione delle maschere a scopo tipografico o per i bromografi.

*Un formato vettoriale memorizza gli elementi che compongono il grafico. Un formato bitmap memorizza una matrice di punti.*

## 2.9. Opzioni da linea di comando

L'applicazione viene distribuita tramite un file `.jar`, che è un archivio Java.<sup>5</sup> In molti sistemi operativi, per far girare l'applicazione è sufficiente fare doppio click sopra il file, a patto che una versione recente di Java sia installata sulla macchina. Usando la terminologia Sun, il così chiamato JRE, o anche Java Runtime Environment, è tutto ciò che serve per far girare un programma scritto in Java (ma non per scriverlo: in questo caso è necessario il SDK...). La versione minima di Java per far girare FidoCadJ è la 1.5, la quale è stata pubblicata da ormai molti anni.

In alcuni casi, potrebbe essere utile per far girare FidoCadJ da linea di comando (il terminale su un sistema Unix, o sul MS-DOS Prompt in Windows). Per fare ciò, è sufficiente lanciare il comando `java`, con l'option `-jar`:

```
java -jar FidoCadJ.jar
```

<sup>4</sup>Il codice di FidoCadJ è strutturato in modo da permettere l'aggiunta di un nuovo formato di file in modo piuttosto semplice. Contattatemi se intendete partecipare al progetto.

<sup>5</sup>Ad eccezione per la versione Macintosh questa è una applicazione stand alone.

## 2. Disegnare con FidoCadJ

Formato	Commento
JPG	Formato bitmap davvero diffuso. Dato che la compressione utilizzata è con perdite, non è adatto all'esportazione di un file FidoCaJ. E' reso disponibile a causa della sua diffusione.
PNG	Formato bitmap compresso , adatto all'esportazione di schemi elettrici. E' probabilmente il miglior formato col quale esportare un disegno FidoCadJ quando il formato vettoriale non può essere utilizzato.
SVG	Formato vettoriale W3C standard. Qualche browser (come le più recenti versioni di Safari) permettono di visualizzarlo in una pagina web. E' un formato molto valido per grafici e schemi elettrici, permette la visualizzazione e la modifica di disegni fatti con FidoCadJ in applicazioni quali Inkscape. Attualmente c'è qualche limite per l'esportazione di testi ruotati e specchiati.
EPS	Encapsulated Postscript vectorial format. Molto utile per coloro che usano applicazioni grafiche professionali, o vogliono utilizzare FidoCadJ con $\text{\LaTeX}$ . L'esportazione grafica dovrebbe essere completamente funzionante. Questo è stato il modo con il quale si sono ottenute le figure <a href="#">2.11</a> , pagina <a href="#">17</a> in questo manuale (passando attraverso la conversione PDF, poichè io uso $\text{\LaTeX}$ ).
PGF	Vectorial format da usare direttamente in un documento $\text{\LaTeX}$ , quando si usa il pacchetto <i>pgf</i> disponibile nell'archivio CTAN. L'opzione di esportazione è pensata in particolare per esportare schemi elettrici e usa uno script facilmente modificabile. Gli attributi testuali non saranno tradotti. Questo permette l'introduzione di codice $\text{\LaTeX}$ direttamente nel disegno ed è la tecnica utilizzata in questo manuale per ottenere la figura <a href="#">2.5</a> , pagina <a href="#">10</a> .
SCR	FidoCadJ permette l'esportazione di uno script che può essere importato in CadSoft Eagle. Per usare questa caratteristica, è necessario installare la libreria <code>FidoCadJLIB.1br</code> nella cartella <code>1br</code> della installazione corrente di Eagle. La libreria può essere scaricata dal sito web di FidoCadJ. Quando è stato scritto questo manuale, questa opzione funziona solo con gli schemi elettrici contenenti solo i simboli più comuni. Qualche elemento grafico, quali i pad e le piste non sono ancora disponibili. Quest'ultime non verranno esportate nello script Eagle.

Tabella 2.2.: Lista di tutti i formati esportabili disponibili in FidoCadJ.

## 2.9. Opzioni da linea di comando

Se viene specificato un file a seguito della linea di comando, , FidoCadJ cercherà di aprirlo. Per esempio (su una macchina Unix):

```
java -jar FidoCadJ.jar ~/FidoCadJ/test.fcd
```

FidoCadJ verrà lanciato cercando di aprire il file `~/FidoCadJ/test.fcd` (ammesso che esista).

Ci sono alcune cose interessanti che FidoCadJ può fare.

L'opzione `-h` mostra una lista delle opzioni FidoCadJ:

```
[davidebucci@Darwin]$ java -jar FidoCadJ.jar -h

This is FidoCadJ, version 0.24.1.
By Davide Bucci, 2007-2012.

Use: java -jar FidoCadJ.jar [-options] [file]
where options include:

-n      Do not start the graphical user interface (headless mode)

-d      Set the extern library directory
Usage: -d dir
where 'dir' is the path of the directory you want to use.

-c      Convert the given file to a graphical format.
Usage: -c sx sy eps|pdf|svg|png|jpg|fcd|sch outfile
If you use this command line option, you *must* specify a FidoCadJ
file to convert.
An alternative is to specify the resolution in pixels per logical unit
by preceding it by the letter 'r' (without spaces), instead of giving
sx and sy.

-s      Print the size of the specified file in logical coordinates.

-h      Print this help and exit.

-t      Print the time used by FidoCadJ for the specified operation.

-p      Do not activate some platform-dependent optimizations. You might try
this option if FidoCadJ hangs or is painfully slow.

-l      Force FidoCadJ to use a certain locale (the code might follow
immediately or be separated by an optional space).

[file] The optional (except if you use the -d or -s options) FidoCadJ file to
load at startup time.

Example: load and convert a FidoCadJ drawing to a 800x600 pixel png file
without using the GUI.
java -jar FidoCadJ.jar -n -c 800 600 png out1.png test1.fcd

Example: load and convert a FidoCadJ drawing to a png file without using the
graphic user interface (the so called headless mode).
Each FidoCadJ logical unit will be converted in 2 pixels on the image.
java -jar FidoCadJ.jar -n -c r2 png out2.png test2.fcd

Example: load FidoCadJ forcing the locale to simplified chinese (zh).
java -jar FidoCadJ.jar -l zh

[davidebucci@Darwin]$
```

L'opzione più semplice è `-n`, con la quale il programma... non fa nulla, non attiva la GUI ed esce. In questo caso la variabile Java di ambiente `java.awt.headless` viene

## 2. Disegnare con FidoCadJ

impostata al valore 'vero'. Ovviamente questa opzione è pressochè inutile da sola, ma potrebbe essere preziosa in combinazione con altre funzioni che andremo a descrivere. L'opzione `-d` permette di specificare il percorso dove FidoCadJ cercherà per le librerie da caricare all'avviamento. L'opzione `-c` converte un file FidoCAD (che deve essere specificato) in una immagine vettoriale o raster. Questo può essere molto utile quando FidoCadJ viene utilizzato quale convertitore non interattivo (insieme all'opzione `-n`) Prendiamo in considerazione il primo esempio mostrato nell'help:

```
java -jar FidoCadJ.jar -n -c 800 600 png out1.png test1.fcd
```

FidoCadJ verrà avviato senza lanciare la GUI e esporterà in formato png il file `test1.fcd`. Il file di output verrà chiamato `out1.png` avrà una dimensione di 800x600 pixel.

C'è una versione alternativa dell'opzione `-c`, la quale permette di specificare quanti pixel dovranno essere utilizzati per la conversione di una unità logica (chiamiamo questo fattore  $r_p$ ). FidoCadJ non considera le mezze unità logiche (i numeri devono essere interi). Scegliendo, diciamo,  $r_p$  uguale a due pixel per unità logica ci si assicurerà che lo schema elettrico sia comprensibile (anche se probabilmente un po' piccolo). il fattore  $r_p$  potrebbe non essere intero, come in questo esempio:

```
java -jar FidoCadJ.jar -n -c r1.25 png out2.png test2.fcd
```

per sapere la dimensione complessiva (in unità logiche) di uno schema elettrico, è possibile utilizzare l'opzione `-s`. Ci si ricordi, però, che durante una esportazione, FidoCadJ somma sempre un margine di  $t_{\text{margin}} = 3$  unità logiche per ogni lato del disegno. In questo modo, se  $t_w$  è la larghezza del disegno in unità logiche date da `-s`, ci si potrebbe aspettare che la larghezza del disegno in pixel  $p_w$  sia calcolata nel modo seguente

$$p_w = r_p(t_w + 2t_{\text{margin}}) \quad (2.1)$$

Un'altra opzione interessante, dovuta più a Java che a FidoCadJ, è la possibilità di modificare l'aspetto dell'applicazione (chiamata in gergo Java look & feel). E' possibile scegliere il look&feel preferito senza modificare una singola linea di codice. Riporto una cosa che gli utenti Linux apprezzeranno, il look GTK+:

```
java -Dswing.defaultlaf=com.sun.java.swing.plaf.gtk.
GTKLookAndFeel -jar FidoCadJ.jar
```

E' anche disponibile il più classico Motif look & feel, mostrato in figura 2.21<sup>6</sup>:

```
java -Dswing.defaultlaf=com.sun.java.swing.plaf.motif.
MotifLookAndFeel -jar FidoCadJ.jar
```

---

6

Questo stile potrebbe lasciare di stucco coloro che sono abituati a stili grafici rifiniti come Aqua su MacOSX. Ho visto qualche anno fa un sistema di controllo per un sincrotrone con una interfaccia grafica basata su Motif. Niente male!

## 2.10. Gestione delle librerie

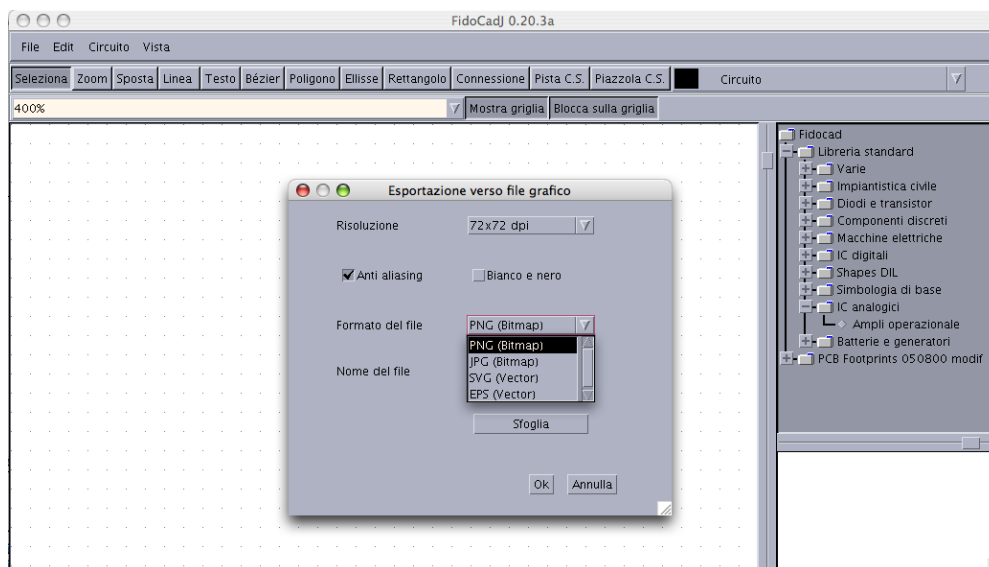


Figura 2.21.: Come appare il programma su MacOSX, usando il look & feel Motif.

Ovviamente, i comandi descritti devono essere inseriti da terminale, facendo attenzione che la directory corrente contenga il file `FidoCAD.jar`. Bisogna fare inoltre attenzione a scrivere tutto sulla stessa linea di comando.

## 2.10. Gestione delle librerie

### 2.10.1. Usare i file di libreria

Una libreria è una raccolta di simboli i quali possono essere inclusi in un disegno. C'è un certo numero di librerie incluse nel pacchetto FidoCadJ, come anche la possibilità di crearne di nuove, simboli inclusi. FidoCadJ permette di specificare una cartella nella quale sono presenti tutti i file di libreria (con l'estensione `.fcl`), tramite il menu "File/Options".

In qualche caso speciale (o per fare test), le librerie incluse in FidoCadJ possono essere sostituite con altre esterne. Se è presente un file di nome `FCDstdlib.fcl` il suo contenuto prenderà il posto della libreria standard e verrà reso disponibile nell'applicazione. Analogamente, se è presente un file di nome `PCB.fcl` il suo contenuto prenderà il posto della libreria per i PCB.<sup>7</sup>

Dalla versione 0.23, grazie a Roby IZ1CYN, ho potuto includere la libreria IHRaM 3.1 direttamente all'interno della distribuzione FidoCadJ. L'ho fatto perchè questa, tra tutte le librerie che ho avuto modo di vedere, mi è sembrata una delle più complete

<sup>7</sup>Si prega di fare attenzione all'uso delle lettere maiuscole, in particolare se il sistema operativo utilizzato distingue le lettere maiuscole dalle minuscole nella gestione dei file

## 2. Disegnare con FidoCadJ

e razionalmente organizzate. Come per tutte le altre librerie FidoCadJ, se è presente un file chiamato `IHRAM.FCL` nel percorso di ricerca delle librerie, esso verrà caricato al posto della versione presente all'interno del programma. Sono presenti anche i simboli per elettrotecnica, il cui file di libreria è chiamato `elettrotecnica.fcl`.

Altri file con l'estensione `.fcl`, nel path delle librerie, saranno considerati come librerie e FidoCadJ cercherà di aprirle una volta avviato. Queste sostituzioni e aggiunte vengono effettuate quando l'applicazione parte, quando l'utente cambia il percorso di ricerca delle librerie oppure quando l'opzione "Update libraries" nel "Circuit menu" viene selezionata.

FidoCadJ permette di separare le macro non standard (come fa anche FidoCAD originale). Questo è molto utile quando si posta un disegno in un newsgroup, poichè ogni macro che non appartiene alla libreria standard di FidoCAD viene espansa nelle sue primitive grafiche. Chi leggerà il post, quindi, non avrà la necessità di possedere la stessa libreria di chi ha postato il disegno.

Per effettuare il copia-incolla, nel menu "Edit" è disponibile il comando "Copy, split non standard macros" (o si può usare la scorciatoia `[Control]+[M]`<sup>8</sup>), in modo da separare tutte le macro appartenenti a librerie non standard. E' anche possibile salvare un lavoro separando le macro non standard, scegliendo "Save as..., split non standard macros" dal menu "File". Comparirà una finestra di dialogo che permetterà di scegliere un nuovo nome file, in modo da non sovrascrivere quello sul quale si sta lavorando.

### 2.10.2. Definire nuovi simboli

FidoCadJ permette di creare nuove librerie e nuovi simboli. La tecnica adottata è stata scelta in modo da essere la più intuitiva possibile. Ecco i passi da seguire:

- Assicurarsi che sia stata definita una cartella contenente le librerie. Se non è così, definirla usando le impostazioni di FidoCadJ.
- Disegnare quanto si intende convertire in un simbolo.
- Selezionare ciò che si è disegnato e cliccarci sopra con il tasto destro. Apparirà un menu popup, come in figura 2.22.
- Selezionare "Symbol-o-matic" e apparirà una finestra di dialogo che permetterà di definire i dettagli del simbolo appena creato (vedere figura 2.23).
- Nel primo campo "Library filename" si trova il nome della libreria nella quale sarà incluso il nuovo simbolo. Digitando qui un nuovo nome, FidoCadJ creerà un nuovo file.
- Il secondo campo "Library complete name" è il nome completo della libreria, quello mostrato nell'albero delle librerie. E' possibile scrivere una cosa qualunque, qui, ma è chiaramente preferibile scegliere un nome corto ed esplicativo.

---

<sup>8</sup>Sui sistemi MacOSX, la scorciatoia è `[Command]+[M]`



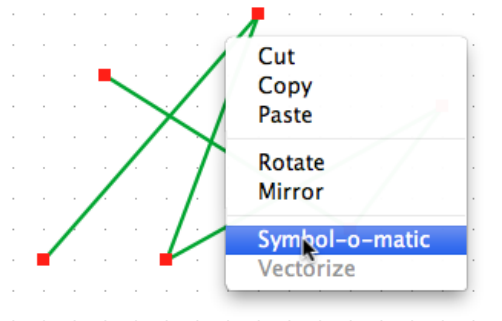


Figura 2.22.: Il menu popup che appare cliccando con il tasto destro permette di trasformare un disegno in un simbolo

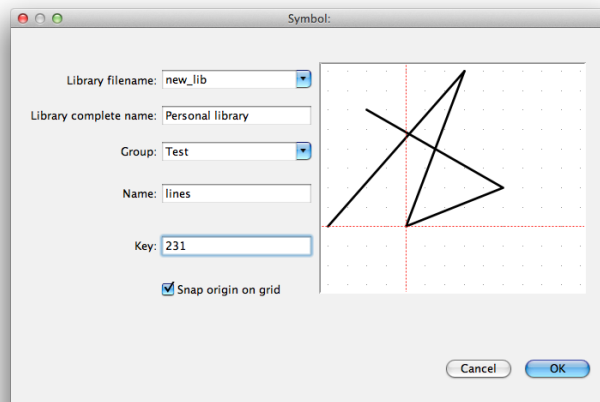


Figura 2.23.: La finestra di dialogo per il nuovo simbolo, tramite la quale è possibile impostare tutte le caratteristiche del simbolo. Si noti che l'origine è definita dai due assi rossi.

## 2. Disegnare con FidoCadJ

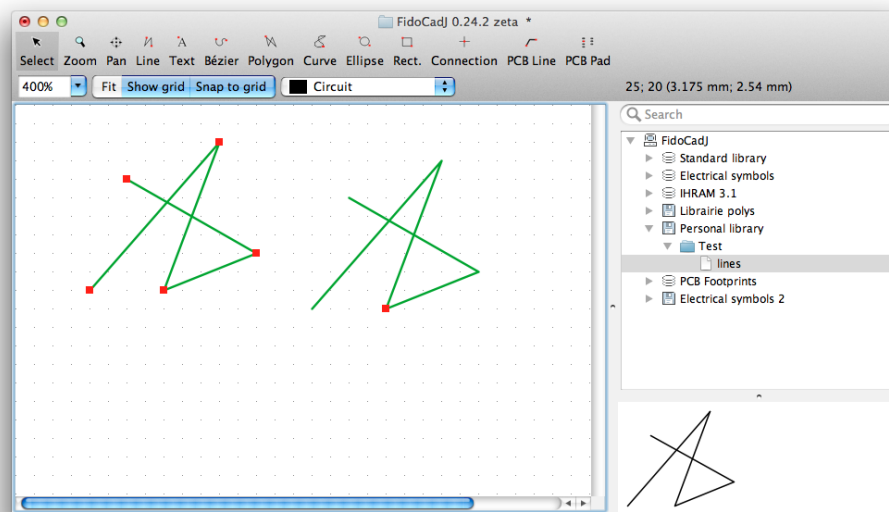


Figura 2.24.: Il simbolo appena creato, mostrato nella lista dei simboli e nel disegno. Sulla sinistra è ancora presente il disegno utilizzato per creare il simbolo. Notare che il punto di controllo è presente nel disegno, per il nuovo simbolo.

- Il terzo campo “Group” permette di scegliere in quale gruppo inserire il nuovo simbolo, dentro la libreria. Nuovamente, digitando un nuovo nome, verrà creato un nuovo gruppo.
- Il quarto campo “Name” è il nome del simbolo: quello mostrato nell’albero delle librerie. Convienne anche in questo caso scegliere un nome corto ed esplicativo.
- Il quarto campo “Key” è una etichetta piuttosto corta che identifica il simbolo nel codice. Dovrebbe essere unico all’interno di ogni libreria e non dovrebbe contenere spazi come anche caratteri quali: parentesi, puntini eccetera. FidoCadJ propone un codice numerico, ma sarebbe meglio una etichetta mnemonica.
- Cliccando nel campo a destra, si deve scegliere l’origine del simbolo, cioè il punto che verrà utilizzato per piazzare il simbolo all’interno del disegno. E’ possibile scegliere di posizionare il simbolo sulla griglia cliccando su “Snap origin on grid”.
- Quando si clicca su “OK”, il simbolo appena creato dovrebbe comparire nella libreria ad albero nella finestra principale di FidoCadJ.

Un esempio di un simbolo creato da un utente e usato in un disegno viene mostrato in figura 2.24. Notare la differenza tra la parte selezionata a sinistra nel disegno (in

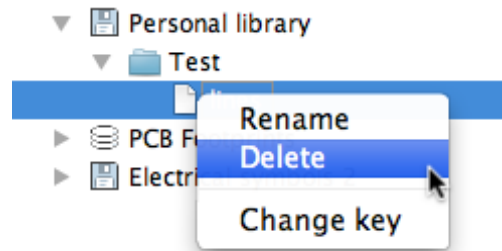


Figura 2.25.: Il menu popup utilizzato per modificare le proprietà dei simboli nella libreria utente.

realtà ciò che è stato usato per definire il nuovo simbolo) e il nuovo simbolo piazzato sulla destra. In effetti, un solo punto di controllo è disponibile per il piazzamento del simbolo e questo punto di controllo corrisponde alla scelta del punto di origine mentre viene definito il simbolo. In questo modo si sceglie un punto particolarmente importante all'interno del simbolo. E' possibile separare un simbolo nelle sue primitive facendo click su "Vectorize".

### 2.10.3. Modificare un simbolo esistente

E' disponibile usufruire di alcune voci di menu nell'albero della librerie, compreso il drag and drop. Selezionando un simbolo in una libreria utente e cliccandoci sopra con il tasto destro apparirà un menu popup, come mostrato in figura 2.25. Ci si troveranno le opzioni seguenti:

- "Rename" rinomina il simbolo.
- "Delete" elimina il simbolo. Si prega di prestare attenzione: se un disegno contiene il simbolo eliminato esso non sarà più riconosciuto e scomparirà.
- "Change key" è utile per modificare la proprietà key associata al simbolo. Anche qui si prega di prestare la dovuta attenzione: questa opzione è da utilizzarsi quando il simbolo non è stato utilizzato nel disegno.

Usando il drag and drop è possibile cambiare la libreria o la categoria entro la quale il simbolo viene classificato. Si prega di prestare attenzione al fatto che un simbolo viene completamente identificato tramite il nome di libreria seguito dalla sua key. A causa di questo lo spostamento di un simbolo da una libreria ad un'altra renderà non più valido il disegno che lo contiene. Le operazioni sulle librerie non sono annullabili, esattamente come le operazioni sul disegno. Librerie o gruppi vuoti non vengono mostrati.

## 3. Formato di disegno, macro e librerie FidoCAD

Questo capitolo contiene una descrizione dettagliata del formato usato da FidoCAD e, di conseguenza, del formato usato da FidoCadJ per salvare un disegno. Il formato è testuale, ma con il vantaggio di essere compatto ed efficiente. Cercherò di spiegare ciò che ho imparato su questo formato, poichè esso non è descritto in dettaglio in nessun documento. Cercherò di spiegare anche le estensioni che ho aggiunto a FidoCadJ, inoltre. Si ricordi che dalla versione 0.23.4, FidoCadJ usa solo la codifica UTF-8 su tutte le piattaforme.

### 3.1. Descrizione dell'intestazione

Tutti i file contenenti un disegno in formato FidoCAD iniziano con l'etichetta *[FidoCAD]*. Un programma può quindi riconoscere la presenza di comandi FidoCAD leggendo questa etichetta. A questo proposito, FidoCadJ è più tollerante rispetto a FidoCAD originale perchè riconosce e interpreta un file anche se non contiene l'intestazione originale. Anche se i comandi sono testuali, essi possono essere correttamente interpretati fino a quando il numero delle linee di codice errato non supera un valore predefinito dal programma (circa 100). Questo fa sì che FidoCadJ non si blocchi per troppo tempo, per esempio cercando di interpretare un grosso file binario.

### 3.2. Sistema di coordinate

FidoCadJ funziona utilizzando un semplice sistema di coordinate. In pratica ha a disposizione un'area grande identificata unicamente da coordinate positive. La lunghezza di ogni unità in x e y è fissata a  $127\mu\text{m}$ , un valore che permette di ottenere una buona risoluzione anche per il più piccolo componente SMD, senza essere troppo fine per l'uso normale. In termini tipografici, la risoluzione di FidoCadJ è di 200 punti per pollice.

Il FidoCAD originale opera in due modi differenti: PCB e schema elettrico. In FidoCadJ questa differenza è più evanescente e si manifesta unicamente nel momento della stampa del disegno. Sarebbe, a tal proposito, consigliabile di impostare il programma in modo che ridimensioni lo schema elettrico in modo da utilizzare al meglio il foglio, altrimenti è possibile che il risultato in stampa abbia le dimensioni di un francobollo.

### 3.3. Elementi di disegno

FidoCadJ può trattare 12 elementi grafici, come segue

- Line
- Filled o empty rectangle
- Simple text (obsolete)
- Advanced text
- Filled o empty polyline
- Filled o empty ellipse
- Bézier curve
- Natural cubic spline curve
- Electrical junction
- PCB pad
- PCB track
- Macro

Analizzeremo ognuno di loro. In generale, ogni elemento è identificato da un comando e un certo numero di parametri (solitamente numeri interi o stringhe di testo) posizionati sulla stessa linea e separati da uno spazio.

#### Line

La primitiva line viene identificata dal comando **LI** e la sua definizione richiede unicamente le coordinate di inizio e fine, oltre che al livello:

```
LI x1 y1 x2 y2 l
```

I punti  $(x_1, y_1)$  e  $(x_2, y_2)$  rappresentano rispettivamente le coordinate di inizio e di fine, mentre il carattere  $l$  rappresenta il livello, dato da un numero intero tra 0 e 15.

**Estensioni FidoCadJ:** dalla versione 0.23 di FidoCadJ, **LI** può essere seguito, alla linea seguente, da questa estensione:

```
FCJ a b c d e nv
```

dove  $a$  è un intero che rappresenta la presenza o no della freccia agli estremi del segmento (confrontare la tabella 3.1),  $b$  rappresenta lo stile della freccia (tabella 3.2). I parametri  $c$  e  $d$  indicano rispettivamente la lunghezza totale e la mezza larghezza della freccia, mentre  $e$  è un intero che indica il tipo di tratteggio. Se  $nv$  è eguale a 1, il comando **FCJ** deve essere seguito da due comandi **TY** contenenti il nome e il valore associati all'elemento.

*I Matematici troverebbero probabilmente più appropriato il termine "segmento".*

### 3. Formato di disegno, macro e librerie FidoCAD

$a$	Freccia
0	nessuna
1	all'inizio
2	alla fine
3	sia all'inizio che alla fine

Tabella 3.1.: Significato del parametro  $a$  per la presenza di una freccia alle estremità di una linea o di una curva Bézier.

$b$	Stile della freccia
0	riempimento standard della freccia
1	riempimento standard della freccia con indicazione di quota
2	freccia vuota
3	freccia vuota con indicazione di quota

Tabella 3.2.: Significato del parametro  $b$  per lo stile della freccia.

#### Filled o empty rectangle

Un rettangolo pieno o vuoto viene definito dal comando **RP** e **RV** rispettivamente, seguito dalle coordinate dei due vertici su una delle due diagonali, e dal livello.

```
RP x1 y1 x2 y2 l
RV x1 y1 x2 y2 l
```

i punti  $(x_1, y_1)$  e  $(x_2, y_2)$  rappresentano i due vertici e  $l$  è il livello, caratterizzato da un numero intero tra 0 e 15.

**Estensioni FidoCadJ:** Dalla versione 0.23 di FidoCadJ, **RP** e **RV** possono essere seguiti, alla linea successiva, dall'estensione:

```
FCJ e nv
```

dove  $e$  è un intero che descrive lo stile di tratteggio. Se  $nv$  è uguale a 1, il comando **FCJ** dovrebbe essere seguito da due comandi **TY** che descrivono il nome e il valore associati all'elemento.

#### Simple text (obsoleto)

La primitiva simple text è stata la primitiva supportata dalle prime versioni di FidoCAD. FidoCadJ la riconosce e la interpreta scrivendo il testo di dimensione 12, indipendentemente dallo zoom corrente.

Poichè la primitiva era considerata obsoleta da Lorenzo Lutti, il creatore di FidoCAD, FidoCadJ si adegua e, anche se essa viene correttamente interpretata, non viene resa disponibile nella barra degli strumenti. FidoCadJ memorizza questo elemento come testo avanzato e utilizza il comando **TY** quando il file viene salvato.

Il comando è **TE** e il formato è il seguente:

Bit	Peso	Comportamento
0	1	testo in grassetto
2	4	testo specchiato

Tabella 3.3.: Funzione dei bit nel campo dello stile di testo.

```
TE x1 y1 testo da scrivere
```

Il punto  $(x_1, y_1)$  corrisponde alla posizione alla quale “testo da scrivere” viene posizionato. Notare che manca l’informazione sul livello. FidoCadJ posiziona questo oggetto sul livello zero (circuito).

### Advanced Text

La primitiva advanced text offre molta più flessibilità rispetto a quella simple text descritta qui sopra.

Viene identificata dal comando **TY**, seguito da un certo numero di parametri per determinare l’orientazione del testo (ruotato o specchiato), e dalle dimensioni  $x$  e  $y$  e dal font usato. a causa della grande quantità di informazioni da specificare, il comando risultante è piuttosto complesso:

```
TY x1 y1 sy sx a s l f testo da scrivere
```

Il punto  $(x_1, y_1)$  corrisponde alla posizione della stringa “testo da scrivere”. I valori di  $s_y$  and  $s_x$  indicano le dimensioni verticali e orizzontali del testo in unità logiche. Per mia scelta FidoCadJ rispetta la dimensione verticale di inizio partendo da quella orizzontale, e questo rapporto di aspetto viene cambiato solo se strettamente necessario. La rotazione del testo è gestita dal termine  $a$ , espresso in gradi sessagesimali, mentre il valore  $s$  determina lo stile del testo, ci si riferisca alla tabella 3.3. Il livello utilizzato è dato dal termine  $l$ , e  $f$  indica il font da usare, oppure viene specificato un asterisco, usato per indicare l’uso del font standard Courier New font. Se il nome contiene degli spazi essi verranno sostituiti con il simbolo +.

La massima lunghezza della stringa di testo è di circa 80 parole. Il conteggio viene effettuato in parole e non in caratteri poichè nella struttura interna del programma le parole (e i comandi) vengono separati quando una linea viene interpretata.

### Filled or Empty polyline

Una poligonale piena o vuota viene indicata dal comando **PP** e **PV** rispettivamente, seguita dalle coordinate dei vertici che definiscono la poligonale e, dopo, dal livello. Il comando è

```
PP x1 y1 x2 y2 ... l
PV x1 y1 x2 y2 ... l
```

### 3. Formato di disegno, macro e librerie FidoCAD

dove i punti  $(x_1, y_1), (x_2, y_2) \dots$  sono i vertici che definiscono la poligonale e  $l$  è il livello, caratterizzato da un numero intero tra 0 e 15. La lunghezza della linea di comando può variare dipendentemente dal numero di vertici utilizzati. Il numero massimo di vertici disponibili utilizzato è stato arbitrariamente fissato a 20, per evitare linee di comando troppo lunghe.

**Estensioni FidoCadJ:** dalla versione 0.23 di FidoCadJ, **PP** e **PV** possono essere seguiti, alla linea successiva, da una estensione:

```
FCJ e nv
```

dove  $e$  è un intero che descrive lo stile di tratteggio. Se  $nv$  è uguale a 1, il comando **FCJ** dovrebbe essere seguito da due comandi **TY** che riportano il nome e il valore associati all'elemento.

#### Filled o Empty ellipse

Una ellisse, piena o vuota, è identificata dal comando **EP** e **EV** rispettivamente, seguiti dalle coordinate dei due vertici sulla diagonale e dal valore del livello.

```
EP x1 y1 x2 y2 l
EV x1 y1 x2 y2 l
```

il punto  $(x_1, y_1)$  rappresenta il primo vertice della diagonale,  $(x_2, y_2)$  il secondo, e  $l$  è il livello, identificato da un numero intero compreso tra 0 e 15.

**Estensioni FidoCadJ:** dalla versione 0.23 di FidoCadJ, **EP** e **EV** possono essere seguiti, alla linea successiva, da una estensione:

```
FCJ e nv
```

dove  $e$  è un intero che descrive lo stile di tratteggio. Se  $nv$  è uguale a 1, il comando **FCJ** dovrebbe essere seguito da due comandi **TY** che riportano il nome e il valore associati all'elemento.

#### Bézier curve

Un segmento di curva Bézier, nella sua variante cubica, è identificato da quattro vertici, i quali sono richiesti dal suo comando associato **BE**:

```
BE x1 y1 x2 y2 x3 y3 x4 y4 l
```

dove  $P_1 \equiv (x_1, y_1)$ ,  $P_2 \equiv (x_2, y_2)$ ,  $P_3 \equiv (x_3, y_3)$  e  $P_4 \equiv (x_4, y_4)$  sono i quattro punti di controllo del segmento di curva Bézier, dove  $l$  è il livello, identificato da un numero intero compreso tra 0 e 15. Dati i quattro punti così definiti, il segmento di curva viene calcolato attraverso l'espressione

$$B(t) = (1 - t)^3 P_1 + 3t(1 - t)^2 P_2 + 3t^2(1 - t) P_3 + t^3 P_4, \quad (3.1)$$

dove  $t \in [0, 1]$  è un parametro.

**Estensioni FidoCadJ:** dalla versione 0.23 di FidoCadJ, **BE** può essere seguito, alla linea successiva, da una estensione:



```
FCJ a b c d e nv
```

dove  $a$  è un intero che rappresenta o meno la presenza di una freccia agli estremi della curva (confrontare la tabella 3.1),  $b$  rappresenta lo stile della freccia (come da tabella 3.2). I parametri  $c$  e  $d$  definiscono la lunghezza totale e la metà lunghezza della freccia, mentre  $e$  è un intero che definisce lo stile di tratteggio. Se  $nv$  è uguale ad 1, il comando **FCJ** dovrebbe essere seguito da due comandi **TY** che definiscono il nome e il valore associato all'elemento.

### Natural cubic spline (complex curve)

Una spline cubica viene definita da un certo numero di vertici. La curva incrocia ogni vertice ed è calcolata in modo tale per cui assuma una buona levigatezza.<sup>1</sup> Nel formato FidoCadJ, una spline viene identificata dal comando **CV** e **CP**:

```
CV aa x1 y1 x2 y2 ... l
CP aa x1 y1 x2 y2 ... l
```

il parametro  $aa$  è uguale a 1 se la curva è chiusa, 0 altrimenti. I vertici  $(x_1, y_1)$ ,  $(x_2, y_2) \dots$  definiscono la spline e  $l$  è il livello, un numero intero che va da 0 a 15. La lunghezza della linea di comando varia quindi in funzione del numero di vertici presenti. Come nel caso della poligonale, il massimo numero di vertici disponibile internamente fissato è circa 100, in modo da evitare linee di comando troppo lunghe. La primitiva è stata introdotta dalla versione 0.24 e non è presente nella versione originale di FidoCAD.

**estensioni FidoCadJ:** **CV** e **CP** possono essere seguite, alla linea successiva dall'estensione:

```
FCJ a b c d e nv
```

dove  $a$  è un intero che rappresenta la presenza o meno di una freccia alle estremità della curva (si confronti anche la tabella 3.1),  $b$  rappresenta lo stile della freccia (si confronti anche la tabella 3.2). I parametri  $c$  e  $d$  definiscono rispettivamente la lunghezza totale e la metà larghezza della freccia, mentre  $e$  è un intero che definisce lo stile di tratteggio. Se  $nv$  è uguale a 1, il comando **FCJ** dovrebbe essere seguito da due comandi **TY** che definiscono il nome e il valore associati all'elemento.

### Electrical junction

La giunzione elettrica è un semplice cerchietto di dimensione costante e viene usato per rappresentare una connessione in uno schema elettrico. Viene identificata dal comando **SA** e richiede unicamente le coordinate e il livello di piazzamento:

```
SA x1 y1 l
```

<sup>1</sup> <http://www.cse.unsw.edu.au/~lambert/splines/natcubic.html>

### 3. Formato di disegno, macro e librerie FidoCAD

FidoCadJ fissa internamente il diametro del cerchio in due unità logiche.

Se **SA** è seguito da **FCJ**, il comando **FCJ** dovrebbe essere seguito da due comandi **TY** che definiscono il nome e il valore associati a questo elemento.

#### PCB Pad

Una piazzola PCB è identificata dal comando **PA** e viene caratterizzata dal suo stile (tondo, rettangolare e rettangolare smussato) e dal suo foro centrale:

```
PA x1 y1 dx dy si st l
```

dove il punto  $(x_1, y_1)$  rappresenta la posizione della piazzola,  $d_x$  è la larghezza della piazzola lungo l'asse  $x$ ,  $d_y$  è l'altezza (lungo l'asse  $y$ ). Il valore  $s_i$  è il diametro del foro interno, mentre  $s_t$  è lo stile della piazzola:

0 piazzola ovale

1 piazzola rettangolare

2 piazzola rettangolare con angoli smussati

Il valore di  $l$  deve essere un numero intero che indica il livello dove la piazzola deve essere posizionata.

Se **SA** è seguito da **FCJ**, il comando **FCJ** dovrebbe essere seguito da due comandi **TY** che definiscono il nome e il valore associati a questo elemento.

#### PCB track

La traccia PCB è essenzialmente un segmento, la cui larghezza può essere specificata. Gli angoli del segmento vengono arrotondati per facilitare la connessione con altre tracce PCB o piazzole. Il comando che viene usato è **PL**, con il formato seguente:

```
PL x1 y1 x2 y2 di l
```

La traccia viene disegnata tra i punti  $(x_1, y_1)$  e  $(x_2, y_2)$ , di larghezza complessiva  $d_i$ , e il livello utilizzato è  $l$ . Se **PL** è seguito da **FCJ**, il comando **FCJ** dovrebbe essere seguito da due comandi **TY** che specificano il nome e il valore dell'elemento.

#### Macro

Una macro consiste in un disegno o in un simbolo contenuto in una libreria. Generalmente, questo il modo con cui i simboli elettrici vengono sovente rappresentati. Il comando per fare una chiamata ad una macro è **MC**, e la chiamata avviene in questo modo:

```
MC x1 y1 o m n
```

La macro viene disegnata posiziando il riferimento alle coordinate  $(x_1, y_1)$ , e l'orientazione è definita tramite il valore di  $o$  (moltiplicato per  $90^\circ$  in senso orario) e se  $m$  è posto uguale ad 1 a macro viene specchiata. L'ultimo parametro,  $n$ , è il nome della macro da richiamare specificata all'interno della libreria, come *library.code*.

Se **MC** è seguito da **FCJ**, il comando **FCJ** dovrebbe essere seguito da due comandi **TY** che specificano il nome e il valore dell'elemento.

## 3.4. Estensioni FidoCadJ

Dalla versione 0.21, FidoCadJ ha introdotto alcuni affinamenti al formato originale FidoCAD format. Le estensioni FidoCadJ sono rappresentate nel codice dal comando **FCJ**. Questo comando non mai usato da solo, ma significa che FidoCadJ richiede di specificare informazioni aggiuntive su cosa è stato scritto alla linea precedente. Facciamo un esempio:

```
[FidoCAD]
MC 40 30 0 0 080
FCJ
TY 50 35 4 3 0 0 0 * R1
TY 50 40 4 3 0 0 0 * 47k
```

La presenza del comando **FCJ** indica che il nome e il valore della macro specificate nella prima riga sono date dai due comandi **TY** che seguono. Solo le coordinate e i font vengono presi in considerazione per la interpretazione del testo.

FidoCadJ permette di attivare lo “strict FidoCAD compatibility mode”, tramite il quale vengono disabilitate tutte le estensioni. In questo modo FidoCadJ sarà perfettamente compatibile con il FidoCAD originale, eccetto per il fatto che FidoCadJ continuerà ad utilizzare la codifica UTF-8 invece della vecchia CP-1252 usata da FidoCAD. FidoCAD per Windows non può comprendere il significato delle informazioni aggiuntive portate dal comando **FCJ**. Quando FidoCAD legge un disegno FidoCadJ che contiene delle estensioni restituisce diversi errori e interpreta un disegno differente da quello originale FidoCadJ. Il risultato dipende da quale estensione è stata utilizzata: una linea tratteggiata viene resa continua e le frecce non vengono rappresentate. Il testo associato al nome e al valore di una macro verrà, invece, perfettamente interpretato.

La figura 3.1 mostra cosa può essere ottenuto utilizzando FidoCAD per leggere il file FidoCadJ che disegna la figura 2.19. Ignorando gli errori rilevati da FidoCAD si osserva che, nonostante ci siano alcuni dettagli mancanti (il tratteggio e la freccia), in generale il disegno è comunque comprensibile.

Una differenza importante dal FidoCAD originale è che FidoCadJ permette di salvare un certo numero di configurazioni di parametri nel file che viene generato. Esse sono quelle che sono usate da **FJC** e dovrebbero essere posizionate all'inizio del file. Il prossimo paragrafo le descriverà.

### 3. Formato di disegno, macro e librerie FidoCAD

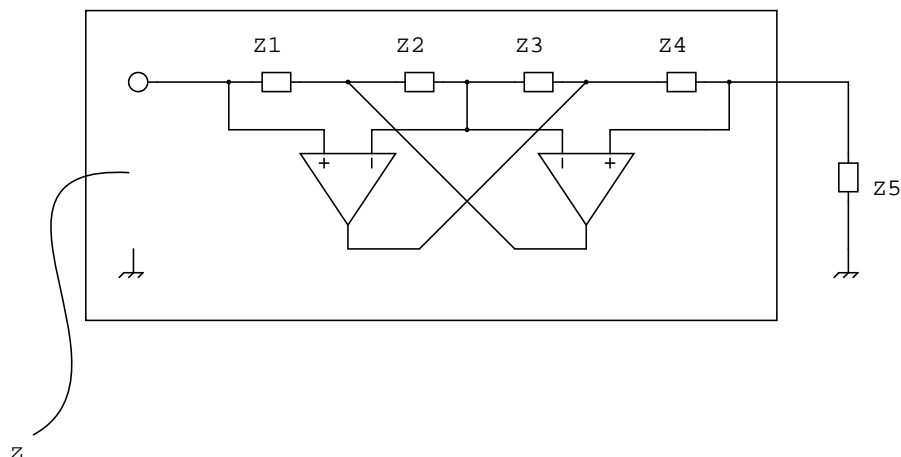


Figura 3.1.: La figura 2.19 come apparirebbe su FidoCAD per Windows.

#### 3.4.1. Impostazione dei livelli

L'impostazione dei livelli viene specificata tramite il comando **FJC L**. FidoCadJ salverà questi dati solo se il livello corrispondente è stato modificato dalle sue impostazioni base. La sintassi è la seguente:

```
FJC L n xxxx yy
```

dove  $n$  rappresenta il numero del livello (da 0 a 15),  $xxxx$  è un intero a 32 bit contenente il codice RGB da usarsi per il livello. Le componenti del rosso sono contenute dal bit 16 al 23, il verde dall'8 al 15 e il blu dal bit 0 al bit 7. Il valore  $yy$  è una costante a virgola mobile in singola precisione che permette di impostare la trasparenza del livello, compresa tra 0.0 (completamente trasparente) e 1.0 (completamente opaco).

Una seconda importante informazione è il nome del livello, specificata come segue (se l'utente l'ha modificata):

```
FJC N n aaaaa
```

dove  $n$  rappresenta il numero del livello (da 0 a 15), mentre  $aaaa$  è il nome da assegnare al livello. Se il comando non è presente, il nome del livello è quello assegnato da FidoCadJ, dipendentemente dalla lingua impostata e dalle configurazioni locali del sistema operativo in uso.

#### 3.4.2. Impostazioni delle connessioni elettriche

La dimensione dei cerchietti neri utilizzati per indicare la connessione elettrica possono essere modificati dall'utente. Quando le estensioni FidoCadJ sono attive, il valore

### 3.5. Tolleranza agli errori di sintassi

selezionato viene salvato nel file con il comando **FJC C** come segue:

```
FJC C aaaa
```

dove *aaaa* è un valore in virgola mobile a doppia precisione (positivo) che permette di impostare il diametro della connessione elettrica, in unità logiche FidoCadJ.

#### 3.4.3. Larghezza del tratto

La larghezza del tratto per la “penna” utilizzata durante il disegno di schemi elettrici può essere modificata. FidoCadJ utilizza il comando **FJC A** per specificare la larghezza del tratto di linee, ellissi, Bézier, spline, rettangoli and poligoni. La larghezza è una costante in doppia precisione.

```
FJC A aaaa
```

Dove *aaaa* rappresenta la larghezza del tratto (in unità logiche). Le impostazioni di base la definiscono pari a 0.5 unità logiche. Nelle vecchie versioni di FidoCadJ, il comando **FJC B bbbb** era usato per specificare la larghezza del tratto per linee curve (ellissi, Bézier). Dalla versione 0.23.6 questa differenza è stata eliminata e questo comando non viene più adottato.

### 3.5. Tolleranza agli errori di sintassi

FidoCadJ è progettato per tollerare errori di sintassi. Certamente, a meno di non avere la palla di cristallo collegata alla USB, il programma non potrà correggere errori, ma si limiterà semplicemente a saltare (e cancellare) le linee errate. Una eccezione a questo comportamento è che un certo numero di elementi può essere specificato senza il livello, che verrà assunto come livello 0 (schema elettrico). E' stato implementato ciò per compatibilità con FidoCAD.

### 3.6. Formati delle librerie

La struttura dei file di libreria è piuttosto semplice:

```
[FIDOLIB Librairie de base]
{Syboles de base}
[000 Terminal]
LI 100 100 102 100
EV 102 98 106 102
[010 Terminal +]
LI 100 100 102 100
EV 102 98 106 102
LI 103 100 105 100
LI 104 99 104 101
[020 Terminal -]
LI 100 100 102 100
```

### 3. Formato di disegno, macro e librerie FidoCAD

```
EV 102 98 106 102
LI 103 100 105 100
...
```

La prima linea contiene, tra parentesi quadre, il nome della libreria (preceduta da FIDOLIB). La seconda linea contiene, tra parentesi graffe, la categoria della libreria sotto la quale le macro, specificate più avanti nel file, verranno immagazzinate. Ogni macro è composta da una intestazione (tra parentesi quadre) e una sequenza di comandi. L'intestazione è costituita da un *part name* (che deve essere unico nella libreria) e la sua descrizione. Il part name verrà utilizzato all'interno di uno script FidoCadJ, mentre la descrizione sarà di aiuto all'utente al momento della scelta delle varie macro contenute nel file. I comandi non sono altro che un disegno FidoCadJ, dove sono utilizzate le coordinate (100,100) quale punto di origine. Questo punto verrà utilizzato come riferimento quando la macro verrà chiamata. In uno script FidoCadJ una macro è identificata tramite "library.macro" usato con il comando MC.

Nulla impedisce la chiamata di una macro attraverso un'altra macro. Tuttavia la ricorsione (per esempio una macro che chiama se stessa) deve essere evitata.

Una libreria *non deve contenere*, attraverso le definizioni della macro, alcuna informazione concernente la configurazione di FidoCadJ. In altre parole il comando FJC non deve *mai* apparire dentro un file di libreria.

## 3.7. Librerie standard

FidoCadJ contiene le due librerie fornite tradizionalmente con FidoCAD. In particolare, esse sono la libreria standard e la libreria che contiene i simboli PCB. Un'altra libreria, sviluppata originariamente per FidoCAD FidoCAD e che è di tale comprovata utilità che è stata introdotta quale parte di FidoCadJ è la libreria IHRAM. IHRAM è l'acronimo di *it.hobby.radioamatori.moderato* il quale è un gruppo di discussione Usenet Italiano.

D'ogni modo, è possibile sovrascrivere il contenuto di queste librerie interne specificando (in "File/Options/Libraries Directory" menu) una cartella contenente le librerie da caricare. Se FidoCadJ troverà nella cartella un file di nome `FCDstdlib.fcl` il suo contenuto verrà utilizzato in luogo della libreria standard. Se è presente invece un file di nome `PCB.fcl` il suo contenuto verrà utilizzato invece della libreria standard PCB interna. Le altre librerie che hanno altri nomi di file (ma sempre con l'estensione `fcl`) verranno caricate insieme alle librerie standard. Se è presente un file di nome `IHRAM.FCL` esso verrà caricato in luogo della libreria interna. E' anche disponibile una libreria di simboli elettrici chiamata `elettrotecnica.fcl`.

FidoCadJ considera le librerie descritte poco sopra quali parti integranti del medesimo. In altre parole, i simboli contenuti nei disegni, appartenenti a queste librerie, non vengono separati. Questo a meno che non venga selezionata l'opzione "Strict compatibility with FidoCAD". In questo caso solo la libreria originale FidoCAD viene considerata standard, esattamente come farebbe il software originale.

## 4. Conclusioni

In questo manuale è stato descritto come utilizzare FidoCadJ per disegnare uno schema elettrico o un semplice PCB. A questo punto il lettore dovrebbe avere in possesso tutti gli elementi necessari per utilizzare FidoCadJ efficacemente per i propri scopi.

FidoCadJ può non essere utilizzato unicamente per produrre disegni concernenti il mondo elettronico. Può essere utilizzato per produrre qualunque tipo di disegno 2D e in molte altre occasioni, a patto che siano disponibili librerie specifiche.

Il vantaggio di un programma libero è che esso sia completamente a disposizione della comunità. Per questa ragione, l'opinione del lettore viene tenuta in grande considerazione (se non altro per capire se il progetto debba essere sviluppato ulteriormente e in che direzione). Per contattarmi potete partecipare al forum di SourceForge dedicato a FidoCadJ.<sup>1</sup>

---

<sup>1</sup><https://sourceforge.net/projects/FidoCadJ/forums/forum/997486>

## A. Informazioni relative alla piattaforma specifica

### A.1. MacOSX

Una delle più frequenti critiche mosse verso le prime versioni di FidoCadJ dagli utenti Macintosh (come me, a proposito) era la scarsa integrazione del programma con MacOSX. Dalla versione 0.21.1, sono stati compiuti degli sforzi per rendere FidoCadJ più compatibile con l'aspetto e la filosofia delle applicazioni Mac. Per questa ragione, qualche dettaglio nel programma è leggermente differente quando FidoCadJ viene eseguito su una piattaforma Apple:

- per impostazione di fabbrica FidoCadJ usa il Quaqua<sup>1</sup> look and feel, quando viene fatta girare l'applicazione completa (FidoCadJ.app invece di FidoCadJ.jar). Poichè Quaqua potrebbe rallentare le prestazioni delle macchine meno recenti, è possibile disabilitarlo tramite il menu relativo alle impostazioni preferite.
- La barra del menu è al proprio posto, cioè sulla sommità dello schermo
- La voce di menu "Preferences" e "About FidoCadJ" sono al loro posto, cioè sotto il menu FidoCadJ.
- Il programma comunica al sistema operativo che può aprire file di tipo `.fcd`; i quali vengono associati alla loro icona specifica, che dovrebbe essere sufficientemente esplicativa.

#### A.1.1. Come scaricare ed eseguire FidoCadJ su MacOSX

FidoCadJ può essere eseguito su una versione di MacOSX più recente della 10.3.9 (Panther). La ragione è che FidoCadJ ha bisogno almeno della versione 1.5 di Java, che Apple fornisce con il sistema operativo. Per ragioni commerciali, Apple non sembra incline ad installare Java sulle ultime versioni di MacOSX. Questo software è essenziale per FidoCadJ, ma fortunatamente è possibile scaricarlo ed installarlo. A tal proposito, Apple non permette di distribuire nel proprio App Store software basato su Java o distribuito sotto licenza GPL. Questa è una delle ragioni più importanti per le quali è improbabile che FidoCadJ verrà reso portabile verso iPad e iPhone.

Anche se è possibile utilizzare direttamente l'archivio Java `FidoCadJ.jar` come si farebbe per altri sistemi operativi, su MacOSX è possibile utilizzare l'applicazione per

---

<sup>1</sup><http://www.randelshofer.ch/quaqua/>



esso specificatamente progettata. Tutto funziona come per una applicazione nativa: è possibile scaricare una immagine utilizzando il seguente collegamento:

[http://sourceforge.net/projects/FidoCadJ/files/FidoCadJ\\_MacOSX.dmg/download](http://sourceforge.net/projects/FidoCadJ/files/FidoCadJ_MacOSX.dmg/download)

A questo punto è possibile aprire l'immagine e spostare **FidoCadJ.app** nella cartella **Applications**, dove l'esecuzione dell'applicativo avviene nello stesso modo come per qualunque altra applicazione Macintosh. Per disinstallare FidoCadJ, è possibile trascinare semplicemente il file **FidoCadJ.app** nel cestino.

## A.2. Linux

di Roby IZ1CYN

Prerequisiti: Installare JRE 6 da Sun e/o OpenJDK 6 JRE (o versioni precedenti compatibili con le specifiche del programma) Nel paragrafo A.2.1 verrà descritto come installare il programma utilizzando unicamente comandi da terminale.

Nel paragrafo A.2.2, verrà utilizzata l'interazione con l'ambiente grafico. Una configurazione scadente del Java Runtime Environment determinerà prestazioni scendenti di FidoCadJ<sup>2</sup>

### A.2.1. Usare una piattaforma qualunque, da terminale

Scaricare il programma utilizzando il comando `wget`:

```
$ wget http://downloads.sourceforge.net/project/FidoCadJ/FidoCadJ
.jar?use_mirror=garr
--00:48:18-- http://downloads.sourceforge.net/project/FidoCadJ/
FidoCadJ.jar?use_mirror=garr
=> 'FidoCadJ.jar?use_mirror=garr'
Resolution of downloads.sourceforge.net is being done...
216.34.181.59
Connection to downloads.sourceforge.net|216.34.181.59:80...
connected.
HTTP request sent, waiting for answer... 302 Found
URL: http://garr.dl.sourceforge.net/project/FidoCadJ/FidoCadJ.jar
--00:48:24-- http://garr.dl.sourceforge.net/project/FidoCadJ/
FidoCadJ.jar
=> 'FidoCadJ.jar'
Resolution of garr.dl.sourceforge.net is being done...
193.206.140.34
Connection to garr.dl.sourceforge.net|193.206.140.34:80...
connected.
HTTP request sent, waiting for answer... 200 OK
Length: 343,207 (335K) [application/java-archive]
```

<sup>2</sup>N.d.r. Vi assicuro che è vero! Per favore, non insultatemi se la vostra distribuzione di Linux viene fornita con una versione inaffidabile di Java.

#### A. Informazioni relative alla piattaforma specifica

```
100% [=====>] 343,207      422.48K/s
00:48:30 (420.55 KB/s) - "FidoCadJ.jar" saved [343207/343207]
$
```

Alternativamente, o se si sono riscontrati dei problemi, è possibile scaricare il file tramite qualunque browser dall'URL seguente:

<http://sourceforge.net/projects/FidoCadJ/files/FidoCadJ.jar/download>

e salvare il file nella propria cartella `/home/`.

Creiamo ora una nuova cartella (ma come prima cosa accediamo ai privilegi di superuser tramite `su` o `sudo -s`):

```
# mkdir /usr/bin/FidoCadJ
```

... spostiamo ora dentro di essa il file precedentemente scaricato (sostituendo `<user>` con il nome utente dell'account presso il quale il file è stato scaricato:

```
# mv /home/<user>/FidoCadJ.jar /usr/bin/FidoCadJ
```

Rendiamo eseguibile il file

```
# chmod +x /usr/bin/FidoCadJ/FidoCadJ.jar
```

Non dimentichiamoci di tornare ad essere utenti normali:

```
# exit
```

Finalmente possiamo eseguire il programma:

```
$ /usr/bin/FidoCadJ/FidoCadJ.jar
$
```

#### A.2.2. Su di un sistema grafico

In questo esempio, la piattaforma grafica è Ubuntu 8.04, ma le cose non cambiano molto usando altre versioni:

- Scarichiamo il file tramite un browser, o con Gwget o strumenti simili.
- Possiamo quindi avviare il nostro File Manager (Nautilus, Konqueror...) con i privilegi di root (qualora non esista un comando specifico nel menu, è possibile avviarlo dal terminale utilizzando il comando `sudo nautilus`). Creiamo quindi la seguente cartella:

```
/usr/bin/FidoCadJ
```

e quindi spostiamoci dentro il file scaricato. Sarà da qualche parte in:

```
/home/<user>/
```

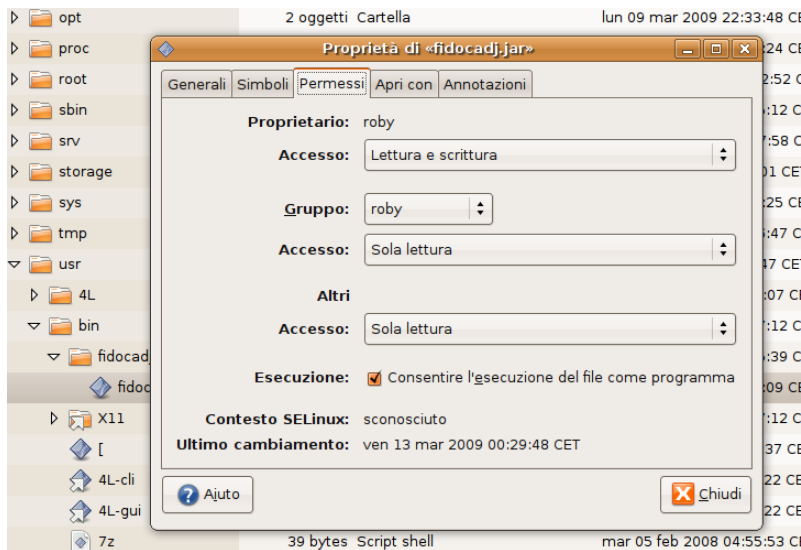


Figura A.1.: L'impostazione dei permessi del file, su Ubuntu 8.04.

- Facciamo ora click con il tasto destro sul file, dalla finestra che comparirà selezioniamo “rights” e aggiungiamo un segno di spunta a “Allow the execution of the file as a program”, come mostrato in figura A.1
- Selezioniamo quindi “Open as” e poi selezioniamo “OpenJDK Java 6 Runtime” o “Sun Java 6 Runtime”, come si può vedere in figura A.2.
- Clicchiamo quindi su “Close” e siamo pronti all'esecuzione di FidoCadJ: facciamo un semplice doppio click sull'eseguibile oppure aggiungiamolo al menu principale. Il comando da aggiungere è `/usr/bin/FidoCadJ/FidoCadJ.jar`.

## A.3. Windows

Dalla versione 0.23, cioè da quando FidoCadJ riconosce la piattaforma Windows, viene usato il Look and Feel nativo per questo sistema.

### A.3.1. come scaricare ed eseguire FidoCadJ

Molto spesso, avendo Java installato, è sufficiente scaricare il file `FidoCadJ.jar` ed eseguirlo con un doppio click. Se, dopo averlo scaricato, il sistema operativo lo riconosce come un archivio zip probabilmente Java non è installato. Oracle permette di scaricare ed eseguire una versione gratuita di java da:

<http://www.java.com/it/download/>

A. Informazioni relative alla piattaforma specifica

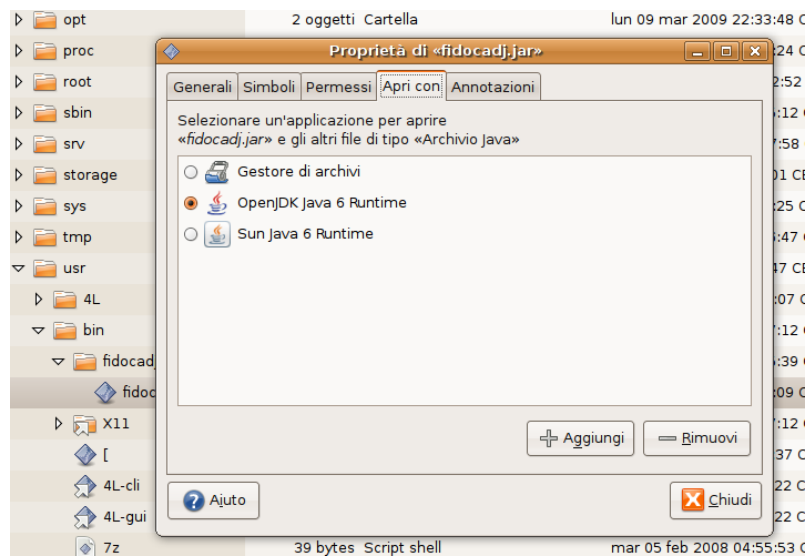


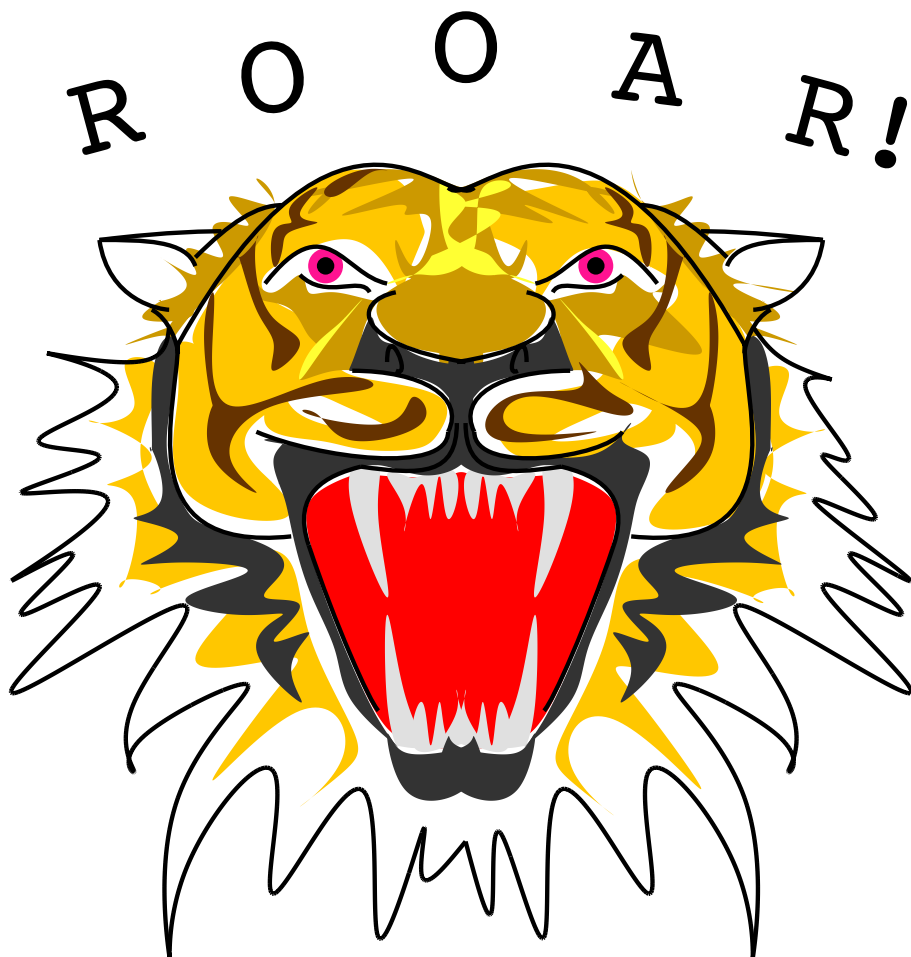
Figura A.2.: Impostazione dei diritti di esecuzione per la Java virtual machine, su Ubuntu 8.04.

## B. FidoCadJ art



Uno dei padri dell'analisi circuitale, in un ritratto di Zeno Martini.

*B. FidoCadJ art*



Tiger, tiger, burning bright... Di Igor Arbanas “elettrodomus”.

# Indice analitico

- [+](#), [37](#)
- [L<sup>A</sup>T<sub>E</sub>X](#), [26](#)
- [A4](#), [19](#)
- [advanced text](#), [37](#)
- [App Store](#), [46](#)
- [asterisco](#), [37](#)
- [Bézier](#), [8](#), [35](#), [38](#)
- [barra degli strumenti](#), [6](#), [11](#)
- [barra di comando](#), [6](#)
- [BE](#), [38](#)
- [bromografo](#), [19](#)
- [CAD](#), [15](#)
- [CadSoft](#), [26](#)
- [CadSoft Eagle](#), [1](#)
- [Change key](#), [33](#)
- [Cinese](#), [3](#)
- [circuito](#), [37](#)
- [codice](#), [13](#), [15](#)
- [codifica](#), [34](#), [41](#)
- [colore](#), [15](#)
- [compatibilità con FidoCAD](#), [44](#)
- [compressione con perdite](#), [26](#)
- [connection](#), [35](#)
- [connessioni PCB](#), [19](#)
- [Courier New](#), [37](#)
- [CP](#), [39](#)
- [CP-1252](#), [41](#)
- [curve](#), [8](#)
- [CV](#), [39](#)
- [Delete \(simbolo di libreria\)](#), [33](#)
- [dimensioni dei caratteri](#), [19](#)
- [disegno vettoriale](#), [6](#)
- [Drag and drop](#), [33](#)
- [e-mail](#), [13](#)
- [Eagle](#), [v](#), [26](#)
- [elemento](#), [35](#)
- [ellipse](#), [8](#), [35](#)
- [ellisse](#), [38](#)
- [EP](#), [38](#)
- [EPS](#), [1](#), [26](#)
- [esportazione](#), [25](#)
- [Estensioni FidoCadJ](#), [13](#)
- [estensioni FidoCadJ](#), [41](#)
- [EV](#), [38](#)
- [FCJ](#), [41](#)
- [FidoCAD](#), [1](#), [2](#), [9](#), [13](#), [23](#), [25](#), [30](#), [34](#), [36](#), [39](#), [41](#), [43](#), [44](#)
- [FidoCadJ.app](#), [46](#)
- [fidocadj.jar](#), [46](#)
- [FIDOLIB](#), [44](#)
- [FidoReadJ](#), [2](#)
- [FJC](#), [41](#), [44](#)
- [FJC A](#), [43](#)
- [FJC B](#), [43](#)
- [FJC C](#), [43](#)
- [FJC L](#), [42](#)
- [formato bitmap](#), [25](#)
- [formato vettoriale](#), [25](#)
- [forum](#), [13](#)
- [Francese](#), [3](#)
- [frecce](#), [23](#)
- [Giapponese](#), [3](#)
- [GIG](#), [23](#)
- [giunzione](#), [39](#)
- [griglia](#), [16](#)
- [Group](#), [32](#)
- [GTK+](#), [28](#)

## *Indice analitico*

- incroci di piste, 16
- Inglese, 3
- Inkscape, 26
- interprete, 2
- intestazioen, 44
- intestazione, 34
- iPad, 46
- iPhone, 46
- it.hobby.elettronica, v, 2
- it.hobby.fai-da-te, 2
- Italiano, 3
  
- jar, 25
- Java, 1, 2, 25, 28
- JPG, 26
- JRE, 25, 47
- junction, 8
  
- Key, 32
  
- L<sup>A</sup>T<sub>E</sub>X, 3
- LI, 35
- Library complete name, 30
- Library filename, 30
- libreria IHRaM, 29
- libreria PCB, 44
- libreria standard, 44
- Librerie
  - libreria standard, 29
- librerie, 29
- line, 8, 35
- linea di comando, 27
- Linux, v, 28
- livello, 15, 18, 43
- look & feel, 28
- Lorenzo Lutti, 1, 36
- lunghezza massima del testo, 37
  
- Macintosh, 7, 25
- MacOSX, 6, 7, 29, 46
- macro, 9, 35, 40, 44
- massimo numero di vertici, 39
- Matematicamente, 5
- MC, 40, 41, 44
- Metal, 6, 7
  
- modalità selezione, 18
- Motif, 28
- move, 8
- MS-DOS prompt, 25
  
- Name, 32
- newsgroup, v, 13
- newsgroups, 2
- numero massimo di vertici, 38
  
- Object oriented programming, 2
- obsoleto, 36
- Olandese, 3
- OpenJDK, 47
- Origine, 32
  
- PA, 40
- palla di cristallo, 43
- Panther, 46
- parametri, 35
- parentesi quadre, 44
- PCB, 1, 15, 16, 19, 20, 34
  - footprint, 9, 18
- PCB line, 18
- PCB pad, 35
- PCB pad, 8
- PCB track, 8, 35
- PDF, 1
- PDF<sup>L</sup>A<sub>T</sub>E<sub>X</sub>, 26
- PGF, 26
- piazzamento automatico, 16
- piazzola PCB, 40
- PL, 40
- PNG, 26, 28
- poligonale, 37
- poligono, 16, 18
- polyline, 8, 35
- Postscript, 26
- PP, 37, 38
- Press&Peel, 19
- primitiva, 6
- primitive, 6
- PV, 37, 38
  
- Quaqua, v, 46



- rectangle, 8, 16, 35
- Rename (simbolo di libreria), 33
- resistore, 11
- rettangolo, 36
- ricerche rapide, 9
- ricorsione, 44
- righello, 22
- risoluzione, 15
- RP, 36
- RV, 36
  
- SA, 39, 40
- Safari, 26
- saldature, 15
- sbroglio automatico, 16
- schema elettrico, 34
- schemi elettrici, 1, 9, 15
- SCR, 26
- segmento, 35
- Seleziona, 8
- seleziona, 9
- Separare un simbolo, 33
- silk-screen, 15, 16
- simple text, 36, 37
- sistema di coordinate, 34
- SourceForge, 3
- Spagnolo, 3
- specchiare le scritte, 18
- spline, 35, 39
- spostamento diagonale, 15
- stampa, 19
- standard library, 9
- stile di tratteggio, 23
- stili di freccia, 23
- stili di tratteggio, 23
- Sun, 25, 47
- Sun/Oracle, 6
- superuser, 48
- SVG, 26
  
- TE, 36
- Tedesco, 3
- terminale, 25
- testo
  - ruotato, 37
  - specchiato, 37
  - stile, 37
- text, 8, 13, 35
- tolleranza agli errori, 43
- traccia PCB, 40
- transistor, 9
- trasferibili R41, 16, 18
- TY, 36, 37
  
- Ubuntu, 48–50
- unità logica, 15
- unità logiche, 23, 37
- Unix, 25, 27
- Usenet group, 1
- UTF-8, 34, 41
  
- Windows, 1, 25
- WInE, 1
- www.electroyou.it, 4, 5
- www.grix.it, v, 5
  
- zoom, 8, 36

*Indice analitico*

Questo manuale è stato scritto utilizzando PDF $\LaTeX$  su MacOSX. Il listing è stato composto utilizzando il pacchetto `listings`. Il pacchetto `pgf` è stato utilizzato per la figura 2.5 come anche per il ritratto a pagina 51. Tutti questi pacchetti sono disponibili come archivi CTAN.