Fondamenti d'Informatica

Introduzione all'architettura dei calcolatori

F. Fabris

Bozza del 12 ottobre 2020

Corso di Laurea Triennale in *Ingegneria Elettronica e Informatica*

A.A. 2019/20

Indice

Prefazione

Questa dispensa contiene il materiale didattico necessario per la prima parte del corso di *Fondamenti d'Informatica*, inserito al I anno della laurea triennale in *Ingegneria Elettronica e Informatica*, presso l'Università degli Studi di Trieste. Essa è stata pensata come un'introduzione all'architettura dei calcolatori e alle tecnologie elettroniche che la supportano.
Si parte da un'introduzione storica all'Informatica (capitolo 1) seguita da una riflessione sul significato dell'informazione (capitolo 2), sottolineando la differenza tra *informazione sintattica* e *informazione semantica*. Si tratta successivamente della *codifica* dell'informazione dal mondo esterno - alfabeti finiti, notazioni numeriche, segnali analogici e immagini - al mondo interno del calcolatore, fatto essenzialmente di stringhe binarie.
Nel capitolo 3 si fornisce una brevissima introduzione alle tecnologie elettroniche, mostrando la storia e i principi di funzionamento dei principali dispositivi elettronici elementari (diodi a vuoto, triodi, diodi a semiconduttore, transistor).
Attraverso lo studio formale dell'*Algebra Booleana* (capitolo 4), autentico pilastro di tutta l'elettronica digitale, si perviene poi alla realizzazione delle *porte logiche*, che vengono caratterizzate anche dal punto di vista circuitale mediante i transistor, in modo da creare una connessione tra funzione astratta e sua attuazione circuitale.
I capitoli 5 e 6 sono dedicati rispettivamente ai *circuiti combinatori* e a quelli *sequenziali*, con l'obiettivo di realizzare i moduli circuitali - registri, contatori, *multiplexer*, sommatori, *flip-flop*, ecc.- che rappresentano le cellule costitutive di base per la realizzazione di un qualunque dispositivo digitale, primo fra tutti il *computer*.
Tutti i temi relativi ai dispositivi elettronici, alle porte logiche e ai circuiti combinatori e sequenziali, saranno ripresi e ampliati, anche nell'ambito di un'attività nei laboratori, nei corsi di *Reti Logiche* ed *Elettronica* e nei corsi di ulteriore specializzazione.
Il capitolo 8 illustra il *modello di computazione* di *Shepherdson-Sturgis* (detto anche modello RAM), che costituisce una chiave di lettura del livello *assembler* della macchina; contemporaneamente esso anticipa i temi che saranno trattati nel corso di *Complessità e Crittografia* del IV anno, dove si ragionerà sui limiti intrinseci dell'approccio procedurale-algoritmico, delineati rispettivamente dalla *Teoria dell Computabilità* (problemi indecidibili) e dalla *Teoria della Complessità* computazionale (problemi intrattabili).
Il capitolo 7 illustra invece, in modo sintetico, la struttura architetturale generale di un calcolatore, soffermandosi sull'organizzazione della CPU, sull'esecuzione del ciclo *fetch-execute* e sulla gerarchia di memoria.

Lo spirito col quale è stato organizzato il materiale è soprattutto quello di sviluppare, nell'allievo ingegnere, l'idea che la straordinaria domanda di complessità che caratterizza i progetti odierni dei computer trova risposta nel rigore metodologico dell'astrazione matematica in generale, e dell'Algebra Booleana in particolare; per questo motivo si è dedicato a quest'ultima disciplina uno spazio maggiore del solito, in modo che sia molto solida la base sulla quale erigere le infrastrutture e le gerarchie circuitali di ordine superiore. Si sono viceversa trascurati alcuni particolari legati allo sviluppo tecnologico e/o al dettaglio fine dell'architettura, che andrebbero eventualmente affrontati in un corso avanzato di architetture.

La maggior parte del materiale didattico riguardante la parte centrale e più significativa del corso (Algebra Boolena, porte logiche, circuiti combinatori e sequenziali, moduli circuitali) è stata organizzata sulla base delle dispense realizzate a suo tempo dal prof. Antonio D'Amore [1], che per moltissimi anni è stato docente dei corsi di *Circuiti logici e impulsivi* e di *Reti logiche* presso l'Università degli Studi di Trieste; sono stati molto utili anche

i testi di Bucci [2] e di Clements [3], ai quali rimandiamo i lettori che volessero approfondire gli aspetti tecnici più sofisticati che in questa sede non si sono potuti affrontare.

Il materiale trattato nel corso avrà il suo naturale completamento nei corsi di *Reti Logiche* (II anno) e in quello di *Sistemi Operativi* (III anno per il curriculum di *Applicazioni Informatiche*) della Laurea Triennale in Ingegneria Elettronica e Informatica.

Capitolo 1

Introduzione storica

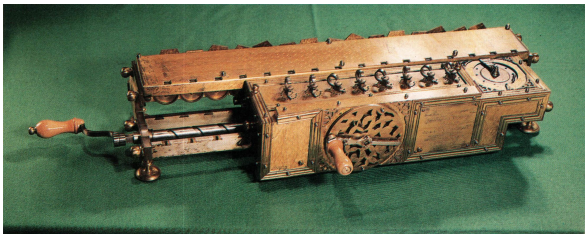
Quando si pensa alla parola *Informatica*, che deriva dalla contrazione francese di *Informa*tion Automa*tique*, l'immagine corre necessariamente al calcolatore e ai suoi accessori periferici (la tastiera, lo schermo, il *mouse* ecc.), al punto che la traduzione inglese viene resa con la locuzione *Computer Science*. Anche se è inevitabile riconoscere l'importanza del calcolatore, inteso come macchina fisica che attua gli schemi concettuali evocati dall'Informatica e che ha decretato il successo e la permeabilità delle tecnologie informatiche, è necessario prendere coscienza del fatto che l'Informatica non è *riducibile* alla macchina. Essa non solo è indipendente dalla tecnologia specifica impiegata per costruire i calcolatori (nella fattispecie la tecnologia elettronica dei semiconduttori), ma è indipendente persino dall'esistenza di una macchina fisica che la renda operativa, tant'è che i fondamenti dell'Informatica, dati dalla *Teoria della Computabilità*, furono sviluppati *prima* della costruzione materiale del primo calcolatore digitale, lo *Z1*, attuata dall'ingegnere tedesco *Konrad Zuse* tra il 1936 e il 1938.

La tecnologia informatica si sviluppa a partire dalla metà degli anni '30, in un momento felice di congiunzione tra due correnti operative e di pensiero ben distinte: da una parte c'era chi inseguiva il sogno millenario di una macchina per fare i calcoli in modo automatico (meccanica nelle prime versioni, elettromeccanica ed elettronica nelle ultime); dall'altra c'era chi si occupava dei *fondamenti logici e assiomatici della Matematica*, sognando una sorta di "meccanizzazione" della stessa, che consentisse di ricavare tutti i *teoremi* di una certa teoria matematica a partire dai suoi *assiomi* e dalle *regole di inferenza*. L'interazione tra queste due correnti di pensiero costituì il contesto fecondo attraverso il quale si passò dai sogni alla realtà.

1.1 Dalle calcolatrici meccaniche al primo computer

La storia della computazione numerica parte dagli abaci cinesi del 1200 D.C., mentre la prima realizzazione di una macchina automatica per il calcolo aritmetico viene fatta risalire a *Blaise Pascal*, filosofo, matematico e fisico francese, che nel 1643 realizzò un dispositivo meccanico per eseguire automaticamente addizioni e sottrazioni, la cosiddetta *Pascalina* (fig.1.1a). È però acclarato che già 150 anni prima *Leonardo da Vinci* aveva progettato una macchina analoga, anche se non arrivò mai a una sua costruzione. Qualche anno dopo, a partire dal 1674, il famoso filosofo e matematico tedesco *Gottfried Wilhelm Leibniz* presentò il progetto di una macchina calcolatrice a ruote e ingranaggi (le *Ruote di Leibniz*) , che era in grado di effettuare moltiplicazioni e divisioni (si veda figura 1.1b). Leibniz è però famoso soprattutto per il suo contributo fondamentale all'individuazione delle basi della Logica Simbolica ("*L'Arte Combinatoria*"), su cui si regge il funzionamento di moderni calcolatori. I successivi sviluppi in tale settore, ad opera di *George Boole*, *Alfred Whitehead*, *Bertrand Russell* e *Giuseppe Peano*, diedero consistenza al sogno di Leibniz di un ragionamento simbolico universale, con la nascita di una nuova disciplina matematica, la *Logica Simbolica*. L'idea di fondo dell'*Arte Combinatoria* è quella di trovare una logica capace non

(a) La macchina *Pascalina* per fare somme e sottrazioni

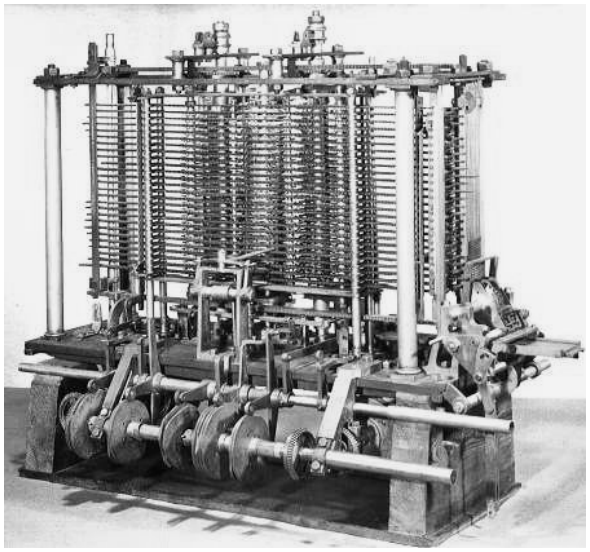


(b) Le *Ruote di Leibniz* per effettuare moltiplicazioni e divisioni

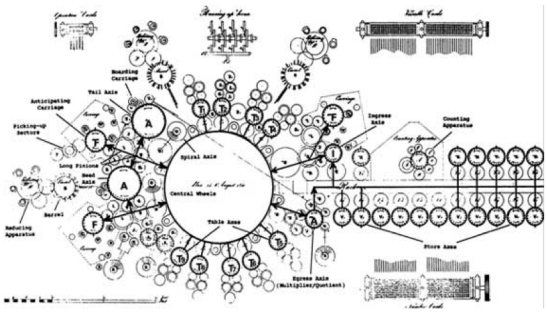
Figura 1.1: Le prime macchine calcolatrici di tipo meccanico

soltanto di dimostrare la verità di ogni proposizione vera, ma anche di costruire nuove proposizioni con la certezza dei procedimenti matematici.

Il primo modello di calcolatore così, come noi lo intendiamo oggi, che fosse cioè in grado di manipolare non solo numeri, ma anche simboli, lo si deve a *Charles Babbage*, matematico, filosofo e ingegnere britannico, il quale descrisse nel 1834 il progetto della *Macchina Analitica*, modello per tutti i successivi calcolatori digitali universali (fig.1.2). La macchina non fu mai realizzata per le difficoltà legate alla complessità meccanica delle



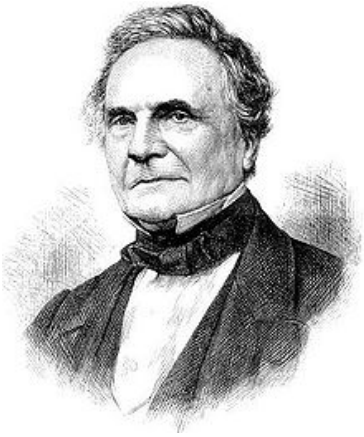
(a) Ricostruzione della *Macchina Analitica*



(b) Schema del 1840 della *Macchina Analitica*

Figura 1.2: La *Macchina Analitica* dell'ingegnere inglese *Charles Babbage*

sue 25 mila parti, anche perché i concetti sui quali avrebbe basato il suo funzionamento anticipavano di almeno cent'anni il livello tecnologico necessario alla loro attuazione pratica. Per questa macchina egli aveva infatti immaginato la possibilità di introdurre da un lato le "regole" della computazione (che noi oggi chiameremmo *algoritmi*) e dall'altro i valori da associare alle variabili e alle costanti, e tutto ciò impiegando schede o nastri perforati del tutto simili a quelli usati nei telai tessili di *Jacquard* fin dai primissimi anni dell'ottocento. I concetti che stanno alla base della *Macchina Analitica* sono gli stessi usati oggi per i moderni calcolatori elettronici. La macchina era costituita da due parti: la memoria (*Store*) che immagazzinava variabili e costanti e nella quale erano conservati anche tutti i risultati intermedi dei calcoli; l'unità di calcolo (*Mill*) che conteneva il programma vero e proprio. Lo schema generale del suo calcolatore è talmente simile a quello dei computer moderni che la tardiva riscoperta dei



(a) Charles Babbage

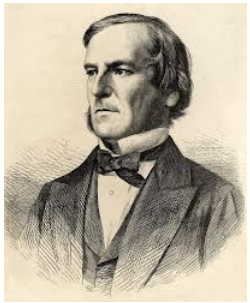
(b) Ada Byron contessa Lovelace

(c) Medaglia commemorativa della *Association for Computing Machinery* (ACM)

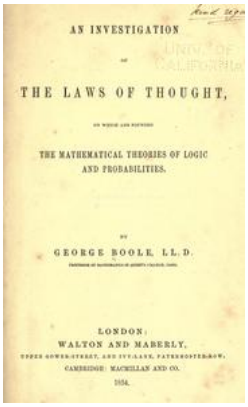
Figura 1.3: Charles Babbage e Ada Byron

suoi scritti invalidò alcuni brevetti della IBM. L'opera di Babbage venne poi esaltata da una singolare nobildonna inglese, *Ada Byron*, contessa di *Lovelace* (figlia del poeta Lord Byron), che per prima intuì l'universalità delle idee espresse da Babbage. Tra i due iniziò un fitto scambio di lettere, piene di numeri, idee, fatti e fantasie e nel 1843, in uno scritto ormai famoso, Ada Byron descrisse le possibili applicazioni della macchina nel calcolo matematico, ipotizzando persino il concetto di *Intelligenza Artificiale* e affermando che la macchina, quando realizzata, sarebbe stata cruciale per il futuro della scienza. A titolo di esempio spiegò il modo in cui la macchina avrebbe potuto effettuare il calcolo dei *numeri di Bernoulli*, e così facendo scrisse quello che viene unanimemente riconosciuto come il primo *programma per computer* della storia. In onore di Ada Byron, nel 1979 il Dipartimento della Difesa degli Stati Uniti battezzò *Ada* un linguaggio di programmazione che era stato appena realizzato.

Nel frattempo *George Boole* (fig.1.4a), logico e matematico britannico, cominciò a lavorare sullo strumento concettuale che sta alla base del funzionamento dei moderni calcolatori, cioè la logica binaria, o *Logica Booleana*, scrivendo l'opera "*An investigation of the Laws of Thought*" (fig (1.4b)). Si tratta di un calcolo logico a due valori



(a) George Boole


AN INVESTIGATION
OF
THE LAWS OF THOUGHT,
ON WHICH ARE FOUNDED
THE MATHEMATICAL THEORIES OF LOGIC
AND PROBABILITIES.
BY
GEORGE BOOLE, LL.D.
LONDON:
WALTON AND MABERLY,
CAMBRIDGE: MACMILLAN AND CO.

(b) Il frontespizio dell'opera *An Investigation on the Law of Thought*, di *George Boole*

AND			OR			NOT	
X	Y	$Z = X \cdot Y$	X	Y	$Z = X + Y$	X	$Z = \overline{X}$
0	0	0	0	0	0	0	1
0	1	0	0	1	1	1	0
1	0	0	1	0	1		
1	1	1	1	1	1		

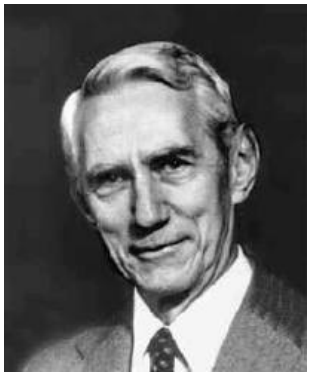
(c) Le tabelle di verità per le funzioni Booleane AND, OR e NOT

Figura 1.4: George Boole, il padre della Logica Booleana

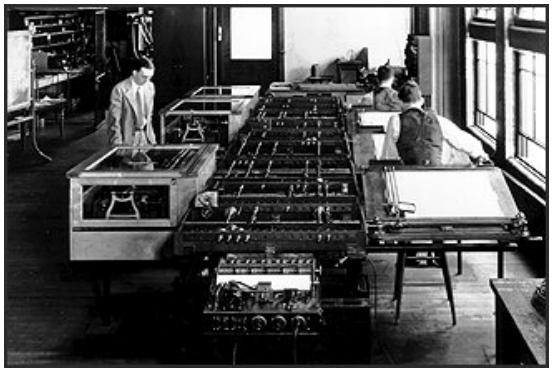
di verità, *Vero* e *Falso*, che consente di operare su proposizioni allo stesso modo in cui si opera su entità matemati-

che. Nel suo lavoro Boole mostrò che la logica Aristotelica può essere rappresentata tramite equazioni algebriche. Boole sviluppò i concetti precedentemente espressi da Leibniz sul sistema binario e descrisse gli operatori logici che da lui presero poi il nome di *Operatori Booleani* (AND, OR, NOT), oggi attuati circuitalmente mediante le cosiddette *porte logiche*.

Il lavoro di Boole fu considerato però d'interesse solo matematico-speculativo, almeno fino al 1937, anno in cui *Claude Elwood Shannon* (fig.1.5a), matematico e ingegnere americano, pubblicò la sua tesi di master intitolata *A Symbolic Analysis of Relay and Switching Circuits*. Shannon stava lavorando al MIT sotto la direzione di *Vannevar Bush*, inventore dell'*Analizzatore Differenziale* (fig.1.5b), il primo calcolatore *analogico* per risolvere equazioni differenziali (1930); in particolare egli era interessato alla teoria e alla progettazione dei complessi circuiti di relay che controllavano le operazioni della macchina di Bush.



(a) Claude Elwood Shannon



(b) L'*Analizzatore Differenziale* di *Vannaver Bush* del MIT

Figura 1.5: Claude Elwood Shannon nel laboratorio del MIT

Fu in questo contesto che si rese conto che la Logica Booleana, così come si applicava alla rappresentazione di *Vero* e *Falso*, poteva essere usata per rappresentare le funzioni degli interruttori nei circuiti elettrici, il cui funzionamento è caratterizzato da due stati, acceso e spento. Ciò divenne la base della progettazione dell'elettronica digitale, con applicazioni pratiche nella commutazione telefonica e nell'ingegneria dei computer. I meriti di Shannon vanno però ben oltre, poiché il suo nome è indissolubilmente legato ai due celeberrimi articoli "*A Mathematical Theory of Communications*" del 1948, e "*Communication Theory of Secrecy Sysytems*" del 1949, che gettarono le fondamenta della *Teoria dell'Informazione* e della *Crittografia* moderna.

(a) Archimedes

(b) Brunsviga

(c) Curta

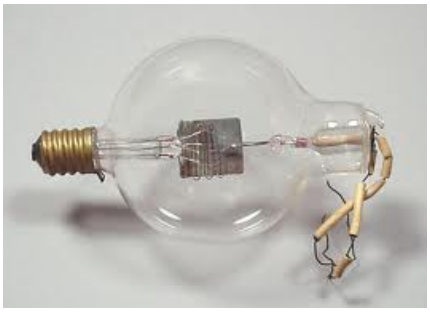
Figura 1.6: Alcuni modelli di macchina calcolatrice meccanica di fine 800, inizi 900

Alla fine dell'ottocento le calcolatrici erano ancora meccaniche, e non essendo stata ancora realizzata la Macchina Analitica prefigurata da Babbage, non esisteva alcuna macchina "programmabile". Nella figura 1.6 ve-

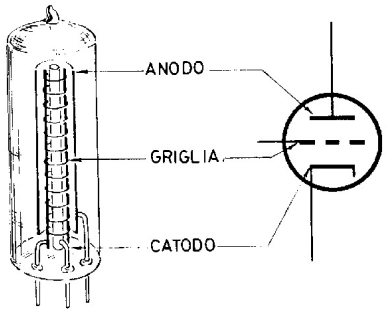
diamo alcuni esempi di calcolatrici in uso all'epoca; tra queste la *Brunsviga*, che ebbe una diffusione notevole e la *Curta*, vero e proprio gioiello nell'arte della meccanica, prodotta fino al 1943.

I primi anni del '900 furono determinanti per il trapasso tra la tecnologia elettromeccanica e quella elettronica dei *tubi termoionici*, che nasce con l'invenzione nel 1904 del *diodo a vuoto* (fig.1.7a), ad opera dell'ingegnere inglese *Sir John A. Fleming*; due anni più tardi l'americano *Lee de Forest*, aggiungendo un terzo elettrodo al diodo di Fleming, la *griglia*, crea il primo *triodo* a vuoto (fig.1.7b), che consente di amplificare un segnale analogico, ma anche di fungere da interruttore comandato in tensione (senza dispendio di potenza), sostituendo così i lenti e pesanti *relay* elettromeccanici, che necessitano per altro di una rilevante potenza per il controllo. La strada è segnata, anche se l'impatto della nuova tecnologia nell'ambito delle macchine da calcolo non sarà immediato, a causa dei problemi di affidabilità ancora presenti.

(a) Diodo a vuoto di *Fleming* (1904)



(b) Triodo a vuoto di *De Forest* (1906)

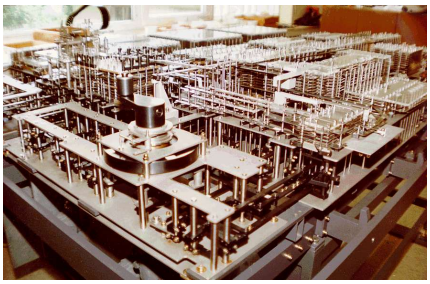


(c) Struttura schematica del triodo

Figura 1.7: I tubi termoionici inventati nei primi anni del 900

Il 1936 è l'anno in cui l'ingegnere tedesco *Konrad Zuse* (fig1.8a) inizia la costruzione del primo calcolatore moderno, la macchina logica "*V1*", successivamente ribattezzata "*Z1*" per evitare qualsiasi riferimento ai tristemente noti razzi V1 tedeschi (fig1.8b). Si tratta di un calcolatore meccanico realizzato artigianalmente e con mezzi rudimentali dallo stesso Zuse, nella propria abitazione (fig1.8c). Il prototipo rappresenta la prima macchina al mondo, basata su codice binario, completamente programmabile. Zuse, convinto che i programmi composti da combinazioni di bit potessero essere memorizzati, chiese anche un brevetto in Germania per l'esecuzione automatica di calcoli.

(a) L'ingegnere tedesco *Konrad Zuse*, che costruì il primo calcolatore nel 1936



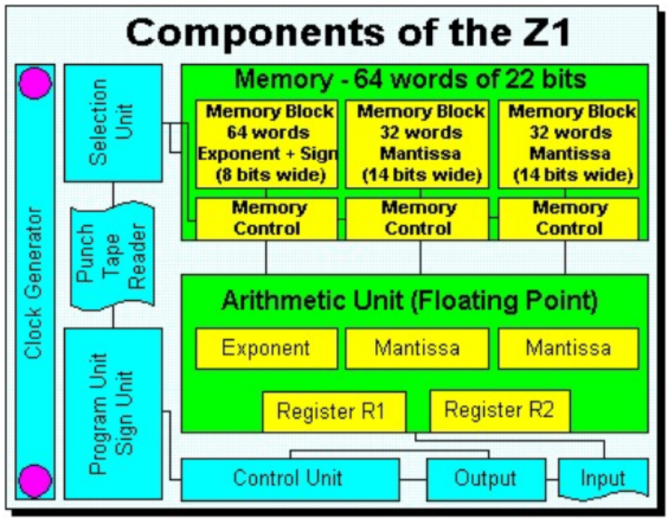
(b) Il primo calcolatore del mondo, lo Z1, del 1937

(c) Soggiorno della casa di Zuse dove venne costruito lo Z1

Figura 1.8: Lo Z1 venne distrutto subito dopo la costruzione, a seguito di un bombardamento di Berlino

Lo *Z1* era un apparecchio programmabile, in grado di processare numeri in formato binario e le cui caratteristiche più apprezzabili, viste con il senno di poi, furono la netta distinzione fra memoria e processore. Questa architettura (fig1.9a), che non venne adottata dall'*ENIAC* o dal *Mark I*, (i primi computer realizzati negli Stati Uniti

quasi dieci anni più tardi), rispecchia l'architettura del calcolatore ipotizzata solo nel 1945 da *John von Neumann*. Lo *Z1* conteneva tutti i componenti di un moderno computer, anche se era completamente meccanico, come ad esempio le unità di controllo, la memoria, la rappresentazione a virgola mobile, ecc. Aveva una frequenza di lavoro di 1 *Hertz*, era in grado di effettuare una moltiplicazione in 5 secondi, disponeva di 64 celle di memoria a 22 bit e usava al posto dei *relay* circa 20.000 piastre in metallo (fig.1.9b). Il calcolatore venne poi distrutto assieme ai progetti dai bombardamenti di Berlino, durante la seconda guerra mondiale, ma nel 1941 venne costruita la sua terza versione, denominata *Z3* (figura 1.10a), che diventerà operativo per qualche tempo, prima di essere a suo volta distrutto da un bombardamento.



(a) Architettura dello Z1

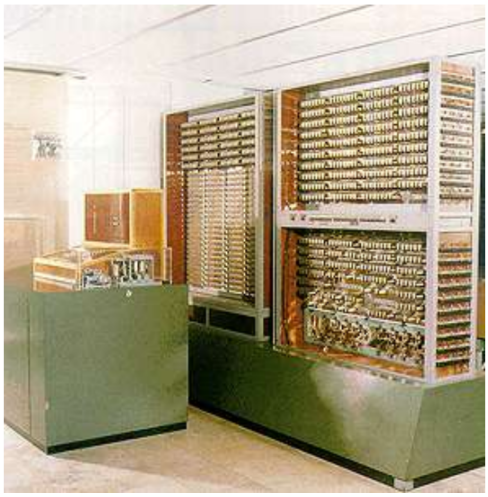
Data sheet

Name of the machine	Z1
Implementation	Thin metal plates, worked with fret saw
Frequency	1 Hertz
Numeric unit	Floating point unit, 22 Bit word length
Average calculation speed	Multiplication approx. 5 seconds
Input	Decimal keyboard, automatic binary coding
Output	Decimal digits
Word length	24 Bit mantissa, 8 bit exponent, 1 sign
Number of relays	No relays, thousands of metal plates, approx. 20,000 parts
Memory	64 cells à 22 Bit
Power consumption	Approx. 1000 watts for the electric cycle motor
Weight	Approx. 500 kg
Area of application	Experimental model, no application, developed for scientific calculations
Number of machines sold	0
Cost in DEM	No price
Comments	The Z1 was not reliable. A

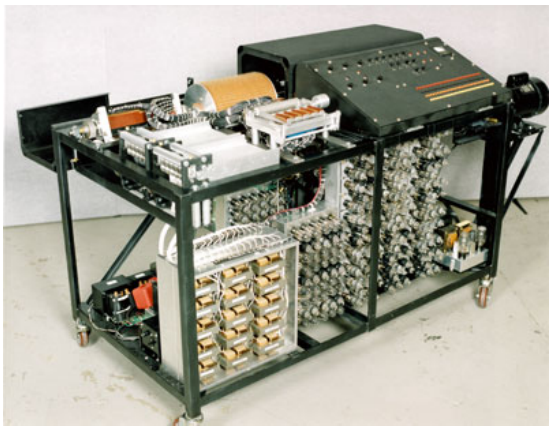
(b) Le principali caratteristiche tecniche dello Z1

Figura 1.9: Architettura e principali caratteristiche dello Z1, basato su una tecnologia puramente meccanica

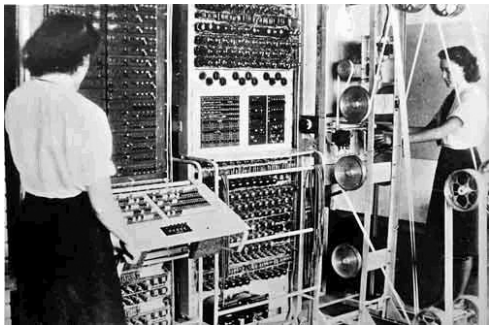
Negli Stati Uniti inizia nel 1939 il progetto dell'*Automatic Sequence Controlled Calculator* (*ASCC*) della IBM, che in seguito verrà ceduto all'università di Harvard e prenderà il nome di *Mark I* (fig.1.10d). Quasi contemporaneamente parte anche il progetto del calcolatore "*ABC*" di *J.V. Atanasov* e *C. Berry* (fig.1.10b), sul quale si sarebbe basato successivamente *J.W. Mauchly* per la costruzione dell'*ENIAC* (fig.1.10e). L'ABC è il primo computer che utilizza la nuova tecnologia dei *tubi termoionici* (o a vuoto). Il prototipo, che realizza somme a 16-bit, non arriverà mai in produzione, ma i concetti contenuti nell'*ABC*, come l'*Unità Aritmetico Logica* (ALU) e la memoria riscrivibile, compariranno nei moderni computer. Negli ultimi anni ci sono state molte controversie su chi avesse veramente inventato il primo computer elettronico digitale, e una corte di giustizia americana decise in favore di Atanasov. Nel Regno Unito si costruisce invece, nel 1943, il *Colossus* (fig.1.10c), progettato per poter forzare i cifrari tedeschi basati sull'impiego della macchina cifrante di Lorenz SZ40/42, in uso al comando generale del III Reich. L'ultimo calcolatore d'interesse storico che menzioniamo è l'EDVAC (fig.1.10f), che fu progettato all'inizio da John von Neumann, il matematico e ingegnere ungherese cui si deve l'odierna architettura dei computer, basata su un *Unità Logico-Aritmetica* (ALU), su dei *registri* di memoria e su una *memoria RAM* per memorizzare i dati e il programma. Diversamente dal suo predecessore decimale ENIAC, l'EDVAC era binario.



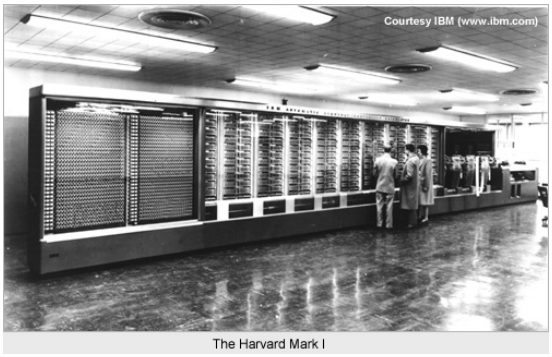
(a) Il calcolatore Z3 di Zuse (Maggio 1941), con un'architettura simile a quella di von Neumann, era basato sulla tecnologia elettromeccanica dei relè



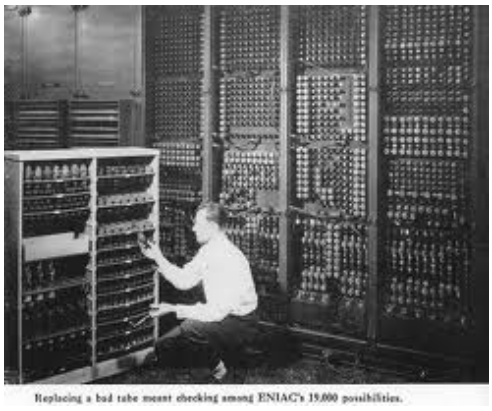
(b) Il primo calcolatore a tubi termoionici, l'ABC di *Atanasov* e *Berry* (Estate 1941)



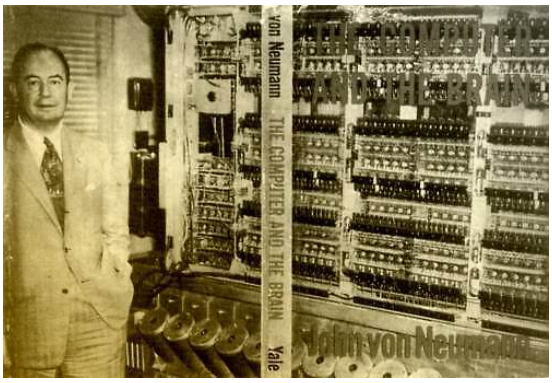
(c) Il calcolatore *Colossus*, prodotto in Inghilterra nel 1943 e usato per la decifrazione della macchina cifrante it Lorenz, in uso al quartier generale di Hitler



(d) Il *Mark I* fu il completamento del progetto *ASCC* realizzato dall'università di Harvard (1944), ma era ancora basato sulla tecnologia elettromeccanica dei relè



(e) Il calcolatore *ENIAC*, basato sulla tecnologia a tubi termoionici dell'ABC (1946-48)



(f) L'EDVAC fu uno dei primi calcolatori con l'architettura ideata da *von Neumann* (1951)

Figura 1.10: I calcolatori del periodo 1941-1951, realizzati da Germania, Regno Unito e USA

1.2 Dal programma di Hilbert ai teoremi di incompletezza di Gödel

L'ideazione e la realizzazione delle prime macchine calcolatrici, secondo il processo storico delineato brevemente nel paragrafo precedente, ebbe come spinta propulsiva la necessità di effettuare in modo automatico le quattro operazioni elementari con i numeri (somme, moltiplicazioni, differenze e divisioni). Tuttavia la complessità strutturale via via crescente di tali macchine, che ebbero come capostipite la *Macchina Analitica* di Babbage, trasformò completamente la loro natura: infatti il "programma" di calcolo, inizialmente incarnato negli ingranaggi meccanici o nei circuiti elettromeccanici delle macchine più avanzate, e deputato alla soluzione di un problema specifico, lascia a un certo punto il posto a un "programma" non cablato, che può essere modificato dall'esterno con lo scopo di poter risolvere un problema nuovo, e ciò senza dover riassemblare la macchina. La macchina acquista dunque una flessibilità che le consente di essere usata più volte per risolvere problemi di natura diversa, e ciò senza dover cambiare la sua topologia. Diventa cioè una *macchina programmabile*. All'inizio questi problemi erano legati soprattutto al calcolo di fattori numerici, ma il potere della *codifica simbolica*, ossia la libertà di attribuire un qualunque significato a un simbolo, coniugato con la possibilità di manipolare i simboli in modo logicamente strutturato, portò a disvelare potenzialità inizialmente insospettabili per le macchine sia pur rudimentali dell'epoca.

È a questo punto che il destino di coloro che inseguivano il sogno di una macchina automatica per fare i calcoli s'intreccia con quello di coloro che miravano a una ricostruzione logica e unitaria di tutta quanta la Matematica, che avrebbe dovuto consentire di ricavarne i teoremi in qualsiasi ambito (analisi, geometria, algebra, ecc.) a partire dagli *assiomi* e dalle *regole di inferenza*, secondo un approccio che si inquadra perfettamente col pensiero razionalista e riduzionista di inizio secolo.

Ricordiamo a tal proposito che ogni *Teoria Matematica*, quale ad esempio l'*Aritmetica*, la *Teoria degli Insiemi* ecc., scaturisce da alcune affermazioni iniziali considerate vere e denominate *assiomi* (o *postulati*), e dall'applicazione ad essi di specifiche *regole di inferenza*, che esprimono le modalità lecite per costruire altre affermazioni vere, denominate *teoremi*. In quest'ottica la *Teoria* è l'insieme di tutti gli assiomi, che giocano il ruolo di verità primitive, e di tutti i teoremi che si possono provare usando le regole di inferenza. Ecco ad esempio i celebri *postulati di Peano*, sui quali si fonda l'Aritmetica dei numeri naturali:

Postulati di Peano

1. "0" è un numero;
2. se n è un numero, il suo successore è un numero;
3. "0" non è successore di alcun numero;
4. numeri diversi non possono avere lo stesso successore;
5. se un insieme S di numeri comprende lo 0, come anche il successore di qualunque numero in S, allora S comprende tutti i numeri.

Per quanto riguarda le regole di inferenza, possiamo dire che esse costituiscono i cardini logici del ragionamento matematico; alcuni esempi sono la *Modus Ponens*, la *Modus Tollens* e la *Reductio ad Absurdum*, illustrate sinteticamente dalla tabella sotto riportata. La barra indica che a partire dalle premesse che stanno sopra, si trae la conseguenza che sta sotto:

Modus Ponens	$\frac{A \to B,\ A}{B}$	
Modus Tollens	$\frac{A \to B,\ nonB}{nonA}$	Il *Modus Ponens* si legge così: se da A segue B, e A
Reductio ad Absurdum	$\frac{A \to B,\ A \to nonB}{nonA}$	

è vero, allora è vero anche B; analogamente le altre.

Il rappresentante sommo dell'impostazione riduzionista prima citata fu il grande matematico tedesco *David Hilbert* (1862–1943, fig1.11a). Al *Secondo Congresso Internazionale di Matematica* di Parigi del 1900, Hilbert tenne un intervento di portata storica - probabilmente la più influente conferenza matematica di ogni tempo - proponendo un elenco di 23 problemi aperti che, a suo giudizio, costituivano una sfida per i matematici del secolo a venire. La tabella di figura 1.12 riporta l'elenco completo. La natura di questi problemi era varia e disomogenea: se

(a) David Hilbert (b) Kurt Gödel (c) Alan Turing

Figura 1.11: La formulazione dei 23 problemi di Hilbert consentì a Gödel e a Turing di sviluppare le riflessioni che portarono alla fine alla formulazione del primo modello di computazione nel 1936

alcuni erano molto specifici e tecnicamente ben delineati (p.es. il Problema 3, che venne risolto immediatamente), altri (p.es. il Problema 6, sull'assiomatizzazione della Fisica, o il Problema 4) erano troppo generali o troppo vaghi per ammettere una risposta incontrovertibile. Altri ancora, i Problemi 1, 2 e 10, portarono a una soluzione inaspettata per Hilbert: essi ci riguardano da vicino, per la loro stretta connessione con i fondamenti della *Teoria della Computabilità*, e quindi con un inquadramento formale dei fondamenti dell'Informatica. Data la loro importanza li esaminiamo a parte.

1^o Problema - Ipotesi del continuo (IC)

Non esiste una cardinalità intermedia tra quella dei naturali e quella dei reali.

Si tratta di accertare se esista un insieme S (infinito) dotato di cardinalità inferiore a quella dei reali e superiore a quella dei naturali. Nel 1938 *Kurt Gödel* stabilì che IC non è refutabile nell'ambito assiomatico della teoria (di Zermelo-Fraenkel) degli insiemi; nel 1963, *Paul Cohen* stabilì che in tale ambito non è neppure dimostrabile.

2^o Problema - Assiomi dell'aritmetica

Accertare che gli assiomi dell'aritmetica sono consistenti.

Gödel dimostrò (1931, *1^o Teorema di Incompletezza*) che in qualunque sistema assiomatico sufficientemente espressivo da contenere almeno l'aritmetica, si può individuare un enunciato circa i numeri naturali che:

non può essere nè dimostrato nè refutato all'interno del sistema (sistema *incompleto*);
oppure
può venire sia provato che refutato all'interno del sistema (sistema *inconsistente*).

Problema	1	risolto (1963)	*L'Ipotesi del Continuo*
Problema	2	risolto (1930)	*Consistenza degli assiomi dell'aritmetica*
Problema	3	risolto	Uguaglianza di volumi tra tetraedri
Problema	4	troppo vago	Costruzione di tutte le metriche con linee geodetiche
Problema	5	risolto	Differenziabilità dei gruppi continui di trasformazioni
Problema	6	aperto	Assiomatizzazione della Fisica
Problema	7	risolto	Trascendenza di a^b, con $a \neq 0, 1$ e b irrazionale
Problema	8	aperto	Ipotesi di Riemann e congettura di Goldbach
Problema	9	parzialm. risolto	Trovare la legge più generale di reciprocità in un qualunque campo algebrico numerico
Problema	10	risolto (1970)	*Risolubilità delle equazioni Diofantee*
Problema	11	parzialm. risolto	Forme quadratiche con coefficienti numerici algebrici
Problema	12	aperto	Estensioni di campi numerici algebrici
Problema	13	risolto	Risoluzione di equazioni di 7-imo grado usando funzioni di due argomenti
Problema	14	risolto	Dimostrazione di finitezza di certi sistemi completi di funzioni
Problema	15	parzialm. risolto	Fondamenti del calcolo enumerativo di Schubert
Problema	16	aperto	Topologia di curve e superfici algebriche
Problema	17	risolto	Espressione di funzioni razionali definite come quozienti di somme di quadrati
Problema	18	risolto	Riempimento spaziale tramite poliedri non regolari
Problema	19	risolto	Analiticità delle soluzioni di Lagrangiani
Problema	20	risolto	Risolvibilità di ogni problema variazionale fissate certe condizioni al contorno
Problema	21	risolto	Esistenza di equazioni differenziali lineari aventi un gruppo monodromico assegnato
Problema	22	risolto	Uniformizzazione di relazioni analitiche per mezzo di funzioni automorfiche
Problema	23	risolto	Sviluppi ulteriori del calcolo variazionale

Figura 1.12: I 23 Problemi di Hilbert

In altre parole *ogni sistema assiomatico sufficientemente espressivo è o inconsistente o incompleto*. Se escludiamo la prima eventualità, possiamo esprimere il teorema più semplicemente, dicendo che non è detto che un enunciato vero sia un teorema, e cioè che discenda dagli assiomi tramite le regole di inferenza del sistema.
Da qui Gödel dedusse (*2^o Teorema di Incompletezza*) che quando un sistema assiomatico è consistente e sufficientemente espressivo da contenere almeno l'aritmetica, esso *non può* provare la propria consistenza. Ciò fornisce una risposta, per quanto negativa e imprevista, al 2^o problema di Hilbert.

10^o Problema - Risolubilità delle equazioni Diofantee.

Trovare una procedura in grado di stabilire se una qualunque equazione Diofantea assegnata ammette soluzioni intere.

Un'equazione Diofantea è un'equazione polinomiale $p(x_1, x_2, ..., x_n) = 0$ a coefficienti interi, che s'intende risolvere assegnando valori interi alle incognite x_i. *Yuri Matiyasevich* dimostrò, nel 1970, che una procedura risolutiva generale non può esistere: a meno che non si pongano fortissime limitazioni al numero di incognite o al grado del polinomio; siamo dunque di fronte a un ulteriore importante problema della matematica che viene risolto in senso tanto negativo quanto inaspettato.

È evidente che l'impostazione riduzionista di Hilbert, implicita per altro già negli enunciati dei problemi (nei quali si chiede di trovare *la* soluzione, e non *se* una certa soluzione esista o meno), subì un duro e inaspettato contraccolpo dall'enunciazione dei *Teoremi di Incompletezza* di Gödel, che rimangono sicuramente una delle più importanti scoperte matematiche del '900. Essi gettarono lo scompiglio tra le fila dei matematici dell'epoca, poiché l'idea che qualcosa di matematicamente vero potesse non esser dimostrabile implicava un ridimensionamento essenziale, anche se circoscritto a singoli problemi, nella capacità argomentativa del metodo matematico basato sull'approccio *ipotetico-deduttivo*. D'altra parte il programma riduzionista, chiaramente perseguito anche dai matematici *Gottlob Frege* e *Giuseppe Peano*, aveva già subito qualche incrinatura, soprattutto ad opera di *Bertrand Russell* e in singolare contemporaneità con l'elencazione dei 23 problemi di Hilbert. Il suo famoso *paradosso* aveva destabilizzato l'opera di *Frege*, aprendo un periodo di crisi dei fondamenti logici della matematica. Tale paradosso riguarda *gli insiemi che non sono membri di sé stessi*. A prima vista la loro stessa definizione potrebbe sembrare mal posta; e in effetti, se prendiamo come riferimento un insieme di numeri, esso non è un numero, per cui sembra privo di senso chiedersi se appartenga o meno a sé stesso. Tuttavia,, tanto per fare un esempio, l'insieme degli argomenti trattati in questo paragrafo è esso stesso un argomento (ne stiamo parlando ora!), e dunque è un insieme che appartiene a sé stesso.

L'antinomia di Russell

Se chiamiamo T l'insieme di tutti gli insiemi che non appartengono a sé stessi si ha:

se $T \in T$ allora $T \notin T$, poiché T contiene per definizione solo insiemi che *non appartengono a sé stessi*

se $T \notin T$ allora $T \in T$, poiché T contiene per definizione tutti gli insiemi che *non appartengono a sé stessi*

Il paradosso aveva famosi precedenti storici, quali il *Paradosso del mentitore*, attribuito ad *Eubulide di Mileto* (filosofo greco del IV secolo A.C.): *Un uomo dice che sta mentendo. Sta dicendo il vero o il falso?* Di questo paradosso è nota anche la variante *Questa frase è falsa*, e una sua versione precedente, attribuita ad *Epimenide*, cretese: *I cretesi son tutti bugiardi*, che non sembra però essere stata scritta con l'intento di illustrare un paradosso. Tuttavia il *Paradosso del mentitore* è una contraddizione logica che gioca sull'autoreferenzialità in un contesto, come quello linguistico, che non è formalizzato matematicamente; infatti la spiegazione più semplice consiste nell'assumere che ogni frase pronunciata (o scritta) esprima implicitamente un'affermazione di verità sull'oggetto della frase stessa, per cui la frase *Questa frase è falsa* andrebbe letta in realtà come *Questa frase è vera e questa*

frase è falsa, il che corrisponde all'enunciazione di una semplice contraddizione del tipo A *e non* A, che è falsa. Nel caso del paradosso di Russell le cose erano invece molto più compromesse: il suo argomento evidenziava che una teoria matematica proposta come fondamentale, la *Teoria Elementare degli Insiemi* di *Cantor* nell'assetto formale raggiunto a fine '800, era minata da irriducibili contraddizioni interne.

Nel 1908 *Ernst Zermelo* riuscirà a sanare l'antinomia di Russell, impostando un nuovo sistema noto oggi come *Teoria assiomatica degli Insiemi di Zermelo-Fraenkel*; ma con l'effetto destabilizzante causato dai teoremi di Gödel, il programma Hilbertiano, teso alla riorganizzazione di tutta la matematica in un edificio formale che avrebbe dovuto autocertificare la propria consistenza, dovrà venire definitivamente archiviato.

1.3 La nascita dell'Informatica

Sul solco delle riflessioni inerenti gli aspetti logico-fondazionali della Matematica sopra evocati, si sviluppò quella corrente di pensiero che riuscì in seguito a delineare il nucleo fondante dell'Informatica, cioè la *Teoria della Computabilità*, intesa come studio, modellizzazione e individuazione dei limiti relativi all'approccio computazionale basato sulle *procedure effettive* o *algoritmi*. Di nuovo lo spunto iniziale partì da Hilbert, che nel 1928 scrisse, con W. Ackermann, il libro "*Grundzüge der theoretischen Logik*"; in quest'opera compare per la prima volta l'enunciazione del famoso *Entscheidungsproblem* (Problema della decisione) per la *Logica dei predicati (del Primo Ordine)*, cioè per il sistema formale che incorpora la logica classica basata sugli operatori *and* ($\wedge$), *or* ($\vee$), *not* ($\neg$), *implica* ($\Longrightarrow$), *per ogni* ($\forall$), *esiste* ($\exists$). Per capire il significato del *Entscheidungsproblem*, ricordiamo che in tale sistema formale si possono formare delle formule, le cosiddette *formule ben formate*, come per esempio

$$(\exists F)\{(F(a) = b) \wedge (\forall x)[p(x) \implies (F(x) = g(x, F(f(x))))]\}$$

che si legge come "esiste una funzione F tale che $F(a) = b$ e tale che $\forall x$, se è vero il predicato $p(x)$, allora $F(x) = g(x, F(f(x)))$". Ciascuna formula è suscettibile di una *interpretazione*, che consiste nell'assegnazione delle funzioni, delle variabili e delle costanti. Per esempio, assegnando $f(x) = x - 1$ e $g(x) = xy$ sui numeri naturali, l'interpretazione diventa:

Interpretazione 1: $f(x) = x - 1$ e $g(x, y) = xy$

$$(\exists F)\{(F(0) = 1) \wedge (\forall x)[x > 0 \implies (F(x) = xF(x - 1)]\}$$

che si legge come "esiste una funzione F tale che $F(0) = 1$ e tale che $\forall x$, se $x > 0$ allora $F(x) = xF(x - 1)$; tale interpretazione è vera, poiché corrisponde alla funzione fattoriale. Viceversa, l'interpretazione seguente

Interpretazione 2: $f(x) = x$ e $g(x, y) = y + 1$

$$(\exists F)\{(F(0) = 1) \wedge (\forall x)[x > 0 \implies (F(x) = F(x) + 1]\}$$

risulta evidentemente falsa. Una formula si dice allora *valida* se è vera in tutte le interpretazioni. L'oggetto del *Entscheidungsproblem* riguarda proprio la validità delle formule nella logica dei predicati.

Entscheidungsproblem

Trovare una procedura effettiva (algoritmica) per decidere se una qualunque formula nella logica dei predicati è valida (p.es. se una qualunque formula dell'aritmetica è un teorema, cioè derivabile dagli assiomi mediante le regole di inferenza).

Il problema fu risolto indipendentemente da *Alonzo Church*, che pubblicò nel 1936 un articolo intitolato "*An Unsolvable Problem in Elementary Number Theory*", e da *Alan Turing* (fig 1.11c), che nello stesso anno pubblicò

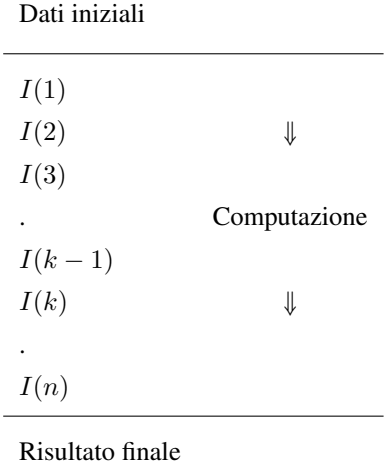
l'articolo "*On Computable Numbers, with an Application to the Entscheidungsproblem*". Essi dimostrarono, con argomentazioni molto diverse, *la non esistenza di un siffatto algoritmo*. Pertanto, in particolare, è impossibile decidere algoritmicamente se un qualunque enunciato sui numeri naturali è vero o meno. Il lavoro di Church fu l'atto di nascita di un formalismo matematico, denominato λ-*calcolo*, che costituisce un vero e proprio *modello di computazione*. Tuttavia l'approccio di Turing, basato su un semplice dispositivo chiamato *macchina di Turing* (MdT), e che oggi riconosciamo come la descrizione del primo modello formale di calcolatore, risultò subito molto più convincente e credibile, al punto che Gödel stesso rimase inizialmente dubbioso sulla correttezza del λ-*calcolo*, ma immediatamente convinto dal modello di Turing. La *Macchina di Turing* incarna implicitamente la prima definizione del concetto di *algoritmo*, al punto che oggi la stretta corrispondenza tra ciò che si considera intuitivamente calcolabile mediante una procedura effettiva di tipo algoritmico e la *Macchina di Turing* costituisce il nucleo forte della cosidetta *Tesi di Church-Turing*. Turing risolse (negativamente, come accennato sopra) l'*Entscheidungsproblem* facendo riferimento al problema della *fermata della macchina di Turing*, e dimostrando che, assegnata una qualunque MdT, non è possibile decidere algoritmicamente se essa si fermerà o meno a partire da certe condizioni iniziali. Il successivo concetto di *macchina di Turing Universale*, cioè di una macchina che sia in grado di simulare la computazione di qualunque altra macchina, getta poi le basi teoriche del calcolatore programmabile.

L'impetuoso sviluppo dell'informatica e il suo rapido affermarsi come disciplina a sé non si può dunque ricondurre solamente alla riflessione millenaria sui limiti del procedimento ipotetico-deduttivo o all'incontro fra questa componente del pensiero matematico e la tecnologia elettronica: la nuova disciplina si sviluppò anche a partire da nuove idee fondanti, la più importante delle quali è proprio quella di *macchina programmabile*, per l'espletamento dei piú diversi compiti senza modifiche della sua architettura fisica. Il sogno di Leibniz di una logica universale, il fervore progettuale di Babbage e i lavori fondamentali di Turing e Church, hanno instradato il pensiero scientifico verso la conquista di un concetto esplicito di *computabilità*, che ha affiancato la realizzazione dei primi calcolatori. Altra idea-chiave, ben manifesta nei progetti di Turing e Zuse, è che l'autoreferenzialità - tradizionale fonte di intriganti paradossi - può essere sfruttata anche in senso positivo. Non c'è una ragione per cui i programmi debbano risiedere all'esterno del calcolatore (come avveniva per i nastri perforati rispetto ai telai Jacquard); un programma caricato in memoria, viceversa, potendo non solo indirizzare le azioni del computer, ma anche subire modifiche per effetto delle sue elaborazioni, avrebbe la duplicità di ruolo necessaria all'apprendimento automatico e alla gestione delle altre tematiche proprie dell'*Intelligenza Artificiale*. Mentre questa seconda idea tarda a dispiegare tutte le potenzialità presenti nella visione di Turing, l'obiettivo di *universalità* del calcolatore può dirsi largamente raggiunto: in effetti, un modesto *laptop* del giorno d'oggi surclassa di molto i colossali calcolatori realizzati da pionieri dell'informatica quali Zuse e von Neumann, che peraltro suscitavano grandi entusiasmi in chi aveva la consapevolezza delle ambizioni che tali prototipi incarnavano. Sarebbe un vero peccato se proprio oggi, mentre si fa un gran parlare di "informatica pervasiva" (o "*ubiquitous computing*") in quanto aspetti tecnologici particolari dell'informatica sono migrati all'interno di palmari, di dispositivi legati alla casa, all'auto, ecc., l'idea originaria venisse persa di vista a favore di logiche proprietarie e di mercato tendenti a riportare in auge soluzioni *ad hoc* o linguaggi programmativi di nicchia, riducendo di fatto il calcolatore alle funzionalità di una mera calcolatrice cablata, sia pure di tipo sofisticato.

1.4 Il metodo procedurale algoritmico

Come già ricordato il termine informatica deriva da *informatique*, contrazione delle parole francesi *infor*mation auto*matique*. Esse sottintendono l'idea di una elaborazione *automatica* delle informazioni, dove l'automatismo è in qualche misura caratterizzato da un aspetto procedurale della successione di operazioni elementari che vengono applicate all'informazione da elaborare con lo scopo di risolvere un problema del mondo reale. Un tale procedimento parte dunque da un insieme di informazioni fornite dall'esterno al sistema di elaborazione, i cosiddetti *dati iniziali*, che vengono successivamente elaborati secondo una procedura ordinata e a passi, descrivibile in modo preciso ed esauriente come successione finita $I(1), I(2), \ldots, I(n)$ di n *istruzioni elementari*. Il risultato dell'esecuzione dell'istruzione $I(k)$ dipende, oltre che dal tipo di istruzione, anche dai dati d'ingresso allo stesso passo, che sono costituiti tanto da eventuali dati esterni resi disponibili al passo k, quanto dai dati che derivano

dall'esecuzione dell'istruzione $I(k-1)$ al passo precedente. E' evidente che lo svolgimento di una tale elaborazione procedurale, detta anche *computazione*, ha uno scopo preciso, che dal punto di vista oggettivo corrisponde molto spesso alla *risoluzione* di un qualche problema numerico. Il legame col problema del mondo reale viene stabilito grazie al potere simbolico della matematica; in tale contesto ogni simbolo (o numero) può rappresentare un qualunque ente esterno al contesto della computazione, e ciò grazie a un'operazione astratta di *codifica*, che si esplica mediante la definizione di una corrispondenze tra oggetti del mondo interno (simboli o numeri) e oggetti del mondo esterno (enti che a essi corrispondono). La codifica ci consente dunque di trasformare l'enunciazione di un problema del mondo reale in una equivalente di natura simbolica, che va poi risolta mediante il sistema di elaborazione. In ambito informatico, alla procedura in questione viene attribuito il nome di *algoritmo*, la sua esplicitazione rigorosa in un linguaggio comprensibile alla macchina viene chiamata *programma*, mentre si riserva il termine di *computazione* al processo che consiste nell'esecuzione dell'algoritmo (del programma) a partire dai dati iniziali. Il risultato della computazione, cioè i dati generati in uscita al termine della stessa, consente di esprimere in modo codificato la soluzione al problema del mondo reale.



Definizione 1.1. *Dicesi algoritmo una procedura effettiva a passi, descritta in modo preciso ed esauriente da un numero finito di istruzioni elementari $I(1), I(2), \ldots, I(n)$, la quale a partire dai dati iniziali fornisce in un tempo finito i dati finali, che vengono interpretati come la soluzione di un determinato problema. Per computazione s'intende l'esecuzione dell'algoritmo.*

Lo schema che abbiamo testé delineato non è l'unico possibile che consenta di ottenere la soluzione di un problema (il più delle volte di natura matematica) descrivibile secondo un approccio simbolico. Esso è tuttavia quello largamente più usato, e ciò essenzialmente per tre ordini di motivi.
Il primo deriva senza dubbio dal fatto che gli aspetti logico-fondazionali del modello computazionale legato alle procedure effettive furono acquisiti solidamente già nel corso degli anni Trenta, grazie al lavoro dei logici di cui si è accennato precedentemente (Gödel, Church, Turing ecc.). Sulla base di quegli studi si fu in grado di descrivere in termini precisi, cioè in termini matematici, che cosa significhi effettuare una computazione, quali siano i possibili modelli di computazione e quali siano (se esistono) i limiti strutturali all'approccio computazionale. Si riuscì inoltre a stabilire l'equivalenza dei vari modelli proposti, giungendo alla descrizione del modello più generale possibile di elaboratore, la cosiddetta Macchina di Turing (MdT). Essa è in grado di effettuare qualunque computazione, per complessa che possa essere, effettuata da un qualunque elaboratore reale. La definizione della MdT portò inoltre all'enunciazione della celebre tesi di Church-Turing, secondo la quale tutto ciò che si ritiene computabile effettivamente è computabile con una Macchina di Turing. Ricordiamo ancora che le fondamenta teoriche dell'Informatica furono consolidate ben prima che nascessero materialmente i calcolatori, cioè le macchine per eseguire le computazioni in modo efficiente.

Il secondo è legato alla circostanza che tale procedimento è di gran lunga il più immediato e intuitivo. Tutti noi, nella vita quotidiana, adottiamo inconsapevolmente delle strategie di tipo *algoritmico* per risolvere alcuni dei nostri problemi. Per strategia algoritmica intendiamo non solo che essa possa essere esplicitata nei termini precisi di una sequenza di azioni elementari effettive, eseguite le quali si perviene alla soluzione del problema, ma anche che la necessità di dover dare struttura procedurale alla nostra attività ci aiuta a delineare i sotto-problemi che via via siamo chiamati a risolvere, prima di giungere alla soluzione globale. La tecnica algoritmico-procedurale ci offre dunque anche la possibilità di progettare meglio e analizzare con maggior efficacia l'insieme globale delle operazioni elementari da svolgere per conseguire l'obiettivo.
Il terzo ordine di motivi che porta alla popolarità delle procedure algoritmiche è che esse sono state facilmente ed efficacemente cablate nella struttura architetturale delle macchine che devono svolgere materialmente la computazione, cioè degli odierni calcolatori. Vedremo infatti nel seguito che la quasi totalità degli elaboratori oggi impiegati dispone di un'architettura (quella di von Neumann, dal matematico ungherese che per primo la ideò) che si basa sul paradigma di una elaborazione procedurale a passi, di tipo *seriale*, eseguita sequenzialmente lungo la linea temporale da un'unità centrale di elaborazione chiamata *processore*.

Nell'architettura di von Neumann, nota in gergo tecnico anche con la denominazione di *architettura seriale*, il processore esegue una singola istruzione elementare per unità di tempo, e l'intera computazione, anche quando molto complessa, si sviluppa attraverso la concatenazione di un certo numero di passi elementari, basati sul programma che descrive l'algoritmo. La soluzione al problema la si ricava come effetto dell'esecuzione dell'ultima istruzione oppure a seguito di un'uscita forzata dal programma, in quanto la soluzione è stata raggiunta prima.

L'architettura seriale si contrappone alle architetture di tipo *parallelo*, nelle quali la soluzione del problema emerge invece dal comportamento collettivo di un insieme di unità computazionali elementari (per esse si usa talvolta il termine metaforico di *neurone*), il cui comportamento e le cui interazioni non sono descrivibili secondo il paradigma della computazione procedurale. Tali unità computazionali sono connesse in una rete, nella quale ciascun neurone può essere potenzialmente connesso a ciascun altro neurone, dotando così la struttura di un massiccio parallelismo che richiama la topologia usata dalle strutture biologiche nella costituzione del cervello, cioè l'organo che esegue le "computazioni" necessarie all'organismo per la risoluzione dei "problemi" legati alla sua sopravvivenza.
La *computazione parallela* ha avuto dei momenti di grande entusiasmo, che si sono però sempre smorzati a causa della mancanza di un vero e proprio modello matematico generale che fosse in qualche misura la controparte di quello scoperto da Gödel, Church e Turing per l'approccio algoritmico-procedurale. A tutt'oggi non è del tutto chiaro se l'approccio parallelo possa, almeno in linea di principio, collocarsi su un piano essenzialmente diverso rispetto a quello tradizionale, anche se si è ragionevolmente portati a ritenere che la tesi di Church-Turing sia valida anche per le computazioni realizzate secondo una modalità operativa parallela. Da questo punto di vista il passaggio da computazione procedurale a computazione parallela corrisponderebbe, essenzialmente, a una semplice mutuazione della *complessità algoritmica* (cioè del numero di passi elementari necessari per eseguire una determinata computazione, fissati che siano i dati iniziali) con la *complessità strutturale* della rete, che in prima approssimazione si può assumere equivalente alla quantità di memoria necessaria per costruire materialmente la rete stessa.

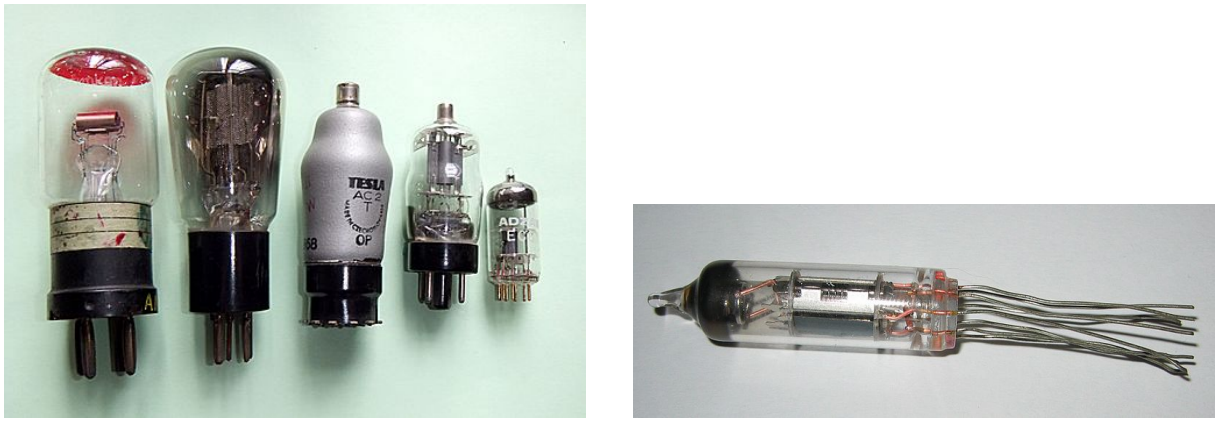
Anche se finora per le computazioni di tipo esatto (cioè quelle che richiedono una soluzione esatta del problema) l'approccio algoritmico si è dimostrato di gran lunga il più efficiente, non si può escludere che in futuro, anche se solo limitatamente a problemi per i quali ci si accontenti di una soluzione approssimata, diventi più conveniente l'impiego delle cosidette *reti neuronali* (artificiali), oppure di altri paradigmi di computazione che hanno suscitato l'interesse degli esperti, quali la *computazione DNA*, gli *algoritmi genetici* e la *computazione quantistica*. A corroborare tale aspettativa c'è comunque il dato di fatto che la natura, nell'organizzare la struttura dei suoi organismi biologici, "ha scelto" il modello parallelo, che gode di una serie di vantaggi derivanti dal fatto che la computazione è sostanzialmente innervata nella struttura della rete. In questo testo ci occuperemo tuttavia del solo approccio algoritmico-procedurale.

In conclusione vale la pena riflettere ancora sulla circostanza, già anticipata nella prefazione, che l'Informatica intesa come studio delle procedure effettive (algoritmi) prescinde dalla tecnologia che s'impiega per l'attuazione delle procedure stesse, al punto che è concepibile, in linea di principio, un'informatica senza calcolatori. La prova di ciò risiede nella comprensione del modello computazionale astratto della macchina di Turing (o di qualunque altro ad esso equivalente) che consente di eseguire una procedura algoritmica facendo uso solo di carta

e penna. Poiché sulla base della tesi di Church-Turing tutto ciò che è computabile nel senso comune e intuitivo del termine lo è anche nel senso della Macchina di Turing, si deduce che tutto ciò che siamo in grado di attuare coi nostri moderni calcolatori sarebbe in linea di principio realizzabile in modo effettivo anche solo con carta e penna. Pur tuttavia gli straordinari successi dell'informatica sono stati resi possibili solo da uno sviluppo impetuoso della tecnologia microelettronica, che rappresenta oggi il sostrato fisico sul quale viene cablata l'architettura del calcolatore. Da questo punto di vista il vero salto di qualità lo si ebbe alla fine degli anni '40, quando fu inventato il *transistor*, che diede il via al processo di miniaturizzazione che consente oggi di disporre, sul palmo di una mano, della potenza di calcolo equivalente a quella di grossi elaboratori che, prima di tale invenzione, occupavano intere ali di edifici consumando decine di kilowatt. Di questa *rivoluzione microelettronica* daremo conto nella prossima sezione.

1.5 La rivoluzione microelettronica e la legge di Moore

L'Informatica odierna è una tecnologia pervasiva, che sta modificando non solo le nostre capacità di elaborazione e di risoluzione di problemi complessi, ma persino le nostre relazioni interpersonali. La possibilità di integrare le capacità di elaborazione dei moderni computer con la connettività offerta da *Internet* sta facendo emergere nuovi paradigmi di partecipazione e di interazione sociale (si pensi al fenomeno dei cosiddetti *Social Network*), che portano profonde modifiche in seno alla società.
Anche se siamo ormai abituati all'idea di stabilire relazioni con altre persone, consultare pagine web o fare fotografie con uno *smartphone* che si porta agevolmente in un taschino, bisogna rendersi conto che tutto ciò non è un prodotto scontato dell'evoluzione tecnico-scientifica. L'acquisizione di un solido modello di computazione, introdotto nei primi anni '30 secondo il processo storico delineato nella sezione 1.3 e la costruzione dei primi calcolatori basati sul modello architetturale di von Neumann, non avrebbero in ogni caso consentito lo sviluppo dell'Informatica che oggi conosciamo se, contemporaneamente, non si fosse realizzata una vera e propria *rivoluzione microelettronica*, che partendo dai primi tubi a vuoto (i diodi e i triodi di figura 1.7), che erano in grado di dare consistenza circuitale alle variabili Booleane necessarie per la costruzione di un computer, ha portato alla costruzione di circuiti integrati ad altissima scala di integrazione (VLSI - *Very Large Scale Integration*), che comprendono milioni o anche miliardi di dispositivi logici equivalenti, dal punto di vista della funzione, ai citati tubi. Diamo allora un breve cenno sull'evoluzione della tecnologia elettronica per comprendere come, dai primi dispositivi a vuoto, si sia giunti agli straordinari *chip* che corredano i nostri computer.

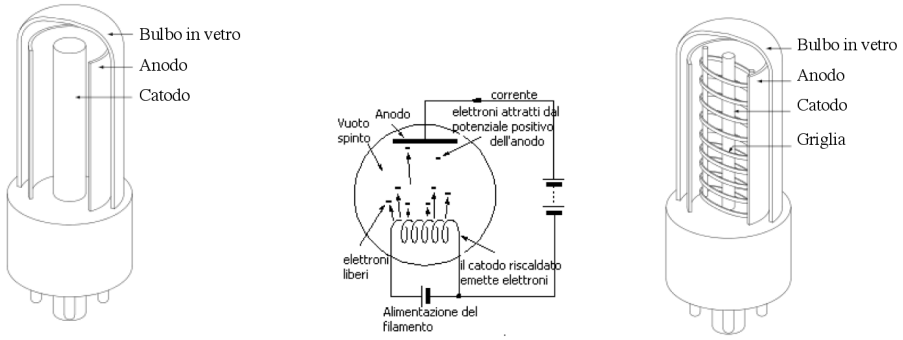


(a) Alcuni tipi di tubi a vuoto o termoionici

(b) Tubo con involucro *subminiatura*

Figura 1.13: Evoluzione dei tubi termoionici, dai primi modelli con zoccolo in bakelite (a), alle ultime realizzazioni degli anni '80 (b), denominate *subminiatura*

Subito dopo il raggiungimento di un'affidabilità adeguata dei tubi a vuoto, che si ottiene solo a partire dalla metà degli anni '20, inizia una produzione industriale su vasta scala, sostenuta inizialmente dalla spinta della tra-



(a) Struttura interna di un diodo a vuoto

(b) Principio di funzionamento del diodo

(c) Struttura interna del triodo; si noti l'elettrodo di controllo, denominato *griglia*

Figura 1.14: Struttura e principio di funzionamento dei tubi a vuoto

sformazione delle prime radio a *galena*, completamente prive di componenti elettronici *attivi*, in radioricevitori ad *amplificazione diretta*, nei quali i tubi termoionici vengono usati per *amplificare* i deboli segnali ricevuti dall'antenna. Lo sviluppo tecnologico porta a una diminuzione dei volumi d'ingombro (si veda la figura 1.13a) e a una integrazione dello zoccolo (inizialmente realizzato in *bakelite*) nel bulbo di vetro. Il culmine della tecnologia dei tubi a vuoto si raggiunge nei primi anni '80, durante i quali vengono realizzati i tubi *subminiatura*, che avevano una lunghezza di circa 3 cm. Il funzionamento dei tubi termoionici è illustrato in figura 1.14. Il tubo è costituito da un bulbo di vetro nel quale viene praticato il vuoto. Un *filamento*, simile a quello di una comune lampadina a incandescenza, riscalda una superficie cilindrica chiamata *catodo* (3.3a), ricoperto di ossidi speciali; l'energia associata al riscaldamento libera un certo numero di elettroni del catodo, che formano una nuvola intorno ad esso; poiché gli elettroni hanno una carica elettrica negativa, essi possono essere attirati da un elettrodo polarizzato positivamente e denominato *anodo*. Questo fenomeno determina un flusso di elettroni che va dal catodo all'anodo, cioè una corrente unidirezionale (fig. 3.3b). Se tra anodo e catodo si interpone un terzo elettrodo, denominato *griglia* (3.6a), il flusso di corrente può essere comandato dal potenziale (negativo) con il quale viene caricata la griglia. Ciò significa che una piccola variazione del potenziale della griglia, che è molto vicina al catodo, può determinare una grossa variazione della corrente anodica, e da ciò nasce l'attitudine del tubo termoionico ad *amplificare* i segnali.

Il principio fisico che sostiene il funzionamento dei tubi a vuoto è anche il principale responsabile dell'impossibilità di scendere, con la miniaturizzazione, al di sotto di certe dimensioni minime; infatti i potenziali anodo-catodo necessari per un corretto funzionamento di un generico tubo a vuoto sono tipicamente dell'ordine delle centinaia di Volt, e non possono scendere in ogni caso al di sotto dei 50-60 V, che rappresenta la minima tensione usata per i tubi subminiatura. Di conseguenza un'ipotetica ulteriore diminuzione dei volumi e delle distanze interelettrodiche comporterebbe delle scariche disruptive tra gli elettrodi, poichè verrebbe superata la tensione di isolamento tra gli stessi. La spinta verso la miniaturizzazione dei tubi a vuoto si fermò dunque ai tubi subminiatura, che non avrebbero consentito neanche la costruzione di calcolatori da tavolo.

Tuttavia già dai primi anni '50 iniziava a farsi strada una nuova tecnologia elettronica, quella dei *semiconduttori*. Nel 1947 *William Shockley, John Bardeen* e *Walter Brattain*, dei laboratori *Bell*, realizzarono il primo prototipo funzionante di *transistor* (che era del tipo *a punta di contatto*, vedi fig.1.15), costituito da un cristallino di *germanio* e da tre elettrodi. Il transistor realizza la stessa funzione dei tubi termoionici, cioè quella di poter controllare, con un elettrodo intermedio denominato *base*, un flusso di corrente che scorre tra altri due elettrodi, l'*emettitore* e il *collettore*. La differenza sta però tutta tra le dimensioni dei due componenti, poiché il cuore del transistor, nella più moderna versione *a giunzione*, è costituito da un microscopico cristallino di germanio o silicio quasi privo di spessore e di meno di un millimetro quadrato, mentre il volume di un tubo è di diversi cm cubici. Il punto di partenza per la costruzione di un transistor è l'impiego della citata *giunzione* PN; essa è costituita da un cristallino di silicio nel quale vengono inseriti, tramite un drogaggio chimico, atomi di elementi diversi, realizzando in tal modo una sezione P e una sezione N. Il silicio ha 4 elettroni nell'orbita esterna; se ad esso vengono aggiunti atomi di un elemento chimico con tre elettroni nell'orbita esterna (p.es. il *Boro* o l'*Indio*) si ottiene un cristallo

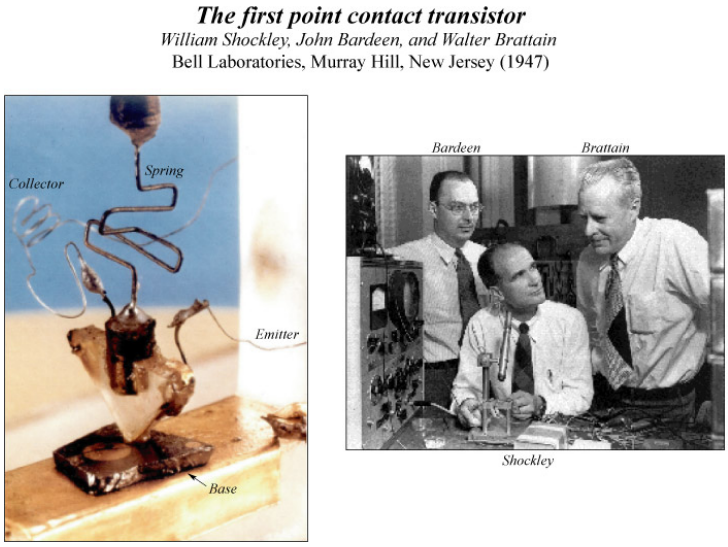


Figura 1.15: Il primo transistor realizzato nei laboratori della Bell da Shockley, Bardeen e Brattain

drogato P. Viceversa, se aggiungiamo atomi di un elemento chimico con cinque elettroni nell'orbita esterna (p.es. l'*Arsenico* o il *Fosforo*) si ottiene un cristallo drogato N. In corrispondenza di ogni atomo dell'elemento drogante di tipo P, poiché ci sono solo tre elettroni nell'orbita esterna, resta disponibile uno spazio vuoto o *lacuna*, che può essere riempita dagli elettroni in eccedenza della giunzione N (vedi fig.1.16a, dove le lacune sono in giallo e gli elettroni in rosso). Se si fornisce una tensione positiva al lato P della giunzione, si instaura un flusso di elettroni da N a P (fig.1.16c), mentre se si inverte la polarità il flusso di corrente si blocca (fig.1.16d). Una giunzione PN ha quindi la funzione di un *diodo*, poiché consente un flusso solo unidirezionale della corrente.

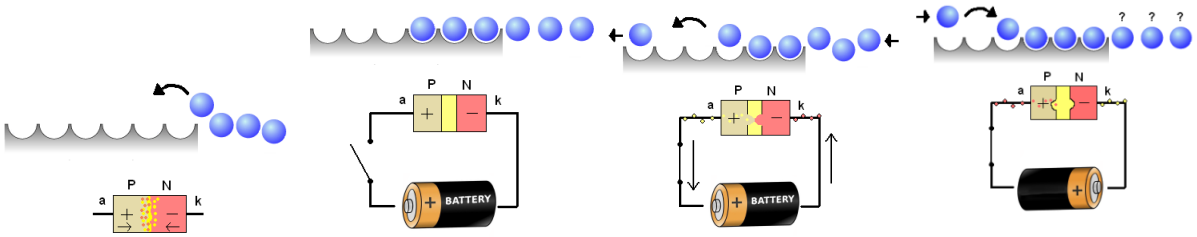
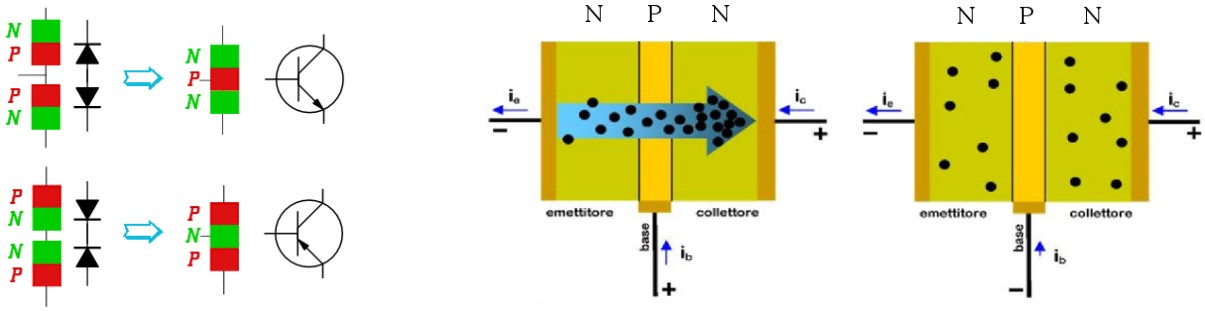


Figura 1.16: Principio di funzionamento della giunzione PN

Il transistor viene realizzato accoppiando due giunzioni PN nei due modi possibili, cioè NPN e PNP (fig. 1.17a). Il funzionamento del dispositivo è brevemente illustrato dalla figura 1.17b. Quando la base è polarizzata positivamente rispetto all'emettitore (negativamente per il caso PNP), la giunzione conduce e una percentuale rilevante di elettroni attraversa la base, che è molto sottile, e perviene al collettore. Se la polarizzazione viene invertita il flusso di corrente si blocca. Il guadagno che ne deriva, dal punto di vista dei volumi occupati dai due dispositivi, è ben evidente in figura 1.18a. Ad esso si aggiunge anche una maggior robustezza rispetto agli urti e la possiilità di impiegare basse tensioni (pochi Volt) al posto delle tipiche tensioni in uso per i tubi, che come detto erano superiori ai 100V.
La tecnologia dei transistor si sviluppa molto rapidamente, e la tecnica *a punta di contatto*, con la quale venne realizzato il primo prototipo, venne subito abbandonata dallo stesso Shockley, che l'anno successivo ideò il transistor a giunzione appena descritto. I primi dispositivi prodotti in larga scala erano realizzati con cristalli di



(a) Transistor a giunzione NPN e PNP

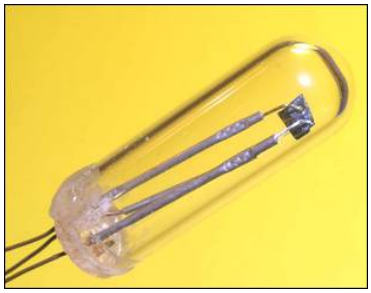
(b) Principio di funzionamento del transistor

Figura 1.17: Transistor e suo principio di funzionamento

germanio, incapsulati in un contenitore di vetro (verniciato in nero) o di metallo per le tipologie che necessitavano un migliore smaltimento del calore (fig.1.18b). La figura (fig.1.18c) ci consente di evidenziare il fatto che quasi tutto l'ingombro, pur minimo, del transistor, serve per poter consentire il collegamento del cristallo con l'ambiente esterno, tramite dei reofori che devono garantire un'adeguata resistenza alle sollecitazioni.

(a) Confronto tra le dimensioni di un tubo termoionico degli anni '80 e di un transistor al germanio per basse potenze

(b) I primi transistor al germanio prodotti su larga scala negli anni '70

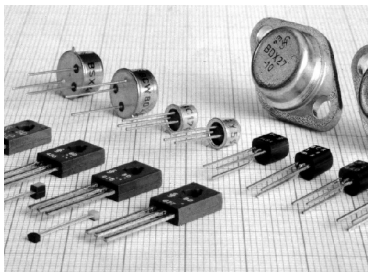


(c) Particolare della giunzione di un transistor al germanio

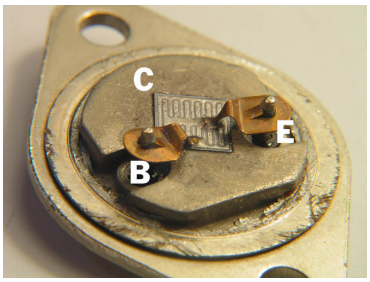
Figura 1.18: I primi transistor prodotti in larga scala negli anni '70 a confronto con la tecnologia dei tubi a vuoto, oramai al tramonto

Il germanio viene presto sostituito col silicio per motivi di affidabilità, in quanto quest'ultimo è molto più resistente alle temperature ($180-200\,^{\circ}$C contro $80-100\,^{\circ}$C) e con una minore deriva termica (aumento della corrente con la temperatura della giunzione). La tecnologia si consolida rapidamente, sviluppando una molteplicità di dispositivi adatti a tutte le applicazioni (fig. 1.19a), anche quando queste prevedono una dissipazione intensa di potenza, per la quale si progettano dei contenitori molto robusti in metallo, per poter connettere il dispositivo a dei dissipatori mediante l'uso di viti (fig. 1.19b).

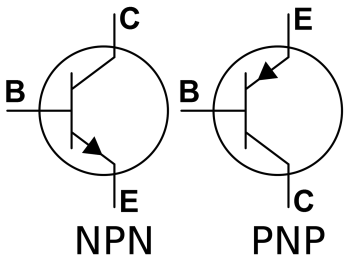
La possibilità di drogare un microscopico cristallino di silicio in modo P e N per realizzare un transistor fa quasi subito prospettare l'idea che su uno stesso sostrato di silicio, denominato *wafer*, fosse in linea di principio possibile realizzare più transistor connessi circuitalmente assieme. E in effetti nel 1958 *Jack Kilby* realizza alla *Texas Instrument* il primo *circuito integrato* della storia, che contiene due transistor connessi su una barretta di germanio (fig.1.20a). Due anni dopo viene annunciato il primo circuito integrato prodotto commercialmente, il multivibratore TI 502 (fig. 1.20b), che contiene 2 transistor, due diodi, 6 resistenze e 4 condensatori (fig. 1.20c) e che costa oltre 500 $. Nel 1965 la stessa *Texas Instrument* realizza un amplificatore integrato di circa 1,5 mm^2, con 7 transistor, che viene prodotto su larga scala con un significativo abbattimento dei costi (fig.1.21a). E' iniziata l'era dell'integrazione su larga scala.



(a) Alcuni esempi di transistor in plastica e metallo

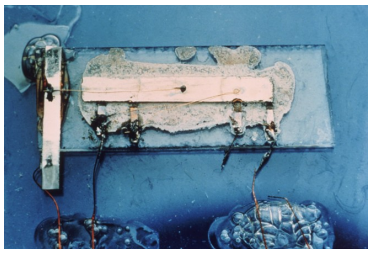


(b) Interno di un transistor di potenza

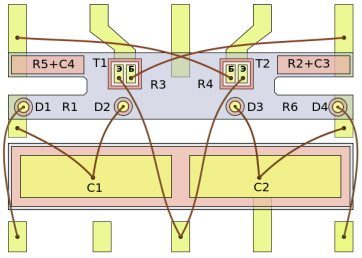


(c) Simbolo elettrico dei transistor NPN e PNP

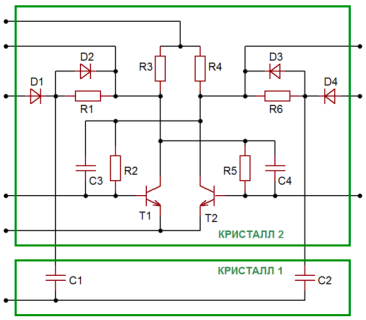
Figura 1.19: Alcuni tipi di transistor e i simboli elettrici relativi



(a) Il primo circuito integrato, realizzato da Jack Kilby nei laboratori della *Texas Instrument*



(b) Struttura interna del multivibratore TI 502

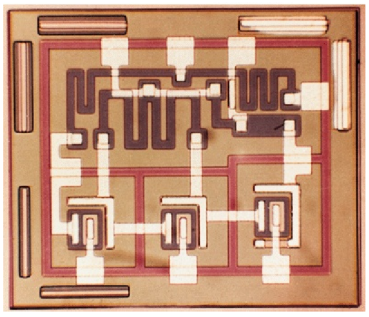


(c) Schema del multivibratore TI 502

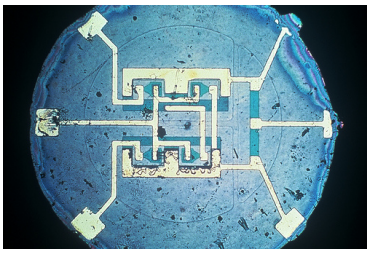
Figura 1.20: Primi circuiti integrati prodotti dalla *Texas Instrument*

Dal punto di vista informatico l'aspetto più rilevante è legato alla possibilità di poter integrare tutti i circuiti necessari per poter realizzare le *porte logiche booleane*, cui si è fatto cenno nella figura 1.4 e che costituiscono il nucleo della circuiteria dei computer. Come esempio si può vedere in figura (1.22) la tabella dei valori di verità per una porta NOR, la sua realizzazione circuitale e il circuito integrato corrispondente. Il funzionamento della porta NOR è il seguente: se entrambi gli ingressi sono posti al valore 0 (False) allora l'uscita è al valore 1 (True); se invece anche uno solo o entrambi hanno valore 1, allora l'uscita va al valore 0. I valori logici 0 e 1 sono realizzati circuitalmente distinguendo tra tensione bassa (0 V) e tensione alta (p.es. 5 V). La linea di sviluppo dei circuiti integrati segue due direttive indipendenti. La prima è quella dei circuiti che amplificano o manipolano un segnale *analogico*, cioè un segnale continuo che si sviluppa nel tempo (p.es. un segnale sonoro nel caso di un amplificatore audio). La seconda è quella delle *porte logiche*, nella quale su ciascun involucro (o *case*) vengono integrate un certo numero di porte AND, OR, NOT, NAND, XOR (vedi fig.1.23). Con questi circuiti logici, che appartengono alla grande famiglie degli integrati TTL (*Transistor-Transistor-Logic*) introdotti sul mercato dalla *Sylvania* nel 1963 e dalla *Texas Instrument* nel 1964, si costruiscono i primi computer personali, quali ad esempio il *Kenbak-1* (fig. 1.23c), del 1971, fino all'Olivetti 6060, che nel 1975 fu uno degli ultimi a usare le porte TTL come elementi costitutivi dell'*Unità Centrale* (o *Central Processing Unit*) del computer.

Nel frattempo, nei laboratori della *Intel*, il progettista capo *Federico Faggin* porta a compimento la realizzazione del primo *microprocessore*, il 4004 (si veda fig.1.24a e 1.24b), che è un circuito integrato di 2300 transistors contenente tutti i principali blocchi logici della CPU in un unico *case*. L'innovazione è notevole, poiché consente di abbattere ulteriormente i costi di produzione dei computer, rendendoli nel contempo meno ingombranti e più affidabili. Al 4004 segue l'anno successivo il modello 8008 (fig.1.24c e 1.24d), che conta ben 3500 transistors. La *Intel* diventa azienda leader nel settore della produzione di microprocessori, e uno dei suoi co-fondatori, *Gordon E. Moore*, osserva in un suo articolo del 1965 che l'evoluzione dei circuiti integrati è stata così veloce da consentire un



(a) Ingrandimento del primo circuito integrato prodotto a basso costo su larga scala



(b) Connessione con l'esterno di un circuito integrato a 6 piedini

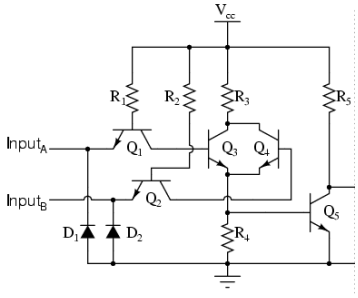
(c) Struttura interna di un circuito integrato per montaggio superficiale

Figura 1.21: Alcuni esempi di struttura interna di circuiti integrati commerciali

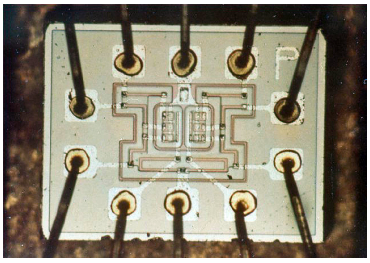
Truth Table

Input A	Input B	Output Q
0	0	1
0	1	0
1	0	0
1	1	0

(a) Tabella di verità della funzione NOR



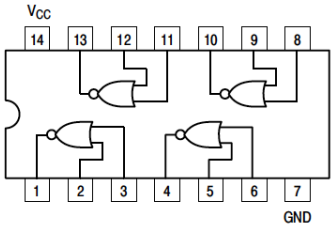
(b) Schema circuitale corrispondente



(c) Circuito integrato che la realizza circuitalmente

Figura 1.22: Circuito integrato che realizza una funzione NOR

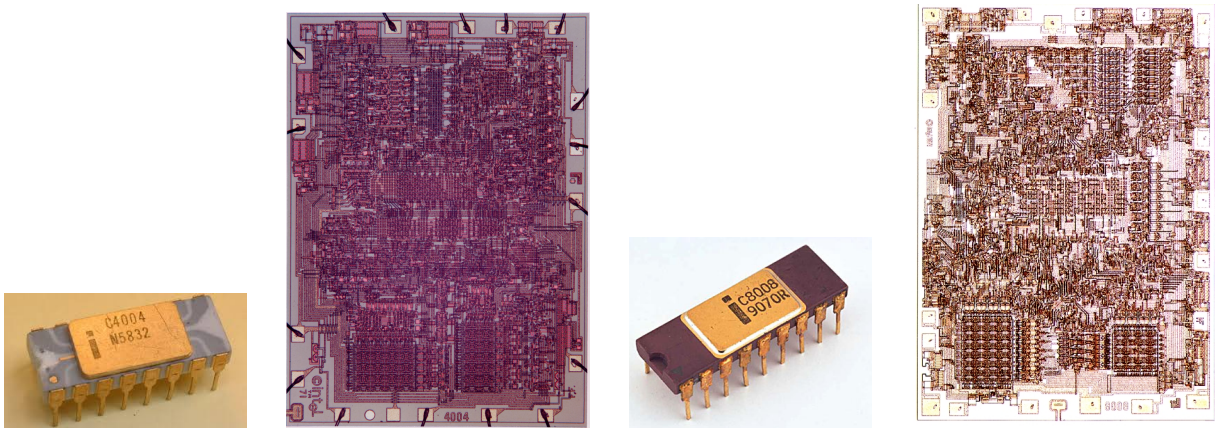
(a) Contenitore per un circuito integrato con quattro porte logiche di tipo NOR



(b) Connessione interna delle porte

(c) Uno dei primi *personal computer*, il *Kenbak*-1

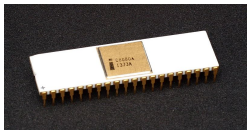
Figura 1.23: La famiglia di circuiti integrati TTL e uno dei primi *computer* che fece uso di questa tecnologia



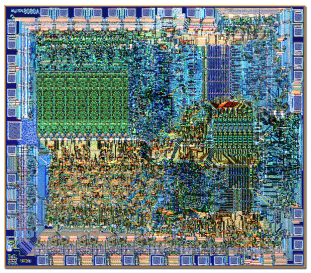
(a) *Intel* 4004 (1971) (b) 4004 con 2300 transistor (c) *Intel* 8008 (1972) (d) 8008 con 3500 transistor

Figura 1.24: *Case* e circuito integrato dei processori 4004 e 8008 della *Intel*

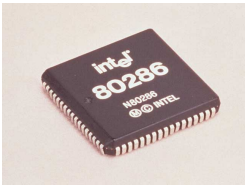
raddoppio del numero di componenti attivi (transistor) ogni (circa) due anni dal 1958, anno della costruzione del primo integrato, al 1965. Moore azzarda anche una previsione, e cioè che una tale legge esponenziale di crescita sarebbe durata ”almeno per una decina d’anni“. Oggi son passati quasi cinquant’anni da quella previsione, e la sua validità non è stata minimamente scalfita dal passare del tempo. Nella figura 1.25 si possono vedere le più importanti famiglie di processori prodotte fino a oggi, e osservando il numero di transistor contenuti in ciascun processore ci si rende conto che l’incremento è stato effettivamente esponenziale; se il 4004 aveva 2300 transistor, i più recenti processori a 8 *core*, cioè con 8 processori fisici montati su un unico contenitore e che lavorano in modo coordinato, si arriva al numero astronomico di 2270 milioni di transistor! La previsione nel prevedere la crescita della scala di integrazione fu talmente precisa che alla stessa fu attribuita, successivamente, la qualifica di *legge di Moore*; si osservi che a tutt’oggi tutte le aziende del settore la usano come elemento di previsione per poter pianificare i progetti futuri e stabilire gli obiettivi della miniaturizzazione. Il diagramma di figura 1.26 ci mostra in ordinata il numero di transistor (in scala logaritmica) mentre nel diagramma sono riportate le varie famiglie di processori che si sono succeduti nella produzione *Intel*. Si può notare che la curva di crescita è effettivamente esponenziale (una retta su scala logaritmica).
I circuiti integrati hanno subìto un abbattimento dei costi e un incremento di prestazioni che non hanno eguali in tutta la storia della tecnologia, poichè il raddoppio dei componenti ogni 24 mesi comporta di fatto un raddoppio delle prestazioni ogni 18 mesi circa. A tutt’oggi la crescita esplosiva garantita dalla legge di Moore non sembra dar segni di cedimento, anche se le dimensioni dei componenti e delle connessioni tra essi cominciano ad essere paragonabili con le dimensioni delle strutture atomiche. Come vedremo nel capitolo relativo agli aspetti architetturali, la crescita esponenziale nel numero di transistor porta con se una serie di conseguenze positive un po’ su tutta l’architettura dei computer, a cominciare dalle memorie, che implica un incremento relativo delle prestazioni ancora maggiore. di quanto ci si aspetti.



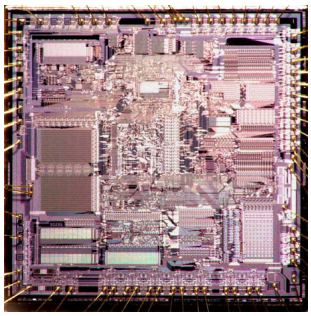
(a) *Intel* 8080 (1974)



(b) 8080 con 4500 transistor

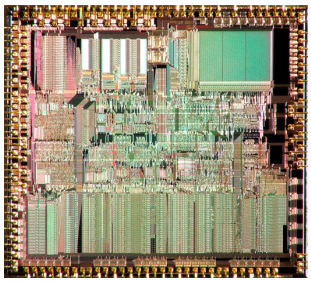


(c) *Intel* 80286 (1982)



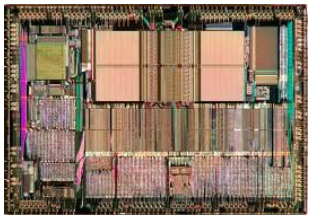
(d) 80286 con 134000 transistor

(e) *Intel* 80386 (1985)



(f) 80386 con 275000 transistor

(g) *Intel* 80486 (1989)

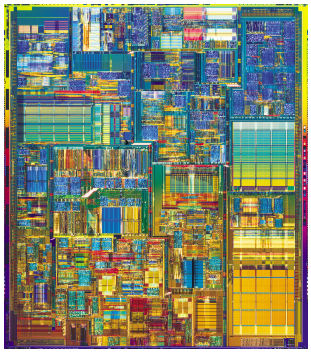


(h) 80486 con 1,2 M di transistor

(i) *Intel* Pentium (1993)

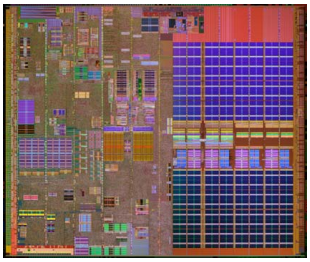
(j) Pentium con 3,1 M di transistor

(k) *Intel* Pentium 3 (1999)



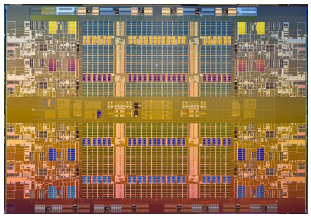
(l) Pentium 3 con 9,9 M di transistor

(m) *Intel* Pentium 4 (2004)



(n) Pentium 4 con 125 M di transistor

(o) *Intel* Xeon 8 *core* (2010)



(p) Xeon con 2270 M di transistor

Figura 1.25: Sviluppo temporale dei principali processori della *Intel*

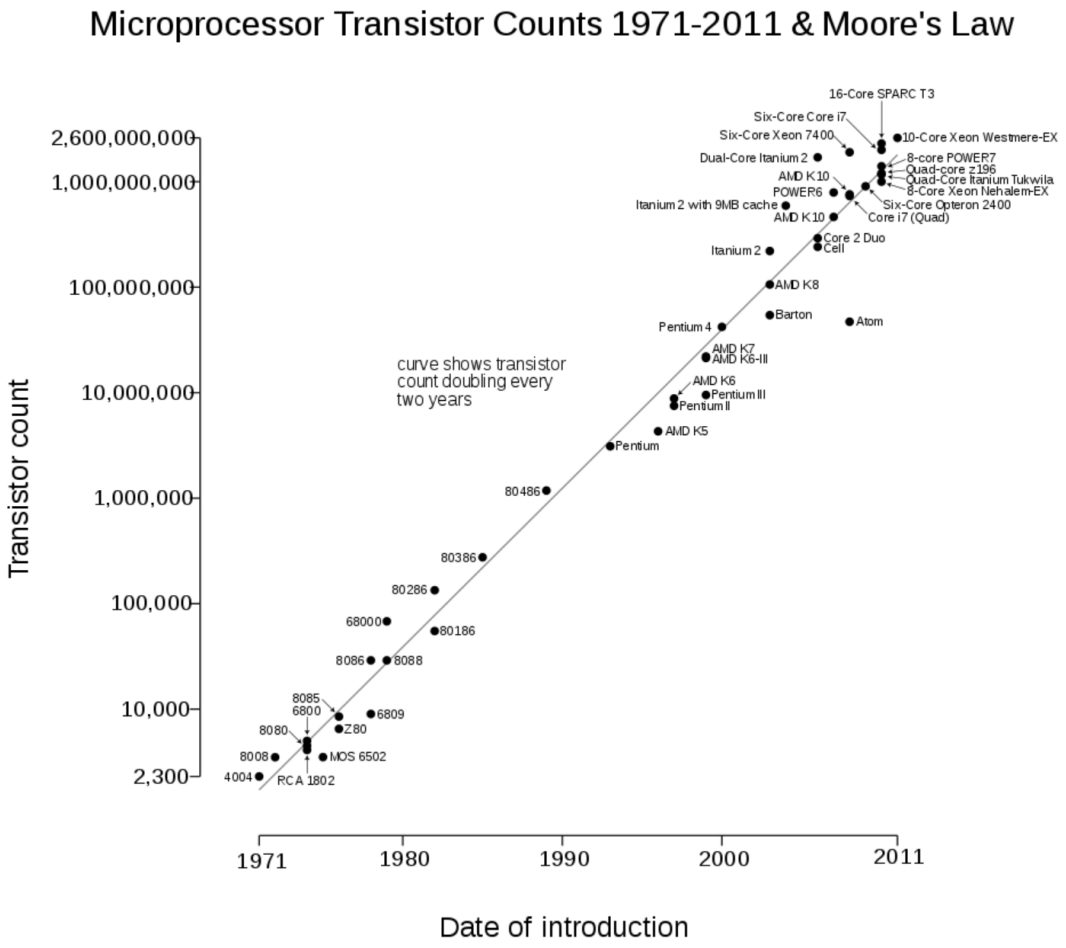


Figura 1.26: Diagramma che illustra la validità della legge di Moore