di Andrea de Prisco

A partire da questo numero MCmicrocomputer ospiterà tra le sue pagine una nuova rubrica denominata, non a caso, appunti di informatica. Quella pura.
Non troverete in queste pagine specifici riferimenti a personal computer oggi in commercio. Faremo semmai un passo indietro, non tanto nel tempo, ma nel livello di alfabetizzazione informatica. Sappiamo infatti come oggi un certo tipo di informatica sia completamente immersa nel vivere moderno. I personal stanno praticamente dappertutto, degli home non ne parliamo: non c'è ragazzetto (dei pochi che ancora non ce l'hanno) che non ne vorrebbe uno. È nata così l'informatica del registratore e del joystick.
In questa sede ci occuperemo dei grossi calcolatori, di linguaggi di programmazione speciali, di processor, di multitasking e delle teorie di base dell'informatica. Argomenti assai cari a chi studia scienze dell'informazione oggi, o più semplicemente si occupava di informatica sei o sette anni fa (prima dell'avvento dei personal).
Obiettivo principale di «Appunti di informatica» è dare al lettore alcuni articoli di facile lettura (almeno speriamo) con l'importante caratteristica della non consequenzialità degli stessi. In altre parole ogni articolo tratterà un tema senza fare necessariamente riferimento alle «puntate» precedenti. In questo modo sarà data al lettore la facoltà di scegliere liberamente gli argomenti che interessano, senza correre il rischio di non raccapezzarcisi più qualche puntata più in là.

La struttura di un calcolatore

Il primo argomento scelto, guardacaso, tratterà in generale la struttura di un sistema di calcolo. Teniamo a sottolineare la parola «sistema» in quanto un calcolatore è sempre fatto da più parti collegate tra loro e con il resto del mondo. Tanto per chiarire subito questo basilare concetto (scemo ma importante) basta pensare a un calcolatore senza né video né stampante né tastiera (in generale senza periferiche I/O), dentro al quale sta girando un programma. Ammesso pure che il programma termini con successo e che i risultati del calcolo siano correttamente mantenuti in memoria, in qualche modo bisognerà pur leggerli, se no ha lavorato per se stesso e non per l'uomo e quindi è inutile.

La tecnologia, si sa, «inventa» macchine, e le macchine esistono in quanto al servizio dell'uomo: se ciò non accade, ritrovando uomini al servizio delle macchine, abbiamo sbagliato qualcosa e occorre ricominciare da capo. Beh, disquisizioni filosofiche a parte (di livello basso, molto basso) andiamo dunque a incominciare.

Dicevamo prima che un calcolatore è composto da più parti tra loro interagenti. Abbiamo già anticipato che un ruolo molto importante è svolto dalle periferiche di ingresso/uscita (I/O Device) che permettono di comunicare col mondo esterno.

Poi, si sa, un calcolatore è dotato di una memoria, per ricordare programmi e dati. A questo bisogna aggiungere un vero e proprio cuore (non osiamo dire «cervello») che elabora le informazioni contenute in memoria e trasmette in qualche modo i risultati alle periferiche di uscita, non senza adoperare periferiche di ingresso per acquisire dati. Questo cuore è la CPU, o se preferite l'unità centrale di elaborazione.

In figura 1 abbiamo schematizzato queste tre parti indicando anche i relativi collegamenti. Andando avanti nella lettura di questo e dei prossimi articoli, potrete rendervi conto di quanto sia drastica questa schematizzazione. Vedremo infatti che le connessioni tra queste varie parti sono tutt'altro che pezzi di filo elettrico, che i dispositivi sono spesso asserviti da veri e propri processor di I/O, e che la CPU può benissimo essere «tante CPU» che lavorano insieme per sbrigare prima i lavori.

Control Process Unit (CPU)

Iniziamo la nostra «zoomata» sulle varie componenti di un calcolatore, scendendo maggiormente nei dettagli,

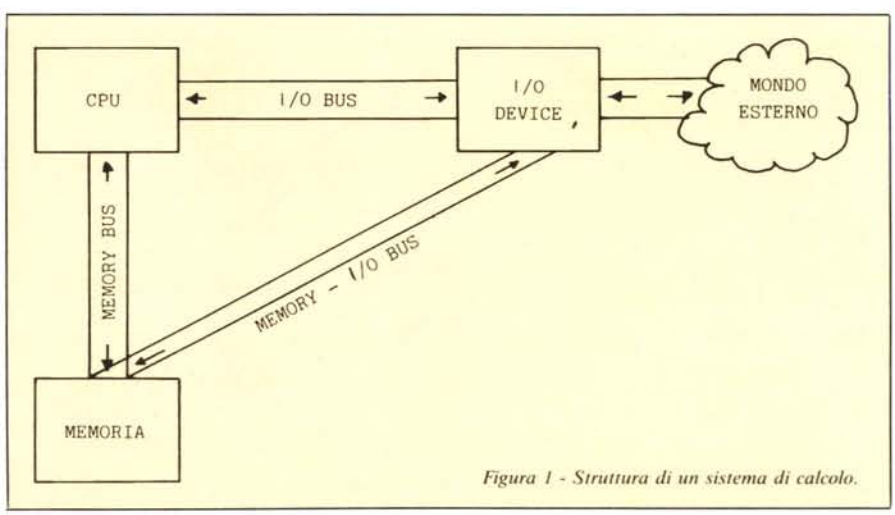

Figura 1 - Struttura di un sistema di calcolo.

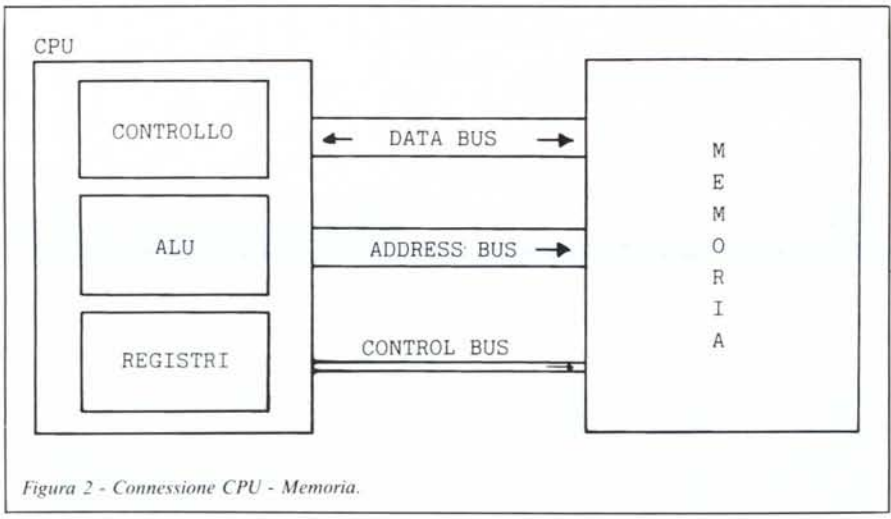

Figura 2 - Connessione CPU - Memoria.

cominciando dall'unità centrale di elaborazione. Essa è collegata con la memoria e con i dispositivi di I/O, come mostrato sempre in figura 1. Ogni CPU ha un proprio linguaggio macchina grazie al quale è possibile scrivere programmi (sequenze di istruzioni) per risolvere problemi. Generalmente il linguaggio macchina di una CPU difficilmente dispone di istruzioni più complicate di sommare due numeri, controllare se un numero è uguale a zero o muovere un dato da una parte della memoria ad un'altra. Infatti anche con questi soli meccanismi è possibile risolvere (scrivere il corrispondente programma) qualsiasi problema risolvibile. È possibile però che il linguaggio macchina sia qualcosa di più complesso, anche un linguaggio ad alto livello. Per esempio esiste la LISP-Machine che è un calcolatore che ha il LISP (linguaggio di programmazione ricorsivo di tipo funzionale) come Linguaggio Macchina. Nulla vieta ad esempio di realizzare una CPU che parli direttamente in PASCAL, o in BASIC o in COBOL: tutto sta nel valutare l'effettiva convenienza in quanto a costi di progettazione (no, ndr). Di solito, dato che programmare in L.M. non è molto divertente, per le varie CPU vengono realizzati compilatori e interpreti che possono tradurre in linguaggio macchina un programma scritto in un altro linguaggio, generalmente più facile da programmare. Comunque non perdiamo il filo del discorso: di questo magari ne riparleremo tra qualche numero.

All'interno della CPU, risiedono un'unità aritmetico-logica, una parte controllo e un insieme di registri per mantenere ad esempio risultati intermedi senza scomodare continuamente la memoria. Il ciclo di funzionamento di una CPU è detto FETCH-EXECUTE, tradotto in linguaggio made in Italy: preleva ed esegui (l'istruzione). Per i più pignoli anticipiamo che alcune CPU funzionano diversamente, magari sovrapponendo le due fasi di due cicli successivi. Ne riparleremo tra qualche puntata: questo mese ci manterremo volutamente sull'elementare.

Il ciclo preleva ed esegui (lo dice il ragionamento stesso) preleva un'istruzione dalla memoria e l'esegue. A seconda del tipo di istruzione la parte controllo scomoderà o meno altre parti della CPU o del calcolatore. Senza andare troppo per il sottile, potrebbe essere una somma di due valori: è implicata l'unità aritmetico-logica; potrebbe essere un test sul risultato di un'operazione: si usa un apposito registro detto di stato; oppure semplicemente un trasferimento CPU-Memoria o viceversa. Prima di terminare questa breve descrizione delle CPU, è doveroso parlare un momentino di un importante registro interno il Program Counter. Come vedremo tra poco, la memoria principale di un computer è organizzata in celle, tutte numerate: in tal modo è possibile accedere ad una qualsiasi cella, semplicemente specificando il suo numero. Siccome il programma che l'unità centrale di elaborazione sta eseguendo è contenuto in un insieme di celle, il Program Counter conterrà sempre l'indirizzo della cella della prossima istruzione da prelevare. È ovvio che tale registro è automaticamente incrementato dopo l'esecuzione di ogni istruzione, eccezion fatta per le istruzioni di salto. In questo caso, non il valore successivo, ma un nuovo valore sarà introdotto nel Program Counter.

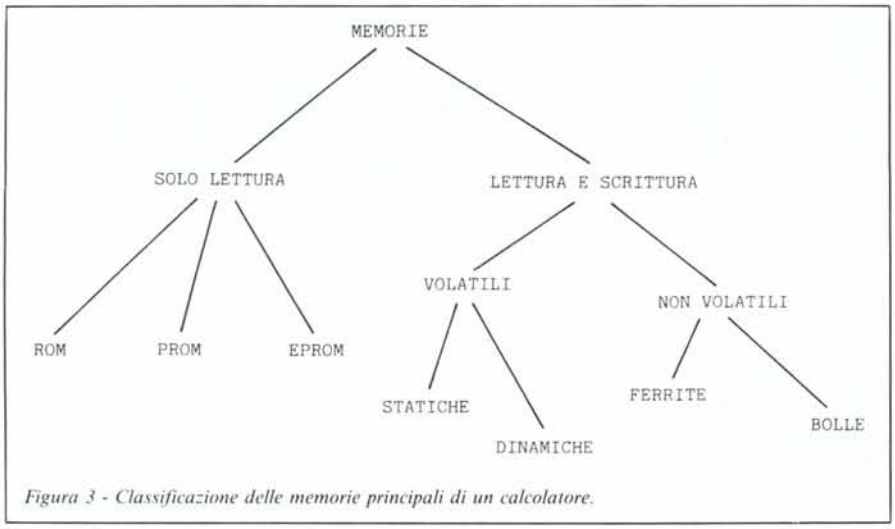

Figura 3 - Classificazione delle memorie principali di un calcolatore.

La memoria

La memoria di un computer è un insieme più o meno esteso di celle, ognuna direttamente indirizzabile, dove programmi e/o dati sono parcheggiati per l'elaborazione. Generalmente una cella può contenere Byte di 8 bit, ma esistono casi sia di indirizzamento al singolo bit (ogni cella può contenere uno 0 o un 1, come nel caso del Burroughs B1700) che casi in cui la minima quantità di memoria indirizzabile è lunga 60 bit, come nel caso del Cyber 70.

Per prima cosa vedremo come la memoria è collegata al processor: in figura 1 abbiamo visto che tale connessione è assicurata dal memory bus, in figura 2 notiamo come questo sia a sua volta composto da tre Bus: l'address bus, il data bus e il control bus. Il primo serve per comunicare alla memoria il numero (l'indirizzo) della cel-

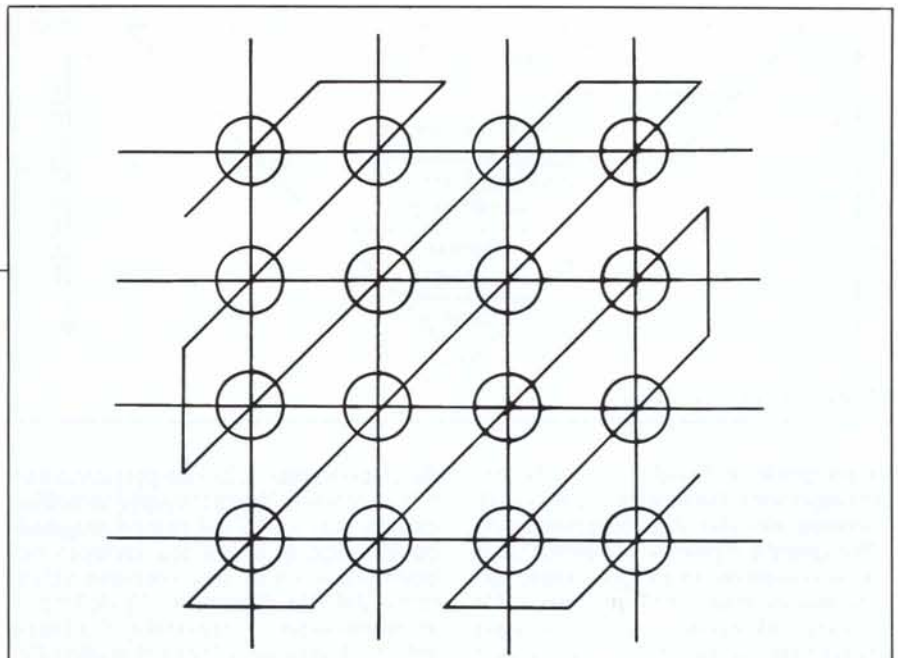

Figura 4 - Memoria a nuclei di ferrite.

la che ci interessa. Sul data bus corre il dato, in un senso o nell'altro, a seconda che sia stata effettuata una operazione di lettura o scrittura. Per specificare questo si usa il control bus che per l'appunto invia alla memoria un segnale o un altro a seconda di quale operazione si deve compiere. Operativamente, la CPU, quando deve leggere il contenuto di una cella invia per prima cosa l'indirizzo sull'address bus; sul control bus specifica l'operazione da compiere (in questo caso «lettura»), e immediatamente dopo può prelevare dal data bus il contenuto della cella che la memoria ha provveduto a mandarle. Per le operazioni di scrittura il ciclo è leggermente diverso: la CPU specifica la cella da utilizzare usando come sempre l'address bus, insieme a questo sul data bus è posto il dato da inviare alla memoria, infine il control bus è posto in stato «scrittura»: il resto lo fa la memoria.

In figura 3 è stata rappresentata una possibile classificazione delle principali memorie centrali dei calcolatori. Esistono infatti sia memorie per lettura e scrittura, come quella appena descritta, sia memorie a sola lettura usate per tenere stabilmente programmi all'interno dei calcolatori. È il caso ad esempio dei sistemi operativi dei personal computer (economici) che all'accensione sono già pronti per funzionare. I grossi cacolatori invece caricano in RAM (memoria lettura e scrittura) anche il sistema operativo che quindi in qualsiasi momento può essere modificato o sostituito per adattarsi a nuove esigenze. Tali calcolatori hanno su ROM (memorie a sola lettura) solo il cosiddetto Caricatore Minimo, una porziuncola di S.O. che ha solo il compito di dialogare con l'unità a dischi per caricare il vero e proprio sistema operativo.

Esistono tre tipi di memorie a sola lettura. Le ROM sono programmate direttamente in fabbrica al momento della costruzione: per l'utente non c'è modo di variarne il contenuto. Le PROM (Programmable ROM) sono memorie che si comprano vuote e possono essere programmate dall'utente tramite un apposito apparecchietto. Come queste, le EPROM (Erasable Programmable ROM) in più permettono di essere cancellate un certo numero di volte semplicemente esponendo la finestrella che portano «in groppa» ai raggi ultravioletti.

Le memorie a lettura e scrittura, usate per le vere e proprie memorie centrali dei calcolatori, si dividono a loro volta in volatili e non volatili a seconda che abbiano bisogno o meno dell'alimentazione per mantenere l'informazione registrata.

Tenendo sempre sott'occhio la figura 3, vediamo che le prime si distinguono tra statiche e dinamiche. Le memorie statiche necessitano della sola alimentazione, quelle dinamiche oltre a questa hanno bisogno del ciclo del refresh per rinfrescare continuamente il contenuto delle celle che altrimenti tenderebbero a dimenticare. Dato che l'informazione in ogni cella pian piano svanisce, il ciclo di refresh, quando la memoria non è usata dalla CPU (ad esempio dopo aver prelevato un'istruzione, mentre la sta analizzando), provvede a leggere e riscrivere ad una ad una tutte le celle in modo da avere sempre byte freschi ossia scritti da poco.

Le memorie non volatili si dividono in due particolari classi: le memorie a nuclei di ferrite e le memorie a bolle. La particolarità sta nel fatto che (combinazione) le memorie a nuclei sono state le prime in assoluto e oggi certamente non esistono più (sono passate alla storia dei computer) mentre quelle a bolle, tuttora oggetto di studio in tutto il mondo, rappresentano davvero il futuro delle memorie per calcolatore.

In questa sede, vedremo più da vicino il funzionamento delle memorie a nuclei di ferrite. Esse sono composte da un numero più o meno grande di anelli di dimensioni minime, ognuno dei quali (rappresentante un bit) è attraversato all'interno (nel buco, per intenderci) da più fili elettrici. In figura 4 è rappresentata schematicamente una memoria a nuclei di $4 \times 4 = 16$ bit. Notiamo fili orizzontali, fili verticali e un filo diagonale che attraversa tutti i nuclei.

Se all'interno di un nucleo passa una corrente maggiore o uguale a un certo I, questa genera un campo magnetico tale da magnetizzarlo permanentemente. Se la corrente sul filo diminuisce o si annulla, il nucleo resterà ugualmente magnetizzato, mantenendo cioè un'informazione. Se mandiamo invece una corrente pari a -1, il nucleo si smagnetizzerà di botto per magnetizzarsi in senso contrario. Chi ha giocato qualche volta con una calamita, avrà notato l'esistenza dei poli Nord e Sud: mandando la corrente -1 Nord e Sud si scambieranno di posto. A questo punto associando i valori 0 e 1 ai due possibili sensi di magnetizzazione abbiamo realizzato una vera e propria memoria. Restano due problemi: come magnetizzare un determinato nucleo senza coinvolgere gli altri e come leggere l'informazione. Il primo è abbastanza semplice, disponendo di fili orizzontali e fili verticali (fig. 4) per magnetizzare ad esempio il nucleo di coordinate (3,2) terza colonna seconda riga manderemo sul filo verticale 3 e filo orizzontale 2 una corrente per ognuno pari a 1/2. Solo nel nucleo interessato passerà corrente

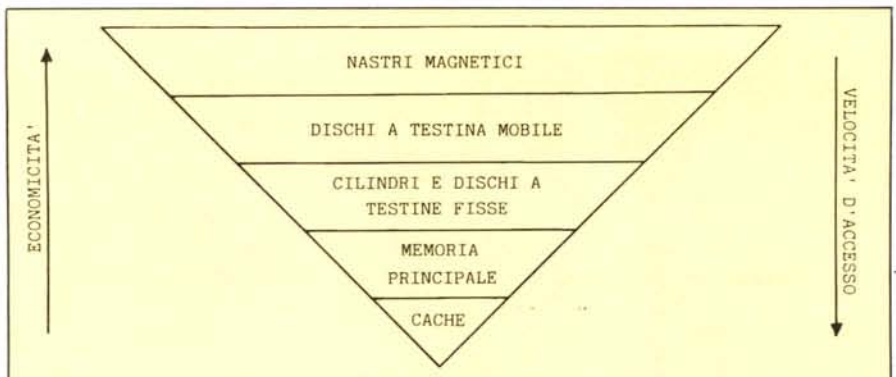

Figura 5 - Gerarchia di memorie.

per un totale di I (sufficiente alla magnetizzazione) trovandosi questo all'incrocio dei due fili. Semplice, no?

Per quanto riguarda la lettura, l'affare si complica un po' in quanto occorre mandare una tensione pari a -1 al nucleo interessato. Se questo era già magnetizzato in tal senso, non accade nulla. Se invece era stato magnetizzato da una corrente 1, la smagnetizzazione e la successiva magnetizzazione provoca una variazione del campo magnetico all'interno del nucleo (sempre nel buco) inducendo una corrente all'interno del filo diagonale. In definitiva se mandando -1 troviamo l'impulso sul filo diagonale allora nel nucleo c'era «scritto» un 1, altrimenti 0. I più attenti avranno notato che l'operazione di lettura è distruttiva in quanto corrisponde a scrivere uno 0 e vedere se è successo qualcosa. Per ovviare a questo inconveniente un apposito circuito di ritardo (non mostrato in figura) provvede a riscrivere quanto appena letto.

Gerarchie di memorie

Quello che abbiamo appena descritto è il funzionamento della memoria principale. In qualsiasi computer, a questa è sempre aggiunta, per motivi di carattere economico, una o più memorie secondarie o di massa. Tra le più diffuse, possiamo indicare le unità a dischi magnetici, a cilindro rotante, a nastro magnetico, tutte caratterizzate (rispetto alla memoria centrale) da un costo per bit assai contenuto, ma con lo svantaggio di avere tempi di accesso al dato cercato superiori di diversi ordini di grandezza. Comunque, dato che gli accessi alla memoria secondaria non sono così frequenti come quelli in memoria principale, il problema dei tempi di accesso si riduce notevolmente, non lasciando nemmeno ipotizzare l'esistenza di computer senza dischi, nastri o cilindri.

Alcuni calcolatori, in contrapposizione alla lentezza delle memorie di massa, hanno accanto alla memoria principale una piccola memoria ausiliaria detta Cache Memory caratterizzata da tempi d'accesso prossimi allo zero (e perciò molto, molto costosa) nella quale vengono caricate ad una ad una le porzioni di programma che devono essere eseguite. In questo modo la CPU può accedere man mano alle varie istruzioni senza nemmeno attendere il già piccolo tempo di accesso proprio delle memorie principali. In figura 5 è rappresentata una gerarchia tra i vari tipi di memorie.

Come abbiamo già detto, al crescere della velocità di accesso, diminuisce l'economicità espressa in costo per bit. Le più lente e più economiche memorie di massa sono senz'altro i nastri magnetici; seguono i dischi a testina mobile; i cilindri rotanti e i dischi a testine fisse; la memoria principale e in ultima la Cache Memory. Anche di queste avremo occasione di parlarne nei prossimi numeri.

Arrivederci. **MC**

L'aritmetica binaria

I calcolatori, si sa, ragionano internamente per mezzo di numeri binari o se preferite in base 2. Questo perché è più facile per i circuiti elettronici mantenere e/o riconoscere stati binari che possono assumere solo valori del tipo SI/NO, vero/falso o acceso/spento. In altre parole è facile stabilire con esattezza se un filo porta corrente o meno semplicemente misurando un'eventuale differenza di potenziale.

I numeri binari sono concettualmente identici ai numeri decimali di uso comune. L'unica differenza è che invece di disporre di 10 simboli (da 0 a 9) si dispone di soli due simboli (0 e 1) per rappresentare un numero. Se ad esempio vogliamo contare in binario, procederemo come abbiamo sempre fatto sin dall'asilo infantile semplicemente tenendo a mente che abbiamo a disposizione solo zeri e uno. Quindi:

0
1
10
11
100
101
110
111
1000 ecc.

Sembra strano, ma è la stessa cosa: in decimale cosa avremmo fatto? Si parte da 0 e si elencano tutti i numeri ad una sola cifra, quindi da 0 a 9. Poi si passa alle decine e si ripete tutto anteponendo alle unità nuovamente tutti i simboli, fino al nuovo esaurimento. Poi si cicla con le centinaia e così via. La numerazione binaria funziona allo stesso modo: controllare per credere.

Altra differenza ovvia è che, essendo la base 2, non avremo unità, decine, centinaia ecc. ma unità duine, quattrine, ottine (tutte potenze di due). Da qui la facile conversione binario decimale nel seguente modo: per esempio 101 è 1 unità +0 duine +1 quattrina = 1+0+4=5. Altro esempio: 101101 è 1 unità +0 duine +1 quattrina +1 ottina +0 sedicine +1 trentaduina = 1+4+8+32=45.

Per convertire un numero da decimale a binario si procede per divisioni successive per due conservando i vari resti.

Per esempio convertiamo 47 in binario. Allora:

47:2 = 23 col resto di 1
23:2 = 11 col resto di 1
11:2 = 5 col resto di 1
5:2 = 2 col resto di 1
2:2 = 1 col resto di 0
1:2 = 0 col resto di 1

Se mettiamo in fila tutti i resti ottenuti partendo a ritroso dall'ultimo otteniamo 101111 che è la rappresentazione in binario di 47. Se non ci credete provate a ritrasformarlo in decimale come abbiamo fatto sopra.

Funziona.

Per quanto riguarda le operazioni ancora niente di nuovo, ad esempio la somma:

0+0=0
0+1=1+0=1
1+1=10 o meglio 0 col riporto di 1
1+1+1=11 o 1 col riporto di 1

Con queste sole 4 somme è possibile sommare qualsiasi coppia di numeri binari, tanto per cambiare procedendo come abbiamo sempre fatto. E qui cadranno miseramente quei furbi (naturalmente giovanissimi) che alle elementari portavano di nascosto la calcolatrice di Papà per fregare il Signor Maestro.