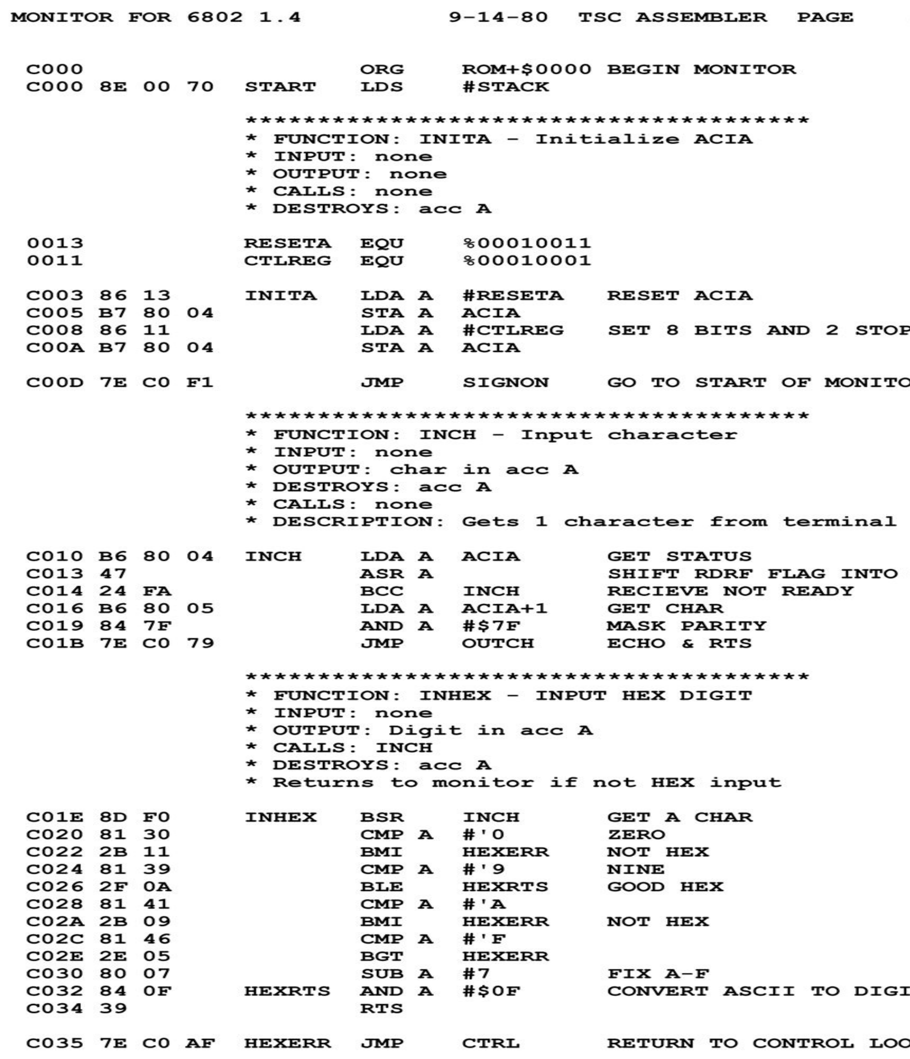
ASSEMBLY

Fabio Danubbio III C INF

PROGRAMMARE
PROGRAMMA
SOFTWARE
ALGORITMI

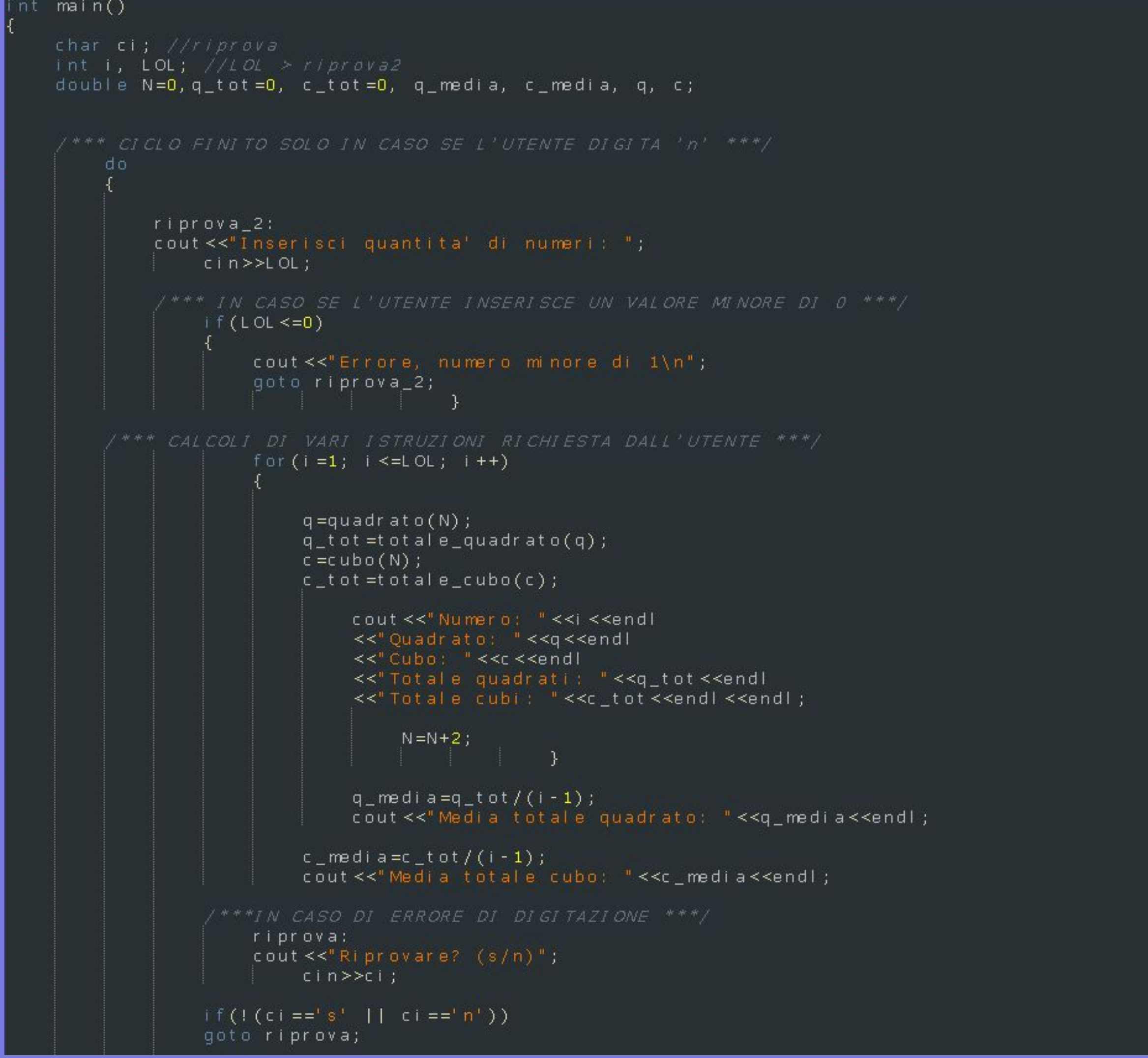
BREVE STORIA DELLA PROGRAMMAZIONE

LINGUAGGIO BASSO LIVELLO



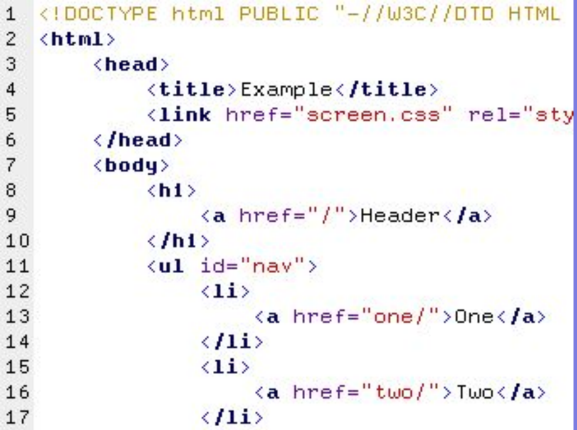
```
MONITOR FOR 6802 1.4             9-14-80   TSC ASSEMBLER   PAGE

 C000                          ORG      ROM+$0000 BEGIN MONITOR
 C000 8E 00 70   START         LDS      #STACK

                 ***************************************
                 * FUNCTION: INITA - Initialize ACIA
                 * INPUT: none
                 * OUTPUT: none
                 * CALLS: none
                 * DESTROYS: acc A

 0013            RESETA        EQU      %00010011
 0011            CTLREG        EQU      %00010001

 C003 86 13      INITA         LDA A    #RESETA    RESET ACIA
 C005 B7 80 04                 STA A    ACIA
 C008 86 11                    LDA A    #CTLREG    SET 8 BITS AND 2 STOP
 C00A B7 80 04                 STA A    ACIA

 C00D 7E C0 F1                 JMP      SIGNON     GO TO START OF MONITO

                 ***************************************
                 * FUNCTION: INCH - Input character
                 * INPUT: none
                 * OUTPUT: char in acc A
                 * DESTROYS: acc A
                 * CALLS: none
                 * DESCRIPTION: Gets 1 character from terminal

 C010 B6 80 04   INCH          LDA A    ACIA       GET STATUS
 C013 47                       ASR A               SHIFT RDRF FLAG INTO
 C014 24 FA                    BCC      INCH       RECIEVE NOT READY
 C016 B6 80 05                 LDA A    ACIA+1     GET CHAR
 C019 84 7F                    AND A    #$7F       MASK PARITY
 C01B 7E C0 79                 JMP      OUTCH      ECHO & RTS

                 ***************************************
                 * FUNCTION: INHEX - INPUT HEX DIGIT
                 * INPUT: none
                 * OUTPUT: Digit in acc A
                 * CALLS: INCH
                 * DESTROYS: acc A
                 * Returns to monitor if not HEX input

 C01E 8D F0      INHEX         BSR      INCH       GET A CHAR
 C020 81 30                    CMP A    #'0        ZERO
 C022 2B 11                    BMI      HEXERR     NOT HEX
 C024 81 39                    CMP A    #'9        NINE
 C026 2F 0A                    BLE      HEXRTS     GOOD HEX
 C028 81 41                    CMP A    #'A
 C02A 2B 09                    BMI      HEXERR     NOT HEX
 C02C 81 46                    CMP A    #'F
 C02E 2E 05                    BGT      HEXERR
 C030 80 07                    SUB A    #7         FIX A-F
 C032 84 0F      HEXRTS        AND A    #$0F       CONVERT ASCII TO DIGI
 C034 39                       RTS

 C035 7E C0 AF   HEXERR        JMP      CTRL       RETURN TO CONTROL LOO
```

LINGUAGGIO ALTO LIVELLO



```
int main()
{
    char ci; //riprova
    int i, LOL; //LOL > riprova2
    double N=0,q_tot=0, c_tot=0, q_media, c_media, q, c;


    /*** CICLO FINITO SOLO IN CASO SE L'UTENTE DIGITA 'n' ***/
        do
        {

            riprova_2:
            cout<<"Inserisci quantita' di numeri: ";
                cin>>LOL;

            /*** IN CASO SE L'UTENTE INSERISCE UN VALORE MINORE DI 0 ***/
                if(LOL<=0)
                {
                    cout<<"Errore, numero minore di 1\n";
                    goto riprova_2;
                                    }

        /*** CALCOLI DI VARI ISTRUZIONI RICHIESTA DALL'UTENTE ***/
                    for(i=1; i<=LOL; i++)
                    {

                        q=quadrato(N);
                        q_tot=totale_quadrato(q);
                        c=cubo(N);
                        c_tot=totale_cubo(c);

                            cout<<"Numero: "<<i<<endl
                            <<"Quadrato: "<<q<<endl
                            <<"Cubo: "<<c<<endl
                            <<"Totale quadrati: "<<q_tot<<endl
                            <<"Totale cubi: "<<c_tot<<endl<<endl;

                                N=N+2;
                                            }

                            q_media=q_tot/(i-1);
                            cout<<"Media totale quadrato: "<<q_media<<endl;

                        c_media=c_tot/(i-1);
                        cout<<"Media totale cubo: "<<c_media<<endl;

                /***IN CASO DI ERRORE DI DIGITAZIONE ***/
                    riprova:
                    cout<<"Riprovare? (s/n)";
                        cin>>ci;

                if(!(ci=='s' || ci=='n'))
                goto riprova;
```

LINGUAGGIO MEDIO-ALTO LIVELLO

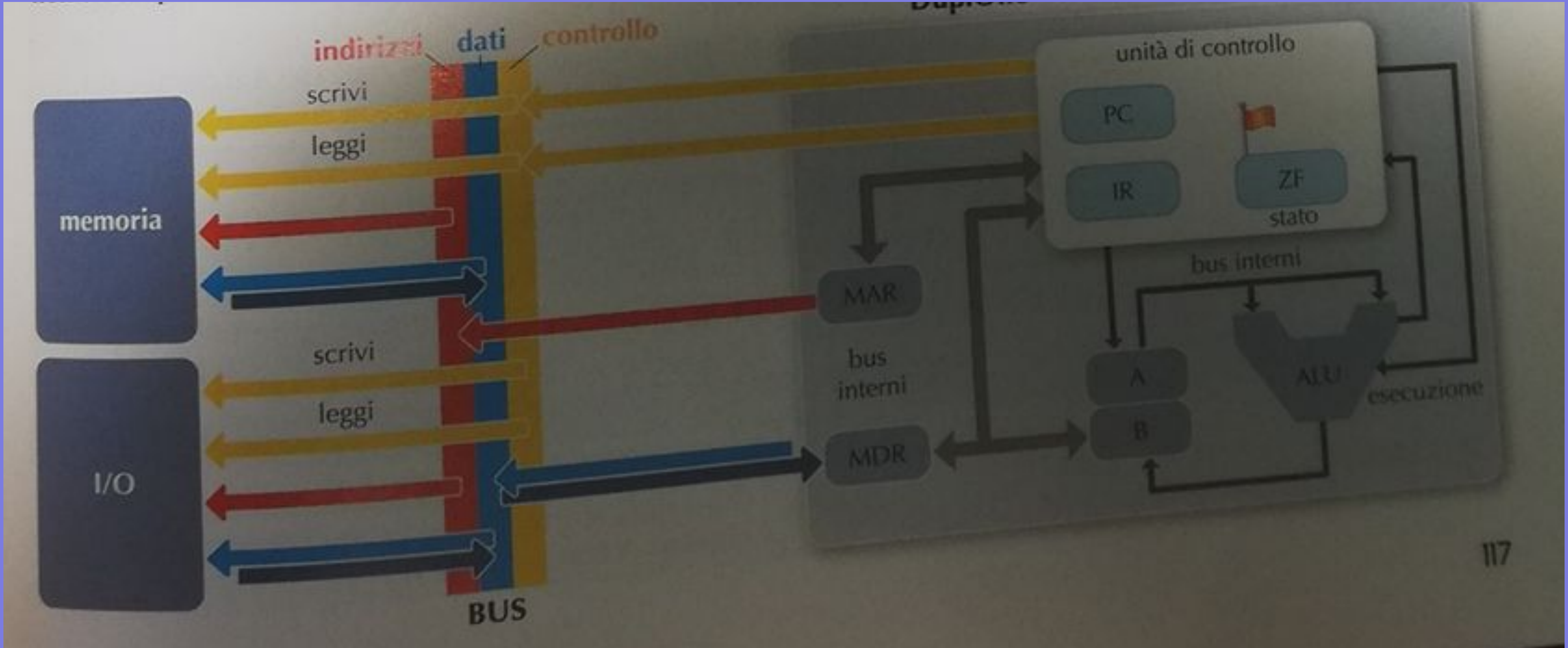


```html
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML
<html>
    <head>
        <title>Example</title>
        <link href="screen.css" rel="sty
    </head>
    <body>
        <h1>
            <a href="/">Header</a>
        </h1>
        <ul id="nav">
            <li>
                <a href="one/">One</a>
            </li>
            <li>
                <a href="two/">Two</a>
            </li>
```

SINTASSI GENERICA ASSEMBLY

`<CODICE OPERATIVO> <OPERANDI>`

DUPLONE


indirizzi
dati
controllo
scrivi
leggi
memoria
scrivi
leggi
I/O
BUS
unità di controllo
PC
IR
ZF
stato
bus interni
MAR
bus
interni
MDR
A
B
ALU
esecuzione
117

CATEGORIE DI ISTRUZIONI

(GENERALI)

ISTRUZIONI ARITMETICHE/LOGICHE

ADD - SUB - DIV - AND - OR - NOT

ECC… . .

ISTRUZIONI DI SALTO
JMP (NC) - JNZ - JZ

ISTRUZIONI DI TRASFERIMENTO
MOV B [LOCAZIONE]

CALL CONDIZIONATO

CALL NON CONDIZIONATO

METODI DI INDIRIZZAMENTO

INDIRIZZAMENTO A REGISTRO

INDIRIZZAMENTO IMMEDIATO

INDIRIZZAMENTO DIRETTO

INDIRIZZAMENTO INDIRETTO