CRONOS

Cronometro avanzato e automatico per uso sportivo

Tesina per Esame di Stato
IPSIA Massa Marittima - a.s. 2004/2005
Francesco Galgani - classe 5^a AP

17 giugno 2005

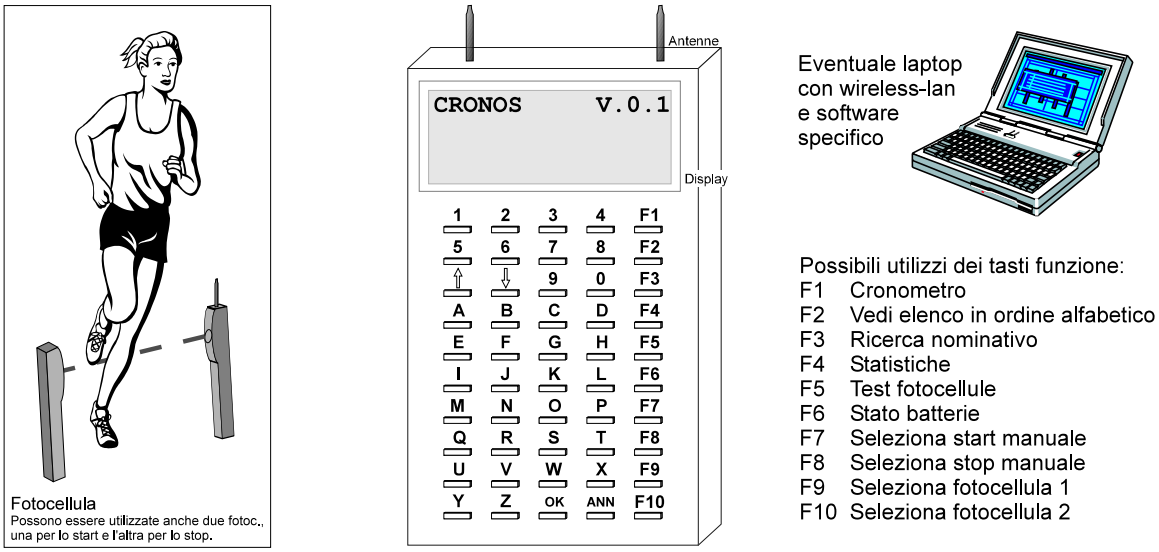
Indice

1 Il progetto in sintesi

Il modello di base del prodotto che ho progettato, composto da un dispositivo palmare e da una coppia di fotocellule senza fili, è destinato all'uso sportivo e, in particolare, serve per misurare con precisione il tempo impiegato per percorrere una certa distanza. A differenza di un comune cronometro, il dispositivo permette di memorizzare i tempi associandoli ad uno o più nominativi, di confrontarli, di realizzare statistiche. La semplicità dell'utilizzo e l'agevole possibilità di spostamento e di trasporto lo rendono adatto per l'allenamento degli atleti. Tramite connessione wireless-lan, sarebbe possibile integrare o sostituire il palmare con un computer portatile, in maniera da poter gestire l'allenamento tramite un apposito software. Il kit potrebbe anche essere completato con un piccolo sensore per misurare le pulsazioni cardiache, anch'esso senza fili e in grado di interfacciarsi sia con il palmare sia con il computer portatile.

Ho progettato e realizzato gli schemi elettrici allegati, che riguardano il dispositivo palmare, le fotocellule e i relativi sistemi di trasmissione, mettendo alla prova le competenze acquisite nel corso del quinquennio di studio. Benché il tempo dedicato a questo progetto abbia superato le 100 ore, non mi è stato possibile, data la complessità e i mezzi utilizzati, realizzare un prodotto finito.

2 Le funzioni



Il disegno sopra riportato schematizza come si dovrebbe presentare il prodotto.
Seguono alcuni possibili utilizzi del palmare:

1. **Cronometro**
   E' possibile selezionare il modo con cui verranno attivati lo start e lo stop (manuali o con fotocellula), inserire la distanza percorsa, calcolare la velocità e memorizzare il tempo.

2. **Vedi elenco in ordine alfabetico**
   Il metodo più semplice per memorizzare i tempi è quello di associarli ciascuno ad un nome. Più tempi possono essere associati alla stessa persona. La consultazione del semplice elenco alfabetico, però, può presentare dei limiti nel caso in cui si memorizzino centinaia di nomi, nel qual caso sarà possibile organizzarli in gruppi e sottogruppi. Ad esempio, in un contesto scolastico sarà utile suddividere gli alunni in gruppi corrispondenti alla scuola di appartenenza e in sottogruppi costituiti dalle classi.

3. **Ricerca nominativo**
   La ricerca per nominativo permette di trovare i dati relativi ad una data persona, indipendentemente dal gruppo o dal sottogruppo.

4. **Statistiche**
   Possono essere utilizzate per valutare l'andamento nel tempo delle proprie prestazioni, con riferimento non soltanto alla distanza percorsa e alle velocità raggiunte, ma anche al grado di allenamento eventualmente stimato sulla base delle pulsazioni cardiache. Inserendo età, sesso, peso e altezza può anche essere calcolato il dispendio calorico.

5. **Test fotocellule**
   In assenza di ostacoli tra fotocellule e palmare e di interferenze radio (che sono possibili ma poco probabili, poiché la trasmissione avviene su frequenze appositamente riservate

per i telecontrolli, in base alle leggi vigenti), la loro distanza massima può essere di alcuni km. Il test delle fotocellule permette comunque di accertarsi che la comunicazione avvenga correttamente. Questo test viene comunque ripetuto automaticamente quando si utilizza la funzione cronometro.

6. **Stato batterie**
   Permette di verificare lo stato delle batterie sia del palmare sia delle fotocellule.

Queste funzioni dovrebbero essere sufficienti per coprire le più comuni esigenze di monitoraggio durante e dopo l'allenamento.

3 Descrizione della fase progettuale

3.1 Generalità

La progettazione ha richiesto innanzitutto la suddivisione del sistema in diversi blocchi, attribuendo a ciascuno di essi le relative specifiche e il ruolo di input/output rispetto alla cpu centrale.

Questo tipo di approccio semplifica il lavoro, permettendo sia una visione d'insieme, sia la possibilità di modificare, correggere o migliorare i singoli blocchi, sia di aggiungerne altri, in maniera tale da integrare nuove funzionalità nel dispositivo. Il sistema di base del palmare è stato così schematizzato:

input: tastierino
ricevitore 433 Mhz (fotocellula 1)
ricevitore 866 Mhz (fotocellula 2)
memoria eeprom

output: display lcd
memoria eeprom

Il compito della cpu centrale, in questo caso costituita da un microprocessore della famiglia ST6, è quello di gestire le periferiche in input e in output, a seconda della funzione richiesta dall'utente. Ulteriori periferiche potranno essere aggiunte in seguito.

Definiti i singoli blocchi, il passaggio successivo è stato la definizione dell'hardware necessario per il loro funzionamento; solo successivamente, una volta accertata la correttezza di circuiti, è stato possibile scrivere il firmware dei sei microcontrollori utilizzati. Ulteriori modifiche all'hardware e al software sono state apportate durante le varie fasi di collaudo.

Per evitare di sprecare materiale, che potrebbe rivelarsi inservibile a seguito di modifiche nel progetto, i circuiti, prima della stampa, sono stati provati su basette bread-board (costituite da numerosi fori ordinati in righe e colonne), che permettono il montaggio e lo smontaggio dei componenti in maniera provvisoria, senza saldature, e possono essere riutilizzate un numero indeterminato di volte. Solo successivamente sono stati realizzati i circuiti stampati su cui sono stati saldati i componenti.

3.2 Descrizione dei componenti principali

Microprocessori ST6

I microprocessori svolgono un ruolo centrale e hanno due vantaggi fondamentali: il primo è quello di far svolgere determinate funzioni al circuito in cui sono inseriti senza modificare l'hardware, il secondo è quello di semplificare sensibilmente lo schema elettrico, condensando in un solo integrato funzioni che altrimenti richiederebbero migliaia o milioni di transistor.

I microprocessori utilizzati sono dei vecchi ST6, con soli 4kb di rom e concepiti nel lontano 1988: nonostante le evidenti limitazioni rispetto ai più recenti micro, sanno svolgere egregiamente le funzioni necessarie per pilotare il sistema.

L'emulatore DSE622 e i micro con eprom cancellabile mediante raggi UV mi hanno permesso di provare i firmware durante il loro sviluppo. Gli ST6 usano un linguaggio assembler formato da non più di trenta istruzioni diverse (molte delle quali svolgono funzioni simili, per cui le istruzioni principali non sono più di cinque o sei).

Per il momento ho scritto 2800 righe complessive di codice, che allego al presente documento come dimostrazione del lavoro svolto. Il codice è abbondantemente commentato per facilitarne la lettura.

Nella cpu centrale, costituita da un ST62T25, sono disponibili 20 I/O, 4kb di rom, 60 byte di ram, un convertitore A/D, un timer e alcuni piedini per funzioni speciali. E' necessario un oscillatore esterno da 8Mhz. Per quanto riguarda la struttura dei programmi adottati, si veda la sez. 5.

Tastierino

Non potendo disporre di un tastierino nuovo con 50 tasti in tempi ragionevoli, me ne sono procurato uno smontando un vecchio telecomando di un televisore ormai in disuso. Ciò ha richiesto un attento studio dello schema elettrico preesistente e la creazione di uno nuovo.

Per il funzionamento del tastierino sono state sufficienti 16 linee (8 ingressi e 8 uscite): il relativo micro, opportunamente istruito, è in grado di rilevare quale dei 50 tasti viene premuto confrontando lo stato degli ingressi e delle uscite, secondo delle combinazioni stabilite dal produttore (a cui sono risalito studiando le piste e servendomi di un semplice multimetro).

Display

Ho usato un display lcd che interagisce bene con il microprocessore ST6 e che supporta un codice ASCII incompleto (mancano ad esempio le lettere accentate) ma personalizzabile (posso aggiungere fino a 32 caratteri). Il display può mostrare due righe di testo con 16 caratteri ciascuna, è pilotato da due microprocessori montati sul display stesso, gli i.c. HD44100FS e HD44780A00, mediante un complesso set di istruzioni documentato nei relativi datasheets. E' stato necessario creare alcune specifiche routines per l'ST6.

Memoria esterna

I quattro i.c. 24C64 (64kbit ciascuno) forniscono una memoria complessiva di 32kbyte e sono idonei per lavorare in combinazione con i micro ST6.

Moduli Aurel

I moduli Aurel, utilizzati per trasmettere al palmare lo stato delle fotocellule, sono quattro piccoli circuiti, due trasmettitori e due ricevitori, che lavorano su frequenze diverse e che usano la tecnica di modulazione impulsiva ASK-OOK (on-off-keying): di semplice realizzazione, è

stata la prima tecnica di modulazione numerica (telegrafia) ed è idonea per controlli di tipo on/off.

Questi piccoli moduli non hanno bisogno di ulteriore circuiteria esterna per il loro funzionamento. L'unica difficoltà è stata quella di creare un protocollo di trasmissione, non ancora completamente implementato, per identificare i possibili stati delle fotocellule e delle relative batterie:

- fotocellula non connessa (spenta o troppo distante dal palmare),
- fotocellula oscurata,
- fotocellula non oscurata,
- batteria della fotocellula scarica,
- batteria della fotocellula carica.

Alimentazione

Considerando che per il corretto funzionamento dei componenti del palmare sono necessari 5V, stabilizzati con un errore massimo del 5%, ho previsto uno stabilizzatore di tensione i.c. 7805, che fornisce, nella peggiore delle ipotesi, una tensione entro i limiti previsti. L'alimentazione dovrebbe essere fornita da una pila a 9V e la tensione in uscita dallo stabilizzatore è stata innalzata di 0,6V rispetto ai 5V nominali per compensare le cadute di tensione sui diodi.

L'alimentazione delle fotocellule e dei relativi trasmettitori è fornita da tre batterie da 9V, da cui sono ricavate, mediante appositi stabilizzatori, le tensioni 24V (per le fotocellule), 17V (per un trasmettitore), 5V (per il contatto comune delle fotocellule, per un trasmettitore, per ST6 e relativa circuiteria).

Successivamente, mediante degli amplificatori operazionali configurati come differenziatori e comparatori, o tramite il convertitore A/D interno all'ST6, sarà possibile completare il circuito di alimentazione con la possibilità di misurare la carica delle batterie.

3.3 Le principali abilità richieste

Al di là della realizzazione pratica dei circuiti, successivamente descritta, i principali problemi che ho dovuto affrontare nella progettazione, riconducibili al percorso di studio professionale, riguardano soprattutto argomenti affrontati nelle discipline di Elettronica, Controlli Elettronici e Sistemi Programmabili, Terza Area Professionalizzante. Sono stati un importante ausilio anche l'abitudine a ragionare rigorosamente e l'interesse ad analizzare i problemi e a studiare i sistemi più adatti per la loro risoluzione.

Le attività di laboratorio svolte sia a scuola sia nel privato (rese possibili anche grazie alle attrezzature e ai componenti elettrici che mi sono procurato durante la mia attività autonoma di studio estivo) mi hanno permesso di approfondire dal punto di vista pratico le conoscenze teoriche.

Nello specifico, i principali prerequisiti che sono stati necessari per la progettazione riguardano:

1. utilizzo di un programma di disegno assistito al computer (CAD);
2. utilizzo di simulatori, editor, compilatori, programmatori sia nella parte software che hardware, corretta esecuzione di un debug;
3. struttura di un sistema programmabile;
4. architettura di un microcontrollore;
5. analisi e utilizzo di un linguaggio di programmazione;
6. tipi di segnali, livelli logici, configurazioni ingressi e uscite, ingressi flottanti, pull-up e pull-down;
7. telecomunicazioni e modulazioni numeriche;
8. trasmissioni parallele, trasmissioni seriali sincrone e asincrone;
9. tipi di memorie, gestione eeprom;
10. tipi di display, pilotaggio di un display lcd;
11. soluzioni circuitali nelle tastiere alfanumeriche, codifica di una tastiera, codifica ASCII, altre codifiche;
12. regolatori e stabilizzatori di tensione e loro utilizzo;
13. circuiti di protezione;
14. utilizzo di basi di riporto decimale, binaria, esadecimale;
15. caratteristiche d’uscita e utilizzo dei principali componenti elettronici.

4 Realizzazione pratica

La realizzazione pratica degli schemi elettrici allegati si è rivelata indispensabile ai fini della progettazione stessa, in quanto la scrittura del software, cioè del firmware dei microcontrollori, richiede un'imprescindibile base hardware.

4.1 Scelta delle attrezzature e dei componenti

Scelta delle attrezzature per la fase progettuale:

- basette bread-board millefori
- multimetri digitali
- programmatore per STMicroelectronics ST621X / ST622X
- CAD per PC: CadSoft Eagle 4.11
- SofTec DSE622 Real Time Emulator per ST622X e relativo emulatore software
- fili rigidi da elettronica

Scelta delle attrezzature per la creazione dei circuiti stampati:

- stampante laser
- Kruse Density Toner spray
- fogli lucidi semi-trasparenti idonei per stampa circuiti e fotoincisione basette
- basette doppia faccia fotosensibili
- bromografo
- soda caustica diluita
- tricloruro ferrico
- alcol isopropilico
- trapano a colonna
- saldatore da elettronica e relativa attrezzatura (stagno, spugna, terza mano, aspiratore, ecc.)

Scelta dei componenti per realizzazione del palmare:

- i.c. 7805, ST6225, ST6220, 24C64
- moduli Aurel 8110 RXSTDLC, TX433BOOST, RX8L50SA70, TX8LAVSA05
- display lcd Hitachi LM093XMLN
- tastierino 50 tasti CME DIGIT 2000 702118A SAA 1250 SL
- componenti discreti vari (diodi, condensatori, resistenze)
- quarzi da 8Mhz
- connettori a striscia maschi e femmine
- antenne Aurel flessibili in gomma 50 ohm, 2W, 433 e 868 Mhz

4.2 Creazione dei circuiti stampati

Il procedimento artigianale per la realizzazione dei circuiti stampati si è basato, pur nella sua semplicità, sugli stessi presupposti del processo industriale (fotoincisione e reazioni chimiche) e ha permesso di ottenere risultati professionali, con l'unico limite che non è stato possibile superare un certo livello di compattezza. I vari passaggi della progettazione e del montaggio possono essere sintetizzati in questo ordine:

A. studio degli elementi costitutivi di ogni blocco del sistema e ricerca di componenti idonei a soddisfarne le specifiche;

B. disegno dello schema elettrico con Eagle o altro programma di CAD (disegno assistito al computer);

C. montaggio e collaudo del circuito su bread-board (ed eventuali modifiche allo schema elettrico);

D. creazione delle piste su lato rame e lato componenti con l'ausilio di Eagle o di altro programma di CAD;

E. stampa dei circuiti su appositi fogli lucidi semi-trasparenti con stampante laser;

F. annerimento del toner, per conferirgli maggiore consistenza e impedire il passaggio della luce sulla parte stampata (mediante un apposito spray per uso tipografico: Kruse Density Toner);

G. fotoincisione dei circuiti stampati mediante bromografo che, per mezzo di lampade al neon a luce ultravioletta, consente, nella successiva fase H, di rimuovere la vernice protettiva dove il rame dovrà poi essere eliminato (fase I);

H. bagno della basetta nella soda caustica diluita, che consente la rimozione della vernice protettiva, perfezionata con abbondante lavaggio con acqua corrente;

I. immersione della basetta nel tricloruro ferrico che, con una semplice reazione chimica, porta in soluzione il rame in eccesso: le piste sono realizzate;

L. foratura della basetta;

M. saldatura dei componenti;

N. collaudo.

5 Scrittura dei firware

Come visibile nei codici sorgenti allegati, i programmi contenuti all'interno dei microcontrollori sono strutturati nei seguenti blocchi principali.

Definizione dei registri e delle variabili

Ad ogni cella di memoria, a cui corrisponde un numero esadecimale, viene associata un'etichetta, composta da massimo otto caratteri e possibilmente facile da ricordare.

Configurazione degli ingressi e delle uscite

Attraverso l'uso combinato di tre registri (pdir, popt, port), è possibile stabilire la funzione di ogni piedino di I/O dell'ST6. E' possibile configurare ingressi digitali con o senza pull-up interno o con interrupt, ingressi analogici (da collegare al convertitore A/D interno), uscite open-collector o push-pull. Gli ST6220 dispongono di 12 I/O, gli ST6225 di 20 I/O; come visibile dalla nomenclatura dei piedini riportata negli schemi allegati, gli ingressi e le uscite sono suddivise in porta A, B e C (la terza è presente soltanto negli ST6225).

Subroutine di interrupt

Sono disponibili sei vettori di interrupt, ognuno dei quali viene richiamato a seguito di un determinato evento:

1. fine conversione A/D
2. fine conteggio del TIMER
3. cambio di stato di un piedino sulle porte B o C
4. cambio di stato di un piedino sulla porta A
5. attivazione dell'ingresso NMI (interrupt non mascherabile)
6. attivazione del RESET

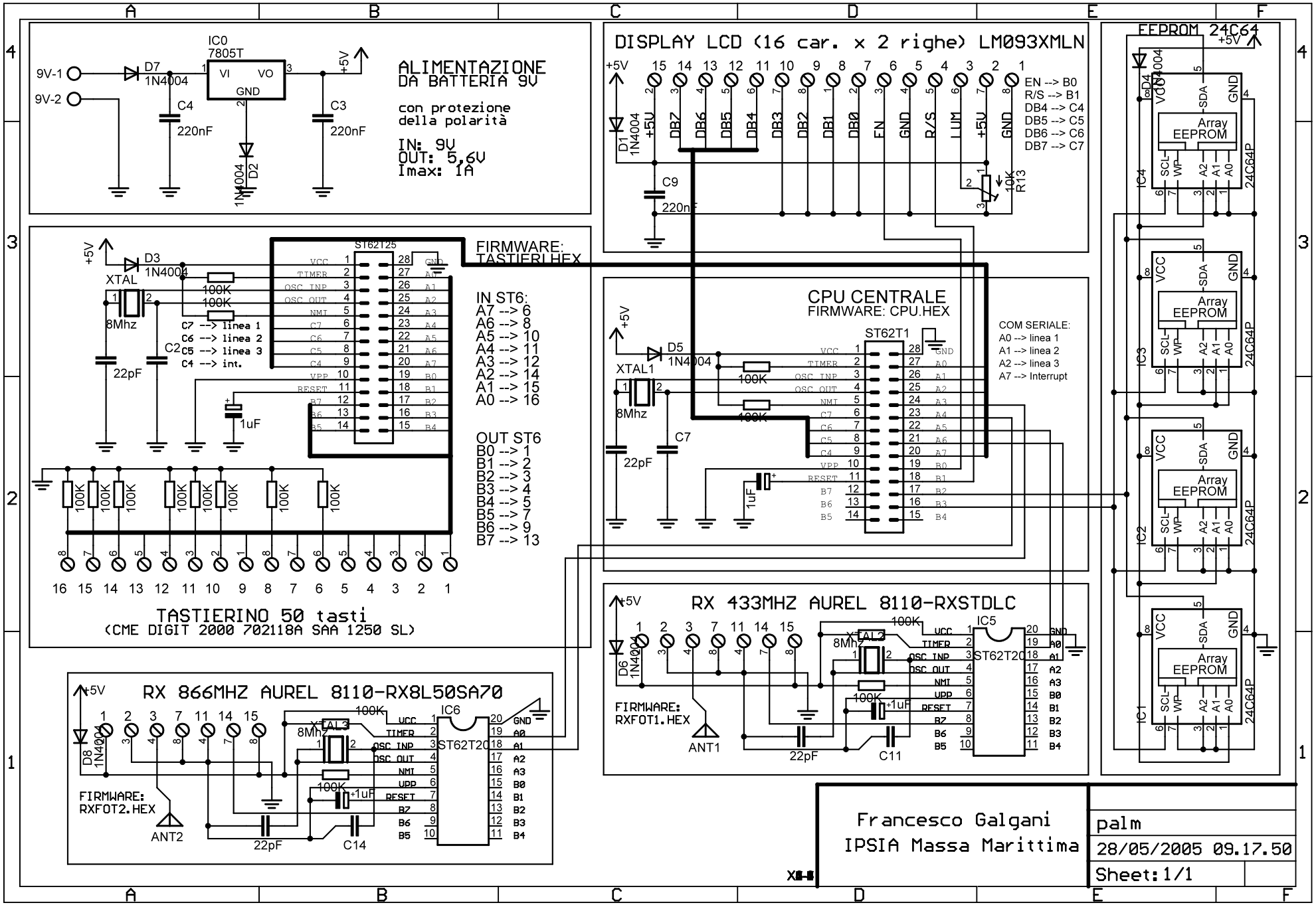
Ogni vettore di interrupt, a sua volta, richiama la rispettiva subroutine di interrupt, che contiene un blocco di istruzioni che viene eseguito indipendentemente dal programma principale.

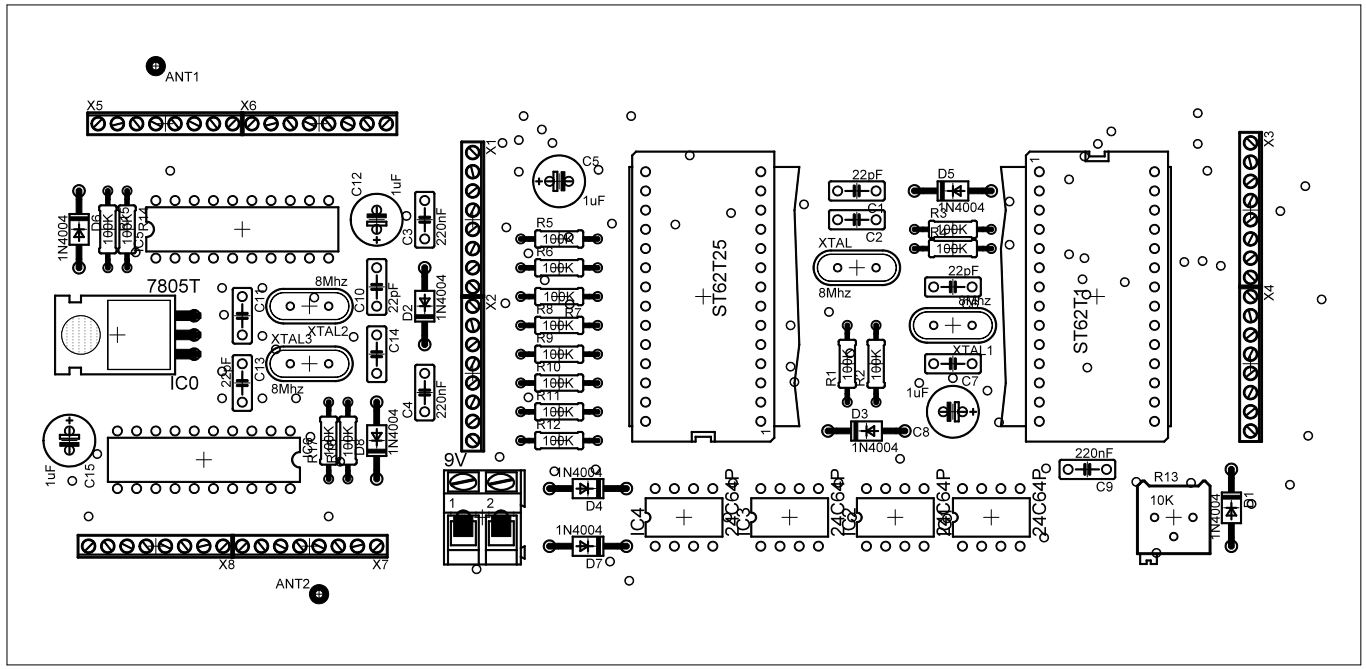
Subroutine

Le subroutine, che non sono associate ad alcun interrupt, contengono blocchi di istruzioni e vengono eseguite solo su specifica richiesta del programma principale. Una subroutine, a sua volta, può richiamare un'altra subroutine, fino ad un massimo di sei livelli.

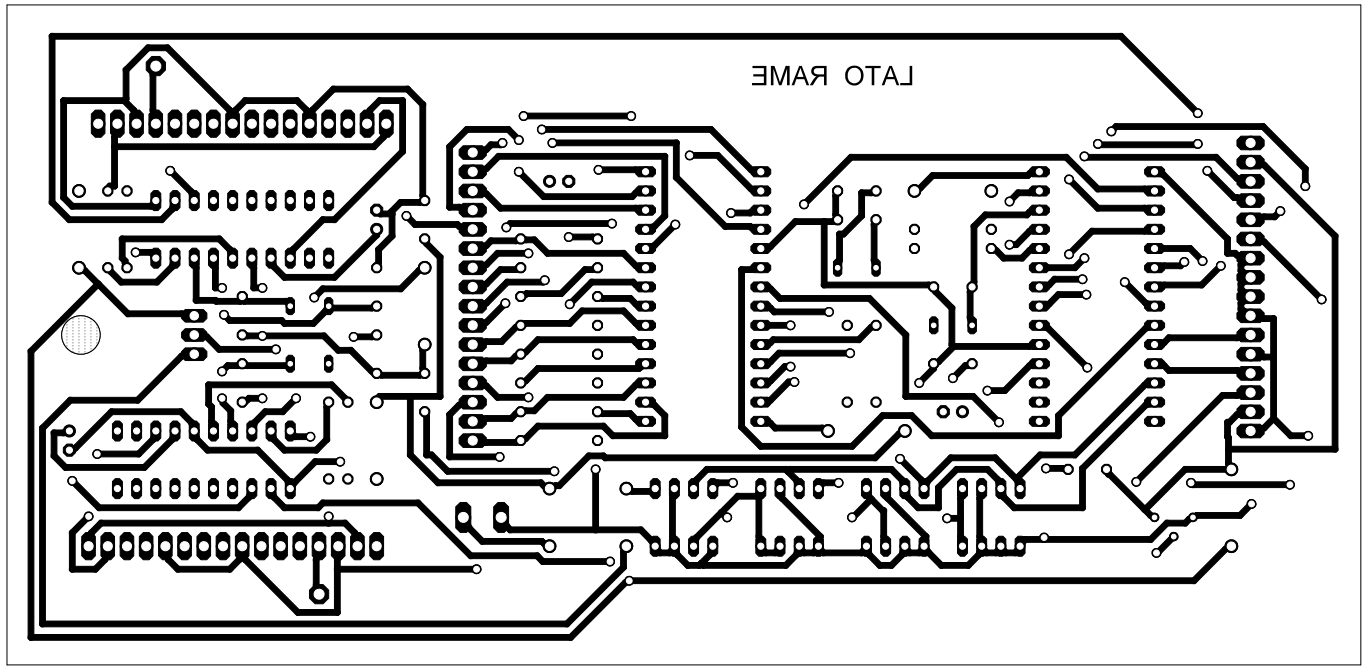
Programma principale

Il programma principale viene richiamato subito dopo la configurazione degli ingressi e delle uscite. Nel caso di una cattiva programmazione o di disturbi elettrici, tali da creare blocchi del programma o cicli di loop non previsti, l'ST6 si resetta automaticamente, grazie alla funzione di Watchdog.

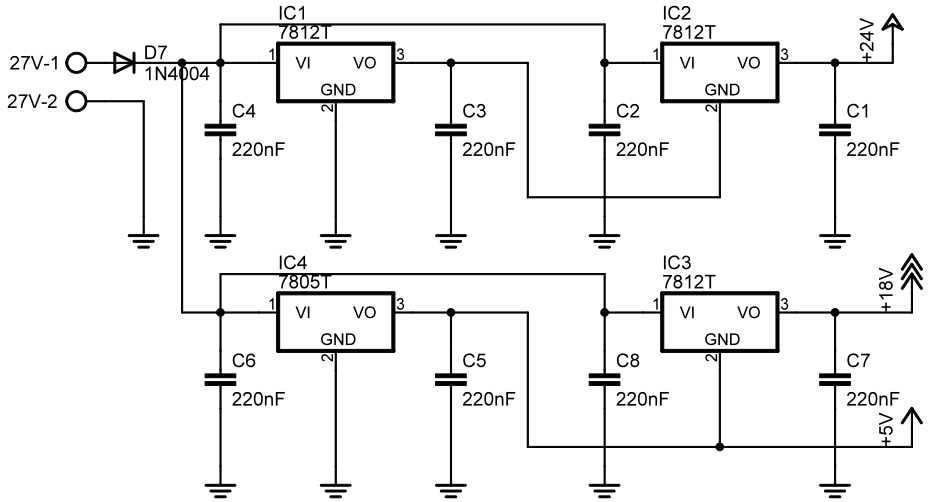

ALIMENTAZIONE
DA BATTERIA 9V
con protezione
della polarità
IN: 9V
OUT: 5,6V
Imax: 1A
IC0
7805T
9V-1
9V-2
D7
1N4004
C4
220nF
C3
220nF
D2
1N4004
+5V
DISPLAY LCD (16 car. x 2 righe) LM093XMLN
EN --> B0
R/S --> B1
DB4 --> C4
DB5 --> C5
DB6 --> C6
DB7 --> C7
D1
1N4004
C9
220nF
R13
10K
EEPROM 24C64
Array
EEPROM
24C64P
IC1
IC2
IC3
IC4
FIRMWARE:
TASTIERI.HEX
ST62T25
XTAL
8Mhz
22pF
C2
100K
1uF
C7 --> linea 1
C6 --> linea 2
C5 --> linea 3
C4 --> int.
IN ST6:
A7 --> 6
A6 --> 8
A5 --> 10
A4 --> 11
A3 --> 12
A2 --> 14
A1 --> 15
A0 --> 16
OUT ST6
B0 --> 1
B1 --> 2
B2 --> 3
B3 --> 4
B4 --> 5
B5 --> 7
B6 --> 9
B7 --> 13
TASTIERINO 50 tasti
(CME DIGIT 2000 702118A SAA 1250 SL)
CPU CENTRALE
FIRMWARE: CPU.HEX
ST62T1
XTAL1
C7
COM SERIALE:
A0 --> linea 1
A1 --> linea 2
A2 --> linea 3
A7 --> Interrupt
RX 433MHZ AUREL 8110-RXSTDLC
ST62T20
IC5
FIRMWARE:
RXFOT1.HEX
ANT1
C11
RX 866MHZ AUREL 8110-RX8L50SA70
IC6
FIRMWARE:
RXFOT2.HEX
ANT2
C14
Francesco Galgani
IPSIA Massa Marittima
palm
28/05/2005 09.17.50
Sheet: 1/1


ANT1
X5
X6
1N4004
D6
100K
R3
100K
R4
C11
7805T
IC0
C12
1uF
C3
220nF
8Mhz
C10
22pF
XTAL2
XTAL3
C14
C13
22pF
8Mhz
D2
1N4004
C4
220nF
1uF
C15
R10
100K
100K
D8
1N4004
X1
X2
C5
1uF
R5
R6
R7
R8
R9
R10
R11
R12
100K
9V
1
2
1N4004
D4
1N4004
D7
ST62T25
IC4
24C64P
IC3
24C64P
IC2
24C64P
22pF
C1
C2
XTAL
8Mhz
D5
1N4004
R2
100K
100K
22pF
8Mhz
XTAL1
C7
1uF
R1
100K
R11
100K
D3
1N4004
C8
ST62T10
IC1
X3
X4
220nF
C9
R13
10K
1N4004
D1
X8
X7
ANT2

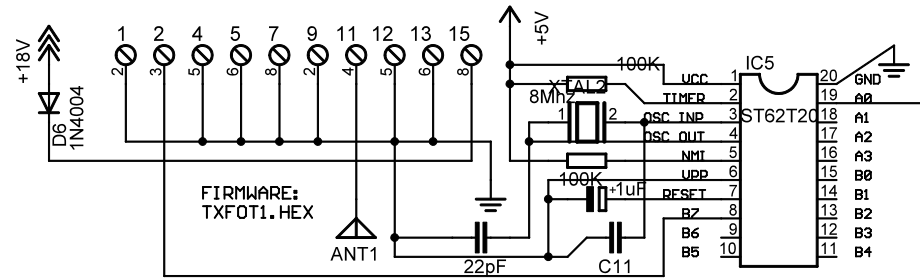
LATO COMPONENTI
DISP
TAST1
RX433
RX866
+
FG004 - Francesco Galgani 28 maggio 2005
ALIM + TAST + RX433 + RX866 + DISP + CPU + EEPROM


LATO RAME

ALIMENTAZIONE CON TRE BATTERIE DA 9V con protezione della polarità

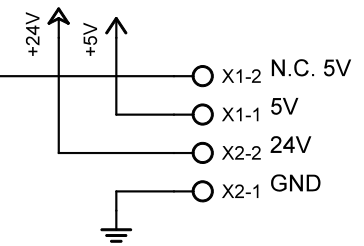


TX 433MHZ AUREL 8110-TX433BOOST

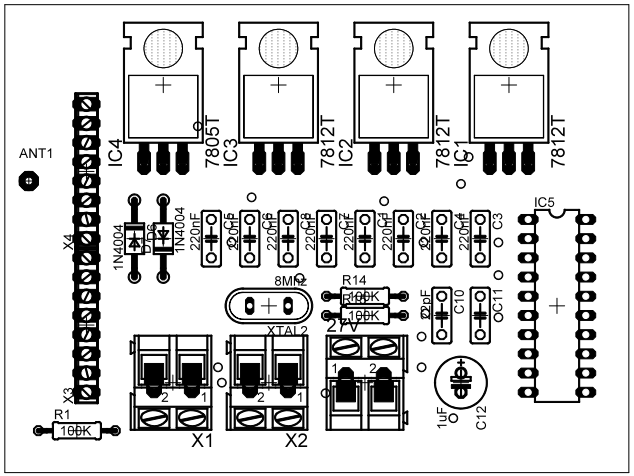


B7 = dati da trasmettere A0 = contatto N.C. fotocellula (5V)

CONTATTI PER FOTOCELLULA



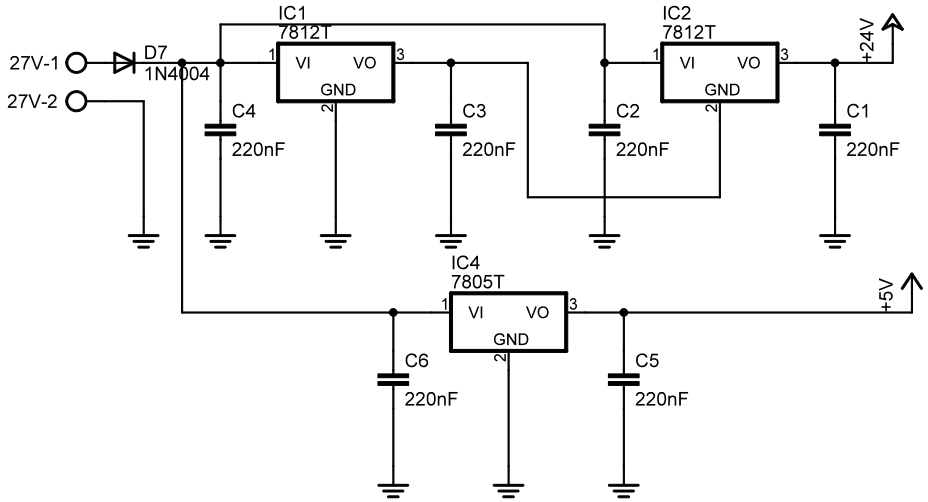
Francesco Galgani IPSIA Massa Marittima	tx433 17/06/2005 19.35.42 Sheet: 1/1


ANT1
IC4
7805T
IC3
7812T
IC2
7812T
IC1
7812T
IC5
X4
X3
1N4004
1N4004
220nF
C3
C10
C11
C12
8MHz
XTAL2
R14
100K
100K
27pF
1uF
R1
100K
X1
X2

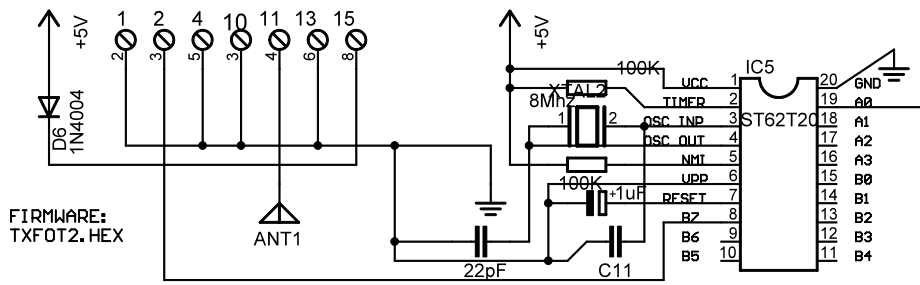
TX433
15
NC 5V 24V GND +27 GND
TX433 FOTI
FRANCESCO
GALGANI
LATO COMPONENTI

LATO RAME

ALIMENTAZIONE CON TRE BATTERIE DA 9V con protezione della polarità

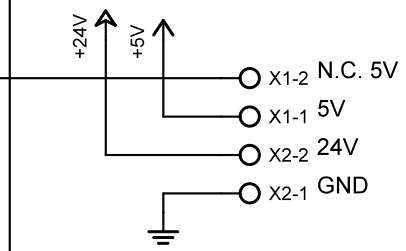


TX 866MHZ AUREL 8110-TX8LAVSA05



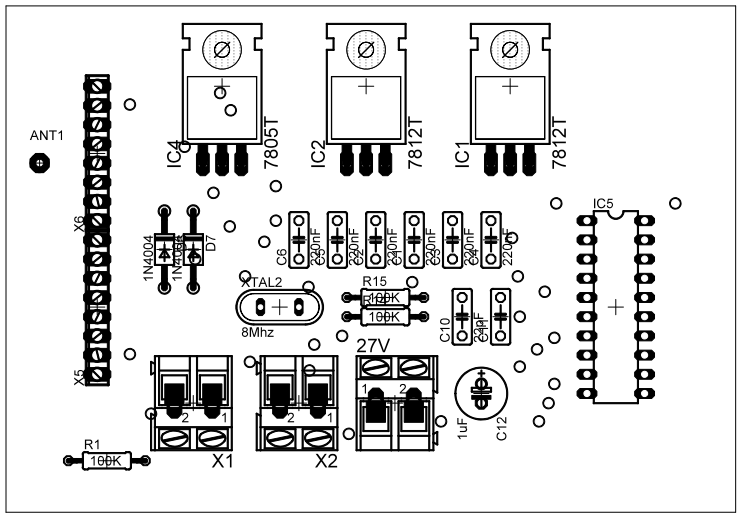
B7 = dati da trasmettere A0 = contatto N.C. fotocellula (5V)

CONTATTI PER FOTOCELLULA

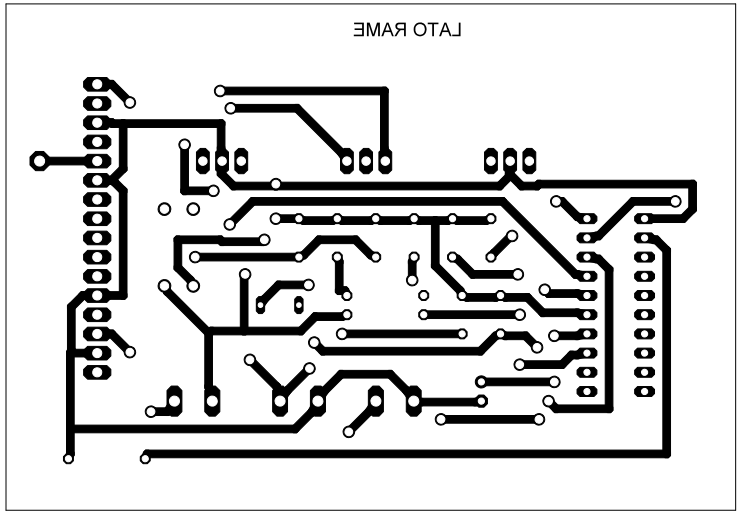


Francesco Galgani IPSIA Massa Marittima	tx866 17/06/2005 19.54.18 Sheet: 1/1

X6-a


ANT1
IC4
7805T
IC2
7812T
IC1
7812T
IC5
X6
X5
1N4004
D7
C6
220nF
XTAL2
8Mhz
R15
10ΘK
C10
24pF
27V
1uF
C12
R1
10ΘK
X1
X2

TX866
16
NC 5V 24V GND 27V
TX866 FOT2
LATO COMPONENTI
FRANCESCO
GALGANI


LATO RAME

```
;*****************************************************************************
;*                                                                           *
;*  FILE  SORGENTE  "CPU"                                                    *
;*                                                                           *
*
;*  INPUTS:  TASTIERINO, RICEVITORE 1 e 2, MEMORIA                           *
;*  OUTPUTS: DISPLAY, MEMORIA                                                *
;*                                                                           *
;*****************************************************************************

; Ricordate sempre che il cuore del programma da realizzare e' il cosiddetto
; PROGRAMMA PRINCIPALE ( MAIN ), che in genere, ma non obbligatoriamente, si
; inserisce alla fine del file. Le SUBROUTINE possono essere inserite
; subito prima del MAIN, anche se e' possibile posizionarle in QUALUNQUE
; punto del programma.
; Fate comunque attenzione che, sia il MAIN, sia le SUBROUTINE devono essere
; compresi fra i due comandi ".ORG" (inizio programma) e ".END" (fine del
; programma).
; In ogni parte del programma e' possibile inserire commenti di ogni tipo,
; purche' preceduti dal punto e virgola, come nel testo che state leggendo.
;

;*****************************************************************************
;*                      NOTE SUI CAMPI (FIELD) PER I COMANDI                 *
;*****************************************************************************

; 1° CAMPO
;      |___
;          | ETICHETTA; massimo 8 caratteri. Deve iniziare sempre con una
;          | lettera e deve partire dalla PRIMA COLONNA. Per passare
;          | al campo successivo e' preferibile usare il tasto TABULAZIONE.

;           2° CAMPO
;                |______
;                       | ISTRUZIONI; sono le istruzioni da far eseguire al
;                 | micro.
;                       | Per passare al campo successivo usare il
;                       | tasto TABULATORE.

;                      3° CAMPO
;                           |____
;                                | OPERANDI; sono gli operandi delle
;                                | istruzioni.

;                                4° CAMPO
;                                     |____
;                                          | COMMENTI; come gia' evidenziato
;                                          | non e' obbligatorio posizionarlo
;                                          | proprio in questa posizione, ma
;                                          | in tal modo consente di spiegare
;                                          | il significato delle varie
;                                          | operazioni.


;                              INIZIALIZZAZIONE
;                              ----------------

        .title  "CPU"

        .vers   "ST62E25C"     ; INDICA IL TIPO DI DISPOSITIVO DA PROGRAMMARE


;*****************************************************************************
;*                        VARIABILI DEL MICRO                                *
```

```
;***************************************************************************

; Questo elenco di variabili e' STANDARD, cioe' non dovrete mai modificarlo.

;      DEFINIZIONE DEGLI INDIRIZZI DI PARTENZA DELLE VARIE AREE DI MEMORIA
;      ------------------------------------------------------------------



;            DEFINIZIONE DELLE LOCAZIONI DEI 5 REGISTRI DEL MICRO
;            ---------------------------------------------------

a       .def    0ffh    ;ACCUMULATOR (REGISTRO ACCUMULATORE)
x       .def    080h    ;X REGISTER  (REGISTRO X)
y       .def    081h    ;Y REGISTER  (REGISTRO Y)
v       .def    082h    ;V REGISTER  (REGISTRO V)
w       .def    083h    ;W REGISTER  (REGISTRO W)


;              DEFINIZIONE DEI REGISTRI DELLE PORTE
;              ------------------------------------

port_a  .def    0c0h    ;PORT A DATA REGISTER
port_b  .def    0c1h    ;PORT B DATA REGISTER
port_c  .def    0c2h    ;PORT C DATA REGISTER
pdir_a  .def    0c4h    ;PORT A DIRECTION REGISTER  (0=Input   1=Output)
pdir_b  .def    0c5h    ;PORT B DIRECTION REGISTER
pdir_c  .def    0c6h    ;PORT C DIRECTION REGISTER
popt_a  .def    0cch    ;PORT A OPTION REGISTER
popt_b  .def    0cdh    ;PORT B OPTION REGISTER
popt_c  .def    0ceh    ;PORT C OPTION REGISTER


;              DEFINIZIONE DEL REGISTRO DEGLI INTERRUPT
;              ----------------------------------------

ior     .def    0c8h    ;INTERRUPT OPTION REGISTER.


;              DEFINIZIONE DEL REGISTRO PER IL CONVERTITORE A/D
;              -----------------------------------------------

addr    .def    0d0h    ;A/D DATA REGISTER (REGISTRO DATI DEL CONVERTITORE)
adcr    .def    0d1h    ;A/D CONTROL REGISTER (REGISTRO DI CONTROLLO)


;              DEFINIZIONE DEI REGISTRI DEL TIMER
;              ----------------------------------

psc     .def    0d2h    ;TIMER PRESCALER REGISTER (REGISTRO DEL PRESCALER)
tcr     .def    0d3h    ;TIMER DATA REGISTER (REGISTRO DATI)
tscr    .def    0d4h    ;TIMER TSCR REGISTER ( REGISTRO PER IL CONTROLLO
                  ;DELLO STATO DEL TIMER)


;              DEFINIZIONE DEL REGISTRO PER IL WATCH-DOG
;              -----------------------------------------

wdog    .def    0d8h    ;WATCHDOG REGISTER


;              DEFINIZIONE DEL REGISTRO PER LO SPAZIO DATI ROM
;              -----------------------------------------------
```

```
drw     .def    0c9h    ;DATA ROM WINDOW REGISTER

;*****************************************************************************
;*                           VARIABILI DEL PROGRAMMA                         *
;*****************************************************************************

; In questo punto del programma potete definire le variabili che intendete
; utilizzare.
; L'etichetta che le contraddistingue
; puo' essere qualunque, ma non ci possono essere due variabili con lo
; stesso nome. Notate che gli indirizzi sono consecutivi ed iniziano
; dall'indirizzo 084h, il primo disponibile della RAM.
; Si possono usare un massimo di 60 variabili.


puls     .def    084h                ; 01 memorizza pulsante premuto
save_a   .def    puls+1              ; 02 da usare per salvare accumulatore
save_x   .def    save_a+1            ; 03 da usare per salvare registro X


; VARIABILI PER DISPLAY

ddata    .def    puls+1              ; 04 variabile per scrittura su LCD
pos      .def    ddata+1             ; 05 posizione su LCD
del1     .def    pos+1               ; 06 variabile per fare un ritardo
del2     .def    del1+1              ; 07 variabile per fare un ritardo
del3     .def    del2+1              ; 08 variabile per fare un ritardo
ndel     .def    del3+1              ; 09 variabile per impostare un rit.
char0    .def    ndel+1              ; 10 variabile per carattere
char1    .def    char0+1             ; 11 variabile per carattere
char2    .def    char1+1             ; 12 variabile per carattere
char3    .def    char2+1             ; 13 variabile per carattere
savea1   .def    char3+1             ; 14 variabile per salvatagg.accumul.
savea2   .def    savea1+1            ; 15 variabile per salvatagg.accumul.
secrig   .def    savea2+1            ; 16 variabile per test su II riga
conta    .def    secrig+1            ; 17 variabile conta loop
conta1   .def    conta+1             ; 18 variabile conta loop
flag     .def    conta1+1            ; 19 variabile flag per on condition
vards    .def    flag+1              ; 20 variabile per shift DX o SX
work1    .def    vards+1             ; 21 variabile di comodo


; VARIABILI PER TIMER

timeh    .def    work1+1             ; 22 byte più significativo del timer
timem    .def    timeh+1             ; 23 byte intermedio del timer
timel    .def    timem+1             ; 24 byte meno significativo del timer
stst     .def    timel+1             ; 25 start/stop manuale o con fotocell.

ccl     .def     stst+1              ; 26 centesimi di secondo
cch     .def     ccl+1               ; 27
ssl     .def     cch+1               ; 28 secondi
ssh     .def     ssl+1               ; 29
mml     .def     ssh+1               ; 30 minuti
mmh     .def     mml+1               ; 31
hhl     .def     mmh+1               ; 32 ore
hhh     .def     hhl+1               ; 33
salvaa  .def     hhh+1               ; 34 salvataggio accumulatore
timer   .def     salvaa+1            ; 35 variabile di comodo per timer

disth   .def     timer+1             ; 36 distanza (byte più significativo)
distl   .def     disth+1             ; 37 distanza (byte meno significativo)

ch1     .def     distl+1             ; 38 caratteri
```

```
ch2     .def     ch1+1               ; 39
ch3     .def     ch2+1               ; 40
ch4     .def     ch3+1               ; 41
ch5     .def     ch4+1               ; 42
ch6     .def     ch5+1               ; 43
ch7     .def     ch6+1               ; 44
ch8     .def     ch7+1               ; 45
ch9     .def     ch8+1               ; 46
ch10    .def     ch9+1               ; 47
ch11    .def     ch10+1              ; 48
nch     .def     ch11+1              ; 49 numero di caratteri


; VARIABILI PER OPERAZIONI MATEMATICHE

res_msb  .def    nch+1               ; 50
res_lsb  .def    res_msb+1           ; 51
number_a .def    res_lsb+1           ; 52
number_b .def    number_a+1          ; 53

; divid_u  .def    number_b+1
; divid_h  .def    divid_u
; divid_l  .def    divid_h
; divis_h  .def    divid_l
; divis_l  .def    divis_h
; ris_h    .def    divis_l
; ris_l    .def    ris_h
; adc_h    .def    ris_l
; adc_l    .def    adc_h


;*************************************************************************
;*                          SETTAGGIO INIZIALE                           *
;*************************************************************************

; Le porte parallele, il timer, il convertitore A/D ed il watch dog debbono
; essere INIZIALIZZATI all'inizio del programma.
; Abbiamo indicato l'operazione di settaggio iniziale con INIZIO.


;             INIZIO DEL PROGRAMMA
;             --------------------

      .org    080h      ; E' DA UTILIZZARE NEI PROGRAMMI DI 4 Kb PER
                        ;    I MICRO ST62E20,ST62T20,ST62E25,ST62T25.

inizio                  ;  INIZIALIZZAZIONE DELLE PERIFERICHE


      ldi     wdog,0ffh       ; RICARICA IL WATCH DOG
; Questa operazione va ripetuta circa ogni 30 istruzioni per evitare che
; il microprocessore si resetti da solo.


        ldi     pdir_a,00000000b   ; A0,A1,A2 = ricevono pulsante premuto
        ldi     popt_a,10000000b   ; A3,A4,A5,A6 = fotocellule
        ldi     port_a,00000000b   ; A7 = interrupt del tastierino


        ldi     pdir_b,00010011b   ; B0 = LCD ENABLE
        ldi     popt_b,00010011b   ; B1 = LCD RS
        ldi     port_b,00010011b   ;
```

```
        ldi     pdir_c,11110000b   ; C4,C5,C6,C7 = dati trasmessi a LCD
        ldi     popt_c,11110000b
        ldi     port_c,11110000b


        ldi     ior,10h         ;Abilita tutti gli interrupt

      reti              ;RIPRISTINA I FLAG PRINCIPALI

        .w_on                   ; abilita la Data Rom Window

      jp      main            ;SALTA AL PROGRAMMA PRINCIPALE


;*************************************************************************
;*                        SUBROUTINE DI INTERRUPT                        *
;*************************************************************************

; In questo punto potremmo scrivere le subroutine di interrupt.
; Queste subroutine devono terminare con l'istruzione   reti.
; La subroutine deve percio' essere scritta tra l'etichetta che la
; identifica e l'istruzione reti.

ad_int                  ;INTERRUPT DEL CONVERTITORE A/D
      reti


tim_int                 ;INTERRUPT DEL TIMER (ogni centesimo di secondo)

        ldi     wdog,0ffh
        ldi     tcr,208
        ldi     tscr,01011101b
        dec     timer
        reti

BC_int                  ;INTERRUPT DELLE PORTE B e C
        reti


A_int                   ;INTERRUPT DELLA PORTA A

        ; riceve il pulsante premuto e lo memorizza in PULS
        ; A2 = b5 e b2
        ; A1 = b4 e b1
        ; A2 = b3 e b0

        ; Questa è la subroutine del RICEVITORE


        ldi     wdog,0ffh    ;                    |
        ld      save_a,a     ; salva registri     | da inizio interrupt
        ld      a,x          ; speciali           | a fine memorizzazione
        ld      save_x,a     ;                    | dei primi tre bit
                             ;                    | trascorrono almeno
        ld      a,port_a     ; carica b5, b4, b3  | 9x4=36 cicli
        rlc     a            ;                    |
        rlc     a            ;                    | (che è lo stesso tempo
        rlc     a            ;                    |  impiegato dal tx per
        andi    a,00111000b  ;                    |  inviare ultimi tre bit)
        ld      x,a          ;                    |

        nop
        nop
        nop
```

```
        nop
        nop
        nop     ; 6x2=12 cicli di delay (per precauzione)

        ld      a,port_a     ; carica b2, b1, b0
        andi    a,00000111b
        add     a,x

        ld      puls,a       ; memorizza in "puls" il numero del pulsante
                             ; premuto


        ldi     wdog,0ffh
        ldi     drw,pulsanti.w  ; carica la tabella con le codifiche ASCII
        ldi     x,pulsanti.d    ; carica in X l'indirizzo del primo byte

        ld      a,x
        add     a,puls
        dec     a
        ld      x,a             ; ottengo in X l'indirizzo della
                                ; codifica ASCII del pulsante

        ld      a,(x)           ; carico su A il valore puntato da X
        ld      puls,a          ; memorizzo la codifica del pulsante in PULS


        ld      a,save_x     ; ripristina registri speciali
        ld      x,a
        ld      a,save_a

      reti

nmi_int                 ;INTERRUPT NON MASCHERABILE

        ldi     wdog,0ffh
        call    incssl
        call    aggcron

      reti


;*************************************************************************
;*                          SUBROUTINE                                   *
;*************************************************************************

; Le SUBROUTINE sono dei pezzi di programmi che verranno utilizzate
; dal programma principale (MAIN) per svolgere determinate operazioni.
; Questi "sottoprogrammi" DEBBONO sempre iniziare con un'ETICHETTA e
; terminare con l'istruzione RET.



; --------------------------- SUBROUTINE N° 1 ----------------------------

; Pulisce tutta la RAM a partire da 080h fino a 0BFh

pulisci
      clr   a
      ldi   y,60  ; 60 = byte di RAM disponibili, più quattro registri
      ldi   x,084h  ; 084h = primo byte di RAM
clean
      ldi   wdog,255
      ld    (x),a
      inc   x
```

```
      dec   y
      jrnz  clean

      clr   x     ; cancellazione dei registri da 080h a 083h
      clr   y
      clr   v
      clr   w
      ret



; --------------------------- SUBROUTINE N° 2 ---------------------------

ritardo        ;NOME DELLA SUBROUTINE (QUANDO VORREMO RICHIAMARLA
              ;NEL PROGRAMMA PRINCIPALE SCRIVEREMO "call ritardo")

        ldi     y,128

rit2    ldi     x,255
rit1    ldi     wdog,0ffH       ; ricarica il watch dog
      dec     x               ; decrementa x
      jrnz    rit1            ; se x non e' zero salta a rit1

      dec     y         ; decrementa y
      jrnz    rit2            ; se y non e' zero salta a rit2

      ret               ; ritorna al programma principale.


ritardo2                        ; ritardo più lungo (caricare in X il
                                ; numero di volte che si vuole richiamare
                                ; la subroutine "ritardo")
        call    ritardo
        dec     w
        jrnz    ritardo2
        ret



; --------------------------- SUBROUTINE LCD ---------------------------

;+--------------------------------------------------------------------------+
;| inizializzazione display                                                 |
;+--------------------------------------------------------------------------+
initdsl                                 ;81 etichetta per inizializzare LCD
      res     1,port_b                ;82 RS = 0 (istruzione)
      ldi     port_c,00110000b        ;83 inizializza a 8 bit
      res     0,port_b                ;84 fronte salita enable LCD
      set     0,port_b                ;85 fronte discesa LCD
      call    delay                   ;86 esegue un ritardo
      call    delay                   ;87 e per averlo piu' lungo
      call    delay                   ;88 la routine delay viene
      call    delay                   ;89 richiamata piu' volte
      ldi     port_c,00110000b        ;90 reinizializza a 8 bit
      res     0,port_b                ;91 fronte salita enable LCD
      set     0,port_b                ;92 fronte discesa LCD
      call    delay                   ;93 esegue un ritardo
      ldi     port_c,00110000b        ;94 reinizializza a 8 bit
      res     0,port_b                ;95 fronte salita enable LCD
      set     0,port_b                ;96 fronte discesa LCD
      call    delay                   ;97 esegue un ritardo
      ldi     port_c,00100000b        ;98 abilita operazioni a 4 bit
      res     0,port_b                ;99 fronte salita enable LCD
      set     0,port_b                ;100 fronte discesa LCD
```

```
      call    delay                 ;101 esegue un ritardo
      ldi     ddata,00101000b       ;102 abilita l'LCD a 2 line
      call    dsend                 ;103 invia all'LCD
      ldi     ddata,00001110b       ;104 accende display
      call    dsend                 ;105 invia all'LCD
      ldi     ddata,00001000b       ;106 spegne display
      call    dsend                 ;107 invia il dato all'LCD
      ldi     ddata,00000001b       ;108 clear display,DD RAM e reset AC
      call    dsend                 ;109 invia all'LCD
      ldi     ddata,00000110b       ;110 DD RAM incr. / no display shift
      call    dsend                 ;111 invia all'LCD
      set     1,port_b              ;112 RS = 1 (dato)
      ret                           ;113 ritorna al programma principale

;+--------------------------------------------------------------------------+
;| scrittura su Display Alfanumerico                                        |
;+--------------------------------------------------------------------------+
dwr                                   ;114 etichetta per scrittura su LCD
      ldi     wdog,0ffh             ;115 ricarica il watch dog
      ld      a,v                   ;116 carica in a la posiz.di visual.
      ld      ddata,a               ;117 carica in "ddata" l'accumulatore
      set     7,ddata               ;118 segnala posiz.in DD-RAM
      res     1,port_b              ;119 RS = 0 (istruzione)
      call    dsend                 ;120 invia all'LCD
      set     1,port_b              ;121 RS = 1 (dato)
      call    delay                 ;122 esegui un ritardo
dwrx2   ldi     wdog,0ffh             ;123 ricarica il watch dog
      ld      a,(x)                 ;124 copia in a carattere da scrivere
      ld      ddata,a               ;125 copia in "ddata" l'accumulatore
      call    dsend                 ;126 invia all'LCD
dwr3    inc     x                     ;127 increm."x" (prossimo carattere)
      dec     y                     ;128 decrementa caratteri da scrivere
      jrz     dwr4                  ;129 arrivato a zero salta a "dwr4"
      jp      dwrx2                 ;130 salta a "drw2"
dwr4    ret                           ;131 ritorna al modulo chiamante


;+--------------------------------------------------------------------------+
;| Questa routine serve per inviare i dati all'LCD 4 bit alla volta       |
;+--------------------------------------------------------------------------+
dsend   ld      savea1,a              ;132 salva il contenuto dell'accumul.
      ld      a,ddata               ;233 carica in accum. "ddata"
      andi    a,11110000b           ;234 esegue AND
      ld      port_c,a              ;235 mette su porta c l'accumulatore
      res     0,port_b              ;236 fronte di discesa enable
      set     0,port_b              ;237 fronte di salita enable
      call    delay2                ;238 esegue un ritardo
      ld      a,ddata               ;239 carica in accum. "ddata"
      sla     a                     ;240 shift del dato di 4 posizioni
      sla     a                     ;241
      sla     a                     ;242
      sla     a                     ;243
      ld      port_c,a              ;244 mette su porta c l'accumulatore
      res     0,port_b              ;245 fronte di discesa enable
      set     0,port_b              ;246 fronte di salita enable
      call    delay2                ;247 esegue un ritardo
      ld      a,savea1              ;248 riprist. contenuto dell'accumul.
      ret                           ;249 ritorna

;+--------------------------------------------------------------------------+
;| ritardo di circa 400 millisecondi                                        |
;+--------------------------------------------------------------------------+
delay   ldi     del1,100              ;250 carica in "del1" 100
dela1   ldi     wdog,0feh             ;251 ricarica il watch dog
```

```
      dec     del1                    ;252 decrementa "del1"
      jrnz    dela1                   ;253 non e' arrivato a zero salta
      ret                             ;254 altrimenti ritorna

;+--------------------------------------------------------------------------+
;| ritardo di circa 80 millisecodi                                          |
;+--------------------------------------------------------------------------+
delay2  ldi     del2,05                 ;255 carica in "del2" 5
dela2   ldi     wdog,0feh               ;256 ricarica il watch dog
      call    delay                   ;257 esegue "delay"
      dec     del2                    ;258 decrementa "del2"
      jrnz    dela2                   ;259 non e' arrivato a zero salta
      ret                             ;260 altrimenti ritorna

;+--------------------------------------------------------------------------+
;| visualizza la scritta iniziale di presentazione                          |
;+--------------------------------------------------------------------------+
disp00  res     1,port_b                ;261 RS = 0 (istruzione)
      ldi     ddata,00001100b         ;262 disp ON, curs OFF, blink OFF
      call    dsend                   ;263 invia all' LCD
      call    delay2                  ;264 esegue un ritardo
      ldi     ddata,00000010b         ;265 return HOME disp curs DD RAM
      call    dsend                   ;266 invia all' LCD
      call    delay2                  ;267 esegue un ritardo
      ldi     ddata,00000110b         ;268 DD RAM incr. no disp shift
      call    dsend                   ;269 invia al LCD
      set     1,port_b                ;270 rs = 1 (dato)
      call    delay                   ;271 esegue un ritardo
        ldi     drw,msg01.w             ;272 carica il Data Reg.Window
        ldi     x,msg01.d               ;273 carica la stringa di presentaz.
      ldi     y,16                    ;274 tot. caratteri da visualizzare
      ldi     v,0                     ;275 posizione di partenza
      call    dwr                     ;276 esegue la visualizzazione
        ldi     x,msg02.d               ;277 carica la stringa di presentaz.
      ldi     y,16                    ;278 tot. caratteri da visualizzare
      ldi     v,64                    ;279 posizione di partenza
      call    dwr                     ;280 esegue la visualizzazione
      ret                             ;290 ritorna al modulo chiamante



;+--------------------------------------------------------------------------+
;| carica una scritta nella prima o nella seconda riga                      |
;+--------------------------------------------------------------------------+

sopra   ldi     wdog,0ffh
        ldi     y,16
        ldi     v,0
        call    dwr
        ret

sotto   ldi     wdog,0ffh
        ldi     y,16
        ldi     v,64
        call    dwr
        ret

cls     ldi     wdog,0ffh
        ldi     drw,msg05.w             ; cancella lo schermo
        ldi     x,msg07.d
        call    sopra
        ldi     x,msg08.d
        call    sotto
        ret
```

```
; ***************************************************************************


; --- Subroutine per attendere pressione start  (PULS=229)

waitst  ldi     wdog,0ffh

        call    cls
        ldi     drw,msg33.w      ; VISUALIZZO IL CRONOMETRO
        ldi     x,msg33.d
        call    sopra
        ldi     x,msg34.d
        call    sotto

        ldi     ior,10h


;       ldi     ior,00h
;       call    cls
;       ldi     drw,msg49.w      ; PREMI START...
;       ldi     x,msg51.d
;       call    sopra
;       ldi     ior,10h

tt1     ldi     wdog,0ffh
        ld      a,puls
        cpi     a,229
        jrnz    tt1
        ret

; --- Subroutine per attendere pressione stop  (PULS=230)

waitsto ldi     wdog,0ffh

        ld      a,puls
        cpi     a,230
        jrz     fine1

        ld      a,timer
        jrnz    waitsto

        ldi     timer,99
        call    incccl
        call    aggcron
        jp      waitsto

fine1   ret


; --- Subroutine per attendere oscuramento fotocellula 1 (A4=1 A3=0) START

waitfo1 ldi     wdog,0ffh

        call    cls
        ldi     drw,msg33.w      ; VISUALIZZO IL CRONOMETRO
        ldi     x,msg33.d
        call    sopra
        ldi     x,msg34.d
        call    sotto


;       ldi     ior,00h
;       call    cls
```

```
;       ldi     drw,msg49.w      ; ATTENDO FOTOCELLULA
;       ldi     x,msg52.d
;       call    sopra
;       ldi     ior,10h

tt2     ldi     wdog,0ffh
        ld      a,port_a
        andi    a,00011000b
        cpi     a,00010000b
        jrnz    tt2
        ret


; --- Subroutine per attendere oscuramento fotocellula 1 (A4=1 A3=0) STOP

waitfob1


tt2b    ldi     wdog,0ffh

        ld      a,port_a
        andi    a,00011000b
        cpi     a,00010000b
        jrz     fine2

        ld      a,timer
        jrnz    tt2b

        ldi     timer,100
        call    incccl
        call    aggcron
        jp      tt2b


fine2   ret



; --- Subroutine per attendere oscuramento fotocellula 2 (A6=1 A5=0) START

waitfo2 ldi     wdog,0ffh

        call    cls
        ldi     drw,msg33.w      ; VISUALIZZO IL CRONOMETRO
        ldi     x,msg33.d
        call    sopra
        ldi     x,msg34.d
        call    sotto


;       ldi     ior,00h
;       call    cls
;       ldi     drw,msg49.w      ; ATTENDO FOTOCELLULA
;       ldi     x,msg52.d
;       call    sopra
;       ldi     ior,10h

tt3     ldi     wdog,0ffh
        ld      a,port_a
        andi    a,01100000b
        cpi     a,01000000b
        jrnz    tt3
        ret
```

```
; --- Subroutine per attendere oscuramento fotocellula 2 (A6=1 A5=0) STOP

waitfob2


tt3b    ldi     wdog,0ffh
        call    aggcron
        ld      a,port_a
        andi    a,01100000b
        cpi     a,01000000b
        jrz     fine3

        ld      a,timer
        jrnz    tt3b

        ldi     timer,99
        call    incccl
        call    aggcron
        jp      tt3b


fine3   ret


; --- Conto gli interrupt del timer


incccl  ;ldi     wdog,0ffh      ; incremento ccl
                               ; (unità dei centesimi)
        ;inc     ccl
        ;ld      a,ccl
        ;cpi     a,10
        ;jrz     inccch
        ;jp      trec


inccch  ;ldi     wdog,0ffh      ; incremento cch
                               ; (decine dei centesimi)
        ;inc     cch
        ;ld      a,cch
        ;cpi     a,10
        ;jrz     incssl
        ;jp      trec



incssl  ldi     wdog,0ffh      ; incremento ssl
                               ; (unità dei secondi)
        inc     ssl
        ld      a,ssl
        cpi     a,10
        jrz     incssh
        jp      trec

incssh  ldi     wdog,0ffh      ; incremento ssh
        clr     ssl            ; (decine dei secondi)
        inc     ssh
        ld      a,ssh
        cpi     a,6
        jrz     incmml
        jp      trec

incmml  ldi     wdog,0ffh      ; incremento mml
```

```
        clr     ssh             ; (unità dei minuti)
        inc     mml
        ld      a,mml
        cpi     a,10
        jrz     incmmh
        jp      trec

incmmh  ldi     wdog,0ffh       ; incremento mmh
        clr     mml             ; (decine dei minuti)
        inc     mmh
        ld      a,mmh
        cpi     a,6
        jrz     inchhl
        jp      trec

inchhl  ldi     wdog,0ffh       ; incremento hhl
        clr     mmh             ; (unità delle ore)
        inc     hhl
        ld      a,hhl
        cpi     a,10
        jrz     inchhh
        jp      trec

inchhh  ldi     wdog,0ffh       ; incremento hhh
        clr     hhl             ; (decine delle ore)
        inc     hhh
        ld      a,hhh
        cpi     a,10
        jrz     incclr
        jp      trec

incclr  clr     hhh

trec    inc     timel           ; incrementa i tre byte
        jrz     itimem          ; che contengono il numero dei secondi
        jp      vaivia          ; (da usare per calcolo della velocità)
itimem  inc     timem
        jrz     itimeh
        jp      vaivia
itimeh  inc     timeh

vaivia  ret


; --- Visualizzo il trascorrere del tempo sul cronometro

aggcron
agg1    ldi     wdog,0ffh    ; visualizzo ssl
        ld      a,ssl
        addi    a,48
        ld      work1,a
        ldi     x,work1
        ldi     y,1
        ldi     v,73
        call    dwr

agg2    ldi     wdog,0ffh    ; visualizzo ssh
        ld      a,ssh
        addi    a,48
        ld      work1,a
        ldi     x,work1
        ldi     y,1
        ldi     v,72
        call    dwr
```

```
agg3    ldi     wdog,0ffh    ; visualizzo mml
        ld      a,mml
        addi    a,48
        ld      work1,a
        ldi     x,work1
        ldi     y,1
        ldi     v,70
        call    dwr

agg4    ldi     wdog,0ffh    ; visualizzo mmh
        ld      a,mmh
        addi    a,48
        ld      work1,a
        ldi     x,work1
        ldi     y,1
        ldi     v,69
        call    dwr

agg5    ldi     wdog,0ffh    ; visualizzo hhl
        ld      a,hhl
        addi    a,48
        ld      work1,a
        ldi     x,work1
        ldi     y,1
        ldi     v,67
        call    dwr

agg6    ldi     wdog,0ffh    ; visualizzo hhh
        ld      a,hhh
        addi    a,48
        ld      work1,a
        ldi     x,work1
        ldi     y,1
        ldi     v,66
        call    dwr

        ret


; ---- Subroutine per inserire cinque caratteri numerici (distanza)

scrivi5 ldi     wdog,0ffh
        ldi     ch1,32  ; inserisco spazi
        ldi     ch2,32
        ldi     ch3,32
        ldi     ch4,32
        ldi     ch5,32
        ldi     nch,5     ; numero di caratteri numerici

waitch  ldi     wdog,0ffh

        clr     puls
        ldi     w,2
        call    ritardo2

        ld      a,puls
        cpi     a,234   ;OK
        jrnz    ckann
        jp      end1
ckann   cpi     a,235   ;ANNULLA
        jrnz    cknum
        jp      scrivi5
```

```
cknum
        cpi     a,48      ; un carattere numerico è >=48 e <58 in ASCII
        jrnc    cknum2
        jp      waitch
cknum2  cpi     a,58
        jrc     memch
        jp      waitch

memch   ldi     wdog,0ffh ; dopo i vari controlli, memorizza il carattere
        ld      v,a       ; metto il pulsante in v
        ld      a,nch     ; carico in a la posizione del carattere

        cpi     a,5
        jrnz    memch4
        ld      a,v
        ld      ch1,a
        jp      visual

memch4  cpi     a,4
        jrnz    memch3
        ld      a,v
        ld      ch2,a
        jp      visual


memch3  cpi     a,3
        jrnz    memch2
        ld      a,v
        ld      ch3,a
        jp      visual


memch2  cpi     a,2
        jrnz    memch1
        ld      a,v
        ld      ch4,a
        jp      visual

memch1  ldi     wdog,0ffh
        ld      a,v
        ld      ch5,a
        jp      visual


visual  ldi     x,ch1     ; visualizza i caratteri
        ldi     y,5
        ldi     v,69
        call    dwr

        dec     nch
        jrz     end1
        jp      waitch

end1    ld      a,ch5
        cpi     a,32
        jrnz    end2
        jp      scrivi5

end2    ldi     wdog,0ffh
        ld      a,puls
        cpi     a,234   ;OK
        jrnz    ckann2
        jp      end3
ckann2  cpi     a,235   ;ANNULLA
```

```
        jrz     ckfin
        jp      end2
ckfin   jp      scrivi5


end3    ret


; --- Moltiplicazione

mul      ldi    wdog,0ffh         ; res_msb = byte più sign. del risultato
         clr    res_msb           ; res_lsb = bute meno sign. del risultato
         clr    res_lsb
         ld     a,number_b        ; number_a = primo fattore
         jrnz   mul1              ; number_b = secondo fattore

end_mul      ret

mul1     ldi    wdog,0ffh
         ld     a,number_a
         jrz    end_mul
         ld     a,number_b
         add    a,res_lsb
         ld     res_lsb,a
         jrnc   mul2
         inc    res_msb

mul2     ldi    wdog,0ffh
         dec    number_a
         jp     mul1




; *** MACRO PER TASTI DI SCELTA RAPIDA

.macro  tastir
        ldi     wdog,0ffh        ; controlla la pressione di un tasto di
        ld      a,puls           ; scelta rapida
        cpi     a,224
        jrnz    $+3
        jp      cronomet
        cpi     a,49
        jrnz    $+3
        jp      cronomet
        ldi     wdog,0ffh
        cpi     a,225
        jrnz    $+3
        jp      elencnum
        cpi     a,50
        jrnz    $+3
        jp      elencnum
        ldi     wdog,0ffh
        cpi     a,226
        jrnz    $+3
        jp      riceraid
        cpi     a,51
        jrnz    $+3
        jp      riceraid
        ldi     wdog,0ffh
        cpi     a,227
        jrnz    $+3
```

```
        jp      testfoto
        cpi     a,52
        jrnz    $+3
        jp      testfoto
        cpi     a,228
        jrnz    $+3
        jp      testalto
        cpi     a,53
        jrnz    $+3
        jp      testalto
        cpi     a,233
        jrnz    $+3
        jp      infoauto
.endm


;*************************************************************************
;*                          PROGRAMMA PRINCIPALE                         *
;*************************************************************************

; Il programma principale iniziera' con l'etichetta main


;     ----------------------------------------------------------------
;      | ISTRUZIONI PER L'UTILIZZO DEL DISPLAY                        |
;      |                                                              |
;      | X = indirizzo del primo byte da scrivere sul display         |
;      | Y = numero di byte (cioè di caratteri) da caricare           |
;      | V = posizione di partenza ("0" per I riga, "64" per II riga) |
;      |                                                              |
;      | "call dwr" per inviare al display                            |
;     ----------------------------------------------------------------
;
;        ESEMPIO:
;
;        ldi     a,44                    ;283 carica il valore ASCII di ","
;        ld      (x),a                   ;284 carica in tabella l'accum.
;        ldi     sette,7                 ;285 carica il val. 7 (| > CG.RAM)
;        ldi     x,sette                 ;286 carica indir. di "sette" in "x"
;        ldi     y,1                     ;287 nr. caratteri da visualizz.
;        ldi     v,64                    ;288 posiz. di visualizz. (II riga)
;        call    dwr                     ;289 esegue la visualizzazione


main                    ;IDENTIFICA L'INIZIO DEL PROGRAMMA PRINCIPALE

        ldi     wdog,0ffh       ;RICARICA IL WATCH DOG

        call    pulisci         ;Pulisce tutta la RAM
                                ;(poiché contiene informazioni casuali)


; ***** INIZIO MESSAGGI ALL'ACCENSIONE *****

        ldi     ior,00h
        call    delay                   ; ritardo iniziale
        call    initdsl                 ; inizializza l'LCD
        call    delay

        ldi     wdog,0ffh               ; ricarica il watchdog
        call    disp00                  ; visualizza msg 1 e 2

        ldi     w,6                     ; imposta ritardo con w=6 (vedi sub)
```

```
        call    ritardo2

        call    cls                     ; cancella lo schermo
        call    ritardo

        ldi     drw,msg01.w             ; ricarica tabella
        ldi     x,msg03.d               ; visualizza messaggio 3 e 4
        call    sopra
        ldi     x,msg04.d
        call    sotto

        ldi     w,4
        call    ritardo2

        call    cls                     ; cancella lo schermo

        ldi     drw,msg05.w             ; visualizza messaggio 5 e 6
        ldi     x,msg05.d
        call    sopra
        ldi     x,msg06.d
        call    sotto

        ldi     w,3
        call    ritardo2

        call    cls                     ; cancella lo schermo
        call    ritardo



;chr     ; permette di visualizzare l'ultimo pulsante premuto
;
;       ldi     wdog,0ffh
;       ldi     x,puls      ; carico su X l'indirizzo del byte PULS
;                           ; che contiene la codifica ASCII dell'ultimo
;                           ; pulsante premuto
;       ldi     y,1
;       ldi     v,75
;       call    dwr
;       jp      chr


; ***** FINE MESSAGGI ALL'ACCENSIONE *****


; ***** INIZIO ELENCO FUNZIONI *****

; Freccia in alto:  B --> 66
; Freccia in basso: 0 --> 48

funz1   ldi     ior,00h         ; disabilita interrupt
        ldi     wdog,0ffh
        ldi     drw,msg09.w     ; ELENCO FUNZIONI
        ldi     x,msg09.d       ;   (USA FRECCE)
        call    sopra
        ldi     x,msg10.d
        call    sotto
        ldi     ior,10h         ; abilita interrupt

        clr     puls            ; resetta il registro del pulsante
fb1     ldi     wdog,0ffh
        tastir                  ; controlla pressione tasto di scelta rapida
        ldi     wdog,0ffh       ; attende freccia in basso
        ld      a,puls
```

```
        cpi     a,48
        jrz     funz2
        jp      fb1


funz2   ldi     ior,00h
        ldi     wdog,0ffh
        ldi     drw,msg09.w     ; 1-CRONOMETRO
        ldi     x,msg11.d       ; 2-ELENCO NUMERIC
        call    sopra
        ldi     x,msg12.d
        call    sotto
        ldi     ior,10h



        clr     puls            ; resetta il registro del pulsante
fa2     ldi     wdog,0ffh
        tastir                  ; controlla pressione tasto di scelta rapida
        ldi     wdog,0ffh       ; attende freccia in alto
        ld      a,puls
        cpi     a,66
        jrnz    fb2
        jp      funz1


fb2     ldi     wdog,0ffh       ; attende freccia in basso
        ld      a,puls
        cpi     a,48
        jrz     funz3
        jp      fa2


funz3   ldi     ior,00h
        ldi     wdog,0ffh
        ldi     drw,msg13.w     ; 3-RICERCA ID
        ldi     x,msg13.d       ; 4-TEST FOTOCELL.
        call    sopra
        ldi     x,msg14.d
        call    sotto
        ldi     ior,10h


        clr     puls            ; resetta il registro del pulsante
fa3     ldi     wdog,0ffh
        tastir                  ; controlla pressione tasto di scelta rapida
        ldi     wdog,0ffh       ; attende freccia in alto
        ld      a,puls
        cpi     a,66
        jrnz    fb3
        jp      funz2


fb3     ldi     wdog,0ffh       ; attende freccia in basso
        ld      a,puls
        cpi     a,48
        jrz     funz4
        jp      fa3


funz4   ldi     ior,00h
        ldi     wdog,0ffh
        ldi     drw,msg13.w     ; 5-TEST ALTOPARL.
        ldi     x,msg15.d       ; 0-INFO AUTORE
```

```
        call    sopra
        ldi     drw,msg17.w
        ldi     x,msg20.d
        call    sotto
        ldi     ior,10h


        clr     puls            ; resetta il registro del pulsante
fa4     ldi     wdog,0ffh
        tastir                  ; controlla pressione tasto di scelta rapida

        ldi     wdog,0ffh       ; attende freccia in alto
        ld      a,puls
        cpi     a,66
        jrnz    fa4b
        jp      funz3
fa4b    jp      fa4


; ***** FINE ELENCO FUNZIONI *****



elencnum        ldi     wdog,0ffh       ; funzione elenco numerico

riceraid        ldi     wdog,0ffh       ; funzione ricerca id

testfoto        ldi     wdog,0ffh       ; funzione test fotocellule

testalto        ldi     wdog,0ffh       ; funzione test segnale sonoro (altop.)


; *** FUNZIONE CRONOMETRO


; --- SELEZIONE START

cronomet        clr     stst
                ldi     ior,00h
                ldi     wdog,0ffh       ; funzione cronometro
                ldi     drw,msg21.w     ; START MANUALE O
                ldi     x,msg21.d       ; CON FOTOCELLULA?
                call    sopra
                ldi     x,msg22.d
                call    sotto
                ldi     ior,10h


                clr     puls            ; l'utente sceglie start manuale
ckcr            ldi     wdog,0ffh       ; o con fotocellula
                ld      a,puls
                cpi     a,229
                jrnz    ckcr2
                jp      startman        ; START MANUALE
ckcr2           cpi     a,231
                jrnz    ckcr3
                jp      startfo1        ; START FOTOCELLULA 1
ckcr3           cpi     a,232
                jrz     ckcr4
                jp      ckcr
ckcr4           jp      startfo2        ; START FOTOCELLULA 2
```

```
startman        ldi     ior,00h
                ldi     wdog,0ffh
                call    cls
                ldi     drw,msg25.w     ; START MANUALE
                ldi     x,msg27.d
                call    sopra
                res     7,stst
                ldi     ior,10h

                ldi     w,3
                call    ritardo2
                jp      stop


startfo1        ldi     ior,00h
                ldi     wdog,0ffh
                call    cls
                ldi     drw,msg29.w     ; START CON FOTOC1
                ldi     x,msg29.d
                call    sopra
                set     7,stst
                res     6,stst

                ldi     drw,msg21.w     ; TEST FOTOC. 1...
                ldi     x,msg23.d
                call    sotto
                ldi     w,3
                call    ritardo2

                ldi     wdog,0ffh
                ld      a,port_a        ; A6 A5
                andi    a,01100000b     ; 0  0   fotocell. NON RILEVATA
                cpi     a,01000000b     ; 0  1   fotocell. non oscurata
                jrz     oscur1          ; 1  0   fotocell. oscurata
                cpi     a,00100000b     ; 1  1   fotocell. NON RILEVATA
                jrz     noosc1
                jp      spent1

oscur1          jp      oscura1         ; fotocellula 1 oscurata
noosc1          jp      nooscu1         ; fotocellula 1 non oscurata
spent1          ldi     wdog,0ffh       ; fotocellula 1 spenta
                ldi     ior,00h
                call    cls
                ldi     drw,msg25.w     ; FOTOC. SPENTA?!
                ldi     x,msg26.d
                call    sotto
                ldi     w,4
                call    ritardo2
                call    cls
                jp      funz1           ; ritorna al menù iniziale

oscura1
nooscu1         ldi     wdog,0ffh
                ldi     ior,00h
                call    cls
                ldi     drw,msg25.w     ; FOTOC. (DI START) RILEVATA!
                ldi     x,msg25.d
                call    sotto
                ldi     w,3
                call    ritardo2
                jp      stop
```

```
startfo2        ldi     wdog,0ffh
                ldi     ior,00h
                call    cls
                ldi     drw,msg45.w     ; START CON FOTOC2
                ldi     x,msg45.d
                call    sopra
                set     7,stst
                set     6,stst
                ldi     w,3
                call    ritardo2


                ldi     ior,00h
                ldi     drw,msg21.w     ; TEST FOTOC. 2...
                ldi     x,msg24.d
                call    sotto
                ldi     w,3
                call    ritardo2


                ldi     wdog,0ffh
                ld      a,port_a        ; A4 A3
                andi    a,00011000b     ; 0  0   fotocell. NON RILEVATA
                cpi     a,00010000b     ; 0  1   fotocell. non oscurata
                jrz     oscur2          ; 1  0   fotocell. oscurata
                cpi     a,00001000b     ; 1  1   fotocell. NON RILEVATA
                jrz     noosc2
                jp      spent1



oscur2          jp      oscura1         ; fotocellula 2 oscurata
noosc2          jp      nooscu1         ; fotocellula 2 non oscurata



; --- SELEZIONE STOP

stop            ldi     ior,00h
                ldi     wdog,0ffh
                call    cls
                ldi     drw,msg45.w     ; STOP MANUALE O
                ldi     x,msg47.d       ; CON FOTOCELLULA?
                call    sopra
                ldi     x,msg48.d
                call    sotto
                ldi     ior,10h


                clr     puls            ; l'utente sceglie stop manuale
sckcr           ldi     wdog,0ffh       ; o con fotocellula
                ld      a,puls
                cpi     a,230
                jrnz    sckcr2
                jp      stopman         ; STOP MANUALE
sckcr2          cpi     a,231
                jrnz    sckcr3
                jp      stopfo1         ; STOP FOTOCELLULA 1
sckcr3          cpi     a,232
                jrz     sckcr4
                jp      sckcr
sckcr4          jp      stopfo2         ; STOP FOTOCELLULA 2
```

```
stopman         ldi     ior,00h
                ldi     wdog,0ffh
                call    cls
                ldi     drw,msg25.w     ; STOP MANUALE
                ldi     x,msg28.d
                call    sopra
                res     5,stst
                ldi     w,3
                call    ritardo2
                jp      distanza



stopfo1         ldi     ior,00h
                ldi     wdog,0ffh
                call    cls
                ldi     drw,msg45.w     ; STOP CON FOTOC1
                ldi     x,msg46.d
                call    sopra
                set     5,stst
                res     4,stst

                ldi     drw,msg21.w     ; TEST FOTOC. 1...
                ldi     x,msg23.d
                call    sotto
                ldi     w,3
                call    ritardo2

                ldi     wdog,0ffh
                ld      a,port_a        ; A6 A5
                andi    a,01100000b     ; 0  0   fotocell. NON RILEVATA
                cpi     a,01000000b     ; 0  1   fotocell. non oscurata
                jrz     oscur3          ; 1  0   fotocell. oscurata
                cpi     a,00100000b     ; 1  1   fotocell. NON RILEVATA
                jrz     noosc3
                jp      spent3

oscur3          jp      oscura3         ; fotocellula 1 oscurata
noosc3          jp      nooscu3         ; fotocellula 1 non oscurata
spent3          ldi     wdog,0ffh       ; fotocellula 1 spenta
                call    cls
                ldi     drw,msg25.w     ; FOTOC. SPENTA?!
                ldi     x,msg26.d
                call    sotto
                ldi     w,4
                call    ritardo2
                call    cls
                jp      funz1           ; ritorna al menù iniziale

oscura3
nooscu3         ldi     wdog,0ffh
                call    cls
                ldi     drw,msg25.w     ; FOTOC. (DI STOP) RILEVATA!
                ldi     x,msg25.d
                call    sotto
                ldi     w,3
                call    ritardo2
                jp      distanza

stopfo2         ldi     wdog,0ffh
                call    cls
                ldi     drw,msg29.w     ; STOP CON FOTOC2
```

```
                ldi     x,msg30.d
                call    sopra
                set     5,stst
                set     4,stst
                ldi     w,3
                call    ritardo2


                ldi     drw,msg21.w     ; TEST FOTOC. 2...
                ldi     x,msg24.d
                call    sotto
                ldi     w,3
                call    ritardo2


                ldi     wdog,0ffh
                ld      a,port_a        ; A4 A3
                andi    a,00011000b     ; 0  0   fotocell. NON RILEVATA
                cpi     a,00010000b     ; 0  1   fotocell. non oscurata
                jrz     oscur4          ; 1  0   fotocell. oscurata
                cpi     a,00001000b     ; 1  1   fotocell. NON RILEVATA
                jrz     noosc4
                jp      spent3


oscur4          jp      oscura3         ; fotocellula 2 oscurata
noosc4          jp      nooscu3         ; fotocellula 2 non oscurata



; --- INSERISCI DISTANZA

distanza        ldi     wdog,0ffh
                call    cls
                ldi     drw,msg29.w     ; DISTANZA IN MT.:
                ldi     x,msg31.d
                call    sopra

                ldi     ior,10h
                call    scrivi5         ; sub per inserire cinque caratteri
                ldi     ior,00h

                ldi     w,2             ; !!! Inserire qui la possibilità
                call    ritardo2        ; !!! di scrivere e mem. distanza


; --- PREMI OK PER INIZIARE

oktostar        ldi     ior,00h
                ldi     wdog,0ffh
                call    cls
                ldi     drw,msg29.w     ; OK PER INIZIARE
                ldi     x,msg32.d
                call    sopra
                ldi     drw,msg49.w     ; ANNUL PER USCIRE
                ldi     x,msg50.d
                call    sotto
                ldi     ior,10h


                clr     puls
waitok          ldi     wdog,0ffh
                ld      a,puls
                cpi     a,234
```

```
                jrz     stro2
                cpi     a,235
                jrz     goout
                jp      waitok
stro2           jp      startcro

goout           ldi     wdog,0ffh
                call    cls
                ldi     drw,msg49.w      ; ANNULLA...
                ldi     x,msg49.d
                call    sopra
                ldi     w,3
                call    ritardo2
                jp      funz1




; --- START CRONOMETRO (INIZIA A CONTARE IL TEMPO)

startcro        ldi     wdog,0ffh
                ldi     ior,10h


                ; Il registro STST definisce lo start e lo stop
                ; b7=0 start manuale
                ; b7=1 start con fotocellula
                ; b6=0 start con fotocellula 1
                ; b6=1 start con fotocellula 2
                ; b5=0 stop  manuale
                ; b5=1 stop  con fotocellula
                ; b4=0 stop  con fotocellula 1
                ; b4=1 stop  con fotocellula 2

                ld      a,stst
                sla     a
                jrnc    staman2
                sla     a
                jrnc    stafx1
                jp      stafx2

; start manuale (attendi pressione F6, cioè PULS=229)

staman2         ldi     wdog,0ffh
                call    waitst
                jp      parti


; start con fotocellula 1 (attende che fotocellula 1 sia oscurata
;                          cioè che A4=1 A3=0)

stafx1          ldi     wdog,0ffh
                call    waitfo1
                jp      parti


; start con fotocellula 2 (attende che fotocellula 2 sia oscurata
;                          cioè che A6=1 A5=0)

stafx2          ldi     wdog,0ffh
                call    waitfo2
                jp      parti
```

```
; stop manuale (attendo pressione F7, cioè PULS=230)

stxpman         ldi     wdog,0ffh
                call    waitsto
                jp      fermo


; stop con fotocellula 1 (attendo che fotocellula 1 sia oscurata
;                         cioè che A4=1 A3=0)

stpfo1          ldi     wdog,0ffh
                call    waitfob1
                jp      fermo


; stop con fotocellula 2 (attendo che fotocellula 2 sia oscurata
;                         cioè che A6=1 A5=0)

stpfo2          ldi     wdog,0ffh
                call    waitfob2
                jp      fermo



parti           ; parte il cronometro!!!

                ldi     wdog,0ffh
                ldi     timer,99


                clr     timeh
                clr     timem
                clr     timel


                clr     ccl
                clr     cch
                clr     ssl
                clr     ssh
                clr     mml
                clr     mmh
                clr     hhl
                clr     hhh


                ldi     tcr,208
                ldi     tscr,01011101b

                jrr     5,stst,stxpman
                jrr     4,stst,stpfo1
                jp      stpfo2


fermo           ; ferma il cronometro!!!

                ldi     wdog,0ffh

                ldi     tscr,0

                call    cls
                ldi     drw,msg33.w    ; time hh:mm:ss:cc
                ldi     x,msg35.d
                call    sopra
```

```
                ldi     a,100
                sub     a,timer
addcch          ldi     wdog,0ffh
                cpi     a,10            ; resetta il carry se a>=10
                jrc     minore
                inc     cch             ; imposta le decine dei centesimi di sec
                subi    a,10
                jp      addcch
minore          ld      ccl,a           ; imposta le unità dei centesimi di sec


                ldi     wdog,0ffh       ; visualizzo ccl
                ld      a,ccl
                addi    a,48
                ld      work1,a
                ldi     x,work1
                ldi     y,1
                ldi     v,15
                call    dwr

                ldi     wdog,0ffh       ; visualizzo cch
                ld      a,cch
                addi    a,48
                ld      work1,a
                ldi     x,work1
                ldi     y,1
                ldi     v,14
                call    dwr


                ldi     wdog,0ffh       ; visualizzo ssl
                ld      a,ssl
                addi    a,48
                ld      work1,a
                ldi     x,work1
                ldi     y,1
                ldi     v,12
                call    dwr

                ldi     wdog,0ffh       ; visualizzo ssh
                ld      a,ssh
                addi    a,48
                ld      work1,a
                ldi     x,work1
                ldi     y,1
                ldi     v,11
                call    dwr

                ldi     wdog,0ffh       ; visualizzo mml
                ld      a,mml
                addi    a,48
                ld      work1,a
                ldi     x,work1
                ldi     y,1
                ldi     v,9
                call    dwr

                ldi     wdog,0ffh       ; visualizzo mmh
                ld      a,mmh
                addi    a,48
                ld      work1,a
                ldi     x,work1
```

```
                ldi     y,1
                ldi     v,8
                call    dwr

                ldi     wdog,0ffh    ; visualizzo hhl
                ld      a,hhl
                addi    a,48
                ld      work1,a
                ldi     x,work1
                ldi     y,1
                ldi     v,6
                call    dwr

                ldi     wdog,0ffh    ; visualizzo hhh
                ld      a,hhh
                addi    a,48
                ld      work1,a
                ldi     x,work1
                ldi     y,1
                ldi     v,5
                call    dwr



finale
                ldi     drw,msg33.w    ; d:  xxxxx  metri
                ldi     x,msg36.d
                call    sotto


                ldi     x,ch1          ; visualizza distanza
                ldi     y,5
                ldi     v,69
                call    dwr



                ldi     w,3
                call    ritardo2

;               ldi     drw,msg05.w    ; cancella seconda riga
;               ldi     x,msg07.d
;               call    sotto

                ldi     drw,msg37.w    ; v: xxxxx m/s
                ldi     x,msg37.d
                call    sotto

                ldi     w,3
                call    ritardo2
                jp      finale



; *** INFO AUTORE

infoauto        ldi     ior,00h
                ldi     wdog,0ffh      ; funzione informazioni sull'autore
                ldi     drw,msg41.w
                ldi     x,msg43.d
                call    sopra
                ldi     x,msg44.d
                call    sotto
                ldi     ior,10h
```

```
                clr     puls
exi2            ldi     wdog,0ffh
                ld      a,puls
                cpi     a,234
                jrnz    au1
                jp      funz1
au1             cpi     a,235
                jrnz    au2
                jp      funz1
au2             jp      exi2


;=========================================================================
;=   DEFINIZIONI DI TABELLE IN PROGRAM SPACE ==============================
;=========================================================================


; Esempi per inserire uno o più byte:
;        .ascii  " VOLT "
;        .byte   01111110b
;        .byte   32,32,32
;        .byte   01111111b


.ifc  ne ($%64)
        .block  64-$%64
.endc

pulsanti ; Tabella per la conversione in codifica ASCII
         ; del numero associato ad ogni pulsante


;        CODIFICA ASCII            PULSANTE PREMUTO

         .ascii "1234"             ; 1  2  3  4
         .byte  224                ; 5
         .ascii "5678"             ; 6  7  8  9
         .byte  225                ; 10
         .ascii "B090"             ; 11 12 13 14
         .byte  226                ; 15
         .ascii "9"                ; 16
         .byte  226                ; 17
         .ascii "AB"               ; 18 19
         .byte  227                ; 20
         .ascii "D"                ; 21
         .byte  228                ; 22
         .ascii "CD"               ; 23 24
         .byte  228                ; 25
         .ascii "EFGH"             ; 26 27 28 29
         .byte  229                ; 30
         .ascii "ILMN"             ; 31 32 33 34
         .byte  230                ; 35
         .ascii "OPQR"             ; 36 37 38 39
         .byte  231                ; 40
         .ascii "STUV"             ; 41 42 43 44
         .byte  232                ; 45
         .ascii "Z"                ; 46
         .byte  234,235,236,233    ; 47 48 49 50



; ---- TAB n° 1
```

```
.ifc  ne ($%64)
        .block  64-$%64                 ; per inserire una tabella in ROM
.endc

msg01    .ascii  "CRONOS V.0.1    "
msg02    .ascii  "IPSIA MASSA M.MA"
msg03    .ascii  "CRONOMETRO      "
msg04    .ascii  "AVANZATO E      "


; ---- TAB n° 2

.ifc  ne ($%64)
        .block  64-$%64
.endc

msg05    .ascii  "AUTOMATICO      "
msg06    .ascii  "PER USO SPORTIVO"
msg07    .ascii  "                "
msg08    .ascii  "                "


; ---- TAB n° 3

.ifc  ne ($%64)
        .block  64-$%64
.endc

msg09    .ascii  "ELENCO FUNZIONI:"
msg10    .ascii  "  (USA FRECCE)  "
msg11    .ascii  "1-CRONOMETRO    "
msg12    .ascii  "2-ELENCO NUMERIC"


; ---- TAB n° 4

.ifc  ne ($%64)
        .block  64-$%64
.endc

msg13    .ascii  "3-RICERCA ID    "
msg14    .ascii  "4-TEST FOTOCELL."
msg15    .ascii  "5-TEST ALTOPARL."
msg16    .ascii  "6-SEL START MAN."


; ---- TAB n° 5

.ifc  ne ($%64)
        .block  64-$%64
.endc

msg17    .ascii  "7-SEL STOP  MAN."
msg18    .ascii  "8-SEL FOTOCELL 1"
msg19    .ascii  "9-SEL FOTOCELL 2"
msg20    .ascii  "0-INFO AUTORE   "


; ---- TAB n° 6

.ifc  ne ($%64)
        .block  64-$%64
.endc
```

```
msg21    .ascii  "START MANUALE O "
msg22    .ascii  "CON FOTOCELLULA?"
msg23    .ascii  "TEST FOTOC. 1..."
msg24    .ascii  "TEST FOTOC. 2..."


; ---- TAB n° 7

.ifc  ne ($%64)
        .block  64-$%64
.endc

msg25    .ascii  "FOTOC. RILEVATA!"
msg26    .ascii  "FOTOC. SPENTA?! "
msg27    .ascii  "START MANUALE   "
msg28    .ascii  "STOP MANUALE    "


; ---- TAB n° 8

.ifc  ne ($%64)
        .block  64-$%64
.endc

msg29    .ascii  "START CON FOTOC1"
msg30    .ascii  "STOP  CON FOTOC2"
msg31    .ascii  "DISTANZA IN MT.:"
msg32    .ascii  "OK PER INIZIARE "


; ---- TAB n° 9

.ifc  ne ($%64)
        .block  64-$%64
.endc

msg33    .ascii  "  cronometro:   "
msg34    .ascii  "  hh:mm:ss      "
msg35    .ascii  "time   :  :  :  "
msg36    .ascii  "d:         metri"


; ---- TAB n° 10

.ifc  ne ($%64)
        .block  64-$%64
.endc

msg37    .ascii  "v:       m/s    "
msg38    .ascii  "MEMORIZZO TEMPO?"
msg39    .ascii  "Inserisci ID:xxx"
msg40    .ascii  "Sovrascrivo xxx?"


; ---- TAB n° 11

.ifc  ne ($%64)
        .block  64-$%64
.endc

msg41    .ascii  "Inserisci nome: "
msg42    .ascii  "xxx: __________"
msg43    .ascii  "FrancescoGalgani"
msg44    .ascii  " www.galgani.it "
```

```
; ---- TAB n° 12

.ifc  ne ($%64)
        .block  64-$%64
.endc

msg45    .ascii  "START CON FOTOC2"
msg46    .ascii  "STOP  CON FOTOC1"
msg47    .ascii  "STOP MANUALE O  "
msg48    .ascii  "CON FOTOCELLULA?"


; ---- TAB n° 13

.ifc  ne ($%64)
        .block  64-$%64
.endc

msg49    .ascii  "   ANNULLA...   "
msg50    .ascii  "ANNUL PER USCIRE"
msg51    .ascii  "PREMI START...  "
msg52    .ascii  "ATTENDO FOTOCELL"


; ---- TAB n° 14

.ifc  ne ($%64)
        .block  64-$%64
.endc

msg53    .ascii  "                "
msg54    .ascii  "                "
msg55    .ascii  "                "
msg56    .ascii  "                "




;*************************************************************************
;*                     VETTORI DI INTERRUPTS                             *
;*************************************************************************

; Questa parte deve essere riportata alla fine del MAIN e prima della
; istruzione finale .END .
; Queste istruzioni non dovranno essere modificate.

      .org    0ff0h
      jp      ad_int  ;INTERRUPT DEL CONV. A/D     vector #4
      jp      tim_int ;INTERRUPT DEL TIMER         vector #3
      jp      BC_int  ;INTERRUPT PORTE A e B       vector #2
      jp      A_int   ;INTERRUPT PORTA A           vector #1
      .org    0ffch
      jp      nmi_int ;INTERRUPT NON MASCHERABILE  vector #0
      jp      inizio  ;INTERRUZIONE PER IL SETTAGGIO INIZIALE

;*************************************************************************

      .end                    ; Termine del programma
```

```
;*************************************************************************
;*                                                                       *
;*  FILE  SORGENTE  "TASTIERINO"                                         *
;*                                                                       *
;*************************************************************************

;*  INDIVIDUA IL PULSANTE PREMUTO SU UN TASTIERINO (50 PULSANTI)         *
;*  E LO TRASMETTE IN FORMA SERIALE                                      *
;*                                                                       *
;*************************************************************************

; Ricordate sempre che il cuore del programma da realizzare e' il cosiddetto
; PROGRAMMA PRINCIPALE ( MAIN ), che in genere, ma non obbligatoriamente, si
; inserisce alla fine del file. Le SUBROUTINE possono essere inserite
; subito prima del MAIN, anche se e' possibile posizionarle in QUALUNQUE
; punto del programma.
; Fate comunque attenzione che, sia il MAIN, sia le SUBROUTINE devono essere
; compresi fra i due comandi ".ORG" (inizio programma) e ".END" (fine del
; programma).
; In ogni parte del programma e' possibile inserire commenti di ogni tipo,
; purche' preceduti dal punto e virgola, come nel testo che state leggendo.
;

;*************************************************************************
;*                    NOTE SUI CAMPI (FIELD) PER I COMANDI               *
;*************************************************************************

; 1¦ CAMPO
;      |___
;          | ETICHETTA; massimo 8 caratteri. Deve iniziare sempre con una
;          | lettera e deve partire dalla PRIMA COLONNA. Per passare
;          | al campo successivo e' preferibile usare il tasto TABULAZIONE.

;           2¦ CAMPO
;                |______
;                       | ISTRUZIONI; sono le istruzioni da far eseguire al
;                 | micro.
;                       | Per passare al campo successivo usare il
;                       | tasto TABULATORE.

;                     3¦ CAMPO
;                          |____
;                               | OPERANDI; sono gli operandi delle
;                               | istruzioni.

;                                  4¦ CAMPO
;                                       |____
;                                            | COMMENTI; come gia' evidenziato
;                                            | non e' obbligatorio posizionarlo
;                                            | proprio in questa posizione, ma
;                                            | in tal modo consente di spiegare
;                                            | il significato delle varie
;                                            | operazioni.


;                              INIZIALIZZAZIONE
;                              ----------------

        .title  "TASTIERINO"

        .vers   "ST62E25C"     ; INDICA IL TIPO DI DISPOSITIVO DA PROGRAMMARE


;*************************************************************************
;*                       VARIABILI DEL MICRO                             *
```

```
;***************************************************************************

; Questo elenco di variabili e' STANDARD, cioe' non dovrete mai modificarlo.

;      DEFINIZIONE DEGLI INDIRIZZI DI PARTENZA DELLE VARIE AREE DI MEMORIA
;      ------------------------------------------------------------------



;            DEFINIZIONE DELLE LOCAZIONI DEI 5 REGISTRI DEL MICRO
;            ---------------------------------------------------

a       .def    0ffh    ;ACCUMULATOR (REGISTRO ACCUMULATORE)
x       .def    080h    ;X REGISTER  (REGISTRO X)
y       .def    081h    ;Y REGISTER  (REGISTRO Y)
v       .def    082h    ;V REGISTER  (REGISTRO V)
w       .def    083h    ;W REGISTER  (REGISTRO W)


;              DEFINIZIONE DEI REGISTRI DELLE PORTE
;              ------------------------------------

port_a  .def    0c0h    ;PORT A DATA REGISTER
port_b  .def    0c1h    ;PORT B DATA REGISTER
port_c  .def    0c2h    ;PORT C DATA REGISTER
pdir_a  .def    0c4h    ;PORT A DIRECTION REGISTER  (0=Input   1=Output)
pdir_b  .def    0c5h    ;PORT B DIRECTION REGISTER
pdir_c  .def    0c6h    ;PORT C DIRECTION REGISTER
popt_a  .def    0cch    ;PORT A OPTION REGISTER
popt_b  .def    0cdh    ;PORT B OPTION REGISTER
popt_c  .def    0ceh    ;PORT C OPTION REGISTER


;              DEFINIZIONE DEL REGISTRO DEGLI INTERRUPT
;              ----------------------------------------

ior     .def    0c8h    ;INTERRUPT OPTION REGISTER.


;              DEFINIZIONE DEL REGISTRO PER IL CONVERTITORE A/D
;              -----------------------------------------------

addr    .def    0d0h    ;A/D DATA REGISTER (REGISTRO DATI DEL CONVERTITORE)
adcr    .def    0d1h    ;A/D CONTROL REGISTER (REGISTRO DI CONTROLLO)


;              DEFINIZIONE DEI REGISTRI DEL TIMER
;              ----------------------------------

psc     .def    0d2h    ;TIMER PRESCALER REGISTER (REGISTRO DEL PRESCALER)
tcr     .def    0d3h    ;TIMER DATA REGISTER (REGISTRO DATI)
tscr    .def    0d4h    ;TIMER TSCR REGISTER ( REGISTRO PER IL CONTROLLO
                  ;DELLO STATO DEL TIMER)


;              DEFINIZIONE DEL REGISTRO PER IL WATCH-DOG
;              -----------------------------------------

wdog    .def    0d8h    ;WATCHDOG REGISTER


;              DEFINIZIONE DEL REGISTRO PER LO SPAZIO DATI ROM
;              -----------------------------------------------
```

```
drw     .def    0c9h    ;DATA ROM WINDOW REGISTER

;*************************************************************************
;*                          VARIABILI DEL PROGRAMMA                      *
;*************************************************************************

; In questo punto del programma potete definire le variabili che intendete
; utilizzare.
; L'etichetta che le contraddistingue
; puo' essere qualunque, ma non ci possono essere due variabili con lo
; stesso nome. Notate che gli indirizzi sono consecutivi ed iniziano
; dall'indirizzo 084h, il primo disponibile della RAM.
; Si possono usare un massimo di 60 variabili.

out     .def    084h
puls    .def    085h
n       .def    086h
save_a  .def    087h

;*************************************************************************
;*                          SETTAGGIO INIZIALE                           *
;*************************************************************************

; Le porte parallele, il timer, il convertitore A/D ed il watch dog debbono
; essere INIZIALIZZATI all'inizio del programma.
; Abbiamo indicato l'operazione di settaggio iniziale con INIZIO.


;             INIZIO DEL PROGRAMMA
;             --------------------

      .org    080h      ; E' DA UTILIZZARE NEI PROGRAMMI DI 4 Kb PER
                        ;    I MICRO ST62E20,ST62T20,ST62E25,ST62T25.

inizio                  ;  INIZIALIZZAZIONE DELLE PERIFERICHE


      ldi     wdog,0ffh       ; RICARICA IL WATCH DOG
; Questa operazione va ripetuta circa ogni 30 istruzioni per evitare che
; il microprocessore si resetti da solo.


        ldi     pdir_a,00000000b   ;
        ldi     popt_a,00000000b   ;
        ldi     port_a,11111111b   ;

        ldi     pdir_b,11111111b   ;
        ldi     popt_b,11111111b   ;
        ldi     port_b,00000000b   ;

        ldi     pdir_c,11110000b   ;
        ldi     popt_c,11110000b   ;
        ldi     port_c,00010000b   ;



      ldi     adcr,0          ;DISABLE A/D INTERRUPT
      ldi     tscr,0          ;DISABLE TIMER INTERRUPT
      ldi     ior,0           ;DISABLE ALL INTERRUPT

      reti              ;RIPRISTINA I FLAG PRINCIPALI

        .w_on                   ; abilita la Data Rom Window
```

```
      jp      main            ;SALTA AL PROGRAMMA PRINCIPALE


;*****************************************************************************
;*                         SUBROUTINE DI INTERRUPT                           *
;*****************************************************************************

; In questo punto potremmo scrivere le subroutine di interrupt.
; Queste subroutine devono terminare con l'istruzione   reti.
; La subroutine deve percio' essere scritta tra l'etichetta che la
; identifica e l'istruzione reti.

ad_int                  ;INTERRUPT DEL CONVERTITORE A/D
      reti

tim_int                 ;INTERRUPT DEL TIMER
      reti

BC_int                  ;INTERRUPT DELLE PORTE A e B
      reti

A_int                   ;INTERRUPT DELLA PORTA A
      reti

nmi_int                 ;INTERRUPT NON MASCHERABILE
      reti


;*****************************************************************************
;*                            SUBROUTINE                                     *
;*****************************************************************************

; Le SUBROUTINE sono dei pezzi di programmi che verranno utilizzate
; dal programma principale (MAIN) per svolgere determinate operazioni.
; Questi "sottoprogrammi" DEBBONO sempre iniziare con un'ETICHETTA e
; terminare con l'istruzione RET.



; --------------------------- SUBROUTINE N¦ 1 ---------------------------

; Pulisce tutta la RAM a partire da 080h fino a 0BFh

pulisci
      clr   a
      ldi   y,60  ; 60 = byte di RAM disponibili, pi¨ quattro registri
      ldi   x,084h  ; 084h = primo byte di RAM
clean
      ldi   wdog,255
      ld    (x),a
      inc   x
      dec   y
      jrnz  clean

      clr   x     ; cancellazione dei registri da 080h a 083h
      clr   y
      clr   v
      clr   w
      ret



; --------------------------- SUBROUTINE N¦ 2 ---------------------------
```

```
ritardo        ;NOME DELLA SUBROUTINE (QUANDO VORREMO RICHIAMARLA
              ;NEL PROGRAMMA PRINCIPALE SCRIVEREMO "call ritardo")

        ldi     y,128

rit2    ldi     x,255
rit1    ldi     wdog,0ffH       ; ricarica il watch dog
      dec     x               ; decrementa x
      jrnz    rit1            ; se x non e' zero salta a rit1

      dec     y         ; decrementa y
      jrnz    rit2            ; se y non e' zero salta a rit2

      ret               ; ritorna al programma principale.



; --------------------------- SUBROUTINE N¦ 3 ----------------------------

sendpuls   ; invia il contenuto del reg. "puls"
           ; C7, C6, C5 --> bit da trasmettere (da b5 a b0)
           ; C4 --> impulso di interrupt

           ; Questa Þ la subroutine del TRASMETTITORE

start   ldi     wdog,0ffh

        ld      a,puls       ; carico "b5, b4, b3" su "C7, C6, C5"
        rlc     a
        rlc     a
        andi    a,11100000b
        set     4,a
        ld      port_c,a

        res     4,port_c     ; invio impulso di interrupt  | da invio int
                             ;                             | a invio ultimi
        ld      a,puls       ; 4 cicli                     | tre bit
        rlc     a            ; 4 cicli                     | trascorrono
        rlc     a            ; 4 cicli                     | almeno
        rlc     a            ; 4 cicli                     | 9x4=36 cicli
        rlc     a            ; 4 cicli                     |
        rlc     a            ; 4 cicli                     | (che equivalgono
        andi    a,11100000b  ; 4 cicli                     |  a 18 NOP)
        set     4,a          ; delay di 4x7=28 cicli       |
                             ;                             |
        ld      port_c,a     ; fine impulso di interrupt   |
                             ; e invio degli altri tre bit |
                             ; b2, b1, b0
        ret


;*************************************************************************
;*                          PROGRAMMA PRINCIPALE                         *
;*************************************************************************

; Il programma principale iniziera' con l'etichetta main
; Per farvi un esempio abbiamo richiamato nel programma principale
; la subroutine ritardo.

main                    ;IDENTIFICA L'INIZIO DEL PROGRAMMA PRINCIPALE

        ldi     wdog,0ffh       ;RICARICA IL WATCH DOG
```

```
startsc ; inizia la scansione

        call    pulisci         ;Pulisce tutta la RAM
                                ;(poichÚ contiene informazioni casuali)

        ldi     port_b,00000001B ; attiva la prima uscita
increm  ldi     wdog,0ffh
        inc     out
        ld      a,port_a
        cpi     a,0
        jrz     nextout          ; se port_a=0 SI, salta all'etichetta
        jp      decod


nextout ; prova l'uscita successiva (port_a=0 SI)

        ld      a,out
        cpi     a,8
        jrnz    fg              ; se out=8, inizia nuovamente la scansione
        jp      startsc
fg      ld      a,port_b
        sla     a
        ld      port_b,a        ; abilita l'uscita successiva
        jp      increm


reset   ; inizia nuovamente il programma
        ldi     wdog,0ffh
        res     5,port_c  ; resetta le linee di trasmissione
        res     6,port_c
        res     7,port_c
        jp      main


decod   ; port_a=0 NO

        ldi     wdog,0ffh
        ld      a,port_b

out1    cpi     a,00000001B ; questa sequenza stabilisce quale decodif. usare
        jrnz    out2
        jp      decod1

out2    ldi     wdog,0ffh
        cpi     a,00000010B
        jrnz    out3
        jp      decod2

out3    ldi     wdog,0ffh
        cpi     a,00000100B
        jrnz    out4
        jp      decod3

out4    ldi     wdog,0ffh
        cpi     a,00001000B
        jrnz    out5
        jp      decod4

out5    ldi     wdog,0ffh
        cpi     a,00010000B
        jrnz    out6
        jp      decod5
```

```
out6    ldi     wdog,0ffh
        cpi     a,00100000B
        jrnz    out7
        jp      decod6

out7    ldi     wdog,0ffh
        cpi     a,01000000B
        jrnz    out8
        jp      decod7

out8    ldi     wdog,0ffh
        cpi     a,10000000B
        jrz     ok
ok      jp      decod8
        jp      reset


; ---- Combinazione n¦ 1

; esempio: 1+6=50
; significa: se mettendo a stato alto l'ingresso 1
;            l'uscita 6 si porta anch'essa a stato alto,
;            allora ho premuto il tasto n° 50.

decod1  ld      a,port_a      ; 1+6=50
        cpi     a,10000000B
        jrnz    dec1a6
        ldi     puls,50
        jp exit

dec1a6  cpi     a,01000000B   ; 1+8=49
        jrnz    dec1a5
        ldi     puls,49
        jp      exit

dec1a5  cpi     a,00100000B   ; 1+10=28
        jrnz    dec1a4
        ldi     puls,28
        jp      exit


dec1a4  cpi     a,00010000B   ; 1+11=2
        jrnz    dec1a3
        ldi     puls,2
        jp      exit


dec1a3  cpi     a,00001000B   ; 1+12=10
        jrnz    dec1a2
        ldi     puls,10
        jp      exit

dec1a2  cpi     a,00000100B   ; 1+14=27
        jrnz    dec1a0
        ldi     puls,27
        jp      exit

dec1a0  cpi     a,00000001B   ; 1+16=39
        jrnz    res1
        ldi     puls,39
        jp      exit
res1    jp      reset
```

```
; ---- Combinazione n¦ 2

decod2  ld      a,port_a      ; 2+6=29
        cpi     a,10000000B
        jrnz    dec2a6
        ldi     puls,29
        jp exit

dec2a6  cpi     a,01000000B   ; 2+8=12
        jrnz    dec2a5
        ldi     puls,12
        jp      exit

dec2a5  cpi     a,00100000B   ; 2+10=23
        jrnz    dec2a4
        ldi     puls,23
        jp      exit

dec2a4  cpi     a,00010000B   ; 2+11=4
        jrnz    dec2a3
        ldi     puls,4
        jp      exit

dec2a3  cpi     a,00001000B   ; 2+12=22
        jrnz    dec2a2
        ldi     puls,22
        jp      exit


dec2a2  cpi     a,00000100B   ; 2+14=37
        jrnz    res2
        ldi     puls,37
        jp      exit
res2    jp      reset


; ---- Combinazione n¦ 3

decod3  ld      a,port_a      ; 3+8=47
        cpi     a,01000000B
        jrnz    dec3a4
        ldi     puls,47
        jp exit

dec3a4  cpi     a,00010000B   ; 3+11=1
        jrnz    dec3a3
        ldi     puls,1
        jp      exit

dec3a3  cpi     a,00001000B   ; 3+12=9
        jrnz    dec3a2
        ldi     puls,9
        jp      exit

dec3a2  cpi     a,00010000B   ; 3+14=26
        jrnz    dec3a1
        ldi     puls,26
        jp      exit

dec3a1  cpi     a,00000010B   ; 3+15=46
        jrnz    dec3a0
        ldi     puls,46
        jp      exit
```

```
dec3a0  cpi     a,00000001B   ; 3+16=34
        jrnz    res3
        ldi     puls,34
        jp      exit
res3    jp      reset

; ---- Combinazione n¦ 4

decod4  ld      a,port_a      ; 4+8=11
        cpi     a,01000000B
        jrnz    dec4a5
        ldi     puls,11
        jp exit

dec4a5  cpi     a,00100000B   ; 4+10=18
        jrnz    dec4a4
        ldi     puls,18
        jp      exit


dec4a4  cpi     a,00010000B   ; 4+11=3
        jrnz    dec4a3
        ldi     puls,3
        jp      exit

dec4a3  cpi     a,00001000B   ; 4+12=21
        jrnz    dec4a2
        ldi     puls,21
        jp      exit

dec4a2  cpi     a,00000100B   ; 4+14=32
        jrnz    res4
        ldi     puls,32
        jp      exit
res4    jp      reset

; ---- Combinazione n¦ 5

decod5  ld      a,port_a      ; 5+6=30
        cpi     a,10000000B
        jrnz    dec5a6
        ldi     puls,30
        jp exit

dec5a6  cpi     a,01000000B   ; 5+8=41
        jrnz    dec5a4
        ldi     puls,41
        jp      exit


dec5a4  cpi     a,00010000B   ; 5+11=7
        jrnz    dec5a2
        ldi     puls,7
        jp      exit

dec5a2  cpi     a,00000100B   ; 5+14=31
        jrnz    res5
        ldi     puls,31
        jp      exit
res5    jp      reset

; ---- Combinazione n¦ 6
```

```
decod6  ld      a,port_a      ; 7+6=44
        cpi     a,10000000B
        jrnz    dec6a6
        ldi     puls,44
        jp exit

dec6a6  cpi     a,01000000B   ; 7+8=40
        jrnz    dec6a5
        ldi     puls,40
        jp      exit

dec6a5  cpi     a,00100000B   ; 7+10=43
        jrnz    dec6a4
        ldi     puls,43
        jp      exit

dec6a4  cpi     a,00010000B   ; 7+11=6
        jrnz    dec6a2
        ldi     puls,6
        jp      exit

dec6a2  cpi     a,00000100B   ; 7+14=38
        jrnz    res6
        ldi     puls,38
        jp      exit
res6    jp      reset

; ---- Combinazione n¦ 7

decod7  ld      a,port_a      ; 9+6=45
        cpi     a,10000000B
        jrnz    dec7a6
        ldi     puls,45
        jp exit

dec7a6  cpi     a,01000000B   ; 9+8=35
        jrnz    dec7a5
        ldi     puls,35
        jp      exit

dec7a5  cpi     a,00100000B   ; 9+10=13
        jrnz    dec7a4
        ldi     puls,13
        jp      exit

dec7a4  cpi     a,00010000B   ; 9+11=5
        jrnz    dec7a3
        ldi     puls,5
        jp      exit

dec7a3  cpi     a,00001000B   ; 9+12=15
        jrnz    dec7a2
        ldi     puls,15
        jp      exit


dec7a2  cpi     a,00000100B   ; 9+14=33
        jrnz    res7
        ldi     puls,33
        jp      exit
res7    jp      reset

; ---- Combinazione n¦ 8
```

```
decod8  ld      a,port_a      ; 13+6=42
        cpi     a,10000000B
        jrnz    dec8a4
        ldi     puls,42
        jp exit

dec8a4  cpi     a,00010000B   ; 13+11=8
        jrnz    dec8a3
        ldi     puls,8
        jp      exit

dec8a3  cpi     a,00001000B   ; 13+12=20
        jrnz    dec8a2
        ldi     puls,20
        jp      exit

dec8a2  cpi     a,00000100B   ; 13+14=36
        jrnz    dec8a1
        ldi     puls,36
        jp      exit

dec8a1  cpi     a,00000010B   ; 13+15=48
        jrnz    res8
        ldi     puls,48
        jp      exit
res8    jp      reset

exit    ; il pulsante identificato viene ora trasmetto con C7, C6, C5, C4
        ; C7, C6, C5 = tre bit
        ; C4 = interrupt

        call    sendpuls ; invia "puls"
        call    ritardo  ; fa una breve pausa (per evitare di spedire
                         ; in continuazione lo stesso pulsante)

        jp      reset    ; resetta il micro



;*************************************************************************
;*                     VETTORI DI INTERRUPTS                             *
;*************************************************************************

; Questa parte deve essere riportata alla fine del MAIN e prima della
; istruzione finale .END .
; Queste istruzioni non dovranno essere modificate.

      .org    0ff0h
      jp      ad_int  ;INTERRUPT DEL CONV. A/D     vector #4
      jp      tim_int ;INTERRUPT DEL TIMER         vector #3
      jp      BC_int  ;INTERRUPT PORTE A e B       vector #2
      jp      A_int   ;INTERRUPT PORTA A           vector #1
      .org    0ffch
      jp      nmi_int ;INTERRUPT NON MASCHERABILE  vector #0
      jp      inizio  ;INTERRUZIONE PER IL SETTAGGIO INIZIALE

;*************************************************************************

      .end                    ; Termine del programma
```