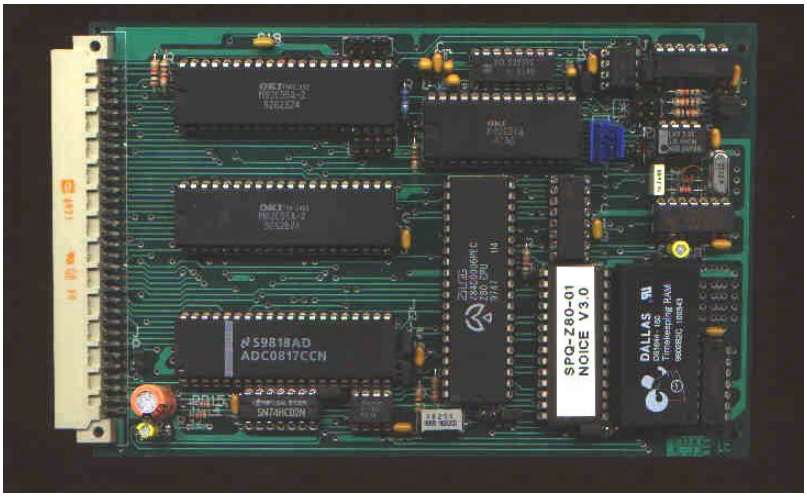
MICROELABORATORE SPQ-Z80



1 - DESCRIZIONE GENERALE
2 - CARATTERISTICHE TECNICHE
3 - CONFIGURAZIONE JUMPERS
3.1 Selezione Baud rate
3.2 Configurazione EPROM
3.3 Configurazione generale
4 - MOS (Micro Operating System) V.3_1X
4.1 Descrizione comandi
4.2 Display LCD
4.3 Tastiera esadecimale
4.4 Autostart
4.5 Bootstrap
4.6 Esempio di caricamento su EPROM di un programma
4.7 Routine di servizio MOS
4.7.1 Trasmissione dati a RS232C
4.7.2 Ricezione dati da RS232C
4.7.3 Visualizzazione caratteri su LCD LM016L
4.7.4 Reset LCD
4.7.5 Visualizzazione data ed ora su LCD
4.7.6 Interrupt
4.7.7 Abilitazione interrupt
4.7.8 Disabilitazione interrupt
4.7.9 Posizionamento cursore terminale
4.7.10 Cancellazione fino a fine pagina
4.7.11 Acquisizione 16 ingressi ADC
4.7.12 Selezione attributo carattere
4.7.13 Composizione di rettangoli

4.7.14 Generazione treno onde quadre
4.7.15 Rientro al Monitor
5 - GESTIONE DEI PERIFERICI
5.1 Convertitore analogico-digitale.
5.2 Interfacce parallele programmabili.
6 - RISORSE DI MEMORIA E I/O
6.1 Indirizzi di I/O
6.2 Indirizzi di memoria
6.3 Mappa area NVRAM riservata a variabili e costanti MOS

1 - DESCRIZIONE GENERALE

L'SPQ-Z80 è un microcalcolatore su singola scheda formato Eurocard, con l'opzione di un interprete BASIC su EPROM (EEPROM), adatto alla realizzazione di sistemi di acquisizione dati e in generale di automazione. Utilizza il microprocessore Zilog Z84C000, molto noto, documentato e compatibile a livello codice oggetto con i microprocessori della famiglia Z80 e Z80180 come HD64180 Hitaci e NSC800 National.

La caratteristica principale dell'SPQ-Z80 è quella di poter essere programmato in BASIC con il solo ausilio di un PC o di un videoterminale. Il BASIC residente permette la gestione di tutte le risorse hardware della scheda e può richiamare routine in assembler mediante l'istruzione CALL; Il mantenimento dei dati è assicurato da una NVRAM da 32k x 8 Dallas DS1644 con real time clock. In alternativa, e` disponibile un potente Monitor/debugger remoto per PC NoICE per lo sviluppo ed il debug di codice in assembler.

L'architettura hardware della scheda è costituita da : MPU Z84C000 Zilog con clock a 6 MHz, 32K di NVRAM / EPROM (EEPROM), un convertitore analogico/digitale a 8 bit con 16 canali di ingresso e generatore di riferimento di precisione, una porta seriale RS-232C, una porta RS-485/422, una porta parallela Centronics, 6 porte I/O a 8 bit programmabili; la scheda ha un connettore DIN a 64 poli che realizza un bus di I/O, utile per espansioni.

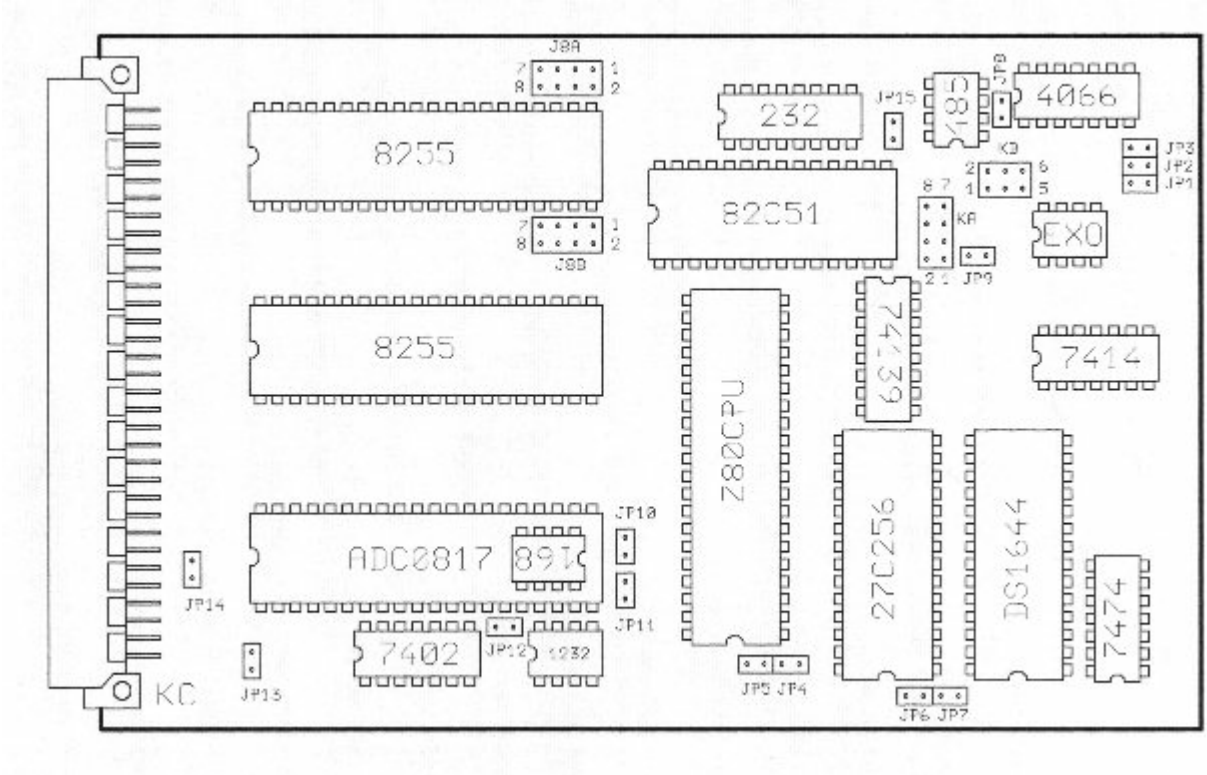
Queste caratteristiche hardware, unite alla flessibilità del software che rende disponibile il programma sorgente in NVRAM per modifiche anche sul campo, rendono l'SPQ-Z80 molto versatile e adatto allo sviluppo di sistemi di automazione e supervisione, anche soggetti a frequenti espansioni o modifiche.

Il software disponibile di supporto alla scheda, comprende : l'interprete BASIC ed un micro sistema operativo (MOS) che permette l'editing a livello codice oggetto, una gestione interrupt vettorizzato, un real time clock, un driver per LCD e tastiera ASCII, un sistema di autoboot e autostart e varie utility in assembler. E`disponibile anche un Monitor remoto NoICE per lo sviluppo in Assembler.

Da un punto di vista operativo l'utilizzazione dell'SPQ-Z80 è descritta dalle fasi seguenti:

- collegamento dell'SPQ-Z80 ad un terminale o PC tramite interfaccia seriale RS232C; in alternativa è possibile utilizzare una tastiera parallela ASCII con un display LCD connessi alle porte I / O come da schemi allegati.
- sviluppo del software col supporto dell'interprete BASIC residente sulla scheda, con possibilità di scrivere routine in codice oggetto Z80 mediante l'utility del MOS
- test del software interattivo con eventuale hardware esterno richiesto dall'applicazione.

2 - CARATTERISTICHE TECNICHE



- CPU Z84C0006 CMOS con clock a 2, 4, o 6 MHz
- RAM NVRAM 32k x 8 Dallas DS1644 con real time clock e batteria Litio >= 10 anni
- EPROM 32k x 8 EPROM / EEPROM
- CONVERTITORE analogico/digitale ADC0817 a 16 canali 0-5V con multiplexer incorporato, 8 bit di risoluzione, tempo di conversione 100 microsecondi. Linee di alimentazione e massa analogica separate.
- GENERATORE LM168 tensione di riferimento per ADC 5.0V - 5ppm / ^C 0, 0005% / V
- MICROPROCESSOR MONITOR DS1232 monitor tensione di alimentazione(5V+/ -5%) e watchdog timer con periodo 1,2 s.
- 2 INTERFACCE PARALLELE I/O industry standard 82C55 programmabili, per un totale di 6 porte TTL da 8 bit.

- 1 PORTA RS-232C (1200-38600 bps) per comunicazione con PC
- 1 PORTA RS-485/422 per reti su doppino twistato (fino a 32 schede SPQ -Z80)
- CONNETTORE DIN 41612 a 64 vie per connessioni I/O ed espansioni su SPQ -BUS
- DIMENSIONI 100 x 160 mm (standard Eurocard).
- ALIMENTAZIONE 5 V (+/- 5%) 50 mA, + opz. 12V 1 mA (riferimento A/D converter)

3 - CONFIGURAZIONE JUMPERS

3.1 Selezione Baud rate

La velocita` di trasferimento dati della interfaccia seriale RS232 e` selezionabile mediante i 3 jumpers posti a fianco della batteria. Le 6 velocità ottenibili sono date dalla tabella :

JP1	JP2	JP3	BPS
NO	NO	NO	1200
SI	NO	NO	2400
NO	SI	NO	4800
SI	SI	NO	9600 (default)
NO	NO	SI	19200
SI	NO	SI	38600

3.2 Configurazione EPROM

E` possibile utilizzare sia EPROM che EEPROM da 32k x 8 con il setup :

JP4	JP5	JP6	JP7	
SI	NO	SI	NO	EPROM
NO	SI	NO	SI	EEPROM

3.3 Configurazione generale

JP8	RE-DE MAX485	SI DEFAULT
JP9	RXRD UART A INT Z80	NO DEFAULT
JP10	VREF(-) ADC0817 A GND	SI DEFAULT
JP11	MUX OUT A COMP IN ADC0817	SI DEFAULT
JP12	VREF(+) ADC0817 A LM168-2	NO CON LM168
JP13	BY-PASS LM168 A +5V	NO CON LM168
JP14	EOC ADC0817 A INT Z80	NO DEFAULT
JP15	CTS UART A +5V	SI DEFAULT

4 - MOS (Micro Operating System) V.3_1X

4.1 Descrizione comandi

L'interprete BASIC residente su EPROM (EEPROM) è integrato da un MOS (micro operating system) che, oltre a gestire le interfacce RS-232C, RS-422 e Centronics, una tastiera ASCII e un display LCD, dispone dei seguenti comandi :

- M : manda in esecuzione un programma di monitor che consente un accesso diretto alle locazioni di memoria; digitando l'indirizzo (HEX), ne viene visualizzato il contenuto. Nel modo monitor sono disponibili i seguenti 3 sotto comandi :

E : consente di modificare il contenuto della locazione aperta

J : determina l'esecuzione del programma all'indirizzo visualizzato

SPAZIO : incrementa l'indirizzo della locazione

- R : determina un restart dell'interprete BASIC, con cancellazione della NVRAM, richiede il limite superiore di NVRAM visualizzando il messaggio "UP RAM limit ?" max 65536 (49152), per dimensionare lo spazio di memoria da rendere disponibile all'interprete BASIC.
- B : manda in esecuzione il BASIC senza cancellazione della RAM è utilizzabile solo dopo aver inizializzato il limite della NVRAM visto sopra.

4.2 Display LCD

Un Display LCD può' essere controllato sia in BASIC che in linguaggio macchina (ASSEMBLER) mediante le routine descritte al paragrafo 4.7.3. Per utilizzare l'LCD mediante l'istruzione PRINT dal BASIC, è sufficiente abilitare un flag, scrivendo il dato 75 (cod.ASCII di "K") nella locazione 65030, questo flag abilita contemporaneamente anche la porta parallela Centronics. Si può' posizionare il cursore scrivendone la posizione nella locazione 65034, prima dell'istruzione PRINT.Es.:

```
10 CALL 880 : REM CANCELLA LCD
20 POKE 65030,75 : REM ABILITA LCD E CENTRONICS IN BASIC 30 POKE
65034,64 : REM CURSORE ALL'INIZIO DELLA SECONDA RIGA 40 PRINT
"SPQ-Z80 BAS 3.7"
50 POKE 65030,0 : REM DISABILITA LCD E CENTRONICS IN BASIC 60 CALL
1168 : REM VISUALIZZA DATA E ORA CON ROUTINE IN ASSEMBLER
```

4.3 Tastiera esadecimale

Seguendo lo schema elettrico allegato, è possibile utilizzare una tastiera esadecimale a matrice di interruttori, codificata tramite un integrato dedicato, per immettere dati o comandi sia in BASIC che nel modo editing del monitor. La tastiera è connessa ad una porta parallela della PPI1, è normalmente abilitata ed è compatibile ad un contemporaneo uso di un terminale collegato alla porta seriale RS232C. Con una diversa configurazione circuitale è possibile utilizzare la stessa tastiera per gestire un interrupt vettorizzato, come descritto in dettaglio al paragrafo 4.7.6.

4.4 Autostart

Le versioni 2.24 e successive del MOS dispongono della funzione di autostart, molto utile per i sistemi di controllo cosiddetti "embedded" dove non sono cioè' necessariamente presenti dei terminali per l'avvio ed il controllo dell'elaborazione. Il programma caricato in RAM viene eseguito dopo un reset, manuale o generato dal watchdog nel caso di anomalie software, senza dover digitare "BRUN", con totale autonomia di funzionamento dell'SPQ-Z80. Per uscire dal modo autostart, normalmente abilitato, occorre operare come segue :

1. digitare ctrl-S e, mantenendo i 2 tasti premuti, dare un reset (è necessaria una sequenza in auto-repeat > 20 caratteri per evitare interruzioni accidentali).
2. dopo il prompt del MOS, rilasciare i tasti ctrl-S
3. digitare ctrl-Q per abilitare la comunicazione con il terminale, precedentemente interrotta. Con un reset si riattiva la funzione di autostart.

NB : i caratteri ctrl-S e ctrl-Q di cui sopra, agiscono come X-OFF e X-ON dell'omonimo protocollo di comunicazione e possono quindi essere utilizzati durante l'output di dati via RS232 per sospenderne temporaneamente la trasmissione.

4.5 Bootstrap

La EPROM (EEPROM) contenente l'interprete BASIC, ha uno spazio libero di circa 18 K da 3900H a 7FFFH che può' essere utilizzato per memorizzare in modo permanente inizializzazioni o programmi in BASIC, che verranno caricati in NVRAM automaticamente dopo un reset. Per utilizzare questa caratteristica occorre duplicare la EPROM (EEPROM) contenente il BASIC caricandovi, a partire dall'indirizzo 3900H il listato codificato in ASCII del programma sorgente scritto in BASIC o le inizializzazioni desiderate. E' importante che al termine del programma utente siano presenti i caratteri BRUN seguiti da un return (0DH). Occorre altresì' aggiornare il parametro Numero Caratteri di Autostart, contenuto nelle locazioni 03F5H-03F6H, sostituendo il numero 5 di default (corrispondente a "B" "R" "U" "N" "0D") con il numero totale di caratteri del programma sorgente aumentato di 5.

4.6 Esempio di caricamento su EPROM di un programma

Per memorizzare in modo permanente nell'area libera della EPROM (EEPROM) da 3900H a 7FFFH un programma utente che verra' poi caricato in NVRAM ed eseguito in modo automatico ad ogni reset, occorre procedere come segue.

- Utilizzare un editor per PC (es. MS-DOS EDIT) per scrivere il programma in SPQ-Z80 BASIC definitivo gia` collaudato in precedenza sulla scheda oppure (preferibile) fare un download dalla SPQ-Z80 al PC del listing del programma utilizzando l’opzione del programma di comunicazione. In questo modo e` possibile un debugging interattivo con successivi upload e download del programma tra SPQ-Z80 e PC fino ad ottenere la versione definitiva.
- Aggiungere all’inizio del programma sorgente i caratteri R (seguito da un return) e NEW (seguito da un return), per inizializzare la NVRAM ed alla fine terminarlo con il comando RUN , sempre seguito da return, se si desidera la funzione di autostart. Il comando R del MOS effettua un restart dell'interprete BASIC e NEW provvede alla cancellazione dell'area di NVRAM disponibile per i programmi utente. In questo modo ad ogni reset della scheda la NVRAM viene cancellata e ricaricata con il programma contenuto nella EPROM e lo stesso viene poi avviato automaticamente.

Utilizzando un programmatore di EPROM (EEPROM), preparare il file oggetto definitivo, che dovra' contenere sia il file ASCII del programma utente appena descritto , sia l’interprete BASIC originale , nel seguente modo:

- Copiare nel buffer del programmatore di EPROM la EPROM Z80BAS3.16 da 0000H a 38FFH e copiare il programma in BASIC allocandolo all'indirizzo iniziale 3900H.
- Calcolare la dimensione del programma BASIC e memorizzare tale numero (HEX) nelle locazioni 03F5, 03F6 con l'LSB in 03F5 e il MSB in 03F6 ; questo parametro specifica al MOS il numero di byte da trasferire dalla EPROM (EEPROM) alla NVRAM; se gli unici caratteri esistenti (come nella EPROM standard) sono "BRUN”, si determina la funzione di autostart del programma gia' presente in NVRAM.
- Il file oggetto cosi' ottenuto, dovra' poi essere trasferito su una EPROM (EEPROM) 32K

Esempio di caricamento in EPROM di una riga in BASIC :

10 PRINT "ESEMPIO DI PROGRAMMA CON RESTART AUTOMATICO"

Dimensione del programma = 63 byte = 3FH byte:

03F5 3F	------> byte basso; 03F6 00 ------> byte alto	
	Dump della EPROM programmata:	
3900H 3910H 3920H 3930H	520D4E45570D31305052494E54224553 454D50494F2044492050524F4752414D 4D4120434F4E20524553544152542041 55544F4D415449434F220D52554E0D00	R.NEW.10PRINT"ES EMPIO DI PROGRAM MA CON RESTART A UTOMATICO".RUN..

4.7 Routine di servizio MOS

Alcune subroutine che costituiscono parte integrante del MOS possono essere utilizzate nei programmi applicativi con la istruzione CALL seguita dall'indirizzo della subroutine, previa una eventuale inizializzazione di opportuni parametri. E' importante ricordare che l'istruzione CALL in BASIC richiede per l'indirizzo l'espressione decimale, mentre in linguaggio macchina viene utilizzata la notazione esadecimale (HEX). Verranno descritte nei successivi paragrafi le routine disponibili.

4.7.1 Trasmissione dati a RS232C

Indirizzo : 0347H = 839 . Questa subroutine permette di trasmettere 1 dato di 8 bit, preventivamente caricato nella cella (FEC0H) all'uscita RS232C con il formato parola standard: 1 bit start,8 bit dato,1 bit stop, no parity, e velocità selezionabile da jumpers sulla scheda da 4800 a 38400 baud.

4.7.2 Ricezione dati da RS232C

Indirizzo : 0510H = 1296. Entra in attesa di ricezione di un bit di start da RS232; ricevutolo, carica il dato relativo nella cella (FEC1H).

4.7.3 Visualizzazione caratteri su LCD LM016L

Indirizzo : 03B0hH = 944 Visualizza sul LCD l'area di RAM FE20H-FE3FH in cui occorre quindi preventivamente caricare, codificati in ASCII, i caratteri desiderati ed inizializzare il numero di caratteri (FE08H) e la posizione iniziale cursore (FE0AH).

4.7.4 Reset LCD

Indirizzo : 0370H = 880 Effettua il reset e l'inizializzazione del LCD, e integra le routine di autostart dell'SPQ-Z80 , utilizzando di default il modo a 2 righe.

4.7.5 Visualizzazione data ed ora su LCD

Indirizzo : 0490H = 1168 Visualizza data ed ora sulla prima riga del LCD, utilizzando i dati caricati dal real time clock in NVRAM.

4.7.6 Interrupt

Quando viene accettato un interrupt (la linea INT Z80 va a livello basso) viene generato un puntatore a 16 bit per accedere ad una tabella posta in RAM FF00H-FF7FH che contiene gli indirizzi di partenza delle routine di servizio, allocate ovunque in NVRAM e/o in EPROM (EEPROM); il puntatore viene generato nel seguente modo: il MSB, che per default ha valore FFH, risiede in NVRAM FE16H; il LSB viene letto dal periferico che ha generato l'interruzione che può essere o, per default, PPI1/a, inizializzato nel MODO 1, INPUT, STROBED, oppure, caricando in NVRAM FE07H l'indirizzo 08, la porta seriale RS232C, utilizzando, con il jumper JP9 inserito, il segnale RXRD della UART; il dato acquisito dal periferico interrompente viene moltiplicato per 8 al fine di ottenere lo spazio necessario alle istruzioni di chiamata alle routine di servizio. Con l'hardware allegato a titolo di esempio, viene gestito un interrupt da tastiera a 16 tasti con decodificatore a

4 bit; i 4 bit alti non utilizzati dell'input PPI sono forzati a livello basso da R1-R4. Si realizza in tal modo la gestione di 16 differenti routine di servizio, allocate liberamente in RAM dall'utente. Il MOS può gestire fino a 32 diverse routine di servizio, corrispondenti ai dati 0,1,2...1FH provenienti dal periferico interrompente, che generano a loro volta gli LSB 0,8,10H,...F8H del vettore. I vettori completi, con MSB=default, saranno quindi: FF00H,FF08H,FF10H,.....FF1FH e daranno luogo alla tabella delle chiamate delle routine di servizio interrupt. Un esempio di tale tabella relativo alla gestione della tastiera esadecimale è il seguente:

4.7.6A Tabella	chiamate routine di servizio	interrupt	
TASTO	INDIRIZZO	OPCODE	ASSEMBLER
0	FF00H FF03H	CDaaaa C36802	CALL aaaa JP 0268H
1 . . F	FF08H FF0BH FF78H FF7BH	CDbbbb C36802 CDpppp C36802	CALL bbbb JP 0268H . . CALL pppp JP 0268H

I parametri aaaa,bbbb, ecc. andranno ovviamente sostituiti con i reali indirizzi di partenza delle routine di servizio interrupt, mentre le istruzioni di rientro JP 0268H vanno ripetute invariate, come invariata deve rimanere la spaziatura di 8 locazioni di memoria tra due routine successive. L'interrupt è normalmente abilitato; per disabilitarlo, occorre eseguire l'istruzione OUT 3,8 , per riabilitarlo ulteriormente OUT 3,9; queste istruzioni agiscono sul set/reset del flip-flop di abilitazione interrupt di PPI1 e quindi disattivano solo la periferica. Per disabilitare l'interrupt a livello processore occorre eseguire l'istruzione DI in Assembler, oppure eseguire CALL 766 in BASIC; in questo modo si può disabilitare anche l'interrupt da RS232C.

4.7.7 Abilitazione interrupt

Indirizzo : 02F0H = 752 Determina l'abilitazione dell'interrupt di modo 1 dello Z80 e programma la PPI1 nel seguente modo : PPI1/A e PPI1/Cup = INPUT MODO 1; PPI1/B e PPI1/Clow = OUTPUT MODO 0; FF di interrupt della PPI1 abilitato (set bit 4 della porta di controllo).

4.7.8 Disabilitazione interrupt

Indirizzo : 02FEH = 766 Provoca la disabilitazione dell'interrupt dello Z80 (lasciando invariato lo stato del FF di interrupt di PPI1).

4.7.9 Posizionamento cursore terminale

Indirizzo : 0519H = 1305 Posiziona il cursore del terminale connesso alla porta RS232, secondo le coordinate riga (FED0H) e colonna (FED1H).

4.7.10 Cancellazione fino a fine pagina

Indirizzo : 0300H = 768 Serve a cancellare tutti i caratteri dalla posizione del cursore fino alla fine della pagina.

4.7.11 Acquisizione 16 ingressi ADC

Indirizzo : 36A0H = 13984 Effettua la conversione A/D dei 16 ingressi e memorizza i valori ottenuti nell'area di RAM.

4.7.12 Selezione attributo carattere

Indirizzo : 36B4H = 14004 In abbinamento alla scheda videografica MT91 Viene abilitato il generatore di caratteri semigrafici con repertorio di 64 simboli, inizializzando il colore con un valore da 1 a 7 (FED2H) ed il set grafico con un valore da 3 a 4 (FED3).

4.7.13 Composizione di rettangoli

Indirizzo : 36C8H = 14024 In abbinamento alla scheda MT91 viene composto un rettangolo con posizione e dimensioni definite dall'utente. Il riferimento è lo spigolo in alto a sinistra.

PARAMETRO	LOC (HEX)	LOC (DEC)
riga (32-55)	FED0H	65232
colonna (32-71)	FED1H	65233
colore (01-07)	FED2H	65234
altezza (1-21)	FED4H	65236
base (1-37)	FED5H	65237

4.7.14 Generazione treno onde quadre

Indirizzo : 3794H = 14228 Viene generato un treno di onde quadre di frequenza = 2 KHz, ampiezza TTL, durata 0,1 s. Questa subroutine è utilizzata unitamente al circuito di watch-dog timer per gestire una supervisione del software a livello alto.

4.7.15 Rientro al Monitor

5 - GESTIONE DEI PERIFERICI

5.1 Convertitore analogico-digitale.

Il convertitore A/D ADC0817 integra al suo interno un multiplexer a 16 canali e quindi, per iniziare una conversione necessita oltre che di un impulso di start, anche dell'indirizzo del canale desiderato. Entrambi questi segnali vengono forniti, in BASIC, con l'istruzione OUT 12,CH dove 12 e l'indirizzo dell'ADC0817 e CH è un numero variabile tra 0 e 15 che indica, appunto, il numero del canale. Dopo circa 50 microsecondi il convertitore rende disponibile sul bus dati il risultato della conversione che può quindi essere letta con l'istruzione R= INP(12) con R variabile qualsiasi; dato che il tempo di esecuzione di queste istruzioni in BASIC è superiore a quello di conversione, non è necessario eseguire un test sulla linea EOC (KC-3) che segnala, andando a livello basso la conclusione del ciclo di conversione; questo segnale può viceversa essere utile per routine di lettura scritte in Assembler Z80, dove i tempi medi di esecuzione sono di pochi microsecondi.

5.2 Interfacce parallele programmabili.

Il PPI 8255 è un dispositivo I/O programmabile che può essere utilizzato in 3 modi principali di funzionamento. Nel MODO 0, che viene utilizzato per l'interfacciamento con dispositivi lenti (pulsanti, display, relè, sensori meccanici ecc.) e quindi il dato in uscita è memorizzato in un registro (latch), mentre non lo e’ quello in ingresso, sono disponibili 2 porte (A e B) da 8 bit e 2 porte (C upper/ C lower) da 4 bit, ognuna definibile singolarmente come ingresso o uscita. La scelta della configurazione voluta si attua inviando un codice opportuno alla porta di controllo del PPI; gli indirizzi delle porte di controllo di PPI1 e PPI2 sono rispettivamente 3 e 7 (cfr. mappa memoria); occorre quindi eseguire l'istruzione di output OUT 3,x oppure OUT 7,x in cui i 16 possibili codici x (decimale per il BASIC e HEX per l'Assembler) sono dati dalla seguente tabella:

HEX	DEC	A	B	C up	C low
80	128	OUT	OUT	OUT	OUT
81	129	OUT	OUT	OUT	IN
82	130	OUT	IN	OUT	OUT
83	131	OUT	IN	OUT	IN
88	136	OUT	OUT	IN	OUT
89	137	OUT	OUT	IN	IN
8A	138	OUT	IN	IN	OUT
8B	139	OUT	IN	IN	IN
90	144	IN	OUT	OUT	OUT
91	145	IN	OUT	OUT	IN
92	146	IN	IN	OUT	OUT
93	147	IN	IN	OUT	IN
98	152	IN	OUT	IN	OUT
99	153	IN	OUT	IN	IN
9A	154	IN	IN	IN	OUT
9B	155	IN	IN	IN	IN

Il modo 0 viene utilizzato per la PPI2 nell'interfacciamento del LCD; il codice utilizzato è 88 (HEX). Il MODO 1 di funzionamento, e' adatto all'interfacciamento con dispositivi veloci e quindi prevede una gestione handshaking dei segnali di I/O, cioè con segnali di strobe; un esempio proposto è l'utilizzo della tastiera esadecimale per cui si adotta il codice di controllo B8 (HEX). Per ricavare i codici di controllo relativi al modo 1 si deve fare riferimento a pag. 2-67/68 del Data-sheet della Intel.

6 - RISORSE DI MEMORIA E I/O

6.1 Indirizzi di I/O

Indirizzo (HEX) I/O

0000	PPI	1/A
0001	PPI	1/B
0002	PPI	1/C
0003	PPI	1/CONT.
0004	PPI	2/A
0005	PPI	2/B
0006	PPI	2/C
0007	PPI	2/CONT.
0008 : 000B	UART	
000C : 000F	ADC	

6.2 Indirizzi di memoria

Ind.	HEX		EPROM/RAM	
0000	:	7FFF	EPROM	
8000	:	FBFF	NVRAM	(BASIC)
FC00	:	FDFF	NVRAM	(libera)
FE00	:	FFFF	NVRAM	(MOS)

6.3 Mappa area NVRAM riservata a variabili e costanti MOS

IND. HEX	IND.(DEC)	Parametro MOS
FE00	65024	Num.car.di bootstrap (LSB)
FE01	65025	" " (MSB)
FE02	65026	Ind.car. di bootstrap (LSB)
FE03	65027	" " (MSB)
FE04	65028	Function set LCD LM016L
FE05	65029	Entry mode set " "
FE06	65030	Flag abilitazione LCD e stampante
FE07	65031	Indirizzo periferico interrupt
FE08	65032	Numero caratteri per LCD
FE0A	65034	Posizione cursore LCD
FE0F	65039	Giorno RTC MOS
FE10	65040	Secondi “”
FE11	65041	Minuti “”
FE12	65042	Ore “”
FE13	65043	Giorno del mese “”
FE14	65044	Mese “”
FE15	65045	Anno “”
FE16	65046	Vettore di interrupt MSB
FE20	65056	Inizio buffer LCD
FE3F	65087	Fine buffer LCD
FEB0	65200	Inizio area dati acquisiti ADC
FEBF	65215	Fine area dati acquisiti ADC
FEC0	65216	Buffer carattere TX RS-232
FEC1	65217	Buffer carattere RX RS-232
FED0	65232	Posizione verticale cursore PC
FED1	65233	Posizione orizzont. cursore PC
FED2	65234	Colore carattere videografica
FED3	65235	Attributo carattere

FED4	65236	Altezza rettangolo
FED5	65237	Base rettangolo
FEE0	65248	Tabella codifica tastiera
FF00	65280	Tabella routine serv. interrupt
FFF0	65520	Routine di restart servizio NMI
FFF8	65528	Registro controllo RTC DS1644
FFF9	65529	Secondi 00-59 “”
FFFA	65530	Minuti 00-59 “”
FFFB	65531	Ore 00-23 “”
FFFC	65532	Giorno 01-07 “”
FFFD	65533	Data 01-31 “”
FFFE	65534	Mese 01-12 “”
FFFF	65535	Anno 00-99 “”