Sessione ordinaria 1997
Seconda prova scritta
04372.2.1/1

Ministero della Pubblica Istruzione

M323 - ESAMI DI MATURITÀ TECNICA INDUSTRIALE

NUOVO ORDINAMENTO

Indirizzo: ELETTRONICA E TELECOMUNICAZIONI

Sopra un nastro che deve scorrere ad una velocità variabile tra 1 e 10 m/sec sono disegnate linee nere dello spessore di 1 mm e distanziate di 10 cm. Un elemento fotosensibile rileva la presenza delle linee generando un impulso in corrispondenza di ciascuna di esse. A partire da questa rilevazione si desidera realizzare un sistema che consenta la rilevazione della velocità in m/sec con la precisione di due decimali e permetta inoltre:

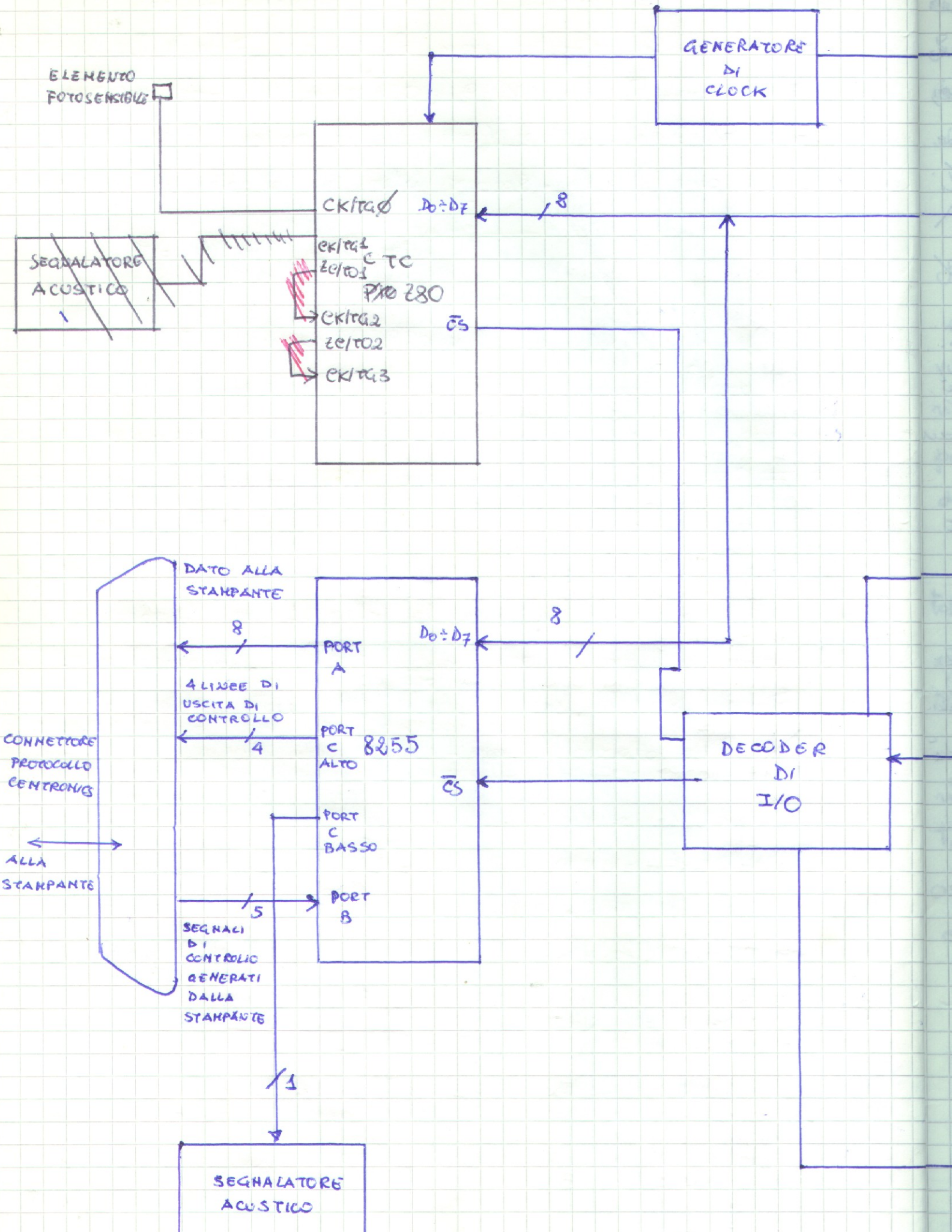
- la sua lettura in ogni momento da parte di un operatore umano,
- l'attivazione di un segnale acustico ed eventualmente di un avviso scritto se la velocità supera il massimo o scende al di sotto del minimo stabiliti,
- la registrazione ogni 5 minuti dei valori della velocità e la creazione di una tabella con i valori rilevati in 24 ore,
- la stampa ogni 24 ore della tabella dei valori associati all'ora del loro rilevamento e di un diagramma che indichi, per intervalli di velocità di un metro al secondo, per quanto tempo nelle 24 ore la velocità si è mantenuta in ciascun intervallo.

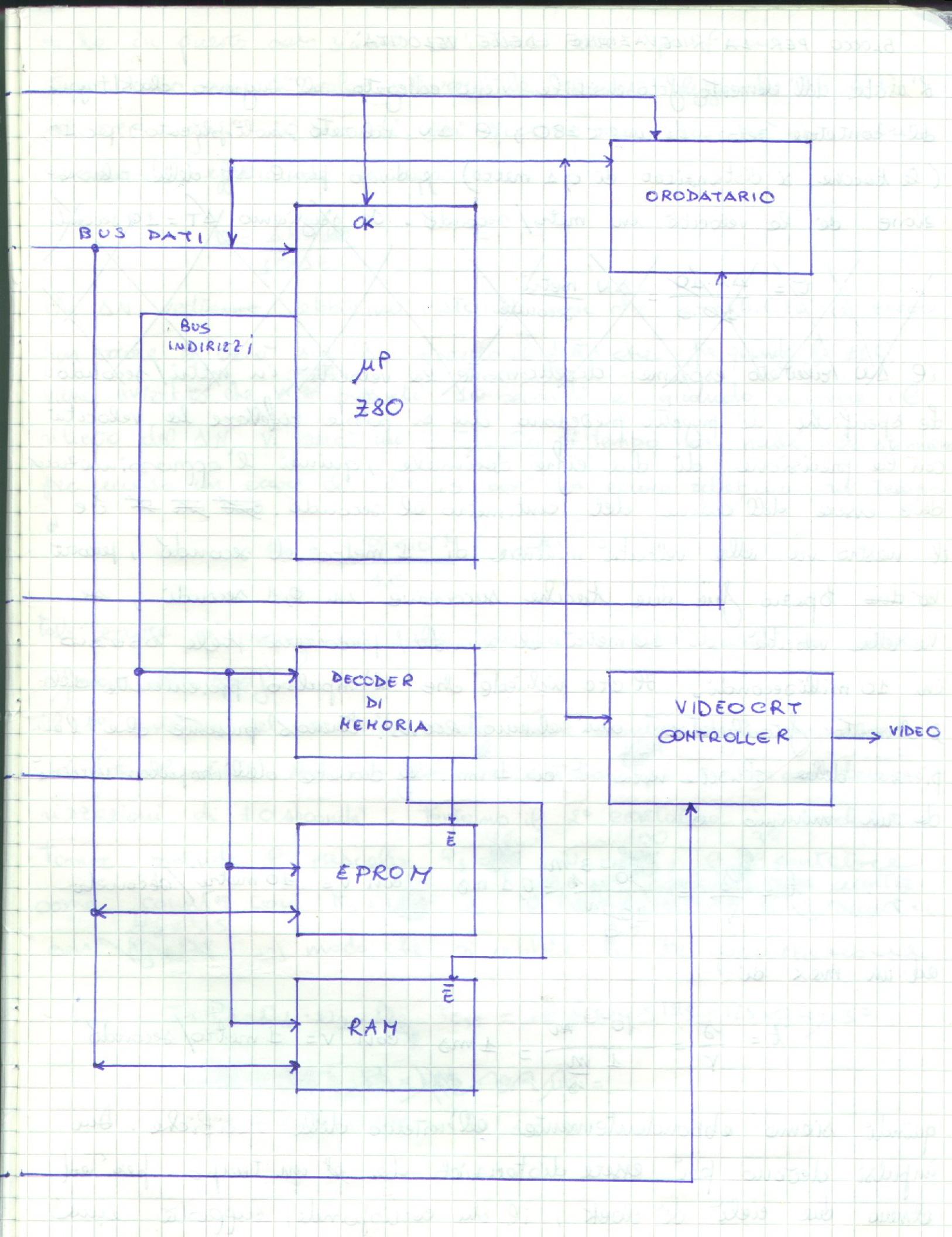
Il candidato deve proporre per il sistema richiesto una soluzione che utilizzi componenti o apparati programmabili. In particolare, formulate le ipotesi aggiuntive che ritiene necessarie, deve:

1. proporre lo schema generale del sistema illustrando la funzione ed il tipo di prestazione richiesta ai singoli blocchi,

2. proporre una realizzazione, con componenti, apparati e linguaggi di sua conoscenza e discutendo in particolare i problemi posti dai diversi valori di velocità a cui può scorrere il nastro:
   a) della interfaccia destinata alla acquisizione dei dati e del programma che la governa,
   b) di almeno un altro blocco di sua scelta oppure del programma per le elaborazioni e le stampe previste ogni 24 ore.

Durata massima della prova: 6 ore.
E' consentito l'uso di manuali tecnici, documentazione sulle caratteristiche dei componenti,


GENERATORE DI CLOCK
ELEMENTO FOTOSENSIBILE
CK/TG0
D0÷D7
8
SEGNALATORE ACUSTICO
1
CK/TG1
ZC/TO1
CTC
PIO Z80
CK/TG2
CS
ZC/TO2
CK/TG3
DATO ALLA STAMPANTE
8
PORT A
D0÷D7
8
4 LINEE DI USCITA DI CONTROLLO
4
PORT C ALTO
8255
CONNETTORE PROTOCOLLO CENTRONICS
CS
DECODER DI I/O
PORT C BASSO
ALLA STAMPANTE
5
PORT B
SEGNALI DI CONTROLLO GENERATI DALLA STAMPANTE
1
SEGNALATORE ACUSTICO


OROLOGIO
CK
BUS DATI
BUS INDIRIZZI
µP
Z80
DECODER DI MEMORIA
VIDEOCRT CONTROLLER
VIDEO
$\overline{E}$
EPROM
$\overline{E}$
RAM

BLOCCO PER LA RILEVAZIONE DELLE VELOCITÀ

L'uscita dell'elemento fotosensibile viene collegata all'ingresso clock/trigger del contatore zero di un CTC280; ie ΔN rilevato, moltiplicato per 10 (le tacche si distanziano di 0,1 metri) e diviso per ie ΔT della rilevazione dà la velocità in metri/secondo. Se scegliamo $\Delta T = 10$ secondi

$$v = \frac{\Delta N \cdot 10}{10\ s} = \Delta N \ \frac{\text{metri}}{\text{secondo}}$$

ie ΔN rilevato esprime direttamente la velocità in metri/secondo. Le specifiche di progetto prevedono che si possa rilevare la velocità con la precisione di due cifre decimale, quindi l'approssimazione deve essere dell'ordine del centimetro al secondo. Se il nastro va alla velocità minima di 1 metro al secondo, percorre lo spazio fra due tacche successive in 0,1 secondi, se va alla velocità di 10 metri al secondo, percorrerà tale spazio in 10 millisecondi. Il CTC richiede che l'impulso proveniente dall'elemento fotorilevatore duri almeno 200 ns. Tenendo presente che la piazza delle strisce nere è di 1 mm. la durata dell'impulso varierà da un minimo di

$$t = \frac{s}{v} = \frac{10^{-3}\ m}{\frac{10\ m}{s}} = 0{,}1\ ms \qquad \text{con } v = 10 \text{ metri/secondo}$$

ad un max di

$$t = \frac{s}{v} = \frac{10^{-3}\ m}{\frac{1\ m}{s}} = 1\ ms \qquad \text{con } v = 1 \text{ metro/secondo}$$

quindi siamo abbondantemente all'interno delle specifiche. Due impulsi devono poi essere distanziati da un tempo pari ad almeno due cicli di clock, il che corrisponde, supposto una

l'uscita dell'elemento fotosensibile viene collegata all'ingresso clock/trigger del contatore zero di un CTC280; ie ΔN rilevato, moltiplicato per 10 (le tacche si distanziano di 0,1 metri) e diviso per ie ΔT della rilevazione dà la velocità in metri/secondo. Se scegliamo $\Delta T = 10$ secondi

$$v = \frac{\Delta N \cdot 10}{10\ s} = \Delta N \ \frac{\text{metri}}{\text{secondo}}$$

ie ΔN rilevato esprime direttamente la velocità in metri/secondo. Le specifiche di progetto prevedono che si possa rilevare la velocità con la precisione di due cifre decimale, quindi l'approssimazione deve essere dell'ordine del centimetro al secondo. Se il nostro va alla velocità minima di 1 metro al secondo, percorre lo spazio fra due tacche successive in 0,1 secondi, se va alla velocità di 10 metri al secondo, percorrerà tale spazio in 10 millisecondi. Il CTC richiede che l'impulso proveniente dall'elemento fotorilevatore duri almeno 200ns. Tenendo presente che la piazza delle strisce nere è di 1 mm. La durata dell'impulso varierà da un minimo di

$$t = \frac{s}{v} = \frac{10^{-3}\ m}{10\ \frac{m}{s}} = 0{,}1\ ms \quad \text{con } v = 10 \text{ metri/secondo}$$

ad un max di

$$t = \frac{s}{v} = \frac{10^{-3}\ m}{1\ \frac{m}{s}} = 1\ ms \quad \text{con } v = 1 \text{ metro/secondo}$$

quindi siamo abbondantemente all'interno delle specifiche. Due impulsi devono poi essere distanziati da un tempo pari ad almeno due cicli di clock, il che corrisponde, supposta una frequenza di 2,5 MHz a 0,8 μs e, come abbiamo visto prima

anche in questo caso siamo all'interno delle specifiche.
Supponiamo ora di voler rilevare la velocità leggendo il ΔN del CTC ogni 10 secondi. Noi otteniamo la velocità in metri al secondo effettuando il rapporto

$$v = \frac{\Delta N \times 100}{10 \cdot \Delta t}$$

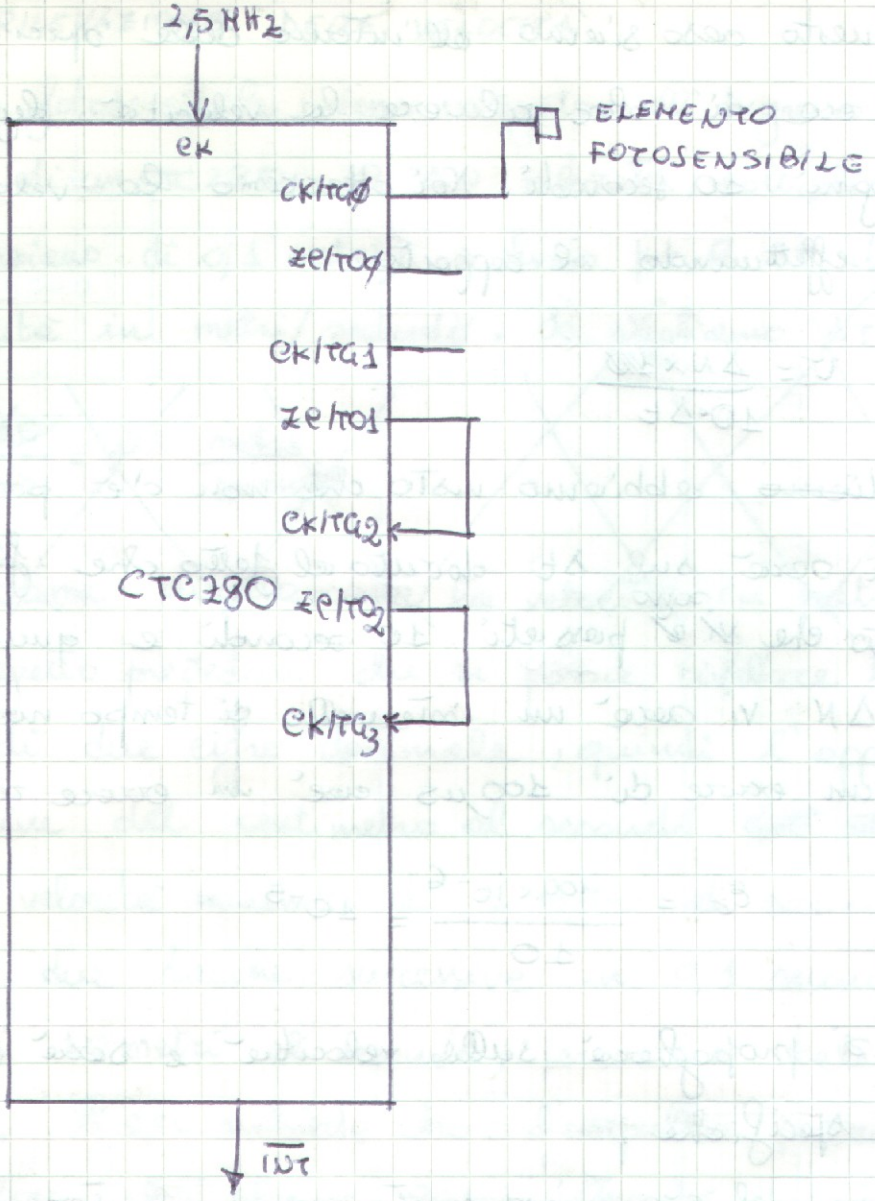
Su ΔN vediamo abbiamo visto che non c'è possibilità di errore un errore ci sarà sul Δt dovuto al fatto che, fra quando lo Z80 viene avvertito che sono passati 10 secondi e quando avviene il rilievo del ΔN vi sarà un intervallo di tempo non nullo. Se stimiamo per eccesso un errore di 100 μs avremo un errore relativo sul tempo di

$$\varepsilon_t = \frac{100 \times 10^{-6}}{10} = 10^{-5}$$

Tale errore si propagherà sulla velocità e sarà quindi un errore inferiore alle specifiche.
Il 1° contatore del CTC viene usato come counter degli impulsi del fotorilevatore. Usiamo gli altri contatori per realizzare temporizzazioni di 10 secondi. Poniamo il 2° contatore in modalità timer ponendo il prescaler $P_1 = 16$, $N_1 = 100$; il 3° contatore come counter con $N_2 = 125$ e il 4° contatore come counter con $N_3 = 125$ in modo che si abbia la temporizzazione

$$T = P_1 \cdot N_1 \cdot N_2 \cdot N_3 \cdot T_{CK} = 16 \times 100 \times 125 \times 125 \times 0{,}4\,\mu s =$$

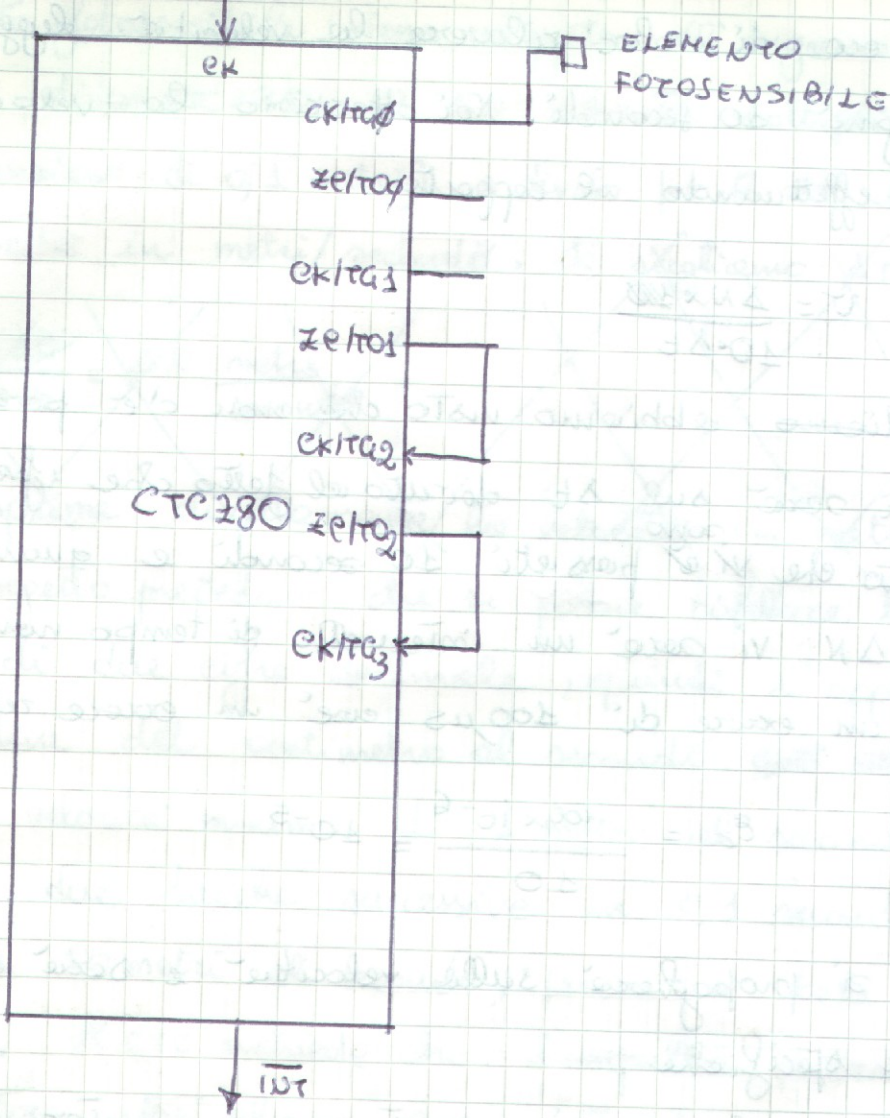
$$= 10.000.000\,\mu s = 10\,s$$



COLLEGAMENTO CON LA STAMPANTE.

Supponiam Colleghiamo il µP alla stampante con un connettore di tipo Centronics. Nella tabella a fianco vi è la enumerazione dei pin di tale connettore con indicazione della loro funzione. Notiamo che occorrono 8 linee di uscita per inviare i dati alla stampante, le 4 linee di uscita per i segnali di controllo di STROBE, per la sincronizzazione, AUTOFD per l'avanzamento automatico di riga, INIT per inizializzare la stampante, SLCT IN per selezionare la stampante; infine vi sono 5 ingressi per i segnali di controllo generati dalla stampante.

Come si vede dallo schema a blocchi, l'interfacciamento è



COLLEGAMENTO CON LA STAMPANTE.

Supponiam Colleghiamo il μP alla stampante con un connettore di tipo Centronics. Nella tabella a fianco vi è la enumerazione dei pin di tale connettore con indicazione delle loro funzione. Notiamo che occorrono 8 linee di uscita per inviare i dati alla stampante, le 4 linee di uscita per i segnali di controllo di STROBE, per la sincronizzazione, AUTOFD per l'avanzamento automatico di riga INIT per inizializzare la stampante, SLCT IN per selezionare la stampante; infine vi sono 5 ingressi per i segnali di controllo generati dalla stampante.

Come si vede dallo schema a blocchi, l'interfacciamento è affidato ad un 8255. La porta A è usata per il trasferimento di dati, le 4 linee di controllo di ~~ingresso~~ uscita sono

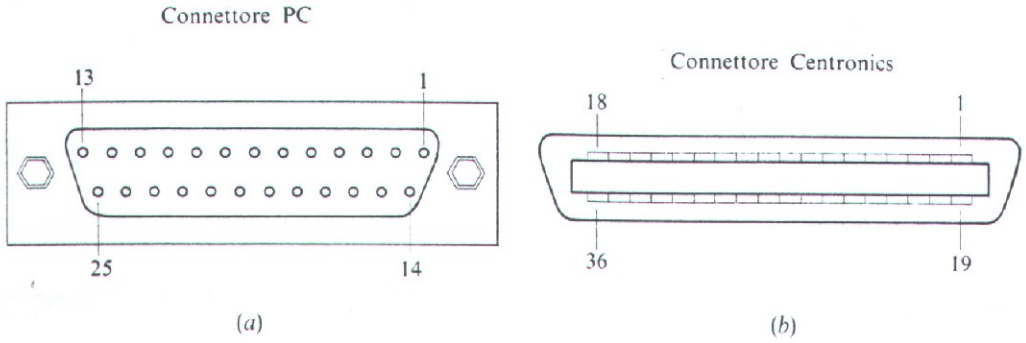


FIG. 1
Connettori: a) *della porta parallela del PC (vista frontale)* e b) *della stampante tipo Centronics (vista frontale).*

disponibili sul connettore del PC sono direttamente legati alla funzionalità delle stampanti, i cui manuali portano sempre una chiara descrizione dei segnali stessi.

Poiché però la stampante è in genere dotata di un connettore di tipo Centronics a 36 pin (*vedi* fig. 1*b*), è utile rappresentare in una tabella (*vedi* tab. 1) le corrispondenze fra i pin del connettore della porta parallela del PC e quelli del connettore della stampante, accanto all'indicazione delle relative funzioni. Il trattino che precede il simbolo di alcune funzioni

TAB. 1

Corrispondenze fra connettore del PC, connettore Centronics e rispettivi indirizzi di I/O.

Connettore PC Pin n.	*Dir. I/O*	*Connettore Centronics Pin n.*	*Funzione*	*Indirizzo di I/O (Hex)*	*Posizione*	*Descrizione*
1	O	1	-STROBE	037A	$\overline{\text{bit 0}}$	Sincronizza il trasferimento
2	O	2	D0	0378	bit 0	Dato da trasferire (LSB)
3	O	3	D1	0378	bit 1	Dato da trasferire
4	O	4	D2	0378	bit 2	Dato da trasferire
5	O	5	D3	0378	bit 3	Dato da trasferire
6	O	6	D4	0378	bit 4	Dato da trasferire
7	O	7	D5	0378	bit 5	Dato da trasferire
8	O	8	D6	0378	bit 6	Dato da trasferire
9	O	9	D7	0378	bit 7	Dato da trasferire (MSB)
10	I	10	-ACK	0379	bit 6	Conferma ricezione dato
11	I	11	BUSY	0379	$\overline{\text{bit 7}}$	Stampante occupata o non pronta
12	I	12	PE	0379	bit 5	Fine carta
13	I	13	SLCT	0379	bit 4	Stampante selezionata (ON-LINE)
14	O	14	-AUTOFD	037A	$\overline{\text{bit 1}}$	Avanzamento riga automatico
15	I	32	-ERROR	0379	bit 3	Errore hardware
16	O	31	-INIT	037A	bit 2	Inizializza la stampante
17	O	36	-SLCT IN	037A	$\overline{\text{bit 3}}$	Seleziona la stampante
18-25		16,19-30,33	GND			

collegate alla parte alta della porta C, alla parte bassa è collegata la sirena di allarme; infine alla porta B sono collegati i segnali di controllo d'ingresso.

Nella pagina seguente si può vedere il diagramma temporale dei segnali di comunicazione secondo il protocollo centronics. Il µP deve inizializzare la stampante con INIT e dopo aver verificato che l'ingresso SLCT è alto (che assicura che la stampante è ON-LINE), invia il dato in uscita, ~~attende~~ controllando che la linea BUSY sia bassa in modo da verificare che la stampante non sia occupata o non pronta, se è bassa, convalida il trasferimento premendo STROBE basso. La stampante conferma l'avvenuta ricezione con ACK basso e disattiva poi BUSY in modo che possa essere inviato un nuovo dato.

ORODATARIO.

Il nucleo dell'orodatario è un circuito integrato M8716A che contiene un orologio digitale a quarzo con un oscillatore al quarzo da 32KHz ed una interfaccia bus seriale (I²C BUS). Il circuito è programmabile per contare secondi, minuti, ore, giorni e mesi o secondi, minuti, ore e giorni della settimana. La scrittura (settaggio dell'orario) e la lettura dei contatori è fatta attraverso un'interfaccia seriale. La frequenza di oscillazione è prima divisa per 256 e poi di nuovo per 128. La frequenza risultante di 1Hz serve come impulso per i contatori. I contenuti dei contatori possono essere letti o modificati attraverso l'interfaccia I²C. In un ciclo di scrittura solo i contenuti dei contatori a partire da quello dei minuti possono essere modificati: il contatore dei secondi ed il blocco di divisione dei secondi vengono resettati a zero. La selezione tra le opzio-

di ottone; infine alla porta B è collegati i segnali di controllo d'ingresso.

Nella pagina seguente si può vedere il diagramma temporale dei segnali di comunicazione secondo il protocollo centronics. Il µP deve inizializzare la stampante con INIT e dopo aver verificato che l'ingresso ~~INIT~~ SLCT è alto (che indica che la stampante è ON-LINE), invia il dato in uscita, ~~ottiene STROBE~~ controllo che la linea BUSY sia bassa in modo da verificare che la stampante non sia occupata o non pronta, se è bassa, convalida il trasferimento mandando STROBE basso. La stampante conferma l'avvenuta ricezione con ACK basso e dopodiché pone BUSY in modo che possa essere inviato un nuovo dato.

ORODATARIO.

Il nucleo dell'orodatario è un circuito integrato M8716A che contiene un orologio digitale a ~~quarzo~~ con un oscillatore al quarzo da 32KHz ed una interfaccia bus seriale (I^2C BUS). Il circuito è programmabile per contare secondi, minuti, ore, giorni e mesi o secondi, minuti, ore e giorni della settimana. La scrittura (settaggio dell'orario) e la lettura dei contatori è fatta attraverso un'interfaccia seriale. La frequenza di oscillazione è prima divisa per 256 e poi di nuovo per 128. La frequenza risultante di 1Hz serve come impulso per i contatori. I contenuti dei contatori possono essere letti o modificati attraverso l'interfaccia I2C. In un ciclo di scrittura solo i contenuti dei contatori a partire da quello dei minuti possono essere modificati: il contatore dei secondi ed il blocco di divisione dei secondi vengono resettati a zero. La selezione tra le opzioni "calendario" o "giorno della settimana" è fatta come segue: se il secondo bit nel primo byte di dati è 1 du-

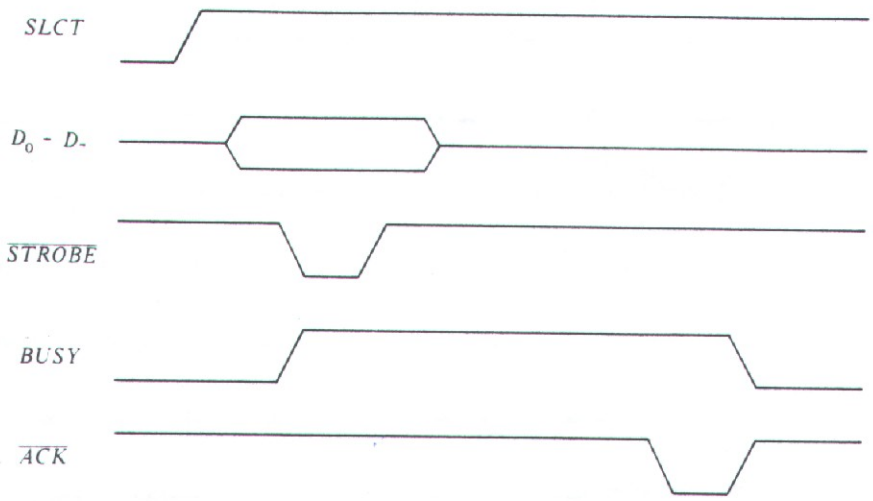
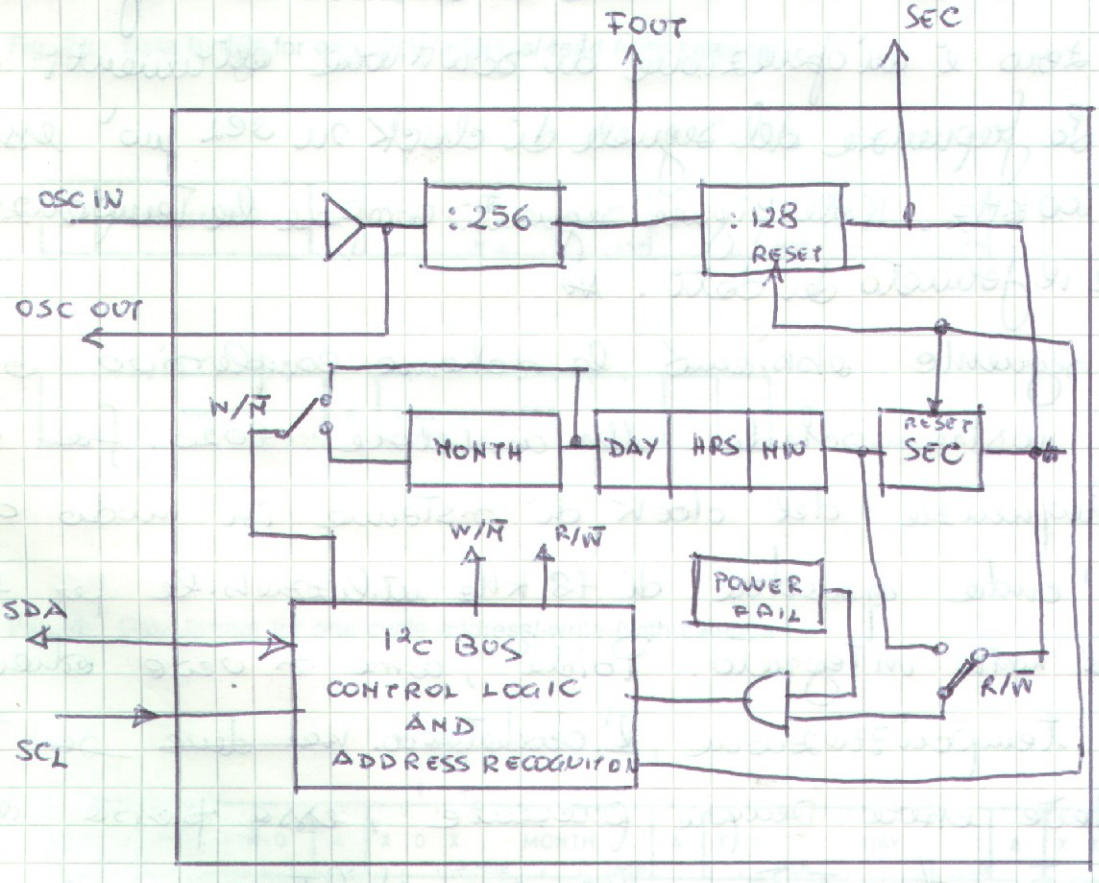


FIG. 3
Rappresentazioni dei segnali attivati secondo il protocollo Centronics.



SCHEMA A BLOCCHI DEL M8716

riante una operazione di scrittura, i contatori sono settati per la modalità "giorno della settimana". Se questo bit

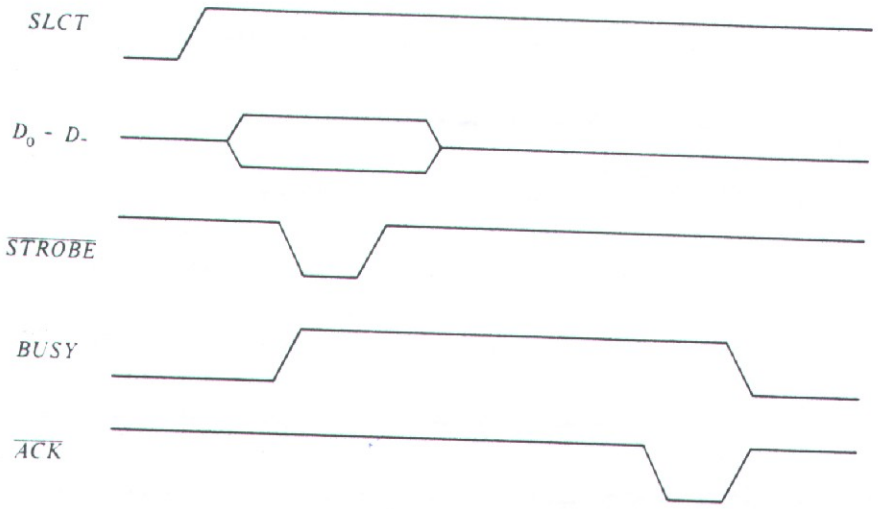
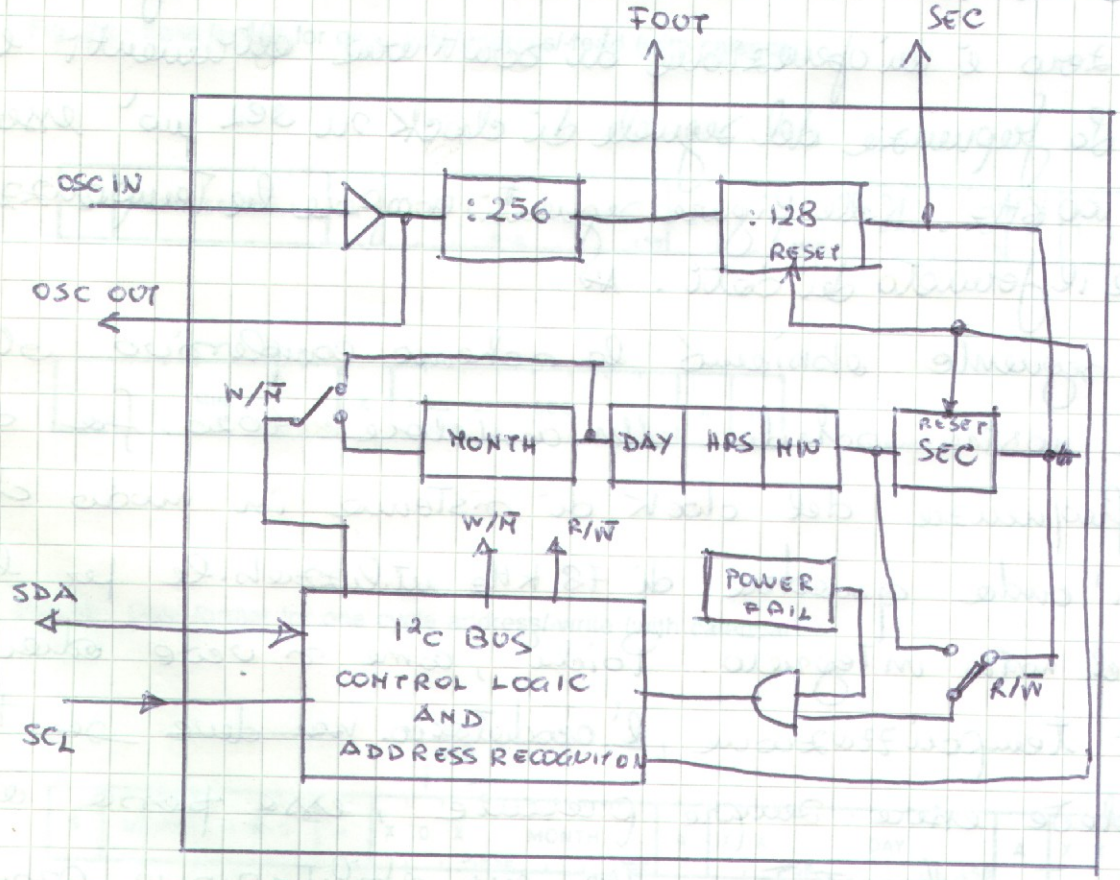


FIG. 3
Rappresentazioni dei segnali attivati secondo il protocollo Centronics.



SCHEMA A BLOCCHI DEL M8716

viene una operazione di scrittura, i contatori sono settati per la modalità "giorno della settimana". Se questo bit rimane a zero è selezionata la modalità "calendario". In questa modalità il riporto per il contatore "giorno"

è effettuato automaticamente alle posizioni 28, 30 o 31, in dipendenza del mese. In caso di anno bisestile il giorno 29 di febbraio può essere settato mediante un'operazione di scrite. Il trasferimento di dati dal circuito al microprocessore e viceversa avviene tramite le due linee SDA e SCL. Indirizzo e dati sono trasmessi sulla linea SDA mentre ~~un~~ gli impulsi di clock di sincronizzazione sono trasmessi sulla linea SCL. Un trasferimento dati, sia in lettura che in scrittura, inizia con una condizione di start (una transizione da 1 a zero sulla linea SDA mentre SCL rimane ad 1) e un successivo byte d'indirizzo. Se l'HS716A riconosce un indirizzo trasmesso sul bus come il suo indirizzo, inizia il trasferimento dato. Il bit meno significativo dell'indirizzo controlla la direzione del trasferimento dati. Se è a zero è un'operazione di scrittura altrimenti è una lettura. La frequenza del segnale di clock su SCL può essere al massimo di 100 kHz. Nelle figure seguenti compare la Temporizzazione completa e il formato dei dati. NO

Nella pagina seguente abbiamo lo schema completo di utilizzo nella nostra scheda. Un contatore 4020 fa da divisore di frequenza del clock di sistema in modo che otteniamo un'onda quadra di 78 kHz utilizzabile per l'ingresso SCL del nostro integrato. Poiché, come si vede dai diagrammi di temporizzazione, l'orologio non deve ~~se~~ tale onda non deve essere sempre presente, esso passa attraverso una porta three state, la cui abilitazione proviene da un flip flop D che memorizza il valore di un uscita del decoder di I/O. In tal modo, quando si attiva questa uscita scrivendo ad un particolare indirizzo di I/O, da

scrite. Il trasferimento di dati dal circuito al microprocessore e viceversa avviene tramite le due linee SDA e SCL. Indirizzo e dati sono trasmessi sulla linea SDA mentre ~~un~~ gli impulsi di clock di sincronizzazione sono trasmessi sulla linea SCL. Un trasferimento dati, sia in lettura che in scrittura, inizia con una condizione di start (una transizione da 1 a zero sulla linea SDA mentre SCL rimane ad 1) e un successivo byte d'indirizzo. Se l'H8716A riconosce un indirizzo trasmesso sul bus come il suo indirizzo, inizia il trasferimento dato. Il bit meno significativo dell'indirizzo controlla la direzione del trasferimento dati. Se è a zero è un'operazione di scrittura altrimenti è una lettura. La frequenza del segnale di clock su SCL può essere al massimo di 100 kHz. Nelle figure seguenti compare la temporizzazione completa e il formato dei dati.

Nelle pagine seguente abbiamo lo schema complessivo di utilizzo nelle nostre schede. Un contatore 4020 fa da divisore di frequenza del clock di sistema in modo che otteniamo un'onda quadra di 78 kHz utilizzabile per l'ingresso SCL del nostro integrato. Poiché, come si vede dai diagrammi di temporizzazione, l'orologio non deve essere tale onda non deve essere sempre presente, essa passa attraverso una porta three-state, la cui abilitazione proviene da un flip flop D che memorizza il valore di un uscita del decoder di I/O. In tal modo, quando si attiva questa uscita scrivendo ad un particolare indirizzo di I/O, da quel momento in poi il valore zero è memorizzato all'uscita del flip flop e la porta three-state viene abi-

Fig. 1 - Complete timing for an address/-read; resp. address/-write cycle

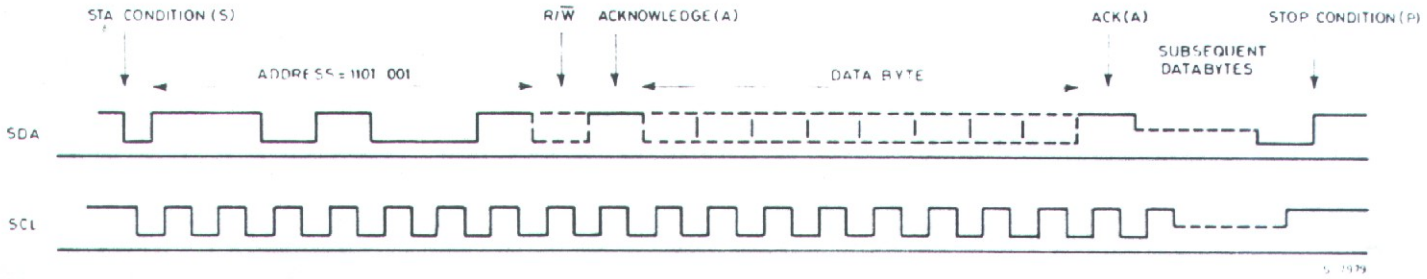


Fig. 2a - Data format for one cycle address/-read (with calendar)

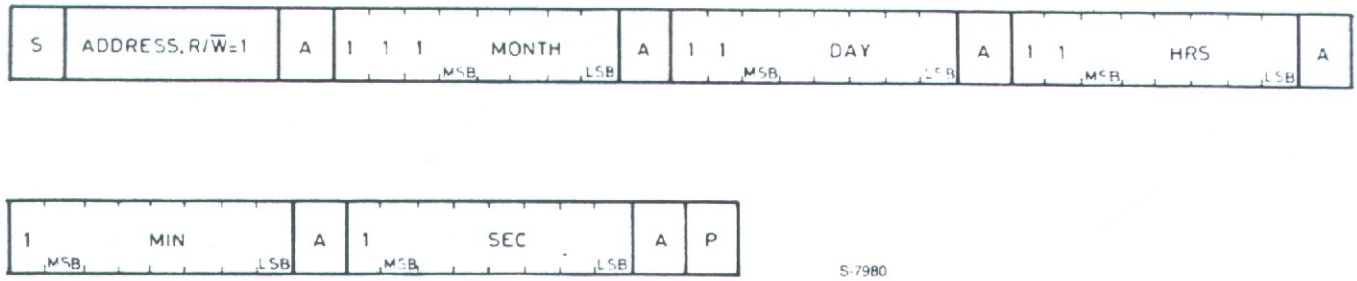


Fig. 2b - Data format for one cycle address/-write (with calendar)

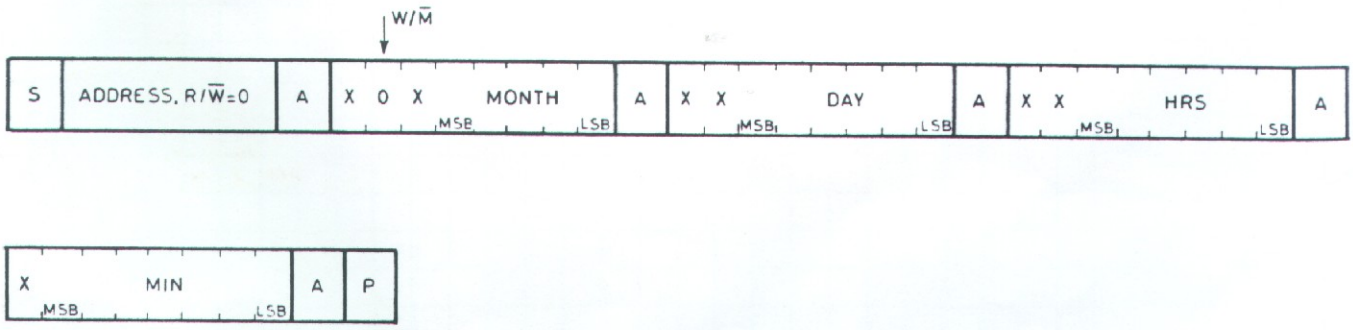


Fig. 3a - Data format for one cycle address/-read (with day of week indication)

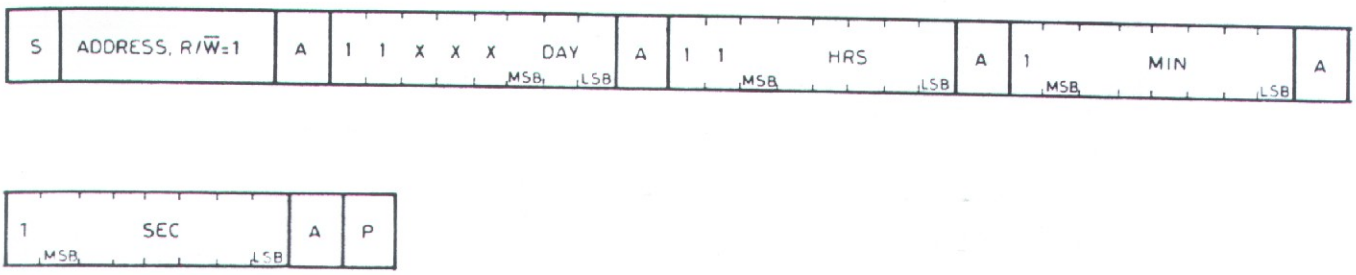


Fig. 3b - Data format for one cycle address/-write (with day of week indication)

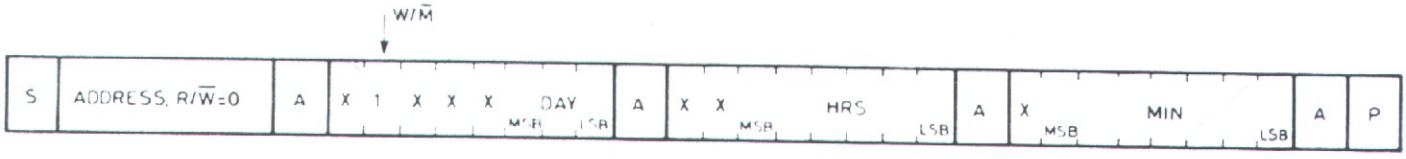


Fig. 4 - Data format for synchronisation (deviation < 30sec)

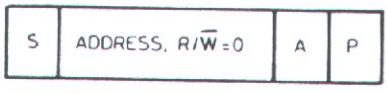


Fig. 5 - Test circuit

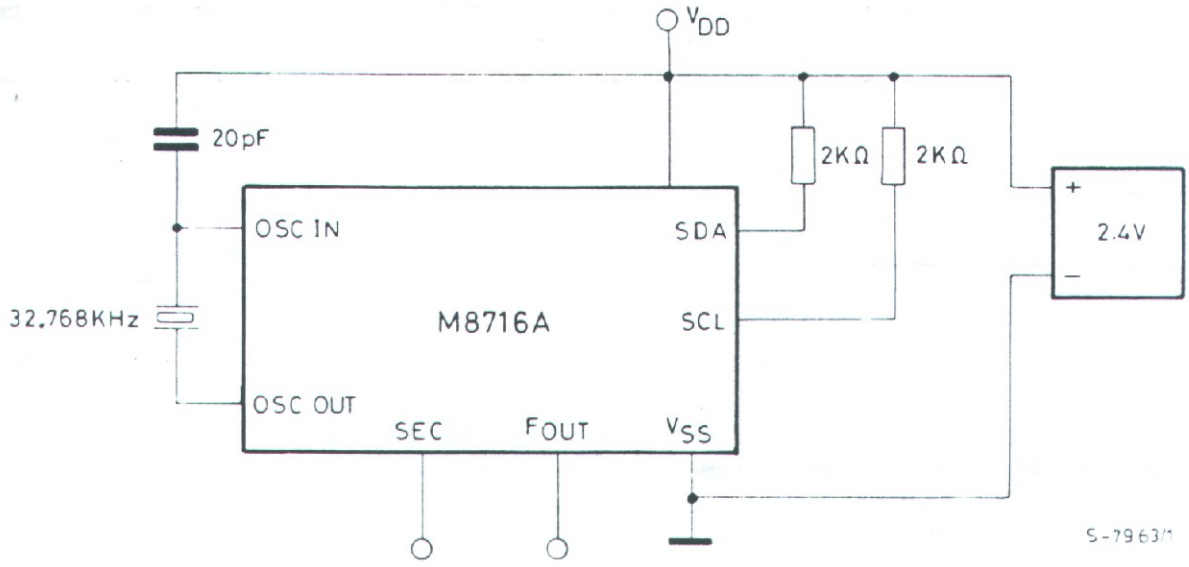
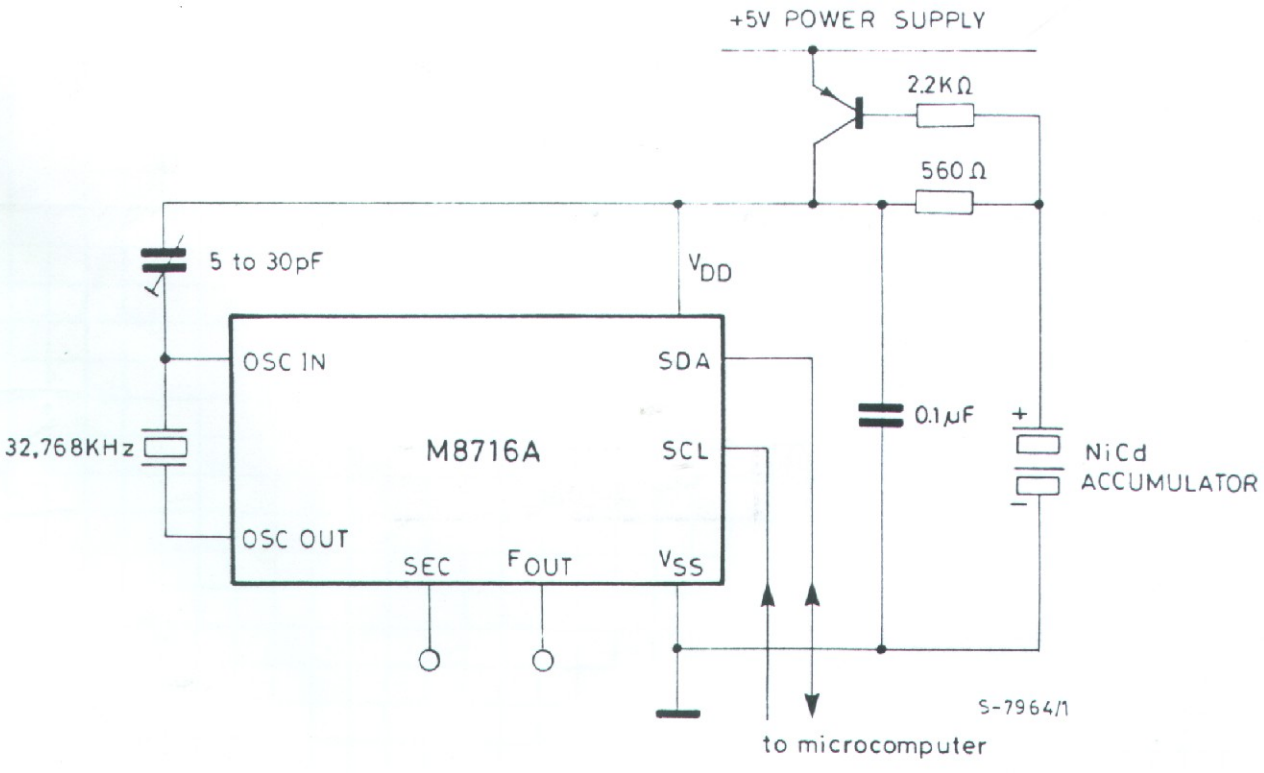
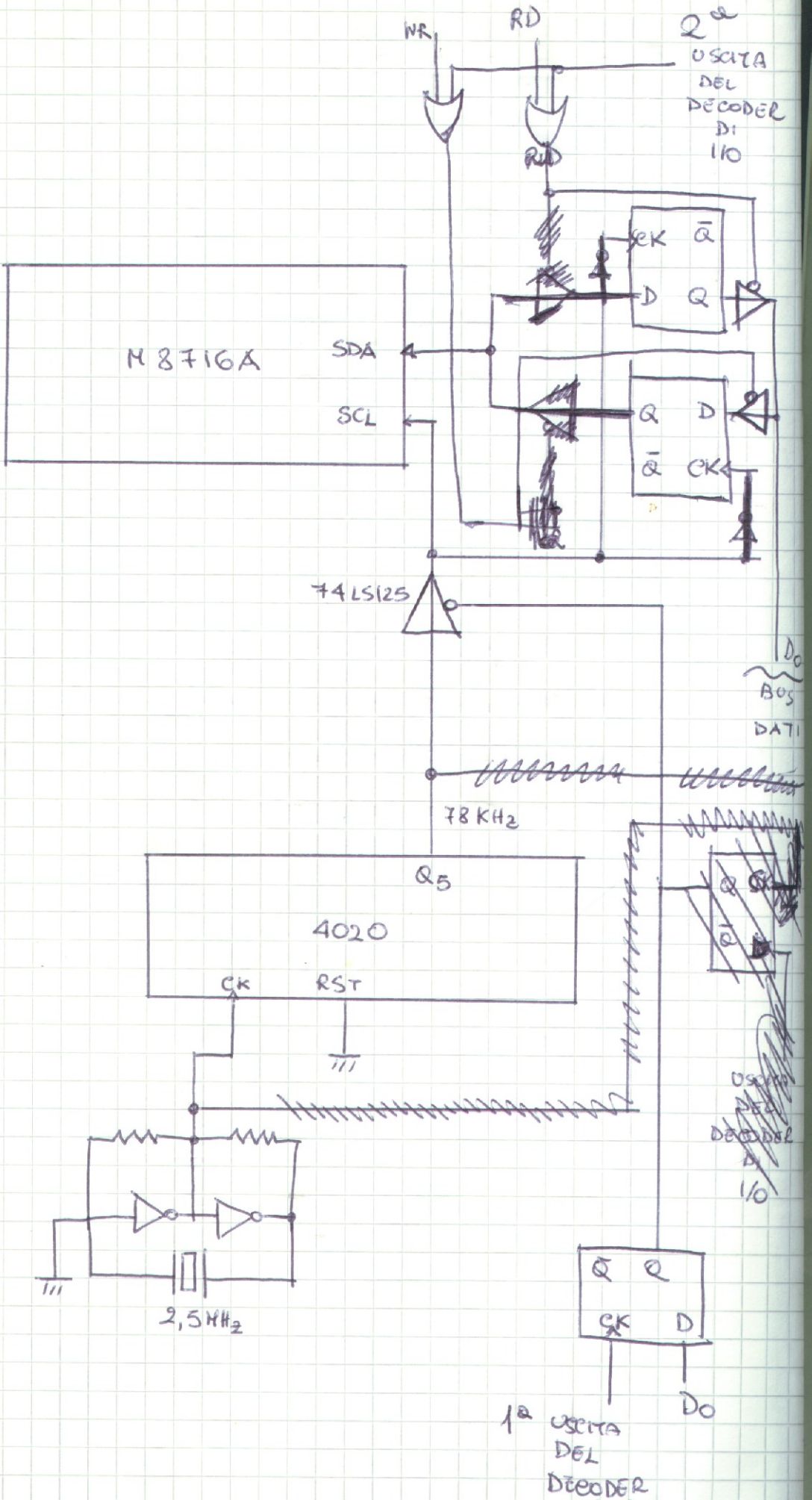


Fig. 6 - Typical application




WR
RD
2ª USCITA DEL DECODER DI I/O
RD
CK
Q̄
D
Q
M8716A
SDA
SCL
Q
D
Q̄
CK
74LS125
D0
BUS DATI
78 KHz
Q5
4020
CK
RST
Q
Q̄
2,5 MHz
Q̄
Q
CK
D
D0
1ª USCITA DEL DECODER

citato consentendo al segnale di clock di passare. Per interrompere l'invio del segnale basta scrivere di nuovo sul flip flop un uno. I dati vengono inviati una bit alla volta in scrittura memorizzandoli in un flip flop D di ingresso in modo che l'8716 abbia il tempo di leggerli alla sua frequenza. Il dato in uscita viene memorizzato sul secondo flip flop. Si tratti Poi three state isolano i flip flop dal bus dati.

CONTROLLER DEL VIDEO.

Un tubo a raggi catodici (CRT) è un trasduttore elettrooptico che trasforma un segnale elettrico in un segnale luminoso. Un cannone elettronico forma un fascio di elettroni che colpisce i fosfori dello schermo i quali emettono un raggio luminoso. Il CRT di è si differenzia da un normale televisore domestico per l'assenza dei circuiti in alta frequenza.

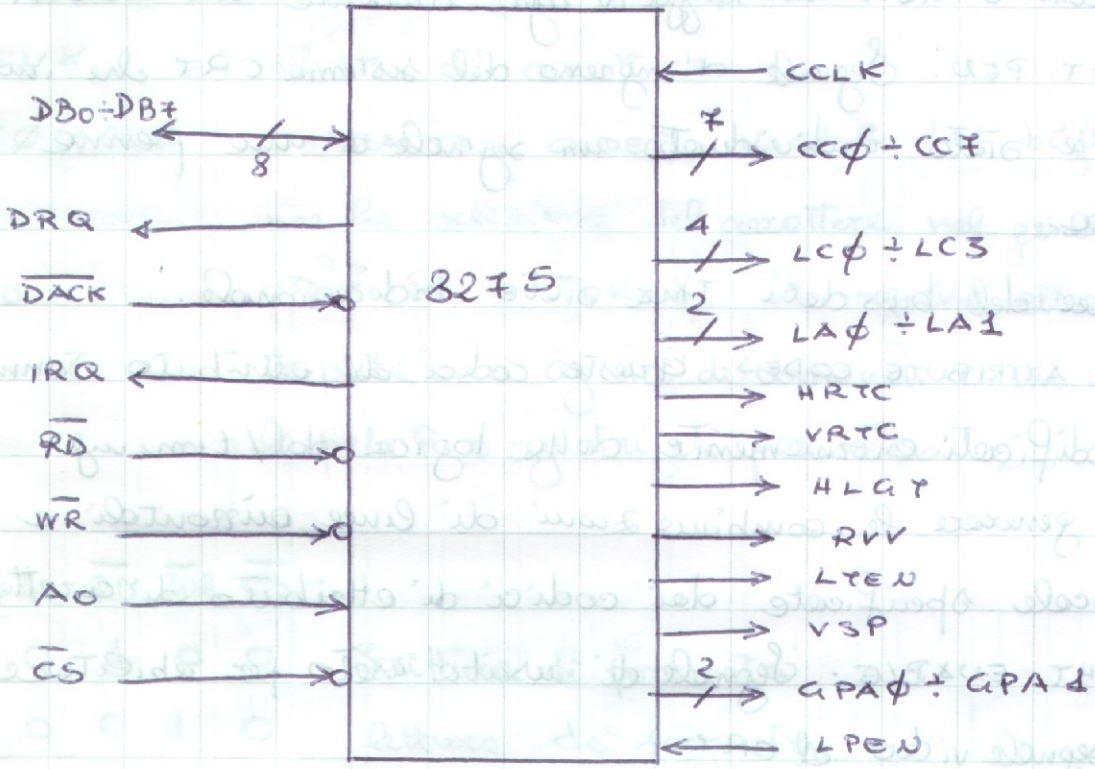
L'immagine sullo schermo è ottenuta da uno spezzettamento orizzontale e verticale del fascio elettronico con una velocità tale da evitare lo sfarfallio.

Ai fini dell'interfacciamento di un terminale video con un sistema basato su µP, basta generare un segnale video composito, cioè senza le componenti in alta frequenza. I caratteri arrivano dal µP in codice ASCII (o altro) e vengono convertiti in matrici di 5x7 punti da un generatore di caratteri residente su una ROM. I caratteri sono completamente rappresentati dopo 7 scansioni di riga

	col 1	col 2	3	4	5					
1e rige	□	■	■	■	□	0	1	1	1	0
2e rige	■	□	□	□	□	1	0	0	0	0
3e //	■	□	□	□	□	1	0	0	0	0
4e //	■	■	■	■	□	1	1	1	1	0
5e //	■	□	□	□	■	1	0	0	0	1
6e //	■	□	□	□	■	1	0	0	0	1
7e //	□	■	■	■	□	0	1	1	1	0

Rappresentazione del numero 6 su 7 righe e 5 colonne

Per generare il segnale di controllo del CRT occorre un controllore, ad esempio l'8275 dell'Intel



DESCRIZIONE DEI PIEDINI

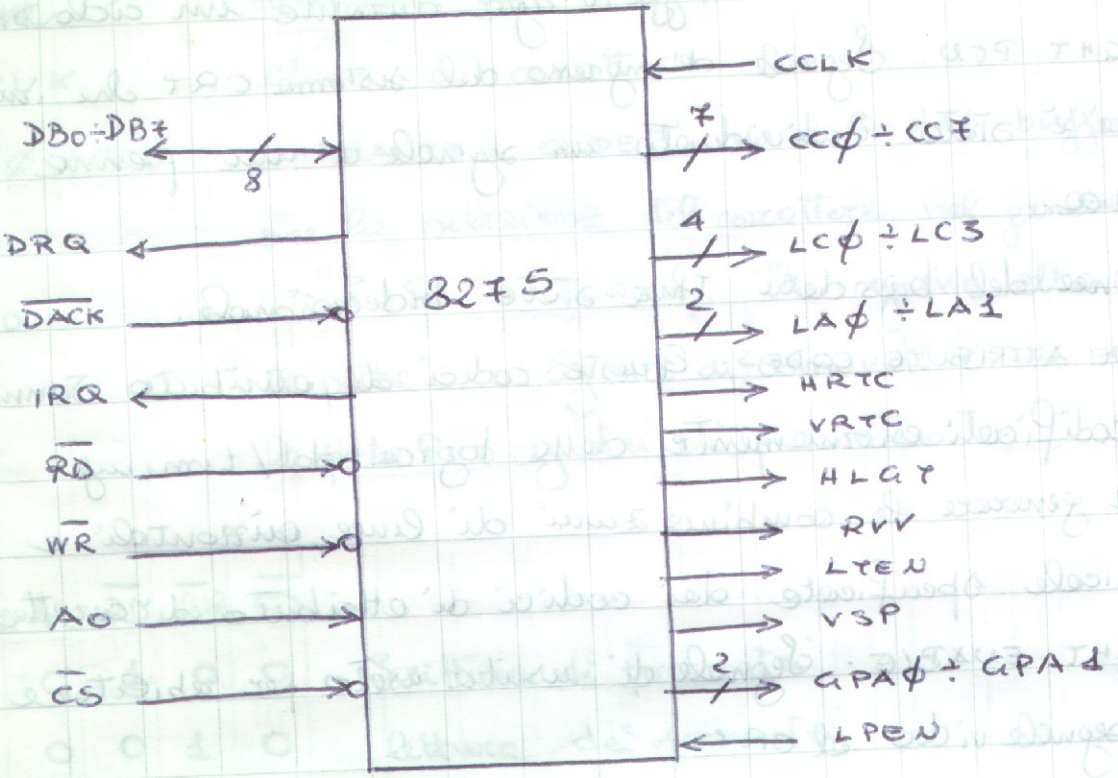
LC0÷LC3 LINE COUNT: contatore di linea. Uscita del contatore di linea che è usato per indirizzare il generatore di caratteri per le posizioni di linea sullo schermo.

DRQ Segnale di uscita indirizzabile ad un DMA per una richiesta di un ciclo di DMA

DACK Segnale di ingresso del controller DMA di acknowledge

HRTC HORIZONTAL RETRACE. Segnale di uscita che è attivo durante il programmato intervallo di retrace orizzontale. Durante questo periodo l'uscita VSP è alta e l'uscita LTEN è bassa.

VRTC VERTICAL RETRACE. Segnale di uscita attivo durante il programmato intervallo di retrace verticale. Durante questo periodo VSP è alto e LTEN è basso.



DESCRIZIONE DEI PIEDINI

LC0÷LC3 LINE COUNT: contatore di linea. Usata dal contatore di linea che è usato per indirizzare il generatore di caratteri per le posizioni di linea sullo schermo.

DRQ Segnale di uscita indirizzabile ad un DMA per una richiesta di un ciclo di DMA

DACK Segnale di ingresso del controller DMA di acknowledge

HRTC HORIZONTAL RETRACE. Segnale di uscita che è attivo durante il programmato intervallo di retrace orizzontale. Durante questo periodo l'uscita VSP è alta e l'uscita LTEN è bassa.

VRTC VERTICAL RETRACE. Segnale di uscita attivo durante il programmato intervallo di retrace verticale. Durante questo periodo VSP è alto e LTEN è basso.

$\overline{RD}$ Segnale di controllo per leggere registri

$\overline{WR}$ Segnale di controllo per scrivere comandi nei registri di controllo o dati nei buffer di riga durante un ciclo DMA

LPEN LIGHT PEN. Segnale d'ingresso del sistema CRT che indica che è stato individuato un segnale di una penna ottica.

DB0 ÷ DB7 Linee del bus dati three-state bidirezionale.

LA0 ÷ LA1 LINE ATTRIBUTE CODES. Questi codici di attributo danno decodificati esternamente dalla logica dot/timing per generare le combinazioni di linee orizzontali e verticale specificate dai codici di attributo di carattere

LTEN LIGHT ENABLE. Segnale di uscita usato per abilitare il segnale video del CRT.

VSP VIDEO SUPPRESSION. Segnale usato per cancellare il ~~video~~ segnale video del CRT. È attivo:

- durante gli intervalli di ritrace orizzontale e verticale
- alle linee di testa e di fondo di una riga se la sottolineatura è programmata per essere a zero numero otto o più grande.
- quando è ritirato un codice di fine riga o fine schermo
- a intervalli regolari per ottenere schermate lampeggianti.

GPA0 ÷ GPA1 GENERAL PURPOSE ATTRIBUTE CODES

Segnali di uscita abilitati dai codici di attributo di uso generale.

HLGT — HIGHLIGHT. Segnale di uscita usato per intensificare la videata ~~display~~ o in posizioni particolari sullo schermo

IRQ — Richiesta di interrupt

CCLK — Clock di carattere

CC0 ÷ CC6 — CHARACTER CODES. Uscite dei buffer di riga usate per la selezione del carattere nel generatore di caratteri.

A0 — INDIRIZZO DI PORTA. Un ingresso alto su A0 seleziona la porta C o registri di comando e un ingresso basso seleziona la porta P o registri di parametro.

A0	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	
0	1	0	0	Scrittura di parametri
0	0	1	0	Lettura dei parametri
1	1	0	0	Scrittura di comandi
1	0	1	0	Lettura dello stato
X	1	1	0	three-state
X	X	X	1	three-state

DESCRIZIONE FUNZIONALE

Un buffer a 8 bit bidirezionale three-state è usato per interfacciare l'8275 al bus dati del sistema. La logica di controllo accetta ingressi dal bus di controllo del sistema e genera segnali di controllo per il chip. Contiene i registri di comando, parametri e stato. Il contatore di caratteri è un contatore programmabile usato per determinare il numero di caratteri che devono visualizzati per riga e la lunghezza dell'intervallo di retrace

HLGT — HIGHLIGHT. Segnale di uscita usato per intensificare dei caratteri video in posizioni particolari sullo schermo

IRQ — Richiesta di interrupt

CCLK — clock di carattere

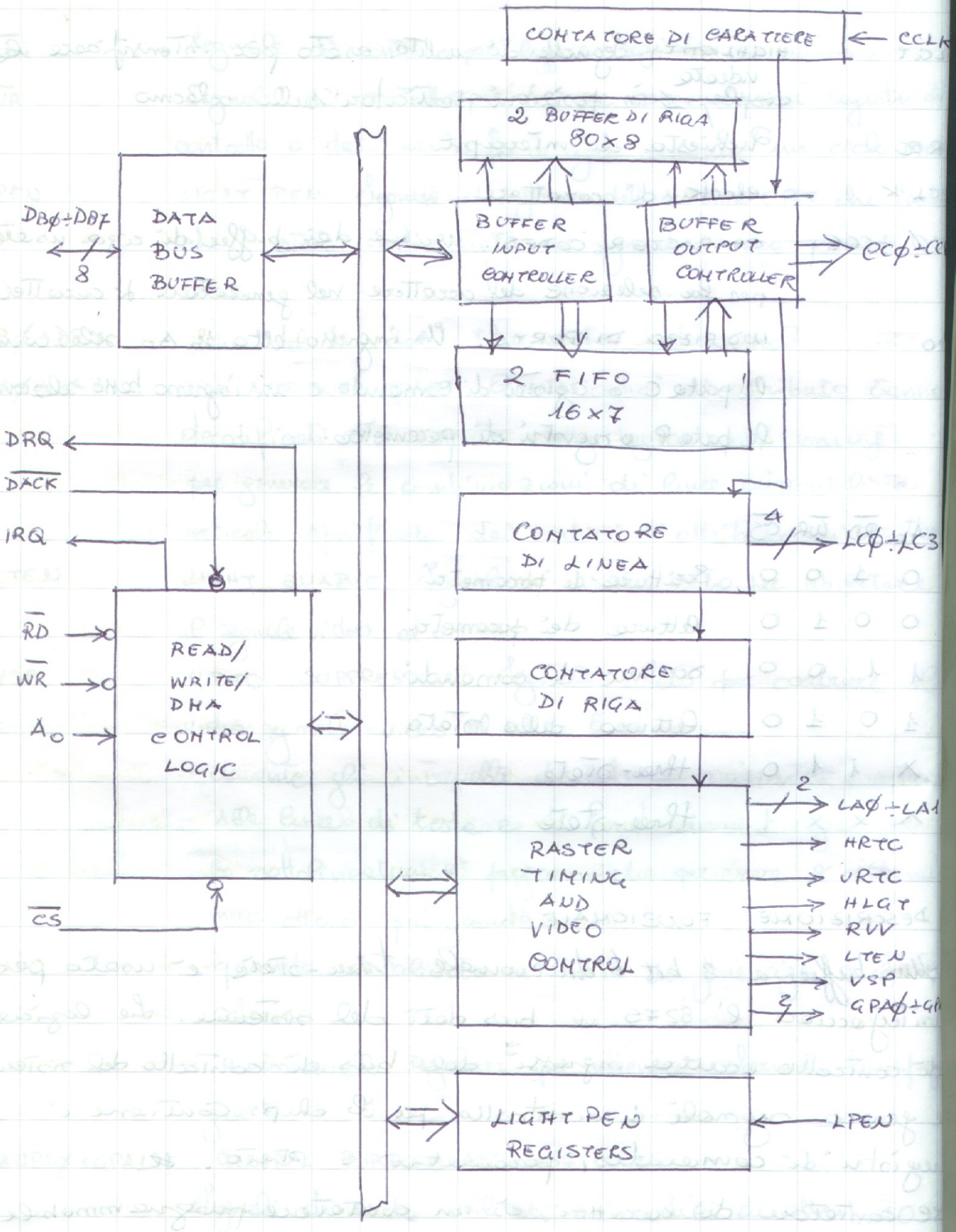
CC0 ÷ CC6 — CHARACTER CODES. Uscite dei buffer di riga usate per la selezione del carattere nel generatore di caratteri.

A0 — INDIRIZZO DI PORTA. Un ingresso alto su A0 seleziona la porta C o registri di comando e un ingresso basso seleziona la porta P o registri di parametro.

A0	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	
0	1	0	0	Scrittura di parametri
0	0	1	0	Lettura dei parametri
1	1	0	0	scrittura di comandi
1	0	1	0	Lettura dello stato
x	1	1	0	three-state
x	x	x	1	three-state

DESCRIZIONE FUNZIONALE

Un buffer a 8 bit bidirezionale three-state è usato per interfacciare l'8275 al bus dati del sistema. La logica di controllo accetta ingressi dal bus di controllo del sistema e genera segnali di controllo per il chip. Contiene i registri di comando, parametri e stato. Il contatore di caratteri è un contatore programmabile usato per determinare il numero di caratteri che devono visualizzati per riga e la lunghezza dell'intervallo di ritrace orizzontale. Esso è guidato dall'ingresso CCLK che


CONTATORE DI CARATTERE
CCLK
2 BUFFER DI RIGA 80×8
DB0÷DB7
8
DATA BUS BUFFER
BUFFER INPUT CONTROLLER
BUFFER OUTPUT CONTROLLER
CC0÷CC
2 FIFO 16×7
DRQ
DACK
IRQ
CONTATORE DI LINEA
4
LC0÷LC3
RD
WR
A0
READ/ WRITE/ DMA CONTROL LOGIC
CONTATORE DI RIGA
CS
2
LA0÷LA1
RASTER TIMING AND VIDEO CONTROL
HRTC
VRTC
HLGT
RVV
LTEN
VSP
2
GPA0÷GPA
LIGHT PEN REGISTERS
LPEN

dovrebbe derivare dal clock esterno di dot.
Il contatore di linea è un contatore programmabile usato per determinare il numero di linee orizzontali per riga di caratteri. Le sue uscite sono usate per indirizzare la Row generatrice di caratteri.
Il contatore di riga è un contatore programmabile usato per determinare il numero di ~~Linee orizzontali~~ righe di caratteri che visualizzare per ogni quadro e la lunghezza dell'intervallo di retrace verticale.
La circuiteria di raster timing controlla la temporizzazione delle uscite HRTC e VRTC. La circuiteria Video Control controlla la generazione di LA0 - LA1, HGLT, RVV, VSP, GPA0 - GPA1.
I Row Buffer sono due buffer di 80 caratteri. Essi vengono riempiti dalla memoria di sistema con i codici dei caratteri da visualizzare. Quando un buffer sta visualizzando una riga di caratteri, l'altro viene riempito con la successiva riga.
Vi sono due FIFO da 16 caratteri usati per fornire lunghezze extra ai buffer di riga nel Transparent Attribute mode.
L'8275 richiede al DMA di riempire il buffer di riga che non è usato per la visualizzazione. Esso visualizza le righe di caratteri una linea alla volta. Il numero di linee per riga di caratteri, la posizione di sottolineatura e l'ammontare delle linee superiore e inferiore sono programmabili. L'8275 fornisce codici di attributi visuali per causare effetti speciali o l'apparizione di simboli speciali sullo schermo senza usare il generatore di caratteri.
L'8275 controlla inoltre la temporizzazione di raster. Per questo genera i segnali HRTC e VRTC. La frequenza di

dovrebbe derivare dal clock esterno di dot.
Il contatore di linea è un contatore programmabile usato per determinare il numero di linee orizzontali per riga di caratteri. Le sue uscite sono usate per indirizzare la Row generatrice di caratteri.
Il contatore di riga è un contatore programmabile usato per determinare il numero di ~~Linee orizzontali~~ righe di caratteri che visualizzare per ogni quadro e la lunghezza dell'intervallo di retrace verticale.
La circuiteria di raster timing controlla la temporizzazione delle uscite HRTC e VRTC. La circuiteria Video Control controlla la generazione di LA0 - LA2, HGLT, RVV, VSP, GPA0 - GPA1.
I Row Buffer sono due buffer di 80 caratteri. Essi vengono riempiti dalla memoria di sistema con i codici dei caratteri da visualizzare. Quando un buffer sta visualizzando una riga di caratteri, l'altro viene riempito con la successiva linea.
Vi sono due FIFO da 16 caratteri usati per fornire lunghezze extra ai buffer di riga nel Transparent Attribute mode.
L'8275 richiede al DMA di riempire il buffer di riga che non è usato per la visualizzazione. Esso visualizza le righe di caratteri una linea alla volta. Il numero di linee per riga di caratteri, la posizione di sottolineatura e l'ordinamento delle linee superiore e inferiore sono programmabili. L'8275 fornisce codici di attributi visuali per ottenere effetti speciali o l'apparizione di simboli speciali sullo schermo senza usare il generatore di caratteri.
L'8275 controlla inoltre la temporizzazione di raster. Per questo genera i segnali HRTC e VRTC. La temporizzazione di questi segnali è programmabile. L'8275 può generare

dovrebbe derivare dal clock esterno di dot.
Il contatore di linea è un contatore programmabile usato per determinare il numero di linee orizzontali per riga di caratteri. Le sue uscite sono usate per indirizzare la Row generatrice di caratteri.
Il contatore di riga è un contatore programmabile usato per determinare il numero di ~~Linee orizzontali~~ righe di caratteri che visualizzare per ogni quadro e la lunghezza dell'intervallo di retrace verticale.
La circuiteria di raster timing controlla la temporizzazione delle uscite HRTC e VRTC. La circuiteria Video Control controlla la generazione di LA0 - LA2, HGLT, RVV, VSP, GPA0 - GPA1.
I Row Buffer sono due buffer di 80 caratteri. Essi vengono riempiti dalla memoria di sistema con i codici dei caratteri da visualizzare. Quando un buffer sta visualizzando una riga di caratteri, l'altro viene riempito con la successiva linea.
Vi sono due FIFO da 16 caratteri usati per fornire lunghezze extra ai buffer di riga nel Transparent Attribute mode.
L'8275 richiede al DMA di riempire il buffer di riga che non è usato per la visualizzazione. Esso visualizza le righe di caratteri una linea alla volta. Il numero di linee per riga di caratteri, la posizione di sottolineatura e l'ordinamento delle linee superiore e inferiore sono programmabili. L'8275 fornisce codici di attributi visuali per ottenere effetti speciali o l'apparizione di simboli speciali sullo schermo senza usare il generatore di caratteri.
L'8275 controlla inoltre la temporizzazione di raster. Per questo genera i segnali HRTC e VRTC. La temporizzazione di questi segnali è programmabile. L'8275 può generare

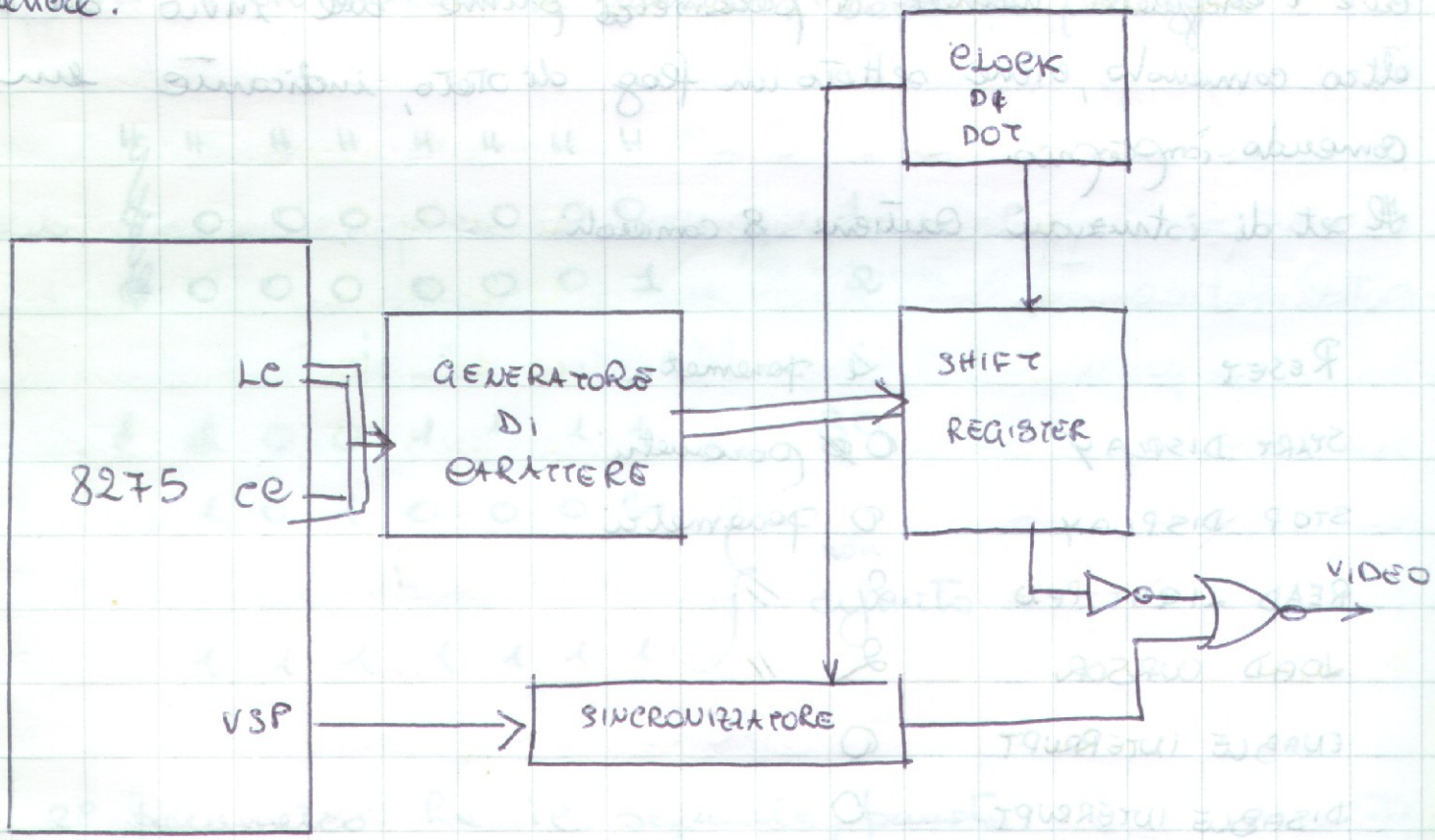
un cursore. La localizzazione del cursore e il suo formato sono progra
l'8275 può essere programmato per generare da 1 a 80 ~~righ~~ caratteri per riga e da 1 a 64 righe per schermata. Esso può essere inoltre programmato per emettere le righe alternativamente in modo che devono visualizzate solo le 1, le 3 e così via
Il numero di linee per riga di carattere è programmabile da 1 a 16.
L'uscita del contatore può essere programmata per essere in modo zero o uno. In modo zero l'uscita del contatore di linea è la stessa del numero di linea per cui si conta

LINE NUMBER	LINE COUNTER MODE 0
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
⋮	⋮
15	1 1 1 1

Nel modo 1 il line counter segue di uno il numero di linea

LINE NUMBER	LINE COUNTER MODE 1
0	1 1 1 1
1	0 0 0 0
2	0 0 0 1
3	0 0 1 0
4	0 0 1 1
⋮	⋮
14	1 1 0 1

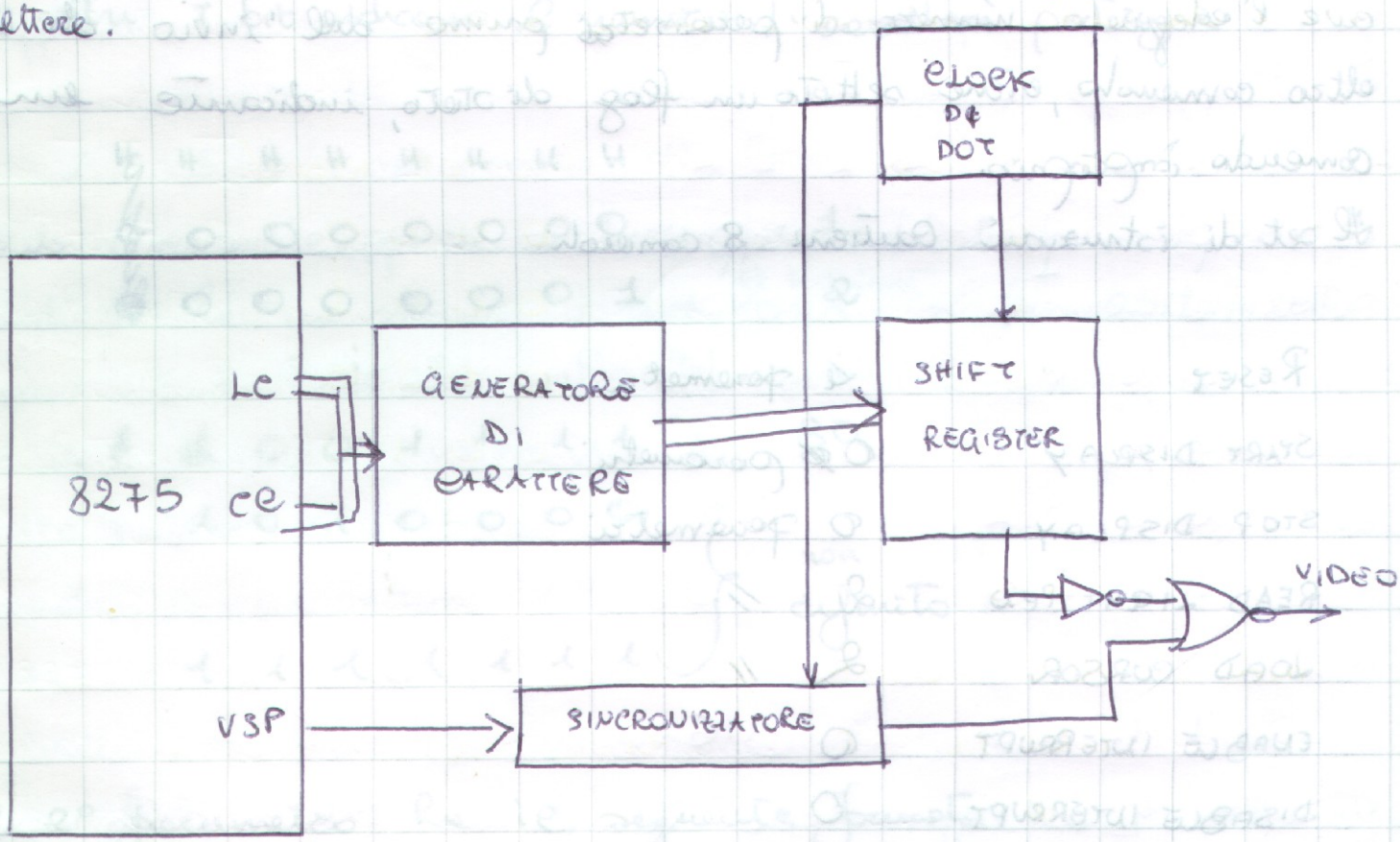
Il posizionamento delle sottolineature è programmabile dalla linea zero alla linea 15. ~~Se il numero di linee della sottolineatura~~ l'ampiezza del dot e l'ampiezza del carattere dipendono dalla circuiteria esterna di temporizzazione e controllo. Tale circuiteria dovrebbe essere disegnata per ottenere l'uscita parallela del generatore di caratteri e shiftarla in uscita seriolmente alla velocità richiesta dal CRT. L'ampiezza del dot è funzione della frequenza di clock. L'ampiezza del carattere è funzione dell'ampiezza del generatore di carattere.



PROGRAMMAZIONE DELL'8275

L'8275 ha due registri di programmazione, il registro di comando (CREG) e il registro dei parametri (PREG). Esso ha inoltre un registro di stato (SREG). Il registro di comando può essere solo scritto e il registro di stato può essere solo letto. Essi vengono indirizzati secondo la seguente tabella

Il posizionamento della sottolineatura è programmabile dalla linea zero alla linea 15. ~~Se il numero di linea della sottolineatura~~ l'empiezza del dot e l'empiezza del carattere dipendono dalla circuiteria esterna di temporizzazione e controllo. Tale circuiteria dovrebbe essere disegnata per accettare l'uscita parallela del generatore di caratteri e shiftarla in uscita seriolmente alla velocità richiesta dal CRT. L'empiezza del dot è funzione della frequenza di clock. L'empiezza del carattere è funzione dell'empiezza del generatore di caratteri.



PROGRAMMAZIONE DELL'8275

L'8275 ha due registri di programmazione, il registro di comando (CREG) e il registro dei parametri (PREG). Esso ha inoltre un registro di stato (SREG). Il registro di comando può essere solo scritto e il registro di stato può essere solo letto. Essi vengono indirizzati secondo la seguente tabella

A_0	OPERAZIONE	
0	lettura	PREG
0	scrittura	PREG
1	lettura	SREG
1	scrittura	CREG

L'8275 si aspetta di ricevere un comando e una sequenza da zero a quattro parametri, a seconda del comando. Se non vi è ove l'adeguato numero di parametri prima dell'invio di un altro comando, viene settato un flag di stato, indicante un comando improprio.
Il set di istruzioni contiene 8 comandi

RESET	4 parametri
START DISPLAY	0 parametri
STOP DISPLAY	0 parametri
READ LIGHT PEN	2 //
LOAD CURSOR	2 //
ENABLE INTERRUPT	0
DISABLE INTERRUPT	0
PRESET COUNTERS	0

RESET

Il comando reset si ottiene, scrivendo, con $A_0 = 1$, nel registro CREG il byte 00H

I 4 parametri sono inviati al PREG ($A_0 = 0$), il 1° parametro ha il seguente formato

S H H H H H H H

dove

S = { 0 righe normali
    1 righe distanziate

gli altri 7 bit indicano il numero di caratteri per riga

H H H H H H H H

0 0 0 0 0 0 0 1

0 0 0 0 0 0 1 0

⋮

1 0 0 1 1 1 1 0 → 80

1 0 1 0 0 0 0 0
⋮
1 1 1 1 1 1 1 1 } non definito

Il 2° parametro ha il seguente formato

V V R R R R R R

I primi due bit codificano il numero di righe da conteggiare per il retrace verticale

V V	n. righe
0 0	1
0 1	2
1 0	3

Gli altri 6 parametri rappresentano il numero di righe per quadr

R	R	R	R	R	R	
0	0	0	0	0	0	1
0	0	0	0	0	1	2
			⋮			⋮
1	1	1	1	1	1	64

Il 3° byte ha il seguente formato

0	0	0	0	L	L	L	L

dove i primi quattro bit rappresentano il posizionamento delle sottolineature

				numero della riga della sottolineatura
0	0	0	0	1
0	0	0	1	2
0	0	1	0	3
		⋮		
1	1	1	1	16

Gli altri 4 bit rappresentano il numero di linee delle righe di carattere

C	C	C	C	
0	0	0	0	1
0	0	0	1	2
		⋮		
1	1	1	1	16

Il formato del 4° byte è

M F C C Z Z Z Z

dove M indica la modalità del contatore 1 linea

M	
0	modo zero
1	modo uno

F rappresenta la modalità dell'attributo di campo (non ci interessa)

CC il formato del cursore (non ci interessa)

Z Z Z Z il numero di ~~linee~~ caratteri da contare per il ritraccia orizzontale (da 2 a 32).

Quando si ha un comando di reset, le richieste DMA vengono fermate, le interruzioni dell'8275 sono disabilitate, e VSP è usato per annerire lo schermo, HRTC e VRTC sono attivi. ## La temporizzazione di HRTC e VRTC sono regolari all'accensione. Appena si accendono i parametri, la composizione dello schermo è definita.

START DISPLAY COMMAND

La struttura di tale byte di comando è

0 0 1 S S S B B

I tre bit SSS codificano il ~~due~~ numero di caratteri tra due richieste DMA (~~da~~

S S S		S S S	
0 0 0	0	1 0 1	39
0 0 1	4	1 1 0	47
0 1 0	13	1 1 1	55
0 1 1	23		

I due ultimi bit permettono di determinare il numero di cicli di clock per impulso (burst) DMA (da 1 a 8)

B	B	
0	0	1
0	1	2
1	0	4
1	1	8

Con questo comando vengono abilitate le interruzioni dell' 8275, cominciano le richieste DMA, il video è abilitato, sono settati i flag Interrupt Enable e Video Enable.

STOP DISPLAY COMMAND

Code comando ha la seguente struttura

0 1 0 0 0 0 0 0

Disabilita il video, le interruzioni permangono abilitate, HRTC e VRTC continuano, il flag VE è resettato ed occorre un comando "Start Display" per riabilitare il display.

READ LIGHT PEN

È un comando necessario se il sistema è dotato di penna elettronica e non ci interessa.

LOAD CURSOR POSITION

Ha il formato 1 0 0 0 0 0 0 0, i successivi sono la posizione del carattere nella riga e il numero della riga. L' 8275 è costretto a posizionare questi due byte nei registri che contengono la posizione del cursore.

ENABLE INTERRUPT COMMAND

1 0 1 0 0 0 0 0

clock per impulso (burst) DMA (da 1 a 8)

B	B	
0	0	1
0	1	2
1	0	4
1	1	8

Con questo comando vengono abilitate le interruzioni dell'8275, cominciano le richieste DMA, il video è abilitato, sono settati i flag Interrupt Enable e Video Enable.

STOP DISPLAY COMMAND

Tale comando ha la seguente struttura

0 1 0 0 0 0 0 0

Disabilita il video, le interruzioni permangono abilitate, HRTC e VRTC continuano, il flag VE è resettato ed occorre un comando "Start Display" per riabilitare il display.

READ LIGHT PEN

È un comando necessario se il sistema è dotato di penna elettronica e non ci interessa.

LOAD CURSOR POSITION

Tale comando 1 0 0 0 0 0 0 0, i successivi due byte sono la posizione del carattere nella riga e il numero della riga. L'8275 è costretto a posizionare questi due byte nei registri che contengono la posizione del cursore.

ENABLE INTERRUPT COMMAND

1 0 1 0 0 0 0 0

Il flag interrupt enable status è settato e le

interruzioni vengono abilitate.

DESABLE INTERRUPT COMMAND

1 1 0 0 0 0 0 0

Le interruzioni vengono disabilitate e il flag IE viene resettato

PRESET COUNTERS COMMAND.

1 1 1 0 0 0 0 0

I contatori di temporizzazione vengono presettati, corrispondendo ad una posizione ~~del~~ nell'angolo in alto a sinistra dello schermo. I contatori rimangono in questo stato finché ogni altro comando deve dato.

REGISTRO DEI FLAG

Ø IE IR LP IC VE DU FO

- IE - (Interrupt Enable). Settato o resettato mediante comando. Esso abilita l'interruzione del retrace verticale.
IR - (Interrupt Request). Questo flag è settato all'inizio della visualizzazione dell'ultima riga della schermata se il flag IE è settato. Viene resettato dopo un'operazione di lettura dello stato.
LP - Questo flag viene settato quando il light pen input viene attivato e i registri della penna ottica

BASILARI SUI CRT

L'immagine mostrata sul CRT è costruita generando una serie di linee (raster) ~~lungo~~ attraverso il video. Usualmente, il fascio elettronico parte nell'angolo in alto a sinistra del display e simultaneamente si muove da sinistra a destra e dall'alto in basso, tracciando

interruzioni vengono abilitate.

DESABLE INTERRUPT COMMAND

1 1 0 0 0 0 0 0

Le interruzioni vengono disabilitate e il flag IE viene resettato

PRESET COUNTERS COMMAND.

1 1 1 0 0 0 0 0

I contatori di Temporizzazione vengono presettati, corrispondendo ad una posizione ~~del~~ nell'angolo in alto a sinistra dello schermo. I contatori rimangono in questo stato finché ogni altro comando viene dato.

REGISTRO DEI FLAG

Ø IE IR LP IC VE DU FO

IE - (Interrupt Enable). Settato o resettato mediante comando. Esso abilita l'interruzione del retrace verticale.

IR - (Interrupt Request). Questo flag è settato all'inizio della visualizzazione dell'ultima riga della schermata se il flag IE è settato. Viene resettato dopo un'operazione di lettura dello stato.

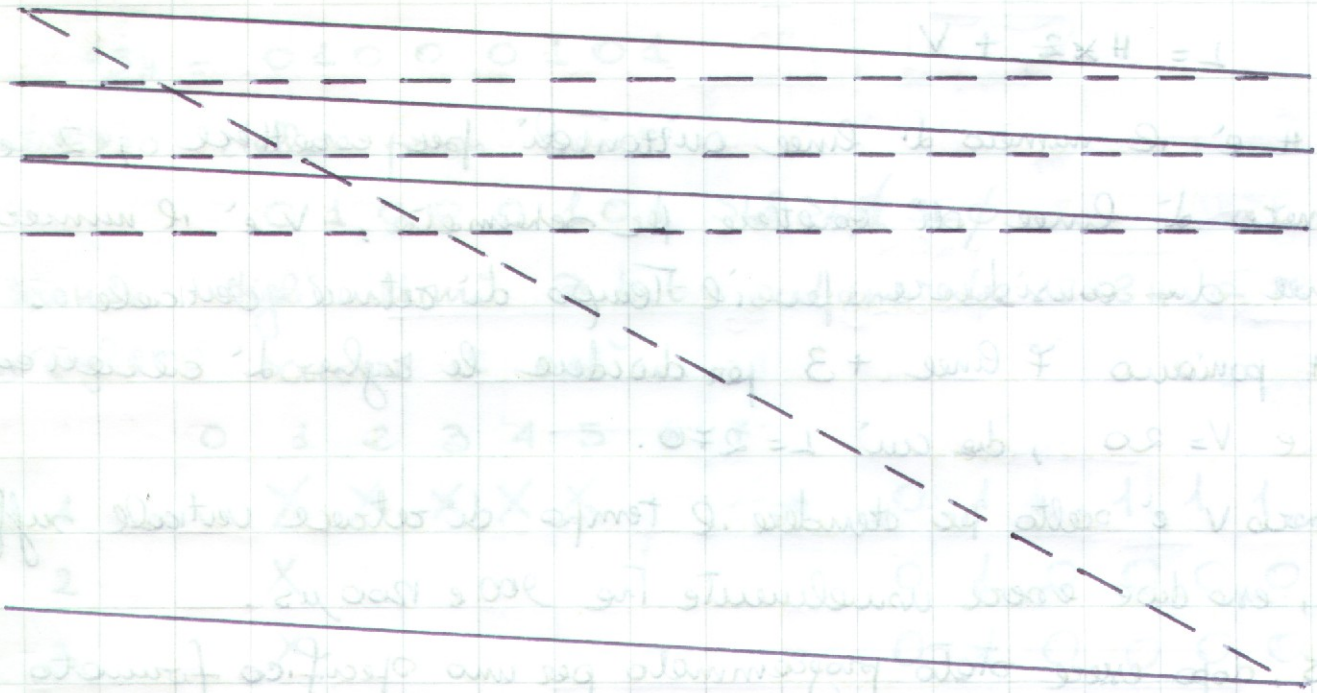
LP - Questo flag viene settato quando il light pen input viene attivato e i registri della penna ottica

BASILARI SUI CRT

L'immagine mostrata sul CRT è costruita generando una serie di linee (raster) ~~lungo~~ attraverso il video. Usualmente, il fascio elettronico parte nell'angolo in alto a sinistra del display e simultaneamente si muove da sinistra a destra e dall'alto in basso ponendo una serie di linee a zig-zag sullo schermo. Due accenni sul pen-

denti, operanti simultaneamente controllano il movimento orizzontale e verticale del fascio. Mentre il fascio elettronico si muove attraverso lo schermo, un terzo circuito controlla la corrente che fluisce nel fascio. Variando la corrente nel fascio, l'immagine nel CRT può essere resa brillante o scura come l'utente desidera.
Quando il fascio raggiunge la fine di una linea, esso è riportato all'inizio della linea successiva ad una velocità che è più veloce di quella usata per generare la linea. Si chiama azione di retrace. Durante il periodo di retrace il fascio è spento.
Mentre il fascio muove lungo lo schermo orizzontalmente, esso si muove inoltre verso il basso. A causa di ciò, ogni linea successiva parte leggermente al di sotto della precedente. Quando il fascio finalmente raggiunge l'angolo inferiore destro dello schermo, esso ritorna verticalmente indietro all'angolo sinistro in alto. Il tempo che occorre affinché il fascio muova dall'alto in basso e indietro di nuovo è il tempo di frame. Un valore tipico per la frequenza di linea (orizzontale) è di 15,75 kHz (63,5 μs) e 60 Hz per la frequenza verticale di frame (16,67 ms).
La prima cosa che un controller di CRT deve fare è generare impulsi che definiscono la temporizzazione orizzontale e verticale. Questo viene fatto dividendo una sorgente di clock di riferimento per qualche numero appropriato.
I caratteri che vengono visti sullo schermo sono formati da una serie di punti che devono shiftati fuori dal controller mentre il fascio elettronico muove lungo lo schermo. I circuiti che creano questa temporizzazione sono chiamati dot clock e character clock. Il clock di carattere è usato al

tale e verticale del fascio. Mentre il fascio elettronico si muove attraverso lo schermo, un terzo circuito controlla la corrente che fluisce nel fascio. Variando la corrente nel fascio, l'immagine nel CRT può essere resa brillante o scura come l'utente desidera. Quando il fascio raggiunge la fine di una linea, esso è riportato all'inizio della linea successiva ad una velocità che è più veloce di quella usata per generare la linea. Si chiama azione di retrace. Durante il periodo di retrace il fascio è spento.
Mentre il fascio muove lungo lo schermo orizzontalmente, esso si muove inoltre verso il basso. A causa di ciò, ogni linea successiva parte leggermente al di sotto della precedente. Quando il fascio finalmente raggiunge l'angolo inferiore destro dello schermo, esso ritorna verticalmente indietro all'angolo sinistro in alto. Il tempo che occorre affinché il fascio muova dall'alto in basso e indietro di nuovo è il tempo di frame. Un valore tipico per la frequenza di linea (orizzontale) è di 15,75 kHz (63,5 μs) e 60 Hz per la frequenza verticale di frame (16,67 ms).
La prima cosa che un controller di CRT deve fare è generare impulsi che definiscono la temporizzazione orizzontale e verticale. Questo viene fatto dividendo una sorgente di clock di riferimento per qualche numero appropriato.
I caratteri che vengono visti sullo schermo sono formati da una serie di punti che devono shiftati fuori dal controller mentre il fascio elettronico muove lungo lo schermo. I circuiti che creano questa temporizzazione sono chiamati dot clock e character clock. Il clock di carattere è uguale al clock di dot diviso per il numero di linee usate per for-



mezzo un carattere lungo l'asse orizzontale e il clock di dot è calcolato mediante la seguente equazione

$$\text{DOT CLOCK(Hz)} = (N+R) \times D \times L \times F$$

dove

N è il numero di caratteri per linea

R è il numero di caratteri da contare per il ritrace orizzontale

D è il numero di dot per carattere

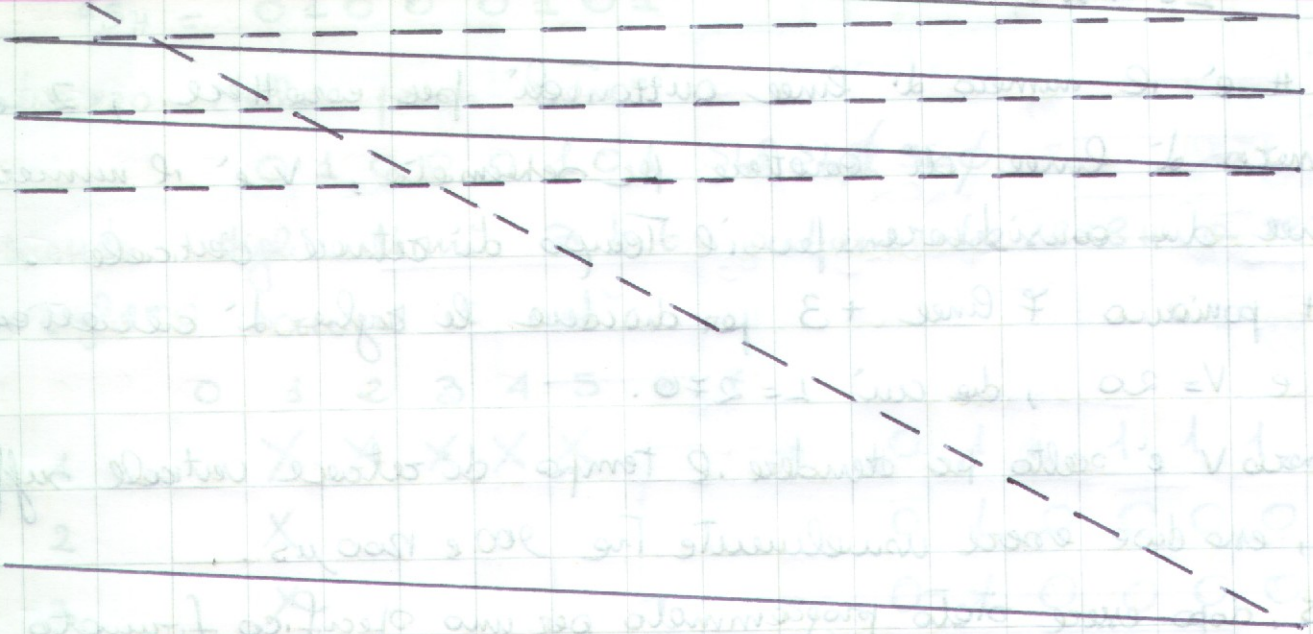
L è il numero di linee per schermata

F è la frequenza verticale.

Se ~~xxxxx~~ poniamo 80 caratteri per linea, 20 caratteri per tener conto del tempo di ritrace orizzontale, D = 7, L = 270 e F = 60 Hz

$$\text{DOT CLOCK} = (80+20) \times 7 \times 270 \times 60 = 11{,}34 \text{ MHz}$$

Il numero R è stato scelto come numero intermedio tra 2 e il max 32 permesso dall'8275. Sono previsti mediamente 5 x 7 dot per ogni carattere più 2 dot per tra 2 caratteri da cui D = 5 + 2 = 7. Il numero di linee per schermata può



mezzo un carattere lungo l'asse orizzontale e il clock di dot è calcolato mediante la seguente equazione

$$\text{DOT CLOCK(Hz)} = (N+R) \times D \times L \times F$$

dove

N è il numero di caratteri per linea

R è il numero di caratteri da contare per il ritrace orizzontale

D è il numero di dot per carattere

L è il numero di linee per schermata

F è la frequenza verticale.

Se ~~xxxxx~~ poniamo 80 caratteri per linea, 20 caratteri per tener conto del tempo di ritrace orizzontale, $D = 7$, $L = 270$ e $F = 60$ Hz

$$\text{DOT CLOCK} = (80+20) \times 7 \times 270 \times 60 = 11{,}34 \text{ MHz}$$

Il numero R è stato scelto come numero intermedio tra 2 e il max 32 permesso dall'8275. Sono previsti mediamente 5 x 7 dot per ogni carattere più 2 dot per tra 2 caratteri da cui $D = 5 + 2 = 7$. Il numero di linee per schermata può essere determinato con la seguente equazione

$$L = H \times Z + V$$

dove H è il numero di linee orizzontali per carattere, Z è il numero di linee di carattere per schermata, V è il numero di linee da considerare per il Tempo di ritrace verticale.
Per H poniamo 7 linee +3 per dividere le righe di caratteri, Z=25 e V=20, da cui L=270.
Il numero V è scelto per ottenere il tempo di ritrace verticale sufficiente, esso deve essere usualmente tra 900 e 1200 μs.
L'8275, dopo essere stato programmato per uno specifico formato di schermo, genera una serie di segnali di richiesta DMA, che danno come risultato il trasferimento di una ~~fila~~ riga di caratteri dalla memoria al buffer di riga dell'8275. L'8275 presenta i codici di carattere ad una ROM esterna generatrice di caratteri usando le linee di uscita CC0 ÷ CC6. La logica di Temporizzazione del dot è poi usata per trasferire il dato parallelo in uscita dalla ROM serialmente all'ingresso video del CRT. Le linee di carattere sono visualizzate sul CRT una linea alla volta. Le uscite LC0 ÷ LC3 sono applicate alla ROM per realizzare la selezione della linea.
Il generatore di caratteri è la EPROM 2716. Le tre linee LC0 ÷ LC3 sono connesse alle tre linee del indirizzo meno significative della EPROM e CC0 ÷ CC6 sono collegati alle linee $A_3 \div A_9$. L'uscita del generatore di caratteri è ~~collegata~~ caricata in uno shift register e l'uscita seriale di questo registro costituisce l'output video del Terminale. Supponiamo di voler visualizzare la lettera E. Il suo codice ASCII è 45H. Allora poremo 45H in ingresso alle linee $A_9 \div A_3$. Di conseguenza le linee da zero a sette per formare il carattere.

$L = H \times Z + V$

dove H è il numero di linee orizzontali per carattere, Z è il numero di linee di carattere per schermata, V è il numero di linee da considerare per il Tempo di ritrace verticale.
Per H poniamo 7 linee + 3 per dividere le righe di caratteri, $Z = 25$ e $V = 20$, da cui $L = 270$.
Il numero V è scelto per estendere il tempo di ritrace verticale sufficiente, esso deve essere usualmente tra 900 e 1200 μs.
L'8275, dopo essere stato programmato per uno specifico formato di schermo, genera una serie di segnali di richiesta DMA, che danno come risultato il trasferimento di una ~~fila~~ riga di caratteri dalla memoria al buffer di riga dell'8275. L'8275 presenta i codici di carattere ad una ROM esterna generatrice di caratteri usando le linee di uscita $cc\phi \div cc6$. La logica di Temporizzazione dei dot è poi usata per trasferire il dato parallelo in uscita dalla ROM serialmente all'ingresso video del CRT. Le linee di carattere sono visualizzate sul CRT una linea alla volta. Le uscite $lc\phi \div lc3$ sono applicate alla ROM per realizzare la selezione delle linee.
Il generatore di caratteri è la EPROM 2716. Le tre linee $lc\phi \div lc3$ sono connesse alle tre linee del indirizzo meno significative della EPROM e $cc\phi \div cc6$ sono collegati alle linee $A_3 \div A_9$. L'uscita del generatore di caratteri è ~~collegata~~ caricata in uno shift register e l'uscita seriale di questo registro costituisce l'output video del Terminale. Supponiamo di voler visualizzare la lettera E. Il suo codice ASCII è 45H. Allora porremo 45H in ingresso alle linee $A_9 \div A_3$. Si considerano così le linee da zero a sette per formare il carattere.

$45_H = 01000101$

L'indirizzo della prom sarà

0 1 0 0 0 1 0 1 SL2 SL1 SL0

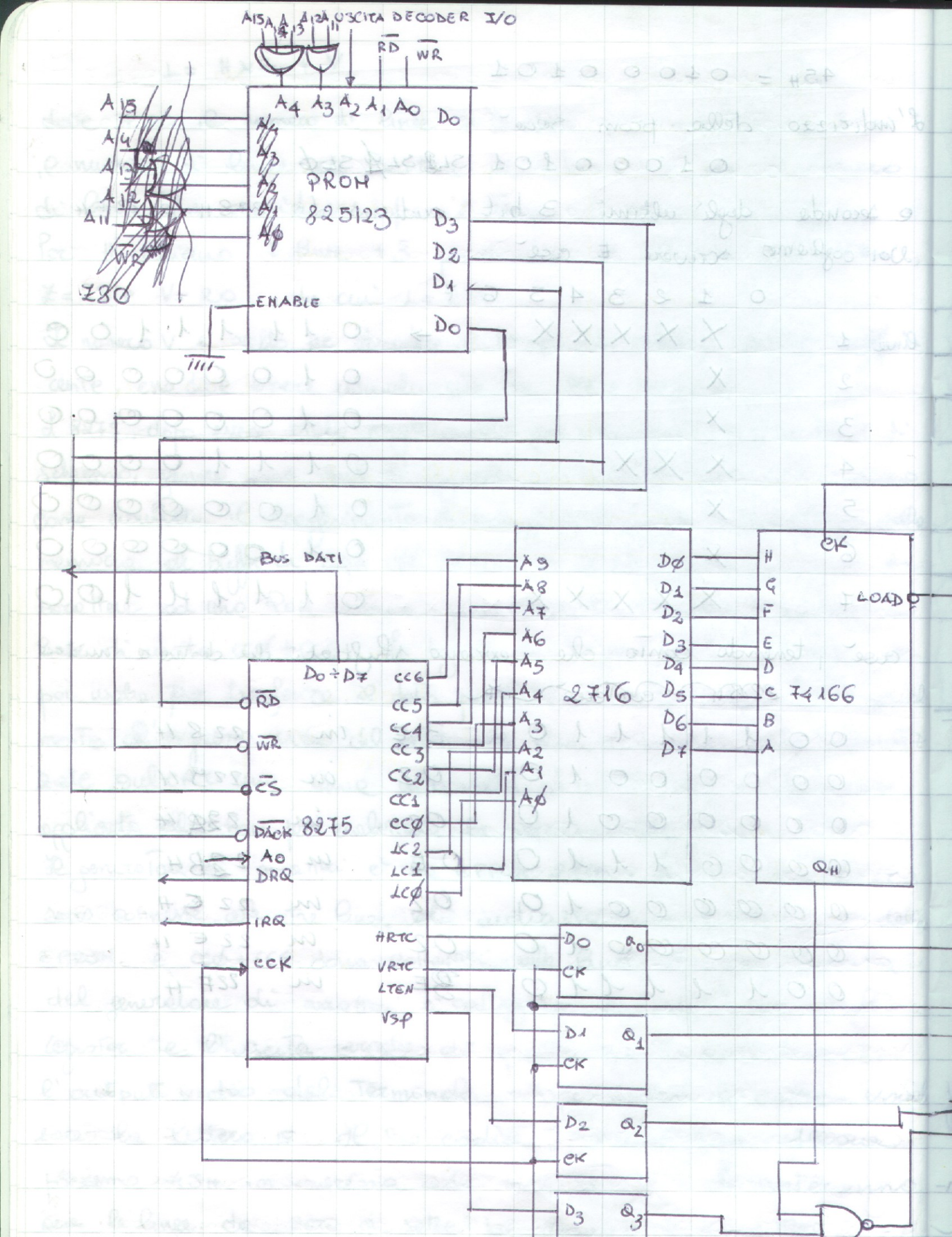
e secondo degli ultimi 3 bit andremo da 228 H a 22F H

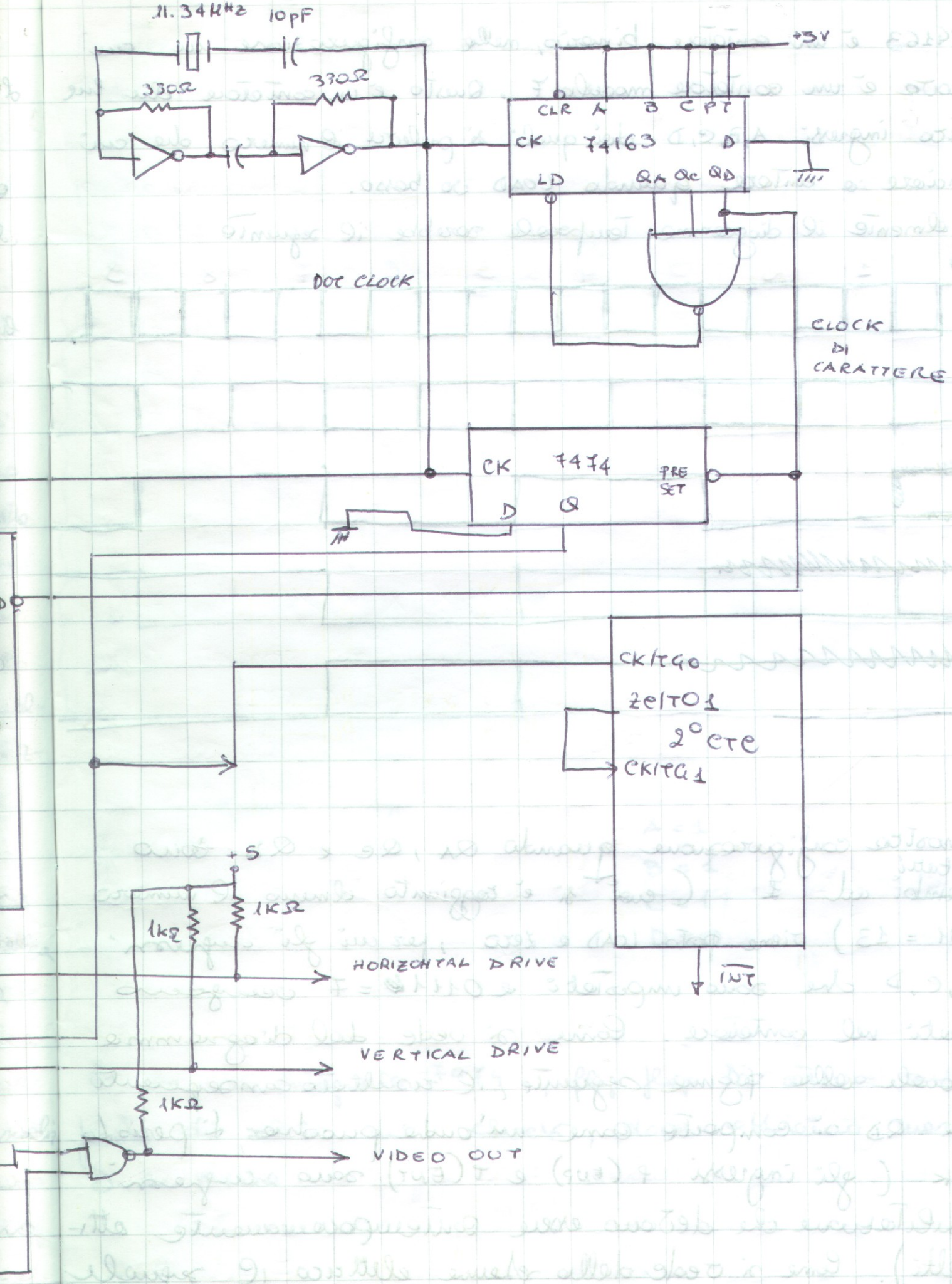
Noi vogliamo scrivere E così

	0	1	2	3	4	5	6	7		
Linea 1		X	X	X	X	X			→	0 1 1 1 1 1 0 0
2		X								0 1 0 0 0 0 0 0
3		X								0 1 0 0 0 0 0 0
4		X	X	X						0 1 1 1 0 0 0 0
5		X								0 1 0 0 0 0 0 0
6		X								0 1 0 0 0 0 0 0
7		X	X	X	X	X				0 1 1 1 1 1 0 0

così, tenendo conto che verranno shiftati da destra a sinistra

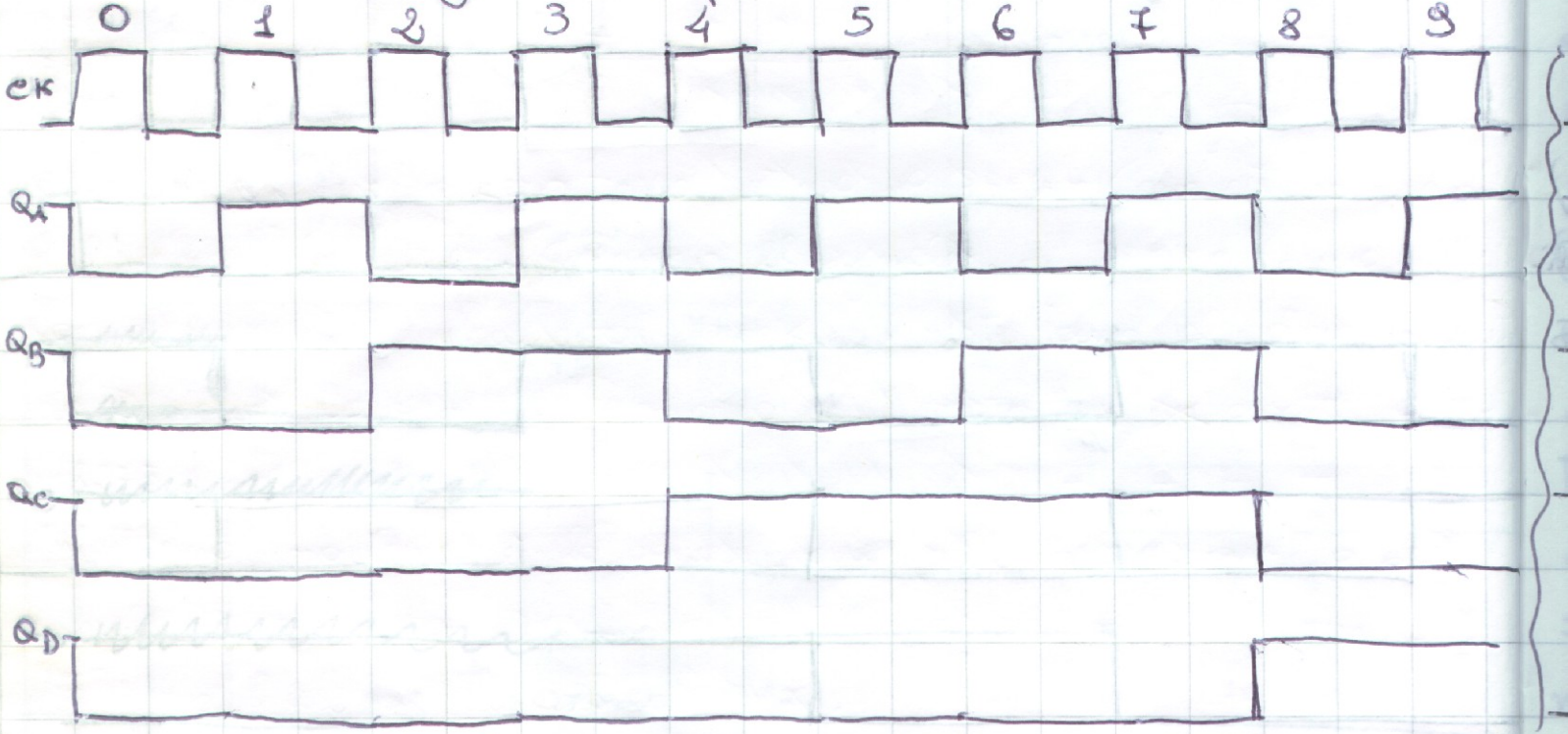
228 H	antenna		
0 0 1 1	1 1 1 0	3E	in 228 H
0 0 0 0	0 0 1 0	02	in 229 H
0 0 0 0	0 0 1 0	02	in 22A H
0 0 0 0	1 1 1 0	0E	in 22B H
0 0 0 0	0 0 1 0	02	in 22C H
0 0 0 0	0 0 1 0	02	in 22E H
0 0 1 1	1 1 1 0	3E	in 22F H


A15 A14 A13 A12 A11 USCITA DECODER I/O
RD
WR
A15
A14
A13
A12
A11
WR
Z80
A4 A3 A2 A1 A0
D0
PROM
82S123
D3
D2
D1
D0
ENABLE
BUS DATI
D0 ÷ D7
RD
WR
CS
DACK
A0
DRQ
IRQ
CCK
CC6
CC5
CC4
CC3
CC2
CC1
CC0
LC2
LC1
LC0
8275
A9
A8
A7
A6
A5
A4
A3
A2
A1
A0
2716
D0
D1
D2
D3
D4
D5
D6
D7
CK
H
G
F
E
D
C
B
A
74166
LOAD
QH
HRTC
VRTC
LTEN
VSP
D0
Q0
CK
D1
Q1
CK
D2
Q2
CK
D3
Q3


11.34 MHz
10pF
330Ω
330Ω
+5V
CLR
A
B
C
PT
CK
74163
D
LD
QA
QC
QD
DOT CLOCK
CLOCK
DI
CARATTERE
CK
7474
PRE
SET
D
Q
CK/TGO
ze/TO1
2° CTC
CK/TG1
+5
1kΩ
1KΩ
HORIZONTAL DRIVE
INT
VERTICAL DRIVE
1KΩ
VIDEO OUT

Il 74163 è un contatore binario, nella configurazione in cui si trova è un contatore modello 7. Questo è un contatore che ha quattro ingressi A, B, C, D dai quali si preleva il numero che da cui cominciare a contare quando LOAD va basso.

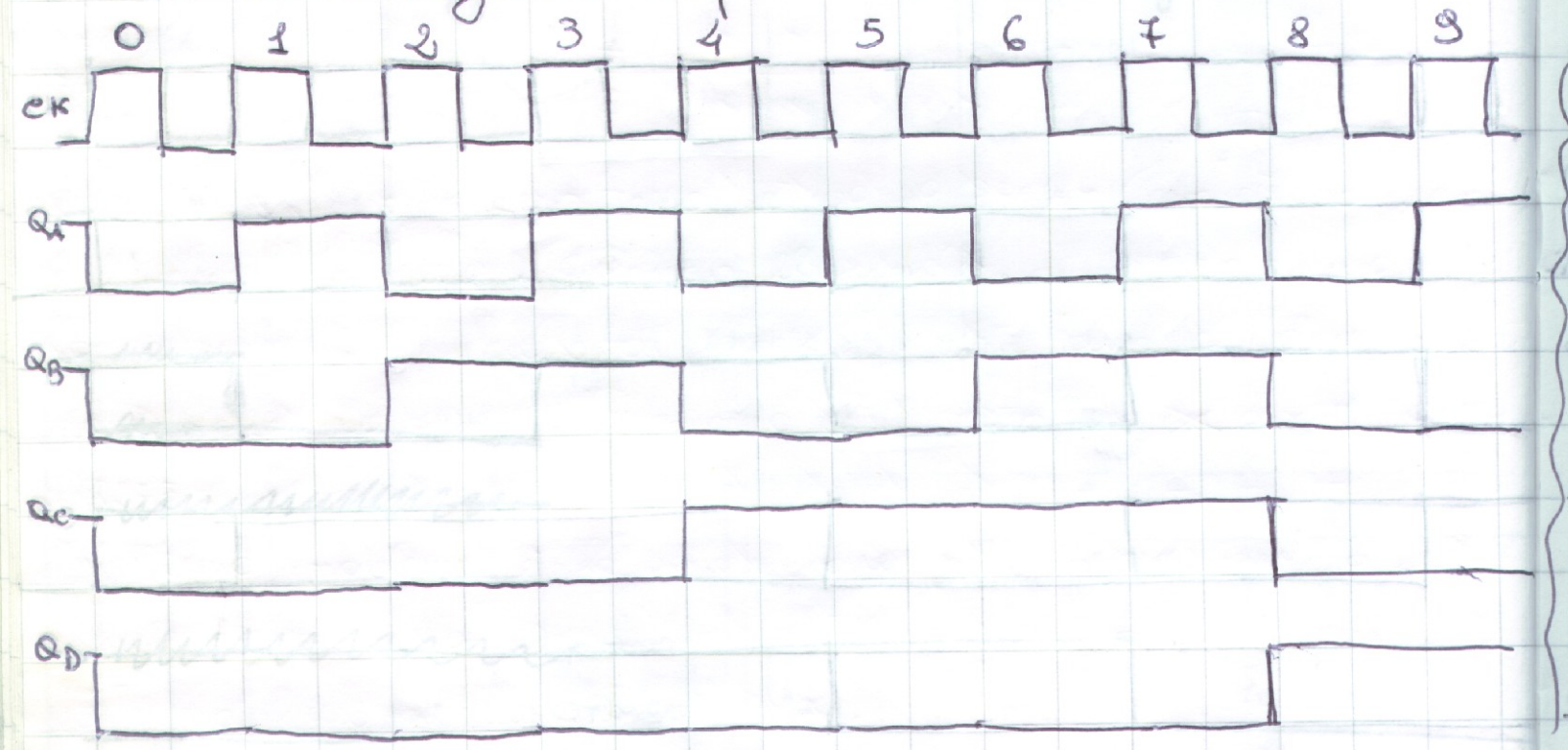
Idealmente il diagramma temporale sarebbe il seguente



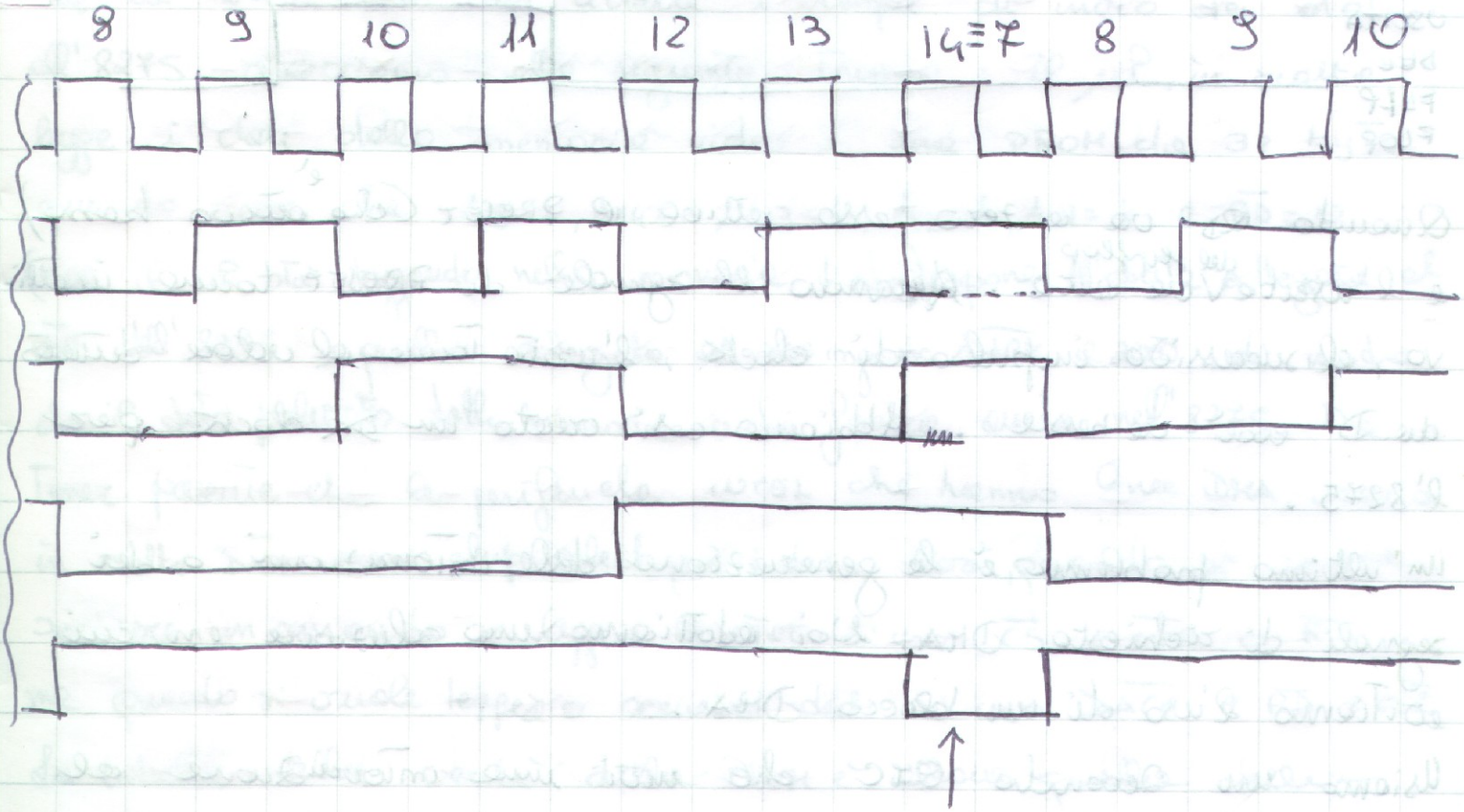
Nella nostra configurazione quando Q_A, Q_C e Q_D sono tutti entrambi ad 1 (cioè si è raggiunto almeno il numero $1101 = 13$) viene posto LOAD a zero; per cui gli ingressi A, B, C, D che sono impostati a $0111 = 7$ vengono caricati nel contatore. Come si vede dal diagramma temporale alla pagina seguente, il risultato conseguente è che Q_D si comporta come un'onda quadra di periodo $7 \cdot T_{CK}$ (gli ingressi P (ENP) e T (ENT) sono ingressi di abilitazione che devono essere contemporaneamente attivi (alti). Come si vede dallo stesso circuito (e segnali

Il 74163 è un contatore binario, nella configurazione in cui si trova è un contatore modello 7. Questo è un contatore che ha quattro ingressi A, B, C, D dai quali si preleva il numero da cui cominciare a contare quando LOAD va basso.

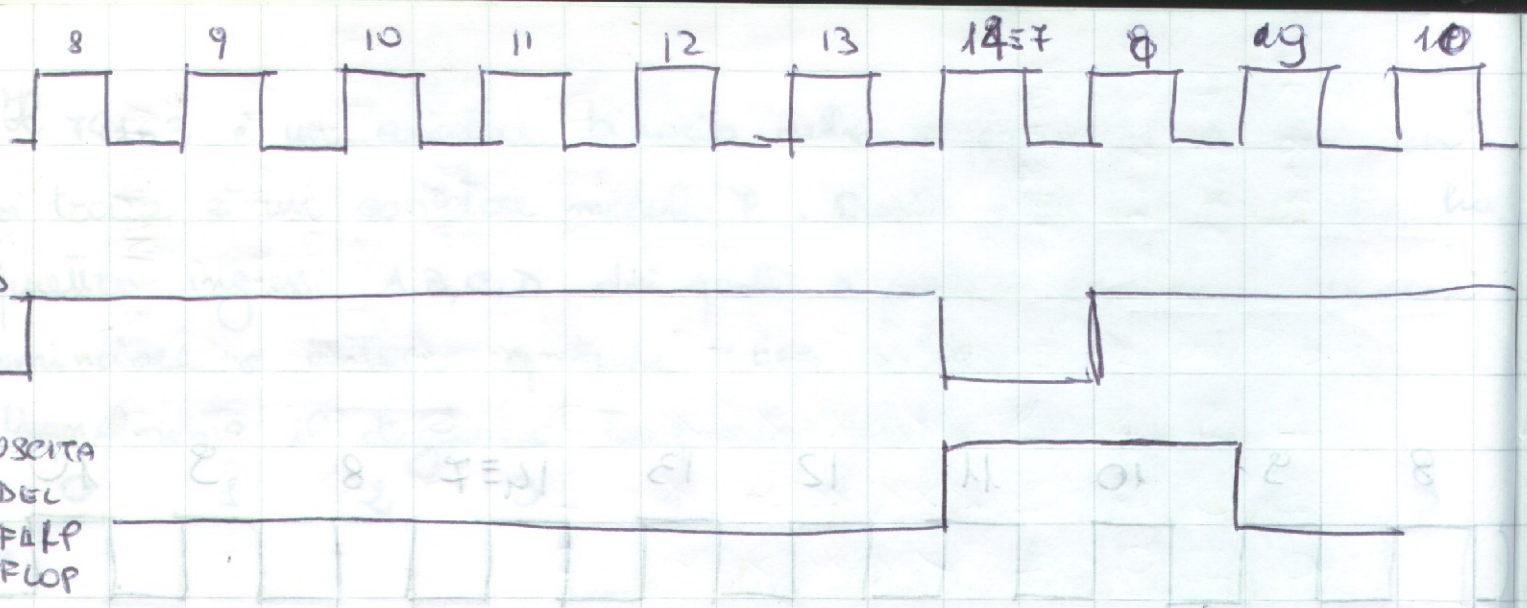
Teoricamente il diagramma temporale sarebbe il seguente



Nella nostra configurazione quando Q_A, Q_C e Q_D sono tutti entrambi ad 1 (cioè si è raggiunto almeno il numero 1101 = 13) viene posto LOAD a zero, per cui gli ingressi A, B, C, D che sono impostati a 0111 = 7 vengono caricati nel contatore. Come si vede dal diagramma temporale alla pagina seguente, il risultato conseguente è che Q_D si comporta come un'onda quadra di periodo $7 \cdot T_{CK}$ (gli ingressi P (ENP) e T (ENT) sono ingressi di abilitazione che devono essere contemporaneamente attivi alti). Come si vede dallo schema elettrico il segnale Q_D va all'ingresso di preset di un flip flop



D in un integrato 7474, flip flop D attivo sul fronte di salita del clock, dove abbiamo messo D a massa.



Quando Q_D va a zero esso attiva il PRESET (che è attivo basso) e l'uscita del flipflop va alta. Quando il segnale di PRESET torna inattivo, al successivo impulso di clock, l'uscita torna al valore datato da D cioè va bassa. Abbiamo così creato un ~~in~~ clock per l'8275.

Un'ultimo problema è la generazione delle interruzioni o dei segnali di richieste DMA. Noi adottiamo una soluzione in cui evitiamo l'uso di un blocco DMA.

Usiamo un secondo CTC che mandi una interruzione al µP ogni volta che è stata inviata una riga di caratteri al video in modo che il µP provveda a riempire un row buffer del µ8275. Tenendo presente che le linee in totale sono 270 e che per una riga occorrono 10 linee si avrà un'interruzione ogni

$$\frac{1}{f_{vertical\ di\ frame}} \times \frac{10}{270} = 16{,}67\ ms \times \frac{10}{270} = 0{,}617\ \mu s$$

questo corrisponde a contare 1000 impulsi del clock di carattere generato dall'uscita del flip flop sopra descritto. Questa uscita verrà allora mandata ad un secondo CTC

Quando $\overline{RD}$ (che è attivo a zero) del flip-flop esso attiva il PRESET (che è attivo basso) l'uscita Q va alta. Quando il segnale di PRESET torna inattivo, al successivo impulso di clock, l'uscita torna al valore dettato da D cioè a basso. Abbiamo così creato un clock per l'8275.

L'ultimo problema è la generazione delle interruzioni o dei segnali di richieste DMA. Noi adottiamo una soluzione in cui evitiamo l'uso di un blocco DMA.

Usiamo un secondo ctc che mandi una interruzione al uP ogni volta che è stata inviata una riga di caratteri al video in modo che il uP provveda a riempire un row buffer del 8275. Tenendo presente che le linee in totale sono 270 e che per una riga occorrono 10 linee si avrà un interruzione ogni

$$\frac{1}{f_{vertical\ di\ frame}} \times \frac{10}{270} = 16{,}67\ ms \times \frac{10}{270} = 0{,}67\ \mu s$$

questo corrisponde a contare 1000 impulsi del clock di carattere generato dall'uscita del flip flop sopra descritto. Questa uscita viene allora inviata ad un secondo ctc dove i contatori 0 e 1 sono in cascata ed entrambi

counter, con $N_0 = 250$ e $N_1 = 4$.

I dati da trasmettere al video sono $80 \times 25 = 2000$ quindi occorrono 2K di memoria che noi poniamo in una RAM che posizioniamo per semplicità all'estremo della mappa di I/O cioè da F800 ÷ FFFF. Per ridurre i tempi di uscita dei dati dall'8275 ricorriamo alla seguente tecnica: il µP, in realtà legge i dati dalla memoria video e una PROM da 32 byte quando vede che $A_{15}=1$, $A_{14}=1$, $A_{13}=1$, $A_{12}=1$, $A_{11}=1$ e $\overline{RD}=0$ (cioè il µP sta leggendo nella memoria video) pone l'uscita collegata al $\overline{WR}$ dell'8275 e quella collegata al suo ingresso $\overline{DACK}$ a zero, di modo che il dato prelevato dalla memoria video finisce anche nell'8275. Da tener presente che le periferiche INTEL che hanno anche DMA usano in realtà $\overline{DACK}$ come chip select per i dati. Così, quando si vuole scrivere un comando o leggere lo stato si usa $\overline{CS}$ e $\overline{WR}$ o $\overline{RD}$, ma quando si vuole leggere o scrivere dati si usa $\overline{DACK}$ e $\overline{RD}$ o $\overline{WR}$. La tabella della verità della PROM è alcune quelle della pagina seguente

WR 8275

RD 8275

CS 8275

DATA 8275

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
X	X	X	X	1	1	1	1
X	X	X	X	1	0	0	1
X	X	X	X	1	0	1	0
X	X	X	X	1	1	1	1
X	X	X	X	1	1	1	1
X	X	X	X	1	1	1	1
X	X	X	X	1	1	1	1
X	X	X	X	1	1	1	1
X	X	X	X	1	1	1	1
X	X	X	X	1	1	1	1
X	X	X	X	1	0	0	1
X	X	X	X	1	0	1	0
X	X	X	X	1	1	1	1
X	X	X	X	1	1	1	1
X	X	X	X	1	1	1	1
X	X	X	X	1	1	1	1

WR Z80

A_0

RD Z80

A_1

OSCITA DECODER I/O

A_2

A12 AND A11

A_3

A15 AND A14 AND A13

A_4

A_4	A_3	A_2	A_1	A_0
0	0	0	0	0
0	0	0	0	1
0	0	0	1	0
0	0	0	1	1
0	0	1	0	0
0	0	1	0	1
0	0	1	1	0
0	0	1	1	1
0	1	0	0	0
0	1	0	0	1
0	1	0	1	0
0	1	0	1	1
0	1	1	0	0
0	1	1	0	1
0	1	1	1	0
0	1	1	1	1

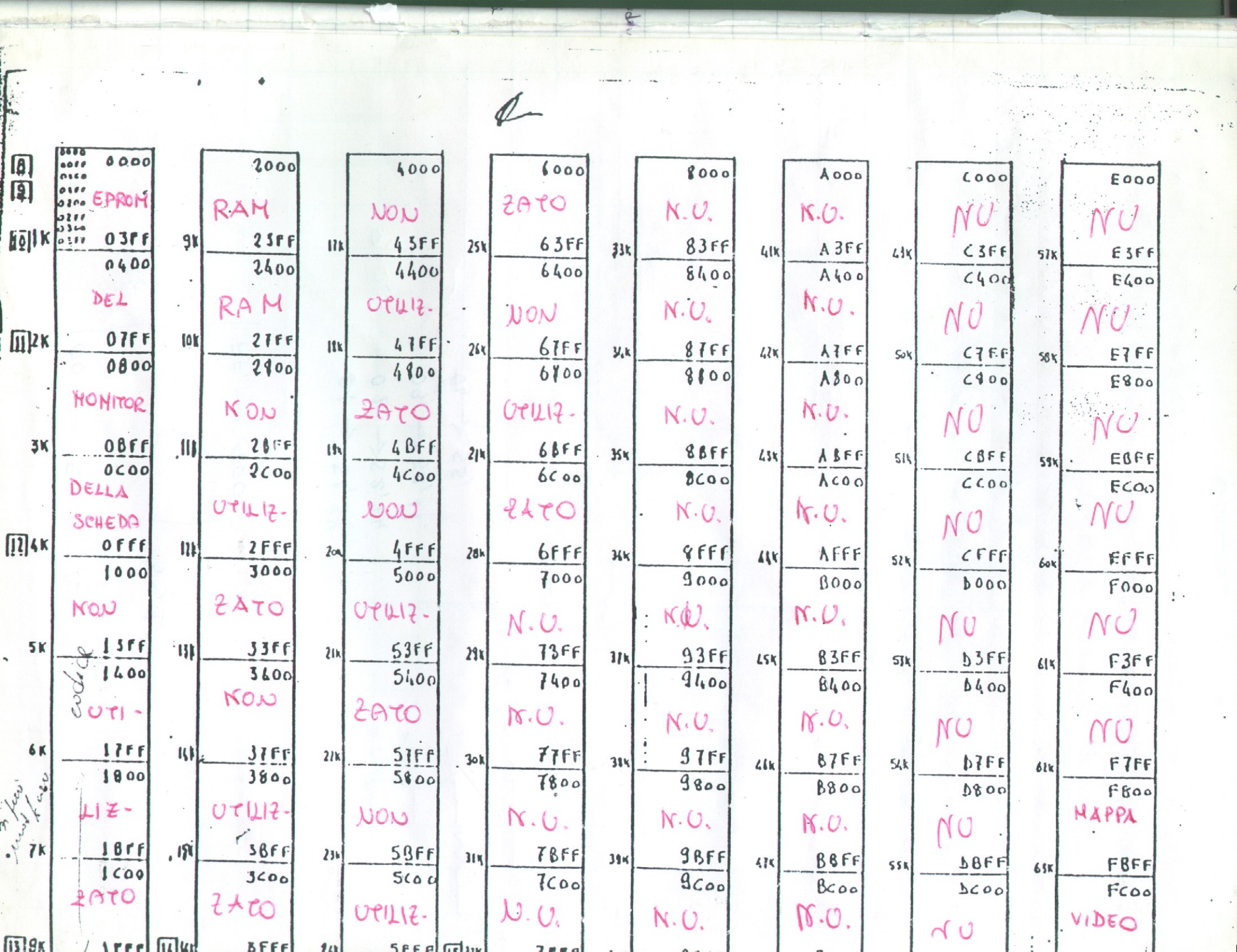
Il µP sta leggendo lo stato dell'8275

Il µP sta scrivendo un comando dell'8275

Il µP sta leggendo lo stato dell'8275

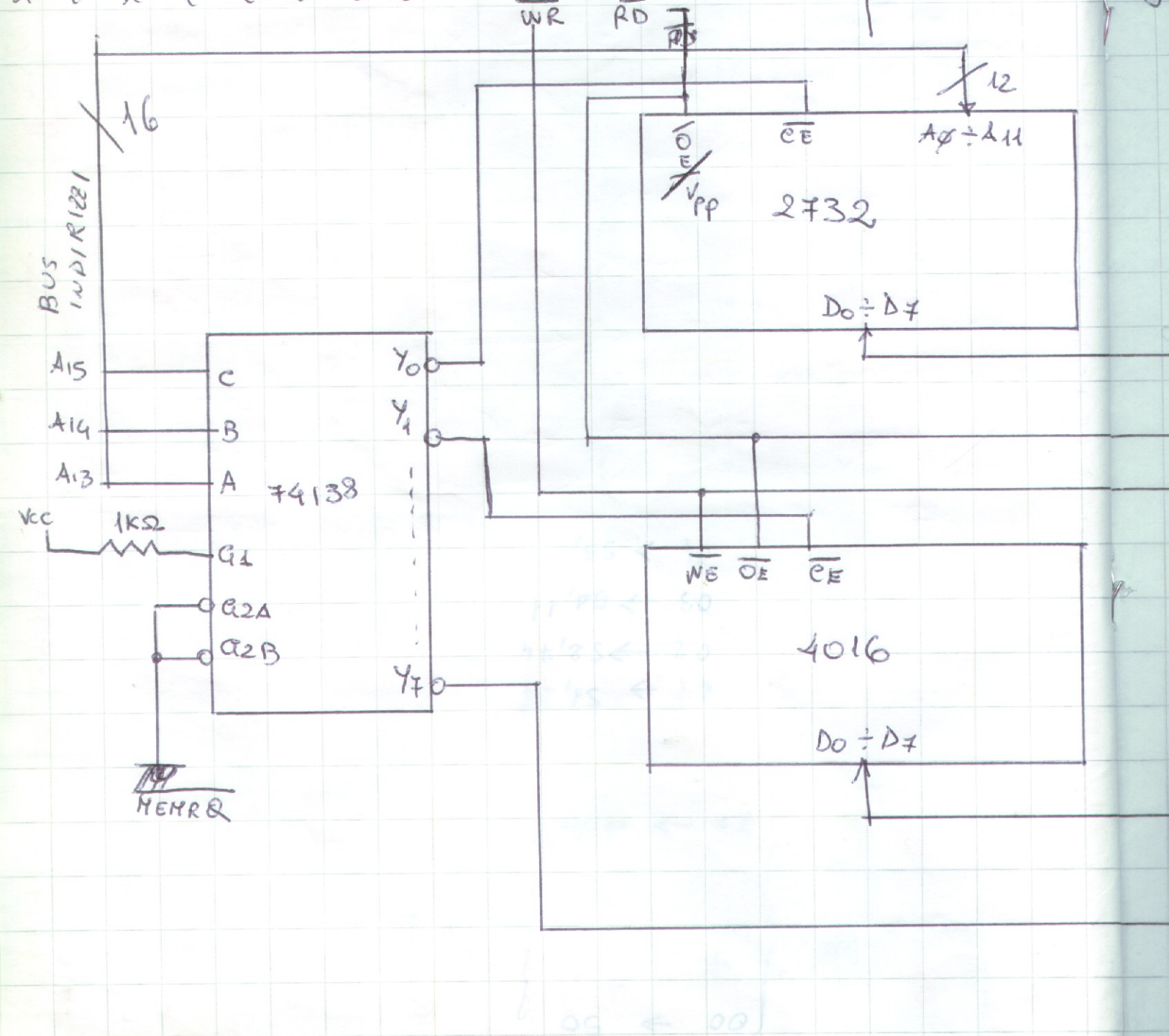
Il µP sta scrivendo un comando dell'8275

A_4	A_3	A_2	A_1	A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
1	0	0	0	0	X	X	X	X	1	1	1	1	
1	0	0	0	1	X	X	X	X	1	0	0	1	
1	0	0	1	0	X	X	X	X	1	0	1	0	
1	0	0	1	1	X	X	X	X	1	1	1	1	
1	0	1	0	0	X	X	X	X	1	1	1	1	
1	0	1	0	1	X	X	X	X	1	1	1	1	
1	0	1	1	0	X	X	X	X	1	1	1	1	
1	0	1	1	1	X	X	X	X	1	1	1	1	
1	1	0	0	0	X	X	X	X	1	1	1	1	
1	1	0	0	1	X	X	X	X	1	0	0	1	lettura stato 8255
1	1	0	1	0	X	X	X	X	1	0	1	0	lettura dato 8255
1	1	0	1	1	X	X	X	X	1	1	1	1	
1	1	1	0	0	X	X	X	X	1	1	1	1	
1	1	1	0	1	X	X	X	X	0	1	1	0	lettura memoria video
1	1	1	1	0	X	X	X	X	1	1	1	1	
1	1	1	1	1	X	X	X	X	1	1	1	1	


0000
EPROM
03FF
0400
DEL
07FF
0800
MONITOR
0BFF
0C00
DELLA
SCHEDA
0FFF
1000
NON
13FF
1400
UTI-
17FF
1800
LIZ-
1BFF
1C00
ZATO
1FFF
2000
RAM
23FF
2400
RAM
27FF
2800
NON
2BFF
2C00
UTILIZ-
2FFF
3000
ZATO
33FF
3400
NON
37FF
3800
UTILIZ-
3BFF
3C00
ZATO
3FFF
4000
NON
43FF
4400
UTILIZ-
47FF
4800
ZATO
4BFF
4C00
NON
4FFF
5000
UTILIZ-
53FF
5400
ZATO
57FF
5800
NON
5BFF
5C00
UTILIZ-
5FFF
6000
ZATO
63FF
6400
NON
67FF
6800
UTILIZ-
6BFF
6C00
ZATO
6FFF
7000
N.U.
73FF
7400
N.U.
77FF
7800
N.U.
7BFF
7C00
N.U.
7FFF
8000
N.U.
83FF
8400
N.U.
87FF
8800
N.U.
8BFF
8C00
N.U.
8FFF
9000
N.U.
93FF
9400
N.U.
97FF
9800
N.U.
9BFF
9C00
N.U.
9FFF
A000
N.U.
A3FF
A400
N.U.
A7FF
A800
N.U.
ABFF
AC00
N.U.
AFFF
B000
N.U.
B3FF
B400
N.U.
B7FF
B800
N.U.
BBFF
BC00
N.U.
BFFF
C000
NU
C3FF
C400
NU
C7FF
C800
NU
CBFF
CC00
NU
CFFF
D000
NU
D3FF
D400
NU
D7FF
D800
NU
DBFF
DC00
NU
DFFF
E000
NU
E3FF
E400
NU
E7FF
E800
NU
EBFF
EC00
NU
EFFF
F000
NU
F3FF
F400
NU
F7FF
F800
MAPPA
FBFF
FC00
VIDEO
FFFF
1K
2K
3K
4K
5K
6K
7K
8K
9K
10K
11K
12K
13K
14K
15K
16K
17K
18K
19K
20K
21K
22K
23K
24K
25K
26K
27K
28K
29K
30K
31K
32K
33K
34K
35K
36K
37K
38K
39K
40K
41K
42K
43K
44K
45K
46K
47K
48K
49K
50K
51K
52K
53K
54K
55K
56K
57K
58K
59K
60K
61K
62K
63K
64K

MAPPATURA DI MEMORIA

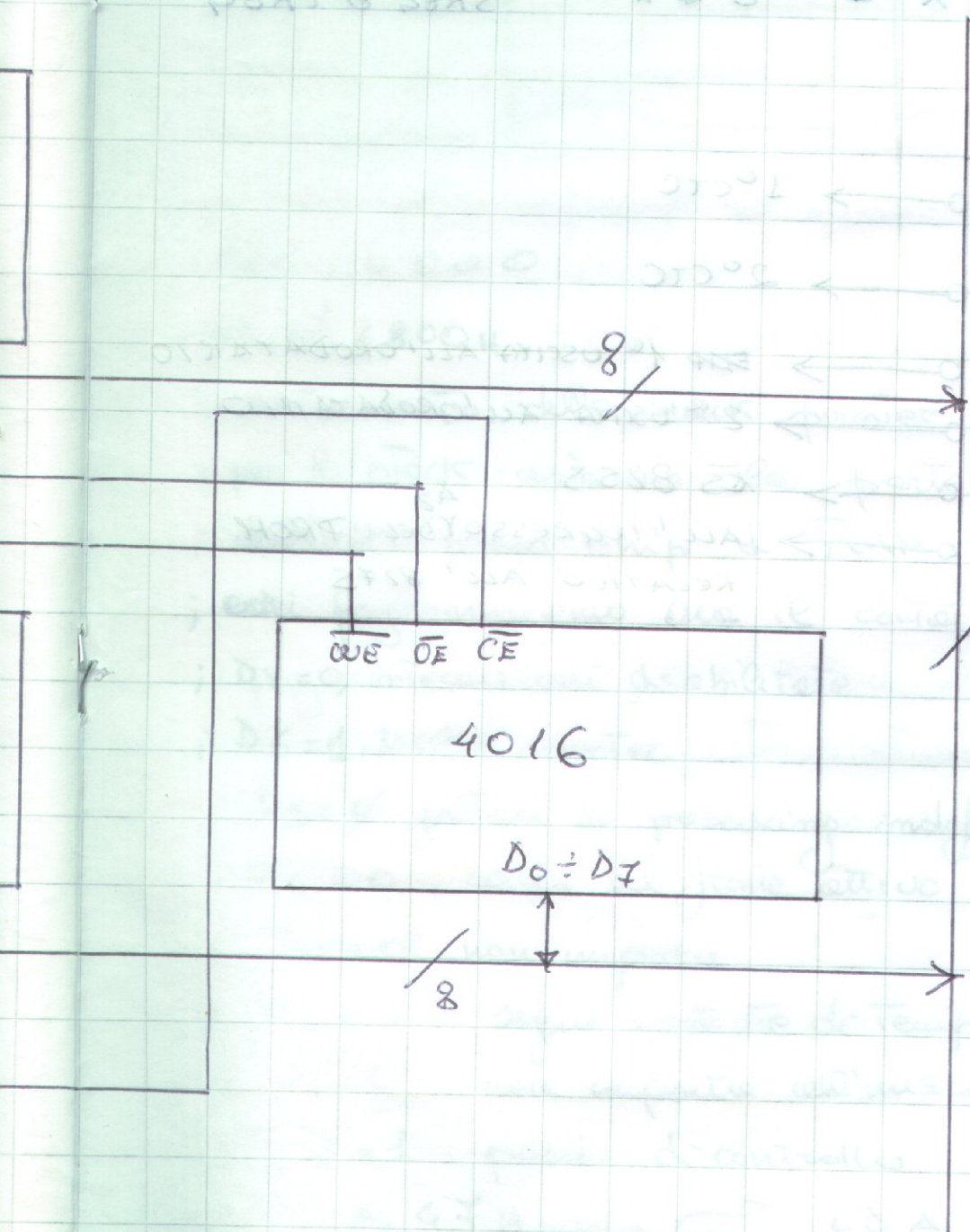
A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	INDIRIZZO HEX
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0 0 0 0
0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0 F F F
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	2 0 0 0
0	0	1	0	0	1	1	1	1	1	1	1	1	1	1	1	2 7 F F
1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	F 8 0 0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	F F F F



EPROM DEL MONITOR

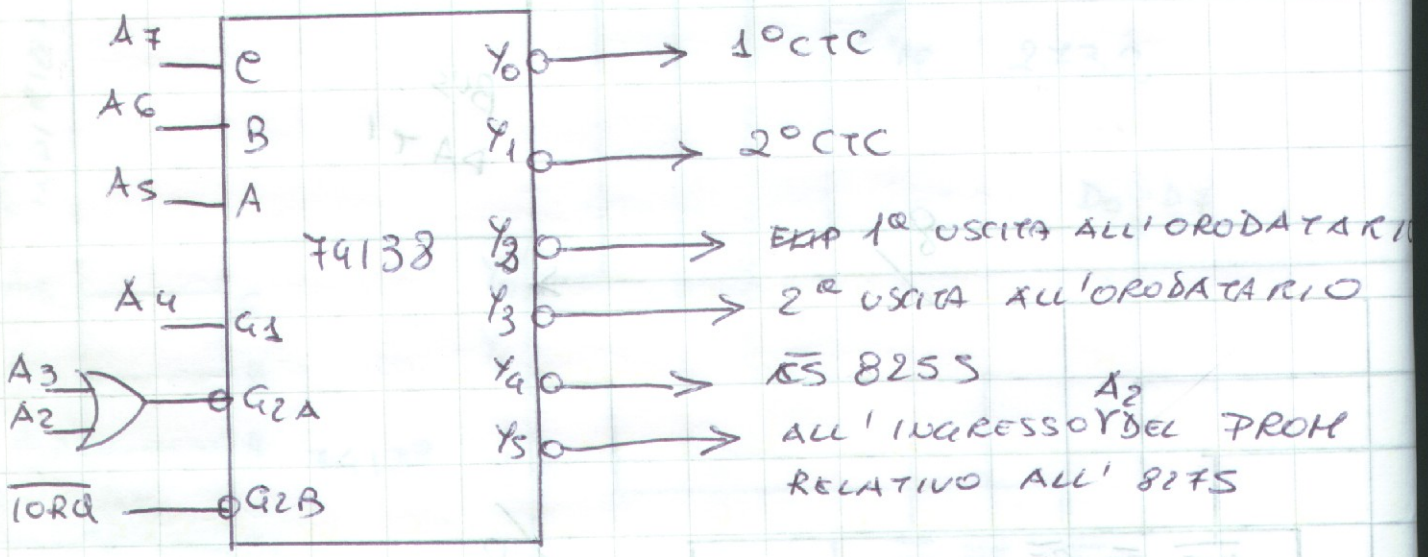
RAM GENERALE

RAM MEMORIA VIDEO



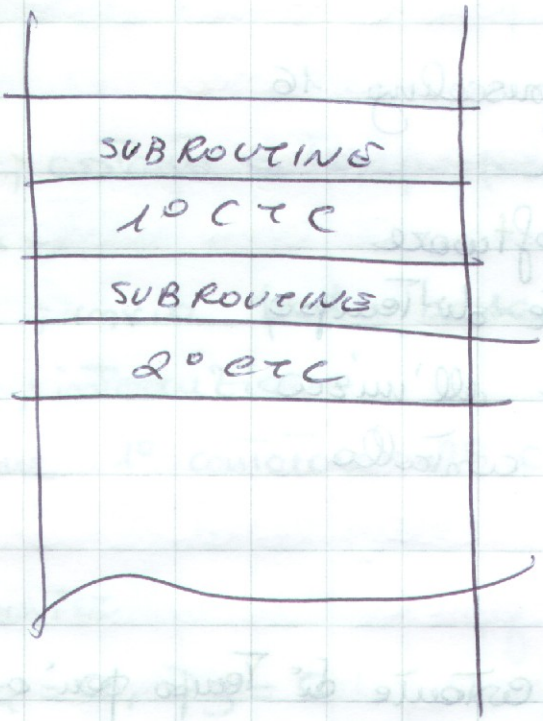
MAPPATURA DI I/O

A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	INDIRIZZO HEX	
0	0	0	1	0	0	X	X	10 H	1° CTC INDIRIZZO BASE
0	0	1	1	0	0	X	X	30 H	2° CTC "
0	1	0	1	0	0	X	X	50 H	ATTIVAZIONE CLOCK ORODATARIO
0	1	1	1	0	0	X	X	70 H	SCRITTURA/LETTURA ORODATARIO
1	0	0	1	0	0	X	X	90 H	INDIRIZZO 8255 BASE
1	0	1	1	0	0	X	0	B0 H	INDIRIZZO 8275 PREG
1	0	1	1	0	0	X	1	B1 H	INDIRIZZO 8275 SREG O CREG



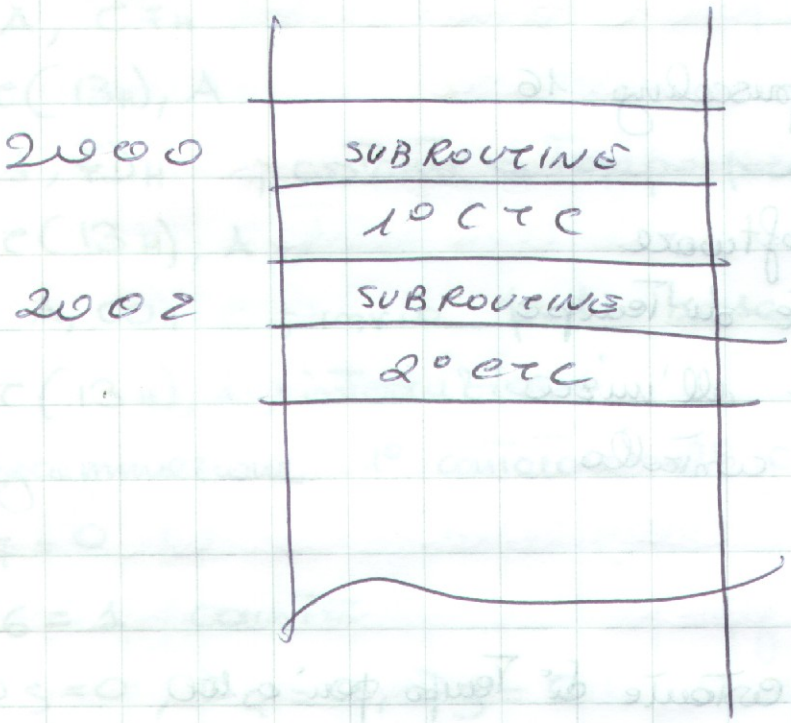
SOFTWARE

I due CTC generano ciascuno un'interruzione, decidiamo di porre la Tabella delle interruzioni all'inizio della RAM




```
ORG  0000
LD SP, 2800H
; caricamento dello Stack pointer
; per lo stack usiamo la parte finale
; dell'ultimo primo chip di RAM
; programmiamo ora il contatore Ø del 1° CTC
; D4 = 0, interruzioni disabilitate
; D6 = 1, counter
; D5 = Ø fattore di prescaling indifferente
; D4 = 0, scelta del fronte attivo del clock/trigger
; D3 = 0 non importa
; D2 = 1 segue costante di tempo
; D1 = 1 non importa all'inizio
; DØ = 1 parola di controllo
LD A, 47H  ↝  LD A, FFH
```

Il due CTC generano ciascuno un'interruzione, decidiamo di porre la Tabella delle interruzioni all'inizio della RAM




```
ORG 0000
LD SP, 2800H
; caricamento dello Stack pointer
; per lo stack usiamo la parte finale
; dell'ultimo chip di RAM
; programmiamo che il contatore Ø del 1° CTC
; D4 = 0, interruzioni disabilitate
; D6 = 1, counter
; D5 = Ø fattore di prescaling indifferente
; D4 = 0, scelta del fronte attivo del clock/trigger
; D3 = 0 non importa
; D2 = 1 segue costante di tempo
; D1 = 1 non importa all'inizio
; DØ = 1 parola di controllo
LD A, 47H                    LD A, FFH
OUT (10H), A                 OUT (10H), A
```

; programmazione del contatore 1 del 1°CTC
; $D_7 = 0$ interruzioni disabilitate
; $D_6 = 0$ timer
; $D_5 = 0$ fattore di prescaling 16
; $D_4 = 0$ non importa
; $D_3 = 0$ partenza software
; $D_2 = 1$ segue costante di tempo
; $D_1 = 1$ non importa all'inizio
; $D_0 = 1$ parola di controllo

```
LD A, 27H
OUT (11H), A
LD A, 64H ; costante di tempo pari a 100
OUT (11H), A
```

; programmazione contatore 2 del 1°CTC
; $D_7 = 0$
; $D_6 = 1$ ⇒ counter
; $D_5 = 0$
; $D_4 = 0$
; $D_3 = 0$
; $D_2 = 1$ segue costante di tempo
; $D_1 = 1$ non importa all'inizio
; $D_0 = 1$

```
LD A, 47H
OUT (12H), A
LD A, 7DH ; 125 in esadecimale
OUT (12H), A
```

; programmazione contatore 3 del 1° CTC
; $D_7=1$, $D_6=1$, $D_5=0$, $D_4=0$, $D_3=0$, $D_2=1$, $D_1=1$, $D_0=1$
LD A, C7H
OUT(13H), A
LD A, FDH ; costante di Tempo pari a 125
OUT(13H), A
LD A, 00H ; invio parte bassa vettore delle
OUT(13H), A ; interruzioni
; programmazione 1° contatore 2° CTC
; $D_7=0$
; $D_6=1$ counter
; $D_5=0$, $D_4=0$, $D_3=0$
; $D_2=1$, $D_1=1$, $D_0=1$
LD A, 47H
OUT(30H), A
; costante di Tempo pari a $250/10 = FA|_H$
LD A, FAH
OUT(30H), A
; programmazione 2° contatore 2° CTC
; l'unica differenza è $D_7=1$ ≡ interruzioni abilitate
LD A, C7H
OUT(31H), A
; costante di Tempo
LD A, 04H
OUT(31H), A
LD A, 02H ; parte bassa
OUT(31H), A ; vettore interruzioni
; programmazione dell'8255

; programmazione contatore 3 del
; $D_7=1$, $D_6=1$, $D_5=0$, $D_4=0$, $D_3=0$, $D_2=1$, $D_1=1$, $D_0=1$
LD A, C7H
OUT(13H), A
LD A, FDH ; costante di Tempo pari a 125
OUT(13H), A
LD A, 00H ; invio parte bassa vettore delle
OUT(13H), A ; interruzioni
; programmazione 1° contatore 2° CTC
; $D_7=0$
; $D_6=1$ counter
; $D_5=0$, $D_4=0$, $D_3=0$
; $D_2=1$, $D_1=1$, $D_0=1$
LD A, 47H
OUT(30H), A
; costante di Tempo pari a 250/10 = FA|H
LD A, FAH
OUT(30H), A
; programmazione 2° contatore 2° CTC
; l'unica differenza è $D_7=1$ ≡ interruzioni abilitate
LD A, C7H
OUT(31H), A
; costante di Tempo
LD A, 04H
OUT(31H), A
LD A, 02H ; parte bassa
OUT(31H), A ; vettore interruzioni
; programmazione dell'8255
; $D_7=1$ identifica come parola di programmazione
; di modo

; $D_6 D_5 = 00$ modo zero porta A
; $D_4 = 0$ porta A impostata come uscita
; $D_3 = 0$ porta alta porta C come uscita
; $D_2 = 0$ porta B in modo zero
; $D_1 = 1$ porta B come ingresso
; $D_0 = 0$ porta bassa C come uscita
LD A, 82_H
OUT (33_H), A
LD HL, 0800_H
; supponiamo che a 0800_H comincia il sotto
; programma che gestisce la richiesta d'
; interruzione da porta del 1° CTC
OUT (2000_H), HL
LD HL, 0600_H
; indirizzo sottoprogramma 2° CTC
OUT (2002_H), HL
~~LD HL~~, LD A, 20_H ; porta alta vettore
LD I, A ; delle interruzioni
~~LD HL~~
LD HL, 2006_H
; facciamo puntare HL alla prima locazione
; in cui inserire le velocità
LD DE, $FB00_H$
; facciamo puntare DE all'inizio della memoria video
~~LD~~ A, 20_H
; poniamo ~~A a zero~~ in A il codice esadecimale

; $D_4 = 0$ porte A impostate come uscite
; $D_3 = 0$ porte alte porte C come uscite
; $D_2 = 0$ porte B in modo zero
; $D_1 = 1$ porte B come ingresso
; $D_0 = 0$ porte basse C come uscite
LD A, 82_H
OUT (33_H), A
LD HL, 0800_H
; supponiamo che a 0800_H comincia il sotto
; programma che gestisce le richieste di
; interruzione dei porte del 1° CTC
OUT (2000_H), HL
LD HL, 0600_H
; indirizzo sottoprogramma 2° CTC
OUT (2002_H), HL
~~LD HL~~, LD A, 20_H ; porte alte vettore
LD I, A ; delle interruzioni
~~HL~~
~~I~~
LD HL, 2006_H
; facciamo puntare HL alla prima locazione
; in cui inserire le velocità
LD DE, $FB00_H$
; facciamo puntare DE all'inizio della memoria video
~~LD~~ A, 20_H
; poniamo ~~A a zero~~ in A il codice Ascii del carattere bianco
LD BC, $07D0$

```
; carico BC con 2000/10 = 07D0H
INIZIO: LD (DE), A
        DEC C
        JR NZ, INIZIO
; decremento C, se non è ancora nullo salta all'inizio
; altrimenti carico C con FFH e decremento di uno B
        LD C, FFH
        DEC B
        DJNZ, ~~toof~~ INIZIO
; con questo ciclo si pone nella memoria video il codice
; ASCII per il carattere null in modo che nel video
; non appaia nulla.
        LD DE, F800H
; ripristino DE originario.
        LD A, 00H
        OUT (B1H), A
; invio all' 8275 del comando di reset
        LD A, 4FH
        OUT (B0H), A
; invio del 1° parametro del comando di reset
; linee nascondi, 80 caratteri per riga
        LD A, 59H
        OUT (B0H), A
; invio del 2° parametro impostando 25
; righe di ~~comandi~~ caratteri per quadro + 2 linee per
; il ritrace verticale.
        LD A, FAH
        OUT (B0H), A
```

```
INIZIO: LD (DE), A
        DEC C
        JR NZ, INIZIO
; decremento C, se non è ancora nullo salta all'inizio
; altrimenti carica C con FFH e decrementa il valore B
        LD C, FFH
        DEC B
        DJNZ, ~~loop~~ INIZIO
; con questo ciclo si pone nella memoria video il codice
; ASCII per il carattere nullo in modo che nel video
; non appaia nulla.
        LD DE, FB00H
; ripristino DE originario.
        LD A, 00H
        OUT (B1H), A
; invio all' 8275 del comando di reset
        LD A, 4FH
        OUT (B0H), A
; invio del 1° parametro del comando di reset
; linee normali, 80 caratteri per riga
        LD A, 59H
        OUT (B0H), A
; invio del 2° parametro impostando 25
; righe di caratteri per quadro + 2 linee per
; il retrace verticale.
        LD A, FAH
        OUT (B0H), A
; 3° parametro, pongo 10 linee per
```

```
; carattere e la linea della sottolineatura per
; e 16 , poiché 16 > 10 non si avrà sottolineatura
    LD A, 05H
    OUT (B0H), A
; il numero di caratteri per la retrace ver
; tale è impostato a 20.
    LD A, E0H
    OUT (B1H), A
; inizio del preset counters command
    IM 2                EXX
    LDDE, 012CH ; carica in DE'
    EI                  EXX
                        ; un contatore per
    HALT                ; capire se sono passati
                        ; 5 minuti

    ORG 0300H
; sottoprogramma che gestisce l'interruzione inviata
; ogni 50 secondi dal 1° CTC
; questo sottoprogramma ha il compito di rilevare
; la velocità del nastro, forse di registrarla in memoria
; dati e in memoria video e controllare se è
; pericolosamente elevata
    IN A, (10H)
; prelievo del numero del contatore 0 del 1° CTC
    LD B, A
    LD A, FFH
    SUB A, B
; calcolo del ΔN
    CALL DIV
```

; e 16 , poiché 16 > 10 non si deve sottolineare pe
LD A, 05H
; OUT (B0H), A
; il numero di cicli per cui deve rimanere
; tale è impostato a 20.
LD A, E0H
OUT (B1H), A
; inizio del preset counters command
IM2 EXX
LD DE', 012CH ; carica in DE'
EI ; un contatore per
EXX ; capire se sono passati
HALT ; 5 minuti

ORG 0300H
; sottoprogramma che gestisce l'interruzione inviata
; ogni 50 secondi dal 1° CTC
; questo sottoprogramma ha il compito di rilevare
; la velocità del nastro, forse registrarla in memoria
; dati e in memoria video e controllare se è
; pericolosamente elevata
IN A, (10H)
; prelievo del numero del contatore 0 del 1° CTC
LD B, A
LD A, FFH
SUB A, B
; calcolo del ΔN
CALL DIV
; supponiamo di dover disporre un

```
; sottoprogramma che calcoli v e ponga in A
; la velocità e in A le sue parte decimale, tutto in BCD
    SUB A,01H
    CALL CONVERT
; questa subroutine converte la velocità in caratteri
; ASCII e la pone nella memoria video
    SUB A, 01H
; si controlla se la velocità è inferiore a 1 metro al
; secondo
    JP P, AVANTI
; se A il risultato della sottrazione è positivo, cioè
; v > 10 m/s si va avanti altrimenti
    LD B, A
    LD A, 01H
    OUT (82H), A
; invio una parola per settare il bit C0 dell' 8155
; ad 1, supponendo che a questo sia collegato il
; segnalatore acustico
    LD A, B
    ADD A, 01H
; ripristino A
AVANTI: SUB A, 10H
; controllo se la velocità è superiore a 10
    LD IX, 0FEA
; nelle ultime locazioni della ROM vi sono
; registrati i codici ASCII per visualizzare i
; messaggi 'VELOCITA' TROPPO BASSA' e
; 'VELOCITA' TROPPO ALTA', che occupano
; ciascuno 32 byte, il 1° messaggio
```

SUB A, 01H

CALL CONVERT

; questa subroutine converte la velocità in caratteri
; ASCII e la pone nella memoria video

SUB A, 01H

; si controlla se la velocità è inferiore a 1 metro al
; secondo

JP P, AVANTI

; se A il risultato della sottrazione è positivo, cioè
; > 10 m/s si va avanti altrimenti

LD B, A
LD A, 01H

OUT (82H), A

; invio una parola per settare il bit C0 dell'8155
; ad 1, supponendo che a questo sia collegato il
; segnalatore acustico

LD A, B

ADD A, 01H

; ripristino A

AVANTI: SUB A, 10H

; controllo se la velocità è superiore a 10

LD IX, 0FEA

; nelle ultime locazioni della ROM vi sono
; registrati i codici ASCII per visualizzare i
; messaggi 'VELOCITA' TROPPO BASSA' e
; 'VELOCITA' TROPPO ALTA', che occupano
; ciascuno 32 byte, il 1° messaggio
; inizia a 0FE0H e il 2° messaggio

```
; inizio a OFD4H
    LD B, 16H
; poniamo B come contatore dei 22 caratteri
ET: LD A, (IX)
    LD (DE), A
; trasferimento dei caratteri del messaggio in
; memoria video
    INC IX
    INC DE
    DJNZ ET
AVANTI: LD A, C
    SUB A, 10H ; ripristino in A la velocità
    ; controlliamo se la velocità è superiore
    ; a 10 m/s
    JP M, AVANTI2
; se non è vero aggiorno avanti altrimenti
    LD B, A
    LD A, 01H
    OUT (92H), A
; attiviamo avvisatore acustico
    LD A, B
    ADD A, 10H
; ripristino in A il valore della velocità
    LD IX, OFD4
    LD C, A
    LD B, 16H
ET2: LD A, (IX)
    LD (DE), A
    INC IX
```

; pongo B come contatore dei 22 caratteri
ET : LD A, (IX)
LD (DE), A
; trasferimento dei caratteri del messaggio in
; memoria video
INC IX
INC DE
DJNZ, ET
AVANTI: LD A, C
SUB A, 10H ; ripristino in A la velocità
; controlliamo se la velocità è superiore
; a 10 m/s
JP PL, AVANTI2
; se non è vero aziono avanti altrimenti
LD B, A
LD A, 01H
OUT (92H), A
; azioniamo avvisatore acustico
LD A, B
ADD A, 10H
; ripristino in A il valore della velocità
LD IX, 0FD4
LD C, A
LD B, 16H
ET2: LD A, (IX)
LD (DE), A
INC IX
INC DE

0FD4	5	6	← 'V'
0FD5	4	5	← 'E'
0FD6	4	C	← 'L'
0FD7	4	F	← 'O'
0FD8	4	3	← 'C'
0FD9	4	9	← 'I'
0FDA	5	4	← 'T'
0FDB	4	1	← 'A'
0FDC	6	0	← '1'
0FDE	2	0	← ' ' ← SPAZIO BIANCO
0FDF	5	4	← 'T'
0FE0	5	2	← 'R'
0FE1	4	F	← 'O'
0FE2	50		← 'P'
0FE3	50		← 'P'
0FE4	4F		← 'O'
0FE5	20		← ' '
0FE6	41		← 'A'
0FE7	4C		← 'L'
0FE8	53		← 'S'
0FE9	41		← 'A'
0FEA	56		← 'V'

0FEA	56	'V'
0FEB	45	'E'
0FEC	4C	'L'
0FED	4F	'O'
0FEE	43	'C'
0FEF	49	'I'
0FF0	54	'T'
0FF1	41	'A'
0FF2	60	'!'
0FF3	20	' '
0FF4	54	'R'
0FF5	52	'R'
0FF6	45	'O'
0FF7	50	'P'
0FF8	50	'O'
0FF9	4F	'O'
0FFA	20	' '
0FFB	42	'B'
0FFC	41	'A'
0FFD	53	'S'
0FFE	53	'S'
0FFF	41	'A'

```
        DJNZ, ET2
        LD A,C
AVANTI2: LD (HL), A
        ; registrazione velocità nella memoria RAM
        INC HL
        LD DE, F800H ; riporto all'inizio della memoria
                     ; video
AVANTI2: EXX
        ; recupero il contatore DE'
        DEC DE
        LD HL, DE
        AND A
        ; pongo il flag di carry a zero
        SBC HL, 0000H
        JP NZ, AVANTI3 ET7
; se DE' non è arrivato a zero si va alla
; fase altrimenti: sul disco sono presenti
; i 5 minuti
        EXX
        LD (HL), A
        INC HL
```

→ `EX AF, AF' ; per`
`LD (HL), A`
`INC HL`

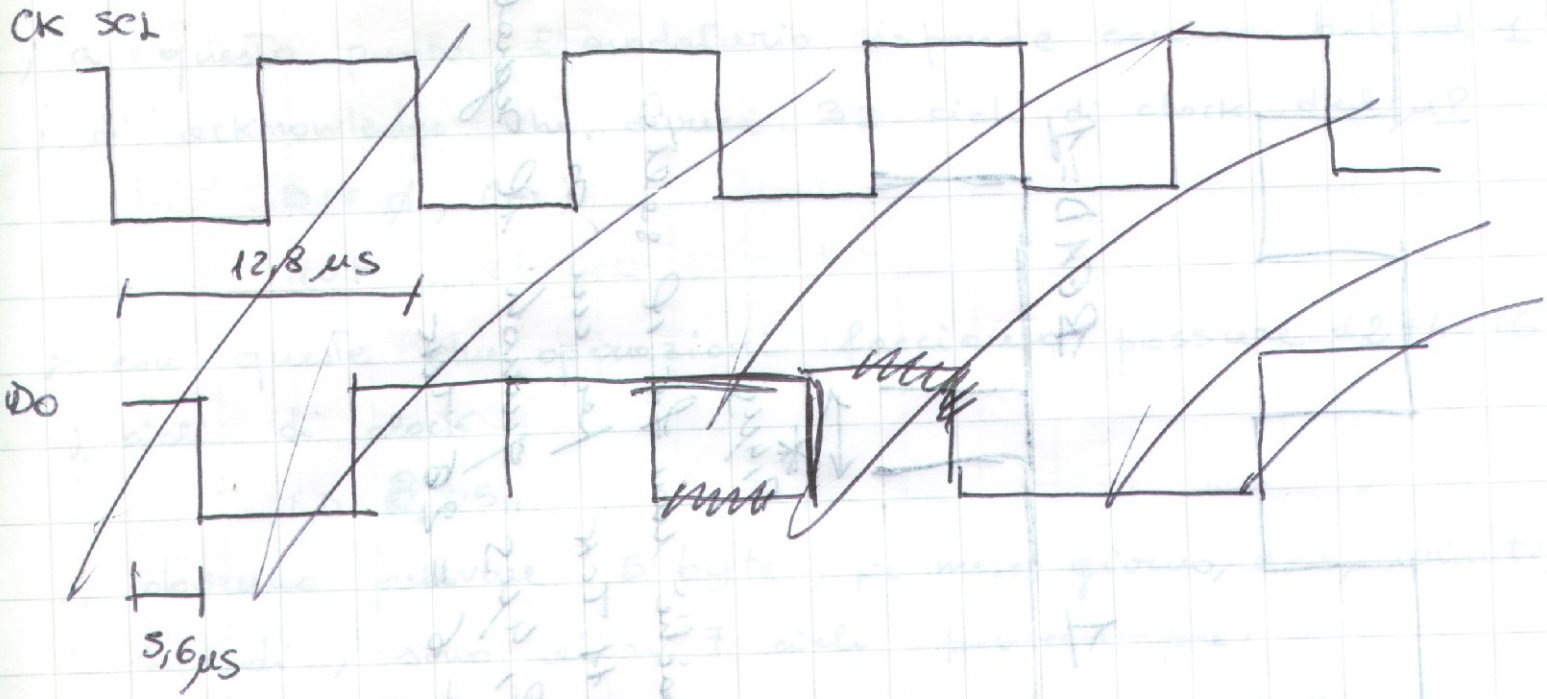
```
; per cui la velocità va registrata in
; memoria
        AND A
        SBC HL, 2005H
; dobbiamo controllare se sono passate 24 ore
; cioè se sono stati registrati 12x24 = 288 dati
; dobbiamo ora registrare l'ora
; della rilevazione
```

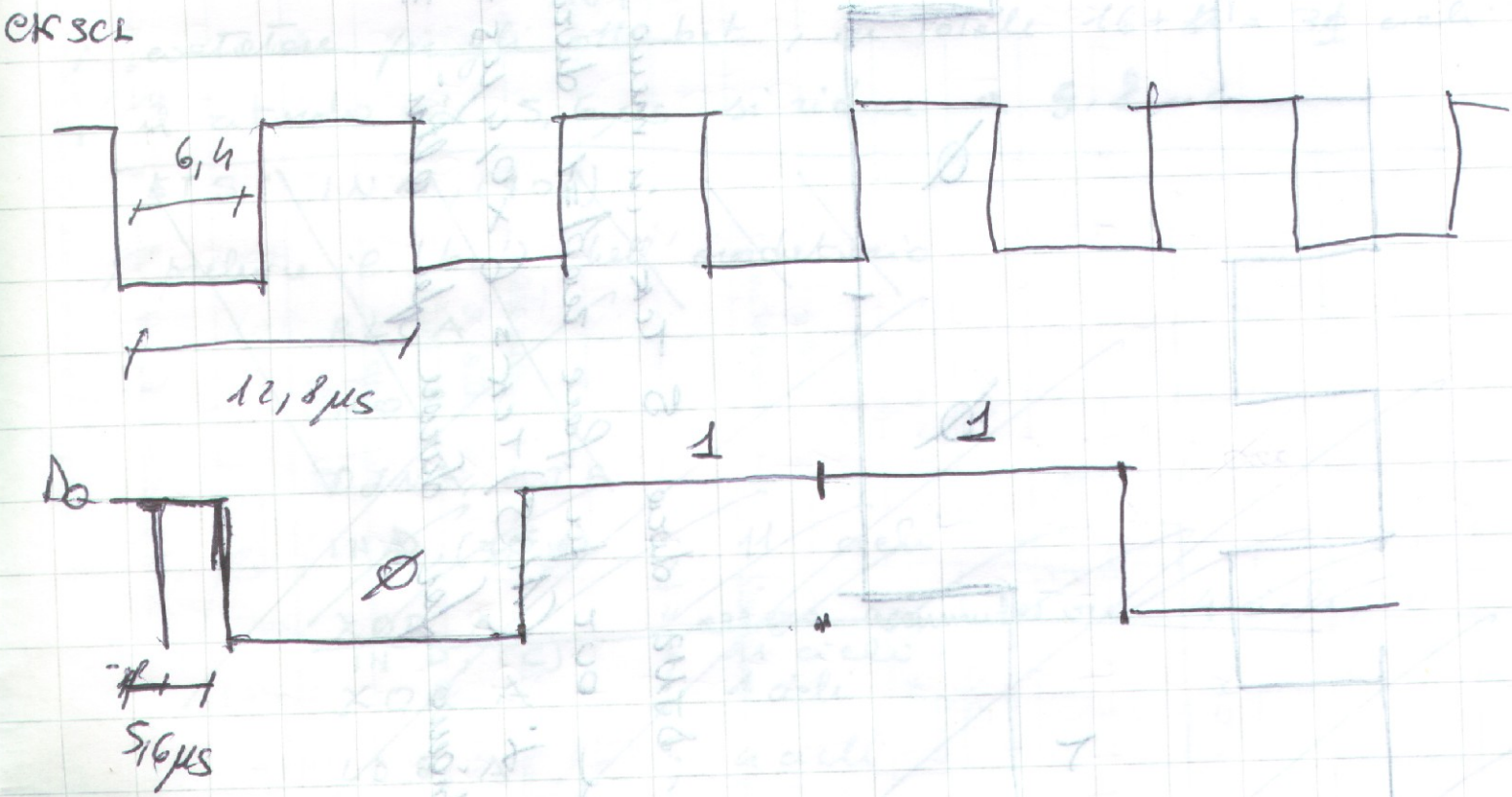
```
        JP NZ, ET2
        LD A, C
AVANTI2 : LD (HL), A
; registrazione velocità nella memoria RAM
        INC HL
        LD DE, F800H ; riporto all'inizio della memoria video
AVANTI2 : EXX
; recupero il contatore DE'
        DEC DE
        LD HL, DE
        AND A
; pongo il flag di carry a zero
        SBC HL, 0000H
        JP NZ, ET7
; se DE' non è arrivato a zero si va alla fine altrimenti sul display sono presenti
; i 5 minuti
        EXX
        LD (HL), A
        INC HL
; per cui la velocità va registrata in memoria
        AND A
        SBC HL, 2005H
; dobbiamo controllare se sono passate 24 ore
; ed se sono stati registrati 12x24 = 288 dati
; dobbiamo ora registrare l'ora della rilevazione
```

EX AF, AF' ; per
LD (HL), A
INC HL

```
        RES 0, A
        OUT (30H), A
; attivazione del clock di lettura dell'orologio
; è una scrittura fittizia, serve solo ad attivare
; la parte che invia il clock all'ingresso SCL dell'o
; dotorio
        LD A, 10010110 1B
; poniamo in A una stringa che rappresenta l'indiriz
; da inviare bit a bit all'ingresso SDA dell'orologi
; questo andrà inviato a partire dal bit meno
; significativo
        LD B, 08H
ETG:    OUT (70H), A
        RRCA
        NOP
; rotazione di A per inviare il bit successivo
        DJNZ, ETG
; il clock dell'orologio è 32 volte più lento di
; quello del μP, poniamo nel ciclo l'istruzione NOP
; in modo che questo duri esattamente 32 cicli di clo
; del μP
        XORA, 00
; il bit 00 di A è ancora 1 e lo invio
        OUT (70H), A
; in modo da inviare il segnale di Read
        LD A, (0000H)
        LD A, (0000H)
; questo dura esattamente 13 cicli di clock
; in modo da fare i 26 che occorrono
```

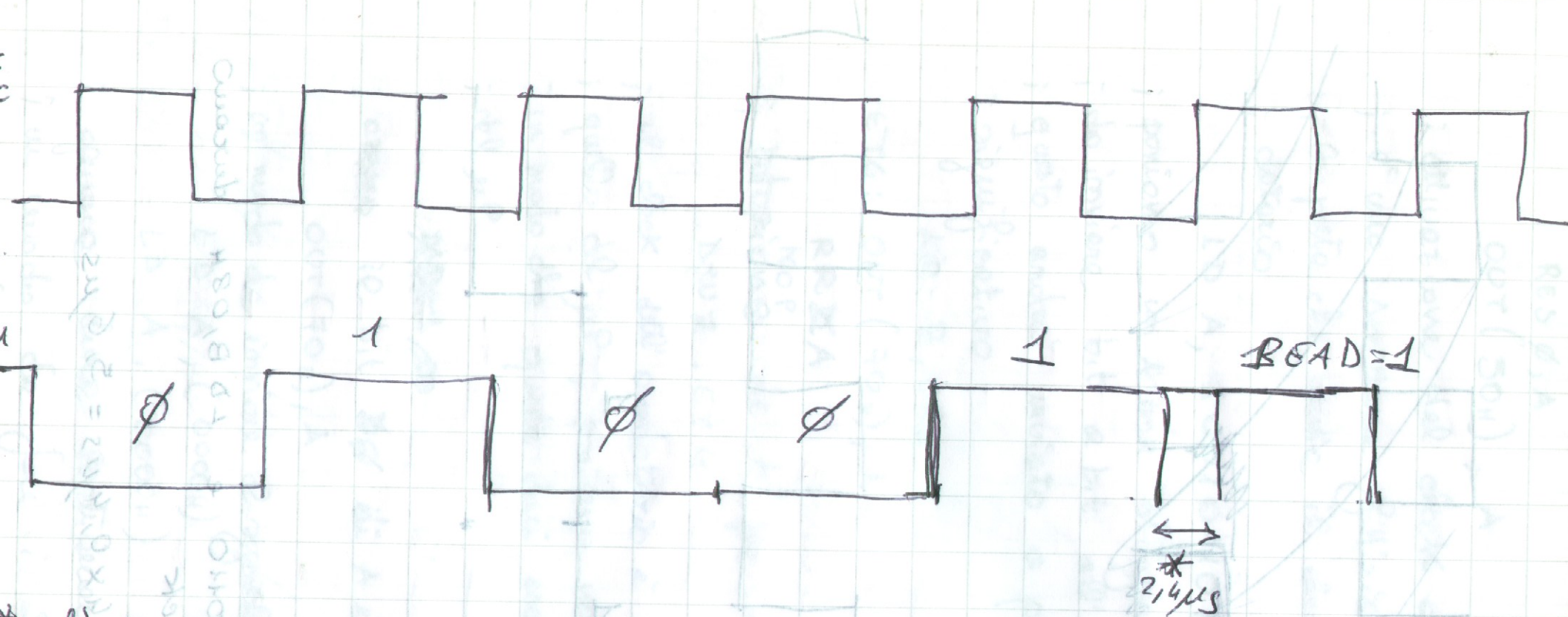
```
        OUT (30H), A
; attivazione del clock di lettura dell'orologio
; è una scrittura fittizia, serve solo ad attivare
; la porta che invia il clock all'ingresso SCL dell'o
; rologio
        LD A, 10010110B
; poniamo in A una stringa che rappresenta l'indiriz
; zo da inviare bit a bit all'ingresso SDA dell'orolo
; gio, questo andrà inviato a partire dal bit meno
; significativo
        LD B, 08H
ETG:    OUT (70H), A
        RRCA
        NOP
; rotazione di A per inviare il bit successivo
        DJNZ, ETG
; il clock dell'orologio è 32 volte più lento di
; quello del µP, poniamo nel ciclo l'istruzione NOP
; in modo che questo duri esattamente 32 cicli di clock
; del µP
        XORA, 00
; il bit 00 di A è ancora 1 e lo invio
        OUT(70H), A
; in modo da inviare il segnale di Reset
        LD A, (0000H)
        LD A, (0000H)
; queste due istruzioni durano esattamente 13 cicli di clock
; in modo da fare i 26 che occorrono
; vedi diagramma
```





Le operazioni LD A, 1001011O e LD B, 08H durano ciascuna 7 cicli di clock

7+7=14 $14 \times 0,4\,\mu s = 5,6\,\mu s$



* l'ultima volta DIN2 dura 5 TCK in meno poiché non fa il salto mentre l'altrove OUT fa fare read=1 dura 11 TCK per cui si avrà un ritardo di (11-5) TCK = 6 × 0,4 μs = 2,4 μs quando il segnale ed 1 lo dobbiamo far durare per altri 32 - 6 = 26 TCK

; a questo punto l'orodatario risponde con un bit ad 1
; di acknowledge che durerà 32 cicli di clock del μP

BIT 0, (HL)

NOP

; con queste due operazioni facciamo passare 12+4=16
; cicli di clock

LD B, 05_H

; dovremo prelevare 5 byte, per mese, giorno, ore, minuti,
secondi, sono altri 7 cicli ~~per cui pre~~.

LOOP: LD ~~B, 70H~~ NOP 7+8
NOP

; ~~contatore per gli otto bit~~, in totale 16+~~4~~ = 30 cicli
; il ritardo di 5,6 μs si riduce a ~~5,2~~ μs

~~ETS: IN A, (70H)~~

; preleva i bit dell'orodatario

~~RLCA~~

~~NOP~~

~~DJNZ, ETS~~

~~IN D, (70H)~~ ; 11 cicli

~~XOR A~~ ; ~~azzera accumulatore~~ 4 cicli

IN D, (C) ; 11 cicli

XOR A ; 4 cicli

LD E, A ; 4 cicli

LD A, D ; 4 cicli

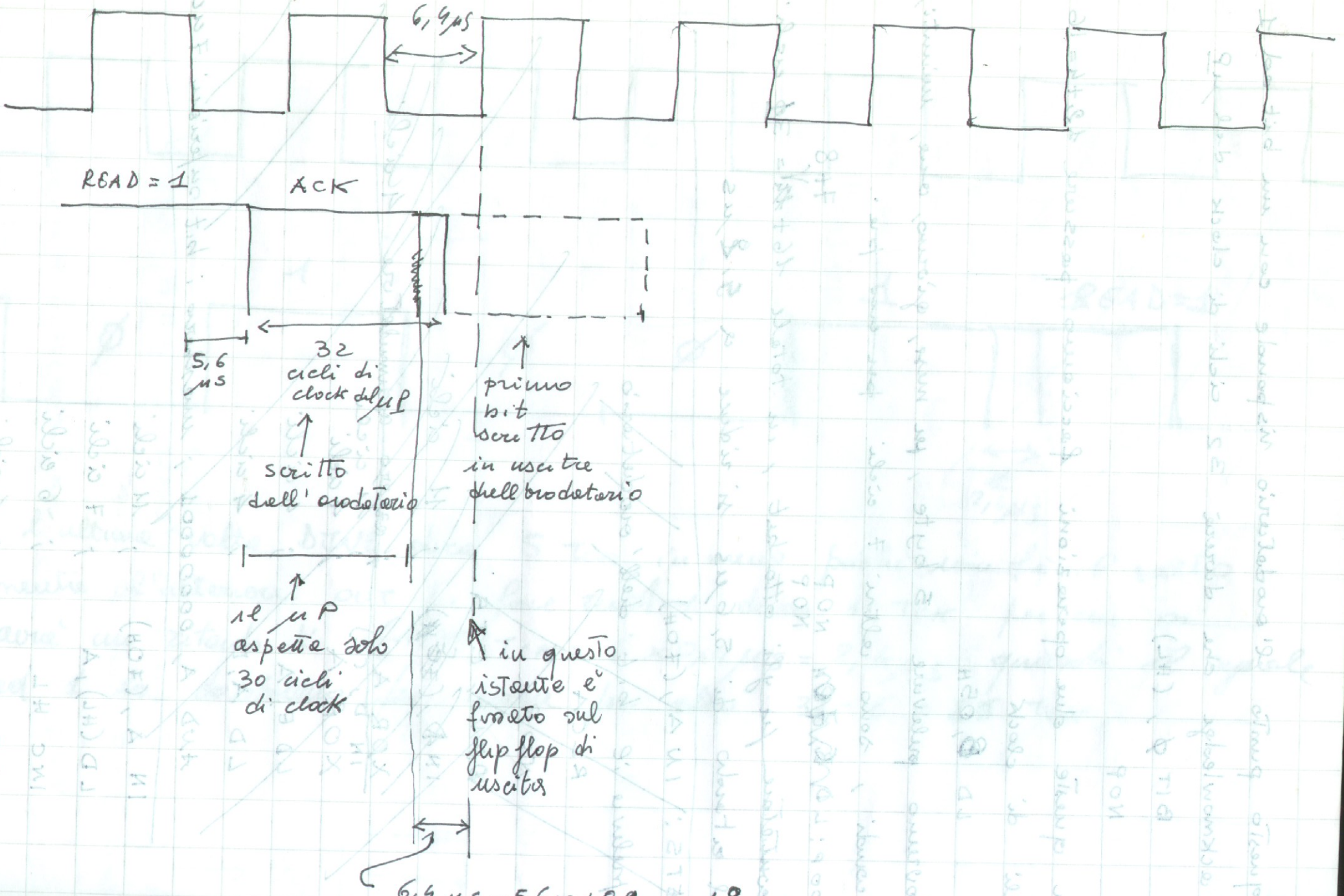
AND A, 00000001 ; mascherare i bit superiori 7 cicli

IN A, (70_H) ; 11 cicli

LD (HL), A ; 7 cicli

INC HL ; 6 cicli

NOP ; 4 cicli


6,4 µs
READ = 1
ACK
5,6 µs
32 cicli di clock del µP
primo bit scritto in uscita dell'arodatorio
scritto dell'arodatorio
il µP aspetta solo 30 cicli di clock
in questo istante è fornito sul flip flop di uscita
6,4 µs - 5,6 µs + 0,4 µs = 1,2 µs

```
; in totale 32 cicli
; dobbiamo farlo 8 volte tutti cicli altrimenti
; il tempo necessario per i salti farebbe saltare
; la sincronizzazione
    IN A, (70H)
    LD (HL), A
    INC HL
    NOP
    NOP
; terzo bit
    IN A, (70H)
    LD (HL), A
    INC HL
    NOP
    NOP
; quarto bit
    IN A, (70H)
    LD (HL), A
    INC HL
    NOP
    NOP
; quinto bit
    IN A, (70H)
    LD (HL), A
    INC HL
    NOP
    NOP
```

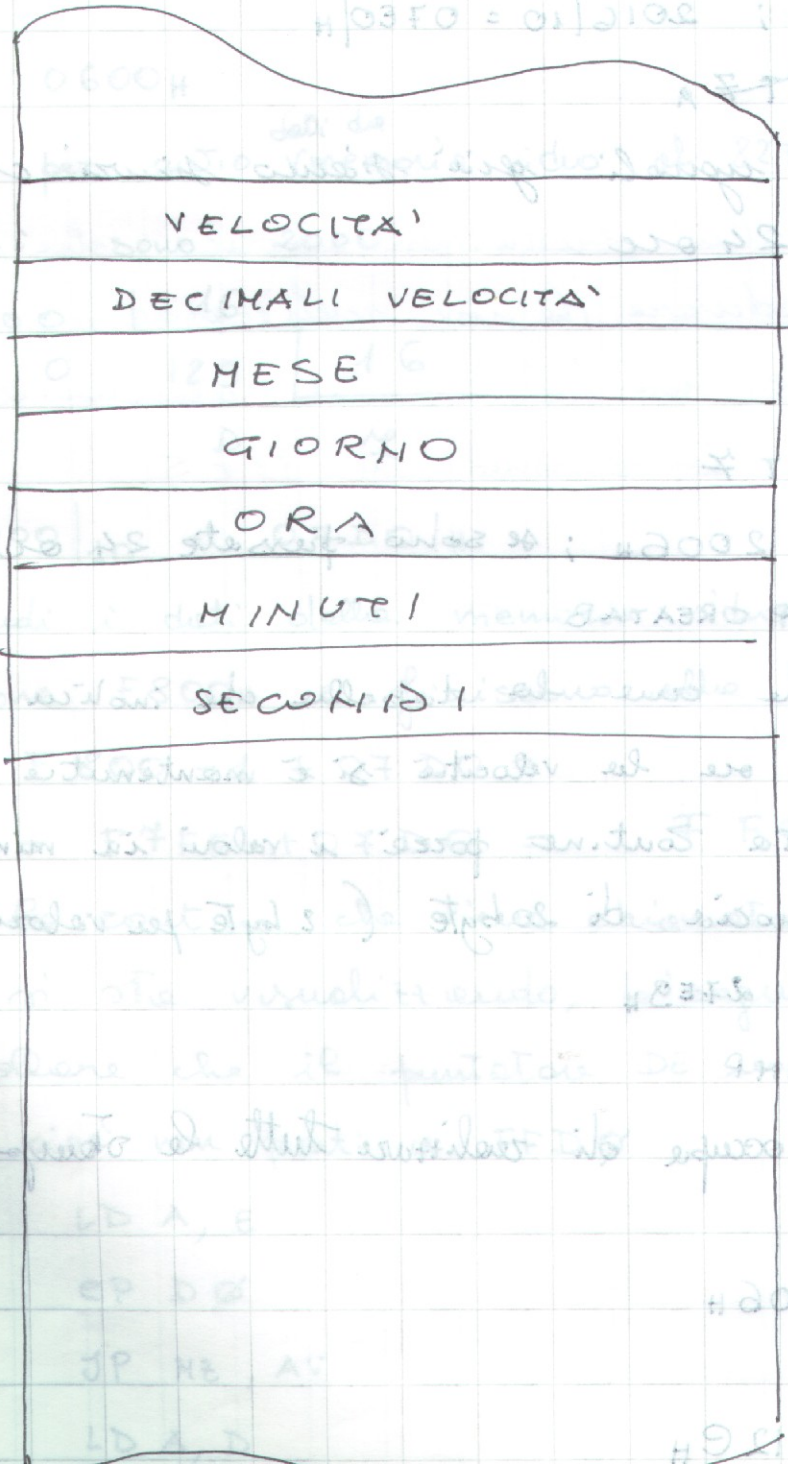
```
; sesto bit
    IN A,(70H)
    LD (HL), A
    INC HL
    NOP
    NOP
; settimo bit
    IN A, (70H)
    LD (HL), A
    INC HL
    NOP
    NOP
; ottavo bit
    IN A, (70H)
    LD (HL), A
    INC HL
    NOP
    NOP
; adesso per 32 cicli di clock di cui 16 bit
; di ACK presente
    NOP
    NOP
    NOP
    NOP
    NOP
; 5x4 = 20 cicli di clock
    DJNZ, LOOP
; 20 + 13 = 33 cicli di clock questo ciclo
```

```
        IN A,(70H)
        LD (HL),A
        INC HL
        NOP
        NOP
    ; settimo bit
        IN A,(70H)
        LD (HL),A
        INC HL
        NOP
        NOP
    ; ottavo bit
        IN A,(70H)
        LD (HL),A
        INC HL
        NOP
        NOP
    ; codice per 32 cicli di clock di cui 16 bit
    ; di ACK presenti
        NOP
        NOP
        NOP
        NOP
        NOP
    ; 5x4 = 20 cicli di clock
        DJNZ,LOOP
    ; 20 + 13 = 33 cicli di clock. questo significa
```

; che per ogni byte la lettura verrà interrotta
; di un ciclo di clock, in cui sull'ultimo byte
; si avrà un ritardo massimo di 5 cicli di clock
; che non fa perdere il sincronismo, c'è lo svantaggio
; che si occupa un byte di memoria per ogni bit

CALL FORMA-BYTE

; questa routine ricomporrà i dati temporali su
; 5 byte secondo il seguente formato



```
; dopo 24 ore saremmo stati registrati
; 2+5=7 byte ogni 5 minuti, quindi
; 7 x 60/5 = 7 x 12 = 84 byte per ore
; 84 x 24 = 2016 byte nelle 24 ore
    XOR A
; azzero il flag di carry
    SBC HL, 2006H
    LD A, L
    CP E0H   ; 2016|10 = 07E0|H
    JP NZ, ET7
; se non sono uguali già siamo sicuri che sono
; sono passate 24 ore
    LD A, H
    CP 07H
    JP NZ, ET7
    ADD HL, 2006H ; se sono passate 24 ore
    CALL CREATAB
; subroutine che crea la tabella che indica per quanto
; tempo nelle 24 ore la velocità si è mantenuta in ciascun
; intervallo, questa routine porrà i valori in minuti in
; una area di memoria di 20 byte (2 byte per valore) della
; RAM a partire da 27E9H
    CALL STAMP
; procedura che si occupa di realizzare tutte le stampe
ET7:
    LD HL, 2006H
    EXX
```

```
; 2+5 = 7 byte ogni 5 minuti, quindi
; 7 x 60/5 = 7 x 12 = 84 byte per ore
; 84 x 24 = 2016 byte nelle 24 ore
    XOR A
; azzero il flag di carry
    SBC HL, 2006H
    LD A, L
    CP E0H ; 2016|10 = 07E0|H
    JP NZ, ET7
; se non sono uguali già siamo sicuri che non
; sono passate 24 ore
    LD A, H
    CP 07H
    JP NZ, ET7
    ADD HL, 2006H ; se sono passate 24 ore
    CALL CREATAB
; subroutine che crea la tabella che indica per quan
; tempo nelle 24 ore la velocità si è mantenuta in ciascu
; intervallo, questa routine porrà i valori in minuti in
; una area di memoria di 20 byte (2 byte per valore) delle
; RAM a partire da 27E9H
    CALL STAMP
; procedura che si occupa di realizzare tutte le stampe
ET7:
    LD HL, 2006H
    EXX
    LD DE, 012CH
```

```
; 2016 | 16
;    0  126 | 16
;         E   7
;
; 2016|10 = 07E0|H
    EI
ET7: RETI

    ORG 0600H
; routine per mujo dati da memoria video al 8275
; i dati sono 2000
;  2000 | 16
;     0  125 | 16
;          D   7
;
;  2000|10 = 07D0|H
; quindi i dati della memoria video partono dalla
; locazione F800H e finiscono alla locazione
;        F800 - 1 + 07D0 =
;      = F7FF + 07D0 =   F FCF|H
; una la routine che viene chiamata non sa quale
; riga si sta visualizzando, bisogna anzitutto
; controllare che il puntatore DE non sia giunto alla
; fine cioè non punti a FFD0
        LD A, E
        CP D0
        JP NZ, AV
        LD A, D
```

```
; 2016   | 16
;   0    126 | 16
;        E    7
;
; 2016|10 = 07E0|H
;
    EI
ET7: RETI

    ORG 0600H
; routine per mujo dati da memoria video al 8275
; i dati sono 2000
;   2000 | 16
;      0 125 | 16
;            D  7
;
;   2000|10 = 07D0|H
; Quindi i dati della memoria video partono dalla
; locazione F800H e finiscono alla locazione
;       F800 -1 + 07D0 =
;       = F7FF +07D0 =   F FCF|H
; una la routine che viene chiamata non sa quale
; riga si sta visualizzando, bisogna anzitutto
; controllare che il puntatore DE non sia giunto alla
; fine cioè non punti a FFD0
        LD A, E
        CP D0
        JP NZ, AV
        LD A, D
        CP FF
```

```
    JP NZ, AV
; se non viene eseguito nessuno dei due salti siamo
; ancora in una linea intermedia, altrimenti DE deve
; puntare all'inizio della memoria video
    LD DE, F800H
AV: LD B, 19H
; leggiamo 25 byte per riempire il buffer dell'827
    LD A, (DE)
    INC DE
    DJNZ, LOOP
; ricordiamo, dalla trattazione teorica, che, quando
; il circuito hardware da noi realizzato, appena
; il µP legge un byte dalla memoria video
; questo da automaticamente all'8275
    RETI
; ~~la stesura delle subroutine~~
```