

Sistemi di Elaborazione: esercizio con il D12

Un sistema basato su PIC18F8720 a 24 MHz è dotato di 32 KB di EPROM agli **indirizzi alti** e 64 KB di RAM statica agli **indirizzi bassi**. Il sistema è dotato inoltre di un controllore di porta parallela 8255, di un controllore USB di tipo D12 e di un DMAC 8237.

Il sistema deve prelevare sequenze di 128 caratteri dalla porta A dell'8255 in Modo 1 in ricezione e trasferirle sull'endpoint 2 del D12 sotto DMA *in fly-by*.

Si progettino:

- 1) segnali di interfaccia dell'8237, dell'8255, del D12;
- 2) eventuale altra logica necessaria per il trasferimento.

Traccia della soluzione

- La modalità richiesta per il trasferimento è particolare: un trasferimento in fly-by da controllore di I/O a un altro controllore di I/O
- In realtà la soluzione è semplice se si fa svolgere al D12 il ruolo della memoria, quindi con un trasferimento DMA in fly-by di tipo “write” da I/O a memoria
- Importante: si assume che la ricezione del carattere sull’8255 avvenga in tempi imprevedibili, **notificata con un interrupt**, e che il D12 sia sempre pronto a ricevere il trasferimento (il buffer dell’endpoint 2 consente fino a 128 byte)
- Il D12 *non* è programmato in modo DMA; il D12 è destinatario di normali cicli di scrittura, di cui devono essere rispettate le temporizzazioni fondamentali
- Un aspetto della programmazione del D12: la CPU, prima dell’inizio del trasferimento, **deve selezionare l’endpoint e scrivere l’header della sequenza dei 128 byte**
- I due dispositivi D12 e 8255 hanno interfaccia a 8 bit, possono essere affacciati al solo emi-data-bus basso, programmate dal micro ai soli indirizzi pari e interfacciate con indirizzi che non debbono mai cambiare, quando pilotate dal DMA per il trasferimento dei dati (i dati vanno sempre dalla porta A dell’8255 all’endpoint 2 del D12).
- La programmazione del DMA è in **single transfer mode**, dato che arriva sull’8255 un solo byte alla volta e ogni volta (in modo 1) viene generato un interrupt dall’8255, oppure in **demand mode**, utilizzando l’EOP# per richiedere la nuova inizializzazione dell’header al micro per la nuova sequenza dei 128 nuovi caratteri che arriveranno.

DMAC 8237

I dispositivi (DMA, 8255, D12) sono tre in più ci sono due chip di memoria, sotto CPU li distinguiamo con tre bit del bus degli indirizzi:

$CS_DMAC = BA19^*/BA18^*BA17^*/HLDA$

Indirizzi, dati, comandi del DMAC: standard

DREQ0 = INTA 8255 (uso del canale 0 del DMAC per il trasferimento del dato arrivato alla porta A in modo 1)

EOP# non è collegato a nessun dispositivo: l'8255 non lo prevede, al D12 non serve, possiamo collegarlo ad un pin di IO del micro per indicare quando è terminata la sequenza dei 128 caratteri (necessaria riprogrammazione dell'header dell'endpoint 2 del D12).

Si trascuri la generazione del segnale HOLDA da parte del micro (che non prevede un DMA, semplicemente si avrà HOLD dell'8237 che va al pin INT0 del PIC18F8720 e un PIN di IO verrà utilizzato per la generazione dell'HOLDA e il micro non utilizzerà il bus durante la fase di utilizzo del bus da parte del DMA controller.

Anche il DMA può essere interfacciato al solo data bus basso e programmato e mappato ai soli indirizzi pari (vedere i driver fatti a lezione in C).

8255

$CS_{8255} = BA19 * BA18 * BA17 * HLDA + DACK0$

$A1 = AD1 * HLDA$ (con $HLDA = 1$, $A1 = 0$)

$A0 = AD0 * HLDA$ (con $HLDA = 1$, $A0 = 0$)

$RD = RD * HOLDA + IORD * HOLDA$ (di sistema, ottenuti dalla fusione dei comandi micro e 8237)

$WR = WR * HOLDA + IOWR * HOLD$ (eventualmente anche solo WR , visto che non deve essere acceduto in scrittura in DMA)

Di al bus dati basso di sistema

INTA dell'8255 in modo 1 sulla porta A al DREQ0 8237

D12

$CS_D12 = BA19^*/BA18^*/BA17^*/HLDA + DACK0$

$A0 = AD0^*/HLDA$ (con $HLDA = 1$, $A0 = 0$, scrittura dati)

$RD = RD^*/HOLDA + MEMRD^*HOLDA$ (di sistema, ottenuti dalla fusione dei comandi micro e 8237)

$WR = WR^*/HLDA + MEMWR^*HLDA$ (con $HLDA = 1$, il controllo delle linee di comando è del DMA e il D12 è visto come il lato memory del trasferimento)

Di al bus dati basso di sistema

INT, EOT#, DREQ, DACK# non utilizzati

Temporizzazioni D12: 8237 a 5 MHz: 600 ns di tempo di ciclo di bus > 500 ns di tempo di ciclo di scrittura del D12 (quindi non ci sono problemi di tempi di wait, non si deve progettare il segnale di ready).

Il clock è 6MHz, poi moltiplicato a 48 interni.

Il clock generato per il micro deve essere 24 MHz, quindi $N=1$ nella formula del calcolo del CLOCKOUT in uscita dal D12.

RAM ed EPROM

$CS_RAM = \overline{BA19} \cdot \overline{BA18} \cdot \overline{BA17} \cdot \overline{HLDA}$

$RD_RAM = \overline{RD} \cdot \overline{HOLDA}$

$WR_RAM = \overline{WR} \cdot \overline{HLDA}$

$CS_EPROM = BA19 \cdot BA18 \cdot BA17 \cdot \overline{HLDA}$

$RD_EPROM = \overline{RD} \cdot \overline{HOLDA}$

$WR_EPROM = \overline{WR} \cdot \overline{HLDA}$

Sia ram che eprom non sono interessate nei cicli di DMA quindi non debbono mai essere selezionate dai CS