

Laboratorul 10 - Calculatorul Didactic: Subsistemul de I/E

Periferele reprezintă o componentă importantă a unui calculator, fără acestea interacțiunea cu utilizatorii fiind foarte dificilă. În lipsa lor introducerea datelor și citirea rezultatelor ar fi necesar să fie făcute manual prin scrierea și respectiv citirea directă a memoriei. Deși această metodă a fost folosită pentru primele modele de calculatoare numerice, unde datele și programul erau introduse prin activarea unui comutator pentru fiecare bit, ea a fost rapid înlocuită de apariția primelor periferice care puteau citi conținutul memoriei de pe cartele perforate și puteau afișa rezultatele prin tipărirea acestora la imprimantă. În zilele noastre orice calculator este dotat cu periferice care permit interacțiunea ușoară cu utilizatorul.

Interfața cu subsistemul de I/E

Calculatorul didactic a fost proiectat având în vedere necesitatea interacțiunii cu utilizatorul. El dispune de un subsistem de intrare/ieșire, al cărui scop este de a permite conectarea de diferite periferice. Orice dispozitiv care respectă interfața de comunicare ar putea fi conectat la calculatorul didactic. Această interfață este asemănătoare cu modul de comunicare cu memoria principală (*ram*).

În mod similar cu memoria, calculatorul generează pentru periferic 2 semnale de comandă: *io_oe* - *output enable* și *io_we* - *write enable* pentru a efectua o operație de scriere sau citire de la periferic. Magistrala calculatorului este disponibilă perifericului pentru a trimite și primi informații, în implementarea Verilog acestea fiind făcute prin două semnale separate: *io_in* - pentru a transmite către periferic și *io_out* - pentru a recepționa de la periferic. În final, deoarece dorim să putem conecta mai multe periferice la calculator în același timp, interfața conține și un semnal de adresă pe 8 biți, numit *io_port*. Acesta este similar cu adresa trimisă memoriei, permițându-ne să accesăm 256 de locații, numite porturi de I/E. Aceste porturi reprezintă niște concepte logice. Ele nu sunt neapărat reprezentate prin porturi fizice ale calculatorului didactic. În implementarea scheletului din acest laborator există un singur port fizic, conectat la modulul *Lcd*.

Pentru a realiza comunicarea efectivă cu un dispozitiv periferic cele 256 de locații sunt împărțite între dispozitive, astfel încât 2 dispozitive diferite să aibă alocate porturi diferite. În momentul în care calculatorul vrea să acceseze un dispozitiv, este obligatoriu ca doar dispozitivul al cărui port se află pe liniile de adresă să răspundă la interogare. Pentru simplitate, în calculatorul didactic alocarea porturilor se face static, fiecare dispozitiv având alocat în prealabil anumite adrese pentru porturile sale. În implementările comerciale, acest lucru se face însă dinamic printr-un protocol de descoperire a dispozitivelor la pornire.

De obicei un dispozitiv are alocate 3 porturi: unul de date, unul de comenzi și unul de stare. Prin intermediul *portului de date* se trimit și se primesc date de la dispozitiv. Acesta sunt dependente de dispozitiv și ar putea reprezenta datele scrise de o imprimantă sau datele primite de un port serial. Prin intermediul *portului de comenzi* se trimit în principal comenzi către dispozitiv. Acestea pot reprezenta comenzi de poziționare a capului de citire pentru imprimantă sau de configurare a parametrilor de transmisie pentru un port serial. A se observa că spre deosebire de operațiile cu

portul de date, operațiile cu portul de comenzi se traduc în diferite acțiuni ale dispozitivului, biții primiți pe acest port fiind decodificați într-o anumită comandă. În final, *portul de stare* este folosit de obicei pentru a citi informații despre starea dispozitivului, care pentru o imprimantă ar putea semnala programului din calculator starea capului de scriere și a tăvii de hârtie, iar pentru un port serial ar putea reprezenta configurația curentă de transmisie sau diferite flag-uri pentru evenimentele de transmisie.

Împărțirea porturilor alocate unui dispozitiv în port de date, de comenzi și de stare, reprezintă însă o convenție de comunicare între program și respectivul dispozitiv. Din punct de vedere al unei implementări a calculatorului informațiile transmise prin porturi fiind doar biți, fără o semnificație anume. Aceștia ar putea conține o comandă trimisă print-un port de date, sau ar putea ca la citire portul de comenzi să returneze starea dispozitivului.

Din punct de vedere al programatorului comunicarea cu un dispozitiv periferic se face prin intermediul instrucțiunilor `IN <port>` și `OUT <port>`, care citesc, respectiv scriu, conținutul registrului RA la portul specificat prin operandul `<port>` al instrucțiunii.

Citirea de pe dispozitiv

Calculatorul didactic execută o citire de pe dispozitiv prin instrucțiunea `IN`. Fig. 1 prezintă semnalele implicate în momentul executării unei citiri de pe dispozitiv. Se observă cum în timpul execuției instrucțiunii `0x4001` (decodificată în `IN 2`), în starea `0x004d`, calculatorul activează semnalele subsistemului de I/E necesare pentru citire de pe dispozitiv: `io_oe` și `io_port`, acesta din urmă trimițând valoarea portului primită în instrucțiune, spre dispozitiv. Dacă dispozitivul își recunoaște vreuna dintre adresele porturilor de citire, el va răspunde în același ciclu de ceas prin trimiterea datelor spre calculator prin semnalul `io_out` (direcția *in/out* este relativă la periferic), acest semnal fiind legat la magistrală. Valoarea este reflectată și pe semnalul `io_in` care este de asemenea legat la magistrală. `io_in` nu este însă implicat în operația de citire de pe dispozitiv.

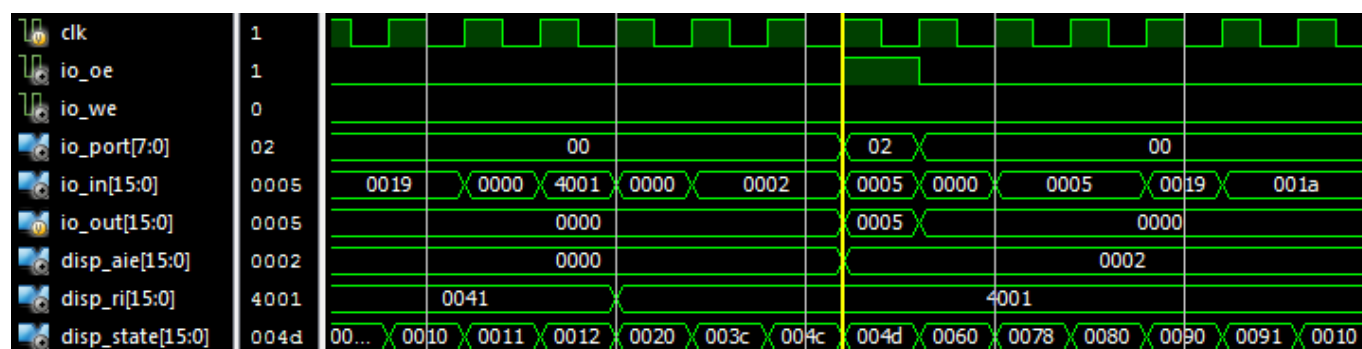


Fig. 1: Citirea de pe dispozitiv

Scrierea pe dispozitiv

Calculatorul didactic execută o scriere pe dispozitiv prin instrucțiunea `OUT`. Fig. 2 prezintă semnalele implicate în momentul executării unei scrieri pe dispozitiv. Se observă cum în timpul execuției instrucțiunii `0x8041` (decodificată în `OUT 1`), în starea `0x008c`, calculatorul activează semnalele subsistemului de I/E necesare pentru scriere pe dispozitiv: `io_we`, `io_port` și `io_in`. `io_port` trimite valoarea portului primită în instrucțiune, iar `io_in` trimite datele care sunt scrise. Dispozitivul

trebuie să reacționeze la aceste semnale, dacă își recunoaște vreuna dintre adresele de porturi alocate lui. Reacția trebuie să aibă loc în acel ciclu de ceas, deoarece aceasta este durata pentru care calculatorul menține active semnalele de scriere.

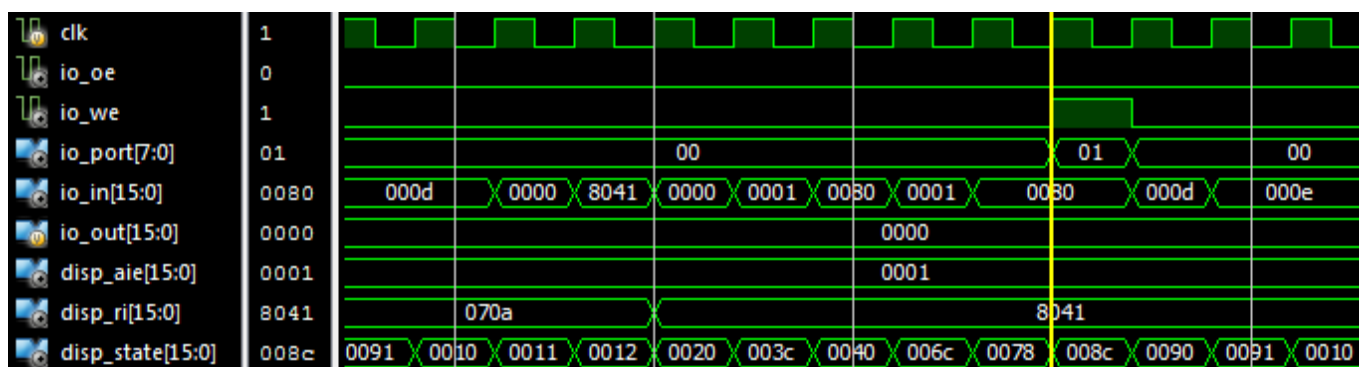


Fig. 2: Scrierea pe dispozitiv

Exerciții

Se dorește conectarea unui display LCD text la calculatorul didactic sub forma unui dispozitiv periferic care poate fi controlat de un program rulat pe calculator. Avem la dispoziție un modul, `lcd_display`, care poate controla LCD-ul text prezent pe placa de laborator. Acest modul primește ca intrare un vector de 32 de valori pe 8 biți, data, care reprezintă codurile ASCII ale caracterelor ce vor fi afișate pe cele 2 linii ale ecranului.

Deoarece folosirea a 32 de porturi din spațiu de I/E al calculatorului didactic pentru un singur periferic este excesivă, vom implementa alocarea obișnuită de porturi pentru un dispozitiv periferic: de date, de comandă și de stare. Pentru aceasta vom avea nevoie de un modul, `lcd`, care să traducă operațiile asupra celor 3 porturi în modificări ale celor 32 de caractere.

Funcționalitățile oferite de cele 3 porturi sunt următoarele:

- portul de date (adresa 0):
 - la scriere suprascrie valoarea caracterului de pe poziția curentă a cursorului cu biții 7..0 ai valorii primite de la calculator
 - la citire returnează o valoare pe 16 biți care conține 0 în biții 15..8 și valoarea caracterului de pe poziția curentă a cursorului în biții 7..0
- portul de comenzi (adresa 1):
 - scrierea valorii 0 șterge conținutul ecranului
 - scrierea unei valori cu bitul 7 egal cu 1 folosește biții 4..0 pentru a suprascrie poziția cursorului
 - scrierea oricărei alte valori este ignorată
 - citirea portului este ignorată
- portul de stare (adresa 2):
 - scrierea portului este ignorată
 - citirea portului returnează o valoare pe 16 biți care conține 0 în biții 15..5 și poziția cursorului în biții 4..0
- 1. (3p) Implementați funcționarea portului de comandă.
 - Hint: ștergerea ecranului se poate face prin scrierea caracterului (spațiu - cod ASCII în hexazecimal 0x20) în toate cele 32 de locații ale ecranului
 - Hint: Cursorul se poate implementa printr-o variabilă a modulului care este modificată în

momentul primirii comenzii corespunzătoare.

- 2. **(3p)** Implementați funcționarea portului de stare.
 - Hint: Starea este dată de valoarea cursorului
 - Hint: Valoarea trebuie returnată în același ciclu de ceas în care s-a primit cererea de la calculator. Veți avea nevoie de un bloc always combinațional pentru a implementa acest lucru.
- 3. **(4p)** Implementați funcționarea portului de date.
 - Hint: Folosiți poziția cursorului pentru a modifica corespunzător datele trimise la modulul `lcd_display`.
 - Hint: Pentru o cerere de citire, returnarea valorii se face în același ciclu de ceas.
- 4. **(1p)** Testați funcționarea perifericului pe placa de laborator. Programul aflat în memoria calculatorului folosește instrucțiunea OUT cu porturile de date și de comenzi ale perifericului pentru a afișa mesajul Hello World pe ecranul LCD-ului.

Resurse

- [Schelet de cod](#)
- [Soluție laborator](#) (disponibilă începând cu 14.12.2018)
- [Surpriză](#)
- [PDF laborator](#)
- [Manualul plăcii de dezvoltare](#)

-
- [Ghid asistent](#)

From:

<https://elf.cs.pub.ro/ac/wiki/> - **AC Wiki**

Permanent link:

<https://elf.cs.pub.ro/ac/wiki/lab/lab10>

Last update: **2018/12/06 11:43**

