Facultatea de Automatică și Calculatoare

Universitatea Politehnica Timișoara

Politehnica University of Timișoara

Facultatea de Automatica si Calculatohas

Microprocessor Design

Implementation of a Microsystem with the 8086 Microprocessor

Created by:

Dobra Mihai

Academic Year 2024-2025

Project Theme

Proiecthasa a microsystem based on microprocesorul 8086, which includes central unit, memories EPROM și SRAM, interfaces seriale și paralele, mini-keyboard, LEDs, 7-segment display și LCD modeule, as well as programs for configuration and operation acestuia.

Microsystem Structure:

- central unit cu microprocesorul 8086;

- 128 KB EPROM memory, using circuits 27C512;

- 64 KB SRAM memory, using circuits 62256;

- serial interface, cu circuitul 8251, plasată în zona 04D0H – 04D2H sau 05D0H – 05D2H, depending on position microswitch S1;- parallel interface, cu circuitul 8255, plasată în zona 0250H– 0256H sau 0A50H – 0A56H, depending on position microswitch S2

;- o mini-keyboard cu 9 contacts;

- 10 led-uri;

- un modeul de display cu 7 segmente, cu 8 ranks (se can afişa maximum 8 characters hex simultan);

- un LCD modeule, cu 2 lines a câte 16 characters each, cu o interfaţă la choice student

Programs:

- routines de programhas ale circuitslor 8251 şi 8255;

- routines de transmission/ recepţie character pe interfaţa serială

;- routine for transmission character pe parallel interface;

- routine for scanning a minitastaturii;

- routine for turning on/ turning off a an LED;

- routine for display a a character hex on a rank with segments

Hardwhas Description

Controle Unit

Unitatea centrală a microsystemului proiectat has la bază microprocesorul 8086, un device pe 16 bits, utilizat for controle și processing given. Acest microprocesor oferă functionalities advanced, incluzând buses separate for given și addresses (viatr-o technology de multiplexing), modees de addressing various și un set complex de instructions.

Microprocessor Configuration and Working Mode

the 8086 Microprocessor funcționează în două modees main which se can set via pin MN/MX#

- **Minimum Mode:** In acest mode microprocesorul manages el itself signals de controle, activeatedion memories si devices de input/output, plus ca un has need de un bus external. Este modeul pe which is used in proiect because is suitable for systems simple si de that is connected la VCC, for a fi activeated MN pe “1” logical.
- **Maximum Mode:** Permite use a coprocessor sau a unor peripherals complex, microprocesorul having un role central de coordination.

Main Signals of the Microprocessor

- **AD0-AD15 (Multiplexed Bus):** Utilizează same pins for given și addresses, reducing number de connections required.
- **A15-A19:** Pini din ranges de magistrala de addresses additional.
- **BHE# (Bus High Enable):** Indica whether has sau nu place un transfer pe half upper a bus de given.
- **ALE (Addres Latch Enable):** Activează storage addresseslor din magistrala multiplexed.
- **RD# si WR#:** Controleează operations de read/input respectiv write/output, synchronizing access la buses.
- **M/IO# (Memory/Input-Output):** Determina whether operation current is intended memoriei (when has value 1) sau a device via ports de input/output (value 0)
- **DT/R# (Data Transmit/Receive):** Ne shows direction transfer de pe magistrala de given (“1” indicates transmission si ”0” indicates reception)
- **DEN# (Data Enable):** Validează transferul de given pe magistrala.
- **RESET:** Pentru reset/initialization processor.
- **CLK:** Este input signal cplacek, in acest case cu o frequency given de device 8284A described mai low.
- **READY (Wait State Controle):** Intrhas for synchronizing microprocessor si devices slow, when is active (level high) microprocesorul continues operation current. Când is inactivee (level low) el enters într-o state de wait (wait state) until when peripheral sau memory completes operation.

Registrul de given 74x373

Acest device cu 8 ranks is utilizat for stowhicha temporara a given, having trei stări de funcționhas (active, neactive si a 3-a state). In lucrhas sunt amplasate 3 registre which sunt utilizate for a memora addressesle multiplexate de la microprocesor plus semnalul BHE#. Pinul

G (Enable) de la each latch sunt legate la semnalul ALE for a sincroniza capturhasa addresseslor din magistrala multiplexată, astfel încât addressesle stocate sa fie corect stabilite înainte de începerea oricărei operații de citire/scriere. Pinul OC# controle activeatedion ieșirilor latch-ului si for a funcționa pe cele doua stări active/neactive trebuie connected la masa (0 logical).

In primul registru (cu identificatorul U6) stochează bits de la A16 la A19 si semnalul BHE. Al doilea (U12) stochează bits de la AD0 la AD7. Si al treilea (U13) stochează bits de la AD8 la AD15.

Circuitul Amplificator/Separator bidirecţional 74x245

Este un circuit which allows transferul bidirecțional de given intre buses. In proiect sunt usede doua, each cu 8 ranks, for a gestiona transferul de given a 8086 si restul componentelor. Pinul DIR (Direction) determines direcția fluxului de given intre cele doua buses si in proiect îl controleam cu semnalul DT/R#. Când citim fluxul de given is direcționat de la peripherals/memorie spre microprocesor si la scriere is inversat. Pinul G# is la fel ca la 74x373 doar ca in caseul acesta is legat la pin DEN# described mai sus.

Primul circuit 747x245 (U14) has intrările cu ranges AD0 la AD7 din magistrala de given/addresses, si cu ieșirile D0-D7 din magistrala de addresses. Circuitul al doilea (U15) cuviade ranges de entersre AD8-AD15 si ieșirile la magistrala D8-D15.

Geratorul de Cplacek 8248A

Acesta generează semnalul de ceas necesar sincronizări operațiunilor interne ale microprocessor si asigura stabilitatea întregului sistem. Generatorul primește semnalul brut de la oscilatorul de cristal, which is connected la el, si îl prelucrează oferind un semnal stabil pe which il va folosi microprocesorul. In plus, furnizează si semnale auxilihas precum READY si RESET which le folosește microprocesorul.

In proiectul nostru (U4) la pins X1 si X2 se conectează cristalul de cuarț de 15MHz plus doua condensatohas de 20pF connectede la masa for a stabiliza oscilația. Pinul EFI (External Frequency Input) is connected la masa for ca nu vom folosi un semnal de cea external. Si la fel F/C# (Frequency/Crystal Select) selectează intre cristal si o sursa externala si in acest case va fi legat la masa (selectând modeul cristal). Si CSYNC (Cplacek Synchronization) va fi legat la masa because el sincronizează generatorul cu alte surse de ceas externale, cea ce nu avem in proiect. La ieșire avem signals CLK, RESET si READY which ajung la intrările din 8086. Semnalul PCLK is o versiune mai lenta a signal CLK which se folosește for a sincroniza peripheralsle which funcționează la o frequency mai mica, si which ne va ajuta la interfaces. Pinul RES# is input for semnalul de resetre external which îl legam la un buton de resetre si un circuit RC si o dioda for a asigura o resetre corecta la pornirea sistemului.

Conecthasa Memoriilor

Proiectul utilizează doua tipuri de memories:

- **128 KB de EPROM memory,** implementata cu doua circuits 27C512 (U9 si U10 , 64 KB flewhich)
- **64 KB de SRAM memory,** implementata cu doua circuits 62256 (U7 si U11, 32 KB each)

Planifiwhicha spațiului de addressing

the 8086 Microprocessor dispune de o magistrala de adresa pe 20 de bits, permițând acceshasa a spațiu maximum de 2^20 = 1MB. Deci toate addressesle posibile for acest procesor cuviad rangul: 00000H-FFFFFH. Când se construiește harta memoriei trebuie stabilite doua given for each circuit (bplace) de memorie, adresa de început si limitele de addressing.

- Pentru SRAM spațiul începe de la adresa 00000H, iar limita upper is determinesta de adăughasa capacității memoriei (64 KB), rezultând) 0FFFFH.
- Pentru EPROM am ales adresa de început 20000H, alowhen următorii 128KB. (20000H + 1FFFFH = 3FFFFH).

Acis ranks pe which le am ales ne va ajuta la decodifiwhicha memories.

Memoria SRAM 62256

SRAM is utilizat for stowhicha given temporhas required procesării, having operații rapide de citire si scriere. Este o memorie volatila which își pierde informația o given ce alimenthasa is întrerupta si stowhicha given se realizează via bistabile de tip flip-flop which nu necesita de operați periodice for refresh. Capacitatea chipului is de 32 KB necesitând 2 chipuri for a ajunge la capacitatea de 64 KB. In plus, se face împărțirea in 2 chipuri for ca procesorul 8086 lucrează cu addresses phas respectiv imphas, de that can sa leg un bplace pe half superioară a bus de given respectiv half inferioara. Utilizează 15 bits de adresa (A0-A14). Conexiunile main:

- **WE# (Write Enable):** is pin which activeatedes scrierea de given si each chip SRAM has un semnale WE# diferit for a asigura access separat la addressesle phas si imphas. Sunt generate de un circuit logical combinațional (U16) format din doua porți sau cu signals WE, BHE si A0. Prima poarta sau combina semnalul BHE# which activeatedes un semnal pe magistrala upper de given D15-D8 cu WR# de la procesor, generând semnalul WE# for bplaceul de memorie which manages addressesle imphas (U7). Si a doua poarta ia semnalul WR# de la procesor si A0 which is bitul cel mai puțin semnificativ de la o adresa si whether is 0 e para, whether e 1 is impara, generând semnalul WE# for bplaceul which manages addressesle phas (U11). Deci procesorul poate sa scrie independent un octet superior in bplaceul (U7), un octet inferior in bplaceul (U11) sau ambii octeți simultan.
- **OE# (Output Enable):** Activează ieșirile for citire si is connected la pin RD# de la 8086.
- **CE# (Chip Enable):** Activează circuitul for acces la given si is connected la o ieșire al decodificatorului de memories.

Memoria EPROM 27C512

EPROM is utilizat for stowhicha codului programului access fiind exclusiv de citire. Fata de SRAM acest tip de memorie is non-volatila iar ștergerea given se face doar via expunerea la radiații ultraviolete. Capacitatea chipului is de 64 KB necesitând 2 chipuri for a ajunge la capacitatea de 128 KB. Utilizează 16 bits de adresa (A0-A15). Conexiunile main:

- **OE# (Output Enable):** La fel ca la SRAM.
- **CE# (Chip Enable):** La fel ca la SRAM.
- **VPP (Voltage Programming Pin):** După ce memory a fost programata pin VPP trebuie connected la VCC because in thatsta configurație memory va funcționa doar in modeul citire, iar givenle programate rămân intacte.

Decodifiwhicha memories

Pentru a genera signals de selecție corect for each memorie in funcție de adresă, am implementat un decodificator based on un circuit 74LS138 si logicala combinațională. Se construiește un tabel which has ca si coloane ranges bus de addresses iar ca rânduri prima si ultima configurație de adresa a fiecărei tip de memorie.

Adresa (Hex)	A19	A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	Memorie
0000fh	**0**	**0**	**0**	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SRAM
0ffffh	**0**	**0**	**0**	**0**	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
20000h	**0**	**0**	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	EPROM
3ffffh	**0**	**0**	**1**	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	

Verde: Spațiul de addresses for SRAM

Albastru: spațiul de addresses for EPROM

Gri: Ranguri neschimbate, constante pe parcursul addresseslor

Decodifiwhicha Completa a semnalelor de selecţie for memories

Decodifiwhicha Completa presupune use tuturor bitslor de adresa relevanți for generhasa semnalelor de selecție. Pornind de la rangul cel mai semnificativ spre cel mai puțin se reține acele ranks which rămân nemodeificate for orice configurație de adresa din zona respectiva. In caseul proiectului, ranges relevante for SRAM sunt A19, A18, A17,A16, iar for EPROM sunt A19, A18 si A17. Deci funcțiile combinaționale ar fi:

$!Sel_{SRAM} = !A19 * !A18 * !A17 * !A16$

$!Sel_{EPROM} = !A19 * !A18 * A17$

Si ne ar trebui un decodificator 4 la 16 in which SRAM sa fie active doar pe combinația 0000 la input decodificatorului iar EPROM va fi active in combinațiile 001X. Sunt multe ieșiri neusede din decodificator plus multe which activeatedes EPROM-ul de in proiect se folosește o decodifiwhich incompleta.

Decodifiwhicha Incompleta a semnalelor de selecţie for memories

Observam ca sunt multe ranks which sunt egale for SRAM si EPROM (A19 si A18), putem sa ignoram acis ranks si sa simplificam decodificatorul. Acum ne ar trebui doar un decodificator 2 la 4.

$!Sel_{SRAM} = !A17 * !A16$

$!Sel_{EPROM} = A17$

Problema is ca un circuit „va fi văzut de procesor” in mai multe zone din spațiul de addressing al processor, for ca cele doua ranks ignorate vor fi „dont-which” si in acest case va ocupa 4 zone egale de memorie cu dimensiunea respectiva.

Se putea folosi in acest case si un circuit simplu logical combinațional, in place de decodificator, cu o poarta si si un invertor for a selecta intre cele doua memories doar cu ranges A17 si A16

Circuitul decodificator 74LS139

In caseul asta, decodificatorul 74LS139 (U) is configurat for a genera semnale de selecție for memoriesle EPROM si SRAM. Când microprocesorul transmite o adresa pe magistrala,

decodificatorul si logicala combinațională determines memory which trebuie activeateda. Are 2 intrări si 4 ieșiri duplicate, adică sunt doua decodificatohas 2 la 4 in același bplace. Am omis unu din ele for ca is necesar doar de unul singur. La intrările 1A si 1B se leagă A16 respectiv A17. Pini Enable funcționează ca un „switch” in which se spune whether decodificatorului is on/off. Pini E1A# si E2B# vor fi totdeauna conectați la masă for ca decodificatorul va funcționa tot timpul timp ce E3 is connected la M/IO# de la 8086 si va activea cu „1” logical funcționhasa decodificatorului. Tabelul de adevăr for ieșirea decodificatorului is următorul:

A17	A16	Output
0	0	1Y0# (Ajunge la CE# de la SRAM)
0	1	1Y1# (Nu is necesar)
1	0	1Y2# (Ajunge la CE# de la EPROM)
1	1	1Y3# (Ajunge la CE# de la EPROM)

Se observa ca 1Y2# si 1Y3# sunt usede for același scop. Pentru a rezolva acest conflict se plasează o poarta Si astfel încât when una din ele is activea va fi pe “0” logical activeând memoriesle EPROM

Nu is rezolvhasa cea mai perfecta, se putea face cum am indicatest mai sus fără un decodificator, cu o poarta sau si un inversor for a nu încarcă mai mult complexitatea proiectului.

Decodificatorul de porturi (interfete)

Interfețele seriala si paralela

Circuitul 8251 (Interfaţa Seriala)

Convertește givenle paralele (din procesor) in given seriale for transmission, sau primește given seriale si le transforma in paralele (U17). Permite transferul de given intre systems which se afla la distante mari unul de altul folosind un număr redus de fire. Exista doua feluri de comuniwhich, cea sincrona in which se folosește o linie de tact comun, si cea asincrona in which trebuie adăugată informație suplimentara la octetul which urmează a fi transferat definind un standard for transmission (Bit de start, bits de stop, factor de multipliwhich)

Pentru configurhasa deviceui se folosește Cuvântul de mode which configurează modeul de operație(bits de given, paritate, stop bit, sincronizhas) si cuvânt de comanda which controle statea de transmission si reception (începere, oprire).

- **Cuvânt de Mod**: In proiect se configurează astfel încât sa funcționeze in mode asincron si de that has 2 bits de stop (11) in caseul asta, fără controle de paritate (00) adică nu se adaugă bit suplimentar for verifiwhicha parități, 8 bits de given (11) unde each character transmis sau recepționat has 8 biti, si un factor de multipliwhich x16 (10) which înseamnă ca rata de transfer a signal de cplacek is împărțita for a genera semnalul intern astfel încât sa fie de 16 ori mai rapid decât baund rate-ul dorit.

In binar -> 11001110 In Hexa -> 0CEH.

- **Cuvânt de Comanda:** bitul de sincronizhas (0) which e used doar in mode sincron for a găsi characterul de sincronizhas, bitul de reinițializhas (0) which resetează statea interna a deviceui, bitul de comanda RTS (0) is un semnal used for controle fluxului de given. Bitul de error reset (1) which resetează bits de erohas, bitul de break (0) transmite un semnal BREAK which reprezintă o state continues de 0 logical pe linia de transmission (TXD), utilizata for sincronizhas sau semnalizhasa unei condiții speciale, bitul de comanda reception (1) which allows 8251 sa primească given via linia RXD,

bitul de comanda DTR (0) which is used for a indicates faptul ca terminalul de given is gata for comuniwhich, si bitul de comanda transmission (1) which allows transmisia de given de la 8251 via lina TXD.

In binar -> 00010101 In Hexa -> 15H.

- **Pini importanți: C/D#** (Comand/Data): selectează intre portul de given (0 logical) si portul de comanda/stări (1 logical), in caseul asta is legat la A1 de pe magistrala de addresses si se explica for ca avem asignata zona de addresses 04D0H -04D2H sau 05D9H -05D2H, in funcție de poziția microswitch S1, in pereche addressesle diferă via bitul de pe rangul A1 cum se observa in tabelul de mai low. Pinii **WR#** si **RD#** controle operations de scriere si citire si sunt legați direct la 8086. Pinul **CS#** is connected la ieșirea comutatorului S1 (U20) si el activeatedes sau dezactiveatedes circuitul. Pinul **RST** is legat la RESET de la 8284A si **CLK** is legat la PCLK, am explicat roleul lui înainte.

- **Pini de reception**: in caseul acesta in proiect doar avem connected doar pin RXD la input R1OUT de la MAX232 because via acis recepționează givenle bit cu bit. RXC# is used in modeul sincron for a sincroniza recepția given cu un semnal de cplacek. RXRDY is used in handshake (sincronizhas cu alte devicee) for ca el indicates faptul whether 8251 has given disponibile for citire. Si SYNDET in mode sincron indicates detecthasa a character de sincronizhas iar in mode asincron poate fi utilizat ca input/output programabila for controle.
- **Pini de transmission**: la fel ca la reception, in proiect se folosește doar pin TXD for a transmite givenle bit cu bit la pin T1IN al MAX232. Pini TXC# si TXRDY funcționează la fel cu cei de la reception doar ca for transmitere de given. Pinul TXE indicates statea buffer-ului, adică whether toate givenle din el au fost transmise si buffer-ul is gol.
- **Pini de controle for handshake** sunt usede for asigurhasa ca givenle sunt transmise si recepționate doar when ambele parți sunt pregătite(DSR#, DTR#, CTS# si RTS#)dar nu se folosesc in proiect.

Prin pins TXD si RXD is connected un transceiver RS-232 (MAX 232 - U47). Este un circuit de conversie de level logical used for a face compatibile levelurile TTL/CMIS de la 8251 cu standardul RS-232 used in multe devicee seriale. Este connected la el condensatohas for a asigura conversia corecta a levelurilor de tensiune de la TTL/CMOS spre RS-232 si viceversa. De la TTL la RS-232 va transforma semnalul cu o Amplitudine mai mhas si inversat. De exemplu: -> TTL (0V – 5V) la RS-232 (-12V - +12V).

Circuitul 8255 (Interfaţa Paralela)

Este utilizat for conecthasa deviceelor peripherals paralele, cum ar fi tastaturi, afișaje (cum is caseul cu un ecran LCD), sau alte modeule I/O. (U22) EL allows comuniwhicha bidirecționala intre procesor si peripheralsle transmițând sau receptând given in mode paralel via intermediul celor 3 porturi de cate 8 bits each (Port A, Port B si Port C), unde each poate funcționa ca entersre sau ieșire. In total ar fi 24 de lines I/O which can fi programate individual sau in grupuri. Are trei modees de operhas, in **Modul 0** (input/output de baza), which is cel used in proiect, each port poate fi configurat ca entersre sau ieșire si nu exista semnale de strobe sau handshake. **Modul 1** (entersre/iesire sincronizata) se folosesc ports A si B for I/O cu posibilitatea de handshaking iar portul C e used for semnale de controle. **Modul 2** (Transfer bidirecțional) Permite transmiterea bidirecționala de given, disponibil doar for portul A ,using linesle din Port C for controle direcție.

Pentru configurhasa deviceui se folosește **cuvântul de controle** which is configurat for a lucra cu ecranul LCD (U45) din proiect. Cel mai semnificativ bit is bitul de which specifica tipul de cuvânt de controle 1 is modees funcționale sau 0 for set/reset port C, așa ca se lasă pe 1 logical. Următori 2 bits sunt for a selecta modeul de operhas al grupului A, in caseul asta va fi modeul 0 (00). Portul C superior which cuviade pins (PC4-PC7) va fi configurat ca ieșire (0). Modul de funcționhas al grupului B va fi mode 0 (0). Portul B va fi sett ca ieșire (0). Portul C inferior which cuviade pins (PC0-PC3) va fi configurat ca ieșire (0). In binar -> 10000000 In Hexa -> 15H.

- **Pini importanți:** Pini **WR#**, **RD#, RESET** si **CS#** au același role ca la 8251 doar ca CS# is legat la microcomutatorul S2 (U21). Pini **A0** si **A1** sunt entersrile which decid ce registru intern al circuitului sa fie selectat si funcționează ca un decodificator 2-la-4 intern in which primește cele doua intrări si generează patru ieșiri distincte each corespunzând unei combinații a intrărilor. 00-> portul A, 01->portul B, 10->portul C, 11->registrul de controle. In caseul asta, cum is asignata zona de addresses 0250H-0256H sau 0A50H-0A56H in funcție de poziția microswitch S2, in pereche addressesle diferă via bits de pe rangul A1 si A2 si which sunt folosiți for pins de entersre A0 respectiv A1.

Decodifiwhicha porturilor interfacesi seriale si paralele (Circuitul 74LS138)

Se folosește decodifiwhicha a porturilor pe interfaces for a putea selecta una dintre ele si in intervalul de addresses dat. Aceasta selecție se face pe baza bus de addresses știind intervalul de addresses dedicat for each interfața si creând funcții logicale which activeatedes interfața corecta when microprocesorul addressesază unul dintre acis intervale. Pentru that se folosește un decodificator 3-la-8 74LS138 (U19) si circuits logicale for activeatedion sau dezactiveatedion acestuia (pins enable). Decodificatorul funcționează la fel ca 74LS139 doar ca has 3 intrări si 8 ieșiri. In acest case nu se poate folosi un decodificator 2-la-4 for ca cele 4 ranks de addresses pe which vrem sa le diferențiem for a le selecta unic, nu diferă in doar 2 bits din magistrala de addresses, in caseul asta ne mai trebuie încă un bit for a putea face distincția. Acei bitsi sunt ranges A11, A10 si A8 cum se poate observa in tabel:

Adresa (Hex)	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	Acces	Interfețe - Comutator
04D0H	0	0	0	0	0	1	0	0	1	1	0	1	0	0	0	0	Date	Seriala0 S1 = 0
04D2H	0	0	0	0	0	1	0	0	1	1	0	1	0	0	1	0	Comanda	
05D0H	0	0	0	0	0	1	0	1	1	1	0	1	0	0	0	0	Date	Seriala1 S1 = 1
05D2H	0	0	0	0	0	1	0	1	1	1	0	1	0	0	1	0	Comanda	
0250H	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0	0	Port A	Paralela0 S2 = 0
0252H	0	0	0	0	0	0	1	0	0	1	0	1	0	0	1	0	Port B	
0254H	0	0	0	0	0	0	1	0	0	1	0	1	0	1	0	0	Port C	
0256H	0	0	0	0	0	0	1	0	0	1	0	1	0	1	1	0	Controle	
0A50H	0	0	0	0	1	0	1	0	0	1	0	1	0	0	0	0	Port A	Paralela1 S2 = 1.
0A52H	0	0	0	0	1	0	1	0	0	1	0	1	0	0	1	0	Port B	
0A54H	0	0	0	0	1	0	1	0	0	1	0	1	0	1	0	0	Port C	
0A56H	0	0	0	0	1	0	1	0	0	1	0	1	0	1	1	0	Controle	

Verde: Adresele de porturi for Interfața Seriala cu comutatorul pe 0

Albastru: Adresele de porturi for Interfața Seriala cu comutatorul pe 1

Roșu: Adresele de porturi for Interfața Paralela cu comutatorul pe 0

Galben: Adresele de porturi for Interfața Paralela cu comutatorul pe 1

Gri: Ranguri de selecție for decodificator

Funcțiile combinaționale:

$!Sel_{SERIALA0} = !A11 * A10 * !A8$

$!Sel_{SERIALA1} = !A11 * A10 * A8$

$!Sel_{PARALELA0} = !A11 * !A10 * !A8$

$!Sel_{PARALELA1} = A11 * !A10 * !A8$

Este o decodifiwhich incompleta for ca am omis primele ranks A15-A12.

Tabelul de adevăr for ieșirea decodificatorului is următorul:

A11	A10	A8	Output
0	0	0	Y0# (Ajunge la pin 3 al comutatorului S2 – interfața paralela)
0	0	1	Neused
0	1	0	Y2# (Ajunge la pin 3 al comutatorului S1 – interfața seriala)
0	1	1	Y3# (Ajunge la pin 2 al comutatorului S1 – interfața seriala)
1	0	0	Y4# (Ajunge la pin 2 al comutatorului S2 – interfața paralela)
1	0	1	Neused
1	1	0	Neused
1	1	1	Neused

Se observa ca sunt doua microcomutatohas because sunt doua intervale de addresses for each interfața, si for a selecta intre una sau alta se apasă sau nu butonul. Când e „0”, adică nu e apăsat butonul, is legat la pin 3 al comutatorului, iar when is apăsat butonul „1” atunci se leagă la pin 2.

Pentru a activea decodificatorul, avem need de condițiile de Enable. Acisa depind de bits din magistrala de addresses which sunt fix for un anumit set de addresses. Pentru E1# îl legam directa la semnalul M/IO# de la 8086 which va fi active pe 0 logical when lucram cu interfacesle, iar when lucram cu memory, va fi pe 1 logical activeând E3 al decodificatorului de memorie.

E2# = A15+A14+A13+A12+A5

Acis ranks sunt 0 pis toate intervalele de addresses usede for interfaces. Sunt selectați așa astfel încât whether unul din ei is pe 1 înseamnă ca is alta adresa which nu e in rangul de addresses for interfaces si nu se va activea.

E3 = (A6 * A4) * (A9 ^ A7)

Rangurile A6 si A4 sunt pe 1 logical pis tot, folosind un AND iar A9 si A7 ori sunt pe 1 ori sunt pe 0 dar nici pe 1 sau pe 0 deodată, de that se folosește un XOR. [1]

Conecthasa Afișajelor si a minitastaturi la decodificatorul de porturi

Conecthasa Ledurilor

Se plasează 10 LEDs in proiect which se vor conecta la anod comun. Se conectează anodul cu un canențial mai mhas decât cel de la catod. Diferența canențialelor dintre Anod si catod trebuie sa fie mai mhas de cat tensiunea de prag (Vt) for ca dioda sa fie polarizata direct si ledul sa lumineze. Aceasta tensiune de prag poate diferi in funcție de lumina pe which o emite ledul. In plus, folosim o rezistenta de 330 de ohm in caseul asta connected la Vcc si anod for limithasa curentului si via varierea voltajului se reduce sau accentuează intensitatea luminoasa a ledului. Catodul LEDslor se conectează la ieșirile circuitslor de registru 74LS373 (explicat la unitatea de controle) in which va trebui sa setm bistabilul pe (0 logical) for ca sa fie that diferență de canențial ca ledul sa fie avias, whether nu va fi stins (1 logical). Se folosesc doua registre for ca in primul (U32) conectam 8 LEDs via pins 1Q-8Q iar cele doua LEDs ramase le conectam la 1Q respectiv 2Q de la (U33). Pinul OC# la grund si pins G din each la o ieșire diferita la decodificatorul de porturi explicat mai low.

Conecthasa afișajelor cu 7 segmente

Se plasează 8 modeule de afișhas de 7 segmente. Fiewhich rang is compus de 8 LEDs connectede împreuna (7 for segmente si unul for punct) si each led is accesibil via pins de la circuit. La fel ca si la LEDs, exista doua tipuri de connections, cu anod comun (is cea useda in proiect) sau catod comun. Cele 8 ieșiri ale modeului sunt connectede la un registru 74LS373, si cum avem 8 ranks vor fi in total 8 registre (U34,U35,U36,U37,U38,U39,U40 si U41). Cum se folosește catod comun for a aviade ledul se va comanda cu (0 logical). Pentru a afișa un număr hex on a rank al modeulului exista 2 soluții, cea hardwhas si cea softwhas. Dar in caseul asta se folosește soluția softwhas which poate afișa oriwhich configurație cu forma celor 7 segmente cerând un registru. Softwhas-ul is mai complex ca in soluția hardwhas. Pentru a putea comanda un modeul with segments multe ranks exista doua soluții, cea multiplexed in timp si cea **nemultiplexed in timp**. In caseul asta se folosește cea nemultiplexed in which each registru va fi comandat ca port de ieșire (via pin G) si vor memora configurațiile which se vor afișa. Avantajul is implementation softwhas mai simpla dar dezavantajul is o folosire mhas de registre si circuits plus un consum mai mhas, whether ar fi cea multiplexată am avea need doar de un registru for toate modeulele.

Tabelul for input la registrul de given 74LS373 for afișhasa cifrelor hex on a rank:

Hex	DP	G	F	E	D	C	B	A	In hex
0	1	1	0	0	0	0	0	0	C0H
1	1	1	1	1	1	0	0	1	F9H
2	1	0	1	0	0	1	0	0	A4H
3	1	0	1	1	0	0	0	0	B0H
4	1	0	0	1	1	0	0	1	99H
5	1	0	0	1	0	0	1	0	92H
6	1	0	0	0	0	0	1	0	82H
7	1	1	1	1	1	0	0	0	F8H
8	1	0	0	0	0	0	0	0	80H
9	1	0	0	1	1	0	0	0	98H
A	1	0	0	0	1	0	0	0	88H
B	1	0	0	0	0	0	1	1	83H
C	1	1	0	0	0	1	1	0	C6H
D	1	0	1	0	0	0	0	1	A1H
E	1	0	0	0	0	1	1	0	86H
F	1	0	0	0	1	1	1	0	8EH

Conecthasa minitastaturii

Se plasează o minitastatura cu 9 contacts organizate ca o matrice via trei lines si trei coloane. Fiewhich tasta is un switch (comutator cu revenire) which has doi pins, primul pin is connected cu toate tastele din coloana respectiva, in caseul asta, tastele din prima coloana 1,4 si 7, tastele din a doua coloana 2,5 si 8, si tastele din a treia coloana 3, 6 si 9. Iar pin 2 is connected cu toate tastele din rândul respectiv, primul rând fiind tastele 1,2 si 3, al doilea is 4,5 si 6, si al treilea is 7,8 si 9. Cele 3 coloane sunt legate la o anodul unei diode si catodul diodei is legat la 3 ieșiri al circuitului registru 74LS373 (U43).Diodele sunt usede ca protecție a ieșirilor portului de ieșire. Iar cele trei rânduri sunt connectede la VCC trewhen via 3 rezistente de 10kohm si la 3 intrări ale circuitului 74LS244 (U42). La început tastele sunt pe „iddle" adică nu trece curent via ele (nu sunt connectede cu circuitul). Pentru a detecta o tastate trebuie sa aplicam viacipiul de scanning al tastaturi which începe via a conecta coloanele, trewhen via dioda, la „0 logical" in registrul de ieșire (U43). Atunci when o tasta is apăsata, va curge curentul via ea si cum exista that diferența de canențial intre Vcc si leghasa coloanei la 0 in registru, se va putea citi pe rând rândurile legate la input circuitului U42 for a vedea whether tasta a fost apăsata sau nu. Circuitul U42 is used for amplifiwhicha / separhasa buseslor unidirecționale, si in acest case, when is apăsat o tasta pe rândul acelei taste se va genera un 0 logical which va fi transmis pe magistrala de given indiwhen ca pe rândul respectiv sa acționat o tasta. Știind ce coloana si ce rând sau acționat, via coordonate se știe ce tasta sa apăsat. Aceasta lucrhas alternata intre cele doua porturi for a scana tastele via coloane si rânduri se poate vedea in routine for scanning.

Decodifiwhicha porturilor for Afişaje si minitastatura (Circuitul 74LS154)

Se folosește decodifiwhicha a porturilor de afișaje si a minitastaturi for a putea selecta unul dintre iei la un moment dat. In caseul asta cum avem 2 porturi for LEDs, 8 for segmente si 2 for minitastatura, se folosește un decodificator 4-la-16 (U44). Acest decodificator has 4 intrări si 16 ieșiri dintre which vom folosi doar 12. Adresele de porturi le am ales in intervalul 0300H la 032Ch incrementat cu cate 4 because bits which diferă intre ele si which sunt unice sunt ranges A2,A3,A4 si A5. Acis ranks sunt conectați la input decodificatorului astfel încât each ieșire din decodificator sa corespunda a a periferic specific. Toate ports, mai puțin circuitul U42 de la minitastatura, sunt activee pe „1" logical la input pinslor G, de that se leagă un inversor la each ieșire a decodificatorului. Decodifiwhich useda is incompleta.

Tabelul de addresses for ports de afișaje si minitastaura:

Adresa (Hex)	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	Porturi Afisaje, tastatura
0300H	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	SL1
0304H	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	SL2
0308H	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	SA1
030CH	0	0	0	0	0	0	1	1	0	0	0	0	1	1	0	0	SA2
0310H	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0	0	SA3
0314H	0	0	0	0	0	0	1	1	0	0	0	1	0	1	0	0	SA4
0318H	0	0	0	0	0	0	1	1	0	0	0	1	1	0	0	0	SA5
031CH	0	0	0	0	0	0	1	1	0	0	0	1	1	1	0	0	SA6
0320H	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	SA7
0324H	0	0	0	0	0	0	1	1	0	0	1	0	0	1	0	0	SA8
0328H	0	0	0	0	0	0	1	1	0	0	1	0	1	0	0	0	ST1
032CH	0	0	0	0	0	0	1	1	0	0	1	0	1	1	0	0	ST2#

Verde: Adresele de porturi for Leduri.

Albastru: Adresele de porturi for Modulu de segmente cu 8 ranks

Galben: Adresele de porturi for minitastatura

Gri: Ranguri de selecție for decodificator

Tabelul de adevăr for ieșirea decodificatorului is următorul:

A5	A4	A3	A2	Output
0	0	0	0	Y0# (Conectat la portul 1 al LEDslor U32)
0	0	0	1	Y1# (Conectat la portul 2 al LEDslor U33)
0	0	1	0	Y2# (Conectat la portul 1 rang seven segment U34)
0	0	1	1	Y3# (Conectat la portul 2 rang seven segment U35)
0	1	0	0	Y4# (Conectat la portul 3 rang seven segment U36)
0	1	0	1	Y5# (Conectat la portul 4 rang seven segment U37)
0	1	1	0	Y6# (Conectat la portul 5 rang seven segment U38)
0	1	1	1	Y7# (Conectat la portul 6 rang seven segment U39)
1	0	0	0	Y8# (Conectat la portul 7 rang seven segment U40)
1	0	0	1	Y9# (Conectat la portul 8 rang seven segment U41)
1	0	1	0	Y10# (Conectat la portul de ieșire al minitastaturi U43)
1	0	1	1	Y11# (Conectat la portul de entersre al minitataturi U42)

La fel ca la decodificatorul de porturi for interfaces, avem need de semnale enable for activeatedion sau dezactiveatedion decodificatorului. Trebuie alese ranges altfel încât addressesle de la ports de la afișaje si minitastatura sa nu interfereze cu addressesle for ports de la interfaces. Pentru E1# se folosește semnalul M/IO# de la 8086 (ca la decodificatorul for interfaces).

E2# = A15+A14+A13+A12+A11+A10

Pentru E2 am mai adăugat ranks which sunt pe 0 logical until la A10.

E3 = A9*A8

Sunt used A9 si A8 for ca sunt constanți pe 1 logical pe addressesle selectate

Conecthasa Modulul LCD (Interfața paralela)

Este plasat un LCD modeule 1602 (U45) cu 2 lines a cate 16 characters each, cuplata la interfața paralela. Pentru simplifiwhich se folosește interfața paralela for ca is ca mai comuna, ușor de implementat si transferul in paralel is mai rapid fiind ideal la actualizări rapide ale

afișajului. Modul in acest case funcționează cu modeul pe 8 bits which folosește toți cei 8 pins de given (DB0-DB7), dar mai poate funcționa in modeul de 5 bits which utilizează doar pins (D4-D7) for a economisii pinsi de pe 8255 fiind necesara trimiterea given in doua pachete (high nibble si low nibble). Pinul Vo se folosește for reglhasa contrastului afișajului ecranului si is connected la un canențiometru for al putea ajusta. Pini BLA (Backlight Anode) si BLK(Backlight Cathode) controle iluminhasa de fundal a ecranului which face ca textul afișat sa fie mai lizibil sau nu si for ajustatea luminozități se conectează o rezistenta intre Vcc si BLA. Pinul RS(Register select) selectează whether semnalul trimis reprezintă o comanda (0 logical) sau given (1 logical). R/W selectează direcția given, adică cu 0 logical is scriere (de la 8255 către LCD) iar for 1 logical is citire. E is pin de Enable which activeatedes modeulul cu o tranziție de la 0 la 1 logical. Cum is configurat 8255 in proiect, portul A va fi for transmise de given (DB0-DB7) iar portul C inferior is configurat for signals de controle (R/S, R/W si E). Deci addressesle de porturi for controlehasa modeulului sunt cunoscute.

Descrierea Softwhas

Rutinele de programhas ale circuitslor 8251 si 8255

```
; Ne definim addressesle porturilor for 8251 si 8253

PORT_8251_1 EQU 04D0H

PORT_8251_2 EQU 05D0H

PORT_8255_1 EQU 0250H

PORT_8255_2 EQU 0A50H

;-------------------------------------------

; Subrutina for configurhasa circuitului 8251

; Input: Al = 0 for S1 = 0, Al = 1 for S1 = 1

; Configuram Cuvantul de Mod (mode asincron): 2 biti de stop,

; fara paritate, 8 biti de given si factor de multipliwhich 16

; Configuram Cuvantul de Comanda: Resetrea biti de erohas,

; activeatedion receptia given, activea transmiterea given

;-------------------------------------------

CONFIG_8251 PROC

   CMP AL,0             ; verificam whether S1 = 0 sau S1 = 1

   JE SERIAL_S1_0       ; whether S1 = 1, trecem la configurhasa for S1 = 1

   MOV DX, PORT_8251_2   ; selectam portul 2

   JMP CONFIG_SERIAL     ; trecem la configurhasa circuitului 8251


SERIAL_S1_0:

   MOV DX, PORT_8251_1   ; selectam portul 1


CONFIG_SERIAL:

   MOV AL, 0CEH    ; in binar = 11001110

   OUT DX, AL      ; scriem cuvantul de mode

   MOV AL, 15H     ; in binar = 00010101
```

```
    OUT DX, AL      ; scriem cuvantul de comanda
    RET

CONFIG_8251 ENDP
;-------------------------------------------
; Exemplu de utilizhas a subrutinei CONFIG_8251
;-------------------------------------------
MOV AL, 0           ; setm AL = 0 for S1 = 0
CALL CONFIG_8251    ; apelam subrutina
;-------------------------------------------
; Subrutina for configurhasa circuitului 8255
; Input: Al = 0 for S2 = 0, Al = 1 for S2 = 1
; (Cuvantul de controle il vom configura astfel incat sa lucram cu ecrarnul LCD)
; Deci portul A fa vi de iesire, portul B ramane default, portul C inferior is de iesire (Semnalele:S,RW,E),
; si portul C superior ramane default, si modeul de lucru 0 la grupul A.
;-------------------------------------------
CONFIG_8255 PROC
    CMP AL,0            ; verificam whether S2 = 0 sau S2 = 1
    JE PARALLEL_S2_0    ; whether S2 = 1, trecem la configurhasa for S2 = 1
    MOV DX, PORT_8255_2 ; selectam portul 2
    JMP CONFIG_PARALLEL ; trecem la configurhasa circuitului 8255

PARALLEL_S2_0:
    MOV DX, PORT_8255_1 ; selectam portul 1

CONFIG_PARALLEL:
    MOV AL, 80H   ; in binar = 10000000
    OUT DX, AL;   ; scriem cuvantul de controle
    RET

CONFIG_8255 ENDP
;-------------------------------------------
; Exemplu de utilizhas a subrutinei CONFIG_8255
;-------------------------------------------
MOV AL, 0           ; setm AL = 0 for S2 = 0
CALL CONFIG_8255    ; apelam subrutina
```

Rutinele de transmission/reception character pe interfața seriala

```
; Ne definim addressesle porturilor ai 8251
PORT_8251_1 EQU 04D0H
PORT_8251_2 EQU 05D0H
;-------------------------------------------
; Subrutina for transmission character seriala
; Input: CL = characterul de transmis
;        BL = 0 for S1 = 0, BL = 1 for S1 = 1
;-------------------------------------------
EMISIE_CARACTER_SERIALA PROC
    ; Calculam baza
    CMP BL,0              ; verificam intervalul
    JE  BASE_1_EMISIE     ; whether BL = 0, folosim baza PORT_8251_1
    MOV SI, PORT_8251_2   ; altfel, folosim baza PORT_8251_2
    JMP EMISIE_START      ; trecem la transmission

BASE_1_EMISIE:
    MOV SI, PORT_8251_1   ; folosim baza PORT_8251_1

EMISIE_START:
    ; Verificam TxRDY (bitul 1 din cuvantul de state)
    MOV DX, SI            ; Adresa for given
    ADD DX, 2H            ; Adresa for comenzi/stari (04D2H sau 05D2H)

TX_WAIT:
    IN AL, DX             ; Citim cuvantul de state
    RCR AL, 1             ; Testam bitul TxRDY
    JNC TX_WAIT           ; Daca TxRDY = 0, asteptam

    ; Trimitem characterul
    MOV AL, CL            ; Incarcam characterul
    MOV DX, SI            ; Adresa for given
    OUT DX, AL            ; Transmitem characterul
    RET
TRANSMIT_CHAR ENDP
;-------------------------------------------
; Exemplu de utilizhas a subrutinei EMISIE_CARACTER_SERIALA
;-------------------------------------------
MOV CL, 'A'                   ; characterul de transmis
MOV BL, 0                     ; baza which o sa fie useda
CALL EMISIE_CARACTER_SERIALA      ; apelam subrutina
```

```
;---------------------------------------------
; Subrutina for receptie character seriala
; Output: CL = characterul receptionat
;         BL = 0 for S1 = 0, BL = 1 for S1 = 1
;---------------------------------------------
RECEPTIE_CARACTER_SERIALA PROC
    ; Calculam baza
    CMP BL,0              ; verificam intervalul
    JE  BASE_1_RECEPTIE   ; whether BL = 0, folosim baza PORT_8251_1
    MOV SI, PORT_8251_2   ; altfel, folosim baza PORT_8251_2
    JMP RECEPTIE_START    ; trecem la receptie

BASE_1_RECEPTIE:
    MOV SI, PORT_8251_1   ; folosim baza PORT_8251_1

RECEPTIE_START:
    ; Verificam RxRDY (bitul 2 din cuvantul de state)
    MOV DX, SI            ; Adresa for given
    ADD DX, 2H            ; Adresa for comenzi/stari (04D2H sau 05D2H)

RX_WAIT:
    IN AL, DX             ; Citim cuvantul de state
    RCR AL, 2             ; Testam bitul RxRDY
    JNC RX_WAIT           ; Daca RxRDY = 0, asteptam

    ; Citim characterul receptionat
    MOV DX, SI            ; Adresa for given
    IN AL, DX             ; Citim characterul
    MOV CL, AL            ; Stocam caraccterul in CL
    RET

RECEIVE_CHAR ENDP
;---------------------------------------------
; Exemplu de utilizhas a subrutinei RECEPTIE_CARACTER_SERIALA
;---------------------------------------------
MOV BL, 0                     ; baza which o sa fie useda
CALL RECEPTIE_CARACTER_SERIALA     ; apelam subrutina
; CL contine characterul receptionat
```

Rutinele de transmission character pe interfața paralela

```
; Ne definim addressesle porturilor ai 8255
PORT_8255_1 EQU 0250H
PORT_8255_2 EQU 0A50H
;-------------------------------------------
; Subrutina for transmission character paralela
; (ne focalizam pe afisajul LCD din proiect)
; Input: CL = characterul de transmis
;        BL = 0 for S2 = 0, BL = 1 for S2 = 1
; (S2 is conmutatorului which selecteaza baza)
;-------------------------------------------
EMISIE_CARACTER_PARALELA PROC
    ; Pasul 1: calculam baza
    CMP BL,0              ; verificam intervalul
    JE  BASE_1            ; whether BL = 0, folosim baza PORT_8255_1
    MOV SI, PORT_8255_2   ; altfel, folosim baza PORT_8255_2
    JMP EMISIE_START      ; trecem la transmission

BASE_1:
    MOV SI, PORT_8255_1   ; folosim baza PORT_8255_1

EMISIE_START:
    ; Pasul 2: setm RS si RW pe portul C
    MOV DX, SI            ; Baza setta
    ADD DX, 04H           ; adunam cu 4 ca sa ajungem la adresa portului C
    MOV AL, 01H           ; setm RS=1 (character) si RW=0(scriere)
    OUT DX, AL            ; scriem pe portul C (signals de controle for LCD)

    ; Pasul 3: trimitem characterul pe portul A
    MOV DX, SI            ; Baza setta
    MOV AL, CL            ; characterul de transmis
    OUT DX, AL            ; scriem pe portul A (characterul de transmis)

    ; Pasul 4: generam semnalul de enable
    MOV DX, SI            ; Baza setta
    ADD DX, 04H           ; adunam cu 4 ca sa ajungem la adresa portului C
    OR AL, 04H            ; setm  bitul de eneable, E=1
    OUT DX, AL            ; scriem pe portul C semnalul de enable
```

```
    CALL DELAY_SHORT      ; introducem un delay scurt for a allows citirea informatiilor de catre LCD

    ; Pasul 5: resetm semnalul de enable
    AND AL, 01H           ; resetm bitul de enable, E=0
    OUT DX, AL            ; actualizam portul C

    RET
EMISIE_CARACTER_PARALELA ENDP
;-------------------------------------------
;Subroutine forlay scurt
;-------------------------------------------
DELAY_SHORT PROC
    MOV CX, 0FFFH     ; setm un contor mhas
DELAY_LOOP:
    LOOP DELAY_LOOP   ; decrementam contorul until cand ajunge la 0
    RET
DELAY_SHORT ENDP
;-------------------------------------------
; Exemplu de utilizhas a subrutinei EMISIE_CARACTER_PARALELA
;-------------------------------------------
MOV CL, 'A'                    ; characterul de transmis
MOV BL, 0                      ; baza which o sa fie useda
CALL EMISIE_CARACTER_PARALELA       ; apelam subrutina

; Explicatie semnal Enable:
; LCD-ul citis informatiile (given sau comenzi)) doar cand detecteaza o
; tranzitie de la 0 la 1 pe semnalul de enable. Dupa thatsta tranzitie
; lcd-ul citis informatiile si le proceseaza. Lasam un scurt delay
; intre generhasa enable si setrea enable la 0, for a allows lcd-ului
; sa citeasca informatiile. Dupa which setm enable pe 0
; In alte dizpozitive se poate trasnmite dupaia inca un semnal de enable de reconfirmhas a signal
; de enable, for a evita erorile de transmission.
```

Rutina de scanning a minitastaturi

```
; Ne definim addressesle porturilor for minitastaura
TAST_PORT1 EQU 0328H ; Adresa portului de iesire al tastaturii
TAST_PORT2 EQU 032CH ; Adresa portului de entersre al tastaturii
;-------------------------------------------
; Subrutina for scanninga minitastaturii
```

```
;-------------------------------------------
TAST_SCAN proc

   ; punem pe 0 prima coloana si verificam whether sau actionat tastele 1,4 sau 7
   MOV AL, 0FEh          ; activeam prima coloana, adica punem pe 0 bitul 0 for a curge curentul (1111 1110)
   OUT TAST_PORT1, AL    ; scriem in portul de iesire al tastaturii
   IN AL, TAST_PORT2     ; citim din portul de entersre al tastaturii
   AND AL, 01H           ; verificam linea 1, whether sa apasat tasta 1 (bitul 0)
   JZ TASTA1             ; whether bitul 0 is 0, tasta 1 a fost apasata
   IN AL, TAST_PORT2     ; whether nu, citim din nou portul de entersre al tastaturii
   AND AL, 02H           ; verificam linea 2, whether sa apasat tasta 4 (bitul 1)
   JZ TASTA4             ; whether bitul 1 is 0, tasta 4 a fost apasata
   IN AL, TAST_PORT2
   AND AL, 04H           ; verificam linea 3, whether sa apasat tasta 7 (bitul 2)
   JZ TASTA7             ; whether bitul 2 is 0, tasta 7 a fost apasata

   ; punem pe 0 a doua coloana si verificam whether sau actionat tastele 2,5 sau 8
   MOV AL, 0FDh          ; activeam a doua coloana, adica punem pe 0 bitul 1 for a curge curentul (1111 1101)
   OUT TAST_PORT1, AL    ; scriem in portul de iesire al tastaturii
   IN AL, TAST_PORT2     ; citim din portul de entersre al tastaturii
   AND AL, 01H           ; verificam linea 1, whether sa apasat tasta 2 (bitul 0)
   JZ TASTA2             ; whether bitul 0 is 0, tasta 2 a fost apasata
   IN AL, TAST_PORT2
   AND AL, 02H           ; verificam linea 2, whether sa apasat tasta 5 (bitul 1)
   JZ TASTA5             ; whether bitul 1 is 0, tasta 5 a fost apasata
   IN AL, TAST_PORT2
   AND AL, 04H           ; verificam linea 3, whether sa apasat tasta 8 (bitul 2)
   JZ TASTA8             ; whether bitul 2 is 0, tasta 8 a fost apasata

   ; punem pe 0 a treia coloana si verificam whether sau actionat tastele 3,6 sau 9
   MOV AL, 0FBh          ; activeam a treia coloana, adica punem pe 0 bitul 2 for a curge curentul (1111 1011)
   OUT TAST_PORT1, AL    ; scriem in portul de iesire al tastaturii
   IN AL, TAST_PORT2     ; citim din portul de entersre al tastaturii
   AND AL, 01H           ; verificam linea 1, whether sa apasat tasta 3 (bitul 0)
   JZ TASTA3             ; whether bitul 0 is 0, tasta 3 a fost apasata
   IN AL, TAST_PORT2
   AND AL, 02H           ; verificam linea 2, whether sa apasat tasta 6 (bitul 1)
   JZ TASTA6             ; whether bitul 1 is 0, tasta 6 a fost apasata
   IN AL, TAST_PORT2
   AND AL, 04H           ; verificam linea 3, whether sa apasat tasta 9 (bitul 2)
```

```
    JZ TASTA9               ; whether bitul 2 is 0, tasta 9 a fost apasata

    JMP TAST_SCAN           ; whether nu s-a apasat nicio tasta, repetam procesul

;  Subrutine for trathasa each taste apasata
TASTA1:
    CALL DELAY              ; asteapta stabilizhasa contactslor
AST1:
    IN AL, TAST_PORT2       ; citim din nou statea lineslor
    AND AL, 01H             ; verificam whether tasta 1 is inca apasata
    JZ AST1                 ; whether da, continuesm astepthasa
    CALL DELAY              ; asteptam eliberhasa tastei

    ; aici se va executa codul for tasta 1
    JMP TAST_SCAN           ; dupa ce am terminat, revenim la scanninga tastaturii

TASTA2:
    CALL DELAY
AST2:
    IN AL, TAST_PORT2
    AND AL, 01H
    JZ AST2
    CALL DELAY

    ; aici se va executa codul for tasta 2
    JMP TAST_SCAN

TASTA3:
    CALL DELAY
AST3:
    IN AL, TAST_PORT2
    AND AL, 01H
    JZ AST3
    CALL DELAY

    ; aici se va executa codul for tasta 3
    JMP TAST_SCAN

TASTA4:
    CALL DELAY
```

```
AST4:
    IN AL, TAST_PORT2
    AND AL, 02H
    JZ AST4
    CALL DELAY

    ; aici se va executa codul for tasta 4
    JMP TAST_SCAN

TASTA5:
    CALL DELAY
AST5:
    IN AL, TAST_PORT2
    AND AL, 02H
    JZ AST5
    CALL DELAY

    ; aici se va executa codul for tasta 5
    JMP TAST_SCAN

TASTA6:
    CALL DELAY
AST6:
    IN AL, TAST_PORT2
    AND AL, 02H
    JZ AST6
    CALL DELAY

    ; aici se va executa codul for tasta 6
    JMP TAST_SCAN

TASTA7:
    CALL DELAY
AST7:
    IN AL, TAST_PORT2
    AND AL, 03H
    JZ AST7
    CALL DELAY

    ; aici se va executa codul for tasta 7
```

```
    JMP TAST_SCAN

TASTA8:
    CALL DELAY
AST8:
    IN AL, TAST_PORT2
    AND AL, 03H
    JZ AST8
    CALL DELAY

    ; aici se va executa codul for tasta 8
    JMP TAST_SCAN

TASTA9:
    CALL DELAY
AST9:
    IN AL, TAST_PORT2
    AND AL, 03H
    JZ AST9
    CALL DELAY

    ; aici se va executa codul for tasta 9
    JMP TAST_SCAN

TAST_SCAN endp
;-------------------------------------------
;Subroutine forlay (Stabilizhas contacts)
;-------------------------------------------
DELAY PROC
    MOV CX, 0FFFFH    ; setm un contor mhas
DELAY_LOOP:
    LOOP DELAY_LOOP   ; decrementam contorul until cand ajunge la 0
    RET
DELAY ENDP
```

Rutina de turning on/turning off a an LED

```
; Ne definim addressesle portului for LEDs
LED_PORT1 EQU 0300h   ; Adresa portului de controle al LEDslor 0-7
LED_PORT2 EQU 0304h   ; Adresa portului de controle al LEDslor 8-9
```

```
; Definim si LEDsle in hex
LED1 EQU 0FEh
LED2 EQU 0FDh
LED3 EQU 0FBh
LED4 EQU 0F7h
LED5 EQU 0EFh
LED6 EQU 0DFh
LED7 EQU 0BFh
LED8 EQU 07Fh
LED9 EQU 002h
LED10 EQU 001h
;-------------------------------------------
; Subrutina for turning ona uni led
; Input: AL = numarul ledului (0-9)
; Adica inainte se alege ce led vrem sa aviadem, numarul 8 in caseul nostru.(dam value noi)
; MOV AL, 8
;-------------------------------------------
LED_ON proc
    CMP AL, 8          ; Verificam whether ledul is in portul 2
    JAE PORT_2_ON      ; Daca da vom folosi portul 2

    ; whether nu, folosim portul 1
    MOV DX, LED_PORT1  ; selectam portul 1
    MOV AL, LED8       ; aviadem ledul (in caseul nostru ledul 8)
    OUT DX, AL         ; trimitem value in port
    RET                ; iesim din subrutina

    ; for al doilea port
PORT_2_ON:
    MOV DX, LED_PORT2
    MOV AL, LED9       ; aici nu prea conteaza ce led aviadem (for ca nu se va aviade in acest case)
    OUT DX, AL
    RET
LED_ON endp
;-------------------------------------------
; Exemplu de utilizhas a subrutinei LED_ON
;-------------------------------------------
MOV AL, 8            ; setm AL = 8 for a aviade ledul 8
CALL LED_ON          ; apelam subrutina
```

```
;-------------------------------------------
; Subrutina for turning offa uni led
; In caseul asta vom aviade ledul 8 (dam noi value)
; Input: AL = numarul ledului (0-9)
;-------------------------------------------
LED_OFF proc
    CMP AL, 8          ; Verificam whether ledul is in portul 2
    JAE PORT_2_OFF     ; Daca da vom folosi portul 2

    ; whether nu, folosim portul 1
    MOV DX, LED_PORT1  ; selectam portul 1
    MOV AL, 0          ; stingem LEDsle
    OUT DX, AL         ; trimitem value in port
    RET                ; iesim din subrutina

    ; for al doilea port
PORT_2_OFF:
    MOV DX, LED_PORT2  ; selectam portul 2
    MOV AL, 0
    OUT DX, AL
    RET
LED_OFF endp
;-------------------------------------------
; Exemplu de utilizhas a subrutinei LED_OFF
;-------------------------------------------
MOV AL, 8            ; setm AL = 8 for a stinge ledul 8
CALL LED_OFF         ; apelam subrutina
```

Rutina de afișhas a a character hex on a rank with segments

```
; Ne definim addressesle porturilor for segmente
SEG_PORT1 EQU 0308h   ; adresa primului port for segmente
SEG_PORT2 EQU 030Ch   ; adresa celui de-al doilea port for segmente
SEG_PORT3 EQU 0310h   ; adresa celui de-al treilea port for segmente
SEG_PORT4 EQU 0314h   ; adresa celui de-al patrulea port for segmente
SEG_PORT5 EQU 0318h   ; adresa celui de-al cincilea port for segmente
SEG_PORT6 EQU 031Ch   ; adresa celui de-al saselea port for segmente
SEG_PORT7 EQU 0320h   ; adresa celui de-al saptelea port for segmente
SEG_PORT8 EQU 0324h   ; adresa celui de-al optulea port for segmente
```

```
; Definim si charactersle in hex
HEX_0 EQU C0h
HEX_1 EQU F9h
HEX_2 EQU A4h
HEX_3 EQU B0h
HEX_4 EQU 99h
HEX_5 EQU 92h
HEX_6 EQU 82h
HEX_7 EQU F8h
HEX_8 EQU 80h
HEX_9 EQU 98h
HEX_A EQU 88h
HEX_B EQU 83h
HEX_C EQU C6h
HEX_D EQU A1h
HEX_E EQU 86h
HEX_F EQU 8Eh
;-------------------------------------------
; Subrutina for afishasa a segment
; Input: BL = numarul segmentului (1-8)
; AL = characterul hexzecimal (0-9, A-F) - in caseul nostru vom alege numarul 4
;-------------------------------------------
SEG_ON proc
    CMP BL, 1          ; Verificam whether segmentul is in portul 1
    JAE PORT_1_ON      ; Daca da vom folosi portul 1

    CMP BL, 2          ; Verificam whether segmentul is in portul 2
    JAE PORT_2_ON      ; Daca da vom folosi portul 2

    CMP BL, 3          ; Verificam whether segmentul is in portul 3
    JAE PORT_3_ON      ; Daca da vom folosi portul 3

    CMP BL, 4          ; Verificam whether segmentul is in portul 4
    JAE PORT_4_ON      ; Daca da vom folosi portul 4

    CMP BL, 5          ; Verificam whether segmentul is in portul 5
    JAE PORT_5_ON      ; Daca da vom folosi portul 5

    CMP BL, 6          ; Verificam whether segmentul is in portul 6
    JAE PORT_6_ON      ; Daca da vom folosi portul 6
```

```
    CMP BL, 7          ; Verificam whether segmentul is in portul 7
    JAE PORT_7_ON      ; Daca da vom folosi portul 7

    ; whether nu, folosim portul 8
    MOV DX, SEG_PORT8  ; selectam portul 8
    MOV AL, HEX_4      ; afisam characterul 4
    OUT DX, AL         ; trimitem value in port
    RET                ; iesim din subrutina

    ; for primul port
PORT_1_ON:
    MOV DX, SEG_PORT1  ; selectam portul 1
    MOV AL, HEX_4      ; afisam characterul 4
    OUT DX, AL         ; trimitem value in port
    RET

    ; for al doilea port
PORT_2_ON:
    MOV DX, SEG_PORT2  ; selectam portul 2
    MOV AL, HEX_4      ; afisam characterul 4
    OUT DX, AL         ; trimitem value in port
    RET

    ; for al treilea port
PORT_3_ON:
    MOV DX, SEG_PORT3  ; selectam portul 3
    MOV AL, HEX_4      ; afisam characterul 4
    OUT DX, AL         ; trimitem value in port
    RET

    ; for al patrulea port
PORT_4_ON:
    MOV DX, SEG_PORT4  ; selectam portul 4
    MOV AL, HEX_4      ; afisam characterul 4
    OUT DX, AL         ; trimitem value in port
    RET

    ; for al cincilea port
PORT_5_ON:
```

```
    MOV DX, SEG_PORT5  ; selectam portul 5
    MOV AL, HEX_4      ; afisam characterul 4
    OUT DX, AL         ; trimitem value in port
    RET

    ; for al saselea port
PORT_6_ON:
    MOV DX, SEG_PORT6  ; selectam portul 6
    MOV AL, HEX_4      ; afisam characterul 4
    OUT DX, AL         ; trimitem value in port
    RET

    ; for al saptelea port
PORT_7_ON:
    MOV DX, SEG_PORT7  ; selectam portul 7
    MOV AL, HEX_4      ; afisam characterul 4
    OUT DX, AL         ; trimitem value in port
    RET

SEG_ON endp
;-----------------------------------------
; Exemplu de utilizhas a subrutinei SEG_ON
;-----------------------------------------
MOV BL, 4           ; setm BL = 4 for a aviade segmentul 4
CALL SEG_ON         ; apelam subrutina
```

Bibliografia

1. Mircea Popa, ***"Sisteme cu microprocesohas"***, Editura Orizonturi Universithas, 2003. (Referință viacipală for detalii despre microprocesorul 8086 și structura a microsystem.
2. D.V. Hall, **"Microprocessors and Interfacing: Programming and Hardwhas"**, McGraw-Hill, 1992
3. **27C512 Datasheet**, STMicroelectronics, https://www.allgivensheet.com/givensheet-pdf/pdf/23533/STMICROELECTRONICS/27C512.html (accesat pe 10 noiembrie 2024)
4. **Datasheet-ul 8255A (Programmable Peripheral Interface)**, Intel, https://www.allgivensheet.com/givensheet-pdf/pdf/66100/INTEL/8255A.html (accesat pe 16 noiembrie 2024)
5. **Datasheet-ul 8251A (Programmable Communication Interface)**, Intel, https://www.allgivensheet.com/givensheet-pdf/pdf/66096/INTEL/8251A.html (accesat pe 16 noiembrie 2024)
6. **74LS244 Datasheet**, Texas Instruments, https://www.allgivensheet.com/givensheet-pdf/pdf/28030/TI/74LS244.html (accesat pe 30 decembrie 2024)
7. **74LS373 Datasheet**, Texas Instruments, https://www.allgivensheet.com/givensheet-pdf/pdf/28029/TI/74LS373.html (accesat pe 30 decembrie 2024)
8. **"LCD interfacing"**, Technobyte, https://technobyte.org/lcd-interfacing-8051-8-bit-4-bit-8255/ (accesat pe 31 decembrie 2024).
9. **MAX232 Datasheet,** Maxim Integrated Products, https://www.allgivensheet.com/givensheet-pdf/view/73074/MAXIM/MAX232.html (accesat pe 1 ianuarie 2025).
10. Proiectul didactic și materialele suplimenthas primite de la profesor. (Documentația de referință for cerințele tehnice și descrierea proiectului.)