HÁZI FELADAT

Programozás alapjai 2.

Terv

Dézsi Bálint Loránd
NINN8B

2024. március 22.

TARTALOM

1. Digitális áramkör

Készítsen egyszerű objektummodellt digitális áramkör szimulálására! A modell minimálisan tartalmazza a következő elemeket:

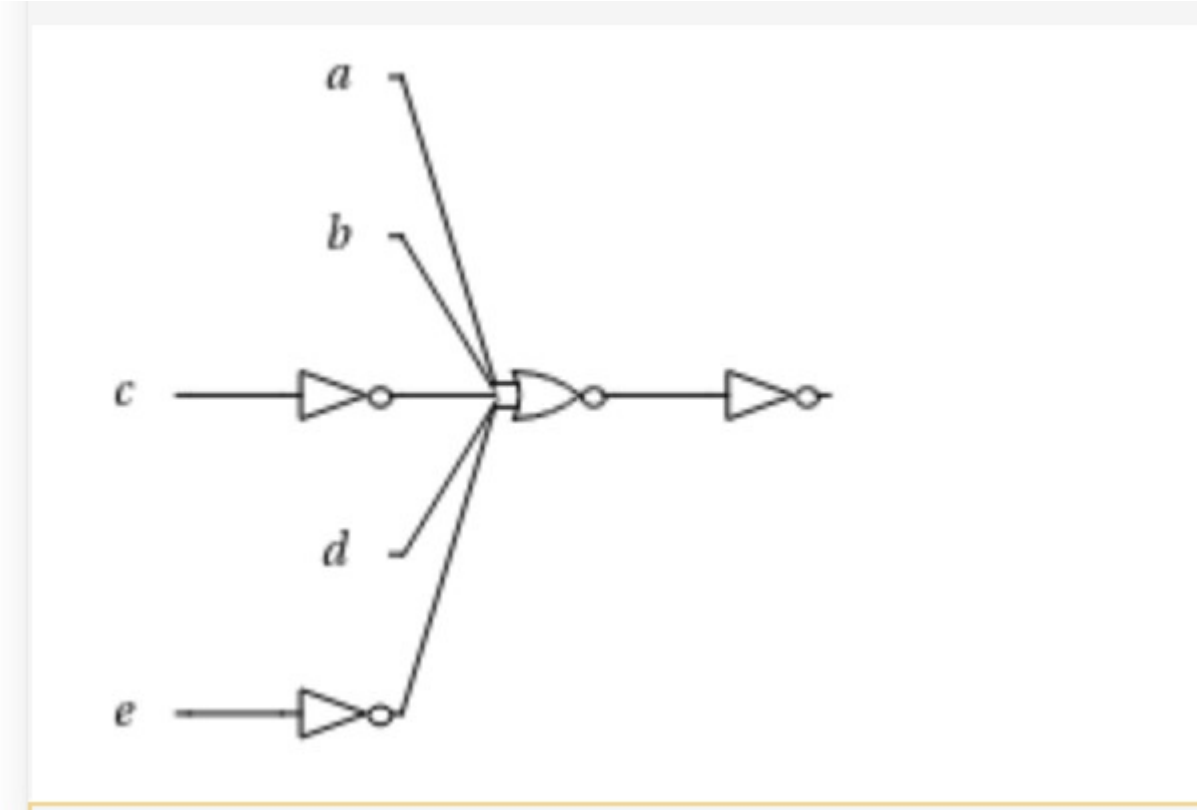
- NOR kapu
- vezérelhető forrás
- összekötő vezeték
- inverter

A modell felhasználásával szimulálja egy olyan 5 bemenetű kombinációs hálózat működését, amely akkor ad a kimenetén hamis értéket, ha bementén előálló kombináció 5!
Demonstrálja a működést külön modulként fordított tesztprogrammal! A megoldáshoz ne használjon STL tárolót!

2. Feladatspecifikáció

A program képes digitális áramköri elemek modellezésére és azokból kombinációs hálózatot létrehozni.
Az áramköri elemeket össze lehet kötni vezeték felhasználásával,vagy vezeték használata nélkül közvetlenül A választott bemenetére lehet kötni B kimenetét.
A modellezet teszt hálózat a feladatleírás szerint akkor fog hamis(logikai 0) értéket adni amikor a bemeneti 5 változó értéke bináris 5(00101), vagyis minden más esetben igazat ad vissza. A működést demonstráló kombinációs hálózat igazságtáblája, és függvénye (diszjunktív normál alakban):

E	D	C	B	A	Y
0	0	0	0	0	**1**
0	0	0	0	1	**1**
0	0	0	1	0	**1**
0	0	0	1	1	**1**
0	0	1	0	0	**1**
0	0	1	0	1	**0**
0	0	1	1	0	**1**
0	0	1	1	1	**1**
0	1	0	0	0	**1**
0	1	0	0	1	**1**
0	1	0	1	0	**1**
0	1	0	1	1	**1**
0	1	1	0	0	**1**
0	1	1	0	1	**1**
0	1	1	1	0	**1**
0	1	1	1	1	**1**
1	0	0	0	0	**1**
1	0	0	0	1	**1**
1	0	0	1	0	**1**
1	0	0	1	1	**1**
1	0	1	0	0	**1**
1	0	1	0	1	**1**
1	0	1	1	0	**1**
1	0	1	1	1	**1**
1	1	0	0	0	**1**
1	1	0	0	1	**1**
1	1	0	1	0	**1**
1	1	0	1	1	**1**
1	1	1	0	0	**1**
1	1	1	0	1	**1**
1	1	1	1	0	**1**
1	1	1	1	1	**1**
a + b + ! c + d + ! e					



1.

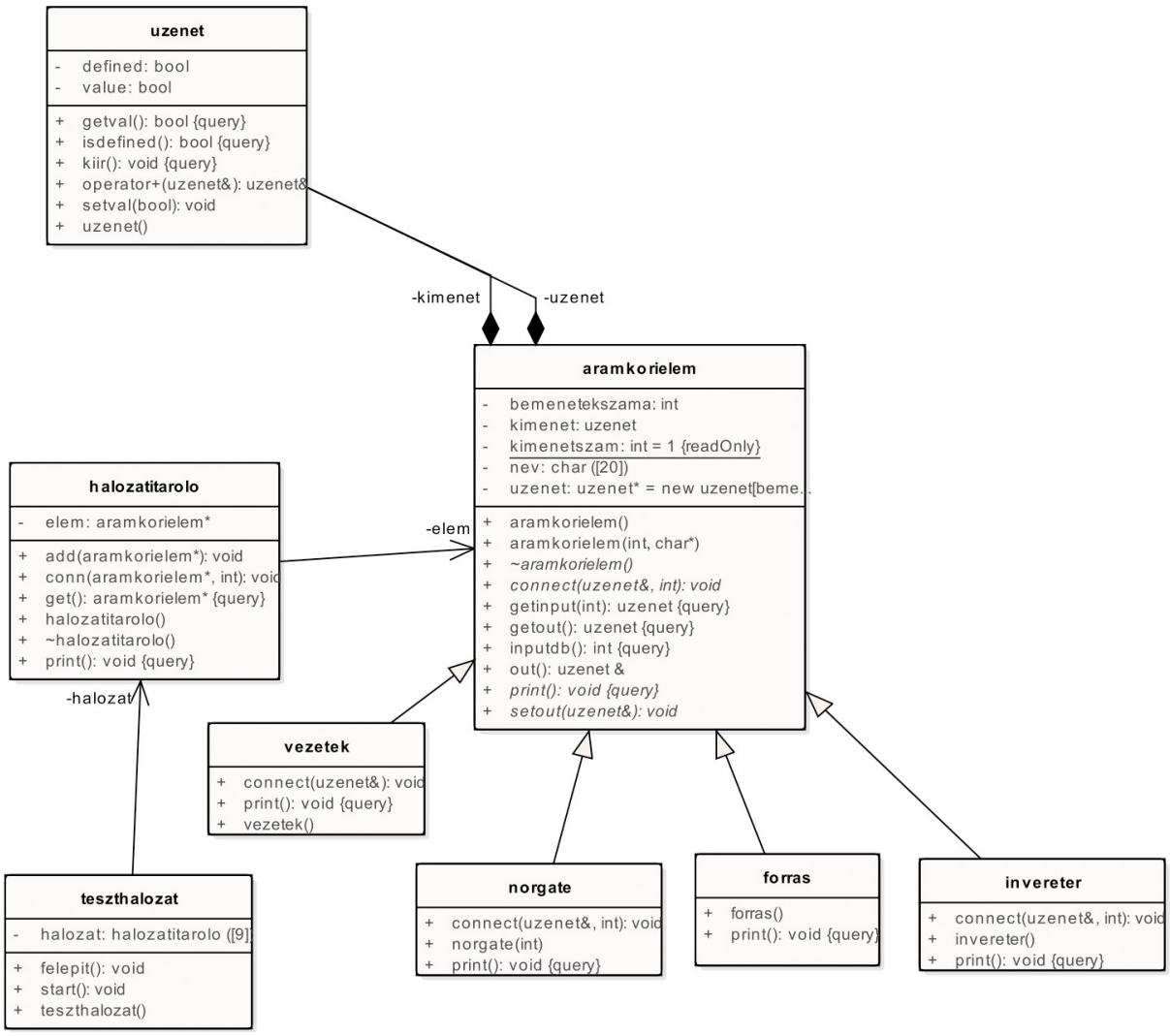
A hálózat megvalósítása „felépítése” a programon belül történik, konzol felületen lesz lehetőség a már felépített hálózat bemeneti változóinak (EDCBA) megadására.

A program nem fogad el csak 1 est vagy 0-át, mást karakternél kivétel keletkezik, amit a program jelezni is fog a felhasználó felé, hogy érvénytelen bemenetet adott meg.

3. Terv

A feladatban 6+1.db objektumra van szükség. 6db áramköri elem és +1 db heterogén tároló. (A program tároló nélkül is működőképes).

3.1. Objektum terv



A digitális áramkör a fent látható objektumokkal lesz megvalósítható. Ezen objektumokkal minden kombinációs hálózat felépíthető és szimulálható. Későbbiekben bővítésre is van lehetőség pl. sima vagy kapu felépíthető egy nor kapu és egy inverter segítségével..

3.2. A program működése/algoritmusai

A program alap működése a következő: Minden áramköri elem egy objektum, mindannyian üzenet objektumokat tárolnak, kompozícióban tehát rendelkeznek az üzenet élettartama felett.

3.2.1. Üzenet

2 bool privát adattaggal rendelkezik:

- defined: Adtunk e már értéket neki(true), vagy még csak létre lett hozva(false)azaz undefined.
- value: Az üzenet értéke 1(true) vagy 0 (false)

Saját insert operátora van, ami, ha definiált az üzenet az értékét tölti be az osstrembe, ha pedig még nem definiált akkor a „Nem definiált” szöveget.

Operator+: összeadja a 2 üzenet értékét, ha definiáltak. Azaz

0+0=0
0+1=1
1+1=0

getval: Ha definiált az üzenet visszaadja az értékét.

Kiir: Kiírja konzolra az üzenet értékét, ha definiálva van, az insert <<operátor felhasználásával.

isdefined: Ha definiált az üzenet visszatér igazzal, ha nem akkor pedig hamissal.

setval: Beállítja az üzenet értékét, valamint átállítja definiáltra.

uzenet: Konstruktor. Létrehozza az objektumot alapértelmezetten nem definiálta. Későbbiekben a setval-lal lehet neki értéket adni.

3.2.2. Forrás

Vezérelhető forrás, csak kimenettel rendelkező áramköri elem. Kimenete std::osstremmel állítható létrehozásakor.

print: Kiírja a kimenetét.

3.2.3. Vezeték

Egyszerű áramköri elem, amilyen üzenetet kap a bemenetére azt továbbítja a kimenetére.

connect: Csatlakoztatja a vezetéket. Megvalósítja a vezeték működését.

print: Kiírja a vezeték be és kimenetén lévő értékeket a hívásakor.

vezetek: Konstruktor, létrehozza a vezetéket 1 bemenettel és vezetéknévvel.

3.2.4. Invereter

A vezetéknél eggyel bonyolultabb áramköri elem,1 be és 1 kimenete van, negálja a bemenetére érkező üzenetet.

connect: Csatlakoztatja az invertert. Megvalósítja az inverter működését.

print: Kiírja az inverter be és kimenetén lévő értékeket a hívásakor.

inverter: Konstruktor, létrehozza az invertert 1 bemenettel és inverter névvel.

3.2.5. Norgate

Nem-vagy kapcsolatot megvalósító áramköri elem. Tetszőleges bemenettel rendelkezik. Amikor minden bemenetére kötöttünk egy áramköri elemet, elvégzi a nem-vagy kapcsolatot.

- A nem-vagy kapcsolat megvalósítása: Egy áramköri elem csatlakoztatásakor, alapértelmezetten 1(true) értéket ad vissza, de végigmegy az összes láb állapotán, ha talál már definiált 1(true) állapotú lábat, visszatér 0(false) értékkel hiszen ilyenkor a vagy kapcsolat az összes többi láb értékétől függetlenül igazat adna, tehát a nem-vagy kapu hamisat.

`connect:` Csatlakoztatja a nemvagy kaput. Megvalósítja a működését.

`norgate:` Konstruktor, létrehozza a nemvagy kaput argumentumban megadott méretű bemenettel és norkapu névvel.

`print:` Kiírja a nemvagykapu bemenetein és kimenetén lévő értékeket a hívásakor.

3.2.6. Teszthálózat

Megvalósítja a feladatban megadott kombinációs hálózatot. (1.ábra)

- Ez csak példa nem szükséges a program működéséhez, az áramköri elemek akár a main-ben is létrehozhatóak és csatlakoztathatok egymáshoz.

`felepit`: Csatlakoztatja a hálózat elemeit egymáshoz az ábrának megfelelően.

`start:` Lefuttatja a szimulációt és kiírja osstremre a végeredményt.

`teszthalozat:` Konstruktor, memóriát foglal a hálózati elemeknek és a tárolóba helyezi őket.

3.2.7. Hálózati tároló

- Példa heterogén kollekcióra

add: Elem behelyezése a tárolóba.

conn: Elem csatlakoztatása a tárolóban lévő elemez.

get: Visszaadja a tárolóban lévő elemre mutató pointert.

halozatitarolo: Konstruktor létrehozza a tárolót NULL-ra mutató elem pointerrel.

~halozatitarolo: Destruktor, felszabadítja a rábízott elemeknek foglalt memóriát.

print: Kiírja a benne lévő elem állapotát.

3.2.8. Áramköri elem

Ez az alaposztály.

- A legtöbb függvénye önleíró egyedül a connect igényel leírást.
- A connect virtuális függvény valósítja meg az elemek csatlakoztatását, de nem közvetlenül áramköri elemet adjuk meg paraméterként, hanem a csatlakoztatni kívánt áramköri elem kimenetén lévő üzenetet (ezt az out tagfüggvénnyel tesszük), valamint paraméterként megadjuk, hogy melyik lábra szeretnénk csatlakoztatni.
- A print függvény, tisztán virtuális, minden leszármazott felülírja, ez a függvény kiírja az adott áramköri elem bemeneténekbemeneteinek állapotát, valamint a kimenetének állapotát a hívása pillanatában.

További függvényei:

aramkorielem: Konstruktor. Létrehozza az objektumot a megadott bemenet mérettel és névvel.

~aramkorielem: Destruktor, felszabadítja az üzeneteknek foglalt memóriát.

getinput: Az argumentumban megadott bemeneten lévő értékkel tér vissza, ha a bemenet létezik.

getout: Lekérdezhető a kimenet állapota.

inputdb: Megadja hány darab bemenettel rendelkezik az adott elem.

out: A kimenettel tér vissza, pontosabban a kimeneten lévő üzenet referenciájával.

setout: Beállítja a kimenet értékét.

4. A tesztprogram

4.1. Tesz A

4.1.1. Forrás tesztjei

Létrehoz egy forrást. Teszteli, hogy létre jön-e.
Beállítja a forrást. A kimenetén elvárt értéket ad a forrás?
Teszteli, hogy tiltott bemenetet elfogad-e a forrás.

4.1.2. Vezeték tesztjei

Létrehoz egy vezetéket. Teszteli, hogy létre jön-e.
Összeköti a már létrehozott forrással a vezetéket. Megnézi a kimenetét

4.1.3. Inverter tesztjei

Létrehoz egy invertert, Teszteli, hogy létrejött-e.
Összeköti a bemenetét a vezetékkel, (tehát így a forrással). Megnézi, hogy tényleg invertálja-e a bemenetre kapott értéket.

4.1.4. Norgate tesztjei

Létrehoz egy 2 bemenetű nemvagy kaput. Teszteli, hogy létrejött-e.
Összeköti az inverter kimenetével az egyik bemenetét. A másik bemenetére létrehoz egy másik forrást és azzal köti össze. Teszteli, hogy megvalósítja-e a nemvagy kapcsolatot.

4.1.5. Hálózati tároló tesztjei

Dinamikusan létrehoz áramköri elemeket és hálózati tárolóba teszi őket. Teszteli, hogy bele kerültek-e és hogy továbbra is megfelelően működnek-e. Ezek után megnézi, hogy a hálózati tároló tényleg felszabadítja e a rábízott objektumok által lefoglalt memóriát, ha már nincs használva.

4.2. Teszt B

4.2.1. Teszthálózat tesztjei

Az elkészített mintahálózatot teszteli, hogy különböző bemeneti változókombinációk esetén az elvárt kimenetet kapjuk-e.

4.3. Teszt C

Felépít egy másik tesszthálózatot, különböző bemenetekre figyeli a kimenet értékét. Közben figyeli „menet közben” is az adott áramköri elemek értékét, tehát nem csak az utolsó kimenetet nézi, hanem nyomon követi az adott üzenet értékének alakulását.