Házi feladat

Feladatválasztás/specifikáció

Programozás alapjai 2.

Pálinkás Lőrinc Mihály - XB0SMF

Tartalom

[1. Feladat 2](#_Toc162177167)

[Digitális áramkör 2](#_Toc162177168)

[2. Feladatspecifikáció 3](#_Toc162177169)

[Feladat általános leírása 3](#_Toc162177170)

[Megvalósított áramköri elemek 3](#_Toc162177171)

[Bemenet formátuma 3](#_Toc162177172)

# Feladat

## Digitális áramkör

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

* NOR kapu
* vezérelhető forrás
* összekötő vezeték
* csomópont

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!

# Feladatspecifikáció

## Feladat általános leírása

A program lehetőséget ad digitális áramkörök szimulálására. A felhasználó áramköröket képes betölteni szöveges file-okból, beállítani a bemeneti jelkombinációt és ez alapján kiolvasni a kimeneti jel(ek)et.

## Megvalósított áramköri elemek

A következő elemeket képes szimulálni az áramkör:

* Forrás: állítható 0 és 1 kimeneti jelekkel
* Vezeték: két komponens egy-egy lábát tudja összekötni
* Csomópont: 1 bemeneti jelet több kimeneti irányba tud továbbítani
* Kapu: 1 vagy 2 bemenetből képes pontosan 1 kimenetet produkálni. Megvalósított kapuk:
  + AND
  + OR
  + XOR
  + NOT
  + NAND
  + NOR
  + XNOR
* Végpont: a kapott jelet adja ki eredmények lekérdezésekor

A bonyolultabb elemeket (pl. több lábú kapuk) egyelőre nem implementáljuk, mert könnyen felépíthető ezekből szimuláció során, de ha marad idő, akkor ezeket is megvalósíthatjuk.

## Bemenet formátuma

Az áramkörök felkonfigurálása szöveges file alapján történik. Ebben megadja a felhasználó a csomópontok számát (ezt pillanat múlva kicsit pontosítjuk) majd felsorolja a további komponenseket, megadva, hogy melyik csomópontra kapcsolódnak. Tehát a konfigurációs file ehhez hasonlóan néz ki:

*test.txt*

VERTEX = 6

SOURCE: 1,2,3

AND: (1,2,4)[;(…,…,…);…] <- ha több van

OR: (2,3,5);…

XNOR: (4,5,6)

END: 6

Fontos megjegyzés: A szimuláció során az összekötő vezetékeket is csomópontnak tekintünk, így tudjuk megadni hogy mely lábakat kell összekötni könnyen formátumosan, illetve áramkörön belül a kiolvasás során is logikus, ha vezetékek jeleit is ki tudjuk olvasni.

A példa azt is mutatja hogy milyen egy kapu megadásának általános formátuma:

GATE\_NAME: (IN1, IN2, OUT1) ; …

Természetesen 1 bementű (gyakorlatilag a NOT) kapu esetén csak 1 db IN láb van. Emellett a SOURCE-oknak csak OUT lába, míg az END-eknek csak IN lába van.

## Kimenet formátuma

A felhasználó képes lekérdezni az áramkör kimeneteinek jeleit, ezeket lekérdezéskor egyesével felsorolja. Emellett konkrét csomóponttól is lehetséges lekérdezni jelszintet (valójában az eredmény is ez, csak ott a felhasználónak nem manuálisan kell egyesével lekérdezni, hanem automatán adja az eredményeket).

# Pontosított specifikáció

## TBA

Majd kiderül hogy ide mi kell…

## Probléma: önhivatkozás

A feladat specifikálásakor már felmerült a gondolat: hogyan kezelhetjük az önhivatkozást, azaz amikor egy elem valamilyen módon (közvetlen vagy közvetett) a saját kimenetére támaszkodik (pl. 2 NOT kapus D flip-flop esetén nem is olyan értelmetlen kérdés).

A problémára jelenleg még nem adok választ, egyeztetés után ezt pontosítjuk.

# Terv

## Objektum terv