

7. Prototípus koncepció

54 – *Override*

Konzulens:

dr. László Zoltán

Csapattagok:

Kriván Bálint	CBVOEN	balint@krivan.hu
Jákli Gábor	ONZ5G1	j_gab666@hotmail.com
Dévényi Attila	L1YRH0	devenyat@gmail.com
Apagyi Gábor	X8SG3T	apagyi.gabooo@gmail.com
Péter Tamás Pál	N5ZLEG	falconsaglevlist@gmail.com

2011. március 23.

Tartalomjegyzék

7	Prototípus koncepciója	4
7.1.	Prototípus interface-definíciója	4
7.1.1.	Az interfész általános leírása	4
7.1.2.	Bemeneti nyelv	4
7.1.3.	Konfigurációs fájlok nyelvtana	5
7.1.4.	Kimeneti nyelv	8
7.2.	Összes részletes use-case	9
7.3.	Tesztelési terv	10
7.4.	Tesztelést támogató segéd- és fordítóprogramok specifikálása	10
7.5.	Napló	10

Ábrák jegyzéke

7. Prototípus koncepciója

7.1. Prototípus interface-definíciója

[Definiálni kell a teszteket leíró nyelvet. Külön figyelmet kell fordítani arra, hogy ha a rendszer véletlen elemeket is tartalmaz, akkor a véletlenszerűség ki-bekapcsolható legyen, és a program determinisztikusan is tesztelhető legyen.]

7.1.1. Az interfész általános leírása

[A protó (karakters) input és output felületeit úgy kell kialakítani, hogy az input fájlból is vehető legyen illetőleg az output fájlba menthető legyen, vagyis kommunikációra csak a szabványos be- és kimenet használható.]

7.1.2. Bemeneti nyelv

A parancsokat a standard bemenetről, illetve fájlból olvassa be a program. Minden parancsot egy sorvége karakter zár le.

Megjegyzés: minden parancs ad visszajelzést a felhasználónak a végrehajtott eseményről, ennek formátuma a Kimeneti nyelv c. fejezetben olvasható.

loadCircuit [file]

- Leírás: A megadott áramkört betölti a szimulációs program.
- Megjegyzés: A file nevét kiterjesztés nélkül kell megadni.
- Opciók: -

loadSettings [file]

- Leírás: A jelenlegi áramkörhöz a megadott konfigurációs fájl betöltése.
- Megjegyzés: A file nevét kiterjesztés nélkül kell megadni.
- Opciók: -

saveSettings [file]

- Leírás: A pillanatnyilag használt konfiguráció fájlba mentése.
- Megjegyzés: A file nevét kiterjesztés nélkül kell megadni.
- Opciók: -

switch [név]

- Leírás: A megnevezett kapcsoló átállítása.
- Opciók: -

setSeqGen [név] [érték1, érték2, ...]

- Leírás: A megnevezett szekvenciagenerátor az értékparaméterek szerint beállítódik.
- Opciók: -

getValue [név]

- Leírás: A megadott áramköri elem kiírja az aktuális kimeneti értékét.

- Opciók: a `getValue` -all parancs az összes áramköri elem kimenetének értékét kilistázza.

step

- Leírás: A parancs hatására lefut egy szimulációs ciklus, melynek két eredménye lehet:
 - véges lépésen belül stabilizálódik a rendszer, ekkor a kapcsoló(k), szekvenciagenerátor(ok) és ki-jelző(k) értéke(i) kiíródnak.
 - nem stabilizálódik az áramkör; hibaüzenet
- Opciók: -

7.1.3. Konfigurációs fájlok nyelvtana

A konfigurációs fájlok *.ovr kiterjesztésűek, ezekben adjuk meg a szimulálandó hálózat paramétereit. Egyszerű szövegfájl, melyben az értelmezendő parancsok soronként tagolódnak. A program feltételezi a konfigurációs fájl hibamentességét, sehol nem ellenőrizzük, hogy a bemenetnek van-e értelme! A fájl létrehozásához az alábbi parancsok, szintaxisok állnak rendelkezésre:

- `X=...`
 - Leírás: komponens létrehozás. Ezzel a paranccsal az egyenlőség jel után megadott komponenst hozzuk létre, melynek kimenetére ezek után az "X"-el hivatkozhatunk. Amennyiben több kimenetű komponensről beszélünk akkor az egyenlőség bal oldala egy tömböt jelent. Ebben az esetben az egyes kimenetekre a későbbiekben `X[i]`-vel hivatkozhatunk (a 0 és N-1 között).
 - Opciók: A lehetséges komponensek az implementált komponensek listájából választható, melyeknek paraméterül az egyes komponensekhez tartozó meghatározott paramétereket át kell adni.
 - * `OR(...)`
 - * `AND(...)`
 - * `INVERTER(...)`
 - * `VCC(...)`
 - * `GND(...)`
 - * `MPX(...)`
 - * `FFJK(...)`
 - * `FFD(...)`
 - * `LED(...)`
 - * `7SEG(...)`
 - * `TOGGLE(...)`
 - * `SEQ(...)`
 - * `WIRE(...)`
 - * `NODE(...)`
- `OR(name,WIRE[n])`
 - Leírás: vagy kapu létrehozása.
 - Opciók:
 - * `name`: meg kell adni a kapcsoló nevét (egyedi azonosításra szolgál - a külvilág felé).
 - * `WIRE[n]`: fel kell sorolni a kapu bemenetére kötött WIRE típusú változók neveit. N bemenetű kapu esetén ide N db WIRE kerül. A kapu számot nem kell megadni, azt a parser automatikusan észleli a megkapott paraméterek számából. Minimum 2 bemenetet meg kell adni.

- Példa: OR(vagy1,w_kapcs1_vagy1,w_kapcs2_vagy1,w_kapcs3_vagy1) - három vezeték rákapcsolása a kapura, mely így egy három bemenetes vagy kapu lesz.
- AND(name,WIRE[n])
 - Leírás: "és" kapu létrehozása.
 - Opciók:
 - * name: meg kell adni a kapu nevét (egyedi azonosításra szolgál - a külvilág felé).
 - * WIRE[n]: u.a. mint a vagy kapu esetén.
 - Példa: AND(es1,w_kapcs1_es1,w_kapcs2_es1) - két vezeték rákapcsolása a kapura, mely így egy két bemenetes vagy kapu lesz.
- INV(name,WIRE_in)
 - Leírás: inverter létrehozása.
 - Opciók:
 - * name: meg kell adni a kapu nevét (egyedi azonosításra szolgál - a külvilág felé).
 - * WIRE_in: az inverter bemeneti vezetéket kell megadni. Csak egy bemenetű inverter létezik.
 - Példa: INV(inv1,w_kapcs1_inv1) - egy vezeték rákapcsolása a kapura.
- VCC(name)
 - Leírás: konstans igaz jel létrehozása.
 - Opciók:
 - * name: meg kell adni a komponens nevét (egyedi azonosításra szolgál - a külvilág felé).
 - Példa: VCC(vcc1)
- GND(name)
 - Leírás: konstans hamis jel létrehozása.
 - Opciók:
 - * name: meg kell adni a komponens nevét (egyedi azonosításra szolgál - a külvilág felé).
 - Példa: GND(gnd1)
- MPX(name,WIRE_in[4],WIRE_S[2])
 - Leírás: 4:1 multiplexer létrehozása.
 - Opciók:
 - * name: meg kell adni a komponens nevét (egyedi azonosításra szolgál - a külvilág felé).
 - * WIRE_in[4]: meg kell adni négy WIRE-t, ami az egyes bemeneteket jelentik.
 - * WIRE_S[2]: meg kell adni két WIRE-t, ami a két select jelet adja
 - Példa: MPX(mpx1,w_in3,w_in2,w_in1,w_in0,w_s1,w_s0)
- FFK(name,WIRE_clk,WIRE_J,WIRE_K)
 - Leírás: konstans hamis jel létrehozása.
 - Opciók:
 - * name: meg kell adni a komponens nevét (egyedi azonosításra szolgál - a külvilág felé).
 - * WIRE_clk: meg kell adni egy WIRE-t, ami az órajel bemenet lesz
 - * WIRE_J: meg kell adni egy WIRE-t, ami az J bemenet lesz
 - * WIRE_K: meg kell adni egy WIRE-t, ami az K bemenet lesz

- Példa: FFJK(ffjk1,w_clk,w_j,w_k)
- FFD(name,WIRE_clk,WIRE_D)
 - Leírás: D flip-flop létrehozása.
 - Opciók:
 - * name: meg kell adni a komponens nevét (egyedi azonosításra szolgál - a külvilág felé).
 - * WIRE_clk: meg kell adni egy WIRE-t, ami az órajel bemenet lesz
 - * WIRE_D: meg kell adni egy WIRE-t, ami az adat bemenet lesz
 - Példa: FFD(ffd1,w_clk,w_d)
- LED(name,WIRE_in)
 - Leírás: LED létrehozása.
 - Opciók:
 - * name: meg kell adni a komponens nevét (egyedi azonosításra szolgál - a külvilág felé).
 - * WIRE_in: meg kell adni egy WIRE-t, ami a LED bemenete lesz
 - Példa: LED(led1)
- 7SEG(name,WIRE_D[8])
 - Leírás: 7 szegmenses kijelző létrehozása.
 - Opciók:
 - * name: meg kell adni a komponens nevét (egyedi azonosításra szolgál - a külvilág felé).
 - * WIRE_D[8]: meg kell adni nyolc WIRE-t, ami a sorra a szegmenseket kapcsolja
 - Példa: 7SEG(7seg1,w_d7,w_d6,w_d5,w_d4,w_d3,w_d2,w_d1,w_d0)
- TOGGLE(name)
 - Leírás: kapcsoló létrehozása.
 - Opciók:
 - * name: meg kell adni a kapcsoló nevét (egyedi azonosításra szolgál - a külvilág felé).
 - Példa: TOGGLE(kapcs1)
- SEQ(name,BITMINTA)
 - Leírás: szekvencia generátor létrehozása.
 - Opciók:
 - * name: meg kell adni a komponens nevét (egyedi azonosításra szolgál - a külvilág felé).
 - * BITMINTA: meg kell adni egy sorozatot, melyet a generátor egymás után kiad magából.
 - Példa: SEQ(seq1,011000110)
- WIRE(name,WIRE_in)
 - Leírás: vezeték létrehozása.
 - Opciók:
 - * name: meg kell adni a vezeték nevét (egyedi azonosításra szolgál - a külvilág felé).
 - * WIRE_in: meg kell adni egy NODE-ot, vagy egy komponens kimenetet a vezeték bemenetének
 - Példa: WIRE(w1,X) - WIRE(w1,Node)
- NODE(name,WIRE_in,out)

- Leírás: csomópont létrehozása.
- Opciók:
 - * name: meg kell adni a csomópont nevét (egyedi azonosításra szolgál - a külvilág felé).
 - * WIRE_in: meg kell adni egy WIRE-t a csomópont bemenetének
 - * out: meg kell adni, hogy a NODE-nak hány kimenete lesz
- Példa: NODE(n1,w1,3) - három kimenetű elosztót hoz létre a w1 vezeték alapján
- Példa áramkör konfigurációs fájl Egy olyan minta hálózatot hozunk létre melyben található két kapcsoló egy és kapura kötve és az és kapu kimenete egy inverteren keresztül egy ledre kapcsolódik.
 - t1=TOGGLE(t1)
 - t2=TOGGLE(t2)
 - w_t1_es1=WIRE(w_t1_es1,T1)
 - w_t2_es1=WIRE(w_t1_es1,T2)
 - es1=AND(es1,w_t1_es1,w_t2_es1)
 - w_es1_inv1=WIRE(w_es1_inv1,es1)
 - inv1=INV(inv1,w_es1_inv1)
 - w_inv1_led1=WIRE(w_inv1_led1,inv1)
 - led1=LED(led1,w_inv1_led1)

7.1.4. Kimeneti nyelv

A program történései, visszajelzése a standard kimeneten jelennek meg, illetve ezek fájlba is kiíródnak. A program minden parancs után visszajelzést ad a felhasználónak a végrehajtott eseményről. A fentebb definiált parancsokra a következő jelzéseket kapja a felhasználó:

loadCircuit [file]

Lehetséges kimenetek

- áramkör betöltése sikeres
 - Leírás: a betöltés sikeres, amennyiben az áramkört tartalmazó fájl szintaktikája megfelel a Konfigurációs fájlok nyelvtana c. fejezetnek.
- áramkör betöltése sikertelen: hibás áramkör
 - Leírás: a betöltés sikertelen, amennyiben az áramkört tartalmazó fájl szintaktikája nem felel meg a Konfigurációs fájlok nyelvtana c. fejezetnek.

loadSettings [file]

Lehetséges kimenetek

- értékek mentése sikeres
 - Leírás: a konfigurációs értékek sikeresen fájlba mentődtek.

saveSettings [file]

Lehetséges kimenetek

- értékek betöltése sikeres
 - Leírás: az értékek betöltése sikeres, amennyiben a konfigurációs fájlban szereplő áramköri elemek megfeleltethetők az aktuális áramkörben szereplő elemekkel, illetve a megadott értékek helyesek.
- értékek betöltése sikertelen: hibás értékek

- Leírás: az értékek betöltése sikeres, amennyiben a konfigurációs fájlban szereplő áramköri elem nem feleltethető meg az aktuális áramkörben szereplő elemek egyikével sem, illetve ha valamelyik érték helytelen.
- Megjegyzés: azon elemek, melyek beállítására nem volt információ a konfigurációs fájlban automatikusan nullázódnak.

switch [név]

Lehetséges kimenetek

- [elem]: [érték]
 - Leírás: az [elem] megadja a módosított kapcsoló nevét, míg az [érték] megmutatja, hogy milyen értékre változott az aktuális kapcsoló kimenete.

setSeqGen [név] [érték1, érték2, ...]

Lehetséges kimenetek

- [elem]: [érték1, érték2, ...]
 - Leírás: az [elem] megadja a módosított generátor nevét, míg az [érték1, érték2, ...] megmutatja, hogy milyen értékekre változott az aktuális generátor kimenete.

getValue [név]

Lehetséges kimenetek

- [elem]: [érték]
 - Leírás: az [elem] megadja a módosított kapcsoló nevét, míg az [érték] megmutatja, hogy milyen értékre változott az aktuális kapcsoló kimenete.
 - Megjegyzés: a getValue -all parancsra az összes elemet kilistázza a megadott formában új sor karakterrel elválasztva

step

Lehetséges kimenetek

- szimuláció sikeres; összes elem kilistázása, mintha hívnánk egy getValue -all függvényt
 - Leírás: a szimuláció sikeres, amennyiben véges lépésen belül stabilizálódni tud az áramkör.
- szimuláció sikertelen: nincs stabil állapot
 - Leírás: a szimuláció sikertelen, amennyiben véges lépésen belül nem tud stabilizálódni az áramkör.

7.2. Összes részletes use-case

Use-case neve	Áramkör betöltése
Rövid leírás	Az áramkört leíró fájl betöltése
Aktorok	Felhasználó
Forgatókönyv	A loadCircuit parancsot használva betöltheti az áramkört leíró fájlt, amely a program követelményeinek megfelel

Use-case neve	Konfiguráció betöltése
Rövid leírás	Egy áramkör konfigurációjának betöltése
Aktorok	Felhasználó
Forgatókönyv	A loadSettings paranccsal betölt egy egyedi a konfigurációt az áramkörhöz, amely például tartalmazhatja a szekvencia generátorok által kiadott bitsorozatok vagy a kapcsolók állását.

Use-case neve	Konfiguráció mentése
Rövid leírás	Áramkör konfigurációjának mentése
Aktorok	Felhasználó
Forgatókönyv	A saveSettings parancs kiadásával menti az aktuális áramkör konfigurációját.

Use-case neve	Kapcsoló kapcsolása
Rövid leírás	Kapcsoló állásnak módosítás
Aktorok	Felhasználó
Forgatókönyv	Az adott áramkörben a neve alapján azonosított kapcsoló állásának módosítása a switch parancs használatával.

Use-case neve	Szekvenciagenerátor módosítás
Rövid leírás	Szekvenciagenerátor bitsorozatának megadása
Aktorok	Felhasználó
Forgatókönyv	Az adott áramkörben a neve alapján azonosított szekvenciagenerátor által kiadott bit-sorozat megadása a setSeqGen paranccsal.

Use-case neve	Érték lekérdezése
Rövid leírás	Egy, az áramkörben lévő alkatrész értékének lekérdezése
Aktorok	Felhasználó
Forgatókönyv	Az adott áramkörben a getValue parancs használatával a megadott nevű alkatrész értékének lekérdezése.

Use-case neve	Áramkör szimulálása
Rövid leírás	A betöltött áramkör szimulálása
Aktorok	Felhasználó
Forgatókönyv	A step parancs kiadásával szimulálja a betöltött áramkört.

7.3. Tesztelési terv

[A tesztelési tervben definiálni kell, hogy a be- és kimeneti fájlok egybevetésével miként végezhető el a program tesztelése. Meg kell adni teszt forgatókönyveket. Az egyes tesztek elég informálisan, szabad szöveggént leírni. Teszt-esetként egy-öt mondatban. Minden teszthez meg kell adni, hogy mi a célja, a proto mely funkcionálisát, osztályait stb. teszteli. Az alábbi táblázat minden teszt-esethez külön-külön elkészítendő.]

Teszt-eset neve	...
Rövid leírás	...
Teszt célja	...

7.4. Tesztelést támogató segéd- és fordítóprogramok specifikálása

[Specifikálni kell a tesztelést támogató segédprogramokat.]

7.5. Napló

Kezdet	Időtartam	Résztevők	Leírás
2011.03.22.12:00	2,5 óra	Apagyi G.	Prototípus konfigurációs nyelvének definiálása