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	devenyiat@gmail.com
Apagyi Gábor	X8SG3T	apagyi.gabooo@gmail.com
Péter Tamás Pál	N5ZLEG	falconsaglevlist@gmail.com

2011. március 22.

Tartalomjegyzék

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 (karakteres) 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 kijelző(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 autómatikusan 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 vezetékét 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)

- FFJK(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_d,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

[Egyértelműen definiálni kell, hogy az egyes bemeneti parancsok végrehajtása után előálló állapot milyen formában jelenik meg a szabványos kimeneten.]

7.2. Összes részletes use-case

[A use-case-eknek a részletezettsége feleljen meg a kezelői felületnek, azaz a felület elemeire kell hivatkozniuk. Alábbi táblázat minden use-case-hez külön-külön.]

7.1. ábra. x

Use-case neve	**...**
Rövid leírás	...
Aktorok	...
Forgatókönyv	...

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 teszteket elég informálisan, szabad szövegként leírni. Teszt-esetenként egy-öt mondatban. Minden teszthez meg kell adni, hogy mi a célja, a proto mely funkcionalitását, 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**	**Résztvevők**	**Leírás**
2011.03.22.12:00	2,5 óra	**Apagyi G.**	Prototípus konfigurációs nyelvének definiálása