BOHUS MIKLÓS—CSOPAKI GYULA—GEFFERTH LÁSZLÓ—HALÁSZ EDIT—DR. NÉMETH GÁBOR—TRÓN TIBOR—VARRÓ LÁSZLÓ
BME Híradástechnikai Elektronika Intézet

A LOGAN logikai szimulációs program*

ETO 621.3.049.7-111:681.3.06 LOGAN

Digitális rendszerek realizálása során elengedhetetlen gazdaságossági követelmény, hogy a rendszer konkrét kivitelezése, gyártása előtt meggyőződjünk az elkészített tervek helyességéről. Az ellenőrzés hagyományos „kézi" módja rendkívül fáradtságos és sok időt igénylő munka, hiszen még az egyes funkcionális egységek is nagyon sok elemből állhatnak. A tervek fokozatos részletezése, illetve a logikai kapcsolási rajzok többszöri átrajzolása során esetleg előforduló elkötések, téves bekötések helyének kimutatása, a még megengedhető terhelési és késleltetési viszonyok vizsgálata, a meg nem engedett vezérlések és állapotok felderítése részletes és alapos ellenőrzést kíván. Egy számítógépes logikai szimulációs program létjogosultságát a felsoroltakon túlmenően az is indokolja, hogy a tervek kisebb-nagyobb módosításának hatása — ha csak nem elemi változtatásról van szó — csak a tervezés ismételt elvégzésével azonos nagyságrendű munkával állapítható meg.

A Számítástechnikai Koordinációs Intézet megbízásából a Budapesti Műszaki Egyetem Híradástechnikai Elektronika Intézetében kidolgozott és most ismertetésre kerülő LOGAN programrendszer [1] alkalmas szinkron működésű szekvenciális hálózatok logikai, áramköri és vezérlési tulajdonságainak vizsgálatára. A programrendszer kidolgozásánál az alábbi szempontokat vettük alapul:

- A vizsgálati eljárás legyen teljes, tegye lehetővé bármely elem és a teljes hálózat működésének ellenőrzését;
- a hálózat elemeinek a számításokhoz szükséges adatai egy mágnesszalag-katalóguson álljanak rendelkezésre, hogy azokat ne kelljen a felhasználónak minden egyes feladat során megadnia;
- a katalógusba felvett elemkészlet elegendően bő választékot biztosítson;
- legyen független az elemek konkrét fizikai tulajdonságaitól (sebesség, hőmérséklet határ, nagyjelű és nagy zavarérzékenységű elemek stb);
- bármely feladathoz tartozó adatok gépi módon tárolhatók legyenek, hogy esetleges későbbi részletesebb vizsgálatokhoz újból egyszerűen hozzáférhetők legyenek;
- a programrendszer által biztosított vizsgálatok, ellenőrzések közül a felhasználó kívánsága szerint (fakultatíve) választhasson;
- a programrendszer felépítése tegye lehetővé a későbbi bővítéseket mind a vizsgálható hálózatok (pl. aszinkron), mind a felhasználható elemkészlet (pl. funkcionális egységek, MSI elemek, nagysebességű elemek) terén;
- szolgáljon alapul a konstrukciós tervezéshez;
- biztosítsa több hálózat egymás után sorozatban történő vizsgálatát külön külső beavatkozás nélkül is;
- a programrendszer legyen felhasználó orientált, alkalmazása ne igényeljen programozási ismereteket, használható legyen a programrendszer részletes ismerete nélkül is.

1. A programrendszer felépítése, működése, szolgáltatásai

A programrendszerrel végezhető ellenőrzések, vizsgálatok két nagyobb csoportra oszthatók.

A hálózat felépítésére és az áramköri működés ellenőrzésére vonatkozó vizsgálatok során kimutathatók és meghatározhatók

- a hiányzó, nem értelmezhető és tilos összeköttetések,
- a forrás nélküli bemenetek és fogyasztó nélküli kimenetek,
- a megépítéshez szükséges integrált áramköri tokok száma,
- az egyes elemek terhelési viszonyai és
- a hálózaton belüli késleltetések.

A hálózat logikai analízise során meghatározható

- a hálózat bármely pontján levő jel logikai értéke a bemenő jelek és a tárolók belső állapotának tetszőleges értéke esetén,
- adott bemeneti vezérléssorozat hatására létrejövő jelsorozat a hálózat bármely pontján vagy kimenetén,
- a bemeneti vezérlés hatására kialakuló tárolóállapotok,
- a hálózat minden olyan eleme, melynek kimenetén a vezérléstől függetlenül állandó logikai szint található,
- a hálózat tetszőleges pontjához rendelhető logikai egyenlet.

A fenti vizsgálatok előtt minden esetben szükséges a hálózatra vonatkozó adatok beolvasása és ellenőrzése, továbbá — kiegészítve a katalógus adatokkal — a programrészek számára egységes adatbázissá alakítása.

A felsorolt vizsgálatokat, feladatokat a programrendszer egy-egy szubrutinja végzi, melyek egymástól függetlenül működnek és hívhatók. Aktivizálásukat — részben a felhasználó igényeinek megfelelően (vezérkártyák) — egy vezér programszegmens

Beérkezett: 1972. III. 4.
* A munka a Számítástechnikai Koordinációs Intézet megbízásából készült.

biztosítja. A szubrutinok közös adatmezőből (COMMON) dolgoznak, a vizsgálatok eredményeit a sornyomtatón közlik. Egyes szubrutinok külön input rendszerrel is rendelkeznek a közös alapadatokon kívül szükséges további információk beolvasására.

A programrendszer szerves részét képező katalógusszalag bővítésére, elemeinek cseréjére vagy törlésére, a katalógusba felvett elemek listájának vagy a katalógus teljes tartalmának kiíratására külön program (KATAL) készült, mivel ezek a feladatok az áramköri és logikai analízistől elkülönülő egységet képeznek.

A LOGAN programrendszer a fentebb elmondottak alapján az alábbi egységekből épül fel:

1. VEZER programszegmens:

Az egész programrendszer keretprogramja, irányítja az egyes vizsgálatokat végző szubrutinokat. A vizsgálatok elvégeztével biztosítja — amennyiben szükséges — a hálózat összes adatának mágnesszalagra vitelét, illetve újbóli vizsgálat során onnan beolvasását, valamint a programrendszer futásának újraindítását, ha további hálózat vizsgálatára is szükség van.

2. INPUT szubrutin:

Új feladat esetén beolvassa és ellenőrzi a vizsgálandó hálózat alapadatait, közli az adatmegadás hibáit. Hibátlan adatmegadásnál összeállítja a közös adatbázist az alapadatok és a katalógusszalag alapján.

3. ELLEN szubrutin:

A hálózat működésére vonatkozó vizsgálatok előtt ellenőrzi a hálózat felépítését. Megkeresi az összes bekötési rendellenességet (hiányzó bekötés, nem értelmezhető bekötés stb.), az összes áramköri szempontból tilos összeköttetést (melyek esetén az egyes elemekben károsodás állhat elő) és az összes logikai szempontból tilos összeköttetést (pl. egy kapu bemenete és saját kimenete nem köthető össze) a hiba típusának és helyének megjelölésével.

4. TOK szubrutin:

Meghatározza a hálózat megépítéséhez szükséges integrált áramköri tokok számát típusonként és összesítve. Hasonló számítást végez a szabadon maradt (be nem kötött) tokon belüli elemekre nézve is.

5. TERH szubrutin:

Meghatározza minden egyes elem kimenetének tényleges terhelését, ezt összehasonlítja az elemre megengedett terhelhetőséggel. A számítás TTL egységterhelésben történik. A túlterhelt elemekről külön listát is ad. A szabad kollektoros kimenetekhez kiszámítja (beépített adatok alapján) az alkalmazandó ellenállás értékének alsó és felső határát. Végül megadja a hálózat bemeneteinek terhelését és a kimenetek terhelhetőségét.

6. KESES szubrutin:

A hálózat bármely pontpárja között, a két pontot a kombinációs részhálózatban összekötő valamennyi útra nézve kiszámítja a maximális késleltetési időt és kiírja az utat alkotó elemeket. A kezdőpontokhoz kezdeti késleltetés rendelhető.

7. EXAMKO szubrutin:

Azokat az elemeket keresi meg, melyek kimenetén a vezérléstől függetlenül állandó logikai szint van. Megadja az egyes elemek azon bemeneteit is, melyek az állandó kimeneti szintet okozzák.

8. LOGEGY szubrutin:

A hálózat kiválasztott pontjához rendelhető logikai egyenletet adja meg. Az egyenlet változói a hálózat bemenetei és a tárolókimenetek, formátuma az ún. lengyel írásmód egy módosított változata. Az egyenletek felírásánál semmilyen egyszerűsítés vagy minimalizálás nincs figyelembe véve, vagyis a jel terjedésének útjához tartozó, a hálózatot felépítése alapján jellemző logikai egyenletet adja meg.

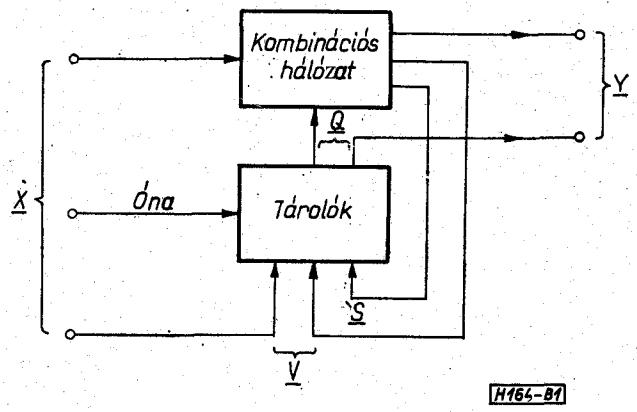
9. LIZA szubrutin:

Meghatározza a hálózat kívánt pontjainak logikai értékét a hálózat bemenő jeleinek és a tárolók belső állapotának előírt értéke mellett. A tárolók tiltott vezérlését üzenet formájában közli. A bemenetek sorozatos vezérlése esetén a kiválasztott pontok logikai értékének kiíratásával a hálózat időrendi működése ellenőrizhető.

2. A programrendszerrel vizsgálható hálózat

A LOGAN programrendszer szinkron működésű szekvenciális hálózatok vizsgálatára alkalmas. A vizsgált hálózat két fő részre bontható: kombinációs hálózatra és a hálózat belső állapotait realizáló tárolókra (1. ábra). A tárolók lehetnek mind szinkron, mind aszinkron vezérlésűek.

A hálózat szinkron működése feltételezi, hogy minden változás azonos időben, az órajelek egy meghatározott szintváltozásának pillanatában következzék be. A szinkron vezérelt tárolók órajelet csak a hálózat bemenetéről kaphatnak, azaz az órajelek kapuzása nincs megengedve, továbbá minden szinkron vezérelt tárolónak az órajel azonos szintváltozására kell átbillennie. Az aszinkron vezérlésű tárolók vagy közvetlenül a hálózat bemenetéről, vagy a kombinációs hálózaton keresztül egymásról vezérelhetők. A hálózat bemenő jelei — így az aszinkron tárolók és a teljes rendszer időzítő vagy vezérlő jelei is — csak az órajel idején változhatnak. Az egymásról történő aszinkron vezérlés hatására bekövetkező



1. ábra

állapotváltozásokat a program szintén az órajellel azonos idejűnek tekinti (a késleltetéseket figyelmen kívül hagyja). A számítás megfelelő sorrendjéről a program maga gondoskodik (lásd 3. fejezet).

A programrendszer megengedi a tárolók vegyes (szinkron és aszinkron) vezérlését is. A kétféle vezérlés egyidejű fellépte esetén az aszinkron vezérlést tekinti meghatározónak. Ily módon előfordulhat, hogy a program egyes tárolókat vezérlésük időbeli lefolyása alapján egyszer szinkron, másszor aszinkron tárolóként kezel.

A kombinációs hálózatrész a fentiek alapján egyrészt vezérli az aszinkron tárolókat, másrészt biztosítja a szinkron tárolók kijelölő vezérlését, ami meghatározza a tárolók következő órajel idején felvett állapotát, továbbá a hálózat bemenő jeleiből a tárolók állapota alapján előállítja a hálózat kimenő jeleit.

A hálózat szinkron működése csak akkor biztosított, ha a kombinációs hálózat elemei között nincs visszacsatolás és az aszinkron vezérelt tárolóknak egymásról — esetleg a kombinációs hálózaton keresztül — való vezérlése szintén egy nyílt lánc mentén történik. A programrendszer a logikai analízis előtt a hálózat elemeit (kapukat és tárolókat egyaránt) ún. kiszámíthatósági sorrendbe rendezi, ami megfelel a logikai jelek „terjedésének". Ha ezt nem tudja elvégezni, mert egy elem logikai állapotának meghatározásához szükséges lenne kimenetének logikai értéke (pl. kapukból kialakított RS tároló), a hálózat aszinkron működésű. Az ilyen aszinkron működést a programrendszer felismeri és kimenő üzenet formájában közli, hogy a hálózat meg nem engedett belső visszacsatolást tartalmaz.

3. A logikai analízis algoritmusa

A programrendszer — mint említettük — a hálózat bármely pontján a logikai érték megváltozását az órajel adott szintváltozásával azonos idejűnek tekinti. Az analízis során tehát nincs időkezelés, a változások ütemét az órajel frekvencia szabja meg, az órajelek közötti időben fellépő esetleges hazárdok és az aszinkron vezérelt tárolók átmeneti állapotainak kimutatására nincs lehetőség. Az analízis feltételezi, hogy a logikai értékek változása az órajel idején „végigfut" az egész hálózaton. Nagy hálózatok esetén az órajelperiódus nagyságrendjébe is eshető kritikus késleltetésekről a késleltetési időket vizsgáló részprogram eredményei alapján lehet képet kapni.

A logikai analízis algoritmusának egyszerűbb bemutatása érdekében vezessük be a következő logikai vektorokat (1. ábra). Jelölje a hálózat bemenő jeleit az

$$\underline{X}=[X_1, \ \dots \ X_i],$$

a hálózat kimenő jeleit az

$$\underline{Y}=[Y_1, \ \dots \ Y_j],$$

a tárolók állapotát a

$$\underline{Q}=[Q_1, \ \dots \ Q_l]$$

vektor, továbbá a szinkron vezérlésű tárolók kijelölését (J, K, D bemenetek) az

$$\underline{S}=[S_1, \ \dots \ S_m]$$

és az aszinkron vezérlő jeleket (CLEAR, PRESET, R, S bemenetek) a

$$\underline{V}=[V_1, \ \dots \ V_n]$$

vektorok. Az egyes vektorok k-adik ütemben felvett értékét felső indexezéssel jelöljük (pl. $\underline{X}^k$).

A tárolók állapotát a k-adik ütemben szinkron vezérlés esetén az

$$\underline{S}^{k-1}=F_S(\underline{X}^{k-1}, \ \underline{Q}^{k-1})$$

kijelölés és az előző ütem során felvett állapot határozza meg a

$$\underline{Q}^k=F_{QS}(\underline{S}^{k-1}, \ \underline{Q}^{k-1}, \ \text{CLOCK})$$

függvény alapján, míg aszinkron vezérlés esetén a

$$\underline{V}^k=F_V(\underline{X}^k, \ \underline{Q}^k)$$

vezérlő jelek a

$$\underline{Q}^k=F_{QV}(\underline{V}^k)$$

függvény alapján. A kimenetek logikai értéke az

$$\underline{Y}^k=F_Y(\underline{X}^k, \ \underline{Q}^k)$$

függvényből határozhatók meg.

A programrendszer az egymás utáni órajelek ($k=1, \ 2, \ \dots \ N$) hatására meghatározza minden előre megadott $\underline{X}^k$ bemenő vektorhoz tartozó $\underline{Q}^k$ állapot- és $\underline{Y}^k$ kimenő-vektort, valamint a szinkron vezérlésű tárolók következő ütembeli állapotát eldöntő $\underline{S}^k$ kijelölő vektort. A számítások elvégzéséhez tárolni kell a $(k-1)$-edik ütemhez tartozó $\underline{S}^{k-1}$ és $\underline{Q}^{k-1}$ vektorokat.

A hálózat logikai analíziséhez meg kell adni a bemenő változók mindazon kombinációját, melyek mellett a logikai működést vizsgálni akarjuk. Az analízis megkezdéséhez szükséges ezenkívül a tárolók kezdeti állapotának ($\underline{Q}^\circ$) és a szinkron tárolók kezdeti kijelölésének ($\underline{S}^\circ$) az ismerete. Utóbbit a bemenő vektor kezdeti értékének ($\underline{X}^\circ$) ismeretében már a program határozza meg. Az aszinkron vezérlésű tárolók kiindulási állapota tetszőleges lehet, mivel azok $\underline{X}^\circ$-val és egymásról is vezérelhetők, így a programrendszer $\underline{X}^\circ$ ismeretében beállítja a bemenő változók kezdő értékének megfelelő helyes kiinduló állapotokat.

A programrendszer tehát a megadott $\underline{X}^\circ$, $\underline{Q}^\circ$ adatokból meghatározza a hálózat kiindulási (vezérlés előtti) állapotát ($\underline{Q}^\circ$, $\underline{Y}^\circ$, $\underline{S}^\circ$). Az első órajel idején érvényes $\underline{X}^1$ bemenő vektor és a hálózat kiindulási állapota alapján kiszámítja az új állapotnak megfelelő $\underline{Q}^1$, $\underline{Y}^1$, $\underline{S}^1$ vektorokat és további órajelek hatására a számítás ciklikusan ismétlődik.

A számítás eredményeit a program a kiválasztott pontok logikai értékének oszlopfolytonos kiírásával közli. A kiíratás formátuma (a két lehetséges logikai

érték egymáshoz képest eltolva jelenik meg) alapján a megadott pontokon lévő logikai jelek idődiagramja is rendelkezésre áll.

4. A programrendszer elemkészlete és korlátozásai

A programrendszer — használatának (adatmegadás) megkönnyítése érdekében — egy katalógusszalaggal rendelkezik. Ezen megtalálhatók a szinkron működésű hálózatokban leggyakrabban használt logikai elemek minden olyan adata, melyre a programrendszer által elvégezhető vizsgálatok során szükség van. Az elemkészlet kialakításánál a hazai viszonyokra való tekintettel célszerűnek látszott a TEXAS TTL integrált áramköri készlet SN74..N sorozatát alapul venni. A katalógus-szalag jelenleg az 1. táblázatban megadott elemek adatait tartalmazza.

1. táblázat

Típus	Logikai funkció
SN7404N, SN7405N	inverterek
SN7400N, SN7401N, SN7403N, SN7410N, SN7420N, SN7430N, SN7440N	2-, 3-, 4-, 8- bemenetű NAND kapuk
SN7402N	2-bemenetű NOR kapuk
SN7450N, SN7451N, SN7453N, SN7454N	2×2-, 4×2- bemenetű AND-OR-INVERT kapuk (a bővíthetők is)
SN7460N	4-bemenetű EXPANDER-ek
SN7472N, SN7473N, SN7476N	J-K-MASTER-SLAVE FLIP-FLOP-ok
SN7474N	D-FLIP-FLOP-ok

Megjegyzés: az aláhúzott típusok szabadkollektoros kimenettel rendelkeznek

2. táblázat

A hálózat bemeneteinek és kimeneteinek együttes száma	max. 100
A hálózatban lévő tárolók száma	max. 100
A hálózatban lévő elemek száma	max. 200
Az elemtípusok száma	max. 20
A hálózat csomópontjainak száma	max. 300
Az elemek bekötött lábainak száma	max. 1000
A logikai analízis során egyszerre kiírható logikai változók száma	max. 30
A kiírható leghosszabb logikai egyenlet karaktereinek száma	max. 4000

A katalógusba felvehető elemek száma tetszőleges lehet. Bővítés esetén, ha az új elem logikai funkciója egy már meglévővel megegyezik, csak az új elem katalógusadataira van szükség. Ha az új elem bevezetése új logikai funkció megjelenésével jár, szükséges az egyes vizsgáló részprogramok kibővítése is.

A programrendszer jelenlegi kiépítésében csak bizonyos korlátozások betartásával használható. A korlátozások egyrészt a felhasznált algoritmusból adódnak (tiltott órajel kapuzás, meg nem engedett aszinkron visszacsatolás), másrészt abból, hogy a számítógép kapacitása alapvetően meghatározza a vizsgálható hálózat méreteit. Utóbbi korlátokat a 2. táblázatban tüntettük fel.

5. Alkalmazási lehetőségek

Jelen fejezetben — a teljesség igénye nélkül, felsorolásszerűen — megkíséreljük összefoglalni azokat a területeket és problémaköröket, ahol a programrendszer a tervezési vagy ellenőrzési munkában hasznos segítséget nyújthat.

Rendszertechnikai szempontból nézve a programrendszer segítségével a digitális áramkörök széles köre vizsgálható. Digitális számítógépek és vezérlő rendszerek területén pl. aritmetikai egységek, vezérlő egységek és rendszerek, szelekciós áramkörök (mágneslemezes, mágnesdobos, ferritmagos tárolók kiválasztó áramkörei), valamint ellenőrző logikai hálózatok működésének szimulációja és ellenőrzése végezhető el. Vizsgálhatók digitális jelátviteli rendszerek hibajelző, hibajavító áramkörei; ipari szekvenciális vezérlő rendszerek; szabályozó szerveket realizáló logikai hálózatok — pl. PD, PI, PID stb. algoritmusok ellenőrizhetők. Használható a program továbbá digitális rendszerek elemi funcionális egységeinek (számláncok, kódkonverterek, üzemmód átalakítók, időzítő áramkörök stb.) vagy egy hálózat tetszés szerint kiválasztott részének a vizsgálatára.

A hálózatok ellenőrzésére — az időbeni sorrendet tekintve — először a tervezés folyamán van szükség. A logikai tervek elkészülésekor, a további konstrukciós tervezés előtt meg kell győződnünk a tervezés helyességéről, az áramkör működőképességéről. Az ellenőrzés folyamán nemcsak a logikai működés, hanem a logikai tervezés során figyelembe nem vehető szempontok alapján az áramköri működés is vizsgálható (terhelési viszonyok, késleltetések, meg nem engedett, illetve hiányzó bekötések). A késleltetési idők ismeretében meghatározható a maximális órajel frekvencia, vagy adott frekvencia esetén ellenőrizhető, hogy a tranziensek lezajlanak-e két órajel között, illetve megfelelő-e az időtartalék. Ugyancsak így ellenőrizhető a tárolók egyszerre történő billenésének feltétele. A kezdeti késleltetések megadásával figyelembe vehetők az elektromos szerelés okozta késleltetések is. Kimutathatók a forrás vagy fogyasztó nélküli csomópontok, az áramkör belső aszinkron jellegű visszacsatolásai.

A hálózat logikai analízise során meghatározható a tárolók vezérlési sorrendje, felvehető az állapottáblázat, kimutathatók a meg nem engedett állapot átmenetek, ellenőrizhető, hogy a hálózat előre felvett állapota fenn áll-e a bemenetek adott értékénél. Kimutathatók a vezérléstől függetlenül állandó logikai értéket képviselő csomópontok és elemek. Bármely belső pont logikai értékének változása vizsgálható a bemeneti vezérlés hatására, előállítható a logikai jelek idődiagramja.

Részekre bontott nagyobb hálózat esetén egyszerűen meghatározhatók a következő részek bemenő jelei, a külső vezérlő jelek könnyen figyelembe vehetők. Megadhatók, hogy az egyes egységek milyen terhelést fejtenek ki egymásra.

A hálózatban végrehajtott minden változtatásnak, egy-egy rész cseréjének vagy új elem bevezetésének hatása minden követelmény alapján vizsgálható (logikai működés, terhelési és késleltetési viszonyok, IC tok igény stb.). Az elemkészletben nem szereplő bonyolultabb integrált áramköri elemek (dekódolók, összeadók, paritás-ellenőrzők) fiktív elemekre bontva helyettesíthetők, így a logikai analízis ilyen hálózatokra is elvégezhető.

Használható a programrendszer már meglévő hálózatok élesztésével kapcsolatos automatizált vizsgálatok előkészítésénél, karbantartási dokumentációk készítésénél, különböző teszt-programok kialakításánál és a hálózatok diagnosztikai vizsgálatánál. Hathatós segítséget nyújthat a diagnosztikához szükséges hibaszótár elkészítéséhez. Mivel a program segítségével a hálózat bármely belső pontja „elérhető", az áramkör jelentéktelen módosításával bármely belső pont logikai értéke tetszés szerint beállítható, így valamely hiba hatása adott bemeneti vezérlés mellett a hiba helyétől a kimenetekig bárhol vizsgálható és kimutatható. Ez lehetőséget ad hardware mérésekkel való összehasonlításra, sőt a belső pontok vizsgáihatósága révén ahhoz képest jelentős többletet is nyújt.

6. A vizsgálandó hálózat megadása

A hálózat megadása a program számára — amennyiben a hálózat elemeit a meglévő elemkészletből vesszük — tulajdonképpen a hálózat struktúrájának leírására és az egyes elemek típusainak közlésére korlátozódik. Az adatmegadás további egyszerűsítését szolgálja a szimbolikus nevek használata.

A felhasználónak névvel (azonosítóval — max. 4 karakter) kell ellátnia a hálózat minden egyes bemenetét, kimenetét és elemét. Szintén szimbolikus névvel (max. 12 karakter) hivatkozhatunk az egyes elemek típusára (pl. SN7400N). Az elemek be- és kimeneteit ún. lábtámasszal látjuk el, mely megegyezik az integrált áramköri tokok katalógusokban közölt lábszámával. A nem elemhez tartozó, logikai funkciót nem teljesítő kivezetések (tápfeszültség, föld, üres lábak) figyelmen kívül hagyandók. A minden szempontból azonosnak tekinthető elemekre azonos lábszámok használhatók.

A hálózat megadásához ezek után fel kell sorolni:

1. a hálózat külső csatlakozási pontjait (csatlakozási lista),
2. típusonként a hálózat elemeit (típuslista),
3. csomópontonként az egymással összekötött elemkivezetéseket és külső csatlakozási pontokat (név, vagy név és lábszám — kötési lista).

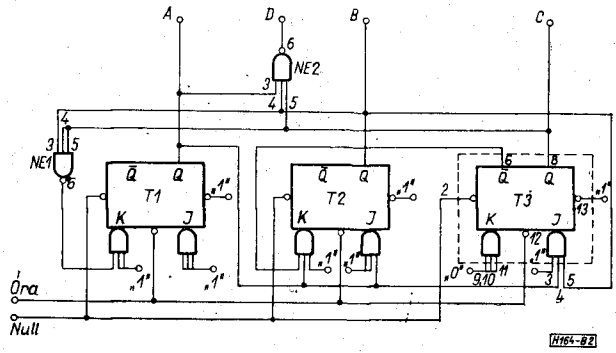
A kötési listában — a nem használt tároló kimeneteken kívül — minden elemkivezetésnek szerepelnie kell. Az elemek nem vezérelt vagy „felesleges" bemeneteit vagy egymással, vagy a tápfeszültség, illetve föld pontokkal kell összekötni. A tápfeszültségre, illetve földre kötött elemlábakat a kötési lista utolsó két csomópontjaként kell megadnia.

Az adatmegadás formátumával, továbbá az egyes áramköri és logikai vizsgálatok végrehajtásához szükséges további adatok megadásával kapcsolatban a LOGAN programrendszer használati utasítására utalunk [2].

A program az eredmények közlésénél a vizsgálat helyére szintén szimbolikus nevekkel és lábszámokkal hivatkozik.

7. Példa

A programrendszer használatára vonatkozóan tekintsük illusztratív példaként a 2. ábrán megadott hálózatot. A hálózat egy 7-ig számláló, majd önmagától leálló szinkron számlánc, újraindítása a NULL bemeneten keresztüli aszinkron törléssel lehetséges.



2. ábra

Az ábrán feltüntettük a szimbolikus neveket és az elemek lábszámozását is (a tárolóknál a lábszámozás azonos). A hálózat megadása az alábbi módon történhet:

Csatlakozási lista:

ÓRA, NULL, A, B, C, D

Típuslista:

SN7472N T1, T2, T3
SN7410N NE1, NE2

Kötési lista:

ÓRA, T1 12, T2 12, T3 12
NULL, T1 2, T2 2, T3 2
A, T1 8, T2 10, T2 5, T4, NE2 3
B, T2 8, T3 5, NE1 3, NE2 4
C, T3 8, NE1 4, NE1 5, NE2 5
D, NE2 6
T3 6, T2 9
T1 9, NE1 6
T3 9, T3 10, T3 11 — földpont (logikai 0)
T1 13, T2 13, T3 13, T1 3, T1 4, T1
T1 10, T1 11, T2 3, T2 4, T2 11,
T3 3 — tápfeszültség (logikai 1)

A logikai analízis megindításához szükséges kiinduló tárolóállapotok megadása tetszőleges lehet (pl. olyan is, mely a normális működés során nem fordulhat elő), ha a bemenetek kiinduló értékének megadásánál törlőjelet adunk a NULL bemenetre. Az

3. táblázat

Sorszám	ÓRA	NULL	T1 8	T2 8	T3 8	D
0	0	0	0	0	0	1
1	1	1	1	0	0	1
2	1	1	0	1	0	1
3	1	1	1	1	0	1
4	1	1	0	0	1	1
5	1	1	1	0	1	1
6	1	1	0	1	1	1
7	1	1	1	1	1	0
8	1	1	1	1	1	0
9	1	1	1	1	1	0
.						
.						
.						
n—í	1	1	1	1	1	0
n	1	0	0	0	0	1
n+1	1	1	1	0	0	1
.						
.						
.						

elsődlegesnek tekintett aszinkron vezérlés hatására ugyanis a program automatikusan beállítja a helyes alapállapotot.

A logikai működést az analízis eredményeként a 3. táblázat alapján követhetjük. A bemenetek vezérlését meg kell adni, a program azonban csak akkor írja ki, ha a kiíratandó pontok listáján a táblázat teljes fejlécét felsoroljuk. A futtatás során az A, B, és C kimenetek helyett a tárolók Q-kimeneteit irattuk ki.

8. Programnyelv, helyfoglalás, futási idő

A LOGAN programrendszer FORTRAN—IV nyelven készült az SzKI SIEMENS 4004/45-ös számítógépre. Helyfoglalása a gép memóriájában 56K szó (32 bites szóhossz). A felhasznált perifériák: kártyaolvasó, sornyomtató és két mágnesszalag egység. Tartozékai: két mágnesszalag, egyrészt a programrendszer tárolására, másrészt a katalógus és a vizsgált hálózatok adatainak tárolására.

A KATAL katalógus kezelő program nyelve szintén FORTRAN—IV, helyfoglalása 21K, ugyanazokat a perifériákat használja, mint a LOGAN programrendszer.

A programrendszer szalagról beolvasása és részeinek összetűzése 4 percet vesz igénybe. A példaként bemutatott számlánc esetében az áramkör felépítésére és logikai működésére vonatkozó vizsgálatokhoz 20 s gépidő szükséges.

Egy bonyolultabb feladat várható időigényének megbecslésére közöljük még, hogy egy bináris, négy bites, párhuzamos üzemmódú összeadó egység, hibaellenőrzéssel (21 csatlakozási pont, 46 elem, 6 elemtípus, 62 csomópont) és egy BCD kódú összeadó egy decimális helyértékét kezelő részének (14 csatlakozási pont, 25 elem, 6 elemtípus, 47 csomópont) egymás utáni teljes vizsgálatához 4 perc gépidőre volt szükség (tiszta futási idő) [2].

A szerzők köszönetüket fejezik ki Farkas György, dr. Flesch István, dr. Géher Károly, Jagudits László, Pápai Zsolt, dr. Szittya Ottó és Theisz Péter kollegáiknak a programrendszer kidolgozása során tőlük kapott hasznos tanácsokért és az értékes eszmecserékért.

IRODALOM

[1] A LOGAN logikai áramkört vizsgáló számítógép programrendszer. Tanulmány az SZKI számára, 1970.

[2] Használati utasítás a LOGAN logikai hálózatok analízisére szolgáló programrendszerhez. 1971.

[3] TTL Integrated Circuits Catalog from Texas Instruments.

[4] *Bohus M.—Géher K.:* Logikai hálózatok számítógépes vizsgálata. Híradástechnika (ebben a számban).