HADMÉRNÖK

IV. Évfolyam 4. szám - 2009. december

Deák Ferenc
deak@nct.hu

LÉTRADIAGRAM FORDÍTÓK ELMÉLETE PLC VEZÉRLÉSEK SZÁMÁRA I.

Absztrakt

A létradiagram egyszerű, programozási képzettséggel nem rendelkező szakemberek számára is érthető nyelv, ennek köszönhető, hogy az elmúlt 20-25 évben az egyik legnépszerűbb ipari programozási formává vált. Bár létradiagramos PLC (Programmable Logic Controller) rengeteg létezik a piacon, az ezeket feldolgozó fordítóprogramok irodalma meglehetősen szegényes. Az alábbi cikk az NCT új vezérléscsaládja számára kidolgozott PLC fordító elméleti hátterét mutatja be. Ez az algoritmus szakít az irodalom által ismertetett megközelítéssel, és gráfelméleti oldalról közelíti meg a kérdést, ami egy jól áttekinthető, szemléletes többlépcsős megoldást eredményez.

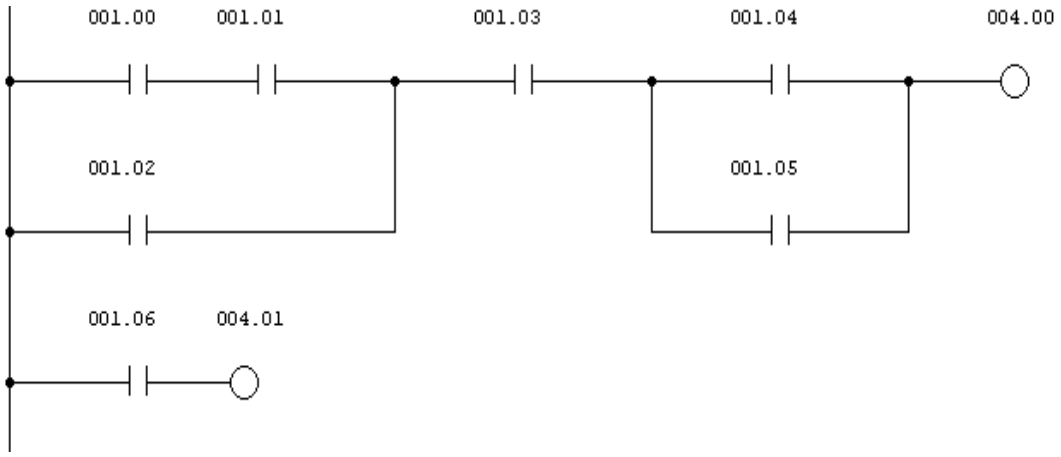
Relay ladder logic has become one of the most popular discrete control programming systems in the last 20-25 years. Programmable Logic Controllers (PLCs) usually can be programmed by wiring up relay contacts and coils on screen. This virtual circuit is transformed into a list of instructions in sequence. In this paper, the translation theory of relay ladder logic for new generation NCT controllers is examined. In contrast to solutions accessible in the literature this algorithm is multiphase, expressive and based on graph theory.

Kulcsszavak: *PLC, Létradiagramm, fordító ~ PLC, Programmable logic controllers, relay ladder logic, compiler*

BEVEZETÉS

Az NCT Kft. – akkor még gazdasági munkaközösségként – 1982-ben alakult. Munkatársai kezdetben az Elektronikus Mérőműszerek Gyára által gyártott Hunor vezérlők fejlesztésével foglalkoztak. A cég a kilencvenes évek elején NCT90 néven saját 16 bites eszterga-, majd pár évvel később maróvezérlővel lépett a piacra, majd szervomotorok és szervohajtások, CNC gépek fejlesztésével és gyártásával bővült a paletta. Az esztergavezérlők NC programnyelve

még a Hunor vezérlőkével egyezett meg, a maróvezérlők programozása azonban már Fanuc-kompatibilis volt.



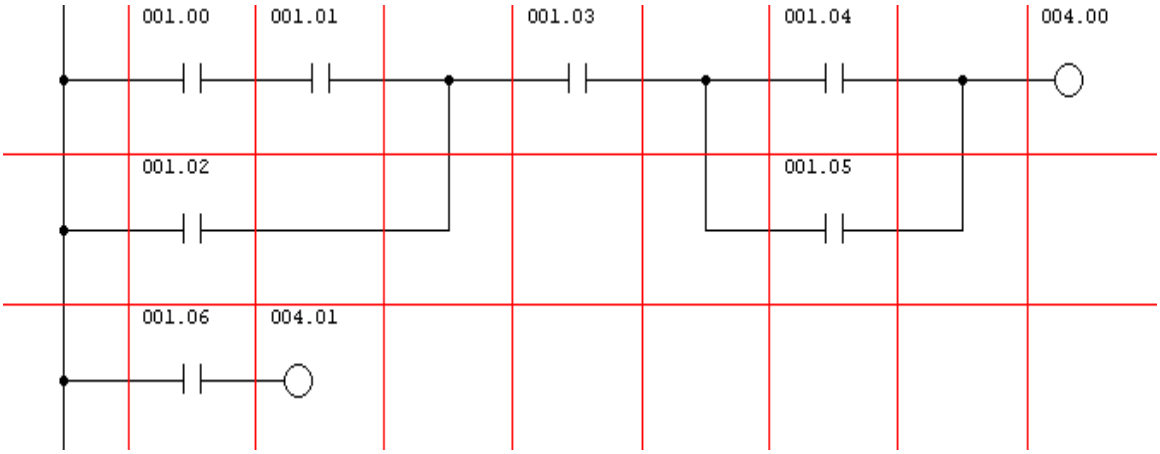
1. ábra. Létradiagram

A következő generációváltás a kilencvenes évek végén történt a 32-bites NCT99 megjelenésével, melynél mind a maró- mind az esztergavezérlők programozása és jelentős részben a kezelése is Fanuc kompatibilissé vált.

```
LD 001.00
AND 001.01
OR 001.02
AND 001.03
LD 001.04
OR 001.05
AND LD
OUT 004.00
```

1. mnemonic kód: Az 1. ábrán lévő létradiagram mnemonic kódja

Tíz évvel később újabb nagyszabású termékmegújítás előtt állunk. A vezérlő többek között preemptív ütemezésű valósidejű operációsrendszert, a korábbinál lényegesen gyorsabb buszrendszert (EtherCAT [10]) kap.



2. ábra. Létradiagram mátrixos felépítése

Az NCT vezérlők a kezdetektől fogva integrált PLC-vel rendelkeztek, vagyis a programozható logikai vezérlő nem önálló hardver egység, hanem csak egy szoftver modul. Egy szerszámgép esetében a PLC feladata, hogy az általános célú eszterga vagy maró vezérlőt egy konkrét gép működési környezetébe illessze: irányítja a szerszám- és munkadarabcserét, pneumatikus munkahengereket működtet, induktív jeladók jeleit fogadja, illetve kommunikál a vezérlő CNC moduljával, ami a szervohajtások irányítását végzi.

Az NCT vezérlők PLC nyelve [3] jelenleg a Hunor vezérlőktől megörökölt alacsonyszintű szöveges kód, amit azonban nagyon kevesen ismernek. Mivel az NCT eladásaiban egyre nagyobb súllyal van jelen az „OEM" (original equipment manufacturer) cégek (szerszámgépgyártók, szerszámgép-felújító vállalkozások) részére történő értékesítés, ezért szükségessé vált, hogy itt is a Fanuc vezérlőkhöz hasonló rendszert vezessünk be: ez a ma már többé-kevésbé szabványossá vált létradiagram.

A szabvány [1] ötféle programozási módot különböztet meg: utasításlista (mnemonickód), strukturált szöveg, létradiagram, funkcióblokk diagram és a sorrendi folyamábra [5]. Az állapotdiagramon [6, 7] és Petri-hálón [8,9] alapuló PLC programozási módok nem terjedtek el széleskörűen.

A létradiagram egyszerű, programozási képzettséggel nem rendelkező szakemberek számára is érthető nyelv, ennek köszönhető, hogy az elmúlt 20-25 évben az egyik legnépszerűbb ipari programozási formává vált.

Bár létradiagramos PLC rengeteg létezik a piacon, az ezeket feldolgozó fordítóprogramok irodalma meglehetősen szegényes. A témában született publikációk többsége azt tárgyalja, hogyan lehet egy megadott szabályrendszerből létradiagramot generálni [4]. Ennek további feldolgozásáról értekezik John T. Welch professzor cikke [2]. Az általa ismertetett algoritmus [2] csupán az alapelemek (kontaktusok, tekercsek) fordítását mutatja be, nem írja le a mai PLC-kben elterjedt bonyolult „soklábú" elemek számlálók, szubrutinok valamint ideiglenes relék kifejtését.

Az alábbi cikk az NCT új vezérléscsaládja számára kidolgozott PLC fordító elméleti hátterét mutatja be. Ez az algoritmus szakít az irodalom által ismertetett megközelítéssel, és gráfelméleti oldalról közelíti meg a kérdést, ami egy jól áttekinthető, szemléletes többlépcsős megoldást eredményez.

A cikksorozat három részből áll: az alábbi cikk ismerteti a létradiagram létrafokokra bontását, a második rész foglalkozik magával a fordítással, a harmadik rész pedig a speciális eseteket és az ipari alkalmazásokat mutatja be.

A LÉTRADIAGRAM

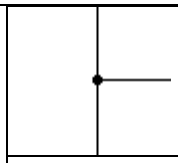
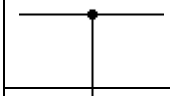
A PLC programot létradiagram formájában kell megírni, ami a hagyományos villamos áramút-tervek szoftveres megfelelője.

Kód	Magyarázat	A verem tartalma, az utasítás végrehajtása után
LD 001.00	A 001.00 bit tartalmát betöltjük a verembe	001.00
AND 001.01	- Kivesszük a verem tetején levő értéket - Elvégezzük a logikai ÉS műveletet a 001.01 tartalmával - Az eredményt betesszük a verembe	001.00 ÉS 001.01
OR 001.02	- Kivesszük a verem tetején levő értéket - Elvégezzük a logikai VAGY műveletet a 001.02 bit tartalmával - Az eredményt betesszük a verembe	001.00 ÉS 001.01 VAGY 001.02
AND 001.03	- Kivesszük a verem tetején levő értéket - Elvégezzük a logikai ÉS műveletet a 001.03 bit tartalmával - Az eredményt betesszük a verembe	(001.00 ÉS 001.01 VAGY 001.02) ÉS 001.03
LD 001.04	A verem tetejére betöltjük a 001.04 bit tartalmát	001.04 ---------- (001.00 ÉS 001.01 VAGY 001.02) ÉS 001.03
OR 001.05	- Kivesszük a verem tetején levő értéket - Hozzávagyoljuk a 001.05 bit tartalmát - Az eredményt betesszük a verembe	001.04 VAGY 001.05 ---------- (001.00 ÉS 001.01 VAGY 001.02) ÉS 001.03
AND LD	- Kivesszük a verem tetején található két értéket - Összeéseljük őket - Az eredményt betesszük a verembe	(001.04 VAGY 001.05) ÉS ((001.00 ÉS 001.01 VAGY 001.02) ÉS 001.03)
OUT 004.00	Kivesszük a verem tetején található értéket, és beírjuk a 004.00 bitbe	üres

1. táblázat. A mnemonickód értelmezése

A PLC program fordítása a következő lépesekből áll:

1. A létradiagramot létrafokokra bontjuk.
2. Az egyes létrafokokat gráffá alakítjuk.
3. Kifejtjük az ideiglenes reléket.

Vezeték darabok		Engedélyezett szomszéd: föntt, jobbra, lent
		Engedélyezett szomszéd: jobbra, lent, balra
		Engedélyezett szomszéd: föntt, jobbra
		Engedélyezett szomszéd: jobbra, balra
Érintkező		Engedélyezett szomszéd: jobbra, balra
Kimenet		Engedélyezett szomszéd: balra

2. táblázat. Néhány létradiagram elem

A piacon léteznek olyan berendezések, amelyeket létradiagramban nem, hanem csak mnemonickódban lehet programozni. Ez esetben általában azt tanácsolják a leírások, hogy a PLC programozó rajzolja meg kézzel a létradiagramot. Ekkor a létradiagram lefordítása mnemonickóddá a PLC programozó feladata. A létradiagram és a mnemonickód között nincsen egyik irányban sem egyértelmű megfeleltetés. Az 1. ábrán szereplő létradiagram fordítása látható az 1. mnemonickód mezőben.

Az **1**. táblázatban foglalt kód némi magyarázatot érdemel.

A mnemonic-kódra fordítást a PLC leírások intuitív folyamatként tárgyalják számos példával illusztrálva. A PLC programozók átlátva a létradiagramban levő kapcsolatokat „kisakkozzák” a megoldást. Ma már a létradiagram fordítók elvégzik ezt a munkát. Az alábbi cikk ennek a folyamatnak egy új módszerét ismerteti.

Ebben a leírásban létradiagram *vezeték darabok*ból, *feltételek*ből (*érintkező*kből), és *kimenet*ekből állnak. A vezeték darabok segítségével a feltételek között logikai (és/vagy) kapcsolatokat hozhatunk létre. A feltételeknek két fajtája van: az alaphelyzetben nyitott és az alaphelyzetben zárt érintkező. Kimenetből nagyon sokféle van, amelyektől jobbra már nem állhat más elem ezeket *jobboldali elem*nek is szokás nevezni. A létradiagram *létrafok*okból áll, melyek a baloldali alapvonalból indulnak ki, és jobboldalon pedig egy *kimenet* zárja őket. Lényegében az egy kimenethez tartozó létradiagram elemeket nevezzük *létrafok*nak.

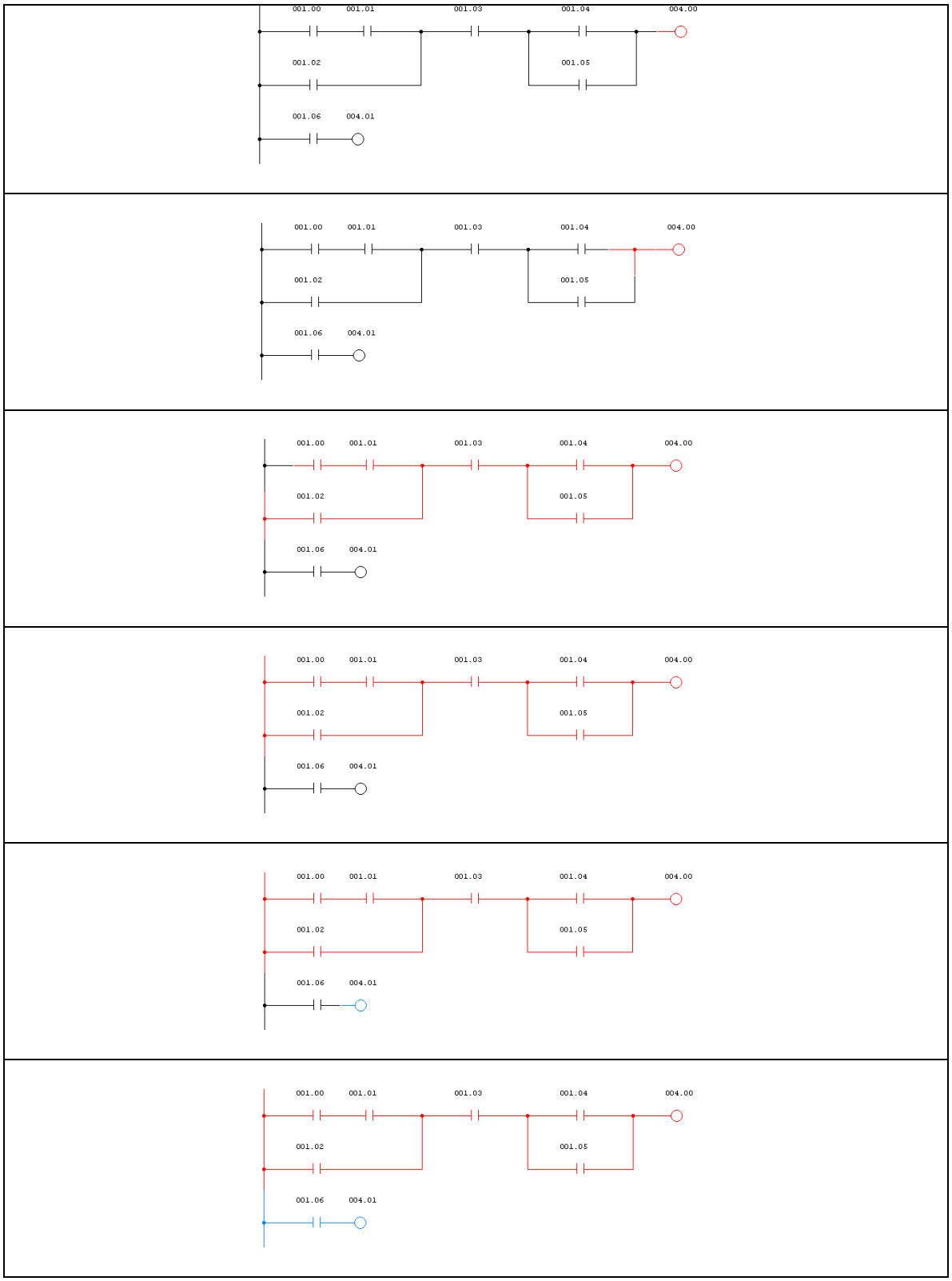
A létradiagram elemek egymáshoz tehát vezetékekkel kapcsolódnak. Ha egy adott létradiagram elem és szomszédja között van vezetékes kapcsolat, akkor azt mondjuk, hogy a létradiagram-elem számára ez a szomszéd engedélyezett. Az 1. ábrán használt létradiagram elemek engedélyezett szomszédait a **2**. táblázat mutatja.

```
minden (K kimenet típusú elemre)
{
        Választunk egy eddig nem használt C színt
        LétraFokokraBontás( K,  C )
}

LétraFokokraBontás(Létradiagram-elem E, Szín C )
{
        ha ( E már a jobboldali alapvonal vagy E már be van színezve) akkor
                nem csinálunk semmit
        különben
        {
                E-t beszínezzük C színűre

                minden ( S engedélyezett szomszédra )
                {
                        LétraFokokraBontás( S, C )
                }
        }
}
```

1. Pszeudó kód: A létrafokokra bontás

Létradiagram mátrixos felépítése

A létradiagram elemeit egy kétdimenziós tömbben tároljuk a 2. ábrán látható módon. Az alábbi leírás nem tér ki minden részletre, a fordítási folyamatot a könnyebb érthetőség kedvéért csak fő vonalaiban ismerteti. A feldolgozás első lépése, hogy a létradiagramot felbontjuk olyan létradiagramok sorozatára, melyek mindegyike csak egyetlen létrafokot tartalmaz.



3. táblázat. A létrafokokra bontás néhány lépése egy példán

A LÉTRAFOKOKRA BONTÁS

A létrafokokra bontás célja, hogy független szegmensekre bontsuk a feladatot, melyeket önállóan lehet feldolgozni.

A kimeneteket a mátrixban balról jobbra és fentről lefele haladva keressük. A fentiek szerint végigmegyünk a mátrix összes elemén, és ha az adott elem kimenetnek számít, akkor választunk egy még nem használt színt és végrehajtjuk rajta a *LétreFokokraBontás* nevű eljárást. Az művelet algoritmusát 1. Pszeudó kód mutatja

Ebben az eljárásban az óramutató járásának megfelelően bejárjuk az adott elem összes szomszédját, és ha az engedélyezett, akkor rekurzív módon elvégezzük azon is a *LétraFokokraBontás* eljárást. A rekurzív hívásoknak akkor van vége, ha elértük a baloldali alapvonalat, vagy ha egy korábban színezett elemre lépünk. Ezt a bejárási módot *mélységi keresés*nek (*depth first search*, *DFS algoritmus*) hívják. Erre mutat egy példát a 3. táblázat

Az azonos színűre színezett létradiagram elemek jelentenek egy lépcsőfokot. A továbbiakban a lépcsőfokokat egymástól elkülönítve dolgozzuk fel.

ÖSSZEFOGLALÁS

A cikksorozat első részében a PLC program létrafokokra bontása került bemutatásra. A következő rész ezeknek az önálló programként értelmezhető programrészeknek végrehajtható szöveges kóddá alakítását fogja majd ismerteti. A harmadik rész tárgyalja, hogy a fordítási algoritmus kidolgozása után a létradiagram szerkesztő kényelmesebbé tétele a fejlesztés fő iránya. Hangsúlyozni kell, hogy a cikkben ismertetett elmélet alapján kialakított szoftver alkalmazásaként már egyre több termelő berendezés működik, főleg a járműiparban.

Felhasznált Irodalom

[1] IEC 61131-3 Programmable controllers - Part 3: Programming languages International Standard, International Electrotechnical Commission, 2003

[2] John T. Welch: Translating unrestricted relay ladder logic into Boolean form. Computers in Industry, 20 (1992) 45-61

[3] NCT szerszámgép vezérlések PLC programozási leírása
http://www.nct.hu/pdf/NC_Documents/Magyar/Telepites/magplc.pdf

[4] H. A. Barker, J. Song, P. Townsend: A rule based procedure for generating programmable logic controller code from graphical input in the form of ladder diagrams, Eng. Appli. of AI, 1989, Vol. 2, December

[5] R. Wareham, Ladder diagram and sequential function chart languages in programmable controllers, Can. Mach. Metalwork., December 1988, pp. 25-26.

[6] R. Devanathan, Computer aided design of relay ladder diagram from functional specification" Int. Conf. on Industrial Electronics, Control and Instrumentation – IECON, 1990, pp. 527-531.

[7] D. Harel, Statecharts: a visual formalism for complex systems, Sci. Comput. Programming Vol. 8, 1987, pp. 231-274.

[8] M. Courvoisier, R. Valette, J. Bigou and P. Esteban, A programmable controller based on a high level specification tool, IECON 1983, pp. 174-179.

[9] T. Murata, N. Komoda, K. Matsumaoto and K. Haruna, A Petri-net based controller for flexible and maintainable sequence control and its applications in factory automation, IEEE Trans. Ind. Electron, Vol. IE-33, No. 1, February 1986, pp. 1-8.

[10] IEC PAS 62407 Real-time Ethernet control automation technology (ETHERCATTM), International Electrotechnical Commission, 2005