КОНЦЕПТУАЛНИ И АЛГОРИТМИЧНИ РЕШЕНИЯ ПРИ СЪЗДАВАНЕ НА ГРАФИЧЕН КОНСТРУКТОР НА КОМБИНАЦИОННИ СХЕМИ

Доц. д-р Христо Кискинов

1. Въведение

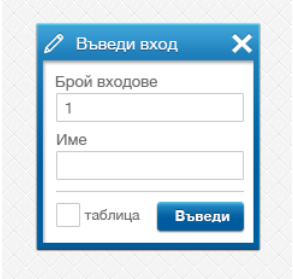
В настоящия доклад са разгледани някои концептуални и алгоритмични решения в създадения конструктор на комбинационни схеми Logical Circuits (LC) за представяне на булеви функции. Конструкторът представлява програмна система, позволяваща графично или таблично задаване, изчертаване и изчисляване на комбинационни схеми. Той е създаден от мен и докторантите Вилислав Радев и Мая Стоева с образователна цел и е предназначен да подпомага изучаването на теорията на булевите функции в лекционни курсове по дискретна математика в университетите както и в профилираната подготовка по информатика в средните училища. Подробно описание на конструктора има в [12].

От една страна, заедно с таблиците и формулите, комбинационните схеми са един от най-популярните методи за представяне на булеви функции. От друга – те са математически модел на реални електронни схеми. По тази причина, типичният инженерен прагматизъм е довел до стереотипи, непривични за математическия мироглед. Например във всички разгледани от авторите на конструктора графични системи за създаване на комбинационни схеми [2, 4, 5, 6, 7, 9, 11], задължително съществува фиксиран набор от „примитивни" функционални елементи, измежду които неизменно присъстват AND, OR и NOT. В LC няма „примитивни" готови функционални елементи. Всички функционални елементи, без изключение, се задават или таблично, или с помощта на комбинационна схема. Вече създадените функционални елементи се използват, (ако няма рестрикции с педагогическа цел) при създаването на комбинационна схема, която от своя страна, ако е коректна, става функционален елемент. Конструкторът LC предоставя автоматизирано изчертаване и интерактивен контрол още в процеса на проектирането на схемата, който не само следи за нейната коректност, но и подпомага създаването и.

2. Описание на функционалните възможности на графичния конструктор LC

Работната област за конструиране на комбинационни схеми представлява правоъгълна решетка. Във всяка клетка на тази решетка може да се разполага само по един функционален елемент. Може да се използва меню с бутони, както и лента, на която фигурират всички налични функционални елементи (Фиг. 3).

При създаване на комбинацинна схема се задава броят на входовете (т.е. броят на променливите на булевата функция) (Фиг.2). На първия ред на решетката се изчертават автоматично функционални елементи, представящи съответните идентитети (Фиг. 3).



Фиг. 1 – *Задаване на брой входове, името на елемента и начин на задаване (таблично или графично)*

Задава се името на схемата, което ще е име и на функционалния елемент, дефиниран с нея. Следва въпрос как ще се дефинира схемата – с таблица (Фиг. 2) или графично (Фиг. 3).

	X1.X2.X3.X4	Function Result
	0.0.0.0	
	1.0.0.0	
	0.1.0.0	

Фиг. 2 – *Таблично задаване на функционален елемент*

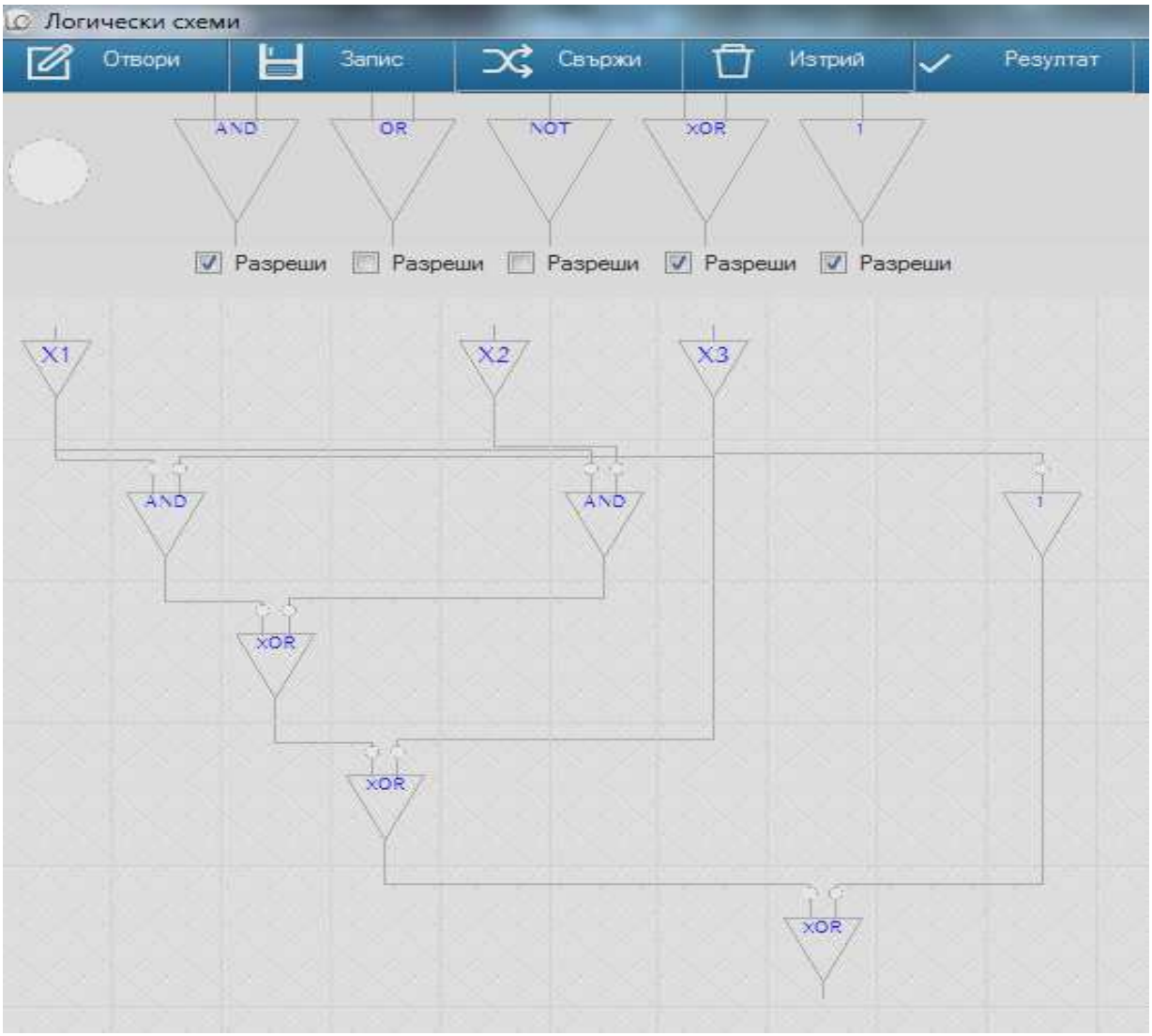
2.1. Таблично задаване на комбинационна схема

Дефинирането с таблица означава, че се създава функционален елемент тип Black box, т.е. такъв елемент, от чиято реализация не се интересуваме, но ще го използваме при конструирането на бъдещи комбинационни схеми. Този елемент автоматично се изчертава на втория ред на решетката под идентитетите като нов функционален елемент със съответния брой входове и име и се свързва с тях: Отваря се прозорец, в който булевата функция, реализирана с тази схема, се задава таблично. След попълването на цялата таблица, схемата се съхранява като нов функционален елемент. Препоръчително, но не задължително е по този начин да се реализира поне една пълна система от булеви функции, което ще гарантира възможността за графична реализация на всякакви комбинационни схеми за в бъдеще. Ако например дефинираме таблично функционални елементи AND, OR, NOT съответстващи на пълната система от булеви функции - конюнкция, дизюнкция и отрицание, ще получим като частен случай добре известната ситуация на налични „примитивни” функционални елементи.

2.2. Графично задаване на комбинационна схема

Графичното дефиниране на комбинационна схема е основната цел на конструктора LC (Фиг. 4).Това става посредством провлачване на елементи от лентата с налични такива до решетката на екрана. Тъй като първият ред е зает с идентитетите, възможното разполагане в решетката е от втория ред надолу. Както е известно, една от възможните грешки при построяването на комбинационна схема е така нареченото „зацикляне” – за вход към даден елемент да се зададе елемент, който зависи при някой от входовете си от изхода на първия. Такова зацикляне при схема, конструирана с LC е невъзможно, поради ограничението, всеки елемент от даден ред на решетката да има изход само към елементи, разположени по-долу от него. Това ограничение не само прави невъзможно наличието на цикъл в схемата, но и подобрява нейния външен вид и много улеснява нейното конструиране. След провлачването на елемент от лентата до решетката, конструкторът го позиционира автоматично в центъра на избраната решетъчна клетка.

Потребителят показва свързването на изхода на даден елемент с вход на друг посредством клик с мишката. Системата проверява и не допуска свързване към вече свързан вход. Изчертаването на тази връзка става автоматично от системата LC и гарантира прегледност на чертежа, т.е. да не се припокриват части от свързващите линии. Това е едно от най-големите достойнаства на настоящия конструктор, защото позволява потребителят да се съсредоточи единствено върху логиката на свързване на елементите, а не върху тяхното разполагане и изчертаване. Конструкторът позволява преместване на вече поставен елемент, като автоматично изчертава наново направените преди това негови връзки. Когато потребителят е готов с конструирането на комбинационната схема, той може да я съхрани. Конструкторът проверява коректността на схемата като изчислява таблицата на реализираната булева функция. Поради факта, че създаването на цикли се предотвратява още на етап разполагане и свързване на елементите, единствената грешка, която може да се получи е липсата на връзка към вход на елемент. В такъв случай, несвързаните входове се маркират и процесът на графично задаване на комбинационната схема продължава.
Ако схемата е коректна, то тя се съхранява и се появява като наличен вече елемент в лентата с функционални елементи.



Фиг. 3 – *Примерна комбинационна схема*

2.3. Преглед на вече създадени комбинационни схеми

По всяко време на работа с конструктора е възможно в нов прозорец да се изиска детайлна информация за всеки от записаните в системата функционални елементи. Тази информация включва комбинационната схема, създадена за този елемент и таблицата на съответстващата булева функция. Възможността за визуализация в отделни позорци на схемите на елементите, участващи в дадена комбинационна схема, позволява създаването и използването на „подсхеми", които да бъдат вграждани в други схеми и т.н. Това дава възможност за избягване на създаването на необозрими схеми, съдържащи много елементи.

2.4. Средства за подпомагане на учебния процес

Основното предназначение на конструктора е за учебни цели и затова в него са предвидени и средства за подпомагане на учебния процес. Конструкторът предлага две нива на достъп до функционалните си възможности – на учител и на ученик. От нивото на учителя е възможно да се определя кои от функционалните елементи в лентата да са достъпни и кои да не са. Такова ограничаване на достъпа до наличните елементи дава възможност за задаването на много и най-разнообразни задачи за решаване от учениците. Например конструирането на нова схема само с помощта на точно определени елементи е една от възможните учебни задачи. Друга допълнителна възможност, предоставена на учителя е автоматичната проверка на конструираната схема от ученика – т.е. сравняването на таблицата на нейната съответстващa булева функция с таблица, зададена от учителя.

3. Софтуерна реализация

Софтуерната реализация на конструктора LC е описана подробно в статията [12] и няма да бъде разглеждана тук.

4. Алгоритмични решения

4.1. Алгоритми за разполагане и изчертаване

Вече споменахме, че работната област, в която се разполагат функционалните елементи, участващи в комбинационната схема представлява правоъгълна решетка. На първия ред (първо ниво) от тази решетка автоматично се разполагат толкова елементи-идентитети, колкото е броя на променливите. На този първи ред не се допуска разполагането на други елементи. Разполагането на различните елементи в решетката и на свързването на техните входове и изходи става от потребителя, като по време на разполагането на елементите се контролира да няма повече от един елемент във всяка колона на решетката. Елементът се провлачва до желаната клетка и се центрира от системета. Освен разполагането на различните елементи е необходимо и да се укажат връзките между тях. Изходът на всеки елемент може да се свързже с един или повече входове на елементи, разположени на по-ниско ниво. След като потребителят укаже такава връзка посредством последователно кликване върху изхода на единия елемент и входа на другия, тя се изчертава автоматично. При изчертаването на връзката, за избягване на припокриване на части от свързващите линии, конструкторът изчертава линията на три етапа: Най-напред, започвайки от изхода на елемента, линията се изчертава вертикално до достигане на реда от решетката на

елемента, собственик на входа. В горната част на всеки ред на решетката се образува "виртуална магистрала", по която да минават хоризонталните свързващи линии. За да се избегне припокриването на различни такива, тази "виртуална магистрала" се разделя на толкова „писти", колкото е сумата от входовете на елементите от този ред. После, по съответната писта се прави хоризонталната част на свързващата линия до достигане на точка, стояща над входа, към който се извършва включване. Най-накрая се изчертава вертикална линия от тази точка до съответния вход.

4.2. Контроли, гарантиращи коректността на комбинационната схема

- При свързване на елементите се извършват следните контроли:
  - Контролира се изходите да се свързват само към входове на елементи от по-ниско ниво.
  - Не се допуска изход на елемент да се свързва с вход, който вече е свързан с изхода на друг елемент.
- Преди съхраняване на схемата се контролира следното:
  - В най-ниското ниво да има един единствен елемент, което гарантира съществуването на един единствен изход на цялата комбинационна схема.
  - Дали всеки изход (освен на най-долния елемент) е свързан с вход на елемент. Това контролира и не допуска наличието на ненужни и неизползвани части от схемата.
  - Дали всеки вход на елемент (освен при идентитетите в най-горния ред) е свързан с изход на елемент (за който разбира се вече е контролирано, че може да бъде само един).

4.3. Алгоритъм за преместване на елемент

Преместването на вече съществуващ в схемата елемент се различава съществено от разполагането на нов такъв поради възможността, неговите входове и изход да са вече свързани с други елементи. По тази причина по време на преместването на елемента се извършват едновременно както контролите за разполагане на елемент, така и контролите за всички негови връзки поотделно. При пропадането на която и да е контрола, преместването се отказва, като се маркира, по каква причина (например заради коя връзка) е отказано преместването. Ако преместването на елемента е допустимо, то освен него се извършва и ново изчертаване на цялата схема, защото се променят параметрите (броя на лентите в тях) на „виртуалните магистрали" по които минават всички хоризонтални свързващи линии.

4.4. Алгоритъм за изчисляване на таблицата на булевата функция

При изчисляването на таблицата на съответната булева функция алгоритъмът е стандартен: На входа на идентитетите се подават последователно всички възможни наредени n–орки от нули и единици като аргументи и последователно, ниво по ниво, се изчисляват резултатите, давани от разположените на това ниво елементи. По този начин, на изхода на елемента от най-долното се появява резултатът, който съответната булева функция дава за подадените аргументи. Резултатите се записват в таблицата на булевата функция.

5. Заключение

Създаденият конструктор на комбинационни схеми съчетава мощ и простота. Това са уникални черти, които го отличават от всички други графични средства за създаване на комбинационни схеми. Графичните системи Logisim [6], TkGate [9], HADES [4], Logic Sim [7], Digital Simulator [2] са предназначени предимно за конструиране на електронни компоненти и поради тяхната сложност не са подходящи за обучение. Въпреки това, в някои отношения конструкторът LC ги превъзхожда. Например, липсата на ограничен набор от фиксирани базови елементи и възможността да се дефинират таблично елементи от тип „black box". LC има предимство спрямо повечето от по-горе споменатите системи и по отношение на автоматичното изчертаване на схемите.
От известните ми графични системи, за обучение са предназначени xLogicCircuits [11] и JLS [5]. Първата е примитивна и отстъпва на LC по всички компоненти. JLS [5] е много мощна система, което обаче я прави доста сложна – факт, който пречи на масовото и използване в обучението по дискретна математика. Конструкторът LC превъзхожда и двете по няколко показателя: Първо – той не само контролира, но и подпомага процеса на конструиране, като улеснява разполагането и изцяло поема свързването на елементите, така че схемата да стане максимално прегледна. И второ – не позволява конструирането на грешна схема. И в [11] и в [5] е възможно например да се конструират схеми със „зацикляне" – нещо което LC не допуска.
Без съмнение LC превъзхожда всички изброени графични системи и с неповторимата си визия, благодарение на специално проектирания за нея дизайн.
Конструкторът LC е предназначен основно за обучение. Неговите възможности бяха тествани и чрез използването му по време на упражненията по теория на булевите функции, базирани на учебниците [1], [3], [8] и [10].
Смятам, че конструкторът LC спокойно може да заеме своето достойно място сред съществуващите графични средства за създаване на комбинационни схеми.

6. Литература

1. Денев Й., Павлов Р., Деметрович Я., Дискретна математика, Наука и изкуство, Sofia 1984
2. Digital Simulator: http://web.mit.edu/ara/www/ds.html, (Last visited 18.10.2013)
3. Grossman, J. W. :*Discrete Mathematics*. Macmillan, New York 1990, ISBN 0-02-348331-8
4. HADES: http://tams-www.informatik.uni-hamburg.de/applets/hades/webdemos/index.html, (Last visited 18.10.2013)
5. JLS: http://www.cs.mtu.edu/~pop/jlsp/bin/JLS.html, (Last visited 18.10.2013)
6. Logisim : http://ozark.hendrix.edu/~burch/logisim/, (Last visited 18.10.2013)
7. LogicSim: http://www.tetzl.de/java_logic_simulator.html, (Last visited 18.10.2013)
8. Манев К., Въведение в дискретната математика, КЛМН София 2005, ISBN 954-535-136-5
9. TkGate: http://www.tkgate.org/index.html, (Last visited 18.10.2013)
10. Wuttke H.-D., Henke K.: *Schaltsysteme – Eine automatenorientierte Einfuehrung.* Pearson Studium, Muenchen 2003, ISBN 3-8273-7035-3
11. xLogicCircuits: http://math.hws.edu/TMCM/java/xLogicCircuits/, (Last visited 18.10.2013)
12. Kiskinov H., Radev V., Stoeva M., A graphic constructor for logic circuits design, *International Journal of Recent Development in Engineering and Technology*, Vol. 2 (2014), No. 4, 24-29, ISSN 2347-6435