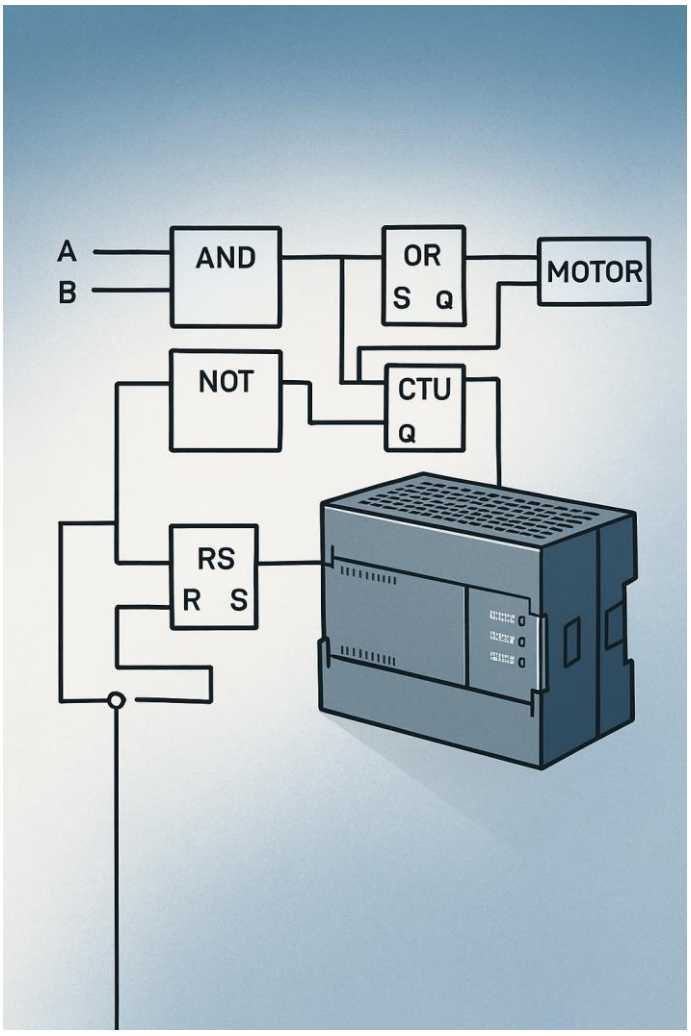
БОРИСЛАВ ГЕОРГИЕВ

РЪКОВОДСТВО
ЗА ЛАБОРАТОРНИ УПРАЖНЕНИЯ
ПО ЕЛЕМЕНТИ НА АВТОМАТИКАТА



УНИВЕРСИТЕТСКО ИЗДАТЕЛСТВО
ГАБРОВО
2025

БОРИСЛАВ ГЕОРГИЕВ

РЪКОВОДСТВО
ЗА ЛАБОРАТОРНИ УПРАЖНЕНИЯ
ПО ЕЛЕМЕНТИ НА АВТОМАТИКАТА

УНИВЕРСИТЕТСКО ИЗДАТЕЛСТВО
"ВАСИЛ АПРИЛОВ"
ГАБРОВО, 2025

РЪКОВОДСТВО ЗА ЛАБОРАТОРНИ УПРАЖНЕНИЯ
ПО ЕЛЕМЕНТИ НА АВТОМАТИКАТА

За студенти от специалност Мехатроника

ISBN: 978-954-683-729-5

Въведение

Настоящото ръководство е предназначено за студентите от специалност **„Мехатроника"** и подпомага усвояването на знания и умения по дисциплината *Елементи на автоматиката*. То е изградено така, че да обхваща целия процес на обучение – от основите на теорията до практическата работа с програмируеми логически контролери (PLC).

В съдържанието са включени три **теоретико-практически упражнения**, които разглеждат структурата на PLC, принципа на тяхната работа и критериите за класификация. Те дават базови знания и развиват умения за анализ и сравнение на различни архитектури, входно-изходни сигнали и методи за програмиране.

Следват **учебно-практически упражнения**, посветени на работа с отделни модули и изграждане на примерни конфигурации. В тях студентите изпълняват задачи, свързани със свързване, наблюдение и анализ на работата на контролера.

В заключителния модул са представени **лабораторни упражнения**, чрез които студентите работят с реално оборудване и прилагат наученото в практически експерименти.

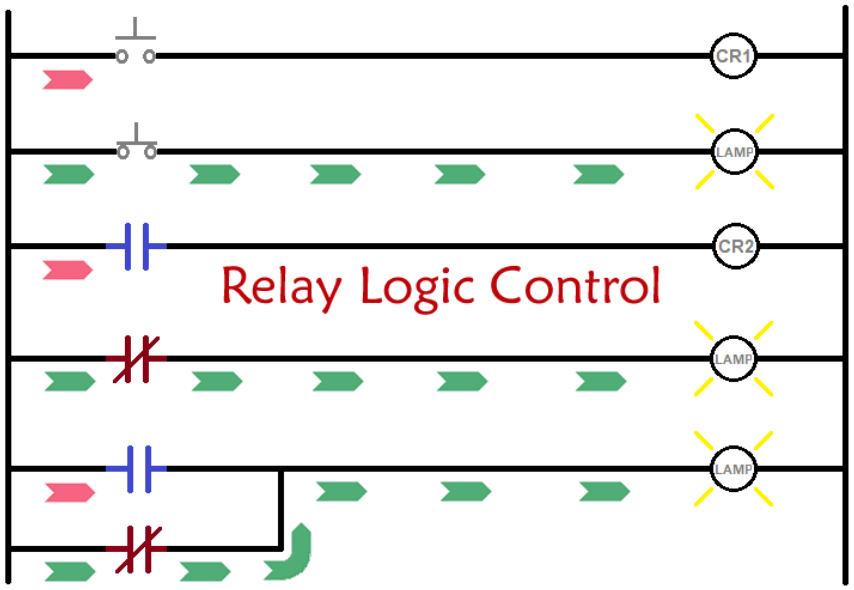
Накрая е включено и едно **модулно упражнение**, което има по-специфичен характер. Неговата цел е да обедини знанията и уменията, придобити от предходните упражнения, като поставя студента в приближени до реалната практика задачи за управление и автоматизация. В него се изисква самостоятелно изграждане на блокови диаграми, логически изрази и решения на по-комплексни задачи, които интегрират различни аспекти на PLC управлението.

По този начин ръководството съчетава теория, упражняване на знанията в учебна среда и прилагането им в реални практически задачи, осигурявайки плавен преход от основите на автоматиката към самостоятелна работа с PLC в индустриални приложения.

Развитие на управляващата техника – от релейната логика до PLC

Управляващата техника се е развивала с течение на времето. В миналото хората са били главните действащи лица за управление на средствата за производство и на производствените процеси. В днешно време електрическата енергия е използвана за управление и първоначално се основава на релейно-контактна логика. Релетата позволяват захранването да бъде включено и изключено без механичен превключвател. Развитието на електрониката, компютърната техника и комуникациите води до качествен скок в управляващата техника. Разработването на нискобюджетен компютър доведе до последната революция- програмируемия логически контролер (PLC). В последните години все по масово навлиза автоматизацията на производствените процеси. Тя позволява по-качествено и по-бързо да се произвежда продукция, като се улеснява работата на човека и се намаляват неговите задължения. PLC-тата се използват в много приложения работещи в реалния свят. Ако има производство, то е почти сигурно че има и контролер, ако няма то тогава се губят пари и време. Почти всяко приложение което се нуждае от някакъв вид електронен контрол има нужда от програмируем контролер. В последните години все по масово навлиза автоматизацията на производствените процеси. Тя позволява по-качествено и по-бързо да се произвежда продукция, като се улеснява работата на човека и се намаляват неговите задължения.

Ранната история на програмируемия логически контролер започва от 60-те години, когато системите за управление са релейни. Релейно ориентираните управляващи машини (релейни логически контролери) (фиг.1.1) са електромеханични устройства, използвани за автоматизация и контрол на различни процеси. Те се базират на релейни технологии, при които електрически сигнали контролират работата на релетата, които от своя страна включват и изключват различни изходни устройства като двигатели, лампи, клапани и др. По онова време в диспечерските зали има цели стени с шкафове, препълнени с релета, клемни блокове и неизброимо количество кабели. Тези системи срещат редица проблеми, по-съществените от които са липса на гъвкавост при необходимост от разширение или изменение на технологичния процес, затруднено отстраняване на повреди свързани със замърсени контакти, разхлабени връзки, несъответствия между остарелите планове на схемите и реалното им положение и т.н. Така назрява моментът за революционно решение на купищата проблеми с които инженери и техници се сблъскват всекидневно.



Фиг.1.1. *Схема на релеен логически контролер.*

През 1968 г. Бил Стоун от отдела за разработка и производство на автоматичните скоростни кутии на Дженерал Мотърс представя на конференция доклад за проблемите с надеждността на производствените системи в техния завод. В доклада той посочва и критериите, на които трябва да отговаря един "стандартен машинен контролер", а именно:

- Да елиминира нуждата от скъпо излизащата подмяна на старата релейна логика при изменения в поточната линия, свързани с внедряването на нови модели, а също така и като цяло да замени ненадеждните електромеханични релета;
- Да намали времето на принудителен престой на машините при възникване на някакъв проблем в системата за управление и да може лесно да бъде програмиран на място с вече възприетата тогава релейна ладер логика (relay ladder logic);
- Да има възможност за разширяване на обхвата и функциите, което значи да бъде с модулна архитектура и да могат да се подменят/добавят компоненти;
- Да може да работи в промишлени условия, т. е. в замърсена среда, при влага, електромагнитни смущения и вибрации.

Спецификацията с горните критерии е предоставена на четири фирми, които да разработят прототип на този "стандартен машинен контролер" - Allen-Bradley, DEC, Century Detroit и Bedford Associates.

DEC предлага "миникомпютър", който бива отхвърлен поради редица причини, но едно от най-сериозните му ограничения е статичната памет.

Allen-Bradley по онова време е сред водещите производители на релета и управления за двигатели и включвайки се в този проект, попада в положението да се конкурира със себе си в една от най-силните си области -

електромеханичните релета. Предлага два прототипа, но и двата, макар и в различна степен, са били твърде обемни, твърде сложни и трудни за програмиране.

Печелившото предложение идва от страна на Bedford Associates и по-точно от колектив в състав Ричард "Дик" Морли, Майк Грийнбърг, Джонас Ландау, Джордж Швенк и Том Бойсевайн. Екипът и дотогава работи по проект за такова устройство с модулна и достатъчно здрава, надеждна архитектура, работещо по зададен алгоритъм, а не само по външни прекъсвания, и използващо пряко изобразяване на адресите на основната памет в кеш-паметта на процесора. Това устройство е наречено 084 - поредният номер на проекта в Bedford. След като си осигурява достатъчна финансова поддръжка, този колектив се отделя и създава фирмата Modicon (от анг. MOdular DIgital CONtroller). Именно под наименованието "Modicon 084" през 1969 г. на Дженерал Мотърс е представен прототипът на програмируемо логическо устройство, изпълнено с полупроводникови елементи. Modicon 084 (фиг.1.2) има три обособени части – процесорна платка, памет и логическо устройство, което изпълнява основния алгоритъм, зададен по правилата на релейната ладер логика (ladder logic).

Фиг.1.2. *MODICON 084*

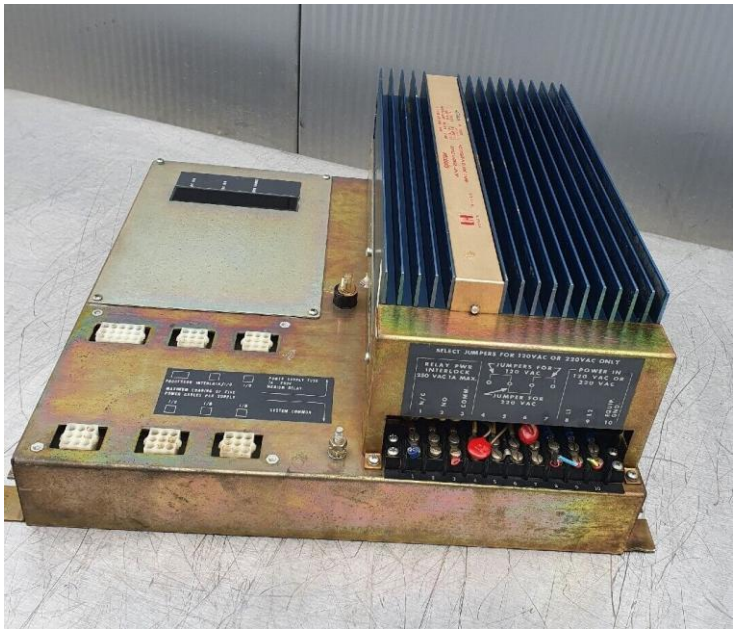
Modicon 084 е проектиран с изключително здрава и надеждна конструкция, без вентилатори за охлаждане, плътно затворен, дори без ключ за включване и изключване. От друга страна, стои изискването той да обслужва някакво устройство, което да не зависи от климата и въобще да не се нуждае от поддръжка. Именно Modicon 084 се смята за първия програмируем логически контролер, макар и създателите му да са го наричали просто "програмируем

контролер" (programmable controller - PC). Modicon набира опит и през 1973 г. пуска следващия много успешен модел PLC, Modicon 184 (фиг.1.3). За разлика от компютрите по онова време, програмируемите логически контролери са проектирани така, че да бъдат надеждни.

Фиг.1.3. *Modicon 184*

Наименованието "програмируем логически контролер" (PLC) се въвежда малко по-късно от Одо Щругер от Allen-Bradley, където работата по това направление продължава, въпреки първите не особено успешни и неприети от Дженерал Мотърс прототипи. Най-общо казано PLC е самостоятелно микропроцесорно устройство с модулна архитектура, което приема входни сигнали, изпълнява програма и формира изходни сигнали за управление на промишлен обект. Отличителни белези са високата надеждност и пригодността за работа в индустриална среда.

През 1971 г. Одо Щругер и Ернст Дамермът създават първия контролер на Allen-Bradley под наименованието Bulletin 1774 PLC (фиг.1.4). Най-точен превод на български би трябвало да бъде "контролер с програмируема логика", но отдавна в България е възприето понятието "програмируем логически контролер". Има държави, които използват собствено название поради ред причини, включително и вследствие на оригинални собствени разработки. В немскоезичното пространство могат да се срещнат със съкращението SPS (нем. Speicherprogrammierbare Steuerung – управление чрез програмируема памет). Във Франция ги наричат "програмируем индустриален автомат" (съкр. API, от фр. Automate programmable industriel). Но във всички случаи същността на устройството е една и съща.



Фиг.1.4. *Bulletin 1774 PLC*

През средата на 70-те, преобладаващите PLC технологии са представлявали машини с последователни състояния. Процесорите AMD 2901 и 2903 са били доста известни в MODICON и A-B програмируемите контролери.

На конвенционалните микропроцесори е липсвала силата да решават бързо PLC логиката, с изключение на най-малките контролери. Все пак с развитието на микропроцесорите, все по големи и по големи PLC-та са се възползвали от възможностите им, но дори днес някои контролери са базирани на AMD 2903 CPU. Комуникационните възможности започват да се развиват приблизително през 1973 година. Първата подобна система е била MODBUS на MODICON. Програмируемия контролер е можел вече да общува с други PLC-та. Те са могли да бъдат и доста далече от машината която е управлявана. Можели са също да предават и да получават различни напрежения, което им е позволило да влязат в аналоговия свят. За нещастие липсата на стандарти свързани с постоянно развиващата се технология е превърнала комуникацията между контролерите в кошмар от несъвместими протоколи и физически мрежи. Въпреки всичко това е било един голям напредък.

През 80-те е направен опит да се стандартизира комуникацията чрез протокола MAP(Manufacturing automation protocol) на General Motor. Също така са били времена през които са се смалили размерите на PLC-тата, както и че те вече са можели да бъдат програмирани чрез програми на персонален компютър вместо чрез програмиращ терминал или специализирани handheld програматори. Най-малкия контролер в света е с размерите на едно контролно реле.

През 90-те почти не се представят нови протоколи. Стандарта IEC 1131-3 се опитва да слее езиците за програмиране под едни общи международни изисквания. Вече е възможно едно PLC да бъде програмирано чрез функционално блокова диаграма, чрез релейно контакторен език, чрез лист от инструкции и структуриран текст. Започва и използването на персонални

компютри за заместители на програмируемите контролери в някои приложения. Компанията която първа е представила MODICON 084 е започнала също да изгражда PC ориентирани контролни системи. Реално погледнато програмируемите контролери са компютри програмирани да изпълняват релейно контакторен език или някой от другите езици за програмиране на PLC-та.

Проблема е, че няма хардуерна съвместимост между различните производители. Всички резервни части трябва да идват от фирмата разпространител на съответния контролер. Потребителя е затворен в рамките, в които производителя е решил да създаде устройството си. Пример за това е че потребителя не е способен да закупи процесор от една фирма и с него да управлява входовете и изходите на устройство на конкурентна на производителя фирма. В повечето случаи потребителя също не може да използва други функции освен вече създадените от производителя модули. Начинът по който се осъществяват много от индустриалните процеси днес е резултат от дългогодишна изследователска и развойна дейност, насочена към подобряване на функционалността, управлението и организацията им.

Сериозен проблем при използването на ранните модели PLC е фактът, че за тяхното програмиране са необходими специализирани терминали. Тези терминали представляват голямо предизвикателство за хората, програмиращи PLC. В това направление Скот Зиферер, съосновател на софтуерната фирма ICOM, и Нийл Тейлър, начело на собствена фирма за промишлен софтуер, започват да развиват програмирането за PLC и съответното документиране, като по този начин оказват огромно влияние върху промишлената автоматизация, каквато я познаваме днес. Скот Зиферер съсредоточава усилията си единствено към продуктите на Allen-Bradley. Той иска да използва компютър за програмирането на PLC и за съставяне на съответната документация, вместо специализирания терминал, който в Allen-Bradley наричаха Терминал Т-3 (фиг.1.5). Постепенно потребителите на Т-3 започват да работят много по-удобно с усъвършенствания потребителски интерфейс, разработен от ICOM. Този подход допринася за по-доброто възприемане на PLC на Allen-Bradley от страна на инженерите по КИП и А и техниците по поддръжката и така разширява полето на приложение на контролерите.

Фиг.1.5. *Allen-Bradley терминал T-3*

Основна роля за унифициране на езиците за програмиране на PLC изиграва споменатият вече Одо Щругер от Allen-Bradley. По-късно той става и вицепрезидент по технологиите в Rockwell Automation. Негова е водещата роля при разработката на стандарта IEC61131-3, на който езиците за програмиране на PLC отговарят днес.

Упражнение 1. *Теоретико-практическо упражнение – Структура на програмируемите логически контролери*

Цел на упражнението:

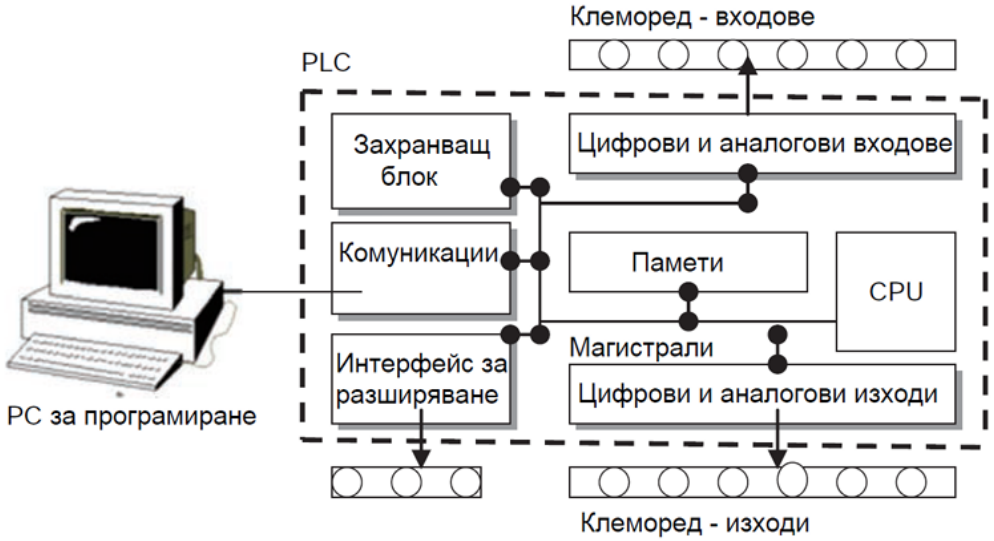
Студентите да се запознаят със състава и функционалното разпределение на блоковете в един PLC – централно процесорно устройство, памет, входни и изходни блокове, захранващ блок.

Задачи за работа:

1. Да бъдат разгледани основните модули и тяхната роля в процеса на управление.
2. Да бъдат анализирани примерни схеми на различни PLC, като се сравнят броят входове, изходи и типовете памет.
3. Да се изследват предимствата и ограниченията на разглежданите архитектури.
4. Да се обобщят факторите, които са решаващи при избора на PLC за дадено приложение.

1.2. Общи положения

Програмируемия логически контролер се състои от **централно процесорно устройство (CPU)**, **памет** и съответните вериги за получаване и изпращане на информация (фиг.1.6).



Фиг.1.6. *Структура на програмируемите логически контролери.*

1.2.1. Централно процесорно устройство (CPU).

Това устройство е основният блок на PLC и определя основните му характеристики като разрядност, бързодействие и др. Тези характеристики са важни при избора на контролер за съответното приложение. Съдържа микропроцесор или микроконтролер с общо предназначение. В първите промишлени контролери са използувани специализирани интегрални схеми като AMD2901, AMD2903 и др. Основният набор инструкции (операционна система за реално време) е програма на високо ниво, инсталирана в постоянна памет (ROM). Програмата за конкретното управление обикновено е съхранявана в постоянна памет EPROM, EEPROM или FLASH. Последните два типа постоянна памет се използват в съвременните устройства и позволяват извършването на следните операции– 1) запис на информация при отпадане на захранването и 2) редактиране или промяна според нуждите на програмата на място. Програмиращото устройство се свързва към процесора, когато операторът трябва да наблюдава, отстранява грешки, редактира или програмира системата, но не е задължително да бъде включено по време на нормалните операции. Информацията се съхранява основно в регистри. Използват се предимно като временни клетки за съхранение, за математически действия или манипулации с информацията. Могат да бъдат използвани също и за съхранение на данни при отпадане на захранването, при възстановяване на напрежението те ще имат същото съдържание.

1.2.2. Памет.

Основно паметта е разделена на постоянна и на оперативна. Постоянната памет се изгражда на базата на ROM, EPROM, EEPROM или FLASH. Съдържанието на цялата или на част от постоянната памет може да бъде променяно. Като оперативни памети най-често се използват памети от статичен тип. Нормално част от оперативната памет (а може и цялата) е с резервирано захранване– батерийно или акумулаторно. Потребителската памет е разделена на блокове със специални функции. Някои части от паметта се използват за съхранение състоянието на входовете и изходите. Всеки вход или изход има един съответстващ бит в паметта. Истинското състояние на даден вход се запаметява като "1" или като "0" в конкретен бит памет. Използват се и други части от паметта за съхраняване на оперативните стойности на променливите, използвани в потребителските програми. Например, текущите стойности на таймерите или на броячите ще се съхраняват в тази част от паметта. Основните подобласти на паметта са:

- **Образ на входния процес - I (Process Image Inputs).** В тази подобласт операционната система записва състоянията на входните точки. Достъпът до информацията от избран вход е например I0.3. Непосредствен достъп до физическия вход се осъществява чрез използване на I0.3:P;
- **Образ на изходния процес - Q (Process Image Outputs).** От тази подобласт операционната система копира изчислените от потребителската програма състояния на изходните точки и ги записва във физическите адреси, например Q1.7. Непосредствен запис във физически изход може да се извърши аналогично на

Q1.7:P;

- **Вътрешни релета** - **M (Bit Memory).** Тук се запазват състоянията на т.нар. „оперативни" релета, имащи отношение към управлението на обекта, но които не са свързани към конкретен изход. Обикновено в тази област се запомнят и междинните резултати при работата на програмата с променливи от тип бит;
- **Таймери - T (Timers).** В тази подобласт се разполагат атрибутите на вградените програмни таймери. Операционната система на контролера поддържа няколко типа таймери, обикновено с фиксирана база време.
- **Броячи - C (Counters).** Подобластта съдържа атрибутите на вградените програмни софтуерни броячи. Нормално се поддържат няколко типа софтуерни броячи;
- **Релета за последователност - SRC (Sequence Control Relay).** Тази подобласт се използва за организиране на постъпково изпълнение на потребителската програма (работа на краен автомат);
- **Памет на променливите - V (Variable Memory).** Предназначена е за запазване на резултатите от междинни изчисления, извършвани върху числови операнди, както и за числови константи;
- **Системна памет / Специални битове - SM (System Memory / Special Bits).** Чрез нея се подържа комуникацията между операционната система на контролера и логическата програма;
- **Аналогови входове (Analog Inputs).** Подобластта е предвидена за адресиране на многовходови аналогово-цифрови преобразуватели, разположени в основния модул или върху допълнителен хардуерен модул. Стойността на аналоговата величина (например температура, ток, напрежение и др.) се съхранява като 16-битова цифрова стойност. Подобластта е позволена само за четене от потребителската програма.
- **Аналогови изходи (Analog Outputs).** Подобластта е предвидена за адресиране на цифрово-аналогови преобразуватели, разположени върху допълнителни хардуерни модули. Модулите преобразуват 16битова цифрова стойност в ток или напрежение. Подобластта е позволена само за запис от потребителската програма. Най-използваните обхвати на аналоговите величини са ±10V, ±5V, ±2,5V, или 0- 20mA;
- **Акумулатори (Accumulators).** Подобластта се използва за предаване на параметри от и към отделните подпрограми, както и за запомняне на междинни резултати от математически изчисления.

1.2.3. Входни блокове.

Интелигентността и възможностите на дадена автоматизирана система зависят до голяма степен от способността й да приема сигнали от различни видове сензори и входни устройства. Бутони, клавиатури и функционални

превключватели са основата на връзката между човека и машината. От друга страна за следене на работата на изпълнителен механизъм, за проверка на налягането или нивото на флуида, има нужда от специални автоматични устройства, като сензори за присъствие, крайни превключватели, фотоелектрически датчици, датчици за нивото и т.н. Входните сигнали на преобразувателите могат да бъдат цифрови (ON/OFF) или аналогови. Обикновено цифровите входове са развързани галванично. По-малките програмируеми логически контролери обикновено имат само цифрови входове, докато по-големите притежават и аналогови входове чрез специални разширителни блокове, присъединени към тях. Едни от най-често използваните аналогови сигнали са токови в обхвата 4-20 mA и напрежителни в обхвата от 0-10 V. Съществуват и специализирани аналогови модули за обработка на сигнали от определен тип датчици- термодвойки, PT100, тензометрични мостове и др.
Физическите места, където постъпват входните сигнали в промишления контролер се наричат входни точки (Input Points). При по-малките PLC входовете обикновено са вградени и техните параметри са посочени в спесификацията на контролера. За по-големите PLC входовете се избират като модули или карти с 8 или 16 входове от един и същи тип на всяка карта.

1.2.4. Изходни блокове.

Системата за индустриално управление не може да изпълнява своите функции без изходни устройства. Някои от най-често използваните изпълнителни устройства са двигатели, соленоиди, релета, индикатори, звукова сигнализация и т.н. Чрез пускането на двигател или реле, PLC може да управлява проста система като система за сортиране на продукти по целия път до сложни системи като сервизна система за позициониране на главата на роботизирана машина. Изходите могат да бъдат от аналогов или цифров тип. Цифровият изходен сигнал има две състояния– чрез него се установяват или прекъсват връзките. Аналогов изход се използва за генериране на аналогов сигнал (например за управление на скоростта на електрически двигател чрез напрежение, чиято стойност е пропорционална на скоростта). Обикновено цифровите изходи са развързани галванично.
Физическите места, през които промишленият контролер изпраща изходните сигнали към управляемия обект се наричат изходни точки (Output Points). Изходните елементи включват външните изпълнителни механизми към източник на напрежение. Изходните елементи са релета, биполярни и MOS транзистори, и триаци. Релетата са най-гъвкавите изходни устройства. Те могат да превключват както променливо, така и постоянно напрежение. Но те са по-бавни (около 10ms типично време за превключване), те са по-обемисти, струват повече и имат определен брой цикли. Изходите с релета често се наричат сухи контакти. Транзисторите се използват за постояннотокови изходи, а триаците – за променливотокови. Изходите с транзистори и триаци се наричат комутируеми изходи.

1.2.5. Захранващ блок.

Захранващият блок осигурява енергия за процесора и другите модули. Най-често използваните вътрешни напрежения са +5V, ±12V и +24V. Допълнително, захранващият блок може да осигури на контролера сигнал за отпадане на входното напрежение (Power Down). Повечето PLC контролери използват като входно напрежение 24 VDC или 220 VAC. При някои PLC контролери захранващият блок е отделен модул. Обикновено това са по-големи PLC контролери, докато малките и средните серии съдържат собствен модул за захранване. Потребителят трябва да определи мощността, която ще се консумира от I/O модула, за да се увери, че електрическото захранване може да я осигури. Различни видове модули имат различна консумация на енергия. Захранващият блок обикновено не се използва за захранване на външните входове и изходи. Потребителят трябва да осигури отделни захранващи източници за входовете и изходите на контролера.

Упражнение 2. *Теоретико-практическо упражнение – Принцип на работа на PLC*

Цел на упражнението:

Студентите да придобият знания за фазите на оперативния цикъл на PLC и тяхното значение за правилното функциониране на системата за управление.

Задачи за работа:

1. Да бъдат разгледани основните фази на оперативния цикъл (Input Scan, Program Scan, Output Scan, Communication, System Functions).
2. Да се анализират ограниченията, свързани с минималната продължителност на входните сигнали и максималното време на реакция.
3. Да се изследват примерни времедиаграми, илюстриращи цикличната работа на PLC.
4. Да се обобщят подходи за осигуряване на надеждно четене и обработка на входните сигнали.

1.3. Общи положения

Програмируемите логически контролери работят под управлението на операционна система за реално време (ОСРВ), която осигурява циклично изпълнение на логическата програма. Операционната система управлява последователното изпълнение на няколко основни действия. Тази последователност, както и самото й изпълнение, се нарича оперативен цикъл (operating cycle) на промишления контролер. Основните действия на промишления контролер са 5 (фиг.1.7). Те се изпълняват последователно и се наричат още „фази на оперативния цикъл”. Последователното изпълнение на отделните фази води до някои ограничения, които трябва да се имат предвид при избора на подходящ контролер за дадено приложение. Двете основни ограничения които съществуват са минималната продължителност на входните сигнали и максималното време на реакция. Минимална продължителност на входните сигнали е параметър, който определя избора на контролер и на подходящи алгоритмични и апаратни подходи за правилното четене на входовете. Обстоятелствата, които налагат взимането на подобни решения са:

- Асинхронно активиране на входните точки спрямо оперативния цикъл;
- Логическата програма няма достъп до физическите адреси на входните точки, а работи с областта от паметта Process Image Inputs;
- Операционната система обновява тази област само във фазата Input Scan.

Фиг.1.7. *Основни действия на PLC.*

Поредните номера и означенията на фазите са както следва:

- **1 - IN - сканиране на входовете (input scan);**
- **2 - PRG - сканиране на програмата (program scan);**
- **3 - OUT - сканиране на изходите (output scan);**
- **4 - COM - обслужване на комуникацията (service communications);**
- **5 - H&O - системни функции (housekeeping and overhead).**

1.3.1. IN - сканиране на входове (Input Scan).

Фаза 1 се нарича сканиране на входове (Input Scan). Физическите места, където постъпват входните сигнали в промишления контролер се наричат входни точки (Input Points). За всяка входна точка е определен по един бит в оперативната памет– входен бит (Input Bit). По време на тази фаза се прочитат входните точки и информацията от тях се записва във входните битове. Създава се „образ" на входните въздействия към дадения момент в областта на входните битове на оперативната памет (Process Image Inputs).

1.3.2. PRG - сканиране на програмата (program scan).

Фаза 2 е сканиране на програмата (Program Scan). По време на нея се изпълняват всички инструкции от логическата програма по отработване на входните въздействия и формиране на изходните реакции. Програмируемите логически контролери са последователностни устройства, в които при формирането на изходните реакции освен входните променливи участват и вътрешни за контролера променливи (таймери, броячи, компаратори и т.н.). В много ограничени случаи, програмируем логически контролер може да изпълнява само комбинационни логически функции (изходните реакции се определят еднозначно само от входните въздействия). Продължителността на тази фаза зависи от броя и вида на използваните инструкции.

1.3.3. OUT - сканиране на изходите (output scan).

Фаза 3 се нарича сканиране на изходите (Output Scan). Физическите места, през които промишленият контролер изпраща изходните сигнали към управлявания обект се наричат изходни точки (Output Points). Както при входните точки, така и за всяка изходна точка е определен по един бит в оперативната памет– изходен бит (Output Bit). В тази област от оперативната памет се създава „образ" на изходните реакции в дадения момент (Process Image Outputs). По време на тази фаза се извършва прочитане на изходните битове и прехвърлянето на информацията от тях в изходните точки (физическите адреси на изходите).

1.3.4. COM - обслужване на комуникацията (service communications).

Фаза 4 е за обслужване на комуникацията (Service Communication). По време на нея се извършва комуникация с други устройства, например програмиращо устройство, централен компютър, устройство за интерфейс човек-машина (операторски пулт, панел, терминал) и др. В повечето програмируеми логически контролери, комуникацията се обслужва главно от операционната система на контролера, според специални директиви, записани в управляващата програма. Някои контролери позволяват директно управление на комуникацията от потребителската програма чрез съответната системна функция.

1.3.5. H&O - системни функции (housekeeping and overhead).

Фаза 5 е за изпълнение на системни функции (Housekeeping and Overhead). По време на нея се извършват разнообразни действия, поважните от които са за управление на:– паметта;– вътрешните регистри на микропроцесора;– вградените специални апаратно-програмни функции (Firmware Function) като броячни входове, импулсни входове и изходи, допълнителни комуникационни портове и др.; - вградените логически функции, които имитират работата на хардуерни устройства, използвани за целите на логическото управление като устройства за времезадържане (таймери), устройства за отброяване на външни или вътрешни за контролера събития (броячи) и др.;– тестване на апаратно-програмните функции на промишления контролер (Self Diagnostic Test).

Упражнение 3. *Теоретико-практическо упражнение – Класифициране на PLC контролерите*

Цел на упражнението:

Студентите да придобият знания за основните критерии за класификация на PLC контролерите и да се запознаят с техните разновидности според мащаба, архитектурата, вида на входно-изходните сигнали и начина на програмиране.

Задачи за работа:

1. Да бъдат разгледани типовете PLC според мащаба (нано, микро, мини, средни и големи).
2. Да се анализират архитектурите на PLC (фиксирана, модулна, разпределена, софтуерно базирана).
3. Да се изследват различните видове входно-изходни сигнали (дискретни, аналогови, хибридни и специализирани).
4. Да се обобщят основните начини за програмиране на PLC, като се посочат предимства и недостатъци.
5. Да се подготви кратко сравнение между различните групи PLC и тяхната приложимост в индустриални системи.

1.4. Общи положения

Съвременните PLC (Програмируемите Логически Контролери) могат да бъдат класифицирани по различни критерии, като:

- **Мащаб (размери);**
- **Архитектура;**
- **Вид на входно-изходните сигнали;**
- **Начин на програмиране;**
- **Приложение;**
- **Начин на комуникация**.

Разнообразието от PLC разновидности се дължи на различните изисквания на индустрията и приложенията, където се използват, за да покрият нуждите им. Няма универсален PLC, който да е идеален за всички случаи.

1.4.1. Мащаб (размер).

Според мащаба (размера) си PLC се разделят на:

- **Нано;**
- **Микро;**
- **Мини;**
- **Средни;**

- **Големи.**

1.4.1.1. Нано PLC.

Нано PLC (Nano Programmable Logic Controllers) са ултракомпактни устройства с ограничен брой входове и изходи (10-50), подходящи за малки автоматизации, които предлагат висока производителност и функционалност в малък размер. Те са идеални за приложения, където пространството е ограничено и се изисква висока скорост и точност. Основните характеристики на нано PLC са:

- **Компактни размери:** Нано PLC са изключително малки, което ги прави подходящи за инсталации в ограничени пространства;
- **Висока производителност:** Въпреки малкия си размер, нано PLC предлагат висока производителност и могат да управляват сложни процеси;
- **Множество входно/изходни точки:** Нано PLC обикновено разполагат с множество цифрови и аналогови входове и изходи, което ги прави гъвкави за различни приложения;
- **Комуникационни възможности:** Те поддържат различни комуникационни протоколи като Ethernet, Modbus TCP/IP и RS485, което позволява лесна интеграция с други устройства и системи;
- **Интегрирани функции:** Нано PLC често включват функции като часовник за реално време, високоскоростни броячи, PWM изходи и други.

Пример за нано PLC е Nano-10 PLC (фиг.1.8), който предлага висока производителност и множество функции в малък размер.

Фиг.1.8. *Nano-10 PLC.*

Друг пример е em4 nano-PLC (фиг.1.9), който е лесен за използване и предлага разширяемост до 46 входно/изходни точки.

Фиг.1.9. *Em4 nano-PLC*

Нано PLC са най-малките програмируеми логически контролери, предназначени за прости автоматизирани задачи с малък брой входове и изходи. Те са компактни, лесни за интеграция и осигуряват надеждност в приложения като управление на осветление, HVAC системи, малки машини и битови автоматизации. Основните им предимства са ниската цена, енергийна ефективност и лесно програмиране. Въпреки това, те имат ограничени възможности за разширение и комуникация, което ги прави неподходящи за сложни индустриални системи.

1.4.1.2. Микро PLC.

Микро PLC (Micro Programmable Logic Controllers) са компактни устройства малко по-мощни от нано PLC, с разширяеми входове/изходи (50-500), които предлагат висока производителност и могат да управляват множество процеси едновременно. Те са подходящи за малки и средни приложения, където е необходимо надеждно и гъвкаво управление. Основните характеристики на микро PLC са:

- **Компактни размери:** Микро PLC са сравнително малки, което ги прави лесни за инсталация в ограничени пространства;
- **Разширяемост:** Те обикновено предлагат опции за разширяване с допълнителни модули за входове и изходи, което ги прави гъвкави за различни приложения;
- **Висока производителност:** Микро PLC могат да изпълняват сложни задачи и да управляват множество процеси едновременно;
- **Лесно програмиране:** Те са съвместими с различни програмни езици, включително Ladder Logic, FBD (Function Block Diagram) и други;
- **Комуникационни възможности:** Микро PLC поддържат различни комуникационни протоколи като Ethernet, Modbus TCP/IP и RS485, което позволява лесна интеграция с други устройства и системи.

Пример за микро PLC е Siemens S7-1200 (фиг.1.10), който предлага висока производителност и множество функции в компактен размер.

Фиг.1.10. *Siemens S7-1200*

Друг пример е Mitsubishi FX3S (фиг.1.11), който е модулен мини PLC, често използван в машиностроенето предоставя гъвкавост и разширяемост за малки и средни приложения.

Фиг.1.11. *Allen-Bradley MicroLogix 1400*

Микро PLC предлагат повече входове и изходи, разширяемост и по-добри комуникационни възможности. Те са широко използвани в малки и средни

автоматизационни проекти като управление на машини, транспортни системи и индустриални процеси. Основното им предимство е балансът между цена, функционалност и мащабируемост, което ги прави гъвкаво решение за различни приложения. Въпреки че не могат да се конкурират с големите модулни PLC по изчислителна мощност, те са ефективни за повечето стандартни автоматизационни задачи.

1.4.1.3. Мини PLC.

Мини PLC (Mini Programmable Logic Controllers) са малки и компактни устройства, които са предназначени за прости контролни приложения и предлагат основните функции за логическо управление, мониторинг и комуникация. Обикновено са лесни за инсталация и програмиране, което ги прави идеални за малки индустриални и домашни автоматизирани системи. Основните характеристики на мини PLC са:

- **Малки размери:** Мини PLC са компактни и могат да бъдат инсталирани в ограничени пространства;
- **Ограничен брой входове и изходи (I/O)**: Обикновено между 10 и 50, но някои модели поддържат разширения;
- **Ниски разходи:** Те са по-евтини в сравнение с по-големите и по-мощни PLC, което ги прави икономически изгодни за малки проекти;
- **Основни функции**: Мини PLC предлагат основни функции за управление на процесите, като входно/изходни операции, таймери и броячи;
- **Лесно програмиране**: Те могат да бъдат програмирани с помощта на различни програмни езици, включително Ladder Logic, FBD (Function Block Diagram) и други;
- **Вградени комуникационни възможности**: Някои мини PLC имат вградени комуникационни интерфейси като Ethernet, RS232 или RS485 за свързване с други устройства и системи.

Пример за мини PLC е Siemens LOGO!, който е лесен за инсталация и програмиране и предлага множество входно/изходни опции (фиг.1.12).

Фиг.1.12. *Siemens LOGO!.*

Друг пример е Mitsubishi FX3S (фиг.1.13), който е модулен мини PLC, често използван в машиностроенето и предоставя гъвкавост и разширяемост в малък размер.

Фиг.1.13. *Mitsubishi FX3S*

Мини PLC са отличен избор за малки и средни автоматизационни системи, където не е необходимо сложно управление, но се изисква надеждност, гъвкавост и компактност. Те запълват нишата между релейните схеми и големите PLC, осигурявайки ефективно решение за множество приложения.

1.4.1.4. Средни PLC.

Средните PLC (Medium Programmable Logic Controllers) са устройства, които предлагат баланс между производителност и размер. Те са по-мощни от нано, мини и микро PLC, но по-малки и по-лесни за инсталация и поддръжка в сравнение с големите и комплексни PLC. Предназначени са за управление на по-сложни автоматизационни системи с по-голям брой входове и изходи (500-5000 I/O). И предлагат модулна архитектура, която позволява добавяне на

допълнителни модули за разширение, комуникация и специализирани функции. Използват се в индустриални производства, транспортни системи, енергийни инсталации и други процеси, където е необходимо гъвкаво и надеждно управление.

Основните характеристики на средните PLC са:

- **Умерени размери**: Средните PLC са по-големи от микро и мини PLC, но все пак компактни в сравнение с големите системи. Това ги прави удобни за инсталация в средни и големи приложения;
- **Разширяемост**: Те обикновено предлагат опции за разширяване с допълнителни модули за входове и изходи, както и за комуникация и обработка на данни;
- **Висока производителност**: Средните PLC предлагат висока производителност и могат да управляват сложни процеси и множество входно/изходни сигнали;
- **Лесно програмиране**: Те са съвместими с различни програмни езици, включително Ladder Logic, FBD (Function Block Diagram), ST (Structured Text) и други, което ги прави лесни за програмиране и адаптиране към различни нужди;
- **Комуникационни възможности**: Средните PLC поддържат различни комуникационни протоколи като Ethernet, Modbus TCP/IP, Profibus, и други, което позволява интеграция с други устройства и системи;
- **Надеждност и стабилност**: Те предлагат висока надеждност и стабилност, което ги прави подходящи за критични приложения в индустриалната автоматизация.

Пример за среден PLC е Siemens S7-1500 (фиг.1.14), който предлага висока производителност и множество функции за различни индустриални приложения.

Фиг.1.14. *Siemens S7-1500*

Друг пример е Allen-Bradley CompactLogix (фиг.1.15), който предоставя гъвкавост и разширяемост за средни и големи системи.

Фиг.1.15. *Allen-Bradley CompactLogix.*

Средните PLC са идеалният компромис между функционалност и цена, като предлагат мащабируемост и надеждност за автоматизация на средно големи системи. Те намират широко приложение в заводи, логистични центрове и производствени линии, където се изисква повече гъвкавост в управлението.

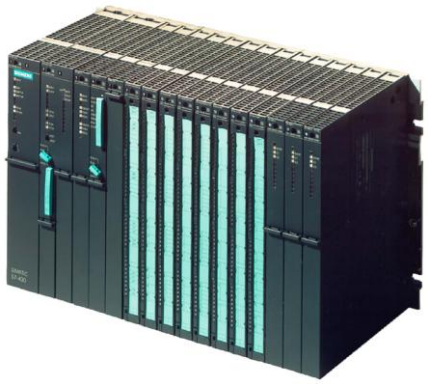
1.4.1.5. Големи PLC.

Големите PLC (Programmable Logic Controllers) са мощни и мащабируеми устройства, които са предназначени за големи индустриални системи и приложения с висока сложност, управление на цели производствени линии, електроцентрали, химически заводи и дори интелигентни градове.. Те предлагат много функции и възможности за управление на широк спектър от процеси и системи. Поддържат разпределени входове и изходи (5000+ I/O), висикоскоростна обработка на данни и многозадачност. Основните характеристики на големите PLC са:

- **Мащабируемост**: Големите PLC могат да бъдат разширени с множество входно/изходни модули и комуникационни модули, което ги прави изключително гъвкави и подходящи за различни приложения;
- **Висока производителност**: Те разполагат с мощни процесори, които могат да обработват голям обем от данни и да изпълняват сложни алгоритми за управление в реално време;
- **Надеждност**: Големите PLC са проектирани за работа в тежки индустриални условия и предлагат висока надеждност и стабилност;
- **Множество функции**: Те включват различни функции като управление на движение, обработка на аналогови сигнали, високоскоростни броячи и много други;
- **Комуникационни възможности**: Големите PLC поддържат разнообразие от комуникационни протоколи като Ethernet/IP, ProfiNet, Modbus и други, което позволява лесна интеграция с други устройства и системи;
- **Разширени възможности за диагностика и мониторинг**: Те предлагат мощни инструменти за диагностика и мониторинг на системите, което улеснява поддръжката и управлението на процесите.

  Пример за голям PLC е Siemens S7-400 (фиг.1.16), който е предназначен

за мащабни индустриални системи и предлага висока производителност и множество функции. В автомобилните заводи на BMW и Volkswagen, SIMATIC S7-400 управлява цялата производствена линия, като синхронизира роботи, транспортьори, CNC машини и контролира качеството.



Фиг.1.16. *Siemens S7-400*

Друг пример е Allen-Bradley ControlLogix 5580 (фиг.1.17), който предоставя мащабируемост, висока надеждност и комуникация за големи индустриални приложения. Той е част от Rockwell Automation и се използва широко в автомобилната индустрия, енергетиката, фармацевтиката, хранително-вкусовата промишленост и тежкото машиностроене.

Фиг.1.17. *Allen-Bradley ControlLogix 5580*

Големите PLC са предназначени за мащабни индустриални процеси, където надеждността, мащабируемостта и бързината на работа са от съществено значение. Те осигуряват максимална гъвкавост, но изискват значителни инвестиции и специализирана поддръжка.

1.4.2. Архитектура.

При избор на контролер за решаване на конкретна задача на автоматизираното управление се взимат предвид някои основни характеристики:

- **Брой и тип на входните и изходните точки;**
- **Бързодействие– определя се от минималната продължителност на входните сигнали;**
- **Обем и организация на паметта на работната програма.**

В зависимост от възможностите на контролерите, тяхната архитектура се разделят на четири основни групи:

- **Фиксирана архитектура (Integrated PLC);**
- **Модулна архитектура (Modular PLC);**
- **Разпределена архитектура (Distributed PLC);**
- **Софтуерно базирана архитектура (Soft PLC).**

1.4.2.1. Фиксирана архитектура (Integrated PLC).

Тези PLC известни още като компактни, имат предварително определен брой входове и изходи, които не могат да бъдат разширени. Всички блокове на контролера са разположени в едно физическо устройство. То съдържа захранване, процесорна система, определени количества входни и изходни точки, интерфейс за програмиране и блок за режим на работа и индикация. Те са подходящи за по-малки и по-прости приложения, където не се изисква голяма гъвкавост – при тях броят на входно-изходните точки е ограничен (обикновено 8 – 24) и няма възможности за разширяване.
Пример за PLC с фиксирана архитектура е Schneider Electric Zelio Logic (фиг.1.18).

Фиг.1.18. *Schneider Electric Zelio Logic*.

1.4.2.2. Модулна архитектура (Modular PLC).

Модулната архитектура на контролерите позволява по-голяма гъвкавост и мащабируемост в различни системи и приложения. Тази архитектура обикновено се състои от отделни модули, които могат да бъдат разработвани, тествани и внедрени независимо един от друг. При тях потребителят решава колко и какви модули да се включат в конфигурацията на контролера, така че да се изпълни задачата за автоматизация на управляемия обект. При някои конфигурации модулите се разполагат в касета (rack), на дъното на която е разположена системна магистрала за връзка между тях. По този принцип се изграждат големите комплексни системи за управление. Производителите на

промишлени контролери предлагат широка гама от допълнителни входно-изходни модули, съобразени с най-често срещаните сензори и изпълнителни механизми. Разширителните модули биват сигнални (SM – Signal Modules - аналогови и цифрови входове и изходи), интерфейсни (IM – Interface Modules - служат за осигуряване на връзка между различни касети с модули), функционални (FM – Function Modules – специализирани модули за броене, позициониране, управление с обратна връзка, регулиране и др.), комуникационни процесори (CP – Communication Processors - за изграждане на локални мрежи и разпределени системи за управление).
Пример за PLC с модулна архитектура са контролерите на **Siemens S7-300/S7-1500** (фиг.1.19).

Фиг.1.19. *Siemens S7-300/S7-1500.*

Модулните PLC са най-гъвкавите и мощни контролери, използвани в индустриалната автоматизация. Те предлагат разширяемост, комуникационни възможности и висока производителност, което ги прави идеални за средни и големи системи.

1.4.2.3. Разпределена архитектура (Distributed PLC).

PLC (Programmable Logic Controllers) с разпределена архитектура предлагат редица предимства пред традиционните централизирани системи. В разпределената архитектура, различни компоненти на PLC системата се разполагат физически на различни места и комуникират помежду си чрез мрежи. Разпределените PLC предлагат модерно и ефективно решение за сложни индустриални системи, където надеждността, мащабируемостта и бързата комуникация са от решаващо значение. Те са особено подходящи за големи предприятия, разпръснати производствени линии и критични инфраструктури, където отказът на едно устройство не трябва да води до спиране на целия процес. Основните характеристики и предимства на разпределената архитектура на PLC са:

- **Разпределена обработка**: Изчислителните задачи са

разпределени между няколко модула, което увеличава ефективността и намалява натоварването на централния контролер.

- **Повишена надеждност**: При отказ на един модул, останалите продължават да функционират, което повишава общата надеждност на системата.
- **Гъвкавост и разширяемост**: Лесно могат да се добавят нови модули и функционалности, без да се засяга цялата система.
- **Подобрена комуникация**: Разпределените PLC системи използват различни мрежови протоколи за комуникация, като Ethernet, PROFIBUS, Modbus и други.
- **Намалени кабелни връзки**: Системата изисква по-малко кабелни връзки между компонентите, което намалява разходите и сложността на инсталацията.
- **Локална автоматизация**: Модулите могат да извършват локална автоматизация, което намалява времето за реакция и увеличава производителността.

Пример за PLC с разпределена архитектура е Schneider Electric Modicon M580 (фиг.1.20). Той е иновативен Ethernet програмируем автоматизационен контролер (ePAC), който предлага висока производителност и гъвкавост. Предлага вградена Ethernet свързаност, която оптимизира комуникацията и свързаността между различните модули и устройства, като има високо ниво на киберсигурност, което гарантира защитата на данните и системата. Контролерът поддържа различни мрежови протоколи като Ethernet, PROFIBUS, Modbus и други, което улеснява интеграцията с различни системи.

Фиг.1.20. *Schneider Electric Modicon M580*

1.4.2.4. Софтуерно базирана архитектура (Soft PLC).

PLC със софтуерно базирана архитектура, известни като Soft PLC, са ново поколение контролери, които използват софтуерни технологии за управление и автоматизация на процеси. Тези контролери предлагат редица предимства и

характеристики, които ги правят подходящи за различни приложения. При тях основната логика за управление не се изпълнява от специализиран хардуерен PLC, а от обикновен индустриален компютър (IPC) или вграден контролер, работещ със специализиран софтуер. Това позволява по-голяма гъвкавост, мащабируемост и интеграция с IoT и облачни технологии, като същевременно запазва стандартните функции на традиционните PLC. Soft PLC са иновативно решение за индустриалната автоматизация, което съчетава гъвкавостта на софтуерните технологии с мощта на класическите PLC. Те са идеални за IoT приложения, облачни платформи и интелигентни индустриални системи, но изискват висока сигурност и надежден хардуер, за да бъдат устойчиви в индустриална среда.

1.4.3. Вид на входно-изходните сигнали.

Класификацията на PLC според вида на входно-изходните сигнали зависи от приложението и изискванията на системата. Те се разделят на няколко основни групи, които са:

- P**LC с дигитални (дискретни) сигнали**: При този вид PLC входните и изходните сигнали са дигитални. Те работят с бинарни сигнали (0 или 1), които представляват състояние "включено/изключено", като входните сигнали управляват PLC а чрез изходните контролера управлява устройства като релета, светодиоди и малки електродвигатели.
- **PLC с аналогови сигнали**: При този вид PLC входните и изходните сигнали са аналогови, като се използват аналогово-цифрови преобразуватели (ADC) и цифрово-аналогови преобразуватели (DAC). Те позволяват прецизно управление на процеси, където е необходима постепенна промяна на параметрите. Входовете приемат непрекъснати сигнали, които варират в определен диапазон от стойности (например 0-10V или 4-20mA), като тези входни сигнали се използват за измерване на параметри като температура, налягане ниво на течности и др., а аналоговите изходи генерират непрекъснати сигнали за управление на устройства като задвижващи механизми, честотни преобразуватели, пропорционални клапани и др.
- **Смесени (Хибридни) PLC:** Смесените (хибридни) PLC обработват както аналогови, така и цифрови сигнали. Това ги прави изключително гъвкави и способни да взаимодействат с различни видове входно-изходни устройства. Използват се в комплексни индустриални процеси, които изискват както дискретно управление на машини, така и аналогова регулация на параметри (например автоматизирани производствени линии, HVAC системи, химически процеси).
- **Специализирани PLC (за специфични сигнали):** Тези PLC имат специализирани входно-изходни модули за комуникация с други устройства и системи чрез мрежови протоколи като Ethernet,

PROFIBUS, Modbus и др. Могат да приемат сигнали от термодвойки и RTD сензори за измерване на температура както и висикоскоростни импулсни сигнали с които обработват енкодери, стъпкови мотори, честотни импулси.

Дискретните PLC са прости и надеждни, аналоговите позволяват прецизно управление, а хибридните комбинират най-доброто от двата вида. Тези различни типове входно-изходни сигнали позволяват на PLC системите да се адаптират към широк спектър от приложения и индустрии.

1.4.4. Начин на програмиране.

PLC (програмируемите логически контролери) могат да се класифицират според различни критерии, като един от основните е начинът на програмиране. Въз основа на този критерий PLC могат да се разделят на няколко основни групи **на класически, визуално програмирани и софтуерни (Soft PLC).** Изборът зависи от нуждите на индустриалната система, сложността на управлението и необходимата интеграция с външни системи.

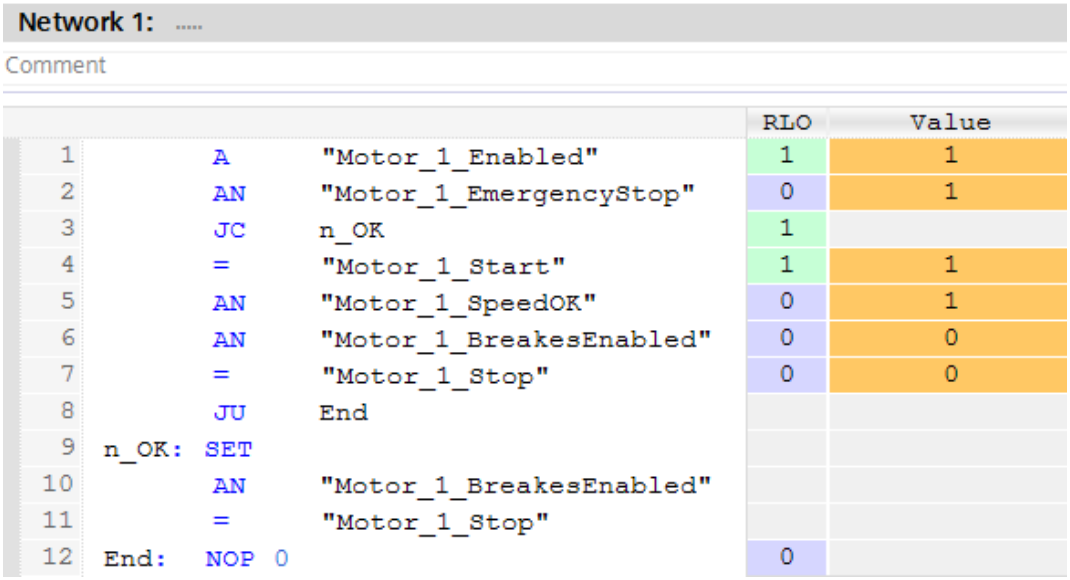
а) Класически PLC (програмируеми чрез традиционни езици): Тези PLC използват стандартни програмни езици, определени от IEC 61131-3, които са специално създадени за индустриалната автоматизация. Типичен примери за такива езици са **Structured Text (ST) и Instruction List (IL).**

- **Structured Text (ST):** Структурираният текст, съкратено като ST или STX (фиг.1.21.), е един от петте езика, поддържани от стандарта IEC 61131-3, предназначен за програмируеми логически контролери. Това е език от високо ниво, който е с блокова структура и синтактично наподобява Pascal, на който се основава. Тъй като структурираният текст е подобен на традиционните езици за програмиране на високо ниво, той може да бъде доста лесен за много хора, които може да нямат опит в програмирането на PLC, но имат опит в традиционното кодиране.

```
1  preset_temp:=12;               //Immediate Expression
2  pushbutton:=1;                 //Immediate Expression
3  new_preset_temp:=preset_temp; //Tag Expression
4  timer_value:=ton1.ACC;      //Tag Expression
5  timer_done:=ton1.DN;       //Tag Expression
6  //timer operation
7  TONR_01.PRE := 500;
8  TONR_01.Reset := Reset;
9  TONR_01.TimerEnable := input3;
10 input3:=1;
11 if TONR_01.ACC >= 500 & TONR_01.DN then //A conditional statement
12     input3:=0;
13 end_if;
14 TONR(TONR_01); //Function Expression
15 timer_state := TONR_01.DN;
16 total:=add1+12+13+add2;//Operators Expression
```

Фиг.1.21. *Structured Text (ST).*

- **Instruction List (IL):** Списъкът с инструкции (IL) (фиг.1.22) е език за програмиране от ниско ниво, наподобяващ assembler, който е специално разработен за програмиране на програмируеми логически контролери (PLC) в технологията за автоматизация. Този цикличен език за програмиране позволява контрол и мониторинг на машини и процеси. В списъка с инструкции програмните команди се показват под формата на текстови инструкции. Тези инструкции са проектирани да бъдат разбрани и изпълнени директно от контролера. Програмирането в IL се извършва по начин, който отчита цикличната последователност на PLC. За постигане на желания контролен ефект се използват различни логически и аритметични операции. Основите на списъка с инструкции включват логически оператори като AND, OR и NOT, но също така и операции за сравнение и математически функции. Тези елементи са комбинирани в структурирана последователност за решаване на сложни задачи за контрол. Четенето и разбираемостта на програмите на IL са важни аспекти за осигуряване на ефективна поддръжка и отстраняване на проблеми. Приложенията на IL са разнообразни и варират от контрола на прости производствени машини до сложни производствени линии в индустрията. Ясната структура и тясната връзка с хардуера правят IL ефективен език за програмиране в технологията за автоматизация. Непрекъснатото развитие на PLC системите означава, че съвременните програми за програмиране на IL включват също графични изображения и интегрирани среди за разработка. Това улеснява инженерите и програмистите да създават, поддържат и оптимизират IL програми. Като цяло списъкът с инструкции (IL) е основен компонент в програмирането на програмируеми логически контролери и играе ключова роля в автоматизацията на индустриалните процеси.

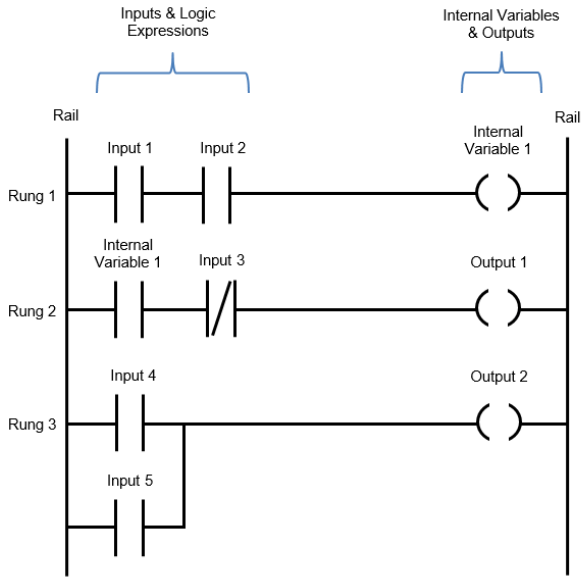


Фиг.1.22. *Instruction List (IL).*

б) **Визуално програмираните PLC (програмируеми логически контролери)** са устройства, които се програмират чрез графичен интерфейс, вместо чрез текстови езици за програмиране като Structured Text (ST) или Instruction List (IL). При тях се използват блокови диаграми, схеми или графични символи вместо текстови команди. Те са по-удобни за инженери и техници без опит в класическото програмиране. Типичен примери за такива езици са **Ladder Diagram (LD), Function Block Diagram (FBD) и Sequential Function Chart (SFC)**

- **Ladder Diagram (LD)**: Представлява графичен език за програмиране, като отначало са използвани т.нар. контактни команди, които симулират отваряне и затваряне на релета. В последствие това програмиране е обогатено с редица функционалности като броячи, таймери, shift регистри и математически операции. Контактите представляват битове, които биват кодирани в ladder диаграмата. Техните инструкции изобразяват състоянието на различните контакти. Например, когато имаме нормално отворен контакт, той се включва когато стойността на адресирания бит е единица, а когато имаме нормално затворен контакт стойността е нула. Ladder диаграмата е пряк наследник на релейно-контактните диаграми. Използва символи като реле, контакт, функционален блок и се оформя във вид на мрежа. Визуално Ladder diagram наподобява електрическа схема. Логическите диаграми (Ladder diagram) (фиг.1.23) са усъвършенствани схеми, широко използвани за записване на логически структури в индустриалното управление. Те се наричат "стълбови диаграми" (Ladder diagrams), защото наподобяват стълба с две вертикални релси (захранващата линия) и толкова

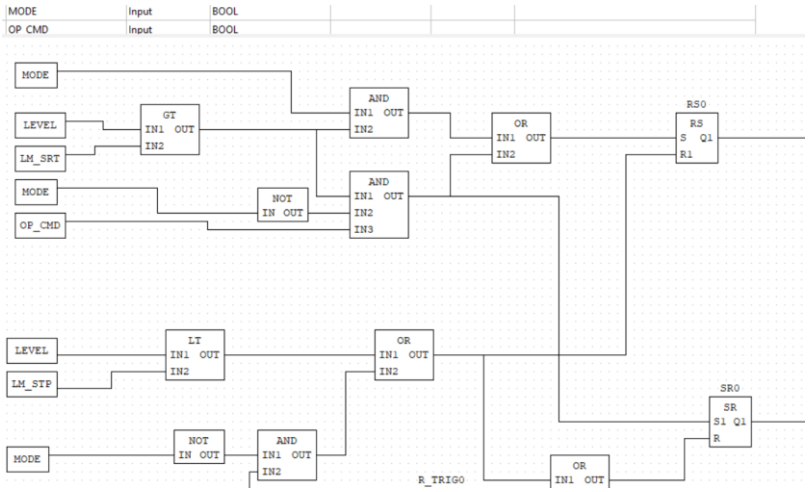
„стъпала" (хоризонтални линии), колкото са необходими за представяне на управляващите вериги. Консуматорът (например лампа, релеен намотка, соленоид) почти винаги се изчертава в дясната част на стъпалото в стълбовите диаграми. Ladder diagram е лесен за разбиране и подходящ за инженери и техници с опит в релейните системи, но може да стане труден за управление при сложни програми.



Фиг.1.23. *Ladder Diagram.*

- **Функционални блокове (Function Block Diagram):** Функционалните блокови схеми са графичен език за показване на потоци от сигнали и данни посредством функционални блокове. Негово предимство е лесното визуализиране на връзката между алгоритмите и логиката на системата за управление. Този метод използва графични блокове, които представляват функционални единици. Блоковете са свързани помежду си, за да формират логическа последователност. Това е език за програмиране на PLC (фиг.1.24), подобен на цифровите логически вериги, който е сравнително нов метод на програмиране, използващ формата на кутии, за да представлява оперативни функции. Има набор от библиотеки с предварително готови програмирани блокове, но потребителят може да създава и програмира свои собствени блокове. Той е лесен за разбиране от хората с основни познания за цифрови вериги, защото функционалните модули могат ясно да изразят функционалната връзка, така че времето за програмиране и отстраняване на грешки да бъде значително намалено. Понастоящем този език е тенденция за развитие. Международната електротехническа

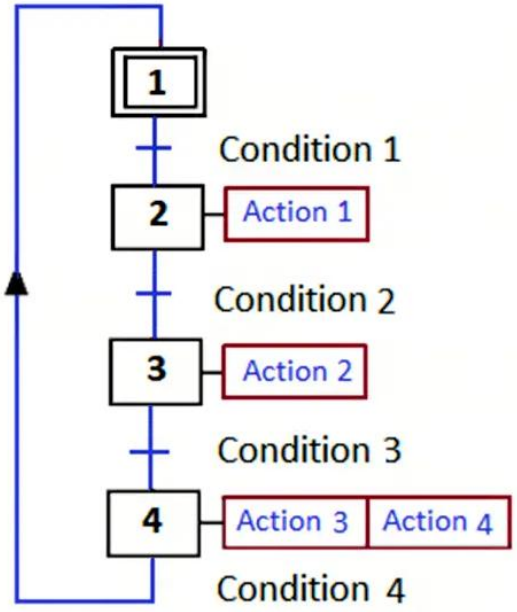
асоциация изпълнява и разработва този нов програмен стандарт, а някои производители на PLC започнаха да поддържат този език в средни и големи PLC.



Фиг.1.24. *Function Block Diagram.*

- **Sequential Function Chart (SFC)**: Метод за програмиране на сложни системи за управление на по-високо структурно ниво. Неговата програмата се състои в преглед на система за управление, в която основните градивни елементи са цели програмни файлове. Всеки програмен файл се създава с помощта на един от другите видове езици за програмиране. Подходът на SFC координира големи, сложни задачи за програмиране в по-малки, по-управляеми задачи. Този метод за програмиране разделя процеса на последователни стъпки. Всяка стъпка съдържа набор от действия и условия за преминаване към следващата стъпка. SFC е поредица от скриптове (фиг.1.25), които са дефинирани в едно място и след това се извикват в последователен ред. Допълнителни елементи в диаграмата могат да определят докъде ще води потокът на диаграмата. Графиките могат да се повтарят за неопределено време или да изпълнят зададен брой пъти преди да приключат. Секвенциалните функционални диаграми са базирани на графичен език за програмиране в стандарта IEC 61131-Този език може да е познат на PLC програмистите, тъй като е един от езиците, които обикновено се предлагат за програмиране на PLC. SFC се използват за изпълнение на логиката по начини, които са по-удобни за структуриране, отколкото само с Python скриптове или PLC програмиране. Поради присъщото им визуално изобразяване, те помагат да се осветява логиката на потребителите и да се улесни интуитивното развитие и усъвършенстване. Диаграмите могат да се наблюдават, докато те се изпълняват визуално, което

прави отстраняването на неизправности по-лесно, отколкото само при скриптове.



Фиг.1.25. *Sequential Function Chart (SFC)*

в) Soft PLC са програмни решения, които заменят класическите физически PLC контролери. Те работят на стандартни индустриални компютри (IPC), сървъри или дори в облака, като изпълняват PLC логика. Характеризират се с гъвкавост тъй като могат да работят на различни хардуерни платформи (PC, Raspberry Pi, индустриални контролери), разширени възможности поради лесната им интеграция с IoT, SCADA и бази данни, както и ниската им цена предвид това че се елиминира нуждата от специализиран PLC хардуер.

1.4.5. Приложение.

Според приложението си PLC могат да се класифицират в две основни категории, PLC за общи приложения (универсални) и PLC за специализирани приложения:

- **PLC за общи приложения:** Тези PLC се използват за широк спектър от автоматизационни задачи. Поддържат дискретни и аналогови сигнали и различни комуникационни протоколи. Гъвкави и лесно адаптируеми към различни индустриални процеси.
- **PLC за специализирани приложения.** Тези PLC са проектирани за конкретни индустриални задачи. Често разполагат с специализирани функции за повишена безопасност, управление на движения и процеси.

1.4.6. Начин на комуникация.

Според начина, по който комуникират с други устройства в системата PLC могат да се класифицират като **самостоятелни, централизирани, разпределени мрежови и софтуерно базирани (Soft PLC):**

- **Самостоятелни (Standalone PLC):** Работят без необходимост от комуникация с други PLC или отдалечени устройства. Всички входно-изходни (I/O) модули са локално свързани към контролера. Подходящи за малки и средни автоматизирани системи.
- **Централизирани (Centralized PLC):** Всички входно-изходни устройства и управлявани процеси са свързани директно към PLC контролера. Входно-изходните сигнали се обработват локално. Подходящи за средни и големи автоматизирани системи, които изискват висока скорост на обработка.
- **Разпределени (Distributed PLC):** Работят в мрежа, където един централен PLC комуникира с няколко отдалечени I/O модули. Използват полеви шини като PROFIBUS, PROFINET, Modbus, EtherCAT. Намаляват дължината на кабелите и повишават гъвкавостта на системата.
- **Мрежови (Networked PLC):** Свързват се един с друг и с други устройства чрез индустриални мрежи (Ethernet/IP, PROFINET, Modbus TCP). Позволяват дистанционно управление и мониторинг. Често са част от SCADA, MES и IoT платформи.
- **PLC със софтуерно базирана комуникация (Soft PLC):** Не са физически устройства, а софтуерни PLC, работещи на компютър, индустриален сървър или в облака. Комуникират с входно-изходни модули и други устройства чрез Ethernet, OPC UA, MQTT. Предоставят висока гъвкавост и могат да се мащабират според нуждите на системата.

Упражнение 4. *Учебно-практическо упражнение – Работа с модулите на PLC*

Цел на упражнението:

Студентите да се запознаят с основните модули на програмируемия логически контролер и тяхното предназначение, както и да придобият практически умения за свързване и конфигуриране на входно-изходни модули.

Задачи за работа:

1. Да бъдат идентифицирани основните модули на даден PLC (CPU, памет, захранващ блок, входни и изходни модули).
2. Да се изследва принципът на свързване на входни/изходни модули към процесорното устройство.
3. Да се опише ролята на захранващия блок и начините за осигуряване на надеждна работа на системата.
4. Да се изготви схема на примерна конфигурация на PLC с два цифрови входа (бутон и краен изключвател) и един цифров изход (лампа).
5. Практическа част:
   - Свържете реални входни устройства (бутон и краен изключвател) към цифровите входове на контролера (по указание на преподавателя).
   - Свържете лампа или светодиод към цифров изход (по указание на преподавателя).
   - Наблюдавайте как промяната на състоянието на входните устройства влияе върху изхода.
   - Запишете наблюдаваните резултати в протокол от упражнението.
   - Да се сравнят наблюдаваните резултати с очакваните (по указание на преподавателя).
6. Да се направят изводи за предимствата на модулния подход и възможностите за разширяване на системата.

2. Общи положения

Siemens LOGO PLC е компактна система за управление, предназначена за малки проекти за автоматизация. Може да се програмира с помощта на LOGO Soft Comfort. Този софтуер предлага графичен интерфейс за създаване и редактиране на програми.

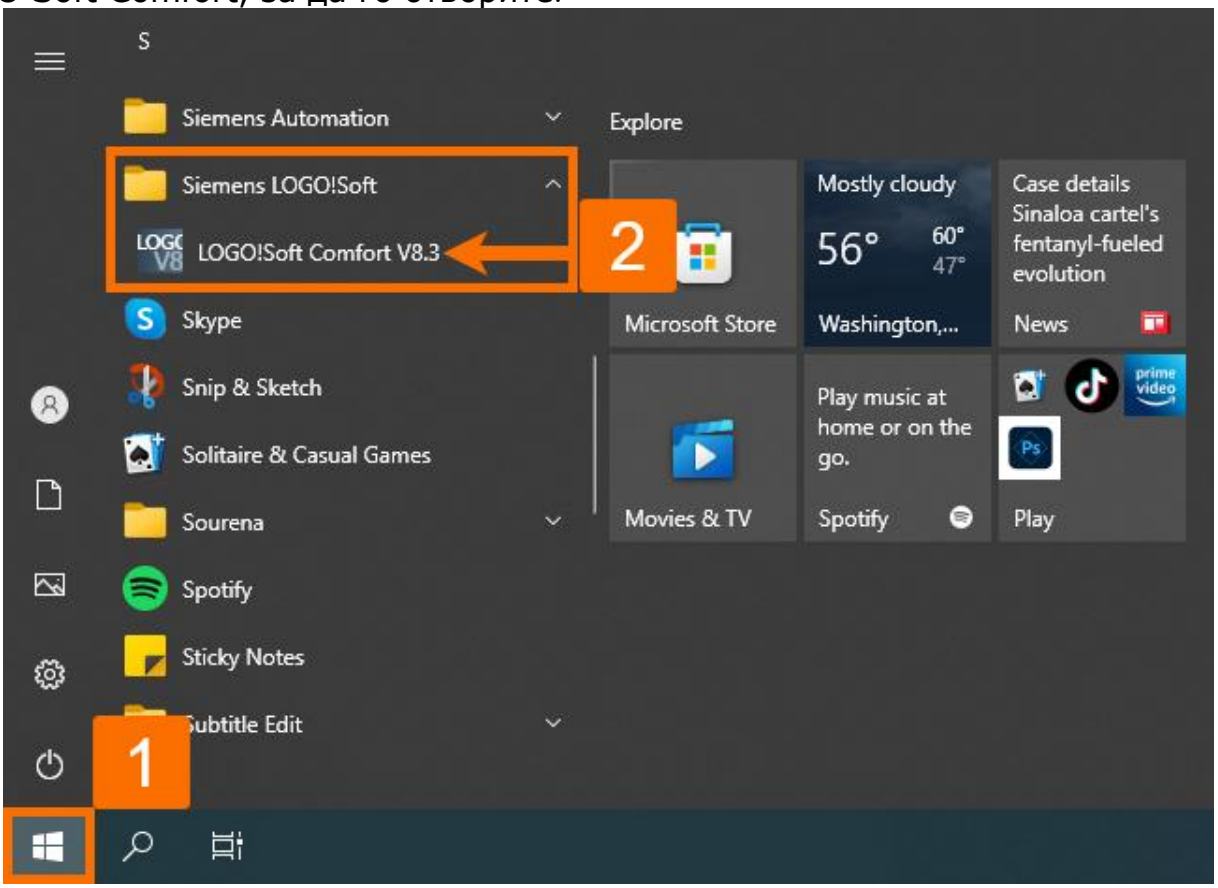
Генерирането на нова схема е от съществено значение за създаването на персонализирани системи за управление. Избирането и поставянето на блокове в програмата позволява на потребителя да проектира логическите функции за системата.

Бързите менюта за редактиране на обекти предлагат по-ефективен начин за

прилагане на промени в програмата. Конфигурирането на блокове позволява персонализиране на програмата, за да отговори на специфични системни изисквания.
Установяването на връзки между блоковете е необходимо, за да функционира правилно програмата. Разбирането на съветите за блокова връзка може да помогне за избягване на грешки и осигуряване на безпроблемна работа.
Като цяло овладяването на тези умения е от решаващо значение за ефективното и ефективно програмиране на Siemens LOGO PLC.

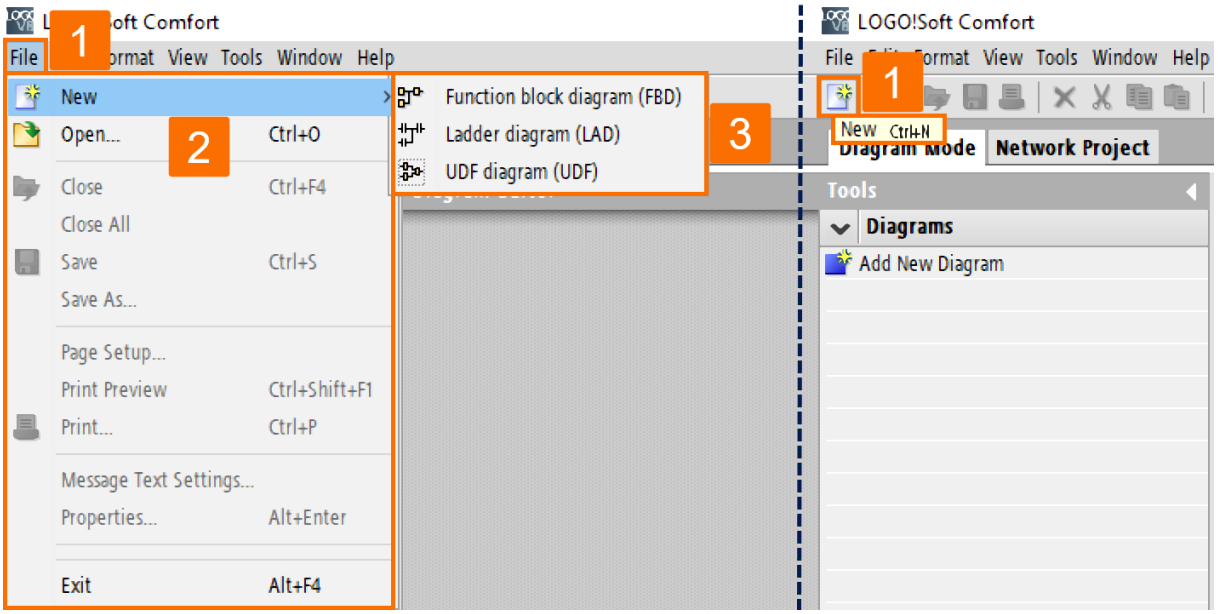
2.1. Генериране на нова програма

Започнете, като отворите стартовото меню на Windows. Превъртете надолу, докато намерите папката Siemens LOGO Soft. Разгънете го и изберете софтуера LOGO Soft Comfort, за да го отворите.



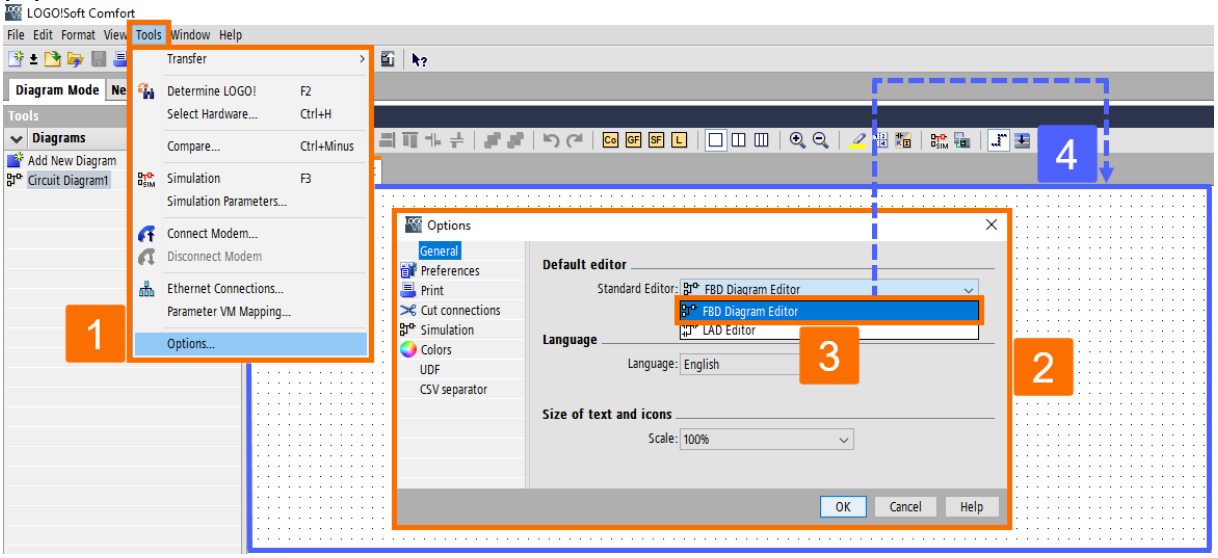
Фиг. 2.1. *Как да стартирате Siemens LOGO soft comfort*

Кликнете върху менюто Файл и изберете Нова команда, за да генерирате нова програма за схема. По този начин можете да отворите функционалната блокова диаграма (FBD), стълбовидна диагр (LAD) или редактора на UDF диаграми. Или можете да изберете бутона Създай, разположен в стандартната лента с инструменти.



Фиг. 2.2. *Как да генерирате нова програма*

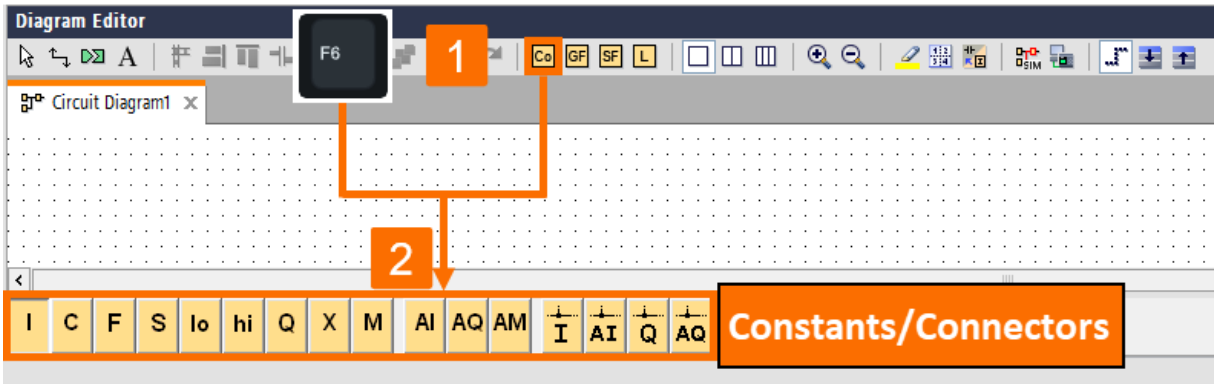
По този начин софтуерът ще стартира FBD редактора (или редактора по подразбиране, зададен в Инструменти/Опции/Стандартен редактор). Тя ви позволява да генерирате новата схема в нов прозорец в програмния интерфейс.



Фиг. 2.3. *Как да определите редактора за програмиране по подразбиране*

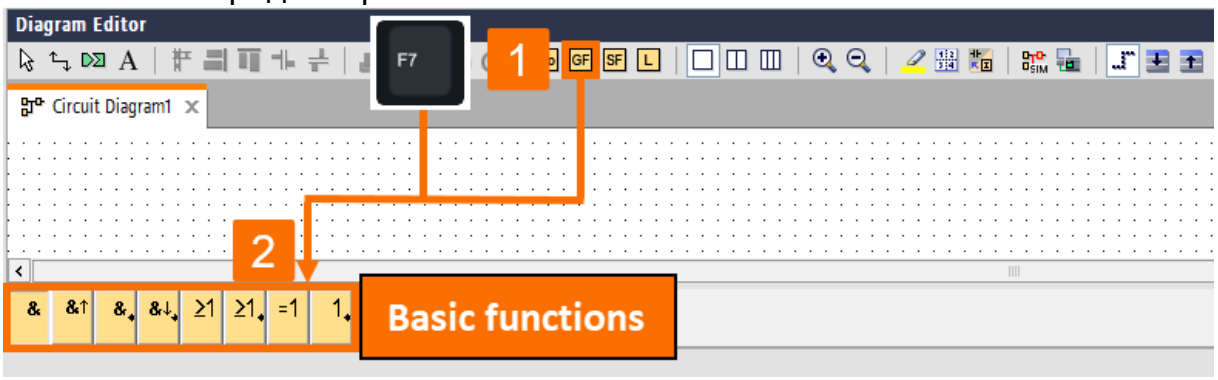
2.2. Избиране на блокове в Siemens LOGO Soft Comfort

Първоначалната стъпка в програмирането е да изберете подходящите блокове за вашата верига. Също така решете последователността, в която искате да вмъкнете I/O и стандартните/SFB блокове. Константите и терминалите, които съставляват набор от I/O и постоянни сигнали, могат да бъдат намерени под *Co* в лентата с инструменти за програмиране.



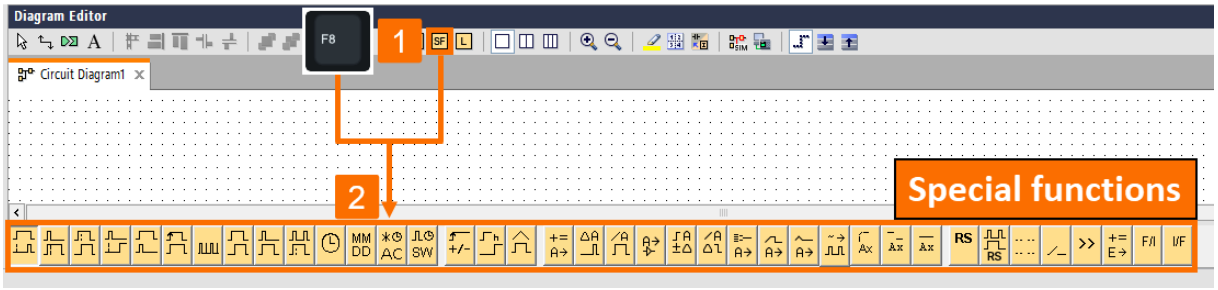
Фиг. 2.4. *Как да покажете групата константи/конектори*

Разположен под GF, можете да получите достъп до основните логически функции на булева алгебра - стандартни цифрови логически блокове. Той е достъпен само за редакторите на UDF и FBD.



Фиг. 2.5. *Как да покажете групата основни функции*

Специалните функции могат да бъдат намерени в SF. Можете също да получите достъп до споменатите функционални групи с помощта на функционалните клавиши.

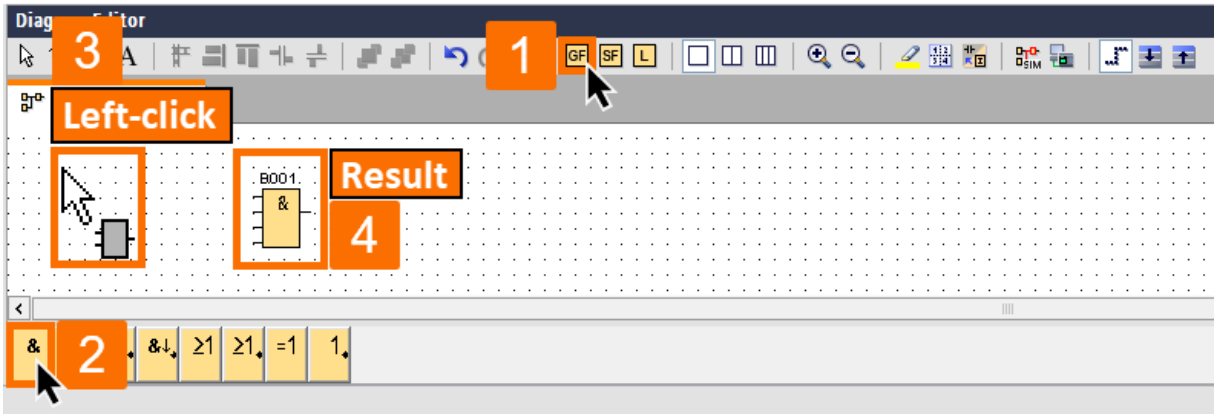


Фиг. 2.6. *Как да покажете групата със специални функции*

2.3. Поставяне на блокове в Siemens LOGO Soft Comfort

За достъп до желания блок щракнете върху групата икони, която съдържа блока. Като алтернатива можете да натиснете съответния функционален клавиш. При избор на функционална група, програмният интерфейс ще представи всички блокове, свързани с нея. Щракнете директно върху интерфейса, за да добавите избраната функция към вашия програмен интерфейс. С помощта на мишката можете да изберете други блокове, преди да ги поставите. Не е необходимо веднага да подравнявате блоковете, тъй като точното подравняване на този етап е безсмислено. Докато блоковете не бъдат свързани помежду си и коментарите не бъдат добавени към програмата на

веригата.

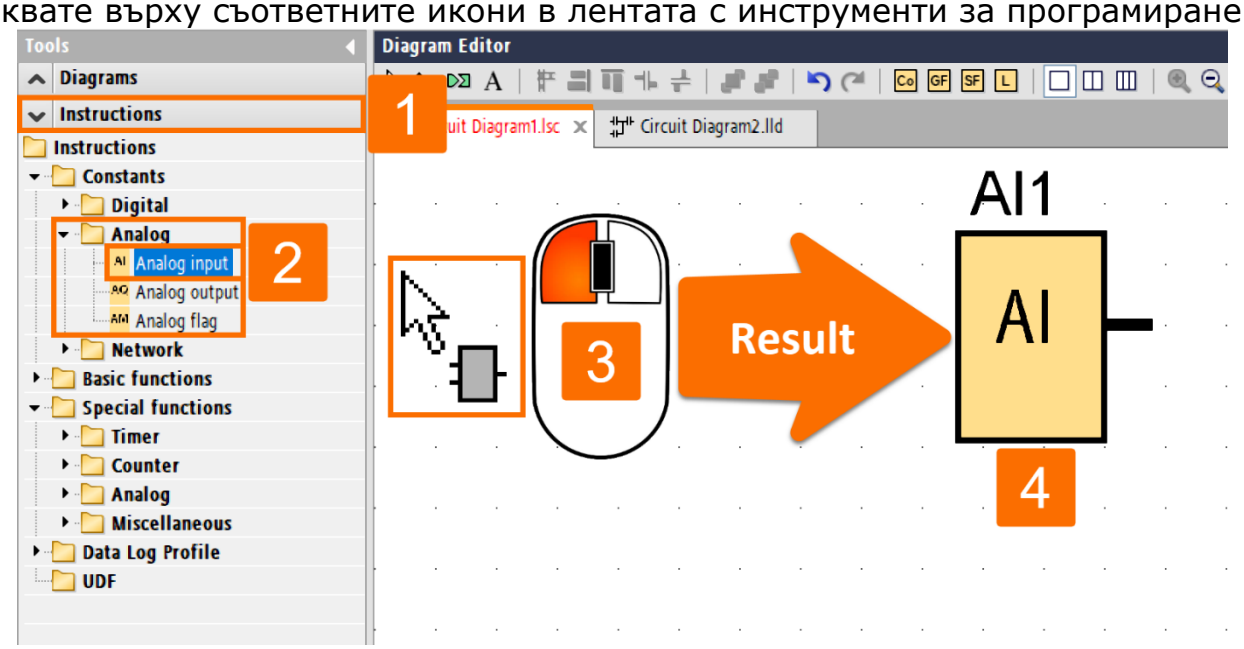


Фиг. 2.7. *Как да поставяте блокове с помощта на икони на лентата с инструменти за програмиране*

Има две опции, достъпни за вас, освен стандартния избор на блокове от иконите на лентата с инструменти за програмиране.

За да използвате първия метод, можете да разширите каталога от дървото на инструкциите и да изберете желания блок. След това щракнете с левия бутон върху позицията за вмъкване на блока във вашата програма за верига и блокът ще бъде поставен на правилното място.

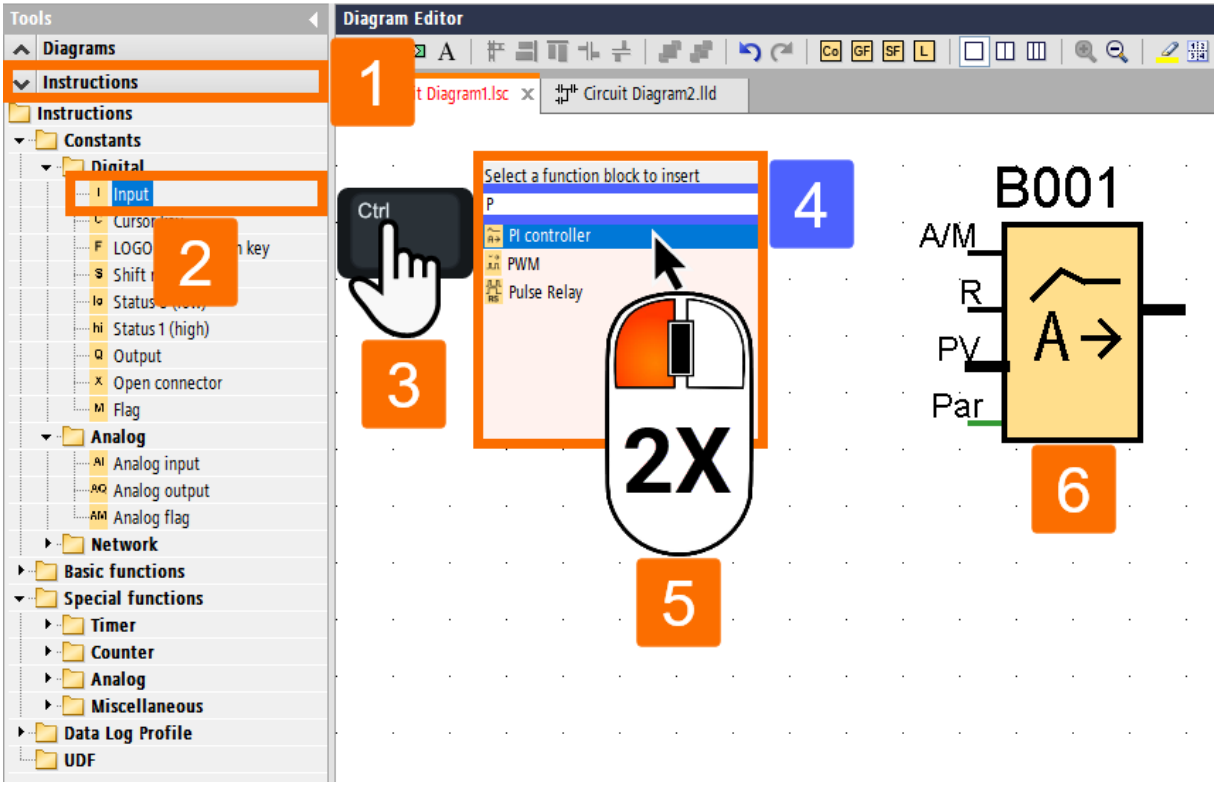
Предимството на този метод е, че можете лесно да превключвате между константи/терминали, SFB и основни функции. Така че не е необходимо да кликвате върху съответните икони в лентата с инструменти за програмиране.



Фиг. 2.8. *Как да поставяте блокове само с помощта на каталога*

Вторият метод включва отваряне на каталога на лентата с инструменти за програмиране и избиране на всеки блок, като щракнете върху него. По-добре е да затворите каталога и да скриете лентата с инструменти за програмиране, ако работите върху масивна програма. След това, като задържите натиснат клавиша Ctrl и щракнете с левия бутон върху позицията за вмъкване на блок,

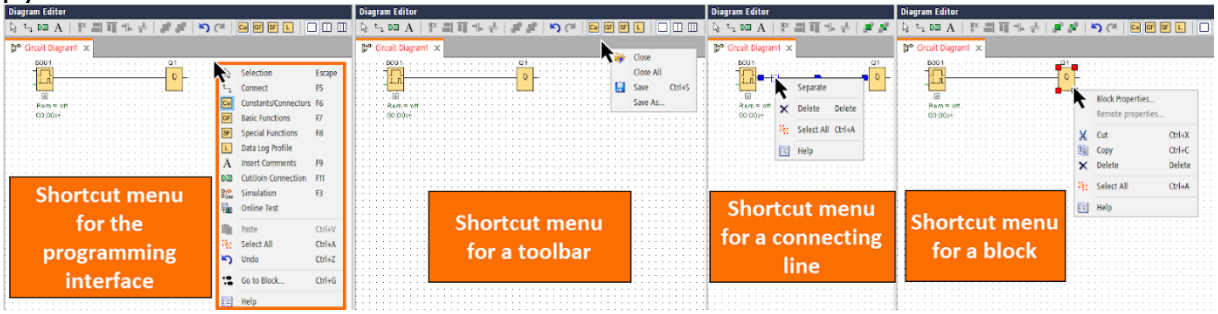
във вашата програма ще се появи маска със списък с блокове. Можете бързо да изберете необходимия блок от този списък, като щракнете двукратно върху него. Ако искате да стесните списъка, използвайте полето за въвеждане в заглавката на маската. За да направите това, въведете началната буква на необходимия блок, за да ограничите показването до списък с блокове с този инициал. Може да спести време и да ви помогне бързо да намерите съответния блок, без да преглеждате цялата маска. След като изберете блока, софтуерът ще го постави на подходящата позиция във вашата верига. Ако трябва да вмъкнете още екземпляри от един и същ блок, щракнете с левия бутон върху съответните позиции за вмъкване.



Фиг. 2.9. *Как да поставите блокове с помощта на каталога заедно с клавиша Ctrl*

2.4. Контекстни менюта

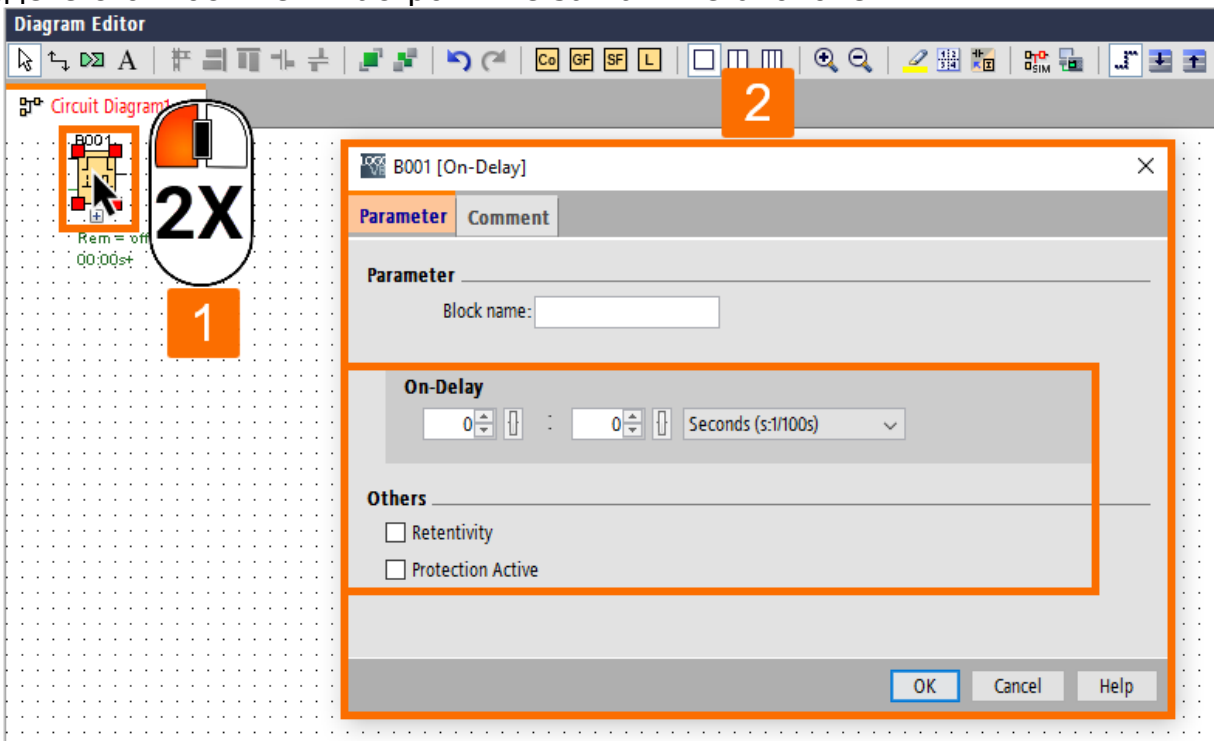
Бързите менюта могат да бъдат достъпни чрез щракване с десния бутон върху обект, което предоставя няколко опции за редактиране. Наличните опции за редактиране ще варират в зависимост от типа на избрания обект. Обектите включват блокове, свързващи линии, програмен интерфейс и ленти с инструменти.



Фиг. 2.10. *Бързи менюта за различни обекти*

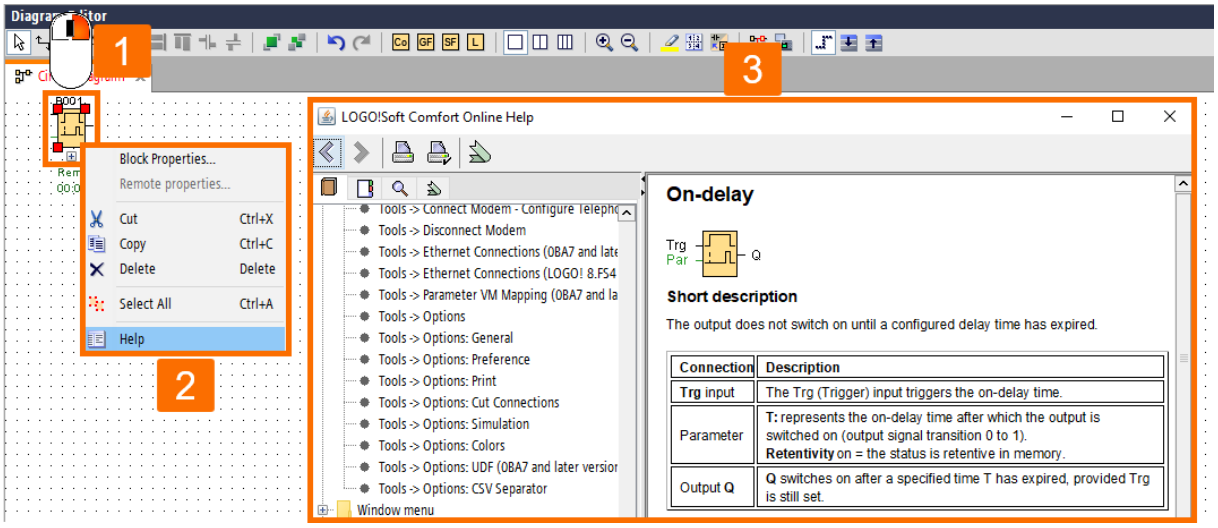
2.4. Конфигуриране на блокове в Siemens LOGO Soft Comfort

Чрез двукратно щракване върху блок можете да получите достъп до диалоговия прозорец за свойства, за да конфигурирате свойствата на блока. Той включва раздела Коментар заедно с раздели с параметри за специални функционални блокове, някои основни функции и константи и конектори. Тук можете да въведете стойностите и настройките за вашите блокове.



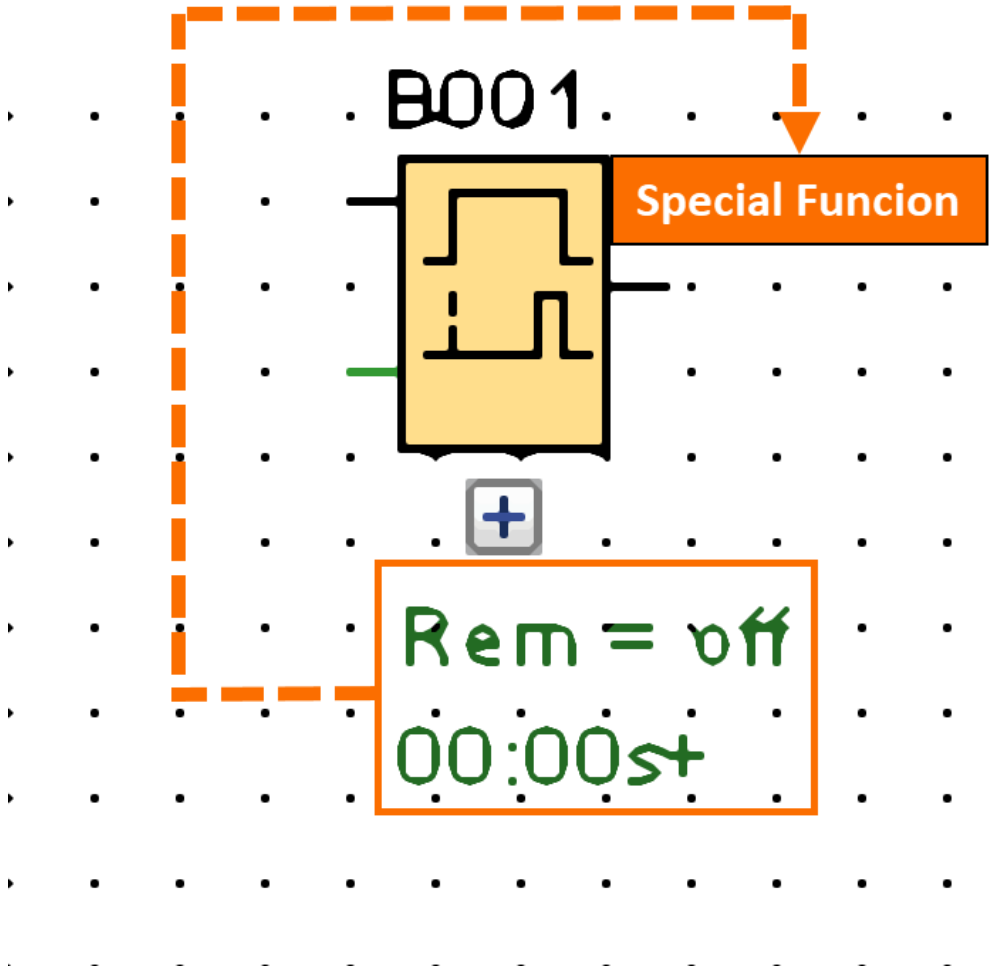
Фиг. 2.11. *Как да получите достъп до свойствата на специален функционален блок*

За да получите помощ за избрания обект и неговите параметри, щракнете с десния бутон върху него в програмния интерфейс. След това изберете елемента "Помощ" от контекстното меню. Той ще покаже подходяща помощна тема, свързана с обекта. Той също така предоставя допълнителни насоки и информация.



Фиг. 2.12. *Как да получите достъп до съответната помощ, свързана с избрания блок*

Зеленият надпис под блока на програмния интерфейс показва, че това е специална функция.

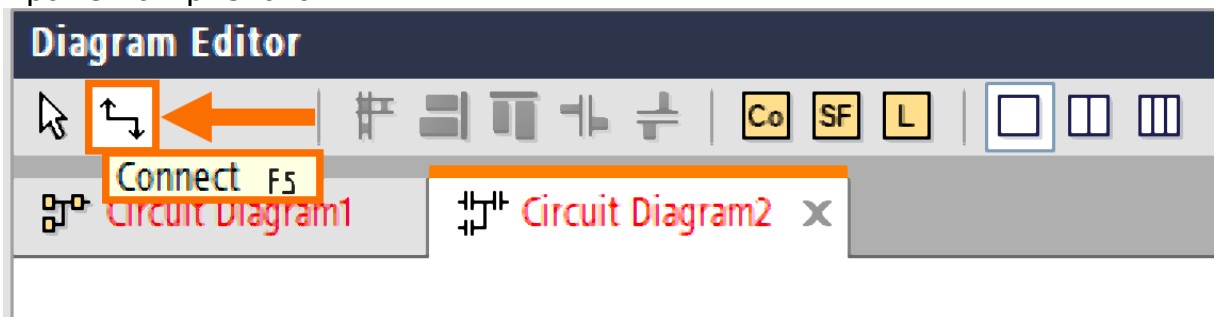


Фиг. 2.13. *Как да разпознаем специален функционален блок*

2.5. Установяване на връзки между блоковете

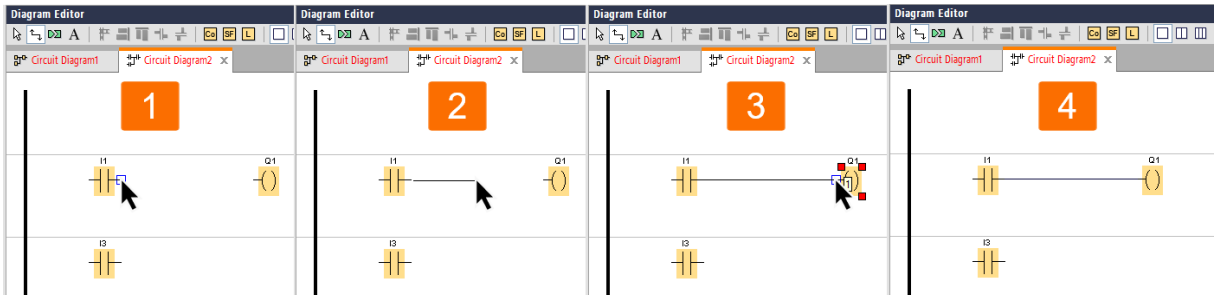
Свързването на блоковете е от съществено значение за завършване на

програмата на веригата. За да направите това, отидете до лентата с инструменти за програмиране и щракнете с левия бутон върху иконата за блокиране на връзката.



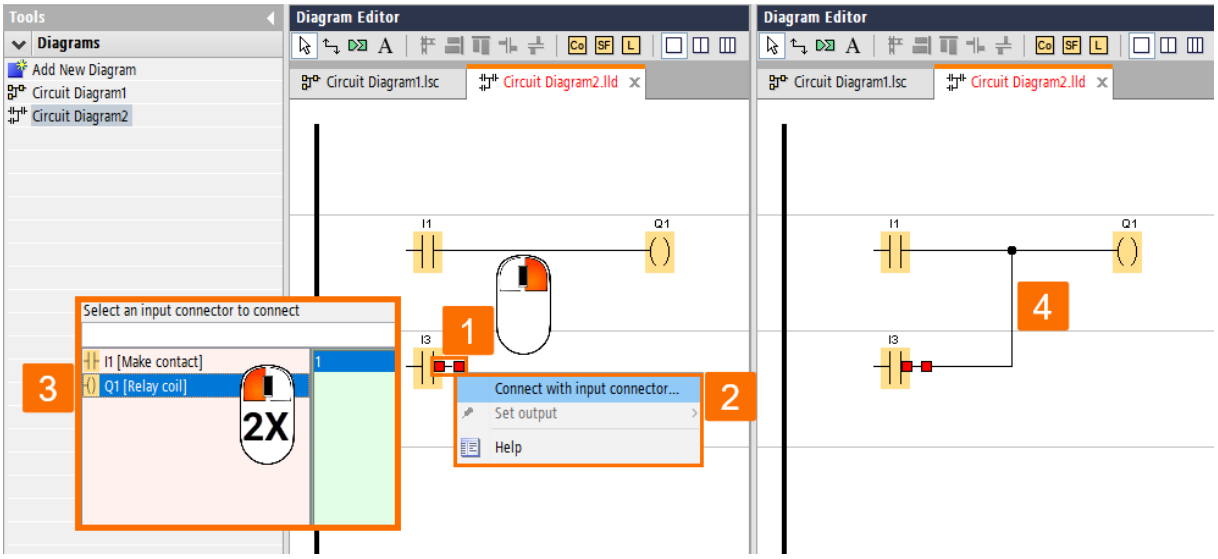
Фиг. 2.14. *Как да получите достъп до иконата за блокиране на връзката*

Как може да се осъществи свързването на два блока? Можете да кликнете върху конектора на един блок и да плъзнете мишката към конектор на друг блок. След това софтуерът ще установи връзка между двата терминала.



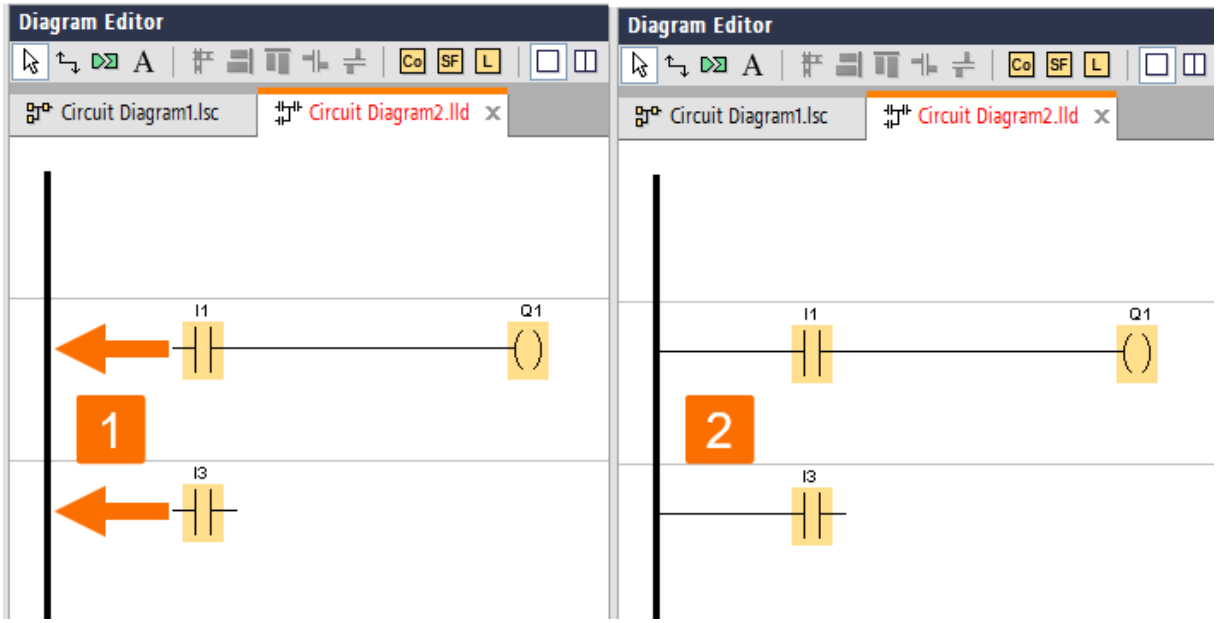
Фиг. 2.15. *Как да постигнете връзка между два елемента с помощта на иконата за блокова връзка*

Когато използвате LOGO Soft Comfort, можете да свържете блокове по различни начини. След като щракнете с десния бутон върху входа или изхода на блока, ще се появи контекстно меню. Оттам изберете "Свързване с..." опция за преглед на списък с всички налични блокове, с които да се свържете. Изберете желания целеви блок и софтуерът автоматично ще начертае свързващата линия. Полезно е, когато е необходимо свързване на източник и целеви блок на по-голямо разстояние в програмния интерфейс.



Фиг. 2.16. *Как да постигнете връзка между два елемента с помощта на контекстното меню*

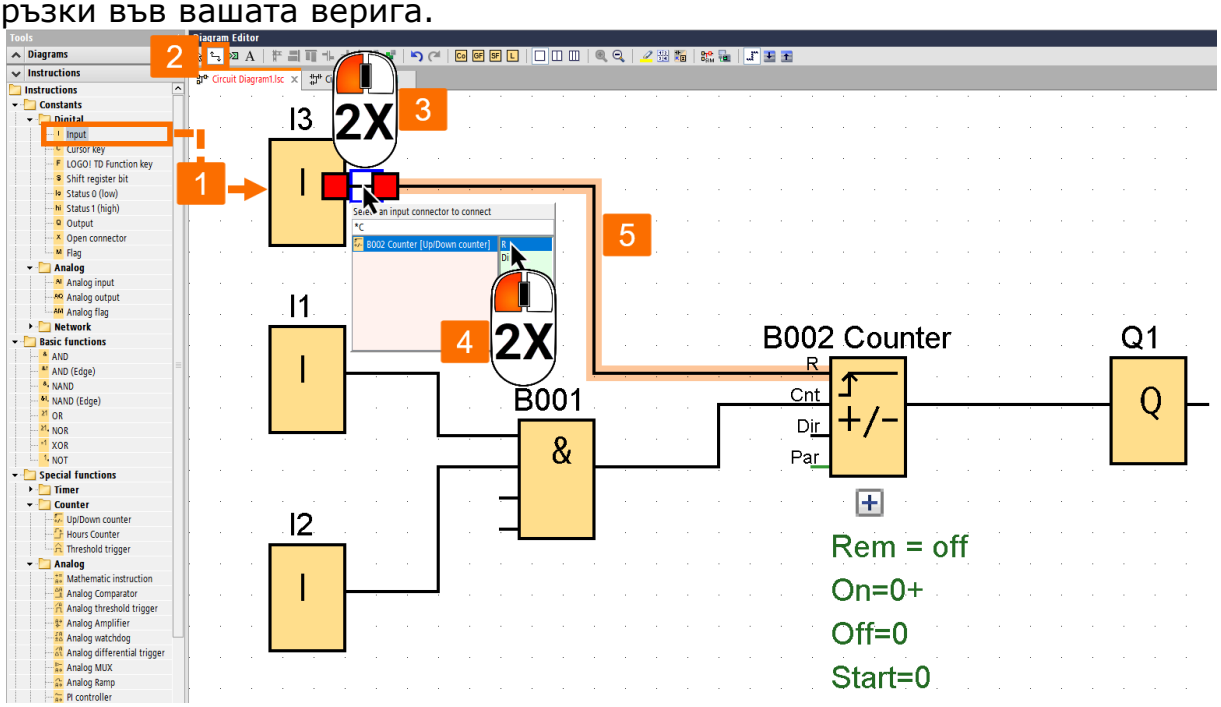
Важно е да се уверите, че входовете/изходите са свързани към лявата шина в прозореца на редактора, когато работите с редактора на стълбове диаграми.



Фиг. 2.17. *Как да установите връзката между I/O елементите и лявата шина на LAD редактора*

Освен стандартния подход за създаване на връзки, е наличен и алтернативен метод. След като поставите желания блок във вашата верига, можете да щракнете двукратно върху входа или изхода на блок, за да отворите маска със списък с целеви блокове. Там можете бързо да изберете желания блок, като щракнете двукратно върху него. Заглавката на маската включва и поле за въвеждане, което можете да използвате, за да стесните списъка с блокове, като въведете началната буква на необходимия блок. Също така можете да използвате заместващи символи като *. Може да спести време и да ви помогне

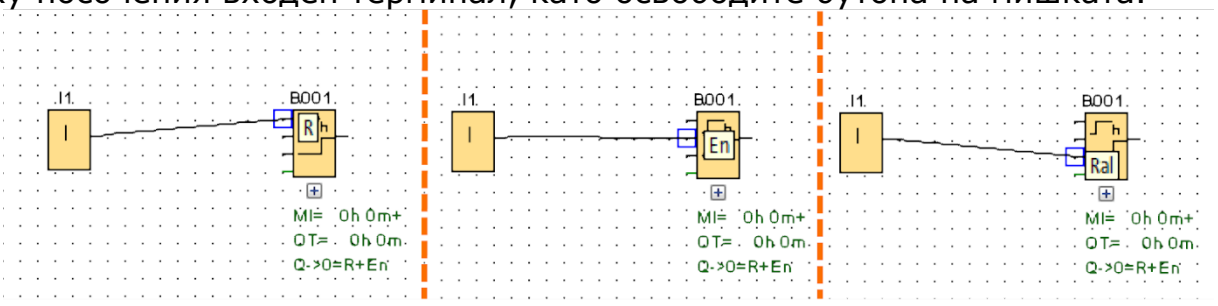
да намерите съответния блок по-ефективно, особено когато работите с големи вериги. След това софтуерът автоматично ще създаде връзката между двата блока. По този начин този метод осигурява бърз и лесен начин за генериране на връзки във вашата верига.



Фиг. 2.18. *Как да установите връзка между два блока с помощта на I/O терминали*

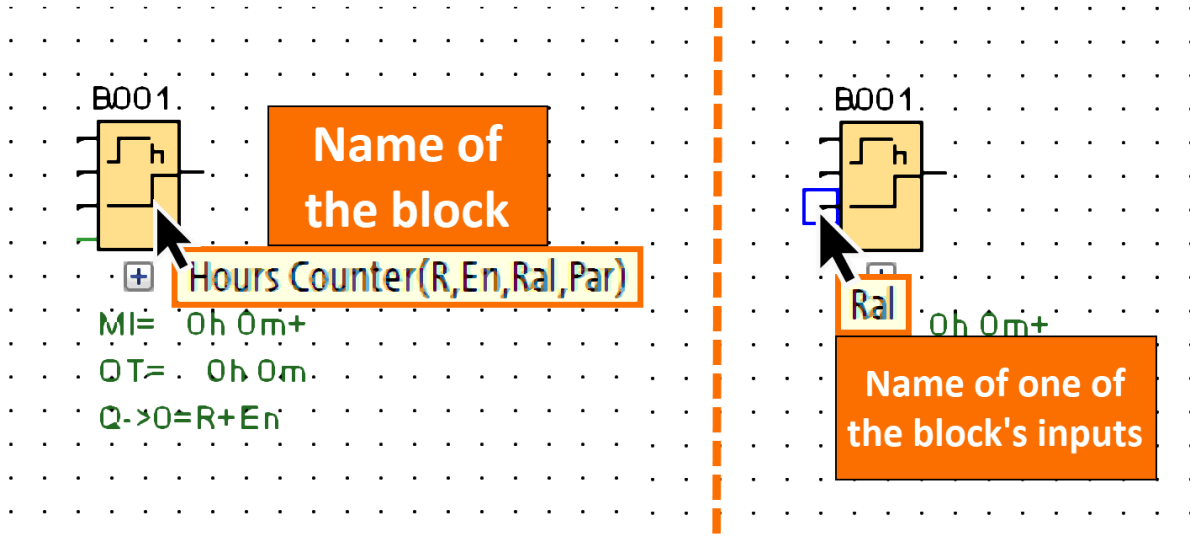
2.6. Съвети за свързване на блоковете

LOGO Soft Comfort ще представи пояснение, показващо връзката, когато свържете вход към изход или обратно. След това можете да щракнете линията върху посочения входен терминал, като освободите бутона на мишката.



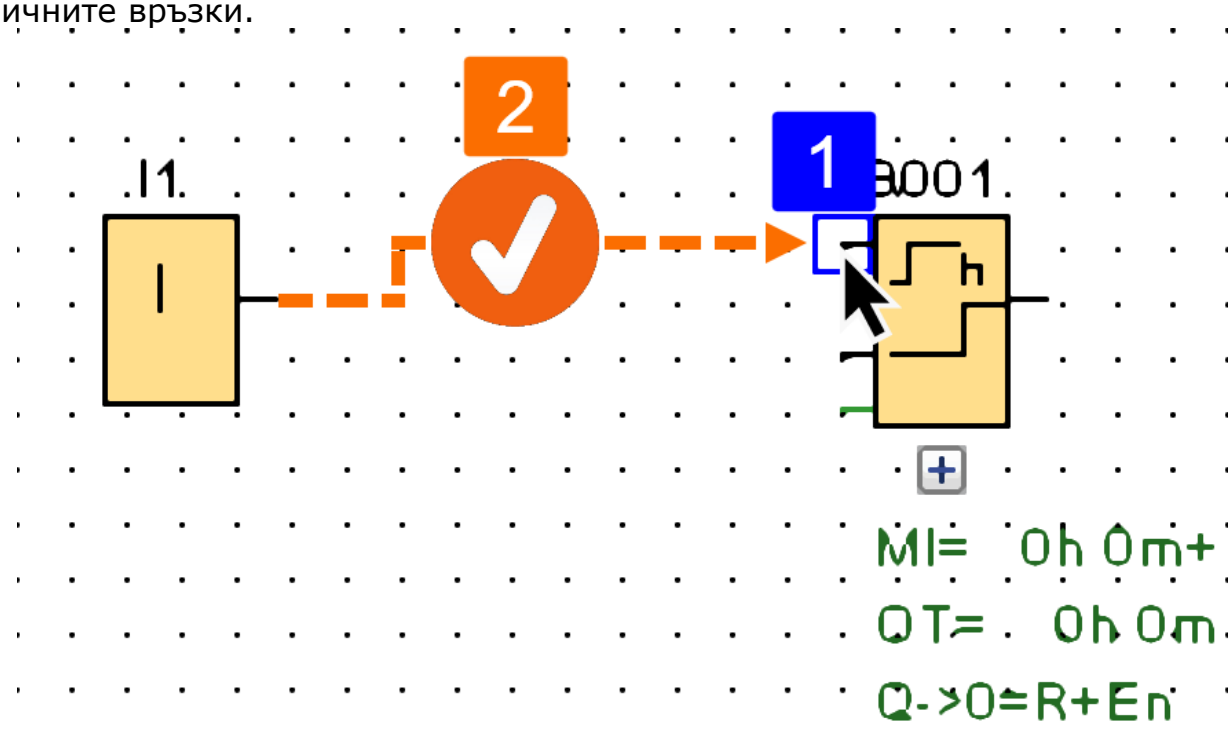
Фиг. 2.19. *Как да покажете подсказки, които показват връзката*

Ако имате нужда от допълнителна помощ при програмирането на схеми, можете да разчитате на кратка информация, наречена подсказки, предоставена от софтуера. Ако задържите курсора на мишката върху блок, можете да видите името му, докато задържането на курсора на мишката върху вход на блок ще разкрие името на входа.



Фиг. 2.20. *Как да получите достъп до името на блока и имената на блоковите терминали*

Когато блок вход (pin) е свързан, Софтуерът ще покаже синя рамка около показалеца на мишката. По този начин улеснява идентифицирането на наличните връзки.



Фиг. 2.21. *Как да проверите възможността за връзка между два терминала на два блока*

В заключение, софтуерът Siemens LOGO Soft Comfort е от съществено значение за програмирането на Siemens LOGO PLC. Софтуерният интерфейс и неговата интуитивна лента с инструменти и менюта за програмиране улесняват начинаещите да започнат. От друга страна, неговите разширени функции и функции отговарят на нуждите на опитни програмисти.

Досега научихте основите на генерирането на нова програма за схеми и избора

и поставянето на блокове. Научихте също как да конфигурирате блокове и да установявате връзки между блокове. Обсъдихме и някои практически съвети и преки пътища, които могат да помогнат за рационализиране на процеса на програмиране. След като следвате тези указания, можете да създадете ефективни схеми, които да отговарят на вашите специфични нужди.

2.7. Първи стъпки със симулиране на верижни програми в Siemens LOGO! Soft Comfort

Овладяване на симулационния процес в Siemens LOGO! PLC може да осигури няколко предимства.

Първо, той позволява откриване на грешки в програмата на управляващата верига, преди да бъде внедрена в реални ситуации. Второ, можете да идентифицирате недостатъци в дизайна и да коригирате параметрите, за да подобрите производителността и безопасността на веригата.

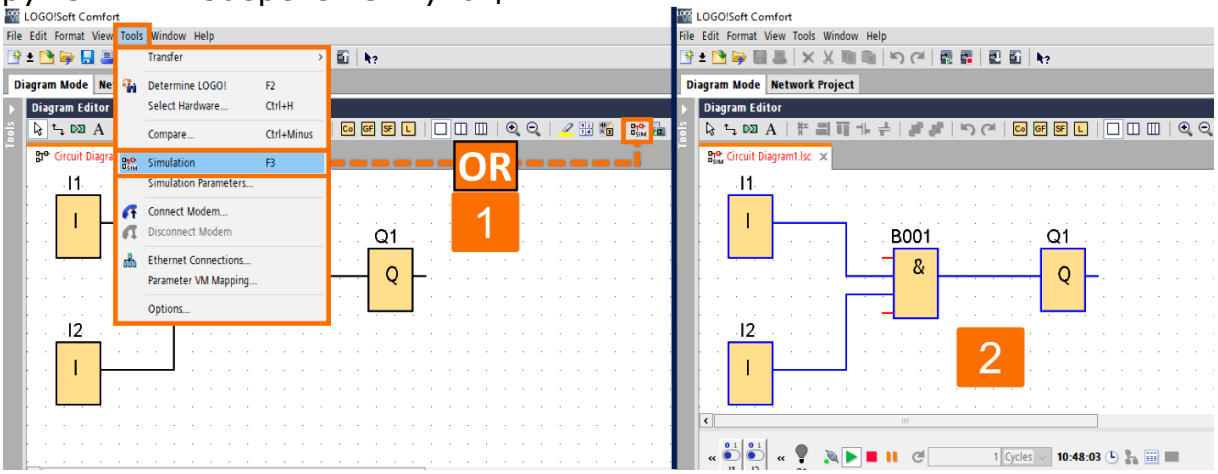
Трето, той позволява тестване на програмата на веригата без нужда от действителен хардуер, което може да спести време и ресурси. Четвърто, получавате разбиране за това как работи програмата, как взаимодейства с различни входове и как произвежда изходи.

Пето, той осигурява по-точно представяне на поведението на веригата от физическото тестване. Защо? Тъй като физическото тестване може да бъде повлияно от външни фактори. Шесто, той позволява тестване на различни сценарии без физически промени в контролната верига. По този начин осигурява по-голяма гъвкавост при експериментирането.

И седмо, помага да се идентифицират потенциалните рискове и опасности, свързани с програмата на веригата. Тя ни позволява да правим промени преди внедряването на веригата, намалявайки риска от грешки и злополуки.

2.7.1. Иницииране на процеса на симулация

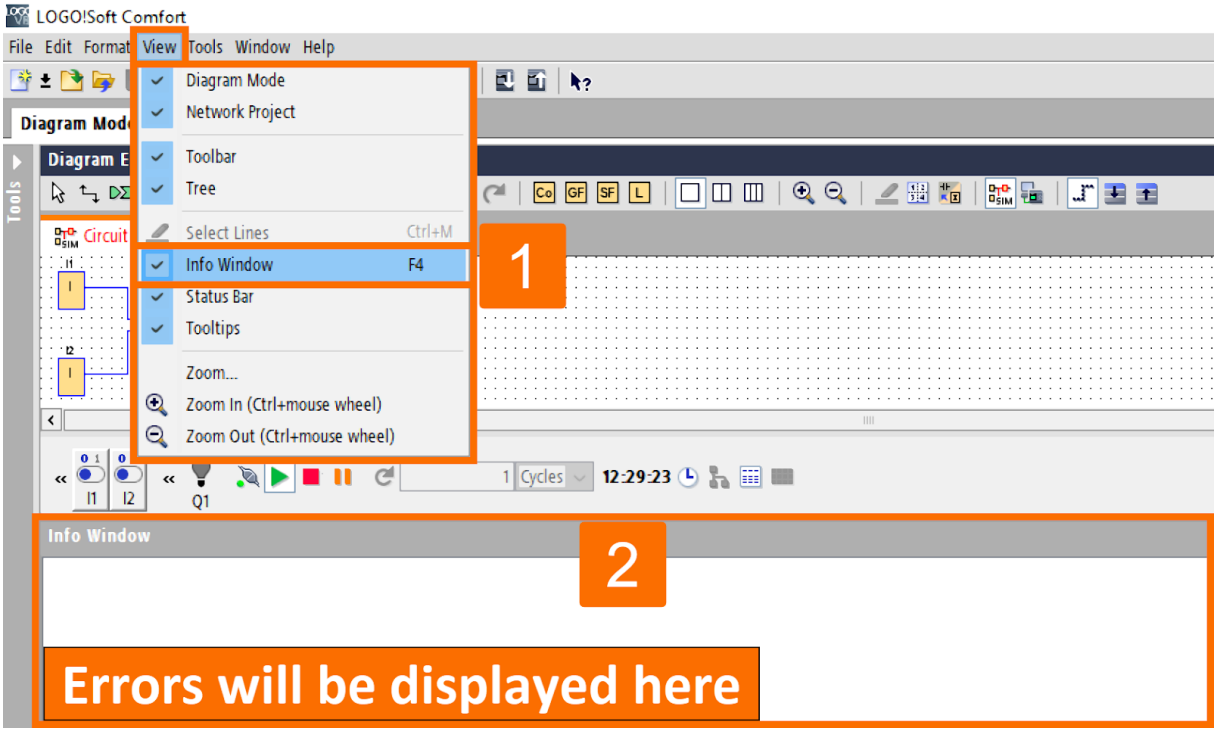
Има два начина да поставите вашата схема в режим на симулация. Първо, можете да използвате иконата за симулация, налична в лентата с инструменти за програмиране. Второ, отидете до менюто "Инструменти" в горната лента с инструменти и изберете "Симулация".



Фиг. 2.22. *Стартиране на процеса на симулация*

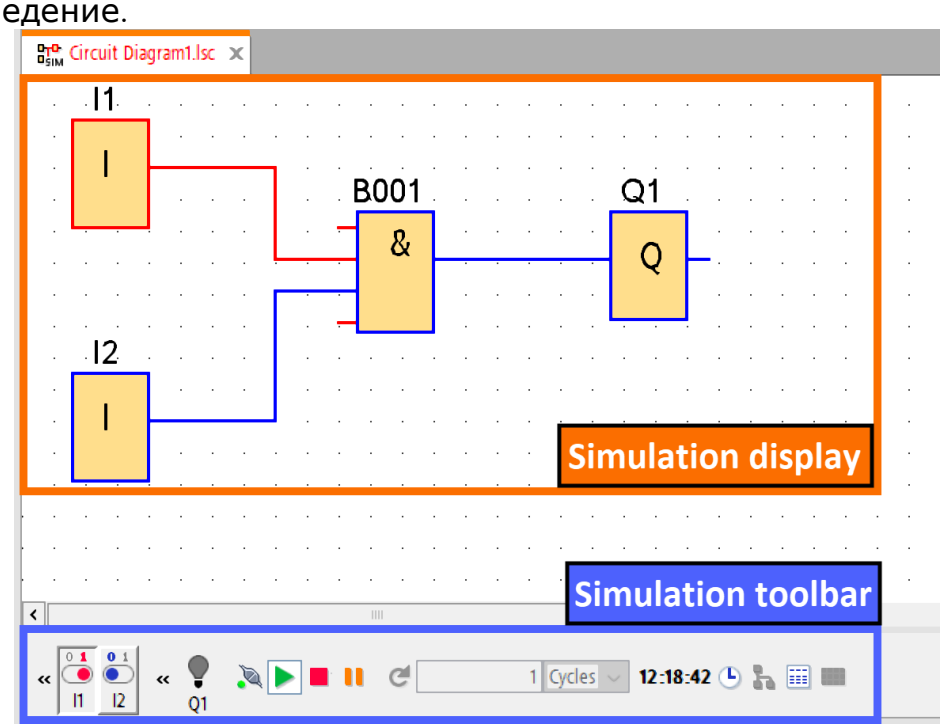
Когато започнете симулацията, LOGO! Soft Comfort провежда проверка на

веригата, за да открие и докладва грешки, показани в "Info Window." Можете да получите достъп до "Info Window" като изберете "View → Info Window" от лентата с менюта или натискане на функционалния клавиш [F4].



Фиг. 2.23. *Достъп до информационния прозорец*

Когато вашата схема е в режим на симулация, ви се предоставя лентата с инструменти за симулация и прозореца за състояние. Тези инструменти ще ви позволят да симулирате вашата програма и да наблюдавате и контролирате нейното поведение.

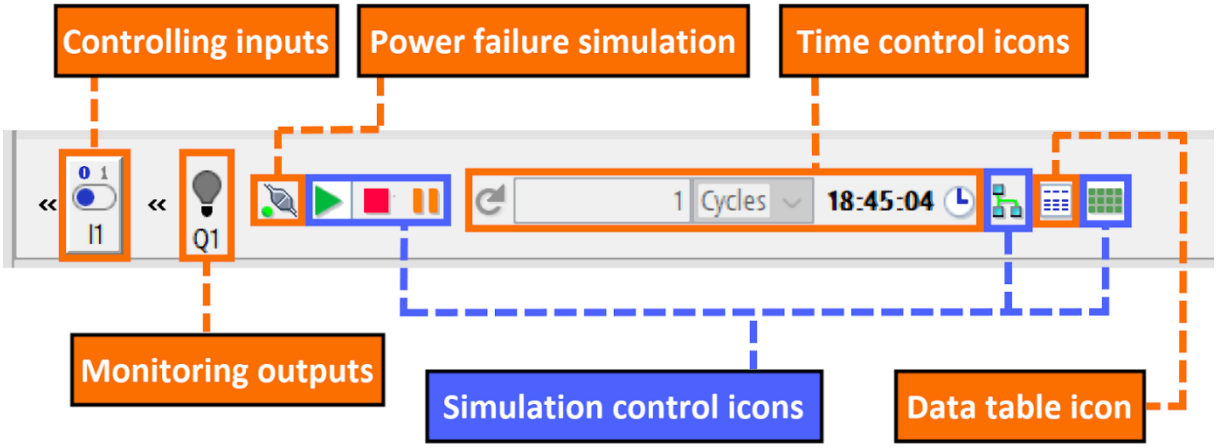


Фиг. 2.24. *Лента с инструменти за симулация и показване на състоянието на*

симулацията

В лентата с инструменти за симулация можете да получите достъп до набор от икони, всяка с уникални функции, които включват следното:

- Иконите, подобно на превключвателите, са предоставени на потребителите да контролират входовете.
- Иконата за симулация на прекъсване на захранването. Позволява тестване на реакцията на превключване по отношение на характеристиките на задържане след прекъсване на захранването.
- Иконите, подобно на крушките, са предоставени на потребителите, за да наблюдават изходите.
- Контролни икони за регулиране на процеса на симулация.
- Контролни икони за управление на времето.
- Икона за управление на таблица с данни.

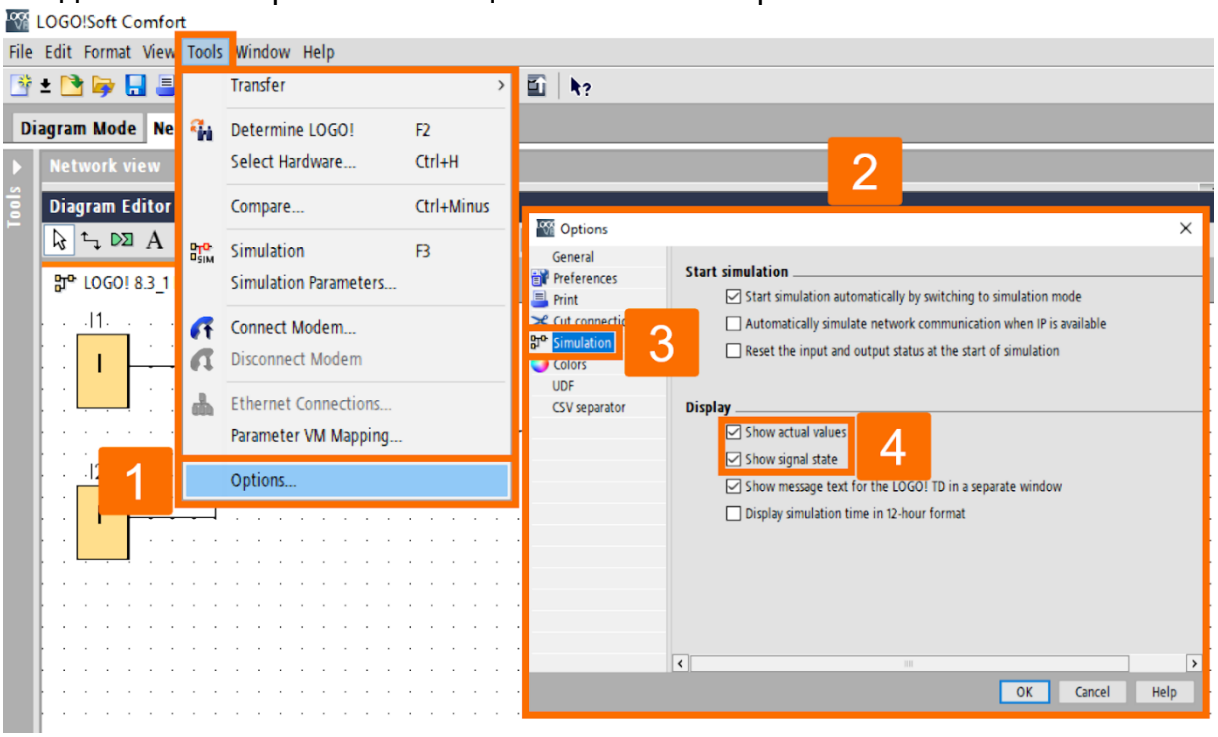


Фиг. 2.25. *Компоненти на лентата с инструменти за симулация*

Като кликнете върху бутона << , можете да накарате определена част от лентата с инструменти да изчезне. Ако искате да го върнете, трябва да кликнете върху бутона >> .

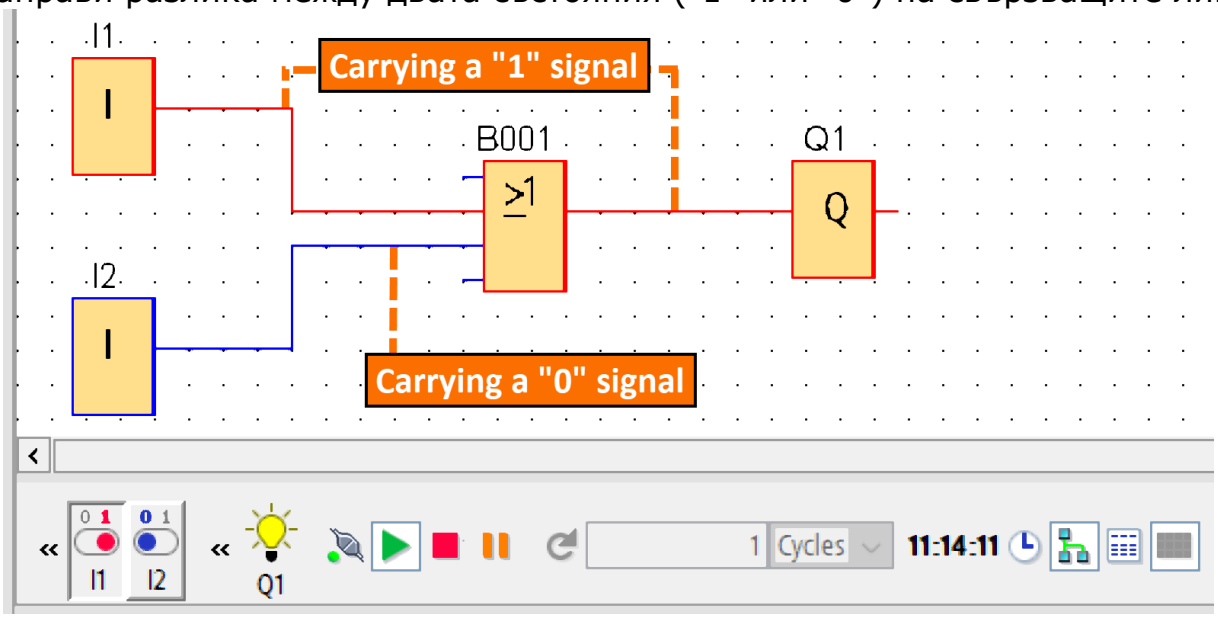
Фиг. 2.26. *Скриване и показване на определена част от лентата с инструменти за симулация*

Ако искате да видите състоянието на сигналите и променливите на процеса, можете да ги активирате с помощта на Tools → Options: Simulation command.



Фиг. 2.27. *Активиране на състоянието на сигнала и променливите на процеса на симулация*

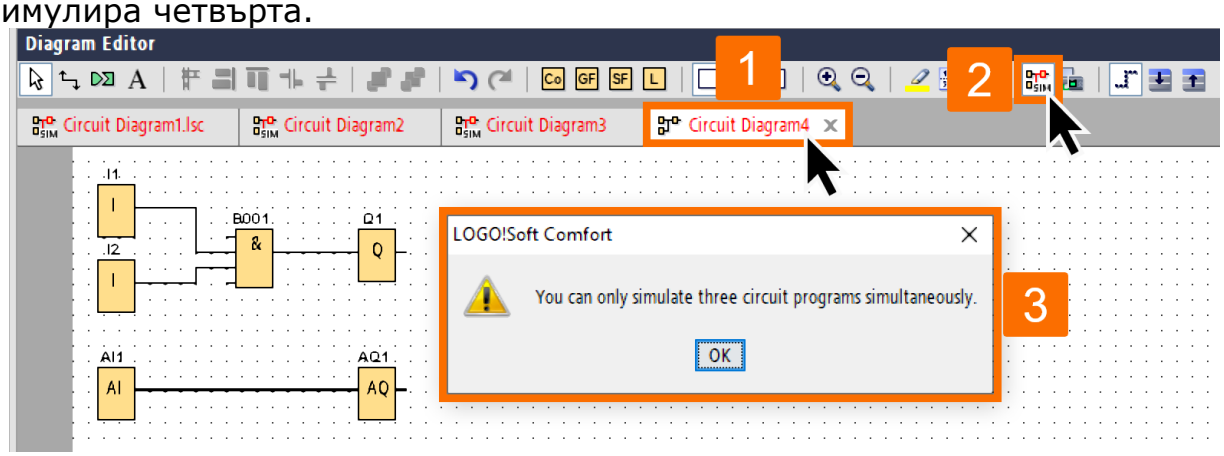
По подразбиране свързващите линии на дисплея са обозначени в червено или синьо, за да покажат дали носят сигнал "1" или "0". Този цветен знак помага да се направи разлика между двата състояния ("1" или "0") на свързващите линии.



Фиг. 2.28. *Идентифициране на състоянието на свързващите линии*

Софтуерът позволява едновременна симулация на до три веригови програми.

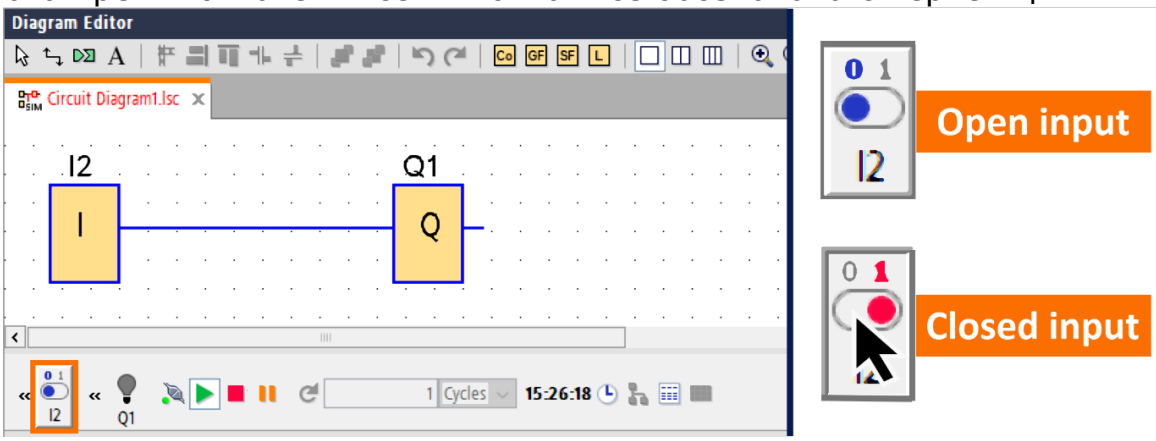
Предупредителен диалогов прозорец ще се появи, ако потребителят се опита да симулира четвърта.



Фиг. 2.29. *Симулиране на програма за четвърта верига, която идва с предупреждение*

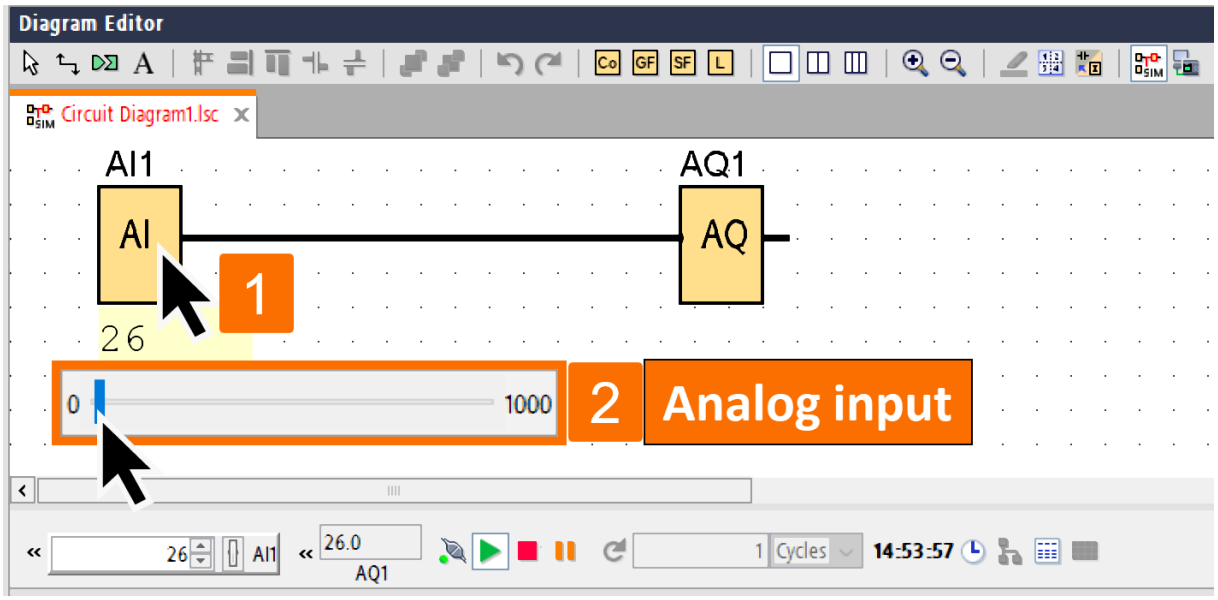
2.7.2. Оформление на входовете

Входове в рамките LOGO Soft Comfort са представени от икони, наподобяващи ключове или превключватели. Също така, съответните им имена ще бъдат показани под иконите. Когато входът е отворен в LOGO Soft Comfort, той се представя като деактивиран превключвател със син цвят. При щракване върху иконата превключвателят се включва и се обозначава с червен цвят.



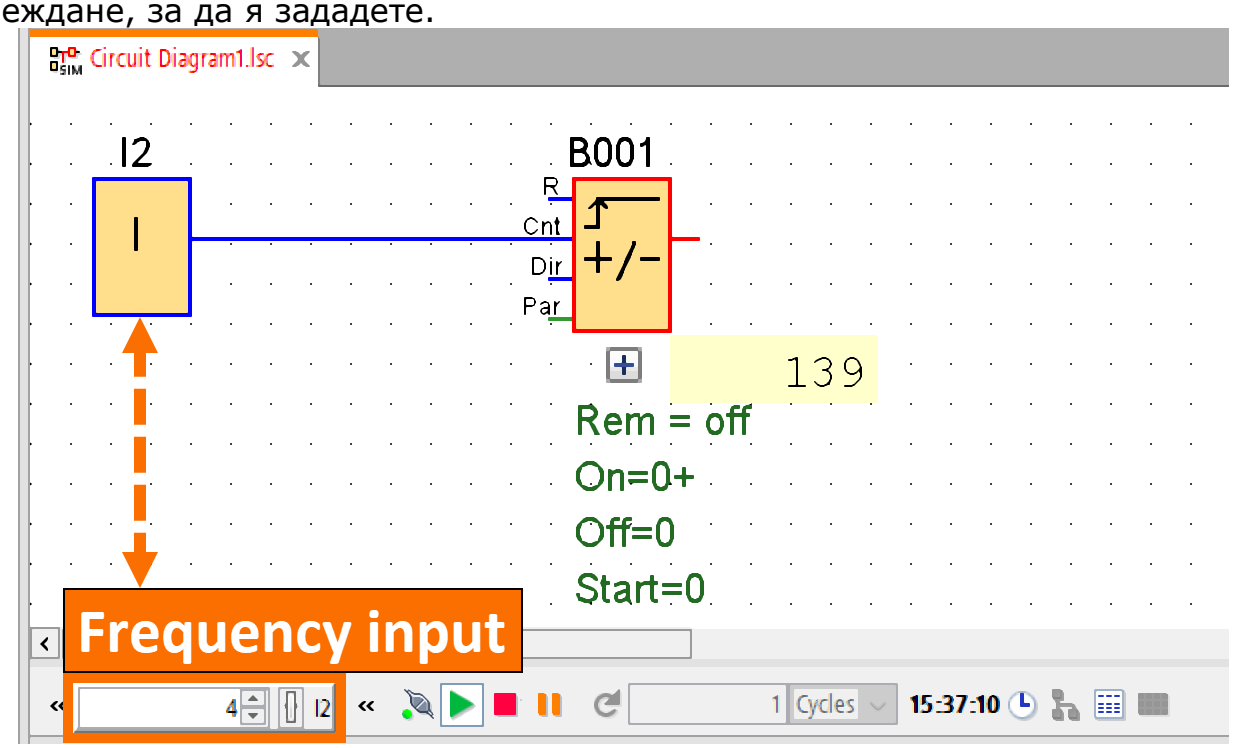
Фиг. 2.30. *Симулационно оформление на цифрови входове*

Ако трябва да зададете стойностите за аналогови и честотни входове, една от опциите е да използвате плъзгащ се резистор за регулиране на аналоговото напрежение или честота. За да направите това, щракнете върху съответния блок на диаграмата, за да получите достъп до плъзгащия се резистор и да го управлявате директно.



Фиг. 2.31. *Симулационно оформление на аналогови входове*

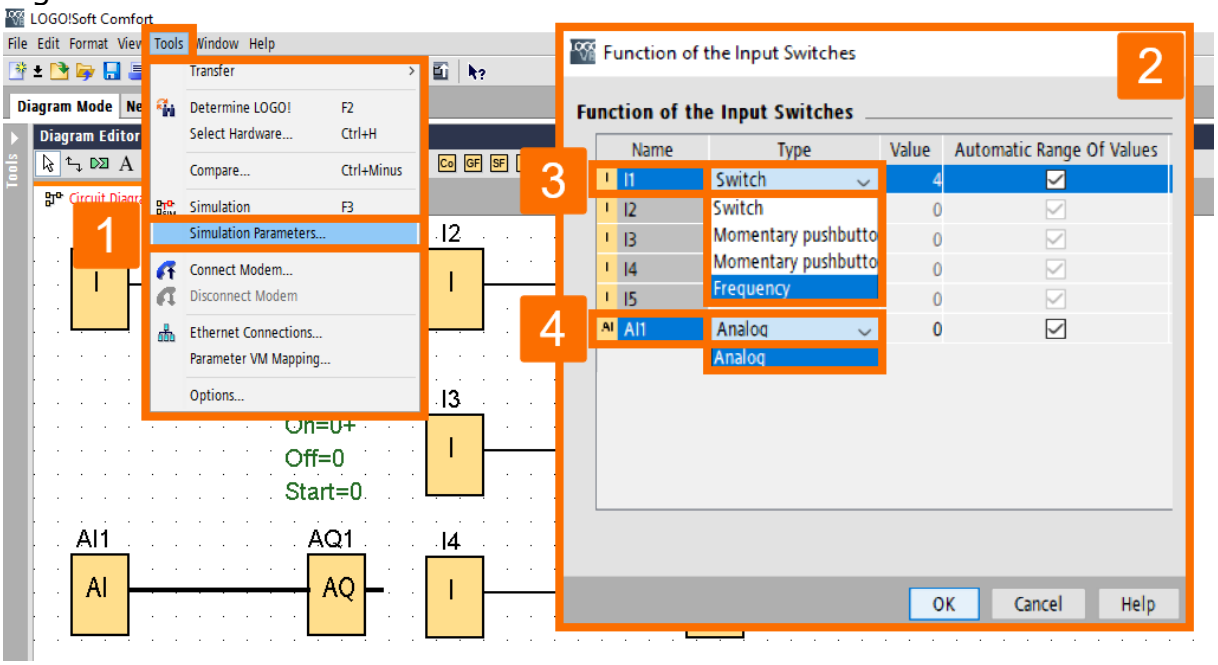
За да въведете по-точна стойност, можете да я въведете директно или да използвате клавишите нагоре/надолу, разположени до прозореца за въвеждане, за да я зададете.



Фиг. 2.32. *Въвеждане на точна стойност за честотен вход*

Можете да посочите входния отговор за целите на симулацията. За да постигнете това, отидете до горната лента с инструменти, за да щракнете върху "Tools," след това изберете "Simulation Parameters." Диалоговият прозорец показва само входовете, използвани в момента в програмата на веригата. Вашите цифрови входове могат да бъдат конфигурирани по един от четирите начина. Тези опции са "Switch," "Momentary pushbutton (make)," "Momentary pushbutton (break)," или "Frequency." Вашият избор е ограничен, когато

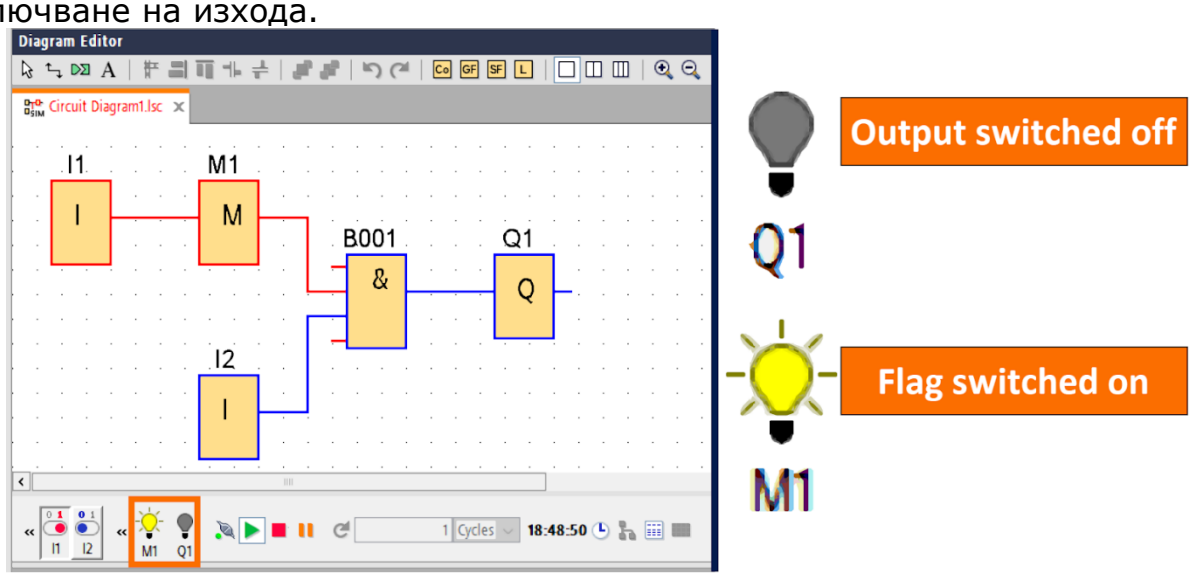
аналоговите входове влязат в действие, тъй като имате само една опция: "Analog."



Фиг. 2.33. *Определяне на входната реакция за целите на симулацията*

2.7.3. Оформление на изходите

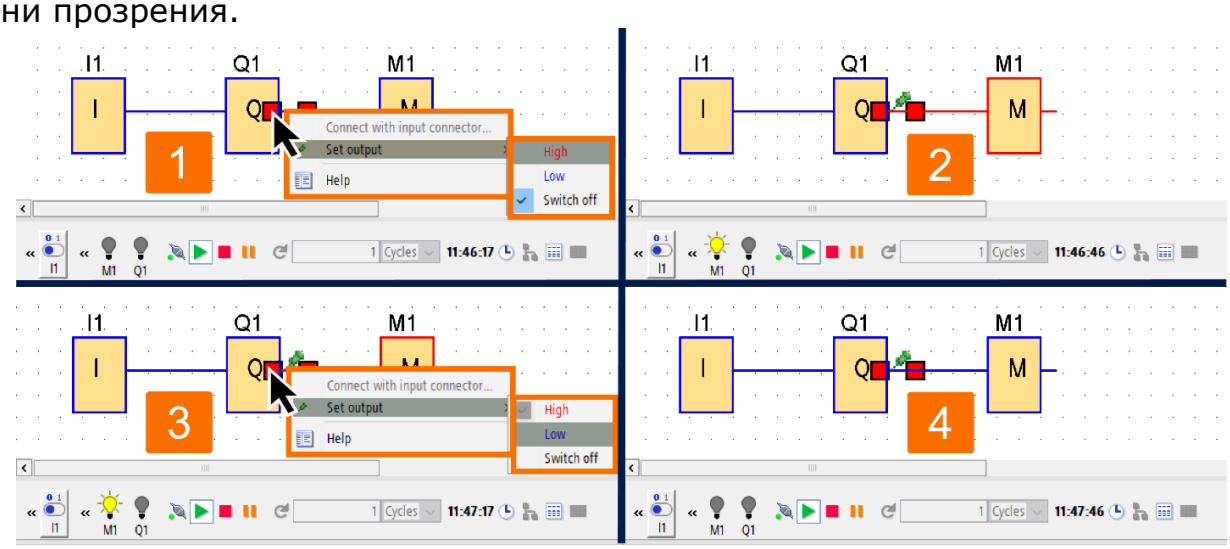
След като влезете в режим на симулация, Logo Soft Comfort ще покаже флаговете M и извежда Q като определени изходни опции. Състоянието на изхода или флага може да се различи с лекота в LOGO Soft Comfort. Постига се чрез икони със светли и тъмни крушки. Изключена крушка показва изключения изход или флаг, а ярка крушка представлява включения изход или флаг. Изходното име в програмата на веригата ще се покаже под иконата на крушката в Logo Soft Comfort. Той помага на потребителите при бърза идентификация. Иконата на крушката в софтуера служи само като визуално представяне на състоянието на изхода и не е превключвател за включване или изключване на изхода.



Фиг. 2.34. *Симулационно оформление за флагове и изходи*

2.7.4. Задаване на изход

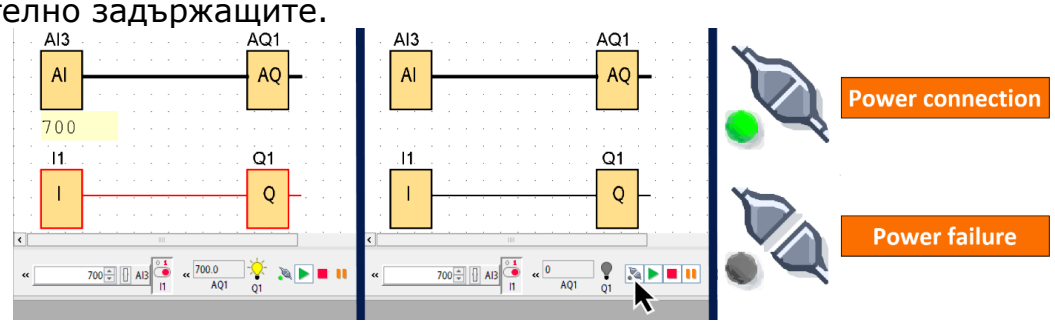
Когато работите в симулационен режим, изходът на блока може да се настрои с помощта на десен бутон върху цифровия изходен щифт на блока. Ако използвате тази команда, можете да конфигурирате изхода, независимо от текущата ситуация на блока, без никакви ограничения. След като бъде зададен, изходът ще остане в това състояние, докато не го активирате отново или спрете симулацията. В резултат на това можете да оцените как дадена програма ще реагира на конкретни състояния, като по този начин предоставяте ценни прозрения.



Фиг. 2.35. *Настройка на изхода на блока, независимо от текущата ситуация на изходния блок*

2.7.5. Прекъсване на захранването

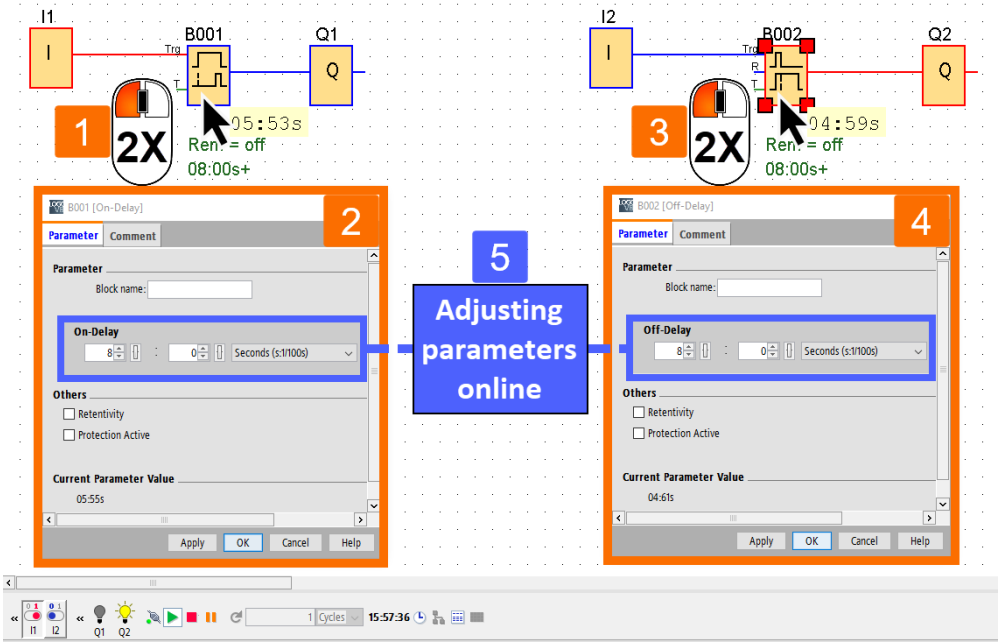
Чрез натискане и задържане на иконата за захранване потребителят може да симулира прекъсване на захранването, което прекъсва захранването на всички входове. По този начин можете да подражавате на ефектите от действително прекъсване на електрозахранването. Използвайки тази функция, можете да оцените поведението на веригата, когато възникне прекъсване на захранването и възстановяване на захранването. Освен това можете да оцените възможностите за задържане на вашата програма. Въпреки че задържането не е проблем в началото на симулацията, то става от решаващо значение при използване на функцията за прекъсване на захранването. LOGO Soft Comfort третира началото на симулацията така, сякаш функцията за зареждане на програмата е активирана. Това води до нулиране на всички стойности, включително задържащите.



Фиг. 2.36. *Емулиране на ефектите от действително прекъсване на захранването*

2.7.6. Конфигурация на параметрите по време на симулационния процес

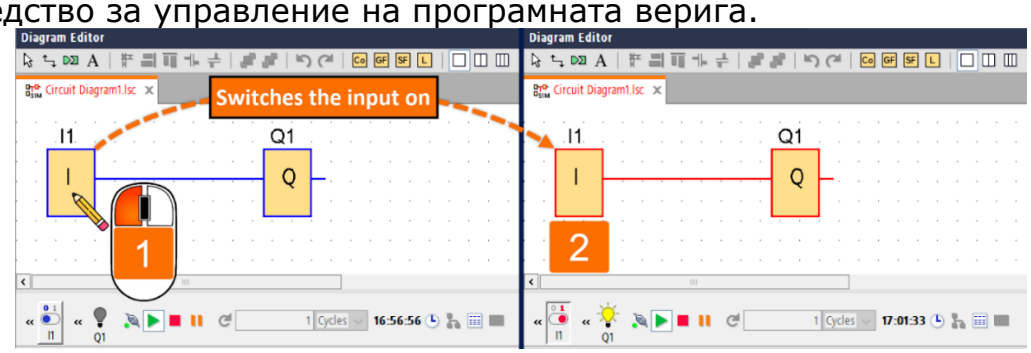
Докато симулационният режим е активен, можете да отворите диалоговия прозорец за свойства на блока, като щракнете двукратно върху съответния блок. Тук, подобно на режима на програмиране, можете да редактирате коментари и параметри в този контекст. Това ви позволява да настроите фино настройките си и да направите необходимите промени. Когато софтуерът е в режим на симулация, можете да наблюдавате стойностите на параметрите в реално време. Може да се използва като техника за анализ за оценка на функционалността на вашата верига. В режим на симулация можете да отворите няколко прозореца за присвояване на параметри едновременно. По този начин можете да направите няколко корекции на програмата на веригата заедно, рационализирайки процеса.



Фиг. 2.37. *Регулиране на параметрите на блоковете в средата на процеса на симулация*

2.7.7. Алтернативен подход

Можете да активирате или деактивирате входовете, като щракнете директно върху тях, като влезете в режим на симулация. Той осигурява ефективно и лесно средство за управление на програмната верига.



Фиг. 2.38. *Алтернативен подход за управление на цифрови входове в симулационен режим*

В заключение, научихте два начина за започване на процеса на симулация. Можете да използвате Tools → Команда на менюто за симулация в горната лента с инструменти или иконата за симулация в лентата с инструменти за програмиране. Научихте къде да проверите откритите грешки при стартиране на симулацията. Запознахте се с лентата с инструменти за симулация и нейния набор от икони, всяка с уникални функции. Разбрахте показването на състоянието на симулацията и как да правите разлика между двата състояния ("1" или "0") на свързващите линии. Научихте се да управлявате цифрови и аналогови входове с помощта на икони на превключватели и плъзгащи се резистори.

Запознахте се с показването на флагове и изходи в режим на симулация. Разбрахте как да конфигурирате изхода, независимо от текущата ситуация на блока, без никакви ограничения. Научихте се да подражавате на ефектите от действително прекъсване на електрозахранването, което прекъсва захранването на всички входове. Накрая научихте как да конфигурирате параметрите на блока точно в средата на процеса на симулация.

По същество овладяването на симулационния процес в Siemens Logo PLC може да осигури ползи, които могат да спестят време, пари и усилия. В същото време това води до подобряване на производителността, ефективността и безопасността на програмната верига в софтуера.

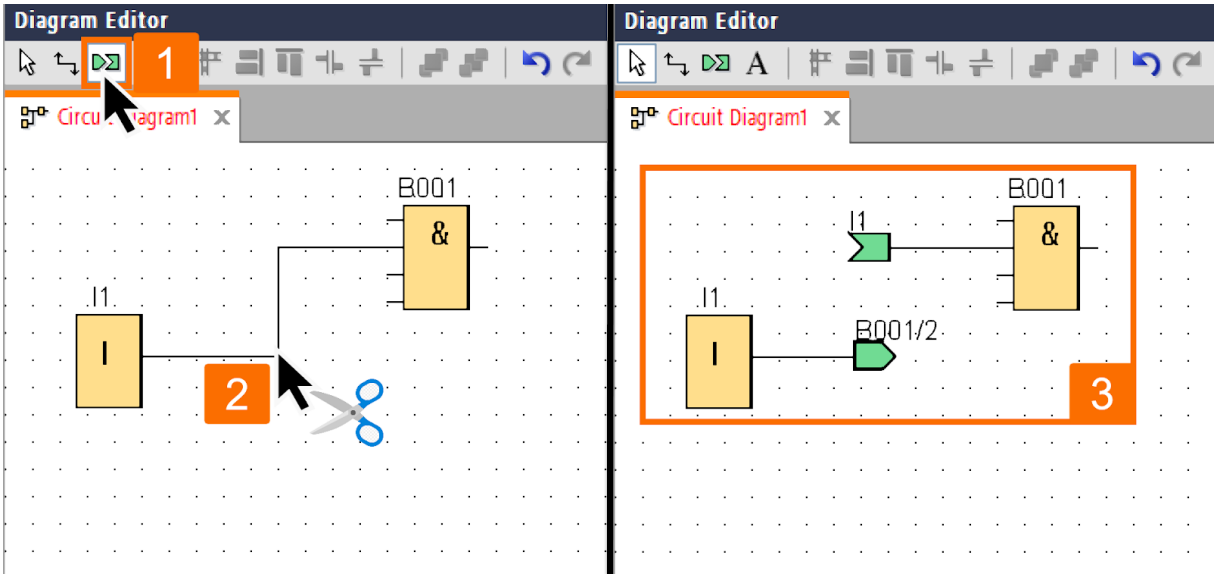
2.8. Режещи връзки и документация на програмната верига

Siemens LOGO! PLC са съществени части от индустриалната автоматизация, използвани в малки приложения. Въпреки че техните възможности са огромни, работата с вериги с много връзки и етикети е сложна. Въпреки това, Siemens LOGO! PLC предлагат практически инструменти, които помагат за опростяване на процеса. Например, инструментът Cut/Join позволява лесно разделяне и повторно свързване на връзките. Друг пример е текстовият инструмент, който ви позволява да създавате независими от блокове или свързани с блокове етикети. Като овладеете тези инструменти, можете да оптимизирате вашите вериги. Освен това ви помага да рационализирате процесите на автоматизация, повишавайки ефективността и производителността.

2.8.1. Cutting Connections

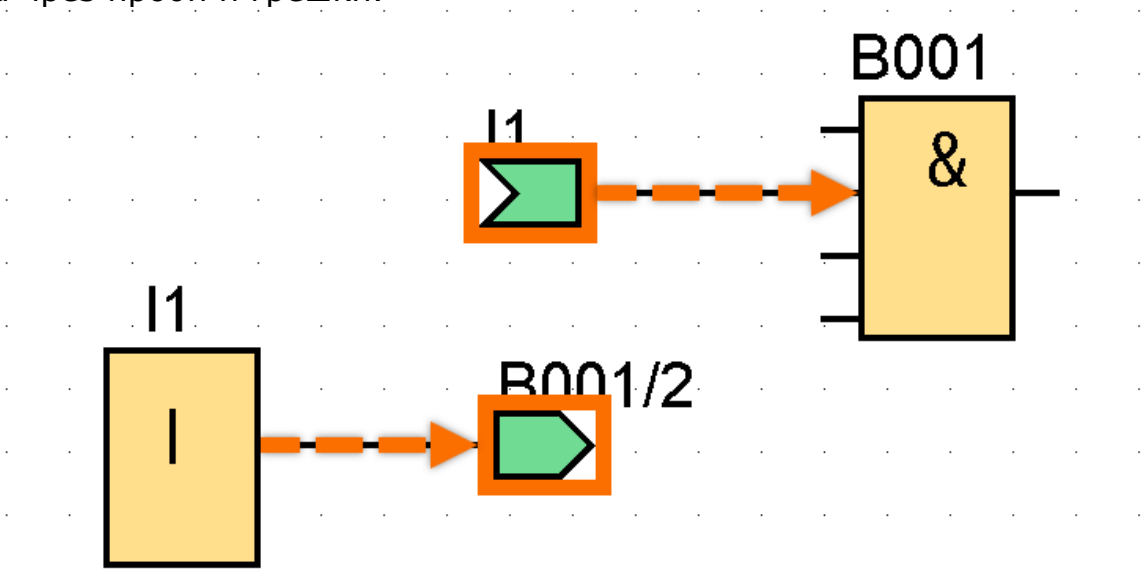
Може да бъде предизвикателство да се разбере оформлението на голяма програма. Може да се забележи, когато включва наличието на множество пресичания на линиите, които могат да бъдат объркващи за следване. Лентата с инструменти за програмиране включва удобен инструмент за изрязване/съединяване, който ви помага да подредите оформлението на връзката си.

След като активирате инструмента Cut/Join, изберете връзка, която искате да разделите, за да разделите свързващата линия визуално. Важно е да се отбележи, че връзката между блоковете ще остане непокътната.



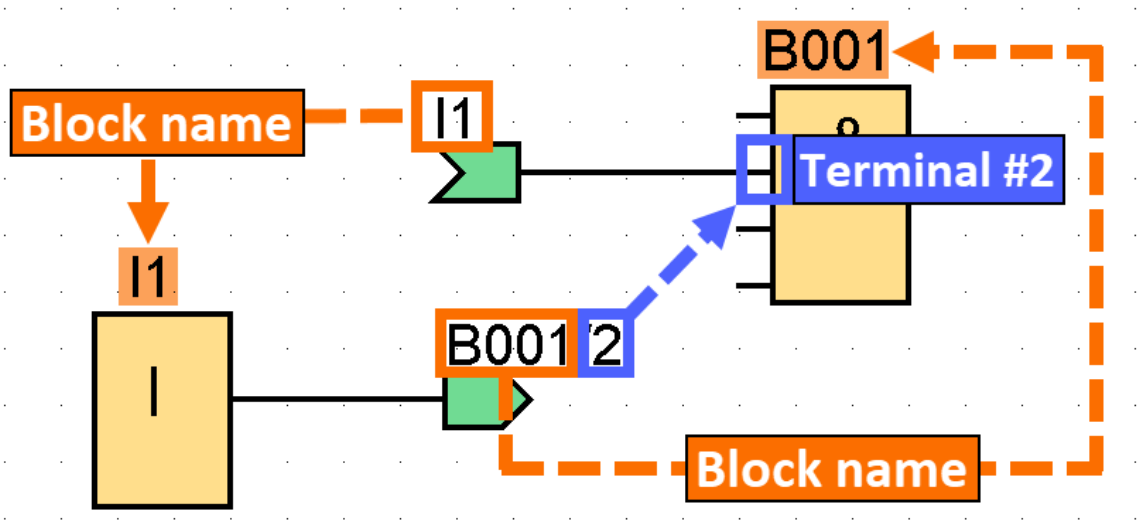
Фиг. 2.39. *Рязане на желаната свързваща линия, като същевременно се запазва връзката между блоковете непокътната*

Иконите със стрелки вече ще се показват в отворените краища на изрязаната връзка. Те служат като визуално представяне на посоката на потока на сигнала. По този начин потребителите не трябва да се обръщат към документацията или да прекарват време в определяне на посоката на потока на сигнала чрез проби и грешки.



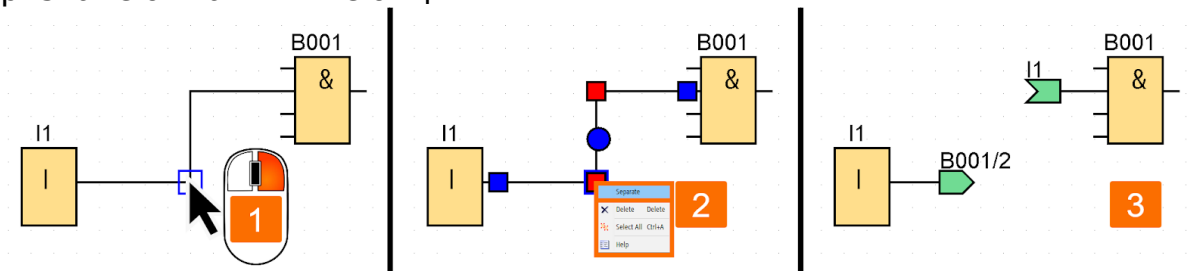
Фиг. 2.40. *Определяне на посоката на потока на сигнала чрез отворени краища на изрязаната връзка*

Препратките, разположени над иконите, показват важни детайли. Той включва номера на страницата на програмата на веригата, името на блока и номера на блок терминала, прикрепен към отворената връзка.



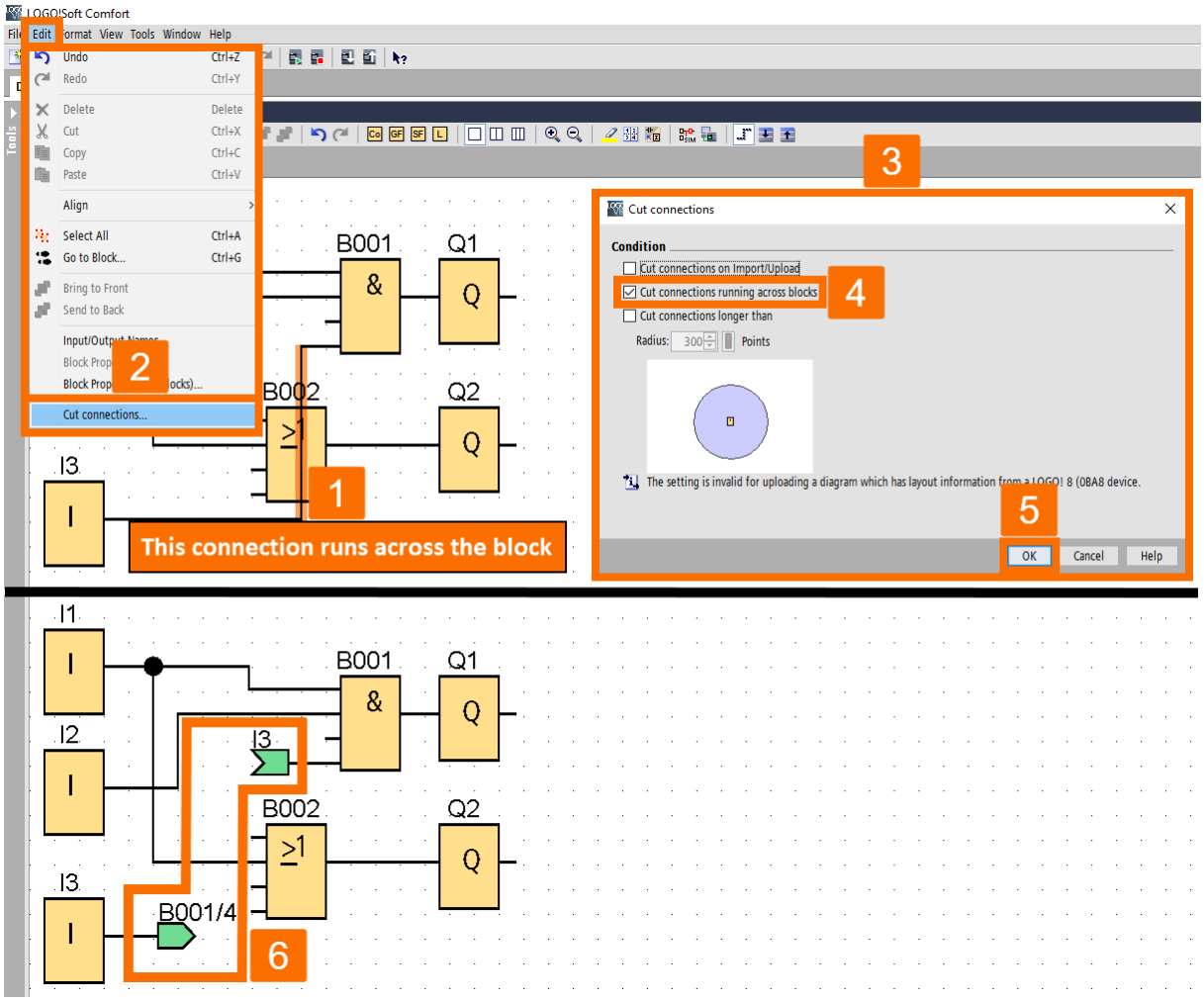
Фиг. 2.41. *Кръстосани препратки на изрязана връзка*

Има и друг начин да разделите линията, свързваща двата блока, които имате предвид. Можете да щракнете с десния бутон върху него и да изберете командата за изрязване от наличните опции.



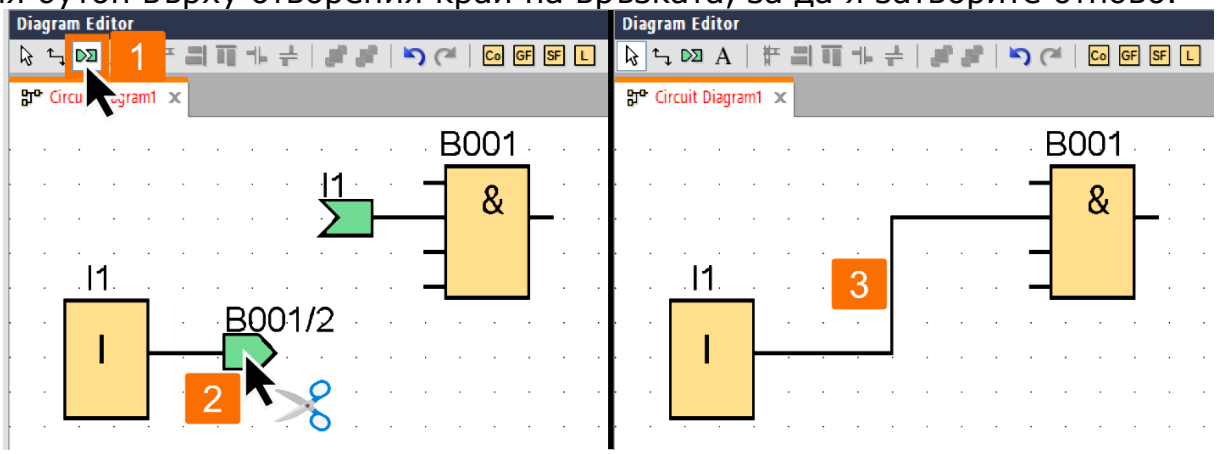
Фиг. 2.42. *Изрязване на желаната линия за свързване чрез менюто за бърз достъп*

Ако искате да изрежете няколко връзки едновременно, използвайте командата Cut Connections от менюто Редактиране. Можете да се уверите, че само връзките, които искате да разделите, са засегнати, като предварително зададете критериите за рязане. Например, можете да конфигурирате рязане на всички връзки, прокарани през блокове.



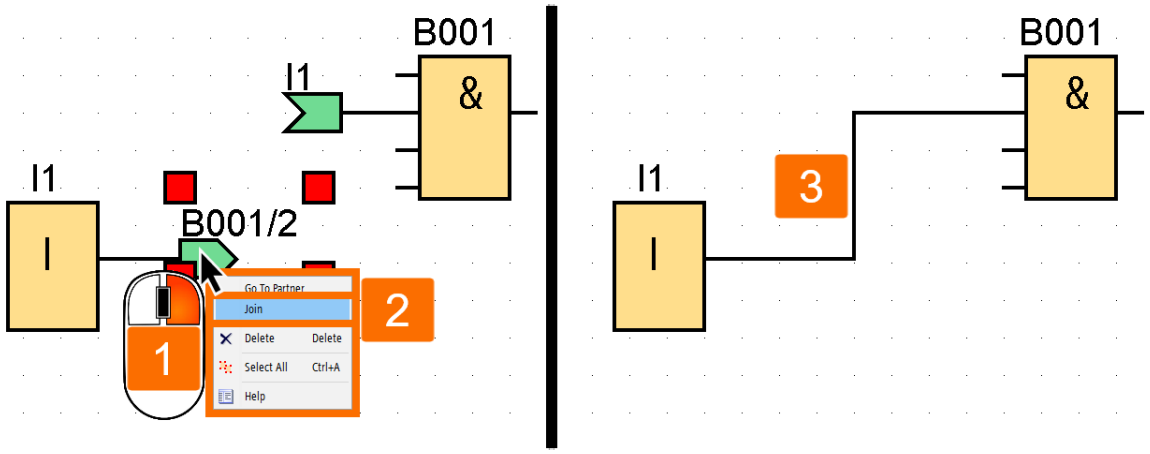
Фиг. 2.43. *Рязане на свързваща линия, прокарана през блок*

Докато Cut/Join инструментът е избран, можете да използвате щракване с левия бутон върху отворения край на връзката, за да я затворите отново.



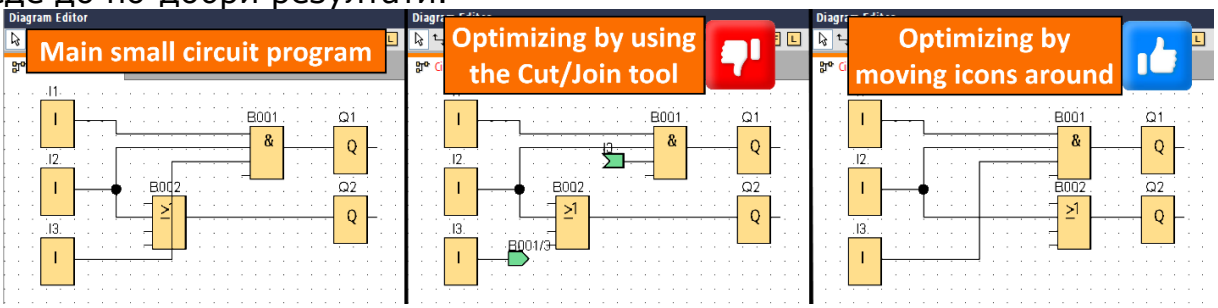
Фиг. 2.44. *Повторно свързване на отделна линия за свързване с помощта на cut/join инструмент.*

Друг начин да затворите връзката си е да щракнете с десния бутон върху един от отворените краища на връзката и да изберете командата Присъединяване от изскачащото меню.



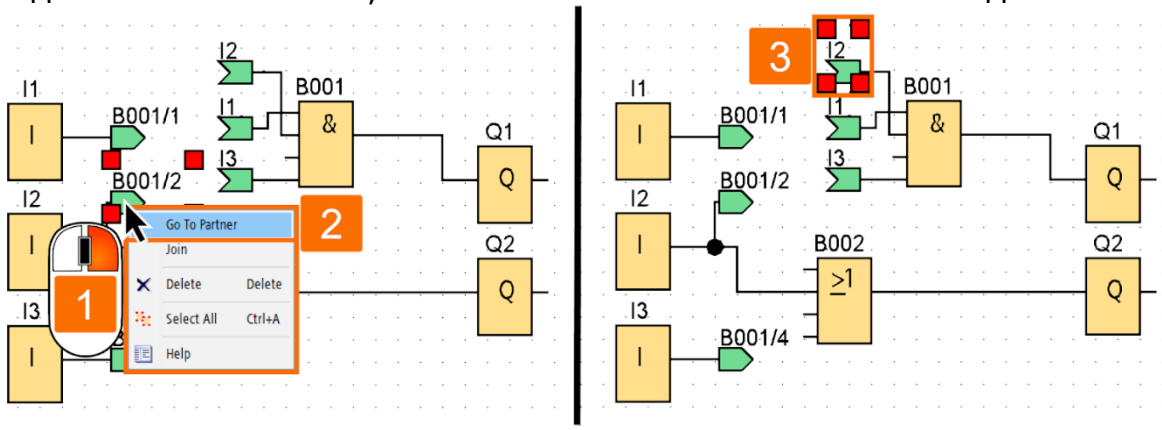
Фиг. 2.45. *Повторно свързване на отделна линия за свързване с помощта на контекстното меню*

Ако работите с по-малка програма, обикновено е по-добре да избягвате използването на този инструмент за редактиране. Вместо това опитайте да оптимизирате оформлението, като преместите иконите, което често може да доведе до по-добри резултати.



Фиг. 2.46. *Оптимизиране на програми за малки вериги чрез преместване на блокове*

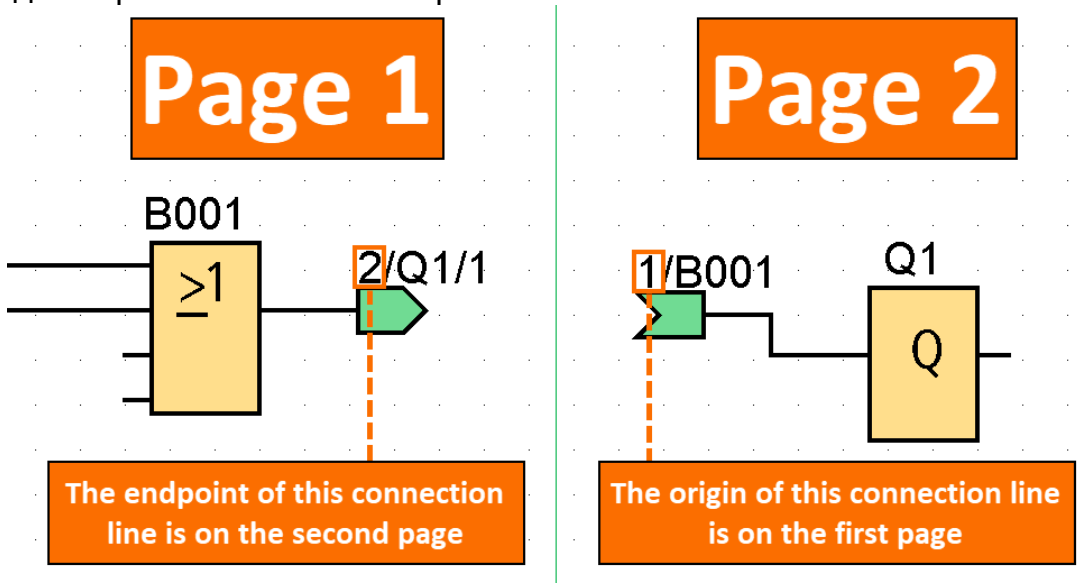
Ако трябва да отидете до партньорския край на прекъсната връзка, можете да го направите бързо, като щракнете с десния бутон върху отворения й край. Той ще изведе контекстно меню, което включва Go to Partner команда за тази цел.



Фиг. 2.47. *Навигиране до партньорския край на изрязана връзка*

Допълнителна полза от Cut/Join инструментът е способността му да обработва вериги, които продължават към множество страници за печат. Той включва тези, които преминават през прекъсване на страница. Линиите, свързващи двата блока на веригата, изобразени на отделни страници, са прекъснати без

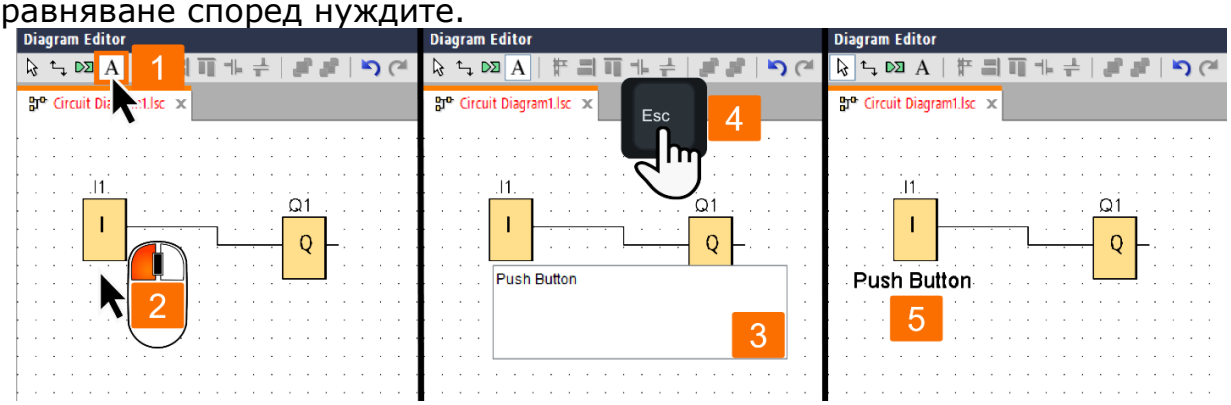
индикация за кръстосана препратка. Въпреки това, с помощта на Cut/Join инструментът за изрязване на тези връзки има предимства. По този начин автоматично се генерира кръстосана препратка, която насочва към източника или целта на връзката. По този начин позволява лесно проследяване на произхода и крайната точка на връзката.



Фиг. 2.48. *Проследяване на произхода и крайната точка на връзката, която е разпределена на две страници*

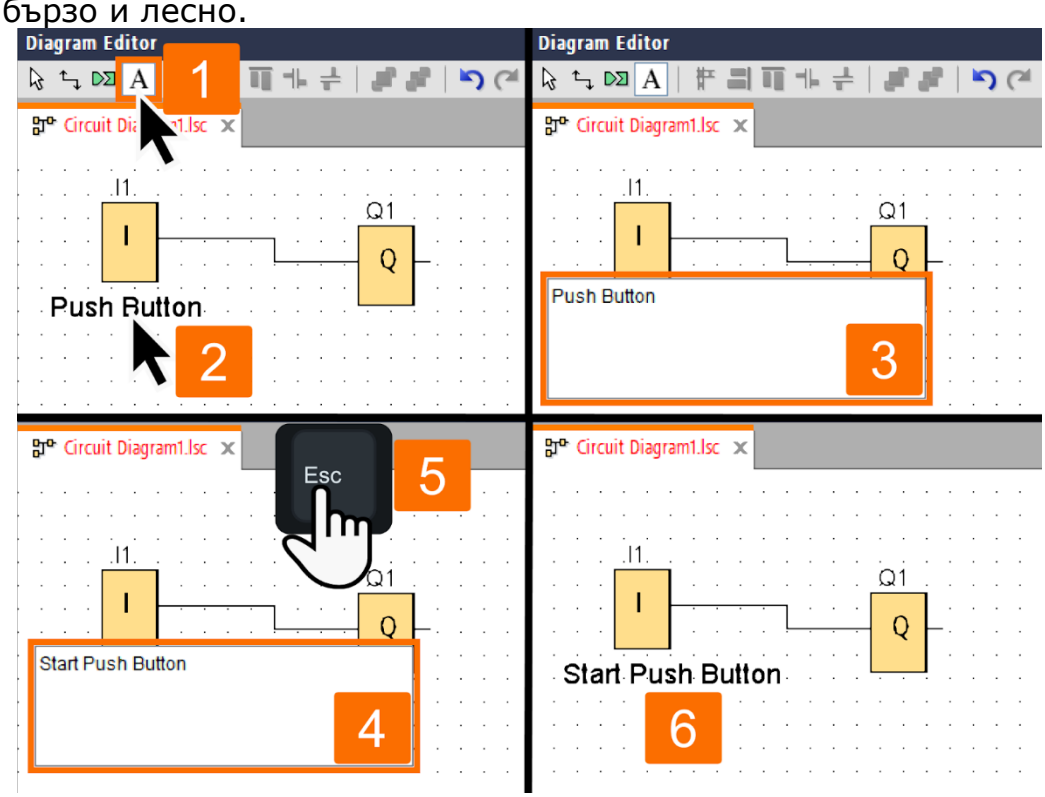
2.8.2. Документация на програмата верига

Ако искате да създадете етикети, които са или без блокове, или свързани с блокове, можете да използвате инструмента за текст в лентата с инструменти за програмиране. За достъп до тази функционалност щракнете върху инструмента за текст. Ако сте активирали тази икона, можете да получите достъп до поле за въвеждане на текст, като щракнете върху незаето място в програмния интерфейс или блок. След като въведете текста на етикета, трябва да излезете от прозореца на етикета. Как? Кликнете върху която и да е част от програмния интерфейс извън прозореца на етикета или просто натиснете клавиша ESC. След като прозорецът бъде затворен, текстът на етикета ще бъде видим на диаграмата и може да се регулира чрез избиране, преместване или подравняване според нуждите.



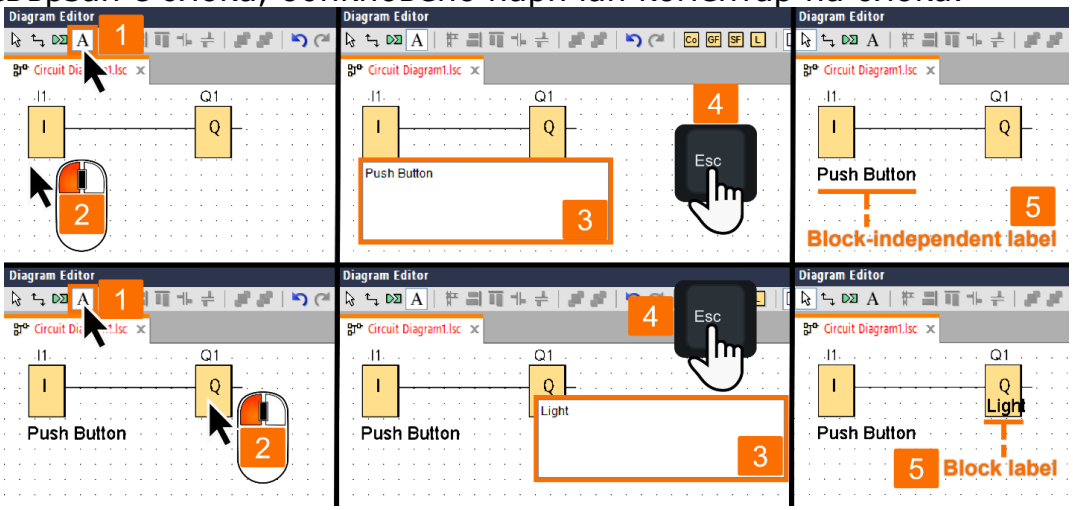
Фиг.2.49. *Създаване на етикет с помощта на инструмента за текст*

За да промените етикет, влезте в текстовия инструмент и щракнете върху етикета, който искате да редактирате. Това действие ще изведе прозореца на етикета за редактиране, което ще ви позволи да направите необходимите промени бързо и лесно.



Фиг. 2.50. *Редактиране на съществуващ етикет*

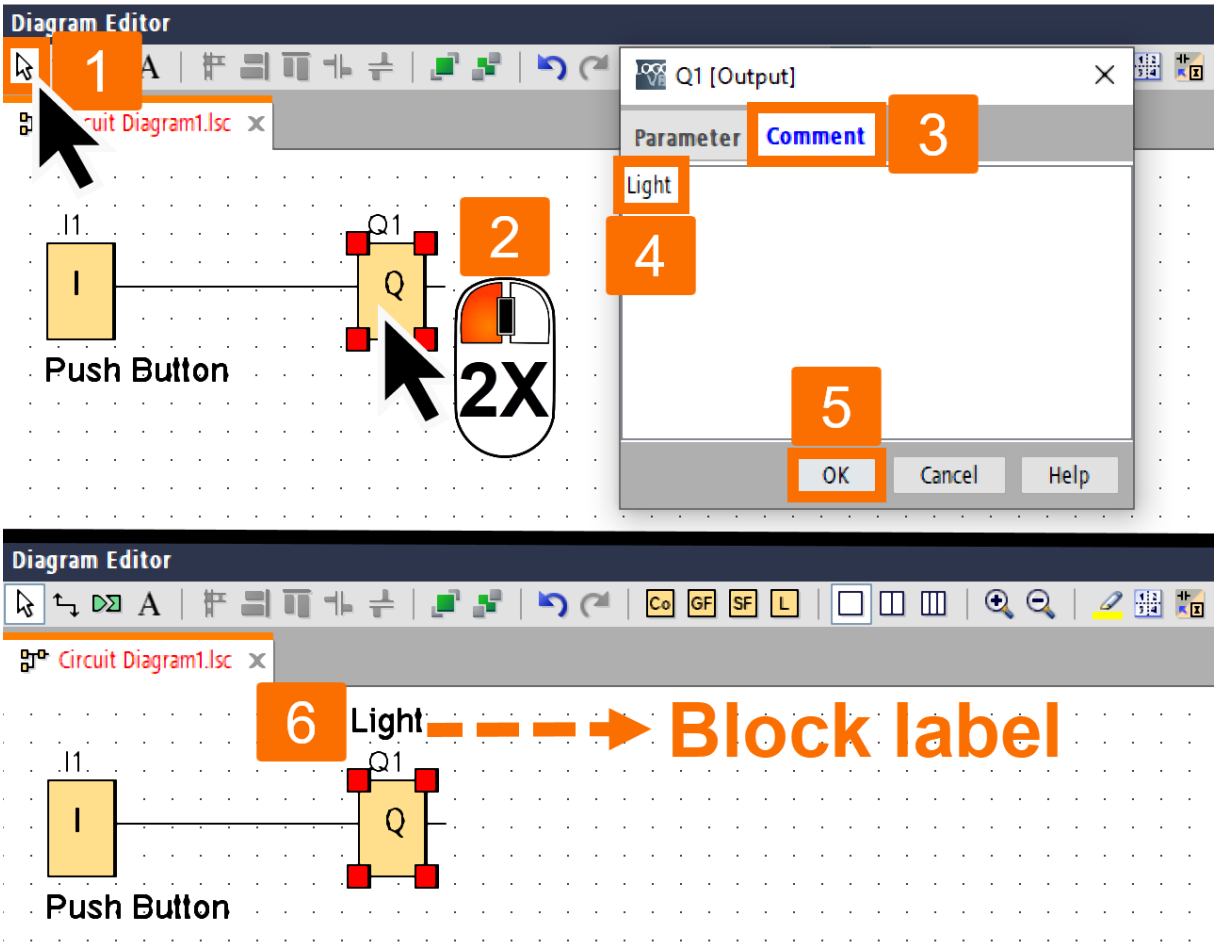
Софтуерът позволява създаване на независими от блокове етикети с помощта на текстовия инструмент и щракване върху незаета зона. По същия начин, като щракнете върху блок с помощта на текстовия инструмент, можете да създадете етикет, свързан с блока, обикновено наричан коментар на блока.



Фиг. 2.51. *Създаване на етикети без блокове и свързани с блокове с помощта на текстовия инструмент*

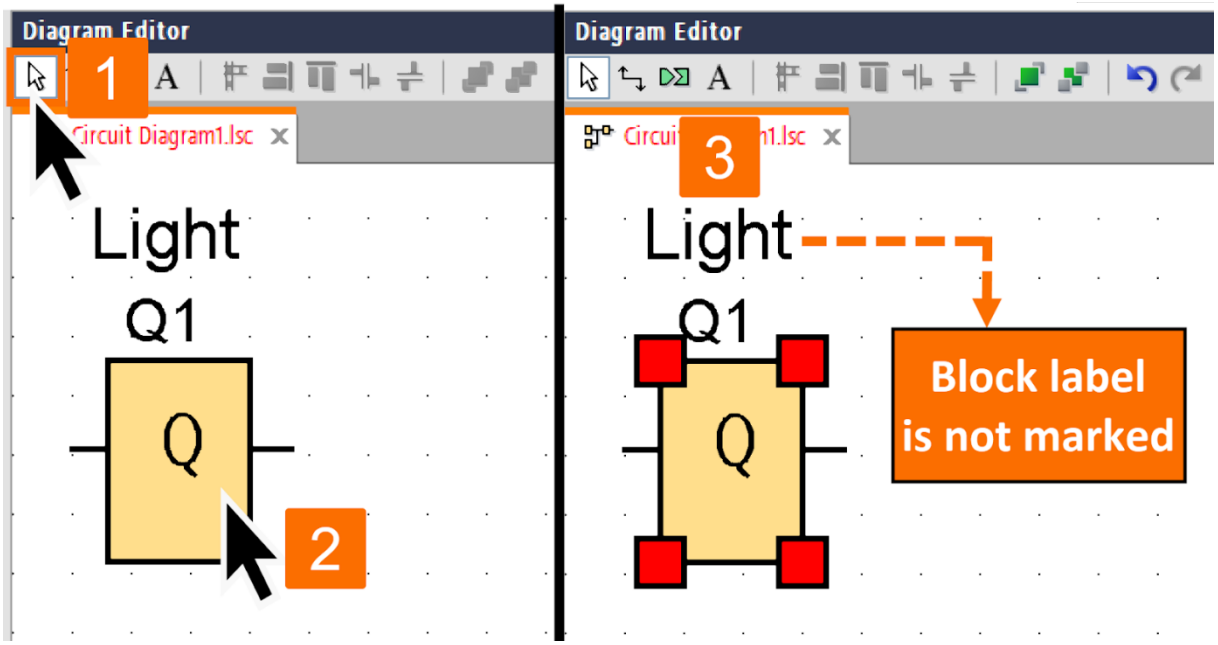
В допълнение към метода, споменат по-рано, има алтернативен подход за създаване на блоков коментар. Можете да щракнете двукратно върху блока,

към който искате да добавите коментар. Отваря се прозорецът със свойства на блока. Оттам можете да отидете до раздела Коментар и да продължите да въвеждате или променяте коментара на блока, ако е необходимо. Когато създавате схема, може да ви е полезно да включите коментари за блокове, които могат да служат като начин за именуване на блокове или описание на техните задачи.



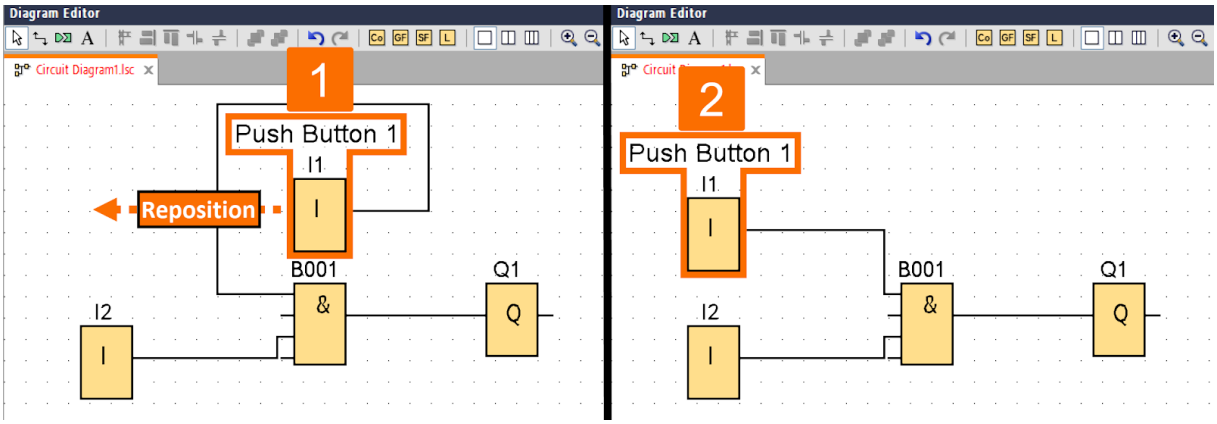
Фиг. 2.52. *Създаване на етикет, свързан с блок, с помощта на раздела за коментари на прозореца със свойства на блока*

Ако изберете блок с прикрепен към него етикет, текстът на етикета няма да бъде маркиран по никакъв начин. Той гарантира, че потребителят може да се съсредоточи върху блока и свързаните с него свойства, без да се разсейва от прикачения етикет.



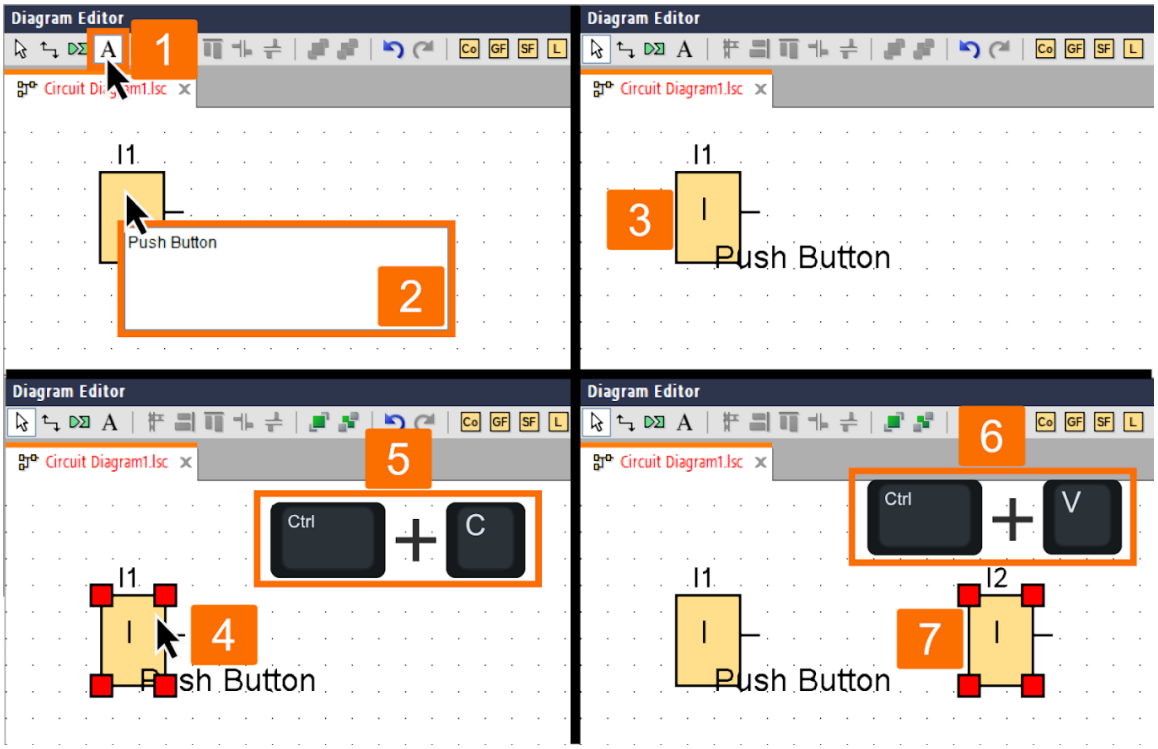
Фиг. 2.53. *Маркирането на свързания етикет не се случва, докато блокът е избран*

Ако преместите блок с присвоен му етикет, етикетът също ще се премести заедно с блока и ще бъде поставен на новата позиция. Той гарантира, че дизайнът остава привлекателен и поддържа ясна визуална връзка между двата елемента.



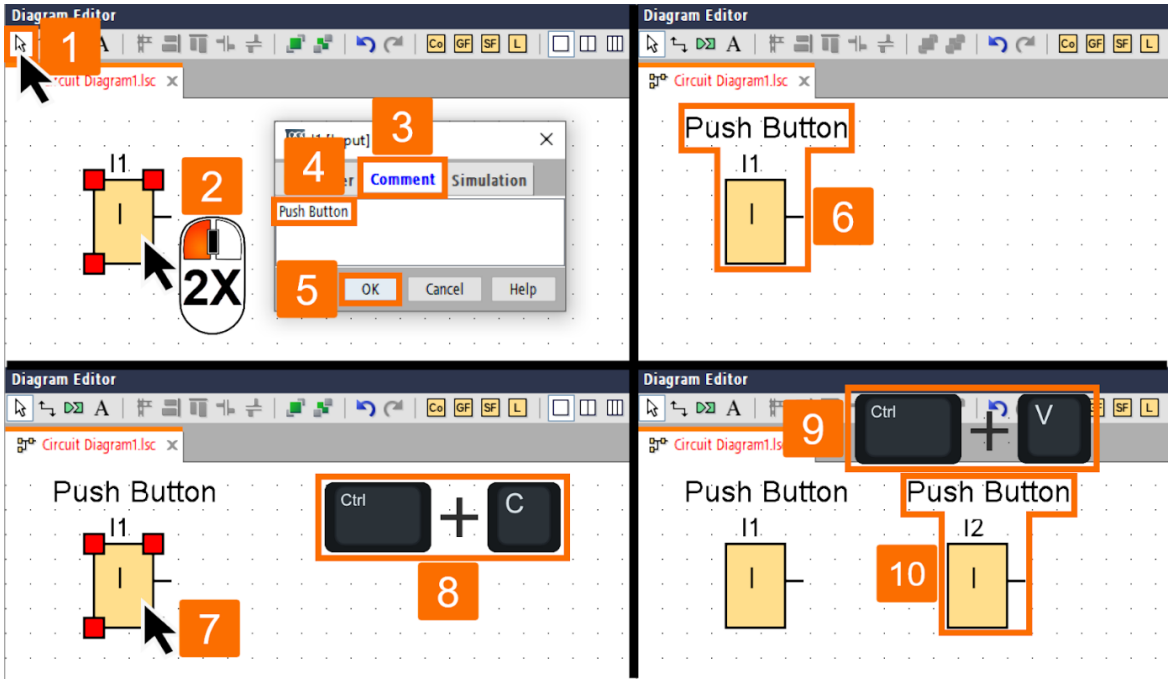
Фиг. 2.54. *Препозициониране на свързан етикет, докато блокът му е преместен*

Когато искате да копирате блок със свързан етикет, от решаващо значение е да помислите как създавате етикета. Има два сценария, които трябва да имате предвид: В първия сценарий етикетът се създава чрез щракване върху блок, докато текстовият инструмент е активиран. Ако копирате блока в този сценарий, само блокът, а не етикетът, ще бъде копиран в клипборда.



Фиг. 2.55. *Копиране на свързан етикет, създаден с помощта на инструмента за текст*

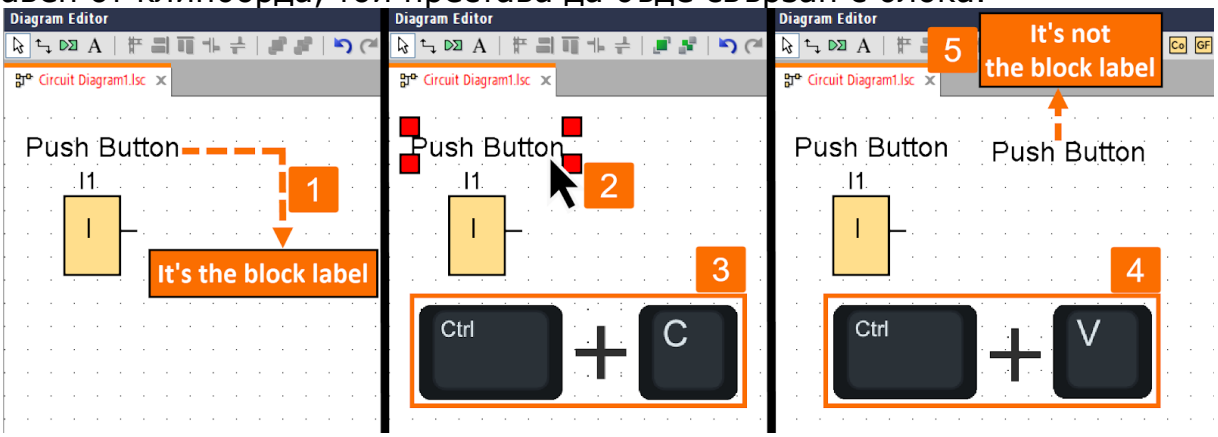
Във втория сценарий етикетът се създава с помощта на раздела за коментари в прозореца на блока свойства. Ако трябва да копирате блока в този сценарий, той ще бъде копиран в клипборда със свързания с него етикет. Когато поставите блока другаде, етикетът също ще бъде включен и ще бъде свързан с блока.



Фиг. 2.56. *Копиране на свързан етикет, създаден с помощта на раздела за*

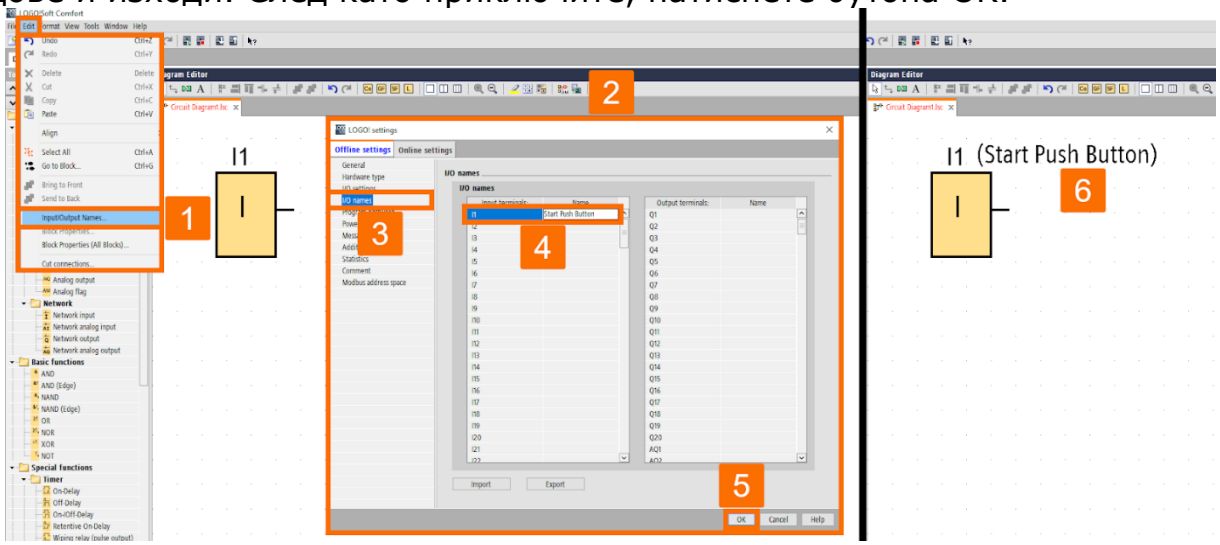
коментари на прозореца със свойства на блока

Изтриването на свързания етикет на блок се случва, когато блокът е изрязан. Свързаният етикет може да бъде избран и да се действа отделно, за да бъде преместен, копиран, изрязан или поставен. След като свързан етикет бъде поставен от клипборда, той престава да бъде свързан с блока.



Фиг. 2.57. *Поставянето на свързан етикет от клипборда води до превръщане в етикет без блокове*

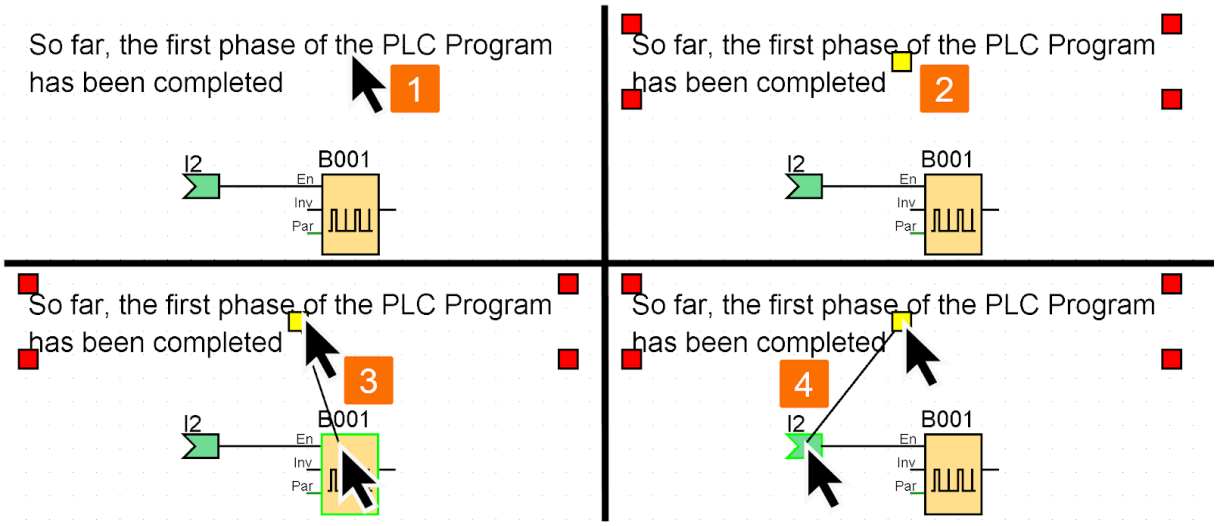
Обозначаването на номера на блокове и имена на конектори за входове и изходи може да се извърши наведнъж. За да направите това, отворете Edit в горната лента с инструменти и изберете Input/Output Names. След това в изскачащия прозорец изберете I/O names и задайте имена на желаните входове и изходи. След като приключите, натиснете бутона OK.



Фиг. 2.58. *Присвояване на имена на конектори за входове и изходи*

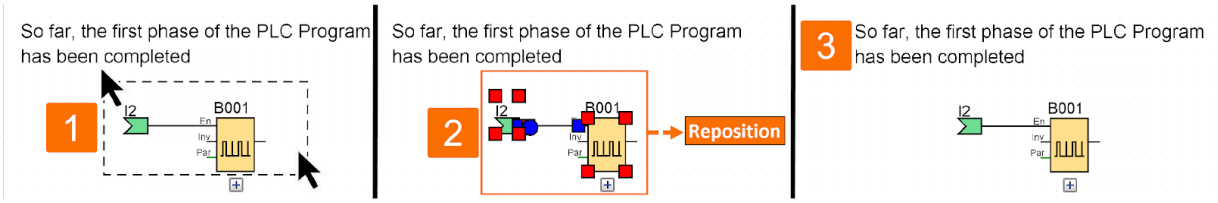
2.8.3. Свързване на коментари

Текстовите коментари имат възможност да бъдат свързани или с функционални блокове, или с изрязани конектори. Когато сте избрали текста, който искате да свържете с функционален блок, щракнете върху жълтия куб в средата. След това, докато държите левия бутон на мишката, преместете показалеца към функционалния блок, с който искате да го свържете.



Фиг. 2.59. *Свързване на коментар към функционален блок и изрязан конектор*

Свързаните коментари се синхронизират с движението на фигурата, към която са свързани. Можете да промените позицията на коментара спрямо фигурата, като използвате същата техника, както бихте направили за блоков коментар.

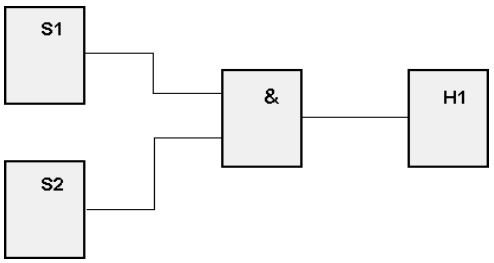


Фиг. 2.60. *Синхронизиране на движението на свързан коментар със свързаната с него фигура*

В заключение научихте как да почистите оформлението на връзката с помощта на инструмента Cut/Join. Научихте как да разделите желаната линия на свързване, като същевременно поддържате връзката между блоковете активна. Освен това разбрахте какъв вид информация може да бъде достъпна чрез кръстосани препратки на отрязани връзки.
Запознахте се със създаването на етикети, свързани с блокове и без блокове, с помощта на текстовия инструмент. Разбрахте как да извиквате и редактирате съществуващи етикети, за да оптимизирате документацията на програмата на веригата. Научихте, че в случай на копиране на блокове трябва да се вземе предвид методът, по който е създаден етикетът на блока.
Накрая се запознахте с свързването на текстови коментари към функционални блокове или изрязани конектори. По този начин коментарите, прикрепени към определена фигура, се движат в тандем с движението на тази фигура.

PLC

Програмиране с функционални блокове



Програмиране с функционални блокове

Цели

След успешно завършване на този модул студентите трябва да могат:

1. Да получат обща представа за основни концепции и функции на PLC в технологията за управление.
2. Да демонстрират как работят различни логически функции, като И, ИЛИ и НЕ.
3. Да представят операции на логическа функция в таблици за истинност.
4. Да използват функциите за тайминг и броене в програмите.
5. Да се запознаят с най-важните функции на LOGO! Soft.
6. Да създават и тестват програми за управление с помощта на функционални блокове.
7. Да демонстрират как се прехвърлят програми от компютъра към ЛОГО Контролера.

СЪДЪРЖАНИЕ НА МОДУЛА:

3.1 PLC

Програмируемият логически контролер е устройство, което може да бъде програмирано да изпълнява контролни функции. Тъй като е цифрово устройство, то съхранява информация под формата на условия за включване/изключване, наречени бинарни или битове. Въпреки че PLC използва както цифрови, така и аналогови сигнали, процесорът може да разбира само цифрови сигнали.

Фиг. 3.1 показва връзката между логическото ниво и състоянието на превключвателя. Logic 1 показва, че има сигнал и превключвателят **е включен**. Logic 0 показва, че сигналът отсъства или превключвателят **е изключен**.

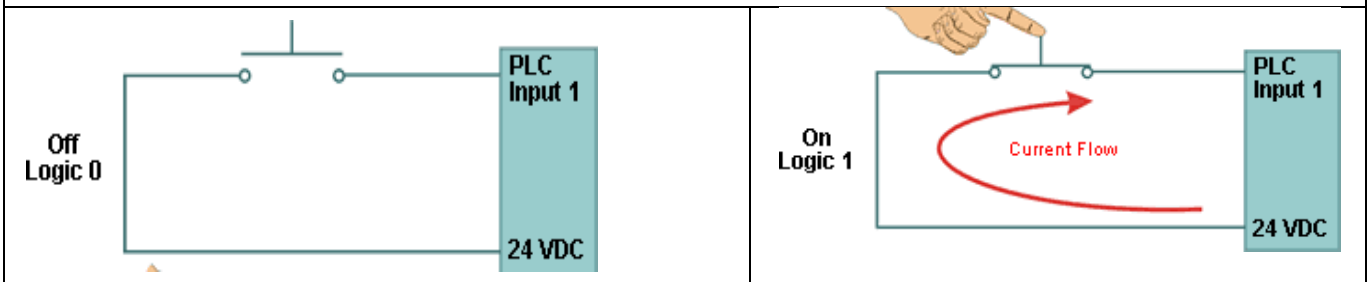


Fig 3.1: Демонстрация на бутон за нормално отваряне

В горния пример се използва нормално отворен (NO) бутон. Когато превключвателят не е натиснат, няма напрежение на PLC входа 1 и го настройва в състояние OFF. Когато превключвателят е натиснат, 24V dc се подава към PLC входа и го настройва в състояние ON. Нормално затвореният бутон (NC) действа срещу нормално отворения (NO) бутон. Фиг. 3.2 показва символите на бутоните.

Фиг. 3.2: Символи с бутони

Основни функции за управление

Основните контролни функции включват следното:

A. Логически функции

B. Функции на паметта

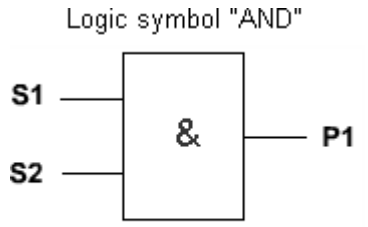
C. Функции за синхронизиране &

D. Функции за броене.

A. Логически функции

PLC системите изпълняват логически функции. Следователно разбирането на логическите функции на двоичните числа е важно за PLC програмирането. Основните логически функции включват AND, OR, NOT, NAND, NOR & EXOR.

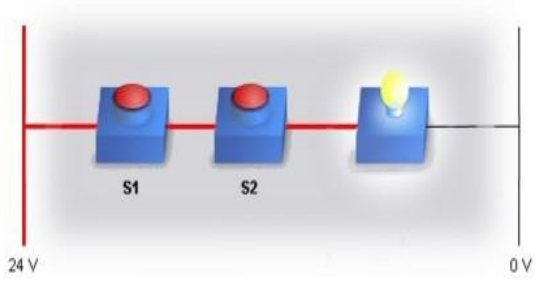
AND Функция: Функцията И има два или повече входа и само един изход. На изхода имаме логическа единица само когато всичките му входове са логически единици.



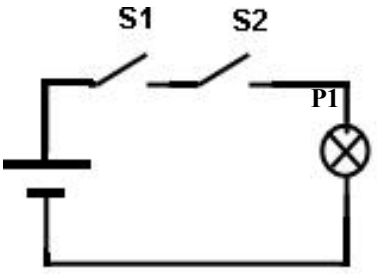
Булев израз: P1=S1.S2

Таблица на истинност

S1	S2	P1
0	0	0
0	1	0
1	0	0
1	1	1

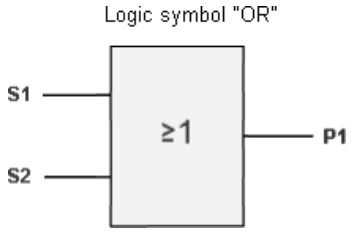


Лампа P1 е включена, само ако S1 и S2 са затворени или включени.



Фиг.3.3: Превключвателен еквивалент на функцията AND

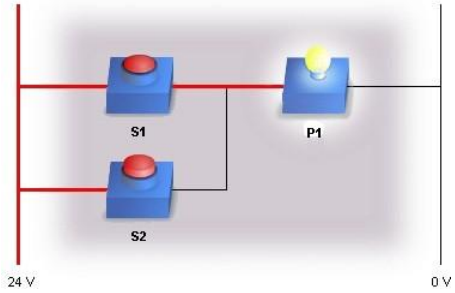
OR Функция: Функцията OR има два или повече входа и само един изход. На изхода имаме логическа единица само ако един или повече от входовете са логически единици.

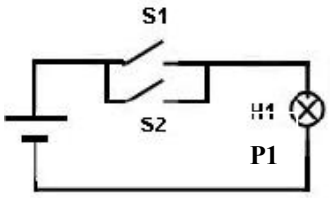


Булев израз: P1=S1+S2

A	B	P1
0	0	0
0	1	1
1	0	1
1	1	1

Таблица на истинност

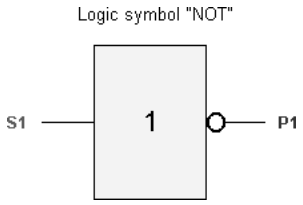




Лампата е включена само ако някое от двете S1 или S2, или и двете са включени.

Фиг. 3.4: Превключвателен еквивалент на функцията ИЛИ

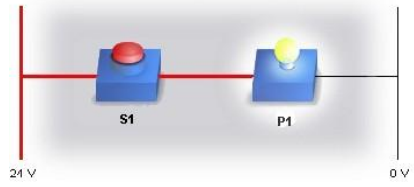
3. **NOT Функция:** Функцията NOT има само един вход и един изход. Тя произвежда изход, който е противоположен на входа.

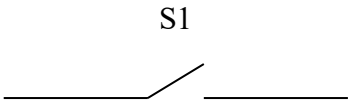


Boolean Expression: $\overline{P1}$=S1

Truth Table

A	P1
0	1
1	0





Фиг. 3.5: Превключвателен еквивалент на функцията NOT

Функция на паметта: Функцията памет се използва за съхраняване на сигнал. Той има входове SET (S) и RESET (R), както е показано на фиг. 3.6. При лог1 на S вход- задава (съхранява 1) функцията на паметта, а при лог.1 на R, входът нулира функцията памет, както е показано на фиг.3.7

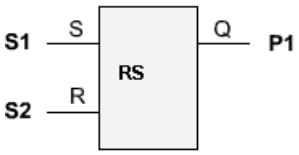


Fig 3.6: Функционален блок на паметта

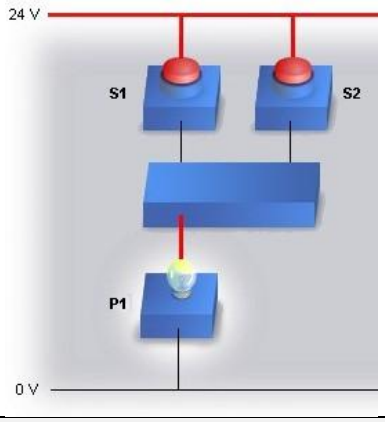
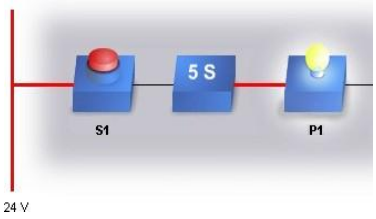
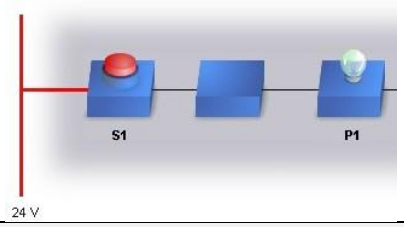


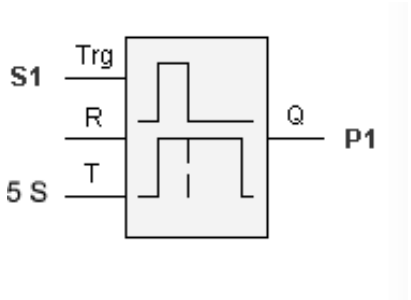
Fig 3.7: Функция за памет SET

Функция за време: Тази функция се използва за осигуряване на забавяне във времето във веригата. В примера, показан на фиг. 3.8, за <u>забавяне на времето за изключване</u> се използва забавяне на изключването. Лампата P1 ще изгасне след 5 секунди.





Фиг. 3.8: Демонстрация на функцията за време



Фиг. 3.9: Функционален блок за време

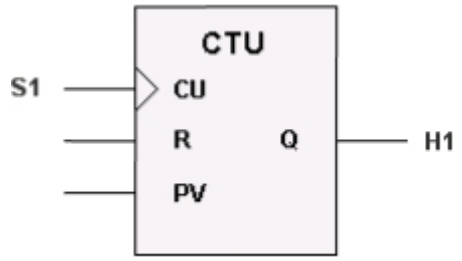
Фиг. 3.9 показва функционалния блок за синхронизиране. Входът T се използва за задаване на часа. Изходът Q ще бъде лог.1 за времето, зададено на входа T.

Има различни видове таймери; Някои примери са дадени по-долу:

1. OFF Таймер за забавяне: Използва се, когато има нужда от спиране на действие след определено време. Вижте примера на фиг. 3.8.
2. ON Таймер за забавяне: Използва се за стартиране на действие след определено време. Например, някои машини не изискват работа веднага при включване, а след определено време, или за безопасност, или за смазване.

Функция за броене

Функциите за броене се използват за откриване на броя на елементите и събитията. Например, може да се наложи брояч за преброяване на точно 10 еднакви части, които трябва да бъдат подадени към конвейер на сортировъчната система.



Фиг. 3.11:Функционален блок Брояч

Фиг. 3.11 показва функционалния блок на брояча. PV е предварително зададената стойност. Стойността на брояча се увеличава с 1 за всеки входен сигнал при CU. Когато се достигне предварително зададената стойност, изходът Q става "1".

3.2 Методи за програмиране

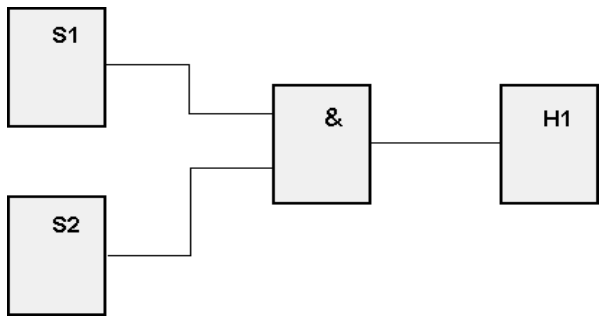
Програмата е набор от инструкции за изпълнение на задача. Следните методи могат да се използват за PLC програмиране:

- Function Block Diagram
- Ladder Logic
- Statement Lists

В този модул ще се използват функционални блокове за разработване на вериги.

Function Block Diagram

Блокове в LOGO! е функция, която преобразува входната информация в изходна. Фиг. 3.13 показва пример за функционална блокова схема. Тук S1 и S2 са клемите, които са свързани към функционалния блок И. Изходът H1 ще бъде включен само ако превключвателите S1 и S2 са включени.



Фиг. 3.13: Илюстрация на функционална блокова диаграма

LOGO! предоставя различни елементи в режим на програмиране.

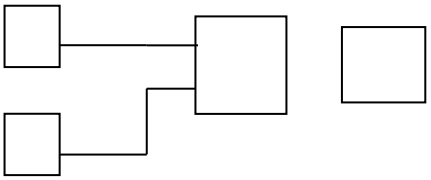
- Co: Списък с конектори
- GF: Списък на основните функции И, ИЛИ и т.н.
- SF: Списък на специалните функции

Логическите функционални блокове са налични в **GF** (General Function) и терминалите могат да бъдат избрани, като изберете функцията **Co**. **Co** е кръстен на термина Connector (терминал).

Умение 1: Създаване на функционални блокови диаграми от задачи с твърдения.

Следвайте пример А и начертайте FBD за логическата операция в пример В:

А. Стълбищното осветление се управлява от два превключвателя (А и В). Светлината трябва да се включи, ако някой от превключвателите или и двата превключвателя са включени. Начертайте функционална блокова диаграма, за да реализирате тази логическа операция.



Б. Проектирайте проста схема за управление на алармата на къщата.

- Къщата има две врати D1 и D2 и два прозореца W1 и W2.

- Ако някоя от вратите ИЛИ някой от прозорците бъде отворен от крадеца, трябва да прозвучи алармата. Начертайте функционална блокова диаграма за тази логическа операция.

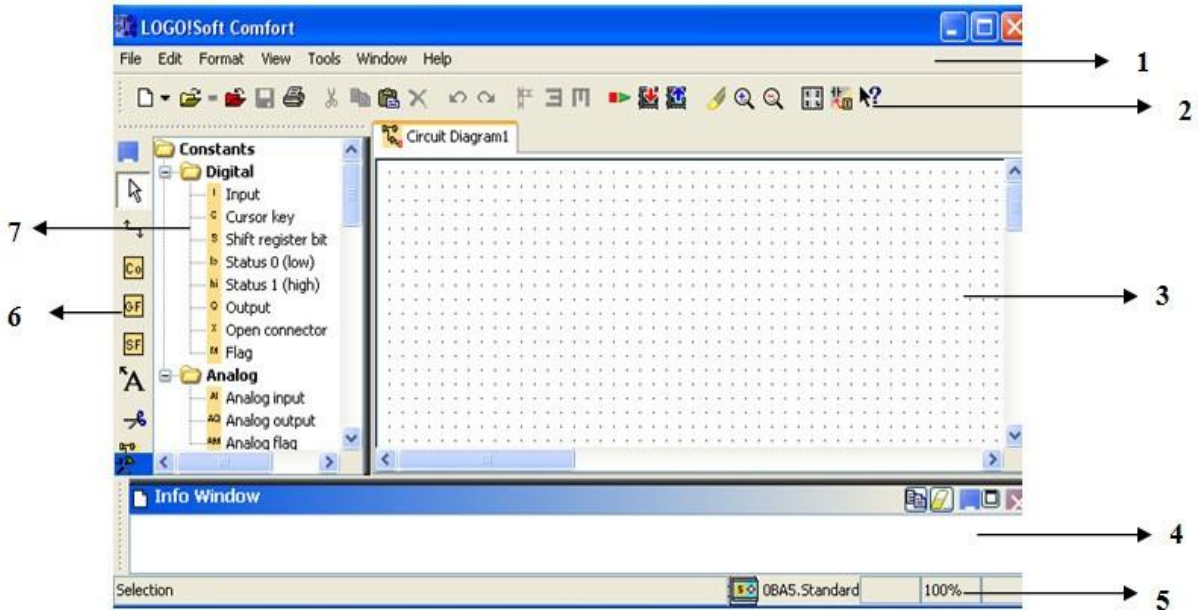
Умение 2: За създаване на функционални блокови диаграми от логически изрази.
Начертайте FBD за следните логически изрази:
A. $Q1 = I1 + \overline{I2} + I3.I4$
B. $Q2 = I1.I2 + I3.\overline{I4}$

3.3 LOGO!Soft Преглед

ЛОГО! Софт е софтуер за програмиране на Сименс лого контролери! С използване на LOGO! Soft, можете да създавате контролни програми и да ги прехвърляте в LOGO!

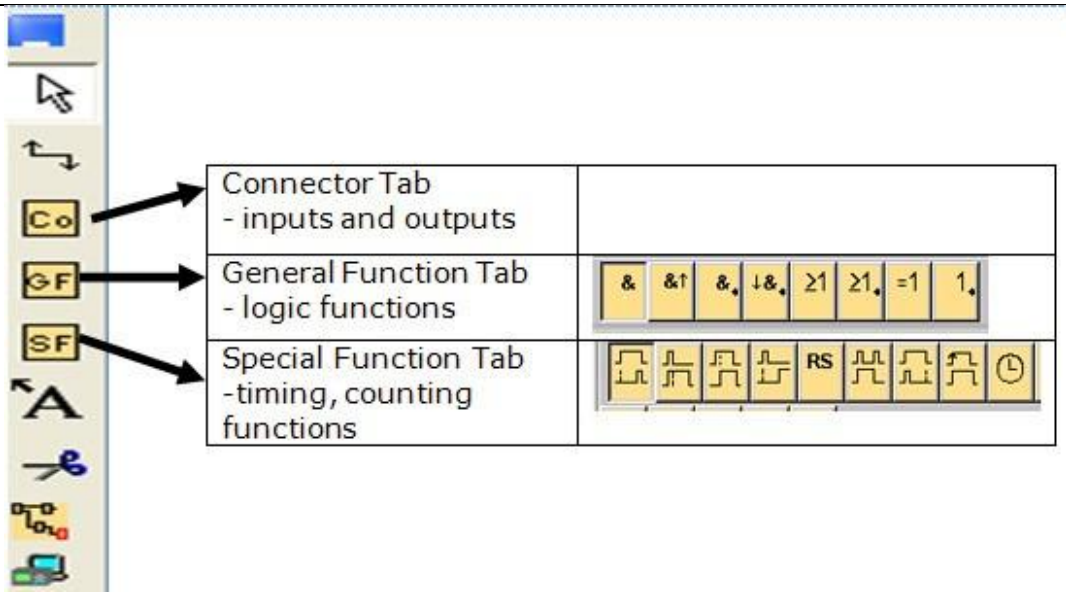
LOGO! Soft Потребителският интерфейс има следните функционални области, както е показано на фиг. 3.13 по-долу:

1. Menu bar
2. Standard tool bar
3. Programming Interface
4. Info box
5. Status Bar
6. Toolbox
7. Constants, connectors, functions



Фиг. 3.14: LOGO!Soft Потребителски интерфейс

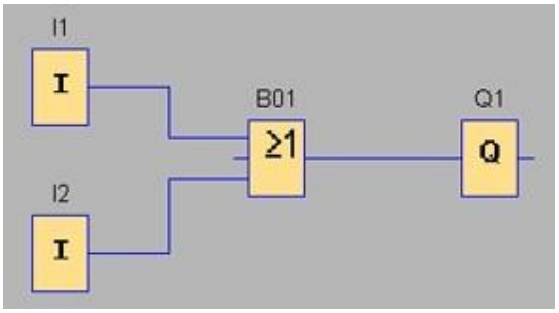
Co, GF and SF раздели (фиг. 3.15) може да се използва за избор на терминали и функционални блокове.



Фиг. 3.15: Раздели на кутията с инструменти

Създаване на програма за управление с помощта на функционални блокове

Следващата програма демонстрира функцията OR. Входните и изходните клеми и функцията OR се използват за създаване на функционалната блокова диаграма, както е показано на фиг. 3.16.



Фиг. 3.16: Функционалната блокова диаграма

При свързването на блоковете трябва да се спазват следните правила:

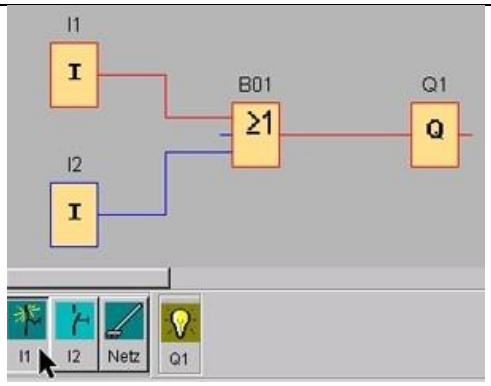
- Може да се осъществи връзка между един блоков изход и един блоков вход.

- Изходът може да бъде свързан към няколко входа.

- Един вход не може да бъде свързан към няколко изхода.

Симулация на програма

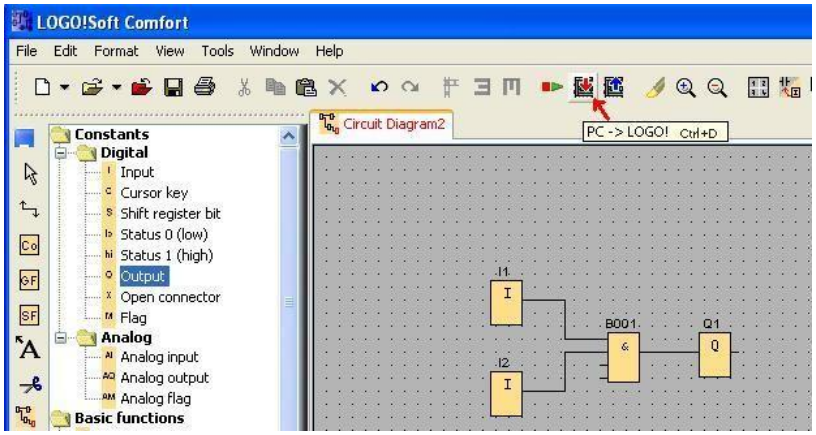
Програмата може да бъде тествана с помощта на симулация. Програмата се тества чрез определяне на входните сигнали и наблюдение на желания изходен сигнал, както е показано на фиг. 3.17. Червените линии носят сигнал 1, а сините линии носят сигнал 0.



Фиг. 3.17: Демонстрация на симулация на програма

Прехвърляне на програмата към ЛОГО!

Програмата може да бъде изтеглена от компютъра на LOGO! и след това тестван.



Фиг. 3.18: Прехвърляне на програмата към ЛОГО!

Стартиране на програмата на ЛОГО!

- ESC key
- Select "START"
- OK key

Лабораторно упражнение 1

Цел: Да се запознаете с логическите функционални блокове

AND Функционален блок

Процедура: Програмирайте PLC за следната операция:

- Обект трябва да се движи напред , когато превключвателят за избор е включен.

- Обект трябва да се движи назад при натискане на бутон1 и бутон2 .

Описание на логиката:

Условия:

- **Обект се движи напред**, когато **Selector Switch** е включен.
- **Обект се движи назад**, когато **бутон1 и бутон 2** са натиснати едновременно.

Стъпки в LOGO! Soft Comfort:

1. Входове:

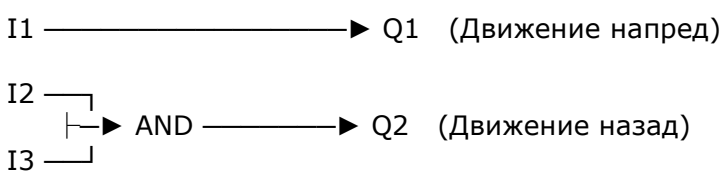
- I1 → Selector Switch
- I2 → бутон 1
- I3 → бутон 2

2. Изходи:

- Q1 → Движение напред
- Q2 → Движение назад

Блокова схема:

- **Движение напред:**
- Свържи I1 директно към Q1
- **Движение назад:**
- Използвай **AND блок**, който приема I2 и I3
- Изходът на AND блока свържи към Q2



Лабораторно упражнение 2

Цел: За да се запознаете с функционалните блокове за време и броене.

Процедура: Програмирайте PLC за следната операция:

- Вентилатор трябва да стартира след натискане на бутон 1 и трябва да остане включен.
- 5 секунди след натискане на бутон 1, трябва да светне лампа.

Стъпка 1:

Използвайте вградените клавиши на ЛОГОТО! Контролен блок за програмиране на PLC. Стартирайте програмата и наблюдавайте резултата. Начертайте FBD в полето по-долу:

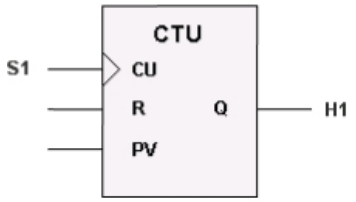
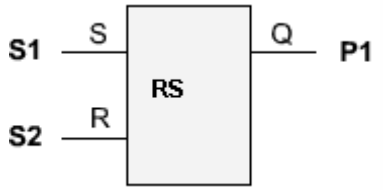
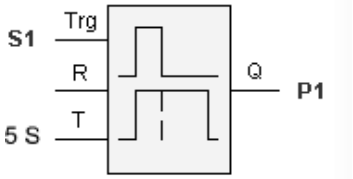
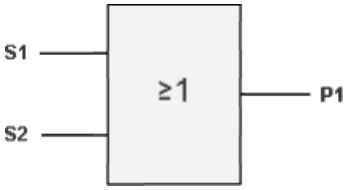
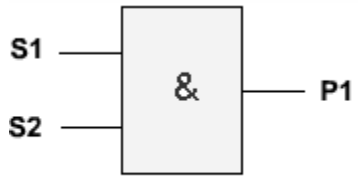
Функционална блокова диаграма:

Стъпка 2:

Използвайте ЛОГО софтуер за създаване на FBD. Изчистете предишната програма на PLC. Изтеглете програмата от компютъра на PLC, стартирайте и наблюдавайте резултата.

Лабораторно упражнение 3

1. Съчетайте следните задачи с правилните функционални блокове:

	Задача		Функционални блокове
1	Включете машината, когато PB е натиснат и вратата на машината е затворена		
2	Пребройте броя на произведените кутии Cola		
3	Изключете машината след определено време		
4	Включете и изключете двигател, като използвате същия функционален блок		
5	Включете лампа, като използвате PB1 или PB2 или и двете		

Лабораторно упражнение 4

За следните задачи *напишете правилните изрази* и *начертайте функционалните блокови диаграми*:

a) Крушка (B) се включва при натискане на превключвател (S1) и се изключва при натискане на (S2)

b) Лампичка (L1) се включва при натискане на превключвател (S1) и превключвател (S2) НЕ е натиснат (освободен)

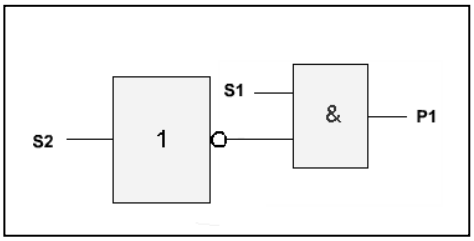
c) Зелена светлина (Q1) ще светне, когато бутонът (I1) и бутонът (I2) са натиснати или бутонът (I3) е натиснат.

d) Машина (Q1) стартира 20 секунди след натискане на превключвателя S1

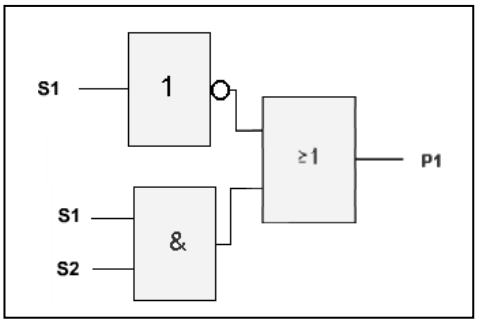
Лабораторно упражнение 5

За следните FBD напишете правилния израз:

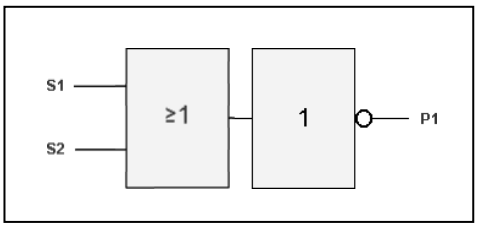
a)



b)



c)



Литература

[1] F. D. Petruzella, *Programmable Logic Controllers*, 5th ed. New York, NY, USA: McGraw-Hill, 2016, ISBN 978-0-07-337384-3.
[2] W. Bolton, *Programmable Logic Controllers*, 6th ed. Oxford, U.K.: Newnes/Elsevier, 2015, ISBN 978-0-08-100622-1.
[3] J. W. Webb and R. A. Reis, *Programmable Logic Controllers: Principles and Applications*, 5th ed. Upper Saddle River, NJ, USA: Prentice Hall, 2003, ISBN 978-0-13-028598-0.
[4] J. R. Hackworth and F. D. Hackworth, *Programmable Logic Controllers: Programming Methods and Applications*. Upper Saddle River, NJ, USA: Pearson/Prentice Hall, 2004, ISBN 978-0-13-060718-8.
[5] L. A. Bryan and E. A. Bryan, *Programmable Controllers: Theory and Implementation*, 2nd ed. Atlanta, GA, USA: Industrial Text Co., 1997, ISBN 978-0-9612084-7-2.
[6] D. H. Hanssen, *Programmable Logic Controllers: A Practical Approach to IEC 61131-3 Using CoDeSys*. Chichester, U.K.: John Wiley & Sons, 2015, ISBN 978-1-119-97134-9.
[7] J. A. Rehg and G. J. Sartori, *Programmable Logic Controllers*, 2nd ed. Upper Saddle River, NJ, USA: Pearson/Prentice Hall, 2009, ISBN 978-0-13-504881-8.
[8] K. Kamel and E. Kamel, *Programmable Logic Controllers: Industrial Control*. New York, NY, USA: McGraw-Hill, 2013, ISBN 978-0-07-181647-2.
[9] Graune, U., Thielert, M., & Wenzl, L. (2009). LOGO! Practical Training. Publicis Publishing, Erlangen. ISBN: 978-3-89578-338-8.
[10] LOGO! 8A Practical Introduction, with Circuit Solutionsand Example ProgramsAuthor: Stefan KrusePublished by: PublicisISBN: 9783895789267
[11] LOGO! Software

Съдържание