ЮЛКА П. ПЕТКОВА

РЪКОВОДСТВО ЗА ЛАБОРАТОРНИ УПРАЖНЕНИЯ

ПО

СПЕЦИАЛИЗИРАНИ КОМПЮТЪРНИ СИСТЕМИ

Ръководството за лабораторни упражнения по дисциплината „Специализирани компютърни системи” е предназначено основно за студентите от ОКС „Бакалавър”, специалност КСТ, изучаващи дисциплината „Специализирани компютърни системи”. То може да се използва и от студентите от ОКС „Магистър”, специалност КСТ, специализация „Компютърни мрежи и комуникации”, както и от всички, които се интересуват и желаят да получат първоначални познания по програмируеми логически контролери.

В учебното пособие са разгледани: програмируемият контролер от серията Twido TWD20Dxxxxx; лабораторният макет, върху който се извършват лабораторните упражнения, както и програмната среда TwidoSuite.

Лабораторните упражнения са представени в осем теми. Структурата на представяне е една и съща и включва: описание на някакъв функционален блок, на определен вид входове и изходи, както и на различни видове променливи, следва задаване на конкретна задача за решаване, последвана от анализ и разпределение на необходимите ресурси на контролера. В първите няколко упражнения са изяснение всички етапи на проектирането на системи за автоматизиране на управлението на определени обекти, а в последните се акцентира само върху особеностите, които трябва да се имат предвид при решаването на поставените задачи. Към отделните теми са добавени и допълнителни въпроси и задачи.

Извън конкретните теми за лабораторни упражнения са добавени и допълнителни задачи за самостоятелна работа.

Доц. д-р Юлка Петкова
Катедра „Компютърни науки и технологии”
Технически университет - Варна

Програмируеми логически контролери (ПЛК). ПЛК от серията TWIDO

1. Въведение

Развитието на съвременната индустрия поставя нови предизвикателства пред проектантите на специализираните компютърни системи. Все по-големи стават изискванията за автоматизация на производствените процеси. Поради тази причина употребата на програмируеми логически контролери (Programmable Logic Controllers - PLC) става все по–необходима и все по-масова.

Програмируемите логически контролери намират широко приложение в различни области на индустрията. Чрез тях се управляват машини или процеси, които имат дискретен характер на действие.

Машината или устройството постоянно изпълнява предопределена последователност от деиствия или в резултат на определени събития, или след изтичане на зададено време. PLC се използват също за управление и на непрекъснати технологични процеси.

В последните години PLC се развиват много интензивно и се утвърждават като основна база на индустриалната автоматизация. Бързото развитие на микроелектрониката води до намаляване на цената и подобряване на характеристиките на PLC. Голямото разнообразие от PLC, предлагани на пазара, позволява да се намерят оптимални решения както за сравнително прости задачи, така и за комплексна автоматизация на производството.

Програмируемите логически контролери са компактни и гъвкави (универсални) устройства, които се използват за управление на машини (обекти) и/или технологични процеси.

Една от официално приетите дефиниции за PLC е следната:

Програмируеми (промишлени) контролери се наричат ***програмно управлявани електронни системи****, проектирани за използуване в качеството на промишлено оборудване за логическо управление на различни машини, съоръжения и технологични процеси през цифрови или аналогови входове и изходи. PLC са програмно управлявани устройства. Програмата извършва управлението на контролера - при активирането на входен сигнал от логическо (on/off) или аналогово входно устройство, се изработва съответстващ отговор - активиране на изход към съответствуващо изходно устройство (логическо или аналогово) в зависимост от съществуващите логически (програмни) връзки между отделните устройства.*

2. Особености на PLC

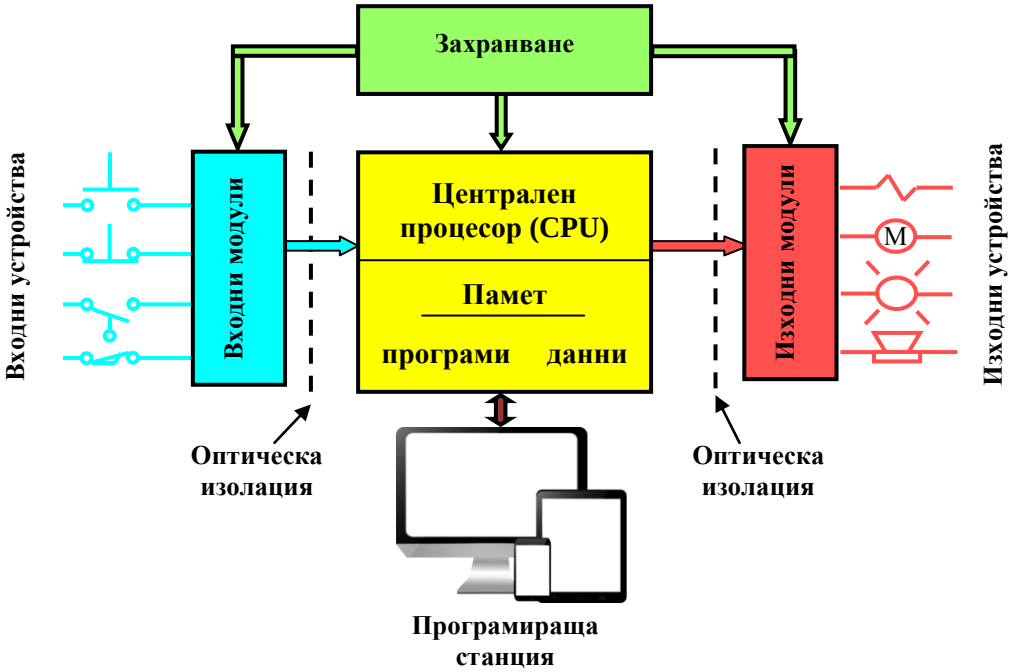
- PLC са програмно управлявани устройства.
- Операционната система на PLC се съхранява в ROM паметта.
- В RAM паметта се зарежда и изпълнява потребителската програма за управление на технологичния процес.
- Често към контролера се предлага и външно програмиращо устройство или специален софтуер за персонален компютър (PC).
- Притежават съединители за връзка с други контролери, външно програмиращо устройство или PC с инсталиран специален софтуер.

3. Основни модули на PLC

Независимо от големината и сложността си, контролерите притежават принципно следните основни модули, представени в блок-схемата, изобразена на фигура 1.1:

- **процесорен** – предназначен е да възприема сигналите от датчиците на системата и въз основа на програмата, записана в паметта на контролера, да издава изходни сигнали. Един контролер може да притежава един или повече процесорни модула, като всеки един от тях е функционално обоснован.

- **програмна памет** - съдържа (кодирана) последователността на работа на програмируемия контролер. Измерва се в килобайтове или в брой програмни инструкции. Варира в широки граници в зависимост от мощността на контролера.
- **памет за данни** - съдържа статуса на управлявания обект. Обемът й определя броя на вътрешните релета, таймери и др.
- **входен интерфейс** – осигурява електрическа обработка на сигналите от управляващия обект.
- **изходен интерфейс** – осигурява електрическа обработка на сигналите към управлявания обект. Обикновено типът и броят на входовете/изходите определят мощността на програмируемия контролер.
- **модул захранване;**
- **модул за връзка с други контролери, горно ниво (при свързване в мрежа) или персонален компютър;**
- **устройство за програмиране.**



Фиг. 1.1. Основни модули на PLC

Функционално програмируемият контролер „чете” входните сигнали и в отговор издава изходни сигнали. Изходните сигнали зависят от логиката, записана в програмата. Програмата предварително е записана в програмната памет на контролера посредством програмно устройство и програмен софтуер. Периодично процесорният модул чете и изпълнява програмата и въз основа на входните сигнали издава управляващи (изходни) синали. Периодът за еднократно изпълнение на програмата се нарича „време на сканиране“. Типично той варира от 5 до 50 милисекунди.

4. Програмируеми контролери TWIDO

Програмируемите логически контролери от серията TWDLMDAxxDxx на **Schneider Electric** са компактни или модулни устройства, позволяващи широк набор от функции и възможности. Съществуват различни комбинации от отделните модули, предлагани в серията TWIDO. В зависимост от приложението си, различните типове контролери са снабдени с основни и възможност за свързванене на допълнителни модули.

На фигура 1.2 са показани: модул „индикатор”, процесорен модул и допълнителен аналогов входно-изходен модул на модулни контролери от серията TWDLMDAxxDxx, а в таблица 1.1 са показани кратки спецификации на основните модели TWIDO контролери.

Фиг. 1.2. Модул „индикатор”, процесорен модул и допълнителен аналогов входно-изходен модул

Таблица 1.1. Спецификации на основните модели TWIDO

Модел контролер	Брой 24VDC входове	Брой и тип на изходите	Съединител	Серийни портове	Максимален брой входно-изходни модули за разширение	Допълнителни модули - тип касета
TWD LMDA 20DRT	12 sink /source	8 transistor source	Сменяема винтова клема	1 x RS 485 опционален 1 x RS 232C/485	7 модула	2 слота: RTC и EEPROM памет
TWD LMDA 20DTK	12 sink /source	8 transistor source	Съединител HE10	1 x RS 485 опционален 1 x RS 232C/485	4 модула	2 слота: RTC и EEPROM памет
TWD LMDA 20DUK	12 sink /source	8 transistor sink	Съединител HE10	1 x RS 485 опционален 1 x RS 232C/485	4 модула	2 слота: RTC и EEPROM памет
TWD LMDA 40DTK	24 sink /source	16 transistor source	Съединител HE10	1 x RS 485 опционален 1 x RS 232C/485	7 модула	2 слота: RTC и EEPROM памет
TWD LMDA 40DUK	24 sink /source	16 transistor sink	Съединител HE10	1 x RS 485 опционален 1 x RS 232C/485	7 модула	2 слота: RTC и EEPROM памет

4.1. Основен модул (процесорен модул) на TWDLMDA20DTK

Процесорният модул представлява хардуерно устройство с включени ограничен брой входове и изходи, които позволяват използване на модула като самостоятелно устройство за решение на определен клас от задачи за автоматизация.

Модулният контролер TWDLMDA20DTK включва:

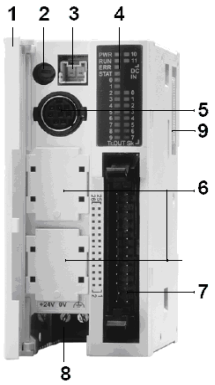
- 1 аналогов и 12 цифрови входа
- 8 цифрови изхода (transistor source)
- Захранващ модул
- Комуникационен модул
- Модул за следене на работата на контролера и състоянието на входовете/изходите (чрез светлинна индикация – LED)

- Възможност за свързване на до два модула („cartridge”):
    - за следене на реално време (RTC – real time clock cartridge)
    - допълнителна памет (EEPROM cartridge)
    - входен симулатор
- Възможност за сързване на до седем разширителни входно/изходни модула.

4.1.1. Описание на предния панел

Предният панел е показан на фигура 1.3. и включва:

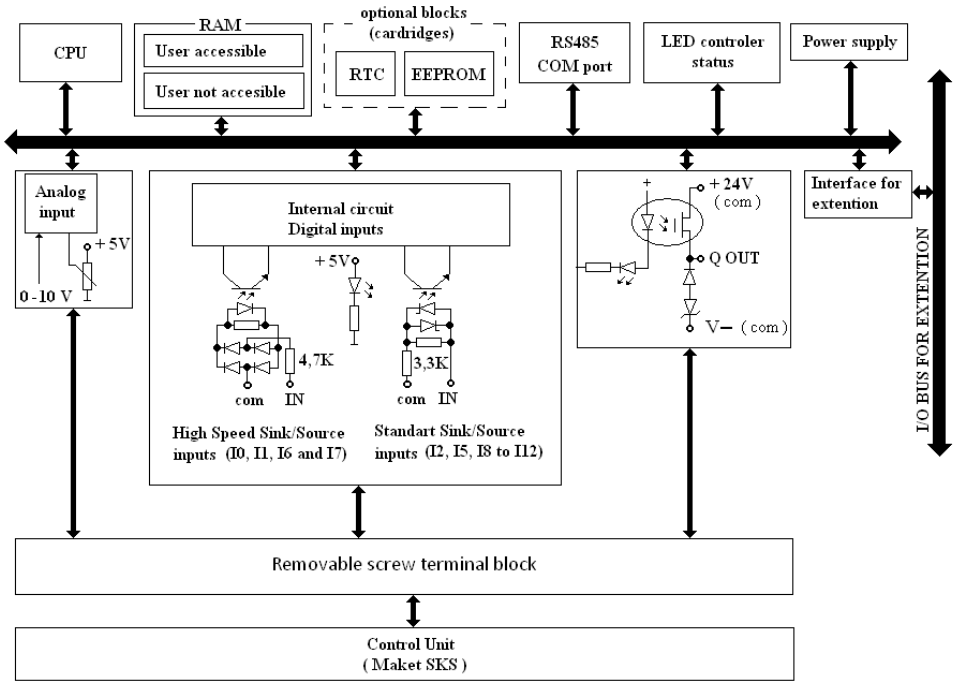
1. Основен преден капак
2. Аналогов потенциометър
3. Съединител за свързване с вградения аналогов вход
4. Блок със светлинна индикация, показващ:
   - състоянието на контролера (PWR, RUN, ERR и STAT),
   - състоянието на входовете и изходите (INi и OUTi).
5. Съединител за сериен порт RS 485 мини (от тип DIN)
6. Два слота (защитени от подвижни капачета)
   - за допълнителна памет - TWD XCP memory MFKxx
   - за външен часовник за реално време - TWD XCP RTC
7. Съединител за свързване на входно/изходни устройства.
8. Съединител за захранване 24 VDC
9. На десният панел: Съединител за допълнителни външни входно/изходни модули TWD Dxx и TWD AMx / AxM (4 или 7 в зависимост от модела).
9. На левия панел: Съединител за TWD XCP ODM дисплей модул или TWD NOZ xxx модул за сериен интерфейс.



Фиг. 1.3. Преден панел

4.1.2. Архитектура на процесорния модул TWDLMDA20DRT

На фигура 1.4 е показана блоковата схема на процесорен модул от типа TWDLMDA20DTK



Фиг. 1.4. Блокова схема на процесорен модул от типа TWDLMDA20DTK

4.2. Описание на блоковете

Централен процесор CPU – основната му задача е декодиране и изпълнение на инструкциите от програмното осигуряване. Процесорът изпълнява набор от команди, реализирани логически от транзисторни схеми. Основни характеристики на процесора са: производителност, разредност, честота на работа на ядрото, честота на работа с входно/изходните канали и обем на кеш-паметта за всяко ниво.

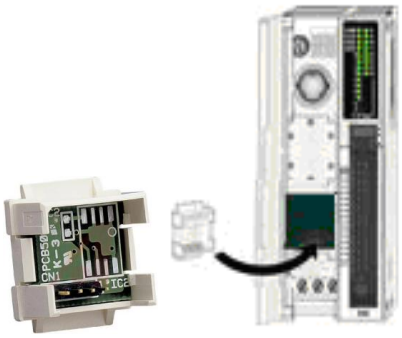
RAM – представлява вид енергозависима памет, която позволява достъп за четене и запис, до произволна част от съдържанието си без ограничения. Този вид памет може да бъде изпълнена като интегрални схеми (процесорни регистри, оперативна памет, различни видове кеш-памет) или на магнитни носители (феритна памет, твърди дискове, гъвкави магнитни дискове). Основната функция на тази памет е съхранение на инструкции за централния процесор (CPU). Целта е да предостави съдържанието си възможно най–бързо на процесора за последваща обработка. Вградената в контролера RAM памет е разделена на два дяла:

- **Памет, до която потребителят няма достъп** - в нея се съхранява програмната поддръжка на контролера (firmware).
- **Памет, до която потребителят има достъп** – в нея се съхраняват някои от потребителски програми и специализиран файл за възстановяване на програмата и съответните настройки (backup).

Предни разширителни слотове – на предния панел процесорният модул разполага с два разширителни слота (cartridge extentions). Чрез тях на потребителя се дава възможност сам да конфигурира модула според нуждите и желанията му.

Допълнителните модули, които потребителят може да използва, са:

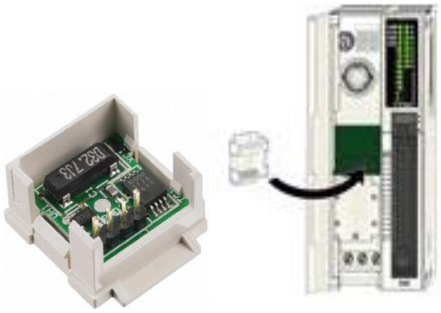
- **EEPROM** ***(E****lectrically* ***e****rasable* ***p****rogrammable* ***r****ead-****o****nly* ***m****emory)* - енергозависима памет, реализирана под формата на чип и се използва се за съхранение на малки обеми от данни (фигура 1.5).



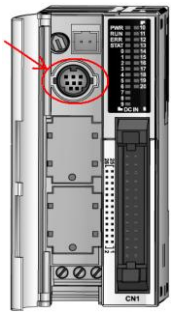
Фиг. 1.5. EEPROM допълнителна памет

- **Часовник за реално време** (**RTC** - **r**eal **t**ime **c**lock) (фигура 1.6) – интегрална схема, която следи настоящото време. Часовникът за реално време присъства в почти всички електронни устройства, в които е необходимо да се следи точно време. В процесорния модул часовникът за реално време също се използва за синхронизиране работата на отделните блокове по план. RTC има изключително малка консумация на ток и когато контролерът се изключи или по някаква причина отпадне захранването му, той запазва работоспособността си до 1000 часа при 25^0C или 300 часа при 55^0C.
- **Комуникационен модул** (фигура 1.7) - чрез комуникационния модул контролерът се свързва с управляващо устройство и синхронизира с други автоматизирани устройства при изграждане на система от машини. Програмируемите контролери от серията TWIDO

разполагат с RS485 последователен интерфейс за връзка с програмиращо устройство. Съединителят за връзка е разположен на предния панел.

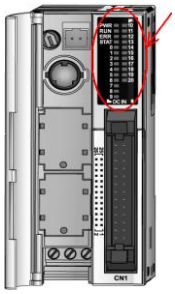


Фиг. 1.6. Часовник за реално време (RTC)



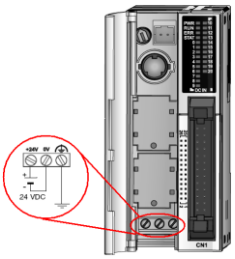
Фиг. 1.7. Комуникационен модул

- **Модул за индикация работата на контролера** (фигура 1.8) – светлинен (LED) модул, чрез който потребителят може да следи работното състояние на контролера и състоянията на всички отделни входове и изходи. Панелът дава следната информация:
  PWR – показва наличието на захранващо напрежение
  RUN – показва режим „работа” на контролера
  ERR – показва „грешка”
  STAT – показва статус при програмиране на контролера
  0 ÷ 12 – показва наличието на сигнал на входа
  13 ÷ 20 – показва наличието на сигнал на изхода



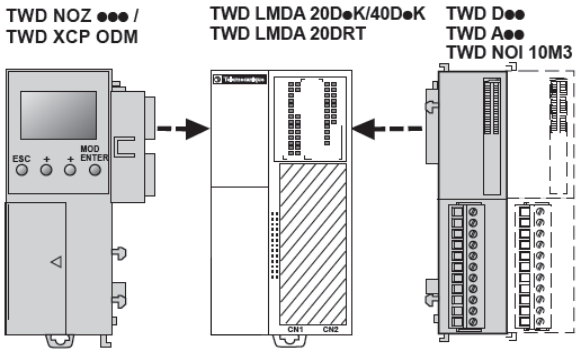
Фиг. 1.8. Модул за индикация

- **Захранващ модул** - осигурява вътрешно необходимите работни напрежения за контролера и неговите блокове. Най-често използваните вътрешни напрежения са +5V, ±12V и +24V. Процесорният модул се захранва от външен захранващ модул, като за целта на предния панел е изведен съединител за захранване (фигура 1.9).



Фиг. 1.9. Съединител за захранване

- **Странични разширителни слотове** – използват се за включване на допълнителни модули. Основният модул е процесорният, който може да работи както самостоятелно, така и с допълнителни модули. Протоколът, който се използва за осъществяване на комуникацията между отделните модули, се нарича “Modbus”. Това е Master/Slave протокол, който позволява на управляващото (Master) устройство да се „обръща” към определено подчинено или към всички подчинени (Slave) устройства. В зависимост от предназначението си комбинацията от модули може да бъде различна, но не неограничена (фигура 1.10).



Фиг. 1.10. Примерна конфигурация за разширение възможностите на процесорния модул

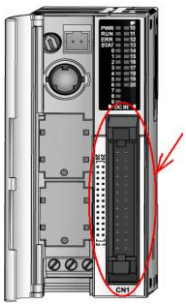
- **Входно-изходни блокове** – те осигуряват интерфейса между промишления контролер и външния апарат или процес, който трябва да се управлява, т. е. те предават активни сигнали и същевременно служат за „преграда” срещу смущенията, породени от силовите вериги на управляваното устройство. Тези смущения са с много високи стойности и могат не само да смутят правилната работа на микропроцесорната система, но и да я повредят. Ето защо, във входните и изходни вериги се вграждат разделителни и филтриращи елементи, които зависят от вида на предаваните сигнали.

Входните сигналите, с които контролерът работи, се делят на:

- **Релейни сигнали** - Изработват се от механични контакти на релета, превключватели, бутони и т. н. Имат две състояния - 0 (0 V, отворена верига) и 1 (обикновено 12-24 V, затворена верига). При включване и изключване, поради трептене на контактите, се получават паразитни импулси, траещи от 5 до 20 ms. Подтискането на смущенията води до времезакъснения при включване и при изключване на възприемането на сигналите.
- **Сигнали от безконтактни електронни датчици** - Сходни са с релейните, но липсват паразитните сигнали при включване и изключване, във връзка с което времезакъсненията при включване и при изключване са намалени.
- **Аналогови сигнали** - Те представляват непрекъсната функция на измерваната величина и следователно входно-изходните устройства не трябва да ги деформират, именно поради тази причина е възможно включване и изключване на филтрите на входа. Обикновено това са сигнали с ниска честота и напрежение до 10-12 V.

Изходните сигнали се извеждат от контролера най-често чрез електромагнитни релета. При по-отговорни случаи разделянето на входно-изходните вериги от микропроцесорната система се извършва чрез оптронни връзки. Оптронът представлява съвкупност от светодиод, чието светлинно излъчване се управлява от входния сигнал и фототранзистор, възприемащ това излъчване. Оптичната връзка е електрически изолатор и гарантира галванично разделяне на входа от изхода.

В процесорния модул TWD LMDA 20DTK всички входове и изходи са изведени на общ съединител HE10 (фигура 1.11).



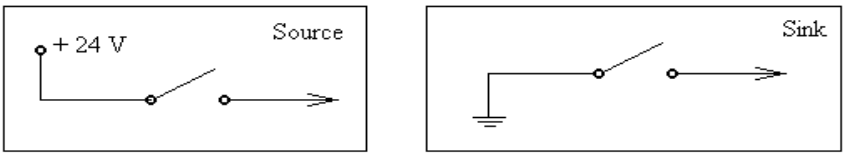
Фиг. 1.11. Входно / изходен съединител

Входове - дванадесетте цифрови входа на процесорния модул са разделени на бързи (high speed) - I0, I1, I6, I7 и бавни (normal) – I2, I5, I8 до I12, както и следните характеристики (Таблица 1.2):

Таблица 1.2. Параметри на цифровите входове

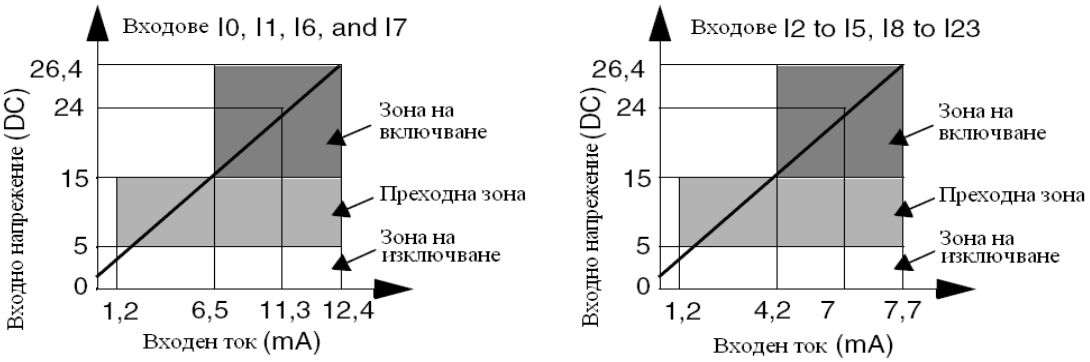
Тип на контролера	**TWDLMDA20DUK** **TWDLMDA20DTK**
Брой входове	12 (подредени в един съединител)
Входно напрежение	24VDC
Обхват на входното напрежение	20,4 - 26,4 VDC
Номинален входен ток	I0, I1, I6, I7 -> 5mA/вход (24VDC) I2 до I5, I8 до I23 -> 7mA/вход (24VDC)
Входно съпротивление	I0, I1, I6, I7 -> 5,7 KΩ I2 до I5, I8 до I23 -> 3,4KΩ
Време за включване (ON Time)	I0 до I7 -> 35μS + с филтъра I8 до I23 -> 40μS + с филтъра
Време за изключване (OFF Time)	I0, I1, I6, I7 -> 45 μS + с филтъра I2 до I5, I8 до I23 -> 150 μS + с филтъра
Изолация	Между входовете – няма; между входовете и вградената схема - оптрон
Филтриране	I0 до I11 -> три разновидности: - няма;- 3ms; -12ms
	I12 до I23 -> без филтър
Тип на входа	Тип 1 (IEC 61131)* - (Фиг. 1.16)
Определяне на сигнала	Статично
Ефект от неподходящ сигнал на входа	Входовете могат да бъдат "sink" или "source"*, ненадвишаващи обхвата на входното напрежение. При поява на различен сигнал от указаното, това води до повреда.
Затихване	3м (9,84 ft) със защита от електромагнитни смущения
Издръжливост на съединителя (при прикачане и откачане на съединителя)	минимум 100 пъти

* Типове входове – различните контролери от серията TWIDO притежават различни входове. Те се различават както по параметри, така и по тип на свързване. Основните видове са (фигура 1.12):



Фиг. 1.12. Видове входове

TWDLMDA20DTK разполага с входове от тип “Sink”. Работният обхват е показан на фигура 1.13.

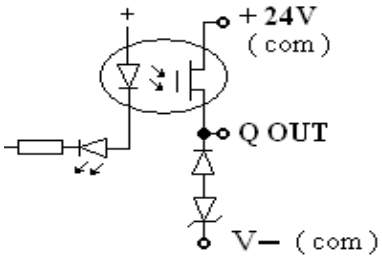


Фиг. 1.13. Работен обхват на входове от тип Sink (IEC 1131)

Голяма част от модулните контролери на TWIDO притежават по един вграден аналогов вход и променлив резистор (потенциометър) за неговото регулиране (фигура 1.14). Аналоговият вход работи с напрежение от 0 до 10V. Стойността на входящото аналогово напрежение се конвертира в цифрово (0 – 512), запаметява се като системна „дума” в контролера и се обновява при всяко сканиране на входовете. Чрез потенциометъра, разположен на предния панел, може да се зададе стойност от 0 до 1024.

Фиг. 1.14. Вграден аналогов входен модул

Изходи – Представеният процесорен модул TWD LMDA 20DTK разполага с изходи от тип transistor source (фигура 1.15) и характеристики, показани в таблица 1.3.



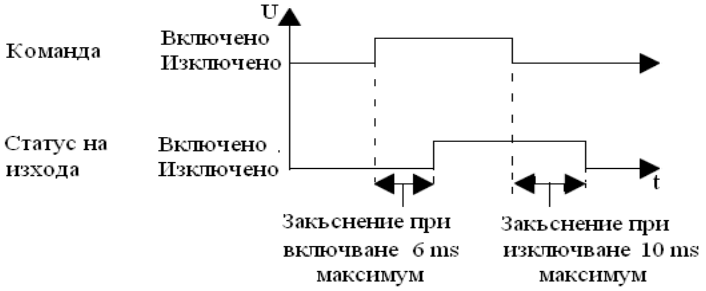
Фиг. 1.15. Електрическа схема на цифровите изходи тип “transistor source”

Таблица 1.3. Характеристики на цифровите изходи

Контролер	LMDA20DTK	LMDA20DUK
Тип на изходите	Source output	Sink output
Брой изходи	8	
Напрежение на товара	24VDC	

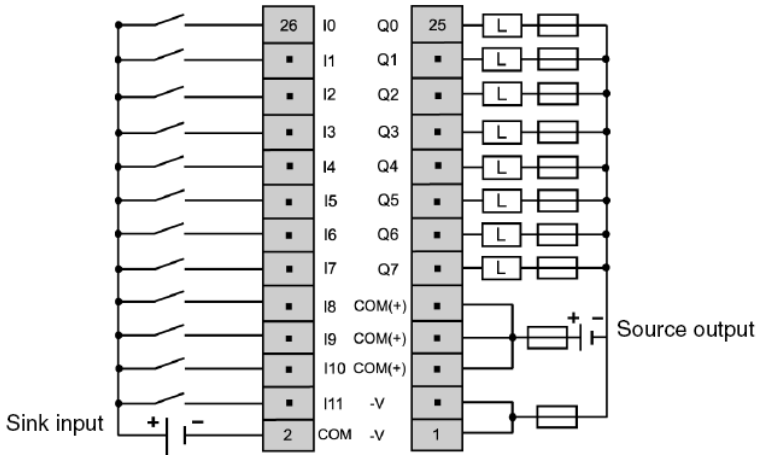
Максимален ток на товара	1A
Обхват на работното напрежение	20.4 - 28.8
Пад на напрежение	1V max (между изход и маса, изходът е активен)
Ток на товара	0,3A / изход
Мксимална мощност на товар (лампа)	8W
Индуктивен товар	L/R = 10ms (28,8 VDC, 1Hz)
Изолация	Между изходния куплинг и вградената схема – оптрон; Между изходните терминали - няма
Брой на вхлючване/ изключване на съединителя	100 пъти минимум
Закъснение при включване	Q0, Q1 -> 5μs maximum; Q2 to Q15 -> 300 μs maximum
Закъснение при изключване	Q0, Q1 -> 5μs maximum; Q2 to Q15 -> 300 μs maximum

Една от най–важните характеристики на изходите е тяхното закъснение. В зависимост от типа на изходите закъснението е различно, като за тип „transistor source” това е показано на времедиаграмата от фигура 1.16.



Фиг. 1.16. Времедиаграма за закъсненията на изходите

Схемата на свързване на входовете, изходите и захранването на процесорния модул е показана на фигура 1.17.



Фиг. 1.17. Схема на свързване на входовете и изходите

- COM (+) терминалите са свързани помежду си вътрешно;

- COM и COM (+) терминалите са различни и не са свързани помежду си;
- V- (V минус) терминалите са свързани помежду си вътрешно;
- Предпазителят се избира според товара.

Основни характеристики на модулните контролери TWDLMDAxxDxx

Таблица 1.4. Основни характеристики на програмируемите контролери TWDLMDAxxDxx

Основен модул		**TWD**	**LMDA 20 DTK**	**LMDA 20 DUK**	**LMDA 20 DRT**	**LMDA 40 DTK**	**LMDA 40 DUK**
Брой входове			12			24	
Брой изходи (вид)			8 transistor source	8 transistor sink	6 релейни 2 transistor source	16 transistor source	16 transisitor sink
Съединители			HE 10		Изваждаем клеморед	HE 10	
Възможност за I/O разширяване	макс. брой допълнителни модули		4		7		
	макс. брой допълнителни точки		132/244 (1)		152/264 (1)		
Максимален обем на програмата		Kb	3000 стъпки		3000 стъпки ;6000 с TWD XCP MFK64 (външна памет)		
Време за изпълнение на програмата	Базови инструкции	ms	1 за 1000 логически стъпки				
	В допълни-телна система	ms	0,5				
Информационна памет	Вградени битове		128				
	Вградени думи		1024(2048 макс.) (2)				
	Таймер		32 (64 макс.) (2)				
	Брояч		16 (64 макс.) (2)				
Захранване	Номинално захранване	VDC	24				
	Граници на захранващото напрежение	VDC	20,4 … 26,4 (включително малки промени в захранването)				
	Максимален ток при 26,4 V	mA	560 при 26,4V		700 при 26,4 V		
	Максимален ток	A	50				
	Максимална мощност	W	15 (базова конФигура 1.ция с 4 вх./изх. разшир.)		19 (базова конФигура 1.ция с 7 вх/изх разширения)		
Комуникационни възможности							
Вид порт			RS 485		RS 232C, TWDNAC 232D адаптер		

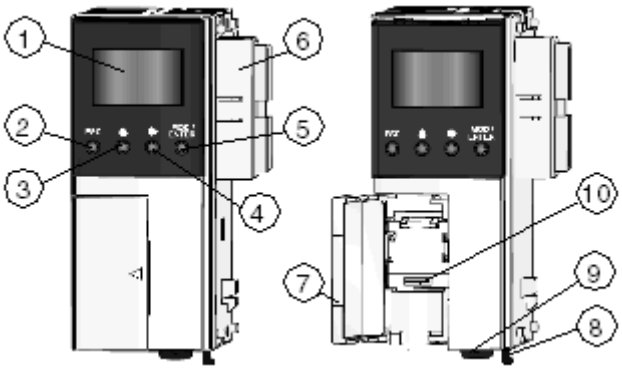
				RS 485, TWDNAC 485Cx адаптер
Максимална скорост		Kbps	19,2	
Програмиране	наличие на порт		Да	Не
Протокол за комуникация			Modbus RTU/ASCII Master/Slave Character mode ASCII	
Отдалечен достъп до вход/изход			Да	
Вградени функции				
Бърз брояч	Брой канали		3	
	Честота	KHz	Еднофазно, 5 (2 pts); Едно/Дву-фазово (избираемо), 20 (2 pts)	
	Капацитет		16 бита (0 ... 65535 точки)	
Позициониране	Брой канали		2	
	Честота	KHz	7	
	Функции		PWM - pulse witdh modulation output; PLS - pulse generator output	
Аналогов вход	Брой канали		1 канал	
	Обхват на входното напрежение		0 ... 10 VDC	
	максимален обем		9 бита (0 ... 511 точки)	
	Импеданс	KΩ	100	
Аналогов потенциометър			1 позиция може да бъде настроена от 0 ... 1023	
Допълнителен часовник				
Точност		Sec / месец	± 30 при 25^0C	
Продължителност		дена	~ 30 (нормално) при 25^0C (77^0 F) след пълно зареждане на батерията	
Вградена батерия			външна литиева батерия	
Време за зареждане		часа	~ 10 за зареждане от 0 до 90% от пълното зареждане	
Допълнителна памет				
Вид памет			EEPROM	
Обем		Kb	32	64
Възможност за възстановяване от резервно копие			Да	
Възможност за увеличаване на размера на резервно копие			Не	Да

1.1. Дисплеен модул – TWDXCPODM

Операторският дисплеен модул е допълнителен, който може да се добави към някои от базовите модули. Операторският дисплей предоставя следните услуги:

- Показва информация за състоянието на базовия модул в какъв режим на работа е.
- Позволява на потребителя да пуска и спира заредената програма в контролера.
- Позволява на потребителя да наблюдава и да настройва данните за RTC.

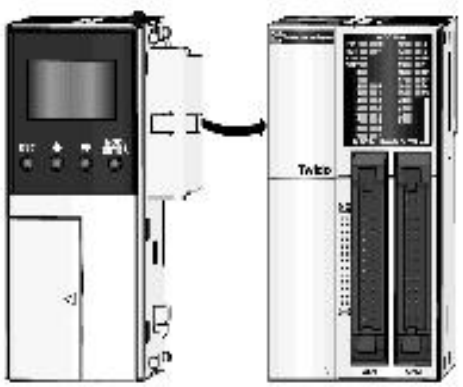
На фигура 1.18 са показани частите на операторския дисплеен модул TWDXCPODM .



Фиг. 1.18. Структура на модул TWDXCPODM

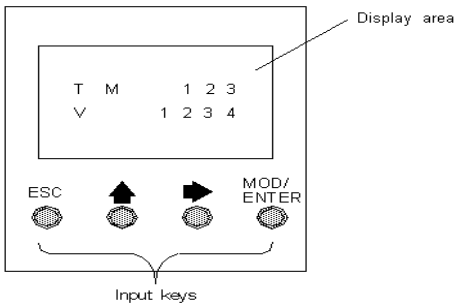
1. Дисплей - показва менюта, операнди и данни.
2. ESC бутон – връща дисплея към предишното състояние и отхвърля промените, направени от потребителя.
3. Бутон със стрелка нагоре - променя текущия елемент; редактиране на следващата стойност.
4. Бутон със стрелка надясно - в дисплеен режим с негова помощ се преминава към следващото състояние на дисплея; в режим на редактиране с негова помощ се преминава към следващия елемент за настройка (елементът, който ще се променя, започва да мига).
5. MOD / ENTER бутон - работи в ENTER режим и се връща към предишното състояние на дисплея и приема промените, направени от потребителя.
6. Съединител за връзка с базовия модул.
7. Вратичка с панти – при отварянето й потребителят получава достъп до втория сериен порт.
8. Резе – то задържа и пуска модула от контролера.
9. Скоба, която осигурява модула с DIN – съединител.
10. Съединител на серийния порт 2 – тук се свързва именно с TWDNAV485D - TWDNAC232D, TWDNAC485D или TWDNAC485T.

На фигура 1.19 е показан начинът на подключване към процесорния модул. Подреждат се в една линия дисплейният модул и програмируемият контролер, така че съединителят за връзка на дисплейния модул да бъде от лявата страна на програмируемия контролер.



Фиг. 1.19. Свързване на дисплейния модул към процесорния

Фигура 1.20 показва поглед на операторския Display, който се състои от една зона на образа (тук е в нормален режим).



Фиг. 1.20. Операторски дисплей

1.2. Аналогов модул TWDALM3LT (фигура 1.21)

Един от основните допълнителни елементи, с които се разширява базовата конфигурация на контролера, е модулът за работа с аналогови сигнали (т. нар. „аналогов модул”). Тъй като контролерът е цифрово устройство, то аналоговите входни сигнали се преобразуват в цифров код, като се използва АЦП (аналогово–цифров преобразувател) и мултиплексор. Вграденият в контролера АЦП е 9 битов.

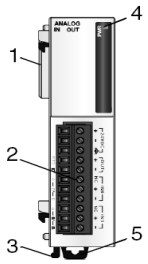
Аналоговите изходи извършват обратното преобразуване – от цифров код в аналогов сигнал чрез отделен ЦАП (цифрово–аналогов преобразувател) за всеки от изходите.

Фиг. 1.21. Аналогов модул TWDALM3LT

Аналоговият модул TWDALM3LT е снабден с два входа и един изход, като за източник на входни сигнали могат да се използват подходящи датчици (например термодвойка или сигнал от резистивен термометър (термистор) и други).

4.4.1. Описание на предния панел (фигура 1.22)

1. Разширителен съединител (Modbus);
2. Сменяем съединител на входно-изходния модул;
3. Бутон за междумодулно заключване (при съединяване на отделните модули);
4. LED индикация;
5. Скоба за монтаж.

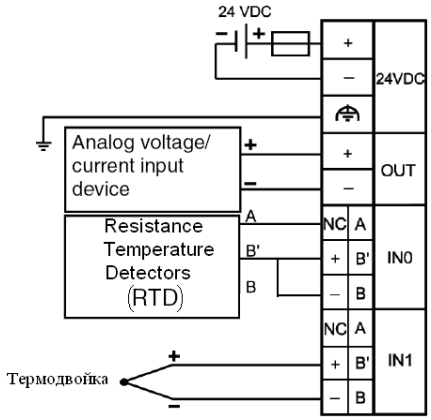


Фиг. 1.22. Преден панел на аналоговия модул

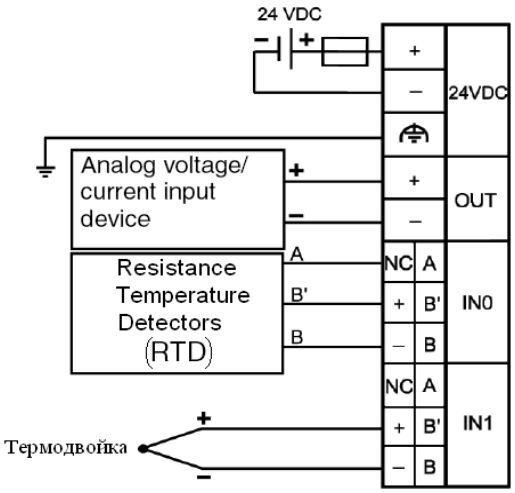
4.4.2. Електрически схеми на свързване на аналоговия модул

Аналоговият модул разполага със стандартен общ съединител за връзка с входно-изходните му модули. В зависимост от предназначението си и очакваната точност, източниците на информация (датчиците) могат да бъдат свързани по три различни начина:

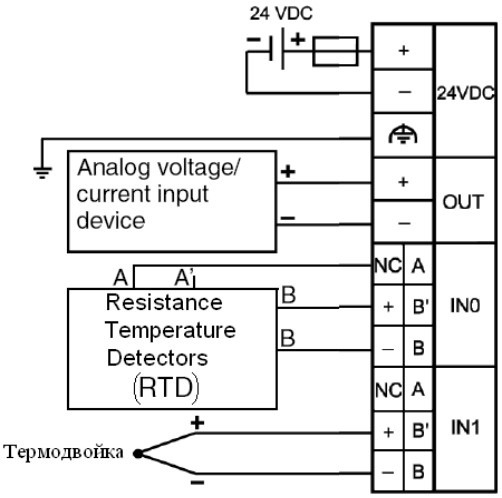
- двупроводно свързани (фигура 1.23)
- трипроводно свързани (фигура 1.24)
- четирипроводно свързани (фигура 1.25)



Фиг. 1.23. Схема на двупроводно свързване



Фиг. 1.24. Схема на трипроводно свързвне

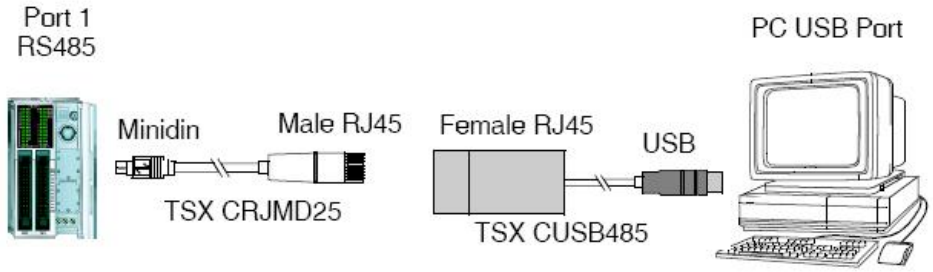


Фиг. 1.25. Схема на четиропроводно свързване

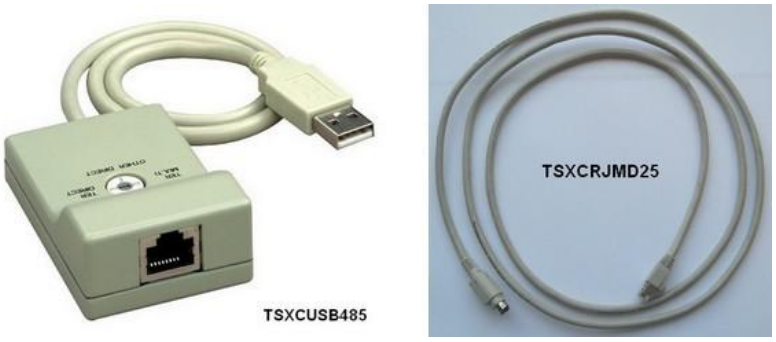
За да осъществи правилен контрол над случващите се действия промишленият контролер има нужда от информация за случващото се в момента. Именно тази информация се получава от свързаните към контролера различни видове датчици. *Датчиците (сензорите)* представляват първични преобразуватели на физически или химически параметри в удобен за ползване сигнал. В зависимост от приложението се подбират датчици с подходящи параметри. Датчиците могат да се разграничат по много и различни параметри, но основните от тях са вида на генерирания от датчика сигнал – цифров или аналогов и неговата чувствителност.

1.3. Модул за връзка с програмиращо устройство

Както подсказва наименованието на програмируемия контролер, преди подключването му за започване на какъвто и да е вид работа, той трябва да бъде програмиран за конкретното задание. Подобно на компютърното програмиране, програмирането на промишлени контролери също представлява процес на писане, тестване и поддържане на програмното осигуряване на контролера. Програмирането на контролера се извършва от специално проектирано за целта устройство, най–често персонален компютър. За програмиране на контролерите от серията TWIDO фирмата производител Shneider Electric е разработила специализиран софтуер за програмиране посредством персонален компютър на име „TWIDO SUITE”. Реализираната на базата на релейно–контактен език програма се записва в паметта на контролера посредством кабел за порограмиране, който свързва персоналния компютър с програмируемия контролер (фигура 1.26).

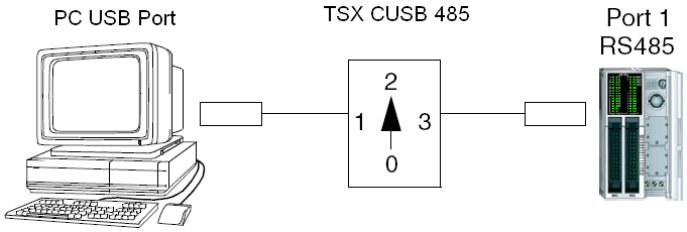


Фиг. 1.26. Връзка между програмиращата станция и PLC



Фиг. 1.27. TSX CUSB485 и TSX CRJMD25 комуникационни кабели за програмиране

TSX CUSB485 и TSX CRJMD25 кабелите (фигура 1.27) служат за конвертиране на сигналите между USB (при компютъра) и RS485 (при контролера). TSX CUSB485 е снабден с 4-позиционен ротационен превключвател, служещ за избор на режим на работа. За връзка между персонален компютър и програмируемия контролер TWIDO се използва позиция 2 на ротационния превключвател (фигура 1.28).



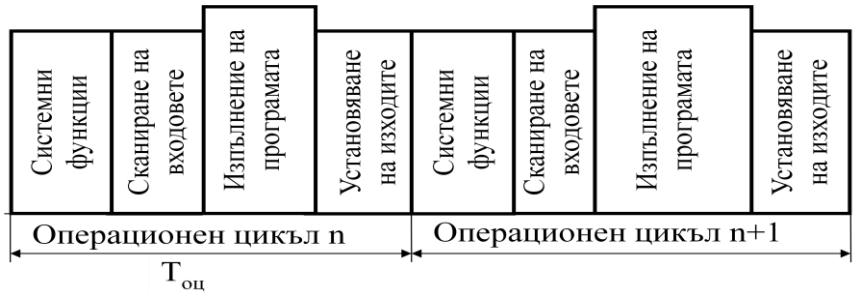
Фиг. 1.28. TSX CUSB485 – ротационен превключвател с 4 позиции

5. Принцип на работа и операционно обслужване на PLC

Работата на PLC се управлява от операционна система. Тя осигурява изпълнението на следните процедури от CPU:

- сканиране състоянието на входовете и записване на това състояние в специално определена област на паметта;
- изпълнение на потребителската програма;
- записване състоянието на изходите в специално определена област на паметта и установяване на изходите в състояние, съответстващо на резултата от изпълнението на програмата;
- изпълнение на системни функции по контрол и управление на апаратни и програмно вградени функции и комуникация.

Тези процедури се повтарят циклично и осигуряват нормалната работа на PLC управлявания обект в съответствие със заложената програма. Всеки работен цикъл (наречен оперативен цикъл) се състои от изброените по-горе фази (фигура 1.29). Продължителността на отделните цикли може да е различна и зависи от изпълняваната програма. Средната продължителност е от порядъка на няколко микросекунди.



Фиг. 1.29. Диаграма на фазите на операционния цикъл на PLC

Ясно е, че за да се регистрира промяната на входен сигнал, минималната продължителност на входните сигнали трябва да бъде по-голяма от продължителността на операционния цикъл, т.е.

$$t_{вх.\ мин} = T_{оц}+T_{вх}$$

където: $t_{вх.мин.}$ е минималната продължителност на входните сигнали;
$T_{оц}$ – продължителността на операционния цикъл;
$T_{вх}$ - продължителността на фазата за сканиране на входовете.

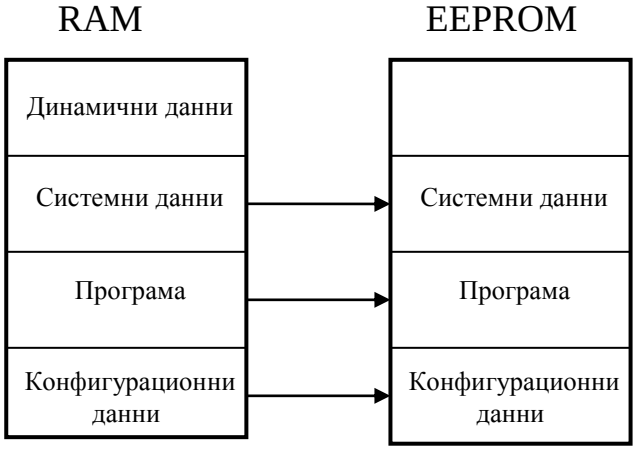
Когато се налага PLC да работи с входни сигнали с малка продължителност (импулси), се използват специални апаратно-програмни средства. Използват се два подхода за тази цел. Първият е чрез тригерни входове, които реагират на входните сигнали в момента на промяната и се запомнят в RS тригери. По време на изпълнение на системните функции, състоянието на тригерите се прочита и се прехвърля в съответната област на паметта, а в края на фазата се нулират. Вторият е чисто програмен. При него се прилага принципа на прекъсване на изпълнението на потребителската програма при наличие на импулсен вход и изпълнение на подпрограма, която записва текущото състояние на входния сигнал в съответната област. След това се възстановява изпълнението на основната програма. Определянето кои физически входове ще се използват за работа с импулсни входни сигнали става по време на инициализация на контролера.

Друга основна характеристика на PLC е времето за реакция. Това време зависи от момента на промяна на входния сигнал спрямо фазите на операционния цикъл. Максималното време, за което контролера ще реагира на промяна на входен сигнал, чрез съответната промяна на изходите в съответствие с програмата, е:

$$t_{реакц.макс.} = 2T_{оц} + T_{вх}$$

Наред с броя на входовете и изходите и размера на паметта, тези характеристики на PLC са основните, които определят възможностите на контролера да се използва за управление на конкретен обект.

Друга характерна особеност на PLC е начинът на разпределение и използване на паметта (фигура 1.30). Както вече стана ясно, паметта на PLC се състои от две части - оперативна памет (RAM) и постоянна памет (EEPROM). По време на работа на PLC конфигурационните данни, потребителската програма, данните, използвани от нея, а също и междинните резултати, получени при изпълнението й, се съхраняват в RAM паметта. За предпазване от аварийни ситуации в управлявания обект при внезапно отпадане на захранването, RAM паметта (или част от нея), е буферирана с батерийно захранване. Това осигурява съхранение на информацията за определено време (примерно в продължение на 30 дена, при изправна и напълно заредена батерия за контролер TWIDO). За трайно съхранение програмата, конфигурационните и системни данни, се записват като резервно копие в постоянната памет.



Фиг. 1.30. Структура на паметта на PLC

Съвременните PLC са микропроцесорни системи за управление, които работят в режим на реално време, т.е. те приемат информацията за състоянието на управлявания обект, обработват я в съответствие със заложената в паметта им програма и подават управляващи сигнали към изпълнителните органи на обекта. От разгледания принцип на работа (от оперативния цикъл на PLC) се вижда, че в процесорния модул (CPU) работят два различни вида програми:

- Операционна система за реално време, която организира оперативния цикъл на PLC и взаимодействието между отделните елементи на потребителската програма в съответствие с настъпилите външни (от управлявания обект) и вътрешни (от изпълнението на потребителската програма) събития.
- Потребителска програма, посредством която се извършва управлението на конкретния обект в съответствие с избрания алгоритъм на работа.

Операционната система осигурява “работна среда” за потребителската програма. Тя осигурява изпълнението на всички фази на операционния цикъл, а също и редица обслужващи функции. Операционните системи от този тип се разделят условно на две части: управляваща част (при повечето производители се нарича ядро) и обслужващи програми. Към втората част се отнасят вградените софтуерни функции и функционални блокове (таймери и броячи, математически, логически и др. функции). Основна част на операционната система е интерпретаторът. Той осигурява интерфейса между потребителската програма и микропроцесора на CPU модула, с други думи превежда на разбираем за процесора код инструкциите, от които е съставена потребителската програма. Множеството от инструкции, поддържано от съответния процесорен модул, определя използването на конкретния PLC в

определен клас задачи за управление. От него зависи времето за изпълнение на потребителската програма, а от там и времето за реакция на управляващата системата като цяло. Стуктурата на отделните инструкции, от друга страна, определя необходимата памет за потребителската програма.

6. Езици за програмиране на PLC

В практиката на използване на PLC най-голямо приложение са намерили следните езици за програмиране:

А. Списък с инструкции (Instruction List - IL)
Б. Структуриран текст (Structured Text - ST)
В. Релеен език (Ladder Diagram - LD)
Г. Функционална блокова диаграма (Function Block Diagram - FBD)

А. Списъкът с инструкции (Instruction List - IL) е език от текстов тип и съдържа последователност от инструкции на отделен ред от програмата:

[етикет] оператор [операнд[,операнд...]] [коментари]

Пример:

```
…….
LD    %IX6.2      (* чете входен бит с адрес 6.2 *)
ST    BIT 1       (* записва резултата в BIT 1 *)
CAL FUNC 1        (* извиква функционален блок FUNC 1 *)
……
```

Б. Структуриран текст (Structured Text - ST) е друг тип текстов език за програмиране на PLC. Основен елемент в ST е изразът. Изразът е конструкция, от която в процеса на изпълнение се получава стойност от определен тип. Изразите представляват описания на операции върху операнди. Операнди могат да бъдат променливи, литерали, функции или други изрази. Типични операции са: извикване на функция, степенуване, смяна на знака, инвертиране, умножение, делене, събиране, сравнение, конюнкция, дизюнкция, сума по модул 2 и др. Пресмятанията на булев израз се прекратяват, ако стойността на израза не зависи от оставащата част, която предстои да се изчисли. Програмите, написани на езика ST, са последователности от оператори, разделени със запетая.

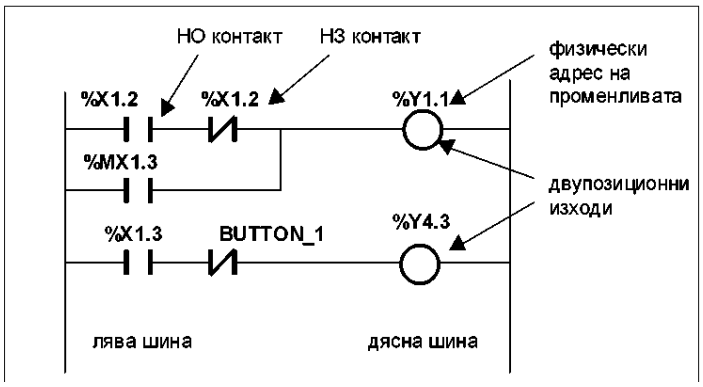
В. Релейният език (Стълбовидна диаграма, Ladder Diagram - LD) е графичен език, удобен за начинаещи програмисти или специалисти, привикнали с графичния вид на релейно-контактните схеми (РКС). Целта тук е в максимално близък до традиционния вид да се представи действието на управляващата програма (фигура 1.31). Използват се следните елементи: променлива, нейно допълнение, начало и край на разклонение при паралелна верига, присвояване на резултата на изходна или междинна променлива. Дължината на веригата може да е фиксирана или променлива. Отделните вериги се обединяват в линия. В най-лявата и най-дясната части на диаграмата се намират "захранващи" шини. Те съответстват на захранващите линии в оперативната верига на една РКС. Между тях се разполага мрежа от елементи, които са свързани както помежду си, така и със страничните шини. Тези елементи са:

- *Връзки* - означават се с хоризонтални и вертикални линии. Състоянието на всяка връзка може да бъде "0" или "1", съответстващи на липса или наличие на напрежение. Състоянието на лявата захранваща шина е винаги "1". Състоянието на дясната шина не е дефинирано, като най-често тя дори и не се изобразява на схемата.
- *Контакти* - биват нормално отворени -| |- и нормално затворени -|/|-, както и контакти, чувствителни по нарастващ (положителен) "0 → 1" или по падащ (отрицателен фронт) "1 → 0".

- *Изходни елементи* с означение -( )- съответстват на бобините на релета или контактори в РКС. Те получават "захранване", само ако състоянието на елементите вляво от тях е "1". Изходните елементи са винаги двупозиционни и могат да бъдат: нормални; инверсни; с нулиране и запомняне; чувствителни по фронт и др.

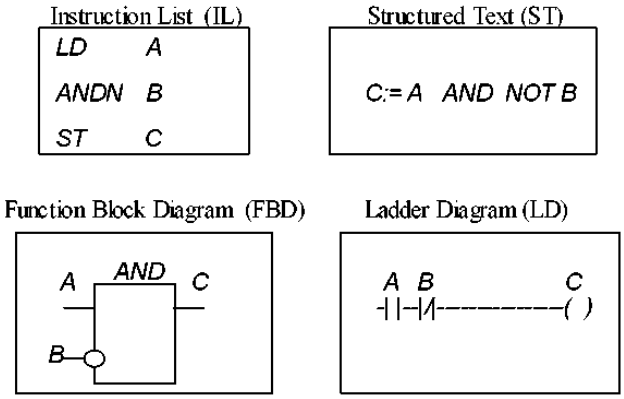
В LD програмите могат също да бъдат използвани функции и функционални блокове, по подобие на текстовите езици IL и ST.

Пример:



Фиг. 1.31. Стълбовидна диаграма

Г. Функционално-блоковата диаграма (Functional Block Diagram - FBD) е език за програмиране, основан на аналогия с алгоритъма за логическо управление на технологичния процес или с неговата схема на функциониране. Приема се, че съществува поток на сигналите, който преминава през диаграмата и определя състоянията на нейните елементи по аналогия с действието на логическите схеми в схемотехниката. На външен вид FBD не се отличават от функциите или функционалните блокове. Означават се с правоъгълници, при които входовете са отляво, а изходите - отдясно на символа. Фигура 1.32 представя един и същ сегмент от програма, реализиран на различните езици.

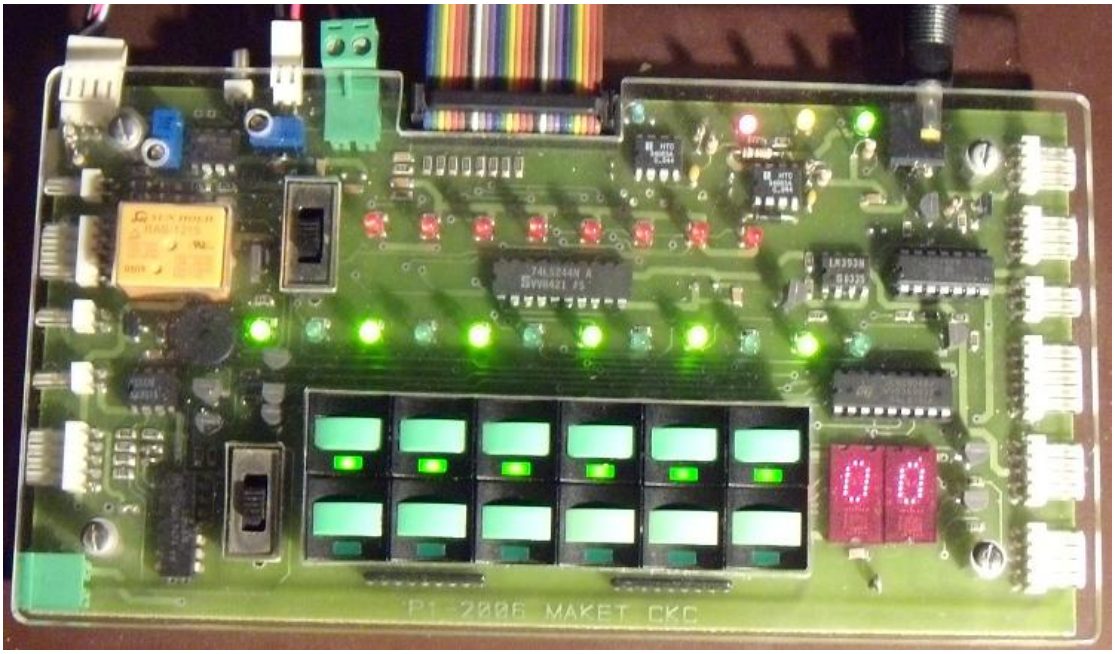


Фиг. 1.32. Сегмент от програма, реализиран на различни езици

Лабораторен макет за управление и симулация на възможностите на ПЛК TWDLMDA**D*K (Макет СКС)

1. Въведение

За целите на обучението по дисциплината „Специализирани компютърни системи” е разработен „**Макет СКС”** (фигура 2.1), който позволява тестване на функционалните възможности на контролерите TWDLMDA**D*K.

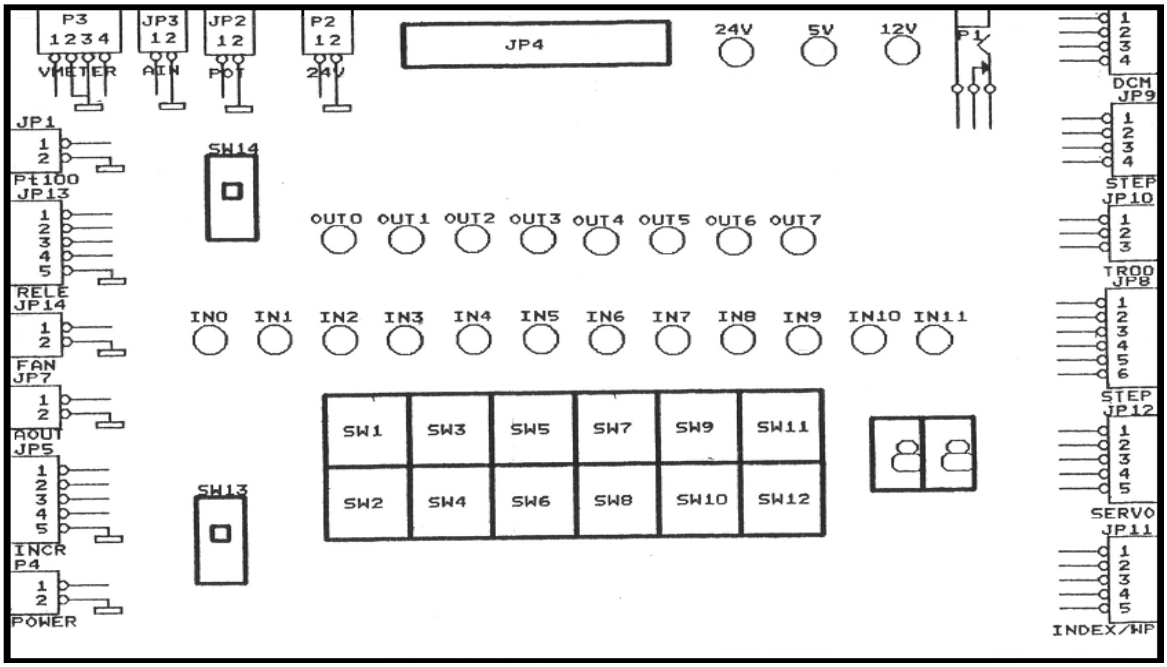


Фиг. 2.1. Макет СКС

Макетът позволява обучение чрез симулативно представяне на работата на контролера, а също така е възможно да се използва и като развойно средство при разработване на програмно осигуряване на контролерите. Той включва няколко по-малки възли за тестване на различните функции на програмируемия контролер. Това са :

- Връзка на макета с програмируемия контролер;
- Модул за захранване на макета;
- Възел за управление на правотоков електродвигател с вграден тахогенератор;
- Възел за управление на правотоков електродвигател с вграден регулатор;
- Възел за управление на стъпков електродвигател (със и без средна точка);
- LED индикация с вграден дешифратор;
- Вход за оптодатчици (пътечка „0”, индекс, защита от запис);
- Вход за фоторастеров преобразувател (инкрементален датчик + симулатор);
- Възел за управление на цифровите входове;
- LED индикация на състоянията на цифровите входове и изходи;
- Блок „широичинно–импулсен модулатор” (PWM);
- Блок за звукова сигнализация (zummer);
- Блок за управление на реле;
- Блок за измерване на температура;
- Блок цифров волтметър.

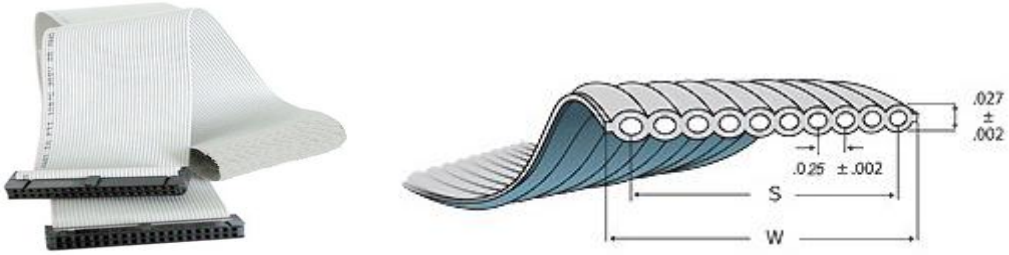
Схемата на разположение на отделните възли е показана на фигура 2.2.



Фиг. 2.2. Макет СКС – разположение на отделните възли

2. Модул за връзка с контролер

Модулът за тестване се свързва с програмируемия контролер посредством 26 линеен кабел, чийто куплунг е изобразен на фигура 2.3 с обозначение „JP4”.



Фиг. 2.3. Кабел за трансфер на информация между модула за симулация и програмируемия контролер

Основната задача на този кабел е пренасянето на информация от и до симулативните входове и изходи.

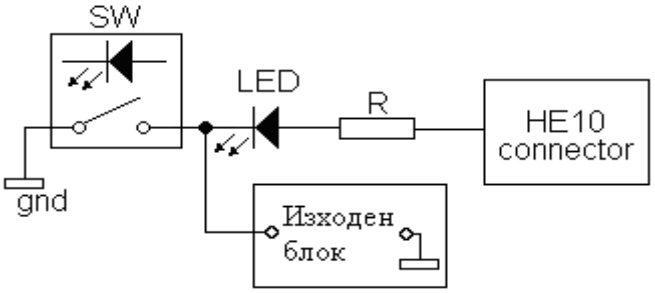
3. Входове

Програмируемият логически контролер работи като циклично обхожда въведената му програма. Работният цикъл се състои основно от три стъпки: проверка на състоянията на входовете, изпълнение на програмата, обновяване състоянието на изходите. При първата основна стъпка контролерът проверява състоянието на всеки свой вход, като записва резултатите в паметта си. При втората стъпка се изпълнява програмата, като се обработват инструкциите една по една и получените резултати се съхраняват за по-нататъшната им употреба. При третата стъпка контролерът, въз основа на състоянието на входовете и резултатите от изпълнението на програмата, определя какво ще бъде състоянието на всеки изход и подава необходимите сигнали и информация към тях.

3.1. Симулатор на цифровите входове

Както бе споменато по–горе, в процеса на работа програмируемите контролери първо проверяват за състоянието на входовете си. Информация за настъпило събитие контролерите

получават посредством различни видове датчици, свързани към различните входове. Основната роля на датчика е да преобразува физически или химически параметър в удобен за ползване сигнал. Симулирането на настъпило събитие се свежда до генериране и подаване на импулс към съответния цифров или аналогов вход. Макет СКС е снабден с дванадесет броя електронни самозадържащи се бутона (SWx), които са свързани в NO (normal open) - нормално отворено състояние. Целта на тези бутони (фигура 2.4) е да затворят веригата, свързана към входа на контролера, като по този начин симулират наличието на импулс (сигнал) на цифровите входове. В схемата на всеки един вход присъства по един свтодиод (освен вграденият в бутона), индикиращ наличието на подаден сигнал. Всички ключове са свързани към съединителя за връзка с програмируемия контролер.



Фиг. 2.4. Схема за симулиране наличието на сигнал на цифров вход

Освен стандартната си работа, една част от електронните ключове могат да се използват и за специализирани цели в макет СКС. Това са следните ключове (таблица 2.1):

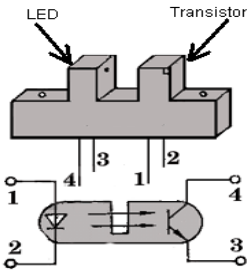
Таблица 2.1. Бутони и функции

Номер на бутон	Специализирана функция
6	"track00" - пътечка нула
7	"index" - индекс
8	"WP" - защита от запис
9	"TG" -
10	"RESET" - рестарт
11	"+1" - прибавяне на 1
12	"-1" - изваждане на 1

Специализираните ключове се използват в комбинация с други възли от макета, като позволяват управление на цял обект. Например ключове 6, 7 и 8 се използват за симулативно управление на модул „флопи–дисково устройство”. Ключове 9, 10, 11 и 12 пък се използват за управление на инкрементален датчик.

3.1.1. Вход за оптодатчици (пътечка „0”, индекс, защита от запис)

Фотоелектрическият датчик (фигура 2.5) се състои от излъчвател (светодиод) и от светлочувствителен приемник (фототранзистор).



Фиг. 2.5. Фотоелектрически датчик

Светодиодът представлява полупроводников електронен елемент, който излъчва светлина, когато през него преминава електрически ток. Тази светлина може да бъде видима или невидима, в зависимост от дължината на вълната на излъчване. Детекцията се реализира, когато даден обект премине през излъчвания от светодиода сноп светлина и по този начин се намали или прекъсне интензитета на светлината, приемана от рецептора, което предизвиква промяна на положението на изхода. В зависимост от модела фотоелектрически датчик, излъчваната от светодиода светлина е инфрачервена, червена или зелена видима светлина. Предимството на светодиодите е много краткото време за реакция. За да неутрализира влиянието на околната среда, електрическият ток, преминаващ през светодиода, е модулиран такам че да се получи импулсен електрически поток.

3.1.2. Вход за фоторастеров преобразувател (инкрементален датчик + симулатор)

Фиг. 2.6. Фотоелектрически растеров преобразувател

Фотоелектрическият растеров преобразувател (фигура 2.6) служи за преобразуване на ъглови премествания в електрически импулси. Има четири изходни сигнала A, B, C и $\overline{C}$. Изходните електрически сигнали A и B са дефазирани на 90° и позволяват определяне на посоката на въртене например на оси. Изходният електрически сигнал C (съответно $\overline{C}$) представлява кратък единичен импулс за едно пълно завъртане на оста на фотоелектрическия растеров преобразувател. Този тип датчик намира широко приложение като средство за азвтоматизация на технологични линии и процеси.

На макет СКС е реализирана електрическа схема с възможност за използване на фотоелектрически растеров преобразувател като датчик (свързан към съединител JP5) или възможност за симулиране на неговата работа, използвайки специализирана схема, генерираща импулси.

3.1.3. Симулиране на сигнал на аналогов вход

Както бе споменато по-горе, програмируемият контролер работи на базата на постъпилата на неговите входове информация. Освен с дванадесетте цифрови входа, контролерът разполага и с един аналогов вход, както и с възможност за допълнително увеличаване на броя на аналоговите входове. Вграденият и допълнителните (разширителни) аналогови входове са предназначени за работа със сензори за температура, термодвойки и резисторни сензори. В контролера е вграден 9-битов аналогово-цифров преобразувател (АЦП), който преобразува информацията, подадена на входа му, в число от 0 до 511. Така при измерване на температура числото 0 (нула) отговаря

на най–ниската, а числото 511 - на най-високата измерена температура. На Макет СКС е реализирана схема на електронен термометър с използване на сензор тип “Pt 100” (платина 100), който се свързва със съединител JP1 (фигура 2.2).

Принципът на работа на тази група сензори се основава на зависимостта на съпротивлението на металите от температурата. Сензори, изработени на основата на платина (Pt), са известни като индустриален стандарт, поради много добрите им качества. Другите (на базта на мед (Cu) и никел (Ni)) се използват, когато изискванията за точност не са много високи, поради по-ниската им цена. Терморезисторите от платина (Pt) се характеризират със:

- съпротивление 100Ω при 0^0C (има 1000Ω, 2000Ω, ...) и 138,5Ω при 100^0C;
- положителен температурен коефициент (TKR) 0,38-0,40 % на градус C;
- слаба нелинейност, в тесен обхват е по-малка от $0,1^0$C;
- много добра повторяемост на параметрите, позволяваща взаимозаменяемост;
- стабилност във времето и устойчивост на външни въздействия.

Тези параметри на резисторите определят и изискванията към електронната апаратура, с която ще се извършва измерването. Повечето електронни схеми са мостовите схеми, при които лесно се отделя началният сигнал (съответстващ на 100Ω) като в изхода остава само изменението. Именно такава е схемата, реализирана в Макет СКС.

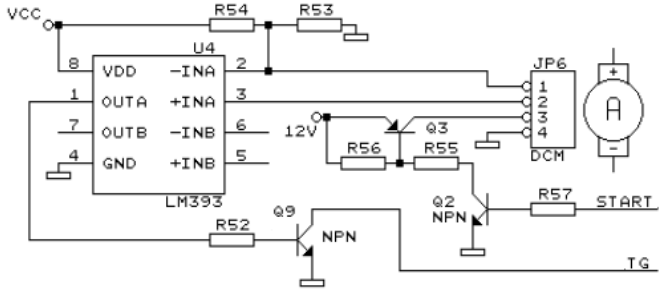
4. Изходи

Изходите на контролера представляват модулът, който подава сигнали към изпълняващите устройства в зависимост от информацията, постъпила на входовете и резултата от изпълнената програма на контролера. Програмируемият логически контролер разполага с осем изхода от тип “transistor source”. Също както за входовете, така и за изходите на Макет СКС е реализирана електронна схема, която да показва състоянието на съответния изход.

Аналогично на входовете – всички изходи на контролера се свързват с макета посредством съединителя HE10. Но за разлика от цифровите входове, всички изходи могат да бъдат използвани като специализирани на макет СКС.

4.1. Блок за управление на правотоков електродвигател с вграден тахогенератор

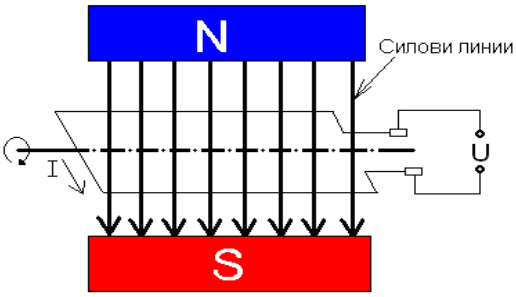
На макет СКС е реализирана електрическа схема (фигура 2.7) за управление на правотоков двигател (фигура 2.8) с помощта на сигнал от специализиран за целта изход.



Фиг. 2.7. Електрическа схема

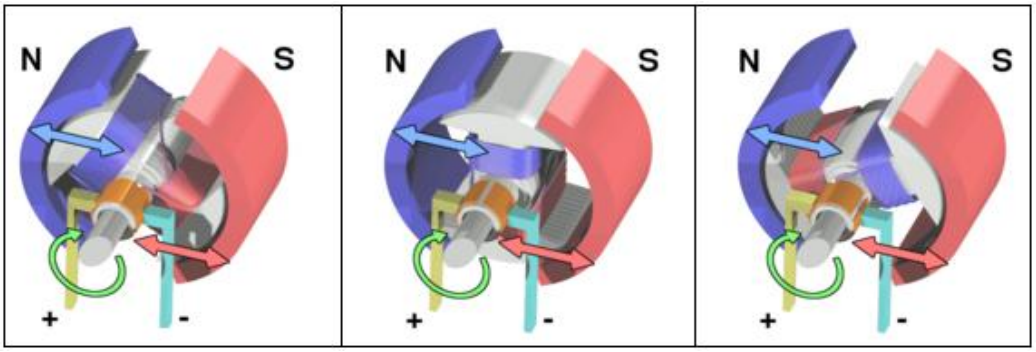
Фиг. 2.8. Правотоков електродвигател с тахогенератор

Правотоковият електродвигател е един от първите създадени електрически двигатели. Работата му се основава на генерираното магнитно поле от преминаващ ток по проводник (фигура 2.9).



Фиг. 2.9. Основна идея за работата на правотоков електродвигател

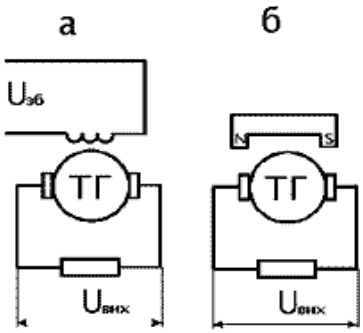
Съвременните правотокови двигатели представляват съвкупност от много правоъгълни рамки от електрически проводник, които се движат в магнитно поле, създадено от постоянни магнити (фигура 2.10).



Фиг. 2.10. Принцип на работа на правотоков електродвигател

Когато става въпрос за управление на двигатели, най-лесен за управление е постояннотоковият двигател – той се върти с обороти пропорционални на подаденото напрежение. При натоварване, поради увеличения консумиран ток, подаваното напрежение спада вследствие на пада на напрежението в бобината му (котва). Ако регулаторът на напрежение, с което се управлява двигателят, има отрицателно изходно съпротивление, равно по абсолютна стойност на съпротивлението на котвата, оборотите ще са стабилни, независимо от натоварването. Когато е необходима по-голяма точност, се използват схеми с обратна връзка, а като източник на такава обратна връзка се използва например постояннотоков тахогенератор (фигура 2.11).

Фиг. 2.11. Тахогенератор



Фиг. 2.12. Видове тахогенератори

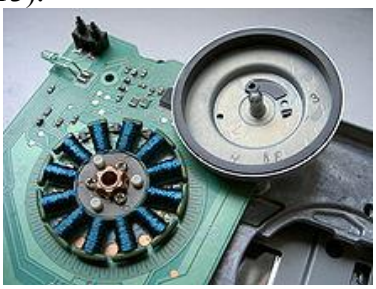
Тахогенераторът изработва напрежение с честота пропорционална на оборотите. Постояннотоковият тахогенератор конструктивно представлява обикновен постояннотоков микродвигател с независимо електромагнитно възбуждане (фигура 2.12а) или с възбуждане от

постоянни магнити (фигура 2.12б). Той има две бобини — възбудителна и котвена, изместени в пространството на 90° електрически една спрямо друга. Към възбудителната (полюсната), която е в статора, се подава възбудителното напрежение, а от изходната (котвената), която е в ротора, се снема изходното напрежение.

Запускането и управлението на този вид електродвигатели става по точно определен начин. За започване на работа на двигателя се подава сигнал “Start” на изход Q3 (фигура 2.7), а сигналът от тахогенератора се връща към контролера по шината “TG”.

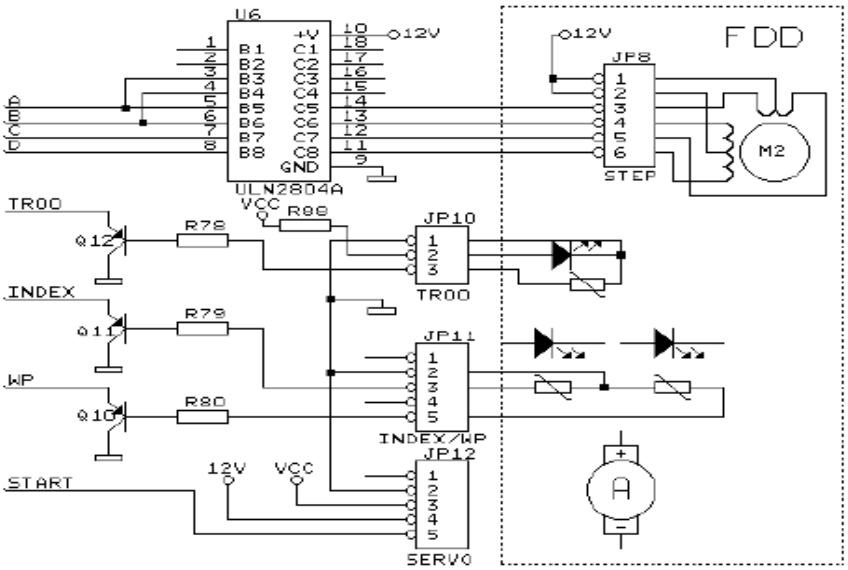
4.2. Блок за управление на правотоков електродвигател с вграден регулатор

Както бе споменато в предходната точка, най-лесен за управление е постояннотоковият двигател, който се върти с обороти пропорционални на подаденото напрежение. Конструкцията и предназначението е аналогично с това на обикновенните двигатели. Както подсказва името на този блок, той на практика управлява вградения в електродвигателя най–често регулатор на напрежение. В зависимост от принципа и предназначението си съществуват много и различни модели електро двигатели с вграден регулатор. Тъй като целта на регулацията е постигането на определен брой обороти в минута – електродвигателят разполага със специализиран датчик (датчик на хол, тахогенератор и др.) чрез който да „каже” на управляващата го схема с колко точно оборота се движи (фигура 2.13).



Фиг. 2.13. Правотоков електродвигател използващ датчик на Хол

На макет СКС е реализирана схема за управление на правотоков двигател с датчик на Хол (шпиндел на флопи-дисково устройство – фигура 2.14).



Фиг. 2.14. Схема за управление на комплексен модул „Флопи-дисково устройство”

Както се вижда от схемата, към електродвигателя са подадени две напрежения (VCC = 5V и 12V), като за запускането на двигателя е необходимо да се подаде сигнал на проводника „Start”. Управлението на този двигател, в комбинация с оптодатчиците и стъпковия двигател с обратна

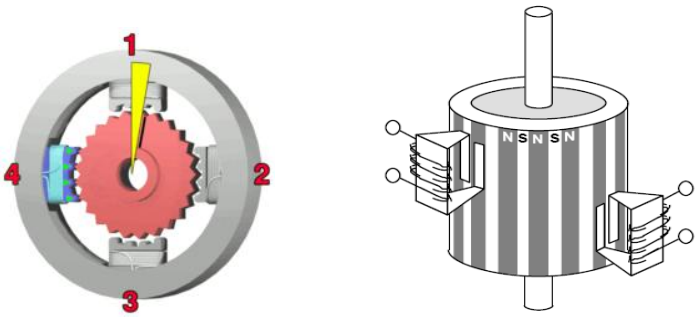
връзка, заедно позволяват управление на комбинирания модул „Флопи-дисково устройство”. Ролята на електродвигателя във флопи–дисковото устройство е да задвижва магнитния носител на съответната дискета, като поддържа постоянни обороти без външна намеса.

4.3. Блок за управление на стъпкови електодвигатели (с и без средна точка)

Освен обяснените по-горе правотокови електродвигатели, съществуват още много различни типове електродвигатели. Такива например са стъпковите електродвигатели (фигура 2.15).

Фиг. 2.15. Стъпков електродвигател

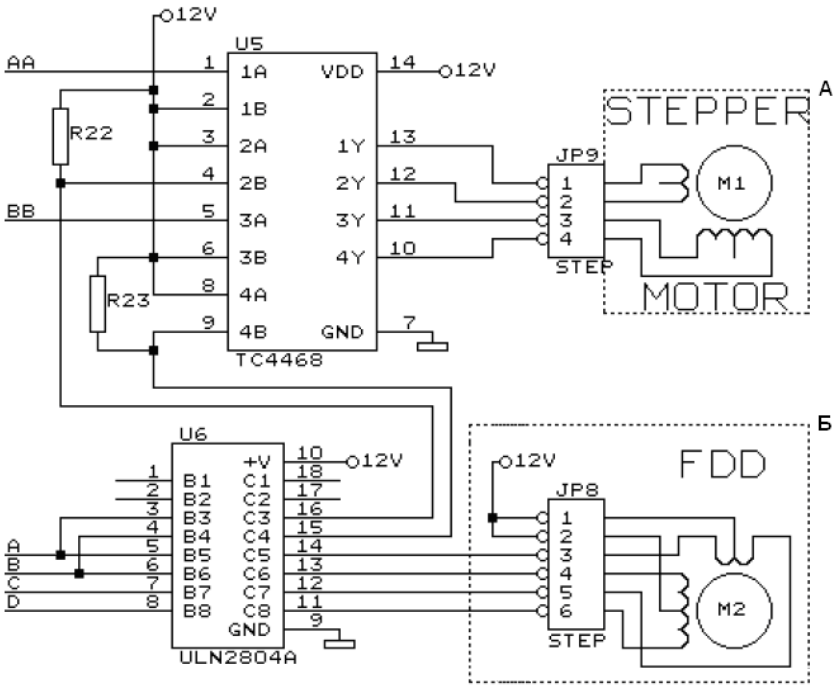
За разлика от правотоковите двигатели, стъпковите двигатели имат коренно различна конструкция (фигура 2.16) и начин на управление. Стъпковите двигатели спадат към групата на синхронните електрически машини.



Фиг. 2.16. Конструкция на стъпков електродвигател

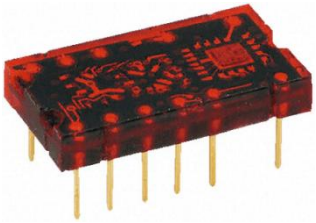
Подобно на обикновените електродвигатели - стъпковият двигател се състои от статор и ротор. На статора са разположени четири бобини под ъгъл 90^0 една от друга, а роторът се състои от назъбен магнитен материал. Характерно за стъпковия електродвигател е, че извършва пълно завъртане, разделено на много стъпки, откъдето идава и наименованието му. Той може да бъде правилно регулиран без обратна връзка, но изисква специализирана външна електрическа схема, за да се задвижва. Такава схема е реализирана на макет СКС. Стъпковите електродвигатели могат да се класифицират според начина и броя изводи за управление.

На следващата фигура 2.17 е показана електрическата схема за управление на стъпкови електродвигатели със и без средна точка, реализирана на макет СКС:



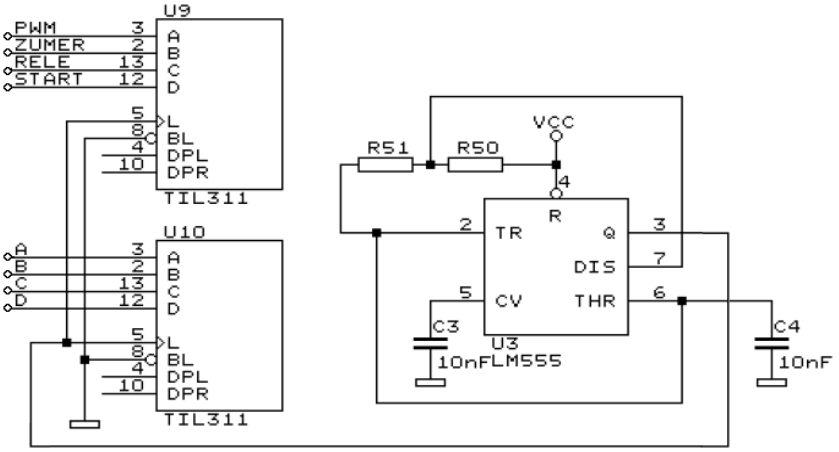
Фиг. 2.17. Електрическа схема за управление на стъпкови електродвигатели
(А – без средна точка; Б – със средна точка)

4.4.. Блок за управление на LED индикация с вграден дешифратор (фигура 2.18)

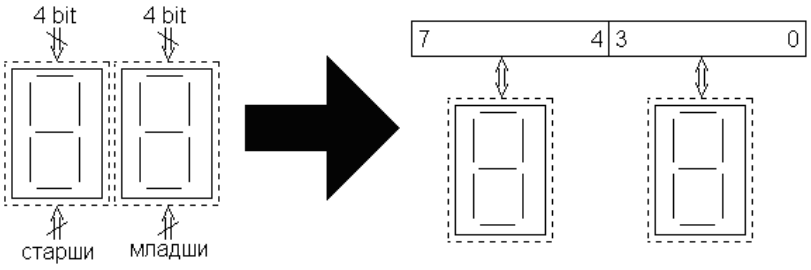


Фиг. 2.18. LED индикатор с вграден дешифратор

На макет СКС е реализирана схема (фигура 2.19) за управление на шестнадесетична LED индикация с вградена TTL схема (TIL311), която дава възможност елементът да приема, съхранява и показва четири битови двоични данни. Основната дейност, която изпълнява, е да изобразява информация в двоично–десетичен код (фигура 2.20).



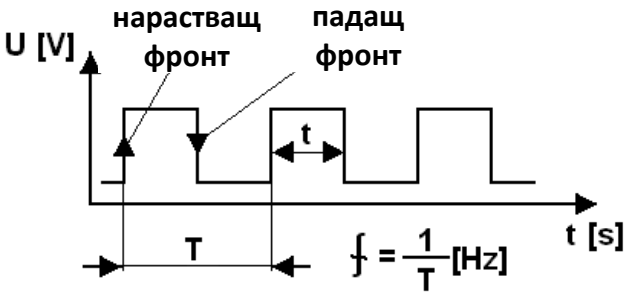
Фиг. 2.19. Схема на свързване на TTL311



Фиг. 2.20. Изобразяване на информация в двоично-десетичен код

4.5. Блок за управление на „широчинно – импулсен генератор” (PWM)

Импулс – може да се опише като рязка промяна на някоя от характеризиращите даден сигнал величина. Съществуват различни начини за модулиране (внасяне на промяна) в импулсните сигнали. Един от начините да се модулират импулсни сигнали е широчинната модулация. PWM - Pulse-width modulation или Широчинно–Импулсна Модулация (ШИМ) представлява метод за генериране на импулси с различен коефициент на запълване на периода. На следващата фигура 2.21. са показани основните параметри на сигнал с правоъгълни импулси.



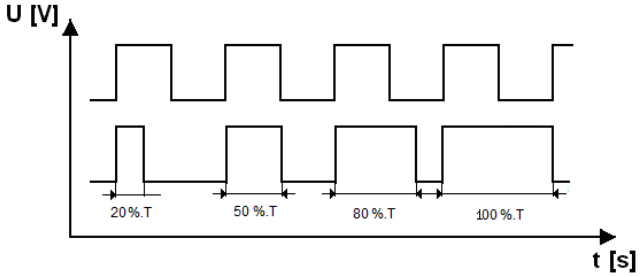
Фиг. 2.21. Параметри на правоъгълен импулсен сигнал

T – период на повторение на импулсите;

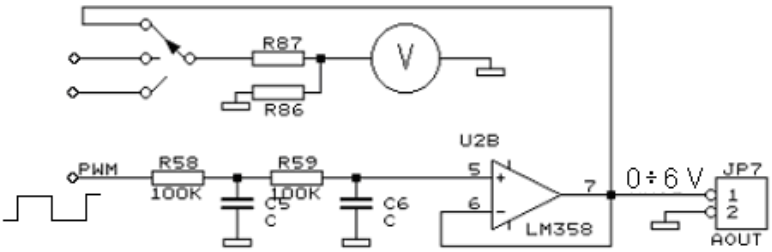
f – честота на повторение на импулсите;

t – продължителност на импулсите.

На фигура 2.22 са показани нормален импулсен сигнал и модулиран (на 20, 50 и 80% и немодулиран 100%.T). На фигура 2.23 е показана електрическата схема на реализирания върху макет СКС широчинно–импулсен генератор, свързан с цифров волтметър.



Фиг. 2.22. Времедиаграма, показваща импулси с различен коефициент на запълване

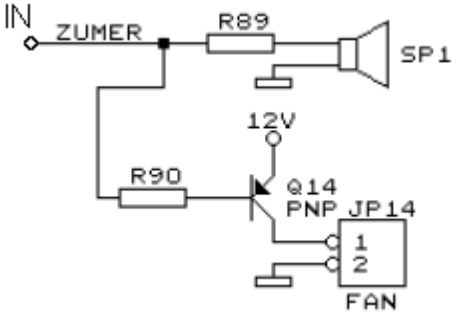


Фиг. 2.23. Електрическа схема на широчинно–импулсен генератор, свързан с цифров волтметър

4.6. Блок за управление на звукова сигнализация „Зумер”

Фиг. 2.24. Зумер

Зумерът (фигура 2.24) представлява устройство, изработено от пиезоелектрик, генериращо звуков сигнал. Той е базиран на обратния пиезоелектричен ефект. Пиезоелектричният ефект е ефект на възникване на поляризация на диелектрик под въздействие на механично напрежение (нарича се пряк пиезоелектричен ефект). Съществува и обратен пиезоелектричен ефект - възникване на механични деформации под въздействие на електрическо поле. При зумера тази деформация генерира звук. На макет СКС е реализирана схема за управление на зумер (фигура 2.25), като се използва сигнал от реализирания на макета широчинно–импулсен генератор. В зависимост от широчината на подаденият импулс се променя честотата на генерирания звук.



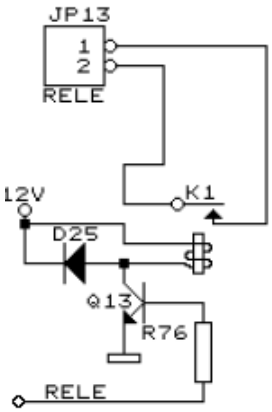
Фиг. 2.25. Схема за управление на зумер

4.7. Блок за управление на реле

Фиг. 2.26. Реле

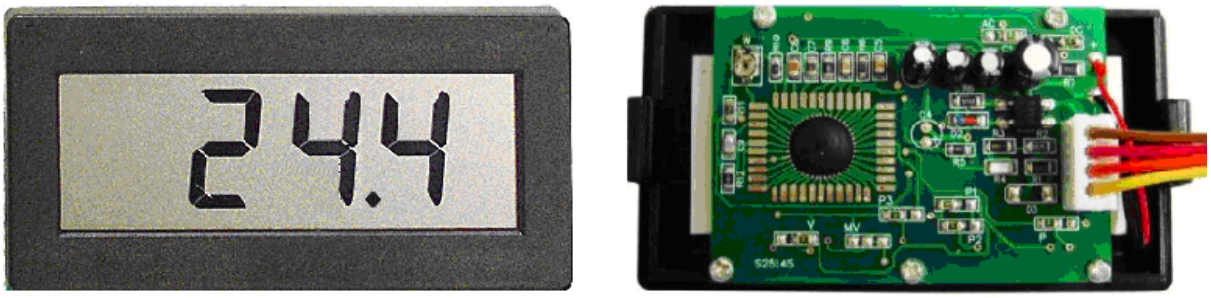
Релето (фигура 2.26) е електромагнитен превключвател, предназначен за комутация на електрическа верига с цел скокообразно изменение на изходната величина при зададено изменение на входната величина. Широко се използва в различни автоматични устройства. По принципа си на действие се различават електрически, пневматични, механични релета, но най-разпространени са електрическите (електромагнитните) релета. Основните части на електромагнитното реле са електромагнит, котва и превключвател. Електромагнитът представлява електрически проводник, навит на бобина със сърцевина от феромагнитен (магнитно мек) материал. Котвата е пластина от магнитен материал, която чрез лостче управлява контактите. При пропускане на електрически ток през бобината на електромагнита възниква магнитно поле, което притегля котвата към сърцевината и тя превключва контактите. Самите превключватели могат да бъдат прекъсващи, свързващи или просто превключващи.

На макет СКС е реализирана електрическа схема, с която е възможно управляването на електромагнитно реле. Електрическата схема е показана на фигура 2.27. Към съединител JP13 се свързва товарът, който ще се управлява.



Фиг. 2.27. Електрическа схема за управление на реле

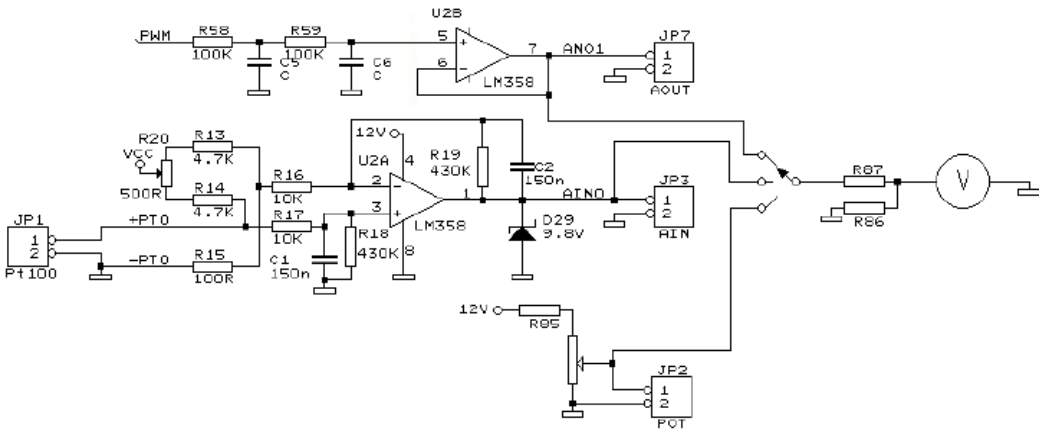
4.8. Цифров волтметър



Фиг. 2.28. Цифров волтметър – дисплей и електронна схема

Волтметърът е уред за измерване на електрическо напрежение. Той се включва в електрическа верига паралелно на консуматора и има голямо вътрешно съпротивление. Обхватът му се разширява чрез последователно свързан добавъчен резистор, делители на напрежение или напрежителни измерителни трансформатори. Съществуват различни видове волтметри, като един от признаците, по които могат да бъдат разграничени, е техният тип – аналогови и цифрови.

На макет СКС е изведен извод за връзка с цифров волтметър, който посредством ключ SW13, може да бъде подключен към схемата за управление на аналогов вход, към широчинно–импулсния генератор и за измерване на захранващото напрежение на макет СКС (фигура 2.29). Използваният цифров волтметър (фигура 2.28) е готов модул с четири извода, два от които за захранване - 9V батерия и два за измерване.

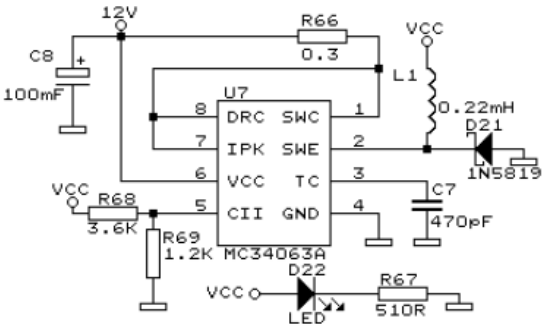


Фиг. 2.29. Схема на свързване на волтметър към ШИМ, аналогов вход и основно захранващо напрежение

5. Захранващ модул

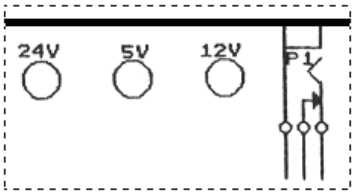
Захранващият модул е един от най–важните модули в една електронна система, тъй като осигурява необходимото захранване за правилна работа на системата. Най-често използваните вътрешни напрежения са +5V, ±12V и +24V. Макетът за симулация се захранва от модул, разделен на две части. Едната част - това е AC – DC конверторът, който представлява отделен елемент, разположен в собствена кутия и чиято основна цел е да преобразува напрежението от захранващата мрежа в удобно такова за ползване от макета за симулации (фигура 2.30). Използвани са готови модули – SYS1308–2412–W2E – за получаване на 12V и 5V, както и SYS1308–2424–W2E – за получаване на 24V. Втората част – това е стабилизаторът (фигура 2.31), който е реализиран физически на макета за симулации. Неговата основна цел е да гарантира постоянно и стабилно захранващо напрежение.

Фиг. 2.30. Електрически трансформатор



Фиг. 2.31. Електрическа схема на стабилизатора

Макетът разполага със съединител за свързване на захранването и три светодиода за показване наличието на различните напрежения - 5V, 12V и 24V (фигура 2.32).



Фиг. 2.32. Индикаторни светодиоди и съединител

Програмна среда TWIDO SUITE

1. Въведение

За реализирането на дадена задача всички компоненти на системата трябва да бъдат правилно сързани и подготвени за работа, като се спази следният ред:

- подготвяне на програмиращото устройство (инсталиране на необходимия софтуер и запознаване с методите за работа с него);
- свързване помежду си на отделните модули на контролера (процесорен модул, разширителен модул за аналогови входове и т.н.);
- свързване на кабелите за комуникация (от и към контролера) и захранване към контролера;
- включване на захранването на контролера и запускане на програмиращият софтуер (*след като бъде подадено захранването към контролера е необходимо да се изчака кратък период от време, за да може контролера да влезе в режим на готовност).

След като контролерът е „влязъл” в режим на готовност, следва неговото програмиране по конкретното задание. Цикълът на програмиране се състои от няколко основни етапа:

- стартиране на програмиращият софтуер;
- установяване на необходимите настройки в удобен за потребителя режим (ако е необходимо);
- изготвяне на приложното програмно осигуряване;
- записване в паметта на контролера;
- тестване, отчитане на получените резултати и, ако е необходимо, внасяне на корекции.

Програмната среда TwidoSuite е предназначена за създаване на проекти (хардуерни конфигурации и приложни програми), базирани на контролери от серията Twido. Разработена е и се предоставя от производителя на контролерите от марката Twido – Schneider Electric.

2. Стартиране на TwidoSuite

Програмата може да се стартира чрез TwidoSuite Application Launcher:

- от **Start → Programs → Schneider Electric → TwidoSuite → Application Launcher**
- или от работния плот **Application Launcher** пряк път, който е бил инсталиран с програмата.

Application Launcher представлява център за стартиране на TwidoSuite. TwidoSuite приложението се предлага в няколко езикови двойки: **Английски-Френски; Английски-Немски; Английски-Испански; Английски-Италиански; Английски-Китайски.**

В зависимост от двойката езици, които са инсталирани, то позволява да се стартира TwidoSuite на английски език (език по подразбиране) или на друг инсталиран език. Приложението за стартиране на TwidoSuite се използва за стартиране в няколко режима:

- **Programming Mode (режим на програмиране);**
- **Monitoring Mode (режим на мониторинг);**
- **PLC Firmware Update (обновяване софтуера на контролера).**

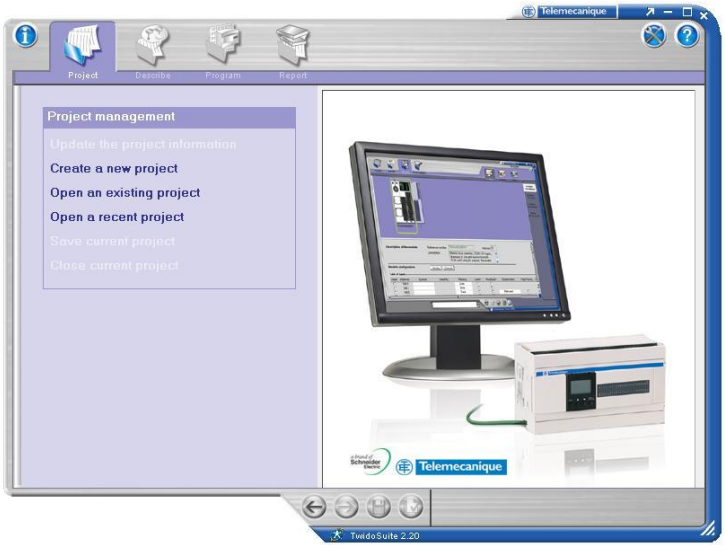
Фигура 3.1 показва как изглежда **TwidoSuite Application Launcher** на английски-френски език:

Фиг. 3.1. Начална страница на Twido Suite

3. Интерфейс на TwidoSuite

За преход към програмиране е достатъчно да се избере съответния режим. Това действие ще доведе до отваряне на основния прозорец на TwidoSuite (фигура 3.2). В горната част на този прозорец са разположени бутоните за превключване в различните режими на работа:

- Project — параметри на проекта
- Describe — описание на оборудването
- Program — непосредствено създаване на управляваща програма
- Report — формиране на отчет (документиране на програмата)

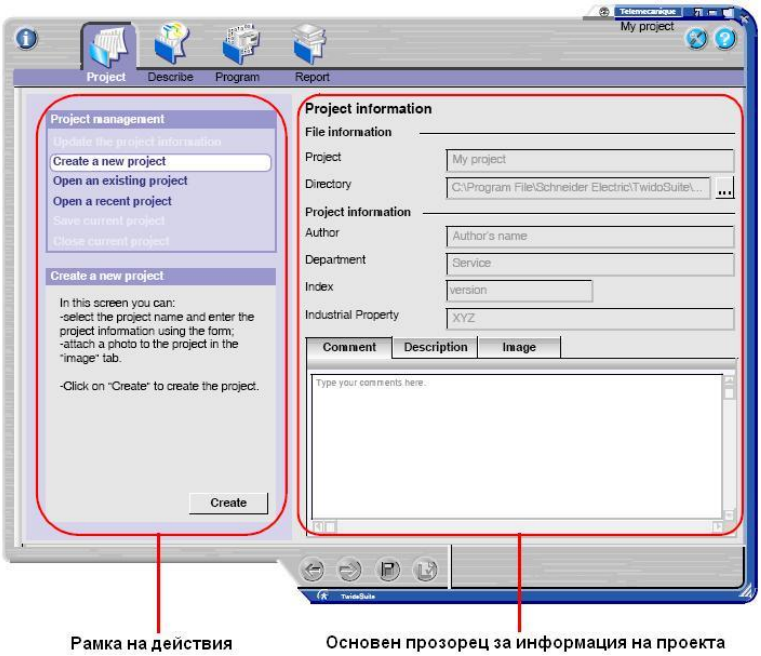


Фиг. 3.2. Основен прозорец на TwidoSuite

Също вдясно горе са разположени допълнителни бутони за настройка на програмата - Preference и описанието Miscellaneous. Средната част на прозореца е за работната зона и може да се раздели на няколко подобласти. В долната част има малък панел за елементите на навигацията, работата с файлове, а също и за специалните инструменти, използвани в различните режими на работа.

Както и в други подобни програми, работата започва със създаването на група от файлове под название Project. Създаването на нов проект се извършва в едноименния режим, където в меню Project Management трябва да се избере Create a new project (фигура 3.3). Вдясно се

появява панел с информация за проекта (Project Information), в който следва да се зададе името и разположението на папката на проекта, а също и авторство, кратко описание и т. н. След това стават достъпни всички останали режими на работа на програмата.



Фиг. 3.3. Създаване на нов проект

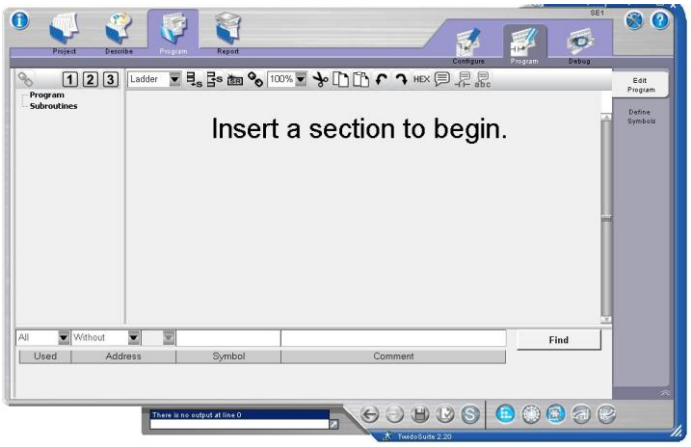
Лявата част на прозореца представлява работното поле, върху което се поставят необходимите елементи. Вдясно е разположен каталогът на достъпните модули. От този каталог трябва да се избери необходимия модел и чрез бутона Place да се постави в работното поле. Под каталога програмата извежда изображение на избрания модул и негово кратко описание.

След задаването на конфигурацията на оборудването може да се премине към описание на входовете и изходите. За целта се избира добавката Program, в която се появяват още три допълнителни пункта (фигура 3.4):

- Configure —конфигуриране на променливите;
- Program — прозорец за разработване на програмата;
- Debug .

Също вдясно се отварят и допълнителни режими за всеки от пунктовете.

В режим Configure на потребителя се предлага да зададе имена на използваните входове и изходи, на константите и друга описателна информация. По принцип това не е задължително, но винаги е по-удобно да се работи със смислени имена, отколкото с машинни променливи. Също при създаването на описанията има възможност да се настроят някои параметри, отнасящи се до вградените елементи (функционални блокове) на контролера – таймери, броячи и др.



Фиг. 3.4. Режим Program

Режимът Program е предназначен за създаване на Ladder диаграма. В този режим се отваря прозореца на редактора и няколко панели с инструменти, съдържащи необходимите елементи за схемите.

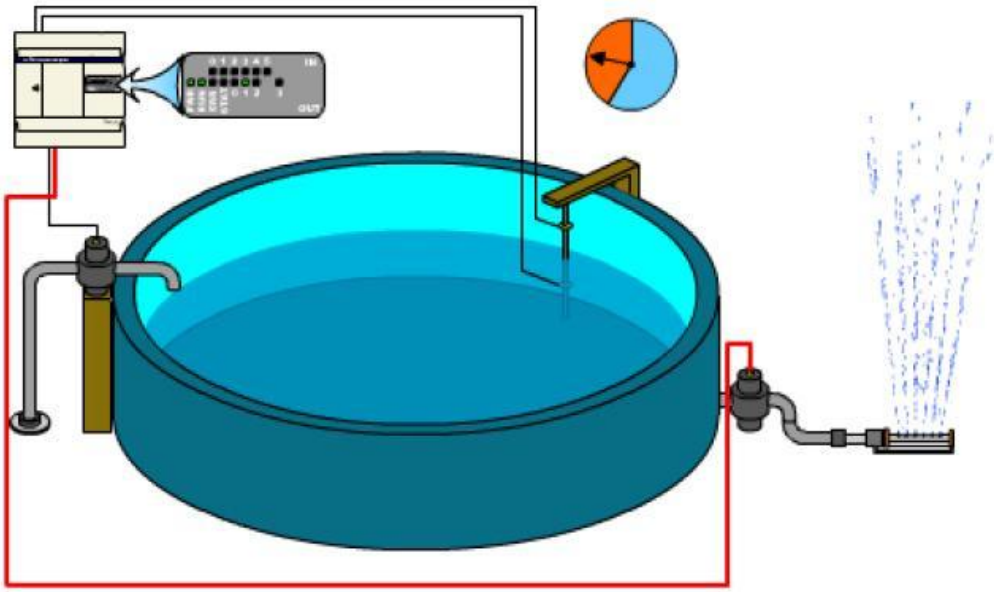
В долната част на прозореца, вдясно са разположени няколко бутона. Те позоляват да се съхрани програмата, да се импортира или експортира в текстов файл. Най-полезен е бутонът Simulate, чрез който се запуска виртуален контролер и така става възможно да се проследи работата на програмата, без да се зарежда в контролера.

След създаването на програмата, тя може да се зареди в контролера, използвайки менюто Debug. По принцип Twido Suite може сам да намери канала за връзка с ПЛК при първото включване, но ако е необходимо, може да се използва възможността за настройка Manage Connection, бутонът за която е разположен в долната част на екрана.

Разбира се, това е много кратко описание на интерфейса, което има за цел първоначално запознаване с програмната среда.

4. Пример за създаване на проект в TwidoSuite

Изясняването на процеса на създаване и тестване на програма в средата TwidoSuite, ще бъде разгледано на базата на пример за управление нивото на водата на резервоар за напояване (фигура 3.5)



Фиг. 3.5. Система за автоматично напояване

Създаването на работещ проект протича в следните стъпки:
Стъпка 1: Създаване на нов проект;
Стъпка 2: Описание на приложението;
Стъпка 3: Определяне на входно-изходни параметри за окабеляване на приложението;
Стъпка 4: Конфигуриране на хардуерните и софтуерните ресурси;
Стъпка 5: Писане на програма;
Стъпка 6: Проверка на програмата;
Стъпка 7: Зареждане на програмата в контролера и стартирането й;
Стъпка 8: Управление на програмния доклад (документация).

Стъпка 1: Създаване на нов проект

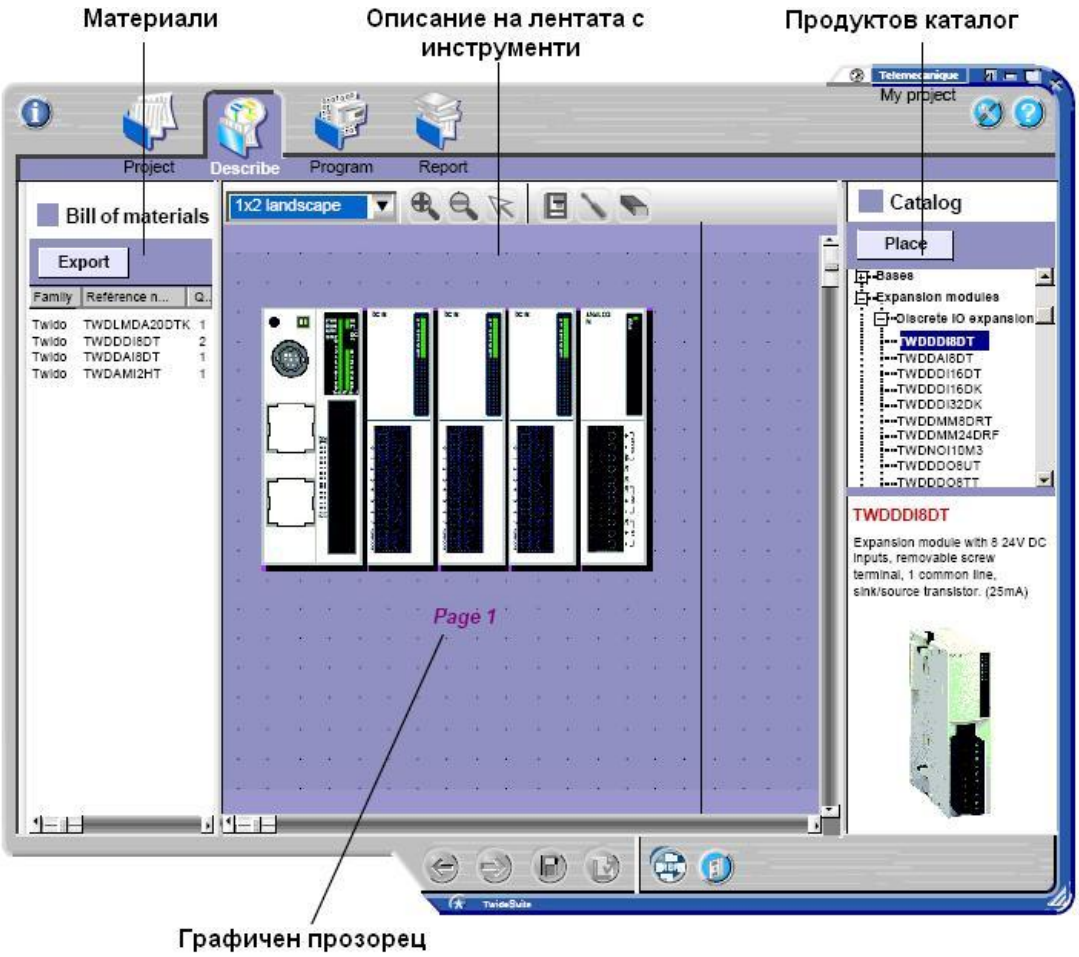
Стартира се иконата на проект в TwidoSuite в работното пространство от лентата с менюта, за да се покаже прозорецът (фигура 3.3) за създаване на нов или отваряне/редактиране на съществуващ TwidoSuite проект.

1.Кликва се върху **Create new project (Създаване на нов проект)** в "Рамка на действия".
2.Въвежда се описателно име на проекта в **рамката на файловата информация**. Валидното име на проекта може да съдържа до 36 букви и цифри.
3. Задава се нов път на директорията на проекта или се използва пътя по подразбиране.
4. Въвеждане на авторска информация на проекта в **рамката на информация на проекта**.
5. Разделът **Comments (Коментари)** включва името на проекта и позволява да се напише кратко описание.
6. Избира се бутон **Create (Създай)**, за да се запазят промените в новия проект.

Забележка: Ако се обновява информация за съществуващ проект, се избира **Apply (Приложи),** за да се запазят промените.

Стъпка 2: Описание на приложението

Избира се иконата **Describe** на TwidoSuite в работно пространство от лентата с менюто, за да се покаже прозореца за описание. В него се декларират елементите на системата за автоматизация и се избират хардуерните ресурси от каталога (фигура 3.6).



Фиг. 3.6. Прозорец за описание

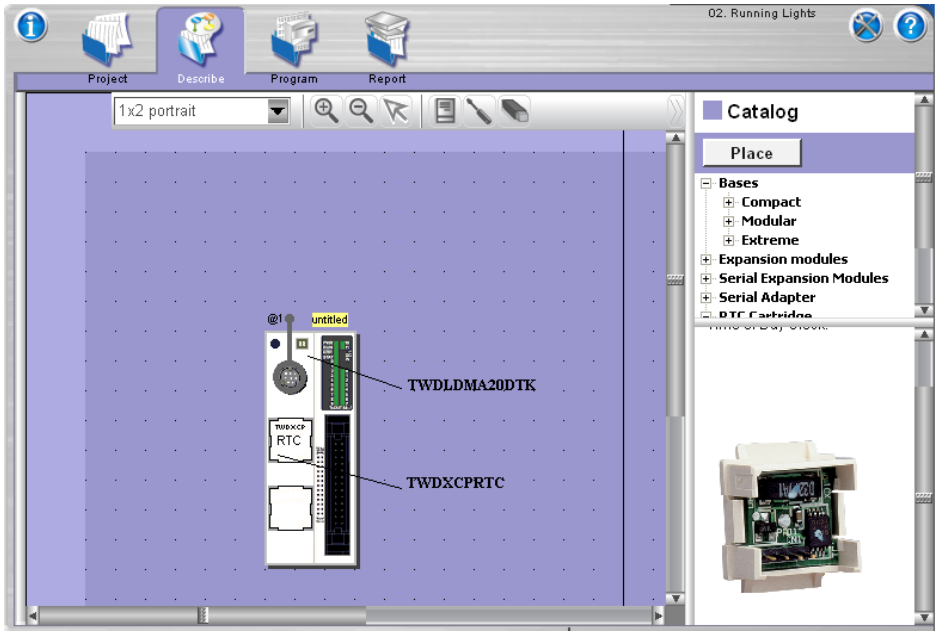
Елементи на прозореца за описание (фигура 3.7):

- **Графичен прозорец**: Това е мястото, където се изгражда графичното представяне на приложенията чрез добавяне на хардуерни компоненти. Чрез плъзгане се избират хардуерни елементи от каталога на графичния прозорец.
- **Лента с инструменти -** управление на графичен екран и елементи.
- **Продуктов каталог:** Този екран показва каталога на хардуерните елементи в Twido, в това число **базовите** – компактни, модулни, разширителни модули, опции, мрежови елементи в дървовидна структура. В долната част на прозореца се предоставя кратко описание на избрания хардуерен елемент.

Чрез използване на влачене и пускане от каталога на елементи в графичния прозорец, първо се избира базовия контролер (компактен или модулен), след което следва добавянето на разширителни модули и възможности.

В този пример трябва:

1. Да се постави TWDLMDA20DTK компактен базов контролер в описващия прозорец;
2. Да се добави RTC (часовник за реално време) вариант на TWDXCPRTC.



Фиг. 3.7. Прозорец за описание

Стъпка 3: Определяне на входно-изходни параметри

Преди да се пристъпи към съставянето на програмата, трябва да се определят входно-изходните спецификации на приложението. По-долу са показани входно-изходните спецификации за автоматична поливна система:

Входни спецификации

- %I0.0 вход от сензор за високо ниво на вода в резервоара
- % I0.1 вход от сензор за ниско ниво на вода в резервоара

Изходни спецификации

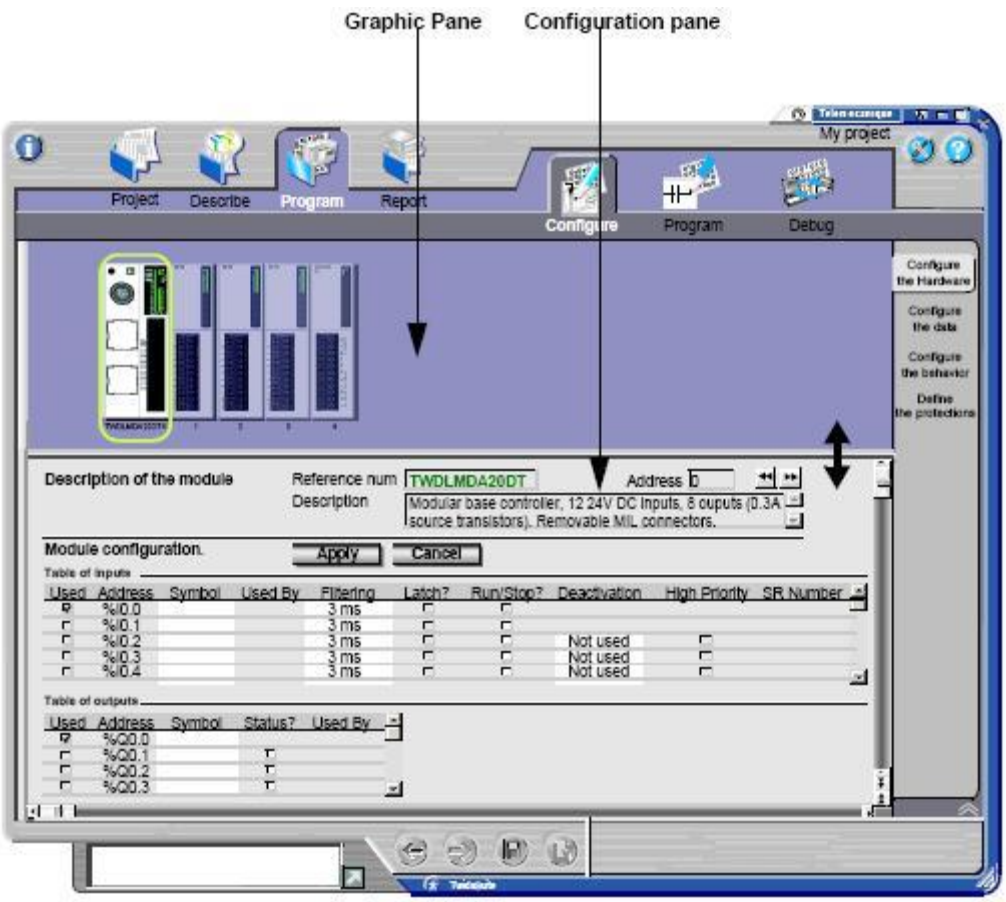
- % Q0.0 изход за задвижване водна помпа
- % Q0.1 изход за контрол на клапаните за напояване

Часовник за реално време - RTC параметри

- RTC модул трябва да бъде инсталиран на Twido контролера.

Стъпка 4: Конфигуриране на хардуерни ресурси

След като са декларирани хардуерните компоненти, трябва да се конфигурират хардуерните и софтуерните ресурси. Избира се: **Program → Configure → Configure the Hardware** (Програма → Конфигуриране → Конфигуриране на хардуер), за да се отвори прозореца на хардуерната конфигурация (фигура 3.8).



Фиг. 3.8. Конфигуриране на хардуерните ресурси

Graphic Pane (Графичен екран): Изображение на контролера и разширенията му

Configuration Pane (Конфигурация): Описание (с конфигурация) на избрания хардуер

След като е избран контролерът, се въвеждат символични имена за входовете и изходите, като например:

- WATER_HIGH_LEVEL, в % I0.0 символно поле
- WATER_LOW_LEVEL, в% I0.1 символно поле
- WATER_PUMP, в% Q0.0 символно поле
- IRRIGATION_VALVE, в% Q0.1 символно поле

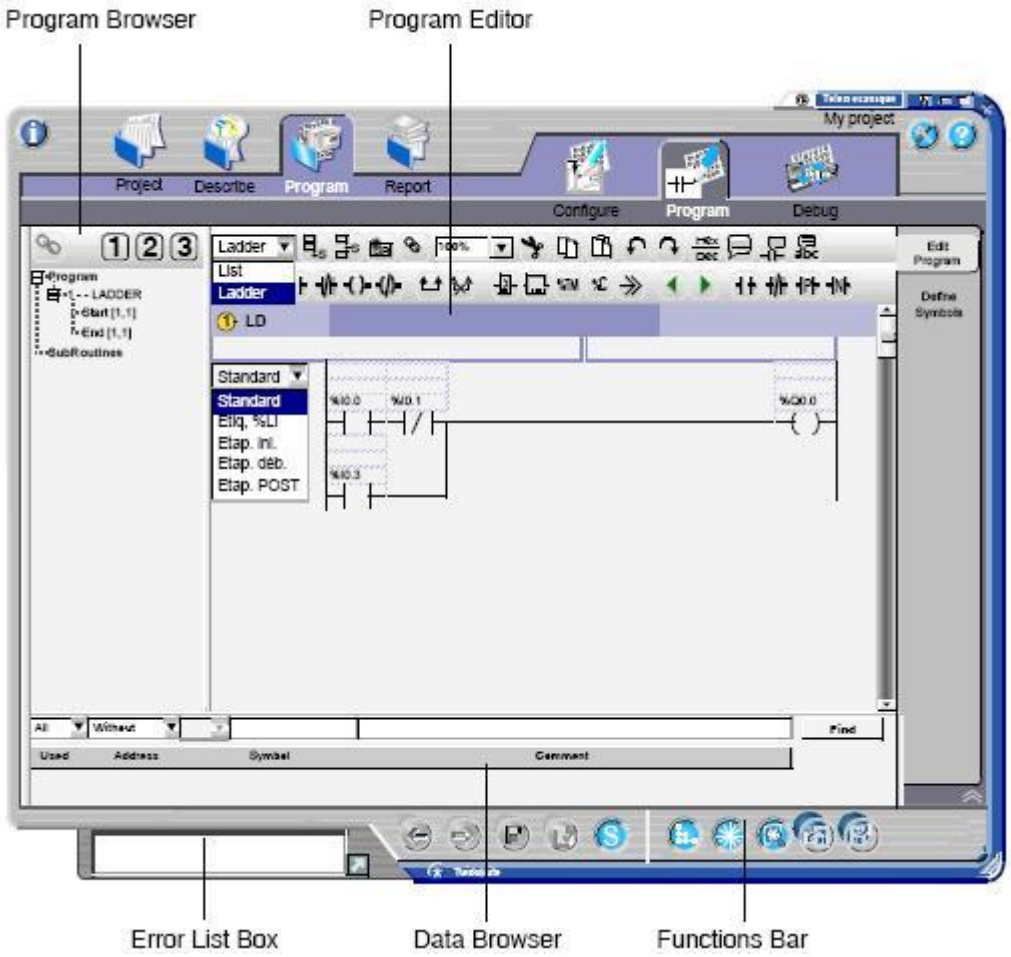
Промените влизат в сила веднага, след като курсорът напусне съответната входна или изходна линия.

Стъпка 5: Писане на програма

Чрез **Редактора на програмата** се разработва основната програма. Основните програмни и конфигурационни функции включват Ladder и списъчен език за програмиране. В този пример ще се използва Ladder езика за програмиране.

За да се достъпи до програмния редактор, се избира:

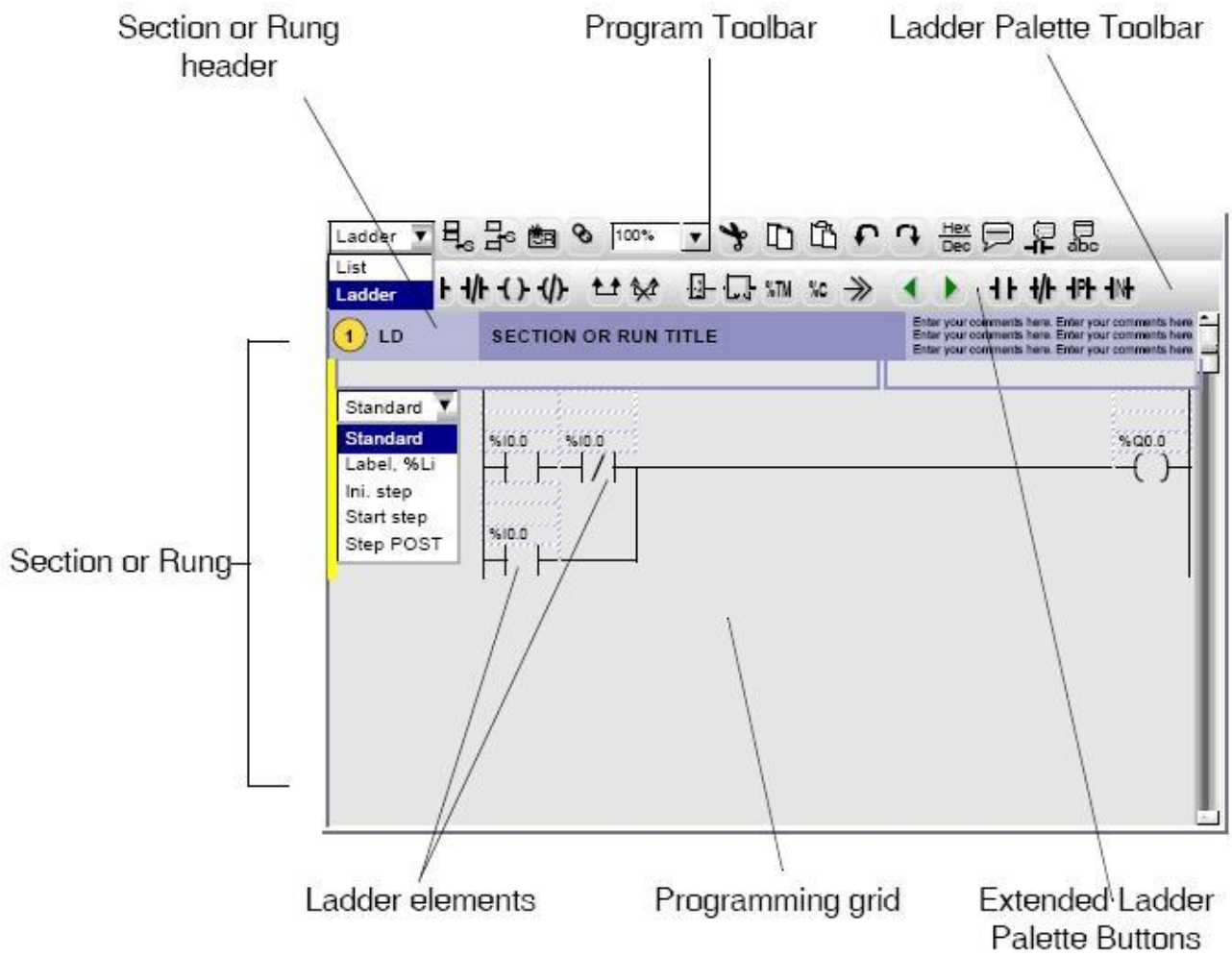
Program → **Program** от TwidoSuite меню и се избира **Редактиране на програма** от лентата със задачи (фигура 3.9).



Фиг. 3.9. Прозорец на програмния редактор

Елементи на Ladder редактора

Ladder Редакторът е графичен програмен редактор, който се използва за създаване и редактиране на Ladder програми. Когато се избере **Ladder** в програмната лента с инструменти от падащото полето, програмният редактор се превключва на режим Ladder. Фигура 3.10 показва различните елементи, от които се състои Ladder редактора:



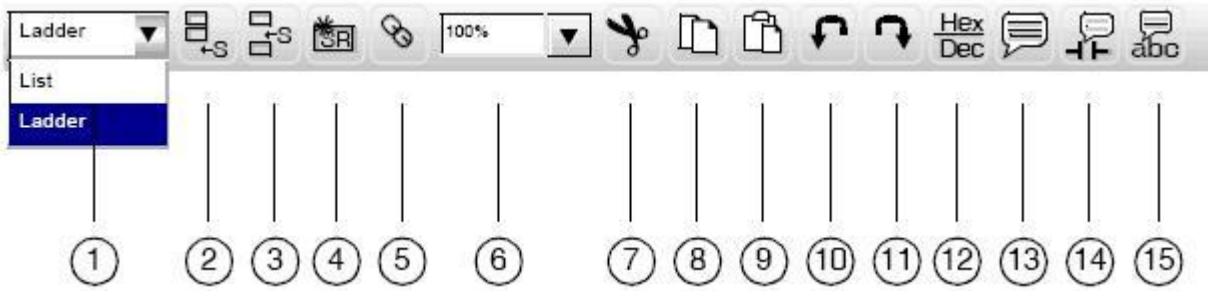
Фиг. 3.10. Елементи на Ladder редактора

Ленти с инструменти за редактиране на програмата

Три Ladder ленти с инструменти за програмиране са на разположение програмиста. Те позволяват лесно редактиране на програми и графично въвеждане на Ladder инструкции:

- Program Toolbar (Програмна лента с инструменти) (фигура 3.11);
- Ladder palette toolbar (Ladder набор от инструменти) (фигура 3.12);
- Extended ladder palette buttons (Разширен Ladder набор от инструменти) (фигура 3.13).

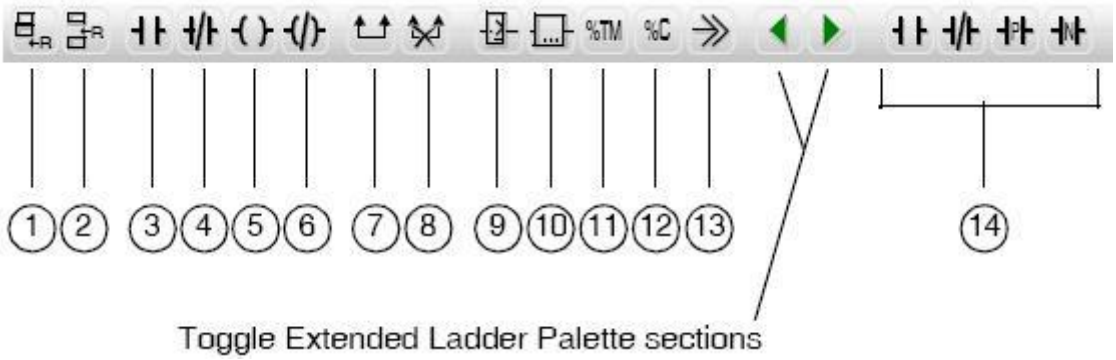
Програмна лента с инструменти



Фиг. 3.11. Програмна лента с инструменти

1. Превключване на Ladder / Списък режим;
2. Добави секция;
3. Вмъкни секция;
4. Добави подпрограма;
5. Изрязване в 2 секции;
6. Мащабиране (по подразбиране е 100%);
7. Изрязване;
8. Копиране;
9. Поставяне;
10. Отменя;
11. Възстановяване;
12. Покажи в шестнадесетична или в десетична бройна система;
13. Покажи / Скрий RUNG коментари;
14. Покажи / Скрий Ladder коментари на елементите;
15. Покажи / Скрий символи.

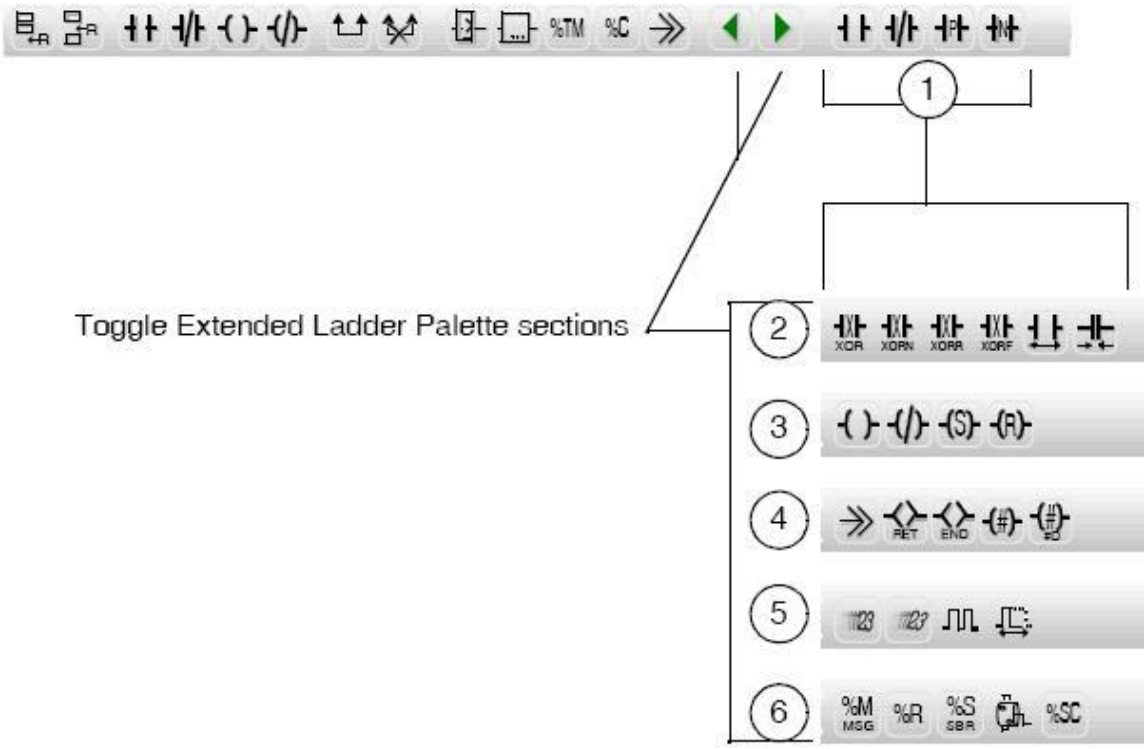
Ladder набор от инструменти



Фиг. 3.12. Ladder набор от инструменти

1. Добави Rung;
2. Постави Rung;
3. Добави нормално отворен контакт;
4. Добави нормално затворен контакт;
5. Добави изход;
6. Добави обратен изход;
7. Добави връзка;
8. Премахване на връзката;
9. Добави блок за сравнение;
10. Добави операционен блок;
11. Добави блок таймер;
12. Добави блок брояч;
13. Добави скок или извикване на подпрограма.

Разширен Ladder набор от инструменти



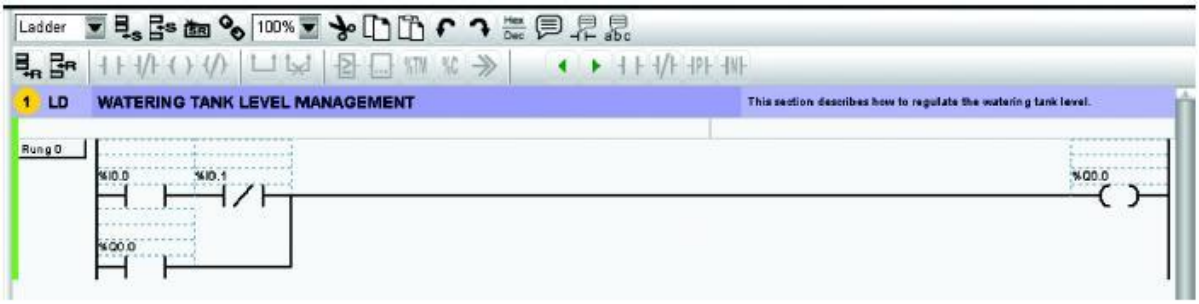
Фиг. 3.13. Разширен Ladder набор от инструменти

1. Контакти: нормално отворен, нормално затворен, нарастващ фронт, падащ фронт;
2. Специални контакти: XOR, XORN, XORR, XORF, OPN, SHORT;
3. Изходи: нормален изход, инвертиращ изход, Set Coil, нулиращ изход;
4. Специални изходи: JMP / SR, RET, END, Grafcet Coil 1, Grafcet Coil 2;
5. Основни функционални блокове: %FC, % VFC, %PLS,% PWM;
6. Разширено функционални блокове:%MSG, %R, %SBR, %DR,%SC.

В примера за автоматичната поливна система има два независими процеса: управление на нивото на водата в резервоара и броя на напояванията. Това изисква да се създадат две секции в програмата, по една за всеки процес.

Секция за управление на нивото на водата в резервоара

Поставя се първата секция на програмата и се създава програмата от фигура 3.14.



Фиг. 3.14. Секция 1 на програмата

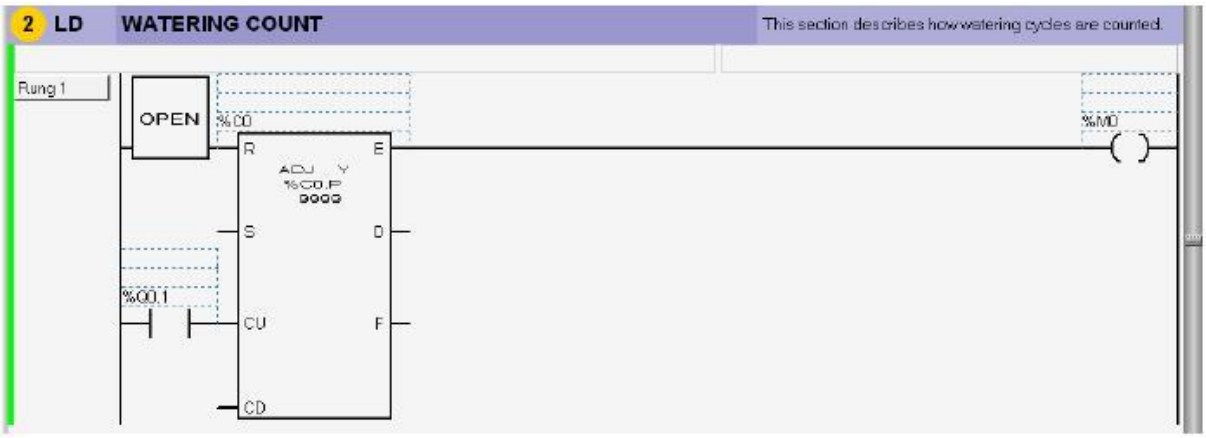
Описание на процеса за управление на нивото на водата в резервоара

Когато е достигнато ниското ниво на водата (% I0.1 е затворен), водната помпа се стартира (%Q0.0 е затворен). %I0.0 остава нормално затворен. Всеки път, когато се достигне високото

ниво на водата (% I0.0 е отворен), водната помпа се спира - изход %Q0.0 се активира и %Q0.0 контакт е затворен. Контакт %Q0.0 позволява да задържи водната помпа пусната, докато е налице високото ниво на водата.

Брой напоявания

Добавя се нова секция в програмата, в която се описва броенето на циклите на поливане (фигура 3.15). Това става, като се щракне двукратно върху елемент %C0 и се зададе **Preset** стойност **100** цикъла в таблицата на броячите.



Фиг. 3.15. Секция 2 – Броене на циклите на поливане

Описание на процеса Брой напоявания

Когато контролът за поливане е активиран (%Q0.1 е затворен) от функцията RTC (часовник за реално време 0), броячът %C0 се увеличава с единица. Когато текущата стойност на бояча достигне 100, вътрешната битова променлива %M0 се установява в 1. Тя може да се използва за контрол на светодиод, който ще покаже, че се изисква поддържане, след завършване на 100-поливни цикъла. Може да се свърже един бутон на входа R за ръчно нулиране на брояча на циклите.

Запазване на програмата

След като се приключи с писането на програмата, тя може да се запише по някой от двата начина:

- В случай че вече е зададено име на приложението и е определена директорията на проекта, се кликва върху **Save current project** (**Запазване на текущия проект)**.
- Избира се **Project → Save current project** и бутон **Save**.

В резултат файловете на проекта, съдържащи програмата на приложението, ще се запишат на диска, в директорията на проекта.

Стъпка 6: Проверка на програмата

TwidoSuite предлага функция *симулатор*, която дава възможност за тестване на ново написана програма, без тя да се зарежда в контролера. Twido симулаторът позволява да се стартира програмата и да се тества нейното поведение при промяна на входните сигнали и параметри.

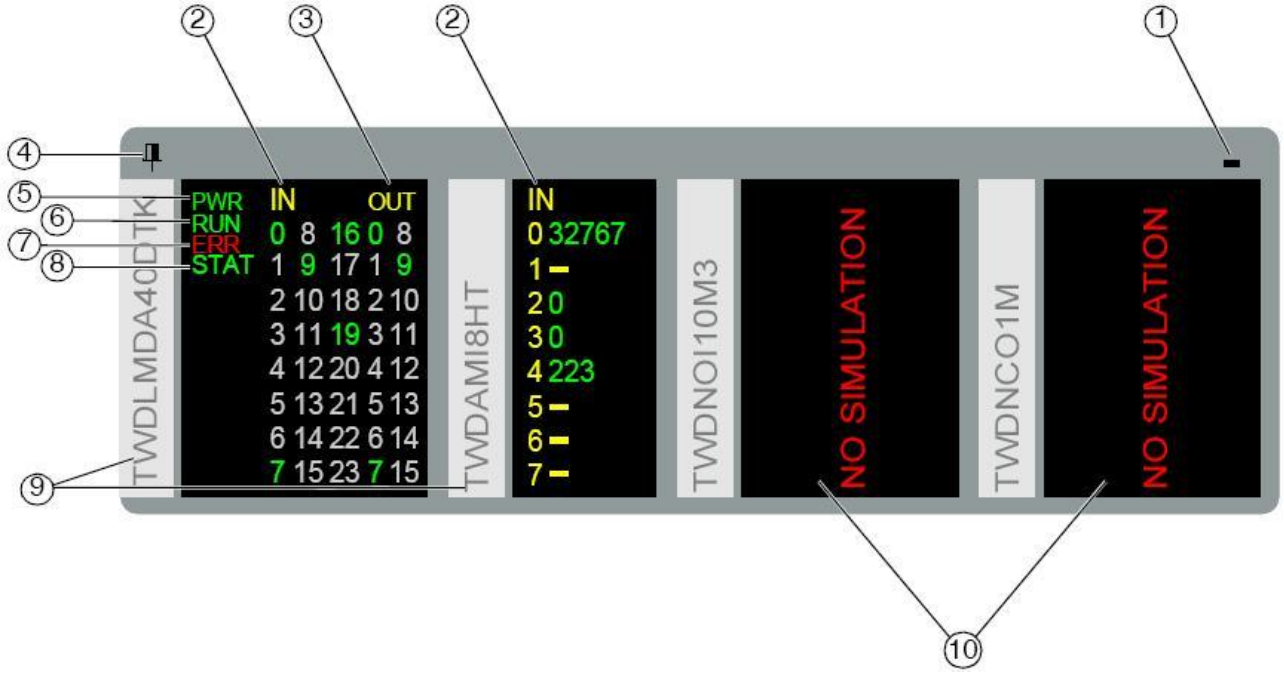
Стартиране на Twido симулатор

За да се стартира Twido симулатор, се избира **Program → Program** и се щрака върху бутона **Simulation** в долната дясна част на TwidoSuite екрана.

Общ преглед на Twido симулатор

Прозорецът на Twido симулатора показва състоянието на входовете и изходите на използвания в програмата контролер, както и на разширителните модули, добавени към него.

Видът на прозореца е показан на фигура 3.16



Фиг. 3.16. Прозорец на Twido симулатор

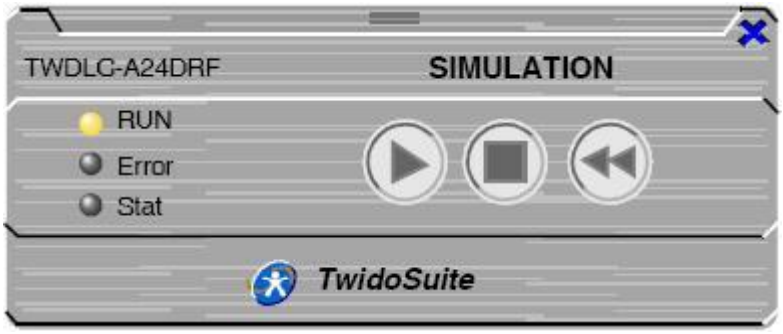
1. Минимизиране.
2. Състояния на входовете:
   - Кликва се за промяна на състоянието (активен/неактивен) на съответния вход.
   - Кликва се два пъти, за да се отвори прозореца с настройки на отделните входове.
3. Състояния на изходите: кликва се два пъти за да се отвори прозореца за проследяване на изходите.
4. Кликва се, за да се постави прозореца най-отгоре.
5. **PWR** LED: показва дали симулираният PLC е захранен.
6. **RUN** LED: Показва състоянието на симулирания PLC.
7. **ERR** LED: Показва състоянието (работно или не) на симулирания PLC.
8. **STAT** LED: Действието на STAT LED се определя от потребителската логика.
9. Препратки към базовия контролер и разширителните му модули.
10. Разширителни модули, които не могат да се симулират.

Twido симулатор табло

Twido симулатор таблото се използва за издаване на следните команди:

- Run (в режим симулиране)
- Stop
- Initialize

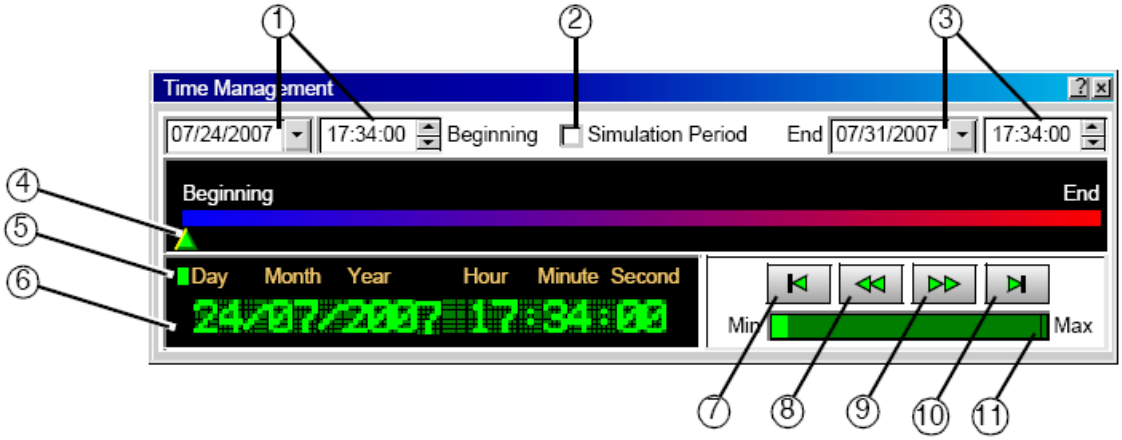
Когато TwidoSuite приложението е в режим на симулиране, тези команди могат да бъдат избрани от таблото на Twido симулатор, както е показан на фигура 3.17.



Фиг. 3.17. Twido симулатор табло

Панел за управление на времето

Панелът за управление на времето (фигура 3.18) позволява да се променя текущото време с цел ускоряване на тестването на програмата в режим на симулация. Може да се избере началото и края на симулацията.



Фиг. 3.18. Панел за управление на времето

1. Начално време - времето в началото на симулацията
2. Продължителност на симулацията:
   - Ако е включена тази опция, след като настъпи краят на симулацията, се появява един прозорец със следния текст: Достигнато е крайното време! Симулаторът ще преминете в режим STOP!
   - Ако това поле не е отметнато, симулацията остава в режим RUN.
3. Краен час - време за край на симулацията.
4. Курсорът показва текущата прогресия на времето. Курсорът може да се премества ръчно, за настройване на текущото време.
5. Състояние на симулатора:
   - Симулатор в режим *стоп*
   - Симулатор в режим *работа*
   - Симулатор в режим *бързо напред*
   - Симулатор в режим *бързо назад*
6. Показва текущото време на симулацията
7. Връщане в началното време
8. Бързо назад във времето
9. Бързо напред във времето
10. Преместване до края на времето

11. Въвеждане нивото на превъртане напред / назад във времето

Спиране на Twido симулатор

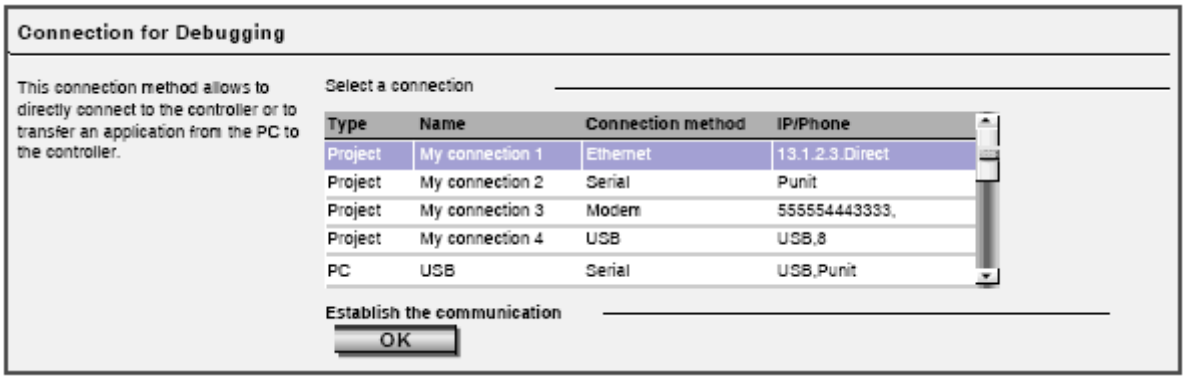
За да се спре Twido симулацията, се щраква върху бутона в долната дясна част на TwidoSuite екрана. След спирането на симулацията TwidoSuite се връща на предходния екран преди началото на Twido симулатор.

Стъпка 7: Зареждане в контролера и стартиране на програмата

Програмата трябва да се запази преди да бъде прехвърлена на PLC.

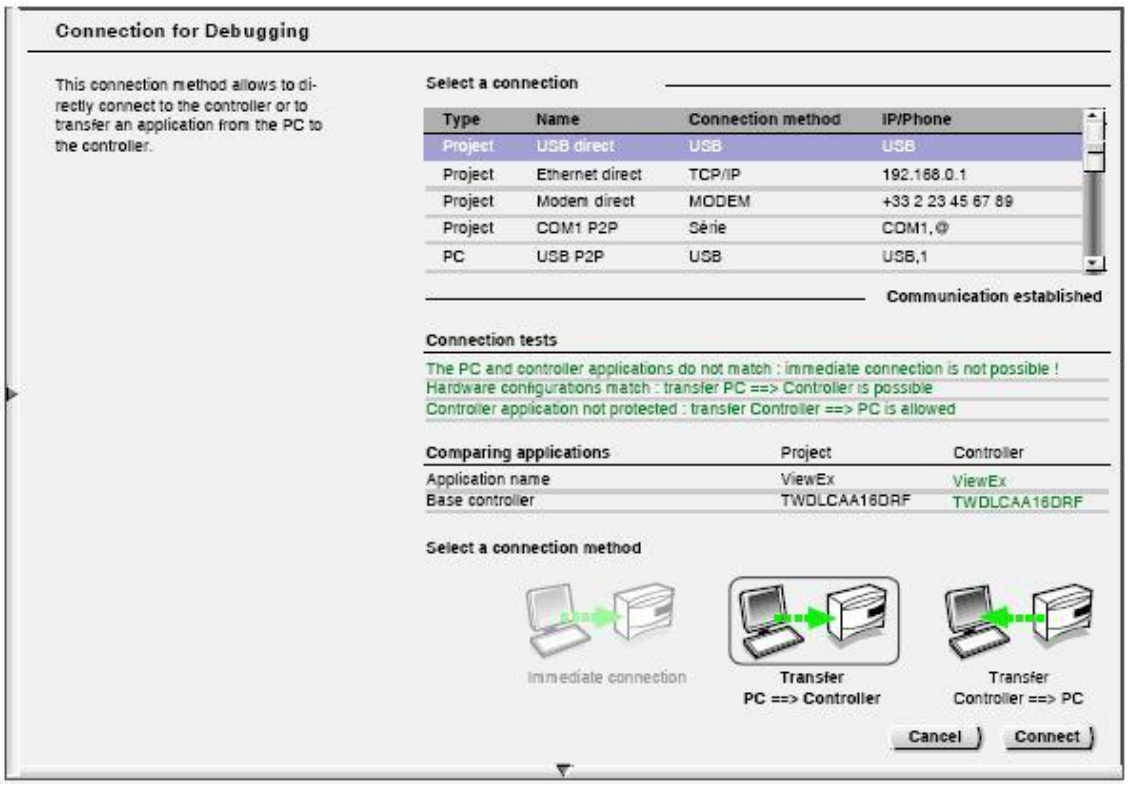
Прехвърляне на програмата в контролера

1. Ако компютърът все още не е свързан към контролера, се избира връзка (фигура 3.19) от таблицата с връзки в **Program → Debug → Connect** и **ОК**.



Фиг. 3.19. Прозорец за избор на връзка

В резултат TwidoSuite се опитва да установи връзка с контролера и извършва проверки на синхронизация между PC и активното PLC приложение (фигура 3.20).



Фиг. 3.20. Прозорец за прехвърляне на програмата

Ако PC и PLC приложението са различни, трябва да се избере **Transfer PC => Controller.**

Ако приложението на контролера е защитено с парола, ще бъдете запитани за парола. Въведете парола и изберете OK за продължаване на прехвърлянето. Ако контролерът работи, се показва прозорец за предупреждение като посочва, че за да продължи прехвърлянето, контролерът трябва да бъде спрян. Ако контролерът е спрян, прехвърлянето започва. Ако контролерът работи, трябва да се избере **ОК**, за да се спре контролера и да се продължи прехвърлянето.

Заредената в контролера програма може да бъде прехвърлена в програмиращата станция с цел анализ и промяна. Това става с избиране на посока на прехвърляне от контролера към PC.

Стартиране на приложението

При успешна връзка, установена между TwidoSuite и контролера, се стартира Twido симулатор табло, снабдено с бутони за управление:

- Стартиране
- Спиране
- Инициализиране

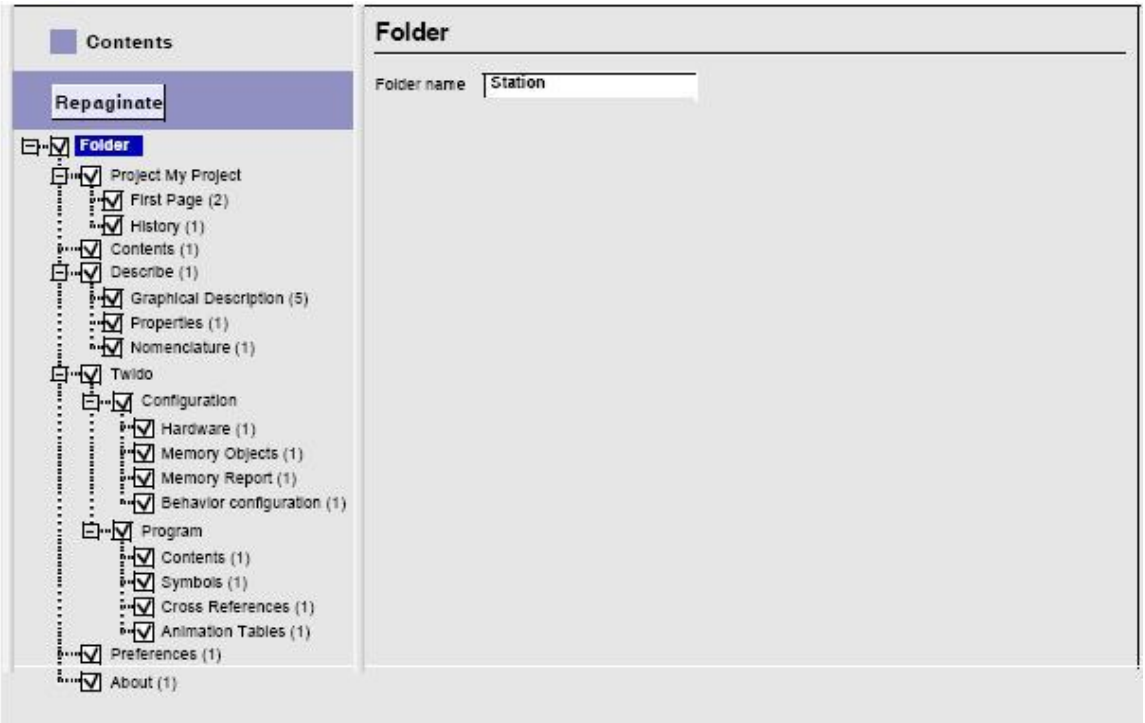
Щраква се върху **Стартиране**, за да се изпълни TwidoSuite приложението, заредено в RAM на контролера. Всяка команда изисква потвърждение, преди изпълнението.

Стъпка 8: Управление на документацията (отчета) на програмата

TwidoSuite предлага възможност за документиране на създадения проект. За достъп до мениджъра на отчети се избира иконата **Report** от TwidoSuite меню.

Конфигуриране на документацията (отчета)

Мениджърът на отчети, показан на фигура 3.21, предоставя дървовидна структура на елементите на отчета и поделементите, които могат да бъдат включени в него чрез маркиране.



Фиг. 3.21. Мениджър на отчети

Настройка на страница

Чрез избиране на **Report → Page Setup** може да се въведе информация, която да се показва в страница, колонтитули и полетата на страницата.

Могат да се конфигурират страниците с отчетите и да се появят по девет различни видове информация в три отделни области на горен и долен ред, включващи:

- Отпечатване на датата;
- Дата на създаване;
- Дата на промяна;
- Наименование на проекта;
- Всеки текст (Свободно);
- Няма текст (Няма);
- Страница номер / Общ брой страници (стр. I / N);
- Номер на страница (стр. I);
- TwidoSuite версия;
- Име на файла.

Отпечатване на документацията (отчета)

За да се отпечата документацията, се избира **Report → Print**. Прозорецът за печат позволява:

- преглед на документацията с няколко страници, формат и опции на мащабиране;
- избиране на отдалечен принтер и достъп до настройките на принтера;
- преномериране на страниците в съответствие с избраните опции за печат;
- изпращане на документацията за печат върху избраното устройство за печат;
- експортиране на документацията в HTML файл.

Лабораторно упражнение № 1

Битови променливи. Логически инструкции за зареждане и съхраняване. Контакти и бобини. Специални логически инструкции. Обработване на логически данни

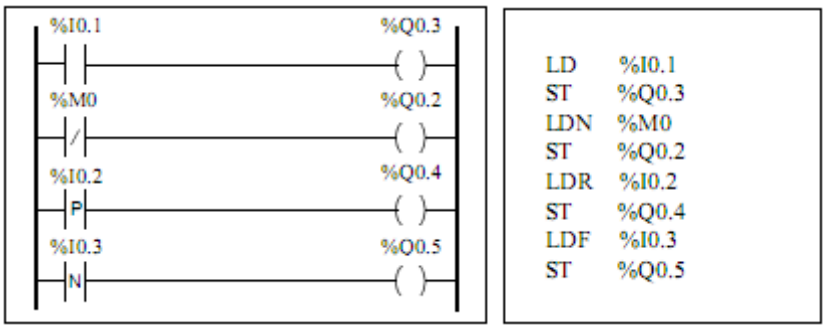
1. Битови променливи

Битовите променливи могат да приемат логическите стойности нула и единица. Като битови променливи могат да се разглеждат:

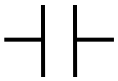
- цифровите входове на контролера, означавани с ***%Ii.j*** (***i*** е номерът на модула на контролера, чийто вход се адресира, ***j*** е номерът на съответния вход);
- цифровите изходи, означавани с ***%Qi.j*** (***i*** и ***j*** имат същите значения както за входовете);
- вътрешни битови променливи - ***%Mi***;
- вътрешни битови системни променливи - ***%Si***;
- изходи на функционални блокове, например - ***%TMi.Q***, ***%Ci.F***.

2. Логически инструкции за зареждане и съхраняване. Контакти и бобини

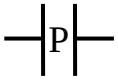
Логическите инструкции за зареждане LD, LDN, LDR и LDF съответстват на отворен, затворен, регистриращ нарастващ фронт на сигнала и регистриращ падащ фронт на сигнала контакти. Логическите инструкции за съхраняване са ST, STN, S и R. На фигура 4.1 е показано съответствието между инструкциите и техните еквиваленти в Ladder диаграмите. Инструкциите за зареждане са свързани с контактите, а инструкциите за съхранение – с бобините.

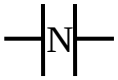


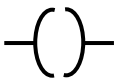
Фиг. 4.1. Съответствие между инструкции за зареждане и контакти


- нормално отворен контакт – затварянето му генерира сигнал логическа единица (представлява проверка дали състоянието на съответния вход, виртуален сигнал или променлива е активно - логическа единица).

- нормално затворен контакт - отварянето му генерира сигнал логическа единица (представлява проверка дали състоянието на съответния вход, виртуален сигнал или променлива е неактивно - логическа нула).


- контакт за откриване на нарастващ фронт на входния сигнал или промяна на входната стойност от 0 на 1. Наличието на нарастващ фронт се интерпретира като логическа единица.


- контакт за откриване на падащ фронт на входния сигнал или промяна на входната стойност от 1 на 0. Наличието на падащ фронт се интерпретира като логическа единица.

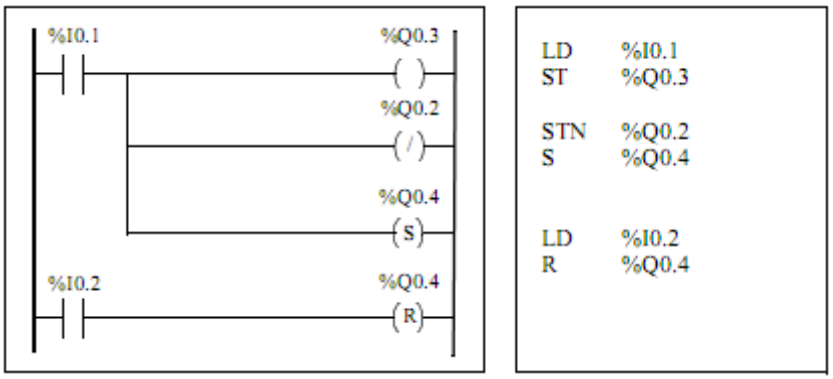

- символ (бобина), указващ установяване в активно ниво на посочения изход или битова променлива. Сигналът се интерпретира като логическа истина (сигнал единица). Времедиаграмата на работа е показана на фигура 4.3.

- символ (бобина), указващ установяване в неактивно ниво на посочения изход или битова променлива. Сигналът се интерпретира като логическа лъжа (сигнал нула). Времедиаграмата на работа е показана на фигура 4.3.

- символ (бобина), указващ, че трябва да бъде подаден активен сигнал на съответната променлива или изход и той да бъде запомнен. Времедиаграмата на работа е показана на фигура 4.4.

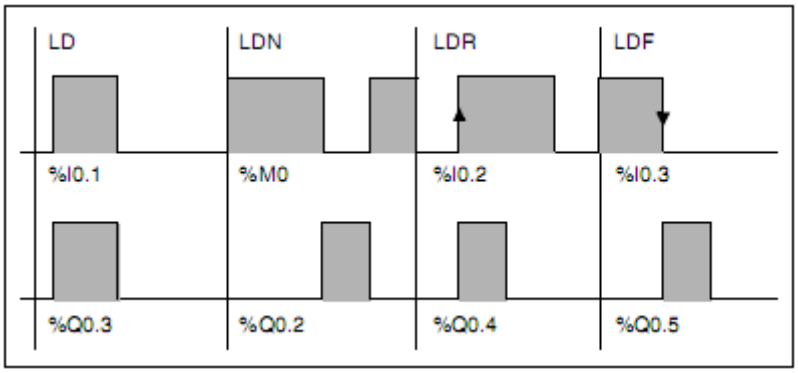
- символ (бобина), указващ, че на указания изход или битова променлива трябва да бъде установен неактивен сигнал (логическа нула). Времедиаграмата на работа е показана на фигура 4.4.

На фигура 4.2 е показано съответствието между инструкциите за съхраняване и използваните в Ladder символични означения на изходите.



Фиг. 4.2. Съответствие между инструкции за съхраняване и контакти

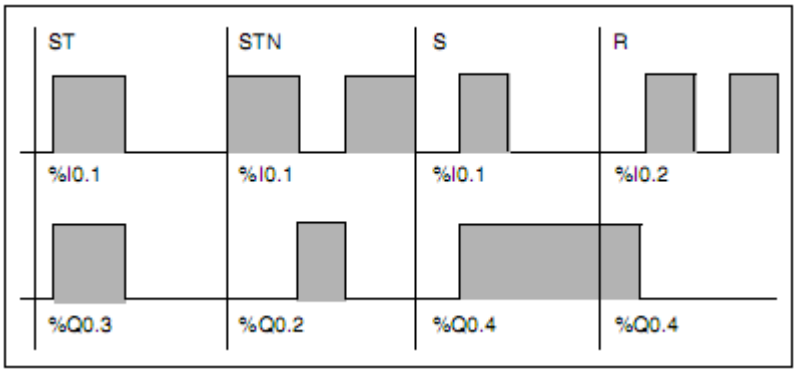
Логическите инструкции обработват битови променливи.



Фиг. 4.3. Времедиаграми на инструкции за зареждане

Инструкцията LD или активно ниво единица на нормално отворения контакт %Ii,j предизвикват установяване на активно ниво единица на изхода %Qi.j, ако е зададена инструкция за съхраняване ST. Инструкцията LDN (активно ниво единица на нормално затворения контакт %Ii.j) предизвикват установяване в неактивно ниво нула на изхода %Qi.j и обратно – при неактивно ниво (нула) на нормално затворения контакт, изходът се установява в активно ниво единица. С други думи – тази инструкция е равностойна на логическата функция отрицание. Инструкцията LDR предизвиква установяване в активно ниво единица на изхода при откриване на нарастващ фронт на входния сигнал за време, равно на продължителността на един цикъл на контролера. Инструкцията LDF предизвиква установяване в активно ниво единица на изхода при откриване на падащ фронт на входния сигнал за време, равно на продължителността на един цикъл на контролера.

На фигура 4.4 са показани времедиаграмите, илюстриращи действието на логическата инструкции за зареждане LD и резултатът от изпълнението на инструкциите за съхранение ST, STN, S и R.

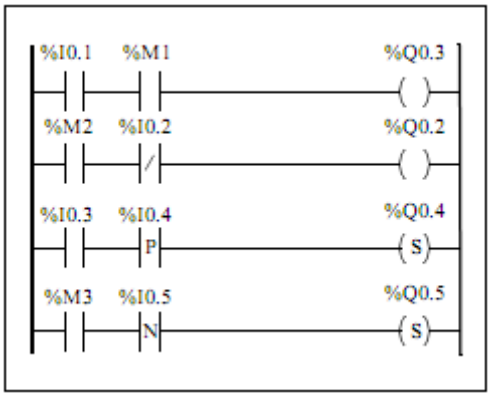


Фиг. 4.4. Времедиаграми на инструкции за съхраняване

Инструкцията LD или активно ниво единица на нормално отворения контакт %Ii,j предизвикват установяване на активно ниво единица на изхода %Qi.j, ако е зададена инструкция за съхраняване ST. При инструкция за съхраняване STN изходът се установява в неактивно ниво нула за времето, за което действа инструкцията LD (на входа има активно ниво единица на сигнала). Инструкцията S предизвиква установяване на изходния сигнал в активно ниво единица, за време, равно на продължителността на цикъла на контролера. Инструкцията R предизвиква установяване в неактивно ниво (нула) на изхода при активно ниво единица на входа, за време, равно на продължителността на един цикъл на контролера.

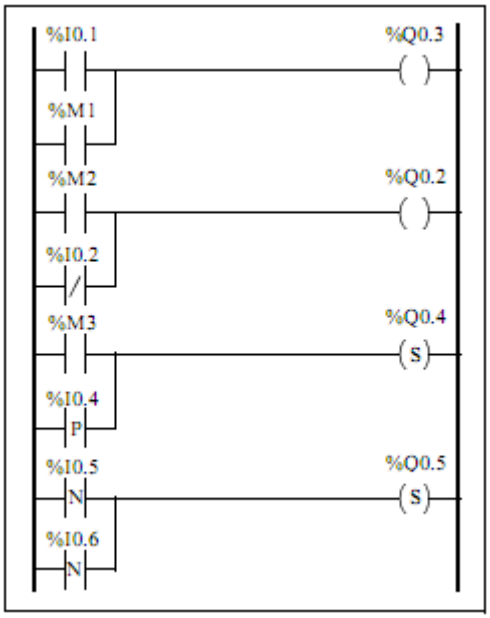
2. Логически инструкции за обработка на логически променливи

- Операция ***логическо произведение*** - Последователното свързване на два или повече контакта е аналогично на логическа операция AND (логическо произведение, операция „И”). Последователно включените контакти могат да бъдат кои да са от изброените по-горе разновидности (виж примерите от фигура 4.5). Така се получават и разновидностите на базовата операция ANDN, ANDR и ANDF.



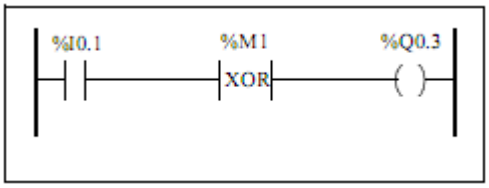
Фиг. 4.5. Логически операции AND, ANDN, ANDR и ANDF

- Операция ***логическа сума*** – Паралелното свързване на два или повече контакта е аналогично на логическа операция OR (логическа сума, операция ИЛИ). Паралелно включените контакти могат да бъдат кои да са от изброените по-горе разновидности (виж примерите от фигура 4.6). Така се получават и разновидностите на базовата операция ORN, ORR и ORF.



Фиг. 4.6. Логически операции OR, ORN, ORR и ORF

- Операция ***изключващо ИЛИ*** (сума по модул две) – Тази операция се изпълнява, когато се свържат последователно два контакта, вторият от които е от вида XOR или разновидностите му XORN, XORR, XORF (виж примера от фигура 4.7). Контакт от вида XOR не може да стои на първа позиция в рънга. Забранено е и включването в паралел на контакта XOR на какви да са други контакти.



Фиг. 4.7. Логическа операция XOR

- Специални инструкции:
  - ✓ OPEN - създава прекъсване в Ladder рънг, независимо от резултатите от последните логически операции;
  - ✓ SHORT - позволява да се премине през рънг, независимо от резултатите от последните логически операции.

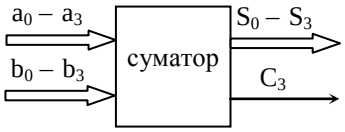
3. Задание

Да се напише програма за събиране две четириразрядни двоични числа. За въвеждане на числата да се използват цифровите входове на контролера. Резултатът да се извежда на цифровите изходи.

3.1. Анализ

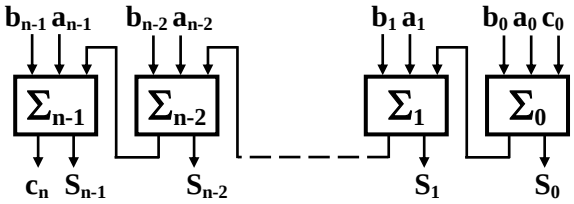
Паралелното събиране на многоразрядни числа се извършва чрез изпълнение на операцията във всеки отделен разряд по един и същи начин (събиране на съответните две цифри и пристигналия от по-младшия разряд пренос). Повтарящата се във всеки отделен разряд операция позволява задачата за синтез на такъв логически възел да бъде сведена до задача за синтез на схема, реализираща операцията в един единствен разряд. Мултиплицирането и подходящото свързване на тази едноразрядна схема позволява да се изгради суматор с произволна дължина.

За реализирането на четириразрядния двоичен суматор са необходими четири входа за едното число, четири – за другото число и пет изхода за сумата с отчитане на преноса (виж фигура 4.8).



Фиг. 4.8. Общ вид на суматора

На фигура 4.9 е показана схемата на изграждането на n-разряден двоичен суматор на базата на едноразряден пълен двоичен суматор Този вид свързване се използва за решаването на поставената задача.



Фиг. 4.9. n-разряден суматор с последователен пренос

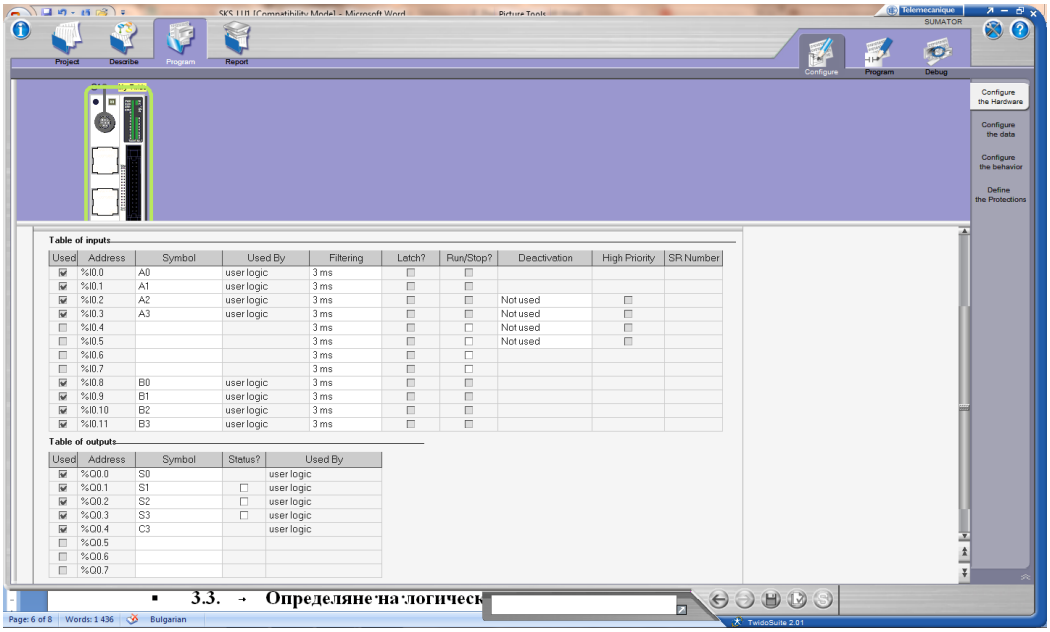
Таблица на истинност на едноразрядния двоичен суматор

a_i	b_i	C_{i-1}	C_i	S_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

3.2. Разпределение на ресурсите на контролера за решаване на задачата

Входове		Изходи
a_0 - %I0.0;	b_0 - %I0.8;	S_0 - %Q0.0
a_1 - %I0.1;	b_1 - %I0.9;	S_1 - %Q0.1
a_2 - %I0.2;	b_2 - %I0.10;	S_2 - %Q0.2
a_3 - %I0.3;	b_3 - %I0.11;	S_3 - %Q0.3
		c_3 - %Q0.4

За междинните преноси C0, C1 и C2 се използват вътрешни битови променливи на контролера. Фигура 4.10 показва екрана с конфигурираните входове и изходи на контролера.



Фиг. 4.10. Конфигурираните входове и изходи на контролера

3.3. Определяне на логическите изрази за функциите на изходите

ai \ bi Ci-1	00	01	11	10
0			1	
1		1	1	1

$c_i = a_i.b_i + a_i.c_{i-1} + b_i.c_{i-1}$ → за съхраняване на междинния пренос използваме временни вътрешни битови променливи % M_i.

$$S_i = \overline{a_i}.\overline{b_i}.c_{i-1} + \overline{a_i}.b_i.\overline{c_{i-1}} + a_i.\overline{b_i}.\overline{c_{i-1}} + a_i.b_i.c_{i-1}$$

3.4. Система уравнения за всеки суматор:

i = 0

$c_0 = a_0.b_0$

$$S_0 = \overline{a_0}.b_0 + a_0.\overline{b_0}$$

i = 1

$$c_1 = a_1.b_1 + a_1.c_0 + b_1.c_0$$

$$S_1 = \overline{a_1}.\overline{b_1}.c_0 + \overline{a_1}.b_1.\overline{c_0} + a_1.\overline{b_1}.\overline{c_0} + a_1.b_1.c_0$$

i = 2

$c_2 = a_2.b_2 + a_2.c_1 + b_2.c_1$

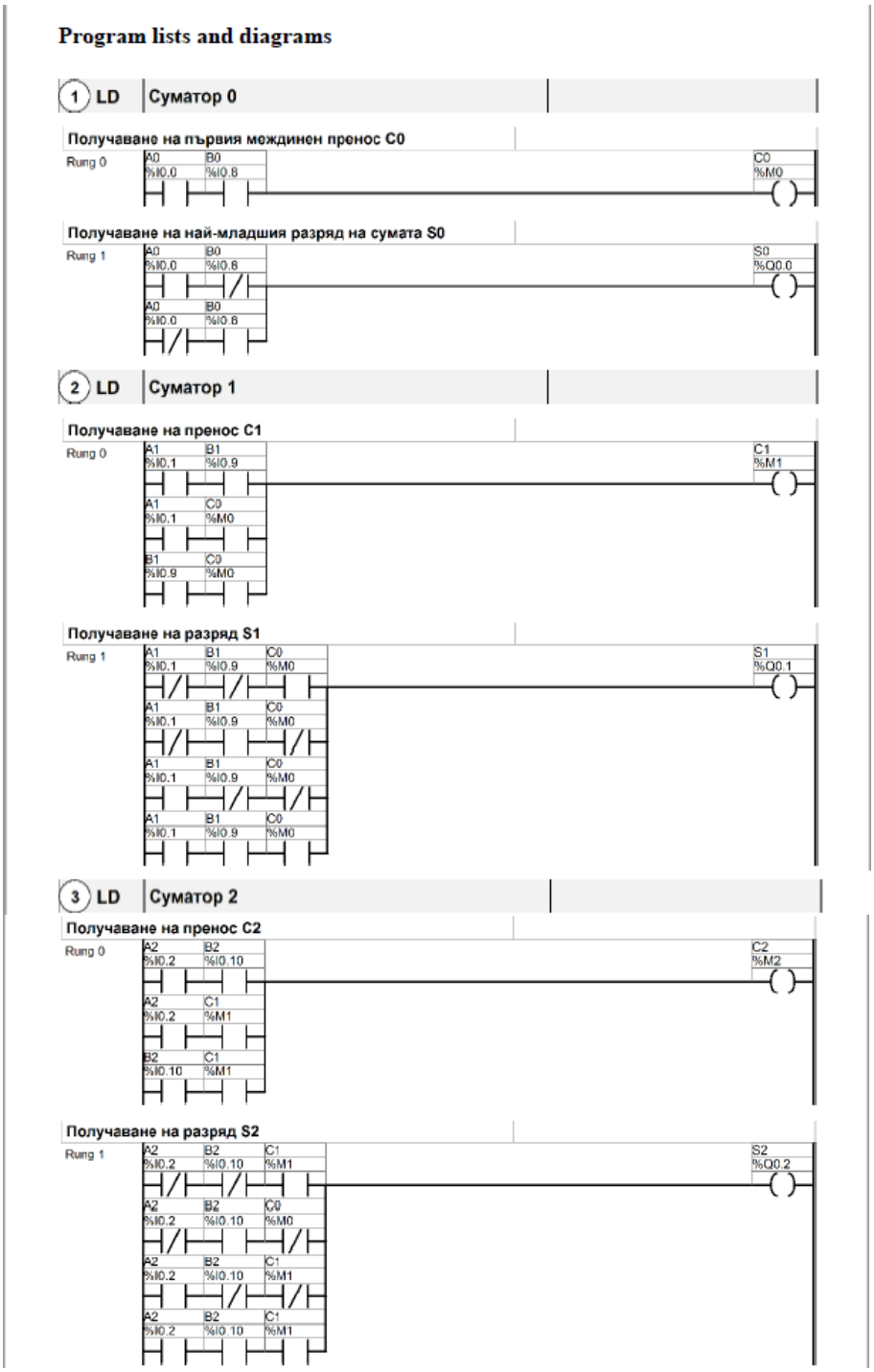
$S_2 = \overline{a_2}.\overline{b_2}.c_1 + \overline{a_2}.b_2.\overline{c_1} + a_2.\overline{b_2}.\overline{c_1} + a_2.b_2.c_1$

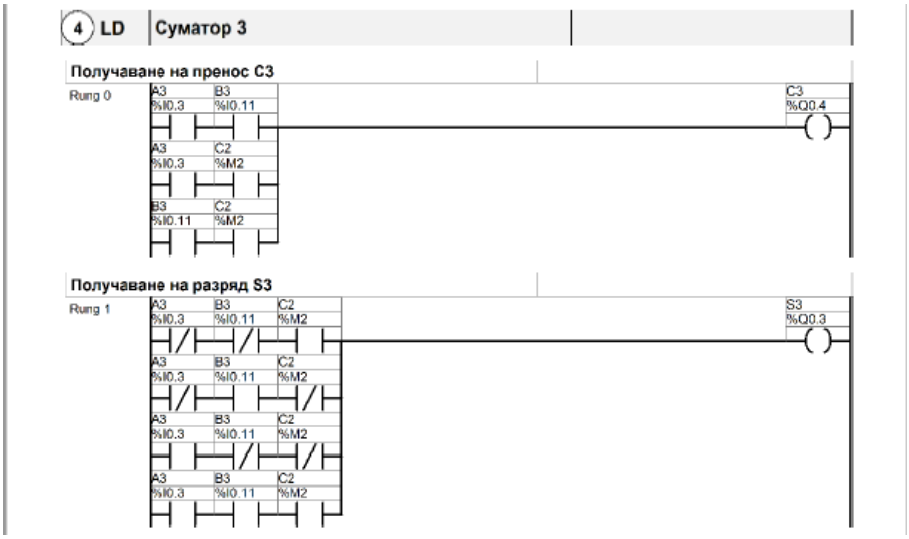
i = 3

$c_3 = a_3.b_3 + a_3.c_2 + b_3.c_2$

$S_3 = \overline{a_3}.\overline{b_3}.c_2 + \overline{a_3}.b_3.\overline{c_2} + a_3.\overline{b_3}.\overline{c_2} + a_3.b_3.c_2$

3.5. Ladder диаграма (фигура 4.11)

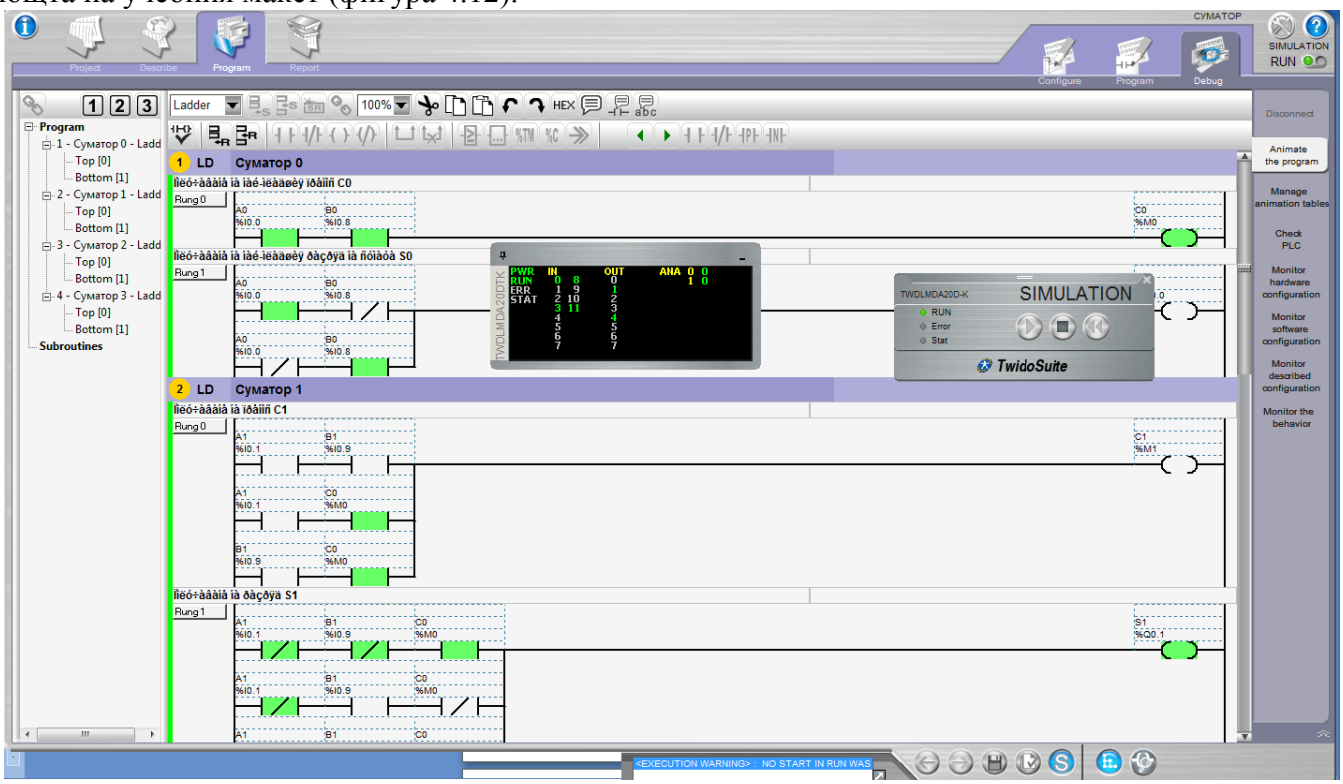




Фиг. 4.11. Програма (Ladder диаграма) за събиране на две четириразрядни двоични числа

3.6. Тестване на програмата

Програмата може да бъде тествана в средата Twido Suite в режим „симулация”, както и с помощта на учебния макет (фигура 4.12).



Фиг. 4.12. Резултати от тестването на програмата в режим симулация

Показано е събирането на числата 9 + 9. Резултатът е равен на 18. Всички числа са представени в двоична бройна система.

Забележка: Разбира се, никой няма да реализира на практика такъв суматор с помощта на промишлен контролер. Това задание е с учебна цел, за да могат по-лесно да се усвоят средата за програмиране и работата с учебния макет.

4. Задачи за самостоятелна работа

4.1. Модифицирайте програмата, като използвате контакт „изключващо ИЛИ”, където е възможно.
4.2. Напишете програма за събиране на две едноразрядни двоично-десетични числа.
4.3. Напишете програма за сравняване на две четириразрядни двоични числа.

Лабораторно упражнение № 2

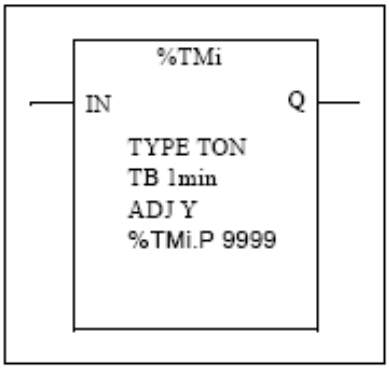
Таймери. Обекти от тип дума. Аритметични инструкции, инструкции за присвояване и за сравнение

1. Таймери

Таймерите са функционални блокове, които отчитат зададен интервал от време. Графичното изображение е показано на фигура 5.1. Таймерите могат да работят в няколко режима, т. е. различават се няколко вида таймери:

TON (Timer On Delay) – включване със закъснение. На изхода на таймера се появява активно ниво на сигнала след изтичането на предварително зададен интервал от време.

TOF (Timer OFf Delay) – изключване със закъснение. Таймерът се изключва след изтичането на предварително зададен интервал от време.



Фиг. 5.1. Функционален блок таймер

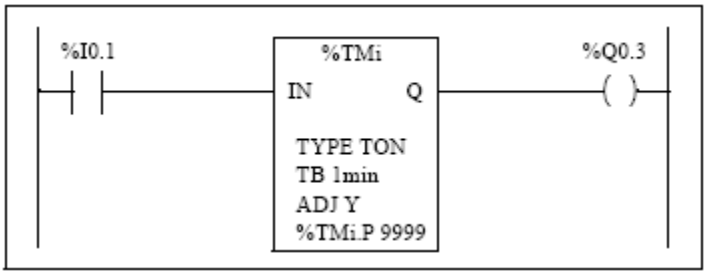
TP (Timer Pulse) – отмерващ импулс. Използва се за генериране на импулси с определена продължителност.

1.4. Параметри на таймерите

- Номер на таймера ***i*** - ***%TMi***. За разглеждания контролер (TWD20DTK, -DTR) могат да бъдат конфигурирани до 128 таймера с номера от 0 до 127;
- Тип на таймера (Type) – ***TON, TOF, TP*;**
- Времева база (Timer Base) ***TB*** – минималната константа на таймера (задава се от програмиста). Времевите бази могат бъдат избирани от следните, заложени в конфигурацията на контролера: 1 min, 1 s, 100 ms, 10 ms, 1 ms. С цел постигане на по-голяма точност на измервания интервал, се препоръчва използването на възможно най-малката времева база.;
- Текуща стойност на брояча на таймера ***%TMi.V*** – дума, която се увеличава от 0 до ***%TMi.P***, когато таймерът е задействан;
- Стойност ***%TMi.P*** – променлива от тип дума, която може да се чете и модифицира от програмата със стойности от 0 до 9999. Съвместно с времевата база определя продължителността на отмервания интервал ***T*** по формулата:

$$T = TB * \%TMi.P$$

- ***ADJ*** (***Y/N***) – може да приема стойност ДА или НЕ и показва дали е позволено стойността на ***%TMi.P*** да бъде променяна посредством редактора на анимационните таблици (Animation Tables Editor);
- Разрешение за вход ***IN*** – стартира таймера по издигащ се или падащ фронт на входния сигнал.
- Изход ***Q*** и асоциирана с него битова променлива ***%TMi.Q*** – установява се в активно ниво по различно време, в зависимост от типа на конфигурирания таймер.

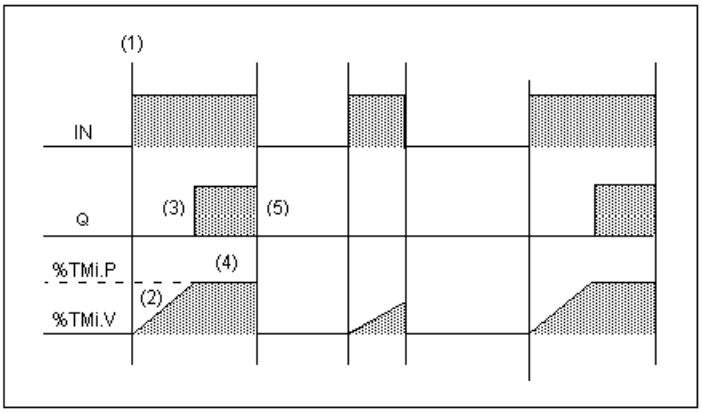


Фиг. 5.2. Използване на таймер в Ladder диаграма

1.5. Времедиаграми

TON (Timer on Delay) - включване със закъснение.

При този тип таймери стартирането на брояча започва по нарастващия фронт на входния сигнал. При достигане на предварително зададената стойност *%TMi.P* изходът на таймера се установява в активно ниво и остава в него до отпадане на входния сигнал. При отпадането на входния сигнал броячът се нулира, а изходът се установява в неактивно ниво. Времедиаграмата на таймера е показана на фигура 5.3, а работата му е описана в таблица 5.1.



Фиг. 5.3. Времедиаграма, илюстрираща работата на TON таймерите

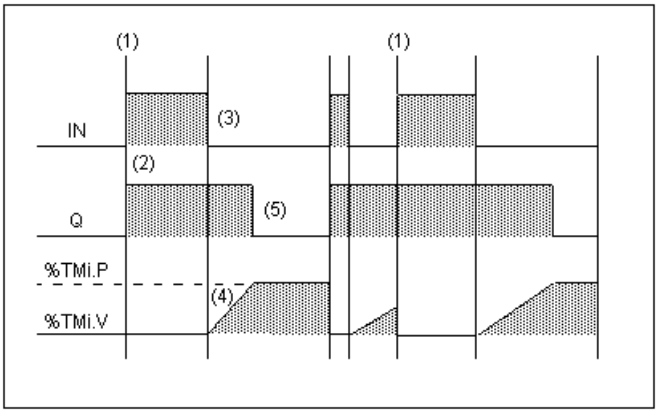
Таблица 5.1. Описва действието на TON таймерите

Фаза	Описание
1.	Броячът на таймера започва броенето по нарастващия фронт на входния сигнал (IN).
2.	Текущата стойност на брояча %TMi.V се увеличава от 0 до %TMi.P, на стъпки от една единица за всеки импулс на базовото време (TB).
3.	Изходът Q (и битовата променлива %TMi.Q) се установява в единица, когато текущата стойност на брояча достигне % TMi.P.
4.	Изходът Q (и %TMi.Q) остава в единица, докато на входа (IN) има активен сигнал (единица).
5.	Когато се открие падащ фронт на сигнала на входа (IN), броенето спира, дори ако броячът не е достигнал %TMi.P, и %TMi.V се установява в нула.

TOF (Timer Off Delay) – Изключване със закъснение

При този тип таймери стартирането на брояча започва по падащия фронт на входния сигнал. Изходът Q на таймера се установява в активно ниво по нарастващия фронт на входния сигнал (IN). При достигане на предварително зададената стойност *%TMi.P* изходът на таймера се

установява в неактивно ниво и остава в него до появяването на нов нарастващ фронт на входния сигнал. В този момент се нулира и текущата стойност на брояча на таймера, за да се подготви за нов цикъл на броене. Времедиаграмата на таймера е показана на фигура 5.4, а работата му е описана в таблица 5.2.



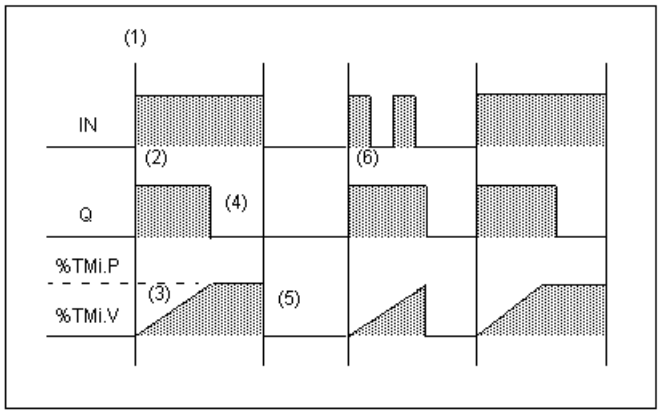
Фиг. 5.4. Времедиаграма илюстрираща работата на TOF таймерите

Таблица 5.2. Описва действието на TOF таймерите

Фаза	Описание
1.	Текущата стойност на брояча %TMi.V се установява в 0 по нарастващия фронт на входния сигнал (IN) и остава в нула до появяването на задния фронт на входния сигнал.
2.	Изходният сигнал Q (и %TMi.Q) се установява в активно ниво единица по нарастващия фронт на входния сигнал.
3.	Броенето започва по падащия фронт на входния сигнал (IN).
4.	Текущата стойност %TMi.V се увеличава до % TMi.P на стъпки от една единица за всеки импулс на базовото време (TB).
5.	Изходът Q (и битовата променлива %TMi.Q) се нулира, когато текущата стойност на брояча достигне %TMi.P.

TP (Timer Pulse) – Отмерващ импулс

При този тип таймери стартирането на брояча започва по нарастващия фронт на входния сигнал. Изходът Q на таймера се установява в активно ниво по нарастващия фронт на входния сигнал (IN). При достигане на предварително зададената стойност *%TMi.P* изходът на таймера се установява в неактивно ниво и остава в него до появяването на нов нарастващ фронт на входния сигнал. Броячът се нулира по падащия фронт на входния сигнал. Така таймерът генерира изходни импулси с точно определена продължителност, започващи по нарастващия фронт на всеки входен сигнал. Времедиаграмата на таймера е показана на фигура 5.5, а работата му е описана в таблица 5.3.



Фиг. 5.5. Времедиаграма, илюстрираща работата на TP таймерите

Таблица 5.3. Описва действието на TOF таймерите

Фаза	Описание
1.	Таймерът стартира броенето по нарастващ фронт на входния сигнал (IN). Първоначалната стойност %TMi.V е 0, ако таймерът не е стартиран.
2.	При стартиране на броенето изходът (както и битовата променлива %TMi.Q) се установява в единица.
3.	Текущата стойност на таймера %TMi.V се увеличава от 0 до %TMi.P, със стъпка една единица на импулс на базовото време (TB).
4.	Изходът (и %TMi.Q) се установява в нула, когато текущата стойност на брояча достигне %TMi.P.
5.	Текущата стойност на брояча %TMi.V се нулира, когато %TMi.V е равно на %TMi.P и едновременно с това входният сигнал е равен на нула.
6.	Този таймер не може да бъде рестартиран, дори когато входният сигнал отпадне, преди броячът да е достигнал стойността %TMi.P.

2. Обекти от тип дума

Обектите от тип дума се съхраняват в паметта като 16-битови думи и могат да приемат целочислени стойности в интервала от -32768 до +32767. Обектите от тип дума могат да бъдат: непосредствени стойности; думи от паметта (%MWi); константи (%KWi); думи за входно-изходен обмен (%IWi, %QWi); системни думи (%SWi) и думи, участващи в конфигурирането на функционалните блокове.

3. Инструкции за присвояване и за сравнение, аритметични инструкции

3.1 Инструкция за присвояване

Инструкцията за присвояване се използва, за да се присвои стойността на операнд 2 на операнд 1. Синтаксисът на инструкцията е Op1 := Op2. Инструкцията се записва в операторен блок.

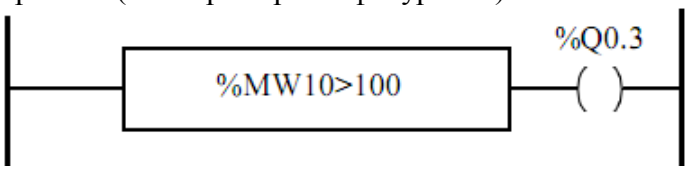
3.2 Инструкции за сравнение

Инструкциите за сравнение (таблица 5.4) се използват за аритметическо сравняване на стойностите на два операнда.

Таблица 5.4. Инструкции за сравнение

Инструкция/ Оператор	Функция	Синтаксис
>	Проверява дали оператор 1 е по-голям от оператор 2	Op1 > Op2
>=	Проверява дали оператор 1 е по-голям или равен от оператор 2	Op1 >= Op2
<	Проверява дали оператор 1 е по-малък от оператор 2	Op1 < Op2
<=	Проверява дали оператор 1 е по-малък или равен от оператор 2	Op1 <= Op2
=	Проверява дали оператор 1 е равен на оператор 2	Op1 = Op2
<>	Проверява дали оператор 1 е различен от оператор 2	Op1 <> Op2

Инструкциите за сравнение се записват в блокове за сравнение, които се поставят в областта за проверка на Ladder програмата (виж примера от фигура 5.6).



Фиг. 5.6. Блок и операция за сравнение

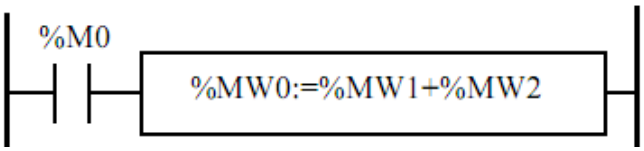
3.3 Аритметични инструкции

Аритметичните инструкции (таблица 5.5) се използват за изпълняване на аритметични операции върху един или два цели операнда.

Таблица 5.5. Аритметични инструкции

Инструкция/ Оператор	Функция	Синтаксис
+	Събиране на два операнда	Op1:=Op2 + Op3
-	Изваждане на два операнда	Op1:=Op2 - Op3
*	Умножение на два операнда	Op1:=Op2 * Op3
/	Деление на два операнда	Op1:=Op2 / Op3
REM	Остатък от деленето на два операнда	Op1:=Op2 REM Op3
SQRT	Корен квадратен от един операнд	Op1:=SQRT(Op2)
INC	Увеличава с единица един операнд	INC Op1
DEC	Намалява с единица един операнд	DEC Op1

Аритметичните инструкции се записват в операторни блокове, които се поставят в изпълнителната област на Ladder програмата (виж примера от фигура 5.7).



Фиг. 5.7. Операторен блок и аритметична операция

4. Примерно задание

Да се реализират шест режима на бягаща светлина, както е показано в таблица 5.6. Б1, Б2 и Б3 са бутони, с които се избират режимите. Бягащата светлина да се реализира върху осем светодиода.

Таблица 5.6. Режими на бягаща светлина

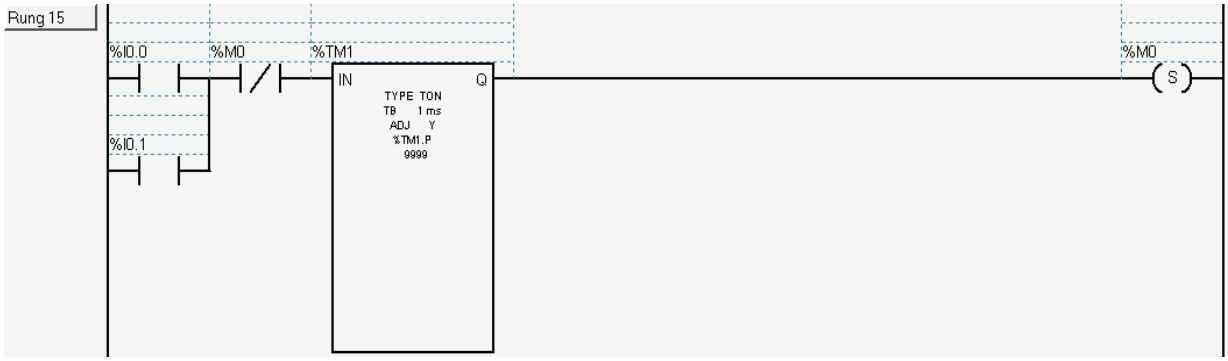
Б1	Б2	Б3	Време Т	Посока
Изкл.	Вкл.	Изкл.	1s	→
Вкл.	Изкл.	Изкл.	0.5s	→
Вкл.	Вкл.	Изкл.	0.25s	→
Изкл.	Вкл.	Вкл.	1s	←
Вкл.	Изкл.	Вкл.	0.5s	←
Вкл.	Вкл.	Вкл.	0.25s	←

Анализ на задачата

Бутоните Б1, Б2 и Б3 имат две състояния – включено и изключено, т. е. сигналите, които идват от тях, могат да се възприемат като логически и е удобно три от цифровите входове на контролера да бъдат свързани с тези бутони. По условие бягащата светлина се реализира върху осем светодиода, чиито състояния са също две – включено и изключено. За целта е удобно да се използват цифровите изходи на контролера Q0.0 до Q0.7.

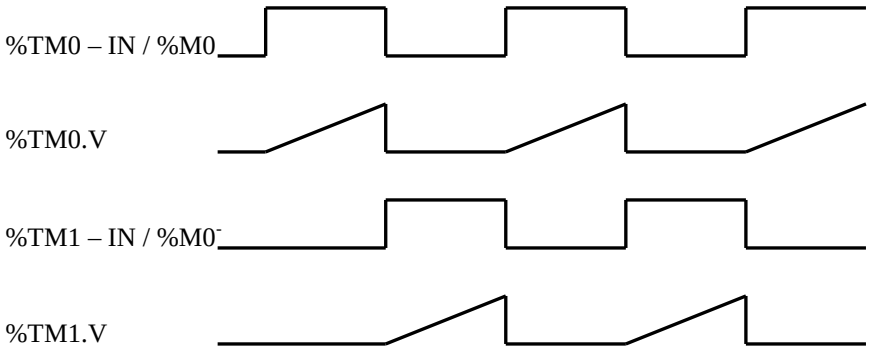
Удобно е номерът на активния (светещия) светодиод (респективно – на съответния изход на контролера) да се съхранява във вътрешна променлива от тип дума, например %MW0. Тъй като номерата се увеличават или намаляват с единица в зависимост от посоката на бягане на светлината, то тази променлива изпълнява ролята на брояч, чиято стойност се увеличава или намалява с единица. За проверка на текущата стойност на брояча се използват операции за сравнение.

Времето за светене на един светодиод се осигурява от мултивибратор, който генерира последователност от правоъгълни импулси с равна продължителност и се реализира с два таймера %TM0 и %TM1 от тип TON, работещи съвместно, посредством вътрешната битова променлива %M0 (виж фигура 5.8).



Фиг. 5.8. Мултивибратор, реализиран с два таймера от тип TON

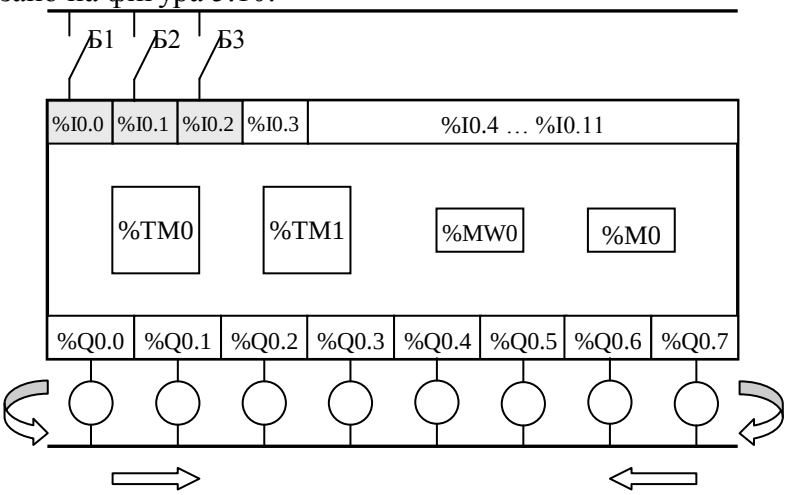
Вътрешната битова променлива %M0 се използва за подаване на входни сигнали на таймерите през равни интервали от време. Времедиаграмата на съвместната работа на таймерите е показана на фигура 5.9.



Фиг. 5.9. Времедиаграма на съвместна работа на два таймера

4.2. Разпределение на ресурсите на контролера

В резултат на анализа на задачата следва следното разпределение на необходимите ресурси на контролера, показано на фигура 5.10.



Фиг. 5.10. Разпределение на ресурсите на контролера за реализация на бягаща светлина

4.3. Настройки на таймера

Времето за светене на един светодиод е T и е равно на продължителността на периода на генерираните от мултивибратора правоъгълни импулси. Всеки от таймерите в него генерира импулс с продължителност t, следователно:

$$T = 2t, \quad \text{т. е.} \quad t = 1/2T$$

$$t = TB * TMi.P \;\; \text{-->} \;\; TMi.P = t/TB \;\; \text{- ->} \;\; TMi.P = T/2*TB$$

При T = 1s	При T = 0.5s	При T = 0.25s
TMi.P = 500	TMi.P = 250	TMi.P = 125
TB = 1ms	TB = 1ms	TB = 1ms

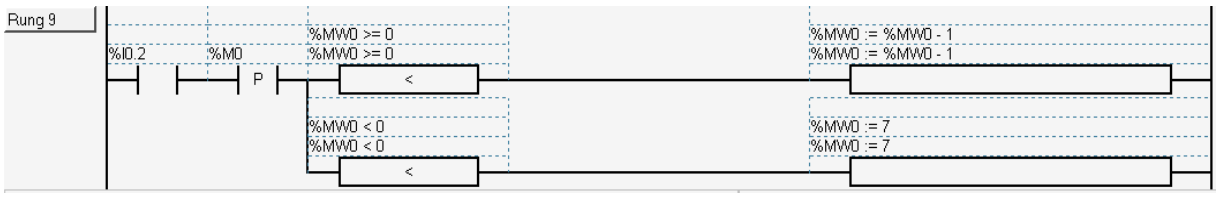
За базова константа на таймерите умишлено е избрана най-малката възможна. Така се осигурява възможно най-голямата точност на продължителността на генерираните импулси.

4.4. Фрагменти от програмата

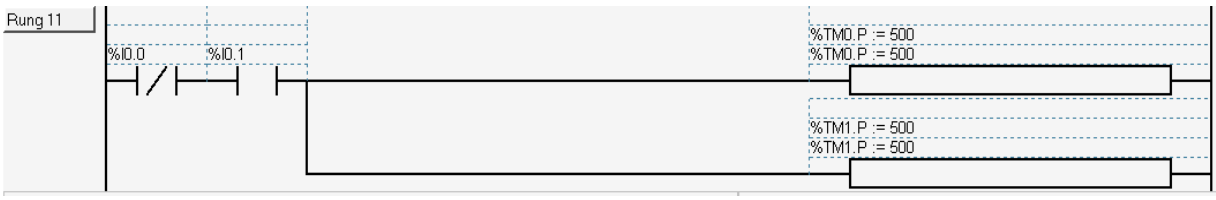
При стартиране на програмата в Rung 0, при нормално затворени контакти, на думата %MW0 се присвоява стойност 0, осигурявайки един вид начално установяване на брояча.

В следващите рънгове се проверява текущата стойност на брояча, като всяка от тях активира съответния изход на контролера.

Бягането на светлината, т. е. последователното активиране на изходите, се осигурява с промяната на текущата стойност на брояча %MW0. В зависимост от избраната с бутон Б3 посока, тази стойност се увеличава или намалява с единица. За да се осигури цикличност на пробягването на светлината открай докрай, при достигане на граничната стойност (0 или 7 в зависимост от посоката), се извършва принудително установяване на променливата с начална стойност. Промяната на съдържанието на брояча трябва да се извърши еднократно в рамките на един период на генерираните от мултивибратора импулси, затова се избира това да става по фронта (нарастващ или падащ) на сигнала, установяващ променливата %M0.



Промяната на скоростта на пробягване на светлината се регулира с продължителността на генерираните от мултивибратора импулси, а това е свързано с промяната на параметъра %TMi.P.



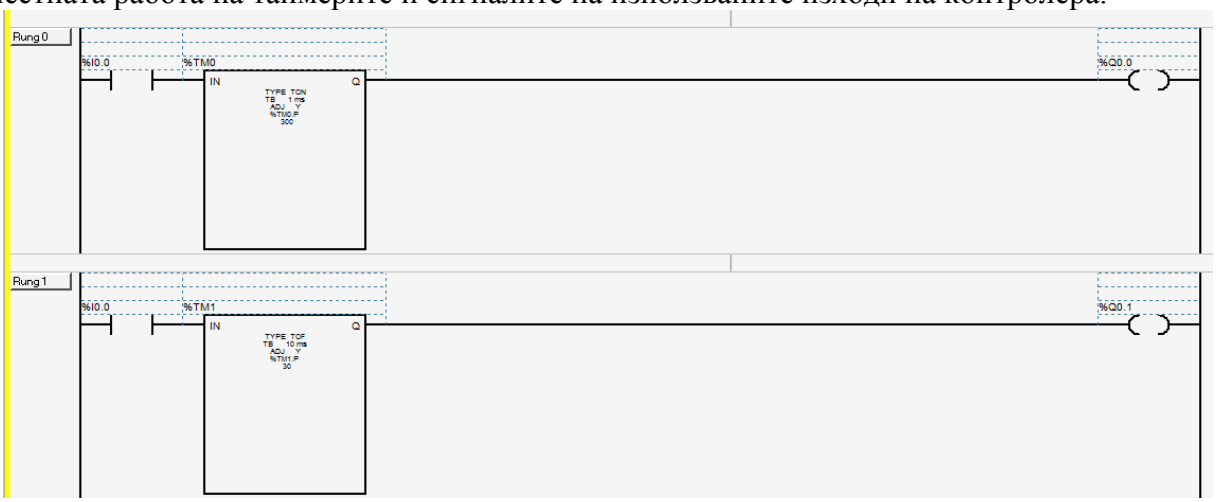
Напишете програмата и я тествайте за работоспособност във всички режими.

5. Въпроси и задачи

5.1 Може ли да се използват друг вид таймери за реализиране на мултивибратора? Обосновете отговора си. Илюстрирайте твърдението си с времедиаграми.

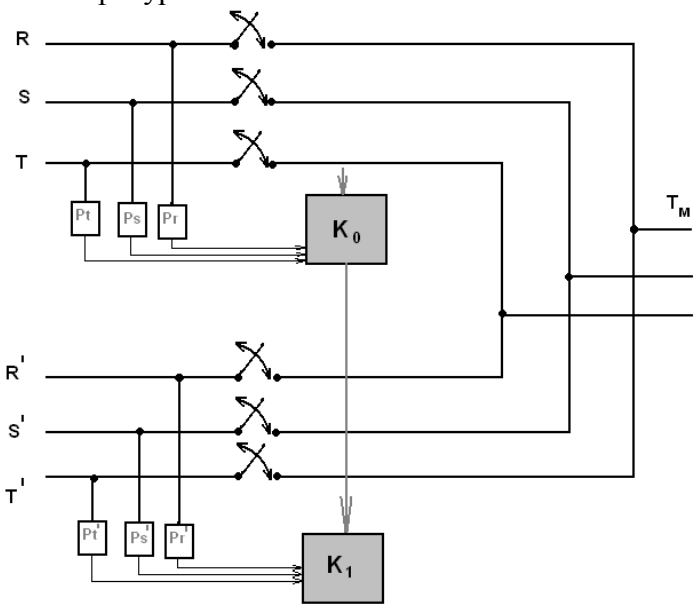
5.2 При същите параметри реализирайте бягаща светлина, но с по два едновременно светещи светодиода.

5.3. Зададен е фрагмент от Ladder диаграма (фигура 5.11). Илюстрирайте с времедиаграма съвместната работа на таймерите и сигналите на използваните изходи на контролера.



Фиг. 5.11.

5.4. Реализирайте автоматизирано осигуряване на непрекъснато токозахранване за постановката, показана на фигура 5.12.



Фиг. 5.12. Схема на постановка с непрекъсваемо токозахранване

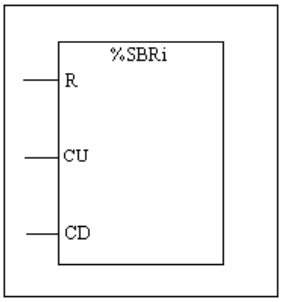
На фигурата с R, S и T са обозначени трите фази на основното захранване, а с R’, S’и T’ – трите фази на резервното захранване. С P са означени измервателите на мощност, по един за всяка фаза. К са контактори, чрез които се включва едното или другото захранване.

Алгоритъмът на работа е следният: Ако отпадне някоя от фазите на основното захранване за повече от една секунда, се включва резервното захранване (чрез K_1), ако са налице и трите му фази. При появяване на основното захранване и ако то е стабилно за повече от една минута, се преминава към захранване на обекта чрез него (K_0).

Лабораторно упражнение № 3

Функционален блок „изместващ регистър” - %SBRi

Функционалният блок „изместващ регистър” (Shift Bit Register - %SBRi) осигурява ляво или дясно изместване на битови данни (0 или 1). Графичното му означение е показано на фигура 6.1.



Фиг. 6.1. Графично означение на функционален блок „изместващ регистър”

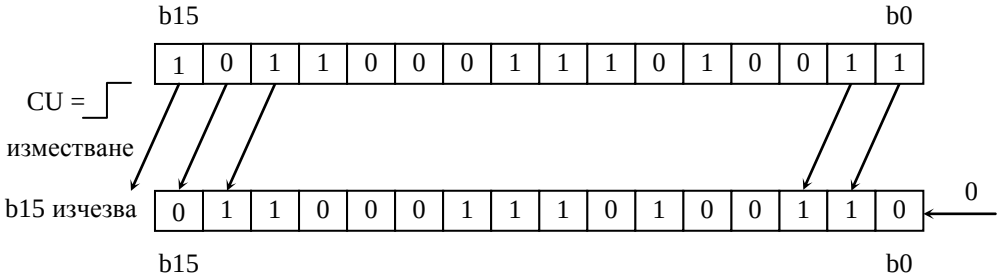
1. Параметри на изместващия регистър

В контролерите от серия Twido има вградени осем шестнадесетбитови изместващи регистъра. Параметрите на функционалния блок „изместващ регистър” са показани в таблица 6.1.

Таблица 6.1. Параметри на изместващ регистър

Параметър/Вход	Етикет	Стойност/Описание
Номер на регистър	%SBRi	от 0 до 7
Адресиране на бит от регистъра	%SBRi.j	Битове от 0 до 15 (j = 0 до 15) на изместващия регистър могат да бъдат тествани чрез тест инструкция.
Вход за начално установяване	R	При логическа единица на сигнала на този вход всички битове в регистъра се установяват в нула.
Вход за изместване наляво	CU	По нарастващия фронт на сигнала, постъпващ на този вход, се осъществява изместване на информацията в регистъра с един бит наляво.
Вход за изместване надясно	CD	По нарастващия фронт на сигнала, постъпващ на този вход, се осъществява изместване на информацията в регистъра с един бит надясно.

На фигура 6.2 е илюстрирано изместването наляво с един бит (при нарастващ фронт на сигнала на входа CU) . При това изместване най-старшият бит (бит 15) се губи, а в най-младшия се записва нула.

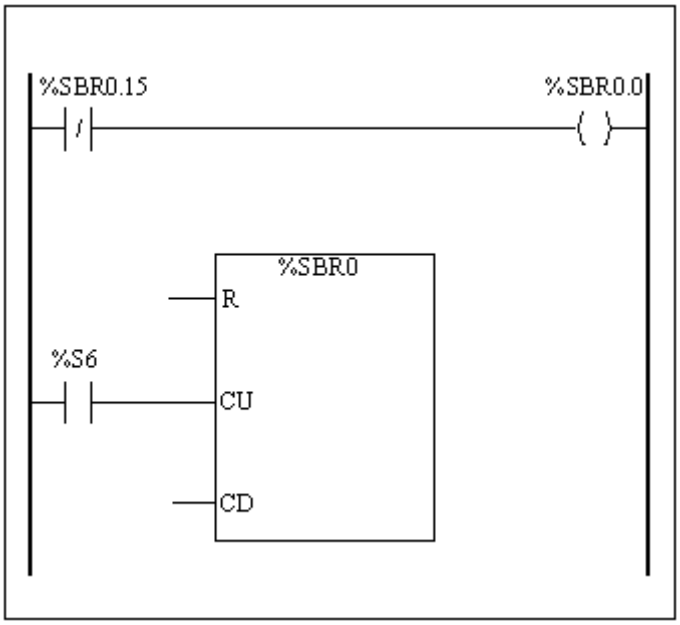


Фиг. 6.2. Изместване наляво

Аналогична е схемата на изместване в посока надясно. Отпадащият бит е b0, а на мястото на b15 се записва нула. Това става по нарастващ фронт на сигнала, подаден на входа CD.

За някои приложения е необходим изместващ регистър с по-голяма дължина. Тогава се налага да бъдат свързани последователно необходимият брой регистри.

На фигура 6.3 е показан пример за изместване наляво на информацията, записана в регистъра, на всяка секунда. Това се осигурява с помощта на системната променлива %S6, която променя състоянието си на противоположното (0 – 1 – 0 - 1) през интервал от 1 секунда и така осигурява нарастващ фронт на сигнала на входа CU на всяка секунда.



Фиг. 6.3. Примерно включване на изместващ регистър

2. Задание

Бягащата светлина от предходната тема да се реализира с помощта на изместващ регистър.

3. Анализ

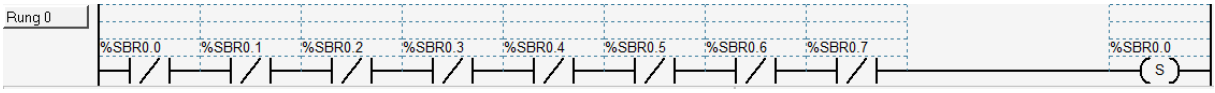
Особеното в тази реализация е, че регистърът е 16-битов, а с него трябва да се управляват осем изхода, т. е. трябва да се осигури повторение на цикъла на изместване при достигане на крайна лява или крайна дясна позиция на бягащата единица в регистъра. Това се реализира лесно чрез адресиране на съответните битове и операции за четене и запис. Вместо вътрешната променлива от тип дума, която е използвана в предходната реализация за съхраняване на текущата позиция на светлината (текущия номер на активния изход), тук се използва номерът на съответния бит от изместващия регистър. Тактовата поредица, която осигурява изместването, се реализира по същия начин както в предходната тема.

4. Разпределение на ресурсите

Всеки изход на контролера се управлява от един бит от изместващия регистър, като най-удобно е това да бъдат битове с номера от 0 до 7. За генериране на тактовата поредица се използват два таймера от тип TON, както в предходната реализация. Константите за настройка на таймерите в различните режими се изчисляват по същия начин.

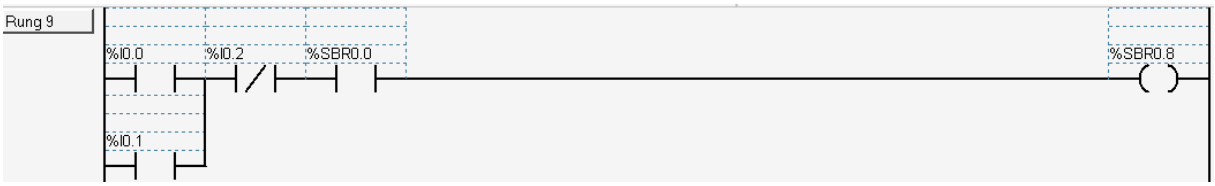
5. Особености на реализацията

В рънг 0 е показано началното установяване на регистъра, т. е. запис на единица в бит b0.

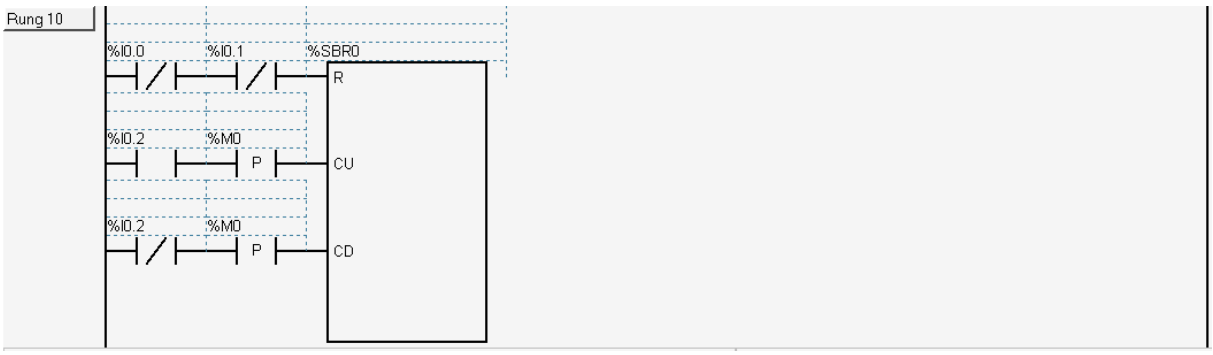


Следващият рънг показва връзката между даден бит от изместващия регистър и съответния изход, който той управлява.

В следващия рънг се прави подготовка за копиране на крайния десен бит в крайна лява позиция при следващо изместване надясно. По този начин се реализира завъртането на светлината. Подобен рънг трябва да бъде включен за осигуряване на изместването в обратна посока.



В Rung 10 е показано включването на изместващия регистър и условията за началното му установяване, за изместване наляво и за изместване надясно.



6. Въпроси и задачи

6.1. В рънг 10 към входовете за изместване са включени контакти, сработващи по нарастващ фронт на сигналите. Как смятате, ще работи ли правилно изместващият регистър, ако се използват контакти от вида „нормално отворен”?

6.2. Реализирайте бягаща светлина, като светодиодите светят по два едновременно.

6.3. Реализирайте брояч на Джонсън.

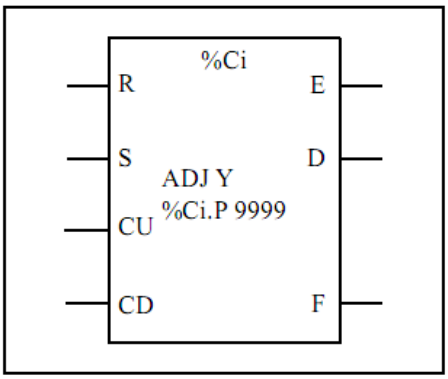
6.4. Обединете и трите реализации в една, за да се получи вид „светлинно шоу”.

Лабораторно упражнение № 4

Броячи

5. Функционален блок брояч

Броячите са функционални блокове, които отброяват някакви събития. Отброяването може да се извършва в права (сумиращ брояч) или в обратна посока (изваждащ брояч). Графичното изображение на функционалния блок е показано на фигура 7.1.



Фиг. 7.1. Функционален блок брояч

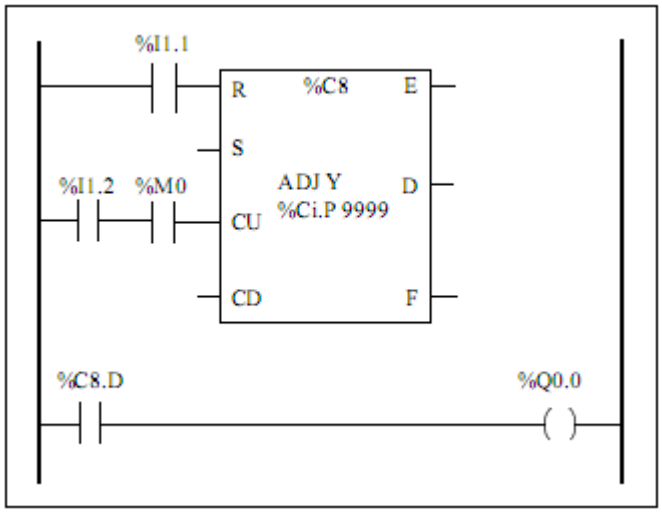
6. Параметри на броячите

- Номер на брояча ***i*** - ***%Ci***. За разглеждания контролер могат да бъдат конфигурирани до 128 брояча с номера от 0 до 127;
- Текуща стойност на брояча ***%Ci.V*** – променлива от тип дума, която се увеличава или намалява с единица в зависимост от активния сигнал на входовете ***CU*** и ***CD***. Съдържанието на тази дума може да бъде четено, но не може да бъде променяно от програмата.
- Инициализараща стойност ***%Ci.P*** – променлива от тип дума, която може да бъде четена и променяна от програмата. Чрез нея се задава коефициентът на броене на брояча. Може да приема стойности от 0 до 9999. Стойността по подразбиране е 9999.
- ***ADJ*** (***Y/N***) – може да приема стойност ДА или НЕ и показва дали е позволено стойността на ***%Ci.P*** да бъде променяна посредством редактора на анимационните таблици (Animation Tables Editor);
- Вход ***R*** – при логическа единица на този вход текущата стойност на брояча става равна на нула. Нулират се също и асоциираните изходи ***%Ci.E, %Ci.F*** и ***%Ci.D***. Този вход е с най-висок приоритет.
- Вход ***S*** - при логическа единица на този вход текущата стойност на брояча става равна на ***%Ci.P***, а асоциираният изход ***%Ci.D*** се установяват в единица.
- Вход ***CU*** – нарастващият фронт на сигнала на този вход предизвиква увеличаването с единица на текущата стойност на брояча ***%Ci.V***.
- Вход ***CD*** – нарастващият фронт на сигнала на този вход предизвиква намаляването с единица на текущата стойност на брояча ***%Ci.V***.
- Изход ***E*** (Empty) – изход за препълване при броене в режим на изваждане. Изходът ***E***, както и асоциираният с него бит ***%Ci.E*** се установяват в единица, когато текущата стойност на брояча, работещ в режим на изваждане, ***%Ci.V*** се променя от 0 към 9999. Изходът ***E*** и текущата стойност на брояча се нулират, когато започне нов цикъл на броене.
- Изход ***F*** (Full) – изход за препълване при броене в режим на събиране. Изходът ***F***, както и асоциираният с него бит ***%Ci.F*** се установяват в единица, когато текущата стойност на

брояча, работещ в режим на събиране, ***%Ci.V*** се променя от 9999 към 0. Изходът ***F*** и текущата стойност на брояча се нулират, когато започне нов цикъл на броене.

- Изход ***D*** (Done) – изход за достигане на инициализиращата стойност ***%Ci.P***. Установява се в логическа единица, когато текущата стойност на брояча ***%Ci.V*** се изравни с инициализиращата ***%Ci.P*** и се нулира при започване на нов цикъл на броене.

На фигура 7.2 е показано примерно включване на брояч.

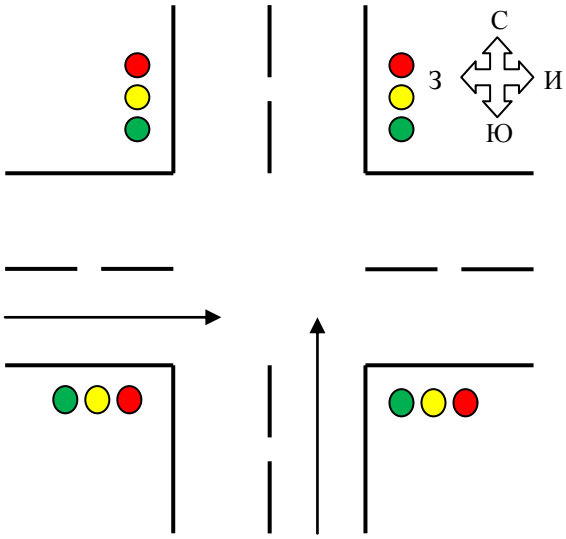


Фиг. 7.2. Пример за използване на брояч

Броячът е настроен за работа в режим събиране (използва се входът CU) с коефициент на броене 10000 (%Ci.P = 9999). Всеки нарастващ фронт на сигнала от логическата операция И между променливите %I0.2 и %M0 предизвиква нарастване с единица на текущата стойност на брояча, до достигането на предварително зададения коефициент на броене. Тогава изходът (и променливата %Ci.D) се установява в активно ниво единица, което активира изхода %Q0.0. Активното ниво на сигнала на входа %I1.1 установява брояча в начално състояние (в случая нула), прекъсвайки текущия цикъл на броене.

7. Примерно задание

Да се реализира управлението на светофарна уредба, чрез средствата на програмируемия логически контролер Twido.



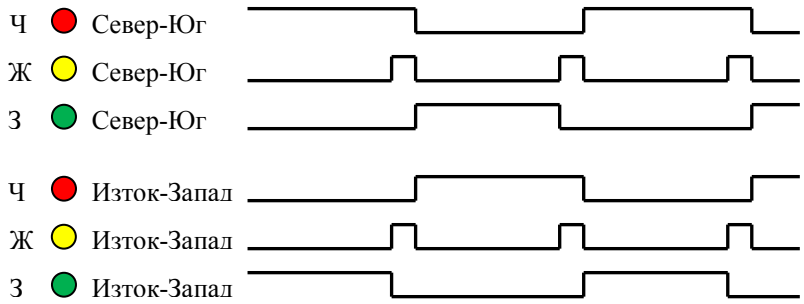
Фиг. 7.3. Регулируемо кръстовище

7.1. Анализ

За да се реши задачата, най-напред е необходимо да се изясни действието на светофарната уредба (виж фигура 7.3). Така например, когато свети червено („Стоп”) на секцията, отговаряща за движение по посока „Север-Юг“, на секцията „Изток-Запад“ свети зелено („Преминаването разрешено”). В следващия момент на секция „Изток-Запад“ угасва зеленото и светва жълто („Внимание предстои сигнал Стоп”), а на секцията „Север-Юг“ продължава да свети червено като за кратък период заедно с него светва и жълто („Внимание, приготви се за потегляне”). Веднага след жълтия сигнал на секция „Изток-Запад“ светва червено („Стоп”), а секцията „Север-Юг“

сигнализира със зелен сигнал („Преминаването разрешено”) и цикълът се повтаря.

На фигура 7.4 работата на светофарната уредба е илюстрирана с времедиаграма.



Фиг. 7.4. Времедиаграма на работа на светофара

Ако приемем за единичен времеви интервал ***t*** продължителността на светене на жълтия сигнал, то по времедиаграмата може да се направи разчет на продължителностите на светене и на другите сигнали, изразени чрез избрания единичен интервал. Така за отмерване на различните продължителности могат да се използват броячи, които ще отброяват необходимия брой единични времеви интервали за всяка светлина. Единичният времеви интервал може да се генерира по познатия начин с използването на два таймера, включени като мултивибратор. Така ще се осигури наличието на нарастващ фронт на сигнала за всеки единичен времеви интервал, по който ще се извършва броенето.

7.2. Разпределение на ресурсите

За генериране на единичния времеви интервал са необходими два таймера, включени като мултивибратор. За стартиране работата на светофарната уредба е необходимо да се използва един от цифровите входове на контролера. За светлините на двете секции на светофара е необходимо да бъдат използвани шест от цифровите изходи на контролера.

Броячите могат да бъдат реализирани по познатия начин чрез вътрешна променлива от тип дума, чието съдържание да се променя и да се следи програмно. Другият вариант е да се използва функционалният блок брояч.

За макета „светофар” се използват следните изходи на контролера: червено „север-юг” - %Q0.0; жълто - %Q0.1; зелено „север-юг” - %Q0.2; червено „изток - запад” - %Q0.3, зелено „иззток - запад” - %Q0.4;

7.3. Настройки на таймерите и броячите

За да се изчислят необходимите константи за настройка на таймерите, се използват познатите формули:

$$T = 2t, \quad \text{т. е.} \quad t = ½*T$$

$$t = TB * TMi.P \;\; \text{-->} \;\; TMi.P = t/TB \;\; \text{-->} \;\; TMi.P = T/2*TB$$

За да се изчислят коефициентите на броене на броячите, трябва да се определят продължителностите на светене на отделните светлини, изразени чрез броя единични времеви интервали. От времедиаграмата се вижда следното (разбира се, това са примерни константи):

Цвят	Свети	Не свети
Червено	7t	7t
Жълто	1t	6t
Зелено	6t	8t

Задължително трябва да се отчете и дефазирането на светлините за двете секции на светофара.

Реализирайте програмата самостоятелно, като използвате функционални блокове броячи, а също и като използвате вътрешна променлива за брояч.

4. Задачи за самостоятелна работа

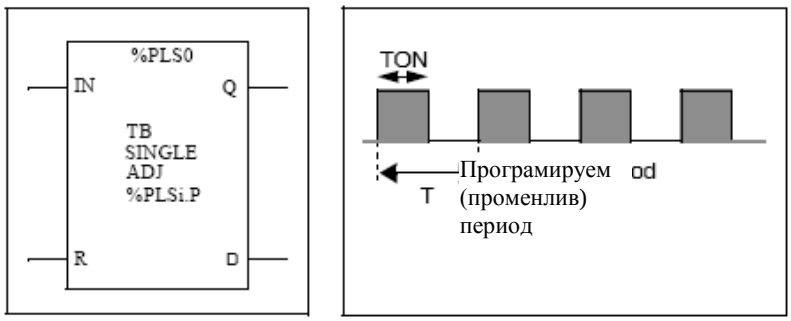
4.1. Модифицирайте програмата, като усложните работата на светофарната уредба, въвеждайки секции за ляв и десен завой.

Лабораторно упражнение № 5

Генератор на правоъгълни импулси

1. Общи сведения

Функционалният блок **генератор на правоъгълни импулси** - %PLS - се използва за генериране на поредица от правоъгълни импулси с коефициент на запълване 50% и програмируема продължителност на периода на повторение (виж фигура 8.1). В конфигурацията на разглеждания контролер са включени два генератора на правоъгълни импулси. Генерираните импулси се насочват към асоциираните изходни канали %Q0.0.0 и %Q0.0.1.



Фиг. 8.1. Генератор на правоъгълни импулси – графично представяне и времедиаграма

Продължителността на импулса TON е равна на половината от периода T.

$TON = T / 2 = \%PLSi.P * TB$ - за базови константи 0,142 ms и 0,57 ms;

$TON = [int(\%PLSi.P / 2)] * TB$ - за базови константи 0,10 ms и 1 s.

Продължителността на периода *T* се изчислява по формулата:

$T = \%PLSi.P * TB$

1.6. Параметри на импулсния генератор

Параметрите на генератора на правоъгълни импулси са представени в таблица 8.1.

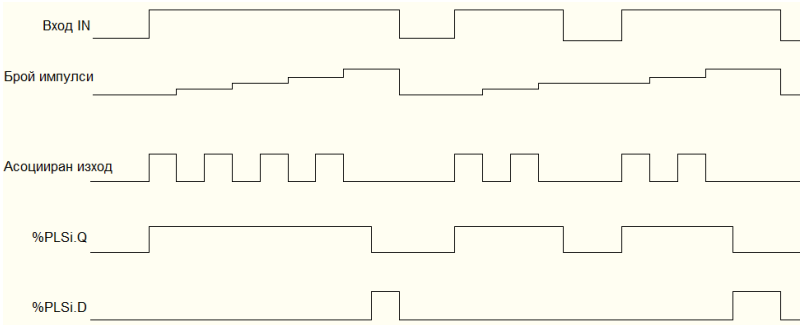
Таблица 8.1. Параметри на импулсния генератор

Функция	Означение	Описание
Номер	i	Показва номера на функционалния блок. i = 1, 2
Базова времева константа	TB	0,142 ms, 0,57 ms, 10 ms, 1 s
Инициализираща константа	%PLSi.P	%PLSi.P = 0 не се използва; 1 < %PLSi.P <= 32762 за TB = 10 ms и 1s; 1 < %PLSi.P <= 255 за TB = 0,142 ms и 0,57 ms. За постигане на по-добра точност при базови константи 10ms и 1s се препоръчва %PLSi >= 100, ако е нечетно число.
Брой на генерираните импулси	%PLSi.N %PLSi.ND	Константи, които определят броя на генерираните импулси: 0 < %PLSi.N <= 32767 в нормален режим 0 < %PLSi.ND <= 4294967295 в режим двойна дума По подразбиране %PLSi.N = 0 (също и %PLSi.ND = 0) и в този случай се генерират неограничен брой импулси.

Регулиране (ADJ)	Y/N	Ако е избрано Y, тогава е възможно да се променя стойността на %PLSi.P чрез Animation Tables Editor. Ако е избрано N, тогава няма достъп до тази стойност.
Вход на импулсния генератор	IN	При логическа единица на този вход се генерират импулси на асоциирания изход.
Вход за принудително преустановяване на генерирането на импулси	R	При логическа единица на този изходите %PLSi.Q и %PLSi.D се установяват в нула. Броячът на генерираните импулси също се установява в нула.
Изход за работно състояние на генератора	%PLSi.Q	Състояние логическа единица показва, че се генерират импулси на асоциирания изходен канал.
Изход за край на генерирането на импулси	%PLSi.D	Установява се в логическа единица, когато определеният брой импулси е генериран.

1.7. Времедиаграма

Времедиаграмата, илюстрираща работата на импулсния генератор, е представена на фигура 8.2. В примера е прието, че желаният брой импулси е четири. Вижда се, че при отпадане на входния сигнал генерирането на импулсите се прекратява, но броячът на генерираните импулси не се нулира. При възстановяване на входния сигнал генерирането продължава до достигане на предварително зададения брой.



Фиг. 8.2. Времедиаграма на работа на импулсния генератор %PLS

1.8. Области на допустимите периоди T

От 0,142 ms до 36,5 ms през стъпки от 0,142 ms (т. е. TB = 0,142 ms). Честотата на генерираните импулси се изменя от 27,4 Hz до 7 KHz.

От 0,57 ms до 146 ms през стъпки от 0,57 ms (т. е. TB = 0,57 ms). Честотата на генерираните импулси се изменя от 6,84 Hz до 1,75 KHz.

От 20 ms до 5,45 min през стъпки от 10 ms (т. е. TB = 0,10 ms).

От 2 s до 9,1 h през стъпки от 1 s (т. е. TB = 1 s).

2. Задание

Да се състави програма, управляваща премигването на лампа (светодиод) с различна продължителност на светене и издаване на звуков сигнал с различна честота. Продължителностите на светене на лампата, както и честотите на звуковите сигнали се

задават с комбинация от бутони К1 до К4 (със задържане), както е показано в следващата таблица.

К1	К2	К3	К4	Време на светене	Честота на звука
0	1	*	*	0,5 s	-
1	0	*	*	0,25 s	-
1	1	*	*	1 s	-
*	*	0	1	-	1 kHz
*	*	1	0	-	2 kHz
*	*	1	1	-	4 kHz

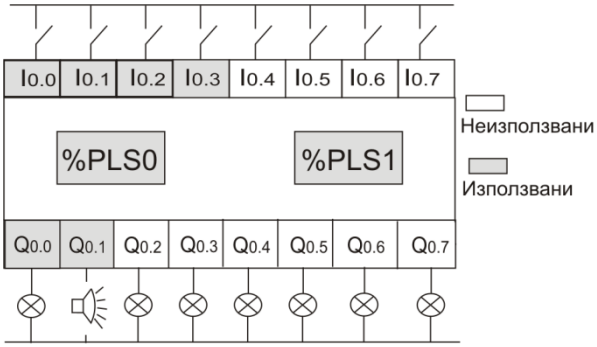
2.1. Анализ

За реализиране на задачата е необходимо да бъдат използвани два импулсни генератора %PLS0 и %PLS1. Генераторът %PLS0 се използва за управление на светодиод, свързан с асоциирания му изход %Q0.0.0. Генераторът %PLS1 се използва за управление на зумер, свързан с асоциирания му изход %Q0.0.1. При задаване на съответната комбинация от бутони К1 и К2, светодиодът на изход %Q0.0 премигва с определена продължителност на светене. Комбинацията от бутони К3 и К4 определя произвеждането от зумера на звук с определена честота.

2.2. Разпределение на ресурсите

Бутоните К1, К2, К3, К4 се свързват към цифровите входове на контролера, съответно %I0.0, %I0.1, %I0.2, %I0.3. С комбинациите от бутоните К1 и К2 (%I0.0 и %I0.1) се определя времето на мигане на светлината, а с К3 и К4 (%I0.2 и %I0.3) - честотата на произвеждания звук. За визуализация на резултата се използват изходите %Q0.0 за светлината и %Q0.1 за зумера.

На фигура 8.3 е показано символично разпределението на ресурсите на контролера за решаването на поставената задача.



Фиг. 8.3. Разпределение на ресурсите

2.3. Изчисляване на константите за настройка на генераторите

Известно е, че периодът на повторение на импулсите е обратно пропорционален на честотата на повторението им, т. е.

$$T = 1/f \qquad и \qquad f = 1/T$$

От друга страна, периодът на повторение на импулсите, издавани от генератора, се определя чрез съотношението:

$$T = \%PLSi.P * TB$$

За честота f = 1 KHz периодът на повторение е $T = \frac{1}{1.10^3} = 1ms$.

Най-подходящата времева база за генериране на импулси с тази честота е TB = 0,142ms.

За константата %PLS1.P се получава:

$$\%PLS1.P = \frac{T}{TB} = \frac{1.10^{-3}}{0{,}142.10^{-3}} = 7{,}04 \approx 7$$

За честота 2 KHz се получава $\%PLS1.P \approx 4$, а за честота 4 KHz - $\%PLS1.P \approx 2$.

За определяне на инициализиращите константи за генератора %PLS0, се извършват следните изчисления:

$$TON = [int(\%PLSi.P / 2)] * TB$$

За TON = 0,5 s и TB = 10 ms се получава:

$$\%PLS0.P = \frac{2.TON}{TB} = \frac{2.0{,}5}{10.10^{-3}} = 100$$

За TON = 1s %PLS0.P = 200 и за TON = 0,25s %PLS0.P = 50.

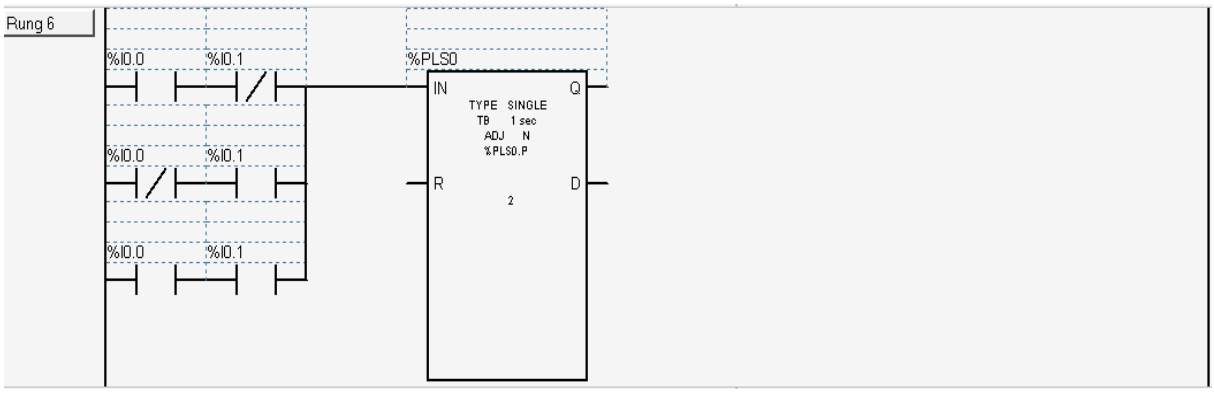
Ако трябва да се постигне по-добра точност, тогава е удачно да се избере базова времева константа TB = 1 ms. Изчислете самостоятелно инициализиращите стойности в този случай.

2.4. Фрагменти от Ladder диаграмата

В първите три рънга се прави проверка за комбинацията от входове, според които се задава съответната стойност на %PLS0.P за времената на мигане на лампата.

Аналогично в следващите три рънга се прави проверка за комбинацията от входове, които определят честотата на звуковия сигнал, издаван от зумера и се задава съответната стойност на %PLS1.P.

Импулсният генератор %PLS0.P работи, когато е налице валидна комбинация от бутоните К1 и К2, респективно – валидна комбинация на входовете %I0.0 и %I0.1.



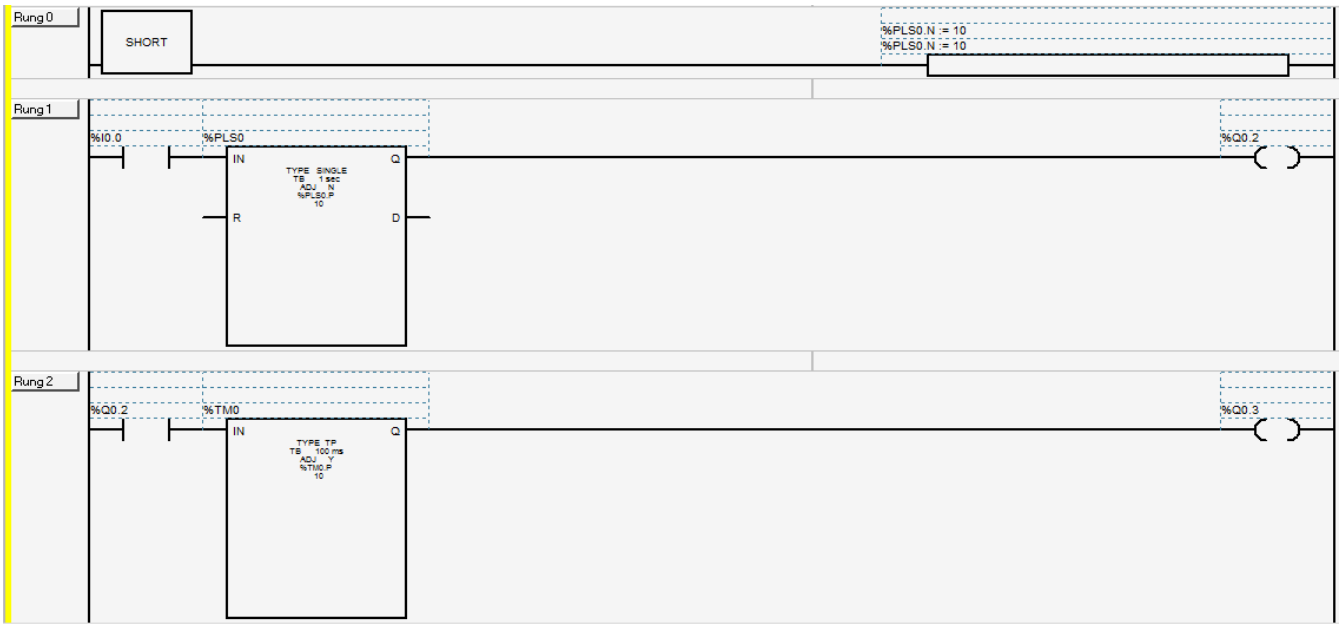
Аналогично, импулсният генератор %PLS1.P работи, когато е налице валидна комбинация от бутоните К3 и К4, респективно – валидна комбинация на входовете %I0.2 и %I0.3.

И двата генератора генерират безкрайно много импулси. Ако желаем да фиксираме броя на импулсите, трябва да зададем съответната стойност на параметъра %PLSi.N.

3. Въпроси и задачи за самостоятелна работа

3.1. Сравнете интензивностите на светене на двата светодиода, свързани с цифровите изходи на контролера %Q0.0 и %Q0.1. Обяснете на какво се дължи разликата.

3.2. Зададен е фрагмент от Ladder диаграма (фигура 8.4). Илюстрирайте с времедиаграма съвместната работа на функционалните блокове и сигналите на използваните изходи на контролера.



Фиг. 8.4.

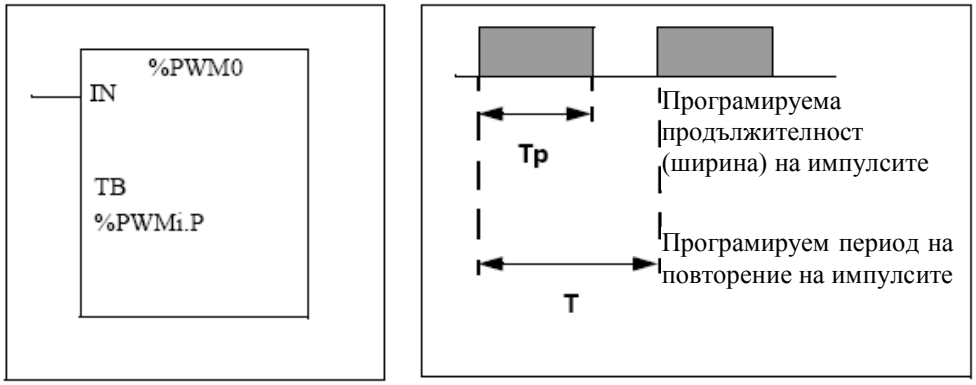
Лабораторно упражнение № 6

Генератор на широчинно модулирани импулси

2. Общи сведения

Функционалният блок **генератор на широчинно модулирани импулси** - %PWM се използва за генериране на правоъгълни импулсни сигнали с променлива продължителност (ширина). В разглеждания контролер има вградени два такива функционални блока, като всеки от тях има асоциран съответен изход %Q0.0.0 и %Q0.0.1, на които се получават генерираните правоъгълни импулси с променлива ширина. На фигура 9.1 е показано условното означение на блока, както и времедиаграмата на генерираните импулси.

Контролерите с релейни изходи не поддържат тази функция заради честотното ограничение. %PLS блоковете си съперничат с %PWM блоковете за използване на асоциираните изходи, което означава, че трябва да се избере между двата (т. е. в една програма могат да се използват или само %PLS, или само %PWM, или по един от двата вида).



Фиг. 9.1. Условно означение на блока и времедиаграма на генерираните импулси

Периодът на повторение на генерираните импулси T се задава в процеса на конфигуриране с избор на времевата база TB и инициализиращата константа %PWMi.P, чрез зависимостта

$$T = TB * \%PWMi.P$$

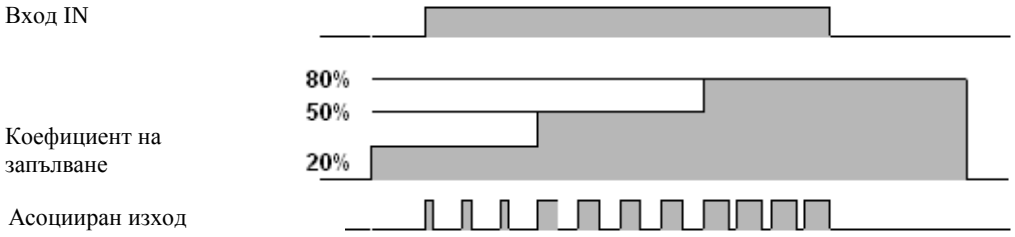
3.3. Параметри на генератора

Таблица 9.1. Параметри на %PWM

Параметър	Етикет	Описание
Времева база	TB	0,142 ms, 0,57 ms, 10 ms, 1 s
Инициализираща константа	%PWMi.P	0 < %PWMi.P <= 32767 с TB = 10 ms или 1 s 0 < %PWMi.P <= 255 с TB = 0,57 ms или 0,142 ms %PWMi.P = 0 не се използва
Коефициент на запълване	%PWMi.R	Определя продължителността на импулса в проценти като част от периода. $T_p = (\%PWMi.R / 100) * T$ %PWMi.R приема стойност по подразбиране 0, а стойности по-големи от 100, се определят като 100.
Вход за разрешаване на работата на генератора	IN	При активно ниво на входния сигнал се генерират импулси към асоциирания изход. При неактивно ниво на входния сигнал генераторът не работи и сигналът на асоциирания изход е с ниво нула.

3.4. Времедиаграма

Времедиаграмата на работа на генератора е показана на фигура 9.2. Вижда се промяната на продължителността на генерираните импулси в зависимост от стойността на променливата %PWMi.R (Ratio). Периодът на повторение на импулсите остава един и същ.



Фиг. 9.2. Времедиаграма на работа на %PWM

4. Задача за изпълнение

В една сграда има две еднакви помещения. Във всяко от тях има по 3 пожарни датчика, бутон за ръчно включване и бутон за ръчно изключване на общата сигнализация. За всяко помещение е предвидена и сигнална лампа. Ако в някое от помещенията се активира поне един от датчиците, светва сигналната лампа на съответното помещение. Лампата изгасва, ако всички датчици в помещението са изключени (не са активни). Ако в някое от двете помещения се задействат кои да са два датчика, се включва общата пожарна сигнализация - звуков сигнал и мигаща светлина. Тя работи, докато не бъде изключена чрез съответния бутон. Сигнализацията може да бъде тествана за работоспособност чрез включването й с бутон за включване, независимо от състоянието на датчиците.

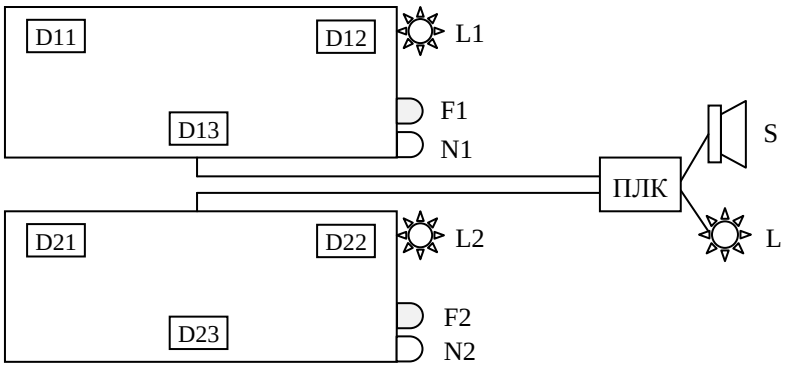
Сигнализацията за пожар се изразява в следното:

1) Сирената издава звуков сигнал, започвайки от честота 7 KHz, като на всеки 5 секунди периодът на повторение на импулсите се увеличава с една времева база, но не повече от 5 пъти, след което започва отначало.

2) Лампата свети с мигаща светлина, като мигането (периодът) е през 0,5 секунди, а продължителността на светене започва от 50 милисекунди и на всеки 5 секунди се увеличава с 10% от периода, докато достигне до пълно светене, след което започва отначало.

2.1. Анализ

На фигура 9.3 е показано условното разположение на елементите на противопожарната инсталация. D11, D12 и D13, както и D21, D22 и D23 са пожарните датчици в помещение 1 и помещение 2, респективно. L1 и L2 са сигналните лампи за всяко помещение. N1 и N2 са бутоните за включване на сигнализацията с цел тестване, а F1 и F2 са бутоните за изключване на сигнализацията след отстраняване на причините за активирането й. L и S са съответно лампата и сирената за цялата противопожарната инсталация.



Фиг. 9.3. Елементи на противопожарната инсталация

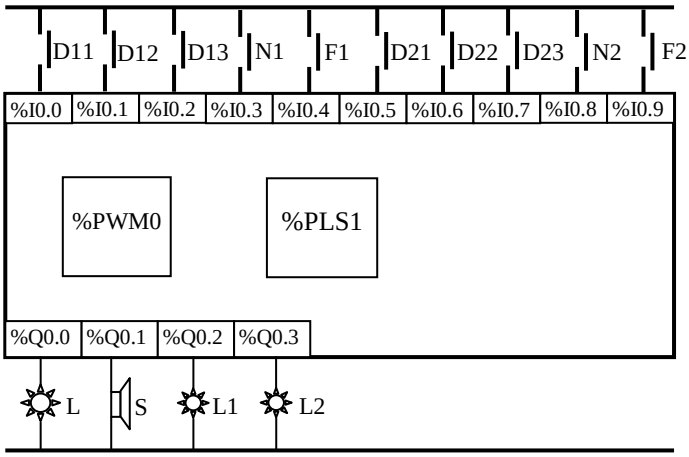
2.2. Разпределение на ресурсите

Датчиците (6 на брой) и бутоните (общо 4 на брой) ще бъдат свързани към цифровите входове на контролера.

Сирената трябва да издава звуци с различни честоти. За целта към нея трябва да бъдат подавани правоъгълни импулси, следващи със съответната честота. За целта е удобно да се използва генератор на правоъгълни импулси. За да бъде симулирана работата на сирената чрез лабораторния макет, е удобно за звука да бъде използван генераторът %PLS1, тъй като неговият асоциран изход е свързан със зумера.

Сигналната лампа L свети с мигаща светлина, през равни интервали от време, но времето за светене се променя. Това означава, че към нея трябва да бъдат подавани правоъгълни импулси с един и същ период на повторение, но с различна продължителност, т. е. с различен коефициент на запълване. Това налага използването на генератор на широчинно модулирани импулси и по-точно генераторът %PWM0, свързан с асоциирания изход %Q0.0.0, а на макета - физически свързан с изхода %Q0.0.

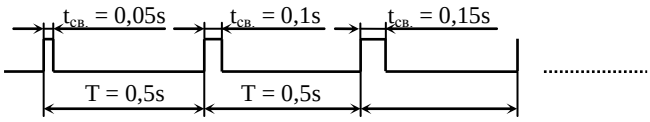
На фигура 9.4 е показано окончателно разпределение на ресурсите.



Фиг. 9.4. Разпределение на ресурсите на контролера за реализиране на задачата за противопожарната инсталация

2.3. Изчисления

- *За мигащата светлина* (фигура 9.5):



Фиг. 9.5. Мигаща светлина

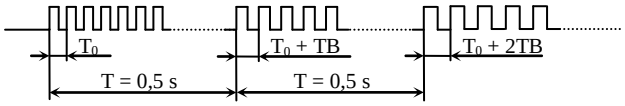
За генератора %PWM0 избираме времева база TB = 10 ms, тъй като периодът на повторение на импулсите е 0,5s и това е най-подходящата стойност от всички възможни. От формулата за периода на импулсите може да се определи инициализиращата стойност на генератора:

$$\%PWM0.P = \frac{T}{TB} = \frac{0{,}5}{10*10^{-3}} = 50.$$

Коефициентът на запълване *%PWM0.R* се променя от 10 до 100.

- *За звука:*

Това, което трябва да се постигне, е генериране на правоъгълни импулси с определена честота (период на повторение), която трябва да се променя през 5 секунди (фигура 9.6).



Фиг. 9.6. Промяна на честотата на звука

За да се генерира звук с честота 7 KHz, е подходящо да бъде избрана времева база TB = 0,142 ms. Така ще се постигне възможно най-добра точност. Трябва да се пресметне инициализиращата стойност на генератора, използвайки формулата за връзката между честотата и периода на повторение на импулсите:

$$f = \frac{1}{T} \quad \text{и} \quad T = \frac{1}{f}.$$

При f = 7 KHz за периода T се получава:

$$T = \frac{1}{f} = \frac{1}{7*10^3}.$$

За инициализиращата стойност се получава:

$$\%PLS1.P = \frac{T}{TB} = \frac{\frac{1}{7*10^3}}{0{,}142*10^{-3}} \approx 1.$$

Тъй като има изискване периодът на повторение на импулсите да се увеличава или намалява с една времева база, трябва да се определи върху кой параметър на генератора ще се отрази това. Ако приемем, че началната стойност на периода е

$$T_0 = TB * \%PLS1.P,$$

то следващата стойност, увеличена с една времева база, ще бъде:

$$T_1 = T_0 + TB = TB * \%PLS1.P + TB = TB * (1 + \%PLS1.P).$$

Това означава, че промяната на периода на импулсите ще се постигне с промяна на инициализиращата стойност.

В заданието е казано, че промяната на продължителността на светене на лампата, както и на честотата на звука трябва да става през 5 секунди. Следователно на всеки 5 секунди трябва да се генерира сигнал, който да предизвиква съответната промяна. Помислете как ще реализирате този сигнал.

Възможно е в началото генераторът за звука да не генерира импулси. Най-вероятно това се дължи на ваша грешка. Каква е тя?

3. Въпроси

3.1. Може ли генераторът на правоъгълни импулси (PLS) да бъде заменен с генератор на широчинно модулирани импулси (PWM)? Обосновете отговора си.

Лабораторно упражнение № 6

Обработване на аналогови сигнали

5. Общи сведения

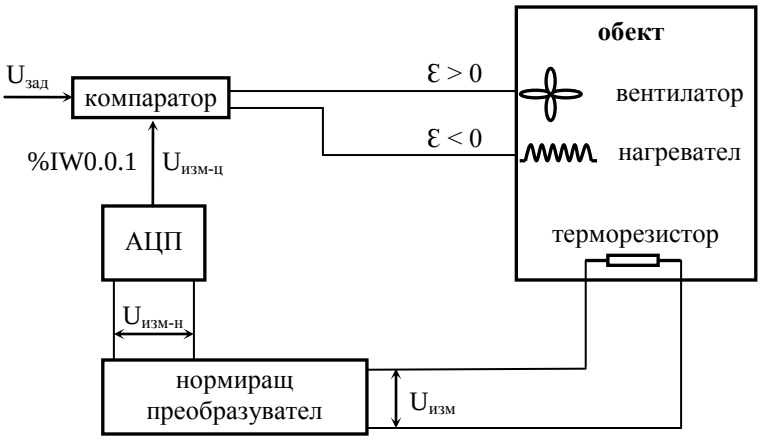
Модулният контролер TWDLMDA20DTK (както и другите модулни контролери от серията Twido - TWDLMDA20DUK, TWDLMDA20DRT, TWDLMDA40DTK и TWDLMDA40DUK) имат вграден аналогов канал, чрез който могат да се обработват аналогови сигнали. Аналоговото входно напрежение може да бъде в диапазона от 0 до 10 волта. Това напрежение постъпва на входа на вграден аналогово-цифров преобразувател (АЦП), който го преобразува в число в диапазона от 0 до 511, т. е. АЦП е 9-битов. Преобразуваното в число напрежение се записва във вътрешната системна дума %IW0.0.1. Тази системна дума е достъпна за потребителя само за четене.

6. Задачи за изпълнение

6.1. Да се състави модел на терморегулатор, който да осигури поддържането на постоянна температура, например 35°C, в обект (термокамера). Загряването на въздуха се осигурява от нагревател, а охлаждането – от вентилатор. Като датчик за температура се използва терморезистор (например от Pt100, за който материал е известно, че има линейно изменение на съпротивлението в много широк температурен диапазон).

6.2. Анализ и изчисления

На фигура 10.1 са показани условно управляваният обект и връзката му с управляващото устройство, базирано на ПЛК.



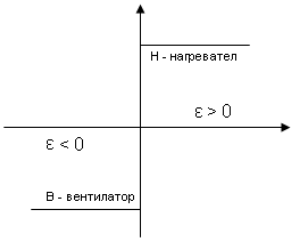
Фиг. 10.1. Управляваният обект и връзката му с управляващото устройство, базирано на ПЛК

Измерването на температурата става чрез измерване на спада на напрежението върху терморезистора – $U_{изм}$. В общия случай това напрежение най-вероятно ще се колебае в някакви малки граници. За да се осигури максимална точност на измерването, е необходимо то да бъде така променено, че тези колебания да бъдат в диапазона от 0 до 10 волта, в който диапазон може да бъде входното напрежение на АЦП. Тази промяна се извършва от нормиращ преобразувател, на изхода на който се получава напрежението $U_{изм-н}$ (измерено - нормирано). АЦП преобразува това напрежение в число $U_{изм-п}$ (измерено – преобразувано) и го записва във вътрешната системна дума %IW0.0.1.

За да се поддържа постоянна температурата в камерата, трябва да се извършва непрекъснато сравняване на измерената стойност със зададената стойност и в зависимост от разликата между тях да се включва или нагревател, или вентилатор.

$$\varepsilon = T^{o}зад - T^{o}изм$$

При ε < 0 се включва вентилатор, а при ε > 0 – нагревател (фигура 10.2).



Фиг. 10.2. Включване на вентилатор или нагревател в зависимост от измерената температурна разлика

Разбира се, няма да се сравняват температури, а напреженията, съответстващи на тези температури, т. е. $U_{зад}$ и $U_{изм-п}$. За целта трябва да бъдат направени необходимите изчисления.

Приемаме, че температурата се изменя в границите от $0^{o}C$ до $50^{o}C$ - с известен аванс, за да се осигури възможност за първоначално влизане в режим. Така, след измерване и нормиране, на температура $0^{o}C$ ще съответства напрежение $U_{изм-н}$ = 0V, а на температура $50^{o}C$ ще съответства напрежение $U_{изм-н}$ = 10V. След аналого-цифровото преобразуване на тези стойности ще съответстват числата 0 и 511 респективно. Лесно могат да се изчислят съответствията, показани в таблицата.

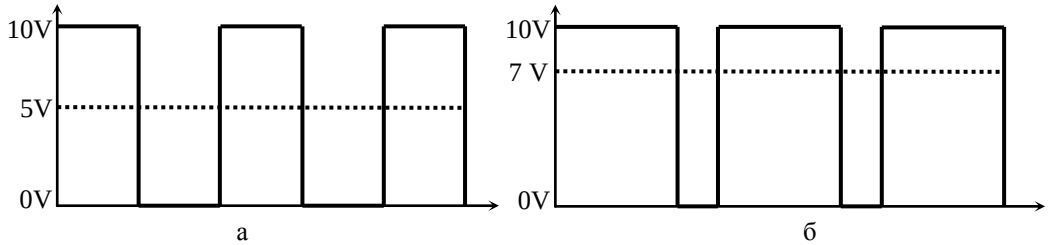
Температура	**$U_{изм-н}$**	**%IW0.0.1**
0°	0 V	0
5°	1 V	51
35°	7 V	358
50°	10 V	511

От тези съответствия следва, че опорната стойност $U_{зад}$, с която трябва да се сравнява $U_{изм-н}$, е 358.

Тъй като в лабораторни условия няма как да се получи динамично изменение на температурата, то промяната на $U_{изм}$ може да се осигури чрез потенциометъра R20 (виж фигура 2.29).

2.3. Симулиране на аналогов сигнал чрез генератор на широчинно модулирани импулси

За целите на изследванията аналоговият сигнал може да бъде симулиран чрез генератор на правоъгълни импулси с променлив коефициент на запълване (ШИМ). Ако импулсите, които генераторът генерира, са с размах 10 V, то минималният коефициент на запълване на импулсите 0% ще съответства на напрежение 0 V, а максималният (100%) – на напрежение 10 V. Коефициент на запълване 50% ще съответства на напрежение 5 V (фигура 10.3а). Така, регулирайки коефициента на запълване на импулсите, може да се регулира напрежението, симулиращо аналоговия сигнал.



Фиг. 10.3. Симулиране на аналогов сигнал чрез ШИМ. а – коефициент на запълване 50%, б – коефициент на запълване 70%

За целите на поставената задача коефициентът на запълване трябва да бъде 70%, тъй като целевата температура T^{o}зад = 35°C е на 70% (фигура 10.3б) от целия температурен диапазон (0°C до 50°C), който ще бъде измерван.

За да се бъде максимално близък до аналоговия сигнал, импулсният сигнал трябва да бъде с възможната най-висока честота. Това налага използването на генератор PWM с базова времева константа TB = 0,142 милисекунди и инициализираща стойност %PWM0.P = 1.

Подаваното на входа на АЦП напрежение може да се види на цифровия волтметър, вграден в макета СКС (виж фигури 2.28 и 2.29).

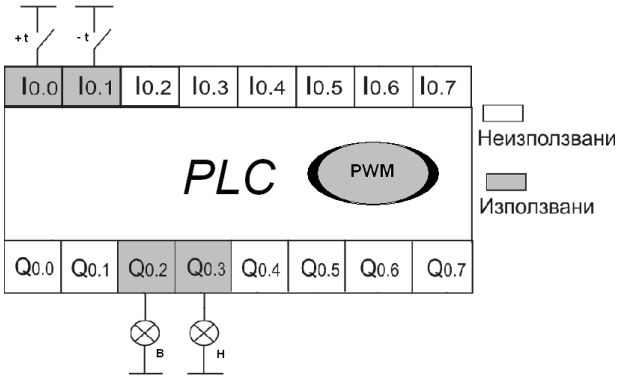
2.4. Разпределение на ресурсите

В случай на симулиране на аналоговия сигнал с ШИМ, бутоните, свързани с входове %I0.0 и %I0.1, могат да се използват съответно за увеличаване и намаляване на коефициента на запълване на импулсите %PWM0.R (респективно – на измерената температура). Тъй като на лабораторния макет тези бутони са от вида „със задържане”, то за по-бърза промяна на %PWM0.R може да се използва контакт от вида „сработващ по фронт”, при това както по нарастващ, така и по падащ.

Ако изменението на температурата се симулира с промяна на съпротивлението чрез потенциометъра, тогава не са необходими никакви цифрови входове за решаването на задачата.

И в двата случая трябва да се изберат два цифрови изхода, които ще управляват включването и изключването на вентилатора и нагревателя.

Примерно разпределение на ресурсите е показано на фигура 10.4.



Фиг. 10.4. Разпределение на ресурсите при симулиране на аналогов сигнал с ШИМ

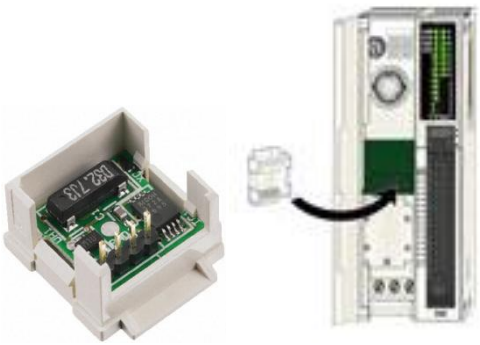
И в двата случая на решаване на задачата, заради стъпката на промяна на „измерената” температура/напрежение, близо до желаната температура/напрежение ще се получава едно непрекъснато включване и изключване („мъждене”) ту на вентилатора, ту на нагревателя. За да се избегне това, е добре да се предвиди някакъв „хистерезис”.

Лабораторно упражнение № 7

Барабанен контролер. Часовник за реално време. Вход за старт/стоп. Инструкции за преход. Подпрограми

1. Часовник за реално време

Часовникът за реално време (RTC - real time clock) е допълнителен модул, който се включва към базовия процесорен модул и чието предназначение е да следи настоящото време (фигура 11.1). Часовникът за реално време присъства в почти всички електронни устройства, в които е необходимо да се следи точно време. В процесорния модул часовникът за реално време също се използва за синхронизиране работата на отделните блокове по план. RTC има изключително малка консумация на ток и когато контролерът се изключи или по някаква причина отпадне захранването му, той запазва работоспособността си до 1000 часа при 25^0C или 300 часа при 55^0C.



Фиг. 11.1. RTC - Часовник за реално време

1.1. Конфигуриране на RTC

Конфигурирането на RTC изисква въвеждането на коефициент за корекция. Всеки RTC има такъв коефицент, който е записан на етикета му. Стойността на коефициента е в десетичен код и е в диапазона от 0 до 127. Той може да се въведе по два ръчни начина - през програмната среда TwidoSuite (или TwidoSoft) или чрез операторския дисплей TWDXCPODM.

1.1.1. Ръчно конфигуриране чрез програмна среда TwidoSuite на корекционна стойност, стъпки:

1. Изберете **Program => Debug и connect,** като изберете свързване по ваш избор.
2. Изберете **Check PLC => Configure** RTC от панела с инструменти. Опциите за настройка на RTC се появяват на дисплея и те трябва да бъдат настроени.
3. Въведете десетична стойност между 0 и 127. Вижте етикета върху корпуса на RTC за тази стойност. Стойността по подразбиране е 0.
4. Щракнете върху Apply, за да се приеме въведената стойност и затворете диалоговия прозорец. Щракнете върху Cancel, за да затворите диалоговия прозорец, без да се правят промени.

Въведената RTC корекционна стойност се добавя към компютърната конфигурация и към конфигурацията на контролера (ако е свързан).

1.1.2. Ръчно конфигуриране чрез програмна среда TwidoSuite на точно време - стъпки:

1. Изберете **Adjust RTC Time** от **Program** => **Debug** => **Check PLC task tool panel.**

2. Въведете текущата дата в полето Date. За да е валидна датата, тя трябва да се въведе във формат ден / месец / година (DD / MM / YYYY). Например: 28-ми Март 2015 се вписва като 28/03/2015.

3. Въвеждане на текущото време в Time. За да е валиден часът, трябва да се въведе в стандартен формат за часа (HH: MM: SS). Например, 02:15:54 PM е 14:15:54.

Забележка: Друг начин да настроите RTC е да щракнете върху бутон Synchronize, който актуализира датата и часа от PC часовник и показва тези стойности.

1.1.3. Програмно конфигуриране чрез програмна среда TwidoSuite

Числовите стойности, които се въвеждат в служебните думи, обслужващи часовника за реално време, са в десетичен формат /Decimal/, контролерът ги преобразува в двоичен формат /Binary/ и ги извежда на дисплейния модул в шестнадесетичен /Hexadecimal/ формат. Следователно, за да получим точно време, трябва да въвеждаме десетичната стойност на шестнадесетичното число или с други думи, за да въведем например 22 часа и 55 минути, служебната дума, отговаряща за часа и минутите, ще има следния вид:

%SW51:=8789 /Hex=2255/ HHMM/Dec=8789/

Така и за останалите служебни думи, обслужващи RTC:

1. Корекционната стойност се въвежда в служебната дума %SW60. Пример: %SW60:= 4 за корекционна стойност 0004.
2. Въвеждането на времето става с помощта на служебните думи от %SW49 до %SW53.

%SW49 := x-ти ден от седмицата. Пример: x=0001 за понеделник; x=0002 за вторник и т. н. до x = 0007 - за неделя.

%SW50:= 00SS секунди. Пример: %SW50:=0009 за 9-тата секунда.

%SW51:= HHMM час и минути. Пример: %SW51:=4181 за 10:55./

%SW52:= MMDD месец и ден. Пример: %SW52:=4357 за 5 ноември.

%SW53:= година. Пример: %SW53:= 8209 за 2011 година.

Тези думи са контролирани от системата, когато %S50 е установен в 0 и стойностите в служебните думи не могат да се променят. Стойностите в тези думи могат да бъдат въвеждани от потребителската програма или от терминала, когато %S50 е установен в 1. Когато %S50 стане равно на нула, RTC се актуализира от стойностите, записани в тези думи.

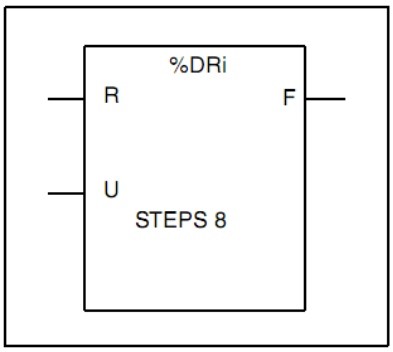
1.1.4. Настройване чрез операторския дисплей TWDXCPODM

Освен софтуерно, часовникът за реално време позволява да бъде настроен чрез операторския дисплей TWDXCPODM и изпълнението на следните стъпки:

1. Посредством навигационните бутони на малкия дисплей на TWDXCPODM се избира полето за въвеждане на корекционна стойност: *RTC Corr*
2. Въвежда се десетична стойност между 0 и 127 (по подразбиране тази стойност е 0).
3. Въвеждането се потвърждава с бутон „OK” и се отменя с избор на бутон „esc”. След потвърждаване на малкия екран се извежда текущото време и дата, а в отделна опция - и текущата корекционна стойност.

2. Барабанен контролер

Функционалният блок **барабанен контролер** (Drum Controller Function Block - %DR) (фигура 11.2) работи по подобие на електромеханичния барабанен контролер, който си превключва стъпките при настъпването на някакви външни събития. При всеки нарастващ фронт на сигнала, постъпващ на входа U на контролера, той се „превърта” с една стъпка, като установява в активно ниво 1 маркираните изходи %Qi.j или вътрешните битови променливи %Mi, и двата вида наричани „управляващи битове”.



Фиг. 11.2. Функционален блок **барабанен контролер**

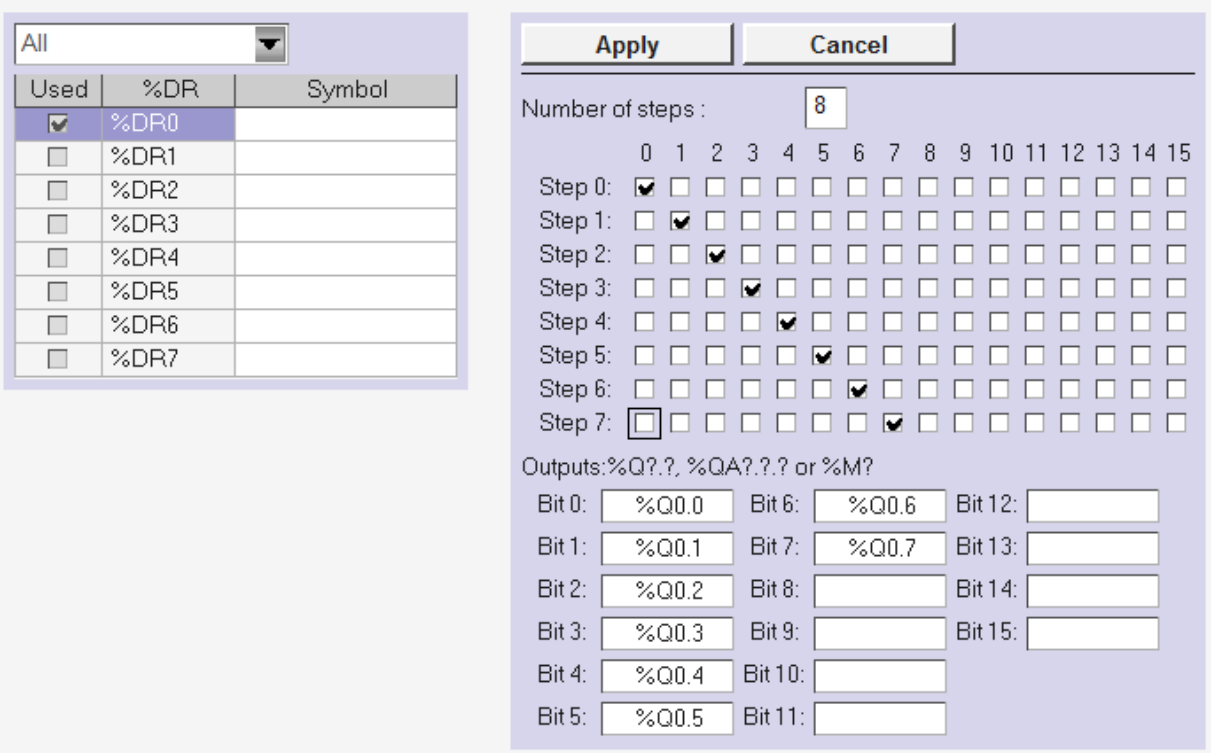
В таблица 11.1 са показани параметрите на барабаннния контролер.

Таблица 11.1. Параметри на барабанния контролер

Параметър	Етикет	Стойност
Номер	%DRi	0 до 3 за компактните контролери и 0 до 7 за модулните
Номер на текущата стъпка	%DRi.S	0 < %DRi.S < 7. Дума, която може да бъде четена и която може да се променя. Стойността, която се записва, трябва да бъде непосредствено десетично число. При записването ефектът се отразява върху следващото изпълнение на функционалния блок.
Брой стъпки		1 до 8 (по подразбиране)
Вход за връщане към стъпка 0	R (Reset)	В състояние 1, установява контролера в стъпка 0.
Вход за превъртане	U (Upper)	Нарастващият фронт на сигнала, постъпващ на този вход, предизвиква превъртането на контролера с една стъпка и обновяване на управляващите битове.
Изход	F (Full)	Установява се в активно ниво, когато е достигната последната дефинирана стъпка. Асоциираният бит %DRi.F може да бъде проверен (например: %DRi.F=1, ако %DRi.S = броя на конфигурираните стъпки - 1).
Управляващи битове		Изходите или вътрешните битове, асоциирани със стъпката (16 управляващи бита) и дефинирани в конфигурационния редактор (Configuration Editor).

Фрагментът от Ladder диаграма на фигура 11.3 показва включването в програма на барабанен контролер, а на фигура 11.4 е показано конфигурирането на този контролер.

Фиг. 11.3. Включване на барабанен контролер

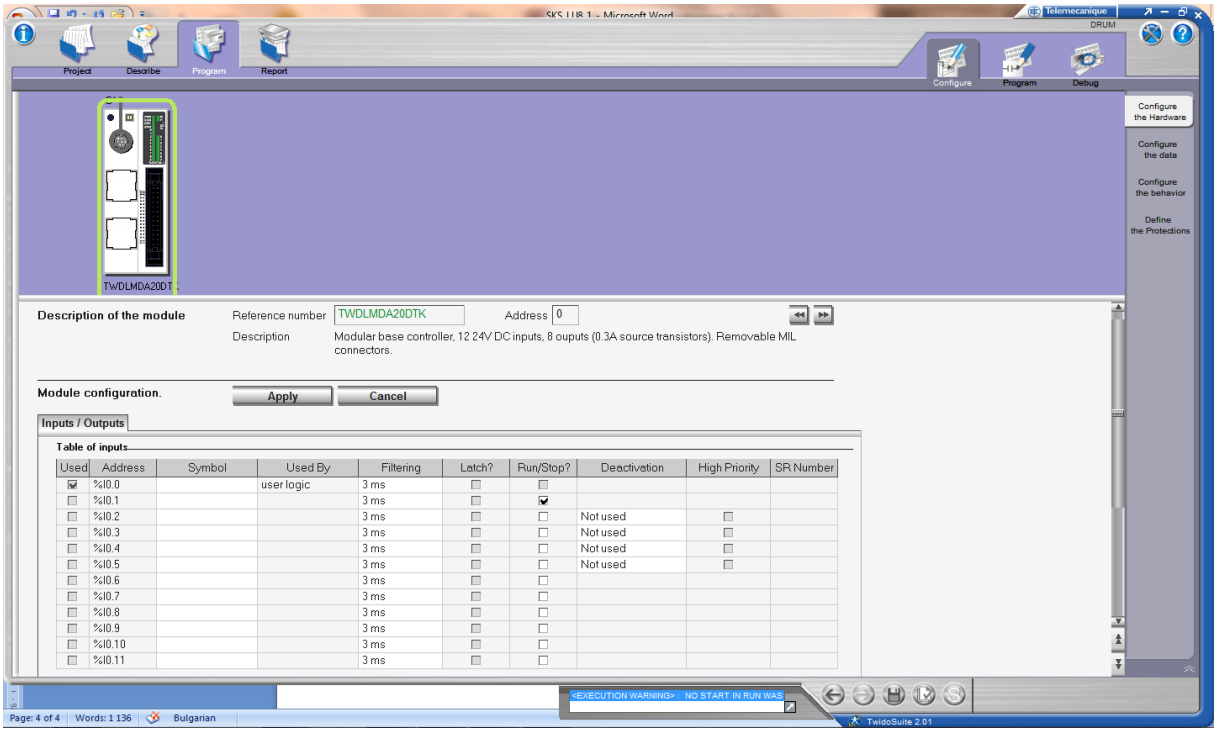


Фиг. 11.4. Конфигуриране на стъпките и асоцииране на битовете на барабанния контролер

При всеки нарастващ фронт на сигнала, постъпващ на входа U на барабанния контролер, той ще се превърта с една стъпка, активирайки последователно изходите на контролера %Q0.0 до %Q0.7, според зададеното в конфигурацията.

3. Вход за старт/стоп

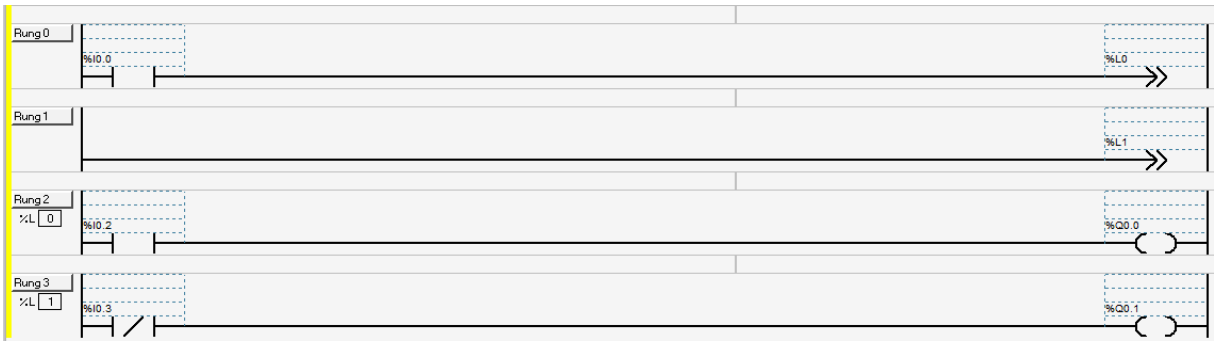
Вероятно всички програми, които сте написали досега, са били сигнализирани с <EXECUTION WARNING> NO START IN RUN WAS SELECTED. Стартирането на такава програма става от програмиращата станция, от бутона . В практиката е по-удобно стартирането и спирането на програмата да става от специализиран бутон, който се свързва към един от цифровите входове на контролера. За целта този вход трябва да се маркира при неговото описание, както е показано на фигура 11.5.



Фиг. 11.5. Конфигуриране на вход за старт/стоп

4. Инструкции за преход

Инструкциите за преход предизвикват последователността на изпълнението на програмата и преминаването към рънга с етикет %Li (i = 1 до 63). Разбира се, преходите могат да бъдат условни или безусловни. На фигура 11.6 в рънг 0 се съдържа инструкция за условен преход. Преходът към етикет %L1 осъществява, когато проверяваното в тестовата част на рънга логическо условие има стойност логическа единица. Възможен е условен преход и когато резултатът от проверката има стойност логическа нула. Преходът, записан в рънг 1, е безусловен.



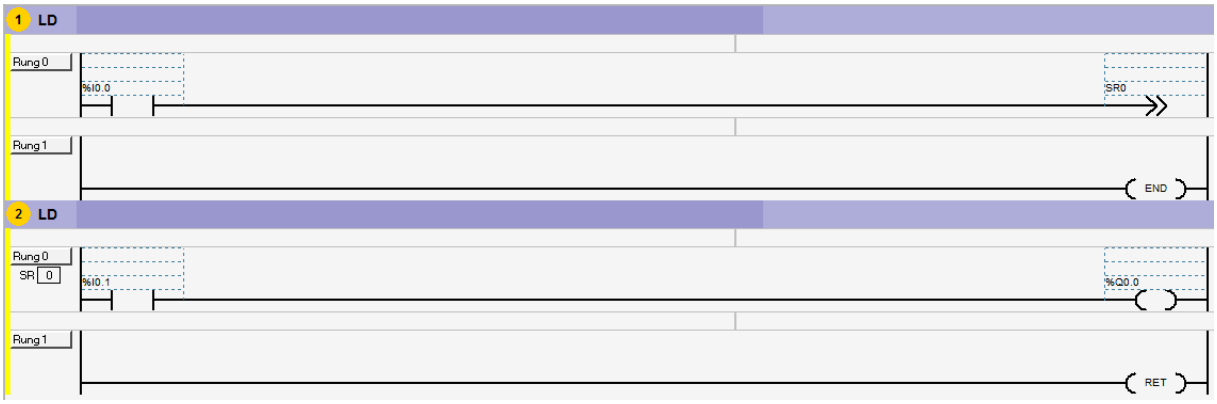
Фиг. 11.6. Условен и безусловен преход

5. Подпрограми

Инструкциите за преход към подпрограма предизвикват преминаване към изпълнението на подпрограмата и след това – връщане към главната програма.

Инструкцията **SRn** извиква подпрограмата, означена с етикет **SRn**, ако резултатът от предшестващата логическа инструкция е единица. Подпрограмите се означават с етикети **SRn**, където **n** може да бъде число от 0 до 63 за разглеждания контролер. Подпрограмата задължително завършва с инструкция за връщане **RET.** Задължително е също главната програма

да завършва с инструкция за край – **END.** На фигура 11.7 е показано извикване на подпрограма и връщане от нея.



Фиг. 11.7. Извикване на подпрограма

Подпрограмите не могат да се влагат, т. е. подпрограма не може да извиква друга подпрограма!

6. Задачи за самостоятелна работа

6.1. Съставете програма за ПЛК от серията TWIDO, която да управлява „светлинно шоу”. Светлинното шоу демонстрирайте върху лабораторния макет. В програмата включете различни режими, всеки от които оформете като подпрогрма. За реализирането на режимите използвайте различни техники и средства, изучени до момента. Например: програмен брояч (с аритметични инструкции), функционален блок брояч, изместващ регистър, барабанен контролер, часовник за реално време (в определен час да се включва съответен режим) и др.

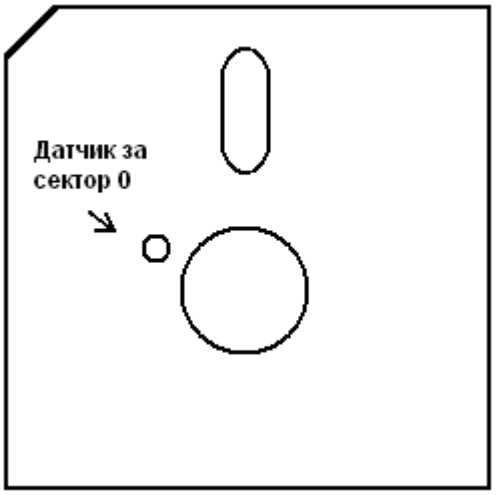
6.2. Съставете програма за ПЛК TWIDO за управление на флопи-дисково устройство. Програмата да извършва следното:

1. Проверка дали главата е позиционирана на нулева пътечка. Това изисква управление на стъпков двигател и следене на цифров вход.
2. Проверка за наличие на дискета. Ако има дискета във флопи-дисковото устройство, се включва основният двигател (шпиндел).
3. Проверка дали дискетата е защипана, но тъй като няма датчик за „защипана дискета”, се проверява за наличие на импулс на цифров вход.
4. Главата да се позиционира на пътечка 10 или 20 в зависимост от натиснат бутон.

Указания:

Управлението на флопи-дисковото устройство чрез ПЛК може да бъде тествано с помощта на лабораторния макет.

Самото флопи-дисково устройство притежава редица датчици, на базата на които ще се извършва управлението му, а управлението ще се реализира, чрез подаването на различни сигнали към изходите на контролера, които физически са свързани с хардуера на самото устройство. Така например изход %Q0.3 е свързан с основния двигател (шпиндел), като при наличие на сигнал на %Q0.3 той се запуска. При наличие на сигнали на изходи от %Q0.4 до %Q0.7 се управлява движението на стъпковия двигател, като съществува точно определена зависимост за посоката на движението на стъпковия мотор. Ако сигнала се „движи” от %Q0.4 към %Q0.7 стъпковият мотор измества електромагнитния четец към по-малките номера на пътечки и обратното - ако сигналът се „движи” от %Q0.7 към %Q0.4 стъпковият мотор измества електромагнитния четец към по-големите номера на пътечки.

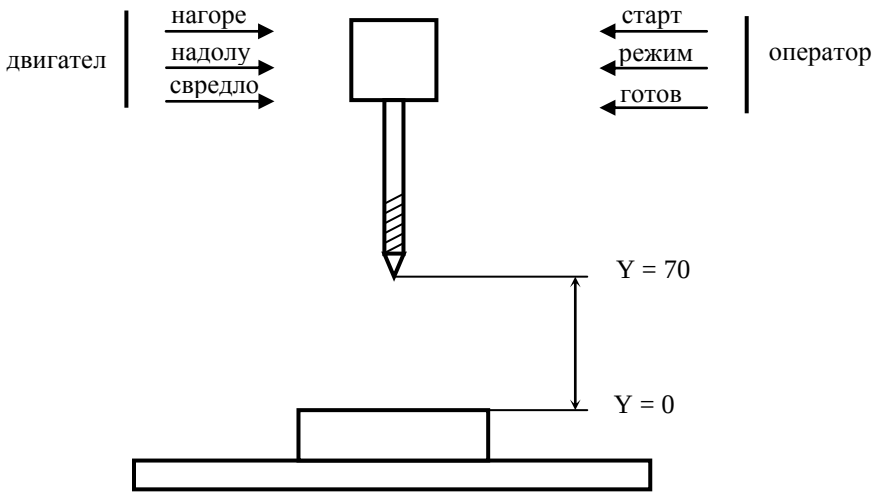


Флопи-дисковото устройство притежава датчик за пътечка ”0”, свързан на лабораторния макет с вход %I0.5 на контролера. Преминаването през пътечка 0 се регистрира със сигнал „0” от датчика. Входът %QI0.6 е свързан с датчика за сектор „0”, а %QI0.7 – с датчика за поставена дискета в устройството.

Задачи за самостоятелна работа

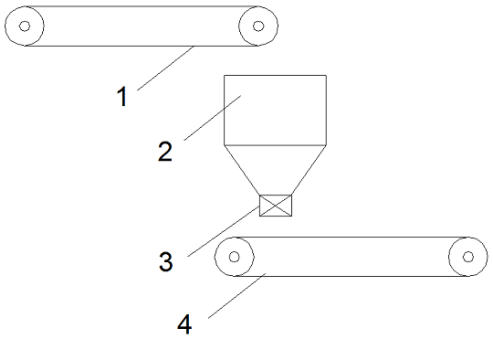
Всички задачи имат едно и също условие: **Да се създаде програмно осигуряване за Twido контролер (TWDLMDA20DTK или DRT), с което да се осигури автоматизирано управление на посочения обект.**

Задача 1. Управление на бормашина



Преди началото на работата операторът, с помощта на бутон, избира режима на работа. След това, с друг бутон, запуска контролера и стартира управлението. Подава се команда за пускане на свредлото и започва обратно броене на координатата. При достигане на най-ниската точка (y = 0) командата за пускане на свредлото отпада и се издава команда за пробиване (за въртене на свредлото). В режим 1 пробиването продължава 5 секунди. В режим 2 въртенето на свредлото се преустановява от оператора (чрез бутон). След приключване на пробиването следва команда за вдигане на свредлото и започва право броене на координатата. При достигане на горно положение (y = 70) командата за вдигане на свредлото се отменя.

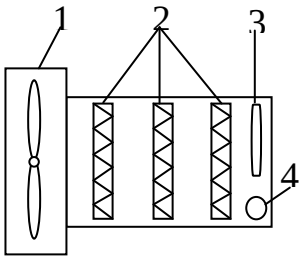
Задача 2. Линия за дозиране на продукти



Продуктът, с помощта на зареждаща транспортна лента 1 попада в бункер 2. Транспортната лента работи, докато теглото на продукта в бункера не стане по-голямо от зададеното предварително. След това транспортна лента 1 спира,

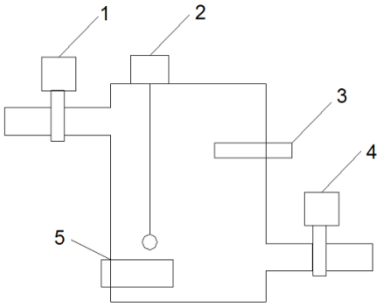
сработва клапан 3 и се включва транспортна лента 4. След изпразването на бункера клапанът се затваря, транспортна лента 4 спира и зареждането започва отново.

Задача 3. „Калорифер”



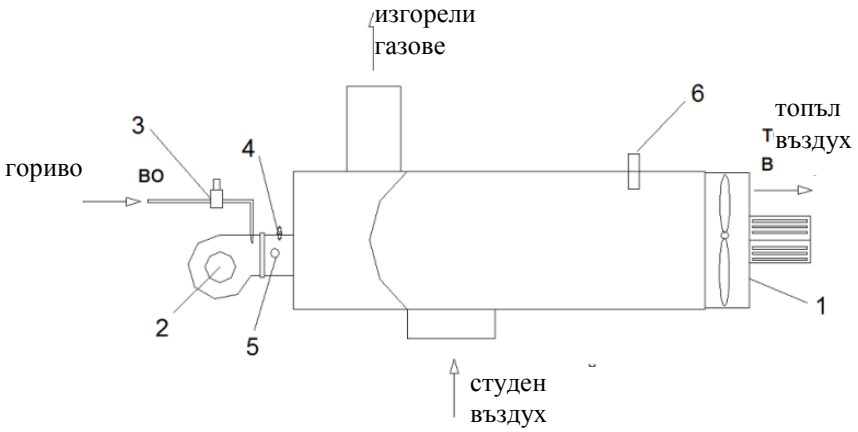
Нагретият въздух се разпръсква от вентилатора 1. В зависимост от зададената температура се включват определен брой нагреватели 2. Нагревателите не трябва да работят при изключен вентилатор. 3 – температурен датчик, 4 – фиксатор (задаване) на температурата.

Задача 4. Водонагревателна система



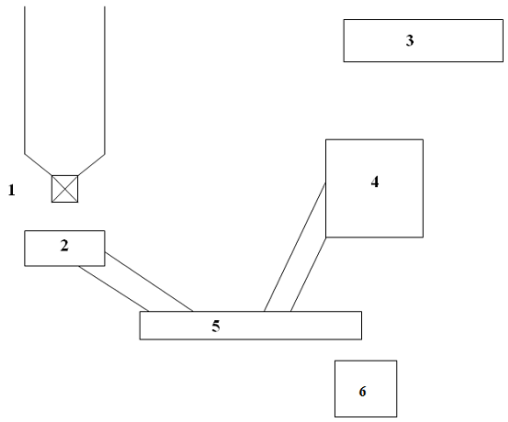
Водата преминава през клапан 1 и запълва вместимостта на резервоара до определено ниво, което се регистрира от датчика за ниво 2. Водата се нагрява от нагревателя 5 до зададена температура, която се измерва от датчика за температура 3, и се източва през изходящия клапан 4.

Задача 5. Топлогенератор



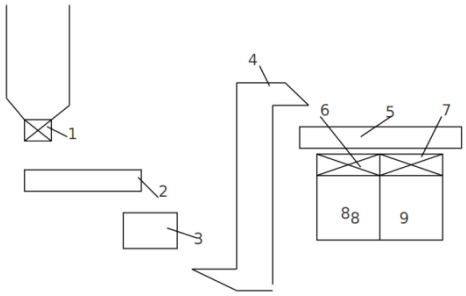
При натискането на бутон за запуск се издава звукова предупредителна сигнализация и се запуска основния вентилатор за топъл въздух 1. След запускането на основния вентилатор се включва топлинния вентилатор 2 за продухване (10 секунди). След това се включва електромагнитният вентил 3 и горивната смес се впръсква в горивната камера (5 секунди). Задейства се запалителната свещ 4 (4 секунди). Релето за пламъка 5 контролира наличието на пламък. Ако пламъкът не се появи в продължение на 5 секунди, процесът на запалване се извършва отново (с продухване на въздуха в продължение на 15 секунди). При неуспешно повторно запускане на агрегата се включва продухване за 1 минута и аварийна сигнализация. При нормално запускане на агрегата системата трябва да контролира температурата на въздуха на изхода на термопреобразувателя 6 и да променя скоростта на въртене на топлинния вентилатор 2. При спиране на агрегата продухването трябва да продължи, докато температура не падне под Tmin.

Задание 6. Технологичен процес 1



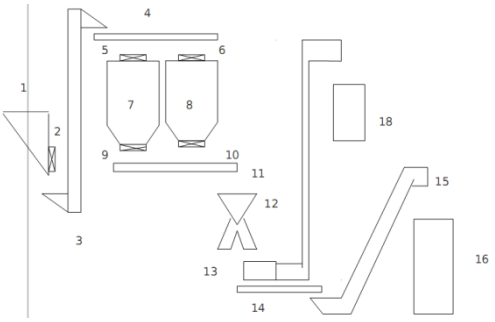
През вентила 1 зърното постъпва в трошачката 2 и след това в конвейер-миксера 5. Там също така постъпват и преработените в миялната-резачка 4 кореноплодни зеленчуци (3 – транспортна лента за нарязаните кореноплодни зеленчуци). Чрез Конвейер-миксера 5 сместа се зарежда в миксера 6. Трябва да се осигури съвместна и самостоятелна работа на линиите за зърно и кореноплодни.

Задача 7. Технологичен процес 2



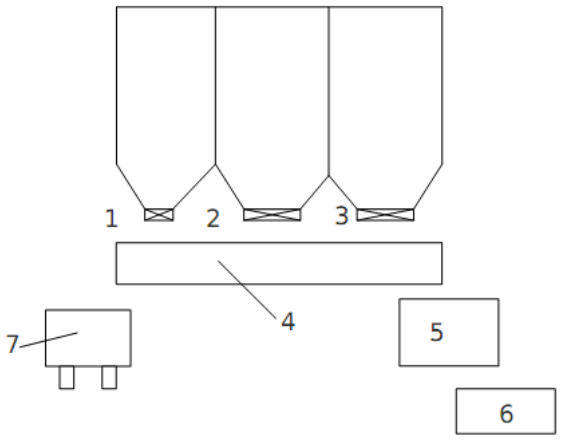
Зърното от бункера през вентила 1 се подава върху конвейера 2 и след това в трошачката 3. Натрошеното зърно чрез кофата-лифт 4 се подава към конвейера 5 и след това или в бункера 8 или в бункера 9. Линията трябва да спре при запълване на един от бункерите. Режимът на работа на електродвигателите на производствената линия е кратък.

Задача 8. Технологичен процес 3



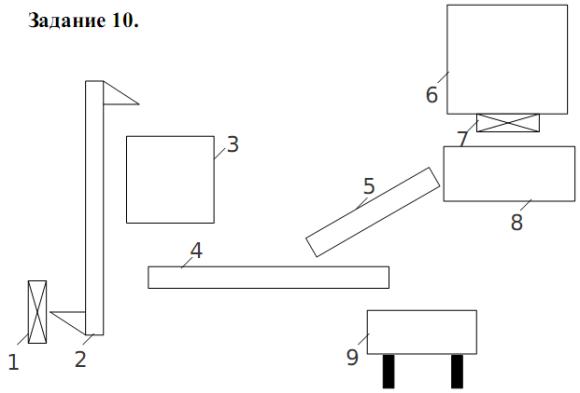
От силоза 1 семената през отвора на вентила 2 постъпват в кофата-лифт 3 към конвейера 4 и след това през клапите 5 и 6 пълнят бункерите 7 и 8. От бункерите 7 и 8 през клапаните 9 и 10 семената попадат върху наклонения транспортьор 11, а оттам – в пресата 12. След пресата маслото от цистерната чрез помпата 13 се подава в резервоара 18. След пресоването кюспето постъпва върху конвейера 14 и кофата-лифт 15, която зарежда бункера 16. Двигателите работят в кратковременен режим.

Задача 9. Технологичен процес 4



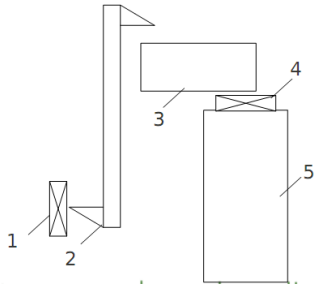
Зърното постъпва на конвейера 4 през един от вентилите 1, 2 или 3, или през всички заедно (по избор на оператора). След това попада или в камиона 7, или в трошачката 5, след което - в бункера 6. Схемата трябва да се изключи при сработване на датчика за ниво в бункер 6 или на датчика за налягане, предизвикано от камиона.

Задача 10. Технологичен процес 5



Производствената линия се състои от една линия за преработка на зърно и друга линия за преработка на кореноплодни. В състава на линията за преработка на зърно влизат клапан 1 на силоза, подемният транспортьор 2 и трошачката 3. Линията за преработка на кореноплоди се състои от бункер за цели кореноплоди 6, клапан на бункера 7, миялна на ножовете 8, транспортьор на настърганите кореноплоди 5. Продуктите и от двете линии постъпват на конвейера-смесител 4 и след това се натоварват в камиона 9. Осигурете отделна и съвместна работа на линиите за преработка на зърно и кореноплодни растения.

Задача 11. Технологичен процес 6



Зърното се подава през клапана 1, чрез подемната транспортна лента 2 към трошачката 3, където се натрошава. Смляното зърно през клапана 4 се зарежда в бункера 5. Осигурете принудително изключване на схемата от работно състояние, както и при сработване на сензорите за ниво. Двигателите в технологичната схема са в режим на краткотрайна работа.

Използвана литература

1. А. Тодоров, Сн. Йорданова, Ст. Джиев, В. Сгурев, „Логическо управление на процеси”, ТУ – София, 2001

2. http://www.automatic-project.eu

3. Twido programmable controllers, Hardware Reference Guide, TWD USE 10AE eng Version 3.2

4. Twido, TwidoSoft Operation Guide, eng Version 3.2

5. Twido programmable controllers, Software Reference Guide, TWD USE 10AE eng Version 3.2