МИННО-ГЕОЛОЖКИ
УНИВЕРСИТЕТ
„СВ. ИВАН РИЛСКИ“

ВОЛОДЯ ДЖАРОВ

КОМПЮТЪРНИ СИСТЕМИ ЗА УПРАВЛЕНИЕ С ОТВОРЕН КОД, ПРИЛОЖИМИ В ПРОМИШЛЕНИТЕ ПРЕДПРИЯТИЯ

СОФИЯ, 2026

МИННО-ГЕОЛОЖКИ УНИВЕРСИТЕТ „Св. ИВАН РИЛСКИ“

Володя Владимиров Джаров

Компютърни системи за управление с отворен код, приложими в промишлените предприятия

Монография

София· 2026

МИННО-ГЕОЛОЖКИ УНИВЕРСИТЕТ „Св. ИВАН РИЛСКИ“

В настоящия монографичен труд е изследвана възможността за изграждане на компютърни системи за управление чрез използване на хардуерни и програмни технологии с отворена архитектура, приложими при автоматизацията на различни технологични процеси. Разгледани са Arduino Uno и Arduino Opta като контролери, работещи с отворена архитектура, и е направен сравнителен анализ между тях. Извършено е изследване на електрозадвижването на лабораторен модел и е разгледано управлението на скоростта на постояннотоковия двигател, както и възможността за реализиране на затворен контур с обратна връзка и PID алгоритъм. Разработен е алгоритмичен подход за функциониране и управление на автоматизирана зонална противопожарна система, основан на последователността „измерване – обработка – оценка – решение – управление“.

Авторът осъзнава, че в настоящия монографичен труд не могат да бъдат обхванати всички аспекти, свързани с приложението на хардуерни и програмни технологии с отворена архитектура при автоматизацията на технологични процеси. Разгледаните примери и проведените изследвания обаче дават възможност да се представят основни подходи за проектиране и изграждане на системи за автоматизирано управление, както и възможностите за практическото им приложение. Наред с научно-приложната си насоченост, монографичният труд може да бъде полезен и в учебния процес при подготовката на студенти в областта на автоматизацията и управлението, като подпомогне изучаването на микроконтролерни системи, програмируеми контролери, електрозадвижвания и системи за автоматично управление.

Автор: гл. ас. д-р Володя Владимиров Джаров
Катедра „Електроенергетика и автоматика“
Минно-геоложки университет „Св. Иван Рилски“
Българска, първо издание
Рецензенти: доц. д-р Мила Илиева Илиева–Обретенова
доц. д-р Николай Лазаров Лаков
Научен редактор: доц. д-р Ромео Фердинандов Александров

ISBN 978-954-353-518-7 (мека подвързия)
ISBN 978-954-353-519-4 (pdf)

© Володя Владимиров Джаров, 2026

Издателска къща „Св. Иван Рилски“ на МГУ „Св. Иван Рилски“
София, 2026

Съдържание

СПИСЪК НА ИЗПОЛЗВАНИТЕ СЪКРАЩЕНИЯ И АКРОНИМИ

- **КСУ** – Компютърна система за управление / Компютърни системи за управление;
- **CAN** – Controller Area Network – комуникационна мрежа за обмен на данни между електронни устройства;
- **CI/CD** – Continuous Integration / Continuous Delivery (Deployment) – непрекъсната интеграция и непрекъсната доставка/внедряване;
- **CPS** – Cyber-Physical Systems – кибер-физични системи;
- **DCS** – Distributed Control System – разпределена система за управление;
- **DI** – Digital Input – цифров вход;
- **DO** – Digital Output – цифров изход;
- **FBD** – Function Block Diagram – език за програмиране чрез функционални блокови диаграми;
- **GNU** – GNU's Not Unix – проект за разработване на свободна операционна система и свободен софтуер;
- **HMI** – Human–Machine Interface – човеко-машинен интерфейс;
- **IDE** – Integrated Development Environment – интегрирана среда за разработка;
- **IEC** – International Electrotechnical Commission – Международна електротехническа комисия;
- **IIoT** – Industrial Internet of Things – Индустриален интернет на нещата;
- **I/O** – Input/Output – вход/изход;
- **IT** – Information Technology – информационни технологии;
- **LD** – Ladder Diagram – стълбовидна диаграма, графичен език за PLC програмиране;
- **MQTT** – Message Queuing Telemetry Transport – лек комуникационен протокол за обмен на съобщения;
- **OPC** – Open Platform Communications – стандарт за обмен на данни в системите за индустриална автоматизация;
- **OPC UA** – Open Platform Communications Unified Architecture – унифицирана архитектура за индустриален обмен на данни;
- **OSH** – Open Source Hardware – хардуер с отворен код;
- **OSHWA** – Open Source Hardware Association – Асоциация за хардуер с отворен код;
- **OSI** – Open Source Initiative – Инициатива за отворен код;

- **OSS** – Open Source Software – софтуер с отворен код;
- **OT** – Operational Technology – оперативни технологии;
- **PCB** – Printed Circuit Board – печатна платка;
- **PID** – Proportional–Integral–Derivative – пропорционално-интегрално-диференциален регулатор;
- **PLC** – Programmable Logic Controller – програмируем логически контролер;
- **PWM** – Pulse Width Modulation – широчинно-импулсна модулация;
- **RAM** – Random Access Memory – оперативна памет с произволен достъп;
- **RS-485** – Recommended Standard 485 – стандарт за серийна комуникация;
- **RTU** – Remote Terminal Unit / при Modbus: режим Remote Terminal Unit – отдалечено терминално устройство/режим за сериен обмен;
- **SCADA** – Supervisory Control and Data Acquisition – система за супервизорно управление и събиране на данни;
- **SFC** – Sequential Function Chart – последователна функционална диаграма;
- **ST** – Structured Text – структуриран текст, език за PLC програмиране;
- **TCP/IP** – Transmission Control Protocol / Internet Protocol – набор от комуникационни протоколи за мрежов обмен;
- **USB** – Universal Serial Bus – универсална серийна шина;

УВОД

През последните десетилетия развитието на информационните технологии, индустриалната автоматизация и комуникационните системи доведе до съществени промени в организацията и управлението на производствените процеси. Концепциите за цифрова трансформация, Industry 4.0, Industrial Internet of Things (IIoT) [1] и кибер-физични системи поставят нови изисквания към компютърните системи за управление, които трябва да бъдат гъвкави, мащабируеми, надеждни и икономически ефективни.

Традиционните индустриални системи за управление, базирани на затворени програмни и хардуерни платформи, предоставят висока функционалност и надеждност, но често са свързани със значителни инвестиционни разходи, ограничени възможности за модификация и зависимост от конкретен производител. Това поражда необходимост от разработване и внедряване на алтернативни решения, базирани на отворени стандарти, свободно достъпен софтуер и възможности за интеграция с различни програмни и хардуерни платформи.

В резултат на тези тенденции все по-голямо значение придобива концепцията за отворен код (Open Source), която променя традиционните модели за разработване и внедряване на софтуерни системи, включително в областта на индустриалната автоматизация. Отворените програмни платформи предоставят нови възможности за изграждане на гъвкави, мащабируеми и икономически ефективни компютърни системи за управление, адаптирани към специфичните изисквания на промишлените предприятия.

Отвореният код (Open Source) представлява модел за разработка и разпространение на софтуер, при който изходният код е публично достъпен и може да бъде използван, изучаван, модифициран и разпространяван съгласно условията на съответния лиценз. Този модел се основава на принципите на прозрачност, сътрудничество, децентрализация и общностно участие в процеса на разработка.

Според дефиницията на Open Source Initiative, софтуерът с отворен код не се определя единствено чрез достъп до изходния код, а чрез съвкупност от права, които гарантират свободно използване, модификация и разпространение при спазване на определени лицензионни условия [2], [3].

Софтуерът с отворен код (Open Source Software – OSS) се характеризира с децентрализиран модел на разработка, при който множество разработчици, изследователи и организации участват съвместно в неговото създаване и усъвършенстване. Този подход позволява прилагането на

механизми за партньорска проверка (peer review), което води до повишаване на надеждността, сигурността и качеството на софтуерните системи [4].

Концепцията за отворен код е исторически свързана с развитието на движението за свободен софтуер и с принципите за свободно използване, изучаване, модифициране и разпространение на програмните продукти [5].

Софтуерът със затворен код (Proprietary Software) представлява алтернативен модел, при който изходният код не е публично достъпен, а правата за неговото използване, модификация и разпространение са ограничени от собственика. В този случай потребителят получава право за използване на софтуера при условията на съответния лиценз, без свободен достъп до изходния код и възможност за неговото независимо модифициране [2].

В съвременната индустриална практика софтуерът с отворен код играе ключова роля в развитието на гъвкави, мащабируеми и икономически ефективни системи за автоматизация. Неговото приложение е особено значимо в контекста на индустриалната дигитализация и конвергенцията между информационните технологии (IT) и оперативните технологии (OT). Open Source решенията се превръщат във важен фактор за реализиране на концепциите за Industry 4.0 и Industrial Internet of Things, като осигуряват отворени интерфейси, гъвкава интеграция и намалена зависимост от конкретни производители.

Развитието на Open Source технологиите е пряко свързано с еволюцията на компютърните системи за управление. Съвременните индустриални системи все по-често използват разпределени архитектури, отворени комуникационни стандарти и модулни софтуерни компоненти, което ги прави съвместими с Open Source екосистеми.

В този контекст настоящата монография разглежда съвременните компютърни системи за управление като интегрирани системи, включващи хардуерни, софтуерни и комуникационни компоненти, функциониращи в реално време за наблюдение, управление и оптимизация на технологични процеси в промишлените предприятия.

Основната цел на настоящата монография е да се изследват възможностите за изграждане и практическо приложение на компютърни системи за управление, базирани на хардуерни и програмни технологии с отворен код, и да се оцени тяхната приложимост при решаването на задачи за автоматизация на технологични процеси и системи в промишлените предприятия.

За постигане на поставената цел са формулирани следните основни изследователски задачи:

1. Да се систематизират и анализират теоретичните основи, архитектурите и принципите на функциониране на съвременните компютърни системи за управление.
2. Да се изследват принципите на софтуера и хардуера с отворен код и да се анализират възможностите за тяхното приложение при изграждането на компютърни системи за управление.
3. Да се анализират и сравнят хардуерни и програмни платформи, приложими при разработването на системи за автоматизация, с акцент върху микроконтролерни и индустриално ориентирани решения като Arduino Uno и Arduino Opta.
4. Да се разработят и експериментално изследват компютърни системи за управление чрез лабораторни модели, включващи измерване на технологични параметри, обработка на информация, управление на изпълнителни устройства и реализация на алгоритми за автоматично регулиране.
5. Да се изследват възможностите за реализация на алгоритми за управление и регулиране, включително PID управление, и да се оцени влиянието им върху динамичните характеристики на изследваните системи.
6. Да се разработят архитектурни и алгоритмични решения за приложение на компютърното управление в автоматизирани системи с повишени изисквания към мониторинга и безопасността.
7. Да се извърши сравнителна оценка на изследваните хардуерни и програмни решения и да се формулират насоки за тяхното приложение и развитие в съвременните компютърни системи за управление.

За реализиране на поставената цел и изпълнение на изследователските задачи в настоящата монография е използван комплекс от теоретични, експериментални и приложни методи на научно изследване.

Теоретичните методи включват анализ и обобщаване на научна литература, нормативни документи, международни стандарти и съвременни публикации, свързани с компютърните системи за управление, софтуера с отворен код, индустриалната автоматизация, Industry 4.0 и Industrial Internet of Things. Използвани са методи за сравнителен анализ на архитектури, програмни платформи, комуникационни протоколи и технологии, използвани при изграждането на системи за управление.

При разработването на компютърните системи за управление с отворен код е приложен системен подход, основан на принципите на модулното проектиране, разпределените архитектури и използването на отворени комуникационни стандарти. Изследвани са възможностите за интеграция между хардуерни и софтуерни компоненти, изградени на базата на Open Source технологии.

Експерименталните изследвания са проведени чрез разработване и реализиране на лабораторни и експериментални постановки, включващи микроконтролерни и програмируеми управляващи платформи, сензорни и изпълнителни устройства и средства за визуализация и мониторинг. Изследвани са функционалните характеристики на разработените системи, поведението им при различни режими на работа и възможностите за реализация на алгоритми за управление и автоматично регулиране.

За оценка на разработените решения са използвани експериментално получени характеристики и сравнителен анализ на основни технически и функционални показатели. При изследването на системите за автоматично регулиране са анализирани преходните процеси и основните показатели, характеризиращи качеството на управление.

При разработването и тестването на експерименталните системи са използвани Open Source програмни средства, среди за разработка и инструменти за визуализация, което позволява оценяване на възможностите за практическото им приложение в условията на съвременните промишлени предприятия.

Експерименталните изследвания са реализирани чрез разработване на компютърни системи за управление с отворен код, базирани на Arduino Uno и Arduino Opta, с използване на индустриални комуникационни протоколи Modbus TCP и MQTT, както и Open Source платформи за визуализация и обмен на данни.

Приема се хипотезата, че чрез интегрираното използване на хардуерни и програмни технологии с отворен код, съвременни програмируеми управляващи платформи и стандартни комуникационни средства могат да бъдат изграждани функционални, гъвкави и разширяеми компютърни системи за управление, приложими при решаването на различни задачи за автоматизация.

Предполага се, че изборът и конфигурирането на управляващата платформа в съответствие с конкретните изисквания на технологичния обект позволява използването на отворени технологии както при лабораторни и

експериментални системи, така и като основа за разработване на индустриално ориентирани решения. Сравнителното изследване на микроконтролерни и програмируеми управляващи платформи, както и практическата реализация на алгоритми за управление, позволява да бъдат определени техните възможности, ограничения и области на целесъобразно приложение.

В частност се приема, че индустриалните контролери с отворена архитектура, какъвто е Arduino Opta, в съчетание с Open Source програмни средства за визуализация, обмен на данни и мониторинг, могат да бъдат основа за изграждане на съвременни компютърни системи за управление, приложими в реални промишлени предприятия.

Научната новост на настоящата монография е свързана със систематизирането, развитието и практическото прилагане на подходи за изграждане на компютърни системи за управление, базирани на хардуерни и програмни технологии с отворен код. Изследването обединява теоретичен анализ, разработване на архитектурни и алгоритмични решения и тяхното приложение при конкретни задачи за автоматизация.

Основните елементи на научната новост могат да бъдат формулирани по следния начин:

1. Систематизирани и анализирани са архитектурни и технологични подходи за използване на хардуерни и програмни средства с отворен код при изграждането на компютърни системи за управление.
2. Предложен е модулен подход за изграждане на компютърни системи за управление, позволяващ интегриране на различни управляващи платформи, сензорни и изпълнителни устройства и комуникационни средства в зависимост от особеностите на управлявания обект.
3. Разработени и изследвани са алгоритмични и програмни решения за управление, реализирани чрез микроконтролерни и индустриално ориентирани програмируеми платформи, като е извършена сравнителна оценка на възможностите им при конкретни задачи за автоматизация.
4. Разработените принципи са приложени при различни по характер обекти и системи – управление на лабораторен модел на гумено-транспортна лента, автоматично регулиране на електрозадвижване и автоматизирано зонално управление на противопожарна система, с което е изследвана възможността за адаптиране на предложените подходи към задачи с различни функционални изисквания.

Практическата значимост на настоящата монография се определя от възможността разработените подходи, архитектури и програмно-технически решения да бъдат използвани при изграждането и внедряването на компютърни системи за управление с отворен код в различни промишлени предприятия.

Предложените решения позволяват изграждането на гъвкави, мащабируеми и икономически ефективни системи за управление, които могат да бъдат адаптирани към специфичните особености на производствени процеси от различни области на индустрията. Използването на Open Source технологии, отворени комуникационни стандарти и програмируеми контролери с отворена архитектура създава условия за намаляване на инвестиционните разходи, ограничаване на зависимостта от конкретни производители и улесняване на интеграцията със съществуващи системи за автоматизация.

Практическата приложимост на разработените решения е потвърдена чрез реализиране на експериментални системи за управление, базирани на Open Source технологии и индустриални програмируеми контролери, както и чрез провеждане на експериментални изследвания на техните функционални и комуникационни характеристики.

Получените резултати могат да бъдат използвани както при проектиране и внедряване на системи за управление в промишлени предприятия, така и в учебния процес при подготовката на студенти и специализанти в областта на автоматизацията, компютърните системи за управление и индустриалните информационни технологии.

Разработените архитектурни и програмно-технически решения създават предпоставки за бъдещо развитие на Open Source базирани индустриални системи, съвместими с концепциите на Industry 4.0, Industrial Internet of Things и интелигентното производство.

Практическата значимост на изследването се изразява и във възможността разработените решения да бъдат използвани като основа за изграждане на достъпни и гъвкави системи за управление, ориентирани към нуждите на малки и средни промишлени предприятия, както и за разработване на лабораторни и учебно-изследователски комплекси.

Глава 1. Теоретични основи на компютърните системи за управление

1.1. Основни понятия за компютърните системи за управление

Теоретичните основи на компютърните системи за управление (КСУ) се базират на принципите и методите за събиране, обработка, съхранение, предаване и използване на информация с цел управление на технически, технологични и организационни обекти. Значителна част от съвременните системи за автоматизация се реализират чрез компютърно базирани средства за управление. Компютърните системи за управление интегрират достиженията на теорията на управлението, кибернетиката, информатиката, компютърната техника и комуникационните технологии, като осигуряват възможност за автоматизирано наблюдение, анализ и управление на процесите в реално време. Чрез прилагането на алгоритми за обработка на данни и вземане на решения тези системи създават условия за повишаване на ефективността, надеждността и безопасността на управляваните обекти, както и за реализиране на автономни и интелигентни режими на работа.

В основата на всяка система за управление стои процесът на обмен и обработка на информация между управляващата и управляваната подсистема. Информацията за текущото състояние на обекта се получава чрез измервателни средства и сензори, обработва се от управляващото устройство и въз основа на зададени алгоритми се формират управляващи въздействия, насочени към постигане на предварително определени цели и показатели за функциониране.

Компютърните системи за управление реализират тези функции чрез съвкупност от хардуерни, софтуерни и комуникационни компоненти, организирани в единна структура, която осигурява надеждна работа в реално време. В зависимост от предназначението, сложността и областта на приложение, тези системи могат да бъдат централизирани, децентрализирани или разпределени, като използват различни методи за обработка на информацията и формиране на управляващи въздействия.

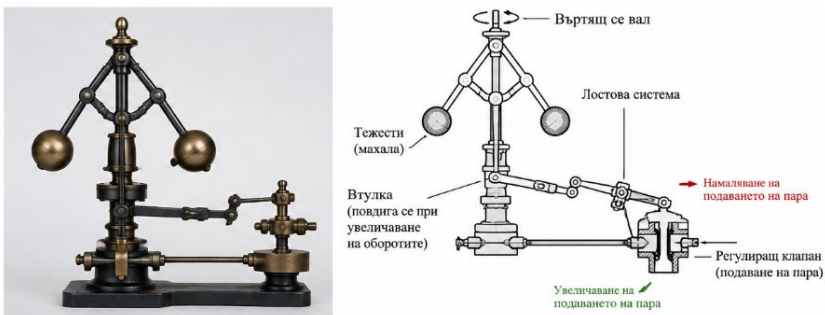
Съвременната структура и функционалните възможности на компютърните системи за управление са резултат от продължително технологично развитие, преминаващо през последователни етапи на механизация, автоматизация, компютъризация и цифровизация на процесите за управление.

1.2. Историческо развитие на системите за управление

Развитието на компютърните системи за управление е тясно свързано с развитието на средствата за автоматизация, изчислителната техника и комуникационните технологии. В исторически аспект могат да бъдат разграничени няколко основни етапа в развитието на системите за управление.

Първият етап е свързан с използването на механични устройства за управление, при които регулирането на технологичните процеси се осъществява чрез механични регулатори и изпълнителни механизми. Характерен

пример за този период е центробежният регулатор (Фиг. 1), използван при парните машини, който представлява един от класическите ранни примери за автоматично регулиране с механична обратна връзка.



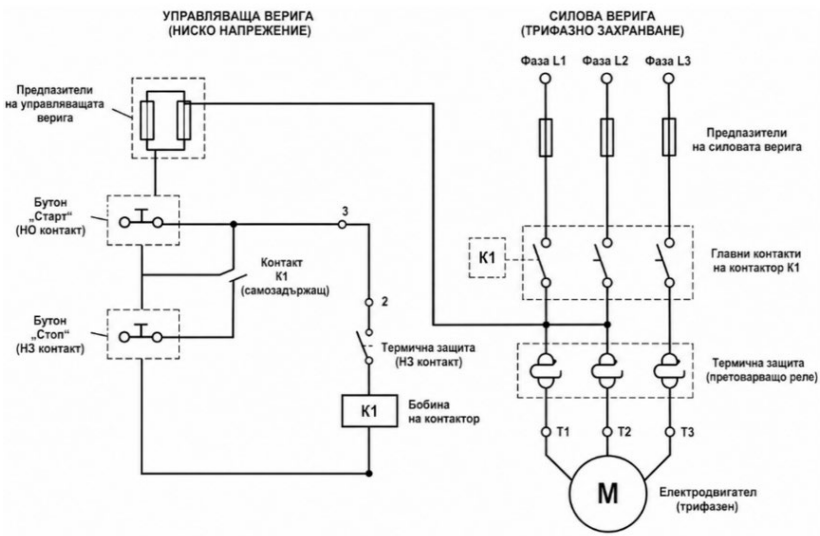
Фиг. 1. Центробежен регулатор на Джеймс Уат (Watt Governor) [6]

Принципът на действие на регулатора се основава на центробежната сила, която действа върху двете сферични тежести (махала), монтирани към въртящ се вал. При увеличаване на скоростта на въртене тежестите се отдалечават от оста на въртене, в резултат на което чрез лостова система се повдига втулката и се намалява подаването на пара към парната машина. При намаляване на скоростта тежестите се приближават към оста, втулката се премества надолу и регулиращият клапан се отваря повече, като увеличава подаването на пара.

По този начин се реализира автоматично поддържане на приблизително постоянна скорост на въртене независимо от изменението на натоварването. Принципът на действие на центробежния регулатор илюстрира една от основните концепции на автоматичното регулиране – използването на обратна връзка, при която изменението на управляваната величина предизвиква коригиращо въздействие върху управлявания процес. Центробежният регулатор на Джеймс Уат представлява една от първите практически реализации на автоматична система с отрицателна обратна връзка и поставя основите на съвременната теория на автоматичното управление.

Вторият етап в развитието на системите за управление е свързан с появата и широкото приложение на електромеханичните системи за управление, базирани на релета, контактори, електромагнитни усилватели и други електромеханични устройства (Фиг. 2). Тези системи започват активно да се използват през първата половина на XX век и поставят основите на индустриалната автоматизация. За разлика от механичните регулатори, електромеханичните системи осигуряват по-висока надеждност, по-голяма гъвкавост при реализиране на логически функции и възможност за управление на по-сложни технологични процеси. В този период се разработват първите релейно-контакторни схеми за автоматично управление на електродвигатели,

производствени линии и технологични агрегати, които намират широко приложение в машиностроенето, металургията, енергетиката и минната промишленост.

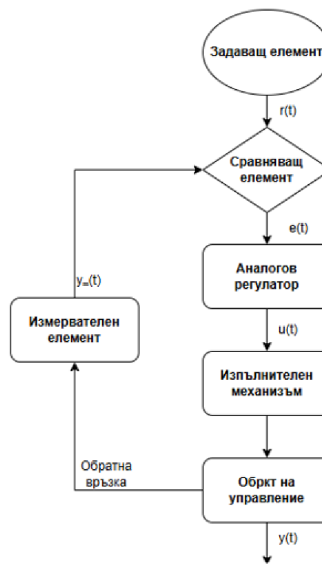


Фиг. 2. Електромеханична система за управление на трифазен електродвигател с реле, контактор и термична защита

На фиг. 2 е представена типична електромеханична система за управление на трифазен електродвигател. Управлението се осъществява посредством бутони „Старт“ и „Стоп“, електромагнитен контактор и термична защита. При натискане на бутона „Старт“ се активира контакторът, който включва електродвигателя към трифазната мрежа. Бутонът „Стоп“ прекъсва управляващата верига и изключва двигателя. Термичната защита осигурява защита от претоварване и прегряване. Подобни системи намират широко приложение в индустриалната автоматизация през първата половина на XX век и поставят основите на съвременните системи за автоматично управление. Развитието на електрониката и изчислителната техника създава предпоставки за постепенно преминаване от твърдо зададени и аналогови схеми за управление към цифрови и програмируеми системи, при които алгоритъмът за управление може да бъде реализиран и изменян чрез програмни средства.

Третият етап в развитието на системите за управление е свързан с навлизането на електронните и аналоговите системи за управление. Използването на електронни усилватели, аналогови регулатори и специализирани електронни схеми позволява значително повишаване на точността,

бързодействието и надеждността на управляващите системи. През този период започват да се използват операционни усилватели, аналогови изчислителни устройства и специализирани електронни регулатори, чрез които се реализират по-сложни алгоритми за управление и регулиране на технологичните процеси (Фиг. 3). Аналоговите системи намират широко приложение в промишлената автоматизация, управлението на електрозадвижвания, химическите производства, енергетиката и други индустриални области. Въпреки високото си бързодействие, тези системи имат ограничена гъвкавост при промяна на алгоритмите за управление, което впоследствие води до развитието и широкото внедряване на цифровите и микропроцесорните системи за управление.



Фиг. 3. Структурна схема на аналогова система за автоматично управление с отрицателна обратна връзка

където:

- r(t) – задаващо въздействие.
- $e(t)$– – сигнал на грешката (изход на сравняващия елемент);
- $u(t)$– управляващо въздействие;
- $y(t)$– изходна величина на обекта на управление;
- $y_m(t)$– измерена стойност (сигнал на обратната връзка).

На фиг. 3 е представена структурна схема на аналогова система за автоматично управление с отрицателна обратна връзка. Основната цел на

системата е да осигури поддържане на зададената стойност на управляваната величина независимо от външните въздействия и измененията в параметрите на обекта на управление.

Задаващият елемент формира референтното въздействие $r(t)$, което се сравнява с измерената стойност $y_m(t)$, постъпваща чрез канала на обратната връзка. Разликата между тези две величини представлява сигнал на грешката $e(t)$, който се подава към аналоговия регулатор. Въз основа на този сигнал регулаторът формира управляващо въздействие $u(t)$, което чрез изпълнителния механизъм въздейства върху обекта на управление. Изходната величина $y(t)$се измерва непрекъснато и отново се подава към сравняващия елемент, с което се реализира отрицателна обратна връзка.

Използването на отрицателна обратна връзка позволява намаляване на влиянието на външните смущения, повишаване на точността на регулиране и подобряване на устойчивостта на системата. Именно този принцип, възникнал още при механичните регулатори, намира широко приложение при аналоговите, а впоследствие и при цифровите компютърни системи за управление.

Въпреки своите предимства, аналоговите системи имат ограничена гъвкавост при промяна на алгоритмите за управление, което стимулира развитието на цифровите системи, базирани на микропроцесори и програмируеми логически контролери (PLC). Това поставя началото на следващия етап в развитието на компютърните системи за управление.

Четвъртият етап в развитието на системите за управление започва с развитието на микропроцесорната техника и появата на програмируемите логически контролери (Programmable Logic Controllers – PLC). Използването на цифрова обработка на информацията, програмируеми алгоритми и микропроцесорни устройства създава предпоставки за изграждането на гъвкави, надеждни и лесно адаптируеми системи за управление, намиращи широко приложение в различни отрасли на индустрията.

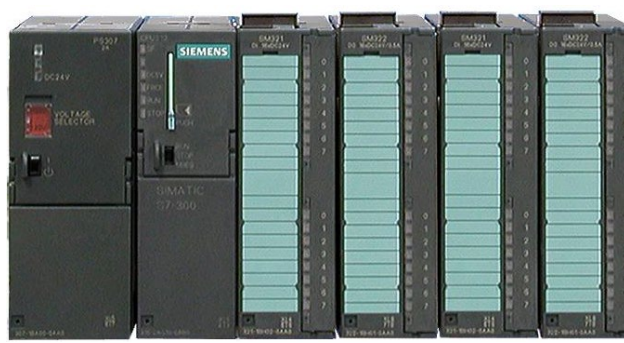
Програмируемите логически контролери (PLC) представляват специализирани индустриални компютърни системи, предназначени за управление на дискретни и непрекъснати технологични процеси. Те изпълняват предварително зададени алгоритми за управление в зависимост от настъпили събития, логически условия или зададени времеви интервали. Благодарение на своята надеждност, гъвкавост и устойчивост на индустриални условия, PLC системите намират широко приложение в различни отрасли на промишлеността.

Първите програмируеми логически контролери се появяват в края на 60-те и началото на 70-те години на XX век като средство за замяна на сложните релейно-контакторни схеми за управление. Те позволяват реализирането на логически, времеви, броящи и впоследствие по-сложни управляващи функции чрез програмни средства, без необходимост от физическо преправяне на електрическите схеми при промяна на алгоритъма за

управление. Това значително улеснява проектирането, експлоатацията и модернизацията на системите за автоматизация и създава предпоставки за широкото навлизане на PLC в индустриалното управление.

Развитието на микроелектрониката през последните десетилетия води до значително подобряване на техническите характеристики на PLC системите, включително повишаване на изчислителната мощност, намаляване на времето за сканиране, увеличаване на броя входно-изходни канали и разширяване на комуникационните възможности. Съвременните PLC поддържат различни индустриални комуникационни протоколи, като Modbus, CAN, Fieldbus, Ethernet и други, както и специализирани интерфейси за свързване на термодвойки, тензодатчици, високоскоростни входове и интелигентни входно-изходни модули.

Впоследствие PLC контролерите се превръщат в основен елемент на индустриалната автоматизация. Те намират широко приложение при управлението на производствени линии, електрозадвижвания, транспортни системи, енергийни съоръжения, минно оборудване и различни технологични процеси. Сред най-известните производители на програмируеми логически контролери са Siemens, Allen-Bradley, Omron, Schneider Electric, Mitsubishi Electric и други (Фиг.4).



Фиг. 4. Програмируем логически контролер SIMATIC S7-300

В България също са разработвани и внедрявани програмируеми контролери, сред които могат да бъдат посочени системите „Изоматик“, „Програма 700“, „Програма 1024“ и „Прокон“, използвани в различни области на промишлената автоматизация.

Развитието на PLC технологиите, комуникационните мрежи и компютърната техника поставя основите на съвременните компютърни системи за управление, които постепенно преминават към разпределени архитектури, отворени комуникационни стандарти и програмни платформи с отворен код.

Програмируемите логически контролери (PLC) работят под управлението на операционна система за реално време, която осигурява цикличното изпълнение на управляващата програма. Всяко изпълнение на програмата преминава през предварително определена последователност от операции, наречена оперативен цикъл (Operating Cycle или Scan Cycle). По време на този цикъл контролерът последователно извършва прочитане на входните сигнали, обработка на логическата програма, формиране на изходните сигнали, обмен на информация с външни устройства и изпълнение на системни функции.

Продължителността на оперативния цикъл е един от основните показатели за производителността на PLC, тъй като определя времето за реакция на системата към измененията в управлявания технологичен процес. Структурата на оперативния цикъл на програмируем логически контролер е представена на фиг. 5.

Фиг. 5. Оперативен цикъл на програмируем логически контролер (PLC)

Оперативният цикъл на PLC включва пет основни фази: сканиране на входовете, сканиране на програмата, сканиране на изходите, обслужване на комуникацията и изпълнение на системните функции. Тези дейности се изпълняват циклично под управлението на операционна система за реално време, което осигурява своевременна реакция на контролера спрямо измененията в управлявания процес.

1. Сканиране на входовете (Input Scan)
   Контролерът прочита състоянието на всички входове и записва стойностите им в оперативната памет, като създава образ на входните сигнали (Process Image Inputs).

2. Сканиране на програмата (Program Scan)
   Изпълнява се логическата програма, като се обработват входните сигнали и се формират изходните реакции.
3. Сканиране на изходите (Output Scan)
   Съдържанието на изходните битове се прехвърля към физическите изходи на контролера и се формира образът на изходните сигнали (Process Image Output).
4. Обслужване на комуникацията (Service Communication)
   Осъществява се обмен на информация с операторски панели, компютри, други PLC и индустриални комуникационни мрежи.
5. Системни функции (Housekeeping and Overhead)
   Извършват се операции по управление на паметта, диагностика, обработка на таймери, броячи и други вградени функции.

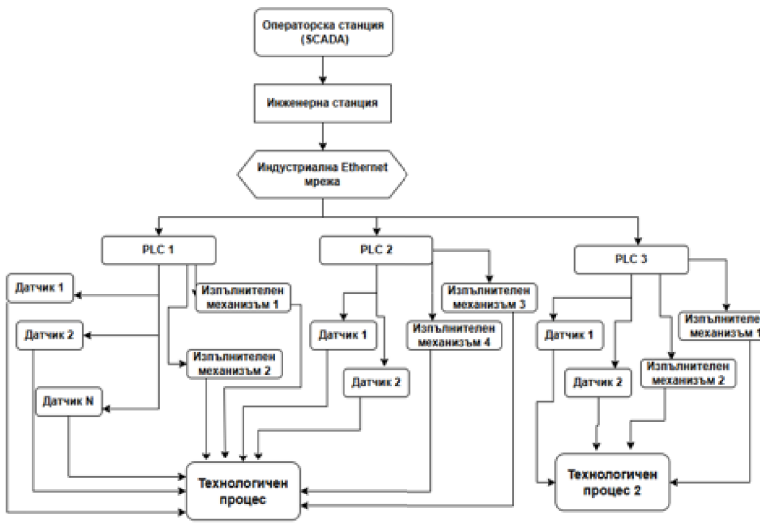
С развитието на промишлената автоматизация и усложняването на технологичните процеси локалното управление чрез отделни контролери постепенно се допълва от разпределени архитектури, при които функциите по измерване, управление, визуализация и съхранение на информация се реализират на различни йерархични нива. Това развитие води до широкото приложение на разпределените системи за управление (DCS) и системите за супервизорно управление и събиране на данни (SCADA).

Петият етап в развитието на системите за управление се характеризира с развитието на разпределените системи за управление (Distributed Control Systems – DCS), системите за супервизорно управление и събиране на данни (Supervisory Control and Data Acquisition – SCADA), както и с широкото използване на индустриални комуникационни мрежи и стандартизирани протоколи за обмен на информация. През този период се реализира преходът от локално управление към разпределени архитектури, при които множество контролери, измервателни устройства и операторски станции обменят информация в реално време чрез комуникационни мрежи.

Системите DCS намират широко приложение при управлението на сложни технологични процеси в химическата, енергийната, нефтопреработвателната и металургичната промишленост. Те осигуряват висока надеждност, модулност и възможност за разпределяне на управляващите функции между множество локални контролери. От своя страна SCADA системите позволяват централизирано наблюдение, визуализация, архивиране и анализ на технологичните параметри, както и дистанционно управление на производствените процеси (Фиг.6).

Развитието на индустриалните комуникационни технологии води до широкото внедряване на стандартизирани протоколи като Modbus, Profibus, Profinet, CAN, Ethernet/IP, OPC и OPC UA, които осигуряват съвместимост между устройства от различни производители и създават предпоставки за изграждане на интегрирани системи за управление.

Именно този етап поставя основите на съвременните кибер-физични системи, Industrial Internet of Things (IIoT) и компютърните системи за управление с отворен код, които се характеризират с висока степен на свързаност, разпределена обработка на данни и използване на отворени програмни и комуникационни стандарти.



Фиг. 6. Структура на разпределена система за управление (DCS) със SCADA система и индустриална Ethernet мрежа

Представената на фиг. 6 структура илюстрира взаимодействието между основните нива на една съвременна разпределена система за управление. На полево ниво се разполагат измервателните и изпълнителните устройства, свързани с локални управляващи контролери, които реализират основните алгоритми за автоматично управление. Чрез индустриална комуникационна мрежа информацията се предава към по-високите нива на системата, където SCADA осигурява централизирано наблюдение, визуализация, регистриране на технологични параметри и операторско взаимодействие.

Интегрирането на PLC, DCS и SCADA системите с индустриални комуникационни мрежи представлява важен етап в развитието на компютърното управление и създава технологичната основа за последващото развитие на мрежово свързаните, кибер-физичните и IIoT базираните системи за автоматизация.

Използването на подобни архитектури създава предпоставки за изграждането на съвременни компютърни системи за управление, характеризиращи се с висока степен на свързаност, мащабируемост и оперативна

съвместимост. Тези характеристики поставят основата за развитието на Industrial Internet of Things (IIoT), кибер-физичните системи и компютърните системи за управление с отворен код.

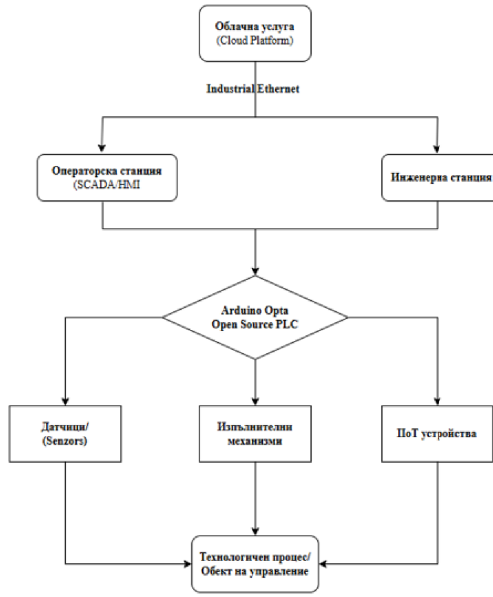
Съвременният етап в развитието на компютърните системи за управление е свързан с концепциите за Industry 4.0, Industrial Internet of Things (IIoT), кибер-физични системи (Cyber-Physical Systems – CPS) и широкото използване на технологии с отворен код (Open Source Technologies). Тези концепции поставят нови изисквания към архитектурата, функционалността, сигурността и комуникационните възможности на системите за управление, като същевременно разширяват областите на тяхното приложение в съвременните промишлени предприятия.

Концепцията Industry 4.0, известна още като четвърта индустриална революция, се основава на интеграцията на физическите производствени системи с цифрови технологии, комуникационни мрежи и интелигентни алгоритми за обработка на данни. Основна характеристика на тази концепция е създаването на интелигентни производства, в които машините, сензорите, изпълнителните механизми и компютърните системи обменят информация в реално време и могат самостоятелно да вземат решения в рамките на определени алгоритми и ограничения.

Важна роля в този процес играе Industrial Internet of Things (IIoT), който осигурява свързаност между индустриалните устройства чрез стандартизирани комуникационни протоколи и мрежови технологии. Благодарение на IIoT се реализира непрекъснат обмен на информация между сензори, контролери, облачни платформи и операторски станции, което позволява дистанционно наблюдение, диагностика, прогнозна поддръжка и оптимизация на производствените процеси.

Кибер-физичните системи представляват интеграция на физически обекти, вградени изчислителни устройства и комуникационни мрежи, функциониращи като единна взаимосвързана система. Те намират широко приложение в роботиката, интелигентните производствени линии, автономните транспортни системи, енергетиката и съвременните системи за автоматизация и управление.

Особено значение през последните години придобиват технологиите с отворен код, които предоставят възможности за разработване на гъвкави, мащабируеми и икономически ефективни системи за управление. Използването на Open Source хардуерни и софтуерни платформи улеснява интеграцията с различни комуникационни протоколи, разработването на собствени алгоритми за управление и ограничаването на зависимостта от конкретни производители. Възможностите за съчетаване на PLC функционалност, индустриални комуникации и Open Source базирани средства за разработка могат да бъдат илюстрирани чрез архитектура, базирана на програмируемия контролер Arduino Opta (фиг. 7).



Фиг. 7. Архитектура на компютърна система за управление с отворен код, базирана на Arduino Opta, в среда Industry 4.0 и Industrial Internet of Things (IIoT)

На фиг. 7 е представена обобщена архитектура на съвременна компютърна система за управление, базирана на програмируемия контролер Arduino Opta. Представената архитектура предвижда интегриране на PLC функционалност, IIoT технологии, облачни услуги и SCADA/HMI приложения, като осигурява възможност за двупосочен обмен на данни между технологичния процес, интелигентните устройства и операторските станции. Използването на отворени програмни средства и стандартни комуникационни технологии създава предпоставки за изграждане на гъвкави и разширяеми системи за управление, съответстващи на тенденциите на Industry 4.0.

По този начин развитието на системите за управление преминава последователно от механични средства за регулиране към електрически и електронни устройства, програмируеми контролери, разпределени архитектури и съвременни мрежово свързани компютърни системи. Независимо от развитието на техническите и програмните средства, основните принципи на управление, свързани с измерване на състоянието на обекта, обработка на информацията, формиране на управляващо въздействие и използване на обратна връзка, запазват своето фундаментално значение. Съвременният етап на това развитие се характеризира с интеграцията на PLC, SCADA/HMI, IIoT и кибер-физични технологии, както и с нарастващото приложение на

отворени хардуерни и програмни решения, които създават нови възможности за изграждане на гъвкави и разширяеми компютърни системи за управление.

Изводи от глава 1

Извършеният анализ на теоретичните основи и историческото развитие на компютърните системи за управление показва, че те са преминали през последователни етапи на развитие – от механичните регулатори и електромеханичните системи, през аналоговите и цифровите системи за управление, до съвременните разпределени архитектури, базирани на програмируеми логически контролери, DCS, SCADA и индустриални комуникационни мрежи.

Развитието на микропроцесорната техника, комуникационните технологии и индустриалните мрежи създава предпоставки за изграждането на разпределени, гъвкави и разширяеми компютърни системи за управление, способни да функционират в реално време и да осигуряват интеграция между различни нива и компоненти на системите за автоматизация.

Анализът на концепциите Industry 4.0, Industrial Internet of Things (IIoT) и кибер-физичните системи показва, че съвременните системи за управление все повече се основават на интеграцията между физически процеси, вградени изчислителни устройства и комуникационни мрежи. Това разширява възможностите за обмен и обработка на данни, дистанционно наблюдение, диагностика и оптимизация на технологичните процеси.

Особено значение придобиват технологиите с отворен код (Open Source), които предоставят възможности за разработване на гъвкави, разширяеми и икономически ефективни компютърни системи за управление. Използването на Open Source хардуерни и софтуерни платформи улеснява интеграцията с различни комуникационни технологии, разработването на собствени алгоритми за управление и ограничаването на зависимостта от конкретни производители.

В резултат от извършения теоретичен анализ може да се обобщи, че използването на технологии с отворен код при изграждането на компютърни системи за управление представлява перспективно направление за развитие на индустриалната автоматизация. Възможността за съчетаване на програмируеми управляващи средства, индустриални комуникации, IIoT технологии и отворени хардуерни и програмни платформи създава предпоставки за разработване на системи с различна архитектура и функционално предназначение. Това определя необходимостта от по-задълбочено изследване на принципите, архитектурите, предимствата и ограниченията на технологиите с отворен код и възможностите за тяхното приложение в компютърните системи за управление.

Глава 2. Open Source технологии и платформи за компютърни системи за управление

Развитието на концепциите Industry 4.0, Industrial Internet of Things (IIoT) и кибер-физичните системи води до значително разширяване на приложението на технологиите с отворен код в областта на индустриалната автоматизация и компютърните системи за управление. Open Source решенията предоставят възможност за разработване на гъвкави, мащабируеми и икономически ефективни системи, които могат да бъдат адаптирани към специфичните изисквания на различни технологични процеси и производствени среди.

За разлика от традиционните затворени системи, при които хардуерът и софтуерът са обвързани с конкретен производител, Open Source технологиите осигуряват достъп до техническата документация, програмния код и възможностите за модификация и разширяване на функционалността. Това създава условия за по-лесна интеграция между различни устройства, комуникационни протоколи и програмни среди, както и за изграждане на иновативни решения в областта на индустриалната автоматизация.

Open Source Hardware предоставя редица предимства при разработването на компютърни системи за управление. Благодарение на свободния достъп до проектната документация се създават условия за по-бързо разработване на нови устройства, по-лесна интеграция с различни хардуерни и софтуерни платформи, както и за адаптиране на съществуващи решения към специфичните изисквания на конкретни приложения. Това намалява разходите за разработка, улеснява обучението и ускорява внедряването на иновативни технологии в индустриалната автоматизация.

През последните години Open Source Hardware се утвърждава като важен елемент при разработването на системи за автоматизация, Industrial Internet of Things (IIoT), роботизирани системи и интелигентни измервателни устройства. Поддръжката на стандартни комуникационни интерфейси и богатата екосистема от хардуерни разширения позволяват изграждането на надеждни и мащабируеми компютърни системи за управление.

Сред широко разпространените Open Source хардуерни платформи могат да бъдат посочени Arduino, Arduino Opta, Raspberry Pi, SparkFun и BeagleBone. Тези платформи предоставят отворени хардуерни спецификации, програмни средства и възможности за разширение, които позволяват разработването на различни системи за управление, мониторинг и Industrial Internet of Things (IIoT) приложения. В съчетание с Open Source софтуерни

решения, като OpenPLC, системи за супервизорно управление с отворен код и индустриални комуникационни платформи, те създават предпоставки за изграждане на компютърни системи за управление с висока степен на гъвкавост, свързаност и адаптивност.

В настоящата глава са разгледани основните Open Source хардуерни и софтуерни технологии и платформи, приложими при изграждането на съвременни компютърни системи за управление. Анализирани са техните основни принципи, архитектурни особености, функционални възможности и приложимост при разработването на системи за автоматизация.

2.1. Open Source Hardware

2.1.1. Същност на Open Source Hardware.

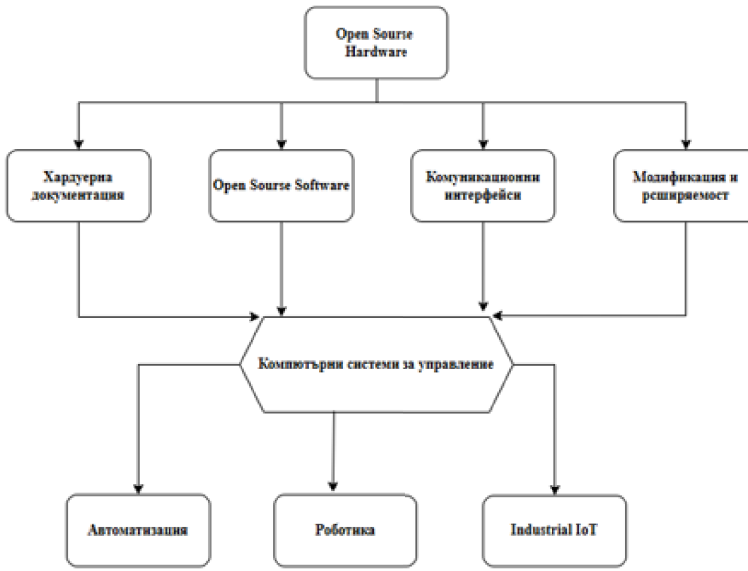
Open Source Hardware (OSH) представлява хардуер, при който проектната документация, електрическите схеми, файловете за печатни платки (PCB), конструктивните чертежи и техническите спецификации са публично достъпни и могат свободно да бъдат изучавани, модифицирани, произвеждани и разпространявани от потребителите при спазване на условията на съответния лиценз. Това определение съответства на официалната дефиниция, публикувана от Open Source Hardware Association (OSHWA) [7].

Основната идея на Open Source Hardware е създаването на отворена среда за разработване на хардуерни решения, която насърчава сътрудничеството между разработчици, научни организации, образователни институции и индустриални предприятия. Свободният достъп до проектната документация създава възможност за непрекъснато усъвършенстване на съществуващите устройства, разработване на нови функционалности и адаптиране на хардуера към специфичните изисквания на различни приложения.

За лицензиране, разпространение и използване на Open Source Hardware широко приложение намира лицензът CERN Open Hardware Licence (CERN OHL), който предоставя правна рамка за свободно използване, изучаване, модифициране и разпространение на хардуерни проекти [8], [9].

В областта на компютърните системи за управление Open Source Hardware намира приложение благодарение на възможностите за модификация, разширяване и интеграция с разнообразни програмни средства и комуникационни интерфейси. Използването на отворени хардуерни платформи значително намалява времето и разходите за разработване на нови

системи, като същевременно улеснява тяхното модифициране, разширяване и последваща поддръжка (Фиг. 8).



Фиг. 8. Концептуална схема на Open Source Hardware и приложението му в компютърните системи за управление

Развитието на Open Source Hardware през последните две десетилетия допринася за широкото му приложение в областта на индустриалната автоматизация, роботиката, Industrial Internet of Things (IIoT), интелигентните измервателни системи и вградените компютърни системи. Благодарение на активната потребителска общност и непрекъснатото развитие на хардуерните платформи, Open Source Hardware се утвърждава като надеждна основа за разработването на съвременни компютърни системи за управление.

2.1.2. Основни принципи на Open Source Hardware.

Основните принципи на Open Source Hardware произтичат от идеята за свободен достъп до проектната документация и възможността за изучаване, модифициране, производство и разпространение на хардуерните разработки при спазване на условията на съответния лиценз. Тези принципи определят начина на разработване, използване и развитие на отворените

хардуерни платформи и могат да бъдат обобщени в няколко основни направления.

Основни принципи на Open Source Hardware са:

Принцип на откритост (Openness). Един от основните принципи на Open Source Hardware е публичният достъп до цялата проектна документация. Тя включва електрически схеми, файлове за печатни платки (PCB), механични чертежи, технически спецификации и програмни библиотеки, което позволява на потребителите свободно да изучават и използват разработените решения.

Принцип на свободно използване. Потребителите имат право свободно да използват, изучават, модифицират, произвеждат и разпространяват хардуерните разработки при спазване на условията на съответния лиценз.

Принцип на модификацията. Потребителите могат да добавят нови функции, да променят схемите, да създават производни устройства, да адаптират платформата за конкретни приложения.

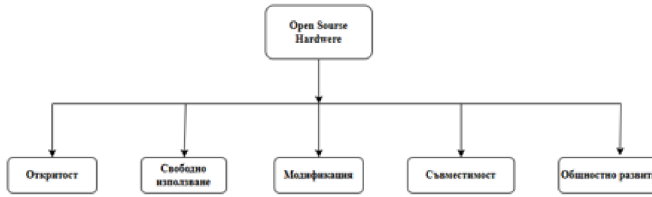
Open Source хардуерните платформи предоставят възможност за интеграция със стандартни индустриални комуникационни протоколи, като Modbus, Ethernet/IP, CAN, RS-485, MQTT и OPC UA, в зависимост от хардуерната архитектура и използвания софтуер.

Принцип на общностното развитие Разработката се осъществява съвместно от изследователи, университети, фирми, независими разработчици.

Принцип на прозрачността - Потребителят има пълен достъп до архитектурата, хардуерните схеми, интерфейсите, комуникационните протоколи.

Прилагането на тези принципи превръща Open Source Hardware в перспективна технологична основа за изграждането на съвременни компютърни системи за управление. Платформи като Arduino, Arduino Opta, SparkFun и BeagleBone предоставят възможност за реализиране на гъвкави, мащабируеми и икономически ефективни решения, отговарящи на изискванията на Industry 4.0 и Industrial Internet of Things (IIoT).

Взаимовръзката между основните принципи на Open Source Hardware е представена обобщено на фиг. 9.



Фиг. 9. Основни принципи на Open Source Hardware

Представените на фиг. 9 принципи са взаимно свързани и формират основата на отворения модел за разработване на хардуер. Достъпът до проектната документация създава възможност за изучаване и модифициране на съществуващите решения, а използването на стандартизирани интерфейси улеснява интеграцията на различни хардуерни и програмни компоненти. Общностният модел на разработка допринася за обмена на технически решения и за последващото развитие и адаптиране на съществуващи проекти към конкретни приложения.

Възможността за свободна модификация и разширяване на хардуерните решения позволява тяхното адаптиране към конкретни технологични процеси и специфични индустриални приложения. Благодарение на това Open Source Hardware намира все по-широко приложение в областта на индустриалната автоматизация, роботиката, вградените системи и Industrial Internet of Things (IIoT).

Практическата реализация на разгледаните принципи намира израз в развитието на множество хардуерни платформи с отворен код, различаващи се по архитектура, изчислителни възможности, комуникационни интерфейси и област на приложение. Основните представители на тези платформи са разгледани в следващия раздел.

2.1.3. Основни Open Source хардуерни платформи

Open Source Hardware намира приложение при разработването на компютърни системи за управление благодарение на достъпа до проектна документация, възможностите за модификация и богатата екосистема от програмни и хардуерни средства. През последните две десетилетия са разработени различни отворени или частично отворени хардуерни платформи и екосистеми, използвани в образованието, научните изследвания, прототипирането и разработването на системи за автоматизация. Сред широко използваните решения могат да бъдат посочени Arduino, Arduino Opta, Raspberry Pi и BeagleBone Black, както и развойни платки, модули и сензори, предлагани

в екосистеми като SparkFun. Тези решения се различават по своята архитектура, степен на отвореност, изчислителни възможности и предназначение. Основните им характеристики и области на приложение са обобщени в табл. 1.

Таблица 1. Основни хардуерни платформи и екосистеми с отворени технологии, приложими в компютърните системи за управление

Платформа	**Тип**	**Основни приложения**
Arduino	Микроконтролерна платформа	Автоматизация, роботика, IoT, обучение и прототипиране
Arduino Opta	Индустриално ориентиран програмируем контролер	Автоматизация, управление и IIoT
Raspberry Pi	Едноплатков компютър (SBC)*	Edge Computing, мониторинг и IIoT
SparkFun	Екосистема от развойни платки, модули и сензори	Прототипиране, измерване и обучение
BeagleBone Black	Едноплатков компютър (SBC)*	Вградени системи, управление и прототипиране

*SBC – Single Board Computer (едноплатков компютър)

Данните от Таблица 1 показват, че Open Source хардуерните платформи се различават по своята архитектура, изчислителни възможности и области на приложение. Микроконтролерните платформи, каквато е Arduino, са предназначени основно за управление на технологични процеси, работа в реално време и взаимодействие със сензори и изпълнителни механизми. От своя страна едноплатковите компютри, като Raspberry Pi и BeagleBone Black, предлагат по-висока изчислителна производителност и се използват при разработването на Edge Computing и IIoT приложения.

Сред разгледаните платформи Arduino заема особено място благодарение на своята модулна архитектура, богатата програмна екосистема и широките възможности за интеграция с различни сензори, изпълнителни механизми и комуникационни интерфейси. Развитието на платформата доведе до създаването на Arduino Opta – индустриален програмируем логически контролер (PLC), който съчетава предимствата на Open Source технологиите с изискванията на съвременната индустриална автоматизация. Поради тези причини Arduino и Arduino Opta са избрани като основен обект на разглеждане и анализ в настоящата монография.

2.1.4. Arduino

Платформата Arduino е една от най-значимите разработки в областта на Open Source Hardware и през последните две десетилетия се утвърждава като световен стандарт за разработване на вградени системи, компютърни системи за управление, роботизирани устройства и приложения в областта на Industrial Internet of Things (IIoT). Arduino представлява хардуерно-софтуерна платформа и развойна екосистема, предназначена за създаване на вградени системи, прототипи и приложения за измерване, мониторинг и управление. Екосистемата включва различни микроконтролерни и индустриално ориентирани платки, програмни средства за разработка, библиотеки и допълнителни хардуерни модули. Благодарение на достъпната среда за програмиране и широките възможности за свързване на сензорни, комуникационни и изпълнителни устройства Arduino намира приложение в образованието, научноизследователската дейност, прототипирането и разработването на системи за автоматизация.

Платформата Arduino е създадена през 2005 г. в Института за интерактивен дизайн в Ивреа (Италия), с основната цел да предостави достъпна и лесна за използване среда за разработване на електронни устройства и интерактивни системи. За разлика от много съществуващи по това време микроконтролерни платформи, Arduino предлага възможност за директно програмиране на микроконтролера, без необходимост от използване на операционна система. Това значително опростява разработването на приложения в реално време и улеснява взаимодействието между микроконтролера и външните устройства.

Сред ранните широко разпространени представители на Arduino екосистемата е Arduino Duemilanove [10], базирана на 8-битов микроконтролер от фамилията ATmega. Впоследствие Arduino екосистемата се разширява с множество платформи, сред които Arduino Uno, Mega, Nano, Due, Portenta и Arduino Opta, предназначени за различни области на приложение – от обучение и прототипиране до разработване на по-сложни вградени и индустриално ориентирани системи. Независимо от съществените различия в тяхната архитектура и технически характеристики, тези платформи са част от общата Arduino екосистема и се поддържат от свързани програмни средства, библиотеки и среди за разработка.

Освен Open Source платформа, Arduino представлява и регистрирана търговска марка, която гарантира качеството, съвместимостта и надеждността на официално произвежданите хардуерни продукти. Благодарение на

публично достъпната хардуерна документация множество производители разработват съвместими платки, които използват същата архитектура и програмна среда, което допринася за значителното разширяване на екосистемата Arduino.

Един от най-разпространените представители на Arduino екосистемата е Arduino Uno, който се използва широко при разработването на учебни, лабораторни и експериментални системи за измерване и управление. Класическата версия Arduino Uno Rev3 е базирана на микроконтролера ATmega328P и предоставя цифрови и аналогови входно-изходни линии, PWM изходи и комуникационни интерфейси, позволяващи свързването на различни периферни устройства. Развойната платка Arduino Uno е представена на фиг. 10.

Фиг. 10. Основни елементи на развойната платка Arduino Uno

Фигура 10 илюстрира основните хардуерни елементи и входно-изходни интерфейси на развойната платка Arduino Uno [10]. Те са обобщени в Таблица 2.

Основните хардуерни компоненти и комуникационни интерфейси на Arduino Uno, представени във фиг. 10 и обобщени в Таблица 2, определят функционалните възможности на платформата и нейното широко приложение при разработването на компютърни системи за управление.

Таблица 2. Основни хардуерни ресурси и комуникационни интерфейси на Arduino Uno

Хардуерен ресурс	Предназначение
ATmega328P	Основен микроконтролер
Захранване	USB или външен източник 7–12 V
Цифрови входове/изходи	Управление на цифрови устройства
Аналогови входове	Измерване на аналогови сигнали
UART	Серийна комуникация
SPI	Високоскоростна серийна комуникация
I²C	Свързване на периферни устройства
PWM	Управление на двигатели, LED и др.

Arduino Uno е една от най-широко използваните платформи от семейството Arduino поради своята проста архитектура, ниска цена, добра документация и голяма съвместимост с разнообразни сензори, изпълнителни механизми и разширителни модули.

Основните елементи на Arduino Uno включват микроконтролер ATmega328P, USB интерфейс за програмиране и комуникация с компютър, бутон Reset, стабилизатор на напрежение, конектор за външно захранване и входно-изходни пинове за свързване на външни устройства [10]. Платката може да се захранва чрез USB порта или чрез външен постояннотоков източник, обикновено в диапазона от 7 до 12 V. При разработване на самостоятелни приложения често се използва външно захранване, което осигурява необходимата мощност за платката и свързаните към нея периферни устройства.

Arduino Uno разполага с 14 цифрови входно-изходни пина, означени от D0 до D13. Част от тях могат да се използват за широчинно-импулсна модулация (PWM), което позволява управление на мощността, подавана към изпълнителни устройства, като електродвигатели, светодиоди и други товари. Цифровите пинове D0 и D1 се използват и за серийна комуникация чрез интерфейса UART, като съответно изпълняват функциите RX и TX.

Платката разполага и с шест аналогови входа, означени като A0–A5. Те са свързани към вградения аналого-цифров преобразувател на микроконтролера и позволяват измерване на аналогови сигнали в определен входен диапазон. Аналоговите входове A4 и A5 могат да се използват и за комуникация чрез интерфейса I²C, като изпълняват съответно функциите SDA и SCL. Това осигурява възможност за свързване на различни сензори, дисплеи и комуникационни модули.

Комуникационните възможности на Arduino Uno включват UART, SPI и I²C интерфейси. Чрез тях платката може да обменя данни с външни

устройства, други микроконтролери, сензорни модули и разширителни платки. Наличието на тези интерфейси прави Arduino Uno подходяща за изграждане на системи за събиране на данни, управление на изпълнителни механизми, роботизирани устройства и малки компютърни системи за управление.

Широкото използване на Arduino Uno се дължи не само на хардуерните ѝ характеристики, но и на наличието на богата програмна екосистема, включваща Arduino IDE, множество библиотеки и активна потребителска общност. Това позволява бързо разработване, тестване и модифициране на приложения, което е особено важно при изграждането на прототипи и експериментални системи за управление.

Хардуерните възможности на Arduino Uno се управляват чрез интегрираната среда за разработка Arduino IDE, която предоставя набор от програмни функции за конфигуриране на входно-изходните пинове, обработка на аналогови сигнали и управление на изпълнителни механизми. Най-често използваните функции са представени в таблица 3.

Таблица 3. Основни функции за работа с входно-изходните пинове в Arduino IDE

Функция	**Предназначение**	**Приложение в системите за управление**
pinMode()	Конфигурира вход/изход	Свързване на датчици и изпълнителни механизми
digitalRead()	Четене на цифров вход	Бутони, крайни изключватели, датчици
digitalWrite()	Управление на цифров изход	Релета, сигнализация, контактори
analogRead()	Четене на аналогов сигнал	Температура, налягане, ниво
analogWrite()	PWM управление	Регулиране на двигатели, LED осветление

Посочените функции представляват основния програмен интерфейс за управление на входно-изходните ресурси на платформата Arduino и намират широко приложение при разработването на системи за управление, мониторинг и събиране на данни.

Платформата Arduino се утвърждава като една от най-широко използваните Open Source хардуерни платформи за разработване на вградени системи и компютърни системи за управление. Благодарение на своята проста архитектура, богатите комуникационни възможности и широката софтуерна поддръжка, тя намира приложение както в образованието и научните изследвания, така и в индустриалната автоматизация. Натрупаният опит при

разработването и внедряването на решения, базирани на платформата Arduino, поставя основите за създаването на индустриални Open Source програмируеми логически контролери, какъвто е Arduino Opta.

Практическото приложение на платформата Arduino при разработването на компютърни системи за управление е разгледано подробно в глава 4, където е представена разработена система за управление на роботизиран манипулатор, реализирана на базата на Arduino Uno.

Независимо от широките възможности на Arduino Uno при лабораторни, учебни и прототипни разработки, приложението на микроконтролерни развойни платки в индустриална среда е свързано с допълнителни изисквания към захранването, електрическата защита, комуникационните интерфейси, конструктивното изпълнение и надеждността на системата. Развитието на Arduino екосистемата в посока на индустриално ориентирани управляващи решения води до появата на платформи като Arduino Opta, чиито архитектура и функционални възможности са разгледани в следващия раздел.

2.1.5. Arduino Opta

Развитието на платформата Arduino и натрупаният опит при нейното използване в областта на автоматизацията и вградените системи водят до създаването на Arduino Opta – индустриално ориентиран програмируем логически контролер (PLC), разработен в партньорство между Arduino и Finder [10]. Arduino Opta съчетава възможностите на Arduino екосистемата с функционални и конструктивни характеристики, ориентирани към приложения в областта на индустриалната автоматизация.

За разлика от традиционните развойни платки Arduino, Arduino Opta е проектиран за приложения в индустриална среда и предоставя възможности за реализация на задачи за автоматично управление, мониторинг и комуникация с други устройства и системи за автоматизация.

Фиг. 11. Общ изглед на програмируемия логически контролер Arduino Opta

Както се вижда от фиг. 11, Arduino Opta представлява компактен програмируем логически контролер, оборудван с релейни изходи, цифрови входове, Ethernet интерфейс, USB Type-C порт и светодиодна индикация за състоянието на входовете, изходите и захранването. Конструктивното и функционалното му изпълнение позволява използването му при разработване на системи за автоматизация, мониторинг и управление, включително приложения в областта на сградната автоматизация и Industrial Internet of Things (IIoT).

Хардуерната архитектура на Arduino Opta е базирана на микроконтролера STM32H747XI с двуядрена архитектура, включваща ARM Cortex-M7 с тактова честота до 480 MHz и ARM Cortex-M4 с честота до 240 MHz. Използването на два процесорни ядра предоставя възможности за разпределяне на различни изчислителни и управляващи задачи и за реализация на приложения с изисквания за работа в реално време. Контролерът разполага с цифрови входове, релейни изходи и комуникационни интерфейси, като поддържа разработване на управляващи приложения чрез програмни средства от Arduino екосистемата и PLC ориентирани среди за програмиране.

Основните технически характеристики на Arduino Opta са обобщени в табл. 4.

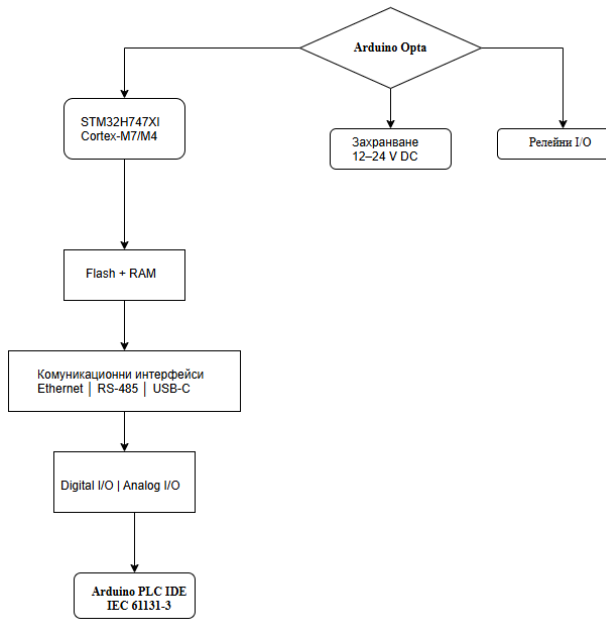
Таблица 4. Основни технически характеристики на Arduino Opta

Параметър	Стойност
Микроконтролер	STM32H747XI
Архитектура	ARM Cortex-M7 + ARM Cortex-M4
Flash памет	2 MB
RAM	1 MB
Ethernet	Да
USB Type-C	Да
RS-485	Да
Wi-Fi/Bluetooth	Да
Релейни изходи	4
Захранване	12–24 V DC

Микроконтролерната архитектура предоставя паметови ресурси, хардуерни таймери и разнообразна периферия, които позволяват реализацията на алгоритми за управление, обработка на измервателни данни и комуникация с външни устройства. Тези възможности разширяват областта на приложение на Arduino Opta в сравнение с традиционните микроконтролерни развойни платки.

Arduino Opta предоставя комуникационни възможности, позволяващи интегрирането му в мрежово свързани системи за автоматизация. Ethernet и USB Type-C са част от основните комуникационни средства, а при отделните модификации се предлагат допълнителни интерфейси и безжична свързаност. Тези възможности позволяват обмен на данни с други управляващи устройства, системи за мониторинг и по-високи нива на автоматизация, включително SCADA и IIoT базирани приложения.

Хардуерната архитектура на Arduino Opta е представена схематично на фиг. 12.



Фиг. 12. Хардуерна архитектура на програмируемия логически контролер Arduino Opta

Както се вижда от фиг. 12, хардуерната архитектура на Arduino Opta е изградена около двуядрения микроконтролер STM32H747XI, който изпълнява ролята на централен изчислителен модул. Към него са свързани оперативната и програмната памет, цифровите и аналоговите входно-изходни модули, комуникационните интерфейси и релейните изходи, което осигурява необходимите функционални възможности за реализиране на индустриални системи за управление.

Използването на двуядрена архитектура създава възможност за разпределяне на изчислителните задачи между ядрата ARM Cortex-M7 и ARM Cortex-M4. По този начин могат да бъдат реализирани паралелно различни функции, свързани с изпълнение на алгоритми за управление, обработка на данни и комуникация с външни устройства.

Освен високата изчислителна производителност, Arduino Opta разполага с индустриални комуникационни интерфейси, които позволяват

интегрирането му в локални индустриални мрежи, SCADA системи и Industrial Internet of Things (IIoT) приложения. Това превръща контролера в подходяща платформа както за изграждане на автономни системи за управление, така и за включването му като елемент от по-сложни разпределени системи за автоматизация.

Представените хардуерни характеристики на Arduino Opta показват, че контролерът съчетава възможностите на съвременните микроконтролерни платформи с функционалността на индустриалните програмируеми логически контролери. Използването на двуядрена архитектура, наличието на комуникационни интерфейси и възможностите за работа с различни входно-изходни сигнали осигуряват необходимата гъвкавост при изграждането на съвременни компютърни системи за управление.

Едно от основните предимства на Arduino Opta е възможността за програмиране както чрез традиционната развойна среда Arduino IDE, така и чрез Arduino PLC IDE, съвместима с международния стандарт IEC 61131-3 [11]. Това позволява контролерът да бъде използван както от разработчици на вградени системи, така и от специалисти в областта на индустриалната автоматизация.

Съчетаването на възможностите на Arduino екосистемата с PLC функционалност и съвременни комуникационни технологии определя Arduino Opta като перспективна платформа за разработване и изследване на системи за автоматизация, IIoT приложения и мрежово свързани системи за управление. Поради тези особености Arduino Opta е избран като една от основните хардуерни платформи при практическите и експерименталните разработки, представени в следващите глави на настоящата монография.

2.2. Open Source Software

Развитието на Open Source Hardware е тясно свързано с развитието на Open Source Software, който осигурява необходимите програмни средства за разработване, управление и интегриране на съвременни компютърни системи за управление. Благодарение на свободния достъп до изходния код и възможността за неговото модифициране, Open Source Software предоставя гъвкава основа за създаване на приложения, адаптирани към конкретните изисквания на различни технологични процеси и индустриални приложения.

Open Source Software представлява софтуер, чийто изходен код е публично достъпен и може свободно да бъде използван, изучаван, модифициран и разпространяван при условията на съответния лиценз. Този подход

насърчава сътрудничеството между разработчици, научни организации, университети и индустриални предприятия и създава условия за съвместно разработване, проверка и усъвършенстване на програмните решения.

В областта на компютърните системи за управление Open Source Software намира приложение при разработването на програмни средства за управление и автоматизация, SCADA системи, IIoT приложения, както и средства за конфигуриране, наблюдение и управление на технологични процеси. Това определя неговото значение като част от съвременните технологии за изграждане и развитие на компютърни системи за управление.

2.2.1. Същност на Open Source Software

Според Open Source Initiative (OSI) Open Source Software се характеризира не само с достъп до изходния код, но и със свободата за неговото използване, модифициране и разпространение при спазване на условията на съответния лиценз [12].

Основната идея на Open Source Software е създаването на отворена програмна среда, която насърчава сътрудничеството между разработчиците и подпомага разработването и развитието на нови програмни решения. Свободният достъп до изходния код позволява непрекъснато усъвършенстване на съществуващите приложения, отстраняване на програмни грешки, разработване на нови функционалности и адаптиране на софтуера към специфичните изисквания на различни приложения.

Разпространението на Open Source Software се осъществява чрез различни лицензи, като едни от най-широко използваните са GNU General Public License (GNU GPL), MIT License, Apache License 2.0 и BSD License [13]. Тези лицензи определят конкретните права и условия за използване, модифициране и разпространение на софтуера, като между отделните лицензи съществуват съществени различия по отношение на изискванията към производните разработки и тяхното последващо лицензиране.

Разнообразието от Open Source лицензи и модели за разработване създава възможности за използване на OSS в различни области, но същевременно изисква внимателен избор на програмните решения в зависимост от техническите, лицензионните и експлоатационните изисквания на конкретното приложение.

2.2.2. Основни принципи на Open Source Software

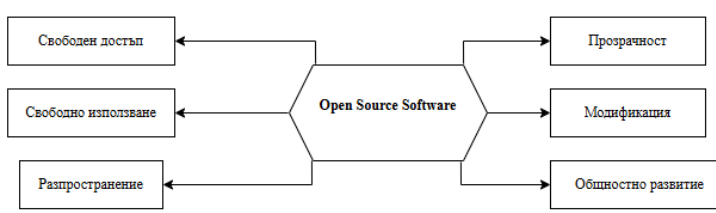
Open Source Software се основава на принципи, които осигуряват свободен достъп до програмния код, възможност за неговото модифициране и

съвместно разработване на програмни продукти. Тези принципи са определени от Open Source Initiative (OSI) и намират широко приложение при разработването на съвременни компютърни системи за управление.
Основните принципи на Open Source Software са:

- **Принцип на свободен достъп до изходния код** – изходният код на програмния продукт е публично достъпен и може да бъде изучаван и анализиран от потребителите.
- **Принцип на свободното използване –** софтуерът може да бъде използван за различни цели при спазване на условията на съответния Open Source лиценз.
- **Принцип на модификацията** – потребителите могат свободно да модифицират програмния код, да добавят нови функционалности и да адаптират софтуера към конкретни приложения.
- **Принцип на свободното разпространение** – модифицираните или оригиналните версии на програмния продукт могат да бъдат разпространявани при условията на съответния лиценз.
- **Принцип на прозрачността –** достъпът до програмния код позволява неговото анализиране, проверка и откриване и отстраняване на програмни грешки и уязвимости.

Принцип на общностното развитие – разработването и усъвършенстването на Open Source Software може да се осъществява с участието на разработчици, университети, научни организации, компании и потребителски общности, което създава условия за обмен на знания, разработване на нови функционалности и развитие на програмните решения.

Основните принципи на Open Source Software могат да бъдат представени схематично, както е показано на фиг. 13.



Фиг. 13. Основни принципи на Open Source Software

Представените на фиг. 13 принципи са взаимно свързани и определят основните характеристики на модела за разработване на Open Source

Software. Достъпът до изходния код създава възможности за неговото изучаване и модифициране, докато условията на съответния лиценз определят начина на използване и разпространение на оригиналните и производните програмни решения. Прозрачността на програмния код и възможността за участие на различни разработчици и организации създават условия за съвместно развитие, проверка и усъвършенстване на софтуера.

Прилагането на тези принципи превръща Open Source Software в технологична основа за разработването на съвременни компютърни системи за управление, програмируеми логически контролери, SCADA системи, облачни платформи и приложения в областта на Industrial Internet of Things (IIoT) [12], [13].

Прилагането на принципите на Open Source Software съществено променя начина на разработване на програмни продукти за компютърни системи за управление. Вместо използването на затворени програмни решения, разработчиците могат свободно да адаптират съществуващи програмни библиотеки, да интегрират различни комуникационни протоколи и да разработват специализирани алгоритми за управление в зависимост от конкретните технологични изисквания. Това може да улесни повторното използване на съществуващи програмни компоненти, внедряването на нови функционалности и адаптирането на системите към конкретни изисквания.

Друго съществено предимство на Open Source Software е широката поддръжка на стандартизирани програмни интерфейси и комуникационни протоколи. Благодарение на това Open Source платформите могат лесно да обменят данни с програмируеми логически контролери, сензорни мрежи, SCADA системи, облачни услуги и Industrial Internet of Things (IIoT) платформи. По този начин се осигурява висока степен на оперативна съвместимост между отделните компоненти на системите за управление.

Активното участие на световната Open Source общност също оказва съществено влияние върху развитието на програмните продукти. Непрекъснатото разработване на нови програмни библиотеки, драйвери и инструменти позволява бързо внедряване на съвременни технологии и значително улеснява разработването на надеждни системи за автоматизация. Това е една от основните причини Open Source Software да се използва все по-широко както в научните изследвания и образованието, така и в индустриалната практика.

В резултат на тези предимства през последните години бяха разработени множество Open Source програмни среди и платформи, предназначени

за разработване на компютърни системи за управление. В резултат от развитието на Open Source екосистемата са създадени различни програмни среди, платформи и инструменти, приложими при разработването на компютърни системи за управление. Сред решенията, представляващи интерес за настоящото изследване, са Arduino IDE, Arduino PLC IDE, OpenPLC, Node-RED и OpenSCADA, които са разгледани в следващите подраздели.

2.2.3. Основни Open Source софтуерни платформи.

Развитието на Open Source Software доведе до създаването на множество програмни среди, платформи и инструменти, предназначени за разработване, програмиране, конфигуриране и управление на компютърни системи. Възможностите за достъп до изходен код, модификация и общностно разработване при редица от тези решения създават условия за тяхното адаптиране и приложение в образованието, научните изследвания и автоматизацията.

В областта на компютърните системи за управление могат да бъдат използвани различни програмни среди, платформи и инструменти, сред които Arduino IDE, Arduino PLC IDE, OpenPLC, Node-RED и OpenSCADA. Всяко от тези решения притежава специфични функционални възможности и е ориентирано към различни задачи – от програмиране на микроконтролерни и програмируеми системи за управление до визуализация, мониторинг, обмен на данни и разработване на IIoT приложения.

Основните Open Source софтуерни платформи, използвани при разработването на компютърни системи за управление, са представени в Таблица 5.

Таблица 5. Основни софтуерни среди, платформи и инструменти, приложими при разработването на компютърни системи за управление

Софтуерна платформа	Основно предназначение	Поддържани технологии	Област приложение
Arduino IDE	Програмиране на Arduino	C/C++	Вградени системи, прототипиране и управление
Arduino PLC IDE	Програмиране на PLC съгласно IEC 61131-3	LD, FBD, ST, SFC	Съвместими Arduino индустриални контролери
OpenPLC	Програмируем логически контролер (Soft PLC)	IEC 61131-3	PLC системи
Node-RED	Визуално програмиране	MQTT, OPC UA, REST API	IIoT
OpenSCADA	SCADA система	Modbus, OPC UA	Мониторинг и управление

Както се вижда от Таблица 5, Open Source софтуерните платформи се различават по своето предназначение, използваните технологии и областите на приложение. Една част от тях са предназначени за разработване на програми за микроконтролерни системи и програмируеми логически контролери, докато други осигуряват средства за визуализация, мониторинг и обмен на данни в индустриални комуникационни мрежи. Различното им функционално предназначение създава възможности за комбинираното им използване при изграждането на компютърни системи за управление, включващи функции за програмиране, управление, комуникация, визуализация и обмен на данни.

В настоящата монография подробно са разгледани програмните среди Arduino IDE и Arduino PLC IDE, използвани като средства за разработване и програмиране на приложения за хардуерните платформи Arduino Uno и Arduino Opta. Това определя тяхното значение при реализирането на представените в следващите глави компютърни системи за управление.

2.2.4. Arduino IDE.

Arduino IDE (Integrated Development Environment) представлява интегрирана програмна среда с отворен код, предназначена за разработване, компилиране, отстраняване на програмни грешки и зареждане на програмно осигуряване в микроконтролерните платформи Arduino. Благодарение на своята опростена организация, широката хардуерна съвместимост и богатия набор от програмни библиотеки, Arduino IDE се утвърждава като една от най - разпространените среди за разработване на вградени системи и компютърни системи за управление [10].

Програмната среда е разработена с цел да улесни създаването на приложения както от начинаещи потребители, така и от професионални разработчици. Arduino IDE предоставя средства за редактиране на програмния код, компилиране на програмите, диагностика на програмни грешки и зареждане на готовия програмен код в микроконтролера чрез USB интерфейс.

Освен това средата разполага с вграден сериен монитор (Serial Monitor), който позволява наблюдение и анализ на обменяните данни между компютъра и управляващото устройство по време на изпълнение на програмата.

Програмите в Arduino IDE се разработват на език, базиран на C/C++, като се използват специализирани програмни библиотеки за управление на цифрови и аналогови входове и изходи, комуникационни интерфейси и разнообразни периферни устройства. Използването на готови библиотеки

значително намалява времето за разработване на приложения и позволява бързо изграждане на прототипи и експериментални системи за управление.

Разглежданата в настоящата монография програмна среда е Arduino IDE версия 2.3.4, която предоставя съвременен графичен потребителски интерфейс, подобрени средства за разработване на програмно осигуряване и разширени възможности за управление на програмни библиотеки и хардуерни платки. Основният прозорец на програмната среда е представен на фиг. 14.

Фиг. 14. Потребителски интерфейс на програмната среда Arduino IDE

Както се вижда от фиг. 14, Arduino IDE предоставя интуитивен потребителски интерфейс, който обединява всички основни средства за разработване на програмно осигуряване за микроконтролерните платформи Arduino. В централната част на прозореца е разположен редакторът на програмния код, в който се създават и редактират програмите. В горната част се намират основните команди за проверка (Verify), компилиране и зареждане (Upload) на програмата, както и меню за избор на използваната развойна платка и комуникационния порт.

Програмната среда автоматично създава основната структура на всяка програма, включваща функциите setup() и loop(). Функцията setup() се изпълнява еднократно при стартиране на микроконтролера и се използва за инициализиране на входно-изходните ресурси и периферните устройства. Функцията loop() съдържа основния алгоритъм на управление и се изпълнява циклично до прекратяване на работата на програмата. Използването на тази структура значително улеснява разработването на приложения за управление и е една от характерните особености на Arduino IDE [10].

Работата с Arduino IDE следва последователност от етапи, които обхващат редактиране на програмния код, компилиране, зареждане на програмата в микроконтролера и тестване на нейното изпълнение. Последователността на тези етапи е представена на фиг. 15 [10].

Фиг. 15. Основни етапи при разработването на програма в Arduino IDE

Както се вижда от фиг. 15, процесът на разработване започва със създаване и редактиране на програмния код. След приключване на редакцията програмата се компилира, при което се проверява нейната синтактична коректност и се генерира изпълним машинен код. При успешно компилиране програмата се зарежда в паметта на микроконтролера, след което започва нейното изпълнение. За наблюдение и диагностика на работата на програмата Arduino IDE предоставя инструмента Serial Monitor, чрез който могат да се визуализират обменяните по серийния интерфейс данни и да се анализира работата на разработеното приложение.

Освен редактиране, компилиране и зареждане на програмния код, Arduino IDE предоставя набор от инструменти, които улесняват разработването, конфигурирането и тестването на приложенията. Сред най-важните от тях са Board Manager, Library Manager и Serial Monitor, чрез които се извършва управление на поддържаните развойни платки, инсталиране на програмни библиотеки и наблюдение на обменяните данни по серийния интерфейс.

Arduino IDE предоставя възможност за работа с голям брой развойни платки, базирани както на микроконтролери AVR, така и на ARM и други съвременни архитектури. Изборът на конкретната хардуерна платформа се

извършва чрез менюто Board, където потребителят определя използваната платка и комуникационния порт. По този начин програмната среда автоматично избира необходимите програмни средства за компилиране и зареждане на програмата в съответния микроконтролер.

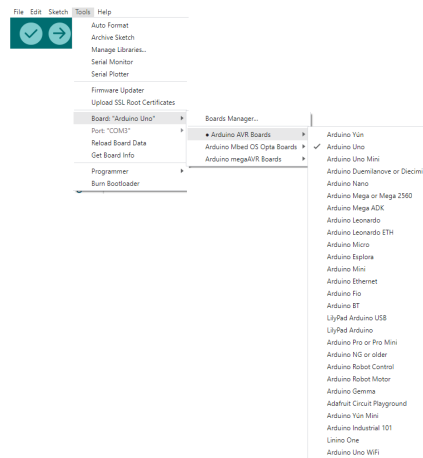
Съществено предимство на Arduino IDE е възможността за използване на голям брой програмни библиотеки, разработени както от официалния екип на Arduino, така и от световната Open Source общност. Библиотеките съдържат предварително реализирани функции за управление на цифрови и аналогови входове и изходи, комуникационни интерфейси, дисплеи, сензори, електродвигатели и разнообразни периферни устройства. Това позволява значително съкращаване на времето за разработване на програмното осигуряване и повишава надеждността на реализираните приложения.

Освен разработването на програмния код, Arduino IDE предоставя средства за диагностика и анализ на работата на програмите. Чрез вградения инструмент Serial Monitor могат да бъдат наблюдавани данните, обменяни между микроконтролера и персоналния компютър посредством серийния интерфейс. Това улеснява проверката на правилното функциониране на алгоритмите за управление, както и откриването и отстраняването на програмни грешки по време на разработването и тестването на приложенията.

За по-сложни приложения Arduino IDE предоставя възможност за организиране на програмния код в отделни файлове и програмни модули, което улеснява разработването, поддръжката и разширяването на проектите. Чрез използването на външни библиотеки и потребителски функции може да бъде реализирана модулна структура на програмното осигуряване, което повишава неговата четимост, повторна използваемост и надеждност.

Благодарение на широката си хардуерна съвместимост, богатия набор от програмни библиотеки и активната потребителска общност Arduino IDE се използва не само за обучение и разработване на експериментални системи, но и при създаването на прототипи на компютърни системи за управление, роботизирани устройства, системи за събиране на данни и приложения в областта на Internet of Things (IoT) и Industrial Internet of Things (IIoT). Тези възможности определят приложимостта на Arduino IDE при разработването на учебни, експериментални и прототипни системи, както и на приложения за събиране на данни, управление и IoT.

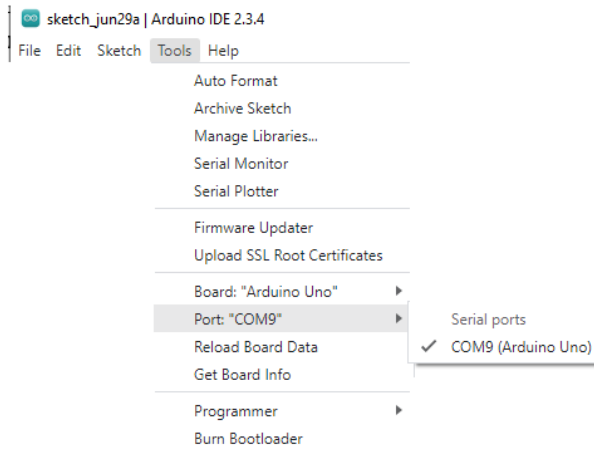
Една от първите стъпки при разработването на приложение в Arduino IDE е изборът на използваната хардуерна платка. От правилното конфигуриране на този параметър зависи изборът на съответния компилатор, настройките за генериране на машинния код и параметрите за зареждане на програмата в паметта на микроконтролера. Програмната среда предоставя възможност за избор измежду голям брой официално поддържани развойни платки, както и на допълнителни платформи чрез инсталиране на съответните пакети. Изборът на хардуерната платка се извършва чрез менюто Board, представено на фиг. 16.



Фиг. 16. Избор на хардуерна платка чрез менюто Board в Arduino IDE

Както се вижда от фиг. 16, Arduino IDE предоставя възможност за избор на различни модели развойни платки от семейството Arduino, включително Arduino Uno, Arduino Nano, Arduino Mega 2560, Arduino Leonardo и други. След избора на конкретна платка програмната среда автоматично конфигурира необходимите параметри за компилиране и зареждане на програмния код, което осигурява съвместимост между разработваното приложение и използвания микроконтролер. При необходимост чрез Board Manager могат да бъдат добавени и програмни пакети за нови хардуерни платформи, което значително разширява възможностите на Arduino IDE.След избора на развойната платка е необходимо да бъде определен и комуникационният сериен порт, чрез който се осъществява обменът на данни между персоналния компютър и микроконтролера. При свързване на платката Arduino чрез USB интерфейс програмната среда автоматично разпознава наличните комуникационни портове и предоставя възможност за избор на съответния COM порт. Коректното конфигуриране на комуникационния порт е необходимо

условие за успешно зареждане на програмата и за обмен на данни между разработваното приложение и управляващото устройство.



Фиг. 17. Избор на комуникационен сериен порт в Arduino IDE

Както се вижда от фиг. 17, Arduino IDE визуализира наличните комуникационни портове и позволява избор на порта, към който е свързана използваната развойна платка. След правилното конфигуриране на комуникационния порт програмната среда може да осъществи зареждане на компилирания програмен код в паметта на микроконтролера, както и последваща серийна комуникация по време на изпълнение на програмата.

Избирайки хардуерна платка и съответния комуникационен порт Arduino IDE е готова за разработване на програмно осигуряване. Следващият етап включва компилиране на програмния код, при което средата извършва проверка за синтактични грешки, анализира използваните програмни библиотеки и генерира изпълним машинен код, предназначен за конкретния микроконтролер. При успешно приключване на компилацията програмата може да бъде заредена във Flash паметта на управляващото устройство посредством USB интерфейса.

При микроконтролерни платки като Arduino Uno зареждането на програмния код се реализира чрез съответния bootloader и комуникационния интерфейс на платката. След приключване на зареждането микроконтролерът автоматично рестартира изпълнението и започва работа с новата програма. Този механизъм позволява бързо тестване и многократно актуализиране на програмното осигуряване по време на разработването и експерименталните изследвания.

За реализиране на взаимодействието между програмното осигуряване и външните устройства Arduino IDE предоставя набор от стандартни входно-изходни функции. Те позволяват конфигуриране на цифровите входове и изходи, управление на изпълнителни механизми, както и получаване на информация от цифрови и аналогови сензори. Използването на унифициран програмен интерфейс значително улеснява разработването на приложения и осигурява висока степен на преносимост между различните хардуерни платформи Arduino.

Конфигурирането на режима на работа на цифровите пинове се извършва чрез функцията pinMode(), която определя дали съответният извод ще бъде използван като вход или изход. След инициализацията управлението на цифровите сигнали се осъществява чрез функциите digitalWrite() и digitalRead(), които позволяват съответно записване и прочитане на логическите състояния на цифровите входно-изходни линии.

При работа с аналогови сигнали Arduino IDE предоставя функциите analogRead() и analogWrite(). Функцията analogRead() извършва аналогово-цифрово преобразуване на входното напрежение и връща числена стойност, пропорционална на измереното напрежение. При платки като Arduino Uno функцията analogWrite() се използва за формиране на PWM сигнал на съответните поддържани изходи, което позволява регулиране на средната стойност на управляващото въздействие към подходящи изпълнителни устройства.

Освен функции за управление на входно-изходните ресурси Arduino IDE предоставя богат набор от вградени математически функции, предназначени за обработка на числови данни и реализиране на инженерни изчисления. Сред най-често използваните са функциите min(), max(), abs(), map(), constrain(), pow(), sqrt(), както и тригонометричните функции sin(), cos() и tan(). Тези функции намират широко приложение при обработката на данни от измервателни сензори, мащабиране на аналогови сигнали, изчисляване на физични величини и реализиране на алгоритми за управление в реално време.

Например функцията map() се използва за преобразуване на стойности от един числов диапазон в друг, което е особено полезно при обработването на аналогови сигнали. При Arduino Uno използваният 10-битов аналого-цифров преобразувател представя измерената стойност в диапазона от 0 до 1023.

При разработването на програми за компютърни системи за управление често се налага вземане на решения в зависимост от текущото състояние на входните сигнали, измерените величини или предварително зададени условия. За тази цел Arduino IDE предоставя възможност за използване на логически изрази и условни оператори, които позволяват реализиране на различни алгоритми за управление. Логическите изрази формират резултат от тип *true* или *false*, като се изграждат чрез оператори за сравнение (==, !=, <,

>, <=, >=) и логически оператори (&&, ||, !). Комбинирането на тези оператори позволява реализиране на сложни условия, използвани при управлението на технологични процеси и обработката на данни от сензори.

На основата на логическите изрази се реализират операторите за разклонение, които определят последователността на изпълнение на програмата. Най-често използваният условен оператор е if, чрез който се изпълнява определен блок от инструкции само при изпълнение на зададеното условие.

При необходимост от избор между две алтернативи се използва конструкцията if...else, а при наличие на множество възможни стойности на една променлива широко приложение намира операторът switch...case. Използването на тези програмни конструкции позволява изграждането на гъвкави алгоритми за управление, способни да реагират на промени в състоянието на управлявания обект или на външната среда.

Освен операторите за разклонение Arduino IDE предоставя и средства за организиране на циклично изпълнение на програмния код. Циклите позволяват многократно изпълнение на определена последователност от инструкции, което е необходимо при реализиране на периодични измервания, обработка на сигнали, управление на изпълнителни механизми и обмен на данни с външни устройства. Най-често използваните циклични конструкции са for, while и do...while, като изборът между тях зависи от начина на управление на повторенията и условията за тяхното прекратяване.

Използването на логически изрази, условни оператори и циклични конструкции представлява основата за разработването на алгоритми за управление в Arduino IDE. Чрез тях могат да бъдат реализирани както прости алгоритми за управление на отделни изпълнителни механизми, така и сложни системи за автоматизация, включващи обработка на данни от множество сензори, вземане на решения в реално време и управление на технологични процеси. Именно тези програмни средства осигуряват необходимата гъвкавост при разработването на приложения за компютърни системи за управление.

Програмите, разработвани в Arduino IDE, използват синтаксис, базиран на езиците C/C++, което позволява прилагане на стандартни програмни конструкции при разработването на алгоритми за управление. Освен задължителните функции setup() и loop(), програмният код може да съдържа променливи, константи, аритметични и логически изрази, условни оператори, цикли, масиви и потребителски функции. Това осигурява необходимата гъвкавост за реализиране както на прости управляващи алгоритми, така и на по-сложни приложения за обработка на данни, мониторинг и автоматизация.

Основните типове данни, използвани в Arduino IDE, включват цели числа, числа с плаваща запетая, логически променливи, символни данни и масиви. Изборът на подходящ тип данни има важно значение при разработването на приложения за микроконтролерни системи, тъй като ресурсите на

микроконтролера са ограничени по отношение на оперативна памет и изчислителна мощност. Поради това при създаване на програми за управление е необходимо да се отчита както диапазонът на представяните стойности, така и необходимата точност на изчисленията.

Променливите се използват за съхраняване на текущи стойности, получени от сензори, междинни резултати от изчисления, състояния на изпълнителни механизми и параметри на управляващия алгоритъм. Чрез аритметични изрази могат да се извършват пресмятания, мащабиране на аналогови сигнали, изчисляване на физични величини и формиране на управляващи въздействия. В този контекст операторите за инкрементиране и декрементиране намират приложение при броячи, циклично сканиране на масиви и реализиране на времеви зависимости в програмата.

Едномерните масиви позволяват съхраняване и обработка на множество еднотипни стойности, което е особено полезно при работа с данни от сензори, буфериране на измервания или реализиране на прости филтриращи алгоритми. Потребителските функции от своя страна дават възможност за структуриране на програмния код в отделни логически блокове, което улеснява разработването, тестването и поддръжката на по-сложни приложения.

Указателите, макар и поддържани от езика C/C++, обикновено не са необходими при разработването на базови приложения в Arduino IDE. Те се използват основно при по-сложни програмни решения, свързани с директна работа с паметта, динамични структури от данни или оптимизиране на програмния код. В контекста на компютърните системи за управление тяхното приложение трябва да бъде внимателно обосновано, тъй като неправилната работа с паметта може да доведе до трудно откриваеми програмни грешки.

При разработването на приложения в Arduino IDE се използват разнообразни програмни конструкции, които осигуряват необходимите средства за реализиране на алгоритми за управление, обработка на данни и взаимодействие с външни устройства. Те включват основни типове данни, променливи, аритметични и логически изрази, условни оператори, циклични конструкции, масиви и потребителски функции. Чрез тях могат да бъдат реализирани както прости програми за управление на входно-изходни устройства, така и сложни алгоритми за автоматизация на технологични процеси. Основните програмни конструкции, използвани в Arduino IDE, са обобщени в табл. 6 [10].

Таблица 6. Основни програмни конструкции в Arduino IDE

Елемент	**Приложение в системите за управление**
Променливи	Съхраняване на измерени стойности и състояния
Аритметични изрази	Пресмятане и мащабиране на сигнали
Логически изрази	Проверка на условия
Условни оператори	Вземане на решения
Цикли	Повтаряща се обработка
Масиви	Съхраняване на множество измервания
Потребителски функции	Структуриране и многократно използване на програмния код

Както се вижда от табл. 6, Arduino IDE предоставя пълен набор от програмни средства, необходими за разработването на приложения за вградени системи и компютърни системи за управление. Комбинирането на различните програмни конструкции позволява изграждането на модулни и лесни за поддръжка алгоритми, които могат да бъдат адаптирани към широк кръг практически приложения.

От изложеното може да се направи изводът, че Arduino IDE представлява интегрирана програмна среда, предоставяща основните средства за разработване, компилиране, зареждане и диагностика на програмно осигуряване за поддържаните Arduino платформи. Благодарение на достъпния потребителски интерфейс, широкия набор от програмни библиотеки и поддръжката на различни хардуерни платформи, средата намира приложение в обучението, научноизследователската дейност, прототипирането и разработването на микроконтролерно базирани системи за измерване и управление.

С развитието на платформата Arduino и навлизането ѝ в областта на индустриалната автоматизация възниква необходимост от програмна среда, която да поддържа стандартите, използвани при програмирането на програмируеми логически контролери. В тази връзка е разработена Arduino PLC IDE – специализирана среда, предназначена за разработване на индустриални приложения съгласно стандарта IEC 61131-3. Нейните основни характеристики и функционални възможности са разгледани в следващия подраздел.

2.2.5. Arduino PLC IDE

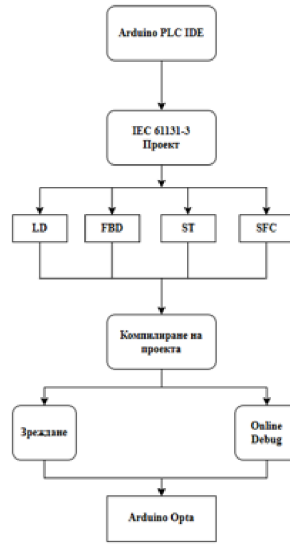
С развитието на платформата Arduino и навлизането ѝ в областта на индустриалната автоматизация възниква необходимост от програмна среда, която да осигурява разработването на приложения за програмируеми логически контролери чрез утвърдените стандарти и подходи в индустриалната автоматизация. Arduino PLC IDE е предназначена за разработване на PLC приложения за съвместими индустриално ориентирани контролери от Arduino екосистемата, включително Arduino Opta. В отговор на тази необходимост Arduino разработва програмната среда Arduino PLC IDE,

предназначена за разработване на приложения за индустриални контролери от серията Arduino Opta и други съвместими устройства.
За разлика от Arduino IDE, която използва езика C/C++ за разработване на вградени приложения, Arduino PLC IDE е ориентирана към разработването на системи за индустриална автоматизация съгласно международния стандарт IEC 61131-3. Средата предоставя възможност за разработване на програмни проекти чрез използване на стандартизирани програмни езици, широко прилагани при съвременните програмируеми логически контролери, като Ladder Diagram (LD), Function Block Diagram (FBD), Structured Text (ST) и Sequential Function Chart (SFC).

Arduino PLC IDE предоставя интегрирана среда за създаване, конфигуриране, компилиране, зареждане и диагностика на PLC приложения. Освен средствата за разработване на програмния код, средата включва инструменти за управление на проектите, конфигуриране на входно-изходните ресурси, наблюдение на стойностите на променливите в реално време, диагностика на изпълняваната програма и обмен на данни с индустриални комуникационни мрежи. Благодарение на тези функционални възможности Arduino PLC IDE позволява реализирането както на самостоятелни системи за управление, така и на приложения, интегрирани в архитектурата на съвременните индустриални автоматизирани системи.

Използването на Arduino PLC IDE разширява възможностите на Arduino екосистемата за разработване на приложения в областта на автоматизацията. В съчетание с контролери като Arduino Opta средата предоставя средства за реализиране на системи за управление, мониторинг и комуникация, включително приложения, свързани с IIoT технологии. Това позволява използването на контролери като Arduino Opta при изграждането на системи за управление, мониторинг и Industrial Internet of Things (IIoT), съответстващи на съвременните изисквания на концепцията Industry 4.0.

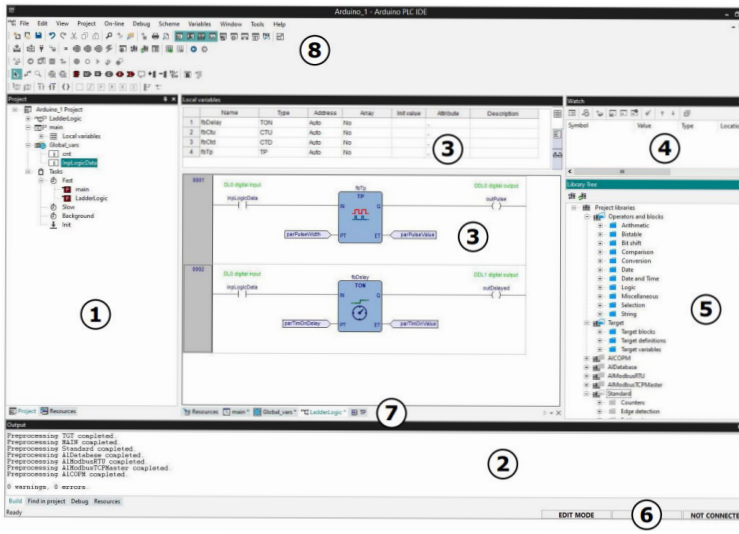
Функционалните възможности на Arduino PLC IDE обхващат целия процес на разработване на PLC приложения – от създаването на проекта и избора на програмен език, през компилирането на програмния код, до неговото зареждане в програмируемия логически контролер и последващата диагностика в реално време. Благодарение на поддръжката на стандарта IEC 61131-3 разработчикът може да използва различни програмни езици в зависимост от спецификата на конкретното приложение, като след компилиране програмата се зарежда в контролера и може да бъде наблюдавана и анализирана чрез средствата за онлайн мониторинг и диагностика. Основните етапи при разработването и изпълнението на PLC проект в Arduino PLC IDE са представени на фиг. 18.



Фиг. 18. Основни етапи при разработване на PLC приложение в Arduino PLC IDE

Както се вижда от фиг. 18, процесът на разработване започва със създаване на PLC проект в Arduino PLC IDE и избор на подходящ програмен език съгласно стандарта IEC 61131-3. След разработването на алгоритъма програмата се компилира, при което се извършва проверка за синтактични и логически грешки. При успешно компилиране проектът се зарежда в контролера Arduino Opta, след което могат да бъдат използвани средствата за онлайн мониторинг и диагностика, позволяващи наблюдение на изпълнението на програмата в реално време и улесняващи процеса на настройка и отстраняване на грешки.

Потребителският интерфейс на Arduino PLC IDE е организиран като интегрирана среда за разработване на PLC приложения, в която са обединени всички необходими средства за създаване, редактиране, компилиране, конфигуриране и диагностика на програмните проекти. Работното пространство е структурирано в отделни функционални панели, които осигуряват бърз достъп до елементите на проекта, редактиране на програмния код, управление на променливите и наблюдение на изпълнението на програмата в реално време. Работното пространство на Arduino PLC IDE е представено на фиг. 19.



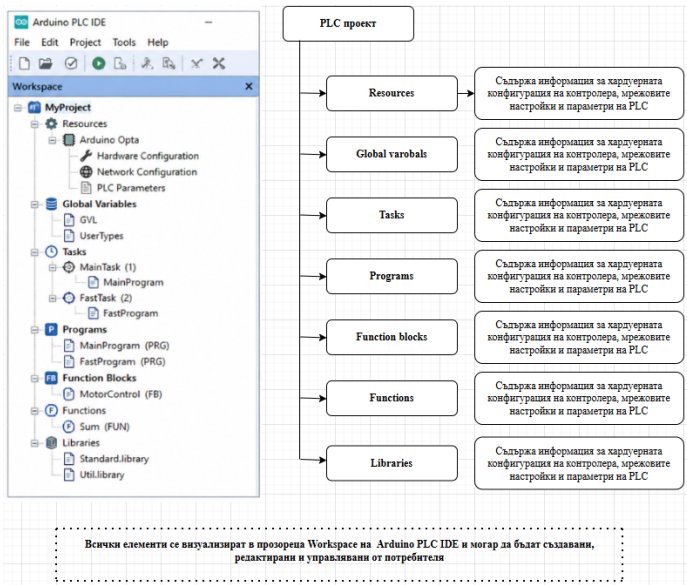
Фиг. 19. Работно пространство на Arduino PLC IDE

Както се вижда от фиг. 19, работното пространство на Arduino PLC IDE е разделено на няколко основни функционални области. С (1) е означен прозорецът Workspace, чрез който се организира структурата на проекта и се осъществява достъп до неговите програмни модули и ресурси. (2) обозначава Output Window, в който се визуализират съобщенията от компилатора, диагностичната информация и резултатите от изпълнението на програмата. С (3) са означени редакторите на програмния код, предназначени за разработване на PLC приложения на езиците, дефинирани в стандарта IEC 61131-3. (4) представлява прозорецът Watch, използван за наблюдение на стойностите на променливите по време на изпълнение на програмата, а (5) обозначава Library Tree, съдържащо библиотеките с програмни функции, функционални блокове и готови програмни компоненти. В долната част на прозореца е разположена лентата на състоянието (6), която предоставя информация за текущото състояние на проекта и връзката с контролера. (7) обозначава лентата на документите, чрез която се осъществява превключване между отворените файлове на проекта, а (8) показва лентите с инструменти, осигуряващи бърз достъп до основните функции за създаване, редактиране, компилиране и зареждане на PLC програмите.

Организацията на работното пространство позволява едновременно разработване, конфигуриране, компилиране и диагностика на PLC приложения, което значително улеснява процеса на разработване и въвеждане в експлоатация на системите за управление.

След запознаване с работното пространство на Arduino PLC IDE следващият етап е създаването на нов PLC проект. Проектът представлява основната организационна единица в програмната среда и съдържа всички програмни модули, конфигурационни файлове, променливи, ресурси и настройки, необходими за разработването на приложението. Чрез проекта се организира цялата логическа структура на програмата и се осигурява възможност за последващо компилиране, зареждане и изпълнение върху програмируемия логически контролер Arduino Opta.

При създаването на нов проект потребителят избира типа на контролера, който ще бъде програмиран, както и основните параметри на програмното приложение. След създаването на нов PLC проект Arduino PLC IDE автоматично генерира неговата основна структура, съдържаща всички необходими програмни и конфигурационни елементи за разработването на приложението. Проектът е организиран йерархично, като включва ресурси, програмни организационни единици (POUs), задачи (Tasks), глобални променливи, библиотеки и други компоненти, които осигуряват модулна организация на програмния код и улесняват разработването на сложни алгоритми за управление. Благодарение на тази структура отделните функционални модули могат да бъдат разработвани, тествани и поддържани независимо един от друг, което повишава надеждността и улеснява последващото разширяване на проекта. Структурата на PLC проект в Arduino PLC IDE е представена на фиг. 20.



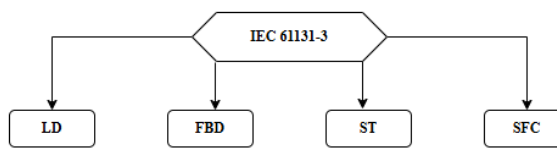
Фиг. 20. Структура на PLC проект в Arduino PLC IDE

Както се вижда от фиг. 20, всеки PLC проект в Arduino PLC IDE е организиран като йерархична структура от взаимосвързани програмни и конфигурационни компоненти. Основен елемент на проекта са ресурсите (Resources), които съдържат информация за хардуерната конфигурация на контролера, комуникационните параметри и останалите настройки, необходими за неговото функциониране. Глобалните променливи (Global Variables) осигуряват обмен на данни между отделните програмни модули, докато задачите (Tasks) определят начина и периодичността на изпълнение на програмите.

Алгоритмите за управление се реализират чрез програмните организационни единици (Program Organization Units – POUs), които включват програми (Programs), функционални блокове (Function Blocks) и функции (Functions). Тази организация позволява разделянето на сложните алгоритми на отделни функционални модули, което улеснява тяхното разработване, тестване, повторно използване и последваща поддръжка. Използването на програмни библиотеки (Libraries) предоставя достъп до стандартни функции, функционални блокове и готови програмни компоненти, което съкращава времето за разработване и повишава надеждността на PLC приложенията.

Йерархичната организация на PLC проекта е едно от основните предимства на Arduino PLC IDE, тъй като осигурява модулност, добра структурираност на програмния код и възможност за разработване на мащабни приложения за индустриална автоматизация. Подобен подход съответства на принципите на стандарта IEC 61131-3 и се използва в съвременните програмни среди за разработване на приложения за програмируеми логически контролери. Йерархичната организация на PLC проекта е пряко свързана с изискванията на международния стандарт IEC 61131-3, който определя общите принципи за разработване на програмно осигуряване за програмируеми логически контролери.

Една от основните характеристики на Arduino PLC IDE е поддръжката на международния стандарт IEC 61131-3, които дефинира програмните езици, използвани при разработването на приложения за програмируеми логически контролери, като всеки от тях е предназначен за реализиране на определен тип алгоритми за управление. Arduino PLC IDE поддържа четири от езиците, включени в стандарта – Ladder Diagram (LD), Function Block Diagram (FBD), Structured Text (ST) и Sequential Function Chart (SFC). Това предоставя възможност за избор на най-подходящия програмен език в зависимост от особеностите на конкретното приложение и предпочитанията на разработчика. Програмните езици, поддържани от Arduino PLC IDE съгласно стандарта IEC 61131-3, са представени на фиг. 21.



Фиг. 21. Програмни езици, поддържани от Arduino PLC IDE съгласно IEC 61131-3

Както се вижда от фиг. 21, Arduino PLC IDE поддържа четири програмни езика, които се различават по начина на представяне на алгоритмите и областите си на приложение. Ladder Diagram (LD) е графичен език, базиран на релейно-контакторната логика и е особено подходящ за дискретно управление. Function Block Diagram (FBD) използва взаимносвързани функционални блокове и намира широко приложение при реализиране на алгоритми за обработка на аналогови сигнали и регулиране. Structured Text (ST) представлява високонивоен текстов език, предназначен за разработване на сложни изчислителни алгоритми, а Sequential Function Chart (SFC) се използва за описание на последователността на изпълнение на технологичните операции.

- Ladder Diagram (LD):

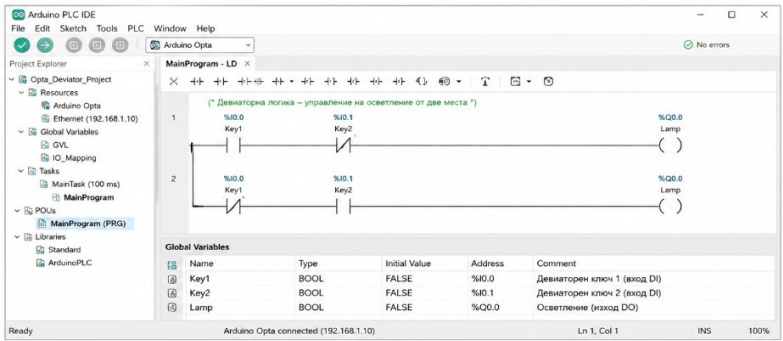
Ladder Diagram (LD) е един от най-широко използваните програмни езици, дефинирани в международния стандарт IEC 61131-3. Езикът е разработен като графично представяне на класическите релейно-контакторни схеми за управление, поради което е интуитивен и лесно разбираем както за програмисти, така и за инженери по автоматизация. Програмите се изграждат чрез последователно свързване на контакти, бобини, таймери, броячи и други функционални елементи, които описват логиката на управление на технологичния процес.

Основното предимство на Ladder Diagram е неговата визуална организация. Програмата се представя като поредица от логически стъпала (rungs), които се изпълняват циклично от PLC контролера. Всяко стъпало съдържа логически условия, реализирани чрез нормално отворени и нормално затворени контакти, както и изпълнителни елементи, представени чрез бобини или функционални блокове. При изпълнение на програмата контролерът последователно анализира състоянието на входните сигнали и определя състоянието на съответните изходи.

Благодарение на своята структура Ladder Diagram намира широко приложение при разработването на дискретни системи за управление, включващи електродвигатели, релета, контактори, пневматични и хидравлични изпълнителни механизми, транспортни линии и производствени машини. Графичното представяне на алгоритмите улеснява тяхното разработване, диагностика и последваща поддръжка, което е една от причините LD да остане най-разпространеният език за програмиране на програмируеми логически контролери.

В Arduino PLC IDE езикът Ladder Diagram е напълно интегриран в работната среда и позволява създаването на PLC приложения чрез използване на стандартизирани графични елементи. След създаването на програмата тя се компилира и зарежда в контролера Arduino Opta, като по време на изпълнение средата предоставя възможност за онлайн наблюдение на логическите състояния на контактите, бобините и останалите програмни елементи. Това значително улеснява настройката, диагностицирането и отстраняването на програмни грешки по време на разработването на приложения за индустриална автоматизация.

За илюстриране на възможностите на езика Ladder Diagram на фиг. 22 е представен пример за PLC програма, реализираща управление на осветление чрез два девиаторни ключа. Програмата е разработена в Arduino PLC IDE и демонстрира използването на основните логически елементи за управление на цифровите входове и изходи на контролера Arduino Opta.



Фиг. 22. Пример за програма, разработена на езика Ladder Diagram (LD) в Arduino PLC IDE

Както се вижда от фиг. 22, програмата използва два цифрови входа (%I0.0 и %I0.1), свързани към двата девиаторни ключа, и един цифров изход (%Q0.0), управляващ осветителното тяло. Логиката е реализирана чрез две паралелни разклонения, които съответстват на операцията „изключващо ИЛИ“ (XOR). При различно положение на двата ключа се активира едно от логическите разклонения и изходът %Q0.0 включва осветлението. При еднакво положение на ключовете и двете логически вериги остават неактивни и изходът се изключва. Представеният пример демонстрира нагледно начина, по който чрез Ladder Diagram могат да бъдат реализирани дискретни логически зависимости, широко използвани в сградната и индустриалната автоматизация.

За по-голяма яснота и с цел онагледяване на връзката между програмната реализация и използваните хардуерни ресурси на контролера Arduino Opta, в Таблица 7 са представени входовете и изходите, използвани при реализирането на примерната LD програма. Посочени са техните адреси в PLC проекта, условните обозначения, функционалното им предназначение и типът на обработваните сигнали.

Таблица 7. Описание на входовете и изходите, използвани в примерната LD програма

Адрес	Име на променливата	Описание	Тип сигнал
%I0.0	Key 1	Девиаторен ключ 1	Дигитален вход
%I0.1	Key 2	Девиаторен ключ 2	Дигитален вход
%Q0.0	Lamp	Осветително тяло	Дигитален изход

От данните в Таблица 7 се вижда, че примерната програма използва два цифрови входа (Key 1 и Key 2) и един цифров изход (Lamp). Входовете

приемат сигналите от двата девиаторни ключа, а изходът управлява осветителното тяло. Използването на именувани променливи вместо директно адресиране на входовете и изходите улеснява разработването, четимостта и поддръжката на PLC програмата, особено при реализиране на по-сложни алгоритми за управление.

Основните програмни елементи, използвани при разработването на приложения на езика Ladder Diagram, са нормално отворени (Normally Open – NO) и нормално затворени (Normally Closed – NC) контакти, бобини (Coils), таймери (Timers), броячи (Counters), логически оператори, функционални блокове и различни типове вътрешни променливи. Чрез комбинацията от тези елементи могат да бъдат реализирани както прости логически зависимости, така и сложни алгоритми за управление на технологични процеси. Всеки от програмните елементи изпълнява строго определена функция при обработването на входните сигнали и формирането на управляващите въздействия към изпълнителните механизми.

За по-голяма яснота основните програмни елементи, използвани при разработването на PLC приложения на езика Ladder Diagram, тяхното предназначение и основните им функционални характеристики са представени в Таблица 8.

Таблица 8. Основни програмни елементи и тяхното графично обозначение в Ladder Diagram

Програмен елемент	**Графично обозначение в LD**	**Предназначение**
Normally Open Contact	NO - ─[ ]─	Проверка за логическа единица (TRUE)
Normally Closed Contact	NC - ─[/]─	Проверка за логическа нула (FALSE)
Coil	Coil - ─( )─	Управление на цифров изход или вътрешна променлива
Set Coil	─(S)─	Установява изхода в логическа единица
Reset Coil	─(R)─	Нулира изхода в логическа нула
Timer ON Delay	TON	Закъснение при включване
Timer OFF Delay	TOF	Закъснение при изключване
Pulse Timer	TP	Генериране на импулс с определена продължителност
Counter Up	CTU	Увеличаване стойността на брояч
Counter Down	CTD	Намаляване стойността на брояч
Up/Down Counter	CTUD	Двупосочно преброяване на събития

Както се вижда от Таблица 8, Ladder Diagram предоставя богат набор от програмни елементи, които позволяват реализиране на разнообразни алгоритми за дискретно управление. Комбинирането на логически контакти, бобини, таймери и броячи осигурява възможност за разработване на надеждни и лесни за поддръжка PLC приложения, като същевременно съхранява логиката на традиционните релейно-контакторни схеми. Благодарение на

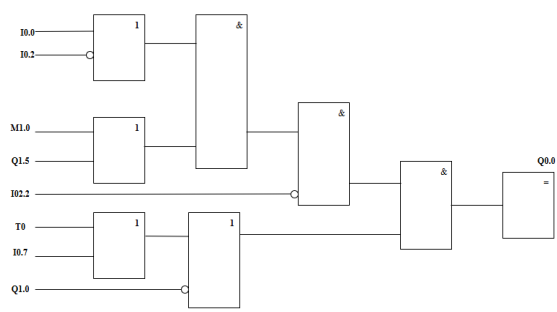
това Ladder Diagram остава един от най-предпочитаните програмни езици в индустриалната автоматизация.

- Function Block Diagram (FBD)

Function Block Diagram (FBD) е графичен програмен език, дефиниран в международния стандарт IEC 61131-3, при който алгоритмите за управление се изграждат чрез взаимно свързани функционални блокове. Всеки функционален блок изпълнява определена логическа, математическа или технологична функция, а обменът на информация между отделните блокове се осъществява посредством входни и изходни сигнали. Благодарение на своята модулна структура FBD позволява изграждането на ясно организирани и лесни за поддръжка програмни приложения.

В Arduino PLC IDE езикът Function Block Diagram се използва при разработването на алгоритми за обработка на аналогови сигнали, управление на технологични процеси, математически изчисления, регулиране и обмен на данни между отделните програмни модули. Използването на предварително разработени функционални блокове намалява сложността на програмния код, ускорява процеса на разработване и повишава надеждността на PLC приложенията.

За онагледяване на възможностите на езика Function Block Diagram на фиг. 23 е представен пример за PLC програма, разработена в Arduino PLC IDE. В програмата логическите зависимости между входните сигнали са реализирани чрез функционални блокове, като резултатите се подават към изходни променливи, свързани с цифровите изходи на контролера Arduino Opta.

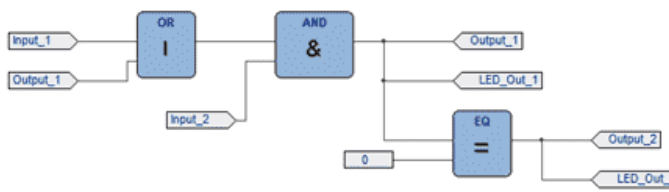


Фиг. 23. Пример за реализиране на логическа функция чрез езика Function Block Diagram (FBD).

Както се вижда от фиг. 23, логическата функция се реализира чрез последователно свързване на функционални блокове, които изпълняват логически операции върху входните сигнали. Всеки блок обработва получените входни данни и формира изходен сигнал, който може да бъде използван като вход за следващ блок. По този начин сложните алгоритми могат да бъдат

изграждани чрез комбиниране на отделни функционални блокове, което осигурява добра четимост, модулност и лесна поддръжка на програмния код. В Arduino PLC IDE функционалните блокове се избират от вградените библиотеки и се свързват графично, като по този начин се изгражда цялостната логика на програмата.

За онагледяване на практическата реализация на езика Function Block Diagram на фиг. 24 е представен пример за PLC програма, разработена в Arduino PLC IDE.



Фиг. 24. Пример за FBD програма, реализирана в Arduino PLC IDE

Както се вижда от фиг. 24, алгоритъмът е реализиран чрез последователно свързване на функционални блокове, които изпълняват логически операции върху входните сигнали. Блокът OR формира логическа единица при активиране на поне един от входовете, след което резултатът се подава към блока AND, където се комбинира с втори входен сигнал. Полученият резултат управлява изходната променлива Output_1 и индикатора LED_Out_1. Втората част на алгоритъма използва блока EQ, който сравнява стойността на входния сигнал с предварително зададена константа. При изпълнение на условието се активират Output_2 и LED_Out_2. Представеният пример демонстрира начина на изграждане на алгоритми чрез функционални блокове, което е основна особеност на езика Function Block Diagram.

Разработването на алгоритми на езика Function Block Diagram се основава на използването на стандартни функционални блокове, дефинирани в стандарта IEC 61131-3. Всеки функционален блок изпълнява конкретна логическа, математическа или технологична операция и може да бъде многократно използван при изграждането на PLC приложения. Комбинирането на различни функционални блокове позволява реализирането както на прости логически зависимости, така и на сложни алгоритми за управление на технологични процеси.

За по-голяма яснота основните функционални блокове, използвани при разработването на приложения на езика Function Block Diagram, са представени в Таблица 9

Таблица 9 Основни функционални блокове, използвани във Function Block Diagram

Функционален блок	Обозначение	Предназначение
AND	AND (&)	Логическо „И“ – изходът е активен, когато всички входове са активни
OR	OR (≥1)	Логическо „ИЛИ“ – изходът е активен, когато поне един вход е активен
NOT	NOT	Инвертира логическото състояние на входния сигнал
XOR	XOR	Изключващо „ИЛИ“ – изходът е активен при различни входни състояния
EQ	EQ (=)	Проверява равенството между две стойности
GT	GT (>)	Проверява дали първата стойност е по-голяма от втората
LT	LT (<)	Проверява дали първата стойност е по-малка от втората
ADD	ADD (+)	Събира две или повече стойности
SUB	SUB (−)	Изважда една стойност от друга
MUL	MUL (×)	Умножава две или повече стойности
DIV	DIV (÷)	Деление на две стойности
TON	TON	Закъснение при включване
TOF	TOF	Закъснение при изключване
TP	TP	Генерира импулс с определена продължителност
CTU	CTU	Увеличава стойността на брояча
CTD	CTD	Намалява стойността на брояча

От данните в табл. 9 се вижда, че езикът Function Block Diagram предоставя богат набор от функционални блокове за реализиране на логически, математически и времеви операции. Благодарение на модулния принцип на изграждане алгоритмите могат лесно да бъдат разширявани, модифицирани и повторно използвани, което е едно от основните предимства на FBD при разработването на PLC приложения.

Съществено предимство на Function Block Diagram е възможността за изграждане на програмите чрез многократно използване на предварително разработени функционални блокове. Това позволява създаването на модулни програмни структури, при които отделните блокове могат лесно да бъдат комбинирани, модифицирани и използвани повторно в различни приложения. Подобен подход значително намалява времето за разработване на програмното осигуряване и улеснява неговата последваща поддръжка.

Графичният начин на представяне на алгоритмите улеснява анализа на потока от данни между отделните функционални блокове и позволява бързо локализиране на логически грешки по време на разработването и тестването на PLC приложения. Поради тази причина езикът Function Block Diagram намира широко приложение при реализирането на системи за автоматично регулиране, обработка на аналогови сигнали, управление на технологични процеси и изграждане на сложни алгоритми за индустриална автоматизация.

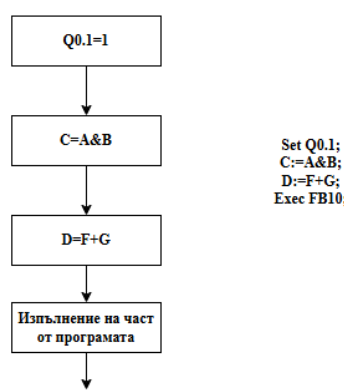
В Arduino PLC IDE езикът Function Block Diagram е напълно интегриран със средата за разработване и може да се използва самостоятелно или в

комбинация с останалите програмни езици, определени в стандарта IEC 61131-3. Това позволява разработването на комплексни приложения, при които различните части на алгоритъма могат да бъдат реализирани чрез най-подходящия програмен език в зависимост от спецификата на конкретната задача.

- Structured Text (ST)

Structured Text (ST) е текстов програмен език, дефиниран в международния стандарт IEC 61131-3, предназначен за разработване на сложни алгоритми за управление, математически изчисления и обработка на данни. За разлика от графичните програмни езици Ladder Diagram и Function Block Diagram, при които алгоритъмът се представя чрез контакти, бобини или функционални блокове, Structured Text използва текстов синтаксис, близък до програмните езици Pascal и C. Това го прави особено подходящ за реализиране на условни конструкции, цикли, изчислителни процедури, обработка на масиви и сложни логически зависимости.

В Arduino PLC IDE езикът Structured Text позволява разработването на програмни модули с висока степен на гъвкавост, четимост и възможност за повторно използване на програмния код. Чрез него могат да бъдат реализирани алгоритми, които трудно биха били представени чрез графичните програмни езици, особено когато се изисква обработка на числови данни, последователни изчисления, работа с голям брой променливи или реализиране на сложни логически зависимости. Поради това ST намира широко приложение при разработването на регулатори, алгоритми за диагностика, системи за обработка на данни и приложения за индустриален мониторинг. За онагледяване на възможностите на езика Structured Text на фиг. 25 е представен пример за програмен фрагмент, разработен в Arduino PLC IDE.

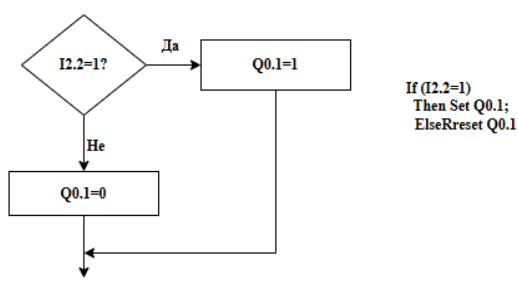


Фиг. 25. Реализиране на безусловни оператори в езика Structured Text (ST).

Както се вижда от фиг. 25, безусловните оператори представляват последователност от програмни инструкции, които се изпълняват без извършване на предварителна логическа проверка. Всяка инструкция се изпълнява в реда, в който е записана в програмата, като резултатът от едно действие може да бъде използван от следващото. В представения пример първоначално се установява логическа единица на изход Q0.1, след което се извършва логическата операция C := A AND B, а накрая се изпълнява аритметичното присвояване D := F + G. След приключване на тази последователност програмата продължава изпълнението си към следващия програмен фрагмент.

Безусловните оператори са основен елемент на езика Structured Text и се използват при реализиране на линейни участъци от алгоритъма, в които не се налага проверка на условия или организиране на циклично изпълнение. Те намират широко приложение при инициализация на променливи, извършване на математически и логически операции, извикване на функции и функционални блокове, както и при обработка на входно-изходни данни. Благодарение на своята простота и последователност тези конструкции осигуряват добра четимост на програмния код и улесняват разработването и поддръжката на PLC приложения.

Освен безусловни оператори, езикът Structured Text предоставя възможност за реализиране на условни конструкции, чрез които изпълнението на определени програмни инструкции зависи от резултата на предварително зададено логическо условие. Пример за такава конструкция е представен на фиг. 26.



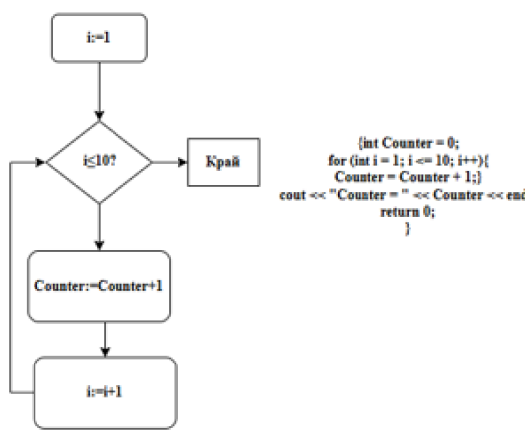
Фиг. 26. Реализиране на условен оператор IF...THEN...ELSE в езика Structured Text (ST)

Както се вижда от фиг. 26, условният оператор IF...THEN...ELSE позволява изпълнението на различни програмни действия в зависимост от резултата на предварително зададено логическо условие. В представения пример се проверява състоянието на входния сигнал I2.2. Ако условието е изпълнено, чрез инструкциите Set Q0.1 се активира изходът Q0.1. При

неизпълнение на условието се изпълнява алтернативната команда Reset Q0.1, с което изходът се деактивира. След изпълнение на съответния клон програмата продължава последователното си изпълнение към следващата програмна инструкция.

Условните оператори са едни от най-често използваните програмни конструкции в езика Structured Text, тъй като позволяват реализирането на логически зависимости между входните сигнали и управляваните изходи. Те намират широко приложение при разработването на алгоритми за управление на технологични процеси, обработка на аварийни състояния, реализиране на защитни функции и управление на изпълнителни механизми. Използването на конструкцията IF...THEN...ELSE осигурява ясно структуриране на програмния код и улеснява неговото разработване, анализ и последваща поддръжка.

Освен последователни и условни програмни конструкции, езикът Structured Text предоставя възможност за организиране на циклично изпълнение на програмните инструкции. Циклите позволяват многократно изпълнение на определена последователност от команди до изпълнение на зададено условие или предварително определен брой повторения. Пример за реализиране на циклична конструкция е представен на фиг. 27.



Фиг. 27. Реализиране на цикъл FOR в езика Structured Text (ST)

Както се вижда от фиг. 27, цикълът FOR започва с инициализиране на брояча, след което се проверява условието за неговата крайна стойност. Докато условието е изпълнено, се изпълняват инструкциите от тялото на цикъла, след което стойността на брояча се увеличава и проверката се извършва отново. При неизпълнение на условието изпълнението на цикъла се прекратява и програмата продължава със следващата инструкция. Поради

предварително определения брой повторения операторът FOR се използва широко при обработка на масиви, последователни изчисления и многократно изпълнение на еднотипни операции.

Цикълът FOR намира широко приложение при обработка на масиви, изпълнение на многократни изчисления, генериране на последователности и реализиране на алгоритми, при които броят на повторенията е известен предварително. Благодарение на компактния си синтаксис и автоматичното управление на брояча той улеснява разработването на четим и ефективен програмен код в езика Structured Text.

Разгледаните програмни конструкции показват, че езикът Structured Text предоставя широки възможности за разработване на алгоритми с различна степен на сложност. Чрез използването на безусловни оператори, условни конструкции и цикли могат да бъдат реализирани както прости логически зависимости, така и сложни алгоритми за управление и обработка на данни. Поради своята гъвкавост, добра четимост и близък до класическите програмни езици синтаксис, Structured Text е един от най-предпочитаните езици за разработване на PLC приложения съгласно стандарта IEC 61131-3.

Изводи от глава 2

В настоящата глава е извършен анализ на основните хардуерни и софтуерни технологии с отворена архитектура, приложими при изграждането на компютърни системи за управление. Разгледани са принципите на Open Source Hardware и Open Source Software, техните основни характеристики и възможностите, които предоставят за разработване, модифициране и интегриране на хардуерни и програмни компоненти. Анализът показва, че използването на отворени технологии създава предпоставки за изграждане на адаптивни и разширяеми системи, при които изборът на конкретна платформа следва да бъде съобразен с техническите, функционалните, лицензионните и експлоатационните изисквания на съответното приложение.

Особено внимание е отделено на Arduino екосистемата и на възможностите за нейното използване при разработването на системи за измерване, мониторинг и управление. Разгледани са особеностите на микроконтролерните платформи Arduino и развитието на екосистемата към индустриално ориентирани управляващи решения. Анализирани са архитектурата, техническите характеристики, входно-изходните и комуникационните възможности на програмируемия контролер Arduino Opta, които създават предпоставки за използването му при разработване на системи за автоматизация и IIoT приложения.

Анализът на програмните средства показва различията в предназначението и функционалните възможности на софтуерните среди и платформи, приложими при разработването на компютърни системи за управление. Специално внимание е отделено на Arduino IDE и Arduino PLC IDE като средства,

използвани при разработването и програмирането на приложения за разглежданите в монографията хардуерни платформи Arduino Uno и Arduino Opta. Arduino IDE предоставя средства за разработване на микроконтролерно базирани приложения, докато Arduino PLC IDE разширява възможностите за създаване на PLC ориентирани системи за управление чрез програмни средства, свързани със стандарта IEC 61131-3.

Разгледаните програмни езици Ladder Diagram (LD), Function Block Diagram (FBD) и Structured Text (ST) предоставят различни подходи за реализиране на алгоритми за управление. LD е подходящ за представяне на логически зависимости и управление на дискретни сигнали, FBD позволява изграждане на алгоритми чрез взаимосвързани функционални блокове, а ST предоставя възможности за реализация на по-сложни изчислителни процедури, обработка на данни и структурирани алгоритми. Изборът на конкретен програмен език или тяхното комбинирано използване се определя от характера и сложността на решаваната задача за управление.

Представените примери, структурни схеми и програмни реализации систематизират основните подходи при разработването на приложения за разглежданите хардуерни и софтуерни платформи. Анализът показва, че съчетаването на микроконтролерни и PLC ориентирани решения с подходящи програмни средства предоставя възможности за изграждане на компютърни системи с различна функционалност и сложност – от лабораторни и експериментални модели до индустриално ориентирани приложения.

Направеният анализ формира теоретичната и технологичната основа за последващото разглеждане на архитектурите, комуникационните възможности и практическата реализация на компютърни системи за управление с използване на отворени технологии. На тази основа в **Глава 3** се преминава към анализ на архитектурните принципи и структурното изграждане на компютърните системи за управление, като се разглеждат взаимодействието между отделните функционални нива и възможностите за интегриране на хардуерни и програмни компоненти.

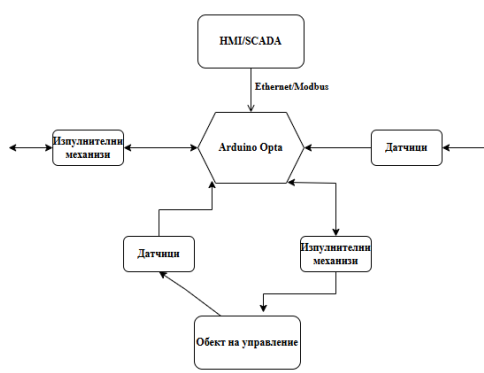
Глава 3. Архитектура на разработената компютърна система за управление.

Разработването на съвременни компютърни системи за управление изисква интегриране на хардуерни и програмни средства, осигуряващи управление, мониторинг и обмен на информация между отделните компоненти на системата. При проектирането на такива системи особено значение имат модулната структура, комуникационните възможности, експлоатационните характеристики и възможността за последващо разширяване.

Въз основа на анализа, представен в предходната глава, за изграждане на разработената компютърна система е избран програмируемият логически контролер Arduino Opta, който съчетава PLC ориентирана функционалност с възможностите на Arduino екосистемата и програмни средства, свързани със стандарта IEC 61131-3. Комуникационните възможности на контролера създават предпоставки за интегрирането му с други компоненти на системата, включително средства за операторска визуализация и системи за мониторинг и управление.

Настоящата глава е посветена на разработването на общата архитектура на компютърната система за управление. Представени са основните функционални модули, използваните хардуерни компоненти, комуникационните връзки между тях и принципът на функциониране на системата. Разгледани са също организацията на програмното осигуряване и взаимодействието между отделните модули.

Въз основа на поставените изисквания е разработена модулна архитектура, осигуряваща ясно разделяне между измервателната част, управляващия контролер, изпълнителните механизми и средствата за операторско наблюдение. Общата структура на разработената система е представена на Фиг. 28.



Фиг. 28. Архитектура на разработената компютърна система за управление, базирана на Arduino Opta

Както се вижда от фигурата, централният елемент на разработената система е програмируемият логически контролер Arduino Opta. Към неговите входове се свързват измервателните датчици и командните устройства, а към изходите – изпълнителните механизми на управлявания технологичен процес. Контролерът осъществява непрекъснато събиране и обработка на входната информация, изпълнение на алгоритмите за управление и формиране на управляващи сигнали към изпълнителните устройства.

Комуникационните възможности на използвания контролер позволяват интегрирането му в мрежово свързани системи за автоматизация. Ethernet свързаността и поддръжката на подходящи комуникационни протоколи, включително Modbus TCP при съответната програмна реализация, създават възможности за обмен на данни със средства за операторска визуализация, мониторинг и управление, включително HMI и SCADA системи.

Архитектурата е разработена на модулен принцип с възможност за разширяване. Към нея могат да бъдат включвани различни групи входни датчици и изпълнителни механизми в зависимост от конкретното приложение, без да се изменя основният принцип на структурното ѝ изграждане. Това позволява архитектурата да бъде адаптирана както към лабораторни и експериментални системи, така и към различни задачи за автоматизация на технологични процеси.

Модулният принцип на изграждане улеснява разработването, тестването и последващото разширяване на системата. Отделните функционални модули могат да бъдат разработвани, конфигурирани и усъвършенствани относително независимо, което улеснява диагностиката, поддръжката и адаптирането на архитектурата към различни приложения.

3.1. Функционални модули на разработената компютърна система.

Разработената компютърна система за управление е изградена на модулен принцип, при който всеки функционален модул изпълнява ясно определени задачи и взаимодейства с останалите модули посредством стандартизирани комуникационни връзки. Подобен подход осигурява висока надеждност, лесна поддръжка и възможност за последващо разширяване на системата чрез включване на нови измервателни средства, изпълнителни механизми и програмни модули.

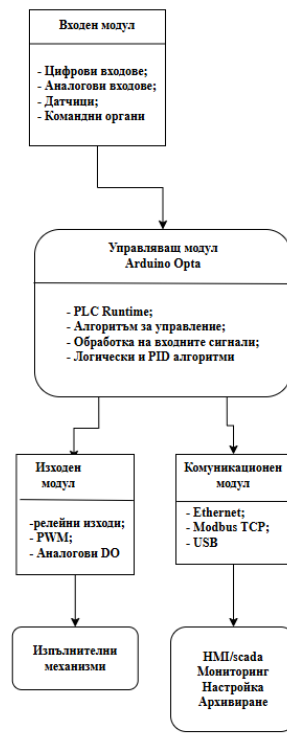
Функционалното разделяне на системата позволява независима разработка, изпитване и модернизация на отделните модули, без да се променя общата архитектура на системата. Това е едно от основните предимства на

модулния подход при изграждането на съвременни компютърни системи за управление.

В разработената система могат да бъдат обособени следните основни функционални модули:

- входен модул;
- управляващ модул;
- изходен модул;
- комуникационен модул;
- модул за визуализация и мониторинг.

Функционалните връзки между отделните модули са представени на фиг. 29.



Фиг. 29. Функционални модули на разработената компютърна система за управление

Както се вижда от фиг. 29, разработената компютърна система за управление е организирана в няколко взаимосвързани функционални модула, всеки от които изпълнява специфични задачи в процеса на автоматизираното управление. Входният модул осигурява събирането на информация от измервателните преобразуватели и командните органи, управляващият модул реализира алгоритмите за обработка на данните и формиране на управляващите въздействия, а изходният модул управлява изпълнителните механизми на технологичния процес. Комуникационният модул осигурява обмена на информация със системите за визуализация, мониторинг и архивиране на данни посредством стандартни индустриални комуникационни протоколи.

Разделянето на системата на функционални модули позволява лесно адаптиране на разработената архитектура към различни технологични процеси чрез промяна на периферните устройства, без да се изменят основните принципи на работа на управляващата система. Разделянето на системата на функционални модули създава възможности за адаптиране на разработената архитектура към различни технологични процеси чрез промяна или допълване на периферните устройства и съответното програмно осигуряване, без да се изменят основните принципи на структурното ѝ изграждане.

Входният модул осигурява приемането на информация от технологичния процес чрез цифрови и аналогови входни сигнали, формирани от измервателни преобразуватели, крайни изключватели, командни бутони и други входни устройства. Измервателните преобразуватели преобразуват наблюдаваните физични величини в сигнали, подходящи за последваща обработка от управляващата система. В зависимост от конкретното приложение могат да бъдат използвани температурни, димни, газови, индуктивни, оптични, Hall датчици и други измервателни средства.

Управляващият модул представлява основният елемент на разработената компютърна система. Неговите функции се реализират от програмируемия логически контролер Arduino Opta, който изпълнява алгоритмите за управление, обработва входните сигнали и формира управляващите въздействия към изпълнителните механизми. Използването на PLC ориентирани програмни средства, свързани със стандарта IEC 61131-3, позволява структурирано разработване и модифициране на алгоритмите за управление в зависимост от изискванията на конкретното приложение.

Изходният модул реализира връзката между управляващия контролер и изпълнителните механизми на технологичния процес. Чрез

подходящи изходни и интерфейсни устройства могат да бъдат управлявани релета, електромагнитни клапани, светлинна и звукова сигнализация, електрозадвижвания и други изпълнителни механизми. В зависимост от конкретното приложение могат да се използват различни типове управляващи сигнали и допълнителни интерфейсни или силови модули, съобразени с характеристиките на съответните изпълнителни устройства.

Комуникационният модул осигурява обмена на информация между управляващия контролер и външни системи за визуализация, мониторинг и управление. Използването на Ethernet свързаност и подходящи комуникационни протоколи, например Modbus TCP при съответната програмна реализация, позволява интегрирането на разработената архитектура с други нива и компоненти на системата за автоматизация.

Модулът за визуализация и мониторинг предоставя възможност за наблюдение на технологичния процес в реално време, визуализиране на измерените параметри, архивиране на данни, генериране на аварийни съобщения и настройка на работните режими. Модулът за визуализация и мониторинг предоставя възможности за наблюдение на технологичния процес, визуализиране на измерените параметри, архивиране на данни, генериране на съобщения за състояния и аварии и, когато е предвидено, настройване на работни параметри. Реализацията му чрез HMI или SCADA система осигурява операторски интерфейс за наблюдение и взаимодействие с управлявания процес.

Описаните функционални модули формират структура, при която функциите по получаване на входна информация, обработка и управление, формиране на изходни въздействия, комуникация и операторска визуализация са ясно разграничени. Модулният принцип създава възможности за адаптиране и разширяване на архитектурата в зависимост от особеностите на конкретния технологичен процес.

След определяне на функционалната структура на разработената компютърна система е необходимо да бъдат разгледани използваните хардуерни средства, чрез които се реализират отделните функционални модули.

3.2. Хардуерна реализация на разработената компютърна система за управление

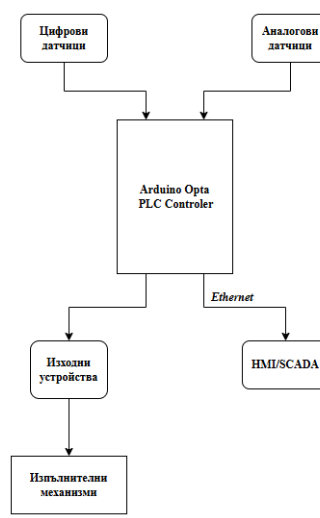
След определяне на функционалната структура на разработената компютърна система за управление е извършен избор на хардуерните средства, необходими за реализиране на отделните функционални модули. При разработването на системата са поставени изисквания за модулност,

възможност за разширяване и съвместимост с използваните комуникационни технологии. Особено внимание е отделено на използването на хардуерни платформи с отворена архитектура и възможности за интегриране на различни периферни устройства и програмни средства.

Хардуерната архитектура на разработената компютърна система за управление е изградена така, че да осигурява събиране и обработка на информация от технологичния процес, изпълнение на алгоритмите за управление и формиране на управляващи въздействия към изпълнителните механизми. За реализиране на тези функции системата включва централен управляващ контролер, входни устройства, изходни устройства, комуникационни средства и операторски интерфейс.

В съответствие с поставените изисквания като централен управляващ модул е избран програмируемият логически контролер Arduino Opta, който съчетава PLC ориентирана функционалност с възможностите на Arduino екосистемата. Контролерът разполага с хардуерни и комуникационни ресурси, позволяващи свързване на входни устройства, управление на изпълнителни механизми и обмен на данни с други компоненти на системата.

Общата хардуерна структура на разработената компютърна система за управление е представена на фиг. 30.



Фиг. 30. Хардуерна структура на разработената компютърна система за управление

Както се вижда от фиг. 30, хардуерната реализация на разработената компютърна система е организирана около програмируемия логически контролер Arduino Opta, който изпълнява функциите на централен управляващ модул. Към неговите входове се свързват измервателните преобразуватели и командните устройства, а към изходите – изпълнителните механизми на управлявания технологичен процес. Чрез Ethernet интерфейса може да бъде реализиран обмен на информация с външни системи за визуализация, мониторинг и управление в зависимост от използваната комуникационна и програмна конфигурация.

Модулната организация на хардуерната архитектура позволява нейното адаптиране към конкретното приложение чрез избор и конфигуриране на подходящи периферни устройства и интерфейсни модули, като се запазва основният принцип на структурното изграждане на системата.

Централният управляващ модул е реализиран чрез програмируемия логически контролер Arduino Opta. Той осъществява обработката на входната информация, изпълнението на програмния алгоритъм и формирането на управляващи сигнали към свързаните изходни и изпълнителни устройства. Наличните комуникационни възможности позволяват интегрирането му с други компоненти на системата за управление.

Входните устройства осигуряват получаването на информация за състоянието на технологичния процес и нейното подаване към управляващия контролер. В зависимост от конкретното приложение могат да бъдат използвани цифрови и аналогови датчици, крайни изключватели, командни бутони, енкодери, Hall датчици, температурни, димни, газови и други измервателни преобразуватели. Използването на модулна архитектура позволява лесна замяна или добавяне на нови входни устройства без промяна в основната структура на системата.

Изходните устройства осъществяват връзката между управляващия контролер и изпълнителните механизми на технологичния процес. Основната им функция е да преобразуват управляващите сигнали, формирани от програмируемия логически контролер, в управляващи въздействия към съответните изпълнителни устройства. По този начин се реализира непосредственото въздействие върху управлявания обект в съответствие с алгоритъма на управление.

В зависимост от конкретното приложение изходната част на системата може да включва релейни, цифрови или аналогови изходни модули, както и допълнителни драйверни и силови интерфейси. Чрез тях могат да

бъдат управлявани електрозадвижвания, електромагнитни клапани, контактори, светлинна и звукова сигнализация и други изпълнителни устройства.

Разработената компютърна система използва вградените релейни изходи на програмируемия логически контролер Arduino Opta, които могат да бъдат използвани за комутационно управление на подходящи външни товари при спазване на допустимите им електрически характеристики.

Комуникационните средства осигуряват обмена на информация между разработената компютърна система за управление и външните устройства за наблюдение, управление и диагностика. Чрез тях се реализира предаването на технологични данни, настройката на параметрите на системата, визуализацията на текущото състояние на процеса и архивирането на измерената информация.

В разработената архитектура Ethernet е основен мрежов комуникационен интерфейс, чрез който контролерът може да бъде свързан към локална мрежова инфраструктура и към външни системи за визуализация и управление. При използване на подходяща програмна реализация комуникационни протоколи като Modbus TCP могат да бъдат използвани за обмен на технологични данни с HMI, SCADA или други мрежово свързани устройства.

Освен Ethernet комуникация, системата разполага и с USB интерфейс, който се използва при разработването, програмирането, настройката и диагностиката на програмируемия логически контролер. Чрез него могат да се извършват операции, свързани със зареждане на програмното осигуряване, конфигуриране и диагностика по време на разработването и експерименталните изследвания.

Разширяването на комуникационните възможности позволява включване на допълнителни устройства, операторски панели, SCADA системи и други мрежово свързани компоненти. С увеличаването на взаимната свързаност и възможностите за дистанционен достъп нараства и значението на подходящите организационни и технически мерки за защита на индустриалната комуникационна инфраструктура [14].

Средствата за визуализация и мониторинг осигуряват наблюдение на технологичния процес в реално време, визуализиране на работните параметри, архивиране на измерените данни и подпомагане на оператора при управлението на системата. Чрез тях се реализира взаимодействието между оператора и компютърната система за управление и се предоставя информация за текущото състояние на технологичния процес.

В разработената компютърна система средствата за визуализация могат да бъдат реализирани чрез операторски панели (HMI), SCADA системи или специализирани програмни приложения за наблюдение и управление. Те предоставят възможност за графично представяне на текущото състояние на технологичния процес, визуализиране на измерените величини, настройка на технологичните параметри и регистриране на аварийни състояния [15].

Интегрирането на средства за визуализация и мониторинг създава възможности за проследяване на състоянието на системата, регистриране на отклонения и анализ на натрупаната информация. Архивираните технологични данни могат да бъдат използвани за последващ анализ на процеса и оценка на работата на реализираните алгоритми за управление.

Описаните хардуерни компоненти формират материалната основа на разработената архитектура и осигуряват реализирането на основните функции по получаване и обработка на входна информация, управление на изпълнителни устройства, комуникация и операторска визуализация. Конкретният състав и конфигурация на хардуерните средства могат да бъдат адаптирани в зависимост от изискванията на съответното приложение. След определяне на хардуерната архитектура следващият етап е разработването на програмното осигуряване, чрез което се реализират алгоритмите за управление и взаимодействието между отделните функционални модули.

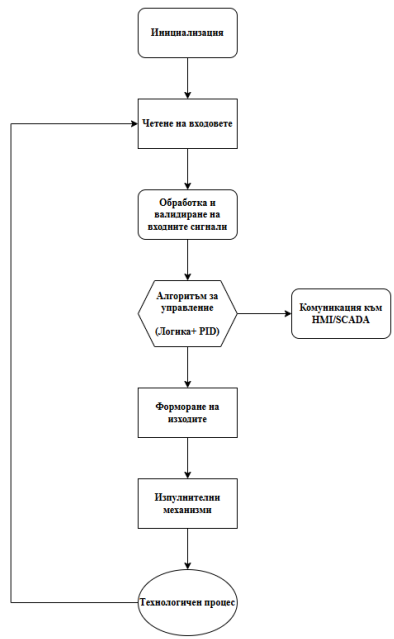
3.3. Програмна реализация на разработената компютърна система за управление.

След изграждането на хардуерната архитектура е разработено програмното осигуряване на компютърната система за управление, което реализира алгоритмите за обработка на входните сигнали, управление на изпълнителните механизми и обмен на информация с външните средства за визуализация и мониторинг. Програмното осигуряване е реализирано в средата Arduino PLC IDE, която предоставя средства за разработване на PLC приложения за Arduino Opta чрез програмни езици и средства, свързани със стандарта IEC 61131-3 [10], [11], [16].

При разработването на програмната реализация е приложен модулен подход, при който отделните функционални задачи са организирани в самостоятелни програмни модули. Това улеснява последващото тестване, диагностика, поддръжка и разширяване на системата и подобрява структурирането и четимостта на програмния код [16]. Основните програмни модули осигуряват обработката на входните сигнали, изпълнението на алгоритмите

за управление, формирането на управляващите въздействия към изпълнителните механизми, както и реализирането на комуникацията със средствата за визуализация и мониторинг.

Структурата на разработеното програмно осигуряване е представена на фиг. 31.



Фиг. 31. Структура на алгоритъма за функциониране на програмното осигуряване на разработената компютърна система за управление

Както се вижда от фиг. 31, програмното осигуряване на разработената компютърна система за управление е организирано като последователен цикличен процес, характерен за програмируемите логически контролери. След стартиране на системата се извършва инициализация на програмните променливи, настройка на комуникационните интерфейси и подготовка на използваните хардуерни ресурси. След приключване на инициализацията програмата преминава към непрекъснат цикъл на изпълнение, при който последователно се извършват четене на входните сигнали, тяхната обработка, изпълнение на алгоритмите за управление и формиране на управляващите въздействия към изпълнителните механизми.

На етапа на обработка на входните сигнали се извършва анализ на информацията, постъпваща от измервателните преобразуватели и командните органи. При необходимост могат да бъдат реализирани филтриране, проверка за валидност на сигналите и преобразуване или мащабиране на аналоговите входни величини в подходящ за обработка формат. Получената информация се използва от алгоритъма за управление, който реализира логическите функции на системата и, при необходимост, алгоритми за автоматично регулиране, включително PID управление.

Въз основа на изчислените управляващи въздействия се формират сигналите към изходните устройства, чрез които се управляват изпълнителните механизми на технологичния процес. При системи с реализирана обратна връзка информацията за състоянието на управлявания обект се регистрира от съответните измервателни средства и отново постъпва към управляващия контролер. При системи с реализирана обратна връзка информацията за състоянието на управлявания обект се регистрира от съответните измервателни средства и отново постъпва към управляващия контролер. По този начин се реализира затворен цикъл на управление, при който управляващото въздействие се коригира в зависимост от текущото състояние на процеса и се осигурява своевременна реакция при промяна на работните условия.

Паралелно с изпълнението на алгоритъма за управление може да се осъществява обмен на информация със средствата за визуализация и мониторинг посредством комуникационния модул. Това позволява визуализиране на текущи технологични параметри, архивиране на данни, настройка на работни режими и диагностика на състоянието на системата в зависимост от конкретната програмна реализация.

Представената структура на програмното осигуряване организира изпълнението на основните функции на разработената компютърна система за управление в последователен цикъл, включващ обработка на входната информация, изпълнение на управляващите алгоритми, формиране на изходни въздействия и обмен на данни. Модулната организация позволява отделните функционални блокове да бъдат разработвани, тествани и модифицирани относително независимо, което улеснява поддръжката, разширяването и адаптирането на системата към различни приложения [16].

След определяне на общата структура на програмното осигуряване следва разработването на отделните програмни модули, реализиращи

логическите функции, алгоритмите за управление, комуникацията и средствата за диагностика на разработената компютърна система за управление.

3.3.1. Инициализация на програмното осигуряване

Инициализацията представлява първия етап от изпълнението на програмното осигуряване на разработената компютърна система за управление. Нейната основна задача е да подготви хардуерните и програмните ресурси за последващото изпълнение на алгоритмите за управление. Този етап се изпълнява еднократно при включване на захранването, рестартиране на програмируемия логически контролер или повторно стартиране на управляващата програма.

По време на инициализацията се задават началните стойности на използваните програмни променливи, конфигурират се входните и изходните канали и се установяват безопасни начални състояния на изпълнителните механизми. Изходните сигнали се привеждат в предварително определено състояние, ограничавайки възможността за непреднамерено активиране на двигатели, клапани, релета и други изпълнителни устройства при стартиране на системата.

В този етап се извършва и настройване на параметрите, използвани от алгоритмите за управление. В зависимост от конкретното приложение могат да бъдат зададени гранични стойности на измерваните величини, времеви закъснения, коефициенти на регулаторите, прагове за задействане на аларми и други технологични параметри. Когато стойностите се съхраняват в енергонезависима памет, при стартиране се извършва тяхното прочитане и проверка за допустимост.

Инициализацията включва също подготовка на комуникационните интерфейси, използвани за обмен на данни със средствата за визуализация и мониторинг. Конфигурират се мрежовите параметри, комуникационните променливи и необходимите регистри за предаване и получаване на информация. При наличие на HMI или SCADA система се установяват началните стойности на визуализираните параметри и диагностичните съобщения.

Преди преминаване към основния програмен цикъл се изпълнява начална самодиагностика. Проверяват се състоянието на входните сигнали, наличието на недопустими комбинации, готовността на комуникационните интерфейси и коректността на основните програмни параметри. При установяване на неизправност или недопустимо начално състояние се изпълняват предвидените защитни и диагностични действия, включително

блокиране на съответните изпълнителни механизми и генериране на диагностично или аварийно съобщение.

След успешно приключване на инициализацията се активира основният цикъл на програмното осигуряване, в който последователно се изпълняват четенето и обработката на входните сигнали, алгоритмите за управление, формирането на изходните въздействия и обменът на информация с външните устройства.

Правилното реализиране на етапа на инициализация е съществено за предвидимото и безопасно стартиране на разработената компютърна система за управление, тъй като осигурява определено начално състояние на програмните и хардуерните ресурси и ограничава възможността за неконтролирани въздействия върху технологичния процес.

3.3.2. Реализация на алгоритъма за управление

Алгоритъмът за управление представлява основният функционален елемент на разработеното програмно осигуряване. Неговата задача е да обработва информацията, постъпваща от входните устройства, да анализира текущото състояние на технологичния процес и въз основа на предварително зададени правила да формира управляващи въздействия към изпълнителните механизми. По този начин се реализира автоматично управление на технологичния процес в съответствие със зададените алгоритми, режими на работа и предвидените условия за безопасност.

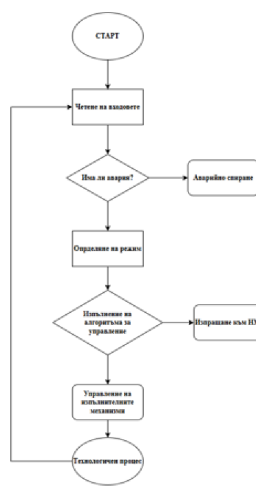
При разработването на алгоритъма е използван модулен подход, при който отделните функционални задачи са реализирани като обособени програмни блокове. Всеки блок изпълнява конкретна функция – обработка на входните сигнали, логическа обработка, управление на изпълнителните механизми, обработка на аварийни състояния или обмен на информация с външни устройства. Подобна организация улеснява тестването, последващото разширяване и повторното използване на програмните модули при различни приложения [16].

Основният алгоритъм се изпълнява циклично и включва последователно изпълнение на логическите операции, необходими за управлението на технологичния процес. След обработването на входните сигнали се извършва оценка на текущото състояние на системата, проверка на предварително зададените условия и избор на съответния режим на работа. В зависимост от резултатите се формират управляващите сигнали към изпълнителните механизми, както и необходимите диагностични и информационни съобщения.

При приложения, включващи непрекъснато регулирани технологични величини, структурата на програмното осигуряване позволява включване на специализирани функционални блокове за автоматично регулиране, включително PID регулатори. По този начин общата програмна архитектура може да бъде адаптирана както за задачи с дискретно логическо управление, така и за приложения, изискващи автоматично регулиране на технологични величини.

Особено внимание при разработването на алгоритъма е отделено на обработката на аварийни и недопустими състояния. При установяване на предварително дефинирано аварийно условие, превишаване на зададени гранични стойности или активиране на защитна функция се изпълняват съответните защитни действия, определени според особеностите на конкретния технологичен процес. Те могат да включват блокиране или изключване на определени изпълнителни механизми, активиране на защитни функции и генериране на диагностични или аварийни съобщения.

За онагледяване на принципа на функциониране на програмното осигуряване е представена блокова схема на алгоритъма за управление. Тя показва последователността на основните програмни операции и логическите връзки между отделните функционални етапи. Блоковата схема е представена на фиг. 32.



Фиг. 32. Блокова схема на алгоритъма за управление на разработената компютърна система

Както се вижда от фиг. 32, изпълнението на алгоритъма започва с прочитане на входните сигнали и определяне на текущото състояние на системата. След обработката на получената информация се извършва проверка за наличие на аварийни или недопустими режими на работа. При установяване на такова състояние се изпълняват предварително зададените защитни действия и се генерира съответната диагностична или аварийна информация. При нормален режим на работа програмата определя съответния режим на управление, изпълнява необходимите логически операции и формира управляващите сигнали към изпълнителните механизми. След изпълнение на съответните операции програмният цикъл се повтаря, като паралелно с това може да се осъществява обмен на информация със средствата за визуализация и мониторинг.

Представеният алгоритмичен подход организира основните функции на системата в циклична и модулна програмна структура. Обработката на входната информация, определянето на текущото състояние, изпълнението на управляващата логика, формирането на изходните въздействия и обработката на аварийни условия са обособени като последователни функционални етапи. Тази организация създава възможности за модифициране и допълване на отделните програмни функции в зависимост от особеностите на конкретното приложение.

Функционирането на алгоритъма за управление е свързано с обмена на информация между отделните функционални модули и външните системи. Поради това в следващата подточка е разгледана реализацията на комуникационния модул и организацията на обмена на данни в разработената компютърна система за управление.

3.3.3. Реализация на комуникационния модул

Комуникационният модул представлява програмен компонент на разработената компютърна система за управление, който осигурява обмен на информация между програмируемия логически контролер и външните средства за визуализация, мониторинг и диагностика. Основната му задача е да организира предаването и приемането на технологични данни, необходими за наблюдение на състоянието на системата, настройка на работните параметри и архивиране на измерваната информация.

Програмната реализация на комуникационния модул е организирана на модулен принцип, като функциите за обмен на данни са логически обособени от основните функции на алгоритъма за управление. Подобна организация улеснява разработването, конфигурирането и последващото

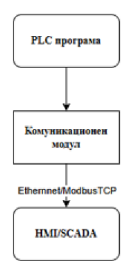
разширяване на комуникационните възможности в зависимост от използваните интерфейси, протоколи и външни програмни приложения [16].

Получената информация може да бъде предоставяна на средствата за визуализация и мониторинг с цел наблюдение на работния процес, регистриране на състояния и събития и последващ анализ на функционирането на системата.

При необходимост комуникационният модул осигурява приемане на управляващи команди и технологични настройки от операторски интерфейс или SCADA система. Получените команди преминават през проверка за допустимост и едва след това се използват от алгоритъма за управление. При необходимост комуникационният модул може да осигурява приемане на управляващи команди и технологични настройки от операторски интерфейс или SCADA система. Прилагането на предварително зададени проверки за допустимост на получените команди позволява ограничаване на изпълнението на некоректни или несъвместими управляващи въздействия.

Модулната организация на комуникационното осигуряване създава възможности за последващо разширяване чрез включване на допълнителни комуникационни протоколи, мрежови услуги или средства за дистанционен мониторинг. Необходимите промени могат да бъдат реализирани основно в комуникационната част на програмното осигуряване, като се запазва общият принцип на организация на управляващия алгоритъм.

За онагледяване на организацията на обмена на информация е разработена принципна схема на комуникационния модул, представена на фиг. 33. Тя показва взаимодействието между програмируемия логически контролер, комуникационния модул и средствата за визуализация и мониторинг, както и основните информационни потоци между тях.



Фиг. 33. Организация на обмена на информация в разработената компютърна система за управление

Както се вижда от фиг. 33, комуникационният модул осъществява двупосочен обмен на информация между програмируемия логически контролер и средствата за визуализация и мониторинг. От една страна, към операторския интерфейс се предават текущите стойности на технологичните параметри, състоянието на входните и изходните сигнали, диагностичната информация и аварийните съобщения. От друга страна, към управляващия контролер могат да бъдат подавани команди за управление, настройки на технологичните параметри и конфигурационни данни, които след проверка за допустимост се използват от алгоритъма за управление.

Представената организация на комуникационния модул осигурява структурирано взаимодействие между управляващия контролер и външните средства за визуализация, мониторинг и управление. Модулното обособяване на комуникационните функции създава възможности за адаптиране и разширяване на обмена на данни в зависимост от комуникационните изисквания на конкретното приложение.

Наред с организацията на комуникационния обмен, съществено значение за функционирането на компютърната система за управление имат средствата за наблюдение на нейното състояние и регистриране на възникнали отклонения и неизправности. Поради това в следващата подточка са разгледани средствата за диагностика и мониторинг, предвидени в програмното осигуряване на системата.

3.3.4. Реализация на функциите за диагностика и мониторинг

Функциите за диагностика и мониторинг са съществен елемент от програмното осигуряване на компютърните системи за управление, тъй като предоставят информация за състоянието на технологичния процес, входните и изходните сигнали, комуникационния обмен и възникналите аварийни или недопустими състояния. В разработената програмна архитектура тези функции са обособени като част от общата структура на програмното осигуряване и могат да бъдат конфигурирани в зависимост от особеностите на конкретното приложение. Получената диагностична информация може да бъде използвана от алгоритъма за управление и предоставяна на средствата за визуализация и мониторинг с цел подпомагане на оператора при наблюдението и диагностиката на системата.

Основните групи параметри и състояния, които могат да бъдат обхванати от функциите за диагностика и мониторинг, са представени в Таблица 10. Тяхното конкретно използване и обработка се определят от особеностите и изискванията на съответното приложение. Всеки от наблюдаваните

параметри изпълнява конкретна функция, свързана с осигуряване на надеждна и безопасна работа на разработената компютърна система за управление.

Таблица 10. Основни параметри, наблюдавани от функциите за диагностика и мониторинг

Наблюдаван параметър	Предназначение
Състояние на цифровите входове	Контрол на сигналите от командни органи, крайни изключватели и цифрови датчици.
Аналогови входове	Наблюдение на измерваните технологични величини и контрол за превишаване на допустимите граници.
Състояние на изходните сигнали	Наблюдение на формираните управляващи команди към изпълнителните устройства.
Комуникация	Контрол на обмена на данни между PLC, HMI и SCADA системите и установяване на комуникационни неизправности.
Аварийни и диагностични състояния	Регистриране на аварийни състояния, блокировки и защитни режими на работа.
Технологични параметри	Наблюдение, архивиране и визуализация на текущите параметри на технологичния процес.

Състояние на цифровите входове - Функцията за наблюдение на цифровите входове осигурява информация за състоянието на командните бутони, крайните изключватели, аварийните бутони и другите свързани цифрови устройства. Промените в състоянието на входните сигнали се обработват от програмния алгоритъм и в зависимост от зададената логика могат да предизвикат съответни управляващи, диагностични или защитни действия.

Аналогови входове – наблюдението и обработката на аналоговите входни сигнали предоставят информация за измервани технологични величини, като температура, налягане, ниво, скорост и други параметри. При наличие на зададени гранични стойности получените данни могат да бъдат сравнявани с тях с цел установяване на отклонения и изпълнение на предварително определени диагностични или управляващи действия.

Състояние на изходните сигнали – наблюдението на изходните сигнали предоставя информация за управляващите команди, формирани от програмируемия логически контролер към свързаните изпълнителни устройства. Самото състояние на изходния сигнал не представлява потвърждение за действителното състояние на изпълнителния механизъм. Когато приложението изисква такова потвърждение, е необходимо използването на

подходящи сигнали за обратна връзка от управляваното устройство или технологичния процес.

Комуникация – функциите за мониторинг на комуникацията могат да бъдат използвани за наблюдение на обмена на информация между управляващия контролер и външните средства за визуализация, мониторинг и управление. При реализирани механизми за установяване на комуникационни прекъсвания или грешки могат да се генерират съответни диагностични състояния и съобщения.

Аварийни и диагностични състояния – програмното осигуряване може да обработва предварително дефинирани аварийни условия, превишаване на зададени гранични стойности и сигнали от защитни устройства. В зависимост от конкретното приложение при установяване на такова състояние могат да бъдат изпълнявани съответни защитни действия, регистриране на събитието и предоставяне на информация към средствата за визуализация и мониторинг.

Технологични параметри – наблюдението на технологичните параметри предоставя възможности за визуализиране, регистриране и, когато е предвидено, архивиране на информация за състоянието на технологичния процес. Натрупаните данни могат да бъдат използвани за последващ анализ на процеса, диагностични цели и оценка на работата на реализираните алгоритми за управление.

Представените функции за диагностика и мониторинг допълват основния алгоритъм за управление чрез обработване и предоставяне на информация за входните и изходните сигнали, комуникационния обмен, технологичните параметри и възникналите диагностични или аварийни състояния. Модулното им организиране позволява конкретният набор от наблюдавани параметри, диагностични условия и реакции да бъде адаптиран в зависимост от особеностите и изискванията на съответното приложение.

Изводи по глава 3

В настоящата глава е разработена модулна архитектура на компютърна система за управление, базирана на програмируемия логически контролер Arduino Opta. В архитектурата са обособени основните функционални компоненти – входен, управляващ, изходен и комуникационен модул, както и средства за визуализация и мониторинг. Приетият модулен принцип създава възможности за конфигуриране и разширяване на системата в зависимост от особеностите на конкретния технологичен процес.

Разработена е обща хардуерна структура, определяща взаимодействието между входните устройства и измервателните преобразуватели, управляващия контролер, изходните и изпълнителните устройства и средствата за комуникация и операторска визуализация. Модулната организация позволява периферните компоненти и интерфейсните средства да бъдат подбирани и конфигурирани в съответствие с изискванията на конкретното приложение при запазване на основния принцип на структурното изграждане.

Разработена е структура на програмното осигуряване, включваща функции за инициализация, обработка на входната информация, изпълнение на управляващи алгоритми, формиране на изходни въздействия, комуникация, диагностика и мониторинг. Модулното обособяване на отделните програмни функции улеснява тяхното разработване, тестване, модифициране и адаптиране към различни задачи за автоматизация.

Предложена е организация на комуникационния обмен между управляващия контролер и външните средства за визуализация, мониторинг и управление. Разгледани са и функциите за диагностика и мониторинг, чрез които могат да се обработват данни за входните и изходните сигнали, технологичните параметри, комуникационния обмен и възникналите диагностични или аварийни състояния.

Разработената в настоящата глава архитектура формира обща методическа и техническа основа за реализиране на компютърни системи за управление с използване на разгледаните хардуерни и програмни технологии. В следващите глави възможностите за практическо приложение на разглеждания подход са изследвани чрез конкретни системи и експериментални разработки, при които архитектурните и програмните решения се адаптират в зависимост от особеностите на съответния технологичен обект.

Глава 4. Реализация и сравнителен анализ на компютърна система за управление на лабораторен модел на гумено-транспортна лента

4.1. Лабораторен модел на гумено-транспортна лента

След разработването на модулната архитектура на компютърната система за управление, представена в глава 3, следващият етап от изследването е нейното практическо приложение върху лабораторен обект. За тази цел е избран лабораторен модел на гумено-транспортна лента, който позволява експериментално изследване на алгоритми за управление, регулиране на електрозадвижването и сравнителна оценка на възможностите на различни хардуерни платформи при съпоставими условия на работа.

В предходно изследване е разработена компютърна система за управление на лабораторен модел на гумено-транспортна лента, реализирана чрез програмируемия логически контролер Arduino Opta. В системата са реализирани функции за управление на електрозадвижването, обработка на сигналите от измервателните преобразуватели, мониторинг на технологичния процес и аварийно спиране на транспортната система [17].

Лентовите транспортьори са сред широко използваните средства за непрекъснат транспорт на насипни и късови материали в минната, металургичната и преработвателната промишленост. Поради тяхното широко приложение разработването и усъвършенстването на системи за управление, диагностика, мониторинг и регулиране на електрозадвижването представлява актуално направление на научни и приложни изследвания [18]–[24], [26], [28], [29], [31].

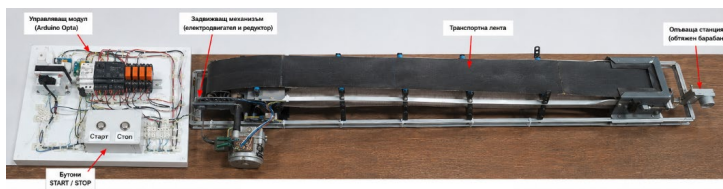
Наред с индустриалните програмируеми логически контролери през последните години все по-широко приложение намират нискобюджетните микроконтролерни платформи, използвани както в учебния процес, така и при разработването на лабораторни модели. Това поставя въпроса за възможностите и ограниченията на подобни платформи в сравнение с PLC ориентирани решения при реализиране на конкретни задачи за управление.

В настоящата глава разработената в глава 3 универсална архитектура на компютърна система за управление е приложена при управление на лабораторен модел на гумено-транспортна лента. С цел обективна оценка на възможностите на различните хардуерни платформи е извършен сравнителен анализ между реализацията с Arduino Uno и предходно разработената система с PLC Arduino Opta[32]. Сравнението е извършено върху един и същ лабораторен обект при съпоставими входни условия и функционални изисквания към управлението, като са анализирани особеностите на реализацията с Arduino Uno и Arduino Opta.

В главата са представени конструкцията на лабораторния модел, реализацията на системата за управление, сравнителният анализ на използваните хардуерни платформи, разработеният алгоритъм за управление и резултатите от проведените експериментални изследвания. На базата на получените резултати е извършена сравнителна оценка на възможностите и ограниченията на Arduino Uno и Arduino Opta при реализиране на разглежданата задача за управление.

За експерименталното изследване е използван лабораторен модел на гумено-транспортна лента, върху който са реализирани и изследвани

различни варианти на компютърна система за управление. Лабораторният модел възпроизвежда основните функционални възли на лентова транспортна система и позволява експериментално изследване на алгоритми за управление, регулиране на електрозадвижването и анализ на работните режими. Общият изглед на лабораторния модел е представен на фиг. 34.



Фиг. 34. Лабораторен модел на гумено-транспортна лента и основни функционални възли

Както се вижда от фиг. 34, лабораторният модел включва управляващ модул, задвижващ механизъм с постояннотоков електродвигател и редуктор, транспортна лента, опъваща станция и командни органи за управление. Конструкцията на стенда е разработена така, че да осигурява възможност за експериментално изследване на алгоритми за управление, мониторинг на технологичните параметри и сравнение на различни хардуерни платформи при съпоставими експериментални условия.

Лабораторният модел включва следните основни функционални възли:

• управляващ модул – Управляващият модул представлява централното звено на компютърната система за управление и изпълнява функциите по събиране и обработка на информацията, постъпваща от входните устройства, както и по формиране на управляващи въздействия към изпълнителните механизми. За целите на сравнителното изследване управляващият модул е реализиран в два варианта – чрез микроконтролерната платформа Arduino Uno и чрез програмируемия логически контролер Arduino Opta. Това позволява възможностите на двете платформи да бъдат оценени при управление на един и същ лабораторен обект.

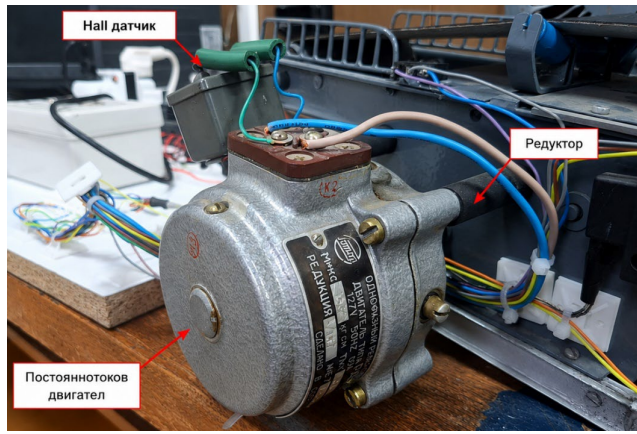
Програмируемият логически контролер изпълнява разработеното програмно осигуряване, представено в глава 3, като реализира логическите алгоритми за управление, обработката на входните сигнали, функциите за автоматично регулиране, диагностиката и комуникацията със средствата за визуализация и мониторинг. Благодарение на поддръжката на стандарта IEC

61131-3 и възможността за програмиране чрез Arduino PLC IDE, управляващият модул позволява изграждането на надеждна и модулна система за управление, която може лесно да бъде адаптирана към различни технологични процеси [10], [11], [16].

Освен изпълнението на основния алгоритъм за управление, управляващият модул осигурява непрекъснат контрол върху състоянието на входните и изходните сигнали, обработва информацията от измервателните преобразуватели и формира управляващите сигнали към изпълнителните механизми. При възникване на аварийна ситуация контролерът изпълнява предварително зададени защитни алгоритми, които привеждат лабораторния модел в безопасно състояние и ограничават възможността за повреда на оборудването.

Благодарение на модулната организация на хардуера и програмното осигуряване управляващият модул представлява универсална платформа, която може да бъде използвана както при лабораторни експериментални постановки, така и при изграждането на реални компютърни системи за управление в промишлени условия.

• задвижващ механизъм - Задвижващият механизъм осигурява движението на транспортната лента и представлява основният изпълнителен възел на лабораторния модел. Той е реализиран чрез постояннотоков електродвигател, свързан с редуктор, който намалява оборотите и увеличава въртящия момент, необходим за задвижване на транспортната лента. Използването на редуктор осигурява плавно движение на системата и създава условия за експериментално изследване на различни режими на управление и регулиране на скоростта. Конструктивното изпълнение на задвижващия механизъм е представено на фиг. 35.



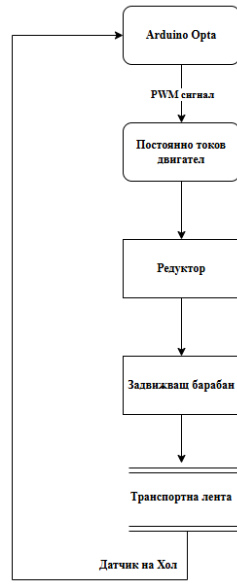
Фиг. 35. Задвижващ механизъм на лабораторния модел на гумено-транспортна лента

Постояннотоковият двигател представлява основният изпълнителен елемент на задвижващия механизъм и осигурява механичната енергия, необходима за задвижване на транспортната лента. Изборът на този тип електродвигател е обусловен от възможността за лесно регулиране на скоростта, което го прави особено подходящ за лабораторни изследвания и разработване на алгоритми за автоматично управление.

За осигуряване на необходимия въртящ момент към изходния вал на двигателя е монтиран механичен редуктор. Чрез него се намалява честотата на въртене и се увеличава въртящият момент, което осигурява стабилна работа на лабораторния модел при различни режими на натоварване.

За измерване на скоростта на въртене е използван Hall датчик, който формира импулсен сигнал, пропорционален на честотата на въртене на двигателя. Получената информация служи за определяне на действителната скорост и се използва при реализиране на обратната връзка в системата за управление.

Принципът на работа на задвижващия механизъм и взаимодействието между основните му елементи са представени на фиг. 36.



Фиг. 36. Функционална структура на задвижващия механизъм

Скоростта на въртене на постояннотоковия електродвигател се регулира чрез широчинно-импулсна модулация (PWM). Чрез промяна на коефициента на запълване на управляващия PWM сигнал се изменя режимът на работа на задвижването, което позволява регулиране на скоростта на транспортната лента. Конкретната хардуерна реализация на управлението при Arduino Uno и Arduino Opta е разгледана в следващия раздел.

Както се вижда от фиг. 36, управляващият сигнал се формира от системата за управление и чрез съответния интерфейсен или силов модул управлява постояннотоковия електродвигател. Генерираният въртящ момент се предава чрез редуктора към задвижващия барабан, осигуряващ движението на транспортната лента. Hall датчикът предоставя информация за скоростта на въртене, която може да бъде използвана като сигнал за обратна връзка при реализиране на затворена система за автоматично регулиране на скоростта.

Задвижващият механизъм е проектиран така, че да позволява работа при различни режими на натоварване и експериментално изследване на динамичните характеристики на системата. Това създава възможност за анализ

на преходните процеси, настройка на параметрите на регулатора и сравнение на различни алгоритми за управление при еднакви експериментални условия.

• транспортна лента - транспортна лента – Транспортната лента, показана на фиг. 34, представлява основния технологичен обект на управление в разработения лабораторен модел. Нейното предназначение е да възпроизвежда принципа на работа на лентовите транспортни системи използвани за непрекъснато транспортиране на насипни и късови материали в различни отрасли на промишлеността. Благодарение на конструкцията на лабораторния стенд могат да бъдат реализирани различни режими на работа, което позволява експериментално изследване на алгоритмите за управление и регулиране на електрозадвижването [18], [20], [21], [26].

Движението на транспортната лента се осъществява посредством задвижващия барабан, който предава въртящия момент от електродвигателя чрез редуктора. От противоположната страна е разположена опъваща станция, която осигурява необходимото предварително опъване на лентата и стабилното ѝ движение по време на работа. По този начин се намалява вероятността от приплъзване и се осигуряват постоянни условия при провеждането на експерименталните изследвания.

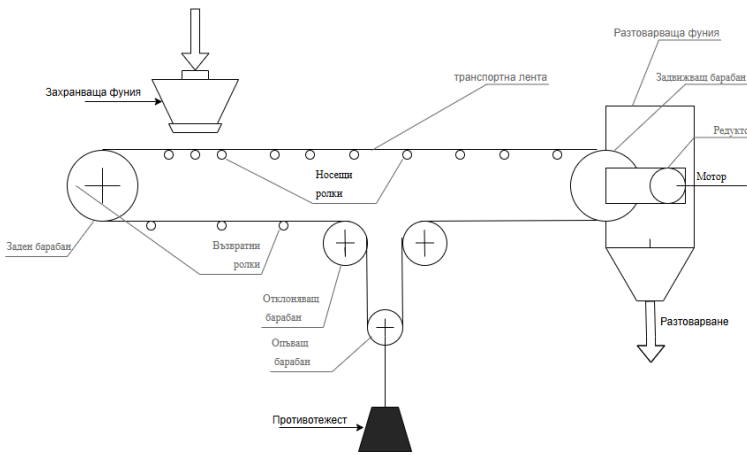
Използваният лабораторен модел позволява изследване на динамичните характеристики на транспортната система при различни режими на управление, както и оценка на ефективността на разработените алгоритми за регулиране на скоростта. Това създава възможност за сравнение на различни хардуерни платформи и алгоритми за управление при еднакви експериментални условия.

• опъваща станция - Опъващата станция осигурява необходимото предварително натягане на транспортната лента и създава условия за нейното стабилно движение по време на работа. Чрез поддържане на постоянно натягане се предотвратява приплъзването на лентата върху задвижващия барабан и се осигурява надеждно предаване на механичната енергия от задвижващия механизъм към транспортната система.

Конструкцията на опъващата станция позволява регулиране на натягането на транспортната лента в зависимост от експлоатационните условия. Това е особено важно при лабораторните изследвания, тъй като осигурява възпроизводими условия при провеждане на експериментите и намалява влиянието на механични фактори върху резултатите от изследването. Правилният избор и настройка на опъващото устройство оказват съществено

влияние върху експлоатационната надеждност, енергийната ефективност и дълготрайността на лентовите транспортьори [18], [22], [24], [33].

Принципна схема на опъваща станция за лентов транспортьор е представена на фиг. 37.



Фиг. 37. Принципна схема на лентов транспортьор с противотежестна опъваща станция (адаптирано по [33])

Както се вижда от фиг. 37, при промишлените лентови транспортьори предварителното натягане на транспортната лента най-често се осигурява посредством противотежестна опъваща станция. Този тип конструкция компенсира измененията в дължината на лентата и поддържа необходимото натягане при различни режими на работа [33]. За разлика от промишлените системи, в разработения лабораторен модел е използвана винтова опъваща станция, която позволява лесно регулиране на натягането и е по-подходяща за лабораторни експериментални изследвания.

Макар опъващата станция да не участва пряко в алгоритъма за управление, нейното конструктивно изпълнение оказва съществено влияние върху динамичните характеристики на лабораторния модел. Поради тази причина нейното правилно настройване е предпоставка за получаване на достоверни експериментални резултати и коректно сравнение на реализираните системи за управление.

• командни органи за управление - Командните органи за управление осигуряват взаимодействието между оператора и разработената компютърна система за управление. Чрез тях се реализират основните функции по пускане, спиране и управление на работните режими на лабораторния модел, както и подаването на команди към програмируемия логически контролер.

В лабораторния модел командните органи са реализирани чрез бутони за управление и превключватели, свързани към входната част на съответния управляващ модул. При задействане на команден орган управляващата система обработва получения сигнал и в зависимост от изпълнявания алгоритъм формира необходимите управляващи въздействия към изпълнителните механизми.

Използването на отделни командни органи позволява реализиране на ръчен режим на управление, както и създава възможност за експериментално изследване на различни алгоритми за управление, защитни функции и аварийни режими. Благодарение на това лабораторният стенд може да бъде използван както за научноизследователска работа, така и за обучение по дисциплини, свързани с компютърните системи за управление и индустриалната автоматизация.

Описаните конструктивни възли формират лабораторния модел на гумено-транспортна лента, използван за експерименталното изследване на разработената компютърна система за управление. Модулното изпълнение на стенда позволява реализиране на различни алгоритми за управление, регулиране и диагностика, както и съпоставимо изследване на различни хардуерни платформи при еднакъв лабораторен обект и предварително определени експериментални условия. В следващия раздел е представена реализацията на компютърната система за управление и сравнителният анализ на платформите Arduino Opta и Arduino Uno.

4.2. Реализация на компютърната система за управление

След представянето на конструкцията и основните функционални възли на лабораторния модел следващият етап от изследването е реализирането на компютърната система за управление. При разработването са използвани основните принципи на модулната архитектура, представена в глава 3, адаптирани към особеностите на лабораторния модел на гумено-транспортна лента и към спецификата на използваните управляващи платформи. Това позволява двете хардуерни реализации да бъдат изградени при обща

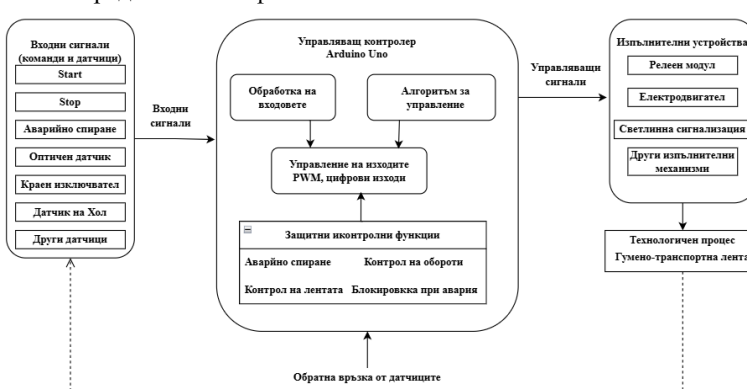
функционална организация и съпоставими изисквания към алгоритъма за управление.

С цел сравнително изследване на възможностите на двете хардуерни платформи са реализирани две функционално еквивалентни системи за управление. Първата система е изградена на базата на микроконтролерната платформа Arduino Uno, а втората – на базата на програмируемия логически контролер Arduino Opta. Двете реализации управляват един и същ лабораторен модел и са разработени при съпоставими функционални изисквания към измерването, управлението и изпълнението на зададените режими. Това създава основа за последващ сравнителен анализ на особеностите и резултатите от двете реализации.

4.2.1. Реализация на компютърната система с Arduino Uno

За експерименталната реализация на разработената компютърна система за управление е използвана микроконтролерната платформа Arduino Uno. Изборът ѝ е обусловен от широкото ѝ приложение в учебния процес, ниската цена, лесната програмна реализация и наличието на богата екосистема от програмни библиотеки и периферни модули. Благодарение на тези характеристики платформата намира широко приложение както в инженерното образование, така и при експериментални научни изследвания в областта на системите за автоматично управление [35]–[37].

Структурната схема на реализираната компютърна система за управление е представена на фиг. 38.



Фиг. 38. Структурна схема на реализираната компютърна система за управление на лабораторния модел на гумено-транспортна лента

Както се вижда от фиг. 38, разработената компютърна система за управление е организирана в три основни функционални части – входни сигнали, управляващ контролер и изпълнителни устройства. При реализирането на системата Arduino Uno изпълнява функциите на централен управляващ модул. Към входовете на контролера са свързани командните органи и измервателните устройства, включително Hall датчикът, които предоставят информация за състоянието на лабораторния модел. На базата на постъпилите входни сигнали контролерът изпълнява разработения алгоритъм за управление и формира съответните управляващи сигнали към изпълнителните устройства.

Изпълнителните устройства включват релеен модул, постояннотоков електродвигател, светлинна сигнализация и други периферни устройства, необходими за реализиране на управлението на лабораторния модел. Чрез тях управляващите сигнали се преобразуват в конкретни действия, които осигуряват функционирането на транспортната система в съответствие със зададения режим на работа. Управлението на постояннотоковия електродвигател се осъществява чрез силов драйвер, управляван с широчинно-импулсен (PWM) сигнал, генериран от Arduino Uno. Чрез промяна на коефициента на запълване на PWM сигнала се изменя управляващото въздействие към електрозадвижването, което позволява регулиране на скоростта на транспортната лента [35], [37].

Освен основния алгоритъм за управление в програмното осигуряване са реализирани защитни и контролни функции, включващи аварийно спиране, контрол на скоростта, наблюдение на състоянието на транспортната лента и блокиране на системата при възникване на аварийни състояния. В програмното осигуряване са предвидени функции за аварийно спиране, контрол на скоростта, наблюдение на състоянието на транспортната лента и обработка на предварително дефинирани аварийни състояния.

Информацията от Hall датчика се използва като сигнал за обратна връзка по скорост при реализиране на затворения контур за регулиране. Измерената скорост се обработва от управляващия алгоритъм и се използва при формиране на управляващото въздействие към електрозадвижването. Благодарение на реализираната обратна връзка се създават условия за стабилно регулиране на скоростта на транспортната лента и за поддържане на зададения режим на работа при различни експериментални условия.

Освен функциите за управление, програмното осигуряване реализира мониторинг на състоянието на системата, обработка на аварийните сигнали

и управление на светлинната сигнализация. Това позволява лабораторният модел да бъде използван както за експериментални изследвания, така и за обучение по дисциплини, свързани с компютърните системи за управление и индустриалната автоматизация.

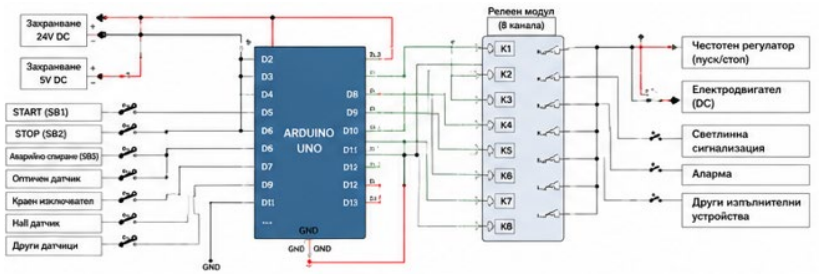
За реализиране на разработената компютърна система за управление са използвани цифровите и аналоговите входове и изходи на Arduino Uno, към които са свързани измервателните преобразуватели, командните органи и изпълнителните устройства. Разпределението на използваните входно-изходни ресурси е представено в Таблица 11.

Таблица 11. Използвани входно-изходни ресурси на Arduino Uno

Вход/Изход	Свързано устройство	Функция
D0	Старт бутон	Стартиране
D1	Стоп бутон	Спиране
D2	Датчик на Хол	Измерване на скорост
PWM	Драйвер	Управление на двигателя
DO	Светлинна сигнализация	Индикация

Както се вижда от таблица 11, използваните входно-изходни ресурси на Arduino Uno са достатъчни за реализиране на основните функции на разработената компютърна система за управление. Цифровите входове се използват за приемане на сигналите от командните органи и Hall датчика, а PWM и цифровите изходи осигуряват управлението на постояннотоковия двигател, релейния модул и светлинната сигнализация. По този начин се реализира пълният цикъл по събиране, обработка и управление на информацията в лабораторния модел.

Представеното в таблица 11 разпределение на входно-изходните ресурси определя начина на свързване на периферните устройства към микроконтролерната платформа Arduino Uno. При реализирането на разработената компютърна система са използвани стандартните цифрови входове и изходи, PWM изходът и външен релеен модул, чрез които се осъществява обменът на информация между управляващия контролер, измервателните преобразуватели и изпълнителните устройства. Електрическата схема на реализираната система за управление е представена на фиг. 39.



Фиг. 39. Електрическа схема на системата за управление с Arduino Uno

Както се вижда от фиг. 39, към цифровите входове на Arduino Uno са свързани командните органи и измервателните преобразуватели, включително бутоните START, STOP, аварийно спиране, оптичният датчик, крайният изключвател и Hall датчикът. Изходните сигнали от контролера управляват външен осемканален релеен модул и PWM канала за регулиране на постояннотоковия двигател, като по този начин се реализира взаимодействието между управляващата и изпълнителната част на системата.

Захранването на отделните компоненти на системата е реализирано в съответствие с изискваните от тях работни напрежения, като за изпълнителната и управляващата част са използвани съответните захранващи вериги. Конкретната организация на захранването е съобразена с електрическите характеристики на използваните контролни, интерфейсни и изпълнителни устройства.

Използването на Arduino Uno позволява реализиране на всички необходими функции за управление, регулиране и мониторинг на лабораторния модел, без необходимост от допълнителни специализирани контролери. Това потвърждава възможността платформата да бъде използвана при разработването на нискобюджетни компютърни системи за управление, предназначени за лабораторни изследвания, обучение и експериментално изследване на алгоритми за автоматично управление.

Разработената система с Arduino Uno успешно реализира всички функционални изисквания към компютърната система за управление на лабораторния модел на гумено-транспортна лента. Използването на микроконтролерната платформа позволява реализиране на функциите за управление, регулиране, мониторинг и защита, необходими за експерименталното изследване на разработените алгоритми. Получените резултати потвърждават, че

Arduino Uno представлява подходящо решение за изграждане на нискобюджетни лабораторни системи за управление.

Реализацията с Arduino Uno осигурява необходимата хардуерна и програмна основа за изпълнение на предвидените функции за управление и експериментално изследване на лабораторния модел. Тази реализация е използвана като един от двата варианта при последващото сравнително изследване с компютърната система, базирана на Arduino Opta.

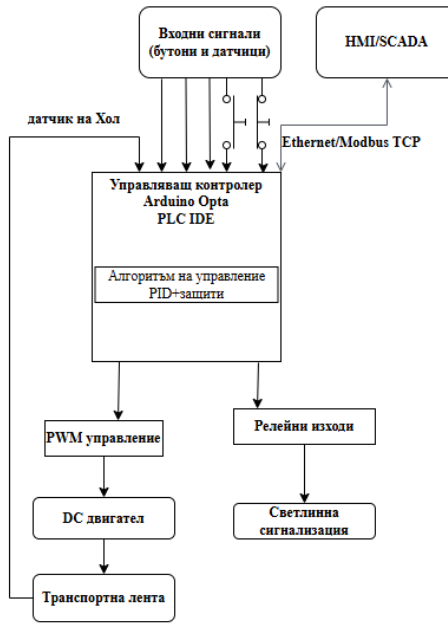
4.2.2. Реализация на компютърната система с Arduino Opta

След реализиране на компютърната система за управление с микроконтролерната платформа Arduino Uno е разработена аналогична система, базирана на програмируемия логически контролер Arduino Opta. Целта е при идентични експериментални условия да бъде извършен сравнителен анализ на възможностите на двете хардуерни платформи при управление на лабораторния модел на гумено-транспортна лента. Използването на Arduino Opta позволява програмната реализация да бъде разработена в PLC ориентирана среда с използване на програмни езици и средства, свързани със стандарта IEC 61131-3 [11], [16], [38].

При реализирането на системата с Arduino Opta е използван същият лабораторен модел, като са запазени основните функционални изисквания към измерването, управлението и работните режими. Това позволява двете хардуерни реализации да бъдат изследвани при съпоставими експериментални условия и да бъдат анализирани особеностите на тяхната хардуерна и програмна реализация. По този начин се елиминира влиянието на външни фактори върху резултатите от изследването и се създават условия за обективно сравнение на функционалните характеристики, надеждността и приложимостта на двете платформи. Подобен подход съответства на съвременните тенденции при проектирането на модулни PLC системи за индустриална автоматизация [16], [39].

Използването на Arduino Opta позволява реализиране на компютърната система за управление чрез индустриален програмируем логически контролер, разработен в съответствие със стандарта IEC 61131-3. Благодарение на това системата може да бъде програмирана чрез Arduino PLC IDE, като същевременно се осигурява възможност за интеграция с индустриални комуникационни протоколи и системи за визуализация и мониторинг [10], [11], [16].

Структурната схема на реализираната компютърна система за управление с Arduino Opta е представена на фиг. 40.



Фиг. 40. Структурна схема на реализираната компютърна система за управление с програмируем логически контролер Arduino Opta

Както се вижда от фиг. 40, реализираната компютърна система за управление с Arduino Opta запазва основната функционална структура на разработената архитектура, като централният управляващ модул е реализиран чрез индустриалния програмируем логически контролер Arduino Opta. Към неговите входове са свързани командните органи и измервателните преобразуватели, които осигуряват информация за текущото състояние на лабораторния модел. Получените входни сигнали се обработват от програмното осигуряване, което реализира алгоритъма за управление, PID регулирането и защитните функции.

Управлението на постояннотоковия двигател се осъществява посредством PWM сигнал, формиран от контролера, а релейните изходи се използват за управление на светлинната сигнализация и изпълнение на защитните функции. Скоростта на въртене на двигателя се определя чрез Hall датчик, който формира сигнал за обратна връзка към управляващия контролер. По

този начин се реализира затворена система за автоматично регулиране, осигуряваща стабилна работа на лабораторния модел при различни режими на натоварване.

Едно от основните предимства на Arduino Opta е възможността за комуникация посредством Ethernet и протокола Modbus TCP, което позволява интегриране на системата със средства за визуализация и мониторинг (HMI/SCADA). Чрез тях могат да бъдат наблюдавани технологичните параметри, състоянието на входовете и изходите, както и да се извършват настройка, диагностика и архивиране на данните в реално време.

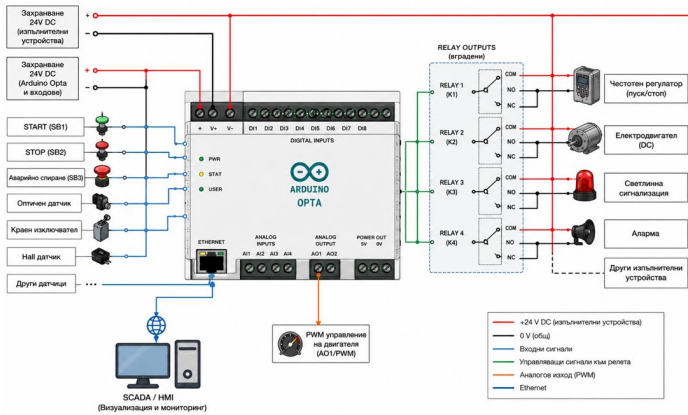
Разпределението на използваните входно-изходни ресурси на Arduino Opta при реализиране на компютърната система за управление е представено в Таблица 12.

Таблица 12. Използвани входно-изходни ресурси на Arduino Opta

Вход/Изход	**Свързано устройство**	**Функция**
I1	Старт бутон	Стартиране
I2	Стоп бутон	Спиране
I3	датчик на Хол	Измерване на скорост
PWM/Q1	Управление на двигателя	Регулиране на скоростта
Q2	Светлинна сигнализация	Индикация
Ethernet	HMI/SCADA	Мониторинг и диагностика

Както се вижда от Таблица 12, Arduino Opta разполага с необходимите входно-изходни ресурси за реализиране на всички функции на разработената компютърна система за управление. Освен основните входове и изходи за управление на лабораторния модел, контролерът предоставя възможност за индустриална Ethernet комуникация и интеграция със SCADA системи, което значително разширява функционалните възможности на системата спрямо реализацията с Arduino Uno.

Представеното в таблица 12 разпределение на входно-изходните ресурси определя начина на свързване на периферните устройства към програмируемия логически контролер Arduino Opta. Благодарение на индустриалната архитектура на контролера се осигурява директно свързване на входните сигнали, вградените релейни изходи и комуникационния Ethernet интерфейс, без необходимост от използване на външни релейни модули. Електрическата схема на реализираната компютърна система за управление е представена на фиг. 41.



Фиг. 41. Електрическа схема на реализираната компютърна система за управление с Arduino Opta

Както се вижда от фиг. 41, командните органи и измервателните преобразуватели са свързани към цифровите входове на Arduino Opta, откъдето постъпилите сигнали се обработват от програмното осигуряване. Управлението на постояннотоковия електродвигател се реализира посредством подходящ интерфейсен/силов модул, управляван от системата с Arduino Opta. Hall датчикът предоставя информация за скоростта на въртене, която се използва като сигнал за обратна връзка при реализиране на алгоритъма за регулиране на скоростта.

За разлика от реализацията с Arduino Uno, при Arduino Opta отпада необходимостта от използване на външен релеен модул, тъй като контролерът разполага с вградени индустриални релейни изходи. Освен това Ethernet интерфейсът позволява директна връзка със SCADA/HMI система за визуализация, мониторинг и диагностика на технологичния процес. Тези особености опростяват хардуерната реализация, повишават надеждността на системата и улесняват нейното внедряване в индустриална среда.

Използването на Arduino Opta като индустриален PLC контролер създава предпоставки не само за реализиране на системи за автоматично управление, но и за прилагане на съвременни методи за верификация, диагностика и повишаване на надеждността на програмното осигуряване, които все по-често намират приложение при разработването на индустриални PLC системи [40].

Комуникационните възможности на Arduino Opta, включително Ethernet свързаността, създават възможност за интегриране на управляващата система със средства за визуализация и мониторинг. В зависимост от конкретната програмна реализация могат да бъдат използвани индустриални комуникационни протоколи и HMI/SCADA приложения за обмен, визуализиране и регистриране на технологична и диагностична информация [41].

Разширените комуникационни възможности и индустриалната архитектура на Arduino Opta създават предпоставки разработената компютърна система за управление да бъде използвана не само като лабораторен стенд, но и като основа за разработване на прототипи на реални индустриални приложения. Подобен подход намира приложение при внедряването на PLC базирани системи за управление в различни отрасли на промишлеността, където се изискват висока надеждност, възможности за дистанционен мониторинг и лесна интеграция с информационни системи [42].

Реализацията с Arduino Opta позволява изпълнение на предвидените функции за управление на лабораторния модел в PLC ориентирана хардуерна и програмна среда и предоставя допълнителни възможности за комуникация и интеграция със средства за визуализация и мониторинг. Получената реализация формира втория хардуерен вариант, използван в последващото сравнително изследване на двете управляващи платформи.

4.2.3. Сравнителен анализ на Arduino Uno и Arduino Opta

След реализирането на компютърната система за управление с микроконтролерната платформа Arduino Uno и програмируемия логически контролер Arduino Opta е извършен сравнителен анализ на двете реализации. Изследванията са проведени върху един и същ лабораторен модел на гумено-транспортна лента при съпоставими експериментални условия и при запазване на основните функционални изисквания към системата за управление. Сравнението е извършено по основни технически, функционални и експериментални показатели, позволяващи да бъдат оценени особеностите и приложимостта на двете хардуерни платформи.

За осигуряване на съпоставимост между двете реализации са използвани едни и същи основни командни органи, измервателни преобразуватели и изпълнителни устройства. Различията са свързани основно с начина на свързване, използваните входно-изходни ресурси и архитектурните

особености на Arduino Uno и Arduino Opta. Съпоставката на използваните входно-изходни ресурси е представена в Таблица 13.

Таблица 13. Съпоставка на използваните входно-изходни ресурси при реализацията на компютърната система за управление

Функция	**Arduino Opta**	**Arduino Uno**
Бутон START	I1	D2
Бутон STOP	I2	D3
Авариен стоп	I3	D4
Оптичен датчик	I4	D5
Краен изключвател	I5	D6
Датчик на Хол	I6	D7
PWM управление на двигателя	A0/PWM	D9 (PWM)
Светлинна сигнализация	DO1	D10
Управление на реле	Relay 1	D11 + релеен модул

Както се вижда от Таблица 13, функционалното предназначение на входните и изходните сигнали е идентично и при двете реализации. Това осигурява функционална съпоставимост на двете реализации при изпълнение на основните задачи за управление на лабораторния модел. Основната разлика се състои в начина на реализиране на входно-изходните интерфейси. При Arduino Uno управлението на релейните товари се осъществява чрез външен релеен модул, докато Arduino Opta разполага с вградени релейни изходи. Освен това индустриалният PLC предлага стандартизирани входно-изходни интерфейси, които улесняват интеграцията на системата в реална индустриална среда. По аналогичен начин PWM управлението на постояннотоковия двигател се реализира чрез съответните PWM изходи на двете платформи, като функционалността на алгоритъма за управление остава непроменена.

Използването на идентична конфигурация на входните и изходните сигнали гарантира, че сравнението между Arduino Uno и Arduino Opta се извършва при еднакви експериментални условия. Това позволява получените различия да бъдат отнесени към особеностите на използваните хардуерни платформи, а не към различия в алгоритъма за управление или периферните устройства.

Реализирането на разработената компютърна система за управление върху две различни хардуерни платформи дава възможност да бъдат оценени техните предимства и ограничения при изпълнение на една и съща задача. Въпреки че и двете реализации осигуряват необходимата

функционалност за управление на лабораторния модел, използваните контролери се различават съществено по своята архитектура, начин на програмиране, комуникационни възможности и предназначение. Поради това е извършена съпоставка на основните им технически и функционални характеристики, представена в Таблица 14.

Таблица 14. Сравнение между Arduino Uno и Arduino Opta

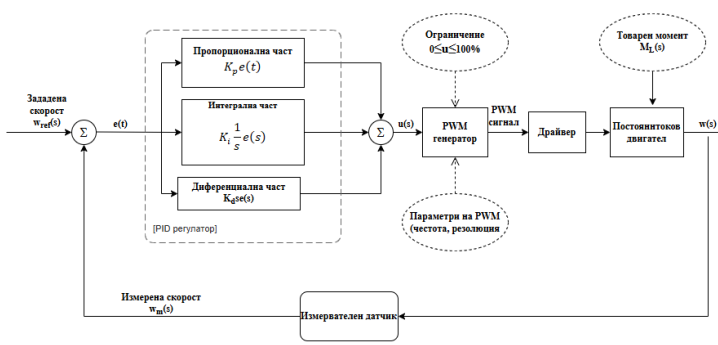
Показател	**Arduino Uno**	**Arduino Opta**
Тип устройство	Микроконтролерна платформа	Индустриален PLC
Програмиране	Arduino IDE (C/C++)	Arduino PLC IDE (IEC 61131-3)
Цифрови входове/изходи	Ограничени	Индустриални I/O
Аналогови входове	Да	Да
PWM	Да	Да
Релейни изходи	Не (необходим външен модул)	Да(вградени)
Ethernet	Не	Да
Modbus TCP	Не	Да
SCADA интеграция	Ограничена	Да
Диагностика	Ограничена	Разширена
Надеждност	Подходяща за лаборатории	Подходяща за индустрия
Цена	Ниска	По висока

Както се вижда от Таблица 14, Arduino Uno и Arduino Opta позволяват реализиране на идентични алгоритми за управление на лабораторния модел, но се различават съществено по отношение на хардуерната архитектура, комуникационните възможности и предназначението си. Arduino Uno представлява нискобюджетна микроконтролерна платформа, подходяща за обучение, лабораторни упражнения и разработване на прототипи, докато Arduino Opta е индустриален програмируем логически контролер, предназначен за изграждане на надеждни системи за управление в реални производствени условия.

Основното предимство на Arduino Uno е неговата достъпност, ниска цена и лесна програмна реализация. Това го прави подходящ за експериментални изследвания и обучение в областта на автоматизацията. От друга страна, Arduino Opta осигурява значително по-големи възможности за индустриална комуникация, диагностика, мониторинг и интеграция със SCADA системи, което разширява областите му на приложение и повишава надеждността на разработената компютърна система за управление.

Освен по отношение на техническите и функционалните характеристики, двете реализации са сравнени и по качеството на регулиране на скоростта на постояннотоковия двигател. Един от основните показатели за ефективността на разработената компютърна система за управление е динамиката на преходните процеси и способността за поддържане на зададената скорост при различни режими на работа. За тази цел е извършен сравнителен анализ между управление без PID регулатор, управление с PID регулатор, реализирано чрез Arduino Uno, и управление с PID регулатор, реализирано чрез Arduino Opta. Анализът е извършен въз основа на преходните характеристики на системата и основните показатели за качеството на регулиране.

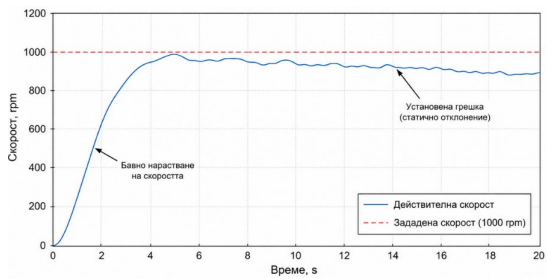
Принципът на работа на разработената система за автоматично регулиране е представен чрез структурната схема на PID управлението на постояннотоковия двигател, показана на фиг. 42. Схемата илюстрира взаимовръзката между зададената скорост, PID регулатора, PWM управлението, електрозадвижването и обратната връзка, реализирана чрез измерване на действителната скорост на въртене.



Фиг. 42. Структурна схема на PID система за управление на постояннотоков двигател

Както е показано на фиг. 42, PID регулаторът формира управляващото въздействие въз основа на грешката между зададената и измерената скорост на въртене. Управляващият сигнал се реализира чрез PWM генератор, който управлява постояннотоковия двигател посредством силов драйвер. Обратната връзка се осигурява от Hall датчика, чрез който се измерва действителната скорост на въртене и се формира сигналът за регулиране.

За да бъде оценено влиянието на PID регулатора върху качеството на регулиране, първоначално е изследвана работата на системата без използване на автоматичен регулатор. В този случай скоростта на постояннотоковия двигател се управлява единствено чрез задаване на коефициента на запълване на PWM сигнала, без корекция на управляващото въздействие въз основа на грешката между зададената и действителната скорост. Получената преходна характеристика е представена на фиг. 43.



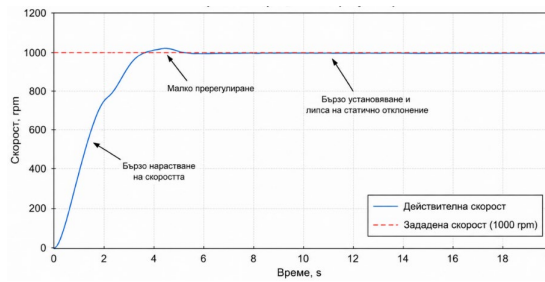
Фиг. 43. Преходна характеристика на системата за управление без PID регулатор

От фиг. 43 се вижда, че при управление без PID регулатор скоростта на двигателя нараства сравнително бавно и не достига напълно зададената стойност. Наблюдава се наличие на установена (статична) грешка, която се дължи на отсъствието на механизъм за автоматична компенсация на измененията в натоварването. Това води до по-ниска точност на регулиране и влошени динамични характеристики на системата.

Освен наличието на статична грешка се наблюдава и по-бавно достигане на установен режим, което води до увеличаване на времето за регулиране и влошаване на динамичните показатели на системата. Тези особености ограничават приложимостта на управлението без обратна връзка при системи, в които се изисква висока точност и устойчивост към измененията в натоварването.

За преодоляване на установените недостатъци в разработената компютърна система е реализиран PID алгоритъм за регулиране на скоростта. Чрез непрекъснато сравняване на зададената и измерената скорост регулаторът изменя управляващото въздействие така, че да минимизира грешката и да подобри динамичните характеристики на системата. Получената

преходна характеристика при използване на PID регулатор е представена на фиг. 44.



Фиг. 44. Преходна характеристика на системата за управление с PID регулатор

От фиг. 44 се вижда, че използването на PID регулатор води до съществено подобряване на качеството на регулиране. Скоростта на двигателя достига зададената стойност значително по-бързо, като се наблюдава само малко пререгулиране, което бързо се компенсира. След приключване на преходния процес статичната грешка практически се елиминира, а системата работи устойчиво около зададената стойност.

Получените резултати показват, че използването на PID регулиране подобрява както бързодействието, така и точността на системата. Благодарение на обратната връзка се компенсират измененията в натоварването и се осигурява стабилно поддържане на зададената скорост. Това е особено важно при системи за управление на транспортни ленти, при които товарът може да се изменя по време на работа.

За количествена оценка на получените резултати е извършено сравнение на основните показатели за качеството на регулиране при трите разгледани реализации. Резултатите са обобщени в таблица 15.

Таблица 15. Сравнение на показателите за качеството на регулиране при различните реализации на системата за управление

Показател	**Без PID (Arduino Uno)**	**С PID (Arduino Uno)**	**Arduino Opta [17]**
Време за установяване	≈ 2.5 s	≈ 0.8 s	≈ 0.8 s
Пререгулиране	няма контрол	≈ 5 %	≈ 5 %
Установена грешка	по-голяма	≈ 0 %	≈ 0 %
Устойчивост при натоварване	ниска	висока	висока
Управление чрез PWM	Да	Да	Да
Използван контролер	Arduino Uno	Arduino Uno	Arduino Opta
Аналогови входове	6	6	8
Цифрови входове/изходи	14	14	8 DI / 4 Relay DO
Подходящ за промишлена среда	Не	Не	Да
Цена	ниска	ниска	висока

Данните, представени в Таблица 15, показват, че използването на PID регулатор води до съществено подобряване на динамичните характеристики на системата за управление. В сравнение с управлението без PID регулатор се наблюдава намаляване на времето за нарастване и времето за установяване, както и практически пълно елиминиране на статичната грешка. Получените резултати потвърждават, че обратната връзка и автоматичната корекция на управляващото въздействие повишават точността и устойчивостта на системата.

Сравнението между реализациите с Arduino Uno и Arduino Opta показва, че при използване на еднакво настроен PID алгоритъм качеството на регулиране остава практически идентично. Това се обяснява с факта, че и двете платформи изпълняват един и същ алгоритъм за управление и използват еднакви измервателни преобразуватели и изпълнителни устройства. Следователно различията между Arduino Uno и Arduino Opta не оказват съществено влияние върху динамичните показатели на регулиране, а се проявяват основно по отношение на комуникационните възможности, надеждността и пригодността за работа в индустриална среда.

Извършеният сравнителен анализ показва, че Arduino Uno представлява подходящо решение за лабораторни изследвания и обучение, докато Arduino Opta осигурява същото качество на регулиране, но предлага значително по-широки функционални възможности за изграждане на

индустриални компютърни системи за управление. Това потвърждава, че изборът на хардуерна платформа следва да бъде съобразен не толкова с качеството на регулиране, колкото с изискванията към надеждността, комуникацията и интеграцията на системата в реална производствена среда.

Получените резултати показват, че и двете платформи успешно реализират разработения алгоритъм за управление на лабораторния модел на гумено-транспортна лента. Arduino Uno е подходящо решение за изграждане на нискобюджетни лабораторни системи, докато Arduino Opta предлага функционални възможности, които го правят значително по-подходящ за внедряване в индустриални системи за управление. Изборът на конкретна платформа следва да се определя от изискванията към надеждността, комуникацията, функционалността и областта на приложение на разработваната система.

Извършеният сравнителен анализ показва, че и двете хардуерни платформи успешно реализират разработения алгоритъм за управление на лабораторния модел на гумено-транспортна лента. Експерименталните резултати потвърждават, че използването на PID регулатор значително подобрява качеството на регулиране, като намалява времето за установяване, елиминира статичната грешка и осигурява стабилна работа на системата. При еднакви настройки на регулатора Arduino Uno и Arduino Opta демонстрират практически идентични динамични характеристики, което показва, че качеството на регулиране се определя основно от използвания алгоритъм, а не от избраната хардуерна платформа.

Основното предимство на Arduino Uno е неговата достъпност, ниска цена и лесно програмиране, което го прави подходящ за лабораторни упражнения, обучение и разработване на експериментални системи за управление. От своя страна Arduino Opta предлага разширени комуникационни възможности, вградени индустриални входно-изходни интерфейси и поддръжка на стандарта IEC 61131-3, което го прави по-подходящ за изграждане и внедряване на компютърни системи за управление в реална индустриална среда. Следователно изборът на конкретна хардуерна платформа следва да бъде съобразен с изискванията към надеждността, комуникацията, функционалността и областта на приложение на разработваната система.

Изводи към глава 4

В настоящата глава са представени две реализации на компютърна система за управление на лабораторен модел на гумено-транспортна лента, базирани съответно на микроконтролерната платформа Arduino Uno и програмируемия логически контролер Arduino Opta. Двете реализации са разработени за управление на един и същ лабораторен обект при съпоставими функционални изисквания и експериментални условия, което създава основа за сравнителен анализ на техните технически и функционални характеристики.

Изследването на системата за регулиране на скоростта показва влиянието на PID алгоритъма върху динамичното поведение на електрозадвижването. Сравнението между реализациите с Arduino Uno и Arduino Opta позволява да бъдат оценени както характеристиките на регулирането, така и влиянието на използваната хардуерна платформа върху изпълнението на алгоритъма за управление. При интерпретацията на получените резултати следва да се разграничават ефектът от използвания алгоритъм за регулиране и особеностите, произтичащи от хардуерната и програмната реализация на двете платформи.

Анализът на функционалните възможности на двете платформи показва, че Arduino Uno представлява подходящо решение за разработване на нискобюджетни лабораторни и учебно-изследователски системи. Arduino Opta, от своя страна, съчетава PLC ориентирана архитектура, вградени релейни изходи, комуникационни възможности и програмни средства, свързани със стандарта IEC 61131-3, което разширява възможностите за интегрирането му в системи за индустриална автоматизация, визуализация и мониторинг.

Получените резултати показват, че изборът на хардуерна платформа следва да бъде съобразен не само с възможността за реализиране на необходимия алгоритъм за управление, но и с конкретните изисквания към входно-изходните ресурси, комуникационните възможности, програмната среда, разширяемостта и областта на приложение на разработваната система. По този начин сравнението между Arduino Uno и Arduino Opta очертава различни области на рационално приложение на двете платформи при изграждането на компютърни системи за управление.

Глава 5. Разработване и експериментално изследване на автоматизирана система за управление на зонална спринклерна пожарогасителна система

Разработената в Глава 3 универсална архитектура на компютърна система за управление позволява адаптиране към технологични обекти с различно предназначение, структура и принцип на действие. След приложението ѝ при управлението на лабораторен модел на гумено-транспортна лента, разгледано в Глава 4, в настоящата глава е изследвана възможността за нейното използване при система за безопасност, предназначена за автоматизирано откриване и потушаване на пожари в затворени пространства. По този начин се разширява областта на приложение на предложената архитектура и се оценява възможността за използването ѝ при обекти, при които надеждността и безопасността имат определящо значение.

Пожарите в затворени пространства представляват съществен риск за живота и здравето на хората, сградите, технологичното оборудване и непрекъснатостта на производствените процеси. Към тази категория могат да бъдат отнесени транспортни тунели, складови помещения, производствени халета, технически помещения и други обекти, при които ограничената естествена вентилация, затруднената евакуация и бързото разпространение на дим и токсични продукти от горенето могат значително да увеличат последствията от възникнал пожар. Поради това ранното откриване на признаците за пожар и своевременното активиране на подходящи пожарогасителни средства имат съществено значение за ограничаване на развитието и последствията от аварийната ситуация [43].

В настоящата глава е разработена и експериментално изследвана автоматизирана система за управление на зонална спринклерна пожарогасителна система, базирана на микроконтролерната платформа Arduino Uno. Разработката стъпва върху резултати, публикувани в авторска научна статия [43], като в монографията изследването е разширено чрез по-подробно представяне на архитектурата на системата, алгоритъма за управление, хардуерната и програмната реализация, както и чрез допълнителен анализ на принципите на функциониране и получените експериментални резултати.

Предложената система използва многосензорно наблюдение на параметри, характеризиращи възникването на пожарна опасност, включително температура, наличие на дим и концентрация на наблюдавани газове. Получената информация се обработва от управляващия контролер, като въз основа на зададен алгоритъм се определя текущият режим на работа, идентифицира се

засегнатата зона и се формират управляващи въздействия към съответните изпълнителни устройства.

Основен акцент при разработването на системата е алгоритъмът за вземане на решения, основан на комбинирана оценка на информацията от няколко независими измервателни канала. Активирането на пожарогасителната система се извършва при потвърждаване на пожарното събитие чрез предварително зададени логически критерии, което позволява да се намали вероятността от неправомерно задействане при единично или краткотрайно изменение на наблюдаван параметър. На тази основа са разграничени три основни режима на функциониране – нормален, предупредителен и авариен.

В главата последователно са разгледани съвременните автоматични пожарогасителни системи, принципът на действие и конструктивните особености на спринклерните системи, архитектурата на разработената автоматизирана система, алгоритъмът за управление, хардуерната и програмната реализация и резултатите от проведените експериментални изследвания. Чрез разработената система се оценява приложимостта на предложените в монографията принципи за изграждане на компютърни системи за управление с отворен код при реализацията на автоматизирани системи с повишени изисквания към безопасността.

5.1. Анализ на съвременните системи за автоматично пожарогасене

Пожарите представляват една от най-сериозните аварийни ситуации в промишлени предприятия, обществени сгради, складови бази и транспортна инфраструктура. Освен значителните материални щети, те могат да доведат до прекъсване на производствения процес, унищожаване на технологично оборудване и създаване на непосредствен риск за живота и здравето на хората. Поради това осигуряването на надеждна противопожарна защита е сред основните изисквания при проектирането и експлоатацията на съвременните инженерни системи [44], [46].

Автоматичните системи за пожарогасене представляват съществен елемент от комплексната противопожарна защита на сгради, промишлени обекти и технологични съоръжения. Основното им предназначение е своевременно откриване на признаци за възникване на пожар и автоматично активиране на подходящи средства за ограничаване и потушаване на горенето. Ранното задействане на пожарогасителната система има особено значение в затворени пространства, където разпространението на дим, топлина и продукти от горенето може бързо да създаде опасни условия за хората и технологичното оборудване.

Развитието на автоматизираните системи за пожароизвестяване и пожарогасене е насочено към намаляване на времето за откриване на пожарното огнище и ограничаване на последствията още в началния етап на неговото развитие. Съвременните системи осъществяват непрекъснат мониторинг на контролираните помещения чрез различни видове измервателни преобразуватели, обработват получената информация в реално време и при необходимост автоматично управляват изпълнителните механизми на противопожарната инсталация. По този начин се свежда до минимум времето между възникването на пожара и активирането на средствата за неговото локализиране и потушаване [46], [49], [51].

В зависимост от използваното пожарогасително вещество и особеностите на защитавания обект автоматичните пожарогасителни системи могат да бъдат реализирани като водни, водомъглови, газови, пенообразуващи и прахови системи. Изборът на конкретен тип се определя от характера на пожарния риск, свойствата на горимите материали, конструктивните особености на защитаваното пространство, наличието на хора и технологично оборудване, както и от нормативните изисквания за пожарна безопасност.

Водните спринклерни системи са широко приложими при защита на обекти, в които използването на вода като пожарогасително средство е допустимо. Газовите системи намират приложение предимно в помещения с чувствително електронно и технологично оборудване, при които е необходимо ограничаване на вторичните щети от пожарогасителното вещество. Водомъгловите системи използват фино диспергирана вода и позволяват ефективно охлаждане при сравнително ограничено водно потребление, докато пенообразуващите и праховите системи се използват при специфични категории пожарен риск и технологични приложения. Различията в принципа на действие на тези системи определят и различни изисквания към алгоритмите за автоматично управление, средствата за детекция и изпълнителните механизми.

Нормативните изисквания към автоматизираните противопожарни системи са регламентирани в европейските стандарти от серията БДС EN 54, които определят изискванията към пожароизвестителните системи, както и в БДС EN 12845 и NFPA 13, регламентиращи проектирането, изграждането и експлоатацията на автоматичните спринклерни инсталации. Стандартът EN 54-14 допълва тези изисквания чрез препоръки относно планирането, проектирането, въвеждането в експлоатация и техническото обслужване на пожароизвестителните системи [46], [49], [51], [52].

Автоматизираните противопожарни системи се изграждат като компютърни системи за управление, които включват измервателни преобразуватели, управляващ контролер, комуникационна инфраструктура и изпълнителни устройства. В зависимост от предназначението на защитавания обект могат да бъдат реализирани различни алгоритми за вземане на решения, включително използване на множество независими сензори, логически зависимости между тях и зонално управление на изпълнителните механизми. Подобен подход повишава надеждността на системата и намалява вероятността от фалшиви сработвания, което е особено важно при промишлени обекти и инфраструктура с повишени изисквания към безопасността [44], [49], [52].

От гледна точка на автоматизираното управление съвременната пожарогасителна система може да бъде разглеждана като комплексна система, включваща измервателна, управляваща и изпълнителна подсистема. Измервателната подсистема осигурява информация за параметрите на защитаваната среда чрез температурни, димни, газови и други видове датчици. Управляващата подсистема обработва получената информация и въз основа на предварително зададен алгоритъм определя текущото състояние на системата и необходимите управляващи действия. Изпълнителната подсистема реализира формираните команди чрез активиране на електромагнитни клапани, помпи, сигнализационни устройства, вентилационни системи и други изпълнителни механизми.

При традиционното централизирано управление активирането на пожарогасителната система може да обхване значителна част от защитавания обект независимо от конкретното местоположение на възникналото пожарно събитие. При зоналния принцип защитаваното пространство се разделя на отделни функционални зони, които могат да бъдат наблюдавани и управлявани независимо. Това позволява локализиране на възникналото събитие и активиране само на изпълнителните устройства, свързани със съответната зона, когато архитектурата и проектът на системата предвиждат такъв режим на действие.

Зоналната организация създава предпоставки за по-селективно управление на пожарогасителния процес и за ограничаване на ненужното задействане на пожарогасителни средства извън засегнатата зона. Особено значение при подобна архитектура има надеждното определяне на състоянието във всяка зона и правилното формиране на управляващото решение.

Въз основа на извършения анализ в настоящото изследване е избран принципът на зонално управление на спринклерна пожарогасителна система. Изборът е обусловен от възможността защитаваното пространство да бъде разделено на отделни контролирани зони, за които да се извършва независимо наблюдение на параметрите и селективно формиране на управляващи въздействия. За повишаване на достоверността при определяне на пожарно събитие е приложен многосензорен подход, при който управляващото решение се формира въз основа на комбинирана информация от няколко измервателни канала.

Извършеният анализ показва, че ефективността на съвременните автоматични пожарогасителни системи зависи не само от вида на използваното пожарогасително средство, но и от архитектурата на системата за управление, начина на обработване на информацията от датчиците и алгоритъма за вземане на решения. От тази гледна точка зоналният принцип на управление предоставя подходяща основа за разработване на модулна автоматизирана система, при която отделните зони могат да бъдат наблюдавани и управлявани в зависимост от текущото им състояние. Поради това в следващия подраздел е разгледан принципът на действие и конструктивните особености на спринклерните пожарогасителни системи, които са използвани като основа при разработването на предложената система за управление.

5.2. Принцип на действие и конструктивни особености на автоматичните спринклерни системи

Спринклерните пожарогасителни системи представляват широко използван вид стационарни автоматични системи за противопожарна защита, при които водата се използва като основно пожарогасително средство. Основното им предназначение е своевременно ограничаване и контролиране на развитието на пожара чрез подаване на вода в защитаваното пространство. Конструкцията и принципът им на действие се определят от предназначението на защитавания обект, характеристиките на пожарния риск и приложимите нормативни изисквания.

При класическите спринклерни системи отделните спринклерни глави обикновено са снабдени с термочувствителен елемент, който реагира при достигане на определена температура. При задействането му се освобождава водният поток през съответната спринклерна глава и водата се разпределя върху защитаваната площ. По този начин активирането се извършва локално в зоната, в която температурното въздействие е достигнало необходимото ниво за задействане на съответния спринклер.

При разработването на автоматизирани пожарогасителни системи принципът на локално действие може да бъде допълнен чрез използване на измервателни датчици, програмируем управляващ контролер и електрически управлявани изпълнителни устройства. В този случай информацията за състоянието на защитаваното пространство се получава от няколко измервателни канала, а решението за активиране на съответната пожарогасителна зона се формира програмно въз основа на предварително зададен алгоритъм.

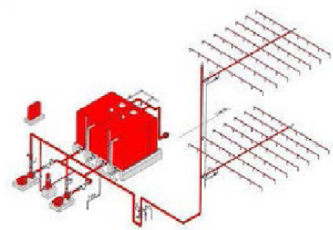
При зоналната организация защитаваното пространство се разделя на отделни контролирани участъци, като за всяка зона могат да бъдат предвидени съответни измервателни средства и изпълнителни устройства. Информацията от датчиците се обработва от управляващия контролер, който определя текущото състояние на наблюдаваните зони и при изпълнение на зададените критерии формира команда за активиране на изпълнителните устройства, свързани със засегнатата зона.

Основният принцип на работа се състои в автоматично подаване на пожарогасително вещество към зоната на възникване на пожара след достигане на предварително определени условия за сработване на спринклерните елементи [48], [49], [51].

5.2.1. Принцип на действие на автоматичните спринклерни системи

Основните конструктивни елементи на една спринклерна пожарогасителна система включват водоизточник, помпена група, тръбопроводна мрежа, контролно-сигнални устройства и спринклерни глави. В зависимост от конкретната конструкция и условията на експлоатация могат да бъдат използвани различни типове системи и допълнителни контролни, сигнализационни и управляващи устройства.

Пример за реализация на такава система е показан на фиг. 45, разработена от Makedonska и съавтори при изграждането на стенд за изпитване на спринклерни дюзи [45].



Фиг. 45. Принципна схема на стенд за изпитване на спринклерни дюзи [45].

Както се вижда от фиг. 45, системата включва воден резервоар, помпена станция, тръбопроводна мрежа и изпитвателна секция със спринклерни дюзи. Макар разработката да е предназначена за експериментални изследвания, тя представя основните конструктивни елементи на класическата автоматична спринклерна инсталация.

Класическите автоматични спринклерни системи осигуряват надеждно пожарогасене, но управлението им обикновено се основава на локалното сработване на отделните спринклерни глави. Развитието на съвременните компютърни системи за управление позволява интегриране на множество измервателни преобразуватели, обработка на информацията в реално време и реализиране на алгоритми за вземане на решения, които повишават надеждността и намаляват вероятността от фалшиви сработвания. На фиг. 46 е представена обобщена принципна схема на автоматизирана зонална спринклерна система за управление.

Фиг. 46. Обобщена принципна схема на автоматизирана зонална спринклерна пожарогасителна система.

Принципът на действие на автоматичните спринклерни системи се основава на непрекъснатата готовност за подаване на пожарогасително вещество при възникване на пожар. В нормален режим тръбопроводната мрежа е запълнена с вода или се намира под налягане, като всяка спринклерна глава остава затворена посредством термочувствителен елемент. При повишаване на температурата в зоната на пожара до предварително определена стойност този елемент се разрушава или освобождава заключващия механизъм, вследствие на което се отваря отворът на спринклера и към защитаваната зона се подава вода под налягане [48], [49].

След задействане на спринклера водната струя преминава през специален разпръскващ елемент (дефлектор), който осигурява равномерно разпределение на водните капки върху защитаваната площ. По този начин се осъществява охлаждане на горящите материали, намаляване на температурата и ограничаване на разпространението на пожара. При правилно проектирана система активирането се извършва само в зоната, в която е възникнал пожарът, което значително намалява разхода на вода и ограничава вторичните щети върху оборудването и помещенията [48], [49], [51].

Съвременните автоматични спринклерни системи все по-често се изграждат като зонални системи за управление. При тях защитаваният обект се разделя на отделни функционални зони, като всяка зона разполага със собствен набор от измервателни преобразуватели и изпълнителни устройства. Управляващият контролер обработва информацията от датчиците, определя местоположението на възникналия пожар и активира единствено съответната секция на пожарогасителната инсталация. Този подход повишава ефективността на пожарогасенето, намалява консумацията на пожарогасително вещество и ограничава възможността за ненужно задействане на останалите секции [49], [51], [52].

Съществено предимство на зоналната организация от гледна точка на автоматизираното управление е възможността за независимо наблюдение на отделните участъци на защитавания обект и селективно формиране на управляващи въздействия. По този начин алгоритъмът за управление може да отчита състоянието на всяка зона и да активира съответните изпълнителни устройства в зависимост от локализирането и потвърждаването на пожарното събитие.

За повишаване на надеждността през последните години все по-широко приложение намират интелигентните системи за управление, при които решението за задействане на пожарогасителната система не се основава

единствено на информацията от един температурен елемент, а се взема след комплексен анализ на данните от множество измервателни преобразуватели – температурни, димни и газови сензори. Подобен подход намалява вероятността от фалшиви сработвания и повишава надеждността на автоматизираната противопожарна защита [47], [54], [58].

За разлика от система, при която управляващото решение се основава на информация само от един измервателен параметър, използването на няколко вида датчици позволява едновременно наблюдение на различни признаци, свързани с възникването и развитието на пожар. В разработената система се използва информация за температурата, наличието на дим и наблюдаваните газови компоненти. Комбинираната обработка на тези сигнали позволява реализиране на алгоритъм с различни режими на функциониране и потвърждаване на аварийното състояние преди активиране на съответната пожарогасителна зона.

Извършеният анализ на принципа на действие и конструктивните особености на спринклерните пожарогасителни системи показва възможността традиционната пожарогасителна инфраструктура да бъде интегрирана със средства за автоматизирано наблюдение и управление. Разделянето на защитавания обект на отделни зони, използването на многосензорно наблюдение и програмното формиране на управляващи решения създават предпоставки за изграждане на система с по-гъвкава логика на функциониране. На тази основа е разработена автоматизираната зонална пожарогасителна система, чиято архитектура и функционална организация са представени в следващия подраздел.

5.2.2. Видове спринклери

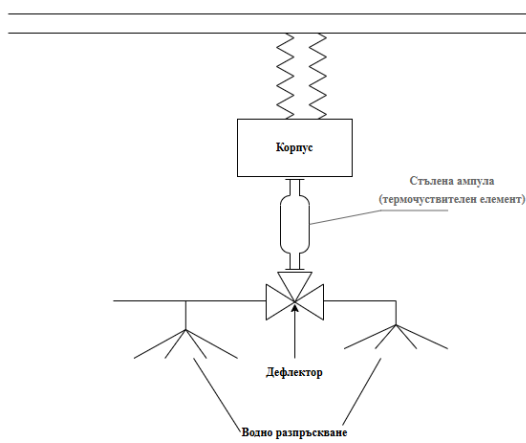
Автоматичните спринклери представляват крайни изпълнителни елементи на пожарогасителната инсталация, предназначени за автоматично разпръскване на вода при достигане на предварително определена температура. Конструктивните и експлоатационните им характеристики са регламентирани в БДС EN 12259-1, а изборът им зависи от предназначението на защитавания обект, височината на помещението и пожарната опасност [48], [49].

- **Спринклери с висящ монтаж (pendent)**

Спринклерите с висящ монтаж, означавани в техническата литература като *pendent sprinklers*, се монтират под хоризонталната разпределителна тръба, като корпусът на спринклера е насочен вертикално надолу.

Термочувствителният елемент и дефлекторът се разполагат под тръбопровода, което позволява водната струя след задействане да бъде насочена към защитаваната площ. Този тип конструкция е сред най-разпространените решения при изграждането на автоматични спринклерни инсталации в обществени, административни, промишлени и складови сгради [48], [49], [51], [61].

Спринклерите с висящ монтаж се свързват към разпределителния тръбопровод така, че корпусът, термочувствителният елемент и дефлекторът да бъдат разположени под него. При нормални условия изходният отвор се поддържа затворен от стъклена ампула, съдържаща термочувствителна течност. Когато температурата в защитаваното помещение достигне определената стойност на сработване, налягането в ампулата нараства и тя се разрушава. Вследствие на това затварящият механизъм се освобождава, водата преминава през корпуса и попада върху дефлектора, който я разпределя върху защитаваната площ. Основните конструктивни елементи и посоката на водното разпръскване са представени на фиг. 47 [48], [49], [60].



Фиг. 47. Конструктивно устройство и принцип на действие на спринклер с висящ монтаж — авторска разработка по [48], [49], [60]

При това конструктивно изпълнение водата се насочва вертикално надолу към дефлектора, след което се разпределя радиално и образува приблизително конусовидна зона на оросяване. Формата и размерите на защитаваната площ зависят от геометрията на дефлектора, работното налягане,

дебита, височината на монтаж и разстоянието до съседните спринклерни глави. Висящите спринклери се използват широко в помещения с открити тръбопроводи или окачени тавани, където е необходимо водата да бъде насочена непосредствено към разположените под тях горими материали [49], [51], [60], [63].

В нормален режим изходният отвор на спринклера е затворен чрез термочувствителен елемент, който може да бъде изпълнен като стъклена ампула или стопяема връзка. При повишаване на температурата до определената температура на задействане термочувствителният елемент се разрушава или освобождава затварящия механизъм. В резултат на това водата преминава през отвора на спринклера и достига до дефлектора, който я раздробява и разпределя върху защитаваната зона.

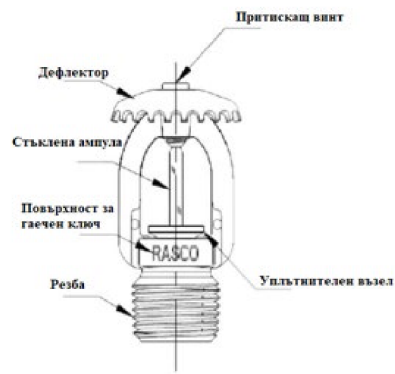
Характерна особеност на висящите спринклери е формата на дефлектора, чрез която водният поток се насочва предимно надолу и странично. По този начин се формира приблизително конусовидна зона на оросяване, чиито размери и равномерност зависят от конструкцията на спринклера, работното налягане, дебита, височината на монтаж и разстоянието между отделните спринклерни глави. Тези параметри трябва да бъдат съобразени с изискванията на приложимите стандарти и с категорията на пожарна опасност на защитавания обект **[**48], [49].

Висящият монтаж е особено подходящ при помещения с окачени тавани или открити хоризонтални тръбопроводи, при които спринклерната глава може да бъде разположена непосредствено под таванната конструкция. В зависимост от архитектурните и експлоатационните изисквания спринклерът може да бъде монтиран открито или чрез декоративна розетка, която прикрива отвора около тръбната връзка. При избора на конкретно изпълнение трябва да се осигури свободно разпръскване на водата и да се избегне наличието на конструктивни елементи, осветителни тела или технологично оборудване, които могат да нарушат проектната форма на водния поток.

Основните предимства на спринклерите с висящ монтаж са сравнително простата конструкция, удобният монтаж и ефективното разпределение на водата върху разположените под тях горими материали. Като ограничение може да се посочи, че този тип спринклери изисква разпределителният тръбопровод да бъде разположен над спринклерната глава. Освен това неправилното позициониране спрямо тавана или наличието на препятствия може да доведе до неравномерно покритие на защитаваната площ.

- **Спринклери с изправен монтаж (upright)**

Спринклерите с изправен монтаж (*upright sprinklers*) се монтират върху горната страна на разпределителния тръбопровод, като корпусът и дефлекторът са разположени над него. При задействане водният поток се насочва към дефлектора, който осигурява равномерното му разпределение върху защитаваната площ. Благодарение на тази конструкция спринклерите с изправен монтаж са подходящи за помещения с открити тръбопроводи, високи тавани, складови и промишлени сгради. Те се използват и в случаи, когато съществува вероятност върху тръбопроводите да се натрупват прах или други замърсявания, които биха могли да повлияят на работата на спринклера. Основните конструктивни елементи и принципът на действие на спринклер с изправен монтаж са представени на фиг. 48 [48], [49], [51], [60].



Фиг. 48. Конструктивно устройство на спринклер с изправен монтаж [60].

Както се вижда от фиг. 48, спринклерът с изправен монтаж се състои от корпус с резбова връзка, стъклена ампула, уплътнителен възел, притискащ винт и дефлектор. При нормална експлоатация стъклената ампула задържа уплътнителния механизъм в затворено положение. При достигане на определената температура ампулата се разрушава, вследствие на което водата преминава през корпуса и се насочва към дефлектора. Последният осигурява равномерното разпределение на водния поток върху защитаваната площ, като формира необходимата зона на оросяване съгласно конструктивните характеристики на спринклера [48], [49], [60].

Основните предимства на спринклерите с изправен монтаж са възможността за монтаж върху открити тръбопроводи, надеждната работа в

помещения с високи тавани и по-ниската вероятност от натрупване на прах или замърсявания върху дефлектора. Благодарение на разположението си те са подходящи за складови, производствени и логистични помещения, където съществуват конструктивни препятствия или повишен риск от механично въздействие върху спринклерните глави. Като ограничение може да се посочи, че при помещения с окачени тавани този тип спринклери не е приложим, тъй като изисква открит монтаж на разпределителния тръбопровод [49], [51], [60].

- **Странични спринклери (Sidewall)**

Страничните спринклери (sidewall sprinklers) се монтират към вертикални стени, като водният поток се разпределя хоризонтално и надолу върху защитаваната площ. За разлика от висящите и изправените спринклери, които се монтират към таванния тръбопровод, при страничните спринклери тръбната връзка е разположена в стената. Този тип спринклери намира широко приложение в хотелски стаи, коридори, офиси, болнични помещения и други обекти, при които таванният монтаж е затруднен или не е желателен по архитектурни причини. Основните конструктивни елементи и принципът на действие на страничния спринклер са представени на фиг. 49 [48], [49], [51], [60], [62].

Фиг. 49. Конструктивно устройство на страничен спринклер (sidewall) [60]

Както се вижда от фиг. 49, страничният спринклер се състои от корпус с резбова връзка, стъклена ампула, уплътнителен възел и дефлектор. След разрушаване на стъклената ампула водата преминава през корпуса и достига до дефлектора, който насочва водния поток хоризонтално и надолу към

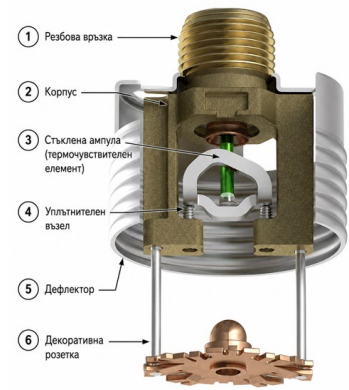
защитаваната зона. По този начин се осигурява ефективно покритие на помещението, без да е необходимо разполагането на тръбопроводи по тавана [48], [49], [60].

Основните предимства на страничните спринклери са лесният монтаж към стени, възможността за запазване на архитектурния облик на помещенията и отпадането на необходимостта от таванен тръбопровод. Те са особено подходящи за помещения с окачени тавани, декоративни таванни конструкции или ограничено пространство над тавана. Като ограничение може да се посочи, че защитаваната площ обикновено е по-малка в сравнение с висящите и изправените спринклери, поради което при по-големи помещения е необходимо използването на по-голям брой спринклерни глави [49], [51], [60].

- **Скрити спринклери (concealed)**

Скритите спринклери представляват разновидност на висящите спринклери, при които спринклерната глава е разположена над равнината на тавана и е закрита с декоративна капачка. След отделянето на капачката и нагряването на термочувствителния елемент стъклената ампула се разрушава, освобождавайки водния канал към дефлектора [48], [49], [51], [60], [65]–[67].

При нормални условия спринклерът не е видим, което позволява запазване на архитектурния облик на помещенията. При възникване на пожар покривната розетка се освобождава вследствие на повишената температура, след което се задейства термочувствителният елемент и започва разпръскването на вода. Скритите спринклери намират широко приложение в хотели, административни сгради, търговски центрове, жилищни сгради и други помещения, при които освен пожарната безопасност се поставят високи изисквания към интериорния дизайн. Основните конструктивни елементи и принципът на действие на скрития спринклер са представени на фиг. 50 [48], [49], [51], [60], [61].



Фиг. 50 Конструктивно устройство на скрит спринклер [59]

Както се вижда от фиг. 50, скритият спринклер включва корпус, термочувствителен елемент, уплътнителен възел, дефлектор и декоративна покривна розетка. Скоростта на задействане се определя не само от номиналната температура на ампулата, но и от нейната топлинна инерция, характеризирана чрез показателя RTI, скоростта и температурата на газовия поток и топлопроводността между отделните конструктивни елементи [65]–[67].

При повишаване на температурата първоначално се освобождава покривната розетка, след което стъклената ампула се разрушава и водата преминава през корпуса към дефлектора. Последният осигурява равномерно разпределение на водния поток върху защитаваната площ съгласно конструктивните характеристики на спринклера [48], [49], [60].

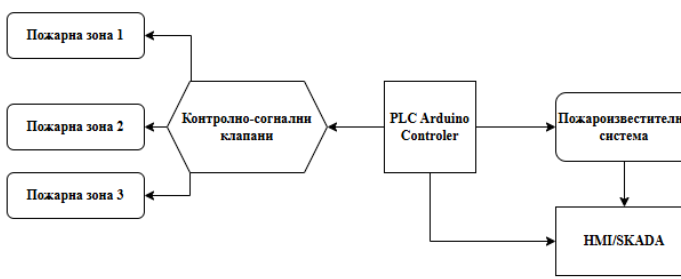
Основните предимства на скритите спринклери са добрата естетика, възможността за интегриране в окачени тавани и минималното влияние върху архитектурния облик на помещенията. Поради тези характеристики те се използват предимно в представителни обществени и жилищни сгради. Като недостатъци могат да се посочат по-високата цена, необходимостта от прецизен монтаж и използването на декоративни розетки, съвместими с конкретния модел спринклер. При подмяна или обслужване трябва да се използват оригинални компоненти, препоръчани от производителя [49], [51], [60].

5.2.3 Зонално управление на спринклерните системи.

При проектирането на автоматичните спринклерни пожарогасителни системи защитаваните обекти обикновено се разделят на отделни пожарни

зони. Всяка зона представлява самостоятелен участък от инсталацията, който включва определен брой спринклери, разпределителни тръбопроводи, контролно-сигнален клапан и средства за контрол на състоянието. Разделянето на сградата на отделни зони позволява локализиране на пожара, ограничаване на водните щети и по-лесно обслужване и диагностика на системата [49], [51], [61].

Съвременните автоматични спринклерни пожарогасителни системи все по-често използват зонален принцип на организация и управление. При този подход защитаваният обект се разделя на отделни пожарни зони, като всяка от тях се обслужва от самостоятелна група спринклери и контролно-сигнален клапан. По този начин се осигурява локализиране на пожара, ограничаване на водните щети и по-ефективно управление на процеса на пожарогасене. В системите с централизирано управление информацията от пожароизвестителната система се използва за определяне на зоната с възникнал пожар и за управление на изпълнителните устройства. Общият принцип на зоналното управление на автоматична спринклерна пожарогасителна система е представен на фиг. 51 [49], [51], [61], [63].



Фиг. 51. Общ принцип на зоналното управление на автоматична спринклерна пожарогасителна система

Както се вижда от фиг. 51, защитаваният обект е разделен на отделни пожарни зони, всяка от които е свързана към съответен контролно-сигнален клапан. Управлението се осъществява от програмируем контролер, който приема информация от пожароизвестителната система и в зависимост от местоположението на пожара формира управляващи сигнали към изпълнителните устройства. Едновременно с това информацията за състоянието на системата се визуализира чрез HMI/SCADA интерфейс, което позволява

наблюдение в реално време, регистриране на събитията и дистанционен контрол на процесите.

Представената схема илюстрира общата концепция за зонално управление и не отразява конкретната реализация на разработената система. Архитектурата, функционалните модули и алгоритъмът на работа на разработената автоматизирана система за управление са разгледани подробно в следващия раздел.

Размерът и конфигурацията на пожарните зони се определят в зависимост от предназначението на сградата, нейната площ, пожарното натоварване, височината на помещенията и нормативните изисквания. При възникване на пожар се активират само спринклерите, разположени в зоната, където температурата достигне температурата на задействане на термочувствителните елементи. По този начин се осигурява локално пожарогасене без едновременно задействане на всички спринклери в сградата [48], [49], [51].

В традиционните мокри спринклерни системи управлението се осъществява по пасивен принцип. Всеки спринклер представлява самостоятелно автоматично устройство, което се задейства единствено при локално повишаване на температурата. След разрушаване на стъклената ампула или стопяемия елемент водата постъпва към дефлектора и се разпръсква върху защитаваната площ. Контролният клапан на съответната зона осигурява подаването на вода към всички спринклери в зоната, като реално се задействат само тези, които са достигнали температурата на сработване [49], [51], [65].

Съвременните тенденции в развитието на противопожарните системи са насочени към използването на интелигентни средства за управление, позволяващи интегриране на пожароизвестителни системи, програмируеми логически контролери, електромагнитни клапани и средства за дистанционен мониторинг. Това позволява предварително определяне на засегнатата пожарна зона, управление на изпълнителните устройства и обмен на информация с диспечерски системи за сградна автоматизация [63], [64], [68].

Въпреки високата надеждност на класическите спринклерни системи, тяхното действие се основава изцяло на локалното термично задействане на отделните спринклери, което ограничава възможностите за централизирано управление и адаптивно реагиране при сложни пожарни сценарии. С оглед преодоляване на тези ограничения в настоящата разработка е реализирана автоматизирана система за зонално управление на спринклерна пожарогасителна инсталация, чиято архитектура и принцип на функциониране са представени в следващия раздел.

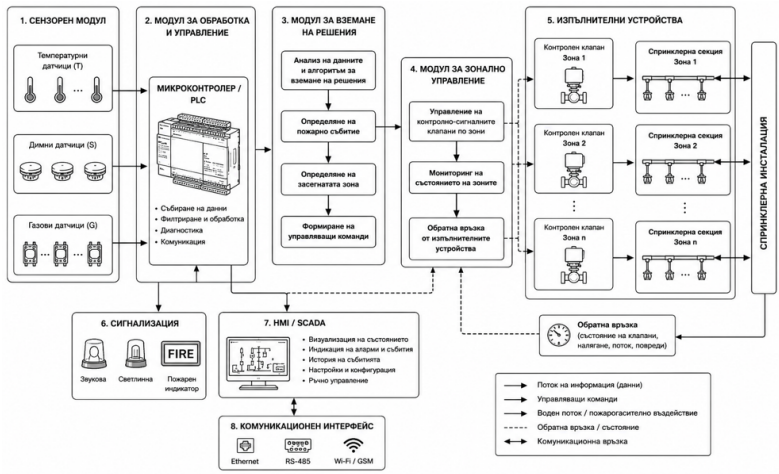
5.3. Архитектура на разработената автоматизирана система

Разработената автоматизирана система е предназначена за непрекъснато наблюдение на параметрите, характеризиращи възникването и развитието на пожар, своевременно откриване на потенциално опасни състояния и автоматично управление на пожарогасителните устройства в зависимост от локализацията на установеното пожарно събитие. Архитектурата на системата е изградена на модулен принцип и обединява средства за измерване, обработка на информацията, вземане на решения, управление на изпълнителните устройства и визуализация на текущото състояние.

За повишаване на надеждността при установяване на пожарно събитие се използва информация от няколко типа измервателни преобразуватели, чрез които се наблюдават основни параметри на защитаваната среда – температура, наличие на дим и концентрация на газове. Получените сигнали се подават към управляващия модул, където се обработват в съответствие със зададения алгоритъм и предварително определени прагови стойности. Използването на повече от един информационен признак позволява да се разграничат нормалното състояние, потенциалната пожарна опасност и потвърденото пожарно събитие.

При установяване на пожар системата определя засегнатата зона и формира управляващи въздействия към съответните изпълнителни устройства. По този начин пожарогасителното въздействие може да бъде насочено към конкретната зона, без да е необходимо едновременно активиране на цялата инсталация. Наред с управлението на спринклерната секция се предвижда активиране на звукова и светлинна сигнализация, както и възможност за обмен на информация със средства за визуализация и мониторинг.

Функционалното взаимодействие между основните измервателни, управляващи и изпълнителни компоненти на разработената автоматизирана система е представено на фиг. 52.



Фиг. 52. Архитектура на разработената автоматизирана система за зонално управление на спринклерна пожарогасителна инсталация

Както се вижда от фиг. 52, архитектурата на разработената автоматизирана система е изградена от взаимосвързани функционални модули, осигуряващи последователното изпълнение на процесите по наблюдение, откриване и потвърждаване на пожарно събитие, определяне на засегнатата зона и формиране на управляващи въздействия към съответните изпълнителни устройства. Модулният принцип позволява отделните функционални части да бъдат разглеждани самостоятелно, като същевременно между тях се осъществява непрекъснат обмен на информация и управляващи сигнали.

- **Сензорен модул.**

Сензорният модул представлява входното информационно ниво на разработената автоматизирана система и има основна задача да осигурява непрекъснато наблюдение на физичните параметри, характеризиращи състоянието на защитавания обект. В разглежданата архитектура се използват три основни групи измервателни средства – температурни, димни и газови датчици. Използването на различни по физичен принцип средства за детекция позволява получаването на комплексна информация за възможното възникване и развитие на пожар. Подобен многосензорен подход се използва при съвременните системи за ранно пожароизвестяване, тъй като съвместната обработка на няколко пожарни признака позволява по-надеждно

разграничаване на реално пожарно събитие от изменения на средата, които могат да предизвикат фалшива тревога [72], [73]. Изследванията върху многосензорното откриване показват, че комбинирането на температура, дим и газови продукти на горенето може да подобри надеждността и бързодействието на детекцията.

Температурните датчици осигуряват непрекъснато измерване на температурата в съответните защитавани зони. Получените стойности се сравняват с предварително зададени прагови нива, като повишаването на температурата над допустимите граници се разглежда като един от основните признаци за възникване на пожарна опасност. Освен абсолютната стойност на температурата, при програмната обработка може да бъде анализирана и динамиката на нейното изменение, тъй като скоростта на нарастване на температурата за определен времеви интервал предоставя допълнителна информация за развитието на наблюдавания процес [72].

Димните датчици се използват за ранно установяване на продукти от горенето в защитаваната среда. Тяхното включване позволява наличието на дим да бъде използвано като самостоятелен информационен признак и едновременно с това като част от комбиниран критерий за оценка на пожарната опасност. При многосензорните системи информацията от димните датчици може да се анализира съвместно с температурата и концентрацията на характерни газови продукти, което подобрява възможността за разграничаване на пожарни и непожарни състояния [72], [73].

Газовите датчици допълват информацията от температурните и димните измервателни средства чрез регистриране на изменения в концентрацията на газове, свързани с процесите на горене. Газовите продукти могат да предоставят информация още в ранните стадии на някои пожарни сценарии, но използването им самостоятелно може да бъде повлияно от други източници в наблюдаваната среда. Поради това комбинирането им с други физични признаци представлява по-подходящ подход за изграждането на надеждна автоматизирана система [72].

Комбинираното използване на трите групи датчици позволява решението за наличие на пожарна опасност да се формира въз основа на съвкупност от независими информационни признаци. В разработената архитектура това създава възможност за реализиране на последователни нива на оценка – нормално състояние, предупредително състояние и потвърдено пожарно събитие. По този начин единичното краткотрайно превишаване на даден параметър не е необходимо автоматично да води до задействане на

пожарогасителната система, а може да бъде подложено на допълнителна логическа проверка чрез останалите наблюдавани параметри [72], [73].

Сензорите се разпределят по отделните защитавани зони, като всеки измервателен сигнал се асоциира със зоната, от която е получен. Това позволява при установяване на опасно състояние управляващата система не само да регистрира наличието на пожарни признаци, но и да определи тяхното пространствено местоположение. Получените измервателни данни се предават към модула за обработка и управление, където се извършват тяхното първично обработване, проверка и подготовка за последващия алгоритъм за вземане на решения.

- **Модул за обработка и управление**

Модулът за обработка и управление представлява основното управляващо ниво в архитектурата на разработената автоматизирана система. Неговото предназначение е да осигурява приемането и първичната обработка на информацията, постъпваща от температурните, димните и газовите датчици, както и обмена на данни с останалите функционални модули. Използването на програмируем контролер позволява в единна хардуерно-програмна среда да бъдат реализирани функциите по събиране на данни, обработка на входните сигнали, диагностика, комуникация и формиране на информация за последващо вземане на управляващи решения [74].

Постъпващите от сензорния модул сигнали се преобразуват във форма, подходяща за програмна обработка, след което се извършва тяхната проверка и оценка. В зависимост от използваните измервателни преобразуватели входната информация може да бъде представена чрез дискретни или аналогови сигнали, както и чрез данни, предавани посредством цифров комуникационен интерфейс. По този начин управляващият контролер формира текущ информационен модел за състоянието на наблюдаваните пожарни зони.

Съществена функция на модула е първичната обработка на измервателните данни. Тя може да включва филтриране на краткотрайни смущения, проверка за допустим диапазон на измерваните стойности, сравнение с предварително зададени прагове и проследяване на изменението на наблюдаваните параметри във времето. Тази обработка е необходима, тъй като единично краткотрайно изменение на даден сензорен сигнал не трябва непременно да се интерпретира като потвърдено пожарно събитие.

Наред с обработката на измервателната информация модулът изпълнява и диагностични функции, свързани с наблюдение на

работоспособността на отделните елементи на системата. При наличие на техническа възможност могат да бъдат регистрирани прекъсване на комуникацията със сензор, недопустима измерена стойност, неизправност на входен или изходен канал и липса на потвърждение за изпълнена управляваща команда. Разграничаването на техническа неизправност от реално пожарно събитие е съществено условие за надеждната работа на автоматизираната система.

Обработената информация се предава към модула за вземане на решения, където се извършва логически анализ на наблюдаваните признаци и се определя текущото състояние на системата. По този начин функциите по първична обработка на измервателните сигнали са логически разграничени от алгоритъма, определящ наличието на пожарна опасност, засегнатата зона и необходимите управляващи действия. Това модулно разделяне улеснява програмната реализация, диагностиката и последващото разширяване на системата.

Управляващият контролер осигурява и необходимия информационен обмен с по-високите нива на системата. Чрез комуникационен интерфейс текущите измервателни стойности, състоянията на зоните, алармените събития и диагностичната информация могат да бъдат предавани към HMI/SCADA среда за визуализация, архивиране и операторски контрол. Интегрирането на PLC-базираното управление със SCADA създава възможност за централизирано наблюдение на пожарогасителния процес и състоянието на отделните компоненти [75].

Следователно модулът за обработка и управление изпълнява ролята на свързващо звено между физическото ниво на измерване и логическото ниво за вземане на решения. Чрез него необработените сензорни сигнали се преобразуват в структурирана информация, която може да бъде използвана за определяне на пожарната опасност, локализиране на засегнатата зона и формиране на последващи управляващи въздействия.

- **Модул за вземане на решения**

Модулът за вземане на решения представлява логическото ядро на разработената автоматизирана система. Неговата основна функция е да извършва комплексна оценка на информацията, постъпваща от сензорния модул след предварителната ѝ обработка, и въз основа на зададени критерии да определя текущото състояние на защитавания обект. За разлика от системите, при които решението за пожарна опасност се формира въз основа на единичен измерван параметър, в разработената архитектура се използва

съвместен анализ на няколко независими признака – температура, наличие на дим и концентрация на газови продукти. Многосензорната обработка позволява да се повиши достоверността на оценката и да се намали влиянието на случайни или краткотрайни изменения на отделен параметър [72], [73].

Процесът на вземане на решение се реализира последователно, като първоначално се анализират текущите измервателни данни и се сравняват с предварително зададени гранични стойности. В зависимост от получената комбинация от сензорни признаци системата може да преминава между различни функционални състояния – нормален режим, предупредителен режим и авариен режим при потвърдено пожарно събитие. По този начин превишаването на единичен параметър не е необходимо непосредствено да предизвиква задействане на пожарогасителната система, а може да активира допълнителна проверка и повишена честота на наблюдение.

За формализиране на процеса на оценка на текущото състояние на защитаваната зона се въвежда логически критерий за наличие на пожарен признак. Критерият се формира въз основа на информацията, получена от температурния, газовия и димния датчик. Наличието на поне един параметър, превишаващ предварително зададената прагова стойност, се разглежда като индикация за потенциална пожарна опасност и се определя съгласно зависимостта:

$$F_i = \left(T_i \geq T_{pr}\right) \vee \left(G_i \geq G_{pr}\right) \vee \left(S_i = 1\right) \qquad (51)$$

където:

- F_i– логически признак за пожарна опасност в зона i;
- T_i– измерена температура в зона i;
- T_{pr}– зададена прагова стойност на температурата;
- G_i– измерена концентрация на наблюдавания газ;
- G_{pr}– зададена прагова концентрация;
- S_i– състояние на димния датчик: $S_i = 1$при наличие на дим и $S_i = 0$при липса на дим;
- $\vee$означава логическо „ИЛИ“.

При $F_i = 0$ нито един от наблюдаваните параметри не показва отклонение, характеризиращо потенциална пожарна опасност, и системата продължава да функционира в нормален режим. При $F_i = 1$е изпълнено поне едно от зададените условия – превишаване на температурния праг, превишаване на праговата концентрация на газ или регистриране на дим. Това

състояние се интерпретира като наличие на потенциален пожарен признак, но не представлява автоматично потвърждение за възникнал пожар.

За да се повиши достоверността на оценката и да се ограничи вероятността от неправилно задействане на пожарогасителната система, следващият етап от алгоритъма отчита броя на едновременно активните и независими пожарни признаци.

$$K_i = I\left(T_i \geq T_{pr}\right) + I\left(G_i \geq G_{pr}\right) + S_i \qquad (5.2)$$

Където $I(\cdot)$е индикаторна функция, която приема стойност **1**, когато съответното условие е изпълнено, и **0**, когато не е изпълнено.

Тоест K_iможе да приема стойности 0, 1, 2 или 3 според броя на едновременно регистрираните пожарни признаци.

Получената стойност на критерия K_iсе използва за определяне на текущия режим на работа на системата за съответната защитавана зона. Разграничаването на отделни режими позволява управляващият алгоритъм да реагира диференцирано в зависимост от броя на едновременно регистрираните пожарни признаци. При липса на такива системата функционира в нормален режим, при наличие на единичен признак преминава в предупредителен режим, а при едновременно регистриране на два или повече независими признака се приема наличие на потвърдено пожарно събитие и се преминава в авариен режим. Логиката за определяне на режима на работа в зона iсе представя със зависимостта:

$$R_i = \begin{cases} R_N, K_i = 0; \\ R_W, K_i = 1; \\ R_A, K_i \geq 2; \end{cases} \qquad (5.3)$$

където:

- R_i– текущият режим на работа за зона i;
- R_N– нормален режим;
- R_W– предупредителен режим;
- R_A– авариен режим (потвърдено пожарно събитие);
- K_i– броят на активните пожарни признаци, определен съгласно зависимост (5.2).

Така при $K_i = 0$ системата работи в нормален режим; при $K_i = 1$ се регистрира единичен пожарен признак и системата преминава в

предупредителен режим; а при $K_i \geq 2$ наличието на два или повече независими пожарни признака се приема като критерий за преминаване в авариен режим.

След определяне на текущия режим на работа на системата се формира управляващо решение за активиране на пожарогасителните устройства в съответната защитавана зона. Задействането се разрешава само при преминаване на системата в авариен режим, съответстващ на потвърдено пожарно събитие. По този начин се предотвратява ненужното активиране на пожарогасителната инсталация при единичен или непотвърден пожарен признак. Условието за формиране на управляващ сигнал към пожарогасителната секция в зона iсе определя съгласно зависимостта:

$$A_i = \begin{cases} 1, R_i = R_A, \\ 0, R_i \neq R_A, \end{cases} \qquad (5.4)$$

където:

- A_i– управляващ сигнал за активиране на пожарогасителната секция в зона i;
- R_i– текущият режим на работа на системата в зона i;
- R_A– авариен режим, съответстващ на потвърдено пожарно събитие;
- $A_i = 1$– подава се команда за активиране на пожарогасителната секция в съответната зона;
- $A_i = 0$– пожарогасителната секция не се активира.

За математическо описание на логиката за вземане на решения в разработената система се въвежда логически критерий за наличие на пожарен признак в съответната зона. Състоянието на зона iсе определя въз основа на измерената температура, концентрацията на газ и сигнала от димния датчик съгласно зависимост (5.1). Броят на едновременно активните пожарни признаци се определя чрез зависимост (5.2), а режимът на работа на системата – чрез зависимост (5.3). При преминаване в авариен режим се формира управляващ сигнал за активиране на съответната пожарогасителна зона съгласно зависимост (5.4).

При липса на активни пожарни признаци системата функционира в нормален режим, като се извършва непрекъснат мониторинг на наблюдаваните параметри. Регистрирането на един признак за пожарна опасност води до преминаване в предупредителен режим. В този режим може да се

активира предварителна сигнализация, да се увеличи честотата на измерване и да се извърши допълнителна проверка на останалите сензорни канали.

При едновременно наличие на два или повече независими пожарни признака събитието може да бъде класифицирано като потвърден пожар, вследствие на което системата преминава в аварийен режим. Формират се команди за задействане на звуковата и светлинната сигнализация, определяне на засегнатата пожарна зона и подготовка за активиране на съответната спринклерна секция. Подобно използване на информация от различни сензорни канали е в съответствие с принципите на многосензорното пожароизвестяване и информационното сливане, при които крайната оценка се формира въз основа на комбинация от независими наблюдавани признаци [72], [73].

- **Модул за зонално управление**

Модулът за зонално управление осъществява функционалната връзка между модула за вземане на решения и изпълнителната част на автоматизираната противопожарна система. Основната му задача е да преобразува формираното управляващо решение в конкретна команда за активиране на пожарогасителните устройства, принадлежащи към зоната, в която е установено и потвърдено пожарно събитие. По този начин се реализира селективно въздействие върху засегнатия участък, без да се налага едновременно задействане на пожарогасителните устройства в останалите защитавани зони.

В разработената архитектура всяка защитавана зона се разглежда като самостоятелна функционална единица, към която са логически асоциирани съответните сензори и изпълнителни устройства. Информацията за зоната, в която е регистрирано пожарното събитие, се определя въз основа на местоположението на активираните сензорни канали. След потвърждаване на пожара модулът за вземане на решения формира управляващ сигнал A_i, определен съгласно зависимост (5.4), който се предава към модула за зонално управление.

При $A_i = 1$ се разрешава активирането на пожарогасителната секция, съответстваща на зона i. Управляващата команда се подава към съответното изпълнително устройство, чрез което се осигурява водоподаването към избраната спринклерна секция. При $A_i = 0$ изпълнителните устройства на съответната зона остават в нормалното си състояние. По този начин се реализира адресно управление, при което всяка пожарогасителна зона може да бъде управлявана независимо от останалите.

Селективното активиране на отделните пожарогасителни секции представлява съществена особеност на зоналния принцип на управление. За разлика от конвенционалното локално термично задействане на отделен спринклер, при автоматизираното управление решението може да бъде формирано въз основа на комплексна информация от няколко сензорни канала.

Това позволява пожарогасителното въздействие да бъде насочено към предварително определена зона в зависимост от резултата от алгоритъма за детекция и локализация на пожарното събитие. Изследванията върху електрически управляваното активиране на спринклерни системи показват възможностите за координирано задействане на избрани спринклерни групи и адаптиране на пожарогасителното въздействие към конкретния пожарен сценарий [63].

- **Обратна връзка и контрол**

Важен елемент на зоналното управление е контролът върху изпълнението на формираните управляващи команди. При наличие на подходящи средства за обратна връзка управляващата система може да следи състоянието на зоналните клапани, наличието на воден поток и други технологични параметри, характеризиращи действителното активиране на пожарогасителната секция. Получената обратна информация се предава към управляващия модул и към системата за визуализация, което позволява да се установи несъответствие между подадена команда и действително изпълнено действие.

При липса на потвърждение за изпълнение на командата може да бъде формиран сигнал за техническа неизправност или аварийно състояние. По този начин обратната връзка създава възможност за разграничаване между логически взето решение за пожарогасене и действително активиране на съответната изпълнителна част. Това е особено важно при автоматизирани системи, при които надеждността на управлението зависи не само от правилното откриване на пожара, но и от гарантираното изпълнение на формираното управляващо въздействие.

- **Взаимодействие между зоните**

При развитие на пожарното събитие алгоритъмът продължава да следи параметрите във всички защитавани зони, включително и след активиране на първоначално засегнатата секция. Ако бъдат изпълнени критериите за потвърдено пожарно събитие и в друга зона, архитектурата позволява формирането на отделна управляваща команда към съответната пожарогасителна секция. По този начин зоналният принцип не ограничава действието на системата само до една предварително определена област, а

позволява последователно или паралелно управление на отделни зони в зависимост от развитието на наблюдавания процес.

След активиране на избраната пожарогасителна зона информацията за текущото състояние на изпълнителните устройства, активната зона и регистрираното пожарно събитие се предава към останалите функционални модули на системата. Това осигурява координирана работа между зоналното управление, изпълнителните устройства, алармената сигнализация и средствата за визуализация и мониторинг. По този начин модулът за зонално управление реализира прехода от логическото решение за наличие и местоположение на пожар към конкретно физическо въздействие върху процеса на пожарогасене.

- **Изпълнителни устройства**

Изпълнителните устройства представляват крайното функционално ниво на разработената автоматизирана система, чрез което формираните от управляващия алгоритъм решения се преобразуват в конкретни физически въздействия върху защитавания обект. Тяхното задействане се извършва в зависимост от установеното състояние на системата, идентифицираната пожарна зона и генерираните управляващи команди. Към тази група се отнасят устройствата за управление на водоподаването към отделните спринклерни секции, както и допълнителни средства за аварийна сигнализация и управление, предвидени в конкретната конфигурация на системата.

Основна роля в изпълнителната част имат електрически управляемите клапани, чрез които се осъществява селективното подаване на вода към съответната пожарогасителна зона. При потвърждаване на пожарно събитие и формиране на сигнал за активиране $A_i = 1$, управляващият модул подава команда към изпълнителното устройство, асоциирано със засегнатата зона. По този начин се осигурява контролирано активиране на необходимата пожарогасителна секция в съответствие с логиката на зоналното управление.

Използването на електрически управлявани изпълнителни устройства създава възможност процесът на пожарогасене да бъде функционално свързан с алгоритъма за многосензорно откриване и локализиране на пожарното събитие. За разлика от изцяло пасивния принцип на действие, при който активирането зависи непосредствено от локалното термично сработване на отделния спринклер, управляваната архитектура позволява решението за задействане да бъде формирано на основата на предварително обработена информация от няколко независими източника. Подобни подходи за

електрически контролирано активиране се разглеждат като възможност за по-гъвкаво управление на пожарогасителното въздействие [63].

След задействане на съответния изпълнителен механизъм се осигурява водоподаване към избраната спринклерна секция. Водата се разпределя посредством спринклерните устройства в защитаваната зона, като основните механизми на пожарогасителното въздействие са свързани с охлаждане на горящите материали, намаляване на температурата на газовата среда и ограничаване на разпространението на пожара. Ефективността на водното пожарогасене зависи от редица параметри, сред които дебитът, налягането, характеристиките на водната струя, размерът и пространственото разпределение на капките, както и условията на развитие на конкретния пожар [48], [49], [51], [64], [68].

- **Звукова и светлинна сигнализация**

Наред с непосредственото управление на пожарогасителните устройства, при преминаване на системата в предупредителен или авариен режим могат да бъдат активирани средства за звукова и светлинна сигнализация. Тяхната функция е да предоставят непосредствена информация за възникналото събитие на намиращите се в защитавания обект лица и на обслужващия персонал. Характерът на сигнализацията може да бъде диференциран в зависимост от текущия режим на системата, като предупредителното състояние и потвърденото пожарно събитие се индикират по различен начин.

При авариен режим сигнализацията функционира едновременно с останалите управляващи действия, което позволява автоматизираното пожарогасене и информирането за възникналата опасност да се осъществяват като части от единен алгоритъм. Информацията за активираните изпълнителни устройства и текущото състояние на системата може едновременно да бъде предавана към HMI/SCADA нивото за визуализация и регистриране на събитията.

При избора и управлението на изпълнителните устройства съществено значение има поведението на системата при прекъсване на електрозахранването, повреда на управляващия контролер или прекъсване на комуникационна връзка. В зависимост от предназначението на конкретното устройство трябва да бъде определено неговото безопасно състояние при възникване на неизправност. Поради това архитектурата на изпълнителното ниво следва да отчита не само нормалното командно управление, но и поведението на системата при отказ на отделни компоненти.

След изпълнение на формираните управляващи команди информацията за състоянието на изпълнителната част се използва за проследяване на развитието на пожарогасителния процес. Сензорният модул продължава непрекъснатото наблюдение на параметрите в защитаваната зона, което позволява да се оцени изменението на температурата, дима и газовите показатели след активиране на пожарогасителната секция. По този начин се реализира затворена информационна последователност **„измерване – оценка – решение – управляващо въздействие – последващо наблюдение“**, която е в основата на функционирането на разработената автоматизирана система.

- **HMI/SCADA и визуализация**

Модулът за HMI/SCADA и визуализация представлява информационното ниво на разработената автоматизирана система, чрез което се осъществява взаимодействието между техническата система и оператора. Основното му предназначение е да предоставя в подходящ графичен вид информация за текущото състояние на наблюдавания обект, измерваните параметри, състоянието на отделните пожарни зони, възникналите предупредителни и аварийни събития и състоянието на управляваните изпълнителни устройства. Използването на средства за визуализация позволява информацията, постъпваща от различните функционални модули, да бъде обединена и представена в единна операторска среда [75].

В процеса на работа данните, получени от температурните, димните и газовите датчици, се обработват от управляващия контролер и могат да бъдат предавани към HMI/SCADA нивото посредством съответния комуникационен интерфейс. По този начин операторът получава информация не само за наличието на пожарно събитие, но и за изменението на наблюдаваните параметри, зоната, в която е регистрирано отклонението, и текущия режим на работа на системата.

Една от основните функции на визуализационния модул е графичното представяне на отделните защитавани зони. За всяка зона може да бъде визуализирано текущото ѝ състояние – нормално, предупредително или аварийно – в съответствие с логиката, определена чрез зависимост (5.3). Това позволява бързо локализиране на участъка, в който е регистрирано опасното събитие, и проследяване на последващите действия на автоматизираната система.

Наред със състоянието на зоните могат да бъдат визуализирани текущите стойности на основните наблюдавани параметри – температура,

концентрация на газ и състояние на димните датчици. Представянето на тези данни в реално време предоставя възможност да се проследи развитието на пожарното събитие както преди, така и след активирането на съответната пожарогасителна секция.

Съществена функция на HMI/SCADA нивото е визуализирането и регистрирането на предупредителни, аварийни и диагностични събития. При преминаване от нормален към предупредителен режим операторът може да бъде информиран за регистриран единичен пожарен признак, без това непременно да означава активиране на пожарогасителната система. При потвърждаване на пожар и преминаване в авариен режим се визуализира съответната пожарна зона и състоянието на задействаните изпълнителни устройства.

Разграничаването на предупредителните и аварийните събития позволява операторът да получава информация, съответстваща на действителната степен на опасност. Освен пожарните събития в системата могат да бъдат регистрирани и технически неизправности, свързани със сензорните канали, комуникационните връзки или изпълнителните устройства, когато съответната диагностична информация е налична.

Интегрирането на системата със SCADA среда създава възможност за регистриране и архивиране на технологични данни и събития. Записването на измерените стойности и моментите на възникване на предупредителни и аварийни състояния позволява последващ анализ на поведението на системата. Архивираната информация може да се използва за оценка на времето за реакция, проследяване на последователността на управляващите действия и анализ на причините за възникнали аварийни ситуации [75].

Историческите данни могат да бъдат представяни под формата на времеви графики, чрез които се проследява изменението на наблюдаваните параметри. Особено значение има съпоставянето на момента на първоначално регистриране на пожарен признак, преминаването в предупредителен и авариен режим, активирането на пожарогасителната секция и последващото изменение на параметрите в защитаваната зона.

В зависимост от конкретната програмна и хардуерна реализация HMI/SCADA интерфейсът може да осигурява и функции за операторско управление, като потвърждаване на алармени съобщения, възстановяване на системата след приключване на аварийно събитие и изпълнение на разрешени команди за ръчно управление. Достъпът до подобни функции следва

да бъде организиран така, че операторските действия да не нарушават автоматичните защитни функции на системата.

Особено значение има разграничаването между автоматичен и ръчен режим на управление. При нормална експлоатация автоматичният алгоритъм изпълнява непрекъснатото наблюдение, оценката на пожарните признаци и формирането на необходимите управляващи въздействия. Ръчните функции могат да се използват при техническо обслужване, изпитване, възстановяване или при необходимост от операторска намеса съобразно приетата логика на системата.

HMI/SCADA модулът изпълнява и важна комуникационна функция, тъй като осигурява връзка между локалното управляващо ниво и средствата за централизирано наблюдение. В зависимост от използваната хардуерна платформа и комуникационна инфраструктура могат да бъдат използвани стандартни индустриални комуникационни интерфейси и протоколи за обмен на технологична информация. Това създава предпоставки разработената система да бъде интегрирана в по-широка инфраструктура за сградна или промишлена автоматизация.

По този начин HMI/SCADA и визуализационният модул допълват автоматичните функции на системата чрез осигуряване на централизирано наблюдение, визуализация на текущите параметри и състояния, регистриране на алармени събития и възможност за последващ анализ на работата. Информационната връзка между управляващия контролер и операторското ниво позволява процесът на пожароизвестяване и пожарогасене да бъде наблюдаван като единна последователност – от първоначалното регистриране на пожарните признаци до активирането на съответната пожарогасителна зона и проследяването на резултата от управляващото въздействие.

5.4. Алгоритъм за управление на автоматизираната противопожарна система

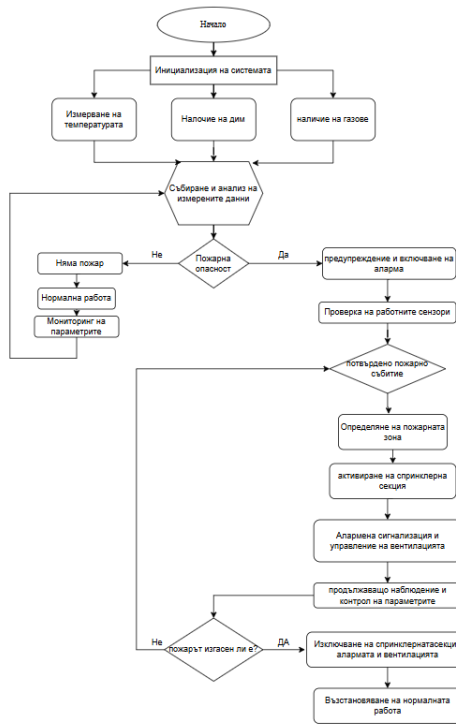
На основата на разработената архитектура е създаден алгоритъм за управление, който определя последователността на операциите при наблюдение на защитавания обект, установяване и потвърждаване на пожарно събитие и активиране на съответната пожарогасителна зона. Алгоритъмът обединява процесите по събиране и обработка на сензорната информация, оценка на текущото състояние, локализиране на възникналата опасност и формиране на управляващи въздействия към изпълнителните устройства.

Основен принцип при разработването на алгоритъма е непрекъснатото циклично наблюдение на параметрите на защитаваната среда.

Информацията от температурните, димните и газовите датчици се анализира с цел установяване на отклонения от нормалното състояние. При регистриране на пожарен признак се извършва допълнителна проверка на постъпващите данни, като решението за активиране на пожарогасителната система се формира след изпълнение на зададените условия за потвърждаване на пожарното събитие.

Алгоритмичната структура е разработена така, че да осигури последователен преход между процесите на **детекция, оценка, потвърждаване, локализация и пожарогасително въздействие**. По този начин се създава възможност за селективно управление на отделните пожарогасителни зони в зависимост от местоположението и развитието на установеното пожарно събитие.

Обобщената последователност на функциониране на разработената автоматизирана система е представена на фиг. 53. Алгоритъмът обхваща целия цикъл на управление – от първоначалната инициализация и непрекъснатото наблюдение на сензорните параметри до активирането на съответната спринклерна секция, контрола върху развитието на пожарното събитие и възстановяването на системата след неговото прекратяване.



Фиг. 53. Обобщен алгоритъм за функциониране на разработената автоматизирана противопожарна система

Както се вижда от фиг. 53, функционирането на системата започва с инициализация на хардуерните и програмните компоненти, при която се установяват началните състояния на входните и изходните канали и се подготвят основните функционални модули за работа. След приключване на инициализацията системата преминава към непрекъснато наблюдение на параметрите на защитаваната среда.

На следващия етап се извършва паралелно получаване на информация за температурата, наличието на дим и наличието или концентрацията на газови продукти, свързани с процеса на горене. Използването на няколко независими информационни признака позволява състоянието на наблюдаваната зона да бъде оценено комплексно, а не само въз основа на единичен

сензорен сигнал. Получените измервателни данни се обединяват и анализират от управляващия алгоритъм в съответствие с предварително зададените критерии за пожарна опасност.

Ако анализираните параметри не показват наличие на пожарна опасност, системата остава в **нормален режим на работа** и продължава цикличното наблюдение на сензорните параметри. По този начин се реализира непрекъснат затворен цикъл на наблюдение, при който състоянието на защитавания обект се оценява през целия период на функциониране на системата.

При регистриране на отклонение, което отговаря на зададените критерии за потенциална пожарна опасност, алгоритъмът преминава към **предупредително състояние**, при което се активира алармена сигнализация и се извършва допълнителна проверка на информацията от работещите сензори. Целта на този етап е да се получи допълнително потвърждение на регистрираното събитие, преди да бъде формирана команда за задействане на пожарогасителната система.

След активирането на предупредителната сигнализация алгоритъмът преминава към проверка за **потвърждаване на пожарното събитие**. На този етап се извършва повторна оценка на информацията от работещите сензори с цел разграничаване на действително възникнал пожар от краткотрайно изменение на наблюдаваните параметри или единично сработване на измервателен канал. Потвърждаването се основава на логическата обработка на регистрираните пожарни признаци в съответствие с критериите, определени чрез зависимости (5.1)–(5.3).

При липса на достатъчно условия за потвърждаване на пожар системата не преминава непосредствено към задействане на пожарогасителните устройства, а продължава наблюдението и анализа на постъпващата информация. По този начин се реализира допълнително ниво на проверка между първоначалното установяване на пожарна опасност и непосредственото пожарогасително въздействие.

При **потвърдено пожарно събитие** следва определяне на засегнатата пожарна зона. Локализирането се извършва въз основа на принадлежността на активираните сензорни канали към съответните защитавани участъци. По този начин управляващият алгоритъм определя не само наличието на пожар, но и зоната, към която трябва да бъде насочено управляващото въздействие.

След определяне на пожарната зона се формира команда за **активиране на съответната спринклерна секция**. Условието за задействане е определено чрез зависимост (5.4), като управляващ сигнал се формира само за

зоната, в която е потвърдено пожарното събитие. Този принцип позволява селективно управление на пожарогасителната система и ограничава ненужното активиране на незасегнатите секции.

Едновременно с активирането на пожарогасителната секция се задействат предвидените средства за **алармена сигнализация**, а при наличие на интегрирана система за управление на вентилацията могат да бъдат формирани и съответните управляващи команди към нея. По този начин действията по пожарогасене, сигнализация и управление на свързаните технически системи се обединяват в общ алгоритъм за реакция при аварийно състояние.

След активирането на пожарогасителните устройства системата продължава непрекъснатото наблюдение на параметрите в защитаваната зона. Този етап има съществено значение, тъй като позволява да се проследи изменението на температурата, наличието на дим и газови продукти след започване на пожарогасителното въздействие. Получените данни се анализират циклично с цел оценка на развитието на пожарното събитие и ефективността на предприетите действия.

Ако наблюдаваните параметри продължават да отговарят на условията за пожарна опасност, пожарогасителната секция остава активна и системата продължава цикъла на наблюдение. По този начин продължителността на управляващото въздействие е свързана със състоянието на наблюдавания процес, а не единствено с първоначалното регистриране на пожара.

При установяване на условия, характеризиращи прекратяване на пожарното събитие, алгоритъмът преминава към процедура за **изключване на активираната спринклерна секция, прекратяване или нулиране на алармената сигнализация и възстановяване на системата към нормален режим на работа**. Преди окончателното възстановяване е необходимо да се отчете състоянието на защитавания обект и готовността на техническите средства за последваща експлоатация.

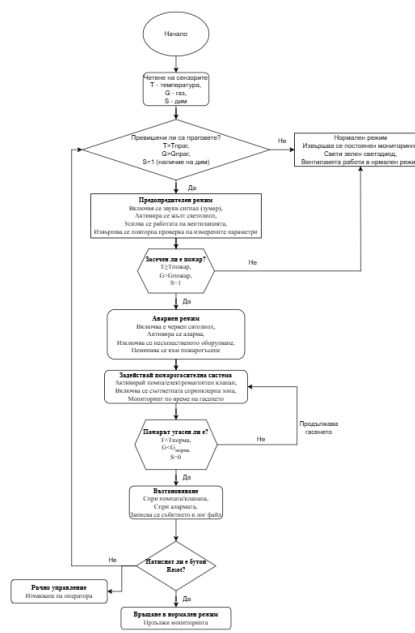
По този начин представеният на фиг. 53 алгоритъм реализира затворен цикъл на управление, включващ последователно **наблюдение, откриване на пожарни признаци, потвърждаване на събитието, локализиране на засегнатата зона, пожарогасително въздействие и последващ контрол на резултата**. Цикличният характер на алгоритъма позволява състоянието на системата да бъде оценявано непрекъснато и управляващите действия да се определят в зависимост от развитието на наблюдавания процес.

Представеният на фиг. 53 обобщен алгоритъм описва основната последователност на функциониране на автоматизираната противопожарна

система, но не представя в детайли логическите условия, определящи преходите между отделните режими на работа. За по-пълно описание на процеса на вземане на решения е разработен детайлизиран алгоритъм, в който са отчетени състоянията на основните сензорни канали, зададените прагови стойности и последователността на управляващите действия при възникване и развитие на пожарно събитие.

В алгоритъма се разграничават три основни функционални състояния – **нормален, предупредителен и авариен режим**. Преходът между тях се определя въз основа на информацията за температурата T, концентрацията на газ Gи наличието на дим S, като логиката съответства на въведените зависимости (5.1)–(5.4). Освен автоматичното преминаване между отделните режими, алгоритъмът включва действията по активиране на пожарогасителната система, наблюдение на развитието на пожара, възстановяване след неговото потушаване и възможност за връщане към нормален режим.

Детайлизираната последователност на процеса на вземане на решения и формиране на управляващите въздействия е представена на фиг. 54.



Фиг. 54. Детайлизиран алгоритъм за вземане на решения и управление на режимите на автоматизираната противопожарна система

Както се вижда от фиг. 54, алгоритъмът започва с циклично прочитане на информацията от основните сензорни канали – температура T, концентрация на газ Gи наличие на дим S. Получените стойности се сравняват със зададените прагови нива, въз основа на което се определя текущото състояние на защитаваната зона.

При липса на превишени прагове системата остава в **нормален режим**, като продължава непрекъснатия мониторинг на параметрите. При установяване на поне един пожарен признак се преминава в **предупредителен режим**, при който се активира звукова и/или светлинна сигнализация и се извършва допълнителна проверка на сензорната информация.

Ако последващият анализ потвърди наличието на пожар, системата преминава в **авариен режим**. Активират се съответните алармени средства, изпълняват се предвидените управляващи действия и се задейства пожарогасителната секция, принадлежаща към установената пожарна зона. По този начин пожарогасителното въздействие се насочва селективно към засегнатия участък.

След активирането на пожарогасителната система наблюдението на параметрите продължава. Ако условията за пожар все още са налице, пожарогасителното въздействие се запазва и алгоритъмът остава в цикъл на контрол. При нормализиране на наблюдаваните параметри се преминава към процедура по **възстановяване**, включваща прекратяване на алармените състояния и подготовка на системата за връщане към нормална работа.

Окончателното връщане към нормален режим се извършва след изпълнение на предвиденото условие за **Reset**, с което се предотвратява автоматичното възстановяване на системата без потвърждение, че пожарното събитие е приключило и защитаваният обект е в безопасно състояние. Предвидената възможност за ръчно управление позволява операторска намеса при обслужване, изпитване или необходимост от контролирано възстановяване на системата.

По този начин алгоритъмът на фиг. 54 реализира последователно управление на състоянията **„нормален режим – предупредителен режим – авариен режим – пожарогасене – възстановяване“**, като преходите между тях се определят въз основа на текущата сензорна информация и зададените логически условия. Това осигурява структурирано реагиране при различни степени на пожарна опасност и намалява вероятността за непосредствено задействане на пожарогасителната система при единичен непотвърден признак.

Изводи към Глава 5

В настоящата глава са разгледани основните принципи на изграждане и функциониране на автоматизираните противопожарни системи, като специално внимание е отделено на автоматичните спринклерни инсталации и възможностите за тяхното интегриране в съвременни компютърни системи за управление. Анализирани са принципът на действие, основните конструктивни разновидности на спринклерите и особеностите на зоналния подход при организацията на пожарогасителния процес.

На основата на извършения анализ е разработена архитектура на автоматизирана система за зонално управление на спринклерна пожарогасителна инсталация. Предложената структура обединява сензорно, управляващо и изпълнително ниво и позволява информацията за състоянието на защитавания обект да бъде използвана за формиране на целенасочени управляващи въздействия. Използването на температурни, димни и газови датчици създава възможност оценката на пожарната опасност да се извършва въз основа на няколко независими информационни признака, вместо решението да зависи единствено от локалното сработване на отделен чувствителен елемент.

Разработен е логически модел за оценка на състоянието на отделните защитавани зони, чрез който се разграничават нормален, предупредителен и авариен режим на работа. Формализирането на процеса чрез зависимости (5.1)–(5.4) позволява последователно да се определят наличието и броят на активните пожарни признаци, текущият режим на системата и условието за активиране на съответната пожарогасителна зона. По този начин се създават предпоставки за ограничаване на неправомерното задействане при наличие само на единичен непотвърден признак.

Зоналният принцип на управление позволява при потвърдено пожарно събитие управляващото въздействие да бъде насочено към конкретния засегнат участък. Това осигурява по-селективно използване на пожарогасителните средства и ограничава необходимостта от активиране на незасегнати секции на системата. Същевременно модулната архитектура създава възможности за разширяване на броя на наблюдаваните зони, включване на допълнителни измервателни средства и интегриране на нови изпълнителни устройства.

Разработените обобщен и детайлизиран алгоритъм определят последователността на функциониране на системата – от непрекъснатото наблюдение и обработката на сензорната информация до потвърждаването и

локализирането на пожарното събитие, активирането на съответната пожарогасителна секция и последващия контрол на състоянието на защитавания обект. Въвеждането на междинен предупредителен режим осигурява допълнително ниво между нормалното състояние и непосредственото пожарогасително въздействие и позволява поетапно формиране на управляващото решение.

Предложената архитектура позволява интеграция с HMI/SCADA системи, чрез които могат да се реализират централизирана визуализация на текущите параметри, наблюдение на състоянието на отделните пожарни зони, регистриране на алармени и аварийни събития и архивиране на технологична информация. Това разширява възможностите за операторски контрол и създава предпоставки за включване на разработеното решение в по-сложни системи за промишлена и сградна автоматизация.

В резултат от проведеното изследване може да се заключи, че съчетаването на многосензорно наблюдение, програмна обработка на информацията, логическо вземане на решения и зонално управление на изпълнителните устройства представлява перспективен подход за развитие на автоматизираните противопожарни системи. Разработеното решение създава основа за изграждане на по-гъвкави системи, при които пожарогасителното въздействие се определя не само от локалното физическо задействане на отделен спринклер, а от комплексна оценка на състоянието на защитавания обект и автоматизирано формиране на управляващо решение.

ЗАКЛЮЧЕНИЕ

В настоящата монография са изследвани възможностите за изграждане на компютърни системи за управление чрез използване на хардуерни и програмни технологии с отворена архитектура, приложими при автоматизацията на различни технологични процеси. Основното внимание е насочено към интегрирането на измервателни средства, управляващи устройства, изпълнителни механизми, комуникационни интерфейси и програмни алгоритми в единна функционална структура, позволяваща адаптиране към различни задачи на автоматизираното управление.

Извършеният теоретичен анализ показа, че развитието на съвременните компютърни системи за управление е свързано с нарастваща интеграция между микропроцесорни и PLC базирани управляващи устройства, индустриални комуникационни технологии, средства за визуализация и мониторинг и концепциите Industry 4.0, Industrial Internet of Things (IIoT) и киберфизични системи. В този контекст технологиите с отворен код предоставят възможности за разработване на гъвкави и разширяеми системи, при които хардуерната и програмната реализация могат да бъдат адаптирани към специфичните изисквания на конкретния технологичен обект.

Въз основа на извършения анализ е разработена модулна архитектура на компютърна система за управление, включваща входен, управляващ, изходен и комуникационен модул, както и средства за визуализация и мониторинг. Модулният принцип позволява отделните функционални компоненти да бъдат адаптирани и разширявани в зависимост от конкретното приложение, без да се променят основните принципи на организация и функциониране на системата.

Като основни хардуерни платформи са разгледани Arduino Uno и Arduino Opta, които представят два различни подхода при изграждането на компютърни системи за управление. Arduino Uno осигурява достъпна микроконтролерна среда, подходяща за обучение, прототипиране и лабораторни изследвания, докато Arduino Opta предоставя PLC ориентирана архитектура, разширени комуникационни възможности и програмни средства, свързани със стандарта IEC 61131-3. Анализът показва, че изборът между двете платформи следва да се извършва в зависимост от функционалните изисквания, необходимите входно-изходни ресурси, комуникационните възможности, условията на експлоатация и изискванията за последващо разширяване на системата.

Практическата приложимост на разгледаните принципи е изследвана чрез разработване на компютърни системи за управление на лабораторен модел на гумено-транспортна лента. Реализирани са варианти, базирани на Arduino Uno и Arduino Opta, което позволява сравнително изследване на различни хардуерни и програмни подходи при управление на един и същ технологичен обект. Разработените реализации демонстрират възможността общите принципи на модулната архитектура да бъдат адаптирани към различни управляващи платформи.

При изследването на електрозадвижването на лабораторния модел е разгледано управлението на скоростта на постояннотоковия двигател и възможността за реализиране на затворен контур с обратна връзка и PID алгоритъм. Получените резултати позволяват да бъде оценено влиянието на алгоритъма за регулиране върху динамичното поведение на системата и показват значението на правилното съчетаване на измервателната, управляващата и изпълнителната част при изграждането на системи за автоматично регулиране.

Сравнителното изследване на Arduino Uno и Arduino Opta показва, че възможността за реализиране на един и същ основен алгоритъм за управление не означава функционална еквивалентност на хардуерните платформи по отношение на тяхната цялостна приложимост. Arduino Uno представлява рационално решение за нискобюджетни лабораторни, учебни и прототипни системи, докато Arduino Opta предоставя допълнителни възможности за изграждане на PLC ориентирани системи, комуникационна интеграция и включване в по-сложни архитектури за автоматизация и мониторинг.

Приложимостта на разработените принципи е разгледана и при автоматизирана система за управление на зонална спринклерна пожарогасителна система. По този начин предложената концепция е приложена към обект с различно функционално предназначение и с повишени изисквания към безопасността. Разработената система използва многосензорно наблюдение и алгоритмична обработка на информацията за определяне на текущото състояние на наблюдавания обект и формиране на съответните управляващи въздействия.

Предложеният зонален принцип позволява защитаваното пространство да бъде разделено на отделни контролирани участъци, за които да се извършва независимо наблюдение и да се формират селективни управляващи въздействия. Разграничаването на нормален, предупредителен и аварийен режим създава възможност реакцията на системата да бъде съобразена

с установеното състояние и с резултатите от обработката на информацията от използваните измервателни канали. По този начин се реализира последователността „измерване – обработка – оценка – решение – управление“, характерна за компютърните системи за автоматично управление.

Резултатите от разработените приложения показват, че ефективността на една компютърна система за управление се определя не само от характеристиките на използвания контролер, а от системното взаимодействие между хардуерната архитектура, програмното осигуряване, измервателните средства, алгоритмите за управление, комуникационната инфраструктура и изпълнителните механизми. Именно модулното организиране на тези компоненти създава възможност една обща архитектурна концепция да бъде адаптирана към технологични обекти с различно предназначение.

Използването на технологии с отворена архитектура предоставя допълнителни възможности за разработване, модифициране и интегриране на хардуерни и програмни компоненти при изграждането на компютърни системи за управление. Съчетаването на достъпни микроконтролерни платформи, PLC ориентирани устройства, стандартизирани програмни средства и индустриални комуникационни технологии създава предпоставки за разработване на решения с различна степен на сложност – от лабораторни и учебно-изследователски системи до приложения, ориентирани към индустриалната автоматизация.

Като перспективни направления за развитие могат да бъдат определени разширяването на сензорната и комуникационната инфраструктура, интегрирането на допълнителни средства за HMI/SCADA визуализация и мониторинг, разработването на методи за дистанционна диагностика, използването на натрупани експлоатационни данни за оценка на състоянието на технологичните обекти и прилагането на адаптивни и интелигентни алгоритми за управление. Перспективно направление представлява и последващото експериментално изследване на разработените архитектури в условия, максимално близки до реалната промишлена експлоатация.

В обобщение може да се приеме, че поставената цел на монографията е постигната чрез теоретично изследване, разработване на обща модулна архитектура и практическо приложение на компютърни системи за управление, базирани на технологии с отворена архитектура. Разработените хардуерни, програмни и алгоритмични решения демонстрират възможността за прилагане на общ подход при автоматизацията на обекти с различно функционално предназначение. Получените резултати и предложените решения

създават основа за последващи изследвания и разработки в областта на компютърното управление, индустриалната автоматизация, мониторинга и автоматизираните системи за безопасност.

ОСНОВНИ НАУЧНО-ПРИЛОЖНИ И ПРИЛОЖНИ ПРИНОСИ

В резултат от проведените теоретични и експериментални изследвания, разработените хардуерни и програмни решения и извършения сравнителен анализ могат да бъдат формулирани следните основни научно-приложни и приложни приноси:

I. НАУЧНО-ПРИЛОЖНИ ПРИНОСИ

1. Предложена е модулна архитектура за изграждане на компютърни системи за управление, базирани на хардуерни и програмни технологии с отворена архитектура. Разработеният подход структурира системата във входен, управляващ, изходен и комуникационен модул и средства за визуализация и мониторинг, което позволява адаптиране на общата архитектура към различни технологични обекти чрез промяна на периферните устройства и управляващите алгоритми. Приложимостта на подхода е изследвана чрез реализации, базирани на Arduino Uno и Arduino Opta.

2. Извършен е сравнителен анализ на възможностите на Arduino Uno и Arduino Opta при изграждането на компютърни системи за управление. Съпоставени са особеностите на хардуерната и програмната реализация, входно-изходните ресурси, комуникационните възможности, средствата за интеграция и областите на приложение. На тази основа са систематизирани критерии за избор на управляваща платформа в зависимост от функционалната сложност, комуникационните изисквания, необходимостта от мониторинг и условията на приложение.

3. Разработен е логически модел за оценка на пожарната опасност и определяне на функционалното състояние на автоматизирана зонална противопожарна система. Моделът използва комбинирана информация от няколко измервателни канала и формализира определянето на нормален, предупредителен и авариен режим. Предложеният подход позволява управляващото решение да се формира въз основа на съвместна оценка и потвърждаване на наблюдаваните признаци, а не единствено въз основа на единичен измервателен сигнал.

4. Разработен е алгоритмичен подход за функциониране и управление на автоматизирана зонална противопожарна система, основан на

последователността „измерване – обработка – оценка – решение – управление“. Алгоритъмът обхваща наблюдение и обработка на сензорната информация, установяване и потвърждаване на аварийно състояние, определяне на засегнатата зона, формиране на управляващи въздействия и последващо наблюдение на състоянието на системата. Модулната организация създава възможност за адаптиране и разширяване към допълнителни контролирани зони.

II. ПРИЛОЖНИ ПРИНОСИ

1. Разработена и експериментално изследвана е компютърна система за управление на лабораторен модел на гумено-транспортна лента, обединяваща управляваща платформа, измервателни средства, електрозадвижване, изпълнителни устройства и програмно осигуряване. Създадената експериментална постановка позволява практическо изследване на алгоритми за управление и сравнително оценяване на различни хардуерни реализации при управление на един и същ технологичен обект.

2. Реализиран и експериментално изследван е алгоритъм за регулиране на скоростта на постояннотоково електрозадвижване с използване на обратна връзка и PID регулатор. Чрез изследване на динамичното поведение на системата при различни режими на управление е създадена практическа основа за оценяване на влиянието на алгоритъма за регулиране и за сравнително изследване на неговата реализация върху различни управляващи платформи.

3. Разработена и практически реализирана е архитектура на автоматизирана система за зонално управление на пожарогасителна система, интегрираща многосензорно наблюдение, програмна обработка на информацията, управляващ контролер и изпълнителни устройства. Реализацията позволява определяне на състоянието на наблюдаваните зони и селективно формиране на управляващо въздействие към съответната пожарогасителна секция.

4. Създадена е методическа и приложна основа за разработване и изследване на компютърни системи за управление с използване на микроконтролерни и PLC ориентирани платформи. Разработените хардуерни, програмни и алгоритмични решения могат да бъдат използвани при изграждането на учебни и лабораторни системи, експериментални стендове и приложни разработки и позволяват последващо разширяване със средства

за HMI/SCADA визуализация, индустриални комуникации и дистанционен мониторинг.

Литература:

[1]Xu, L. D., He, W., Li, S. *Internet of Things in Industries: A Survey*. IEEE Transactions on Industrial Informatics, 10(4), 2233–2243, 2014.
[2] IBM, “What is open source software?”, IBM Think, 2024. [Online]. Available: https://www.ibm.com/think/topics/open-source
[3] Open Source Initiative, “Frequently Asked Questions,” OSI, 2024. [Online]. Available: https://opensource.org/faq
[4] D. Riehle, “Free and Open Source Software,” IEEE Computer, vol. 57, no. 8, pp. 114–118, 2024. doi: 10.1109/MC.2024.3407268
[5] K. Miller, J. Voas, and T. Costello, “Free and Open Source Software,” IEEE IT Professional, 2010. doi: 10.1109/MITP.2010.147
[6] Bennett, S., *A History of Control Engineering 1800–1930*, IET, London, 1979.
[7] Open Source Hardware Association (OSHWA). *Open Source Hardware Definition*. Available at: https://oshwa.org/resources/open-source-hardware-definition/
[8] CERN. CERN Open Hardware Licence (CERN-OHL). Available: https://cern-ohl.web.cern.ch/
[9] Bonvoisin, J., Molloy, J., Haeuer, M., Wenzel, T. Standardisation of practices in Open-Source Hardware. (2020), DOI: 10.5334/joh.22, https://scite.ai/journals/journal-of-open-hardware-mObxy
[10]Arduino. Arduino Opta Documentation. Available: https://docs.arduino.cc/hardware/opta
[11] IEC. *IEC 61131-3:2013. Programmable Controllers – Part 3: Programming Languages*. International Electrotechnical Commission (IEC), Geneva, Switzerland, 2013.
[12] Open Source Initiative (OSI). *The Open Source Definition*. Available at: https://opensource.org/osd
[13] Open Source Initiative (OSI). *Open Source Licenses*. Available at: https://opensource.org/licenses
[14] Upadhyay, D., & Sampalli, S. (2020). SCADA (Supervisory Control and Data Acquisition) systems: Vulnerability assessment and security recommendations. *Computers & Security, 89*, Article 101666. https://doi.org/10.1016/j.cose.2019.101666
[15] Boyer, S. A. (2004). *SCADA: Supervisory control and data acquisition* (3rd ed.). ISA—The Instrumentation, Systems, and Automation Society.
[16] Bolton, W. (2015). *Programmable Logic Controllers* (6th ed.). Newnes, Elsevier. https://doi.org/10.1016/C2014-0-03884-1

[17] Dzharov, V. (2025). Emergency mode of control of rubber belt conveyors via PLC system Arduino Opta. *International Journal of Recent Technology and Applied Science, 7*(2), 104–116. https://doi.org/10.36079/lamintang.ijortas-0702.914

[18] Antoniak, J. (2007). *Przenośniki taśmowe w górnictwie podziemnym i odkrywkowym* (3rd ed.). Wydawnictwo Politechniki Śląskiej. ISBN 978-83-7335-396-1.

[19] Antoniak, J. (n.d.). *Urządzenia i systemy transportu podziemnego w kopalniach*. Śląsk Publishers. ISBN 9321607098.

[20] Bajda, M., & Hardygóra, M. (2019). Laboratory tests of operational durability and energy-efficiency of conveyor belts. *IOP Conference Series: Earth and Environmental Science, 261*, 012002. https://doi.org/10.1088/1755-1315/261/1/012002

[21] Bajda, M., & Hardygóra, M. (2021). Analysis of the influence of the type of belt on the energy consumption of transport processes in a belt conveyor. *Energies, 14*, 6180.

[22] Bugaric, U., Tanasijević, M., Polovina, D., Ignjatović, D., & Jovančić, P. (2014). Reliability of rubber conveyor belts as a part of the overburden removal system—Case study: Tamnava-East Field Open Cast Mine. *Technical Gazette, 21*, 925–932.

[23] Guo, X., Liu, X., Zhou, H., Stanislawski, R., Królczyk, G., & Li, Z. (2022). Belt tear detection for coal mining conveyors. *Micromachines, 13*(3), 449. https://doi.org/10.3390/mi13030449

[24]Jonak, J., & Gajewski, J. (2006). Operating diagnostics and monitoring issues of selected mining belt conveyors. *Eksploatacja i Niezawodność, 4*, 74–78.

[25] Kostov, G., & Александров, Р. (2024). Структура на система на електрозадвижване с превключваем реактивен двигател. *Годишник на Минно-геоложкия университет „Св. Иван Рилски“, 67*, 228–235. https://doi.org/10.5281/zenodo.13762846

[26] Krol, R., Kawalec, W., & Gladysiewicz, L. (2017). An effective conveyor belt for underground transportation systems. *IOP Conference Series: Earth and Environmental Science, 95*, 042047.

[27] Palmero-González, M. A., Batuecas, E., Fernández-Torrijos, M., & Marugán-Cruz, C. (2025). Environmental comparison of dry and hybrid condensing systems in concentrating solar power tower plants. *Energy Conversion and Management, 340*, 119978. https://doi.org/10.1016/j.enconman.2025.119978

[28] Rzeszowska, A., Jurdziak, L., Błazej, R., & Lewandowicz, P. (2025). Analysis of uncertainty in conveyor belt condition assessment using time-based indicators. *Applied Sciences, 15*, 7939. https://doi.org/10.3390/app15147939
[29] Salawu, G., Bright, G., & Onunka, C. (2020). Modelling and simulation of a conveyor belt system for optimal productivity. *International Journal of Mechanical Engineering and Technology, 11*(1), 115–121.
[30] Zabchev, A., & Alexandrov, R. (2025). Determination of the resistances in the rotor's coils of a mine locomotive motor. *Sustainable Extraction and Processing of Raw Materials Journal, 6*. https://doi.org/10.58903/dw18921671
[31] Шейретов, Х. (2016). Изследване влиянието на различни фактори върху натоварването на ролките на лентови транспортьори. Годишник на Минно-геоложкия университет „Св. Иван Рилски“, Том 59, Св. III.
[32] Джаров, В. Сравнителен анализ на системи за управление на лабораторен модел на гумено-транспортна лента, базирани на Arduino uno и plc Arduino opta. Годишник на Минно-геоложкия университет „Св. Иван Рилски“, 2026.
[33] Kulinowski, P. M. (2013). *Analytical Method of Designing and Selecting Take-Up Systems for Mining Belt Conveyors.* Archives of Mining Sciences, 58(4), 1301–1315. https://doi.org/10.2478/amsc-2013-0090
[34] Dai, Y., Shi, N., Lu, Y., & He, Y. (2011). *Analysis and determination of tensioning device of belt conveyor.* Proceedings of ICECENG 2011. DOI: 10.1109/ICECENG.2011.6057384.
[35] *PWM Control of a DC Motor Used to Drive a Conveyor Belt*. Procedia Engineering, Vol. 100, 2015, pp. 299–304. https://doi.org/10.1016/j.proeng.2015.01.371
[36] IFAC-PapersOnLine. *Low Cost Laboratory Plant for Control System Education*, 2018. https://doi.org/10.1016/j.ifacol.2018.07.168
[37] Uyanik, I. et al. *A Low-Cost Feedback Control Systems Laboratory Setup via Arduino–Simulink Interface*. Computer Applications in Engineering Education, 2018. https://doi.org/10.1002/cae.21917
[38] Barrett, S. F. (2025). *Arduino PLC IDE and Ladder Logic*. In Arduino VII (Lecture Notes in Electrical Engineering). Springer. DOI: 10.1007/978-3-031-68609-2_3.
[39] Sidorenko, A., Rezapour, M., Wagner, A., & Ruskowski, M. (2024). *Towards Using Behavior Trees in Industrial Automation Controllers*. arXiv.
[40] Dantas, P., Cordeiro, L., & Junior, W. (2026). *ESBMC-Arduino: Closing the Deployment Gap for Formal Verification of Open-Hardware PLCs*. arXiv.

[41] Watterson, C., Leonardi, A., & Paterniani, G. *Arduino Platform with Analog Devices Technology for Flexible Industrial Control*. Analog Dialogue, Analog Devices.
[42] *From Analog to Digital – Successful Implementation of IoT Solutions in the Petrochemical Industry*. 2024.
[43] Джаров, В. Автоматизирана система за управление на зонална спринклерна пожарогасителна система. Годишник на Минно-геоложкия университет „Св. Иван Рилски“, 2026.
[44] Е. Кърцелин, Р. Александров, Ц. Върбов, „Използване на RCD устройствата като противопожарни средства“, Proceedings of National scientific and technical conference with international participation “Automation in mining industry and metallurgy”, BULCAMC’20, 19-20 November 2020;
[45] D. Makedonska, B. Vladkova, N. Kostadinova, Z. Dinchev and B. Valev, *Design and Construction of a Stand for Sprinkler Nozzle Testing*, Annual of the University of Mining and Geology "St. Ivan Rilski", Vol. 68, 2025.
[46] БДС EN 54-1 – БДС EN 54-31. *Пожароизвестителни системи. Съставни части на пожароизвестителни системи и системи за управление на дим и топлина*. Български институт за стандартизация.
[47] K. Gurung, S. R. Maharjan, S. Pokhrel, Y. Sigdel and B. Bhatta, "Fire Detection and Extinguishing System," *International Journal on Engineering Technology (InJET)*, vol. 2, no. 1, pp. 60–70, 2024.
[48] БДС EN 12259-1:1999+A1:2001+A3:2006. *Стационарни пожарогасителни инсталации. Съставни части на спринклери и инсталации за разпръскване на вода. Част 1: Спринклери.*
[49] БДС EN 12845:2015+A1:2020. *Стационарни пожарогасителни инсталации. Автоматични спринклерни инсталации. Проектиране, монтиране и поддържане.*
[50] БДС EN 671-1:2002. *Стационарни противопожарни инсталации. Инсталации с маркуч. Част 1: Макари с полутвърд маркуч.*
[51] NFPA 13. Standard for the Installation of Sprinkler Systems. National Fire Protection Association, USA.
[52] European Committee for Standardization, *EN 54-14 Fire Detection and Fire Alarm Systems – Guidelines for Planning, Design, Installation, Commissioning, Use and Maintenance*, 2018.
[53] N. A. Sathiabalan et al., "Autonomous Robotic Fire Detection and Extinguishing System," *Journal of Physics: Conference Series*, vol. 2107, 2021.

[54] Hamood Alqourabah , Amgad Muneer , Suliman Mohamed Fati, “A smart fire detection system using IoT technology with automatic water sprinkler”, International Journal of Electrical and Computer Engineering (IJECE), Vol. 11, No. 4, August 2021, pp. 2994~3002, ISSN: 2088-8708, DOI: 10.11591/ijece.v11i4.pp2994-3002

[55] Milanoski, Mile, “ТСМВ112 Инфраструктурно изграждане на мрежи”, 2021/01/25, DOI: 10.13140/RG.2.2.28625.15205.

[56] S. Basu, S. Pramanik, S. Dey, G. Panigrahi and D. K. Jana, "Fire Monitoring in Coal Mines Using Wireless Underground Sensor Network and Interval Type-2 Fuzzy Logic Controller," *International Journal of Coal Science & Technology*, 2019.

[57] K. Saxena, N. B. Kalyan, A. Aswath et al., "Real-Time Monitoring of Underground Mines with IoT and Machine Learning Integration," *Journal of Mines, Metals and Fuels*, vol. 73, no. 6, pp. 1559–1568, 2025.

[58] A Comprehensive Review and Bibliometric Analysis of IoT-Based Fire Safety Systems AlQahtani A.A.S. et al., *Fire*, Vol. 11, No. 2, 2025.

[59] E. F. Raquin, A. P. Ramirez, K. R. Meredor, A. M. Francisco, K. N. Guillena and B. C. Fabro, *Automated Fire Suppression System*, International Journal of Research and Scientific Innovation (IJRSI), Vol. XIII, Issue I, pp. 661–674, 2026, DOI: 10.51244/IJRSI.2026.13010057.

[60] Viking Group Inc. *VK202 Micromatic® Standard Response Pendent Sprinkler (K8.0). Technical Data Sheet.*, Viking Group Inc., USA.

[61] Madrzykowski, D., Vettori, R. L. *A Sprinkler Fire Suppression Algorithm for the GSA Engineering Fire Assessment System.* NISTIR 4833, National Institute of Standards and Technology, Gaithersburg, MD, 1992. DOI: 10.6028/NIST.IR.4833.

[62] Patterson, N. Y., Madrzykowski, D. *Review of Residential Sprinkler Systems: Research and Standards.* NISTIR 6941, National Institute of Standards and Technology, Gaithersburg, MD, 2002.

[63] Tanklevskiy, L., Tsoy, A., Snegirev, A. “Electrically Controlled Dynamic Sprinkler Activation: Computational Assessment of Potential Efficiency.” *Fire Safety Journal*, Vol. 91, 2017, pp. 614–623. DOI: 10.1016/j.firesaf.2017.04.019.

[64] Vetter, M., Dinkov, I., Schelb, D., Trimis, D. “Experimental Study on the Effects of Sprinkler Activation on the Stratified Smoke Layer Inside a 36 m² Enclosure.” *Fire Safety Journal*, Vol. 121, 2021, Article 103313. DOI: 10.1016/j.firesaf.2021.103313.

[65] Frank, K., Spearpoint, M., Baker, G., Wade, C., Fleischmann, C. “Measuring Modified Glass Bulb Sprinkler Thermal Response in Plunge and Compartment Fire Experiments.” *Fire Safety Journal*, Vol. 91, 2017, pp. 662–670. DOI: 10.1016/j.firesaf.2017.03.017.
[66] Heskestad, G., Bill, R. G. “Quantification of Thermal Responsiveness of Automatic Sprinklers Including Conduction Effects.” *Fire Safety Journal*, Vol. 14, 1988, pp. 113–125
[67] Ingason, H. “Investigation of Thermal Response of Glass Bulb Sprinklers.” *Fire Safety Journal*, 1998. DOI: 10.1016/S0379-7112(97)00026-X.
[68] Hua, J., Kumar, K., Khoo, B. C. “A Numerical Study of the Interaction of Water Spray with a Fire Plume.” *Fire Safety Journal*, 2002. DOI: 10.1016/S0379-7112(02)00026-7.
[69] Ruffino, P., et al. “The Simulation of Fire Sprinklers Thermal Response in Compartment Fires.” *Fire Safety Journal*, 2004.
[70] Yao, C. “The Development of the ESFR Sprinkler System.” *Fire Safety Journal*, 1988
[71] Lai, C. M., et al. “Influence of Fire Source Locations on the Actuation of Wet-Pipe Sprinkler Systems.” *Building and Environment*, 2010.
[72] L. Wu, L. Chen, X. Hao, “Multi-Sensor Data Fusion Algorithm for Indoor Fire Early Warning Based on BP Neural Network,” *Information*, Vol. 12, No. 2, Article 59, 2021. DOI: 10.3390/info12020059.
[73] Q. Ding, Z. Peng, T. Liu, Q. Tong, “Multi-Sensor Building Fire Alarm System with Information Fusion Technology Based on D-S Evidence Theory,” *Algorithms*, Vol. 7, No. 4, pp. 523–537, 2014. DOI: 10.3390/a7040523.
[74] Su, L. et al., “Using Edge Computing Technology in Programmable Logic Controllers for Intelligent Industrial Safety and Fire Protection Systems,” *Sensors and Materials*, 2023.
[75] Chamorro-Atalaya, O. et al., “Supervision and Control by SCADA of an Automated Fire Fighting System,” 2021

Володя Владимиров Джаров

Компютърни системи за управление с отворен код, приложими в промишлените предприятия

Минно-геоложки университет „Св. Иван Рилски“

Българска, първо издание

Рецензенти: доц. д-р Мила Илиева Илиева–Обретенова
доц. д-р Николай Лазаров Лаков

Научен редактор: доц. д-р Ромео Фердинандов Александров

ISBN 978-954-353-518-7 (мека подвързия)
ISBN 978-954-353-519-4 (pdf)

© Володя Владимиров Джаров, 2026

Формат: 60/90/16
Печатни коли: 10,5

Издателска къща „Св. Иван Рилски“ на МГУ „Св. Иван Рилски“
София, 2026

МИННО-ГЕОЛОЖКИ
УНИВЕРСИТЕТ
„СВ. ИВАН РИЛСКИ“

Отпечатано в Издателска къща „Св. Иван Рилски“
на МГУ „Св. Иван Рилски“, София.

mgu.bg