

Міністерство освіти і науки України

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

І.М. Бондаренко, О.В. Бородін, В.П. Карнаушенко

МІКРОПРОЦЕСОРНІ СИСТЕМИ КОНТРОЛЮ ТА КЕРУВАННЯ

Навчальний посібник
для студентів спеціальностей
«Мікро – та наносистемна техніка», «Електроніка»

РЕКОМЕНДОВАНО
Вченою радою університету.
Протокол № 7/6 від 31.05.2019р.

Харків 2020

Бондаренко І.М., Бородін О.В., Карнаушенко В.П. Мікропроцесорні системи контролю та керування: Навч. посібник для студентів ЗВО. – Харків: ХНУРЕ. – 2020. – 244 с.

Викладено основні принципи побудови та функціонування автоматизованих систем керування та контролю на базі мікропроцесорної техніки, сучасні та перспективні напрями розвитку засобів автоматизації технологічних процесів в промисловості. Розглянуті основні типи засобів керування та контролю, алгоритми їх роботи. Визначені базові варіанти реалізації автоматизованих систем та сучасна компонентна база, що застосовується при конструюванні засобів контролю та керування, етапи та складові частини процесу проектування із застосуванням мікропроцесорів. Визначений порядок та рівні проектування, основні підходи до реалізації проектів.

Розглянуті особливості застосування мікропроцесорних систем в засобах автоматизації та наведені приклади реалізації систем автоматичного регулювання. Розглянуті засоби автоматизованого проектування і напрями удосконалення та розвитку засобів контролю та керування.

Рекомендовано як навчальний посібник для студентів усіх форм навчання спеціальностей 153 «Мікро – та наносистемна техніка», 171 «Електроніка».

Рецензенти:

Обод І.І., доктор технічних наук, професор кафедри МТС, лауреат Державної премії України;

О.Ю. Панченко, д-р. фіз.-мат. наук, професор, завідувач кафедри ПЕЕА ХНУРЕ.

Іл. 184 . Бібліогр. 17 назв.

Бондаренко І.М.
Бородін О.В.
Карнаушенко В.П., 2020

ЗМІСТ

ВСТУП	5
1. АВТОМАТИЗОВАНЕ УПРАВЛІННЯ ПРОМИСЛОВИМИ ОБ'ЄКТАМИ	7
1.1 Системи автоматичного управління	8
1.2 Склад і класифікація систем автоматичного регулювання	10
1.3 Приклади систем автоматичного регулювання в процесі формування структур інтегральних схем	14
1.4 Контрольні запитання і завдання	16
2 ТИПИ РЕГУЛЯТОРІВ І НАЙПОШИРЕНІ ЗАКОНИ РЕГУЛЮВАННЯ	18
2.1 Закони регулювання	18
2.2 Пропорційно-інтегральний закон регулювання	20
2.3 Пропорційно-інтегрально-диференціальний закон регулювання	22
2.4 Контрольні запитання і завдання	24
2.5 Короткий тест	25
3 ЗАСТОСУВАННЯ МІКРОПРОЦЕСОРІВ В СИСТЕМАХ АВТОМАТИЧНОГО КЕРУВАННЯ	31
3.1 Типова мікропроцесорна система керування	31
3.2 Структура програмного забезпечення	33
3.3 Початковий етап проектування	35
3.4 Приклади застосування мікропроцесорних систем контролю та керування	37
3.5 Програмно-апаратні засоби сполучення з об'єктом	45
3.6 Контрольні запитання і завдання	47
4 РЕАЛІЗАЦІЯ ЗАСОБІВ КЕРУВАННЯ НА МІКРОПРОЦЕСОРАХ SITARA	49
4.1 Модулі на основі мікропроцесорів Sitara	56
4.2 Програмні засоби розробки для МП Sitara	64
4.2.1 Середовище розробки Code Composer Studio	65
4.3 Контрольні запитання і завдання	75
5 ЗАСТОСУВАННЯ МІКРОКОНТРОЛЕРІВ В СИСТЕМАХ КОНТРОЛЮ ТА КЕРУВАННЯ	76
5.1 Загальні характеристики сімейства	77
5.2 Вбудовані інтерфейси комунікації	78
5.3 Структура периферії	80
5.4 Короткий огляд ЦПП Cortex в мікроконтролерах	88
5.5 Програмні засоби розробки для мікроконтролерів	88
5.6 Контрольні запитання і завдання	100
6 ПРОГРАМОВАНІ ЛОГІЧНІ КОНТРОЛЕРИ В ЗАСОБАХ АВТОМАТИЗОВАНОГО УПРАВЛІННЯ	102

6.1 Програмування ПЛК	110
6.2 Огляд програмованих логічних контролерів фірми SIEMENS	118
6.3 Контрольні запитання і завдання	125
7 ПРОГРАМОВАНІ СИСТЕМИ НА КРИСТАЛІ В СИСТЕМАХ УПРАВЛІННЯ	126
7.1 Мікросхеми типу система на кристалі фірми Cypress першої генерації	126
7.2 Основні характеристики сімейства PSoC®3	128
7.3 Мікросхеми сімейства PSoC® 4	134
7.4 Сімейство PSoC® 5LP	141
7.5 Сімейство PSoC®6	149
7.6 Середовище розробки проектів PSoC® Creator™	151
7.7 Приклади розробки елементів систем на PSoC	158
7.8 Контрольні запитання і завдання	168
8 ЗАСТОСУВАННЯ ПРОГРАМОВАНИХ ЛОГІЧНИХ ІНТЕГРАЛЬНИХ СХЕМ ДЛЯ РОЗРОБКИ СИСТЕМ КЕРУВАННЯ	169
8.1 Архітектура ПЛІС. Логічні елементи	174
8.2 Базовий логічний елемент ПЛІС Cyclone IV	175
8.3 Приклад розробки елементу системи керування на ПЛІС	180
8.4 ПЛІС типу «програмована система на кристалі»	192
8.5 Контрольні запитання і завдання	210
9 МЕРЕЖИВІ ПРОТОКОЛИ ІНДУСТРІЇ 4.0	212
9.1 Розподілені системи автоматизації	213
9.2 Багаторівнева мережива архітектура	216
9.3. Загальні відомості про промислові мережі	220
9.4. Модель взаємодії відкритих систем OSI	223
9.5. Інтерфейси RS-485, RS-422 і RS-232	226
9.6 Інтерфейси RS-232 і RS-422	228
9.7 Інтерфейс "струмова петля"	229
9.8 Магістральний протокол HART	232
9.9 CAN протокол	235
9.10 Мережа Profibus	236
9.11. Протокол Modbus	238
9.12 Промисловий Ethernet	241
9.13 Контрольні запитання і завдання	245
РЕКОМЕНДОВАНА ЛІТЕРАТУРА	246

ВСТУП

Одним із важливих досягнень мікроелектроніки та обчислювальної техніки є створення аналогових і цифрових інтегральних схем та мікропроцесорів. На їх основі на сьогодні створено велику кількість цифрових систем різного призначення, які можна в першому наближенні розділити на такі класи: цифрові системи передачі та обробки даних, цифрові інформаційно – вимірювальні системи та цифрові системи контролю та керування. Основну частину роботи в системах управління і автоматики зі збирання, зберігання та опрацювання інформації, необхідної для формування керівних дій в обмежений проміжок часу і їх реалізації, виконують мікропроцесорні пристрої і інші засоби цифрової, вимірювальної та регулюючої техніки. Тому мікропроцесорні системи (МПС) різного призначення становлять собою базу не тільки існуючих, але й перспективних вискоефективних автоматичних і автоматизованих систем керування окремими технологічними процесами і виробництвом взагалі, систем автоматизації пошуку інформації, наукових досліджень та ін. Найбільшого поширення в останній час набули системи управління, ядром яких є спеціалізовані мікропроцесори або мікроконтролери.

Сучасні мікропроцесорні засоби автоматизації представлені трьома основними групами: індустріальні комп'ютери, промислові контролери, однокристальні мікропроцесори і мікроконтролери.

Індустріальні комп'ютери побудовані на базі архітектури і програмних можливостей персональних комп'ютерів. Взятий за основу персональний комп'ютер просто адаптується до конкретних умов завдання управління. Він доповнюється необхідними інтерфейсами сполучення з керованим об'єктом, його конструкція модернізується під жорсткі виробничі умови. Фірми HP, Advantech, Agilent та інші пропонують великий набір стандартних засобів сполучення комп'ютерів з об'єктами контролю. Як засоби автоматизації індустріальні комп'ютери універсальні і здатні вирішувати практично будь-які за складністю завдання. Але при цьому для багатьох завдань вони надмірні за потужністю, габаритами, надмірні за вартістю і за складністю їх програмування.

Промислові контролери відрізняються вузькою спрямованістю їх архітектури та програмного забезпечення на вирішення завдань програмно-логічного управління. Промислові контролери, по суті, є програмованими логічними контролерами (PLC – Programmable Logic Controller). Вони мають вбудовані інтерпретатори з мови релейної автоматики або мови Булевої алгебри. Це дозволяє програмувати контролер неспеціалісту з мікропроцесорної техніки і легко модернізувати системи релейної автоматики з жорсткою логікою. Провідними виробниками промислових контролерів є фірми Siemens (PLC Simatic), Allen Bradley, ABB, Fanuc та інші. Є також низка вітчизняних виробників. Незважаючи на широкий спектр можливостей з управління практично на всіх рівнях, промислові контролери не розраховані для застосування в якості вбудованих керуючих засобів в механотронних

системах.

Проектування цифрових систем керування на базі мікропроцесорної (МП), та зокрема, мікроконтролерної (МК), техніки є одним із складових напрямків інженерної діяльності фахівця зі спеціальностей 153 «Мікро– та наносистемна техніка», 171 «Електроніка». Поряд із знанням широкого кола питань з цифрової електроніки та програмування для успішного проектування МП систем управління потрібні досвід, творчий підхід, відповідне інформаційне та технічне забезпечення. Постійне удосконалення та поновлення номенклатури МП засобів та цифрової схемотехніки, жорсткі обмеження на тривалість і вартість розробки вносять додаткові ускладнення в процес проектування цифрових систем управління і автоматики на базі МП пристроїв та цифрових схем. Тому для успішного подолання таких складнощів в інженерній діяльності фахівців необхідним є формування у студентів знань з апаратних та програмних засобів мікропроцесорної та цифрової схемотехніки, необхідних для побудови сучасних цифрових систем автоматики та управління.

У посібнику також розглянуті особливості застосування засобів контролю та керування, їх архітектура, структурні схеми, базові поняття та перспективи створення й використання систем автоматизованого управління із застосуванням сучасних компонентів. В восьмій главі наданий приклад розробки прототипу системи контролю та керування на базі програмованої логіки. Майже в кожній главі в якості додатку наведені приклади розробки МПС на базі програмованих засобів, що застосовуються при розробці систем керування. Також невід’ємною частиною промислових систем контролю та керування є індустріальні протоколи, що застосовуються згідно з концепцією Індустрії 4.0. Дана тема розглянута в дев’ятій главі.

Матеріал навчального посібника призначений для забезпечення дисципліни «Мікропроцесорні системи контролю та керування», «Елементна база сучасної електроніки» та «Проектування та конструювання електронних пристроїв і систем» для спеціальностей 153 «Мікро – та наносистемна техніка», 171 «Електроніка», освітньо – професійних програм «Мікро – та наноелектроніка», «Електронні пристрої та системи», а також може бути корисним для споріднених спеціальностей.

1 АВТОМАТИЗОВАНЕ УПРАВЛІННЯ ПРОМИСЛОВИМИ ОБ'ЄКТАМИ

Мікропроцесорні засоби контролю та керування є базовим елементом автоматизованих систем управління. В свою чергу, автоматизація процесів управління є основним напрямом модернізації технологій в будь – якій сфері виробництва, від космічної галузі до побутової техніки. Для розуміння важливості засобів контролю та керування в загальній структурі систем автоматичного управління (САУ), потрібно розглянути мету, структуру, різновиди та базові положення систем управління.

Різновиди автоматизованих систем.

1. АСУП (Автоматизована Система Управління Підприємством), що включає ситему вирішення завдань синхронізації, координації, аналізу та оптимізації випуску продукції (MES – *Manufacturing Execution System*) і систему планування ресурсів підприємства (MRP – *Manufacturing Resources Planning*) або (ERP – *Enterprise Resources Planning*) .

2. АСУ ТП (Автоматизована Система Управління технологічними процесами), що включає обладнання з програмованими контролерами (PLC – *Programable Logic Controllers*), розподілені системи управління (DCS– *Distributed Control Systems*), системи диспетчерського управління та збору даних (SCADA – *Supervisory Control And Data Acquisition*), системи забезпечення людино–машинного інтерфейсу тобто двостороннього зв'язку "оператор – технологічне обладнання" MMI (*Man–Machine Interface*) або HMI (*Human–Machine Interface*) (рис. 1.1).

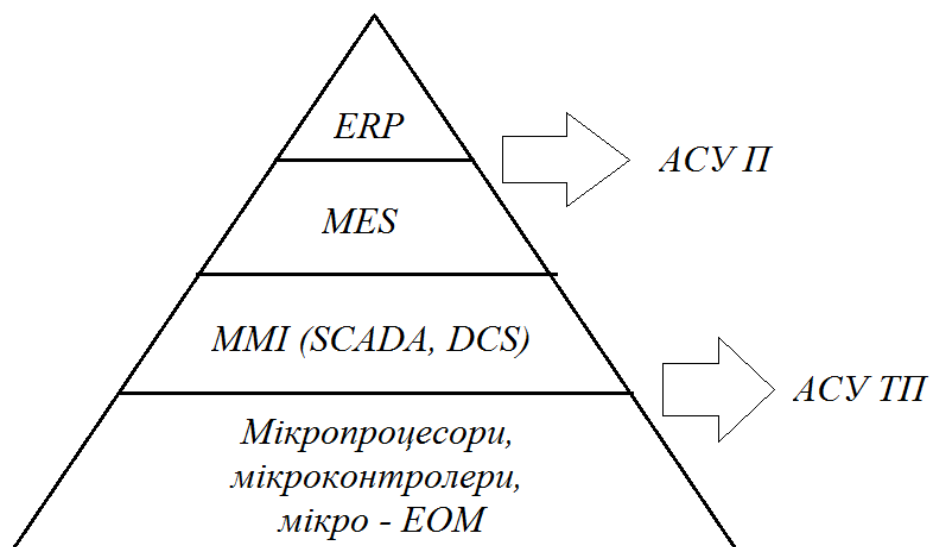


Рисунок 1.1 – Різновиди автоматизованих систем, рівні автоматизації.

Також широко впроваджуються ГАС (Гнучки Автоматизовані Системи, FMS – *Flexible Manufacturing Systems*) із застосуванням роботизованих комплексів. ГАС забезпечують максимальну ступінь гнучкості переналагодження і складаються з одного або декількох гнучких виробничих комплексів, об'єднаних з автоматизованою системою управління виробництвом.

ГАС забезпечує швидкий перехід на нове обладнання і виготовлення нової продукції.

1.1 Системи автоматичного управління

Автоматичне управління в техніці – це сукупність дій, спрямованих на підтримку або поліпшення функціонування керованого об'єкта без безпосередньої участі людини, відповідно до вказаної мети управління. Також, автоматизація широко застосовується в багатьох технічних і біотехнічних системах для виконання операцій, які неможливо здійснити людиною в зв'язку з необхідністю переробки великої кількості інформації в обмежений час, для підвищення продуктивності праці, якості і точності регулювання, звільнення людини від управління системами, що функціонують в умовах відносної недоступності або небезпечних для здоров'я. Мета управління тим чи іншим чином пов'язується зі зміною в часі регульованої вихідної величини керованого об'єкта. Для здійснення мети управління, з урахуванням особливостей керованих об'єктів різної природи і специфіки окремих класів систем, організовується вплив на керівні органи об'єкта – *керуючий вплив*. Управління також призначене для компенсації ефекту зовнішніх впливів, що збурюють, та прагнуть порушити необхідну поведінку регульованої величини. Керуючий вплив виробляється пристроєм управління (ПУ). Сукупність взаємодіючих керуючого пристрою і керованого об'єкта утворює систему автоматичного управління (рис.1.2).

Система автоматичного управління підтримує або підвищує функціональність керованого об'єкта. У ряді випадків допоміжні для САУ операції (пуск, зупинка, контроль, налагодження і т.д.) також можуть бути автоматизовані. САУ функціонує в основному в складі виробничого або будь якого іншого комплексу.

Історія техніки налічує багато ранніх прикладів конструкцій, що володіють всіма відмінними рисами САУ. Першою замкнутої САУ, що отримала широке технічне застосування, була система автоматичного регулювання обертів з відцентровим регулятором в паровій машині Ватта (1784). Новий етап в автоматизації характеризується впровадженням в системи регулювання і управління електронних елементів і пристроїв автоматики і телемеханіки. Це зумовило появу високоточних систем стеження і наведення, телеуправління і телевимірювання, системи автоматичного контролю і корекції. Кінець 20 століття ознаменувалися появою складних систем управління виробничими процесами і промисловими комплексами на базі електронних обчислювальних машин.

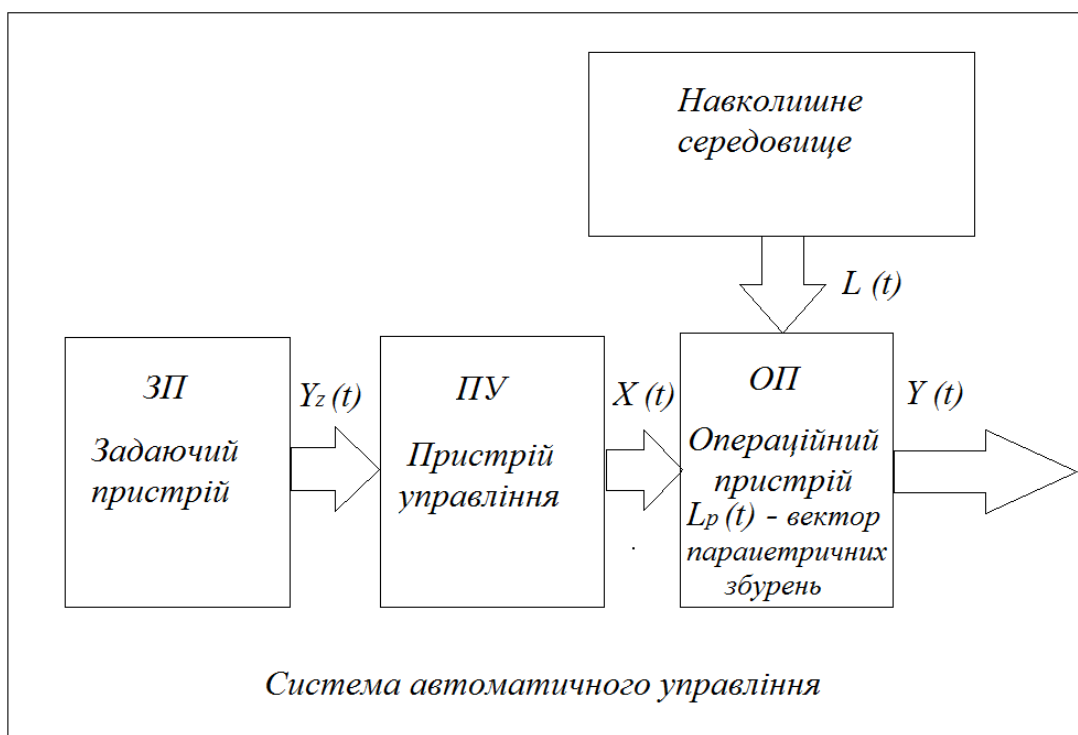


Рисунок 1.2 – Узагальнена структура системи автоматичного управління

Системи управління класифікуються в основному за метою управління, типу контуру управління і способу передачі сигналів. Спочатку перед САУ ставилися завдання підтримки певних законів зміни в часі керованих величин. В цьому класі систем розрізняють системи автоматичного регулювання (САР), в завдання яких входить збереження постійними значення керованої величини; системи програмного управління, де керована величина змінюється за заданою програмою; стежачи, для яких програма управління заздалегідь невідома. Надалі мета управління стала зв'язуватися безпосередньо з певними комплексними показниками якості, що характеризують систему (її продуктивність, точність відтворення і т. п.); системи, що мають функції самостійного налаштування. Останні розрізняються за способом управління: в *самоналагоджувальних* системах параметри пристрою управління змінюються, поки не будуть досягнуті оптимальні або близькі до оптимальних значення керованих величин; в *само організованих* системах з тією ж метою може змінюватися і їх структура. Найбільш широкі можливості систем, що поліпшують алгоритми свого функціонування на основі аналізу досвіду управління. Пошук оптимального режиму в *адаптивних* САУ може здійснюватися як за допомогою автоматичного пошуку, так і без пошуковим чином.

Спосіб компенсації збурень пов'язаний з типом контуру управління системи. У *розімкнутих* САУ на ПУ не надходять сигнали, що несуть інформацію про поточний стан керованого об'єкта, або в них вимірюються і компенсуються головні з обурень, або управління ведеться по жорсткої програмі, без аналізу яких не будь чинників в процесі роботи. Основний тип САУ – *замкнуті*, в яких регулювання здійснюється по відхиленню, а ланцюги проходження сигналів

утворюють замкнутий контур, що включає об'єкт управління і ПУ. Відхилення керованої величини від бажаних значень компенсуються впливом через зворотний зв'язок, незалежно від причин, що викликали ці відхилення. Об'єднання принципів управління по відхиленню і по обуренню призводить до *комбінованих систем*. Часто, крім основного контуру управління, включно з головним зворотним зв'язком, в САУ є допоміжні контури для стабілізації і корекції динамічних властивостей (*багатоконтурні системи*). Одночасне управління декількома величинами, що впливають одне на одного, здійснюється в системах *багато зв'язного* управління або регулювання.

За формою подання сигналів розрізняють *дискретні* та *безперервні* САУ. У перших сигнали хоча б в одній точці кола проходження дискретизуються за часом, або за рівнем, або як за рівнем, так і за часом.

Спеціалізовані САУ широко застосовуються в різних областях техніки, наприклад: система управління копійчально – фрезерним верстатом; САУ металорізальних верстатів з програмним керуванням; система програмного управління реверсивним прокатним станом, що включає в свій контур керуючу обчислювальну машину. У відносно повільних технологічних процесах в хімічній і нафтовій промисловості поширені багато зв'язкові САУ, що здійснюють регулювання великої кількості пов'язаних величин; так, при перегонці нафти інформація про температуру, тиск, витраті і складі нафтопродуктів, що отримується від декількох сотень датчиків, використовується для формування сигналів управління десятками різних регуляторів. У військовій техніці застосовуються високоточні системи, що стежать, (наприклад, система кутового супроводу радіолокаційної станції).

1.2 Склад і класифікація систем автоматичного регулювання.

Системи автоматичного управління в багатьох випадках мають в складі системи автоматичного регулювання різного рівня складності та призначення.

Головним чином *системи автоматичного регулювання* підрозділяються за такими чинниками:

- призначення (температури, тиску, перепаду тиску, рівня, витрати);
- характеристика регулювання (позиційне, статичне, астатичне,);
- вид допоміжної енергії (гідравлічні, електричні);
- швидкодія регулюючого органу (з постійною, змінною швидкістю).

Загальні принципи автоматичного регулювання. САР призначена для ліквідації наслідків всіх збурюючих факторів і приведення регульованого параметра до заданого значення. Завданням автоматичного регулювання є забезпечення заданого технологічного режиму роботи системи без безпосередньої участі людини. Технологічний режим задається через робочі параметри: тиск, температуру, витрату і т. п. Регулятори діляться на регулятори *прямої* і *непрямої* дії. В регуляторах *прямої дії* в одному елементі поєднується сенсор, задаючий і командний пристрій, при цьому вимірювальний орган безпосередньо впливає на регулюючий (виконавчий) орган, крім того, виконавчий орган для приводу ви-

користовує енергію самого регульованої середовища (рис 1.3).

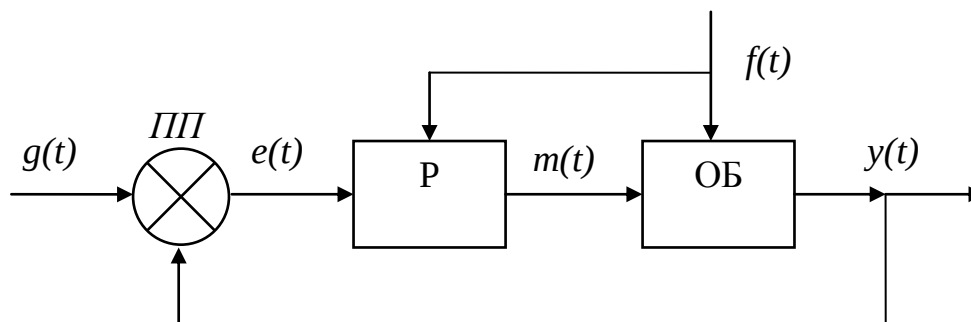


Рисунок 1.3 – Структурна схема системи автоматичного регулювання:

Об – об'єкт регулювання з передавальною функцією $W_{об}(P)$;

ПП – компаратор (пристрій порівняння);

P – регулятор з передавальною функцією $W_p(P)$;

$f(t)$ – збуджуючий вплив;

$y(t)$ – регулюєма величина;

$g(t)$ – похибка регулювання;

$e(t)$ – задаючий вплив;

$m(t)$ – керуючий вплив.

В регуляторах *непрямої дії* вимірювальний орган впливає на виконавчий механізм не прямо, а через командний орган, до якого підводиться енергія від стороннього джерела.

В таблиці 1.1 наведена загальна класифікація систем автоматичного регулювання.

Таблиця 1.1 – Класифікація систем автоматичного регулювання.

Класифікація	Коротка характеристика САР
1 За призна- ченням алго- ритму зміни впливу (або за видом виконуваних функцій)	стабілізуюча програмна стежуча
2 За кількіс- тю контурів регулювання	одно контурні багато контурні
	з управлінням від комп'ютера
	Підтримує регульований параметр на постійному значенні за- даної точки. $X = SP = \text{const}$ Змінює регульовану величину відповідно до функцій завдання в часі – програмні задатчики. $SP = F [SP_{\text{prog}}(t)]$ Завдання полягає в тому, щоб зміни регульованої величини стежили за змінами іншого параметра. $X = \text{var}$ Змінює регульовану величину залежно від завідомо відомої величини заданої точки. Значення задання регулятора форму- ється по інтерфейсу. $X = \text{var}$ Містять один контур регулювання Містять кілька контурів регулювання (в т.ч. каскадні регуля- тори, регулятори обмеження)

Класифікація	Коротка характеристика САР
однокомпонентні	Системи з однієї регульованою величиною
багатокомпонентні незв'язані	Системи з декількома регульованими величинами. Регулятори безпосередньо не пов'язані і можуть взаємодіяти тільки через загальний для них об'єкт регулювання
3 За кількістю регульованих параметрів	Системи з декількома регульованими величинами. Регулятори різних параметрів одного або декількох об'єктів, пов'язані між собою:
багатокомпонентні пов'язані	– регулятори зі статичної та/або динамічною корекцією параметра або заданої точки; – регулятори співвідношення декількох параметрів з постійним або керованим коефіцієнтом співвідношення; – каскадні регулятори; – регулятори обмеження (з макс. або мін. обмеженням).
4 За своїм функціональним призначенням	САР температури, тиску, витрати, рівня, об'єму та ін. З нормованими вхідними та вихідними сигналами і придатні для управління різними параметрами
універсальні	
двох позиційні	
5 За законом регулювання або логікою роботи регулювання	П, ПІ, ПІД – регулятори контуру ШІМ – регулятори адаптивні оптимальні
Дивитися другу главу	Які самостійно налагоджуються Використовують оптимальний закон регулювання
6 За характером сигналів управління (за родом дії)	Аналогові сигнали (струм, напруга). Окремий випадок – вихідний сигнал ШІМ регулятора. Релейні, імпульсні, цифрові. Вихідні пристрої – механічні реле, твердотільні реле, симистори, тиристори, транзисторні ключі, інтерфейси.
7 За характером математичних співвідношень	лінійні нелінійні
Дивитися другу главу	
8 За видом енергії регулювання	електричні пневматичні гідравлічні механічні комбіновані
В т.ч. електронні Мембранні, поршневі, лопатеві. Мембранні, поршневі. Різні механічні приводи Електропневматичні, пневмо – електромеханічні	
9 За принципом регулювання	за неузгодженістю за відхиленням за збуренням комбіновані
Дивитися другу главу	
10 За напрямом дії	прямі зворотні
Регулятори прямої (нормальної) дії Регулятори зворотної (реверсивної) дії	
11 За принципом дії	прямой дії непрямой дії
Використовують енергію самого об'єкта управління (деякі регулятори тиску) Для роботи потрібне зовнішнє джерело енергії	

У САР здійснюється замкнутий контур взаємодії: об'єкт регулювання впливає на регулятор, а регулятор через виконавчий орган впливає на об'єкт регулювання (рис. 1.4). Крім внутрішніх зв'язків на регулятор впливають зовнішні впливи, наприклад, зміна завдання і т. п.

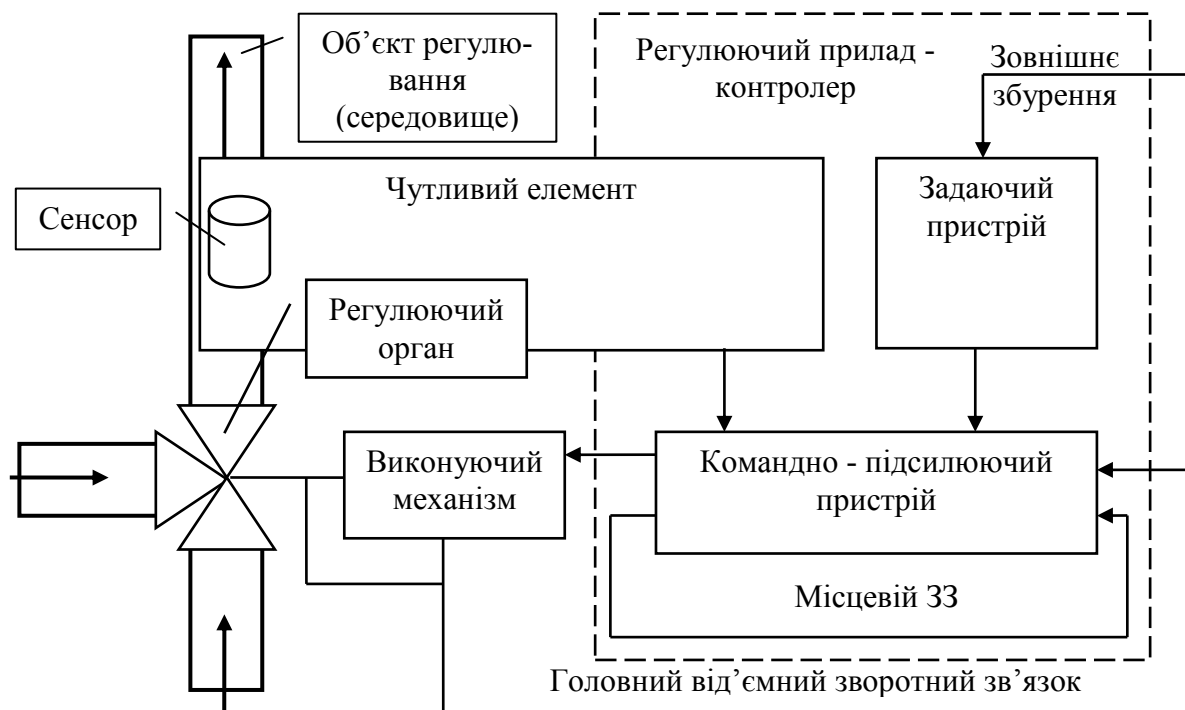


Рисунок 1.4 – Структура автоматичного регулятора

Чутливий елемент – складається з сенсора, вимірювального пристрою, підсилювача і перетворювача; крім датчика все перераховане входить до складу контролера (електронного блоку).

Задаючий пристрій (задатчик) – за допомогою цього пристрою задається необхідне значення регульованої величини (наприклад, уставка температури гарячої води і т.п.).

Командно– підсилюючий пристрій – порівнює сигнали від чутливого елемента і задатчика, виробляє сигнал неузгодженості і підсилює його до величини, необхідної для управління виконавчим механізмом. Ступінь посилення сигналу неузгодженості визначається глибиною негативного зворотного зв'язку.

Виконавчий механізм (привід) – перетворює сигнал від командно – підсилювального пристрою в рух регулюючого органу. *Регулюючий орган* змінює витрату потужності, або регульованого середовища.

Пристрій зворотного зв'язку – передає вплив з виходу виконавчого механізму на вхід командно– підсилюючого пристрою. (в САР застосовується тільки негативний зворотний зв'язок – НЗЗ, інакше, при переважанні позитивного зворотного зв'язку система переходить в режим генератора, тобто буде мати тільки два крайніх положення).

1.3 Приклади систем автоматичного регулювання в процесі формування структур інтегральних схем

Приклад 1. Регулювання температури в процесі формування структур напівпровідникових інтегральних схем.

Показником ефективності регулювання є підтримка температури на заданому рівні. В даному прикладі температура процесу є вихідна керована координата. Стабілізація температури процесу здійснюється за використанням в якості вхідного регулюючого сигналу дані від сенсорів температури, що вбудовані у тигель та в утримувач підкладинок мікросхем (рис. 1.5). Система, що надана в прикладі, здійснює регулювання за відхиленням температури процесу шляхом змінення потужності, що дістається до нагрівачів.

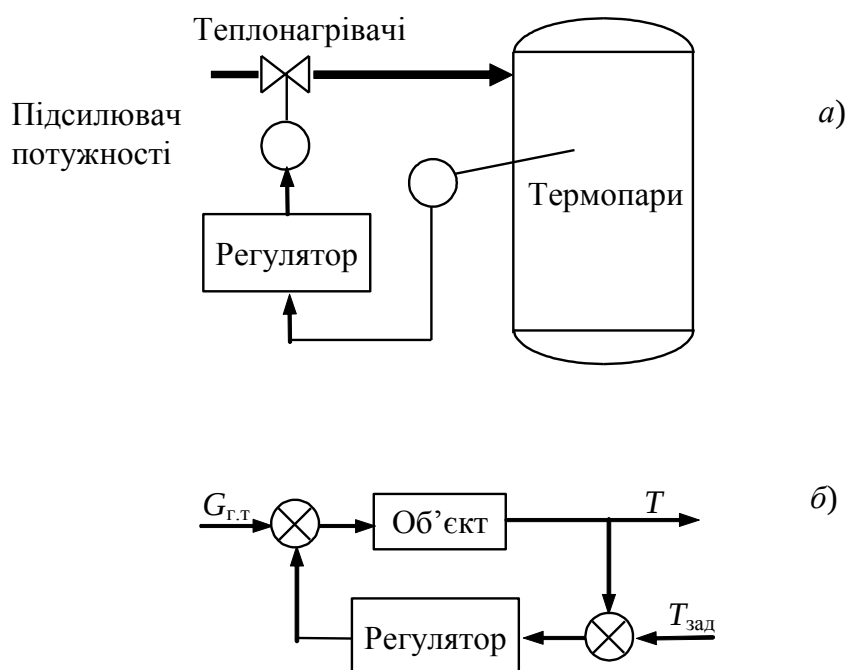


Рисунок 1.5 – Система регулювання температури процесу в вакуумній установці: а – технологічна схема; б – структурна схема

Приклад 2. Регулювання тиску в вакуумній установці.

В вакуумних установках регулювання тиску здійснюється за допомогою вакуумних насосів. На разі керованою величиною є розрядження, а керуючою – величина потужності, що подається до вакуумних насосів (рис 1.6).

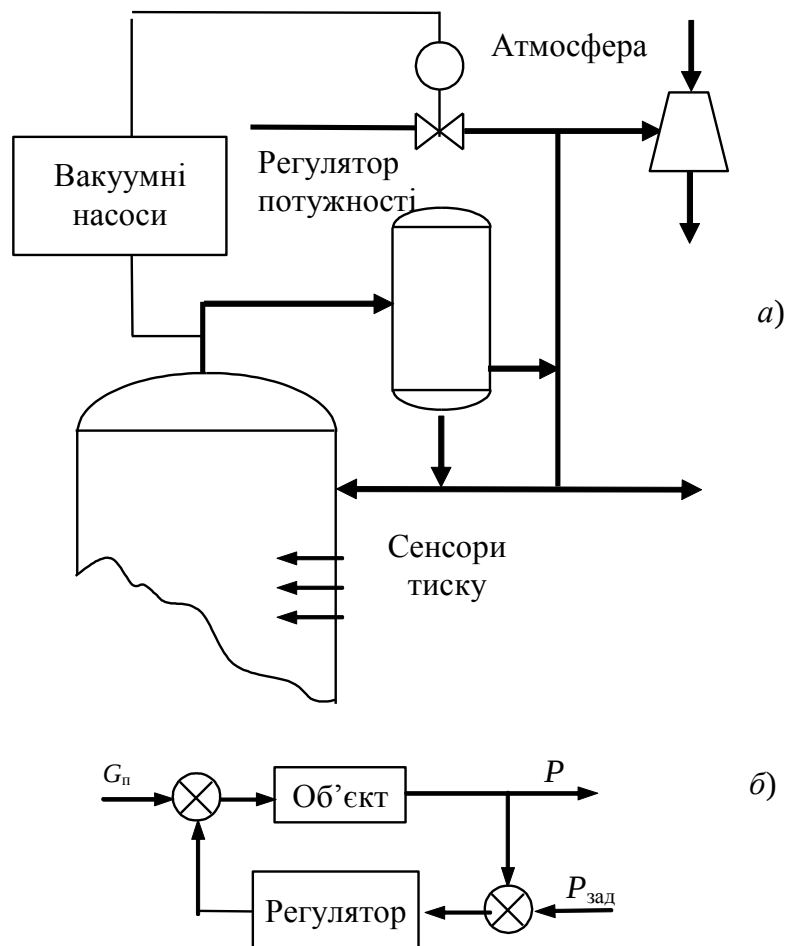


Рисунок 1.6 – Система регулювання тиску в вакуумній установці:
а – технологічна схема; *б* – структурна схема

В розглянутих прикладах структури систем автоматичного регулювання мають спрощений характер. В будь-якій реальній АСР можна виділити наступні складові елементи:

- Об'єкт регулювання;
- Чутливий елемент;
- Підсилюючий та перетворюючий пристрій;
- Регулятор;
- Виконавчий пристрій;
- Регулюючий орган.

Повна структурна схема зображена на рис. 1.7

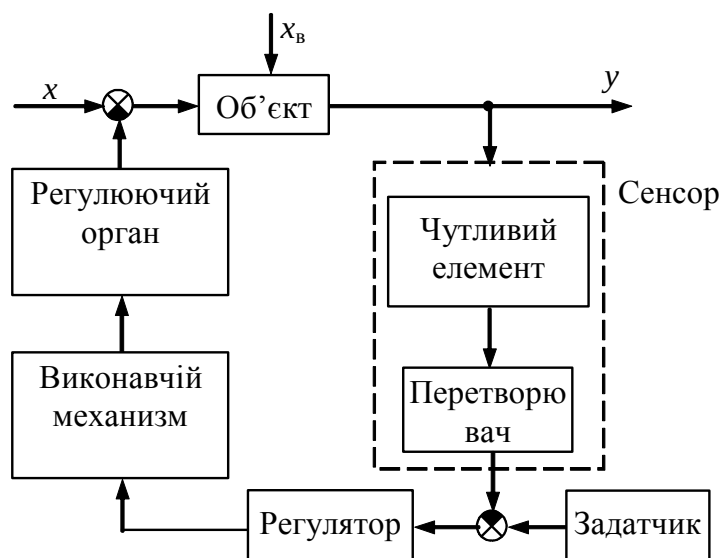


Рисунок 1.7 – Структурна схема систем автоматичного регулювання

В подальшому ми будемо використовувати спрощені схеми, умовно відносячи чутливий елемент, перетворювач та підсилювач, виконуючий та регулюючий елементи до об'єкту регулювання.

1.4 Контрольні запитання і завдання

- 1 Систему управління утворюють:
 - А Сукупність засобів управління і об'єкта управління.
 - В Сукупність засобів управління.
 - С Об'єкт управління.
- 2 Чим характеризується будь-який елемент системи?
 - А Вхідною координатою.
 - В Вихідною координатою.
 - С Вхідною і вихідною координатами.
- 3 Який принцип регулювання був реалізований в першому промисловому регуляторі рівня в котлі парової машини.
 - А Регулювання "за відхиленням".
 - В Регулювання "за збуренням".
 - С Комбіноване регулювання.
- 4 Яка система регулювання називається автоматичною?
 - А Всі робочі операції і операції керування виконують автоматичні пристрої.
 - В Частина операцій управління виконують автоматичні пристрої, ін-

шу частину виконує людина.

С Робочі операції виконують машини і механізми, а операції управління - людина.

5 Детерміновані системи управління відображають:

А Характер подання сигналів.

В Характер процесу управління.

С Характер функціонування.

6 При класифікації систем управління за характером функціонування система автоматичного регулювання може бути:

А Системою програмного регулювання.

В Системою з розподіленими параметрами.

С Стохастичною системою.

7 Система автоматичної стабілізації – це система, в якій підтримується:

А $y_{зад}(t) = const.$

В $y_{зад}(t) = f(t).$

С $y_{зад} = f(x).$

8 За основними видами рівнянь динаміки процесів управління системи підрозділяються на:

А Безперервні і дискретні.

В Детерміновані і стохастичні.

С Лінійні і нелінійні.

9 В оптимальних системах управління показник ефективності залежить від:

А Поточних значень координат.

В Поточних значень координат, а також характеру їх зміни в минулому, сьогодні і майбутньому.

С Власних параметрів системи або структури.

2 ТИПИ РЕГУЛЯТОРІВ І НАЙ ПОШИРЕНІ ЗАКОНИ РЕГУЛЮВАННЯ

Процес регулювання відбувається за *відхиленням* і за *збуренням*.

Регулювання по відхиленню – регулятор вимірює величину відхилення безпосередньо регульованою середовища і виробляє вплив на регулюючий орган. За цим принципом працюють більшість регуляторів (наприклад, регулювання температури гарячої води).

Регулювання по збуренню – регулятор впливає на регульоване середовище в залежності від величини збурення (наприклад, зміна температури зовнішнього повітря). Переваги даного методу в тому, що регулятор починає впливати на об'єкт регулювання ще до того, як відбудеться відхилення регульованої величини.

За характером процесу регулювання регулятори поділяються на *регулятори– стабілізатори, програмні та стежучі*.

Регулятори– стабілізатори – підтримують на постійному рівні заданий параметр (наприклад, регулятор температури гарячої води на базі регулятора потужності і термовентіля).

Програмний регулятор – здійснює регулювання у часі за заданим алгоритмом (наприклад, мікропроцесорний регулятор температури гарячої води з програмованим час– температурним графіком).

Стежучі регулятори – регульований параметр є функцією деякої незалежної величини (наприклад, температура води в опалювальній системі є функцією температури зовнішнього повітря: $T^{\circ} oc = f(T^{\circ} np)$).

2.1 Закони регулювання.

Основною величиною, що визначає роботу регулятора (його входом), є так звана *неузгодженість* – величина відхилення регульованого параметра (μ).

Регулюючий вплив – величина керуючої дії (s), є вихідною величиною регулятора.

Законом регулювання називається залежність між *регулюючим впливом* (s) і *неузгодженістю* (μ) або, інакше, залежність між вхідним і вихідним впливом.

Розрізняють такі закони регулювання.

Позиційне регулювання: відсутня безперервна функціональна залежність між (s) і (μ) (вхідних і вихідних впливом). Регулюючий вплив змінюється стрибкоподібно і може мати або два положення: відкрито– закрито, або кілька фіксованих положень (позицій).

Розрізняють двох, трьох, і багатопозиційні регулятори. При цьому законі регулювання завжди буде мати місце різниця двох крайніх значень регульованого параметра, яка називається – *зоною нерівномірності*.

Нерівномірністю регулятора називають діапазон зміни регульованої величини, необхідний для зміни регулюючого впливу з одного крайнього положення в інше крайнє положення.

Статичне регулювання здійснюється за пропорційною зміною вихідного впливу (s) до відхилення параметра (μ): $s = k * \mu$.

Пропорційні регулятори називаються *П-регуляторами*. У цих регуляторах швидкість зміни регулюючого впливу (вплив s) пропорційна швидкості зміни параметра (неузгодженості μ).

У статичних системах регулювання відхилення регульованого параметра після закінчення перехідного процесу залежить від величини збурення μ (рис. 2.1а) і називається *статичною помилкою*, яка тим більше, чим значніше зміна навантаження (тобто збурення). П-регулятор швидко відновлює задане значення, але зі статичною помилкою.

Астатичне регулювання – при цьому методі регулювання регулюючий вплив може приймати будь-яке значення (при статичному регулюванні регулюючий вплив змінюється пропорційно відхиленню). При астатичному – між вихідним впливом s і відхиленням μ існує інтегральна залежність: регулюючий вплив змінюється при відхиленні на деяку величину в часі і буде змінюватись до тих пір, поки параметр не повернеться до заданого значення. Такий регулятор називається *I-регулятором*.

За астатичною схемою регулювання положення регулюючого органу не залежить від встановленого значення регульованого параметра (рис. 2.1б) і при різних за величиною збурення відхилення регульованого параметра після закінчення перехідного процесу стає рівним нулю.

Перевага астатичного регулювання полягає в підтримці параметра на заданому рівні; недолік – затягування процесу регулювання.

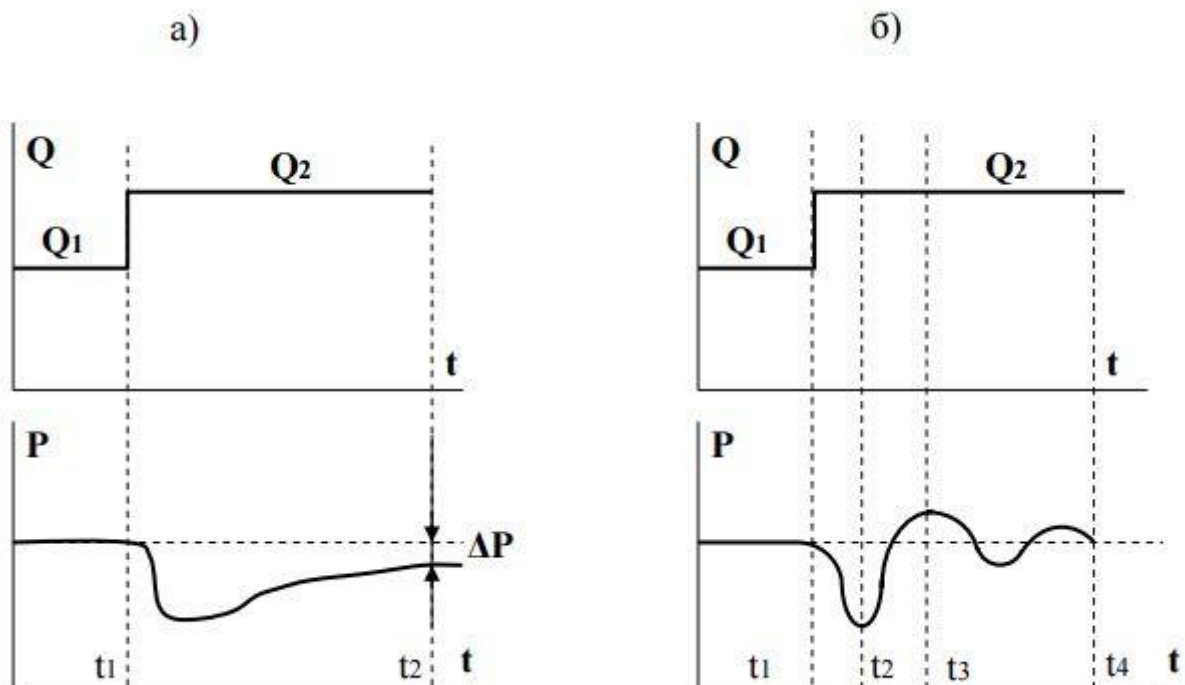


Рисунок 2.1 – Залежність регульованого параметра (P) від внесеного збурення (Q): а) – при статичному (пропорційне, П – регулювання); б) – при аста-

тичному, регульований параметр нестійкий (інтегральне, І– регулювання).

Ізодромне регулювання – має властивості статичного і астатичного регулювання, тобто регулюючий вплив залежить як від величини відхилення, так і від суми відхилень у часі.

Спочатку регулюючий вплив залежить від величини події відхилення (статичну регулювання), а потім регулюючий вплив здійснює додаткову зміну, яка призводить до усунення статичної помилки. Такий вид регулятора називають ПІ– регуляторами. При ПІ – регулюванні процес протікає без значних коливань, параметр швидко повертається точно до заданого значення.

Регулювання з додатковим впливом по відхиленню – в регуляторах типу ПІ і ПІІ застосовують додатковий вплив від прискорення відхилення (μ), таким чином регулятор буде реагувати не тільки на відхилення, але і на швидкість цього відхилення. Регулятори, що працюють за таким законом, називаються ПІД – регуляторами .

Регулювання з додатковим впливом від збурення (по навантаженню) – оскільки відхилення і швидкість відхилення виникають в результаті внесеного збурення, реакцію регулятора можна поліпшити, внісши в закон регулювання додатковий вплив від навантаження.

2.2 Пропорційно-інтегральний закон регулювання

Пропорційно-інтегральний закон регулювання описується рівнянням

$$x_p(t) = -(S_1 \Delta y(t) + S_0 \int_0^t \Delta y(\tau) d\tau) \quad (2.1)$$

та є паралельним з'єднанням пропорційної та інтегральної складової.

Динамічні характеристики ПІ–регулятора:

– передавальна функція:

$$W(s) = -\left(S_1 + \frac{S_0}{s}\right); \quad (2.2)$$

– частотні характеристики (рис. 2.2):

$$\text{АФХ} \quad W(i\omega) = -\left(S_1 + \frac{S_0}{i\omega}\right); \quad (2.3)$$

$$\text{АЧХ} \quad M(\omega) = \frac{\sqrt{S_1^2 \omega^2 + S_0^2}}{\omega}; \quad (2.4)$$

$$\text{ФЧХ} \quad \varphi(\omega) = \frac{\pi}{2} + \arctg\left(\frac{S_1 \omega}{S_0}\right). \quad (2.5)$$

Перехідна функція (рис. 2.3а):

$$h(t) = -(S_1 1(t) + S_0 t). \quad (2.6)$$

Вагова функція (рис. 2.3б):

$$w(t) = -(S_1\delta(t) + S_0). \quad (2.7)$$

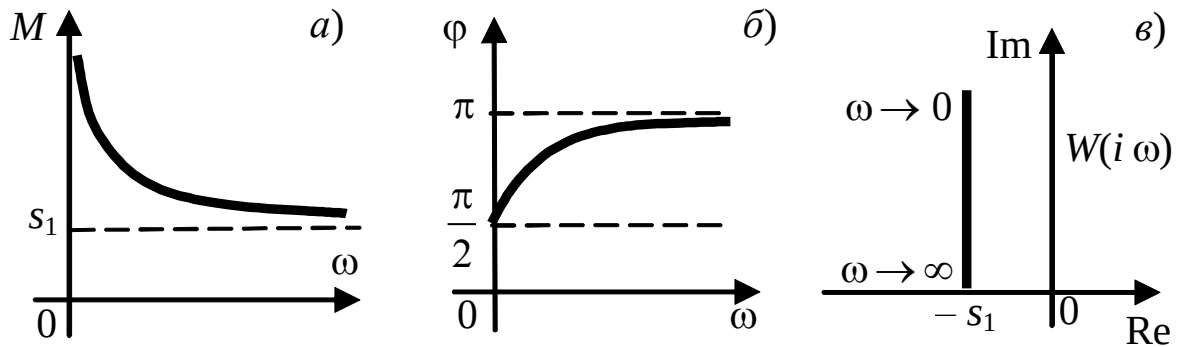


Рисунок 2.2 – Частотні характеристики ПІ-регулятора: *а* – АЧХ; *б* – ФЧХ; *в* – АФХ

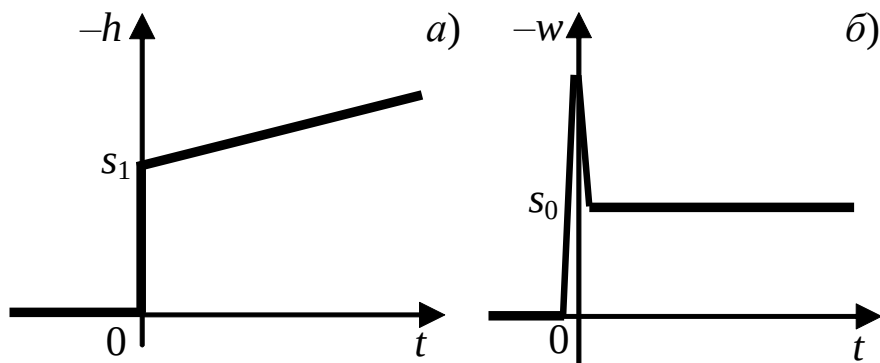


Рисунок 2.3 – Перехідні характеристики:
а – перехідна функція; *б* – вагова функція

Пропорційно-інтегральний регулятор поєднує в собі переваги П- і І-законів регулювання, а саме: пропорційна складова забезпечує достатню швидкість регулятора, а інтегральна складова ліквідує статичну похибку регулювання. Перехідний процес в АСР з ПІ регулятором зображений на рис. 2.4.

На початку процесу регулювання основну роль грає пропорційна складова, так як інтегральна складова залежить не тільки від абсолютного значення, а й від часу. Зі збільшенням часу зростає роль інтегральною складовою, що забезпечує усунення статичної похибки, тобто

$$\begin{aligned}\lim_{t \rightarrow \infty} y(t) &= \lim_{s \rightarrow 0} sy(s) = \lim_{s \rightarrow 0} s \frac{1}{s} \frac{W_{ог}(s)}{1 + W_{ог}(s)(S_1 + S_0/s)} = \\ &= \lim_{s \rightarrow 0} \frac{s W_{ог}(s)}{s + W_{ог}(s)S_1 s + S_0} = 0.\end{aligned}\quad (2.8)$$

Підбором параметрів настройки S_0 і S_1 можна змінювати питому вагу кожної складової. Зокрема, при $S_0 = 0$ виходить П-регулятор, а при $S_1 = 0$, І-регулятор.

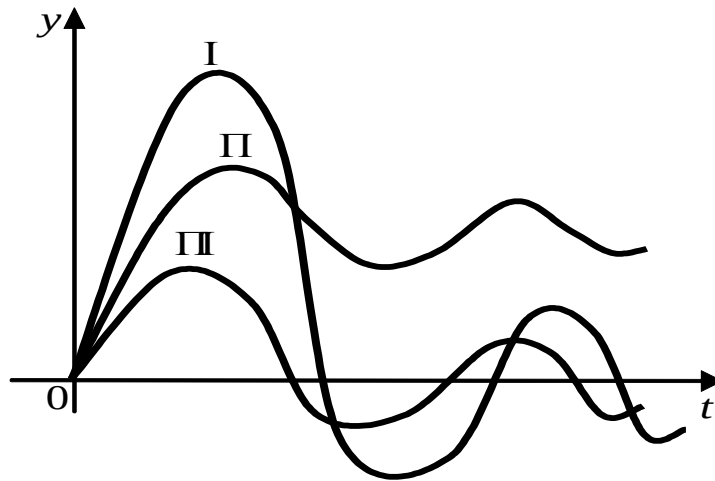


Рисунок 2.4 – Перехідний процес в АСР с ПІ-, П- та І-регуляторами

2.3 Пропорційно-інтегрально-диференціальний закон регулювання.

Пропорційно-інтегрально-диференціальний закон регулювання описується рівнянням:

$$x_p(t) = -(S_1 \Delta y(t) + S_0 \int_0^t \Delta y(\tau) d\tau + S_2 \Delta y'(t)). \quad (2.9)$$

Динамічні характеристики ПІД-регулятора:
передавальна функція

$$W(s) = -(S_1 + \frac{S_0}{s} + S_2 s). \quad (2.10)$$

частотні характеристики (рис. 2.5):

$$\text{– АФХ} \quad W(i\omega) = -(S_1 + \frac{S_0}{i\omega} + S_2 i\omega); \quad (2.11)$$

$$\text{– АЧХ} \quad M(\omega) = \frac{\sqrt{S_1^2 \omega^2 + (S_0 - S_2 \omega^2)^2}}{\omega}; \quad (2.12)$$

$$\text{– ФЧХ} \quad \varphi(\omega) = \frac{\pi}{2} + \arctg\left(\frac{S_1 \omega}{S_0 - S_2 \omega^2}\right). \quad (2.13)$$

Перехідні характеристики:

перехідна функція, при $t > 0$

$$h(t) = -(S_1 + S_0 t + S_2 \delta(t)); \quad (1.14)$$

вагова функція

$$w(t) = -(S_1 \delta(t) + S_0 + S_2 \delta'(t)). \quad (1.14)$$

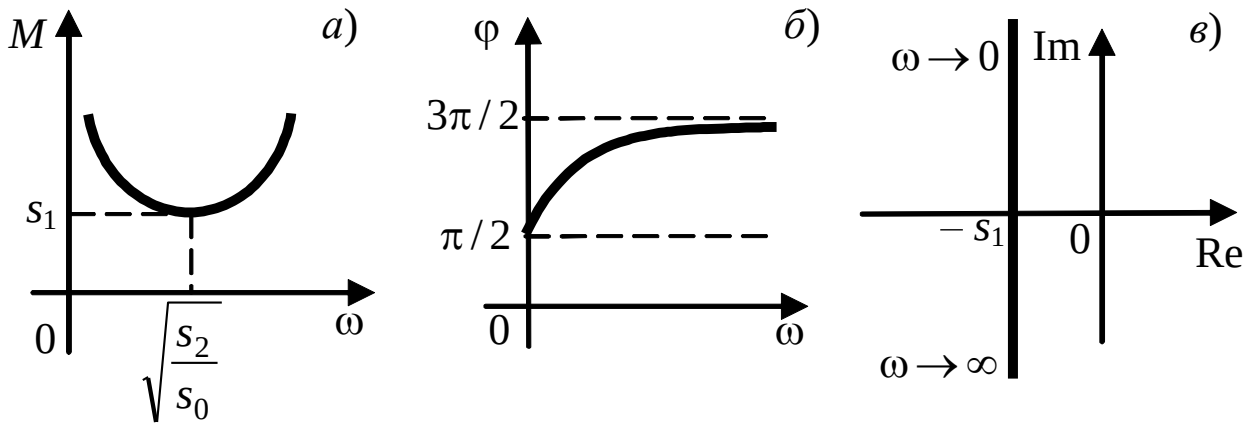


Рисунок 2.5 – Частотні характеристики ПІД-регулятора: a – АЧХ; $б$ – ФЧХ; $в$ – АФХ

Графік перехідної функції ПІД-регулятора наданий на рис. 2.6.

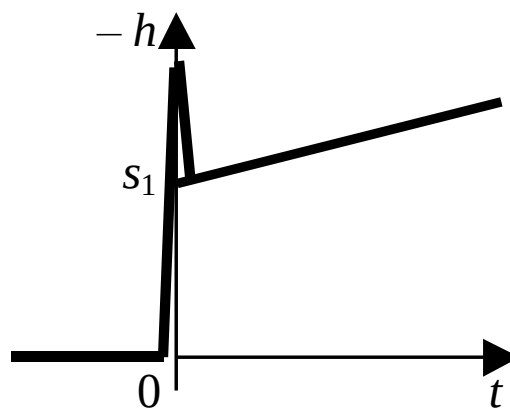


Рисунок 2.6 – Перехідна функція ПІД-регулятора

ПІД-регулятор поєднує в собі переваги всіх трьох найпростіших законів регулювання: високу швидкість завдяки наявності імпульсу по похідною від $dy(t)$ і відсутність статичної помилки, яке забезпечує інтегральна складова (рис. 2.6).

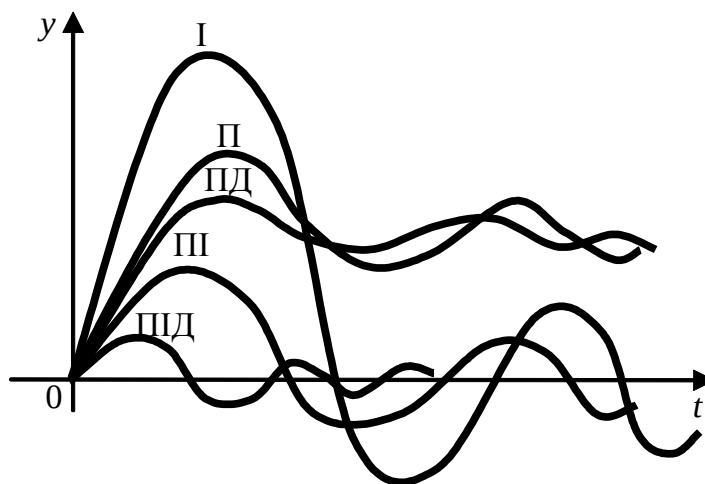


Рисунок 2.7 – Перехідні процеси в АСР с різними законами регулювання

Необхідно відзначити, що застосування регуляторів з диференційними складовими, незважаючи на їхні переваги, не завжди доцільно, а іноді і неприпустимо. Так, для об'єктів з великою запізненням по каналу регулювання марно вводити вплив по похідною від регулюємої величини, так як цей імпульс буде надходити в регулятор по закінченні часу чистого запізнювання після приходу збурення, за яке в об'єкті можуть накопичитися великі відхилення (рис. 2.7). Більш того, в таких випадках ПД або ПІД-регулятор може "розгойдати" об'єкт і система втратить стійкість.

2.4 Контрольні запитання і завдання

1 Колами називаються окремі елементи системи, в яких відбувається перетворення вхідних сигналів у вихідні. Якщо передавальна функція кола має вигляд простого дробу, то таке коло відноситься до групи типових або елементарних кіл, рівняння яких можна отримати з диференціального рівняння:

$$a_2 y''(t) + a_1 y'(t) + a_0 y(t) = b_1 x'(t) + b_0 x(t),$$

прирівнюючи ті чи інші коефіцієнти нулю.

Розрізняють такі кола: підсилююче, інтегральне, ідеальне і реальне диференціюючі, чистого запізнювання, аперіодичне першого порядку, аперіодичне другого порядку, коливальні. Кожна з перерахованих ланок розглядається з позицій аналізу їх динамічних характеристик.

A Які ланки описуються звичайними диференціальними рівняннями?

B Чому ідеальна диференціююча ланка фізично не піддається реалізації?

C На які групи поділяються типові ланки?

2 При аналізі і синтезі систем автоматичного управління широко використовується структурний аналіз. У будь-якій структурній схемі можуть бути присутні тільки три типи з'єднань: послідовне, паралельне, з'єднання зі зворотним зв'язком. Значення передавальних функцій окремих ланок дозволяє записати передавальні функції з'єднань і побудувати їх частотні характеристики.

Реальні об'єкти мають складну структуру: в них є, так звані, перехресні зв'язки, які необхідно розв'язати, використовуючи правила перетворення структурних схем.

A Які передавальні функції можна записати для одно контурної системи автоматичного регулювання?

B Задані передавальні функції ланок $W_1(s) = k$; $W_2(s) = 4 \frac{1}{Ts}$. Записати частотні характеристики послідовного і паралельного з'єднань.

C Перенесення яких елементів при перетворенні схем проводиться без додаткових перетворень?

3 Елементами одно контурної системи автоматичного регулювання є об'єкт і регулятор. Всі закони регулювання підрозділяються на найпростіші: пропорційний, диференційний, інтегральний і промислові: пропорційно-інтегральний, пропорційно-диференційний, пропорційно-інтегрально-диференціальний. Всі закони регулювання розглядаються з точки зору їх динамічних властивостей.

A Який із законів регулювання фізично не реалізується?

B Що дає введення в закон регулювання диференціальної складової?

C Передавальні функції регуляторів записуються зі знаком "-". Яку інформацію дає цей знак?

2.5 Короткий тест

1 Які кола відносяться до групи статичних?

A Статична характеристика відмінна от нуля.

B Статична характеристика не існує.

C Статична характеристика дорівнює нулю.

2 Передавальна функція якого кола має вигляд $W(s) = \frac{1}{Ts}$?

A Підсилюючого.

B Реально диференціюючого.

C Інтегрального.

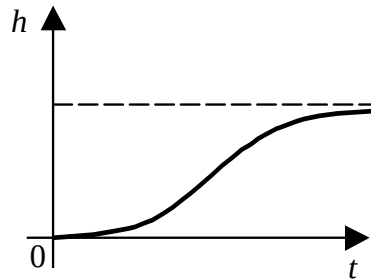
3 Передавальна функція аперіодичної ланки першого порядку має рівняння:

A $W(s) = \frac{Ks}{Ts+1}.$

B $W(s) = K + \frac{1}{Ts}.$

C $W(s) = \frac{K}{Ts+1}.$

4 Крива розгону якої ланки має вигляд?



A Підсилюючої.

B Аперіодичної першого порядку.

C Аперіодичної другого порядку.

5 Яке ланка описується рівнянням $T y'(t) + y(t) = k x'(t)$?

A Аперіодична ланка першого порядку.

B Ідеальна диференціююча.

C Реальна диференціююча.

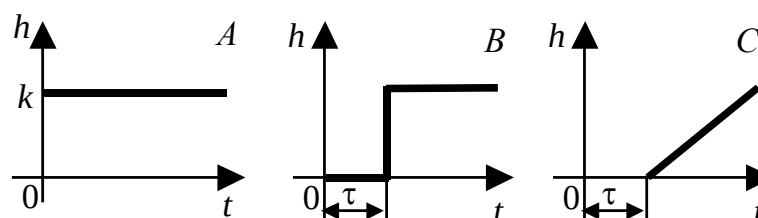
6 Яким рівнянням описується коливальний ланка?

A $T y'(t) + y(t) = k x(t).$

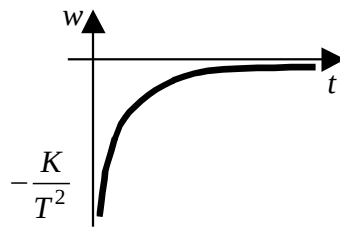
B $T_1 T_2 y''(t) + (T_1 + T_2) y'(t) + y(t) = k x(t).$

C $T_k^2 y''(t) + T_d y'(t) + y(t) = k x(t).$

7 Яку криву розгону має ланка чистого запізнювання?



8 Яке ланка має вагову функцію?

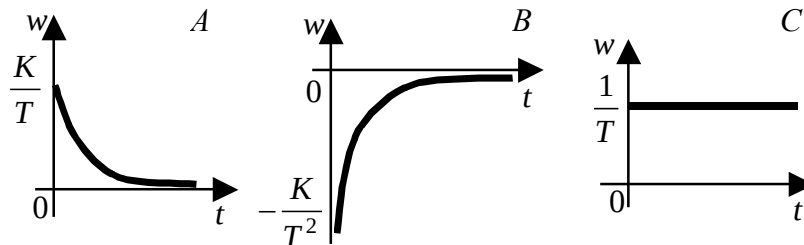


A Аперіодична ланка першого порядку.

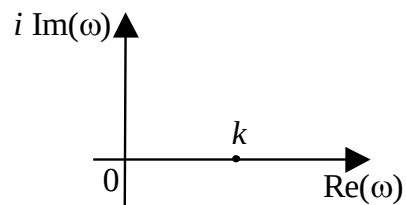
B Реальна диференціююча.

C Інтегральна.

9 Яку вагову функцію має аперіодична ланка першого порядку?



10 Яка ланка має зображену нижче АФХ?

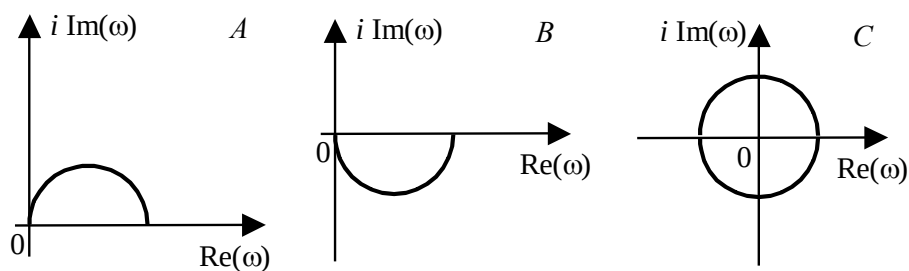


A Підсилююча.

B Інтегральна.

C Коливальна.

11 Яка АФХ відповідає ланці чистого запізнювання?



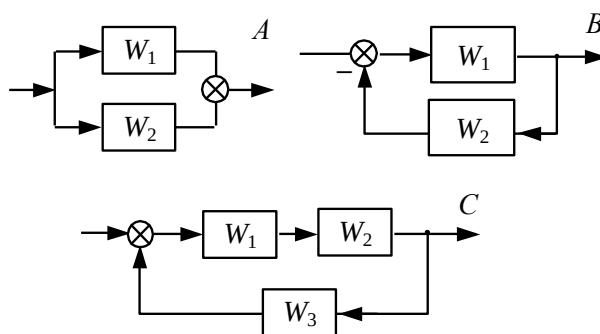
12 Яке ланка з відповідною функцією передачі відноситься до групи особливих ланок?

A $W(s) = \frac{k}{Ts+1}$.

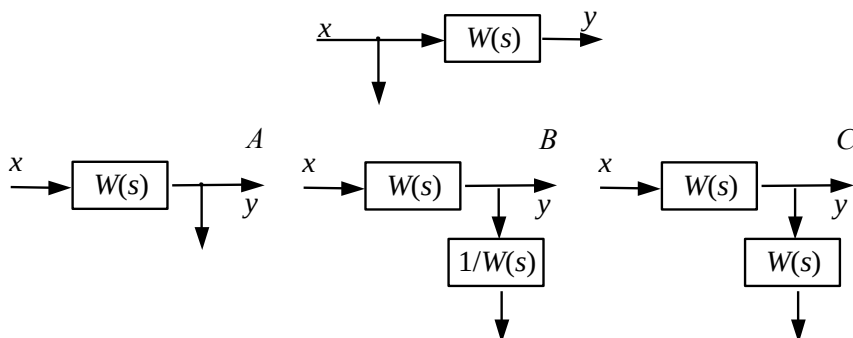
B $W(s) = \frac{k}{Ts-1}$.

C $W(s) = \frac{ks}{Ts+1}$.

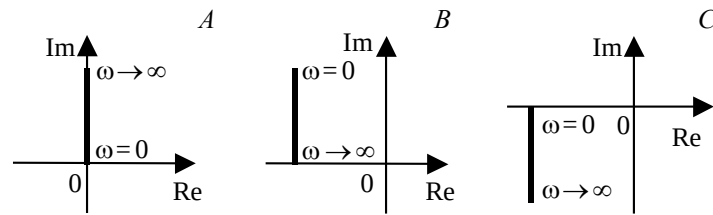
13 Яке з'єднання називається паралельним?



14 В якому наведеному варіанті правильно здійснений перенос вузла через ланку?



15 Яку з наведених АФХ має ПІ-регулятор?



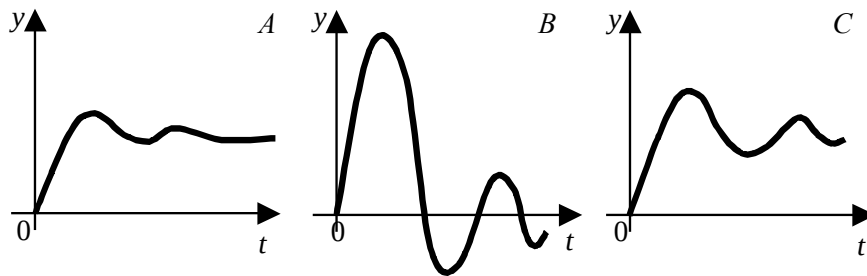
16 Яку передавальну функцію має ПД-регулятор?

A $W(s) = -S_1 - S_2 s.$

B $W(s) = -\frac{S_0}{s} - S_1 - S_2 s.$

C $W(s) = -\frac{S_0}{s} - S_1.$

17 Який перехідний процес буде в АСР с І-регулятором?



18 Який закон регулювання є найбільш поширеним на практиці?

A І-закон.

B ПІ-закон.

C ПД-закон.

19 Який закон регулювання має три налагоджувальних параметри?

A ПІ-закон.

B ПД-закон.

C ПІД-закон.

20 На рис. 2.8 зображено об'єкт з вхідними і вихідними сигналами.

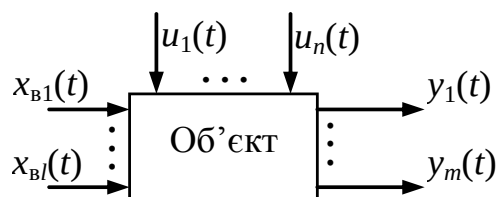


Рисунок 2.8 – Об'єкт управління з вхідними і вихідними сигналами.

- A* Що таке об'єкт управління? Наведіть конкретний приклад.
B Які зовнішні змінні є керуючими?
C Яка змінна є керованою змінною?

21 На рис. 2.9 зображено структурну схему системи автоматичного регулювання.

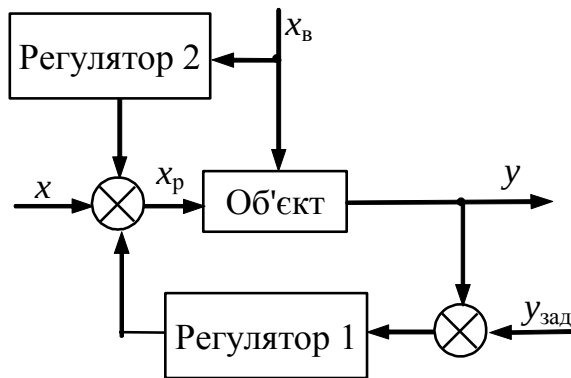


Рисунок 2.9 – Структурна схема АСР

Питання:

- A* Які принципи регулювання реалізовані в САР, зображеній на рисунку 2.9?
B Що означає регулювання за відхиленням?
C Яка система регулювання є найбільш ефективною?
- 3 На які основні класи поділяються системи автоматичного регулювання?
A До якого класу відноситься лінійна система?
B На які підкласи поділяється клас "характер функціонування"?
C Що являє собою клас "характер подання сигналів"?

3. ЗАСТОСУВАННЯ МІКРОПРОЦЕСОРІВ В СИСТЕМАХ АВТОМАТИЧНОГО КЕРУВАННЯ

Дуже жорсткі масогабаритні вимоги до керуючих засобів мехатронних систем роблять актуальним застосування в них однокристальних мікропроцесорів. МП містять достатньо потужні й гнучкі керуючі ресурси для вирішення більшості завдань нижнього або середнього рівнів керування. Вони обов'язково мають канал зв'язку з більш високим рівнем керування. При цьому мінімальні за габаритами і легко можуть бути спеціалізовані під конкретну задачу. Тому мікропроцесори – основний засіб керування в системах вбудованого керування, широко застосовуваних у мехатроніці і робототехніці. Цим обумовлений вибір мікропроцесорів в якості засобів керування для задач, що розглядаються в даному навчальному посібнику.

3.1. Типова мікропроцесорна система керування

Типовою мікропроцесорною системою керування можна вважати систему з найбільш типовими складом, структурою і архітектурною організацією. Її представлення показує загальні принципові особливості в побудові мікропроцесорних систем.

Далі розглянуті структурна організація апаратних і програмних засобів типової мікропроцесорної системи керування, а також початкові кроки в її проектуванні.

Структура апаратних засобів типової мікропроцесорної системи керування представлена на рис. 3.1. В її складі є пов'язані між собою загальною шиною: мікропроцесор, пам'ять та інтерфейсні засоби.

Процесор – головний активний елемент шини. Забезпечує інтерпретацію команд, записаних в пам'яті програм, реалізуючі тим самим програмне керування об'єктом.

Основна пам'ять – місце зберігання програм і даних, з якими оперують програми.

Інтерфейсні засоби забезпечують сполучення з об'єктом керування, пультом і віддаленими мікропроцесорними системами різного рівня (комп'ютерами або мікроконтролерами). Драйвери кожного з інтерфейсів утворюють їх програмні оболонки, дозволяючи програмісту верхнього рівня використовувати всі можливості інтерфейсів, не вникаючи глибоко в особливості функціонування їх апаратної частини.

Інтерфейсні засоби можуть забезпечити взаємодію зовнішніх пристроїв з процесором двома способами: 1) через програмне опитування; 2) через переривання.

Для взаємодії через програмний опитування інтерфейси мають реєстри стану, де відображені прапори готовності зовнішніх пристроїв. Зчитуючи ці прапори при виконанні відповідних команд, процесор може синхронізувати свої дії зі станом готовності зовнішніх пристроїв.

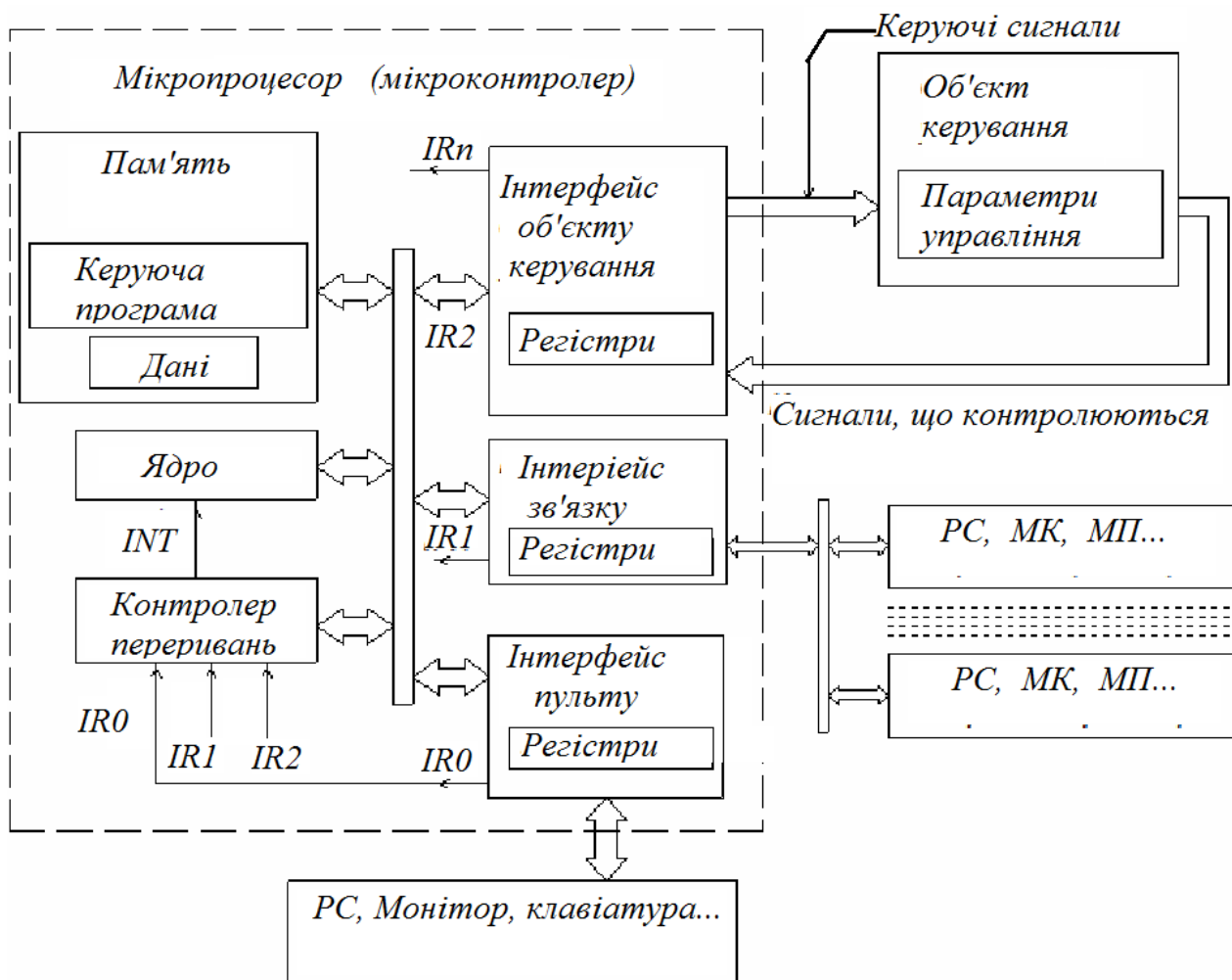


Рисунок 3.1 – Структура апаратних засобів типової мікропроцесорної системи керування

Для взаємодії через переривання в МП організований централізований механізм переривання на основі контролера переривань. Контролер переривань забезпечує переривання процесора сигналом INT за фактами приходу на його входи запитів переривань від інтерфейсів зовнішніх пристроїв: IR0 – від пульта, IR1 – від зв'язку, IR2 – від об'єкта керування.

Інтерфейс сполучення з об'єктом керування, інтерпретуючи отримані від процесора байти як команди, формує керуючі сигнали для впливу на об'єкт. Разом з тим цей інтерфейс сприймає в свої вхідні регістри контрольовані сигнали, що характеризують стан об'єкта. Процесор має можливість виконувати програмне опитування стану вхідних регістрів інтерфейсу, щоб через них оцінити стан об'єкта. Інтерфейс може також забезпечувати генерацію сигналу переривання процесора IR2 в залежності від стану контрольованих сигналів, щоб процесор міг оперативно перейти на підпрограму, призначену для відпрацювання переривання, тобто відреагувати на подію в реальному часі зміни параметрів, що контролюються.

Інтерфейс зв'язку з ЕОМ верхнього рівня забезпечує мікропроцесору на-рощування керуючих можливостей за рахунок підключення через канал послі-довної зв'язку інших мікропроцесорів або комп'ютерів. У складі контролера ін-терфейсу зв'язку є: буферний регістр для прийому в нього бітів байта з лінії зв'язку і буферний регістр для передачі бітів поміщеного в нього байта в лінію зв'язку. Генерація запиту переривання інтерфейсу зв'язку IR1 можлива як при прийомі, так і при передачі. При прийомі переривання виникає з приходом з лі-нії зв'язку останнього біта байта (в момент завершення заповнення приймаль-ного буфера). Це робиться, щоб своєчасно залучити процесор до прийому з приймального буфера прийшов туди байта. При передачі переривання генеру-ється за фактом передачі в лінію зв'язку останнього біта байта (в момент звіль-нення передавального буфера від переданих бітів чергового байта), щоб залу-чити процесор до заповнення передавального буфера наступним новим байтом.

Інтерфейс пульта служить для сполучення з пультом і організації через пульт взаємодії системи керування з оператором (користувачем). Завдання ін-терфейсу пульта – відповідним чином відпрацьовувати команди оператора і формувати повідомлення для оператора на дисплеї. Контролер інтерфейсу пульта забезпечує формування запиту переривання IR0 по кожному натискання клавіші для можливості привернути увагу процесора до дії оператора. Для ке-рування дисплеєм контролер інтерфейсу пульта має регістри, в які можна запи-сати коди повідомлення, а також можливість сканування цього повідомлення на індикаторах дисплея. Таким чином, інтерфейс пульта можна вважати основною частиною коштів підтримки діалогу системи керування з оператором, яка нази-вається операційним інтерфейсом. Розробник операційного інтерфейсу, спира-ючись на драйвер інтерфейсу пульта, проектує системне програмне забезпечен-ня, яке дає користувачеві можливість реалізувати всі необхідні функції програ-много керування об'єктом в максимально простою та зручною для нього формі.

Таким чином, з наведеної на рис. 3.1 структури видно, що основним (ба-зовим) елементом системи керування є мікропроцесор. В даному випадку, мік-ропроцесор – це мікро ЕОМ з високим ступенем інтеграції керуючих можливо-стей в одному кристалі, орієнтована своєю архітектурою на використання в си-стемах вбудованого керування.

3.2. Структура програмного забезпечення

У структурі програмного забезпечення мікропроцесорної системи керу-вання (рис. 3.2) є три рівні: розробника інтерфейсу, розробника операційної си-стеми і користувача.

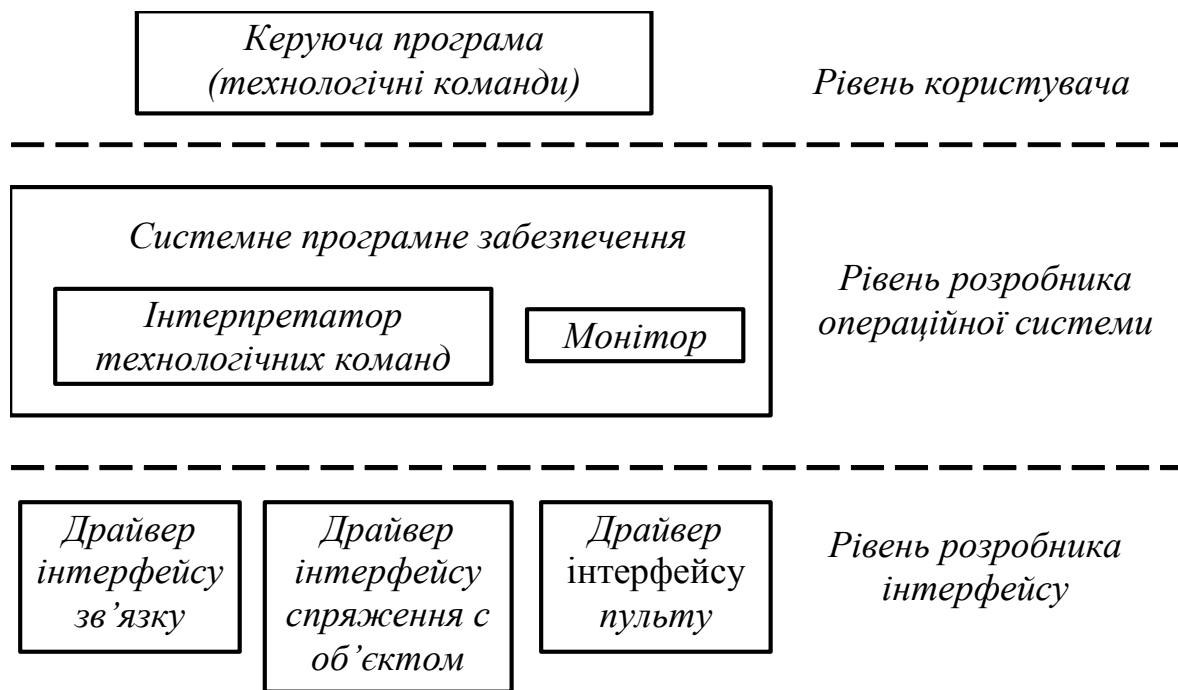


Рисунок 3.2 – Структура програмного забезпечення мікропроцесорної системи керування

Рівень розробника інтерфейсу – найнижчий, близький до апаратних засобів. На цьому рівні знаходяться драйвери інтерфейсів всіх зовнішніх пристроїв.

Кожен драйвер проектується як програмний модуль у складі інтерфейсу, тобто як невід'ємна його частина, невідривна від апаратного блоку. Завдання драйвера – спростити доступ до керованого об'єкту шляхом переходу від програмування на рівні сигналів (яке унікально для кожного керованого об'єкта) до програмування на рівні команд універсальної структури. Тому можна вважати, що драйвер адаптує унікальні особливості об'єкта керування до системного програмного забезпечення мікропроцесора. За своїм місцем серед інших програмних засобів драйвер є модулем, функціонально орієнтованим на реалізацію програмування дій з керування об'єктом. Ці дії організовуються як команди драйвера, тобто кожній керуючій дії ставиться у відповідність окрема команда драйвера, а набору управляючих впливів відповідає набір команд, що зветься системою команд драйвера. Тому завдання драйвера зводиться до того щоб інтерпретувати команди, направлені до нього з більш високого рівня.

На рівні операційної системи реалізовані основні дії щодо програмного керування роботою мікропроцесора в рамках поставленого завдання.

Монітор – програмний засіб керування діалогом з користувачем через пульт. Відразу при включенні монітор бере на себе керування процесором і супроводжує користувача у всіх його взаємодіях з системою керування. Склад модулів монітора і їх функції залежать від поставленого завдання керування об'єктом.

Для забезпечення програмного керування на рівні технологічних команд обов'язковою компонентою системного програмного забезпечення буде модуль

інтерпретатора технологічних команд. У цьому випадку спочатку розробляється так звана система технологічних команд керування даним об'єктом. Її завдання дати можливість оператору найпростішим чином програмувати потрібні йому руху приводу і інші технологічні операції, уникаючи, при цьому, застосування мови машинних команд мікропроцесора. Модуль інтерпретатора виконує роль програмного процесора. Він використовує в якості вхідного параметра код чергової команди керування, ідентифікує її, а потім по результату ідентифікації вибирає послідовність дій, відповідних цій команді і організованих у вигляді підпрограм в машинних командах мікропроцесора.

Самий верхній рівень програмного забезпечення – рівень користувача. На ньому передбачається програмування користувачем на мові технологічних команд. Користувач розробляє керуючу програму, що реалізовує необхідний йому алгоритм керування об'єктом. Далі користувач, за підтримкою монітора, вводить програму в пам'ять мікропроцесора і запускає її на виконання. Інтерпретатор технологічних команд забезпечує послідовне відпрацювання керуючої програми і реалізацію таким чином керуючих дій заданого алгоритму.

3.3. Початковий етап проектування.

Першим кроком в проектуванні, відразу після визначення особливостей об'єкта і завдання керування, є вибір мікропроцесора, здатного за своїми функціональними можливостями впоратися з вирішенням задачі керування і при цьому є мінімальним за габаритами, енергоспоживанням і вартості. Ринок таких мікропроцесорів широкий і різноманітний. Це різні за архітектурою і функціональними можливостями пристрої.

Кожен контролер, з одного боку, вже містить у своєму складі багато апаратних інтерфейсних засобів для підключення об'єктів, засобів зв'язку і пультів, а з іншого боку, вимагає невеликої (в різному ступені залежною від конкретного завдання) побудови апаратної частини цих інтерфейсів. Головне ж завдання розробника – розробка програмного забезпечення для мікропроцесорів, тобто драйверів і системного програмного забезпечення.

Далі, після вибору базового мікропроцесора, завдання проектування поділяється на три часткові задачі, вирішення яких може здійснюватися паралельно, хоча і взаємозалежно:

- розробка інтерфейсу сполучення з об'єктом;
- розробка інтерфейсу зв'язку з комп'ютером верхнього рівня;
- розробка операційного інтерфейсу.

Етапи проектування апаратних і програмних засобів МПСК зображені на рисунку 3.3.

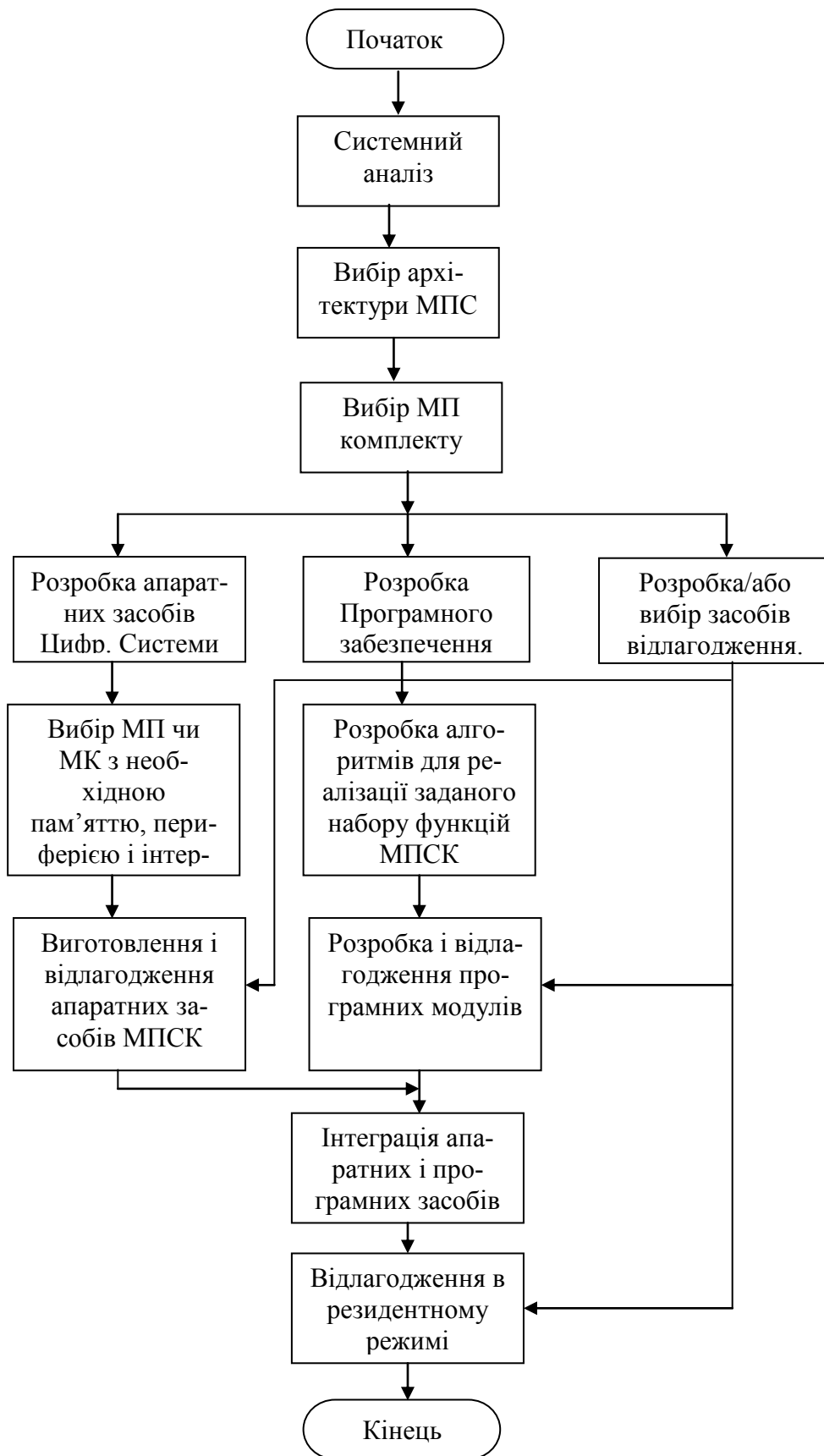


Рисунок 3.3 – Етапи проектування мікропроцесорної системи керування

Одним з найважливіших етапів є системний аналіз, мета якого полягає у визначенні вимог до проектованої системи, а саме:

- Визначити набір функцій, які повинні виконуватися за допомогою системи;
- Необхідну продуктивність;
- Межі вартості на етапах розробки, виготовлення, обслуговування системи;
- Уточнюються критичні функції відповідно до критеріїв розроблюваної системи;
- Проаналізувати склад периферійних пристроїв і режими їх роботи;
- Встановити характер зв'язків окремих функцій (підзадач) системи, наприклад, послідовні, паралельно-послідовні, зв'язок з переривання;
- Визначити структуру і "походження" даних, а також вимоги до вихідних даних і необхідність їх зберігання.

Вибір архітектури – визначаються вимоги до архітектури системи і попередньо вирішується завдання реалізації виділених функцій програмним і апаратним способом. Часто проблема полягає в тому, що приходить проводити синтез модулів для реалізації функцій апаратним шляхом через відсутність в МП комплектах відповідних ВІС.

Вибір мікропроцесорного засобу складається з трьох основних моментів:

1. З точки зору розробки ПЗ необхідно аналізувати такі характеристики МП як набір команд і способи адресації, розрядність, число використовуваних регістрів загального призначення, тип стекової пам'яті, способи обробки переривань і т.д.

2. З точки зору системного проектування, необхідно аналізувати повноту комплексу, який повинен містити крім МП, ВІС ПЗП і ОЗП, модулі для організації інтерфейсу з периферійними пристроями, контролери для обробки переривань, керуванням прямим доступом до пам'яті шинні формувачі, буферні регістри, тактовий генератор, системний контролер.

3. З точки зору розробки та налагодження ПЗ необхідна наявність таких програм, як транслятор з символічної мови на двійковий код (об'єктний), редагуюча програма, програма моделювання, без яких сума і вартість розробки МПС значно збільшуються. Вони називаються крос-засобами для розробки програмного забезпечення проектованої мікропроцесорної системи.

3.4. Приклади застосування мікропроцесорних систем контролю та керування.

Пристрій керування обладнанням з кроковим приводом. В даний час широко застосовуються верстати з числовим програмним керуванням (ЧПК), де створена на комп'ютері деталь перетворюється в набір команд для верстата, який дотримуючись їх самостійно виготовляє деталь. Участь людини в виробництві зводиться до установки заготовки, сервісного обслуговування та прийняття рішень у виняткових ситуаціях. Такий підхід дозволяє виготовляти велику

кількість ідентичних деталей високої якості, а для зміни параметрів деталі потрібно тільки змінити модель деталі на комп'ютері, без виготовлення форм або переробки обладнання. Так само виключається людський фактор, що важливо при серійному виробництві.

Верстати з ЧПК можуть бути найрізноманітніші: від простих свердлувальних і токарних, до складних фрезерувальних. Для переміщення робочого інструменту використовують крокові двигуни або сервоприводи. Переміщення відстежується системою зворотного зв'язку. Таким чином, контролеру ЧПК потрібно приймати команди від комп'ютера (наприклад в G-кодi), інтерпретувати їх в команди для драйверів двигуна і коригувати відповідно до сигналів зворотного зв'язку, і все це одночасно для всіх координат.

Об'єкт керування та постановка задачі. Об'єкт керування являє собою двох координатний привід свердлувального обладнання для виробництва друкованих плат. Приводи обох координат (рис. 3.4) ідентичні. Кожен з них забезпечує поступальне переміщення по координаті в обраному напрямку із заданою швидкістю ω робочого органу маніпулятора (РОМ). Результат поступального руху є переміщення РОМ в положення θ по даній координаті.

Обертання ротора крокового двигуна координати забезпечується почерговими включеннями його обмоток імпульсами струму M_{xi} (M_{yi}). Імпульси подаються на обмотки або в прямій послідовності M_{x1} , M_{x2} , M_{x3} , M_{x4} , M_{x1} , M_{x2} т. Д., Або в зворотній послідовності M_{x4} , M_{x3} , M_{x2} , M_{x1} , M_{x4} , M_{x3} т. і. З кожним новим імпульсом двигун повертається на однаковий елементарний кут або крок. Напрямок повороту залежить від порядку проходження імпульсів. Пряма послідовність забезпечить напрямок обертання двигуна в одну сторону, а зворотна - в протилежну. При цьому швидкість обертання буде визначатися частотою проходження імпульсів обмоток.

Засоби сполучення з об'єктом

Апаратура інтерфейсу сполучення з кроковими приводами координат засобами програмованого таймера і дискретного виводу повинна буде забезпечити реалізацію наступних двох функцій:

1. Формування послідовності імпульсів керування рухом X_v , X_n , Y_v , Y_n . Частота і кількість цих імпульсів повинні бути програмовані;
2. Розподіл імпульсів по обмоткам крокової двигуна в залежності від необхідного напрямку його обертання (реалізація розподільника імпульсів).

Один з варіантів формування імпульсів керування рухом представлений на рис. 3.5. Таймер РТ1 мікросхеми програмованого таймера застосований в ролі генератора імпульсів. Генерація імпульсів таймера РТ1 дозволяється активним рівнем сигналу Е, що виробляється через лінію РА0 порту А мікросхеми РРІ. Імпульси Х с його виходу Out1 логічними ключами можуть комутуватися на лінії імпульсів керування рухами X_v або x_n . Напрямок проходження імпульсів Х залежить від значення сигналу D, який формується з виходу мікросхеми РРІ.

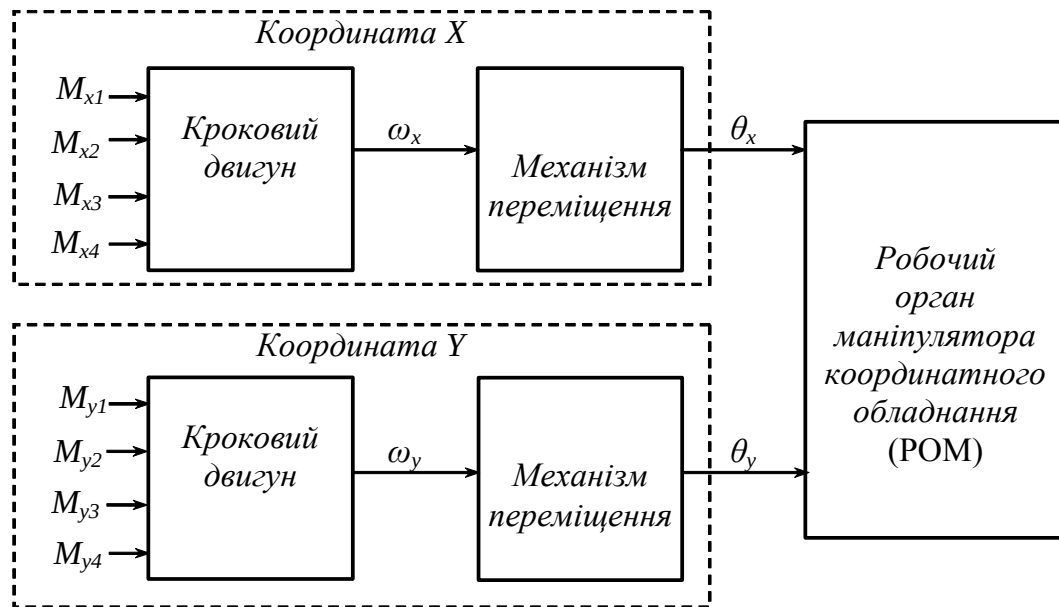


Рисунок 3.4. – Структурна схема об'єкта керування: $M_{x1..4}$, $M_{y1..4}$ - імпульси на обмотки крокових двигунів координат X , Y ; ω_x , ω_y - швидкості обертання двигунів координат X і Y , θ_x , θ_y - переміщення по координатах X і Y

Імпульси X надходять також на тактовий вхід таймера $PT0$. Цей таймер працює в рахунковому режимі. Він вирішує задачу контролю потрібної кількості імпульсів руху. Необхідна кількість імпульсів руху, відповідна заданому переміщенню приводу, завантажується перед початком видачі імпульсів в лічильник таймера $PT0$. З кожним відправленим імпульсом X цей лічильник зменшує своє значення на 1. А коли він дорахував до 0, зміниться стан його виходу $Out0$, тобто сигнал $Out0$ може сигналізувати про видачу запрограмованої кількості імпульсів руху. Тому сигнал $Out0$ з'єднано із входом $IR1$ контролера переривань і використаний для переривання процесора мікроконтролера.

Таким чином, програмуванням таймера $PT1$ можна визначати необхідну частоту обертання двигуна, значенням сигналу D - напрямок його обертання, а сигналом E - дозволити або забороняти рух. При цьому, програмуванням таймера $PT0$ можна управляти величиною переміщення приводу (рис.3.5).

Для реалізації розподілу імпульсів по обмотках крокового приводу необхідно виконати наступну схему. Спочатку вибрати відповідний кроковий двигун, визначитися з кількістю його обмоток і необхідну послідовність подачі імпульсів на ці обмотки. Потім розробити електронну схему розподільника імпульсів, наприклад, на основі включених послідовно лічильника, дешифратора і підсилювача потужності.

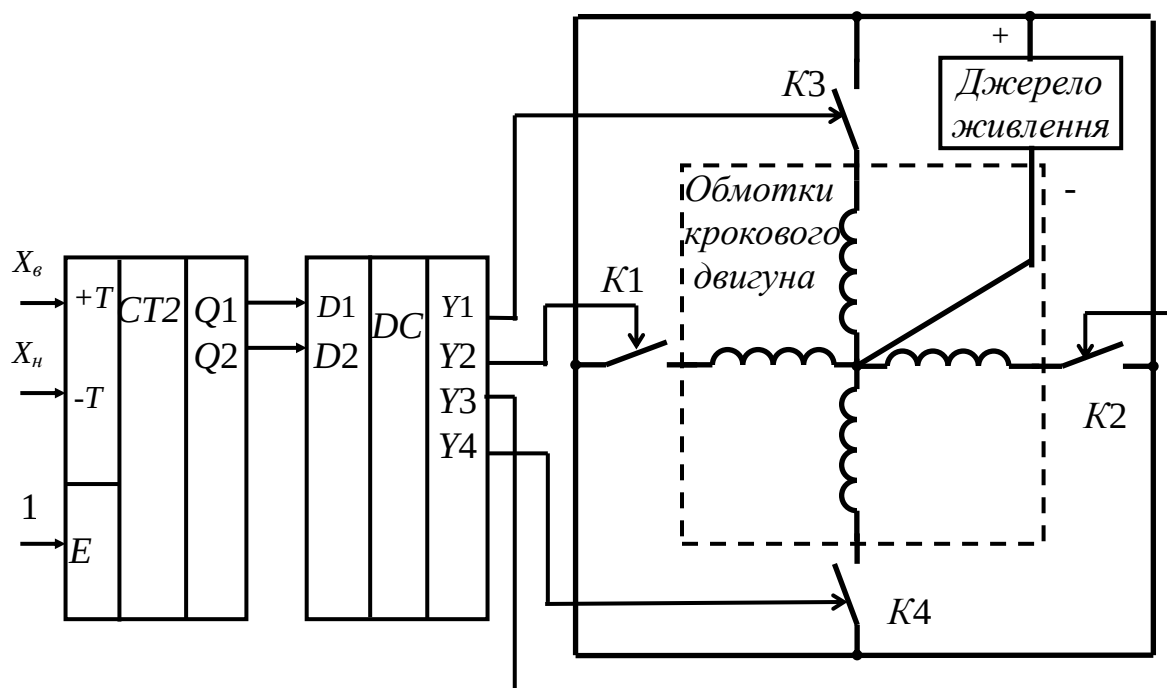


Рисунок 3.6 – Функціональна схема розподілення імпульсів: $X_{в}$, $X_{н}$ – імпульси руху для координати X вперед и назад; $CT2$ – рахівник; DC – дешифратор; $K1...K4$ – ключі включення обмоток двигуна

Операційний інтерфейс. Постановка задачі програмного керування на концептуальному рівні представлена у вигляді структурної схеми (рис. 3.7). Виходячи з неї, в складі операційного інтерфейсу будуть потрібні наступні програмні модулі:

1. Головне меню.
2. Редагування програми пристрою керування (ПК).
3. Введення точок.
4. Ручний.
5. Робота по точках.
6. Автомат.

Крім перерахованих модулів потрібен ще модуль інтерпретатора який буде викликатися з модулів 5 і 6.

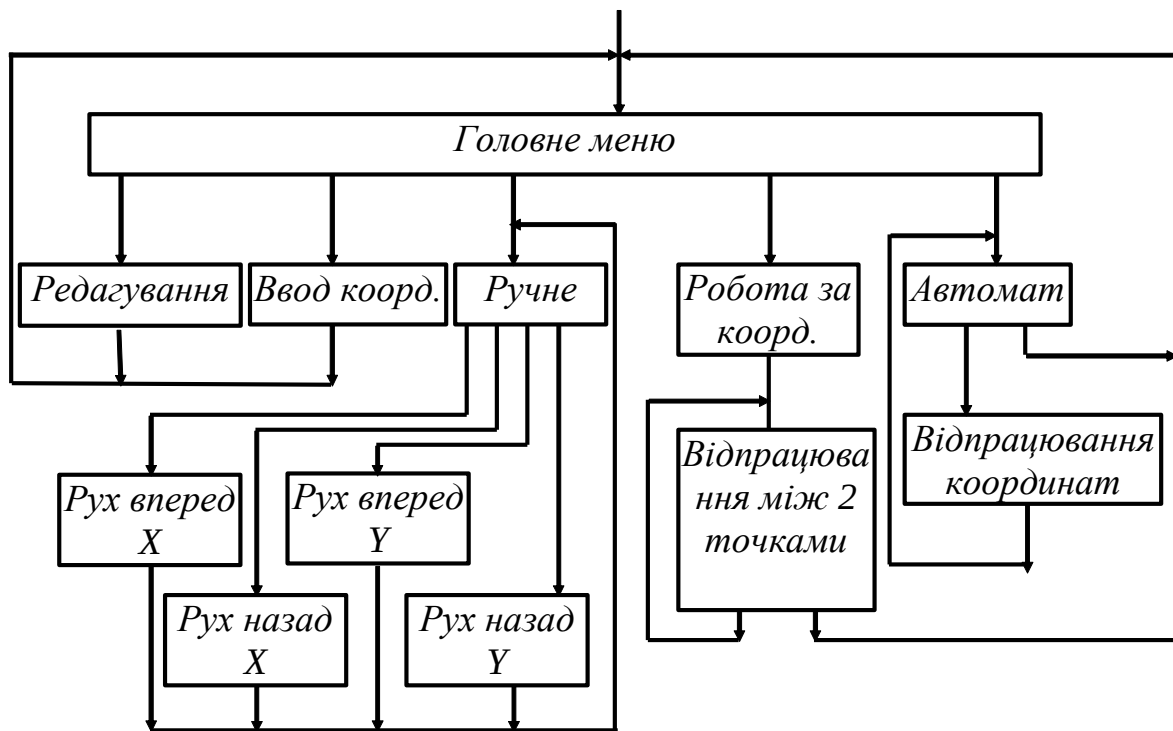


Рисунок 3.7 – Структурна схема концептуального представлення задачі

Головне меню. У цьому програмному модулі повинен бути забезпечений висновок на індикацію цифр, вказувати на клавіші, натискання яких відповідає ініціації переходу на відповідний режим роботи системи: 1 - редагування; 2 - введення точок; 3 - ручний; 4 - робота по точках; 5 - автомат; 6 - вихід з програми.

Ручне керування. Цей програмний модуль повинен забезпечити керування рухами маніпулятора від чотирьох клавіш пульта.

Робота по точках. Цей програмний модуль повинен забезпечити відпрацювання рухів, запрограмованих в ПК, таким чином, що після переміщення в нову точку траєкторії, програма зупиняється і для продовження очікує від оператора натискання будь-якої клавіші. Після натискання виконуються наступні команди ПК до приходу маніпулятора в наступну точку. Далі - знову очікування натискання, і так далі все повторюється до кінця УП.

Автомат. У цьому програмному модулі повинна бути забезпечена відпрацювання ПК від першої команди до останньої без зупинок

Маніпулятор з приводом постійного струму з керуванням від комп'ютера.

Об'єкт керування та постановка задачі. Об'єкт керування являє собою двох координатний маніпулятор. Структурно об'єкт керування представлений на рис. 3.8.

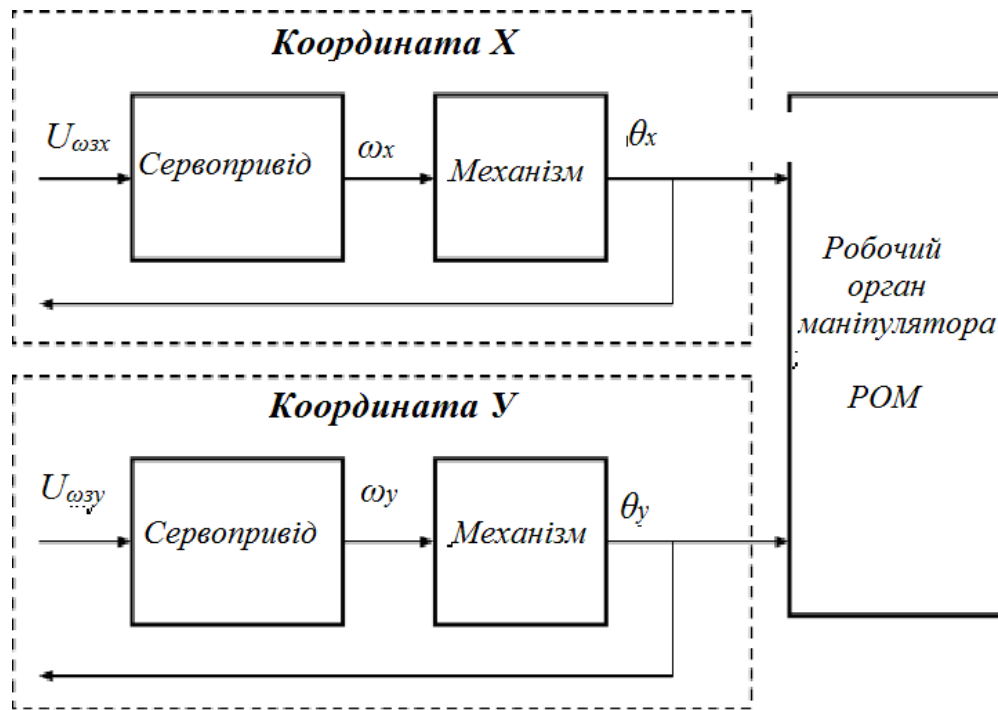


Рисунок 3.8 – Структурна схема об'єкта керування: $U_{\omega z}$ – сигнал завдання швидкості; ω – швидкість обертання двигуна приводу координати; θ – положення робочого органу маніпулятора за координатою

Сервоприводи координат X і Y ідентичні за своїм виконанням. Кожен з них незалежно забезпечує поступальний рух робочого органу маніпулятора (РОМ) за своєю координаті зі швидкістю ω , яка визначається сигналом завдання швидкості $U_{\omega z}$. Поступальний рух РОМ є результат перетворення редуктором механізму вхідного обертального руху вала електродвигуна. А результат поступального руху є переміщення РОМ в положення θ по даній координаті.

Функціональна схема системи керування однієї координати представлена на рис. 3.9. На ній, перш за все, детально показаний об'єкт керування для координати – сервопривід. У складі мікроконтролера вказані тільки додаткові засоби для сполучення з сервоприводом.

Сервопривід забезпечує підтримку відповідності між завданням швидкості $U_{\omega z}$ і реальною швидкістю ω обертання вала двигуна M . Він побудований за класичною схемою двоконтурного підлеглого керування.

Контур швидкості є зовнішнім контуром для контуру струму. Він шляхом формування завдання U_{Iz} для контуру струму управляє діями цього внутрішнього контуру. Підлеглий контур швидкості контур струму цілком входить до складу об'єкта регулювання контуру швидкості. Крім нього до складу контуру швидкості входять: механічна частина двигуна, сенсор швидкості ДС (тахогенератор) і регулятор швидкості РС.

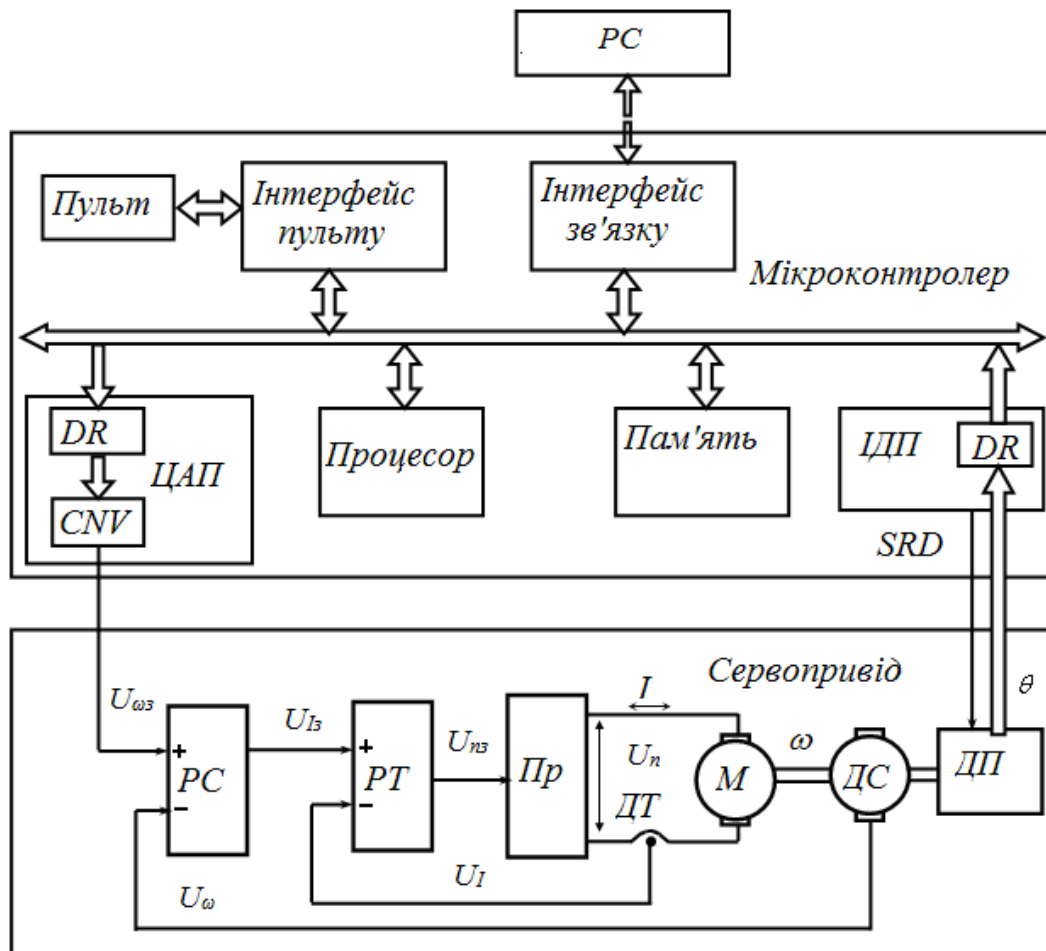


Рисунок 3.9 – Функціональна схема системи керування однієї координати: ЦАП – цифро-аналоговий перетворювач; ІДП – інтерфейс детектора положення; РС – регулятор швидкості; РТ – регулятор струму; Пр – перетворювач; ДС – сенсор швидкості; ДП – детектор положення; ДТ – детектор струму.

Якірну напругу двигуна M формує перетворювач $Пр$ (широотно-імпульсний або тиристорний). Він посилює потужність вхідного сигналу керування U_{nz} і виробляє пропорційну йому напругу U_n для живлення якоря двигуна. Внутрішній контур регулювання – контур струму. Він складається з якоря двигуна M , перетворювача $Пр$ і регулятора струму $РТ$. Контур струму, за допомогою варіювання якірним напругою, забезпечує струм якоря пропорційним завданням струму на вході контуру струму U_{Iz} . Досягається це за рахунок наявності негативного зворотного зв'язку по струму якоря: сенсор струму перетворює вхідний сигнал струму якоря в пропорційний йому сигнал напруги U_I ; різницю $U_{Iz} - U_I$ (неузгодженість) подається на вхід $РТ$; далі регулятором виробляється такий сигнал U_{nz} (при тому перетворювачем виробляється відповідне U_n), щоб оптимальним чином усунути неузгодженість або зробити відміну струму якоря від заданого струму мінімальним.

Контур швидкості, за допомогою варіювання струмом якоря (через керування контуром струму) забезпечує швидкість обертання двигуна ω пропорційною завданням швидкості $U\omega_z$. Досягається це за рахунок застосування негативного зворотного зв'язку за швидкістю. Датчик швидкості виробляє напругу $U\omega$ пропорційно швидкості обертання двигуна. Неузгодженість швидкості $U\omega_z - U\omega$ надходить на вхід РС, яка виробить $U\omega_z$ таким, щоб струм якоря забезпечив момент обертання на валу двигуна, який зменшить або збільшить швидкість його обертання для усунення даного неузгодженості.

Таким чином, головною особливістю керованого об'єкта є аналоговий характер керуючих сигналів $U\omega_z$ і $U\omega$. Значить потрібні цифроаналогові перетворювачі ЦАП, які треба буде спроектувати і додати до апаратних засобів базових мікроконтролерів.

Інша особливість об'єкта пов'язана з необхідністю контролю поточного стану по кожній координаті. Тому потрібно реалізувати канал зворотного зв'язку по положенню на основі сенсора положення та інтерфейсу в складі мікроконтролера.

3.5 Програмно-апаратні засоби сполучення з об'єктом

Для регулювання положення РОМ будуть потрібні програмно-апаратні засоби, що реалізують цифровий контур положення. Це означає, що апаратні засоби для підтримки реалізації контуру положення повинні будуть забезпечити формування: 1) інформації про поточні положення по координатам; 2) аналогових сигналів завдання швидкостей.

Інформація про поточний стан РОМ повинна бути представлена в дискретної формі у вигляді двійкового коду. Інтерфейс сенсора положення повинен забезпечувати:

- 1) вибір сенсора положення;
- 2) читання даних обраного сенсора із записом в регістр інтерфейсу;
- 3) буферне зберігання кодів положень за координатами в регістрах інтерфейсу;
- 4) можливість програмного опитування мікроконтролером значень цих кодів.

Засіб формування аналогового сигналу завдання швидкості $U\omega_z$ - цифроаналоговий перетворювач. Двійкові числа на вході ЦАП повинні бути представлені у формі додаткового коду.

Один з можливих варіантів підключення сенсорів положення і ЦАП через паралельні інтерфейси $PPI1$ і $PPI2$ показаний на рис. 3.10.

Для прикладу обрано 12-розрядні ЦАП, що мають в складі разом з перетворювачем CNV також і вхідний регістр DR для фіксації вхідного коду. Вхідний код завдання швидкості виводиться через порти А (8 молодших розрядів) і С (4 старших розряду). Ці лінії використовуються спільно обома ЦАП. У момент установки коду для ЦАПх на RA0-7 і PC4-7 повинен бути програмно сформований сигнал. Тоді код буде записаний в регістр DR ЦАПх, надійде на вхід

перетворювача CNV, який сформує йому відповідний вихідний аналоговий сигнал U_{w3x} . Там, де необхідно встановити інший вихідний аналоговий сигнал U_{w3y} , відповідний йому код виводиться на PA0-7 і PC4-7, потім повинен бути програмно сформований сигнал.

П'ятнадцятирозрядні кодові сенсори положення координат також об'єднують свої вихідні лінії в загальну шину, яка приходить на входи портів А (8 молодших розрядів) і В (7 старших розрядів) PPI2. Вибір сенсора, що підключається до шини, забезпечується сигналом, програмно формується на виході PC0 порту С.

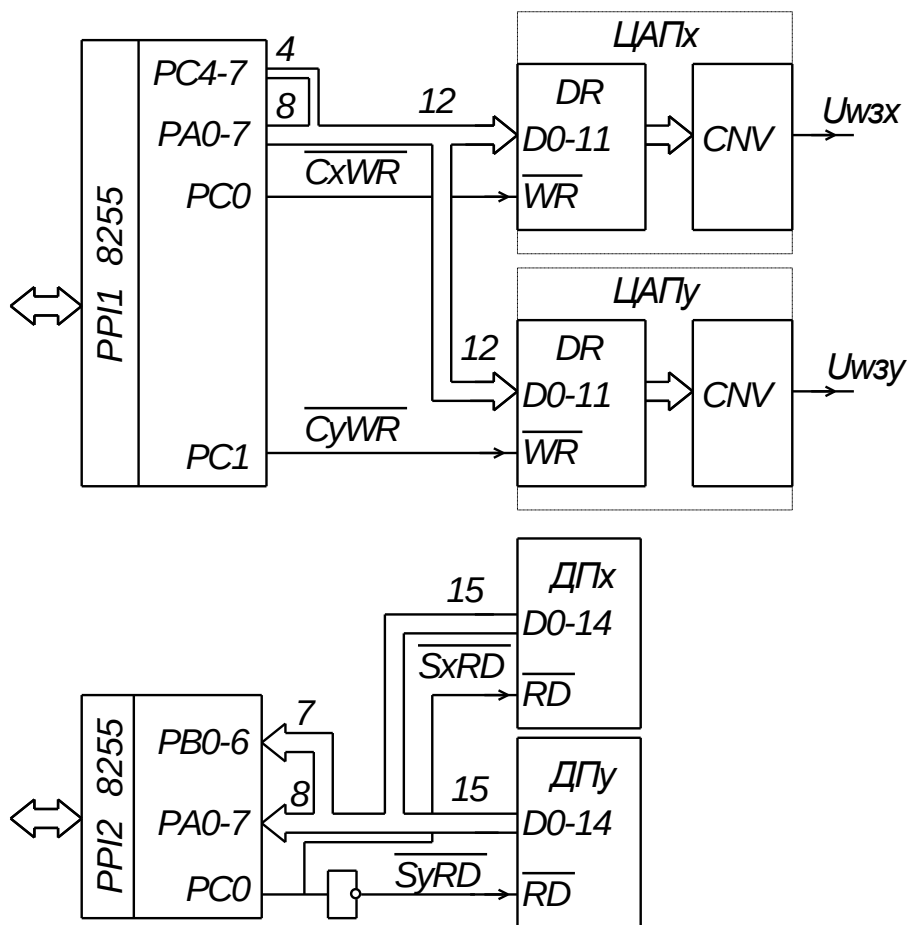


Рисунок 3.10 – Варіант підключення сенсорів положення і ЦАП: ДП – детектор положення; ЦАП – цифро-аналоговий перетворювач; DR – регістр даних; CNV – перетворювач; PPI - паралельний інтерфейс

Операційний інтерфейс.

Постановка завдання програмного керування на концептуальному рівні представлена у вигляді структурної схеми (рис. 3.11).

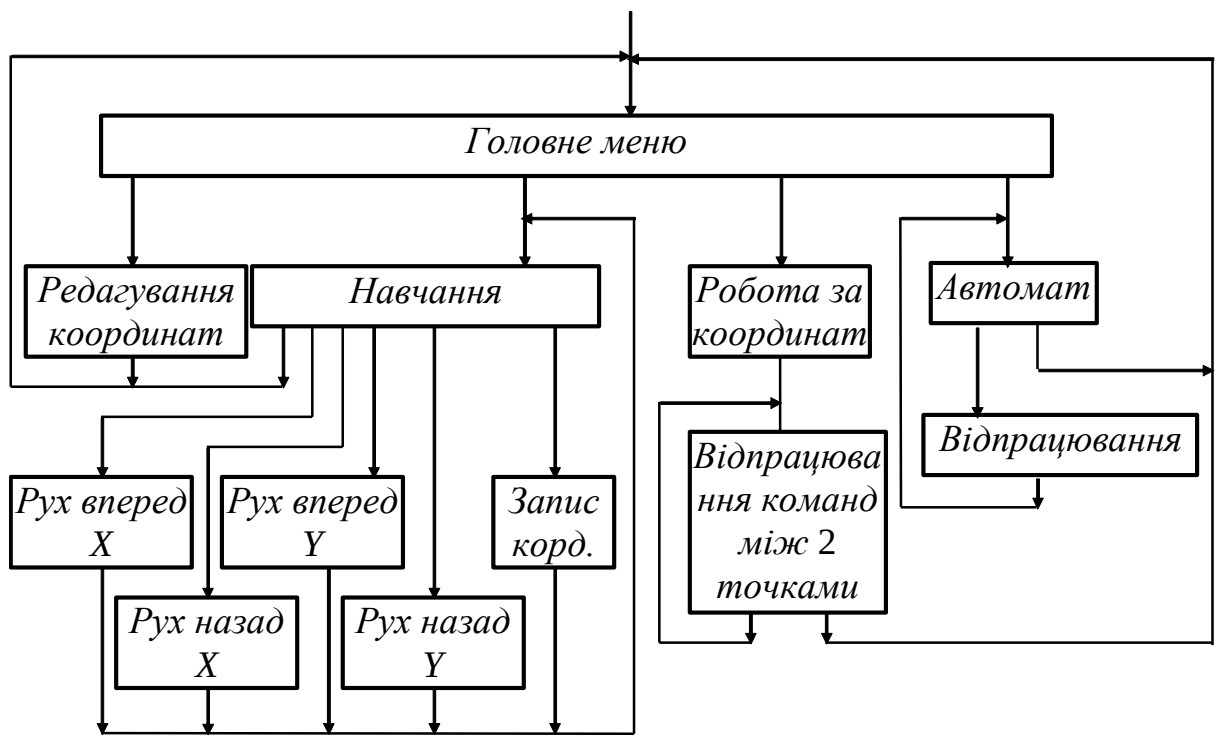


Рисунок 3.11 – Структурна схема концептуального представлення задачі

Виходячи з структури, в складі операційного інтерфейсу будуть потрібні наступні програмні модулі:

1. Головне меню.
2. Редагування ПК.
3. Навчання.
4. Робота по точкам.
5. Автомат.

Крім перерахованих модулів потрібно ще модуль інтерпретатора, який викликається з модулів 4 і 5.

Виводи. У загальному вигляді задача на проектування полягає в розробці одного, або декількох інтерфейсів в системі керування заданим об'єктом.

Керування у кожному завданні здійснюється на основі ресурсів будь-якого з типів мікропроцесорних пристроїв та периферійних засобів, що забезпечують функції збору, первинної обробки та передачі даних до засобу керування.

3.6 Контрольні запитання і завдання

1. Чим обумовлений вибір мікропроцесорів в якості засобів керування?
2. Що вважається типовою мікропроцесорною системою керування?
3. За що відповідають інтерфейсні засоби систем керування?
4. Якими способами інтерфейсні засоби можуть забезпечити взаємодію зовнішніх пристроїв з процесором?

5. В чому полягає функція інтерфейсу сполучення з об'єктом керування?
6. Які функції забезпечує інтерфейс зв'язку.
7. Приведіть структуру апаратних засобів типової мікропроцесорної системи керування.
8. Які рівні існують в структурі програмного забезпечення мікропроцесорної системи керування?
9. Дайте характеристику рівню розробника інтерфейсу.
10. Етапи проектування мікропроцесорної систем керування.
11. В чому полягає функція системного аналізу?
12. В чому полягає функція вибору архітектури?
13. Приведіть структуру приводу свердловального обладнання?
14. Що визначає вибір мікропроцесорного засобу?
15. Яка функція операційного інтерфейсу в приведених прикладах?
16. Яка структурна схема об'єкта керування в прикладах?
17. Дайте характеристику рівню користувача.

4. РЕАЛІЗАЦІЯ ЗАСОБІВ КЕРУВАННЯ НА МІКРОПРОЦЕСОРАХ SITARA

Сімейство мікропроцесорів Sitara™ компанії Texas Instruments включає дві лінійки мікропроцесорів, молодшу – на основі ядра ARM9 і старшу - на основі ядра Cortex-A8. Молодша лінійка процесорів з максимальною частотою 456 МГц позиціонується як орієнтована на мале енергоспоживання з можливістю гнучкого вибору необхідного набору інтерфейсів. Старша лінійка процесорів з максимальною частотою до 1,5 ГГц позиціонується як високопродуктивна, орієнтована на системну інтеграцію в рамках сучасних вимог до промислових розробок.

У лінійці на основі ядра ARM9 збільшення функціональних можливостей здійснюється додаванням до початкової конфігурації (AM1705) контролерів LCD, SATA, цифрової відеокамери, а також комунікаційних інтерфейсів. Особливістю цих мікропроцесорів є вбудований блок програмованих інтерфейсів PRU (Programmable Real-time Unit), конфігуруючи який, розробник може додати до наявних інтерфейсів стандартні порти (UART, CAN) або порти з індивідуальним протоколом. Максимальна тактова частота в молодшій лінійці у більшості мікропроцесорів однакова. Молодші моделі AM17xx здатні працювати тільки з пам'яттю SDRAM, а старші - ще й з пам'яттю DDR/DDR2. Базова модель лінійки, мікропроцесор AM1705, має корпус LQFP, інші укладені в корпуси BGA. Структура мікропроцесорів AM1705/1707 приведена на рисунку 4.1.

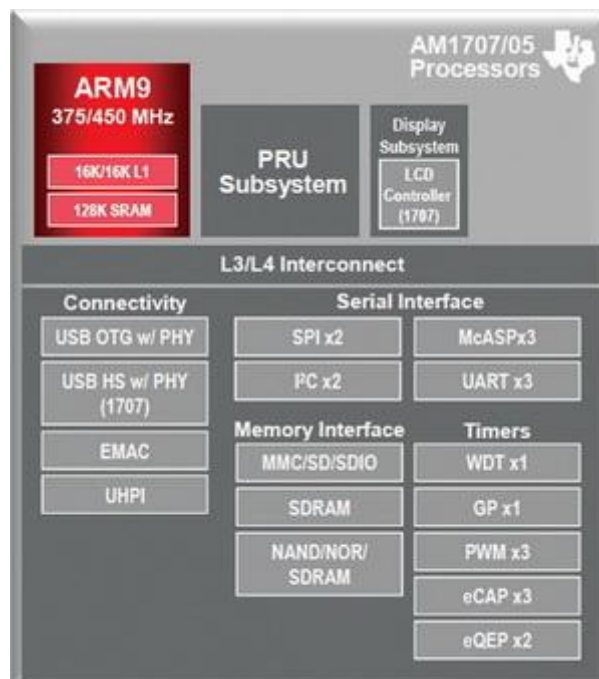


Рисунок 4.1 – Структура мікропроцесорів AM1705/1707

Сімейство мікропроцесорів Sitara™ виробляється компанією Texas Instruments відповідно до концепції масштабованої архітектури, коли в одному

або декількох сімействах присутні як однопоточні процесори з мінімумом інтерфейсів, так і процесори з декількома ядрами, розвиненим набором інтерфейсів, додатковими контролерами. Масштабуються архітектури на програмну, апаратну і топологічну (за виводами корпусів) сумісність між групами мікропроцесорів/мікроконтролерів. В рамках цієї концепції мікропроцесор Sitara™ AM1707 сумісний з OMAP-L137, а мікропроцесор Sitara™ AM1808 - с OMAP-L138 (таблиця 4.1).

У лінійці на основі ядра Cortex-A8 в даний час є чотири групи мікропроцесорів, що відрізняються максимальною тактовою частотою (720 МГц, 600 МГц, 1000 МГц, 1,5 ГГц). У кожній групі по два мікропроцесора, один з графічним прискорювачем, інший без нього.

Розробка систем на основі швидкодіючих 32-розрядних мікропроцесорів трудомістка, вимагає високої кваліфікації і значного досвіду. В системі необхідно забезпечити роботу зовнішньої пам'яті з високою швидкістю обміну, оскільки такі мікропроцесори внутрішньої Flash-пам'яті не мають. Також швидкі мікропроцесори вимогливі до організації живлення, мають багатовивідні корпуси BGA, непрості з точки зору трасування і монтажу. Тому кінцеві вироби часто виготовляються з використанням модулів SOM (System On Module), які містять власне мікропроцесор з системою живлення, зовнішню пам'ять і оціночний роз'єм. Програмне забезпечення при цьому будується на основі операційних систем: Linux, Windows CE та ін.

Таблиця 4.1 – Лінійка процесорів Sitara Arm

	AM335x	AM4x	AM5x	AM6x
Основні ядра	Arm Cortex-A8 ≤ 1 ГГц	Arm Cortex-A9 ≤ 1 ГГц	1-2 Arm Cortex-A15 $\leq 1,5$ ГГц	2-4 Arm Cortex-A53 $\leq 1,1$ ГГц
Додаткові ядра	ні	ні	1-2 C66x DSP ≤ 750 МГц	2-4 Arm Cortex-A53 $\leq 1,1$ ГГц
Підсистема промислової комунікації	2 x промислові порти Ethernet / 1x PRU-ICSS	2 x промислові Ethernet-порти + fieldbus / 2x PRU-ICSS	4x Cortex-M4	2x Cortex-R5F
Мультимедіа	3D-графічний прискорювач	3D-графічний прискорювач	4 x промислові порти Ethernet / 2x PRU-ICSS	6 x Gigabit industrial Ethernet ports / 3x PRU-ICSS
Інші периферійні пристрої	Відеоприскорювач 2x 3D графічні прискорювачі	Відеоприскорювач 2x 3D графічні прискорювачі	PCIe, USB 3.0, CSI-2	Відеоприскорювач 2x 3D графічні прискорювачі
ECC	ні	Паралельна камера	так	PCIe, USB 3.0, CSI-2
	ні	ні	так	так

Деякі особливості сімейства мікропроцесорів Sitara™. Сімейство побудовано для впровадження в промислових об'єктах, що застосовують концепцію Індустрії 4.0. Для цього при розробці головний наголос було зроблено на наступних вимогах та особливостях застосування:

- Довгі виробничі цикли,
- Висока надійність пам'яті з ECC (*Пам'ять з кодом Усунення помилок (англ. Error-correcting code memory, ECC – пам'ять)*),
- Розширений діапазон робочих температур (до 125C),
- Підтримка більш ніж 10 промислових протоколів,
- Комплексна функціональна безпека,
- Функціональна безпека включно з продуктами Hercules.

Підвищена продуктивність системи, захист завантаження, крипто– та інші засоби безпеки:

- 2D/3D графіка, HD відео прискорення,
- Високошвидкісні інтерфейси, такі як USB 3.0 та PCIe,
- Підтримка Wi-Fi підключення за допомогою WiLink 8.

Єдина платформа розробки для всіх сімейств і лінійок:

- Одномісний для багатоядерних ARM процесорів,
- Останній Linux, RTOS та Android у портфоліо,
- Загальне середовище розробки,
- Під'єднання сумісних пристроїв,
- Широка екосистема апаратних та програмних партнерів.

Програмні засоби процесорів Sitara. Процесори Sitara від Texas Instruments надають масштабовані рішення для широкого кола додатків від НМІ (інтерфейс людина – машина) та шлюзів до більш складного обладнання, такого як приводи та обладнання для автоматизації підстанцій та складних виробничих процесів. Процесори Sitara Arm® забезпечують масштабованість апаратної частини і надійність, а також багато підтримку протоколів індустріального зв'язку, таких як EtherCAT, EtherNet/IP та Profinet. Загалом Sitara Arm підтримують більше 11 промислових протоколів у реальному часі через програмований інтерфейс PRU-ICSS (рис.4.2).



Рисунок 4.2 – Підтримка протоколів зв'язку індустріального класу.

Процесори TI Sitara Arm® мають унікальну периферійну підсистему Industrial Communications Industrial Unit (PRU-ICSS), що програмує внутрішні тактові засоби, забезпечуючи інтеграцію протоколів Industrial Ethernet та Fieldbus в реальному часі (рис.4.3). PRU-ICSS (Підпрограма підсистеми промислової комунікації в режимі реального часу) – це підсистема, що містить програмовані сервери реального часу (PRU).

Для PRU-ICSS доступні протоколи Ethernet, Fieldbus, кодера та резервування, включаючи сертифіковані рішення для EtherCAT, Profibus, Profinet та інших. PRU-ICSS підтримує кілька протоколів на одній і тій же апаратній основі, використовуючи прошивку, надану TI, через промислові пакети програмного забезпечення та засоби партнерів в галузі.

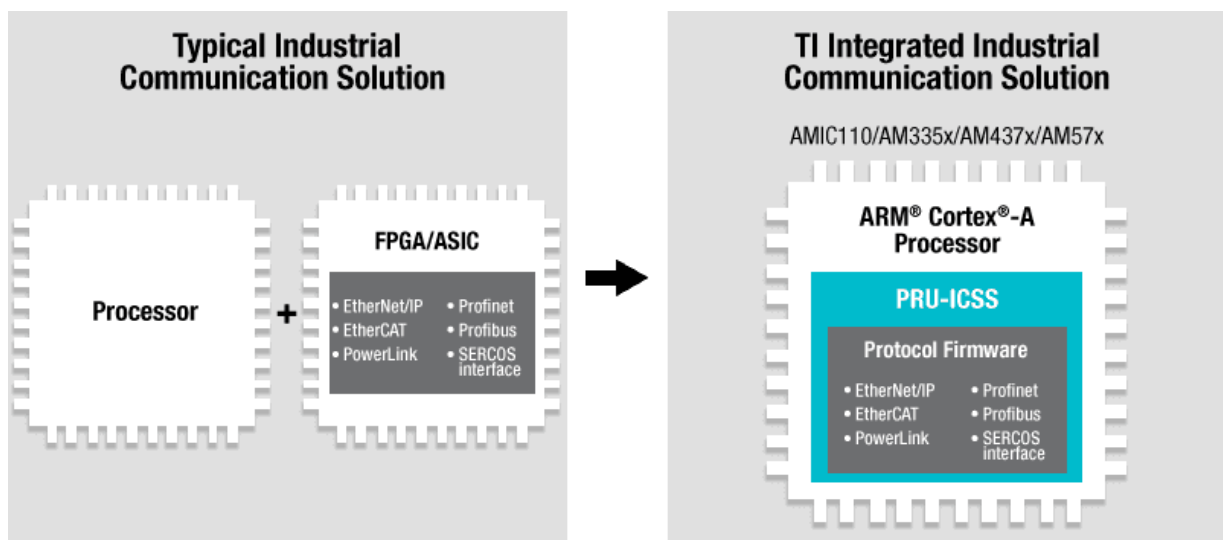


Рисунок 4.3 – Структурна схема периферійної підсистеми PRU-ICSS.

Функції PRU-ICSS та структура підпрограми підсистеми промислової комунікації в режимі реального часу (рис 4.4):

- Підтримка декількох протоколів на одній платформі;
- Протоколи підтримки: PROFIBUS, PROFINET (RT / IRT), SERCOS-III, Ethernet / IP (з DLR), POWERLINK, EtherCAT, CC-Link IE Field Basic та інші;
- Протоколи ведучого та відомого;
- Протоколи резервування, такі як HSR і PRP;
- Протоколи кодера, такі як BiSS C, EnDAT 2.2, Tamagawa і HIPERFACE DSL;
- Стандартний Ethernet 10/100 Мб;
- Повністю протестовані, сертифіковані та підтримувані рішення;
- Спеціальне програмне забезпечення, доступне в складі промислового пакета TI (таблиця 4.2);
- Програмне забезпечення загального призначення доступне від процесорного SDK.

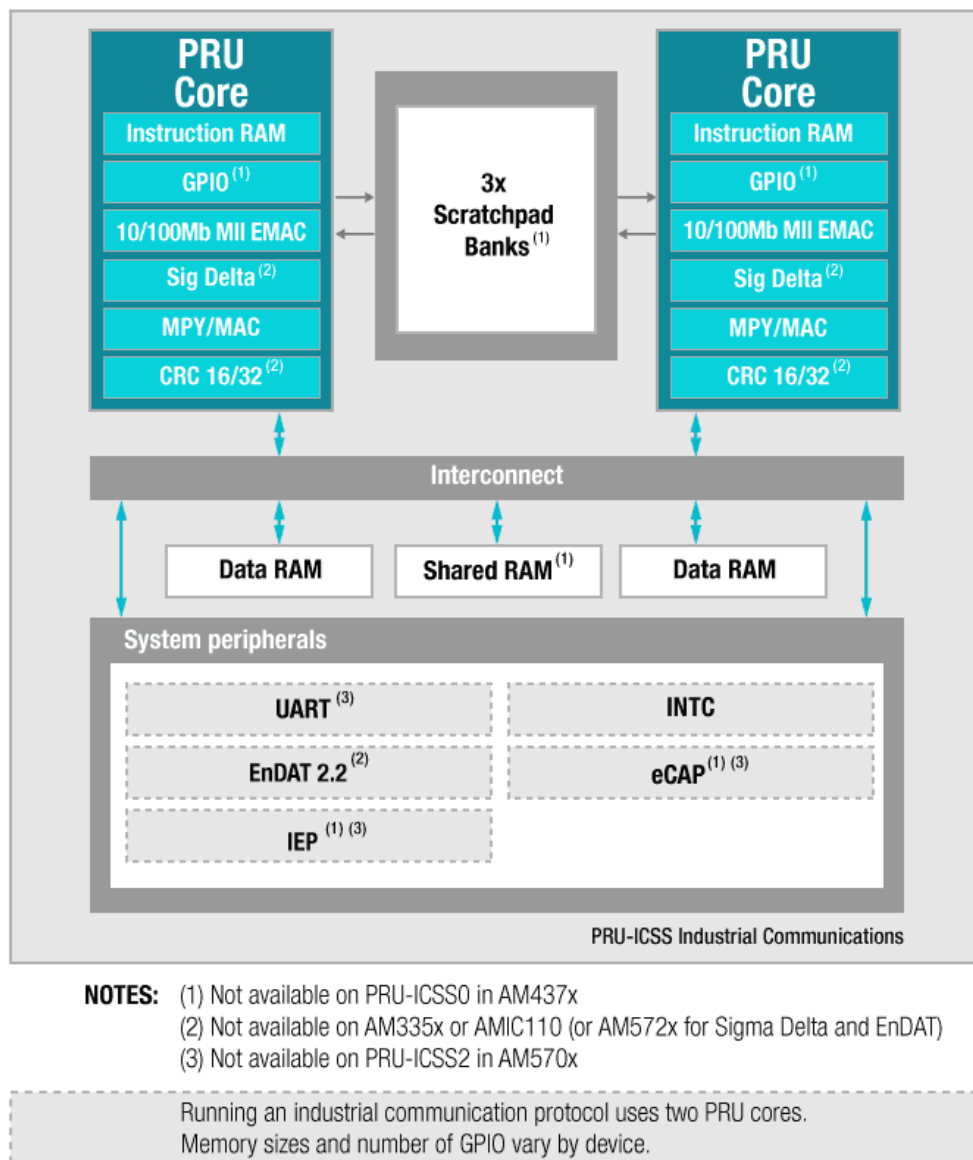


Рисунок 4.4 – Структурна схема підпрограми підсистеми промислової комунікації в режимі реального часу

Таблиця 4.2 – Програмне забезпечення

Пакет програмного забезпечення	Пристрої
<u>PRU-ICSS Industrial Software</u>	AM335x, AM437x, AM57x, 66AK2Gx, AMIC110

Наприклад, для сімейства AM335x кожен під модуль PRU складається з:

- 32-х розрядного процесорного ядра, що працює на частоті 200 МГц,

- 8 Кбайт пам'яті даних,
- 8 Кбайт пам'яті програм,
- Модуль E GPIO (16 виходів і 17 входів загального призначення),
- 32-бітного модуля множення з накопиченням (MAC) з 64-бітовим результатом.

зультатом.

Загальні блоки модуля PRUSS:

- двох портовий модуль Ethernet MII_RT,
- пам'ять даних 12 Кбайт,
- модуль ECAP (Enhanced Capture Model),
- модуль IEP (Industrial Ethernet Peripheral),
- UART0- 16 біт FIFO transmit / receive, генерація переривань, макс швидкість-192Мбіт / с,
- Блок регістрів управління модулем PRUSS,
- Швидкодіюча оперативна пам'ять (Scratch pad) - три банки по тридцять 32-розрядних регістрів, доступ за один цикл,
- Контролер переривань.

Взаємодія між блоками і процесорними ядрами всередині модуля здійснюється за допомогою внутрішньої 32-розрядної шини даних. У свою чергу, система PRUSS має доступ до всіх ресурсів основної системи на кристалі через шину Interface/OCF Master port, так само як і основне ядро має доступ до всіх ресурсів PRUSS. Підмодулі PRU програмуються на асемблері, більшість команд виконуються за один цикл без кеша і конвеєра, дозволяючи точно розраховувати часові проміжки. На частоті 200 МГц виконання одного циклу займає 5 нс.

Модуль PRU-ICSS є вбудованої апаратної платформою для реалізації індустріальних протоколів EtherCAT і PROFIBUS. Реалізація EtherCat можлива завдяки двом фізичним портам Ethernet, а для обробки пакетів «на льоту» використовуються процесорні ресурси модуля. Слід зазначити, що апаратна підтримка EtherCat є не у всіх представників сімейства AM335x.

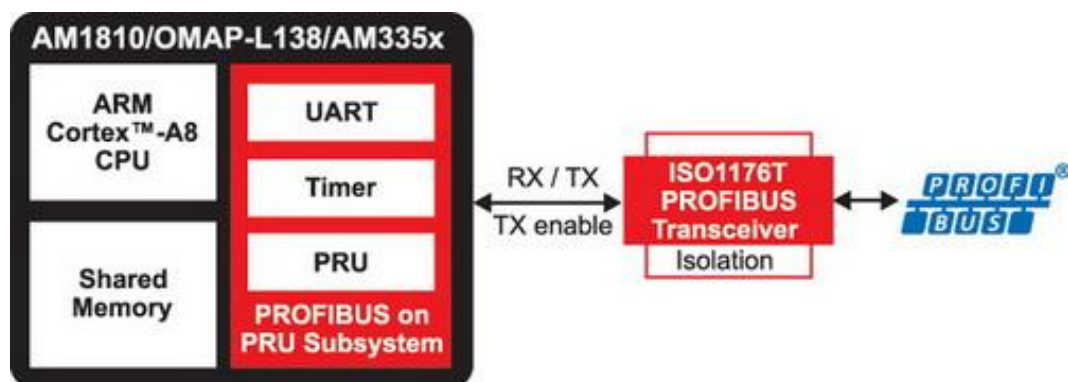


Рисунок 4.5 – Реалізація PROFIBUS на процесорах Sitara

Для реалізації шини PROFIBUS трансивери інтерфейсу 485 підключаються безпосередньо до виводів модуля PRU-ICSS (рис. 4.5). Формування кадру для шини виконується процесорною системою модуля, тому немає необхідності використовувати додаткові мікросхеми ПЛІС. Модуль PRU забезпечує прийом і передачу повідомлень шини, перевірку кадрів, передачу даних основного процесору.

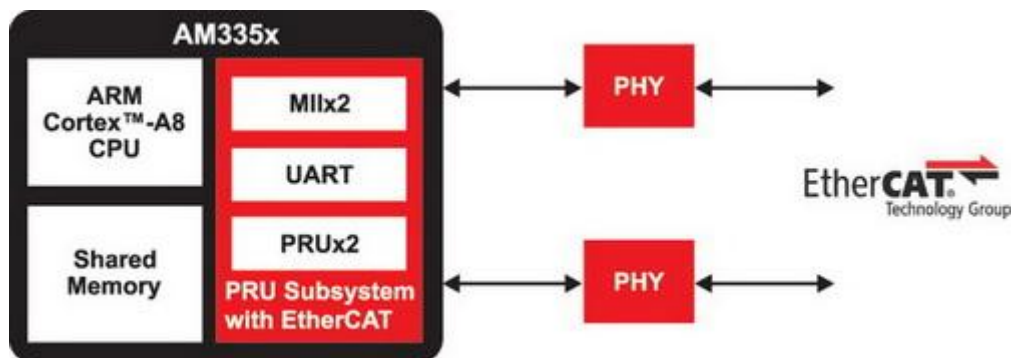


Рисунок 4.6 – Реалізація EtherCat на процесорах AM335x

На сайті ti.com є програмні приклади конфігурації модуля PRU-ICSS (рис. 4.6) для початку роботи і швидкого освоєння.

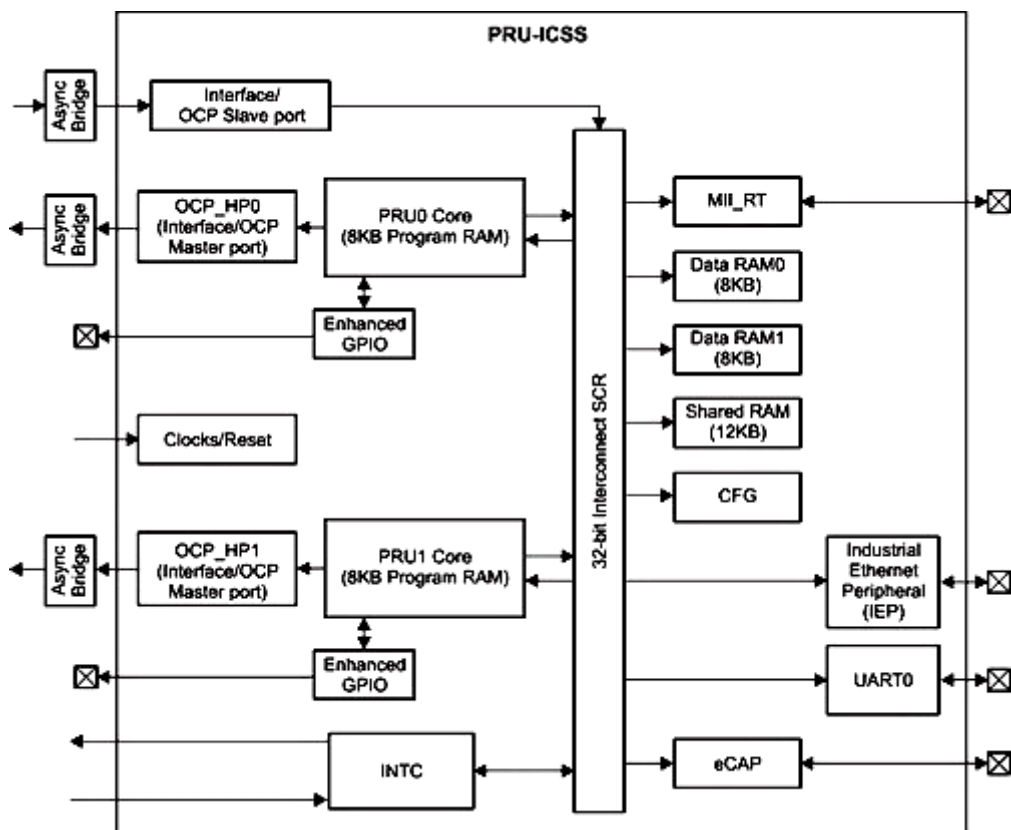


Рисунок 4.7 – Структура модуля PRU-ICSS

Лінійка процесорів AM335X (рис. 4.8) досить гнучка: вона включає в себе різні за складом, продуктивності і ціні кристали. Усередині сімейства є програмна і апаратна сумісність, що дає можливість розробнику без зайвих витрат на-рощувати продуктивність або оптимізувати вартість кінцевого виробу.

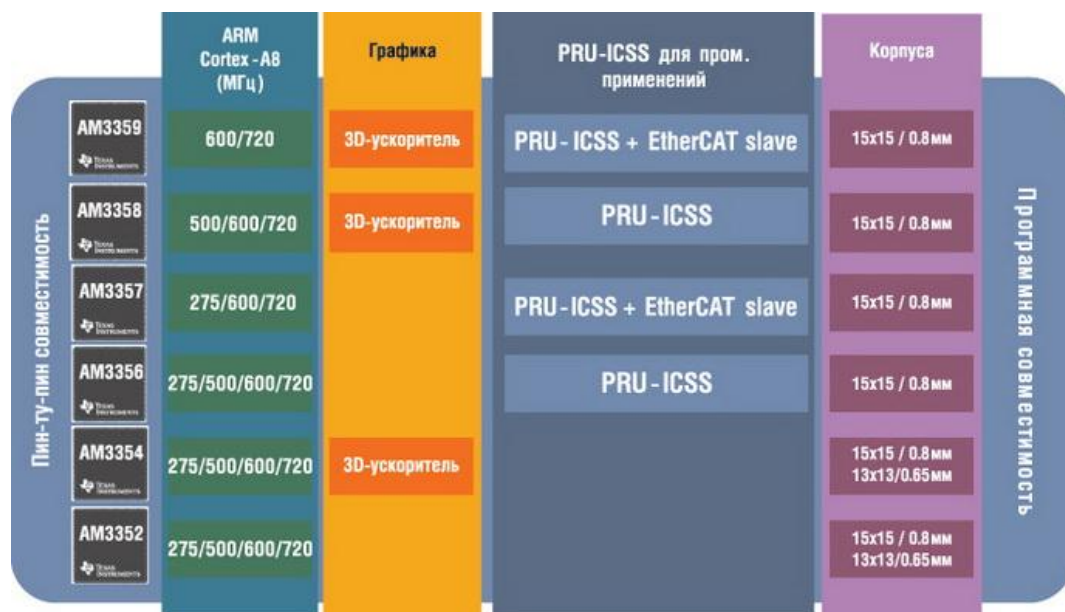


Рисунок 4.8 – Сімейство AM335X

Процесори випускаються в трьох температурних виконаннях:

- Комерційний (0 ... 90 ° C),
- Індустріальний (-40 ... 90 ° C),
- Розширений (-40 ... 105 ° C).

Мікросхеми поставляються в пластикових BGA–корпусах двох видів:

- ZCZ (15,1 × 15,1 × 1,4 мм, 324 виводи, крок 0,8 мм),
- ZCE (13,1 × 13,1 × 1,3 мм, 298 виводів, крок 0,65 мм).

Модуль PRUSS має дві модифікації – з підтримкою EtherCat slave і без. Варто звернути увагу, що кристали з модулем PRUSS випускаються тільки в корпусах ZCZ.

4.1 Модулі на основі мікропроцесорів Sitara.

На основі мікропроцесорів сімейства Sitara з ядром ARM9 розроблено вбудовані модулі AM1705v2 і AM1707LCD70 . Перший з них є продуктивним процесорним модулем, орієнтованим на застосування в складі приладів і промислових систем, другий - модулем на основі мікропроцесора з контролером LCD і 7" дисплея, який дозволяє в розвиненій системі з швидким процесором і різноманітними інтерфейсами реалізувати функцію HMI (Human Machine Interface).

Модуль AM1705v2 виконаний на процесорі AM1705PTP3, який має корпус LQFP176 , звичний для розробників і простий для монтажу. Процесорне

ядро ARM926EJ-S працює на частотах до 456 МГц. На кристалі інтегровані інтерфейси Ethernet MAC 10/100 і USB 2.0 OTG. Для формування широтно-модульованих сигналів є три блоки PWM з поліпшеними характеристиками. У набір інтерфейсів входять два багатоканальних послідовних аудіопорти.

На платі модуля AM1705v2 (рис.4.9), крім мікропроцесора, встановлена зовнішня пам'ять: NAND 128 MB, SDRAM 32 MB, SPI Data Flash 2 MB. Набір інтерфейсів включає порти Ethernet, USB Host, RS-232, слот microSD і міст USB-UART.

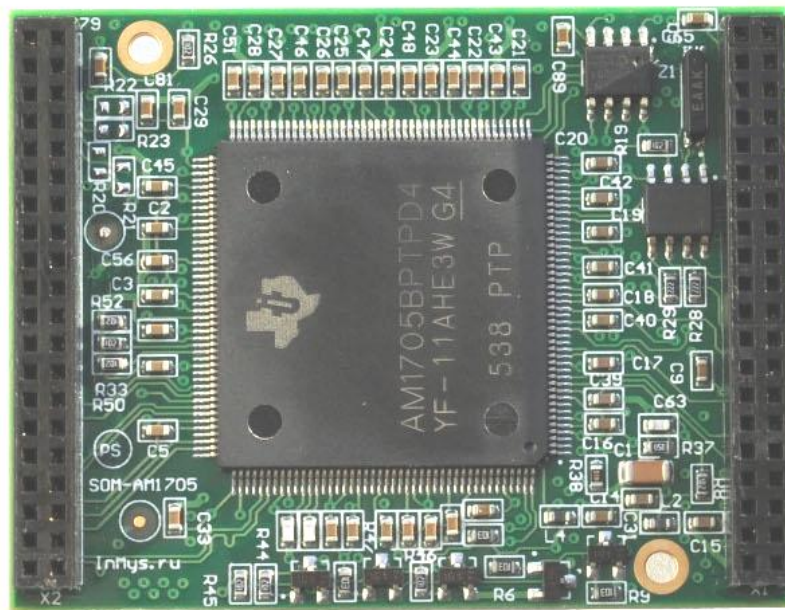


Рисунок 4.9 – Модуль AM1705v2

Для програмування і налагодження є роз'єм JTAG. На два штирьових роз'єми виведені лінії GPIO мікропроцесора і паралельна 8-розрядна шина.

Цей функціональний набір апаратних засобів модуля AM1705v2 відкриває великі можливості для побудови систем управління, обробки і формування сигналів. Модуль може бути також використаний розробниками як апаратна платформа для отримання практичних навичок роботи з ОС Linux. Для цього в комплект поставки включений багатосторінковий документ «Рекомендації по установці ОС Linux на модуль AM1705v2»

Процесорна плата модуля AM1707LCD70 (рис. 4.10) виконана на процесорі AM1707BZKBD4, який має інтегрований контролер LCD та розташований в корпусі BGA256.

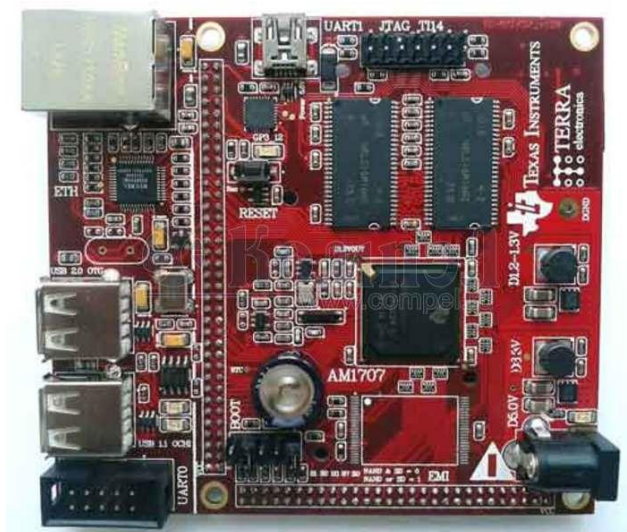


Рисунок 4.10 – Процесорна плата модуля AM1707LCD70

Ядро ARM926EJ-S в цьому процесорі працює на частотах до 456 МГц, на кристал інтегровані порт Ethernet MAC 10/100 і два порти USB (Host, OTG). Є блок PRU і три блоки PWM, три багатоканальних послідовних аудіопорти.

На процесорній платі встановлені два роз'єми, до яких можуть бути підключені дисплеї OSD070T0480, або MTF-TW70SN941. Обидва дисплеї мають діагональ 7", роздільну здатність 800x480 пікселів, сенсорний екран.

Конфігурація зовнішньої пам'яті на платі модуля AM1707LCD70 містить NAND Flash 128 MB, SDRAM 64 MB, SPI Data Flash 2 MB.

Набір інтерфейсів модуля включає порти Ethernet, 2xUSB Host, RS-232, microSD і міст USB-UART.

Для програмування і налагодження є роз'єм JTAG 2 на 7 виводів. На два штирьових роз'єми виведені лінії GPIO мікроконтролера і паралельна 16 – розрядна шина.

На модуль встановлений BSP (Board Support Package) Linux і графічна бібліотека Qt (рис. 4.11).



Рисунок 4.11 – Демонстраційне зображення бібліотеки Qt

Принцип масштабованих архітектур дозволяє встановити на плату цього модуля мікропроцесор OMAP-L137, що має на кристалі додаткове DSP-ядро C674x. Це значно збільшує можливості по обробці сигналів всієї системи без додаткових змін апаратури.

Модулі AM1705v2 і AM1707LCD70 мають перевірену схемотехніку, оптимізовану топологію плат, функції налагодження. Вони можуть бути використані на етапі вивчення сімейства Sitara компанії Texas Instruments, при розробці систем управління, а також у складі серійних виробів в якості OEM-модулів.

Модуль AM1705LCD70 при використанні графічної бібліотеки Qt дозволяє швидко реалізувати продуктивні системи управління з розвиненими функціями HMI.

Програмні пакети BSP (Board Support Package) містять все необхідне для установки і запуску ОС Linux на модулях AM1705v2 і AM1707LCD70. До складу кожного BSP входять U-Boot (завантажувач операційної системи), ядро ОС Linux (Kernel 3.2.1) і набір драйверів пристроїв. Завантажувач і ядро ОС в BSP представлені також у вигляді вихідних текстів і сконфігуровані для двох варіантів джерел завантаження (незалежній пам'яті): NAND або SPI data flash.

У засобі завантаження U-Boot реалізована підтримка інтерфейсу Ethernet і збереження змінних середовища оточення в незалежну пам'ять, що дозволяє гнучко підійти до вибору джерела ядра (завантаження з мережі по протоколу tftp, NAND, SPI, USB і т.д.) і параметрів завантаження ОС (передачі командного рядка ядру) - це скорочує час, необхідний на розробку програмного забезпечення.

Ядро ОС Linux, включене до складу BSP, налаштоване на підтримку модулів AM1705v2 і AM1707LCD70. Набір драйверів підтримує всі функціональні пристрої, реалізовані в модулях: Ethernet, SPI Data flash, NAND, UART, RTC і т.д. Таким чином, після завантаження модуля з'являється можливість використовувати ці пристрої, застосовуючи інтерфейс програмування додатків API (Application Programming Interface) ОС Linux, тобто створювати прикладне програмне забезпечення.

BSP Linux дозволяє в мінімально можливий термін освоїти модулі AM1705v2 і AM1707LCD70 і приступити до розробки прикладного та / або системного програмного забезпечення.

Модуль оцінювання AM65x забезпечує платформу для швидкого запуску процесорів Sitara™ Arm® Cortex®-A53 AM65x (AM6548, AM6546, AM6528, AM6527, AM6526) та прискорення розробки для HMI, мереж, моніторингу систем та інших промислових застосувань. Це платформа розробки на базі чотирьох ядерного процесора Cortex-A53, двоядерного Cortex-R5F процесора, інтегрованого з багатьма інтерфейсами, такими як PCIe, USB 3.0/2.0, Gigabit Ethernet та інше (рис 4.12). В таблицях 4.3 та 4.4 приведені програмне забезпечення та інструменти розробки модулю.

Особливості модулю оцінювання:

- 3 Gigabit Ethernet–порти,
- 4 Гб DDR4 з ECC,
- Бортовий 16 Гб eMMC,
- Бортовий 512 Мб OSPI Flash,
- 1–провідний інтерфейс PCIe Gen3.1, USB 3.1, USB 2.0 та CSI–2.

Склад модулю оцінювання:

- Модуль оцінювання в зборі,
- Флеш–картка,
- USB–кабель,
- Кабель Ethernet,
- Документація.

Таблиця 4.3 – Програмне забезпечення модулю оцінювання AM65x

Назва	Найменування продукту	Тип програмного забезпечення
Процесор SDK для AM65x Sitara процесорів – підтримка Linux і TI–RTOS	PROCESSOR– SDK–AM65X	Набори розробки програмного забезпечення (SDK)

Таблиця 4.4 – Інструменти розробки модулю оцінювання AM65x

Назва	Найменування продукту	Тип інструменту
Інструмент Clock Tree для Sitara, Automotive, OMAP5, Vision Analytics та цифрових сигнальних процесорів	CLOCKTREETOOL	Інструменти розрахунку
Студія Composer Code (CCS) інтегрованого середовища розробки (IDE) для процесорів Sitara ARM	CCSTUDIO– CITAPA	Інструменти розробки SW, IDE, компілятори
Pin Mux Tool	PINMUXTOOL	Інструменти розрахунку



Рисунок 4.12 – Процесорна плата модулю оцінювання AM65x

Висновок. Концепція масштабованих архітектур, що забезпечує програмну, апаратну і топологічну (за висновками корпусів) сумісність між групами мікропроцесорів різної продуктивності, відкриває великі перспективи перед розробниками. Сімейство Sitara™ є базою масштабованих архітектур 32-розрядних мікропроцесорів компанії Texas Instruments.

Використання модулів на базі сімейства Sitara™ має максимально прискорити для розробників темп вивчення нових електронних компонентів та їх застосування в автоматизованих системах. Ці модулі можуть застосовуватися на всіх етапах життєвого циклу виробів:

- при освоєнні мікропроцесорів сімейства Sitara™ як початківцями, так і досвідченими розробниками;
- при розробці, макетування та налагодженні нових виробів;
- в якості готових вузлів серійних систем.

4.2 Програмні засоби розробки для МП Sitara

Для розробників програмного забезпечення компанія Texas Instruments пропонує цілий ряд SDK для різних застосувань. SDK (Software Development Kit) – пакет інструментів для розробника, що містить порт операційної системи, набір додатків і документацію. SDK дозволяє в найкоротший термін ознайомитися з ОС і можливостями апаратної платформи, а також є базою для розробки власних додатків. Компанія TI надає SDK для сімейства Sitara під операційні системи Linux, Android 2.3, SYS/BIOS, Windows Embedded Compact 7. Крім того, доступний для скачування пакет StarterWare для розробників, які не використовують операційні системи.

Особливості пакету StarterWare:

- Програмні приклади для роботи з периферією;
- Синхронізація програмних блоків для роботи з периферією на різних процесорах;
- Інструментарій для C-коду;
- Запуск на Windows і Linux;
- Підтримка компіляторів GCC для ARM, IAR для ARM для процесорів сімейства TMS470;
- Вбудована в оболонку графічна бібліотека і USB-стек з програмними прикладами;
- Online-утиліта для генерації образу файлової системи. Крім генерації файлової системи за допомогою засобів, що входять в SDK, існує можливість створити образ за допомогою утиліти на сайті проекту Angstrom.

4.2.1 Середовище розробки Code Composer Studio

Характеристики та особливості інтегрованого середовища розробки програмного забезпечення для ЦСП фірми Texas Instruments – Code Composer Studio. Інтегроване середовище розробки програмного забезпечення CCS дозволяє виконати повний цикл розробки програми для ЦСП. Це означає, що даний пакет включає в себе:

- вбудований редактор, що дозволяє створювати і редагувати вихідні файли (мова C або асемблер);
- компілятор, необхідний для створення об'єктного файлу;
- компоновщик або линкер, що дозволяє об'єднати в один об'єктний файл кілька вихідних і бібліотечних файлів;
- різноманітні засоби налагодження програмного забезпечення (набір спеціальних засобів перегляду стану внутрішніх регістрів, пам'яті, засоби візуалізації даних і ін.).

Перед початком роботи необхідно встановити на робочому комп'ютері програмне забезпечення Code Composer Studio c2x ezDSP Kit Tools і підключити оцінний модуль. На жаль, дана версія програмного забезпечення не працює в режимі симуляції, такий режим доступний тільки в повній версії пакету Code Composer Studio.

Запуск програми здійснюється з меню за допомогою іконки на робочому столі комп'ютера (рис. 4.13).



Рисунок 4.13 – Середовище розробки Code Composer Studio

Після запуску програми, відкривається її основне вікно, зовнішній вигляд якого представлений на рис. 4.14. Воно поділяється на дві основні області - робочу область і область навігатора проекту. В області навігатора проекту відображається вся інформація про поточний проект – вихідні файли, командний файл компоновщика, спільні бібліотеки тощо.

Основною програмною одиницею в середовищі Code Composer Studio є проект (файл з розширенням * .pjt). Проект включає в собі вихідні файли (тексти програм на мові C (* .c) або асемблер (* .asm), бібліотеки (* .lib), командний файл компоновщика (* .cmd)) і файл налаштувань (компілятора і компоновщика).

В області навігатора проекту відображається вся інформація про поточний проект – вихідні файли, командний файл компоновщика, спільні бібліотеки тощо.

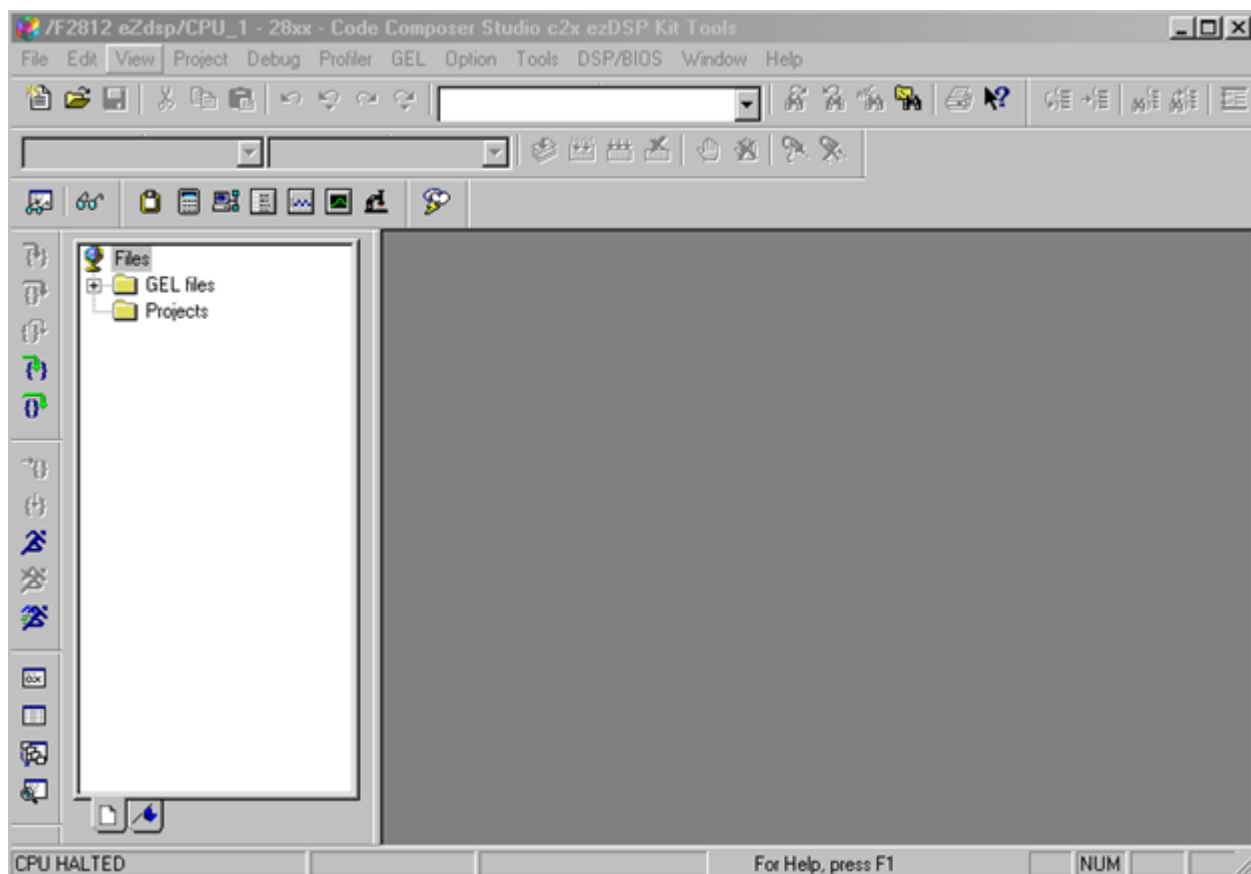


Рисунок 4.14 – Зовнішній вигляд основного вікна Code Composer Studio

Основний програмної одиницею в середовищі Code Composer Studio є проект (файл з розширенням * .pjт). Проект включає в себе вихідні файли (тексти програм на мові С (* .с) або асемблер (* .asm), бібліотеки (* .lib), командний файл компоновщика (* .cmd)) і файл настройок (установки компілятора і компоновщика) .

Після запуску Code Composer Studio відкриється головне вікно, як показано на рисунку 4.14. Тепер необхідно створити проект.

Створення проекту. Для створення проекту в меню Project вибираємо команду New Відкриється діалогове вікно завдання параметрів проекту Project Creation (рис. 4.15). В поле Project Name необхідно вказати ім'я проекту.

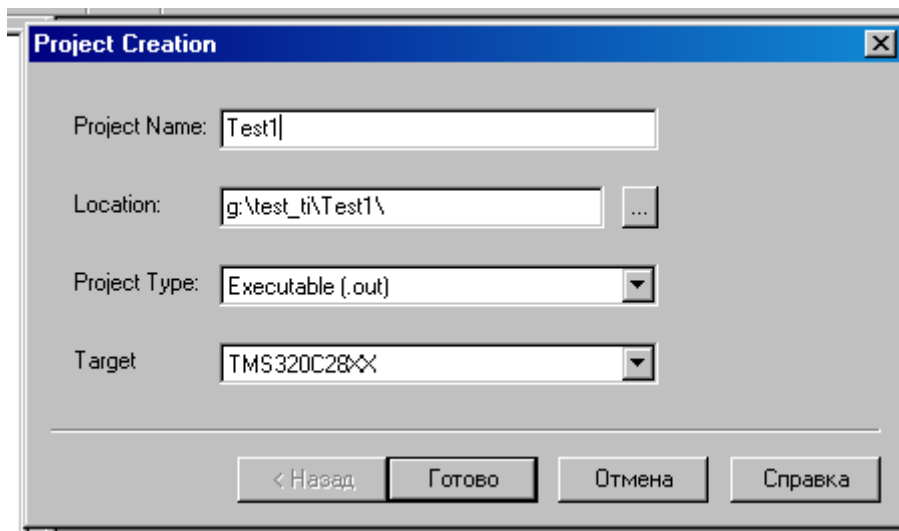


Рисунок 4.15 – Вікно завдання параметрів проекту

В полі Location задається робоча директорія проекту. В полі Project Type вибирається тип проекту - виконується програма (Executable (.out)). Це значення встановлюється в полі за замовчуванням. В полі Target необхідно вказати сімейство ЦСП для якого створюється проект. Потім натисніть кнопку Готово .

В результаті буде створений новий проект Test1 (зверніть увагу на зміни у вікні навігатора проекту). Створений в робочій директорії файл проекту Test1.pjt є звичайним текстовим файлом, в якому зберігаються всі налаштування і установки проекту.

Створення вихідного файлу. Далі необхідно створити вихідний файл, тобто написати першу програму для ЦСП. Для виклику вбудованого текстового редактора в меню File вибрати команду New -> Source File (рис. 4.16). Перша програма буде дуже простий – вона містить нескінченний цикл, змінну *i*, що змінюється від 1 до 99, і проміжну змінну *k*, в якій зберігається результат звенення в квадрат змінної *i*. Але вона дозволить познайомитися з основними командами середовища Code Composer Studio. У вікні текстового редактора наберіть наступну програму:

```
unsigned int k;
void main (void)
{
    unsigned int i;
    while (1)
    {For (i = 0; i <100; i ++)
        k = i * i;
    }
}
```

Збережіть цей файл в робочій директорії проекту під ім'ям «Test1.c» за допомогою команди File -> Save As ... Файл зберігся на жорсткому диску, але він не додався до проекту автоматично. Щоб додати файл до проекту, виконайте команду Project -> Add Files to Project ..., і у вікні, виберіть файл «Test1.c». У вікні навігатора проекту, в папці Source з'явиться вихідний файл.

Перевіримо коректність вихідного файлу. Для цього, за допомогою команди Project -> Compile File, виконайте компіляцію вихідного файлу. Дана команда дозволяє скомпілювати окремий файл, який в даний момент є активним. Якщо у вихідній програмі не було помилок, то у вікні повідомлень з'явиться інформація про успішне закінчення компіляції (рис. 4.17). У цьому ж вікні відображається інформація про поточні налаштування компілятора. Якщо у вихідній програмі були виявлені помилки, то інформація про кількість помилок і їх місцезнаходження теж буде виводитися у вікні повідомлень. Після усунення помилок, необхідно виконати повторну компіляцію вихідного файлу.

Для коректної роботи компілятора мови C, до проекту необхідно підключити бібліотеку функцій мови C, що поставляється в складі середовища Code Composer Studio, і оптимізовану для конкретного сімейства ЦСП.

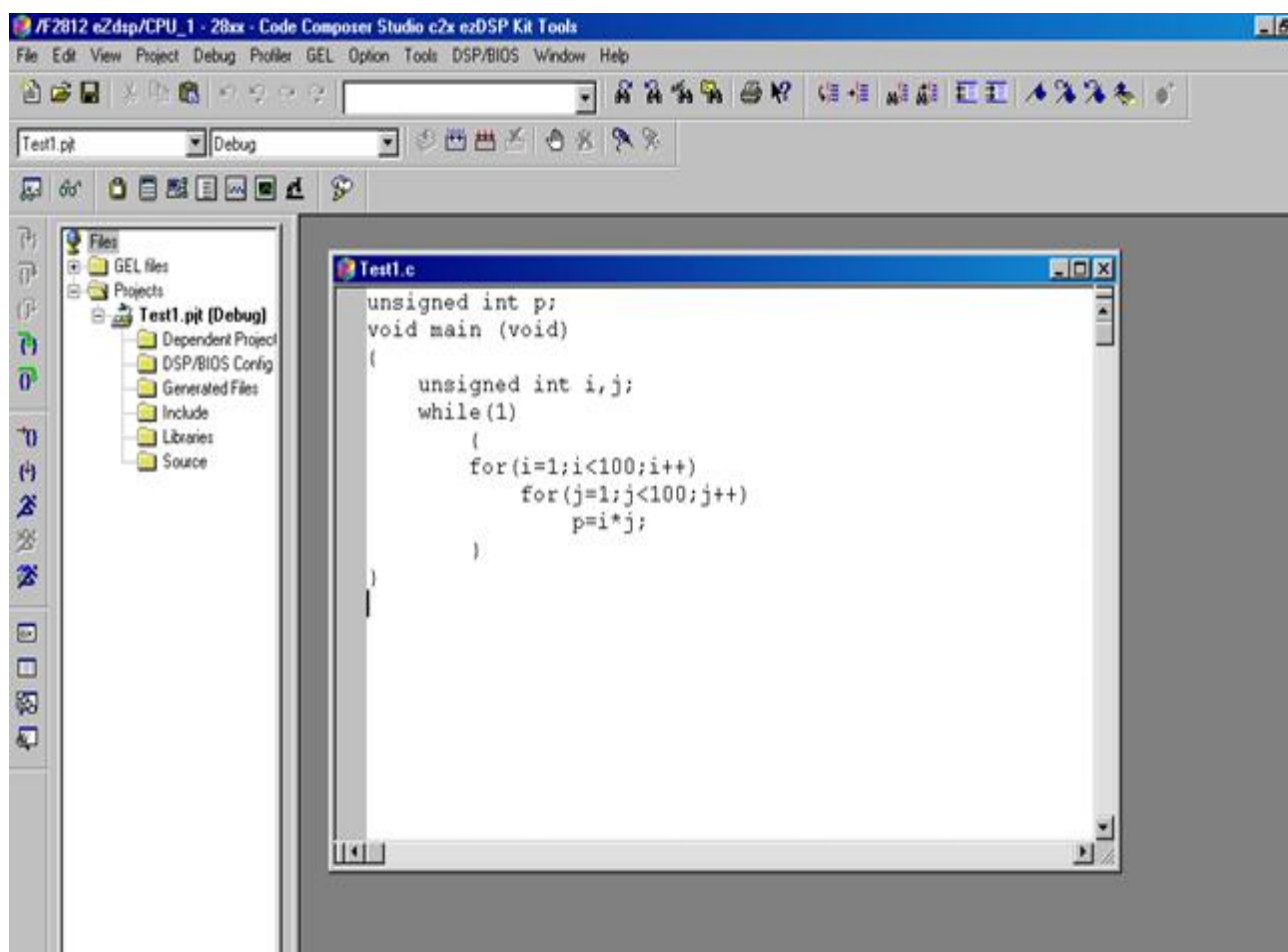


Рисунок 4.16 – Вікно текстового редактора

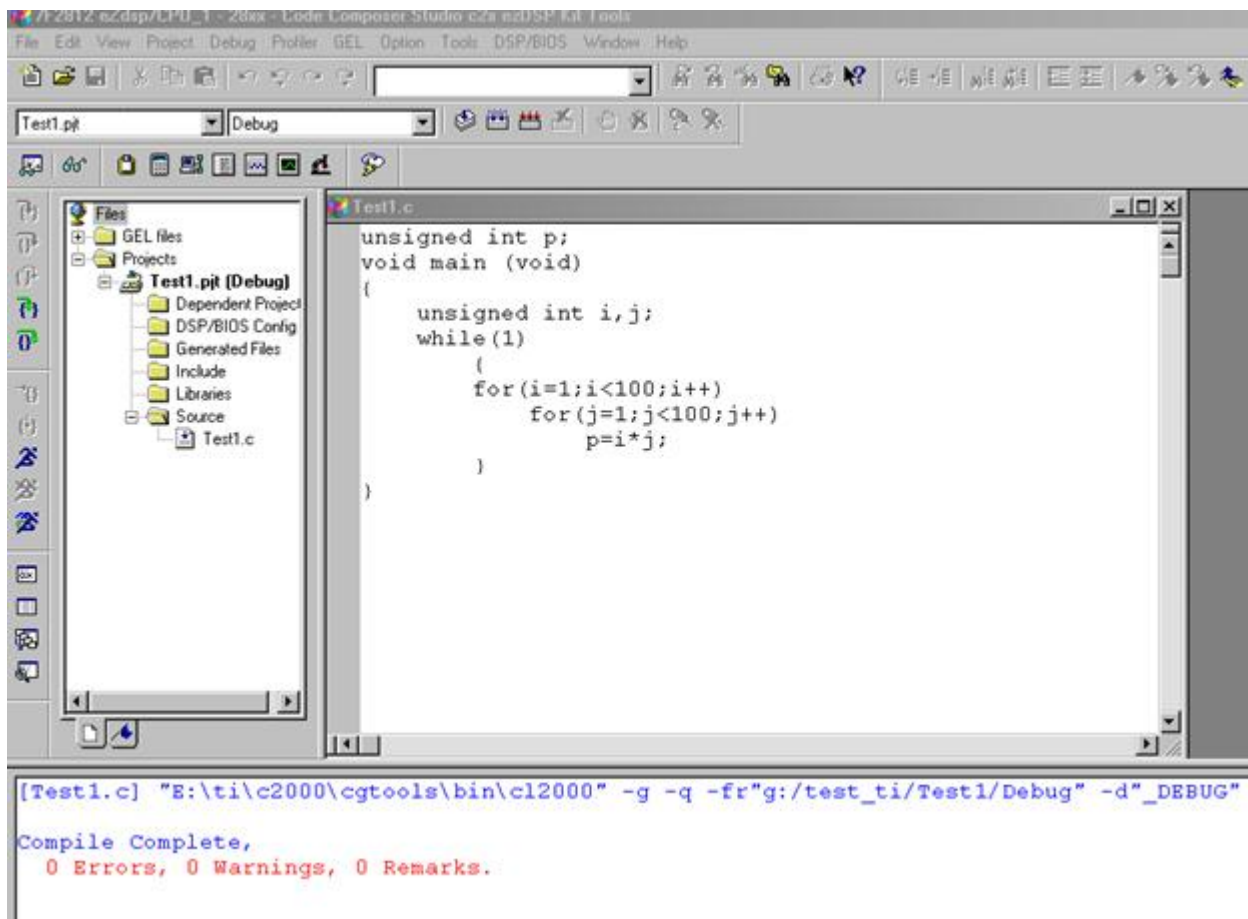


Рисунок 4.17 – Результат компіляції вихідного файлу

Для цього необхідно викликати вікно налаштувань компілятора і компоновщика (за допомогою команди Project -> Build Options ...). У вікні, виберіть закладку Linker. У рядку Library Search Path (-i) - вкажіть директорію розміщення бібліотеки, в рядку Include Libraries (-l) - вкажіть ім'я підключається бібліотеки. У рядку Stack Size (-stack) - вкажіть 0x400 (величина стека). Зовнішній вигляд сторінки налаштувань компоновщика показаний на рис. 4.18. Натисніть кнопку ОК.

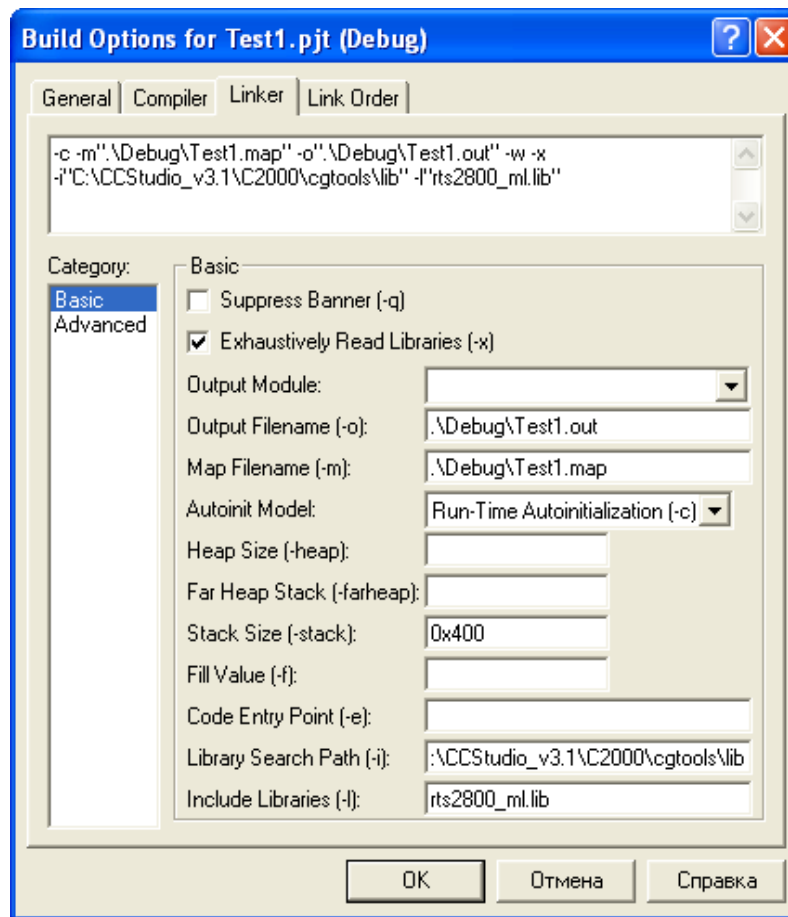


Рисунок 4.18 – Сторінка налаштувань компоновщика

Далі необхідно додати в проект командний файл компоновщика (Linker Command File, * .cmd). Цей файл визначає відповідність між фізичною пам'яттю ЦСП і логічними секціями, створюваними всередині програми. Нижче наведено приклад командного файлу компоновщика:

```
MEMORY
{
    PAGE 0: /* Program Space */
        FLASH: org = 0x3F8000, len = 0x20000

    PAGE 1: /* Data Space */
        M0SARAM: org = 0x000000, len = 0x400
        M1SARAM: org = 0x000400, len = 0x400
}

SECTIONS
{
    .text:> FLASH PAGE 0
    .ebss:> M0SARAM PAGE 1
    .cinit:> FLASH PAGE 0
    .stack:> M1SARAM PAGE 1
}
```

В даному файлі, в розділі MEMORY, описується фізична пам'ять ЦСП. В області пам'яті програм (PAGE 0) користувачеві доступний банк FLASH - з початковим адресою 0x3f8000 і довжиною 0x20000. В області пам'яті даних (PAGE 1) користувачу доступні два банки: M0SARAM - який розташований за адресою 0x000000 розміром 0x400, і M1SARAM - який розташований за адресою 0x000400 розміром 0x400. Наступний розділ - SECTIONS, вказує компонуваньнику, де у фізичній пам'яті ЦСП повинні розташовуватися різні логічні програмні секції. Наприклад, програмна секція .text буде розміщена в області пам'яті програм, в банку пам'яті, який починається з 0x3f8000 адреси.

Далі скористаємося командним файлом, який поставляється разом з середовищем Code Composer Studio. За допомогою команди Project -> Add Files to Project ... додайте до проекту файл EzDSP_RAM_lnk.cmd, який розташовується в папці Solutions (спочатку, скопіюйте файл в робочу директорію проекту). Командний файл з'явиться в самому кінці списку у вікні навігатора проекту.

Тепер всі необхідні файли зібрані, і проект можна компілювати. Запустіть проект на компіляцію з допомогою команди Project -> Build. Якщо проект скопілювати без помилок, можна перейти до наступного етапу - налагодженні проекту.

Налагодження проекту. Для того, щоб приступити до налагодження програми, спочатку її необхідно завантажити в ЦСП (тільки це не означає, що ми будемо програмувати внутрішню пам'ять процесора!). Для цього необхідно спочатку підключити процесор до середовища за допомогою команди Debug -> Connect. У меню File виберіть команду Load Program ... У діалоговому вікні виберіть файл Test1.out, який знаходиться в робочій директорії проекту, в папці Debug.

Після завантаження програми в ЦСП, автоматично відкривається вікно дизасемблера (рис. 4.19). Показчик в цьому вікні (зелена стрілка) – зазначає команду асемблера, розташовану за першою виконуваною адресою.

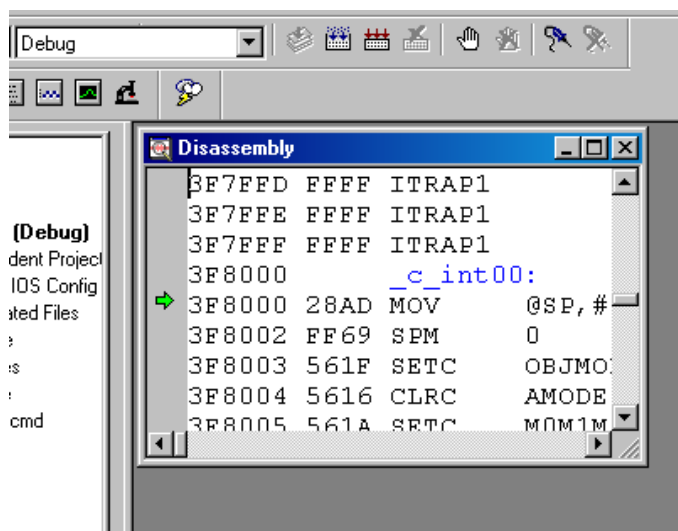


Рисунок 4.19 – Вікно дизасемблера

Для переходу до програмної мітки «main» виконайте команду Debug → Go Main. Відкриється вікно вихідного файлу Test1.c, покажчик в цьому файлі (жовта стрілка) відзначає команду, адреса якої зараз знаходиться в лічильнику команд (Program Counter, PC) (рис. 4.20). Ця команда буде виконуватися на наступному кроці.

Щоб запустити програму, виконайте команду Debug → Run (або натисніть функціональну клавішу F5). Ніяких помітних змін в середовищі Code Composer Studio не відбулося! Про активність процесора говорить пропажа покажчика у вікні вихідного файлу і напис RUNNING в лівому нижньому кутку (рис. 4.21). Для зупинки процесора виконайте команду Debug → Halt (або натисніть комбінацію клавіш Shift + F5). Для повернення процесора в початковий стан виконайте команди Debug → Reset CPU і Debug → Restart. Знову поверніться до програмної мітки «main».

Для перегляду значень змінних, використовуваних в програмі, скористаємося вбудованим вікном візуалізації Watch Window (викликається командою View → Watch Window). Це вікно має закладки Watch Locals і Watch1. Локальна змінна і автоматично відображається у вікні Watch Locals. Для перегляду глобальної змінної її необхідно додати вручну.

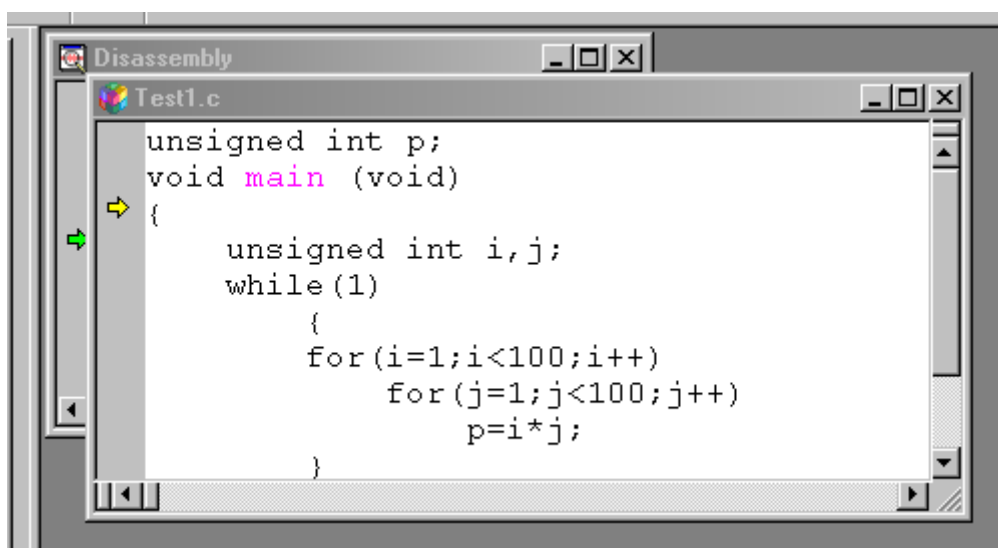


Рисунок 4.20 – Перехід до мітки "main"

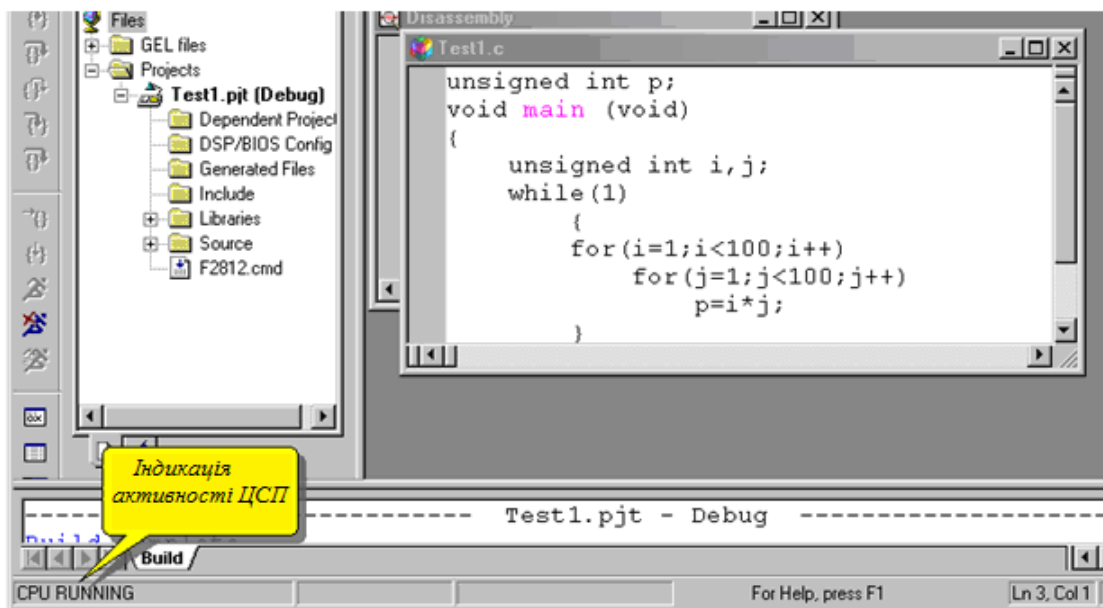


Рисунок 4.21 – Виконання програми процесором

Для цього, у вікні з вихідним файлом виділіть змінну k . Натисніть праву кнопку миші. У контекстному меню, виберіть команду Add to Watch Window. Мінлива k додається у вікно Watch1 (рис. 4.22). Таким же способом додайте в вікно перегляду змінну i . У стовпці Value відбивається поточне значення змінної, яке можна змінити безпосередньо в цьому ж вікні.

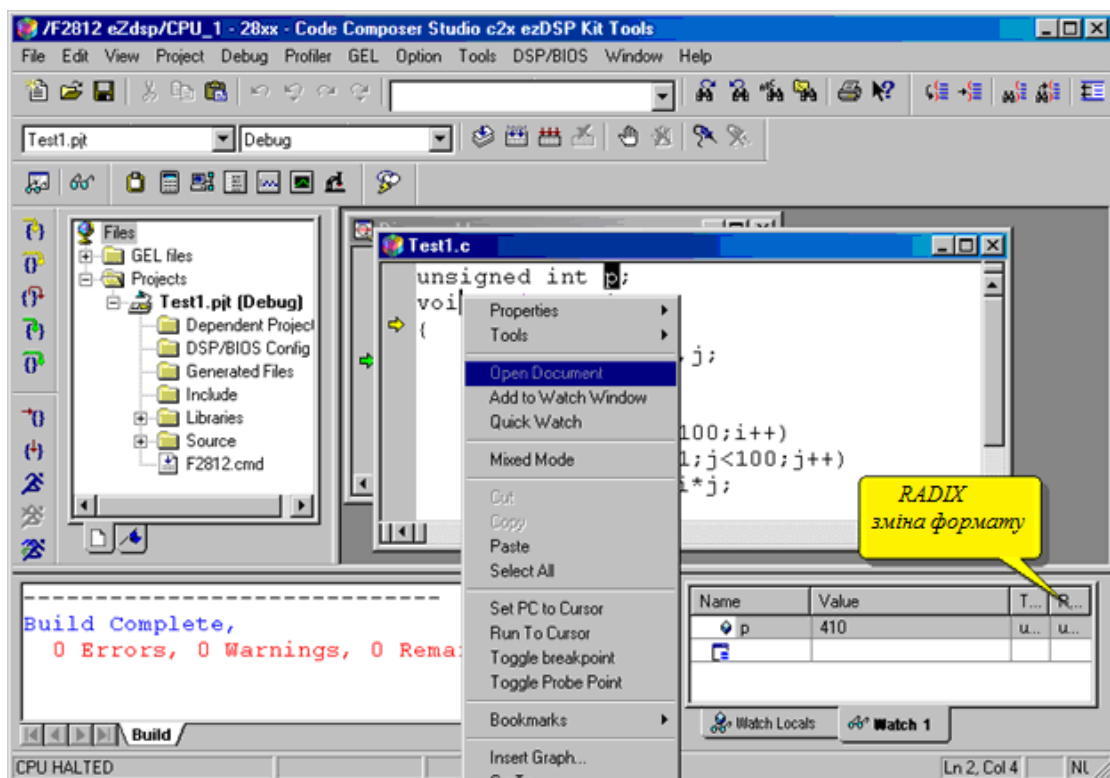


Рисунок 4.22 – Додавання змінної у вікно перегляду

Наступний поширений спосіб налагодження програми - робота в покроковому режимі. Для цього виконайте команду Debug -> Step Into (або скористайтеся функціональною клавішею F11). Виконайте кілька кроків. Зверніть увагу на зміни положення покажчика (жовтої стрілки) у вікні вихідного файлу і на зміни значень змінних k і i у вікні Watch Window. Зліва від вікна навігатора проекту розташовується додаткова панель управління (рис. 4.23) з командами, що використовуються в покроковому режимі налагодження. Спробуйте самостійно розібратися з цими командами.

Ще один спосіб, який використовується при налагодженні програмного забезпечення, застосування точок зупини (Breakpoints). В цьому випадку, виконання програми припиняється при досягненні команди, зазначеної точкою зупинки. Для установки точки зупини розмістіть курсор у рядку вихідного файлу з командою $k = i * i$.Натисканням правої кнопки мишки викличте контекстне меню і виберіть команду Toggle Software Breakpoints. Поруч з покажчиком (жовта стрілка) з'явиться індикатор точки зупинки (червона точка).Повторний вибір команди Toggle Software Breakpoints видаляє точку зупини у вибраному рядку (рис. 4.24).

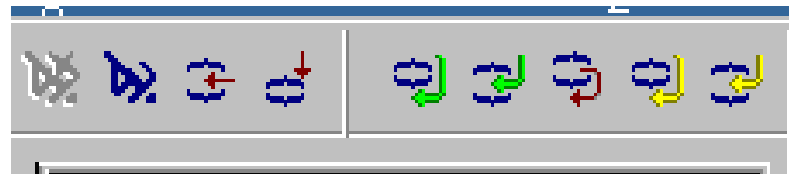


Рисунок 4.23 – Додаткова панель управління

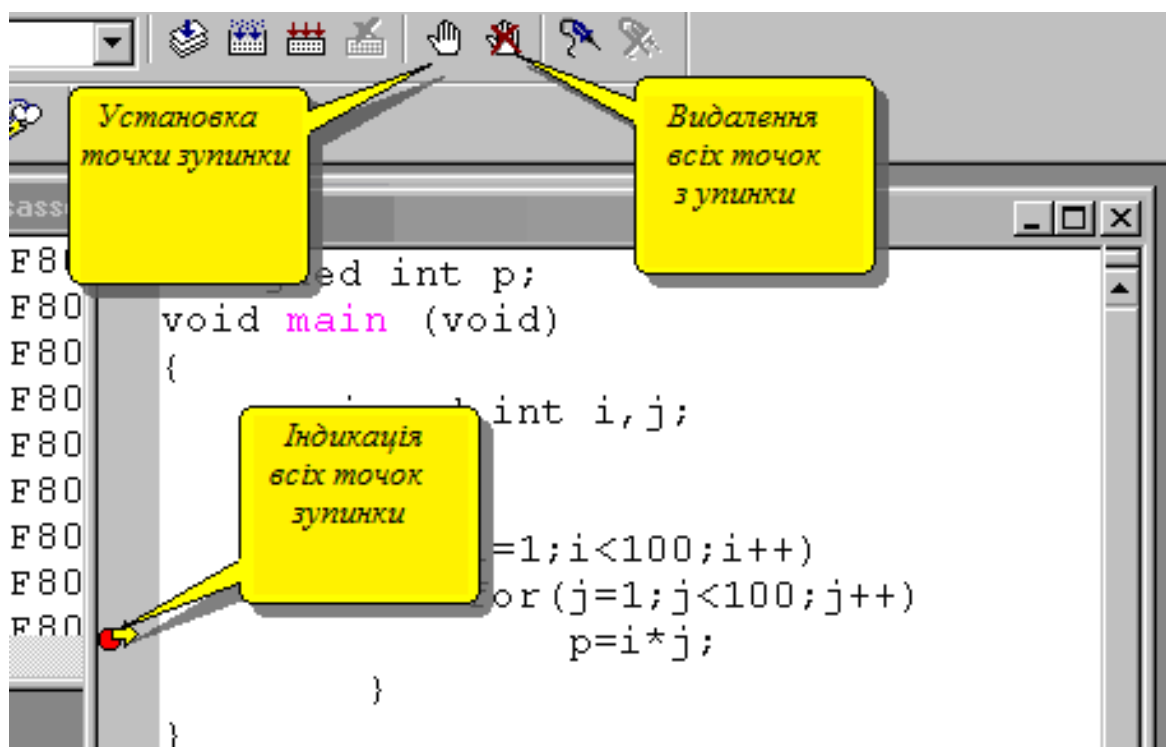


Рисунок 4.24 – Точка зупинки

Після розміщення точки зупинки виконайте повторний запуск програми (команди Debug → Reset CPU , Debug → Restart , а потім Debug → Run (або F5)). Процесор зупиниться в точці зупинки. Зверніть увагу на зміну змінних. Повторіть команду Debug → Run кілька разів.

Крім точок зупинки, в середовищі Code Composer Studio існує можливість роботи з тестовими точками (Probe Points). Якщо в точці зупинки процесор припиняє виконання програми, то в тестовій точці виконання програми призупиняється, процесор обмінюється інформацією з налагоджування середовищем, а потім виконання програми триває. Зараз ми змінимо умови роботи нашої програми. Вікно перегляду змінних буде оновлюватися в момент досягнення програмою тестової точки. Для цього спочатку видаліть точку зупинки. Потім розмістіть курсор в рядку вихідного файлу з оператором $k = i * i$. Натисканням правої кнопки мишки викличте контекстне меню і виберіть команду Toggle Software Probe Point. Поруч з покажчиком (жовта стрілка) з'явиться індикатор тестової точки (блакитний ромб). Тепер необхідно вказати дії, що виконуються при досягненні тестової точки. Для цього відкрийте вікно налаштувань тестових точок (команда Debug → Probe Points ...). У вікні встановлених тестових точок виділіть точку Test1.c line 9 → No Connection за допомогою мишки. Автоматично ця точка пропишеться в рядку Location: даного вікна. У випадваючому меню рядка Connect to: виберіть Watch Window. Натисніть кнопку Replace, а потім ОК (рис. 4.25). Тестова точка підключена! Запустіть програму і зверніть увагу на оновлення інформації у вікні перегляду змінних. Зупиніть виконання програми. Видаліть всі тестові точки за допомогою команди Remove all Probe Points ().

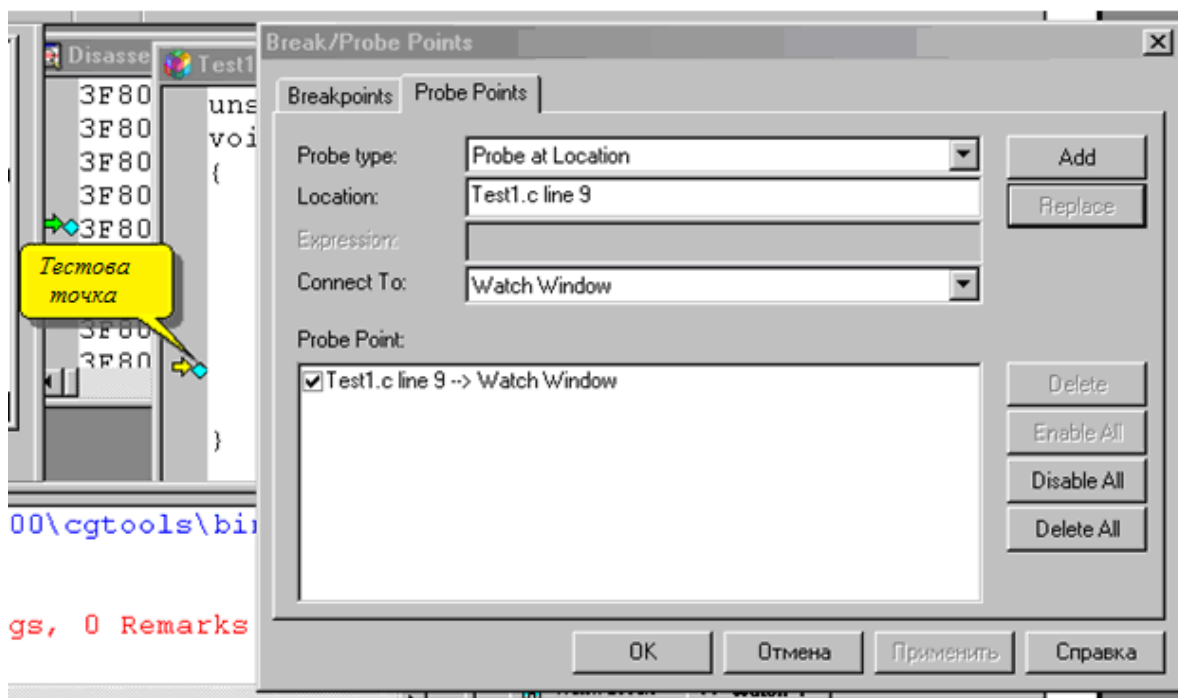


Рисунок 4.25 – Підключення тестової точки

Щоб перевірити ефективність роботи компілятора мови C, можна у вікні з вихідним файлом одночасно переглядати програму на мові C і її реалізацію на мові асемблер. Для цього виберіть команду View -> Mixed Source / ASM. Створені компілятором в асемблері команди будуть відображатися у вікні з вихідним файлом сірим кольором (рис. 4.26). Додатково, у вікні з'явився ще один покажчик (зелена стрілка). Цей покажчик відображає стан лічильника команд для програми на мові асемблер. Перевірте зміну положення цих покажчиків при роботі ЦСП в покроковому режимі.

Для збереження налаштувань середовища Code Composer Studio - встановлені режими відображення, відкриті допоміжні вікна та ін., Виконайте команду File -> Workspace -> Save Workspace As ... і збережете їх під ім'ям Test1.wks. Наступного разу, під час запуску Code Composer Studio можна буде викликати налаштування робочої області с допомогою команди File -> Workspace -> Load Workspace.

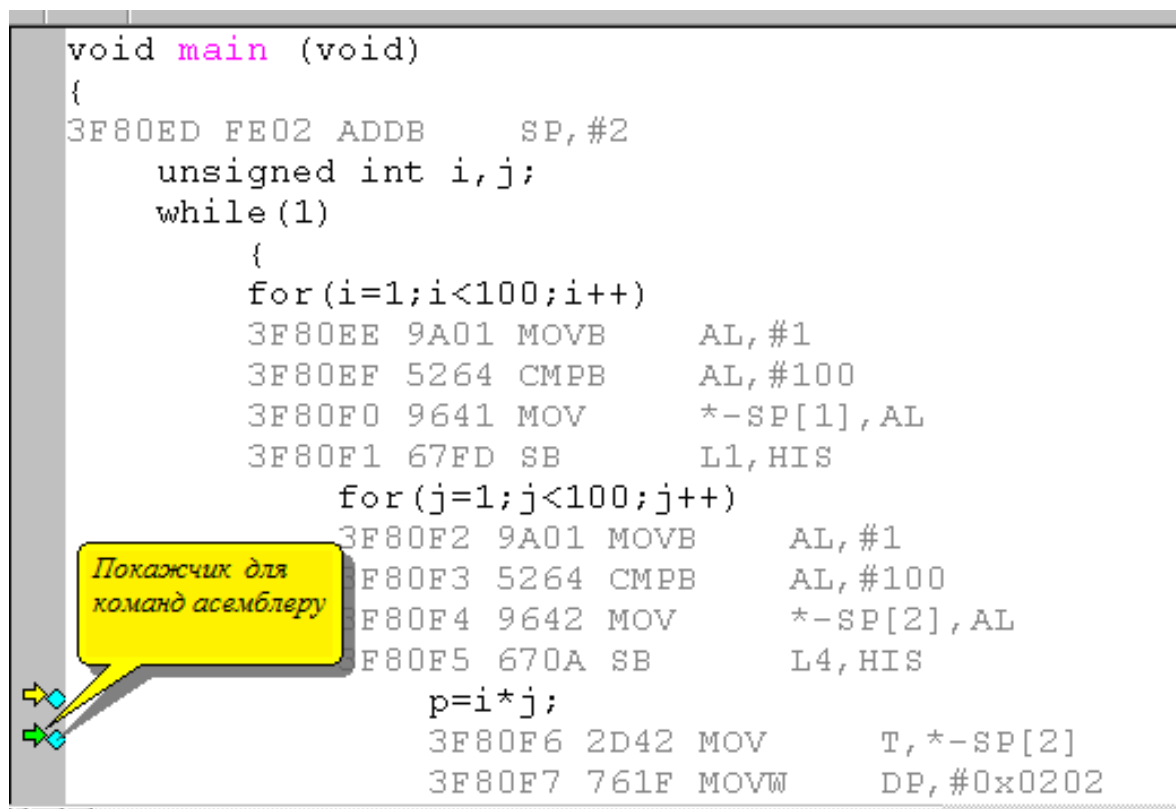


Рисунок 4.26 – Змішаний режим відображення вихідного коду

Для закріплення матеріалу рекомендується самостійно створити проект і перевірити його роботу. Як завдання можна використовувати будь-яку програму – наприклад, сума двох чисел або обчислення квадрата числа і т.п.

4.3 Контрольні запитання і завдання

1. Структура мікропроцесорів AM1705/1707?
2. Що означає масштабованість в рамках сімейства мікропроцесорів?
3. Які особливості сімейства мікропроцесорів Sitara TM.?
4. Які програмні засоби процесорів Sitara.?
5. В чому полягає підтримка протоколів зв'язку?
6. Які функції забезпечує PRU-ICSS (Підпрограма підсистеми промислової комунікації в режимі реального часу)?.
7. Приведіть структуру підпрограми підсистеми промислової комунікації в режимі реального часу.
8. Які функції PRU-ICSS та структура підпрограми підсистеми промислової комунікації?
9. Дайте характеристику сімейству AM335X.
10. Які вбудовані модулі розроблено на основі мікропроцесорів сімейства Sitara з ядром ARM9.
11. Які існують пакети інструментів для розробника?
12. Який склад середовища розробки програмного забезпечення Code Composer Studio?
13. Приведіть структуру модуля PRU-ICSS?
14. В який спосіб виконується налагодження програмного забезпечення?
15. Яка структурна реалізація PROFIBUS на процесорах Sitara?
16. Дайте характеристику лінійці процесорів Sitara Arm.
20. Дайте характеристику мікропроцесорів, що застосовують концепцію Індустрії 4.0.

5. ЗАСТОСУВАННЯ МІКРОКОНТРОЛЕРІВ В СИСТЕМАХ КОНТРОЛЮ ТА КЕРУВАННЯ

Останнім часом завдяки вдалому поєднанні характеристики ціна – якість широке застосування в пристроях систем керування, вбудованих системах та інших додатках отримали мікроконтролери STM, та їх найпоширеніший представник – STM32.

Базові характеристики сімейства мікроконтролерів

Засновані на ARM CortexM4 мікроконтролери серії STM32F4 є продовженням провідної лінійки STM32, втім маючи ще більш високу продуктивність. Так само, як серія STM32F2, ці мікроконтролери виготовляються по 90нм-технології і використовують запатентований STMicroelectronics ART Accelerator для досягнення найкращих результатів тестів серед заснованих на ядрі CortexM мікроконтролерів, досягаючи показників в 225 DMIPS/606 CoreMark і працюючи з флеш – пам'яті на частоті 180 МГц. Інструкції DSP і модуль операцій з плаваючою крапкою дають можливість застосовувати дані контролери в широкому спектрі проектів. Динамічне споживання енергії дозволяє знизити споживання струму при виконанні коду з флеш-пам'яті до 140 мкА/МГц для STM32F401 (максимальна частота до 84 МГц) і до 238 мкА/МГц для STM32F42x/43x, що працюють на частоті до 180 МГц. Мікроконтролери серії STM32F4 є результатом ретельно створеного симбіозу можливості управління МК в реальному часі і продуктивністю обробки сигналів, властивій сигнальним процесорам, доповнюючи таким чином лінійку контролерів STM32 новим класом пристроїв – сигнальними мікроконтроллерами (англ. *Digital signal controller, DSC*). Серія складається з п'яти класів продуктів, які повністю сумісні за виводами, периферією та програмним кодом.

Характеристики серії:

- STM32F401 мають STM32 84 MHz CPU 105 DMIPS. Є продовженням серії STM32 F4, мають низьке енергоспоживання, розміщені в корпуси малих розмірів, на відміну від інших класів серії STM32F4.
- STM32F405/415 – мають 168 MHz CPU/210 DMIPS, до 1 МБайт флеш пам'яті з розширеними можливостями сполучення і шифрування.
- STM32F407/417 мають ядро STM32 168 MHz CPU/210 DMIPS до 1 МБайт флеш пам'яті з наявністю Ethernet MAC і інтерфейсом камери для STM32F405/415.
- STM32F427/437 – мають STM32 ядро 168 MHz CPU/210 DMIPS, до 2 МБайт флеш, доповнюючи STM32F407/F417 розширеними можливостями сполучення і шифрування.
- STM32F429/439 – мають ядро STM32 180 MHz CPU/225 DMIPS, до 2 МБайт двохбанкової флеш-пам'яті з інтерфейсом SDRAM, контролером TFT LCD, технологією акселерації Chrom-ART, послідовним аудіо інтерфейсом,

мають більшу продуктивність і менше енергоспоживання в порівнянні з STM32F4x7 F4x5.

5.1 Загальні характеристики сімейства

Характеристики мікроконтролерів сімейства STM32F4:

- Ядро ARM 32-bit CortexM4 CPU;
- Частота тактового генератора 168МГц, 210 DMIPS / 1.25 DMIPS / МГц (Dhrystone 2.1);
- Підтримка DSP-інструкцій;
- Нова високопродуктивна АНВ-матриця шин;
- До 1 Мбайт Flash-пам'яті;
- До 192 + 4 Кбайт SRAM-пам'яті;
- Напруга живлення 1,8 ... 3,6 (POR, PDR, PVD і BOR);
- Внутрішні RC-генератори на 16МГц і 32кГц (для RTC);
- Зовнішнє джерело тактування 4 ... 26МГц і для RTC – 32,768кГц;
- Модулі налагодження SWD / JTAG, модуль ETM;
- Три 12-біт АЦП на 24 вхідних канали (швидкість до 7,2 мільйонів вибірок, температурний датчик);
- Два 12-бітних ЦАП;
- DMA-контролер на 16 потоків з підтримкою пакетної передачі;
- 17 таймерів (16 і 32 розрядних);
- Два сторожових таймера (WDG і IWDG);
- Комунікаційні інтерфейси: I²C, USART (ISO 7816, LIN, IrDA), SPI, I²S;
- CAN (2,0 В Active);
- USB 2.0 FS / HS OTG;
- 10/100 Ethernet MAC (IEEE 1588v2, MII / RMI);
- Контролер SDIO (карти SD, SDIO, MMC, CE-ATA);
- Інтерфейс цифрової камери (8/10/12/14-бітові режими);
- FSMC-контролер (Compact Flash, SRAM, PSRAM, NOR, NAND і LCD 8080/6800);
- Апаратний генератор випадкових чисел;
- Апаратне обчислення CRC, 96-бітний унікальний ID;
- Модуль шифрування AES 128, 192, 256, Triple DES, HASH (MD5, SHA-1), HMAC;
- Розширений температурний діапазон -40 ... 105 ° С.

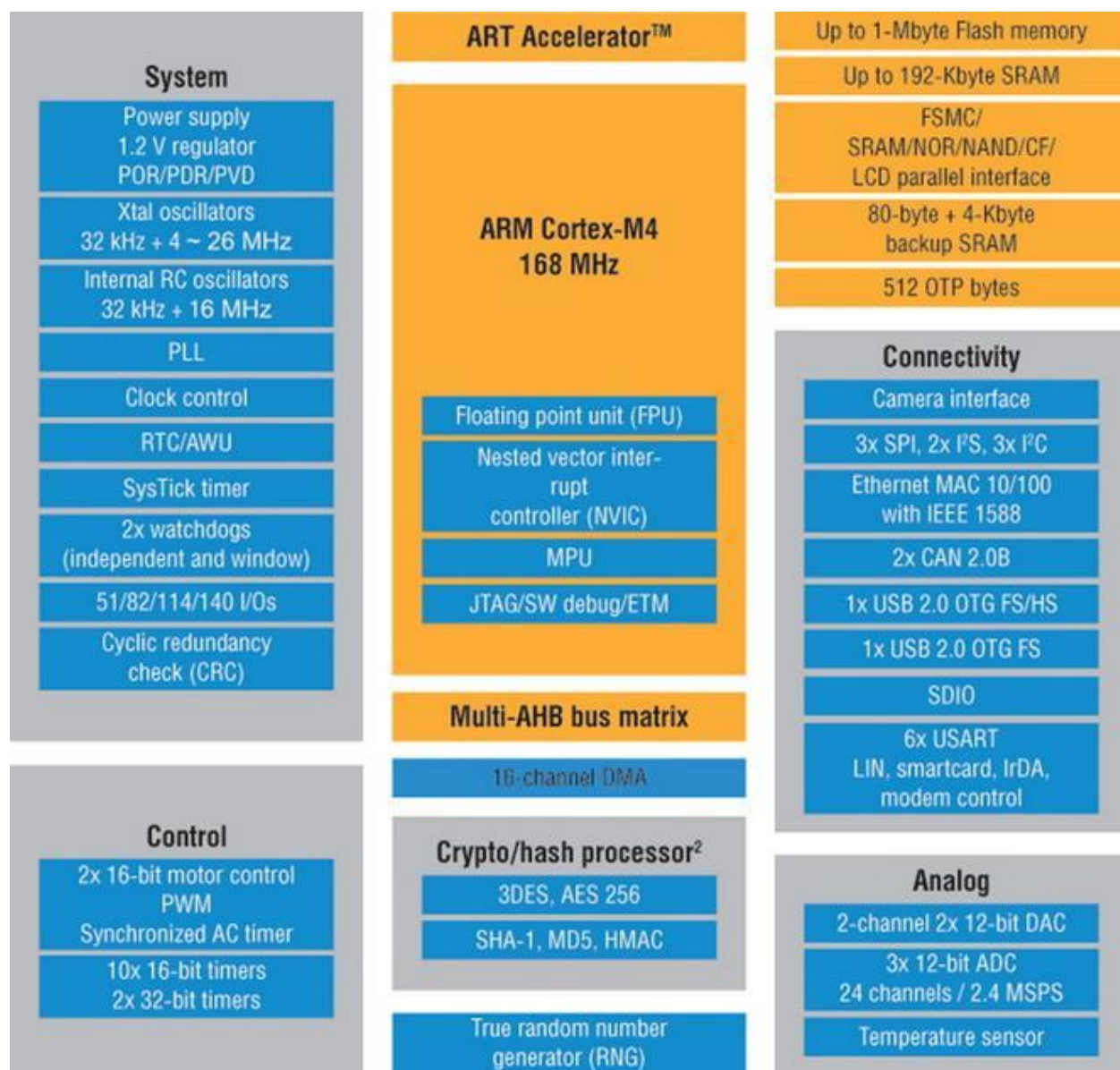


Рисунок 5.1 – Структура мікроконтролерів STM32F4xx

5.2 Вбудовані інтерфейси комунікації

Поговоримо більш детально про комунікаційні інтерфейси сімейства STM.

Ethernet. Даний блок присутній не в усіх продуктах сімейства, а лише в контролерах STM32F407 / STM32F417. Блок виконаний за стандартом IEEE802.3. Можлива передача даних зі швидкістю 10/100 Мб/с. Доступна синхронізація годинника для чого протокол IEEE1588v2 реалізований апаратно. Для оптоволоконної або провідної лінії потрібне застосування стороннього приймача. PHY-трансивер з'єднується безпосередньо з портом MII або RMII.

USB (Universal Serial Bus). На борту присутні два роздільних блоки USB. Перший – USB OTG full-speed, є повністю апаратною реалізацією і сумісний зі стандартами USB 2.0, а також OTG 1.0. Працює на швидкості до 12 Мб/с. Підт-

римується робота в режимі Host/Device/OTG. Присутній SRP (Session request protocol) і HNP (Host negotiation protocol).

Другий – USB OTG high-speed працює в режимі Host Device/OTG з високою швидкістю 480 Мб/с, для чого необхідний блок приймача, що працює на високій швидкості через спеціальний ULPI-інтерфейс.

SDIO (Secure Digital Input/Output). Інтерфейс дозволяє працювати з картами SD SDIO/MMC-картами пам'яті і дисковими контролерами CE-ATA. У восьми бітному режимі несуча частота обміну даними складає 48 MHz. Контролер відповідає таким стандартам: SD Memory Card 2.0, MultiMediaCard System 4.2 (робота в режимах 1/4/8 біт), SDIOCard 2.0 (режими 1го та 4х біт), CE-ATA 1.1.

SPI (Serial Peripheral Interface). Пристрій оснащений трьома блоками SPI, кожен з яких працює в режимі Master (Multimaster) або в режимі Slave, передаючи дані в напівдуплексному, повнодуплексному або сімплексному режимі. Підтримується апаратний підрахунок контрольних сум CRC для підвищення надійності каналу зв'язку: так CRC може бути переданий останнім байтом слова в режимі Tx, присутній авто перевірка правильності CRC останнього прийшов байта. Блок пристрої SPI1 працює на швидкостях аж до 37,5 Мб/с. Інші обмежені максимальною швидкістю в 21 Мб/с.

Передача даних йде у вигляді 8- або 16-бітових блоків, вперед молодшим або старшим бітом. При чому допускається програмна заміна фази і полярності тактового сигналу.

USART (Universal Synchronous Asynchronous Receiver Transmitter) . У мікроконтролер вбудовано чотири блоки USART і два UART (Universal Asynchronous Receiver Transmitter). Блоки USART1 і USART6 допускають високошвидкісний обмін даними на швидкості до 10,5 Мбіт / с. Інші ж підтримують швидкість не більше 5,25 Мб/с. На апаратному рівні є підтримка передачі даних згідно стандарту NRZ (Non Return to Zero).

Обмін даними здійснюється з використанням 8- або 9-бітних блоків, один або два біта яких виділені як стоп-біти і біти перевірки парності. USART можна конфігурувати на режим SPI, блок USART при цьому виступає в ролі ведучого пристрою SPI. Використовуючи блок USART можна організувати підключення до інтерфейсу LIN, яка знайшла застосування в автомобільній промисловості, або налаштувати на енкодинг / декодинг ІК-сигналу IrDA. Можлива робота з модемами по лініях управління RTS і CTS. Підтримується робота зі смарт-картками.

I²C (Inter-Integrated Circuit) . На борту МК міститься три блоки I²C, що підтримують роботу в режимі Master / Slave (головний або підпорядкований), а також в режимі Multimaster (режим в якому на шині присутні кілька Master-пристроїв, що мають спільні ресурси Slave, або по черзі змінюють свій стан з Master на Slave і назад). У складі пристрою є модуль діагностики і виправлення пакетних помилок PEC. Використовується 7-бітний і 10-бітний режим адресації. Підтримуються загальноприйняті для протоколу швидкості обміну даними до 100 kHz в простому режимі і 400 kHz в режимі надшвидкого обміну даними.

Модулі можуть бути сконфігуровані на розширені протоколи SMBus 2.0 і PMBus.

I²S (Inter-Integrated Sound). У мікроконтролері присутні два мультиплексованих блоки *I²S* з вбудованим SPI. Обидва модулі можуть бути сконфігуровані на роботу в режимі Master або Slave. Дані передаються по 16, 24 або 32 біта повнодуплексно або сімплексно.

Серед підтримуваних протоколів такі: Phillips *I²S*, PCM, MSB і LSB з вирівнюванням даних. Інтерфейс *I²S* був розроблений для обміну звуковими даними в цифровому форматі. Відтепер для тактування присутній окремий блок фазового автопідстроювання частоти, який уможливорює генерацію частот зразків аудіо від 8 до 192 kHz з похибкою не більше 0,01%.

CAN (Controller Area Network). На борту знаходиться два CAN-модуля, що працюють за стандартами 2.0A і 2.0B, швидкість роботи при цьому досягає 1 Мбіт / с. Модулі можуть працювати зі стандартними, а також з розширеними кадрами. Модуль CAN містить три буфера передачі, трьохкаскадний FIFO-стек і 28 банків фільтрів (розподілені і масштабуються).

DCMI (Digital Camera Interface). Присутній в контролерах STM32F407 і STM32F417. За допомогою даного інтерфейсу можна організувати пряме підключення до камери або CMOS-матриці. Можлива внутрішня і зовнішня синхронізація покадрово, робота в безперервному режимі, автообрізка зайвих частин зображення. Серед підтримуваних форматів 8/10/12/14-бітове прогресивне відео, YCbCr 4: 2: 2 і RGB 565, JPEG.

FSMC (Flexible Static Memory Controller). Блок використовується для підключення рідкокристалічних дисплеїв або зовнішньої пам'яті безпосередньо. Блок міститься лише в 100, 144- або 176-піновий корпусах.

Можливе сполучення з підключеною синхронної або асинхронної пам'яттю або PCMCIA-пристроями. В основному блок призначений для видачі даних МК у відповідному підключеним пристроєм вигляді без зайвих витрат процесорного часу на перекодування даних.

Таким чином кожне зовнішній пристрій має власну адресу з пулу, власні сигнали для управління. Подавши необхідний сигнал вибору мікросхеми можна отримати доступ до того чи іншого зовнішнього пристрою (одночасне використання не допустимо). Можливе підключення таких типів пам'яті як NAND Flash, Compact Flash, NOR Flash, SRAM і PSRAM. Інтерфейс налаштований для роботи з LCD – контролерами Motorola 6800 і Intel 8080, однак може бути легко використаний для сполучення з контролерами інших виробників.

5.3 Структура периферії МК

На рис.5.2 зображена периферія МК STM32F407VGT6, що використовується в рекомендованій виробником налагоджувальній платі для швидкого старту STM4-DISCOVERY.

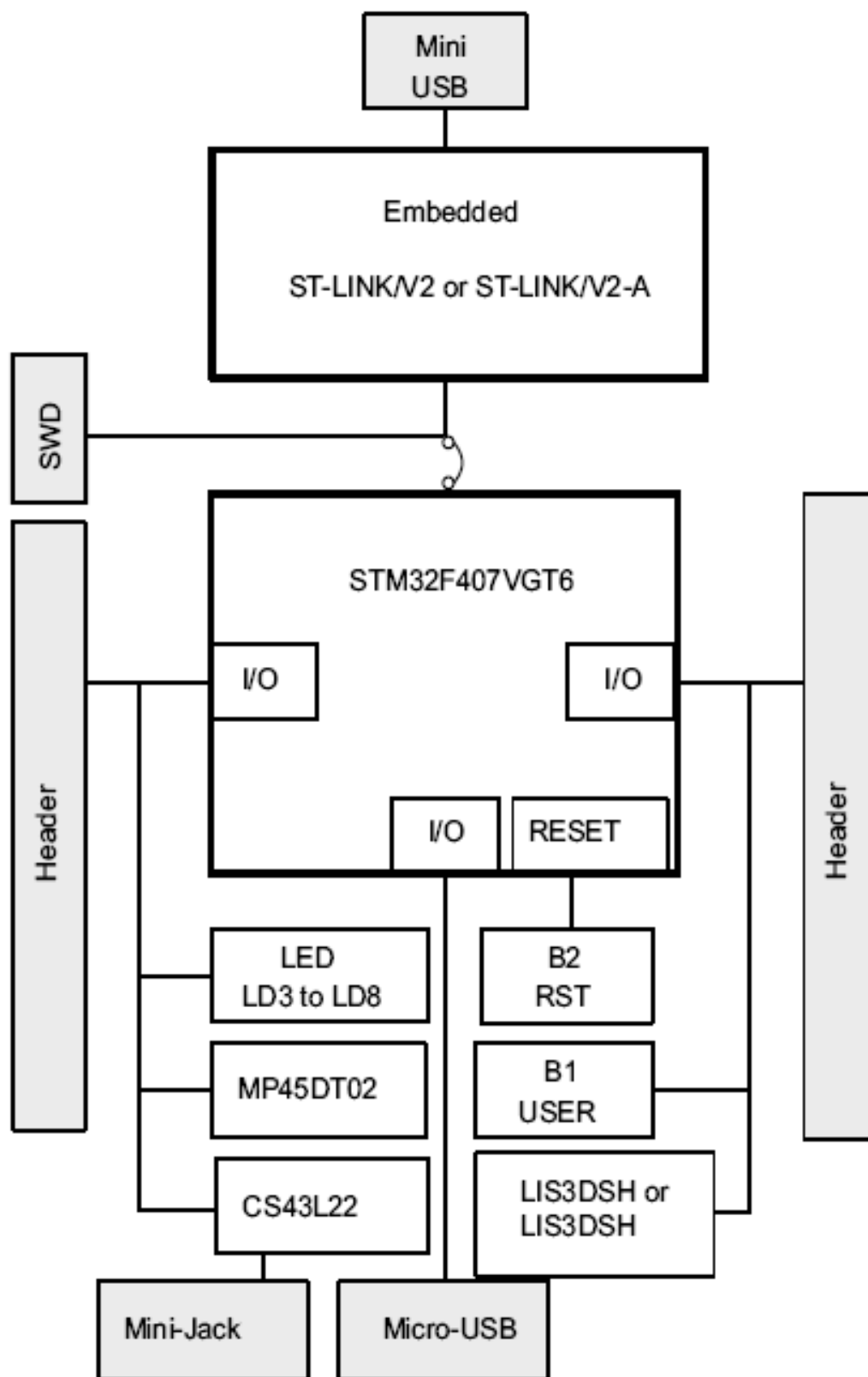


Рисунок 5.3 – Структура периферії налагоджувальної плати для швидкого старту *STMF4-DISCOVERY*

Плата розробника STM32F4DISCOVERY STM32F407 побудована на контролері STM32F407VGT6 з ядром CortexM4F, який можна розігнати до 168 МГц. Об'єм розширюваної оперативної пам'яті складає 192 Кбі. Флеш пам'ять програм має місткість 1 Мб. Є підтримка USB, причому контролер може бути як пристроєм так і хостом. Є апаратний генератор випадкових чисел. Крім звичних інтерфейсів DAC, ADC, SPI, I²C, PWM, RTC є інтерфейси для підключення камери – DCMІ і для підключення SD карт – SDIO. Є підтримка Ethernet.

У плату STM32 F4 DISCOVERY вбудований цифровий мікрофон MP45DT02 і стереофонічний 24-бітний ЦАП CS43L22 для відтворення звуку з вбудованим підсилювачем потужності класу D, регулятором гучності, тембру і ін., що дозволяє побудувати на цій платі звукову карту.

Також ця плата оснащена трьохосевим цифровим акселерометром LIS302DL. Крім стандартних функцій, акселерометр додатково може перемикає діапазони вимірювання прискорення, апаратно обробляти кліки і подвійні кліки (постукування, наприклад), генерувати переривання. Має два інтерфейси - SPI або I²C а також виходи для переривань. Для управління і контролю на платі є 4 різних світлодіоди, розташованих навколо акселерометра, і дві кнопки (призначені для користувача і скидання контролера).

Живиться контролер через стабілізатор напруги LD3985M33R. Для його початкової роботи достатньо струму в 150мА, проте для складних пристроїв з більшою кількістю периферії потрібно більш потужний зовнішній стабілізатор з напругою 3.3В.

Для налагодження програми на платі розміщений програматор-відладчик ST-Link. На відміну від програматорів інших плат DISCOVERY, світлодіод даної плати – двоколірний, що дозволяє розрізняти режими роботи налагодження та програмування.

Засоби комунікації:

- Через mini USB роз'єм підключається відладчик ST-LINK;
- Через micro USB-роз'єм під'єднується контролер.

Живиться відлагоджувальна плата STM32F4DISCOVERY від роз'єму mini USB.

Середовища розробки:

- RealView Development Suite (ARM C / C ++ Compiler);
- Keil MDK-ARM (ARM C / C ++ Compiler);
- IAR Embedded Workbench for ARM (IAR C / C ++ Compiler);
- MULTI IDE for ARM (Green Hills C / C ++ Compiler);
- TASKING VX-toolset for ARM (Altium C / C ++ Compiler);
- Sourcery CodeBench (GCC Compiler);
- Rowley CrossWorks for ARM (GCC Compiler);
- Atollic TrueSTUDIO (GCC Compiler);
- RAISONANCE Ride7 IDE for ARM (GCC Compiler);
- Coocox CoIDE (GCC Compiler);
- mikroC for ARM (MikroElektronika C Compiler).

На рис. 5.4 зображена *системна архітектура* сімейств STM32F405xx, STM32F407xx.

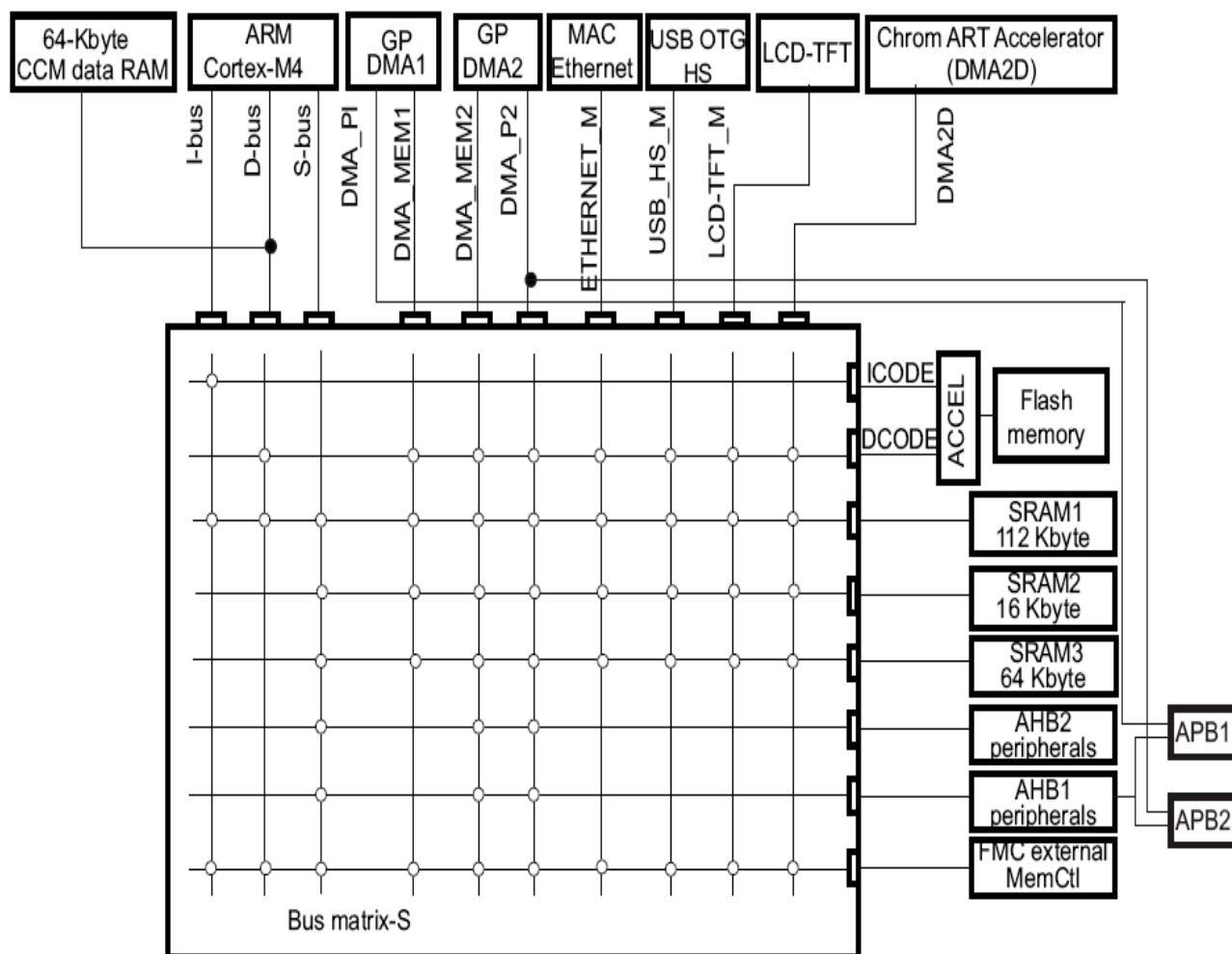


Рисунок 5.4. – Системна архітектура сімейств STM32F405xx, STM32F407xx

Робота з аналоговими сигналами. На кристалі міститься три АЦП і два одноканальних ЦАП. АЦП має роздільну здатність 12 біт і дуже високу швидкість перетворення, складову 2,4 MIPS в одиночному режимі і 7,2 MIPS – в потрійному режимі. Максимально доступне число аналогових каналів – 24. Як і в більшості сучасних МК, присутній генератор опорної напруги. Гнучка система налаштувань вбудованого аналогового мультиплексора дозволяє задавати будь-які послідовності перетворення аналогових каналів (за винятком одночасного перетворення одного каналу на декількох АЦП). Налаштування АЦП дозволяють одноразові і циклічні вимірювання. Для проведення перетворення на максимальних швидкостях необхідно дотримуватися діапазону напруги живлення 2,4 ... 3,6 В. При зниженні напруги до 1,8 (1,7) В швидкість перетворення знижується до 1,2 MIPS. Для контролю внутрішньої температури мікроконтролера вбудований температурний датчик. На його виході формується напруга в

залежності від навколишньої температури. Вихід датчика через мультиплексор підключається до АЦП. Використовуючи температурний датчик, можна вимірювати температуру від -40 до $125\text{ }^{\circ}\text{C}$ з точністю $\pm 1,5\text{ }^{\circ}\text{C}$.

ЦАП має роздільну здатність 12 біт, перетворення можливо в 8/12-бітовому форматі з вирівнюванням цього результату по лівому або правому краях. Так як ЦАП містить два канали, є можливість формування стерео сигналу. Доступна функція автоматичної генерації шумового сигналу з мінливою амплітудою або трикутного сигналу.

Швидкий старт і налагодження. Для підключення мікроконтролера до налагоджувальних засобів використовується чотирьох провідний інтерфейс JTAG, або двох провідний SWD (Serial Wire Debug). Виходи SWD-інтерфейсу мультиплексовані з виходами JTAG. Розробнику дається можливість у виборі інтерфейсу налагодження / програмування. Інтегровані макрокомірки трасування (Embedded Trace Macrocell) значно розширюють функції налагодження, дозволяючи спостерігати за потоком інструкцій і даних всередині ядра CPU в реальному часі. Дані виводяться через внутрішні ETM-виводи на зовнішній TDA-аналізатор (аналізатор телеметричних даних). Існує також спеціальний пакет в Matlab для генерування і створення коду алгоритмів цифрової обробки сигналів для ядра CortexM4.

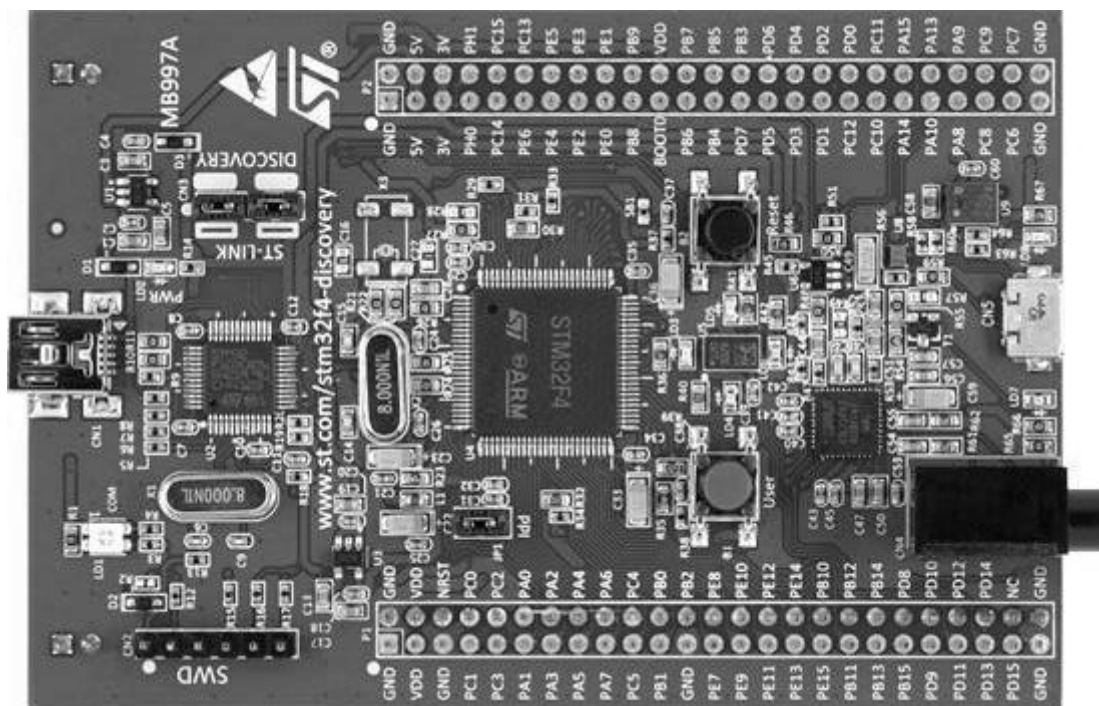


Рисунок 5.5 – Плата STM32F4-DISCOVERY. Вид зверху

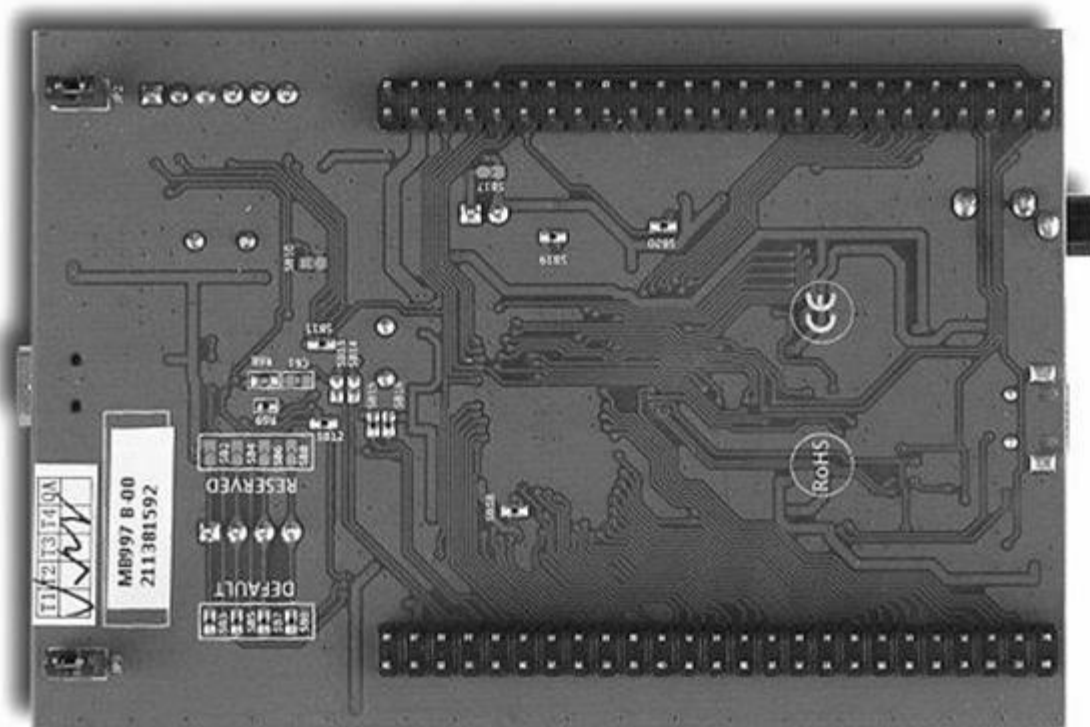


Рисунок 5.6 – Плата STM32F4–DISCOVERY. Вид знизу

Висновки на основі огляду архітектури мікроконтролерів STM32.

На сьогоднішній день колишні лідери ринку в області вбудованих рішень 8 і 16 бітні мікроконтролери були практично витіснені повноцінними 32–бітними мікроконтролерами, на зразок ядра CortexM. Це дозволило розробникам не вникати у внутрішню будову, думаючи на рівні архітектури, а зосередитися на більш високому – програмному рівні. Таким чином грань між персональними комп'ютерами і мікроконтролерами значно розмилася. Проблемами, які відштовхували розробників від 32–бітових контролерів кілька років тому були висока вартість кристала, складність його сполучення із зовнішнім світом, високе енергоспоживання. Коли виробники зменшили енергоспоживання кристалів за рахунок створення безлічі механізмів відключення окремих модулів в непотрібні моменти часу і впровадили в свої пристрої високоякісні модулі шифрування, мнотинні АЦП і ЦАП, модулі зв'язку – почалася епоха цифрових сигнальних процесорів. Однак очевидно, що задовольняє більшість сучасних прикладних задач кристал буде вимагати охолодження, еквівалентного охолодження ядерного реактора, тому в певний момент гонка за максимальною частотою тактування закінчилася. Почалася епоха паралельних обчислень. Можна стверджувати, що МК серії STM32F4 знаходяться посередині – на цьому перехідному етапі, безумовно займаючи свою нішу.

Однак оскільки майбутнє за паралельними обчисленнями, варто звернути особливу увагу на пристрої FPGA, ASIC, CPLD і інші, оскільки саме ці пристрої будуть прабатьками нових технологій в недалекому майбутньому.

5.4 Короткий огляд ЦПП Cortex в мікроконтролерах.

ARM Cortex – це нове покоління процесорів, виконаних в рамках стандартної архітектури і в той же час відповідають строгим технологічним вимогам. У порівнянні з іншими ЦПП ARM, сімейство Cortex об'єднує системну архітектуру зі стандартним ЦПП в завершеному процесорному ядрі. Дане сімейство можна розбити на три основних профілю: профіль А для застосування в складі пристроїв з високою продуктивністю, профіль R для застосування в складі систем, які працюють в реальному часі і профіль М для застосування в складі дизайнів з жорсткими вимогами щодо вартості кінцевого виробу.

В основі мікроконтролерів STM32 лежить профіль CortexM3, спеціально розроблений для проектів, де необхідні розвинені системні ресурси і, при цьому, мале енергоспоживання. Їх низька вартість дозволяє конкурувати з популярними серед розробників електронних пристроїв в пострадянському просторі 8 і 16 бітними мікроконтролерами, практично витісняючи останні з ринку. Не дивлячись на успішну інтеграцію ЦПП ARM7 і ARM9 в стандартні мікроконтролери, в них все ще простежується початкова орієнтованість на системи на кристалі (SoC), це помітно по способу обробки виняткових ситуацій і переривань, оскільки кожен виробник реалізує дані механізми в своїх продуктах по-своєму.

Однак CortexM4 являє собою ядро, що підкоряється суворій системі стандартизації і, крім ЦПП, що містить також інші елементи, що складають основу мікроконтролера такі, як система переривань, системний таймер SysTick, повноцінна система налагодження і карта пам'яті. Чотирьох Гб адресний простір CortexM4 складається з чітко розподілених областей коду програми, статичного ОЗП, пристроїв введення–виведення і системних ресурсів. Основна відмінність від ядра ARM7 полягає в тому, що ядро CortexM4 виконано по Гарвардської архітектури, отже має декілька шин, що дозволяють виконувати операції паралельно. Дане сімейство також має відміну від попередніх ARM–архітектури в можливості проводити операції з фрагментованими даними. Даний підхід гарантує, що внутрішній статичний ОЗП буде використано максимально ефективно. Сімейство також надає можливість доступу до двох розділених банках пам'яті розміром 1 Мбайт в режимі bit–bang. Такий підхід дозволяє раціонально використовувати ресурси контролера для доступу до регістрів і прапорів ПБВ, розташованим в області статичного ОЗП, це виключає необхідність інтеграції повнофункціонального бітового процесора (рис.5.7).

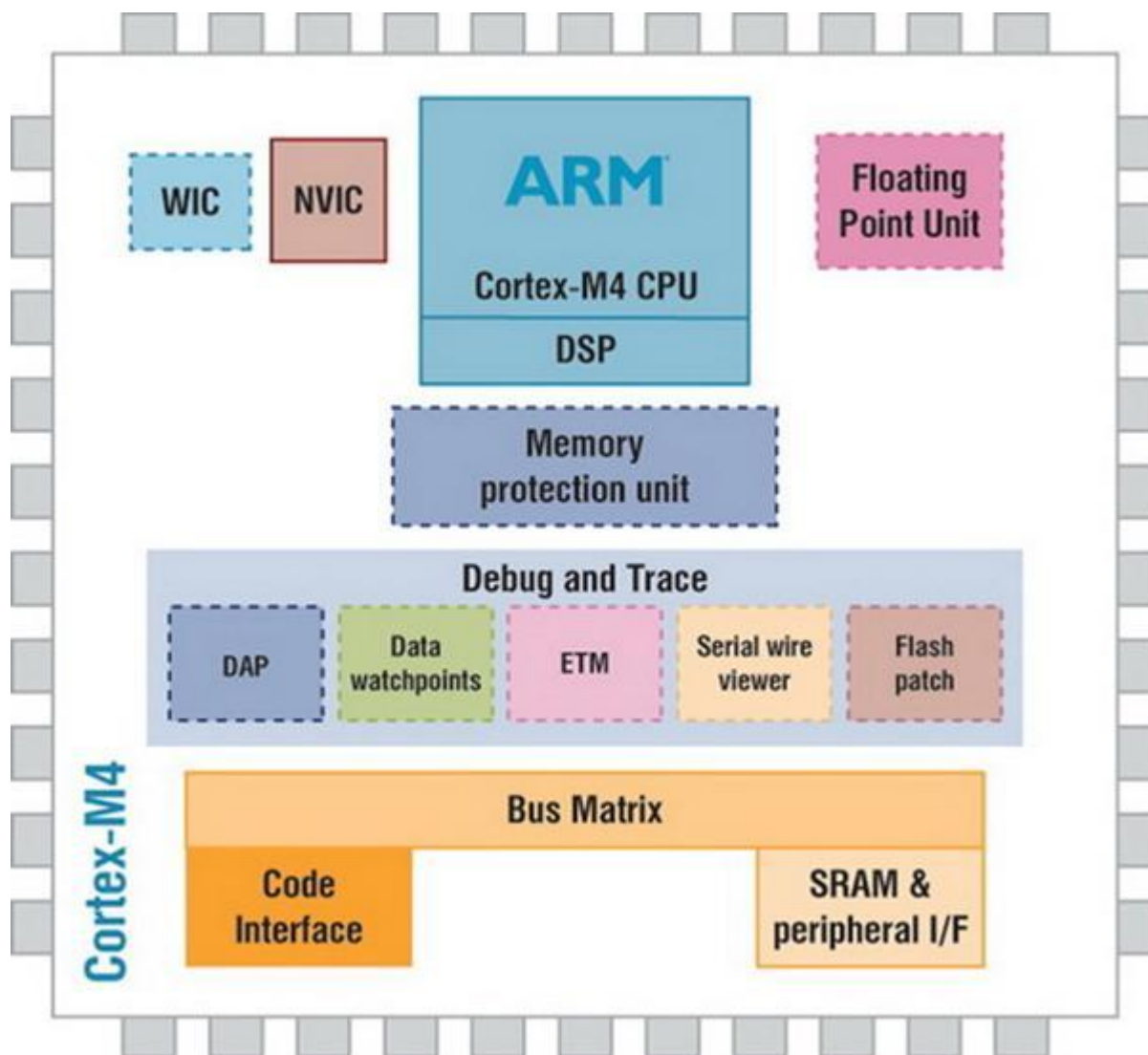


Рисунок 5.7 – Процесорне ядро ARM CortexM4

Один з найважливіших компонентів ядра CortexM4 – контролер векторизованих вкладених переривань (КВВП), що надає узагальнену структуру переривань і методи їх обробки. КВВП визначає вектори переривань для близько двохсот джерел, і кожному може бути призначений свій пріоритет. При розробці такого механізму було зроблено акцент саме на швидкодії обробки переривань, таким чином час з моменту отримання запиту на переривання до його обробки не перевищує дванадцяти циклів. Це досягається з використанням автоматизованих стекових операцій, виконуваних вбудованим в ЦПП спеціальним мікро кодом. При одночасному надходженні переривань КВВП здійснює впорядковану обробку переривань, при чому затримка виклику чергової процедури переривання не перевищує шести циклів. Питання накладення переривань залишається також актуальним (як і для більшості сучасних мікроконтролерів), так як в разі накладення, переривання з більш високим пріоритетом може витіснити більш низько пріоритетне переривання без витрат на додаткові цикли ЦПП. На структуру організації переривань в ядрі CortexM4 значно впливають

економічні режими роботи. Так існує можливість автопереходу в економічний режим роботи після закінчення обробки переривання при відповідній конфігурації ЦПП. Після такого переходу ядро не діятиме до виникнення наступної виняткової ситуації. На структуру організації переривань в ядрі CortexM4 значно впливають економічні режими роботи. Так існує можливість автопереходу в економічний режим роботи після закінчення обробки переривання при відповідній конфігурації ЦПП. Після такого переходу ядро не діятиме до виникнення наступної виняткової ситуації. На структуру організації переривань в ядрі CortexM4 значно впливають економічні режими роботи. Так існує можливість автопереходу в економічний режим роботи після закінчення обробки переривання при відповідній конфігурації ЦПП. Після такого переходу ядро не діятиме до виникнення наступної виняткової ситуації.

Спочатку ядро CortexM4 розроблялося як недороге, однак воно залишається 32-х бітовим ЦПП, відповідно підтримує два режими роботи: *threading і handling*, кожен з яких може мати свої стеки. Це дає можливість розробляти інтелектуальне ПО, використовуючи операційні системи реального часу (RTOS). До складу ядра також входить самостійно перезавантажується 24-х бітовий таймер, який використовується для генерації інтервальних періодичних переривань і широко використовуваний більшістю RTOS. Сімейство Cortex на відміну від ЦПУ ARM7 і ARM9, які мають 32-х бітовий ARM і 16-ти бітовий Thumb набори інструкцій, підтримує набір інструкцій Thumb-2, що представляє собою суміш 16-ти і 32-х бітових інструкцій. Це дозволяє досягти продуктивності 32-х бітового набору ARM і властивій 16-ти бітовому набору Thumb щільності коду.

На сьогоднішній ядро CortexM4 – є останньою сходинкою сімейства CortexM, що дає можливість обробляти сигнали на рівні DSP. Додана підтримка можливості множення з накопиченням за один такт, арифметика з насиченням, обчислення з плаваючою комою, команди управління потоком даних SIMD (single instruction multiple data). Блок обчислень з плаваючою комою повністю відповідає стандарту IEEE 754 і дає процесору можливість здійснення багатьох базових матеріально численних математичних операцій. Ядро CortexM4 розроблено так, що б інструкції DSP виконувалися за один такт, на рівні ядра підтримується *математичний співпроцесор FPU (Floating Point Unit)* – блок, що виконує операції з плаваючою точкою, *FPU* допомагає основному процесору виконувати математичні операції над числами. STM32F4xx застосовується скрізь, де існує необхідність обробляти цифрові сигнали: контроль над кроковими двигунами, обробка звукових і відеосигналів, технології радіолокації, навігації, інше. Мікроконтролер може легко виконати цифрову фільтрацію, згортку або швидке перетворення Фур'є.

5.5 Програмні засоби розробки для мікроконтролерів

Знайомство з інтегрованим середовищем розробки програмного забезпечення для мікроконтролерів STM – STM32CubeMX. Інтегроване середовище

розробки програмного забезпечення STM32CubeMX дозволяє виконати повний цикл розробки програми для МК. Це означає, що даний пакет включає в себе:

- вбудований редактор, що дозволяє створювати і редагувати вихідні файли;
- компілятор, необхідний для створення об'єктного файлу;
- компоновщик (або линкер), що дозволяє об'єднати в один об'єктний файл кілька вихідних і бібліотечних файлів;
- різноманітні засоби налагодження програмного забезпечення (набір спеціальних засобів перегляду стану внутрішніх регістрів, пам'яті, засоби візуалізації даних і ін.).

Перед початком роботи необхідно встановити на робочому комп'ютері програмне забезпечення STM32CubeMX і підключити оцінний модуль (схема модулю приведена в Додатку).

Запуск програми здійснюється з меню за допомогою іконки на робочому столі комп'ютера (рис. 5.8).

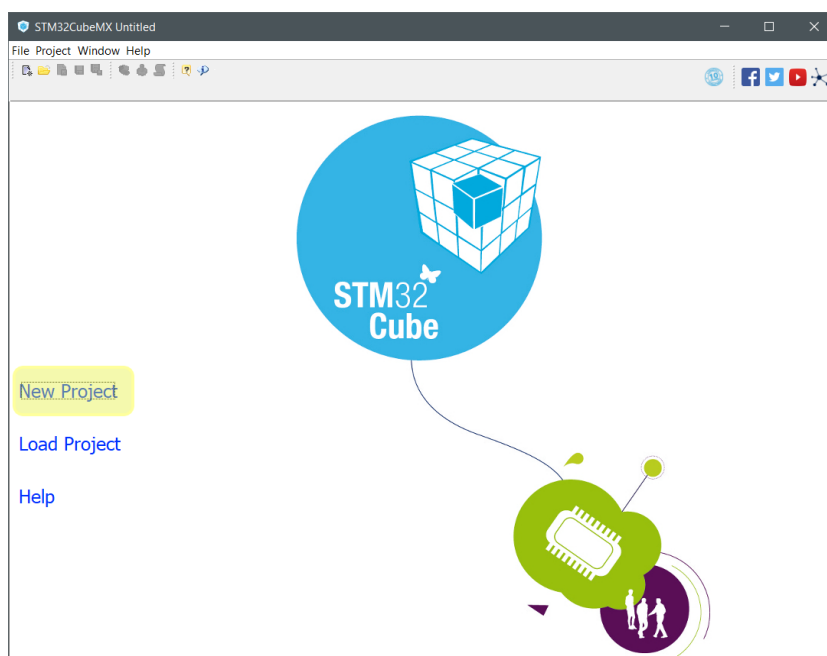


Рисунок 0.8 – Початкова сторінка STM32CubeMX

Вказати потрібний мікроконтролер (STM32F030C8) у полі пошуку (рис.5.9)

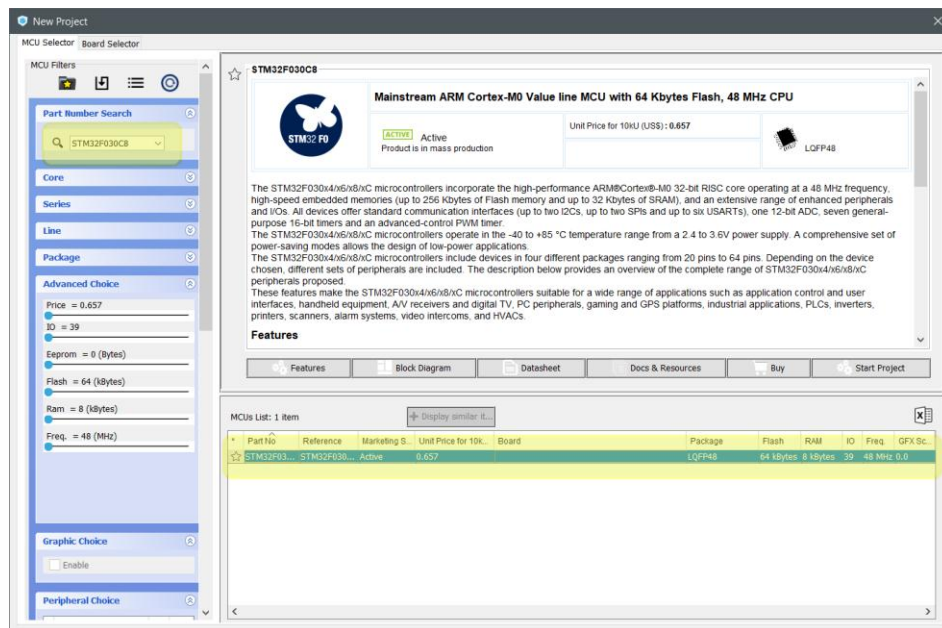


Рисунок 0.9 – Вікно вибору мікроконтролера

Для налаштувати програматора-відладчика STLINK у SYS обрати Debug serial wire (рис.5.10).

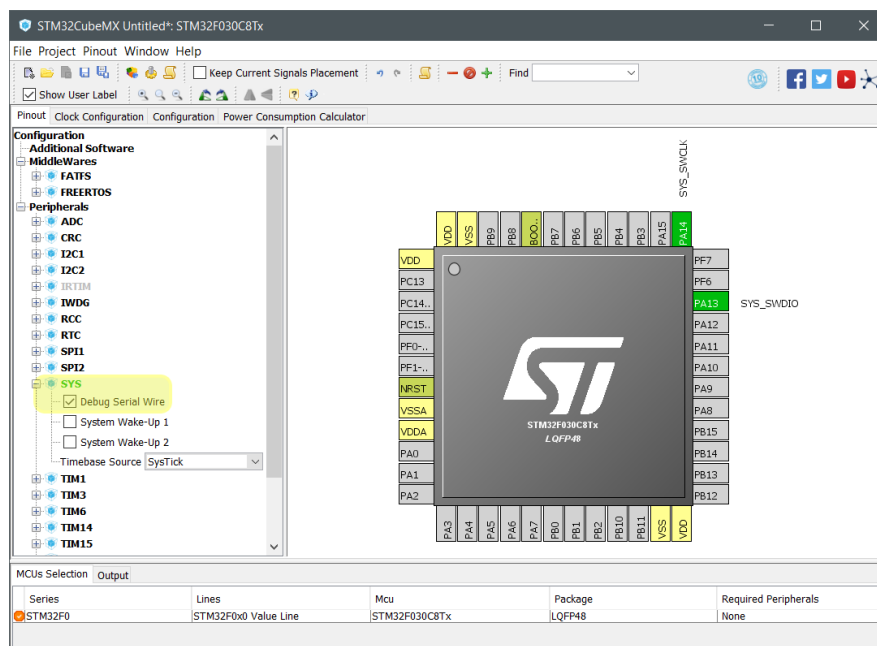


Рисунок 0.10 – Налаштування відладчика

Перейти на вкладку налаштування системи тактування (Clock configuration) (рис.5.11).

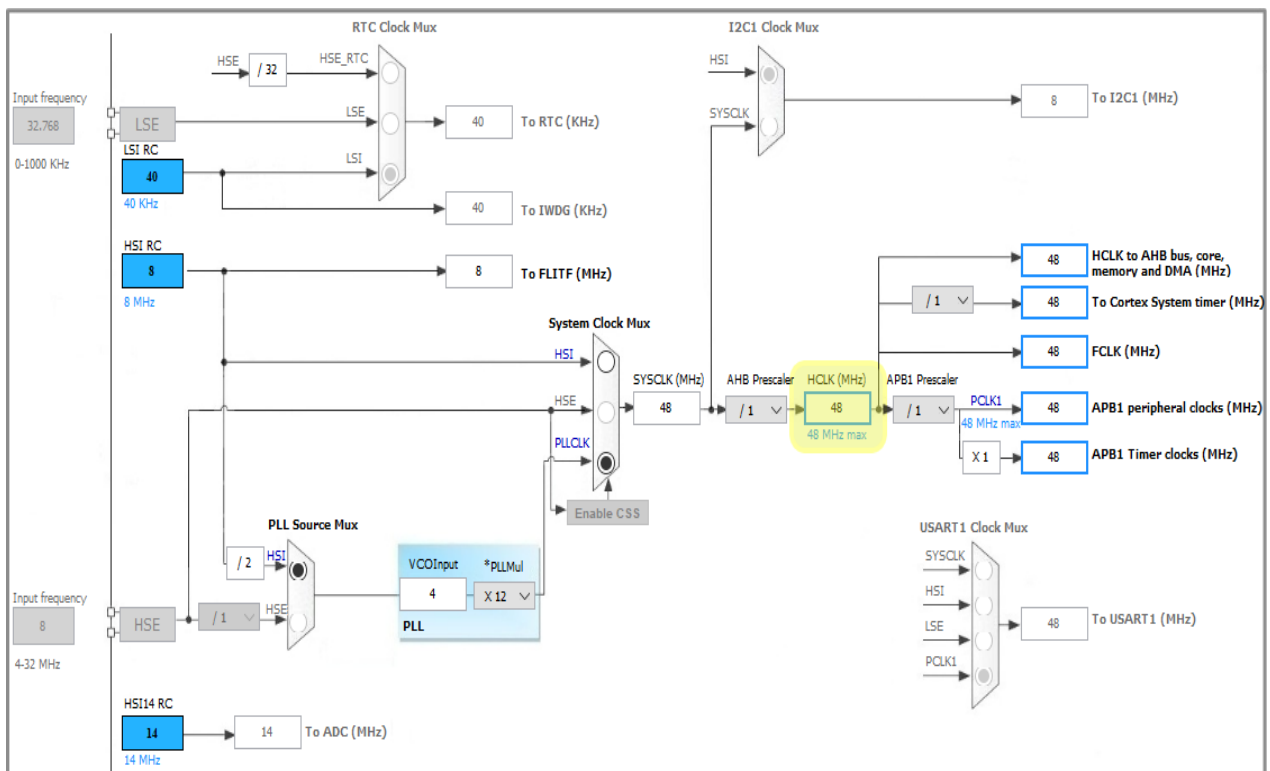


Рисунок 0.11 – Вікно налаштування системи тактування

В якості джерела опорної тактової частоти вибрати внутрішній RC-генератор. У полі HCLK вказати частоту 48 МГц та ввести значення, дільники та множники підберуться автоматично.

У вкладці Pinout (рис.5.12) вибираємо потрібний пристрій вводу-виводу мікроконтролера та обираємо на вхід (GPIO_Input) або вихід (GPIO_Output) він буде працювати.

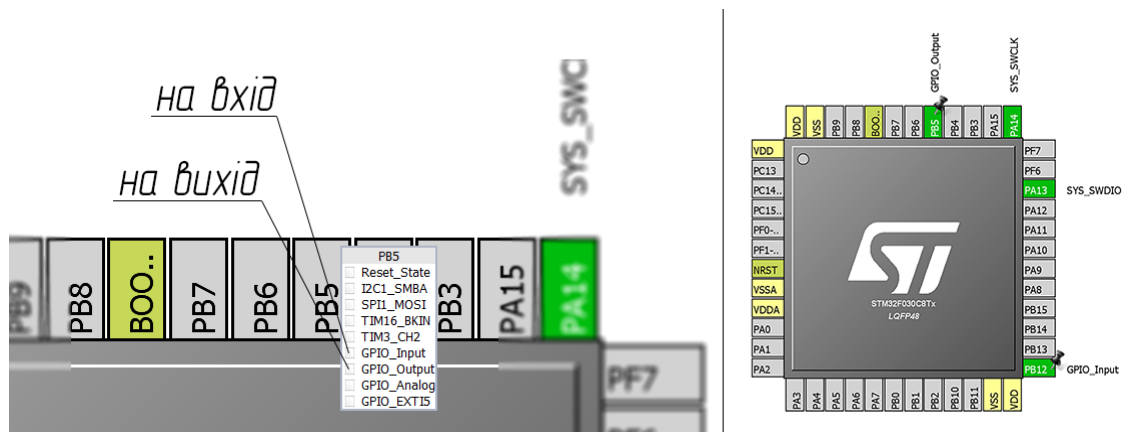


Рисунок 0.12 – Налаштування виводів мікроконтролера

Подальші налаштування проводяться у вкладці Configuration (налаштування). Натискаємо на GPIO (англ. *general-purpose input/output*, порти введення-виведення загального призначення) (рис.5.13).

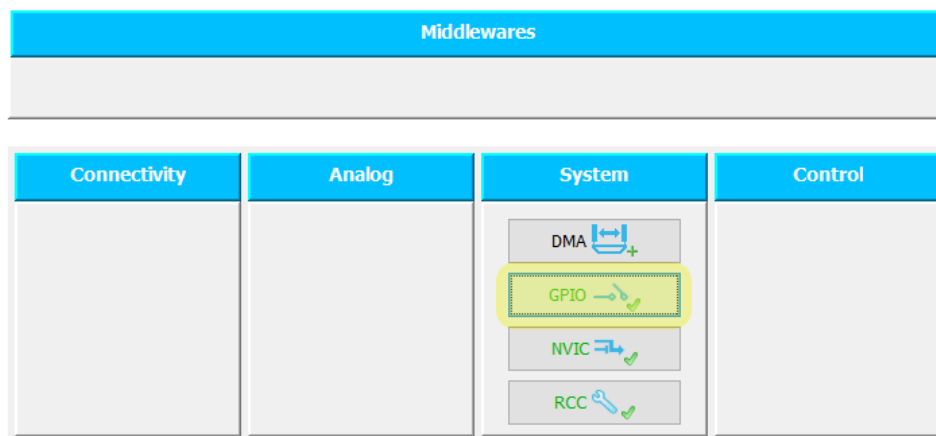


Рисунок 0.13 – Налаштування периферії

Обираємо необхідний пристрій вводу/виводу та налаштовуємо його призначення (рис.5.14).

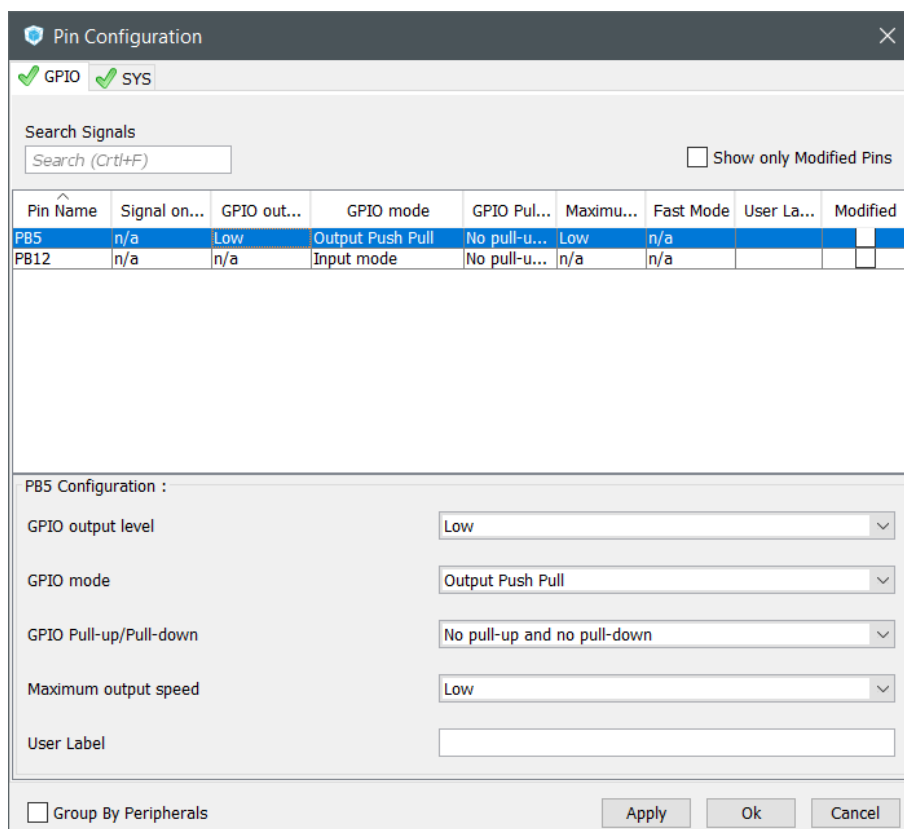
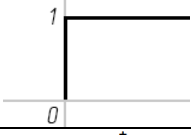
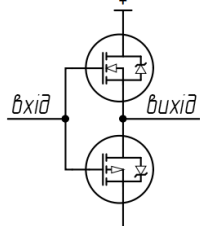
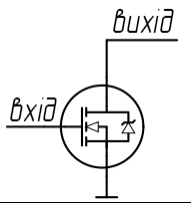
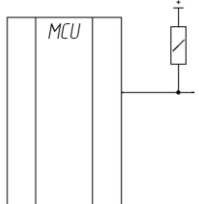
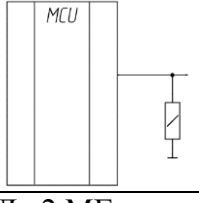


Рисунок 0.14 – Налаштування виводу мікроконтролера, який працює на вихід

Налаштовуємо виводи на двотактний вихід (Output Push-Pull), на виході низький рівень (Low), без підтяжки (No pull-up and no pull-down:), швидкість низька (Low) (табл.5.1).

Таблиця 0.1 – Налаштування для виводу на вихід

GPIO Output level (Що буде на виході)	Low (Низький рівень)	
	High (Високий рівень)	
GPIO mode (Режим роботи)	Output Push-Pull (Двотактний вихід)	
	Output Open Drain (Вихід з відритим стоком)	
GPIO Pull-up/Pull-down (Підтяжка)	No pull-up and no pull-down	Без підтяжки
	Pull-up (Підтяжка до плюса)	
	Pull-down (Підтяжка до мінуса)	
Maximum output speed (Найбільша вихідна швидкість)	Low (Низька)	До 2 МГц
	Medium (Середня)	До 10 МГц
	High (Висока)	До 50 МГц

Налаштовуємо підтяжку до виводу, до якого підключена кнопка. Згідно до схеми (Додаток X) при натисненні на В3 вивід PB12 буде замикатись на мінус живлення, а у часі простою рівень (лог.0 або лог.1) на виводі PB12 не визначений, що призведе до хибних спрацювань. Для того щоб такого не сталося потрібно включити резистор між цим виводом та плюсом живлення, тоді на ньому буде чітко визначена логічна одиниця (високий рівень) весь час, поки кнопка В3 не замкнена. Зручно користуватися внутрішньою підтяжкою, для цього потрібно у полі GPIO Pull-up/Pull-down просто обрати підтяжку до плюса (Pull-up) або підтяжку до мінуса (Pull-down) (рис.5.15).

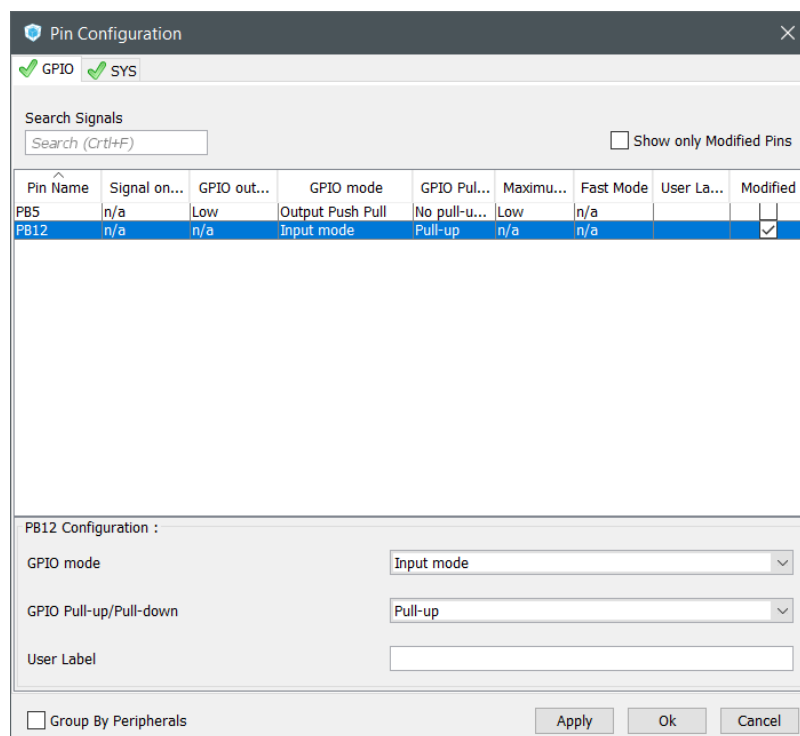


Рисунок 0.15 - Налаштування виводу мікроконтролера, який працює на вхід

Створюємо проект, для цього тиснемо Ctrl + Shift + G або Project -> Generate Code. У полі Toolchain/IDE обрати MDK-ARM v5, назва будь-яка. Після цього відкриваємо проект – Open project (рис.5.16).

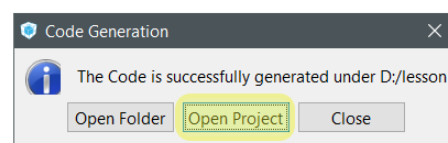
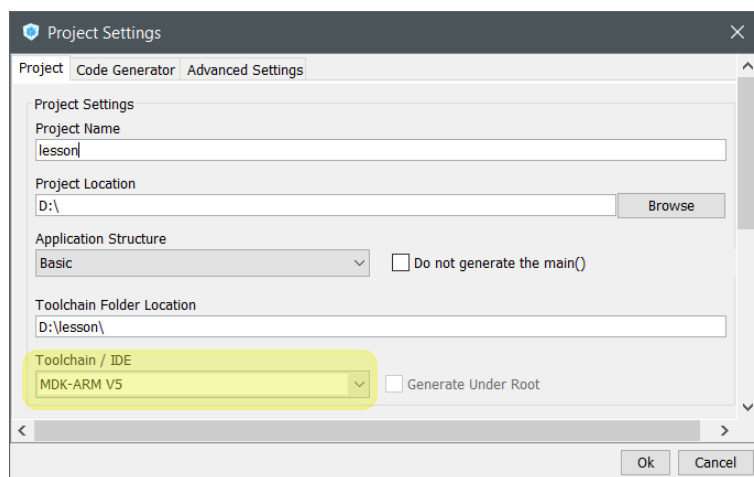


Рисунок 0.16 – Створення проекту

Проект відкриться у середовищі розробки Keil μ Vision. Відкриваємо файл main.c – це головний файл проекту (рис.5.17).

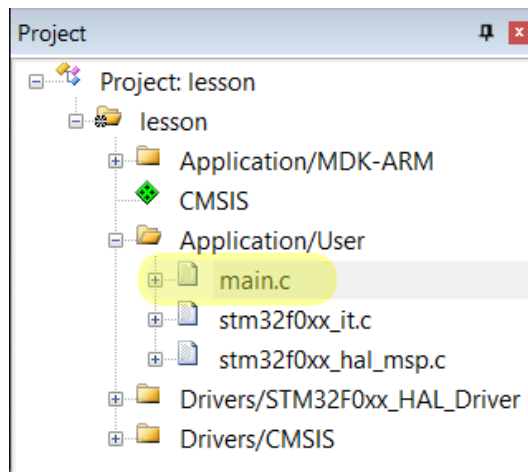


Рисунок 0.17 – Дерево проекту у середовищі Keil

Код потрібно писати у відведених для цього місцях, це потрібно для того щоб він не втрачався після переналаштувань у STM32CubeMX (рис.5.18).

```

34  * OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
35  * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
36  *
37  *****
38  */
39  /* Includes ----- */
40  #include "main.h"
41  #include "stm32f0xx_hal.h"
42
43  /* USER CODE BEGIN Includes */
44  /* USER CODE END Includes */
45
46  /* Private variables ----- */
47
48  /* USER CODE BEGIN PV */
49  /* Private variables ----- */
50  /* USER CODE END PV */
51
52  /* Private function prototypes ----- */
53
54  void SystemClock_Config(void);
55  static void MX_GPIO_Init(void);
56
57  /* USER CODE BEGIN PFP */
58  /* Private function prototypes ----- */
59
60  /* USER CODE END PFP */
61
62  /* USER CODE BEGIN 0 */
63  /* USER CODE END 0 */
64
65  /* USER CODE BEGIN 2 */
66  /* USER CODE END 2 */
67
68  /* Infinite loop */
69  /* USER CODE BEGIN WHILE */
70  while (1)
71  {
72      /* USER CODE END WHILE */
73      /* USER CODE BEGIN 3 */
74      /* USER CODE END 3 */
75  }
76  }

```

включення
(підключення бібліотек або інших заголовних файлів)

глобальні змінні
(такі змінні діють по всьому коду, записаному у цьому файлі)

виконується один раз
(код, записаний тут виконується один раз після підключення живлення/скидання МК)

виконується один раз

буде виконуватися постійно
(код, записаний сюди виконується поки є живлення*)

Рисунок 0.18 – Вміст сформованої головної програми

Для збірки (компіляції проекту) вибрати команду F7 або Project → Build Project.

Для завантаження прошивки до мікроконтролера натиснути F8 або Flash → Download.

Для того, щоб завантажити прошивку до мікроконтролера та перейти у режим відладки натиснути Ctrl + F5 або Debug → Start/Stop Debug Session.

Після кожної прошивки МК потрібно натискати кнопку Скидання B1 (Reset) на платі, щоб прошивка почала працювати. Для налаштування автоматичного скидання:

- 1) Викликати проект → Options for Target
- 2) Перейти на вкладку Debug, вибрати на Settings
- 3) На вкладці Flash download відмітити Reset and Run

Пояснення до роботи. Ознайомлення з основними функціями бібліотеки HAL (Hardware Abstraction Layer), котра значно полегшує програмування на мікроконтролерах STM. (Далі в тексті: x – літера порту, y – номер виводу).

- Встановлення високого рівня (логічна 1) на виводі:
HAL_GPIO_WritePin(GPIOx, GPIO_PIN_y, GPIO_PIN_SET);
GPIOx→BSRR |= GPIO_BSRR_BS_y;
- Встановлення високого рівня (логічний 0) на виводі:
HAL_GPIO_WritePin(GPIOx, GPIO_PIN_y, GPIO_PIN_RESET);
GPIOx→BSRR |= GPIO_BSRR_BR_y;
- Переключення рівня на виводі (інвертування):
HAL_GPIO_TooglePin(GPIOx, GPIO_PIN_y);
GPIOx→ODR |= GPIO_ODR_OD_y;
- Зчитування стану на виводі:
HAL_GPIO_ReadPin(GPIOx, GPIO_PIN_y);
GPIOx→IDR & GPIO_IDR_ID_y
- Програмна затримка тривалістю 1 с (час пишеться у мс):
HAL_Delay(1000);

Наведені вище функції працюють з основними регістрами мікроконтролерів STM, вони встановлюють/скидають біти, перевіряють високий чи низький рівень на виводі, їхня робота на більш низькому рівні теж показана, для їх розуміння потрібно розуміти роботу побітових операцій.

Робота з перемикачами та кнопками. Застосування перемикачів при роботі з цифровими схемами завжди пов'язано з ризиком непевної роботи пристрою внаслідок наявності брязкоту контактів. Є декілька видів програмного усунення впливу брязкоту контактів:

1. Внесення затримки: при першій зміні стану від одного до іншого (передній чи задній фронт) відбувається зчитування стану на виводі, а далі затримка на час брязкоту.

2. Підрахунок числа співпадаючих значень: через визначені проміжки часу (десятки мс) опитується стан на виводі, якщо він підтверджується визначену

кількість разів, то кнопка вважається увімкненою.

Завдання для виконання:

1) Створити проект у STM32CubeMX за прикладом попередньої роботи з налаштуванням виводу до якого підключена кнопка на вхід з підтяжкою до плюсу/мінусу живлення (див. додаток);

2) Визначити змінну з будь-якою назвою (латинськими літерами) типу `uint32_t` у відповідному місці.

Таблиця 5.2 – Опис 32-бітної цілнчисленної без знакової змінної

Тип змінної		Кількість байт	Допустимі значення
<code>uint32_t</code>		4 байти	0...4294967295
unsigned (беззнакова)	integer (цілнчисленна)	4 байт = 32 біт	$0 \dots (2^{32} - 1)$

3) У головній програмі в циклі `while(1){}`, де команди виконуються постійно написати код, у якому при натисненні на кнопку B2 відбувається збільшення значення створеної змінної на одиницю, а при натисненні на кнопку B3 зменшення цієї ж змінної на одиницю.

4) Зібрати проект (F7), завантажитися в режимі відлагодження (Ctrl + F5), вибрати створену змінну та обрати Add to Watch 1, вона з'явиться у вікні Watch 1. Виділивши Add to Watch 1 прибрати позначку з поля Hexadecimal Display (для відображення в десятинній системі) – це дозволяє спостерігати за її значенням у режимі реального часу (рис.5.19).

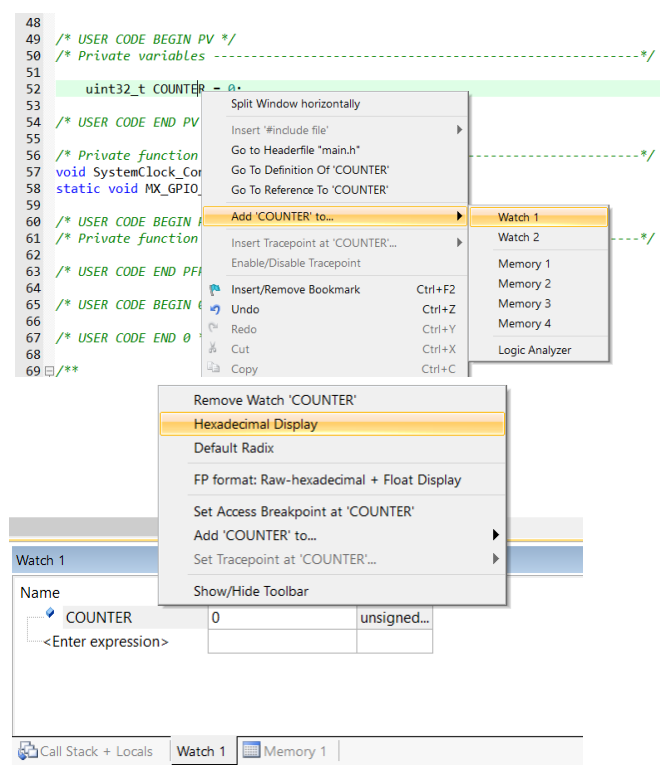


Рисунок 5.19 – Додавання змінної до вікна перегляду

Вибрати Peripheral System → Viewer, обрати GPIOB (рис.5.20), розкрити для перегляду регістр вхідних даних IDR (*Input Data Register*).

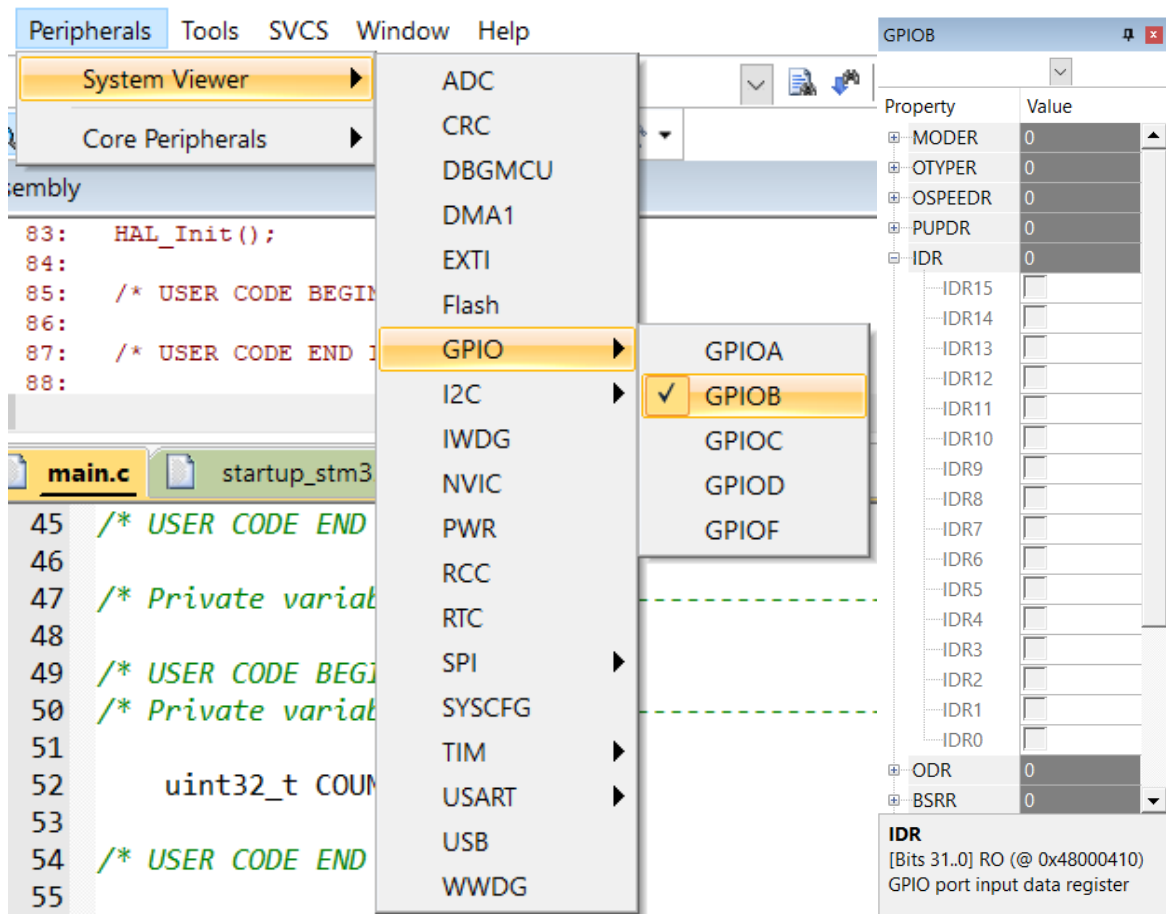


Рисунок 5.20 – Перегляд регістру вхідних даних порту GPIOB

5) Запустити роботу програми (F5). При натисканні на кнопки B2/B3 повинні пропадати та з'являтися позначки навпроти відповідного виводу (таким чином можна пересвідчитись, що кнопка дійсно працює і дізнатися який логічний рівень на виводі). Значення змінної буде приймати хаотичні значення, не буде чіткого збільшення та зменшення її значення при натисканні – все через брязкіт контактів (рис.5.21).

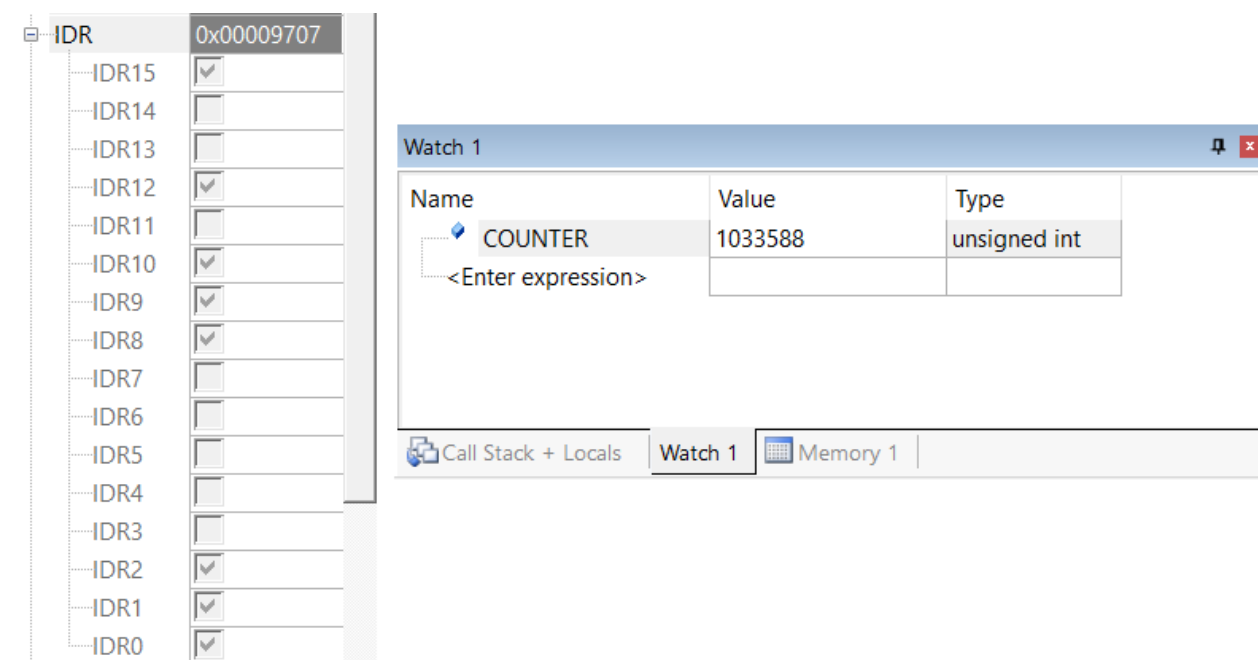


Рисунок 5.21 – Зміна значення бітів (IDRx) при натисненні кнопок B2/B3 та значення змінної (вікно Watch 1)

6) Першим способом усунення впливу брязкоту контактів є внесення затримки (50-400 мс) одразу після того, як мікроконтролер зчитав стан на виводі та виконав дію. Таким чином впродовж часу брязкоту не буде відбуватися зчитувань. Повторити виконання попереднього пункту, пересвідчившись у тому, що тепер збільшення/зменшення значення змінної відбувається чітко при кожному натисканні кнопок.

7) Встановлене значення змінної (рис.5.22).

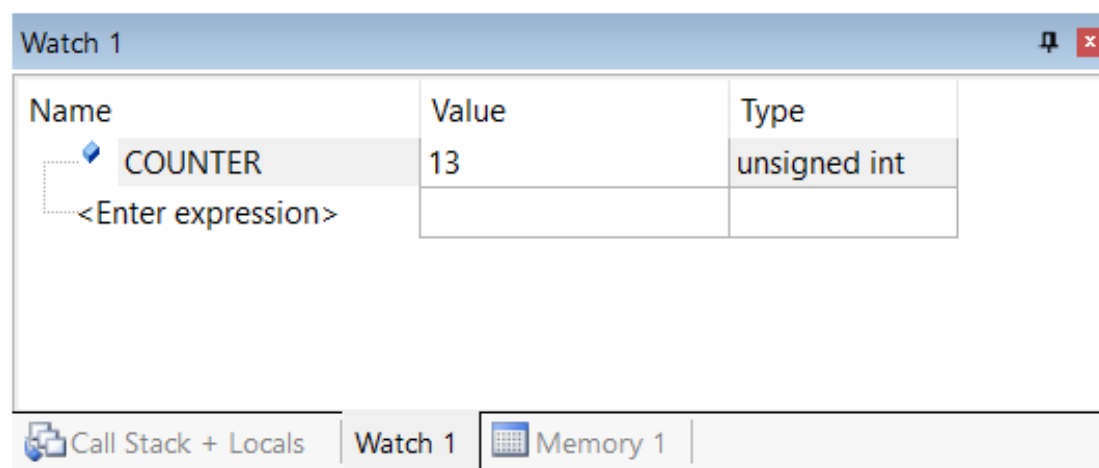


Рисунок 5.22 – Вікно встановленого значення змінної з урахуванням внесеної затримки

8) Код, написаний у головному циклі, який постійно перевіряє натиснення кнопок – це нерозумне використання потужностей МК й годиться тільки у

навчальних цілях. Створимо переривання, яке буде викликатися 30-60 разів у секунду та перевіряти натиснення кнопок, таким чином з'явиться псевдо паралельність. У STM32CubeMX обираємо частоту тактування таймера.

9) Переходимо у вікно налаштувань (Configuration) та одразу у вкладці NVIC Interrupt ставимо позначку, тим самим дозволяємо перериванню викликатися від даного таймера.

10) Налаштовуємо частоту виклику переривань, для цього у строках PSC (Prescaler – перед дільник) та ARR (Auto-Reload Register – регістр перезавантаження) записуємо числа, розраховані за наступним виразом:

Пояснення: частоту роботи таймерів дивимося у вкладці Clock Configuration. Є дві шини, від яких тактуються різні таймери, дізнатися які конкретно можна з технічної документації на цей мікроконтролер:

Бачимо цифру 48 МГц = 48000000 Гц, це означає що теоретично таймер за одну секунду дорахує від 0 до 48000000, таким чином рахунок буде відбуватися з періодом $1/48000000 \approx 21$ нс.

Це дуже велика частота (малий період), нам потрібно зменшити її, для цього у перед дільник PSC записується число 0-(65536-1) (бо регістр 32-бітний).

Таким чином можна отримати частоту роботи таймера ≈ 366 Гц, але нам потрібно ще менше, тому в регістр перезавантаження ARR записується число, до якого буде рахувати таймер і тільки потім його значення буде збільшуватися на одиницю.

11) Перетворюємо проект (Ctrl + Shift + G), заходимо в відкритий проект у Keil, погоджуємося зі зміною файлів.

12) Переривання не будуть викликатися поки ми не запустимо їх (один раз), для цього використовується функція (n – номер таймеру):

HAL_TIM_Start_IT(&htmx);

13) Обробники переривань знаходяться у файлі it (двічі натискаємо на нього. В обробнику від таймера запишемо код, який перевіряє стан на виводі PB12 та PB13 (до них підключені кнопки) та у разі натиснення змінює значення змінної.

14) Запускаємо відладчик (Ctrl + F5), обираємо відповідний таймер у периферії. Запускаємо роботу програми (F5). Значення у полі CNT (Count) повинні змінюватися (таймер в рахунковому режимі), в регістрах PSC та ARR повинні знаходитися числа, які були записані раніше (вони представлені у шістнадцятиричній системі відліку).

5.6 Контрольні запитання і завдання

1. Як завантажити прошивку до мікроконтролера?
2. Які налаштування портів на вхід?
3. Що включає в себе пакет STM32CubeMX?
4. Як налаштувати частоту виклику переривань?
5. Які способи усунення впливу брязкоту контактів?
6. В чому полягає роль змінної в усуненні брязкоту контактів?

7. Які налаштування портів на вихід?
8. Базові характеристики сімейства мікроконтролерів STM32F4?
9. В чому відмінність сімейство Cortex від стандартних ЦПП ARM?
10. Які функції контролеру векторизованих вкладених переривань?
11. Дайте характеристику сімейству STM32F4xx.
12. Які аналогові блоки входять до складу STM32?
13. Які існують пакети інструментів для розробника?
14. Який склад середовища розробки програмного забезпечення STM32CubeMX?
15. Які комунікаційні інтерфейси входять до складу STM32?
16. В який спосіб виконується налагодження програмного забезпечення?
17. Яка структура мікроконтролерів STM32F4xx?
18. Дайте загальну характеристику мікроконтролерів сімейства STM32F4.
19. Які функції інтерфейсу FSMC (Flexible Static Memory Controller)?
20. Які функції інтерфейсу I²S?

6. ПРОГРАМОВАНІ ЛОГІЧНІ КОНТРОЛЕРІ В ЗАСОБАХ АВТОМАТИЗОВАНОГО УПРАВЛІННЯ

Основним і найбільш перспективним методом автоматизованого управління складними динамічними системами та процесами в життєво важливих і критичних, з точки зору безпеки і надійності, областях є системи диспетчеризації, управління та збору даних – SCADA (Supervisory Control and Data Acquisition), що базуються на мікропроцесорних засобах контролю і керування. Використання SCADA–систем дозволяє здійснювати ефективне управління автоматизованими системами в промисловості та енергетиці, транспорті, машинобудуванні, автоматизації будівель, водопостачанні, транспорті.

Основні функції SCADA-системи:

- збір, обробка і передача даних
- відображення даних в цифровому і графічному вигляді (візуалізація)
- архівація (бази даних, бібліотеки, паспортизація обладнання)
- оперативне інформування персоналу (тривоги і події)
- формування звітів, аналіз, планування

Базовим елементом SCADA з початку розробки її концепції стали програмовані логічні контролери. *Програмований Логічний Контролер* (ПЛК; англ.: Programmable Logic Controller або PLC) – це спеціалізована мікропроцесорна система, яка використовується для автоматизації технологічних процесів і загально промислових установок і комплексів.

ПЛК мають ряд особливостей, що відрізняють їх від інших електронних приладів, які застосовуються в промисловості:

- на відміну від мікроконтролера (одно кристального комп'ютера) - мікросхеми, призначеної для управління електронними пристроями - ПЛК є самостійним пристроєм, а не окремою мікросхемою;
- на відміну від комп'ютерів, орієнтованих на прийняття рішень і управління оператором, ПЛК орієнтовані на роботу з машинами через розвинений введення сигналів датчиків і висновок сигналів на виконавчі механізми;
- на відміну від вбудованих систем ПЛК виготовляються як самостійні вироби, окремі від керованого при його допомоги обладнання.

Таким чином, основна сфера застосування ПЛК – це сфера масштабного промислового виробництва. Однак вони також використовуються для автоматизації в інших галузях та додатках (контроль доступу в приміщення, управління освітленням, обігрівом, вентиляцією і кондиціонуванням повітря, управління ліфтами, ескалаторами і т. п.).

Архітектура програмованих логічних контролерів багато в чому нагадує архітектуру ПК загального призначення зі спеціалізованими модулями вводу/виводу (I/O). Однак деякі важливі характеристики відрізняють ці пристрої. По-перше, ПЛК набагато надійніші і розраховані на безвідмовну роботу протягом багатьох років – і це їх найважливіша особливість. По-друге, ПЛК можуть використовуватися в умовах промислового виробництва, де їм доводиться пра-

цювати в умовах серйозного електромагнітного випромінювання, вібрацій, екстремальних температур і вологості. По-третє, ПЛК легко обслуговуються технічним персоналом на виробництві.

Апаратна архітектура ПЛК. Основи.

В архітектурі ПЛК знаходяться такі основні компоненти:

- процесорний модуль,
- блок живлення,
- модулі вводу/виводу,
- підсистема реального часу;
- незалежна пам'ять;
- інтерфейси послідовного введення-виведення (RS-485, RS-232, Ethernet)
- схеми захисту і перетворення напруги на входах і виходах ПЛК.

Процесорний модуль складається власне з ядра, шин обміну даними і пам'яті. Крім того, є інтерфейс програмування, а також можуть бути інтерфейси до віддалених пристроїв вводу/виводу і комунікаційних мереж. Живлення зазвичай реалізується за допомогою окремого модуля, як і ввід/вивід даних. Модулі вводу/виводу бувають дискретними та аналоговими, а також існують спеціалізовані модулі, наприклад, для управління переміщеннями або високошвидкісні лічильники. До модулів вводу/виводу підключаються віддалені пристрої. Узагальнена архітектура програмованих логічних контролерів наведена на рис.6.1.

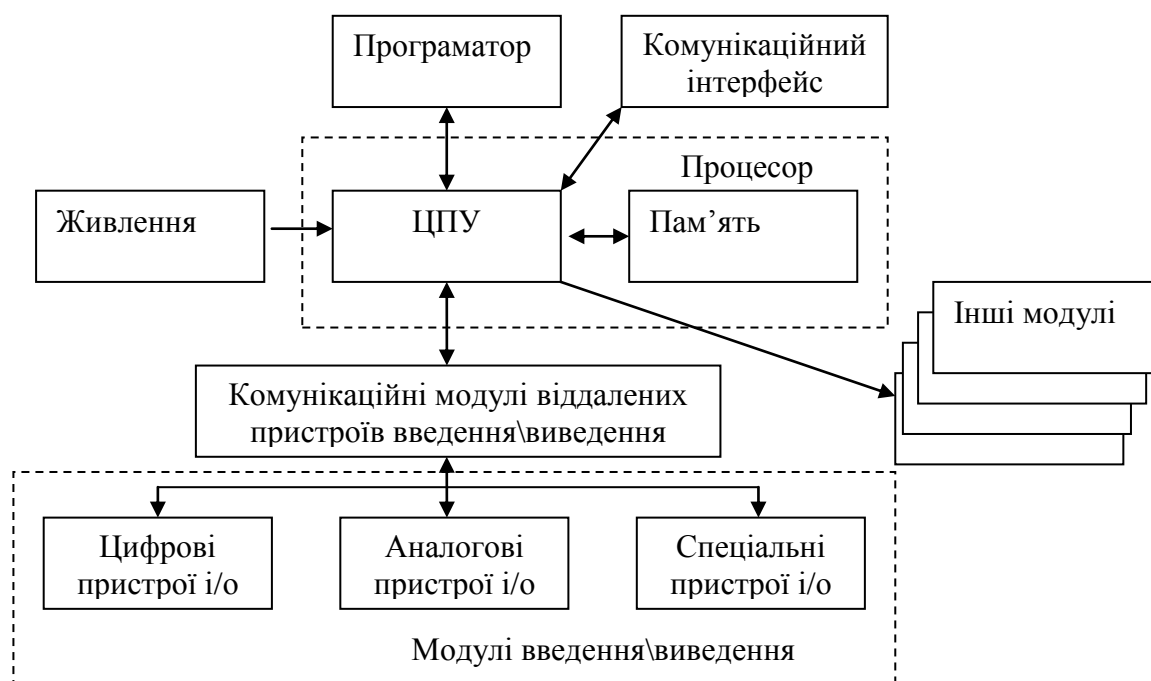


Рисунок 6.1 – Узагальнена архітектура програмованих логічних контролерів.

Залежно від кількості каналів вводу/виводу і конкретного ядра ПЛК, модулі вводу/виводу можуть бути на тому ж шасі, що і процесор, і/або на іншому або, навіть, кількох шасі. У самих компактних ПЛК, таких як мікро-ПЛК або нано-ПЛК, блок живлення, процесорний модуль, і всі пристрої вводу/виводу знаходяться в одному корпусі. Деякі мікро-ПЛК обладнуються вбудованою панеллю операторського інтерфейсу. Для більшості мікро-ПЛК кількість каналів вводу виводу обмежена і не підлягає розширенню.

У загальному випадку ПЛК – це одноплатний міні-комп'ютер, побудований на основі одно кристального мікропроцесора і розташований в типовому корпусі розмірами з жорсткий диск. Також існують модульні контролери. До входів ПЛК можна під'єднати перемикачі, контакти пристроїв позиціонування, органи управління, сенсори і виконавчі механізми (двигуни, лампи, нагрівальні елементи, клапани, вентилі, актюатори і т. і.). Програмований логічний контролер циклічно опитує входні сигнали (органи управління і сенсори), виконує програму користувача (перераховує значення змінних) і видає отримані вихідні дані на виконавчі механізми, тобто ПЛК циклічно, раз по раз виконує одну і ту ж програму (програму користувача).

Конструктивно ПЛК підрозділяються на моноблочні, модульні і розподілені (рис 6.2).

Моноблочні мають фіксований набір пристроїв вводу/виводу.

У модульних контролерах модулі пристроїв вводу/виводу встановлюються в різному складі і кількості в залежності від майбутньої завдання.

У розподілених системах модулі або навіть окремі пристрої вводу/виводу, що утворюють єдину систему управління, можуть бути рознесені на значні відстані.



Рисунок 6.2 – Серійні програмовані логічні контролери.

Компонентна база програмованих логічних контролерів. Напрямок діяльності виробника диктує вибір архітектури пристроїв. Деякі розробляють свою платформу на RISC-процесорах – цей підхід дуже популярний серед виробників, що мають невеликий обсяг продукції. Інші (Beckhoff) використовують платформи вбудованих промислових комп'ютерів, на яких встановлені операційні системи Windows CE, або Windows XP Embedded, при цьому PLC–додаток працює як служба.

Для організації зв'язку з оточенням, ПЛК необхідні пристрої вводу – виводу. Існують різні підходи щодо реалізації конструкції контролера. Існують конструкції «все в одному», коли головний контролер і пристрої вводу–виводу об'єднані в монолітні блоки, що мають набір різних конфігурацій. Так роблять, наприклад, Carel, і багато хто ще. На невеликому проекті такий підхід себе повністю виправдовує.

Більш гнучким є підхід, коли контролер знаходиться в головному блоці, а пристрої вводу–виводу монтуються окремо, як конструктивно-незалежні модулі (рис 6.3).



Рисунок 6.3 – Програмований логічний контролер Beckhoff серії CX9000 (в центрі) з набором модулів вводу-виводу (праворуч).

Програма ПЛК працює циклічно. Час циклу може тривати від 10^{-3} до 10 секунд, в залежності від завдань, які на цей ПЛК покладені. Більшість ПЛК дозволяють задавати час циклу розробнику програми, проте в деяких моделях такої можливості немає. Багато програмованих логічних контролерів, зокрема Beckhoff, дозволяють в одній програмі створити більше одного циклічно виконуваного завдання, і встановити пріоритет для цих завдань. Доцільність такого підходу розглянемо на прикладі.

Приклад. Що дає можливість виконання декількох задач? Уявімо ситуацію, коли ПЛК керує вентиляційною установкою, і до нього підключена панель управління через RS232. Температура в приміщеннях змінюється не швидко, і

запускати алгоритм управління вентиляцією частіше, ніж раз в 50...100 мс просто немає сенсу. Зате панель оператора опитує контролер постійно, і затримка відповіді ПЛК більше 10 мс вже відображається в гальмуванні інтерфейсу користувача, а при затримці 20 мс у нас переповниться апаратний буфер СОМ-порту. Наявність декількох завдань дозволяє нам вирішити цю проблему: нехай «швидке» завдання працює з СОМ-портом, і викликається кожні 2 мс, а «повільне», що реалізує логіку роботи вентиляції, викликається кожні 100 мс. Все працює добре, панель оператора не гальмує, користувач задоволений.

Крім апаратної уніфікації (використання стандартних розмірів, рівнів напруги, видів сигналів), поширенню ПЛК сприяло те, що для них були розроблені інтуїтивні «загально інженерні» мови програмування. Тепер для розробки програми користувача не обов'язково запрошувати програміста високого класу. З цим може впоратися (іноді – і краще) і технолог, і електрик, і хімік, і, звичайно, фахівець з автоматизації. А в разі складних завдань ці мови програмування стирають грань непорозуміння між програмістом і інженером. Вони однаково зрозумілі і замовнику (інженеру) і виконавцю (програмісту).

Таких мов програмування декілька, причому більшість з них – візуальні (об'єктно – орієнтовані). Програма в такому середовищі вводиться не у вигляді тексту, а як набір з'єднаних один з одним графічних елементів (рис. 6.4).

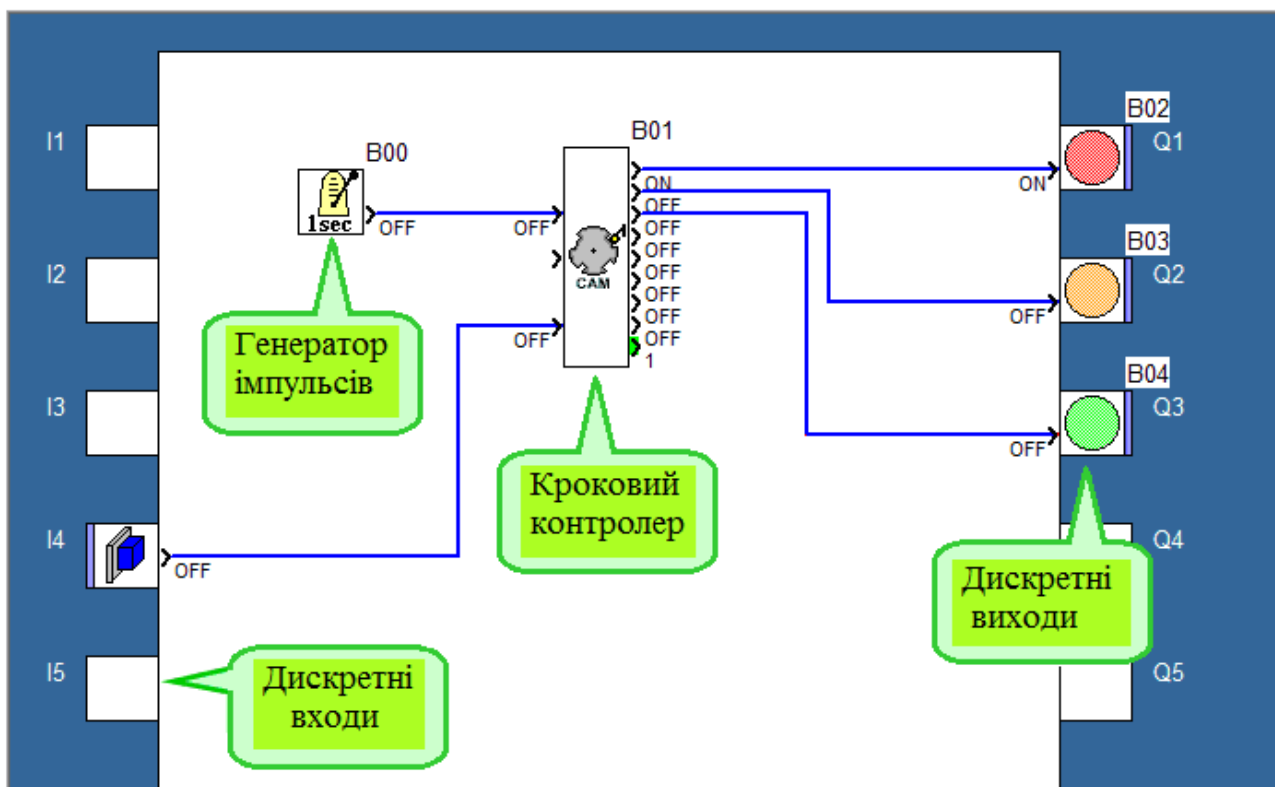


Рисунок 6.4 – Приклад програми для програмованого логічного контролера (графіка середовища розробки).

Програмна архітектура (ІЕС 61131-3). Зазвичай один і той же контролер можна програмувати декількома мовами на вибір користувача. Для цього використовують інструментальні програмні комплекси, що дозволяють не тільки розробити програму, а й налагодити її за допомогою програмної моделі контролера на симуляторі, або в режимі моніторингу (коли програму користувача виконує реальний контролер, а на дисплеї комп'ютера можна стежити за його роботою).

Апаратна і програмна уніфікація ПЛК дає можливість легко переходити на контролери іншого виробника, переносити програми з однієї платформи на іншу. Це підвищує гнучкість систем автоматизації, сприяє конкурентному інноваційного розвитку ринку.

Промислові ПЛК програмуються на мовах стандарту ІЕС 61131-3. Найпопулярніше, напевно, середовище розробки під ПЛК – це CoDeSys . Багато виробників беруть її за основу, і або роблять до неї бібліотеку для роботи зі своїм ПЛК, або доробляють середовище під себе.

Стандарт ІЕС 61131-3 задає програмну модель, відповідну сучасним принципам розробки програмних продуктів. Ця модель характеризується такими властивостями, як розробка «зверху-вниз», структуроване програмування, ієрархічна організація, програмні інтерфейси і інкапсуляція. На щастя, для того, щоб ефективно програмувати ПЛК, не потрібно тривалого навчання; хоча, звичайно, будучи повністю реалізованою, модель ІЕС 61131-3 вельми складна. Це основний недолік моделі, особливо по контрасту з простотою перших архітектур ПЛК.

Нижче описується програмна модель ІЕС 61131-3 (рис.6.5). Розглянемо основні елементи докладніше.

Конфігурація (configuration) – сукупність всього ПО і даних, що відносяться до системи ПЛК. У більшості випадків, конфігурація відноситься до програм і даних одного ПЛК. У складних системах, в яких взаємодіють багато різних ПЛК, у кожного є своя конфігурація. Конфігурація обмінюється даними з іншими конфігураціями ІЕС в системі автоматизації через інтерфейси, звані **шляхи доступу** (access paths). Використання слова «конфігурація» в цьому контексті призводить до термінологічного конфлікту, в зв'язку з початковим використанням цього терміна в галузі автоматизації яка відноситься до таких об'єктів як модель процесора в ПЛК, комунікаційні інтерфейси, віддалений ввід/вивід, розподіл пам'яті і т.д. Тому фірми, що виробляють ІЕС-сумісні ПЛК, використовують термін «конфігурація» в історичному значенні, і описують сукупність програм і параметрів терміном **проект** (project) .

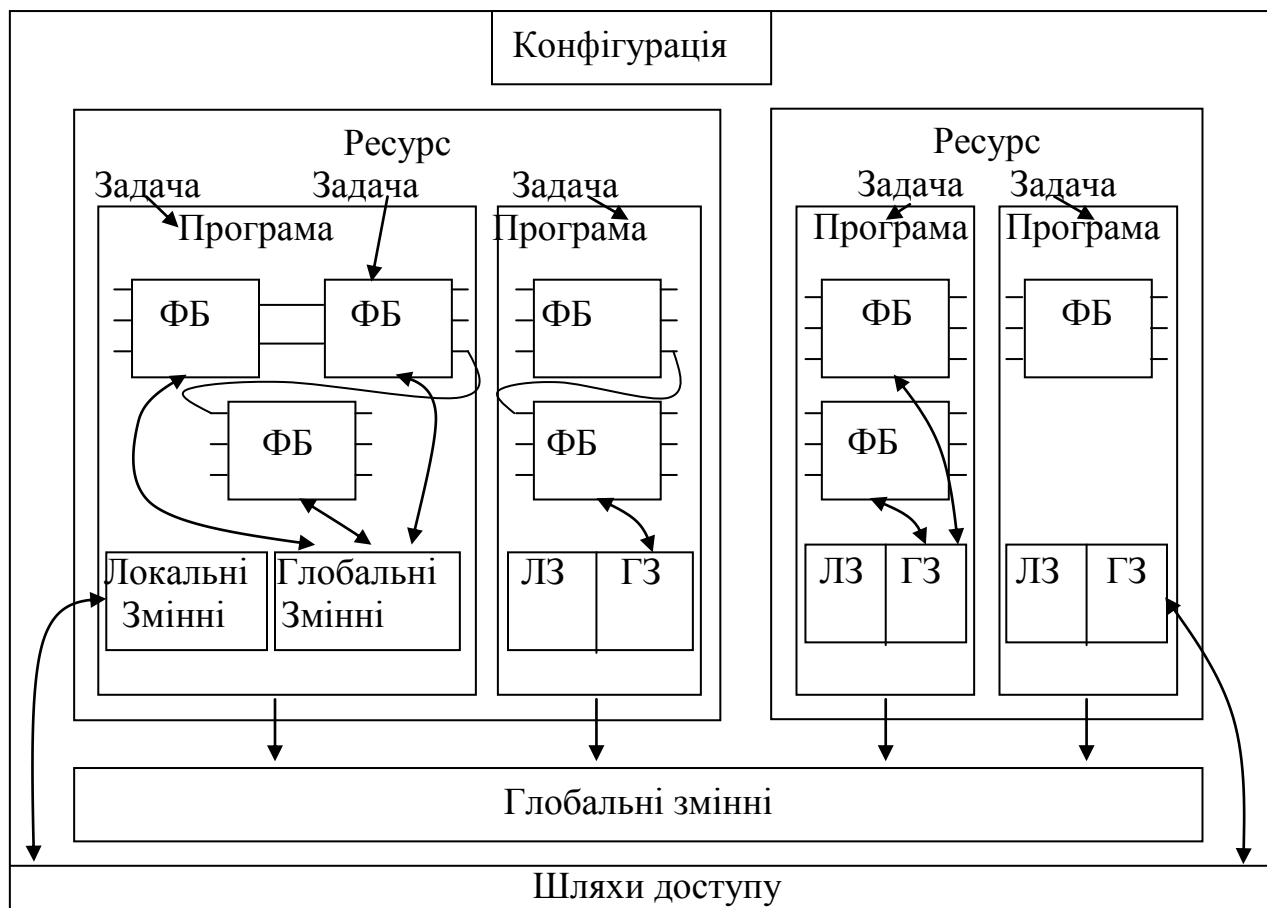


Рисунок 6.5 – Структура програмної моделі ІЕС 61131-3

Ресурс (resource) забезпечує виконання програм. Один або кілька ресурсів становлять конфігурацію. Зазвичай ресурс існує в межах ПЛК, але також він може існувати на ПК, припустимо, для програмного тестування. Одна з основних функцій ресурсу – забезпечувати інтерфейс між програмою і фізичними пристроями вводу/виводу ПЛК.

Програма (program) зазвичай складається з взаємозв'язаних **функціональних блоків** (function blocks), кожен з яких може бути написаний будь-якою мовою ІЕС. Функціональні блоки або програми також називаються **програмними модулями** (program organization unit). Крім функціональних блоків, в програмі декларуються фізичні вводи/виводи і локальні змінні програми. Програма може зчитувати/вести запис в канали вводу/виводу, глобальні змінні, і обмінюватися даними з іншими програмами. Шляхи доступу дозволяють конфігурації обмінюватися даними.

Завдання (tasks) керують виконанням програм або частин програм. Завдання управляє виконанням однієї або декількох програм і/або функціональних блоків. Виконання програми має на увазі, що всі функціональні блоки програми обробляються один раз. Виконання функціонального блоку має на увазі, що всі його програмні елементи виконуються один раз. Для виконання програми призначаються завдання, з налаштуванням для безперервного, періодичного виконання або виконання з тригера (події).

Змінні (Variables) задаються в різних програмних елементах моделі.

Локальна змінна (local variable) визначається програмним елементом і доступна тільки йому. Локальні змінні можуть бути задані для функціонального блоку, програми, ресурсу або конфігурації.

Глобальна змінна (global variable) задана для конфігурації, ресурсу або програми, доступна всім елементам, які в них містяться. Наприклад, глобальна змінна конфігурації доступна всім програмним елементам конфігурації. Глобальна змінна програми доступна всім функціональним блокам програми.

Мови програмування ПЛК.

За останнє десятиліття з'явилося декілька технологічних мов. Більш того, розроблений стандарт ІЕС 61131-3, який концентрує всі новітнє в області мов програмування для систем автоматизації технологічних процесів. Цей стандарт вимагає від різних виробників ПЛК пропонувати команди, які є однаковими і за зовнішнім виглядом, і за дією.

Стандарт специфікує 5 мов програмування:

- Sequential Function Chart (SFC) – мова послідовних функціональних блоків;
- Function Block Diagram (FBD) – мова функціональних блокових діаграм;
- Ladder Diagrams (LAD) – мова релейних діаграм;
- Statement List (STL) – мова структурованого тексту, мова високого рівня. Нагадує собою Паскаль
- Instruction List (IL) – мова інструкцій., Це типовий асемблер з акумулятором і переходах по мітці.

Мова LAD або KOP (з німецького Kontaktplan) схожі на електричні схеми релейної логіки. Тому інженерам який знає модерних мов програмування, не важко написати програму. Мова FBD нагадує створення схем на логічних елементах. У кожному з цих мов є свої мінуси і плюси. Тому при виборі фахівці орієнтуються в основному на особистий досвід. Хоча більшість програмних комплексів дають можливість бстрактувати вже написану програму з одного мови в інший. Так як деякі завдання витончено і просто вирішуються на одній мові, а на іншому доведеться зіткнутися з деякими труднощами

Найбільшого поширення в даний час отримали мови LAD, STL і FBD.

Більшість фірм виробники ПЛК традиційно мають власні фірмові напрацювання в області інструментального програмного забезпечення. Наприклад такі як Concept (Schneider Electric), Step 7 (Siemens).

Програмний комплекс CoDeSys.

Відкритість ІЕС стандартів привели до створення фірм, що займаються виключно інструментами програмування ПЛК.

Найбільшою популярністю в світі користуються комплекс CoDeSys, розроблений фірмою 3S. Це універсальний інструмент програмування контролерів на мовах МЕК, не прив'язаної до якої-небудь апаратній платформі і задовольняє всім сучасним вимогам.

Основні особливості:

- повноцінна реалізація ІЕС – мов,
- вбудований емулятор контролера дозволяє проводити налагодження проекту без апаратних засобів. Причому налагодження здійснюється не просто якогось абстрактного контролеру, а конкретного ПЛК з урахуванням апаратної платформи,
- вбудовані елементи візуалізації дають можливість створити модель об'єкта управління і проводити налагодження, тобто є можливість створення людино-машинного інтерфейсу (НМІ),
- дуже широкий набір сервісних функцій, що прискорює роботу програміста.

6.1 Програмування ПЛК

Особливості мови LAD в CodeSys. Як і було вказано вище, ПЛК здійснює циклічне читання входів, виконання прикладної програми і керування виходами. Тому написання програми для ПЛК відрізняється від традиційного написання програми для мікроконтролерів і ПК. До програм для ПЛК висуваються жорсткі вимоги щодо надійності, одна справа зависає текстовий редактор, а інша справа програма, що керує ядерним реактором. Інший не менш важлива вимога – це своєчасне реагування на подію. А що значить, вчасно не прореагувати на подію в промисловості? Це означає втратити контроль над технологічним процесом. Що в деяких випадках, наприклад з реактором, призведе до непоправних наслідків.

Розглянемо відмінності написання програми для ПЛК і мікроконтролера. Для прикладу візьмемо найпростішу задачу для МК – мигання світлодіоду.

Алгоритм буде наступним:

- Записати в порт лог. 1,
- Часова затримка,
- Записати в порт лог.0,
- Часова затримка,
- Перехід за міткою на початок програми.

За даним алгоритмом програма на ПЛК працювати не буде, вона містить нескінченний цикл. А в ПЛК вся прикладна програма виконується від початку до кінця в кожному робочому циклі, і будь-яка програма повинна віддавати управління системної програми. Тому при такій організації алгоритму наш ПЛК зависне. Навіть якщо і прибрати, перехід по мітці на початок, програма не буде працювати, так як нам хочеться. Порт завжди буде в стані лог.0, так як фізична установка виходів проводиться тільки після виконання всієї прикладної програми. І тому проміжні стану це всього лише програмні змінні в пам'яті, і на апаратній частині вона ніяк не відображаються.

На додаток затримку часу теж потрібно організувати за допомогою таймера, періодично перевіряючи його значення, а не очікувати в порожню поки цей час пройде, напевно для контролера знайдеться інша більш важлива робота.

З урахуванням вище сказаного, правильний алгоритм буде виглядати наступним чином:

– Перевірити таймер. Якщо час паузи вийшов, то:

- а) інвертувати вихід,
- б) почати новий відлік,

– Кінець програми.

Реалізуємо даний алгоритм на практиці в прикладі.

Приклад. Створимо проект для генератора одиночних імпульсів. Після запуску програми CoDeSys, вибираємо пункт меню «Створити новий проект». CoDeSys запропонує вибрати цільову платформу для ПЛК. Вказати цільову платформу необхідно, щоб середовище знало, для якого типу контролера пишеться програма. Вибрати 3S CodeSys Sp PLCWinNT V2.4, ввести – ОК (рис.6.6).

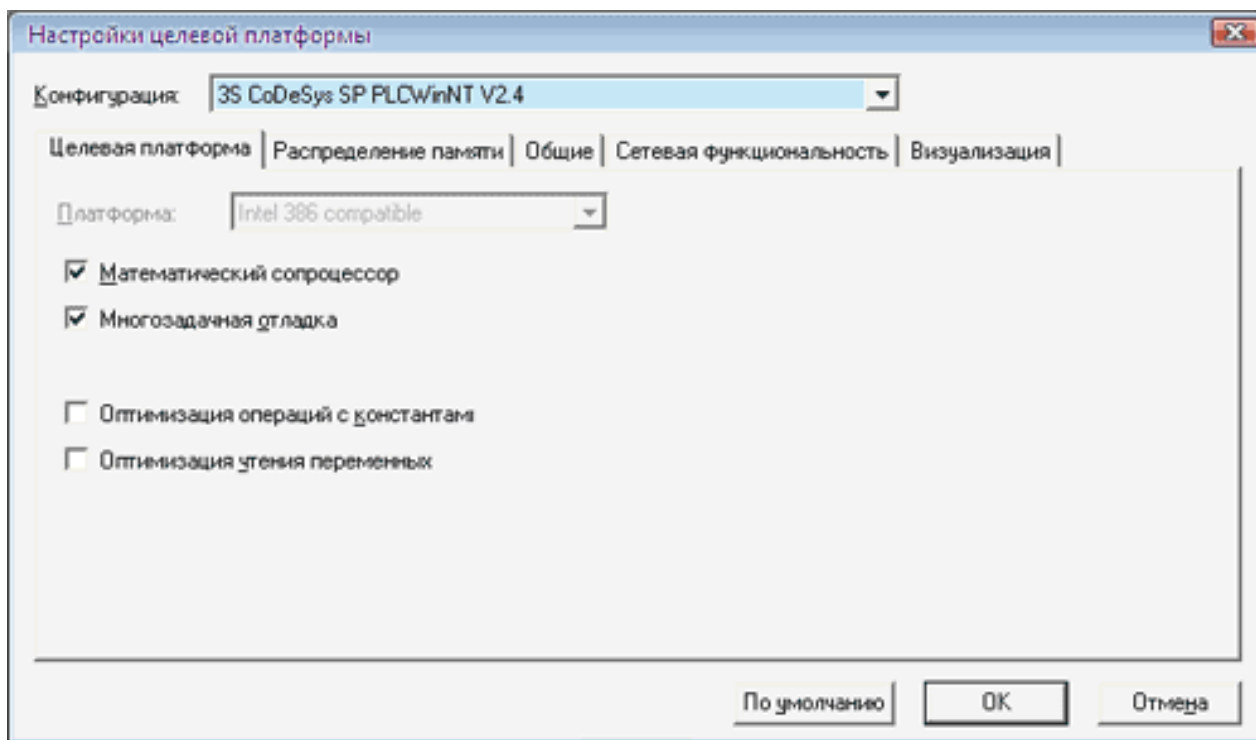


Рисунок 6.6 – Вибір цільової платформи для ПЛК

Після цього вибираємо ім'я проекту і мову програмування (рис.6.7).

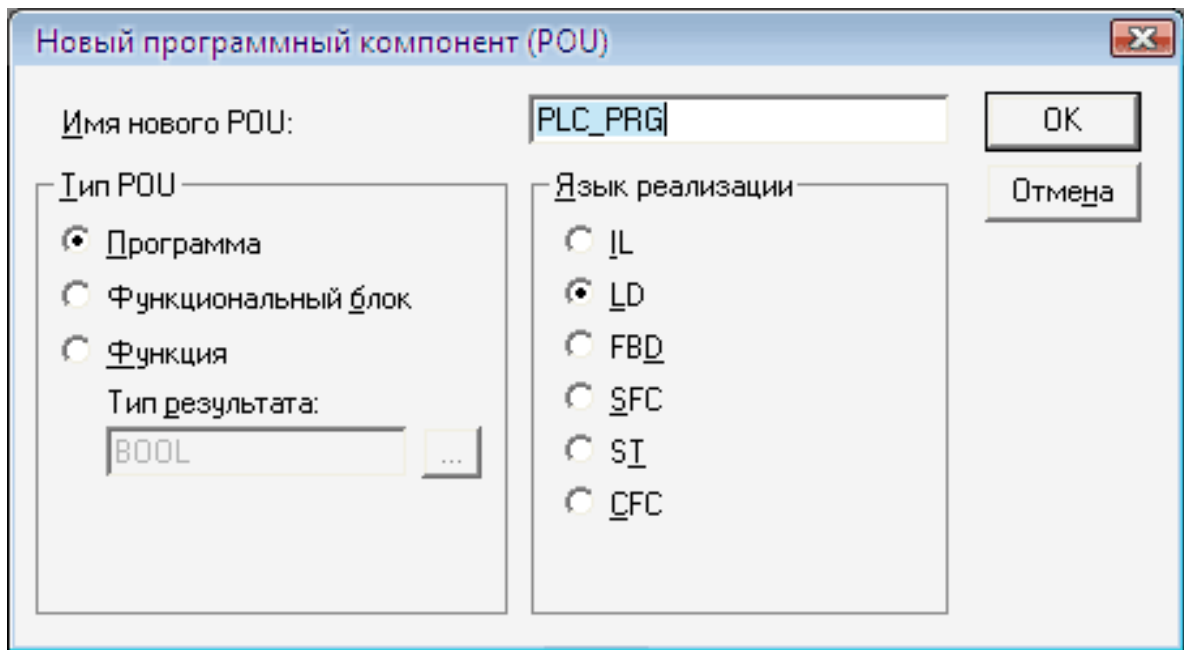


Рисунок 6.7 – Ім'я проекту в прикладі залишаємо за замовчуванням, мову вибираємо LD

Інтерфейс програми інтуїтивно зрозумілий. При наведенні на елемент спливає його ім'я. Для більш глибокого розуміння та вивчення можливостей середовища розробки, рекомендується переглянути всі типи елементів програмного меню POU, а також склад головного меню.

Щоб додати елемент в програму необхідно лівою кнопкою миші позначити точку на робочому полі програми а потім вибрати необхідний елемент меню і перемістити у вікно програми, наприклад, нормально розімкнутий контакт; повинно вийти наступне (рис.6.8).

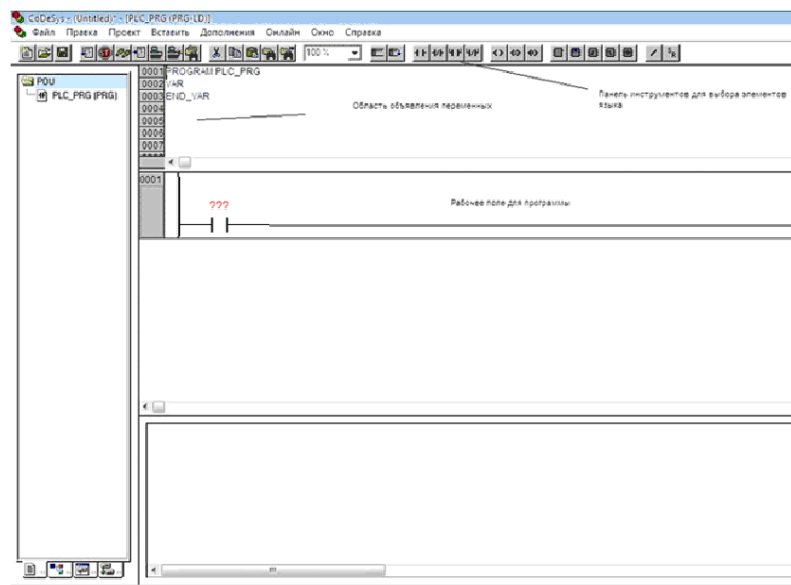


Рисунок 6.8 – Додавання елементів програми

Замість знаків питання пишемо ім'я нашої змінної, наприклад SB, і натискаємо Enter, виходить вікно оголошення змінної, вибираємо Bool і натискаємо OK (рис.6.9).

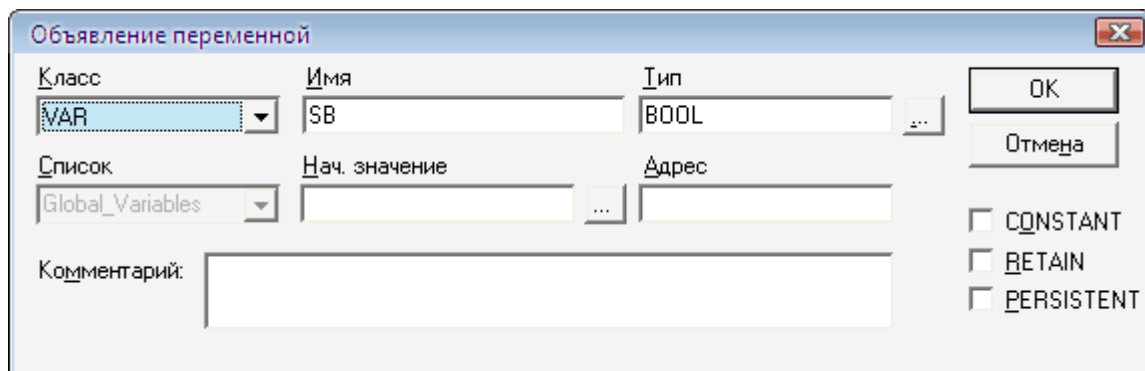


Рисунок 6.9 – Введення імені змінної

Доцільно ознайомитись з типами елементів, які можна вибрати, а також з існуючими класами змінних.

Для безпосередньої реалізації програми для генератора одиночних імпульсів використовуємо функціональний блок таймеру TP. Таймер TP – це таймер одиночного імпульсу із заданою по входу PT тривалістю.

Поки вхід IN дорівнює FALSE, вихід Q = FALSE, вихід ET = 0. При переході IN в TRUE вихід Q встановлюється в TRUE і таймер починає відлік часу на виході ET до досягнення тривалості, заданої PT. Далі лічильник не збільшується. Таким чином, вихід Q генерує імпульс тривалістю PT по фронту входу IN (рис.6.10).

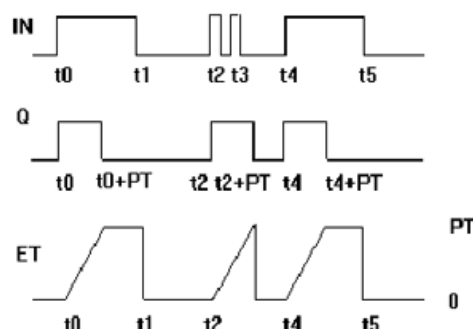


Рисунок 6.10 – Часова діаграма роботи TP.

Для вставки таймеру TP, на панелі елементів вибираємо його позначення (рис.6.11).



Рисунок 6.11 – Вибір позначення таймеру на панелі інструментів.

При цьому на робочому полі спливає асистент вибору функціонального блоку (рис.6.12).

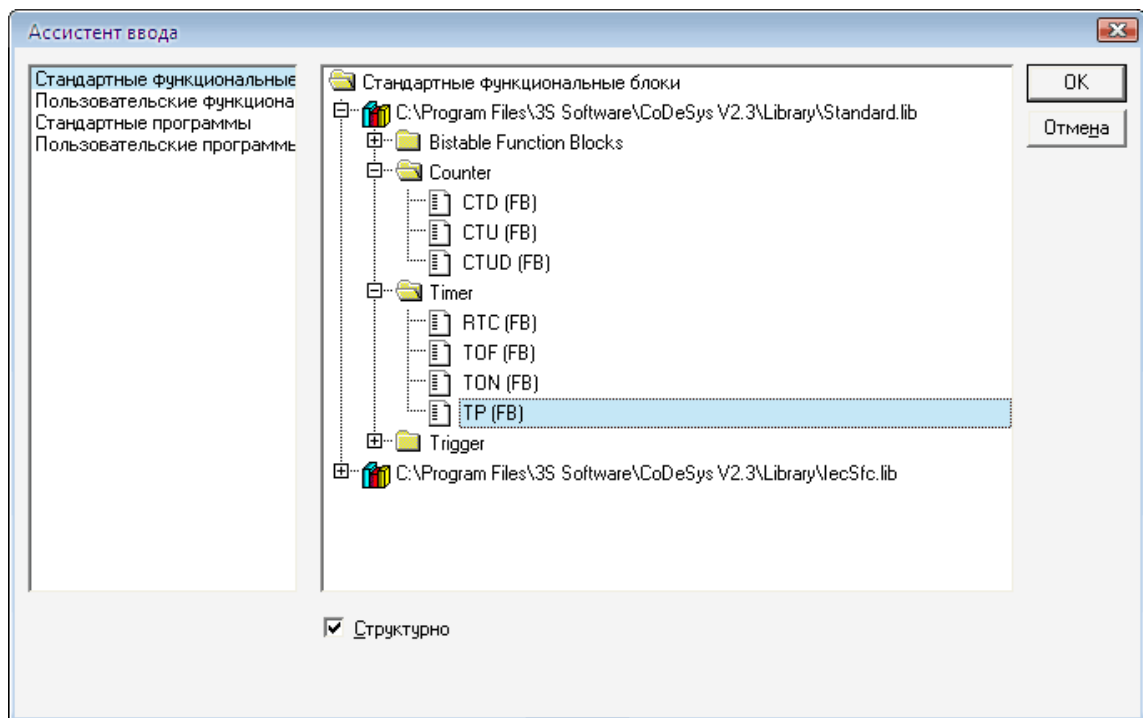


Рисунок 6.12 – Вибір функціонального блоку за допомогою асистенту.

Розглянемо як працює наш проект у вікні програми. У початковий момент $X = \text{False}$, тому інверсний контакт X замкнутий і таймер $T2$ запущений, вихід $Q = \text{True}$, тому ланцюг включена. А так як обмотка в ланцюзі інверсна, значить вона копіює інверсне стан ланцюга в X , і X залишається False , після переповнення таймера $Q = \text{False}$, і інверсна обмотка переводить X в True . Після цього запускається $T1$, після переповнення скидає X в False і все повторюється. Змінна X є виходом генератора. Таймер $T2$ встановлює паузу, а $T1$ тривалість імпульсу (рис.6.13).

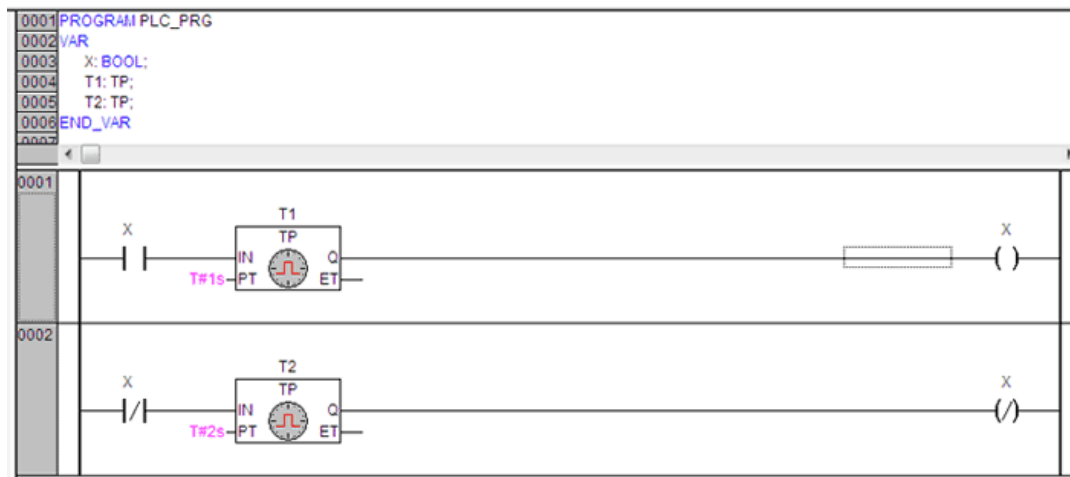


Рисунок 6.13 – Діаграма роботи генератора одиночних імпульсів.

Наступний крок – компіляція проекту **Проект -> Компілювати**.

У пункті онлайн вибираємо **Режим емуляції**, а потім **Підключення і Старт**. В робочому режимі емуляції, зібрана схема починає роботу в режимі перемикавання, ланцюг де «протікає струм» виділяється синім кольором. Також в області оголошення змінних відбивається поточне значення змінних.

Вихід генератора можна подивитися за допомогою цифрового трасувальника, для цього переходимо на вкладку **Ресурси** в нижньому лівому кутку (рис.6.14).



Рисунок 6.14 – Вибір цифрового трасувальника

Вибираємо **Цифровий трасувальник -> Доповнення -> Налаштування трасування**, вийде наступне вікно (рис.6.15).

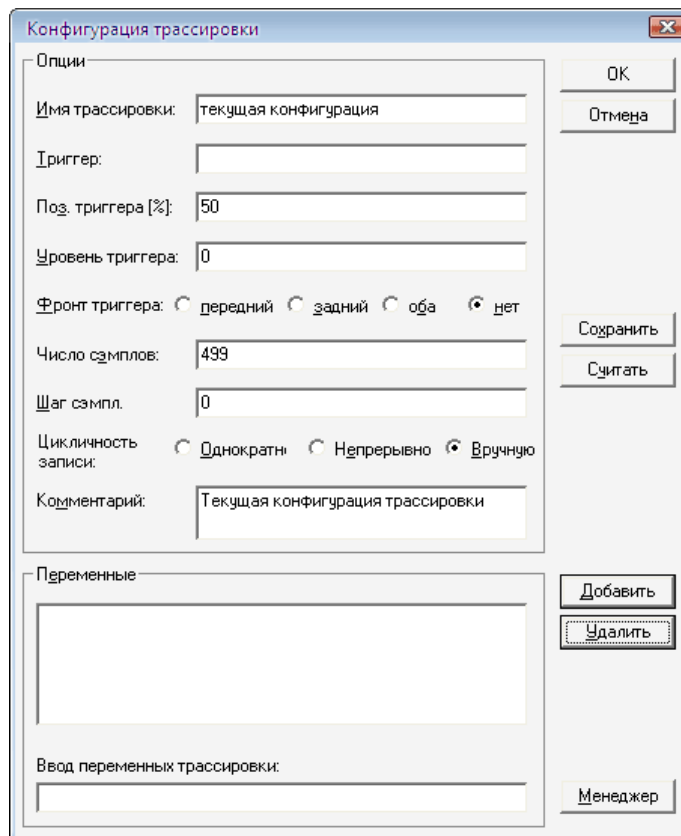


Рисунок 6.15 – Встановлення опцій цифрового трасувальника

Циклічність запису встановлюємо вручну. У вікні асистенту вводу вибираємо змінні X(Bool) (рис.6.16). Натискаємо **Ok**.

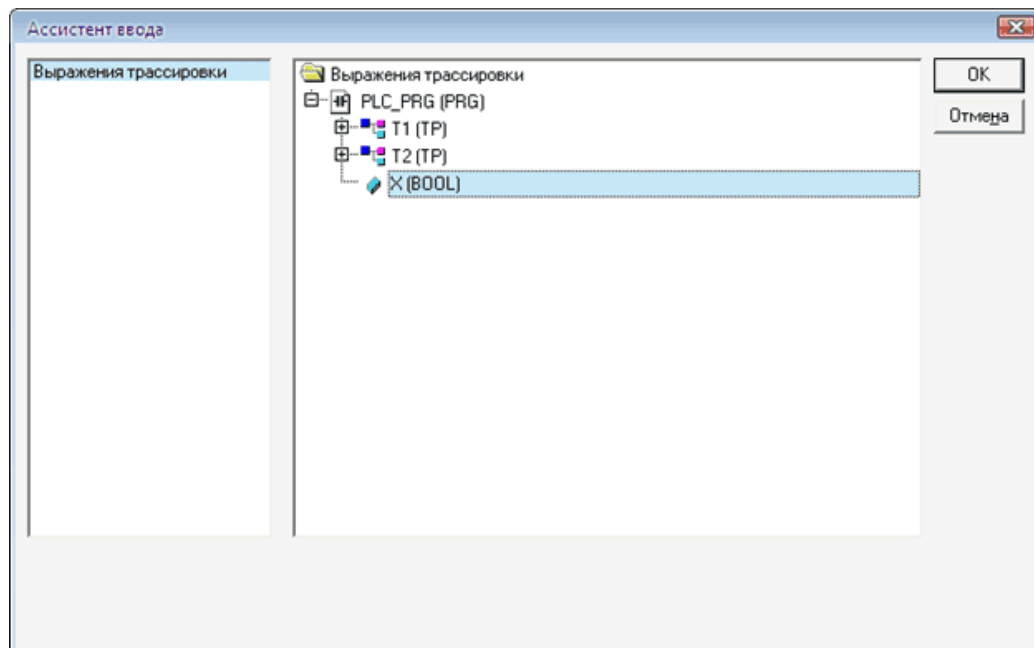


Рисунок 6.16 – Вибір змінні X(Bool) у вікні асистенту вводу

Вибираємо інструмент рисування для візуалізації змінної. Встановлюємо режим онлайн **Підключення**, натискаємо **Старт**, вибираємо пункт **Автоматичне трасування**, далі **Додатково-> Почати трасування** (рис.6.17).

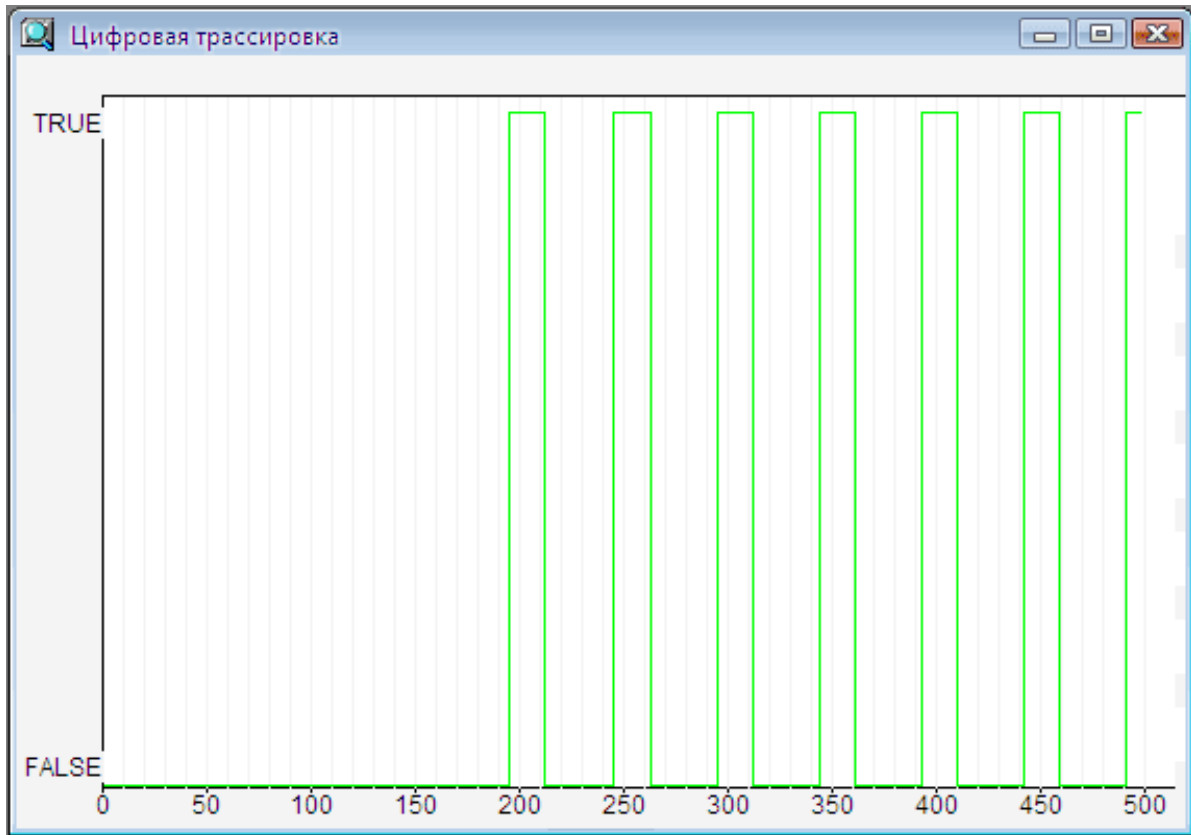


Рисунок 6.17 – Візуалізація роботи генератора одиночних імпульсів за допомогою цифрового трасувальника.

Доповнення. У скороченому вигляді розглянемо ще один приклад – управління двигуном з електронною комутацією обмоток статора. Алгоритм роботи пристрою наступний.

Всі таймери запускаються за сигналом старт. Кожен таймер відміряє момент початку і закінчення фази. Змінні Y1-Y3 є виводами відповідної фази управління. Кожен вихід включається в тому випадку, якщо таймер ще не переповнений і вимкнений попередній вихід. Останній ланцюг, є ланцюгом автоматичного перезапуску циклу роботи керуючого пристрою (рис.6.18).

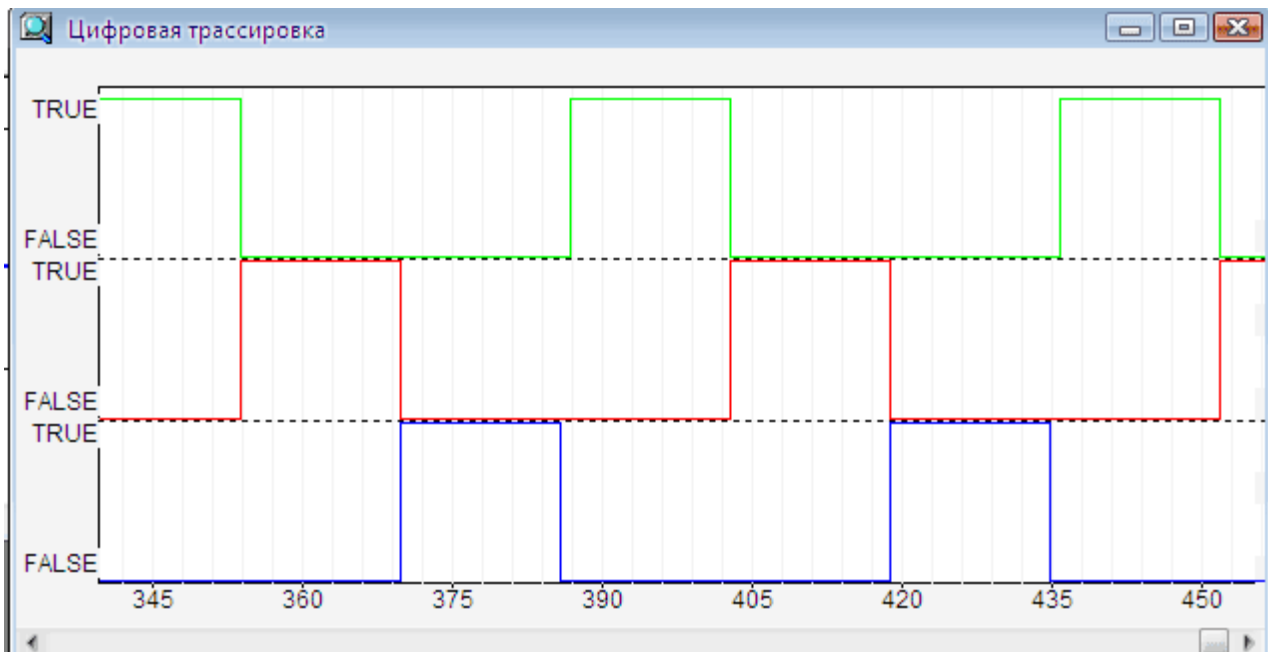


Рисунок 6.18 – Візуалізація роботи пристрою автоматичного управління двигуном з електронною комутацією обмоток статора за допомогою цифрового трасувальника.

6.2 Огляд програмованих логічних контролерів фірми SIEMENS.

У цьому розділі дається короткий опис лінійок ПЛК з описом конструктивних особливостей, основних функціональних можливостей, обмежень, засобів програмування і налагодження.

Лінійка нижнього рівня ПЛК представлена логічними контролерами LOGO (рис.6.19). Вони підходять для створення засобів найпростішої автоматизації та заміни будь-яких таймерів, реле часу і т.п. Більш того для цієї серії передбачено розширення за рахунок додання різноманітних модулів.



Рисунок 6.19 – Контролер LOGO доданими модулями вводу/виводу.

Існує обмеження на розширення системи. Так загальна кількість дискретних пристроїв вводу системи (модуля ПЛК і модулів розширення) не може перевищувати 24 і число дискретних виходів не може перевищувати 16. Додаткові модулі можуть мати дискретні пристрої вводу/виводу (як з транзисторним

ключем на виході (струм до 0,3 А) так і релейним для комутації сигналів зі струмом до 10А, аналогові пристрої мають вихідний струм 0...10 або 4...20мА. При цьому деякі модулі мають уже закладену характеристику для терморезисторів типу Pt100 і Pt1000, що дозволяє мати в системі вимірювач температури для можливості побудови доволі динамічно функціональної системи. Кількість аналогових входів і виходів також обмежені. У системі може бути не більше 8 аналогових входів і 2 аналогових виходів. Подальше розширення можливо при використанні центрального блоку ПЛК типу LOGO! (рис 6.20). Але і в цьому випадку система буде мати обмеження. Максимальне число входних дискретних сигналів в системі не може перевищувати 64, вихідних 64, аналогових входів 32, аналогових виходів 16. При цьому важливо пам'ятати, що програму може виконувати тільки один ПЛК, що працює в режимі Master, а решта працюючи в режимі Slave забезпечують взаємодію Master з модулями розширення ПЛК.

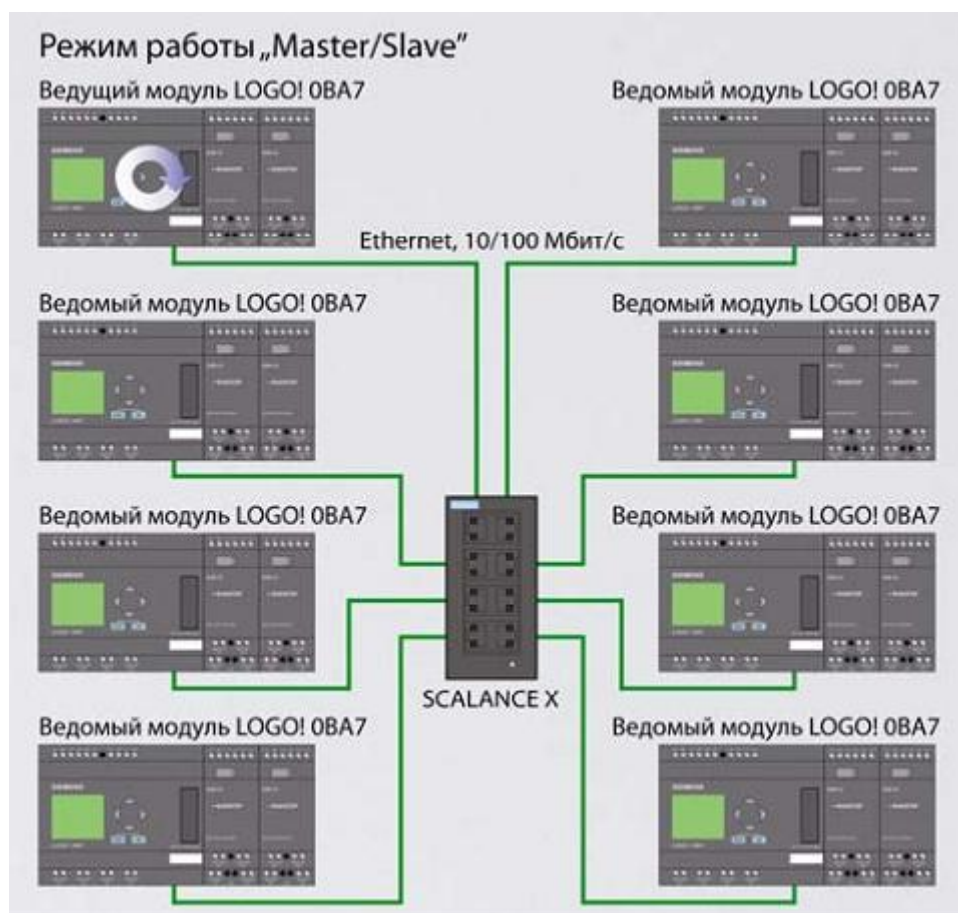


Рисунок 6.20 – Розширення системи керування з використанням центрального блоку ПЛК типу LOGO!

У лінійці LOGO є такі типи центральних ПЛК (рис. 6.21):

- LOGO BASIC тип 0BA6 – містять в своєму складі РК екран, що дозволяє відображати будь-які дані з програми користувача і здійснювати програмування модуля,
- LOGO PURE тип 0BA6 – спрощений варіант, що не має екрану,

– LOGO BASIC тип 0BA7 – містять в своєму складі РК екран, що дозволяє відображати будь-які дані з програми користувача і здійснювати програмування модуля. Модулі типу 0BA7 мають додаткові поліпшення у вигляді виконання призначених для користувача функцій, більшого числа аналогових і дискретних прапорів і т.д.



Рисунок 6.21 – Типи центральних ПЛК LOGO.

Взаємодія з користувачем.

У системах на базі модулів LOGO НМІ інтерфейс (взаємодія з користувачем) може бути реалізований в наступний спосіб. У модулях LOGO BASIC за рахунок наявного РК екрану дозволяє відображати до 4-х рядків буквено-цифрових рядків до 12 символів на рядок (на різних мовах включаючи російську). Підсвічування екрану регулюється з програми.



Рисунок 6.22 – Зовнішній екран LOGO TD

Зовнішній екран LOGO TD має можливість виводити 4 рядки по 24 символу на рядок, до 4-х бар графіків. Також доступні 10 сенсорних кнопок управління (рис.6.22).

Програмування модулів LOGO.

Програмування з екрану модуля здійснюється на мові FBD. Процес програмування полягає у виборі необхідної функції з бібліотеки, визначенні прив'язок до входів виходів або інших функцій, а також установки параметрів

функції при необхідності. Даний вид програмування дуже складний і вимагає великої уваги. Його використання доцільне скоріше для коригування існуючої програми ніж для набору програми з нуля (за винятком лише дуже простих програм).

Програмування за допомогою карти пам'яті полягає в тому, що програма (заздалегідь створена) спочатку записується на карту пам'яті, що вставляється в паз для модуля пам'яті в блоці LOGO. Після подачі живлення на модуль LOGO PURE, програма автоматично копіюється в пам'ять модуля і починає виконуватися. У модулях типу LOGO BASIC на екран виводиться повідомлення про необхідність скопіювати програму у внутрішню пам'ять. Для масового програмування модулів пам'яті може бути використаний спеціальний програма тор (рис. 6.23).



Рисунок 6.23 – Модулі пам'яті для LOGO тип 0BA6. Для LOGO тип 0BA7 використовується стандартна карта SD.

Використовується спеціальна програма, що встановлюється на PC і потім записується в модуль пам'яті, або безпосередньо в блок контролера. Програма підтримує набір і симуляцію програми модуля (рис. 6.24).

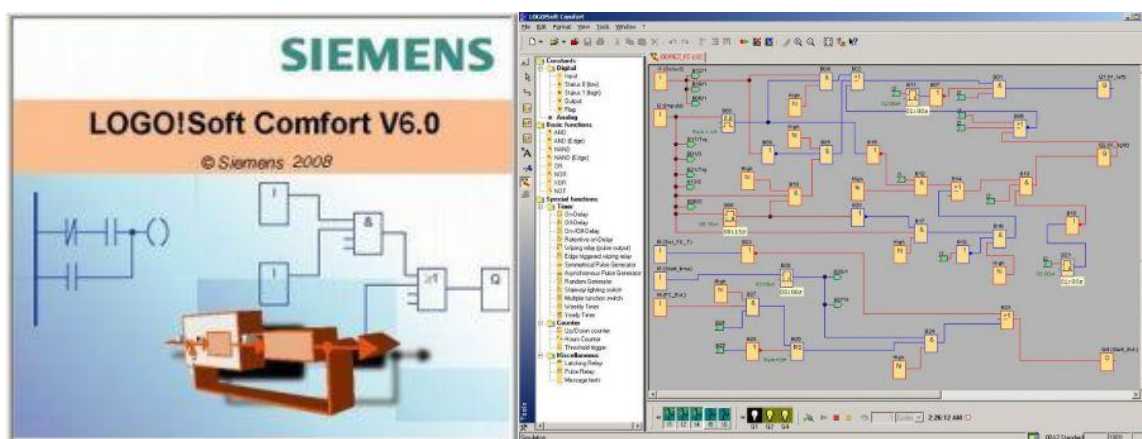


Рисунок 6.24 – Засоби програмування LOGO SOFT COMFORT.

Лінійка початкового рівня ПЛК представлена контролерами SIEMENS S7-200. Це вже цілком адекватні ПЛК дозволяють реалізувати автоматичні сис-

теми високої складності. Природно передбачено розширення за рахунок різних модулів. Також існує обмеження на розширення системи.

Модулі розширення мають в своїй номенклатурі модулі роботи з терморезисторами різних типів, термopаpами, модулі позиціонування, ваговимірювальні модулі, різні комунікаційні модулі (рис.6.25).

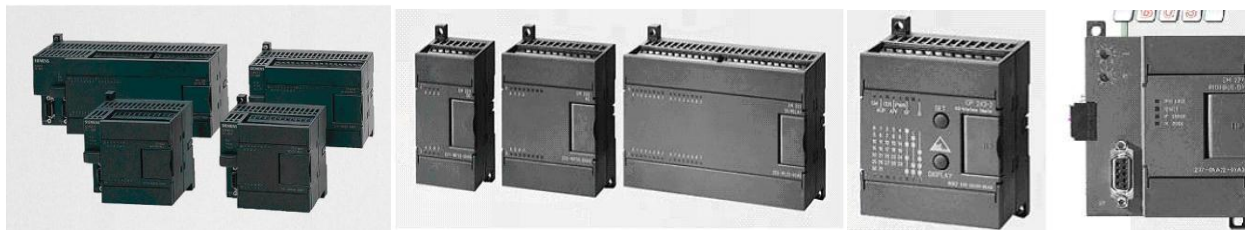


Рисунок 6.25 – Головні ПЛК, модулі вводу/виводу, комунікаційні модулі

Подальше розширення можливо при використанні вбудованого інтерфейсу PPI/MPI. При організації мережі PPI може бути використано до 32 станцій.

Також є модулі типу SIPLUS S7-200 працюють в розширеному температурному діапазоні і високої вологості.

Взаємодія з користувачем.

У системах на базі S7-200 HMI інтерфейс (взаємодія з користувачем) може бути реалізований засобами програм Protool або WinCC (рис.6.26).

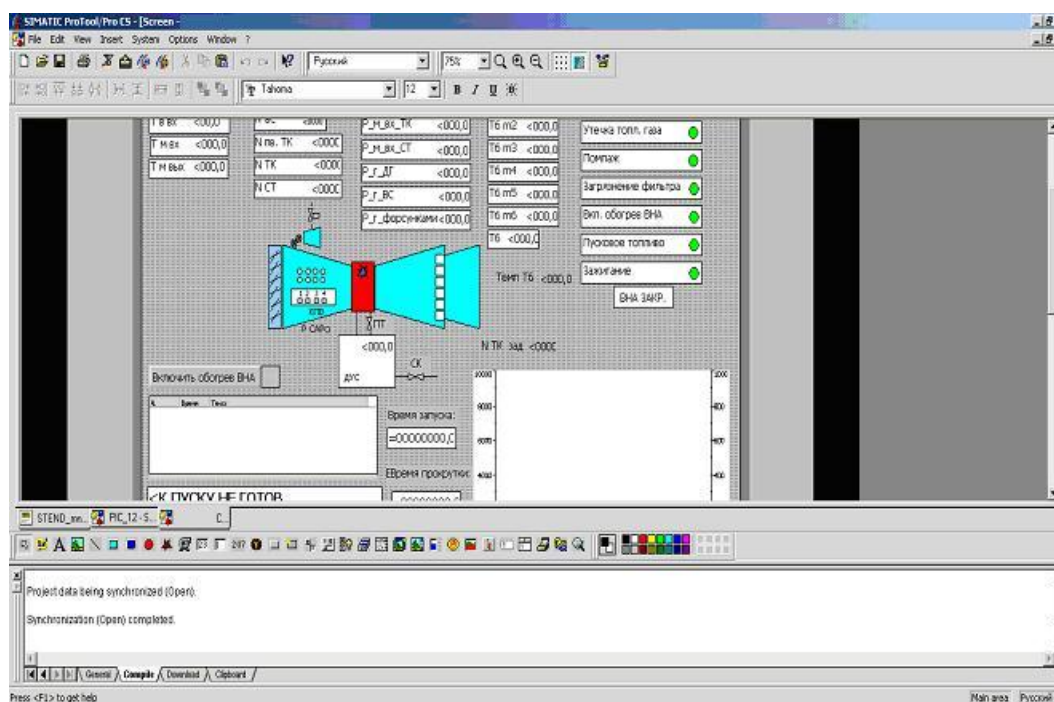


Рисунок 6.26 – HMI інтерфейс на PC: реалізується засобами програм Protool або WinCC.

Програмування завжди ведеться з програми STEP 7 MicroWin, відрізняються тільки способи завантаження програми в ПЛК.

Завантаження здійснюється через кабель PC/PPI, або через Industrial Ethernet при наявності в системі комунікаційного модуля CP243-1, або через модемний зв'язок при наявності в системі комунікаційного модуля CP241.

При програмуванні використовуються мови LAD, STL або FBD. Також в MicroWin здійснюється налаштування панелей оператора TD200 і TD400 і комунікаційних модулів (рис. 6.27).

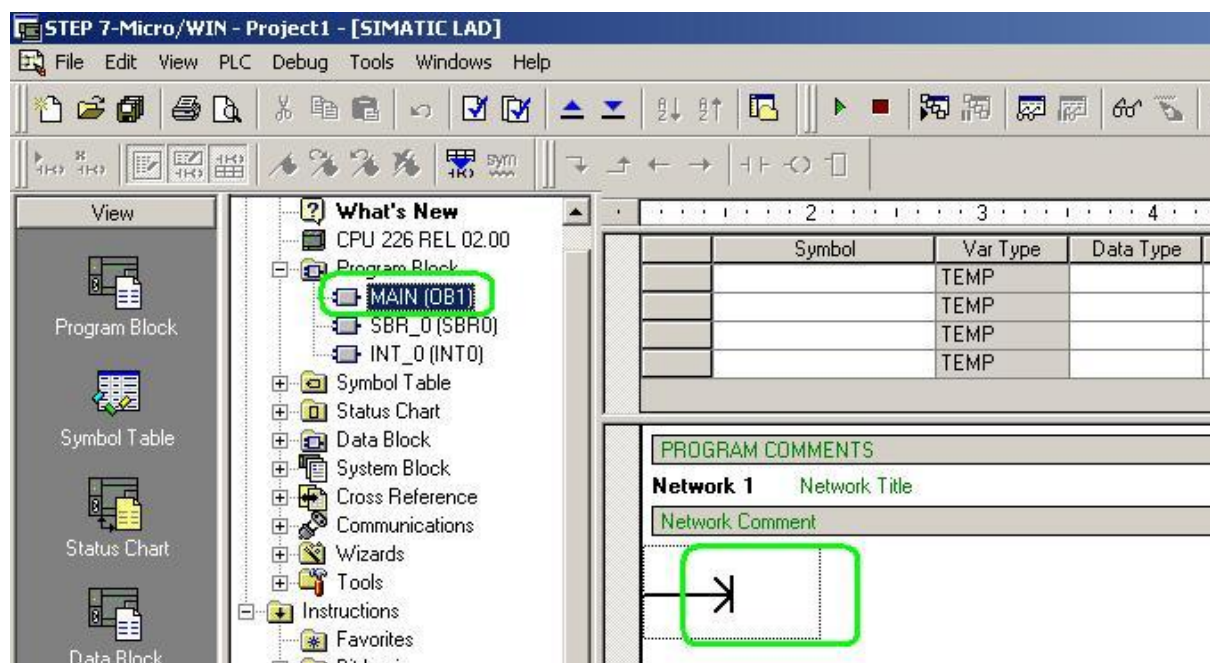


Рисунок 6.27 – Налаштування панелей оператора TD200 і TD400 і комунікаційних модулів в пакеті STEP 7 MicroWin.

Пакет STEP 7 MicroWin має безліч вбудованих бібліотек і майстрів налаштування модулів.

Необхідно відзначити, що на даний момент SIEMENS рекомендує для заміни ПЛК SIMATIC S7-200 модулі S7-1200. SIMATIC S7-1200 мають більші можливості. До них можна віднести: менший розмір корпусів, більший розмір пам'яті, більше число підтримуваних входів в системі і меншу ціну.

Для написання програми використовується пакет STEP 7 BASIC в якій інтегровано засоби написання і налагодження програм для ПЛК, а також пакет WinCC Basic для створення HMI на базі панелей KT і KTR.

Однак, STEP 7 BASIC v10 не підтримує мову програмування STL. Таким чином, ПЛК S7-200 і S7-1200 найзручніше програмувати в мові графічного образу типу LAD або FBD. Складні системи, що реалізуються на ПЛК S7-300 S7-400 і можуть бути запрограмовані на STL.

Контролери SIEMENS S7-300. Модульні програмовані контролери для вирішення завдань автоматизації середнього рівня складності (рис. 6.28).



Рисунок 6.28 – Панель головного контролера S7-300 з комунікаційними модулями

S7-300 є найбільш потужним і найбільш затребуваним представником ПЛК фірми SIEMENS. Істотні можливості для розширення до 32 модулів (4 ряди по 8 модулів в кожному) дозволяють обслуговувати величезну кількість сигналів (до 8200 дискретних і 512 аналогових). Набір вбудованих технологічних функцій дозволяє вирішувати завдання швидкісних розрахунків, вимірювання частоти або тривалості періоду, ПІД-регулювання, позиціонування, переведення частини дискретних виходів в імпульсний режим.

SIMATIC S7-400 – модульний програмований контролер для вирішення складних завдань автоматичного управління.

У системах S7-400 можлива заміна модулів розширення без виключення контролера за рахунок розташування шини безпосередньо в стійці. Також на ПЛК S7-400 можлива побудова так званих H - систем (систем подвійного резервування) мають два CPU програма в яких виконується паралельно і перемикання між керуючими контролерами відбувається без переривання технологічного процесу. Програмування також здійснюється з пакета STEP 7

На остаток флагманський комплекс SIMATIC S7-1500 від SIEMENS (рис.6.29).



Рисунок 6.29 – SIMATIC S7-1500 – модульний програмований контролер останнього покоління.

SIMATIC S7-1500 має розширені можливості за рахунок збільшеної швидкодії і величезної кількості пам'яті. Час виконання логічної операції 1 нс. Швидкість обміну по внутрішній шині до 400 Мб/с. Також змінилася якість програмою підтримки ПЛК. Тепер в програмі зберігаються і символьні імена і коментарі, підтримка 64-бітових даних.

6.3 Контрольні запитання і завдання

1. Основні функції SCADA.
2. Архітектура програмованих логічних контролерів.
3. Конструкції ПЛК.
4. Компонентна база програмованих логічних контролерів.
5. Програмна архітектура ПЛК (IEC 61131-3).
6. Поняття конфігурації та ресурсів.
7. Мови програмування ПЛК.
8. Основні особливості CoDeSys.
9. Особливості програмування ПЛК.
10. Лінійка контролерів LOGO. Їх характеристики.
11. Існуючі обмеження на розширення системи.
12. Відмінності лінійки LOGO!
13. Типи центральних ПЛК LOGO.
14. Варіанти взаємодії контролера з користувачем.
15. Програмування модулів LOGO.
16. Модульні програмовані контролери для вирішення завдань автоматизації середнього рівня складності.
17. Характеристики SIMATIC S7-1500.

7. ПРОГРАМОВАНІ СИСТЕМИ НА КРИСТАЛІ В СИСТЕМАХ УПРАВЛІННЯ

7.1 Мікросхеми типу система на кристалі фірми Cypress першої генерації

Стрімкий розвиток технології виробництва інтегральних мікросхем призвів до того, що до ВІС мікроконтролерів почали включати не тільки АЦП та ЦАП, але багато інших додаткових периферійних пристроїв: програмовані аналогові підсилювачі, компаратори, лічильники - таймери з достатньо гнучким програмуванням, різноманітні послідовні порти введення/виведення, контролери світлодіодних та рідкокристалічних індикаторів та панелей і тощо. Такі ВІС може віднести до більш - менш повних аналого цифрових систем на кристалі. Проте вони не мали однієї важливої для проектування будь-якої системи особливості – можливості налаштування наявних аналогових і цифрових блоків в бажаній структурі та встановлювати їх функції на свій розсуд.

В 2001 – 2003 роки фірма Cypress Semiconductor Corp. оголосила про нове сімейство аналого-цифрових програмованих систем на кристалі (СНК) під торговою маркою PSoC (Programmable System on Chip), що містять базові компоненти цифрових мікроконтролерів (з додаванням пристроїв, які часто використовуються в цифрових процесорних системах управління), і достатньо ефективні набори реконфігуруваних цифрових і аналогових блоків. Аналогові PSoC блоки можуть бути використані дуже гнучке, оскільки, по-перше, мають широкий спектр налаштування конфігурації (фільтри, підсилювачі та компаратори, ЦАП та АЦП), а по-друге, їх можна раціональніше з'єднати між собою за допомогою дуже ефективної матриці перемикачів. Втім, у якості цифрових реконфігуруваних блоків застосовуються 8-розрядні функціонально закінчені пристрої, а не набори програмованих логічних вентилів.

Однак, такі пристрої, реалізовані на основі лічильників і порозрядних суматорів (по модулю), значно зменшили складність і спростили конфігурації налаштування цифрової частини SOC. Як наслідок, архітектура та системо- та схемотехнічні рішення PSoC виявилися досить успішними.

Мікросхеми першої генерації PSoC Cypress в основі своєї архітектури мали вбудовані апаратні ядра:

- центральний процесор;
- пам'ять;
- програмовані аналогові і цифрові блоки.

«Аналогова підсистема» (матриця конфігуруваних аналогових блоків) складається з конфігуруваних блоків на базі матриці з програмованими резисторами (Continuous Time, СТ) і звичайного операційного підсилювача та конфігуруваних аналогових блоків на основі схем на комутованих конденсаторах і операційному підсилювачі (SC).

Аналогові блоки типу дозволяють реалізувати:

- програмовані підсилювачі;

- програмовані компаратори;
- програмовані фільтри нижніх і верхніх частот 2-го порядку;
- програмовані смугові фільтри;
- пристроїв вибирання – збереження;
- АЦП різних типів;
- цифро-аналоговий перетворювач (ЦАП).

Конфігурування, програмування мікросхем PSoC та розробка на їх основі систем відбувалися у середовищі проектування PSoCDesigner. Середовище розробки дозволяє користувачу розробляти структурні і принципові схеми системи, а також основні програми її функціонування. Для спрощення цього процесу застосовуються модулі користувача – це готові програми, що знаходяться у бібліотеці оболонки PSoC Designer. Кожен такий модуль конфігурує внутрішні ресурси кристала для виконання тих або інших функцій. Кількість модулів які одночасно можна розмістити в одній мікросхемі PSoC залежить від числа займаних ними цифрових і аналогових блоків

На базі мікросхем PSoC було розроблено багато різноманітних пристроїв, які було наведено на сайті фірми Cypress Semiconductor Corp. і не тільки.

Переваги мікросхем PSoC:

- зменшують кількість компонентів;
- адаптовані до зміни вимог розробника;
- спрощують розробку і впровадження проекту.

Для прикладу порівнюємо традиційну схему сенсора системи охорони (на мікроконтролері) і схему, в якій застосовується мікросхема PSoC.

Традиційне рішення вимагає операційного підсилювача для нормування вхідного сигналу, ФНЧ для очищення сигналу від завад, контролера з внутрішніми схемами АЦП і ЦАП, зовнішнім кварцом та конденсаторами, вихідного підсилювача, цифрової схеми для роботи з індикаторами (рис. 7.1).

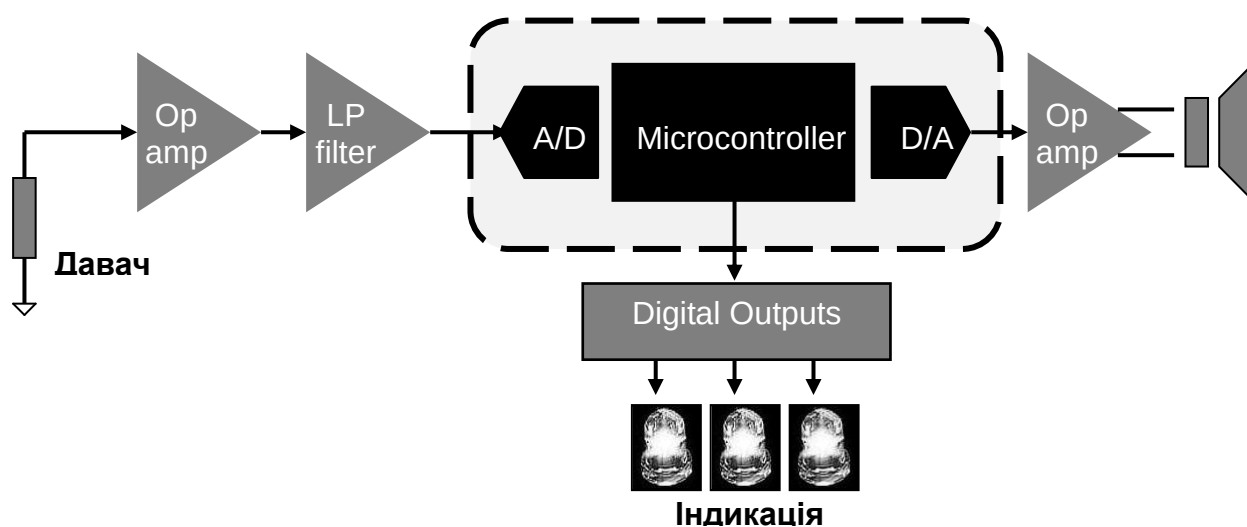


Рисунок 7.1 – Традиційна схема сенсора системи охорони на мікроконтролері

Застосування мікросхеми PSoC усе значно спрощує (рис. 7.2). Схема потребує мінімальну кількість компонентів, зменшується вартість.

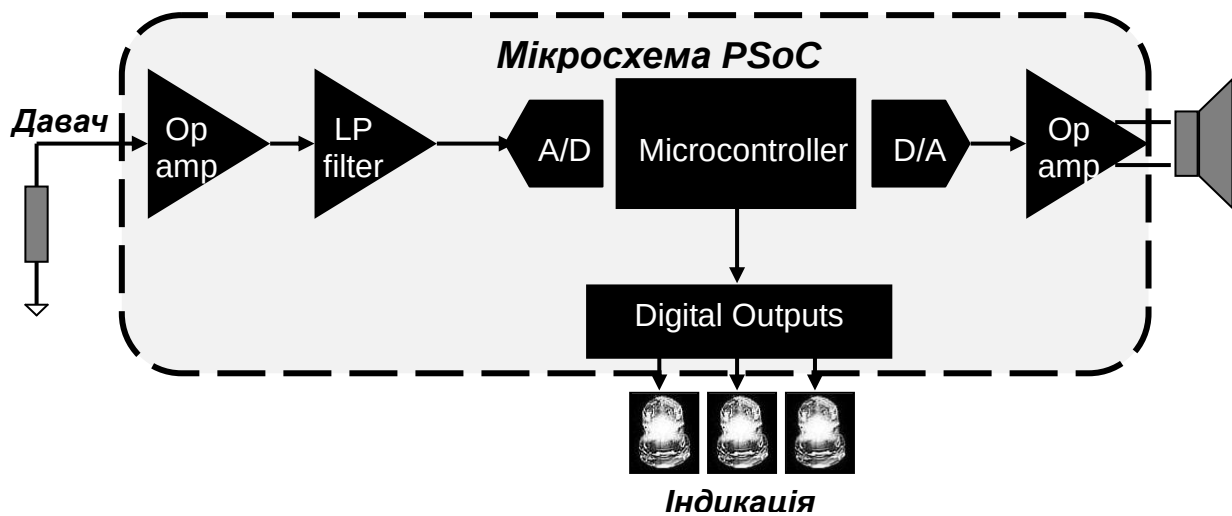


Рисунок 7.2 – Схема сенсора системи охорони на мікросхемі PSoC

7.2 Основні характеристики сімейства PSoC®3.

Мікросхеми сімейства PSoC®3 є подальшим розвитком фірмою Cypress програмованих аналого-цифрових схем на кристалі. У своїй архітектурі PSoC®3 поєднує процесорне ядро (CPU) з дуже гнучкою аналоговою та цифровою підсистемами.

Сімейство PSoC®3 є високопродуктивною конфігурованою цифровою системою, що містить такі інтерфейси, як USB, I2C та CAN. Окрім комунікаційних інтерфейсів, цифрова частина PSoC містить програмовану матрицю логічних елементів, гнучку систему комунікацій до системи введення/виведення та мікропроцесорне ядро 8051. Застосування широко поширеного процесорного ядра 8051 значно розширило коло користувачів. У порівнянні з попереднім сімейством PSoC, де було застосовано у якості цифрових реконфігурованих блоків 8-розрядні функціональне закінчені пристрої, а не набори програмованих логічних вентилів, підвищилася гнучкість і можливості цифрової системи та портів введення/виведення.

Особливості сімейства мікросхем PSoC®3.

У своїй архітектурі PSoC PSoC®3 поєднує процесорне ядро (CPU) з дуже гнучкою аналоговою та цифровою підсистемами (рис. 7.3). До складу PSoC входять такі основні частини:

- Процесорне ядро 8051;
- Підсистема пам'яті ;
- Підсистема програмування, налаштування та тестування;
- Порти введення/виведення;

- Блок генерування тактових імпульсів;
- Блок керування живленням;
- Цифрова підсистема;
- Аналогова підсистема.

Підсистема процесорного ядра. Підсистема CPU PSoC 8051 реалізована на основі одно- циклового конвеєризованого 8-розрядного CPU, здатного працювати на частотах до 67 МГц. Процесорне ядро побудовано на базі конвеєрної RISC архітектури, що виконує більшість інструкцій за 1–2 такти. Це забезпечує пікову продуктивність до 33 MIPS з двома тактами на виконання інструкції. Набір інструкцій CPU є повністю сумісним з набором MCS-51 та оптимізованим для 8-бітної обробки даних та виконання логічних операцій. Основними типами операцій, що підтримуються, є:

- арифметичні операції;
- логічні операції;
- операції пересилання даних;
- бітові операції;
- операції переходів.

Однотактове ядро 8051 забезпечує приблизно на порядок вищу продуктивність порівняно зі стандартним 8051. Тактова частота ядра є програмованою, що дає можливість налаштовувати споживану потужність згідно з вимогами конкретної задачі.

Системна шина та контролер прямого доступу до пам'яті (ПДП) забезпечують обмін даними між процесорним ядром та периферійними пристроями, а також між периферійними пристроями напряму. Окрім того, системна шина та контролер ПДП забезпечують конфігурування PSoC під час завантаження.

Розвинута підсистема пам'яті подібна першому поколінню PSoC.

Периферійні цифрові блоки поповнилися блоками інтерфейсу CAN 2.0b, лічильником та широтне-імпульсний модулятор (ШИМ). На конфігурованих цифрових блоках можуть бути реалізовані стандартні периферійні вузли:

- 8, 16, 24, та 32-бітні таймери, лічильники та ШИМ;
- інтерфейси SPI, UART, I2C;
- інтерфейс LIN Bus 2.0;
- квадратурний декодер.

Аналогова периферія з напругою живлення від 1,71В до 5,5В:

- конфігурований Дельта-Сігма АЦП (12 – 20 розрядне);
- 8- бітні ЦАП за струмом та напругою;
- компаратори сигналів.

Програмована підсистема синхронізації:

- генератор 3 – 62 МГц та вузол ФАПЧ;
- кварцовий генератор частотою 32.768 кГц для годинника реального часу;
- низькоспоживаючий внутрішній осцилятор, що програмується на частоти 1 кГц, 33 кГц та 100 кГц.

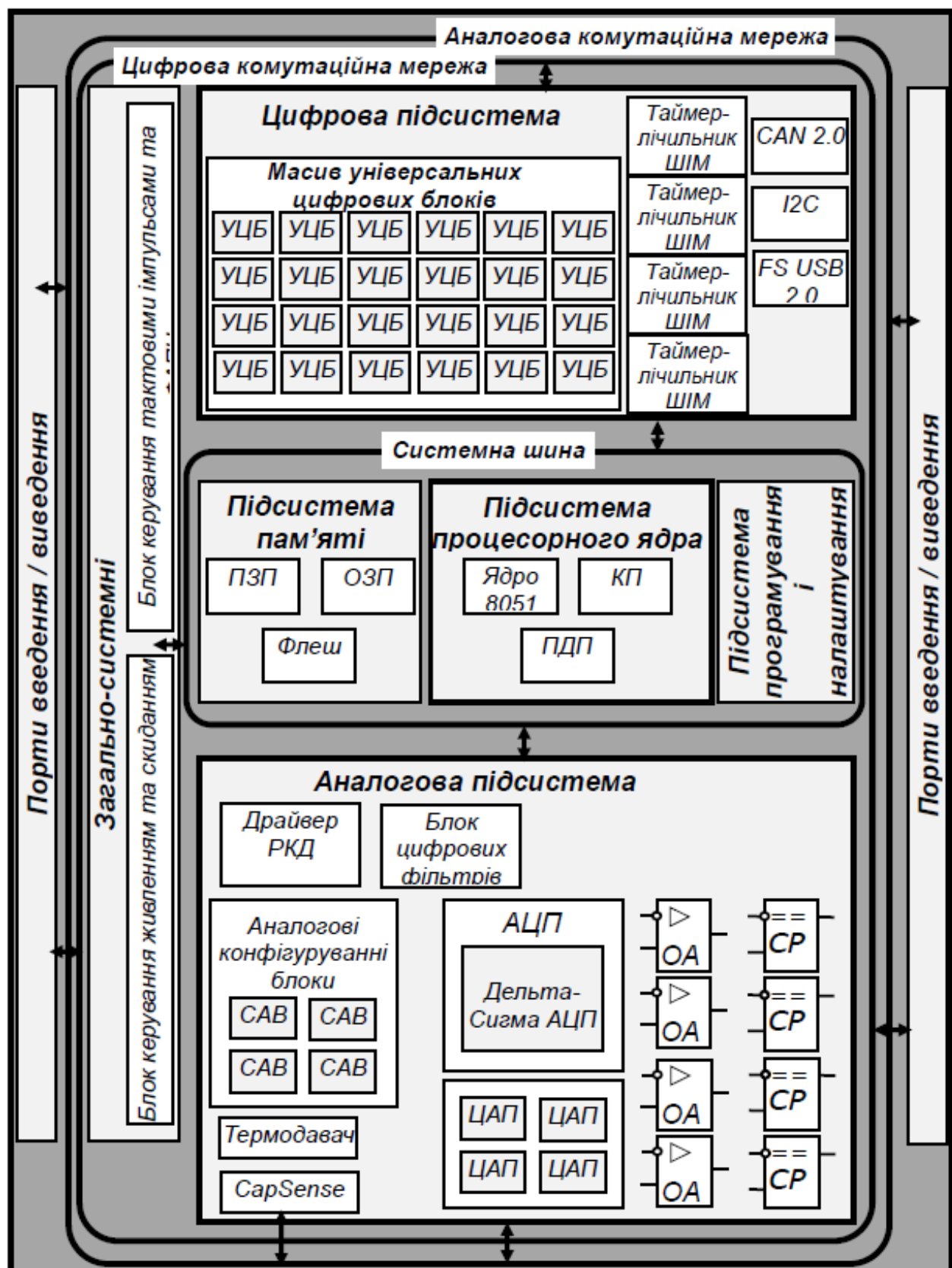


Рисунок 7.3 – Спрощена структурна схема PSoC типу CY8C38

Підсистема пам'яті. PSoC CY8C38 має розвинуту підсистему пам'яті, до складу якої входять: статичний ОЗП, флеш-пам'ять, ПЗП типу EEPROM.

Статичний ОЗП PSoC CY8C38 використовується для тимчасового зберігання даних. Процесорне ядро та контролер ПДП можуть адресувати до 8 кБ ОЗП.

ПЗП типу Flash використовується для зберігання коду вбудованого програмного забезпечення (ПЗ), констант, системних даних, конфігурації PSoC та коду виправлення помилок ECC. Основна область Flash містить до 64 кБ вбудованого ПЗ. До 8 кБ додаткового простору Flash - пам'яті є доступними для даних ECC. Якщо контроль ECC не виконується, то цей сегмент пам'яті може застосовуватись для зберігання даних та конфігурації PSoC. ECC дає змогу виправляти однократні помилки та виявляти двократні для 8 байтів коду вбудованого ПЗ.

Програмування Flash ISSP, що зазвичай застосовується при програмуванні пристроїв під час випуску, може здійснювались через SWD та JTAG інтерфейси. Оновлення вбудованого ПЗ за допомогою спеціальних завантажувачів також можливе через послідовний інтерфейс, такий як I2C, USB, UART та SPI або будь-який інший комунікаційний протокол.

ПЗП типу EEPROM. EEPROM в PSoC є постійною пам'яттю з побайтовим доступом об'ємом 2 КБ. Ця пам'ять використовується для енергонезалежного зберігання даних. Вона має довільний доступ на читання, що здійснюється напряму. Запис та стирання EEPROM здійснюється порядкове через інтерфейс програмування EEPROM. Під час запису EEPROM можливе виконання коду програми з Flash- пам'яті. Весь об'єм EEPROM розділений на 128 рядків по 16 байтів кожен. Процесорне ядро не може вибирати дані EEPROM як виконавчий код. Також EEPROM не має апаратної підтримки ECC.

Підсистема введення/виведення. Підсистема введення/виведення PSoC є надзвичайно гнучкою. Кожна лінія введення-виведення забезпечує можливість роботи як з аналоговими, так і з цифровими сигналами. У різних режимах PSoC також підтримує до чотирьох різних рівнів напруги введення-виведення, що задаються окремими входами Vddio. Існує три типи ліній введення-виведення: загального призначення, спеціального призначення та USB порт.

Лінії загального та спеціального призначення мають однакову функціональність для роботи з цифровими сигналами. Їх основною відмінністю є різні можливості роботи з аналоговими сигналами та навантажувальна здатність. USB порт має дві лінії введення-виведення, що підтримують специфічну USB функціональність та мають обмежені можливості роботи з цифровими сигналами.

Всі лінії введення-виведення є доступними для застосування у вигляді цифрових входів та виходів для CPU та цифрових периферійних пристроїв. Окрім того, всі лінії можуть бути джерелом переривання. Гнучкість підсистеми введення-виведення разом з можливістю комутації будь-якого сигналу з будь-якою лінією введення-виведення надзвичайно спрощує проектування зовнішньої схеми та друкованих плат.

Всі лінії введення-виведення можуть використовуватись як аналогові входи для роботи з CapSense сенсорами та для керування сегментами LCD. Спеціальні лінії введення-виведення можуть працювати з амплітудами сигналами.

лів, що перевищують напругу живлення та видавати на виході сигнали програмованої амплітуди.

Загальносистемні ресурси.

Підсистема синхронізації.

Підсистема синхронізації забезпечує генерування, ділення та розповсюдження тактових імпульсів до всіх частин PSoC. PSoC містить гнучку систему внутрішніх генераторів тактових імпульсів, що мають високу стабільність та налаштовані під час виготовлення з високою точністю. Внутрішній осцилятор ІМО, що є базовим джерелом тактових імпульсів, має точність 1% на частоті 3 МГц. Залежно від конфігурації ІМО може генерувати частоти від 3 до 62 МГц.

Вузол ФАПЧ дає змогу генерувати системні тактові імпульси з частотою від 24 до 67 МГц, використовуючи як джерело ІМО зовнішній осцилятор частотою 4-33 МГц або зовнішній сигнал синхронізації частотою до 33 МГц.

Система генераторів тактових імпульсів містить також окремий внутрішній низькочастотний осцилятор ІЛО (частотою 1 кГц, 33 кГц або 100 кГц), що застосовується в режимах з низьким рівнем споживаної потужності. Підтримка зовнішнього осцилятора частотою 32768 Гц дає змогу легко побудувати годинник реального часу (RTC). За допомогою восьми 16-розрядних подільників формуються необхідні частоти синхроімпульсів цифрової частини PSoC, аналогова частина може використовувати чотири подільника. Процесорне ядро має свій окремий подільник.

Система живлення.

Система живлення складається з окремих ліній живлення аналогової, цифрової систем та системи введення-виведення Vdda, Vddd, та Vddiox відповідно. Два внутрішні регулятори напруги 1,8 В забезпечують живлення внутрішньої логіки ядра. Окрім того, кожен з режимів зниженого енергоспоживання використовує свій власний регулятор напруги.

Системне скидання.

Системне скидання CY8C38 може бути ініційовано внаслідок декількох типів подій у системі, зокрема:

- контроль живлення. Сигнал скидання може генеруватись у результаті відхилення аналогової чи цифрової напруги живлення більш ніж на певну величину від заданого діапазону;
- зовнішній сигнал скидання по лінії XRES;
- генерування сигналу скидання від WDT таймера. WDT контролює виконання інструкцій процесорним ядром. Якщо протягом заданого інтервалу не виконується програмного скидання WDT таймера, то він генерує сигнал системного скидання;
- програмне скидання.

Цифрова підсистема. Особливістю цифрової підсистеми є сукупність стандартних та спеціалізованих периферійних блоків, а також програмованих логічних схем. Це дає змогу створювати вбудовані системи у широкому діапазоні прикладних задач. Взаємозв'язок підсистеми з комутаційною матрицею та портами введення-виведення забезпечує високий рівень гнучкості та захищено-

сті інтелектуальної власності. Основними компонентами цифрової підсистеми є:

- універсальні цифрові блоки (УЦБ), що утворюють базову функціональність підсистеми. Кожен УЦБ поєднує універсальну програмовану логіку на базі ПЛМ та фіксований апаратний блок (тракт даних), оптимізований для створення типових рішень вбудованих систем;

- масив УЦБ, що складається з множини УЦБ та набору з'єднань між ними. Масив має однорідну структуру, що забезпечує високу гнучкість розміщення функціональних блоків. Масив УЦБ під'єднаний до цифрової комутаційної матриці, що забезпечує взаємозв'язок УЦБ з рештою підсистем PSoC;

- цифрова комутаційна матриця. Цифрові сигнали з УЦБ, фіксованих апаратних блоків, портів введення-виведення, лінії переривань, каналів ПДП з'єднуються з цифровою комутаційною матрицею, досягаючи цим самим максимального взаємозв'язку між усіма підсистемами в середині PSoC. Будь-який вхід/вихід цифрової підсистеми комутується з будь-яким зовнішнім входом/виходом кристала.

Аналогова підсистема. Аналогова підсистема за своєю суттю є комбінацією стандартних та спеціалізованих блоків опрацювання аналогових сигналів з можливістю програмування їх функцій під визначену задачу. Кожен блок під'єднаний до аналогової комутаційної мережі, що забезпечує високу гнучкість з'єднань блоків між собою та підсистемою введення-виведення PSoC.

Основні характеристики аналогової підсистеми PSoC PSoC®3:

- Гнучка конфігурована комутаційна мережа аналогових блоків.

- Конфігурований дельта-сігма АЦП з високою роздільною здатністю.

- 4 ЦАП з можливістю конфігурації виходів за струмом чи напругою.

- 4 компаратора.

- 4 універсальні аналогові блоки на основі схем з комутуваними конденсаторами, які призначені для реалізації підсилювачів з програмованим коефіцієнтом підсилення, диференціальних підсилювачів, мікшерів та інших аналогових вузлів.

- 4 операційні підсилювачі для внутрішнього використання або реалізації вихідних буферів високого струму.

- Підсистема CapSense для підключення та опрацювання сигналів з ємнісних давачів. Вимірювання ємності використовують у тих областях, де вимагається створення безконтактних перемикачів (сенсорів) для інтерфейсів з користувачем.

- Інтегровані високоточні джерела аналогових опорних напруг для внутрішніх аналогових блоків.

Інтегроване середовище розробки PSoC Creator.

Всі PSoC фірми Cypress підтримуються та програмуються в інтегрованому середовищі розроблення PSoC Creator. PSoC Creator є середовищем сумісного розроблення апаратних і програмних засобів (hardware/software co-design), що поєднує інструменти розроблення програмних засобів мовою C, апаратних засобів шляхом RTL – описів мовою Verilog та на рівні функціональних схем,

розроблення інтерфейсів користувача мовою C#, засоби тестування та відлагодження тощо. Бібліотека PSoC Creator містить набір готових компонент, які можна вводити в проект системи як завершені функціональні блоки. Для кожної компоненти в бібліотеці поставляється супровідна документація з повним описом її функціоналу. Важливою характеристикою середовища є можливість створення власної бібліотеки компонент, що можуть повторно бути використаними надалі.

7.3 Мікросхеми сімейства PSoC® 4

Мікросхеми PSoC® 4 мають архітектуру реконфігурованої та масштабованої платформи сімейства контролерів змішаного сигналу з програмованим вбудованим у систему процесором ARM®Cortex™-M0. Вони поєднують в собі програмовані і реконфігуровані аналогові і цифрові блоки з гнучкою функцією автоматичного трасування маршрутів. Продукт сімейства PSoC 4200M, заснований на цій платформі, являє собою поєднання мікроконтролера із цифровою програмованою логікою, аналого цифрового перетворення високої продуктивності, операційних підсилювачів з режимом порівняння і стандартними засобами комунікації і тактування периферійних пристроїв. Мікросхеми сімейства PSoC 4200M мають найбільшу кількість вбудованих підсистем і блоків та повністю сумісні з іншими мікросхемами платформи PSoC 4 для проектування нових додатків і потреб (рис. 7.3). У них відсутній блок обробки сигналів від індуктивних датчиків – *Inductive Sensing (MagSense)*, який присутній у деяких інших мікросхемах сімейства, проте вони мають підсистему пам'яті більшої ємності.

Програмовані аналогова і цифрова підсистеми забезпечили гнучкість і налаштування проектування.

До складу мікросхеми PSoC 4200M входять такі основні частини (рис. 7.4):

- Підсистема процесорного ядра;
- Аналогова програмована підсистема;
- Цифрова програмована підсистема;
- Підсистема входу/виходу;
- Блок обробки сигналів з ємнісних датчиків;
- Блок послідовних інтерфейсів;
- Блок інтерфейсу CAN;
- Блок таймера/лічильника та широтно-імпульсної модуляції;
- Блок драйвера рідкокристалічного індикатора (РКІ);
- Системні ресурси.

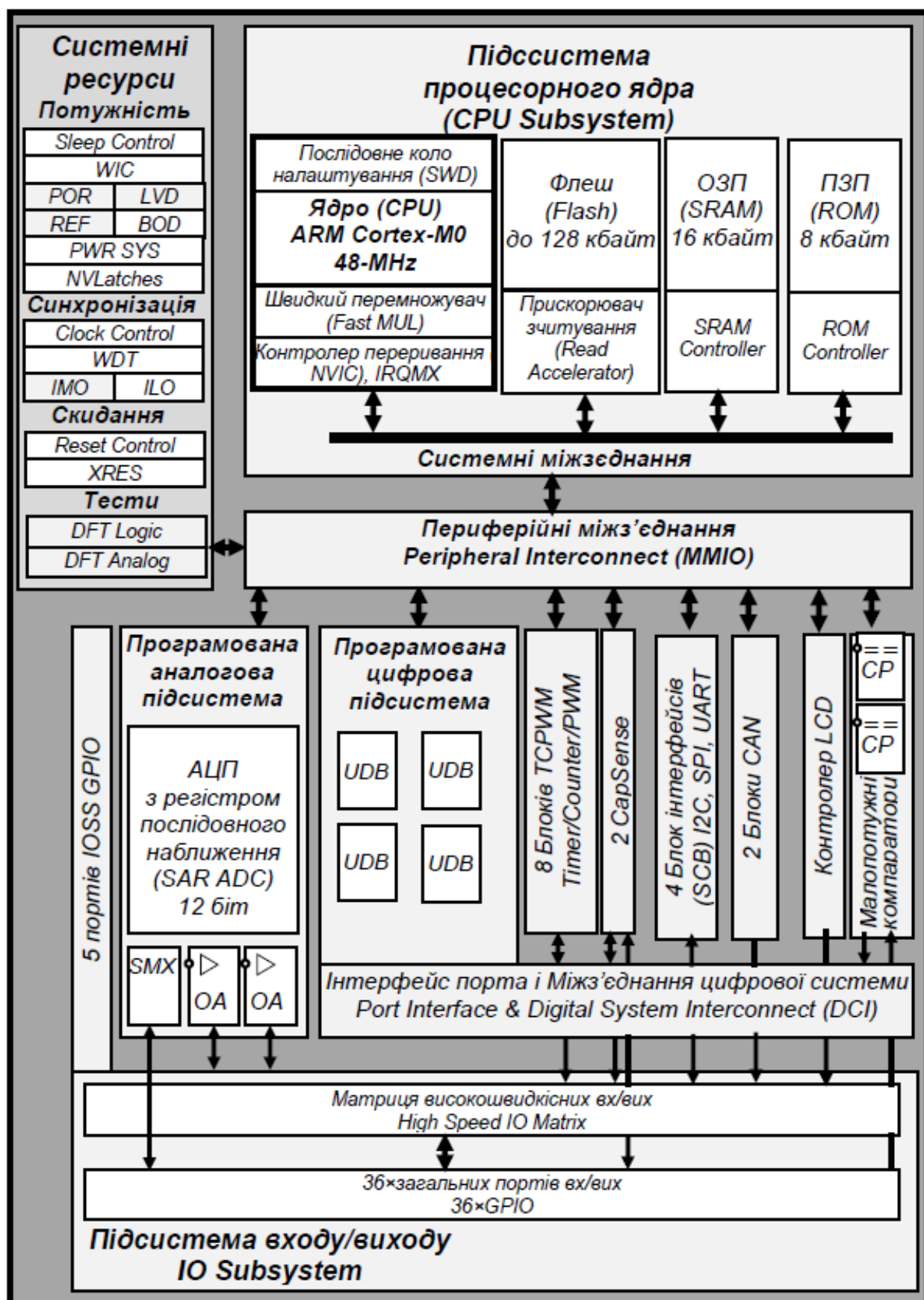


Рисунок 7.4 – Спрощена структурна схема мікросхеми сімейства PSoC 4200M

Підсистема процесорного ядра. Підсистема процесорного пристрою (MCU) об'єднує:

- процесорне ядро на базі 32-розрядного процесора ARM Cortex-M0 з тактовою частотою 48 МГц і з можливістю виконання операції перемноження за один такт;
- флеш-пам'ять ємністю до 128 кБ з прискорювачем читання (Read Accelerator);
- ОЗП (SRAM) ємністю до 16 кБ;
- ПЗП (ROM) ємністю до 8 кБ.

Процесорне ядро ARM Cortex-M0 є частиною 32-розрядної MCU підсистеми, яку оптимізовано для виконання операцій з малою споживаною потужністю та розширеною логікою системи тактування. Воно використовує 16-розрядні команди та виконує набір команд з підмножини Thumb-2. Це гарантує повну сумісність двійкових кодів знизу до гори з більш продуктивними процесорами Cortex-M3 і M4.

Ядро містить апаратний множник, який забезпечує результат для 32 розрядного числа за один такт. Ядро включає також блок контролера вектору вкладених переривань (nested vectored interrupt controller, NVIC) з 32 входами і контролер пробудження переривань (Wakeup Interrupt Controller, WIC). Контролер WIC виводить процесор із режиму глибокого сну (the Deep Sleep mode), коли на ньому відсутня напруга живлення і мікросхема знаходиться в режимі глибокого сну. Процесорне ядро ARM Cortex-M0 забезпечує введення немаскованих переривань (Non Maskable Interrupt, NMI), які доступні користувачеві та потрібні йому, і не використовуються системою.

До CPU належить інтерфейс для налаштування, послідовне коло налагодження (serial wire debug, SWD), що є 2-х магістральною формою інтерфейсу JTAG;

З модулем флеш-пам'яті тісно пов'язаний прискорювач зчитування, який зменшує середній час доступу до даних у флеш. Прискорювач забезпечує час доступу 85% відносно одного циклу зчитування SRAM. Частина модуля флеш, якщо вимагається, може використовуватися для емуляції EEPROM.

Аналогова програмована підсистема. Аналогова програмована підсистема включає: операційні підсилювачі (ОП), АЦП, ЦАП, компаратори і блок ємнісного давача.

Два операційних підсилювачі можна конфігурувати для зовнішніх з'єднань як підсилювач потужності (драйвер), для внутрішніх вживань – широкосмуговий підсилювач. Їх дозволено використовувати для роботи у режимі компаратора та вхідного буфера для АЦП.

Операційні підсилювачі програмуються без застосування зовнішніх компонентів і на них можна реалізовувати: підсилювачі з програмованим коефіцієнтом підсилення (PGA), фільтри, буфери напруги, транс-імпедансі підсилювачі та інші функціональні елементи, що економить споживану потужність, вартість та площу. Вбудовані ОП розроблені з достатньо широкою смугою пропускання і вихідним струмом для схеми вибирання/збереження (S/H circuit), і АЦП не потребує зовнішнього буфера.

АЦП з регістром послідовного наближення (SAR ADC, Successive Approximation Register ADC) з розрядністю 12 біт і продуктивністю 1 Msps працює у диференціальному і несиметричному режимах. До виводів корпусу АЦП підключається через восьми канальний аналоговий мультиплексор (8-input sequencer, SMX), входи якого скануються послідовно. Можливо програмування кожного каналу з потрібною частотою відліків для різних джерел з відмінними опорними напругами і частотами. АЦП має можливість оцифровувати сигнал з вбудованого датчика температури для калібрування та інших залежних від температури функцій. АЦП не працює у сплячому режимі та гібернації, тому що потребує високої тактової частоти (до 18 МГц). Робочий діапазон джерела живлення становить від 1,71 В до 5,5 В.

Два струмових ЦАП (IDACs) для загального призначення або ємнісних датчиків на будь-якому виводі.

Два малопотужних компаратори діють у режимі глибокого сну та гібернації. Їх наявність дозволяє вимикати аналогову частину у малопотужних режимах, втім зберігає можливість контролю зовнішньої напруги. Виходи компараторів синхронізовані для уникнення асинхронного режиму у колі активації режиму пробудження системи.

Датчик температури (Temperature Sensor).

На кристалі розміщено датчик температури, що складається з діоду та джерела струму і які можливо відключати для економії енергії. Датчик температури підключено до АЦП, який оцифровує його данні і виробляє відповідні зміни у програмному забезпеченні, що включає калібрування та лінеаризацію.

Програмована цифрова підсистема включає:

- чотири програмовані універсальні цифрові блоки, УЦБ, (universal digital blocks, UDBs). Кожен блок містить 8 макрочарунок та фіксований апаратний блок (тракт даних), оптимізований для створення типових рішень вбудованих систем. УЦБ можливо тактувати сигналами з блоку дільника, порту інтерфейсу (потрібно для периферійних пристроїв, таких як SPI), і від мережі DSI безпосередньо або після синхронізації. Визначається порт інтерфейсу, регістр якого послуговує джерелом тактового сигналу також і для схем ПЛМ (PLD) в осередку масиву УБЦ.

- УЦБ мають можливість генерувати сигнал переривання для контролера переривань (один УБЦ у один той же час). Вони зберігають можливість підключення до будь-якого виводу на кристалі через DSI.

- надається бібліотека периферійних компонентів користувача, кінцевих автоматів і входи для введення функцій на мові Verilog.

Фіксовані цифрові функції.

Блок таймера/лічильника та широтно-імпульсної модуляції, ТЛШІМ, (Timer/Counter/PWM, TCPWM Block).

Блок складається з 16-розрядного таймер/лічильника та широтно-імпульсного модулятора (TCPWM) з програмованою користувачем довжиною періоду. Існує регістр захвату для запису значення лічильника на момент події (це може бути подія введення/виведення), періоду регістра, який застосовують-

ся для зупинки або автоматичного перевантаження лічильника, коли значення лічильника дорівнює періоду регістру, і порівняння регістрів для утворення сигналу порівнювання, що завдає довжину циклу ШІМ. Блок забезпечує на виходах пряме та комплементарне значення напруги з програмованою величиною зміщення між ними, для того щоб програмно дозволити застосування виводів, які не використовуються. Наприклад, це застосовується у системах приводу, коли вказується стан перевантаження по струму і керовані ШІМ транзистори FET повинні вимкнутися відразу без втручання програмного забезпечення. Мікросхема містить вісім блоків ТЛШІМ у яких також є налаштування режимів вирівнювання імпульсів по центру, краю і псевдовипадковий.

Блок послідовних інтерфейсів (зв'язків) (SCB).

Мікросхема PSoC 4200M має чотири незалежних по часу виконання конфігурованих блока послідовного зв'язку, які можливо налаштувати у якості інтерфейсів I2C, SPI або UART.

Апаратний блок I2C реалізує різні конфігурації режими роботи ведучий/ведомий. Блок може працювати з продуктивністю до 1 Мб/с (у швидкому режимі плюс) та володіє гнучкими параметрами буферізації для зменшення затрат на переривання та затримок для процесорного ядра.

Периферійні блоки I2C сумісні з стандартним режимом інтерфейсу I2C, швидким режимом та швидким режимом плюс. Порти GPIO реалізують входи/виходи I2C у режимі з відкритим стоком.

Режим UART є повнофункціональним до швидкодії 1 Мб/с. Він підтримує автомобільний однодротовий інтерфейс (LIN), інфрачервоний інтерфейс (IrDA), протокол смарт-карт SmartCard, ISO7816), які є незначно відмінними від основного протоколу UART. Зокрема він підтримує 9-біт багатопроцесорний режим, який дозволяє адресацію периферійних пристроїв через загальні RX and TX лінії, визначає помилки парності, перерви, кадра.

Режим SPI. В режимі підтримується повністю інтерфейс SPI фірми Motorola та Texas Instruments (TI SSP) (додається стартовий імпульс для синхронізації SPI кодеків. напівдуплексна форма SPI. Блок SPI може використатися у якості FIFO.

Блоки CAN.

Мікросхема має два незалежних блока CAN 2.0B, які сумісні з сертифікованим інтерфейсом CAN.

Порти входу/виходу загального призначення (GPIO).

Мікросхема PSoC 4200M у корпусі QFN з 68 виводами містить 55 GPIO. Порт GPIO реалізує наступне:

- вісім режимів потужності драйвера, у тому числі потужний двотактний, з резистором, що увімкнено до джерела живлення або до шини «земля», з відкритим витоком або стоком, тільки вхід і відключено;
- вибір порогу на вході (CMOS або LVTTTL);
- індивідуальний контроль відключення вх/вих;
- утримує режим попереднього стану (використаються для зберігання стану введення/виведення у сплячому режимі та режимі глибокого сну);

– вибір швидкості наростання імпульсів відносно рівня шуму для контролю поліпшення електромагнітної сумісності (ЕМІ).

Виводи об'єднуються у групи по вісім. Під час вмикання джерела живлення або перезавантаження виводи групи знаходяться у відключеному стані, щоб не пересікатися з іншими виводами входами і/або підвищення струму увімкнення.

Коло мультиплексування. так звана матриця швидкодіючих вх/вих, використовується для мультиплексування різноманітних сигналів, які можуть бути підключені до закріплених виводів вх/вих. Розміщення контактів для фіксованих периферійних функцій також є фіксованим для зменшення складності внутрішніх мультиплексованих з'єднань (ці сигнали не проходять через коло DSI). Сигнали з будь-якого виводів портів 0, 1, 2, 3 можуть надходити до УБЦ через коло DSI, а також пов'язані між себе.

Кожен контакт вх/вих може генерувати переривання, якщо його відповідно підключено, і кожен порт вх/вих має можливість формувати запит на переривання (IRQ) та служба звичайного переривання (interrupt service routine, ISR). Будь які виводи GPIO можуть послугувати виводами для CapSense, LCD, аналоговими або цифровими;

Спеціальні функції периферії.

Блок драйвера рідкокристалічного індикатора (PKI) (LCD Drive).

Мікросхема містить контролер PKI (LCD), який керує загальними шинами (до чотирьох) і сегментами (до 32) рідкокристалічного індикатора. Для цього використовуються повністю цифрові методи, що не вимагають генерації особливої внутрішньої напруги для PKI.

Модуль ємнісного давача (CapSense).

Модуль ємнісного давача підтримується на всіх виводах мікросхеми через CapSense Sigma Delta (CSD) блок, який може бути підключений до будь-якого контакту GPIO через шини аналогового мультиплексора з'єданого з аналоговими ключами. Функції ємнісного давача підтримуються на будь-якому виводі або групі виводів під контролем програмного забезпечення. Модулі користувача (програмне забезпечення), що постачається, робить легким проектування ємнісних давачів. Можливо автоматичне налаштування апаратного обладнання (SmartSense™).

Модуль ємнісного давача має два струмових ЦАП, які можливо застосовувати для загальних цілей. якщо модуль CapSense не застосовується. У випадку застосування у модулі CapSense одного ЦАП по призначенню, інший можливо застосовувати для загальних цілей.

Сигма Дельта АЦП для обробки сигналів ємнісних давачів (CapSense Sigma-Delta, CSD) забезпечує найкраще у своєму класі співвідношення сигнал/шум (SNR) завбільшки 5:1 і повну відповідність перетворення.

Системні ресурси

Підсистема живлення.

Система живлення забезпечує потрібну точність рівнів напруги для кожного відповідного режиму роботи або затримки режиму запису, скидання на включення (on power-on reset, POR), наприклад, до рівня напруги потрібного

для функціонування сигналу скидання або його генерації (brown-out detect, BOD), або переривань (low-voltage detect, LVD).

Особливості системи живлення[^]

- потрібно одне джерело живлення з напругою від 1,71 В до 5,5 В;
- можливо виконання операцій з низьким енергоспоживанням з напругою живлення від 1,71 В до 5,5 В:
- струм споживання 20-пА у режимі Stop під час активації виводу порту GPIO;
- знайдено компроміс споживаної потужності між режимами гіберназації та глибокого сну (Hibernate and Deep Sleep modes) і часом активації.

Підсистема синхронізації.

Система синхронізації відповідає за забезпечення синхронізації усіх тактовних систем для запобігання помилок перемикання цифрових пристроїв. Крім цього система гарантує відсутність невизнаності стану цифрових елементів.

Система синхронізації складається з внутрішнього основного генератора (IMO), що є базовим джерелом тактових імпульсів, і внутрішнього малопотужного генератора (ILO) та засобів для зовнішньої синхронізації.

Основний генератор IMO піднастроюється для досягнення потрібної точності під час тестування. Величина відхилення частоти запам'ятовується у енергонезалежній пам'яті. Частота генератора IMO по умовчання складає 24 МГц і вона встановлюється у діапазоні частот від 3 МГц до 48 МГц з кроком 1 МГц. Точність калібрування становить $\pm 2\%$.

Внутрішній низькочастотний осцилятор ILO з номінальною частотою 32 кГц застосовується в режимах з низьким рівнем споживаної потужності. Для підвищення точності лічильники у генераторі ILO можуть бути відкалібровані за допомогою генератора IMO. Це забезпечує відповідний програмний компонент.

Кварцовий генератор.

Підсистема синхронізації також включає низькочастотний кварцовий генератор частотою 32 кГц (32-kHz WCO), який застосовується для синхронізації у режимі глибокого сну, реалізації годинника реального часу (Real-Time Clock, RTC) і сторожового таймера (Watchdog Timer).

Сторожовий таймер входить до блоку синхронізації запускається від низькочастотного генератора. Він діє протягом режиму глибокого сну (Deep Sleep) і генерує сигнали скидання та переривання, якщо не служить before the timeout occurs. Сигнал скидання сторожового таймера записано у регістр скидання (Reset Cause register).

Система скидання.

Скидання можливе від різноманітних джерел, включаючи програмне скидання. Скидання є асинхронною подією і гарантує повернення у відомий стан. Вивід забезпечує зовнішнє скидання у випадках зміни конфігурації і збільшенням функцій виводу протягом вмикання живлення та реконфігурації.

Система живлення.

Мікросхема зберігає усі свої функції і режими роботи у діапазоні напруги живлення від 1,71 В до 5,5 В. Можлива робота з напругою живлення у двох режимах: з нерегульованою зовнішньою напругою та регульованою зовнішньою напругою.

Система живлення генерує усі внутрішні опорні напруги. Для 12 розрядного АЦП забезпечується опорна напруга з точністю 1%. Для підвищення співвідношення сигнал/шум (signal-to-noise ratios, SNR) і підвищення абсолютної точності можливо додавати зовнішній фільтруючий конденсатор до внутрішнього джерела опорної напруги через вивід GPIO або застосовувати зовнішнє джерело опорної напруги для SAR.

7.4 Сімейство PSoC® 5LP

Мікросхеми сімейства PSoC® 5LP є справжньою програмованою вбудованою системою на кристалі, що інтегрує настроюванні аналогові і цифрові пристрої, пам'ять і мікроконтролер на єдиному кристалі. Особливості архітектури PSoC 5LP підвищують продуктивність внаслідок застосування:

- процесорного ядра 32-бітного Arm Cortex-M3 плюс контролер DMA і процесор цифрового фільтру з діапазоном до 80 МГц;
- ультра низьке енергоспоживання у своїй галузі з найширшим діапазоном напруги живлення;
- програмованих цифрових і аналогових периферійних пристроїв, з функціями, що активує користувач;
- гнучкої маршрутизації будь-якої аналогової або цифрової периферійної функції до будь-котрого контакту.

Мікросхеми PSoC мають архітектуру кристалу, що налаштовуються для розробки вбудованих систем контролю. На одній мікросхемі PSoC можливо інтегрувати 100 цифрових та аналогових функцій, що зменшує час розробки, споживану потужність, вартість та простір на платі при одночасному підвищенні якості системи.

Особливості мікросхем сімейства PSoC® 5LP.

Експлуатаційні характеристики:

- діапазон напруги живлення від 1,71 В до 5,5 В, до 6 доменів потужності;
- інтервал робочих температур (навколишнього середовища): –40 до 85 °С, з розширеним температурним діапазоном: –40 до 105 °С;
- операції з сигналами від постійного струму до 88 МГц.

Режими споживання:

- струм споживання у активному режимі на частоті 6 МГц становить 3,1 мА, на частоті 48 МГц – 15,4 мА;
 - у сплячому режимі – 2 мкА;
 - у глибокому сплячому режимі (гібернації) з утриманням ОЗП – 300 нА.
- На кристалі розміщено імпульсне джерело підвищення напруги від 0,5 В до 5В.

Продуктивність забезпечують:

- 32 розрядний процесор Arm Cortex-M3, з можливістю 32 переривань на вході;
- 24 каналний контролер прямого доступу до пам'яті (direct memory access, DMA) з передачею даних між периферійними пристроями та пам'яттю;
- 24 розрядний цифровий фільтр(digital filter processor, DFB) з фіксованою крапкою.

Пам'ять складають:

- флеш- пам'ять програм ємністю до 256 кбайт з кешем та функцією безпеки;
- додаткова флеш-пам'ять до 32 кбайт для кодів корекції помилок (error correcting code, ECC);
- ОЗП до 64 кбайт;
- репрограмований постійний запам'ятовуючий пристрій з електричним стиранням, РПЗП-ЕС, (EEPROM).

Цифрова периферія містить:

- 4 блоки 16 розрядних таймер/лічильників та ШІМ (TCPWM blocks);
- блок I2C з шиною продуктивністю 1 Мб/с;
- блок USB 2.0, сертифікований на повну швидкість (Full-Speed, FS) 12 Мб/с периферійного інтерфейсу, що використовує внутрішній генератор;
- блок повного інтерфейсу CAN 2.0b з 16 приймаючими та 8 передаючими буферами;
- від 20 до 24 універсальних цифрових блоків, які програмуються для створення будь якої кількості функцій;
- 8-, 16- 24- та 32 розрядні таймери/лічильники і ШІМ;
- інтерфейси I2C, UART, SPI, I2S, LIN 2.0;
- блок циклічної перевірки переповнення (cyclic redundancy check, CRC);
- генератор псевдовипадкової послідовності (pseudo random sequence, PRS);
- квадратурний декодер;
- нормалізатор рівнів логічних функцій.

Програмовану систему синхронізації забезпечують:

- внутрішній генератор від 3 МГц до 74 МГц, з точністю 1% до 3 МГц;
- зовнішній кварцовий генератор від 4 МГц до 25 МГц;
- внутрішнє коло PLL тактування з частотою генерації до 80 МГц;
- внутрішній малопотужний генератор частотою 1 кГц, 33 кГц та 100 кГц;
- зовнішній годинниковий кварцовий генератор частотою 32,768 кГц;
- 12 годинникових дільників, які можливо зв'язати з будь якою периферією та входом/виходом.

Аналогова підсистема включає:

- конфігурований від 8 до 20 розрядів дельта-сигма АЦП (delta-sigma ADC);
- до двох 12 розрядних з регістром послідовного наближення АЦП (SAR ADC);

- чотири 8-розрядних ЦАП (DAC);
- чотири компаратора;
- чотири операційних підсилювача;
- чотири програмованих аналогових блоки:
- підсилювача з програмованим підсиленням (PGA);
- трансімпедансного підсилювача (TIA);
- змішувач;
- мультиплексований аналоговий вхід (AFE);
- схема вибирання – зберігання.

Блок ємнісного давача (CapSense®) з автоматичним налаштуванням SmartSense™ підтримує до 62 давачів.

Джерело внутрішньої опорної напруги $1,024\text{ В} \pm 0,1\%$.

Система універсальних входів/виходів:

- від 46 до 72 контактів вх/вих, до 62 вх/вих загального призначення (general-purpose I/Os, GPIOs);
- до восьми режимів продуктивності контактів спеціального вводу/виводу (SIO), струм кожного 25 мА.

Програмовані порогова напруга на вході та рівень високої напруги на виході.

Можуть виступати у якості компаратора загального призначення.

Можливість гарячого вмикання та нечутливість до підвищеної напруги.

Два контакти вх/вих інтерфейсу USB (USBIO), які можна використовувати як GPIO.

З'єднання будь-якого цифрового або аналогового периферійних блоків до будь-якого GPIO.

Рідкокристалічним індикатором (LCD) можливо керувати через будь-який GPIO до 46×16 сегментів.

Сигнал на блок ємнісного давача можливо подавати від будь-яких GPIO.

Напруга живлення інтерфейсів від 1,2 В до 5,5 В, до чотирьох доменів потужності.

Програмування, налагодження та трасування.

Забезпечується інтерфейсами: JTAG – 4 дротовим, послідовним колом налаштування (serial wire debug, SWD) – 2-дротовим, одне дротовий – (single wire viewer, SWV), портом трасування (Trace port) – 5-дротовим.

Можливо ручне налаштування та трасування за допомогою модулів, що вбудовані у ядро процесора.

Програми завантажуються через інтерфейси I2C, SPI, UART, USB та інші.

Розробки підтримуються безкоштовною програмою PSoC Creator™. Вона підтримує схемотехнічні проектування та розроблення прошивки. Компоненти знаходяться у вільному доступі у вигляді іконок. Для проектування системи у PSoC Creator потрібно вибрати іконку та встановити її. До неї включені безкоштовний компілятор GCC, підтримує також компілятор Keil/Arm MDK.

Архітектура сімейства CY8C58LP.

Сімейство CY8C58LP володіє ультра низьким енергоспоживанням та є флеш програмованою системою на кристалі на базі 32 розрядної платформи. Сімейство CY8C58LP забезпечує конфігурованість аналогових, цифрових блоків та кіл міжз'єднань навколо підсистеми процесора. Високий рівень інтеграції забезпечує комбінацію процесора з гнучкою аналоговою і цифровою підсистемами, підсистемою маршрутизації і підсистемою входів-виходів. Це забезпечує застосування у промисловості і медицині. Основні складові частини сімейства мікросхем CY8C58LP наведено на рис. 7.5.

До складу сімейства мікросхем CY8C58LP входять:

- процесорна підсистема Arm Cortex-M3;
- енергонезалежна підсистема;
- підсистеми програмування, налаштування і тестування;
- входи та виходи;
- система синхронізації;
- система живлення;
- цифрова підсистема;
- аналогова підсистема.

Цифрова підсистема забезпечує половину унікального конфігурування мікросхеми PSoC. Вона з'єднує цифровий сигнал від будь-якого периферійного пристрою до будь-якого контакту через цифрову систему з'єднань (digital system interconnect, DSI). Вона також забезпечує функціональну гнучкість через матрицю універсальних цифрових блоків UDB. Фірма розробник PSoC надає бібліотеку вбудованих та випробуваних стандартних цифрових периферійних пристроїв (UART, SPI, LIN, PRS, CRC, таймер, лічильник, ШІМ, «І», «АБО», і так далі), які відповідають матриці UDB. Можна також легко створювати цифрові кола, використовуючи булеві примітиви за допомогою введення проекту у графічний формі. Кожен UDB містить програмовану логічну матрицю (PAL), що є функціонально програмованим логічним пристроєм (PLD), яка разом з невеликим автоматом зміни стану підтримує широкий спектр периферійних пристроїв.

Зокрема гнучкості матриця UDB також забезпечує налаштування цифрових блоків на виконання конкретних цільових функцій. Для сімейства CY8C58LP ці блоки можуть включати чотири 16-бітні таймери, лічильники і блоки ШІМ; блоки інтерфейсу I2C у різних режимах; блоки повношвидкісного інтерфейсу USB; блоки повного інтерфейсу CAN 2.0.

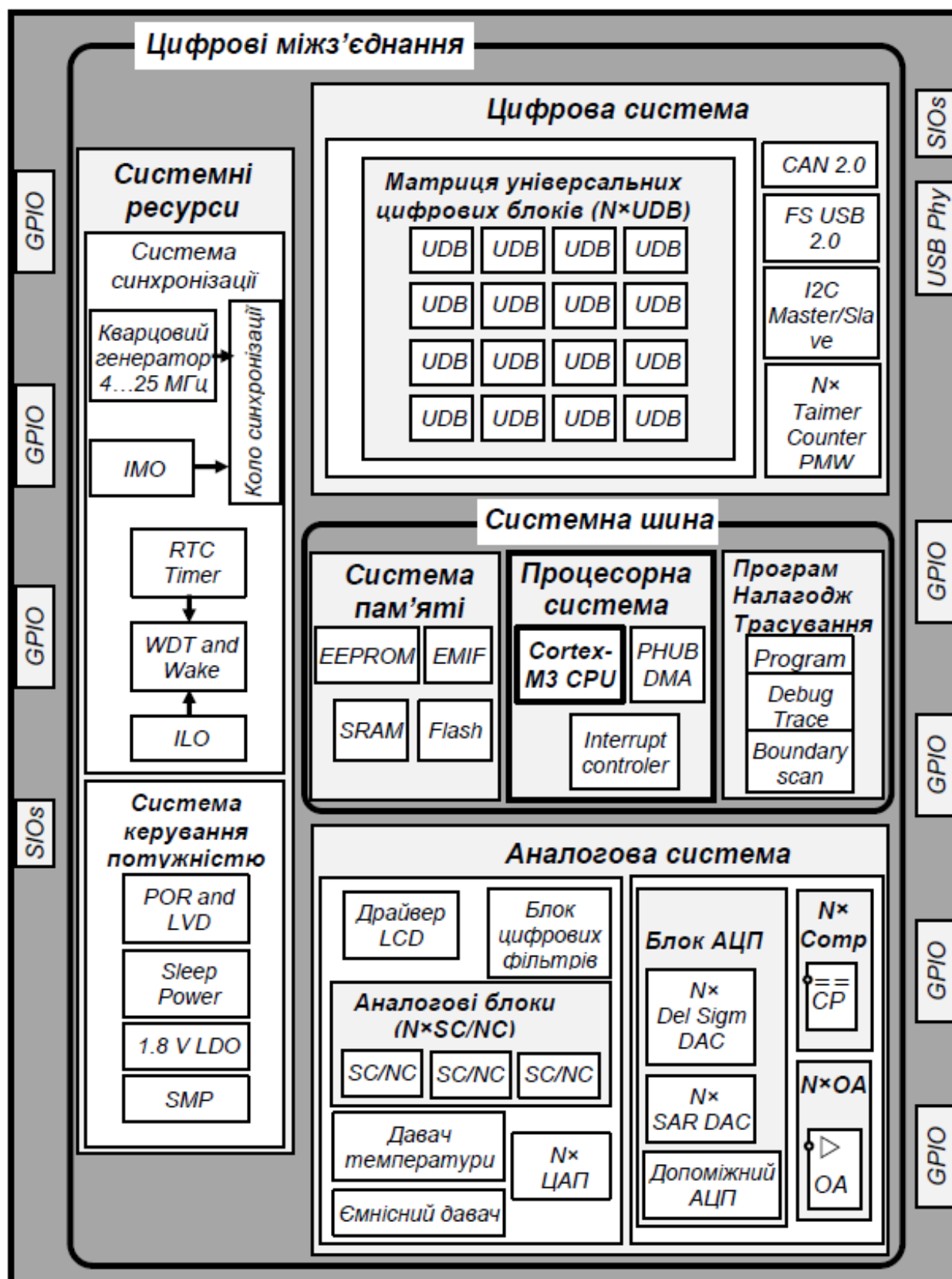


Рисунок 7.5 – Спрощена архітектура мікросхеми сімейства CY8C58LP

Аналогова підсистема. Аналогова підсистема є другою половиною унікального конфігурування мікросхеми PSoC. Всі аналогові застосування основані на високій абсолютній точності опорної напруги з помилкою меншою 0,1% від температури і напруги.

Конфігурована аналогова підсистема містить:

- аналогові мультиплексори;
- компаратори;
- аналогові змішувачі;
- джерела опорної напруги;
- декілька АЦП;
- декілька ЦАП;
- блок цифрового фільтру (DFB).

Всі контакти GPIO можуть перенаправляти аналогові сигнали у пристрій та з пристрою з використанням внутрішньої аналогової шини. Це дозволяє інтерфейсу пристрою обробляти до 62 різних аналогових сигналів. Один з АЦП у аналоговій підсистемі є швидким і точним – це конфігурований дельта сигма АЦП з наступними особливостями:

- напруга зсуву не більше 100 мкВ;
- помилка підсилення 0,2%;
- інтегральна нелінійність (INL) менш ніж ± 2 одиниці молодшого розряду;
- диференціальна нелінійність (DNL) менш ніж ± 1 одиниця молодшого розряду;
- відношення сигнал/(шум + перекручування) (signal to noise and distortion ratio, SINAD) краще ніж 84 дБ у режимі 16 розрядів.

Дельта сигма АЦП призначено для широкого кола прецизійних аналогових застосувань, включаючи обробку сигналів з деяких з найбільш вимогливих давачів.

Сімейство CY8C58LP також пропонує до двох SAR АЦП. Вони виконують 12-розрядні перетворення з продуктивністю 10^6 відліків у секунду і мають низьку нелінійність, помилку зсуву і відношення сигнал/шум (SNR) краще, ніж 70 дБ. Вони добре підходять для реалізації різних аналогових високошвидкісних додатків.

Вихід будь-якого АЦП за бажанням можна зв'язати з програмованими DFB через DMA без втручання процесора. Можливо налаштувати DFB для реалізації цифрових фільтрів з обмеженою імпульсною характеристикою (finite impulse response, FIR) і необмеженою імпульсною характеристикою (infinite impulse response, IIR) та кількох функцій користувачів. Цифровий фільтр може виконувати операцію множення-накопичення (multiply-accumulate, MAC) з 48 розрядним числом за один такт.

Чотири високошвидкісний АЦП з виходом напруги або струму підтримують 8-розрядний вихідний сигнал з швидкістю оновлення до 8 Msps. Вони можуть бути з'єднані з будь-яким контактом GPIO. Є можливість створити

ШІМ ЦАП з вихідною напругою з високою роздільною здатністю, використовуючи масив UDB. Це може бути використано для створення ШІМ ЦАП до 10 біт в до 48 кГц. Цифрові ЦАП у кожному UDB підтримують алгоритми ШІМ, PRS або дельта сигма.

У доповненні до АЦП, ЦАП та цифрового фільтру аналогова підсистема містить:

- компаратори;
- непов'язані (незадіяні) операційні підсилювачі;
- блоки на комутованих конденсаторах та безперервного часу (SC/CT).

Блоки SC/CT реалізують:

- трансімпедансні підсилювачі;
- підсилювачі з програмованим підсиленням;
- змішувачі;
- інші подібні аналогові компоненти.

Підсистема процесора PSoC побудована на 32-розрядному з триступінчатим конвеєром процесорі Arm Cortex-M3 з тактовою частотою до 80 МГц. Процесор включає у себе повністю інтегрований контролер вкладених векторів переривань (NVIC) і різні модулі налаштування та трасування. Загалом підсистема процесора містить контролер прямого доступу до пам'яті (direct memory access) DMA, контролер кеша флеш-пам'яті та ОЗП (RAM). Контролер переривань NVIC забезпечує малу затримку і послідовність ланцюжка переривань та інші функції для підвищення ефективності обробки переривань. Контролер DMA дозволяє периферійним пристроям обмін даними без участі процесора. Це дозволяє процесору працювати повільніше (економія енергії) або використовувати ці цикли процесора для підвищення продуктивності алгоритмів, що прошиті. Кеш флеш-пам'яті також знижує енергоспоживання системи, бо зменшується частота звертань до флеш-пам'яті.

До *енергонезалежної підсистеми PSoC* можна віднести флеш-пам'ять, РПЗП-ЕС (EEPROM), дані у яку записуються по байтам, та енергонезалежні засоби налаштування. Забезпечується ємність флеш-пам'яті до 256 кбайт. Завантажувач дозволяє процесору перепрограмувати окремі блоки флеш-пам'яті. Для високонадійних додатків є можливість підключити частину флеш-пам'яті де знаходяться коди корекції помилок ECC. Вбудована потужна і гнучка функція захисту конфіденційної інформації користувача, що дозволяє пам'яті селективну заборону блоку від читання і запису. Крім того, обрані параметри конфігурації, наприклад швидкість завантаження та режим драйвера на контакті, зберігаються у енергонезалежній пам'яті. Це дозволяє актувати параметри налаштування відразу після режиму POR.

Підсистема виводів. Три різновиди виводів вх/вих мікросхеми PSoC є надзвичайно гнучкими. Всі виводи мають багато режимів роботи драйверів, які встановлюються у режимі POR. Напруга на контактах вх/вих може мати чотири рівня і подається через контакти (виводи) VDDIO.

Кожний GPIO має аналоговий вх/вих, драйвер LCD, модуль ємнісного давача CapSense, гнучку систему переривань, керування швидкістю наростання та

можливість вводу/виводу цифрових сигналів. Спеціальні вх/вих SIO у режимі роботи на вхід мають високий імпеданс, який зберігається коли мікросхема не працює або напруга підіймається вище за напругу живлення. Це робить виводи SIO ідеальними для використання з шиною I2C, коли мікросхема не працює, а інші пристрої підключені до неї. Виводи (контакти) SIO можуть також забезпечувати високий струм для різних застосувань, наприклад у якості драйвера LCD. Програмована величина порогу на вході SIO дозволяє виконувати йому функцію аналогового компаратора загального призначення. Апаратна реалізація інтерфейса USB забезпечує роботу з швидкісним USB, (full-speed) FS USB на вхід та вихід. Якщо ці контакти не використовуються інтерфейсом USB, то їх можна застосувати для цифрових пристроїв обмеженої функціональності та пристроїв програмування.

Система синхронізації. Мікросхема PSoC включає у себе гнучкі внутрішні генератори високої стабільності і точності. Основний внутрішній генератор (IMO) є базовим для системи синхронізації і має точність 1% на частоті 3 МГц. Генератор IMO можливо налаштувати для роботи на частоті від 3 МГц до 74 МГц. На його основі утворюються багато інших тактових частот, які потребують додатки. Систему тактових частот генерує PLL до 80 МГц від генератора IMO, від зовнішнього кварцового генератора або зовнішнього опорного генератора. Він також містить окремий, дуже малопотужний внутрішній низькочастотний генератор (ILO) для сплячого режиму та охоронного таймера. Зовнішній кварцовий генератор частотою 32,768 кГц використовується у годиннику реального часу RTC. Генератори тактових імпульсів разом з програмованими дільниками частоти тактових імпульсів забезпечують гнучкість системи синхронізації.

Система живлення. Сімейство CY8C58LP підтримує широкий діапазон напруги живлення від 1.7 В до 5,5 В. Це дозволяє операції з регульованими джерелами живлення, таких як $1,8\text{ В} \pm 5\%$, $2,5\text{ В} \pm 10\%$, $3,3\text{ В} \pm 10\%$, або $5,0\text{ В} \pm 10\%$, або безпосередньо від широкого спектра типів акумуляторів.

Інтегрований у схему імпульсний перетворювач дозволяє живлення від джерела з напругою від 0,5 В. Це дає можливість пристрою працювати безпосередньо від однієї батареї. Крім того, дає можливість застосувати імпульсний перетворювач для генерування інших напруг, потрібних для пристрою, наприклад, живлення 3,3 В для драйвера LCD. Вихід імпульсного перетворювача підвищення напруги є доступним на виводі VBOOST, що дозволяє іншим пристроям живитися від мікросхеми PSoC.

PSoC підтримує широкий спектр режимів низької потужності. До них відносяться сплячий режим з утриманням оперативної пам'яті – струм споживання 300 нА та сплячий режим з RTC – 2 мкА. Потужністю усіх основних функціональних блоків, включаючи периферійні цифрові та аналогові пристрої, можливо керувати незалежно від прошивки. Це забезпечує низьке енергоспоживання фонові обробки, коли деякі периферійні пристрої не використовуються, і загальний струм споживання становить 3,1 мА, коли процесор працює на частоті 6 МГц.

Для програмування, налагодження та трасування PSoC використовує інтерфейси JTAG – 4 дротовий або SWD – 2-дротовий. За допомогою цих стандартних інтерфейсів можливо програмувати і налагоджувати різні апаратні засоби від Cypress або сторонніх постачальників. Процесор Cortex-M3 для налагодження і трасування містить модулі FPB, DWT, ETM та ITM. Ці модулі володіють багатьма можливостями для вирішення складних проблем налагодження і трасування.

Бібліотека периферійних драйверів PSoC включає до 56 цифрових та 7 аналогових периферійних пристроїв, а також CapSense®, USB, Bluetooth Low Energy (BLE), USB та інтерфейси до серійної пам'яті та цифрових мікрофонів PDM–PCM. Прошивку для керування ними забезпечує бібліотека периферійних драйверів (PDL). PDL – це набір високоефективних драйверів сумісних із MISRA для периферійних пристроїв PSoC із комплексним пакетним документуванням для пошуку, який є HTML–базою для легкого перегляду в будь-якому браузері.

PSoC Creator робить використання PDL максимально спрощеним, автоматично додаючи необхідні драйвери у проект. Коли розробник додає компоненти до свого дизайну і будує проект, потрібний код додається автоматично і включає в себе підтримувані ресурси.

Маршрут програмування та внутрішні з'єднання. Система об'єктно – орієнтованого середовища розробки для проектів на PSoC значною мірою спрощує процедуру впровадження проектів. Це звільняє час на розробку, а також перекладає частину роботи над проектом на саме середовище розробки.

PSoC Creator надає можливість знизити вартість розробки, використовуючи єдиний інструмент розробки системи контролю та керування на базі готових до використання компонентів PSoC, одночасно прискорюючи термін виходу продукції на ринок.

7.5 Сімейство PSoC®6

Архітектура двоядерних МК дозволяє дизайнерам оптимізувати потужність і продуктивність одночасно (рис. 7.6). Власна технологія ультра низької потужності 40-нм SONOS Cypress дозволяє архітектурі PSoC 6 MCU використовувати лише 22 мкА/МГц і 15 мкА/МГц активної потужності на ядрах ARM Cortex-M4 і Cortex-M0 + відповідно. Завдяки динамічному масштабуванню напруги та частоти (DVFS) архітектура PSoC 6 MCU забезпечує високу продуктивність та найменшу споживану потужність. Двоядерна архітектура дозволяє оптимізувати енергоспоживання системи, де допоміжне ядро може використовуватися для розвантаження основного ядра, яке може знаходитися у сплячому режимі. Архітектура PSoC 6 MCU інтегрує апаратне середовище Trusted Execution Environment (TEE) з можливістю безпечного завантаження та вбудованим безпечним сховищем даних для захисту прошивки.

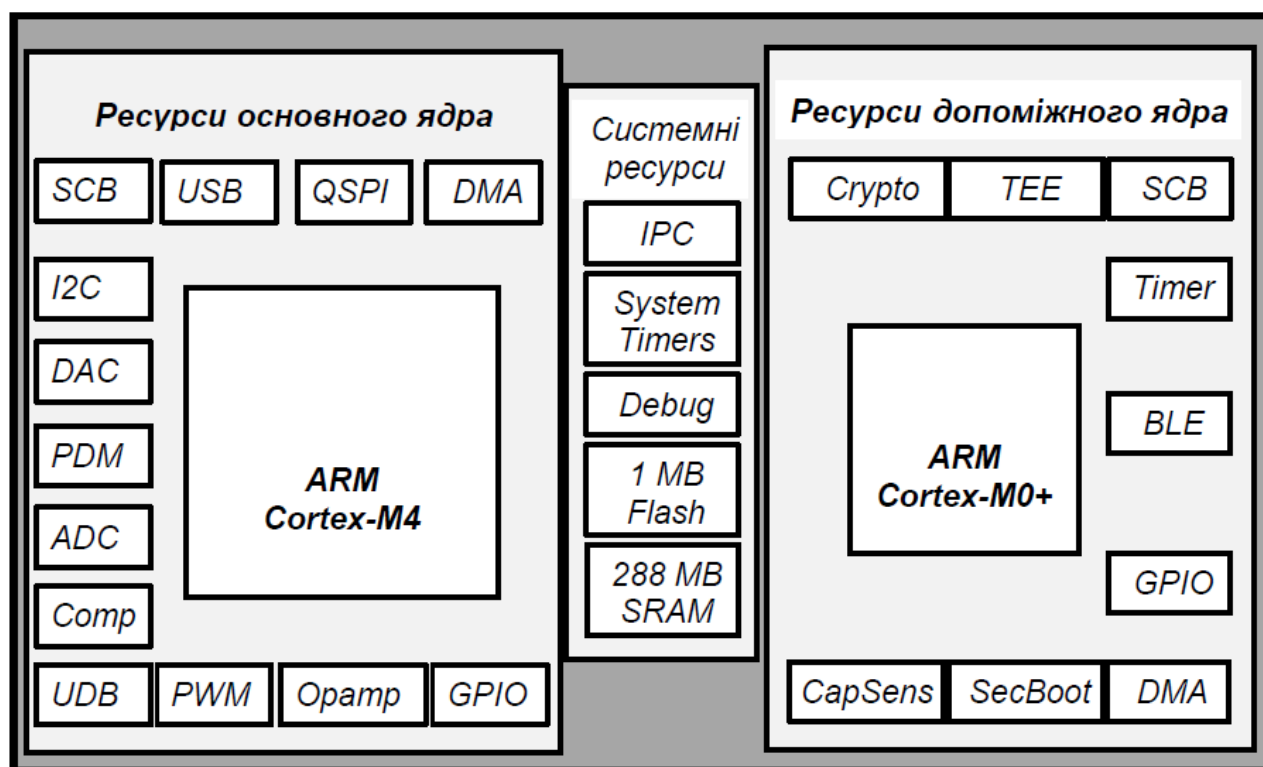


Рисунок 7.6 – Спрощена архітектура мікросхеми сімейства PSoC 6 з двома процесорними ядрами

МК реалізує широкий набір галузевих симетричних і асиметричних криптографічних алгоритмів в інтегрованому апаратному співпроцесорі, призначеному для розвантаження основного процесора за виконанням обчислювальних завдань. Архітектура підтримує декілька одночасних захищених середовищ без необхідності використання зовнішньої пам'яті або захищених елементів і пропонує масштабовану безпечну пам'ять для декількох незалежних підходів до безпеки, які визначаються користувачем. Програмні периферійні пристрої можуть використовуватися для створення власних аналогових інтерфейсів користувача (Active front end – AFE) або цифрових інтерфейсів для інноваційних компонентів системи, а також для підтримки гнучких варіантів бездротового підключення. Архітектура підтримується інтегрованим середовищем дизайну Cypress PSoC Creator (IDE).

Мікросхеми PSoC 6 орієнтовано на вживання у сфері Інтернет технологій IoT. IoT (Internet of Things), або інтернет речей – це мережа зв'язаних через інтернет об'єктів, які збирають дані та обмінюються даними, що надходять з вбудованих сервісів. Пристрої, що входять до інтернету речей – різноманітні автономні пристрої, які підключено до інтернету і за якими можливо слідкувати і/або управляти звіддалі.

Екосистема IoT, або інтернета речей, – усі компоненти, які дозволяють користувачам приєднувати свої пристрої IoT (пульти управління, панелі інструментів, аналітика, мережі, шлюзи, зберігання даних, безпека).

Фізичний рівень – апаратне забезпечення, яке використовується у IoT – пристрої, включаючи сенсори та устаткування мереж.

Мережевий рівень відповідає за передавання даних, що зібрані на фізичному рівні, до різноманітних пристроїв.

Рівень додатків залучає протоколи і інтерфейси, які використовують пристрої для ідентифікації та зв'язку один з іншим.

Пульти управління дозволяють людині використовувати IoT-пристрої, з'єднуючись з ними і контролюючи їх через панель інструментів (мобільний додаток). До пультів управління відносяться смартфони, планшети, ПК, розумні годинники, телевізори та інше.

Панелі інструментів забезпечують відображення інформації об екосистемі IoT для користувачів, що дозволяє управляти екосистемою IoT.

Аналітика — програмні системи, які аналізують дані, отримані від IoT-пристроїв. Наприклад прогнозування технічного обслуговування.

Сфера застосування IoT дуже велика: виробництво, транспорт, сільське господарство, оборона, логістика, банки, лікарні та інше. Це диктує підвищені вимоги до захисту даних пристроїв, що підключені. Вимоги безпеки забезпечуються апаратними та програмними засобами.

На апаратній основі розділено ресурси основного ядра та допоміжного ядра. Допоміжне ядро Arm Cortex-M0 + створює безпечне середовище для виконання надійних додатків. Воно зв'язується з Arm Cortex-M4, яке підтримує базове середовище виконання задачі через канали IPC (рис. 7.7).

Програмно розділені кореневі системи: операції кореневої системи та надійні додатки далі виконуються в ізольованому середовищі виконання, захищаючи цілісність та конфіденційність. Довіренні додатки ізолюванні один від одного: кожний довірений додаток в межах ізольованого середовища виконання може бути відділений один від одного, зменшуючи доступність до вірусної або хакерської атаки.

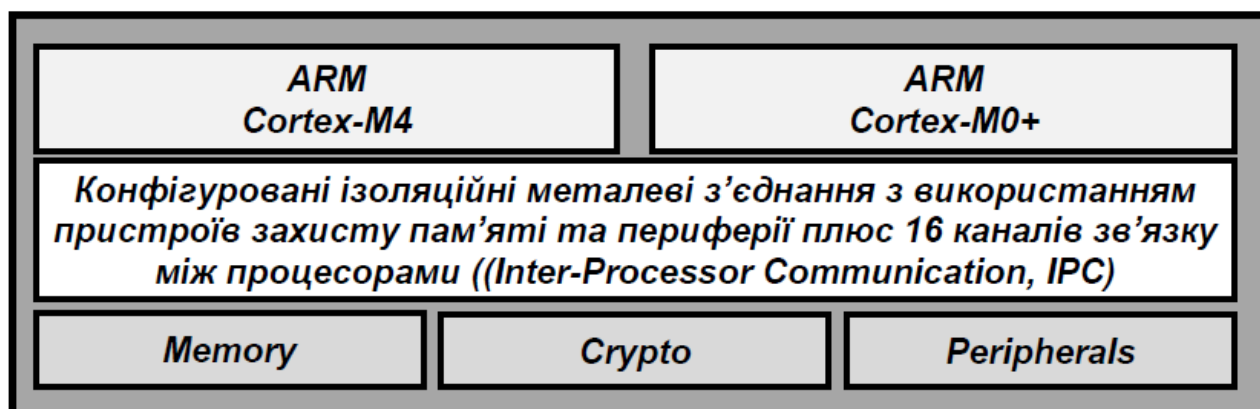


Рисунок 7.7 – Ізоляційні металеві з'єднання у середині мікросхеми Cypress PSoC 6 з двома процесорними ядрами

7.6 Середовище розробки проектів PSoC® Creator™.

Розробка аналого-цифрового проекту в середовищі PSoC® Creator™. PSoC Creator – це інтегроване середовище проектування (IDE), яке забезпечує одночасне редагування апаратного та програмного забезпечення, компіляцію та налагодження систем PSoC та FM0+. Програми створюються за допомогою схематичного (об'єктно – орієнтованого) середовища, що використовує понад 150 розроблених, попередньо перевірених периферійних компонентів.

Компоненти PSoC – це аналогові та цифрові периферійні пристрої, представлені символом, який користувачі встановлюють у свої проекти та налаштовують на відповідність до вимог свого проекту. Кожен компонент у бібліотеці змішаного сигналу налаштовується діалоговим вікном налаштування та містить повний набір бібліотек інтерфейсу програмування (API (*Application Programming Interface*) – *це інтерфейс програмування, або інтерфейс створення додатків, тобто, API – це готовий код для спрощення праці розробника*). Після налаштування всіх периферійних пристроїв, прошивку можна компілювати та налагоджувати в PSoC Creator або експортувати до сторонніх IDE, таких як IAR Embedded Workbench®, Arm® Microcontroller Development Kit та Eclipse™.

PSoC Creator можна завантажити з офіційного сайту виробника. На момент написання, остання версія середовища розробки – PSoC Creator 4.2 SP.

Запуск проекту з нуля може бути складним, тому PSoC Creator надає багато корисних початкових шаблонів, які дозволяють спростити роботу розробника (рис.7.8).

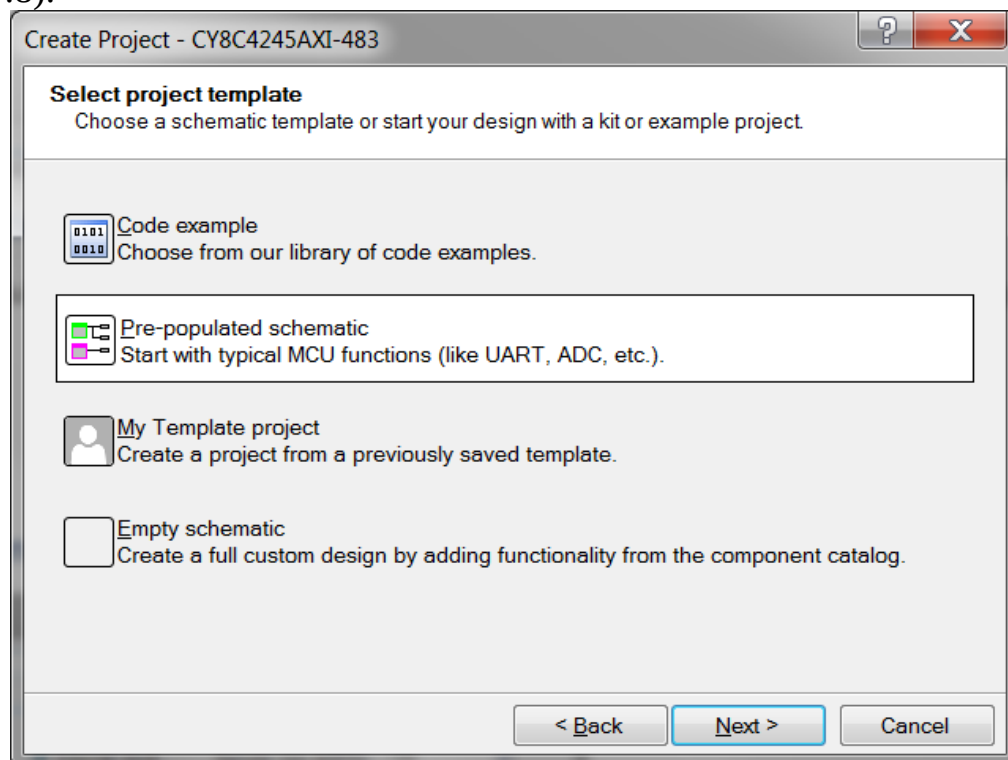


Рисунок 7.8 – Вибір шаблону для проекту

Сама розробка проекту починається із створення схеми. Для цього із каталогу компонентів (Component Catalog) вибирається потрібний компонент і переноситься на схему.

На рисунку 7.9 цифрами відзначено 1—дерево проекту з файлами *.c і *.h, під цифрою 2 власне шар на якому розміщені ті компоненти які використовуються в проекті. Бібліотечні компоненти розміщуються в панелі праворуч і вибираються в залежності від потреб розробника. Далі необхідно з'єднати вибрані компоненти між собою, для цього потрібно скористатися відповідним значком що розташований праворуч від розміщених на робочому полі компонентах. Внизу (3) розташована консоль що дозволяє контролювати помилки не тільки ті які допущені в ході програмування але і ті які виникають в файлі верхнього рівня ієрархії (файл TopDesign.cysch). В цій консолі знаходяться посилання на конкретні помилки – будь то відсутність з'єднання, або невірно позначені вхід або вихід. Тобто система контролює користувача і не дає помилитися.

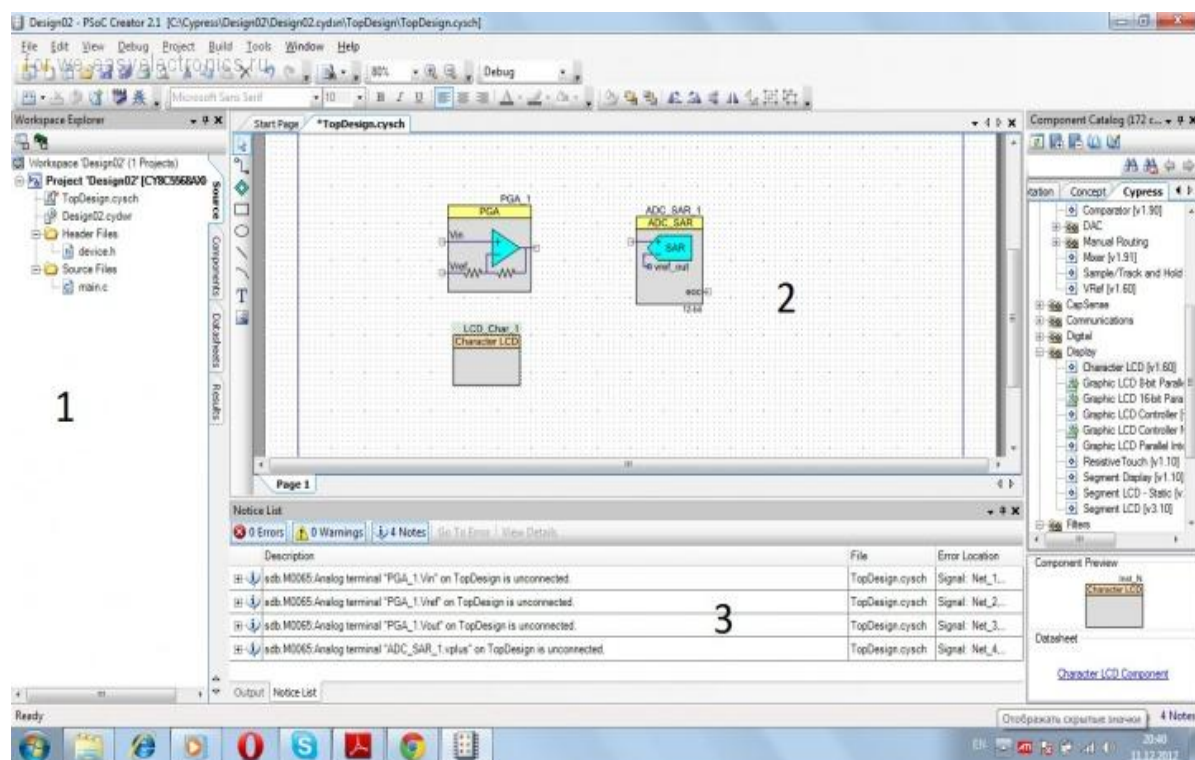


Рисунок 7.9 – Створення схеми із каталогу компонентів

Наступний етап – підключення виводів і налагоджування пристроїв вводу – виводу. Підключаємо програмований підсилювач ОП – PGA_1. На схемі для цього справа знаходиться вкладка Ports & Pins. Неєхидно вибрати аналоговий вхід, встановити його навпроти входу ОП та з'єднати їх між собою. Отримаємо наступне – рисунок 7.10.

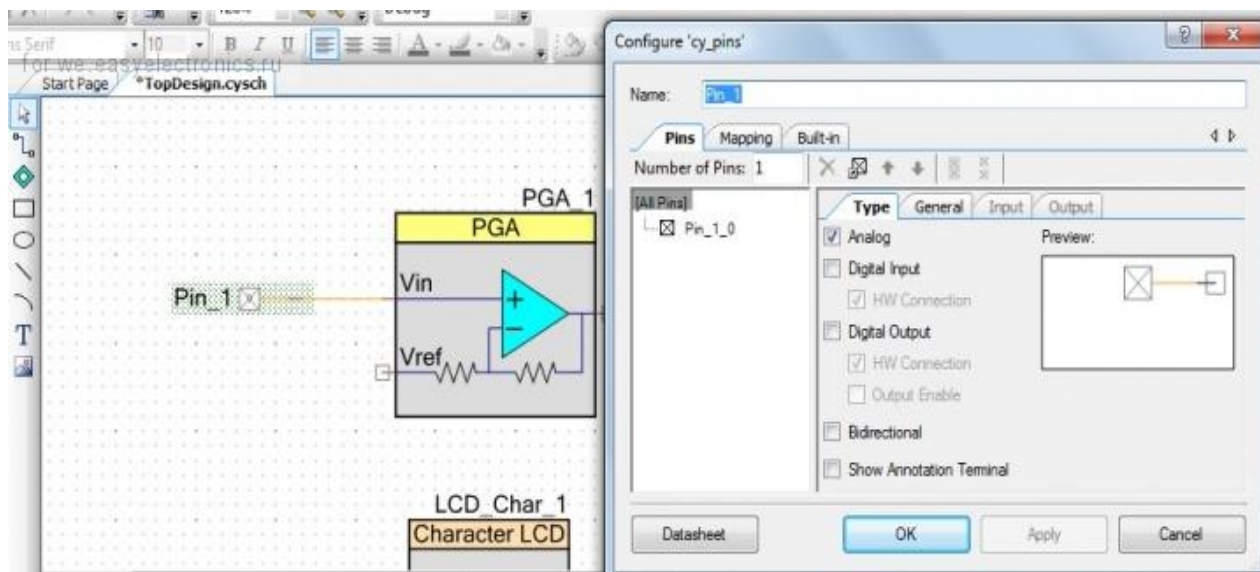


Рисунок 7.10 – Підключення компонентів схеми і налагоджування пристроїв вводу – виводу

На панелі підключення виводів є багато варіантів вибору типів для цифрових входів – виходів, включно зі зміною типу CMOS або TTL.

Відкрити панель настройки ОП Тут можна встановити коефіцієнт підсилення, вибрати опорний сигнал, подивитися залежність коефіцієнту підсилення від частоти. Для збереження змін необхідно натиснути кнопку OK і Apply, в результаті отримуємо налаштований ОП. Щоб переглянути характеристики вибраної функції потрібно натиснути кнопку Datasheet. Там знаходяться описи всіх API, що застосовані в PSoC Creator. В описі вказані функції необхідні для запуску обраного модуля притому можна проводити настройку як вручну так і за допомогою вбудованих бібліотечних функцій.

Вихід ЦАП можна також аналоговий підключаємо його до виходу. Розглянемо ЦАП, їх в даному контролері три: два послідовного наближення, один дельта сигма. Перші АЦП – 12 розрядні 630кSps, дельта сигма АЦП – 16 розрядний, але його можна налаштувати до 20 розрядного. Після цього потрібно зайти у властивості компонента (через контекстне меню) та змінити, за необхідності, параметри на потрібні. На рисунках 7.11, 7.12 показаний приклад налаштування аналогового-цифрового перетворювача.

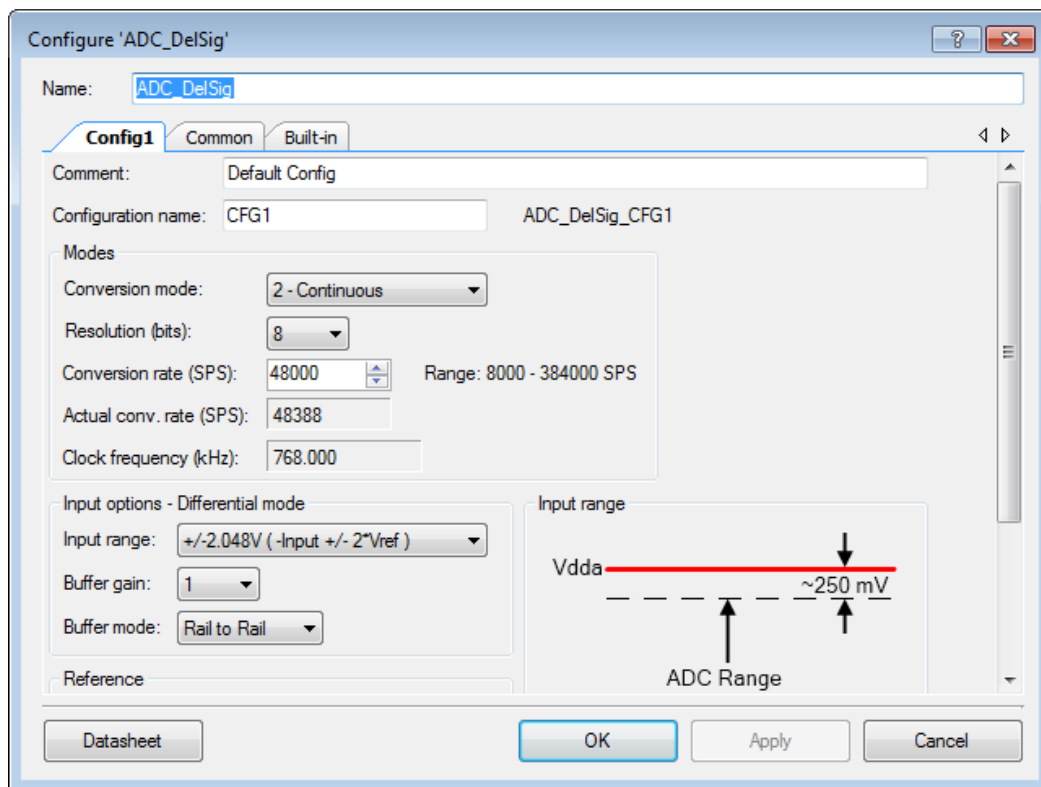


Рисунок 7.11 – Налаштування дельта-сигма АЦП

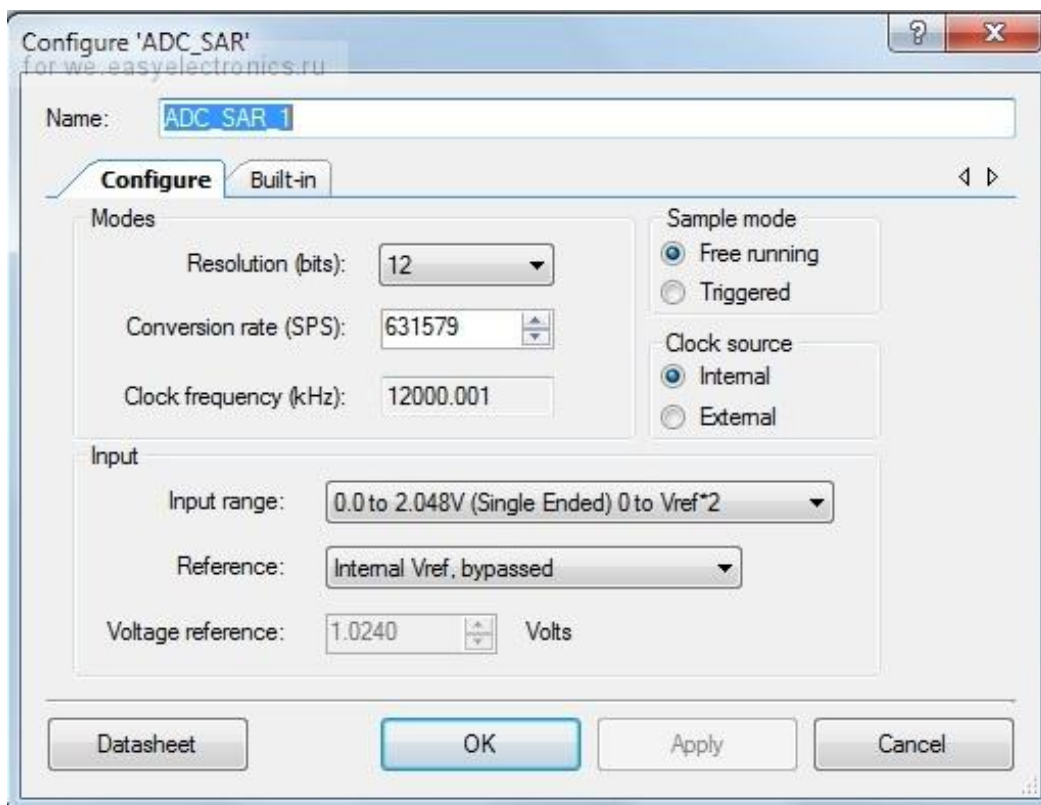


Рисунок 7.12 – Налаштування АЦП послідовного наближення

На наступному кроці потрібно з'єднати компоненти між собою створивши тим самим свою повну схему (рис.7.13). Існують основні компоненти (Cypress Component Catalog) за допомогою яких конфігурується робота пристрою та допоміжні (на схемі відображаються синім кольором) які призначені для відображення з'єднань між виводами мікросхеми та зовнішніми елементами (наприклад показати що до входу на який підключено ADC, підключено резистор). Ці компоненти називаються “Off-Chip” компонентами, тобто, ті, що знаходяться за межами корпусу. Допоміжні компоненти ігноруються середовищем та компілятором і призначені тільки для покращення інформативності схеми. Якщо компоненти прості, наприклад компаратори, лічильники, то після компіляції можна завантажувати проект в мікросхему.

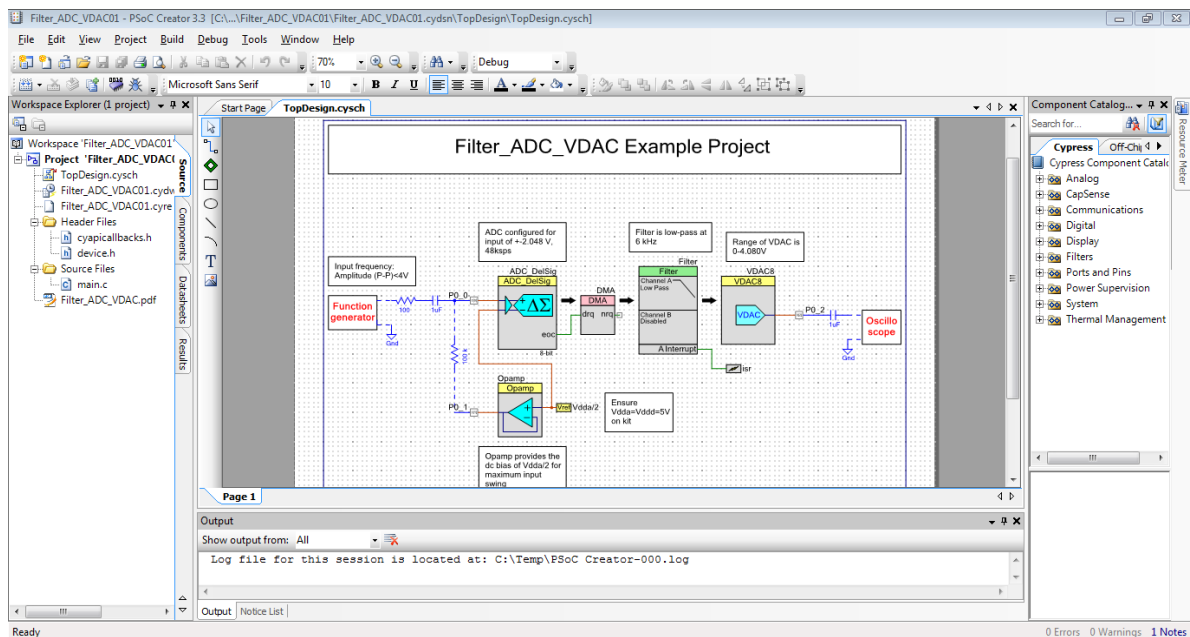


Рисунок 7.13 – Введення повної схеми проекту

Якщо використані більш складні компоненти (ADC, DMA.), потрібно провести їх ініціалізацію, а також долучити їх в main.c. Це відбувається за допомогою API функцій, які доступні після того як розробник зробив генерацію проекту Build – Generate Application. Після цього стають доступні файли *.h в яких є потрібні для роботи API функції. Використовуючи їх можна створювати свої програми (рис. 7.14).

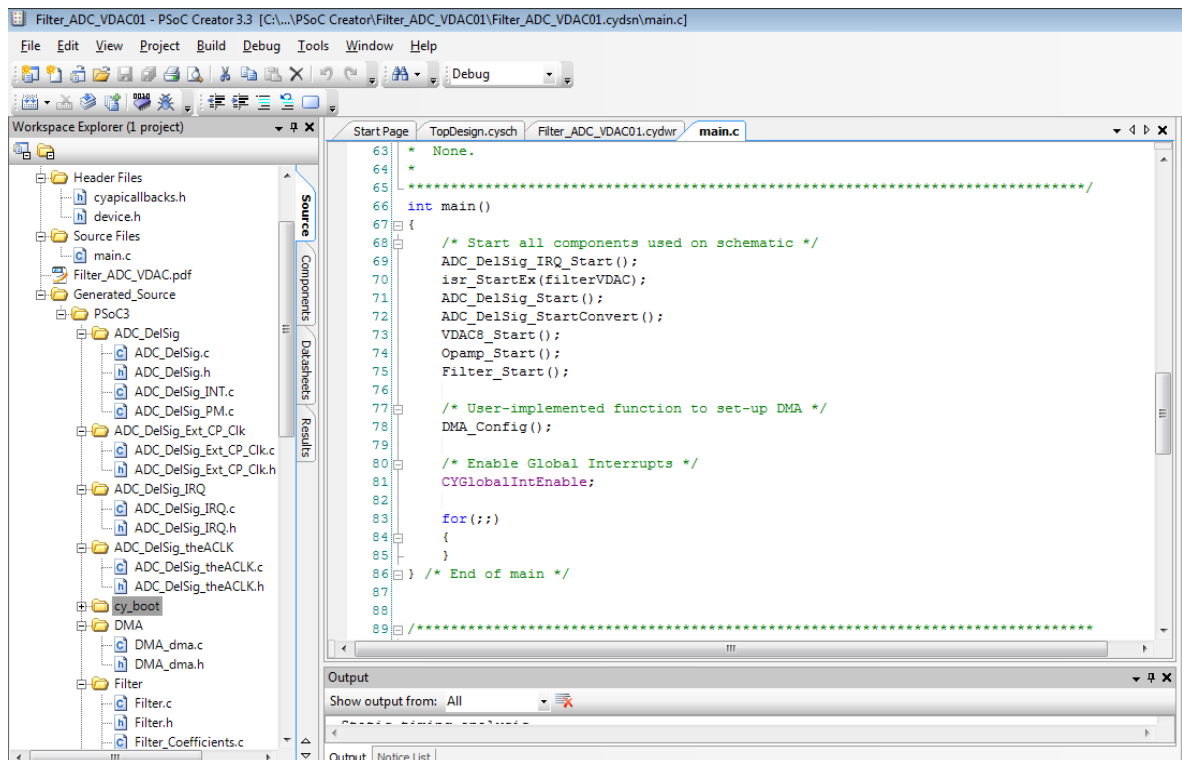


Рисунок 7.14 – Файл main.c після ініціалізації компонентів пристрою.

Після з'єднання і налаштування компонентів, потрібно підключити сконфігуровану схему до портів вводу/виводу (рис. 7.15). В PSoC периферія не закріплена за окремими виводами, тому це дозволяє спростити друковану плату. Переходимо в меню налаштувань портів, для цього запускаємо файл з розширенням *.cydwr, та призначаємо входи і виходи схеми на порти вводу/виводу.

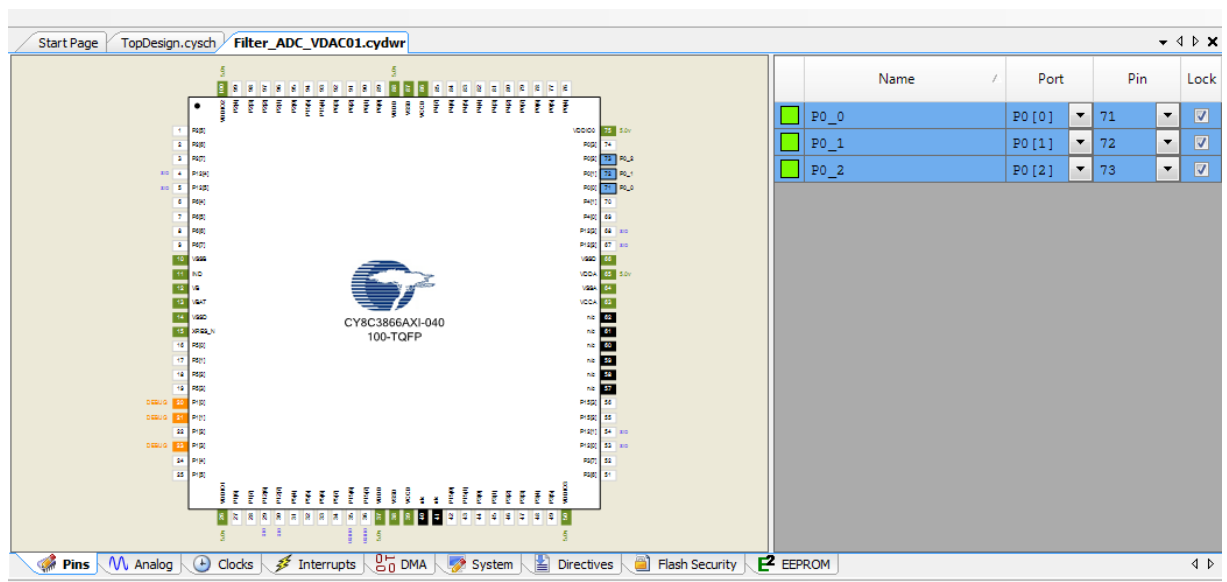
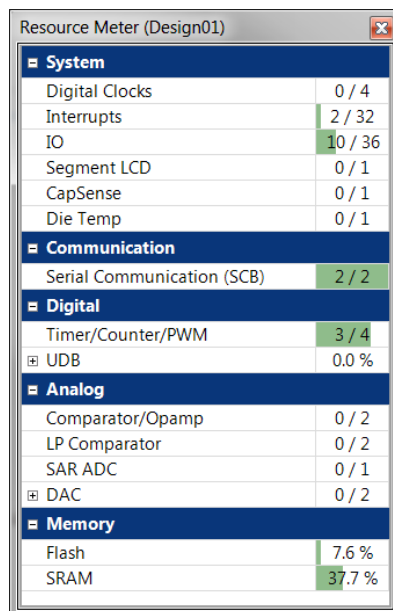


Рисунок 7.15 – Призначення входів і виходів схеми до портів вводу/виводу

Як тільки все налаштовано, написана програма, проект успішно пройшов компіляцію, можна завантажувати *.hex файл в мікросхему.

В процесі створення проекту, в PSoC Creator оновлюється дисплей Resource Meter, який показує компоненти та засоби, які використовуються в дизайні. Розробник може бачити, скільки ресурсів задіяне, і отримати раннє попередження про надмірне використання ресурсів (рис.7.16).



Resource Meter (Design01)	
System	
Digital Clocks	0 / 4
Interrupts	2 / 32
IO	10 / 36
Segment LCD	0 / 1
CapSense	0 / 1
Die Temp	0 / 1
Communication	
Serial Communication (SCB)	2 / 2
Digital	
Timer/Counter/PWM	3 / 4
UDB	0.0 %
Analog	
Comparator/Opamp	0 / 2
LP Comparator	0 / 2
SAR ADC	0 / 1
DAC	0 / 2
Memory	
Flash	7.6 %
SRAM	37.7 %

Рисунок 7.16 – Вікно попередження про використання ресурсів

7.7 Приклади розробки елементів систем на PSoC.

Використання компонентів PSoC Creator для управління модулем L9110S для керування двигунами постійного струму. Модуль L9110S дозволяє дискретно включати/виключати двигун постійного струму, а також змінювати напрям його обертання. Відсутня в модулі функція плавного регулювання швидкістю обертання двигуна реалізується за допомогою PSoC. Основу модуля складають дві мікросхеми HG7881 кожна з яких є Н-міст (рис.7.17).

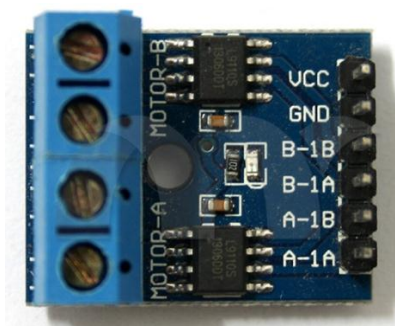


Рисунок 7.17 – Модуль керування двигунами постійного струму L9110S

Призначення виводів драйвера:

VCC – живлення двигунів, 2.5 – 12V;

GND – силова та сигнальна земля;

B-1B – двигун B вхід B;

B-1A – двигун B вхід A;

A-1B – двигун A вхід B;

A-1A – двигун A вхід A.

Таблиця 7.2 – Стани двигуна підключеного до драйвера (одного каналу драйвера)

A-1A	A-1B	Стан двигуна
0	0	відключений
0	1	обертання за годинниковою стрілкою
1	0	обертання проти годинникової стрілки
1	1	відключений

Із таблиці 7.2 видно що в цьому модулі відсутні виводи керування швидкістю (як правило PWM виводи). Щоб можна було керувати швидкістю, замість логічної «1» на вхід потрібно подати pwm сигнал. Проект керування швидкістю розроблений в PSoC Creator. Його основу складає схема, (рис. 7.18), на котрій показано тільки один канал драйвера, щоб мати можливість керувати двома двигунами, додається другий канал.

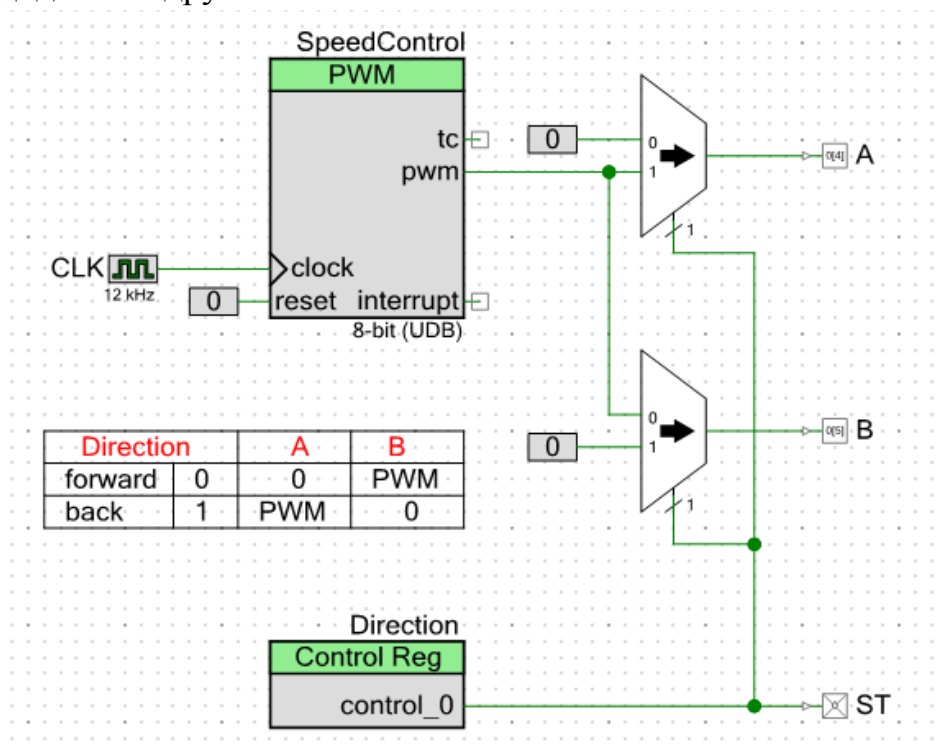


Рисунок 7.18 – Схема керування двигунами постійного струму зібрана в PSoC Creator

Практична частина. Основу проекту складає схема зібрана в PSoC Creator, яка складається з таких основних компонентів:

1. **PWM** – компонент призначений для генерування PWM сигналу. На його вхід подається тактовий сигнал потрібної частоти.
2. **Demultiplexer** – демультимплексор, призначений для комутації кількох сигналів на один вихід.
3. **Control Register** – дає можливість програмі впливати на схему, тобто запис в цей регістр з програми, змінює вихідний стан цього компоненту.

Схема працює наступним чином: компонент **PWM** генерує сигнал, коефіцієнт заповнення (duty cycle) якого задається з програми. Далі цей сигнал подається на входи демультимплексорів. На інші входи демультимплексорів подається логічний «0». Також на демультимплексор подається сигнал перемикавання з **Control Register**, значення якого змінюється в програмі. Цей сигнал задає напрям обертання двигуна. При перемиканні демультимплексорів, на виходах (A, B) з'являється сигнал логічного «0» та ШІМ сигнал. Додатковий вивід ST (стан) – для можливості підключення до цього виводу зовнішніх елементів схеми. Сигнал на цьому виводі повторює сигнал з **Control Register**. Сигнали A, B, ST виведені на виводи PSoC (рис 7.19), при потребі їх легко змінити на інші.

	Name	Port	Pin	Lock
☒	A	P0[4]	24	☒
☒	B	P0[5]	25	☒
☒	ST	P0[0]	19	☒

Рисунок 7.19 – Стан виводів портів пристрою керування

Нижче приведено текст програми керування модулем двигуна.

```
#include "project.h"

#define FORWARD (0u)
#define BACK    (1u)

int main(void)
{
    CyGlobalIntEnable; /* Enable global interrupts. */
    SpeedControl_Start();
    for(;;)
    {
```

```

SpeedControl_WriteCompare(200); /* встановлюємо значення PWM
(максимальне значення 255) */
Direction_Write(FORWARD);
CyDelay(5000); /* пауза 5 секунд */
Direction_Write(BACK);
CyDelay(5000); /* пауза 5 секунд */
}
}

```

Використання компонентів PSoC Creator для вимірювання напруги та перетворення отриманих даних і вивід інформації на ПК дисплей.

Практична частина. Основу проекту складає схема зібрана в PSoC Creator.

Процедура створення проекту показано вище. Після створення перейти на *TopDesign* проекту і додати з бібліотеки компонентів необхідні та змінити їхні властивості:

1. **Analog Pin** – іменувати Vin та дозволити зовнішні з'єднання щоб можна було підключити off-line компоненти для кращої інформативності (рис. 7.15).
2. **Sequencing SAR ADC** – ADC, використовуємо 1 канал ADC, звичайний режим роботи з усередненням (рис. 7.20).

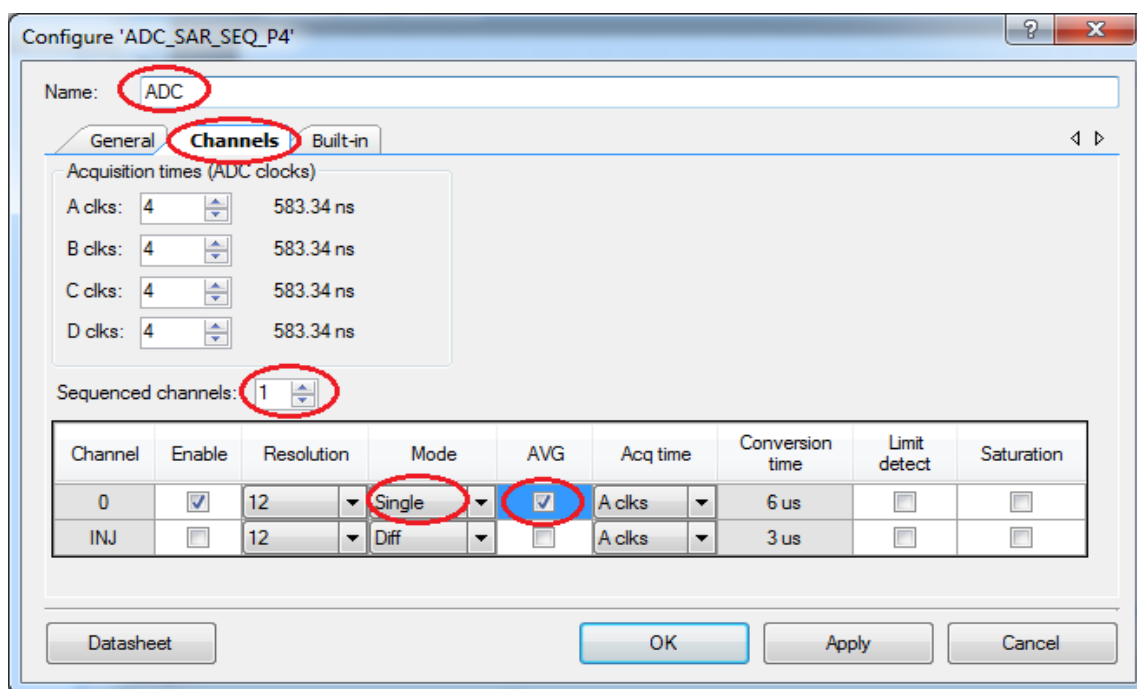


Рисунок 7.20 – Початкові налаштування АЦП

На вкладці *General* змінити частоту дискретизації (sample rate), вибрати джерело опорної напруги (V_{ref}), задати режим вимірювання відносно опорної напруги, оскільки вимірюваний сигнал одно полярний (від 0 до V_{ref}) – результат повинний бути без урахування знаку (Unsigned), вихідний результат буде усередненим за 64 вибірки (рис. 7.21).

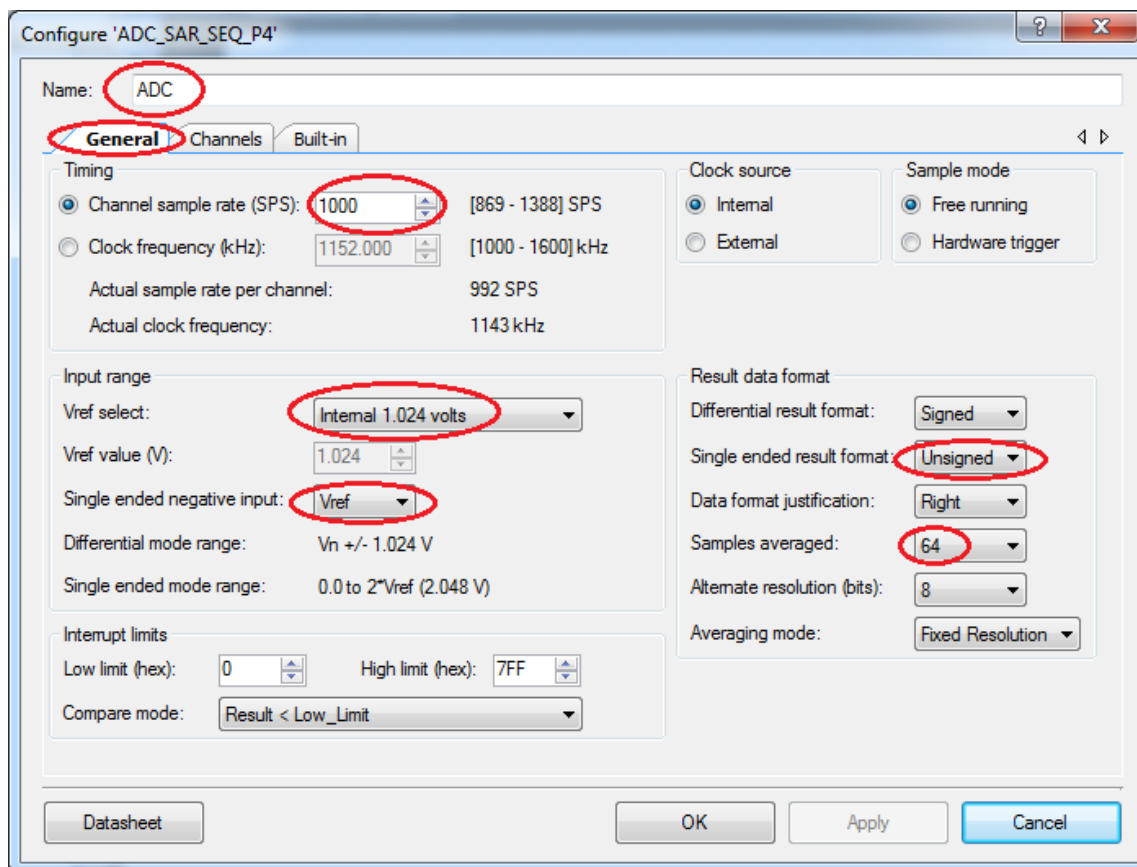


Рисунок 7.21 – Налаштування режиму роботи ЦАП

3. Character LCD – РК дисплей (LCD) (рис. 7.22)

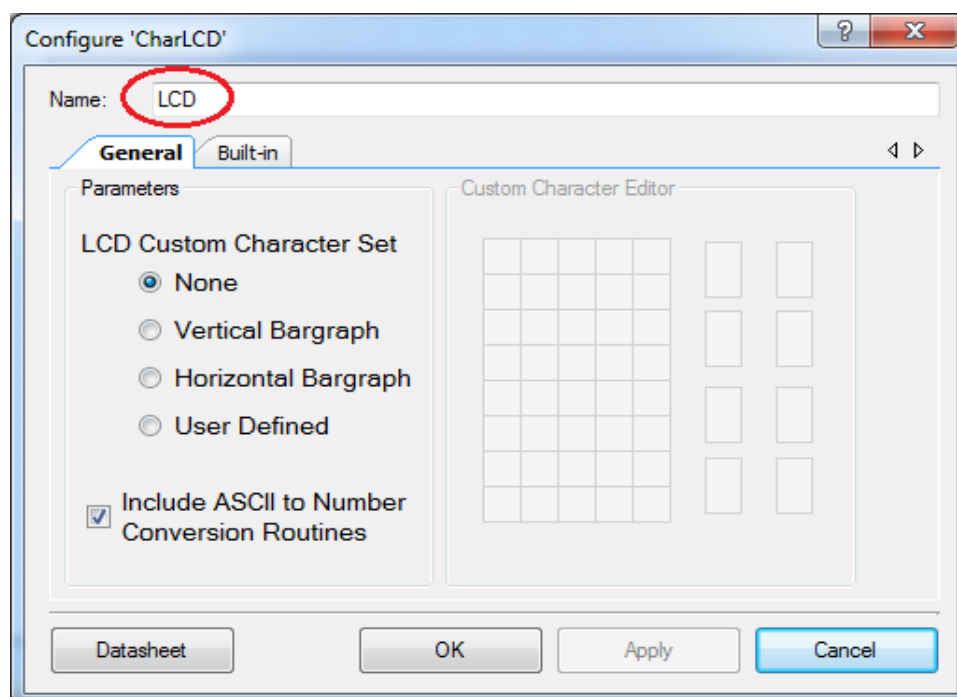


Рисунок 7.22 – Вибір типу РК дисплею

Всі компоненти налаштовані, можна переходити до створення схеми, тобто з'єднання компонентів між собою. В кінці має бути наступна схема. Компоненти “Vdd”, “Vss” та “R1” не беруть участі в роботі схеми, а використовуються тільки для інформативності (рис. 7.23).

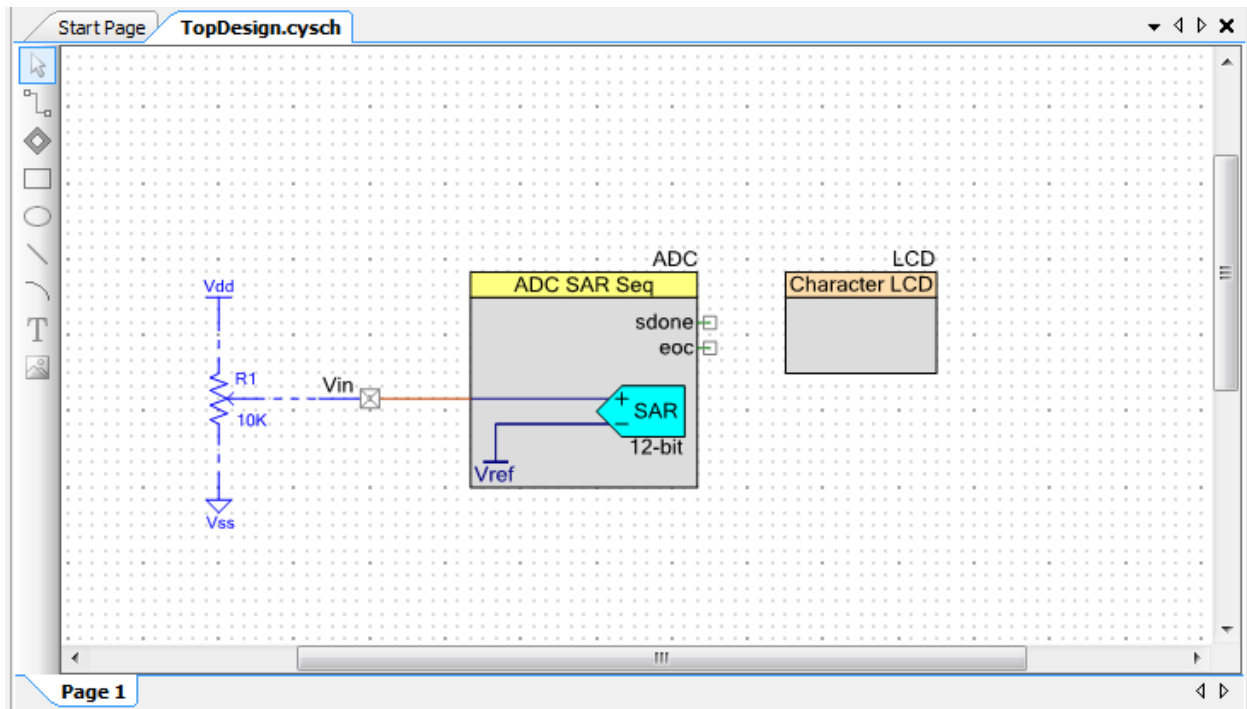


Рисунок 7.23 – Загальна схема пристрою в PSoC Creator

Далі потрібно налаштувати відповідність між позначеними виводами отриманої схеми та виводами мікроконтролера, при цьому виділити порти для підключення периферійних (“Off-Chip”) компонентів: потенціометр підключений до P2[0], а дисплей до P2[7:1] (рис. 7.24).

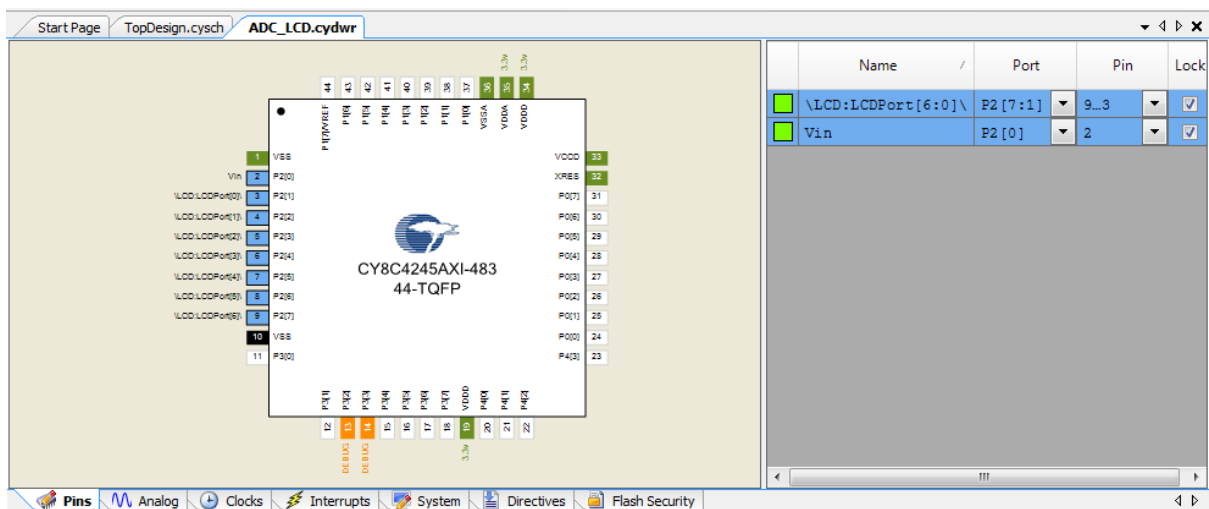


Рисунок 7.24 – Налаштування пристроїв вводу-виводу

Нижче приведено текст програми для вимірювання напруги та перетворення отриманих даних і вивід інформації на РК дисплей.

```

#include <project.h>

0  #define POT_ACD_CHANNEL (0) /* ADC channel 0 */
1
2  int main()
3  {
4      uint16 inResCounts; /* ADC result in counts */
5      uint16 mVolts;      /* ADC result in mVolts */
6
7      ADC_Start();
8      ADC_StartConvert();
9      LCD_Start();
10
11     CyGlobalIntEnable; /* Enable global interrupts. */
12
13     for(;;)
14     {
15         /* Get ADC result and convert to mV */
16         inResCounts = ADC_GetResult16(POT_ACD_CHANNEL);
17         mVolts      = ADC_CountsTo_mVolts(POT_ACD_CHANNEL,
18 inResCounts);
19
20         /* Display value to LCD */
21         LCD_Position(0,0);
22         LCD_PrintNumber(mVolts);
23         LCD_PrintString("mV ");
24     }
25 }
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```

Короткий опис програми: після ініціалізації і запуску АЦП та РК дисплею, відбувається циклічне читання результату АЦП, перетворення його значення в mV за допомогою API функції:

ADC_CountsTo_mVolts()

та відображення цього значення на дисплеї. Як видно більшу частину розробки проекту складає проста конфігурація роботи компонентів. Можна покращити програму, зробивши читання і вивід на дисплей наприклад з періодичністю 1с, використавши для цього переривання апаратного таймеру.

Використання сенсорні кнопки для зміни кольору світлодіоду в залежності від положення пальця на сенсорному повзунку.

Робота із сенсорними кнопками побудована на технології **CapSense**, що дозволяє створювати пристрої з мінімальним використанням зовнішніх компонентів. Ця технологія призначена для заміни механічних кнопок в пристроях, цим самим зробити систему надійнішою та покращити дизайн пристроїв. Для роботи з сенсорними кнопками в середовищі розробки є компонент **CapSense CSD**.

Практична частина. Процес створення проекту описано в попередніх прикладах. Створюємо проект для PSoC 4200. На *TopDesign* добавимо компоненти: CapSense CSD, PWM, Clock та два Digital Output Pin, також добавимо *Off-chip* компоненти для інформативності. Вийде наступна схема (рис.7.25)

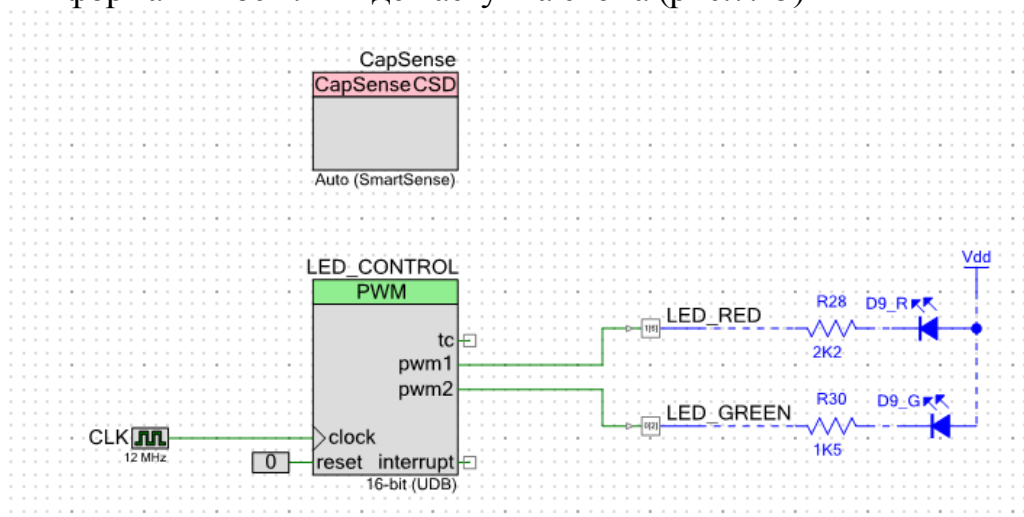


Рисунок 7.25 – TopDesign проекту сенсорної кнопки

Налаштування компонентів:

- **Digital Output Pin** та **CLK (12MHz)**: налаштування залишається без змін;
- **CapSense CSD**: налаштування на вкладці General залишаємо без змін. На вкладці *Widgets Config* додаємо лінійний слайдер (повзунок) – Add linear slider – > Linear sliders (рис.7.26).

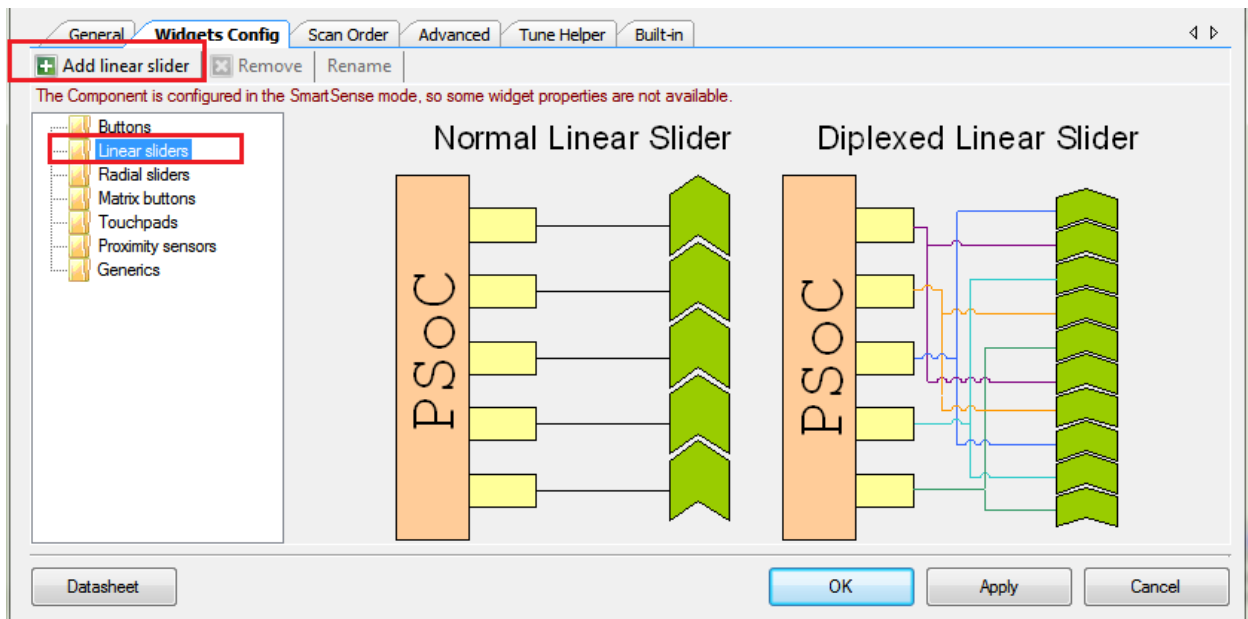


Рисунок 7.26 – Налаштування пристрою CapSense

Параметр “*API Resolution*” визначає діапазон в якому буде змінюватись значення повзунка (роздільна здатність), задаємо значення від 0 до 100, також залишаємо всі інші параметри без змін. На цьому налаштування **CapSense CSD** завершено (рис.7.27).

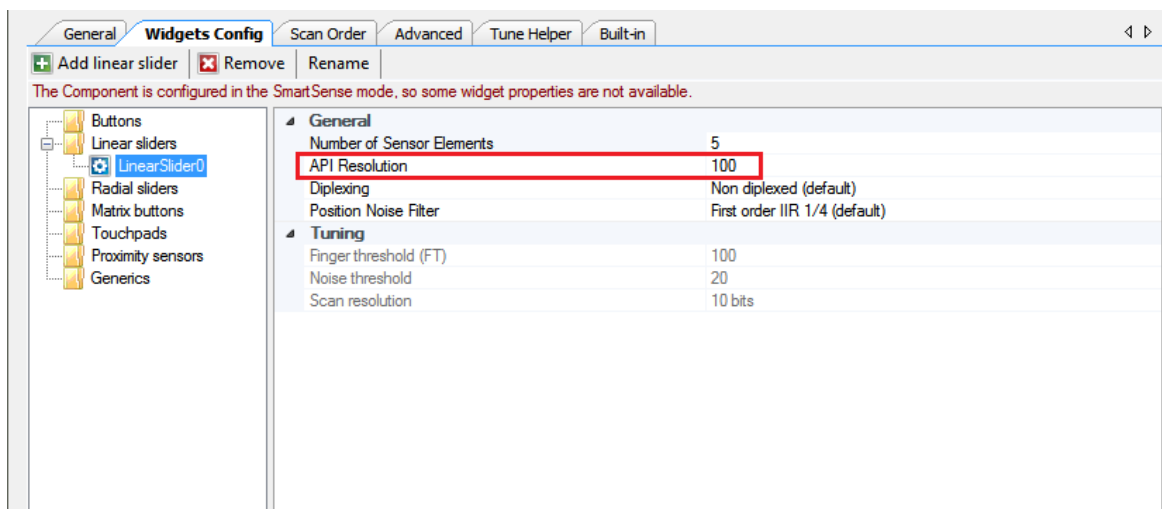


Рисунок 7.27 – Налаштування параметру роздільної здатності

PWM(LED_CONTROL): На платі використовується RGB світлодіод, тому зміна яскравості кожного вбудованого світлодіоду, змінює загальний колір. Налаштуємо **PWM** компонент для керування двома світлодіодами таким чином, що вихідний сигнал на кожному каналі будуть протифазні, це дозволить встановленням 0 – задавати яскравість одного світлодіоду 0%, а іншого 100% і навпаки. Для цього потрібно встановити параметр CMP Type 2 в функцію “Greater” (рис. 7.28).

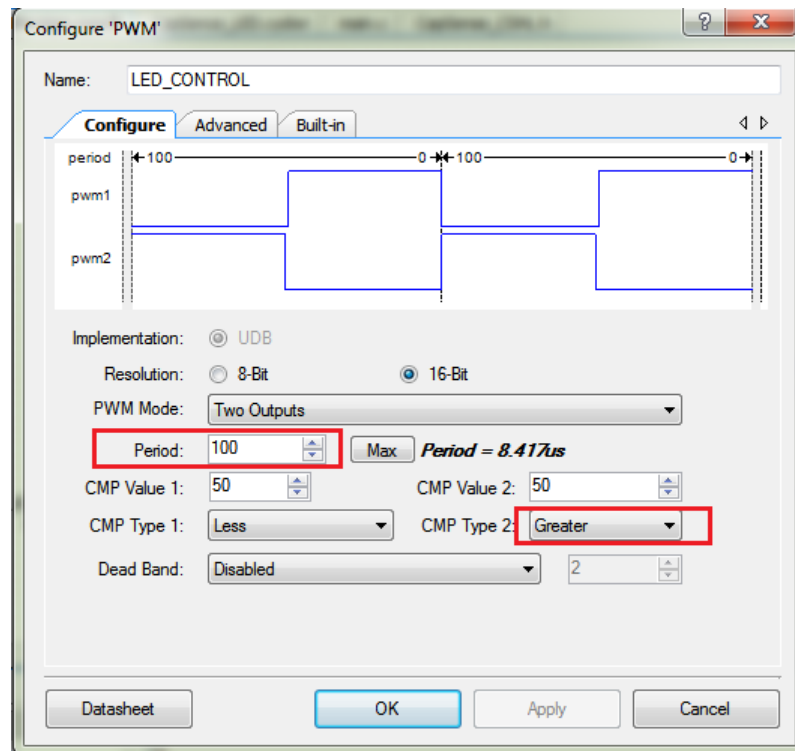


Рисунок 7.28 – Налаштування пристрою контролю світло діодів.

Оскільки **CapSense** видаватиме значення в діапазоні 0...100, налаштуємо період **PWM** також 100 (значення яке відповідає 100% заповнення імпульсу).

Налаштуємо відповідні з'єднання між схемою та портами та пристроями вводу-виводу мікроконтролера: **CapSense** підключений до P1[1:5], світлодіоди до P0[2] та P1[6] (рис.7.29).

	Name	Port	Pin	Lock
	\CapSense:Cmod\ (Cmod)	P4 [2]	22	☒
	\CapSense:Sns[0]\ (LinearSlider0_e0__LS)	P1 [1]	38	☒
	\CapSense:Sns[1]\ (LinearSlider0_e1__LS)	P1 [2]	39	☒
	\CapSense:Sns[2]\ (LinearSlider0_e2__LS)	P1 [3]	40	☒
	\CapSense:Sns[3]\ (LinearSlider0_e3__LS)	P1 [4]	41	☒
	\CapSense:Sns[4]\ (LinearSlider0_e4__LS)	P1 [5]	42	☒
	LED_GREEN	P0 [2]	26	☒
	LED_RED	P1 [6]	43	☒

Рисунок 7.29 – Налаштування пристроїв вводу-виводу мікроконтролера

Текст програми:

```
#include "project.h"
```

```

uint16 sliderPosition = 0;

int main(void)
{
    CyGlobalIntEnable; /* Enable global interrupts. */

    LED_CONTROL_Start();

    CapSense_Start();
    CapSense_InitializeAllBaselines();
    CapSense_ScanEnabledWidgets(); /* Do initial scan */

    for(;;)
    {
        if(!CapSense_IsBusy())
        {
            sliderPosition =
CapSense_GetCentroidPos(CapSense_LINEARSLIDER0__LS);
            CapSense_UpdateEnabledBaselines();
            CapSense_ScanEnabledWidgets();
        }

        if(0xFFFF != sliderPosition) /* 0xFFFF - if there is no touching to
the slider */
        {
            LED_CONTROL_WriteCompare1(sliderPosition);
            LED_CONTROL_WriteCompare2(sliderPosition);
        }
    }
}

```

Алгоритм роботи програми наступний: спочатку проводиться ініціалізацію ШІМ-контролера – **PWM** та сенсорних – **CapSense** компонентів, після цього починається перше сканування сенсорних кнопок, далі в циклі передбачена перевірка закінчення циклу роботи **CapSense**, якщо так, визначається положення пальця на сенсорному елементі і запускається новий цикл сканування, після цього відображається позиція пальця на світлодіоді – початкова позиція – червоний колір, середня – жовтий, кінцева – зелений.

7.8 Контрольні запитання і завдання

1. Узагальнена архітектура програмованої системи на кристалі.
2. З яких основних блоків складається програмована системи на кристалі?
3. В чому відмінність PSoC від класичних мікроконтролерів?

4. Які особливості останнього покоління програмованих систем на кристалі – PSoC® 5LP?
5. В чому полягає особливість обробки сигналів датчиків?
6. Дайте характеристику підсистем процесора PSoC.
7. Приведіть склад бібліотеки периферійних драйверів PSoC.
8. Склад компонентів PSoC Creator.
9. Дайте визначення набору бібліотек API.
10. Які сторонні IDE можна використовувати сумісно з PSoC Creator?
11. Коротко опишіть процес розробки проекту в PSoC Creator.
12. Опишіть процес створення схеми із каталогу компонентів.
13. Які налаштування ЦАП використані в прикладах?
14. Що таке “Off-Chip” компоненти? Навіщо їх застосовують?
15. Яка функція дисплею Resource Meter?
16. Які компоненти використовуються для вимірювача напруги?
17. Опишіть алгоритм роботи програми сенсорної кнопки.

8. ЗАСТОСУВАННЯ ПРОГРАМОВАНИХ ЛОГІЧНИХ ІНТЕГРАЛЬНИХ СХЕМ ДЛЯ РОЗРОБКИ СИСТЕМ КЕРУВАННЯ

Протягом останніх років технології проектування систем керування зазнали кардинальних змін. Було розроблено мови типу VHDL та Verilog, які стали основними інструментами опису апаратних засобів. Також створено низку потужних САПР НВІС (наприклад, QUARTUS II), які підтримують їх розробку, починаючи з опису на названих мовах до створення конфігурації кристалу. В той же час, завдяки досягненням у галузі інтегральної технології суттєво зріс рівень інтеграції мікросхем, і з'явилась можливість реалізації в них надзвичайно складних процесорних пристроїв. У зв'язку з цим існуючі методи проектування систем керування, які передбачають проведення розробки від самого початку (постановки задачі) до реалізації стали малоефективними, оскільки для створення прототипу системи вони вимагають невиправдано багато часу, що негативно впливає на конкурентоспроможність створеного продукту.

Існує підхід, який базується на використанні готового рішення з можливістю конфігурування під конкретну задачу. Основою такого підходу є створення базової моделі пристрою, яку можна конфігурувати, вибираючи значення тих чи інших параметрів. Ще одним підходом до зменшення часу проектування є такий, що базується на використанні мов програмування високого рівня для опису апаратних засобів. Використання такого підходу дозволяє спростити процес проектування і підняти його на рівень проектування програмного забезпечення. Такий підхід дозволяє одночасно отримати і апаратні, і програмні засоби для реалізації поставленої задачі.

Одним з шляхів вирішення даної задачі є широке використання програмованих логічних інтегральних схем (ПЛІС – Programmable Logic Devices, PLD). ПЛІС являють собою елементну базу, що мають гнучкість замовних ВІС і доступність традиційної жорсткої логіки. Головною відмінною властивістю ПЛІС, на відміну від “жорсткої” логіки, є можливість налаштування на виконання заданих функцій самим користувачем. Сучасні ПЛІС характеризуються низькою вартістю, високою швидкістю, значними функціональними можливостями, багаторазовістю перепрограмування, низькою споживаною потужністю й інше. При цьому час розробки навіть досить складних проектів може складати усього кілька годин. Власно кажучи, розробка пристроїв на основі ПЛІС являє собою нову технологію проектування електронних систем, включаючи їх виготовлення і супроводження. Доказом перспективності нової елементної бази служить щорічна поява нових поколінь, а також постійно зростаючий обсяг випуску вже розроблених ПЛІС.

На даний момент існують традиційний та сучасний підходи (рис. 8.1, 8.2) до проектування та розробки готових продуктів.



Рисунок 8.1 – Традиційний підхід до розробки нових систем

Особливістю традиційного підходу є те, що процес виявлення недоліків функціонування або невідповідностей вимогам технічного завдання можливий лише після випуску дослідного зразка. Тобто неминучі помилки при розробці та проектуванні переходять з етапу на етап і виявляються тільки після понесених витрат часу та ресурсів на виготовлення дослідної партії.

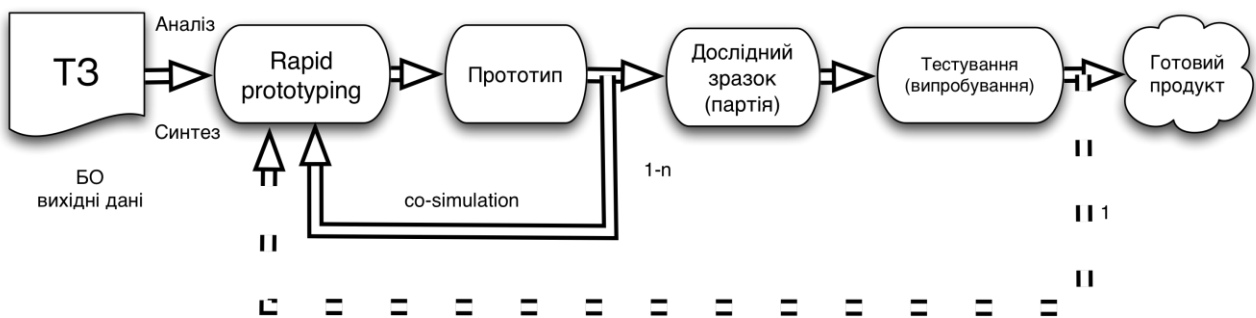


Рисунок 8.2 – Сучасний підхід до розробки нових систем

Найсучаснішим напрямом в розробці спеціалізованих ІС є застосування нового класу інтегральних схем з програмованою користувачем структурою – ASSP (Application-specific standard product), що об'єднав в собі програмовані логічні і аналогові схеми.

Конфігурація ASSP схеми виконується споживачем, виробник в цьому процесі не бере участі і поставляє на ринок продукцію, придатну для багатьох покупців, що створюють апаратуру різного призначення. ASSP мають певною мірою дуальний характер: для споживача вони є спеціалізованими, а для промисловості — стандартними.

На відміну від спеціалізованих інтегральних схем для програмування ASSP не потрібна розробка спеціальних фотошаблонів, що вимагає великих витрат коштів і часу. Застосовуючи для цілей конфігурування програмовані логічні інтегральні схеми, можна виключити розробку шаблонів і порівняно простими способами одержати ефективні засоби побудови цифрових систем, що містять набір схем з необхідним функціонуванням.

Програмування структур спочатку було застосовано в програмованих логічних матрицях (ПЛМ), програмованій матричній логіці (ПМЛ) і базових матричних кристалах (БМК). Слідом за ними виникли нові класи більш складних ASSP, що продовжують лінії розвитку матричної логіки і базових матричних кристалів: CPLD і FPGA, відповідно. Потім були реалізовані ASSP комбінованої (змішаної) архітектури, що поєднували ознаки CPLD і FPGA. Пізніше вдалося розробити ASSP з аналоговими та аналого-цифровими елементами, які можна позначити як ПАІС (програмовані аналогові інтегральні схеми).

8.1 Архітектура ПЛІС. Логічні елементи

Для багатьох розробників ПЛІС уявляється таким собі чорним ящиком, який за допомогою компілятора перетворює вихідний код в реально діючу схему. Але для того, щоб розуміти, що за інструмент ми маємо в руках потрібно, знати – що там всередині. Тому рано чи пізно розробник повинен відмовитися від концепції чорного ящика і вивчити апаратні можливості і структуру ПЛІС.

Кожне сімейство ПЛІС має свої особливості архітектури, проте розглянемо їх на прикладі сімейства Cyclone IV.

У ПЛІС для конфігурації використовується оперативна пам'ять SRAM (Configuration RAM – конфігураційна пам'ять). Ця пам'ять розподілена по всьому кристалу, значення, записані в неї, управляють внутрішнім комутаційним полем, визначаючи структуру синтезованого пристрою. Як правило, в ПЛІС архітектури FPGA ця пам'ять енергозалежна і при подачі живлення на пристрій потрібно завантажити в неї значення з будь-якого зовнішнього (по відношенню до кристалу ПЛІС) носія, часто для цих цілей використовується спеціалізована мікросхема ПЗП.

Однією з найважливіших характеристик ПЛІС є її логічна ємність. Ємність визначає наскільки складні цифрові пристрої можна синтезувати. Основний ресурс ПЛІС – це логічні елементи (Logic Elements). У ПЛІС їх кількість може сягати сотень тисяч

Базовий логічний елемент. З булевої алгебри відомо, що використовуючи якийсь елементний базис, наприклад елемент *I-НІ* чи *АБО-НІ*, можна реалізувати будь-яку логічну функцію. Однак використання лише одного типу елемента не завжди виправдано технічно, при синтезі складних пристроїв велика кількість елементів збільшить час проходження сигналу і тим самим знизить швидкодію. Тому в ПЛІС структури FPGA в якості найпростішого логічного елемента використовують складнішу структуру, що представляє собою з'єднання програмованого комбінаційного пристрою і D-тригера. На рисунку 8.3 показана спрощена структура такого елемента.

Осередок має три логічних входи DATAA, DATAB і DATAC, вхід тактових імпульсів CLK і один вихід LEOUT. У разі якщо від осередку потрібна робота в якості лише комбінаційного пристрою, то вихідний мультиплексор комутує вихід елемента LUT на вихід всієї комірки, якщо вихід повинен бути регістровий, то сигнал з LUT замикається по сигналу синхронізації в D-тригер,

вихід якого через мультиплексор з'єднується з LEOUT. Керуючий вхід мультиплексора (на малюнку не показаний) підключений до відповідного біту пам'яті конфігурації *CRAM*.

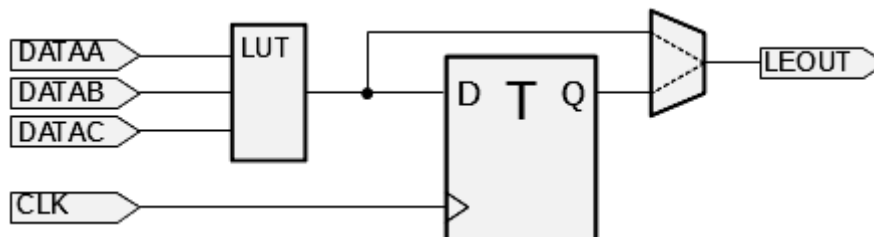


Рисунок 8.3 – Узагальнена структура логічного елемента (LE) ПЛІС

Абревіатура LUT розшифровується як Look-Up Table, що дослівно можна перекласти як "довідкова таблиця" або "таблиця відповідності". Однак, LUT – це скоріше метод реалізації функції, в якому безпосереднє обчислення замінюється пошуком по таблиці готових рішень. Стосовно до ПЛІС це дозволяє реалізувати будь-яку логічну функцію у вигляді пам'яті SRAM, де адреса – це аргумент, а вміст комірки – значення. Таким чином, для того, щоб описати логічну функцію трьох змінних (в наведеному прикладі саме три змінні: DATAA, DATAB і DATAC) досить пам'яті на 8 комірок. Необхідна таблиця істинності зберігається у вигляді маски (LUT-mask) у відповідному осередку *CRAM*. За допомогою мультиплексорів вибирається потрібне значення. Мультиплексорами керують сигнали входних портів для побудови k -входову LUT (k -LUT),^k біт SRAM і 2^{k-1} мультиплексорів. На рисунку 8.4 зображено трьохвходовий LUT.

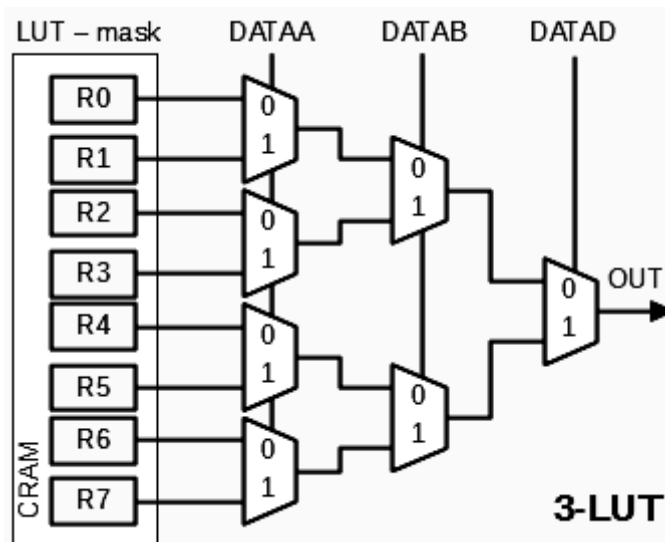


Рисунок 8.4 – Пристрій LUT

За таким підходом можна досить точно спрогнозувати час проходження сигналу і він не буде залежати від реалізованої логічної функції. Ця важлива особливість робить можливим часовий аналіз схеми.

Використовуючи логічний елемент, показаний на рис.8.3, можна отримати будь-яку логічну функцію трьох змінних і її регістровий варіант. Для реалізації деяких типів досить наявного комбінаційного пристрою (LUT) на вході D-тригера, але для реалізації всіх відомих тригерів доведеться внести деякі зміни в базову схему. Необхідно ввести зворотний зв'язок з виходу D-тригера на один з входів LUT. А також потрібно замінити LUT на 4-х входовий. Також потрібно збільшити функціональність самого D-тригера, за рахунок додаткової лінії управління: вхід дозволу ENA (рис.8.5).

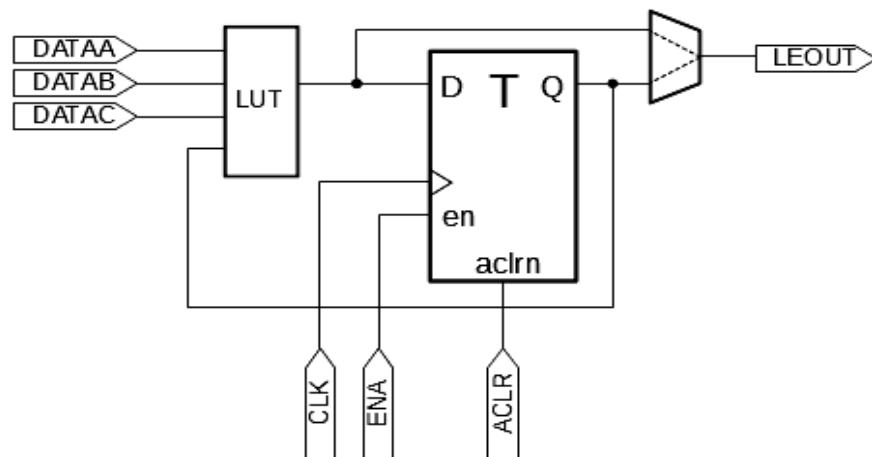


Рисунок 8.5 – Структура логічного елемента (LE) зі зворотним зв'язком

Для вирішення типових в цифровій техніці арифметичних задач, для роботи з перенесенням даних в базовому логічному елементі передбачений спеціальний канал.

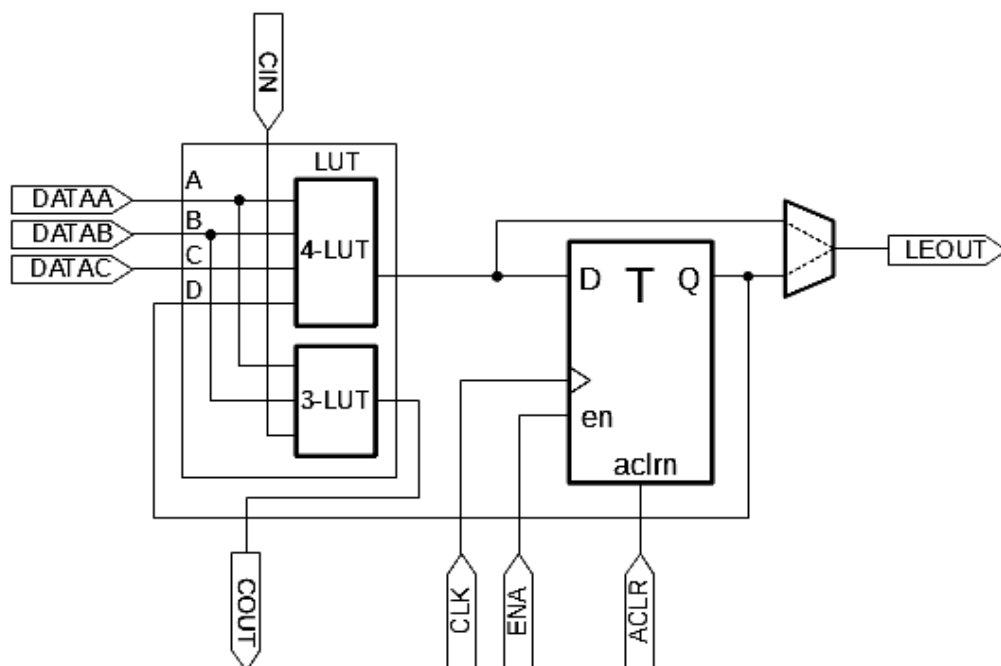


Рисунок 8.6 – Структура логічного елемента (LE) з каналом перенесення

Фактично для перенесення потрібно ввести додаткову LUT, як показано на рис.8.6, але для простоти сприйняття комбінацію LUT будемо розглядати як один цілий модуль.

Використовуючи канал перенесення, легко об'єднати комірки для отримання багато розрядного суматора. На рис.8.7 показана схема 4-розрядного суматора з послідовним переносом, побудована на чотирьох базових логічних елементах.

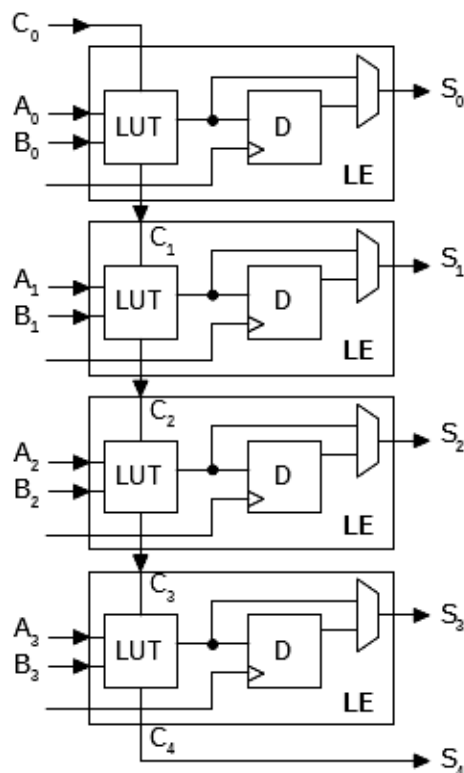


Рисунок 8.7 – Чотирьох розрядний суматор

При розробці базової логічної комірки вирішувалися два завдання: по-перше, синтезовані пристрої повинні володіти максимальною швидкістю, по-друге, використання ресурсів має бути якомога повнішим. У попередньому прикладі, якщо потрібна робота суматора виключно як комбінаційної схеми, вихідні мультиплексори сигнал візьмуть з виходів LUT, а весь ланцюжок тригерів виявиться невикористаною. На противагу цьому, при синтезі послідовного або паралельного регістра, все LUT будуть по суті виконувати функцію провідника: з'єднувати вхід LE з входом D-тригера. Невелике доповнення до схеми, дасть можливість при необхідності використовувати комбінаційний пристрій і тригер елемента окремо для синтезу незалежних модулів. Мультиплексор на вході тригера дозволить вибирати джерело сигналу: або з входу DATAC, або з виходу LUT. Крім того, з'являється можливість організувати додатковий канал з'єднання тригерів сусідніх LE для збільшення швидкодії при побудові послідовних регістрів. На рис.8.8 показана схема отриманого логічного елемента. Вхід REGIN і вихід REGOUT утворюють виділений канал для з'єднання тригерів,

вхід SLOAD (Synchronous Loading – Синхронне завантаження) управляє вибором джерела сигналу для входу тригера.

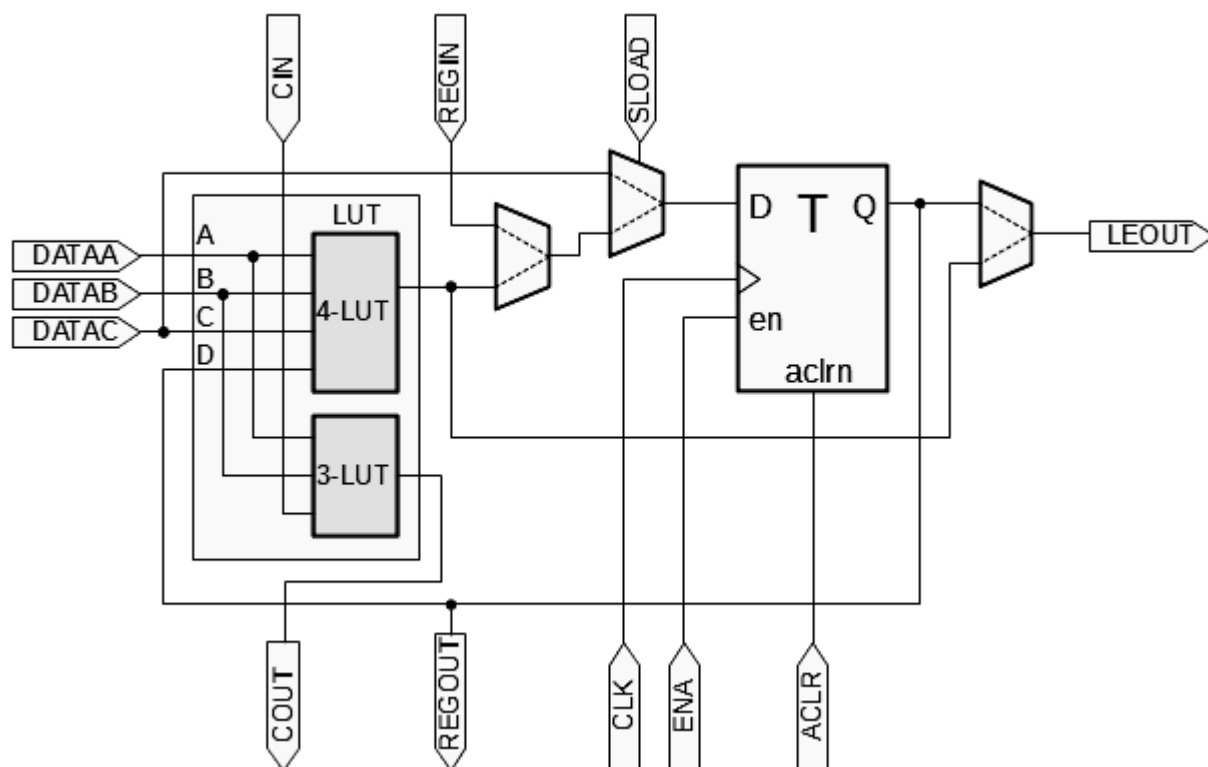


Рисунок 8.8 – Структура логічного елемента (LE) з можливістю поділу LUT і тригера

8.2 Базовий логічний елемент ПЛІС Cyclone IV

Розглянутий елемент LE в тому чи іншому вигляді присутній в різних сімействах ПЛІС, в ньому може бути збільшено кількість входів і складність LUT, додані додаткові з'єднання всередині LE і порти для інтеграції в глобальну комутаційну мережу ПЛІС. Так, наприклад, в топовому сімействі Stratix IV в один елемент входить дві шестивходові LUT, два виділені повних суматора і чотири тригери.

На рис.8.9 показана схема логічного елемента ПЛІС Cyclone IV.

Також, як і в базовому логічному елементі, кожен тригер має вхід даних, вхід тактових імпульсів CLK, вхід дозволу ENA і вхід асинхронного скидання (очищення) ACLR. Додався сигнал синхронної очищення SCLR (від англійського "Synchronous Clear" – Синхронне Очищення): якщо на цьому вході з'явиться логічна одиниця, то наступним тактовим імпульсом в тригер буде записаний логічний нуль. Все це дозволяє налаштувати програмований тригер кожної LE на роботу в режимі D, T, JK або RS тригера.

Комбінаційний пристрій в порівнянні з базовою схемою отримало деяке ускладнення. На вході C LUT мультиплексор вибирає джерело сигналу, завдяки чому LUT може реалізовувати логічну функцію чотирьох змінних, крім того, в

якості змінної може бути використаний прапор переносу або вихід власного тригера.

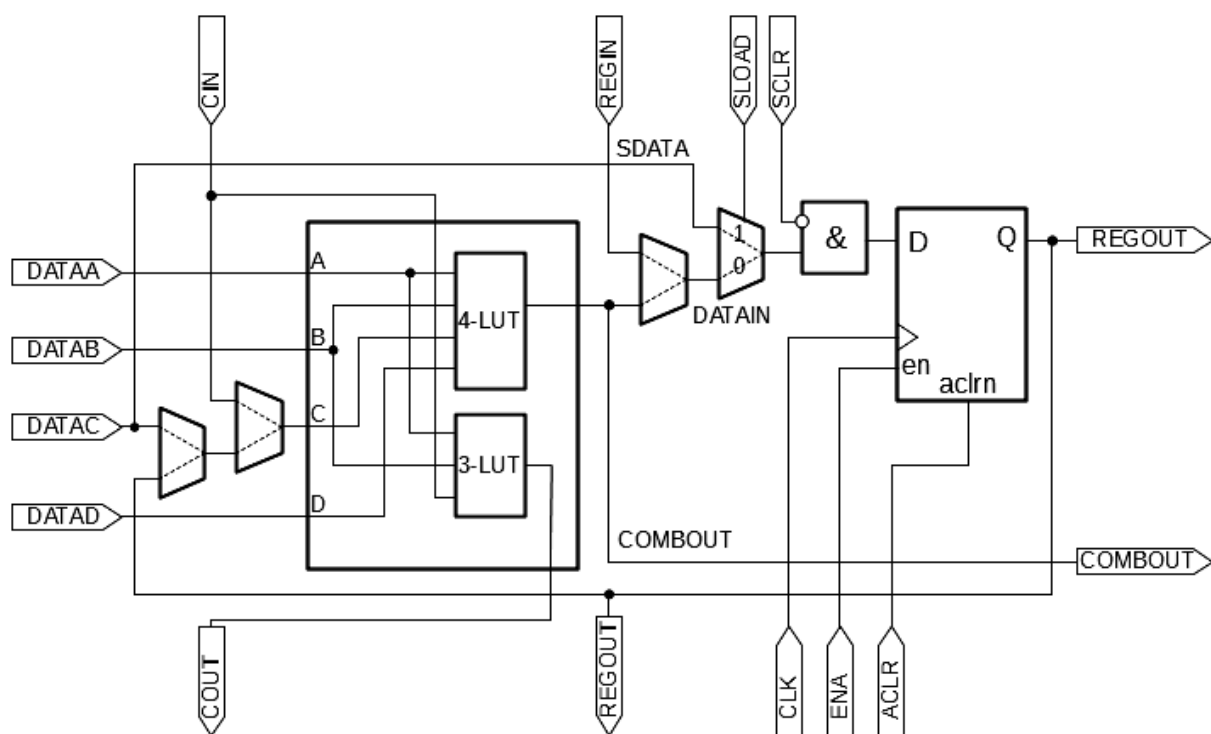


Рисунок 8.9 – Логічна структура елемента (LE) Cyclone IV

У Cyclone IV LUT може працювати в двох режимах: нормальному і арифметичному. САПР Quartus при компіляції автоматично вибере оптимальний режим для реалізації необхідної функції. Нормальний режим призначений для реалізації основної логіки і різних комбінаційних функцій. В цьому режимі чотири входи LE (DATAA, DATAB, DATAC, DATAD) надходять на чотири входи LUT. Компілятор автоматично вибирає вхід перенесення CIN, вхід DATAC або вихід тригера (коло зворотного зв'язку) в якості одного з входів LUT. Арифметичний режим підходить для синтезу суматорів, лічильників, акумуляторів і компараторів (ланцюгів порівняння). В цьому режимі LUT – повний однорозрядний суматор, що включає обробник логіки прапора переповнення. Компілятор сам створює кола перенесення під час синтезу багаторозрядних арифметичних пристроїв.

Можна звернути увагу, що на рисунку логічного елемента Cyclone IV відсутній вихідний мультиплексор, насправді він є і не один. Але для того, щоб зрозуміти логіку їх роботи, потрібно розглянути LE в контексті загальної архітектури ПЛІС.

Логічні елементи LE об'єднуються в логічні блоки **LAB** (Logic array blocks). У Cyclone IV кожен LAB містить:

- 16 логічних осередків;
- сигнали управління LAB;
- кола прапора перенесення LE;
- кола каскадного об'єднання регістрів;

– кола локальних з'єднань.

Кола локальних з'єднань передають сигнали між осередками LE в одному LAB. Кола об'єднання регістрів з'єднують вихід регістра одного LE з входами регістрів прилеглих осередків LE. Компілятор Quartus II розміщує пов'язану логіку в LAB або в сусідніх LAB, дозволяючи використовувати локальні кола зв'язку і зв'язку регістрів для збільшення продуктивності і ефективності розміщення.

На кола локальних з'єднань надходять сигнали з рядків і стовпців глобального комунікаційного поля і з виходів осередків LE, що належать цьому ж блоку LAB. Сусідні логічні блоки, блок генератора з фазовим автопідстроюванням частоти (PLL), осередки пам'яті M9K RAM, вбудовані помножувачі, розташовані з правого або лівого боку через спеціальні з'єднувачі, можуть бути безпосередньо підключені до кіл локальних з'єднань LAB. Таким чином, будь-який осередок LE може мати з'єднання з 16 LE зі свого блоку (включаючи саму себе) і тридцятьма двома LE з LAB, розташованих зліва чи справа. Всього до 48-ми сполук! Такі безпосередні прямі з'єднання мінімізують використання глобальних маршрутів, забезпечують більшу гнучкість при синтезі схеми і збільшують загальну швидкодію.

На рисунку 8.10 показана інтеграція блоку LAB в глобальну комунікаційну мережу ПЛІС.

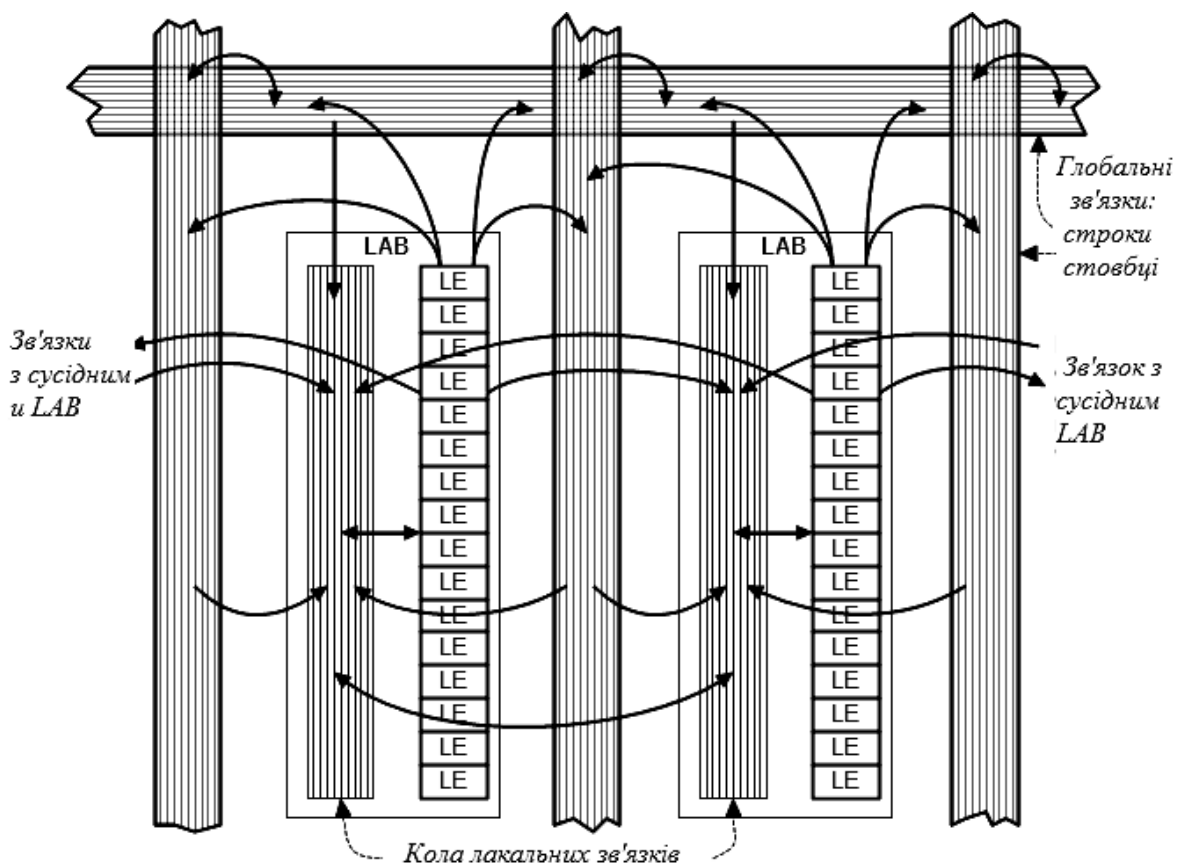


Рисунок 8.10 – Структура з'єднань LAB в комутаційному полі ПЛІС

Кожен LE має три виходи, які забезпечують з'єднання з комутаційним полем ПЛІС. Ці виходи надходять на рядки і стовпці глобальних сполучних трас і на маршрути локальних з'єднань. Як і в базовому логічному елементі, LUT або тригер можуть незалежно управляти цими виходами.

Для того, щоб можна було керувати всіма осередками LE в межах одного LAB одночасно, в логічний блок вбудована спеціальна логіка і виділені особливі лінії – канали управління. За таким каналам поширюються широкосмугові (в межах одного LAB) сигнали управління. Архітектура дозволяє одночасно використовувати до восьми керуючих сигналів:

- два тактових сигнали (labclk1 і labclk2);
- два сигнали дозволу (labclkena1 і labclkena2);
- два сигнали асинхронного скидання (labclr1 і labclr2);
- сигнал синхронного скидання / очищення (synclr);
- сигнал синхронного завантаження (syncload).

Сигнали синхронної завантаження і скидання зручно використовувати для синтезу різних лічильників і регістрів. Ці сигнали впливають на всі тригери LE в межах одного LAB.

На рисунку 8.11 показано підключення керуючих сигналів LAB і ланцюгів локальних з'єднань до логічної чарунки. Виводи LE відповідають рис. 8.9.

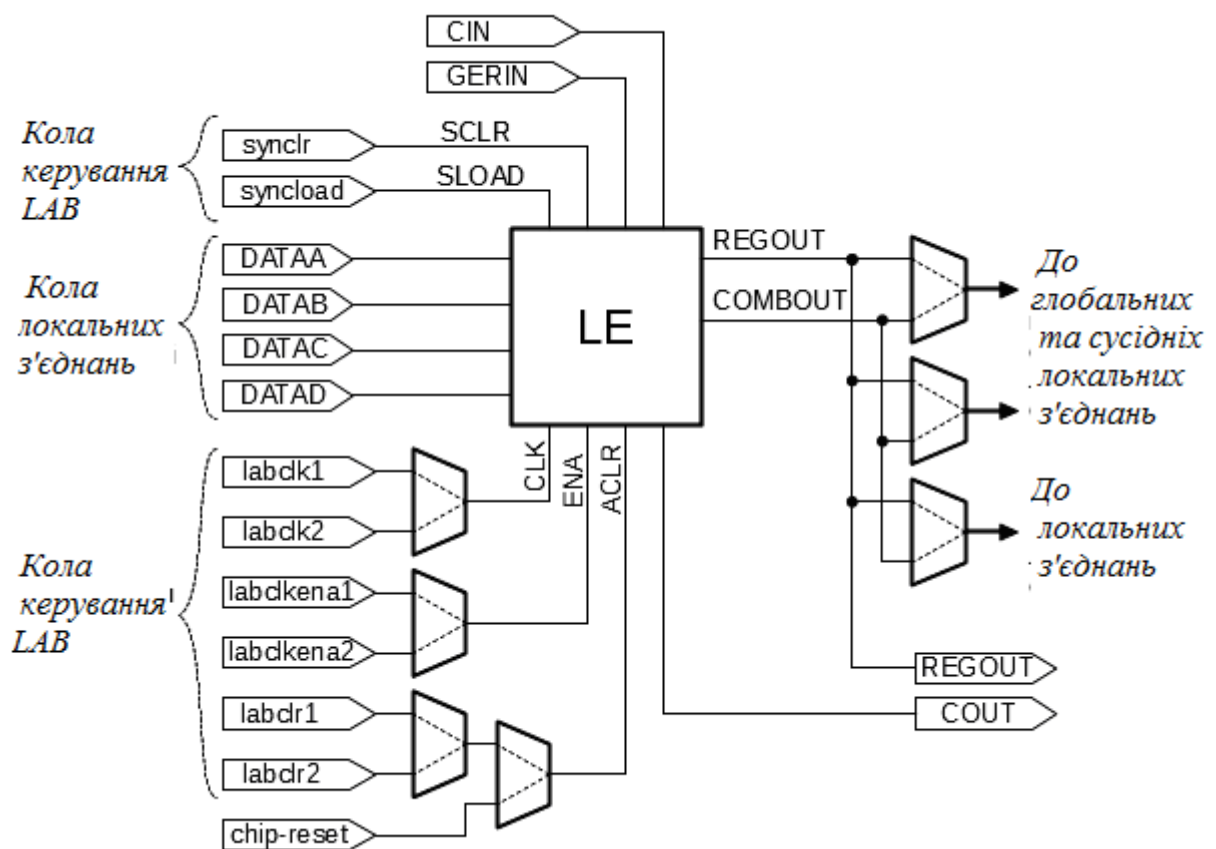


Рисунок 8.11 – Структура взаємодії LE з сигналами управління

Кожен LAB має два тактових сигналу і два сигнали дозволу. На рівні логічного елемента вибирається який сигнал буде поданий на тригер, але тактовий сигнал пов'язаний з сигналом дозволу. Тому, хоча для будь-якого LE в конкретній LAB можна вибрати який тактовий сигнал використовувати: `labclk1` або `labclk2`, разом з ним повинен використовуватися відповідний сигнал дозволу. Наприклад, якщо використовується `labclk1`, разом з ним буде використовуватися тільки `labcken1`. Якщо LAB використовує обидва фронти (наростаючий і падаючий) тактового сигналу, то будуть задіяні обидва канали тактування, при цьому логіка вибору ускладниться.

Після компіляції проекту в Quartus II в вікні Flow Summary можна подивитися відсоток використаних ресурсів (рис.8.12).

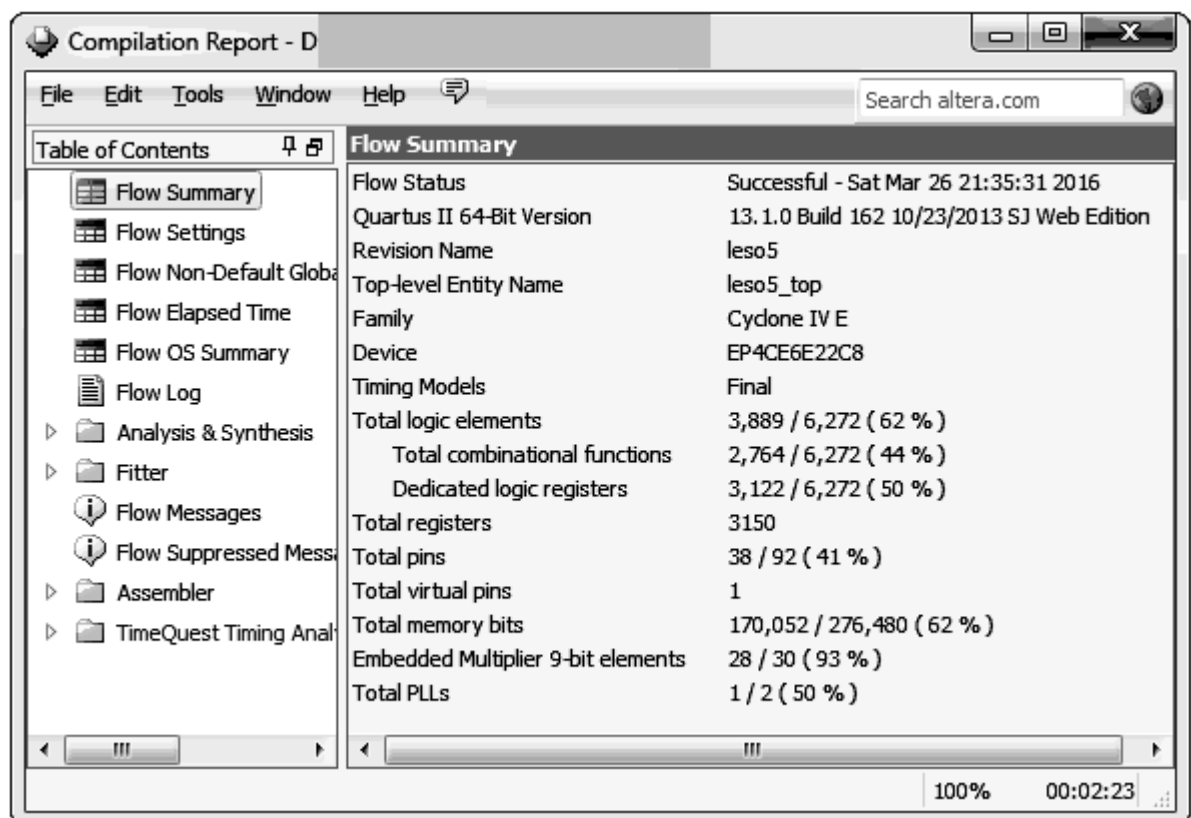


Рисунок 8.12 – Звіт компіляції

Для візуального контролю використовуваних ресурсів в Quartus II застосовується утиліта **Chip Planer**. Вона показує розташування і використання елементарних блоків в загальній архітектурі цільової ПЛІС. Запустити Chip Planer можна з середовища Quartus: меню Tools -> Chip Planer. На карті кристала (рисунок 8.13) показані всі ресурси ПЛІС: LE, об'єднані в LAB, апаратні помножувачі (DSP block), осередки пам'яті, буфери введення виведення, генератори PLL. Кольором показана ступінь використання: світлим - невикористовувані блоки, темним - максимально завантажені. Якщо збільшити масштаб (відповід-

ний інструмент на панел або ctrl + коліщатко мишки), то доступна деталізація на рівні LE, тут синім показаний LUT, темно червоним - тригер.

Велику деталізацію можна отримати якщо на панелі Layers Settings змінити Basic на Detailed. У цьому режимі при наближенні видно локальні і глобальні ланцюга сполук, глобальні лінії управління та керуючі сигнали LAB.

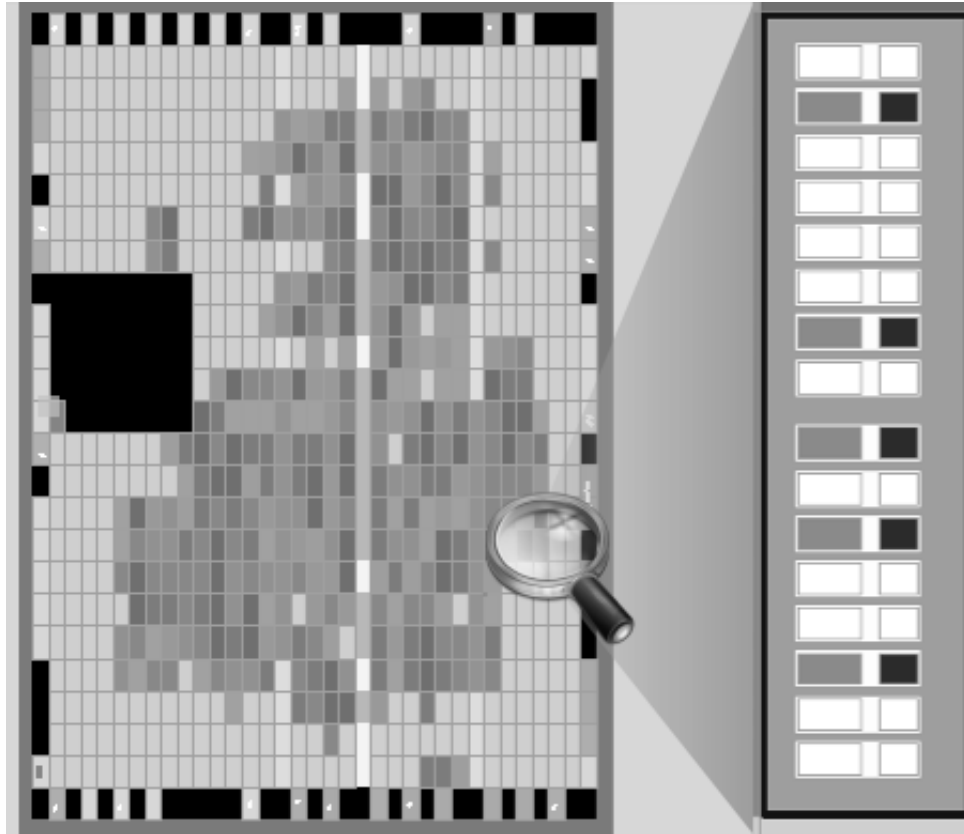


Рисунок 8.13 – Карта ресурсів ПЛІС

Якщо в LE виділити LUT або тригер, то на панелі Node Properties можна побачити схему LE і опис властивостей і режимів роботи. Подвійний клік миші по LUT або тригеру запустить в новому вікні інструмент Resource Property Editor, в якому можна досліджувати з'єднання всередині логічного елемента.

8.3 Приклад розробки елемента системи керування на ПЛІС

Сучасна технологія, що отримала назву технології швидкого прототипування, дозволяє отримувати прототипи, на яких вже можливо виконувати пошук різного роду помилок функціонування ще до виготовлення дослідних зразків.

Для скорочення терміну створення проектів ІС застосовують засоби розробки прототипів із застосуванням програмованої логіки. Даний розділ присвячений розгляду процедури розробки проекту цифрової ІС типу системf на кристалі (SOC) на ПЛІС.

Для цього потрібен комплекс, що складається з IBM-PC сумісного комп'ютера із установленим пакетом САПР ПЛІС Quartus, програматора типу

Byte Blaster MV, макета із установленою інтегральною схемою програмованої логіки й периферійних пристроїв відображення, призначених для візуалізації роботи проектованої цифрової схеми.

Створення елементу системи керування на базі процесорного ядра Nios II. Запустити програму Quartus. Ознайомитися з повним списком додатків, що входять до складу САПР.

Nios II – програмне процесорне ядро, написане на мові програмування апаратури і призначене для реалізації в ПЛІС фірми ALTERA. Для створення системи на кристалі до процесорного ядра необхідно додати додаткові компоненти – пам'ять, порти вводу/виводу, таймер, необхідні інтерфейси. Для цієї мети використовується спеціальна утиліта, що входить до складу САПР Quartus II – SOPC Builder, що дозволяє формувати необхідну архітектуру процесорної системи. Приклад одного з варіантів реалізації архітектури процесорного ядра показаний на рис. 8.14.

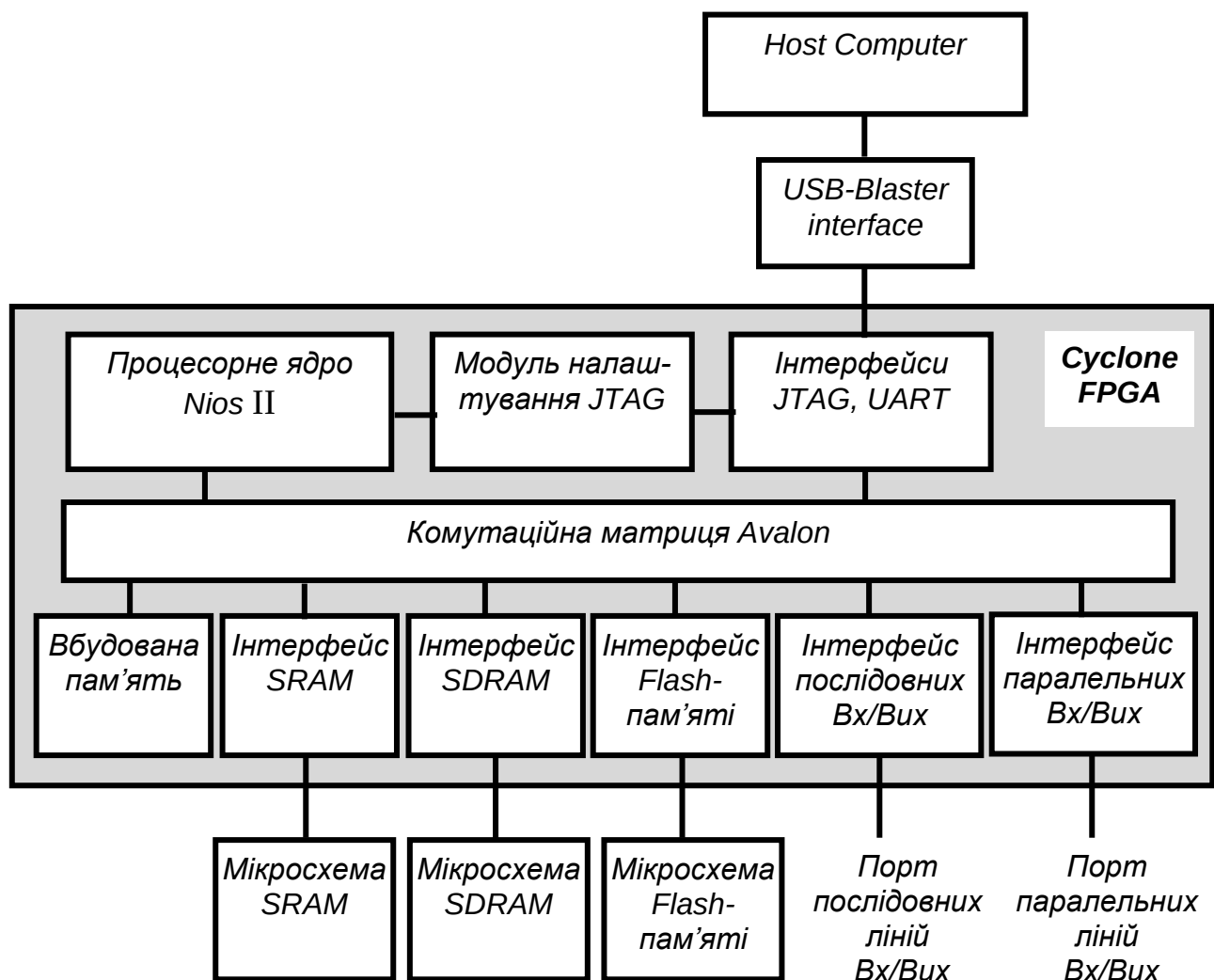


Рисунок 8.14 – Приклад архітектури процесорного ядра з Nios II

Для обміну даними з периферійними пристроями (наприклад, SRAM, SDRAM, Flash та інші) всередині ПЛІС формуються спеціальні компоненти – інтерфейсні блоки. Вони пов'язані з процесорним ядром за допомогою спеціальної шини – Avalon switch fabric. До складу процесорного ядра можуть входити і спеціалізовані блоки: JTAG UART – для обміну інформацією з комп'ютером за допомогою завантажувального кабелю (USB-Blaster або ByteBlaster), модуль JTAG Debug – для виконання налагодження програмного забезпечення за допомогою комп'ютера та інші. Користувач може підключати до процесорного ядра як бібліотечні компоненти (що входять до складу САПР Quartus II), так і власні модулі.

Для створення проекту може бути використано будь-який модуль виробника, наприклад, **DE2 Cyclone II Edition**. Порядок роботи з модулем наступний.

1. Створіть новий проект у робочій директорії. Вкажіть назву проекту, але виключно латинськими буквами (рис. 8.15).

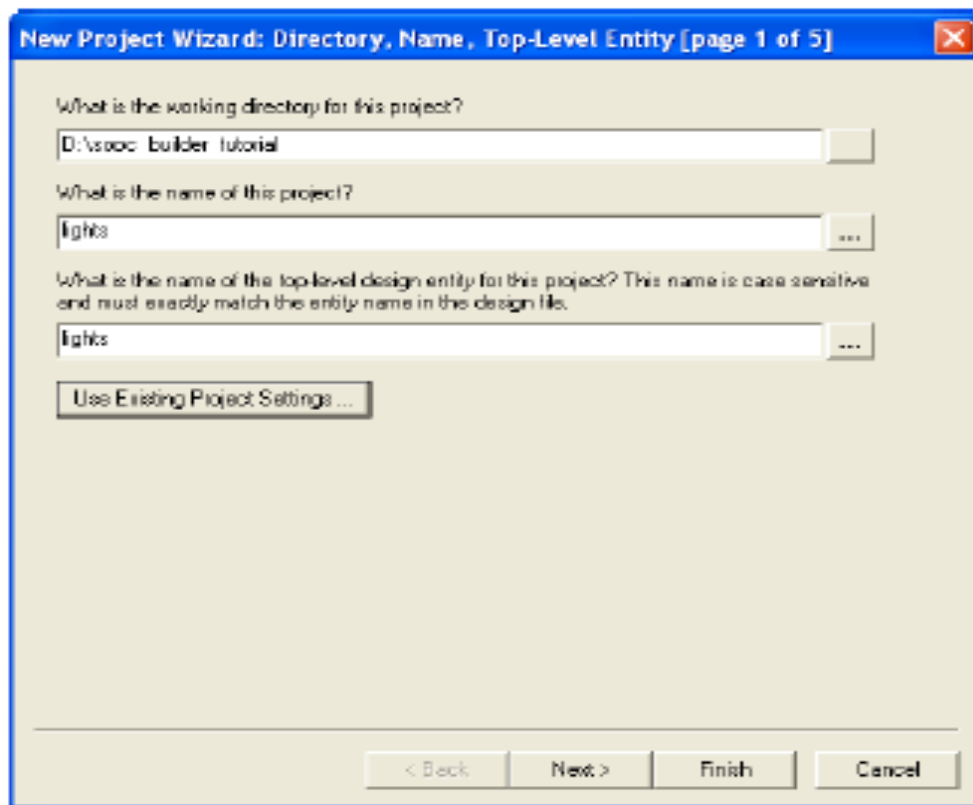


Рисунок 8.15 – Створення нового проекту

2. В якості мікросхеми, для якої створюється проект, вкажіть мікросхему **EP2C35F672C6** сімейства Cyclone II (встановлена на модулі DE2).

3. Створюваний процесорний модуль повинен містити наступні компоненти:

- процесорне ядро (**Nios II CPU**);

- внутрішній блок пам'яті для зберігання програми і даних (**onchip memory**);
 - контролер послідовного каналу зв'язку **JTAG UART**;
 - паралельні порти вводу/виводу для зв'язку з зовнішніми компонентами модуля – світлодіодами, кнопками (**PIO peripherals**).
4. Викличте утиліту формування архітектури процесорного ядра (меню **Tools** → **SOPC Builder...**). Вкажіть ім'я процесорної системи – **nios_system** і мову опису – **Verilog**.
5. У діалоговому вікні (рис. 8.16) перевірте системні налаштування – сімейство ПЛІС **Cyclone II**, тактова частота (**clk_0**) – зовнішня, 50МГц.

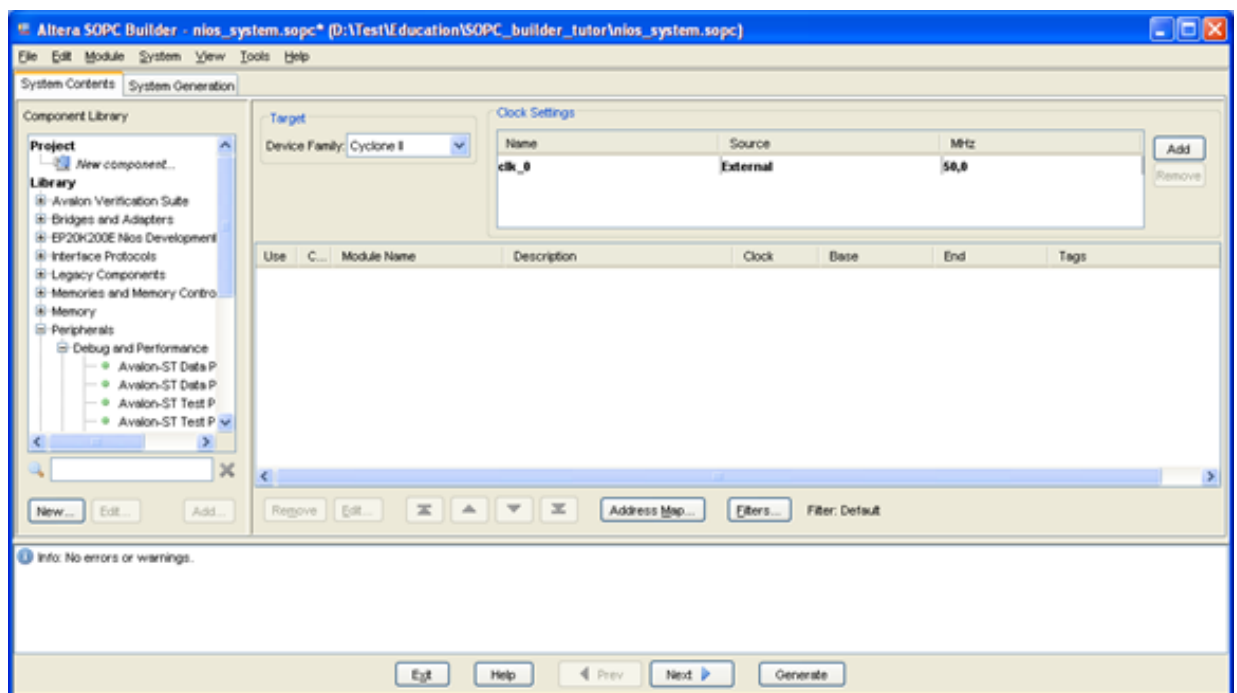


Рисунок 8.16 – Перевірка системних налаштувань

6. Додайте процесорне ядро в проект. У вікні бібліотеки системних компонентів (**Component Library**) в папці **Processors** виділіть **Nios II Processor** і натисніть кнопку **Add**. У діалоговому вікні виберіть архітектуру **Nios II/e** (рис. 8.17).
7. Натисніть кнопку **Finish**. У вікні повідомлень утиліти SOPC Builder можуть з'явитися попередження або повідомлення про помилку. Вони пов'язані з тим, що для процесора Nios II був пропущений ряд налаштувань. Їх можна зробити пізніше.

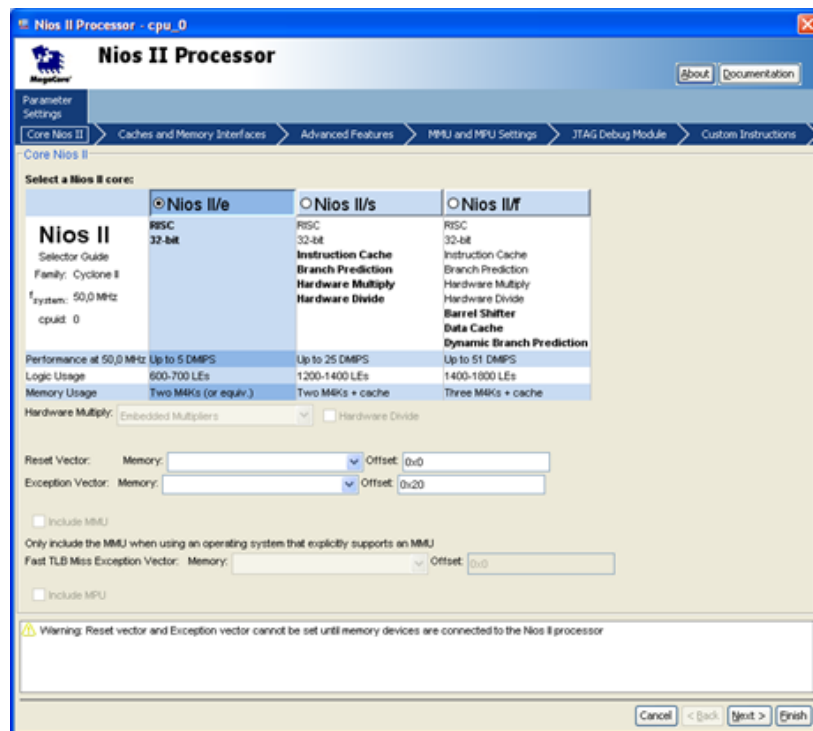


Рисунок 8.17 – Налаштування процесорного ядра

8. Зовнішній вигляд діалогового вікна утиліти SOPC Builder повинен виглядати так, як показано на рис. 8.18.

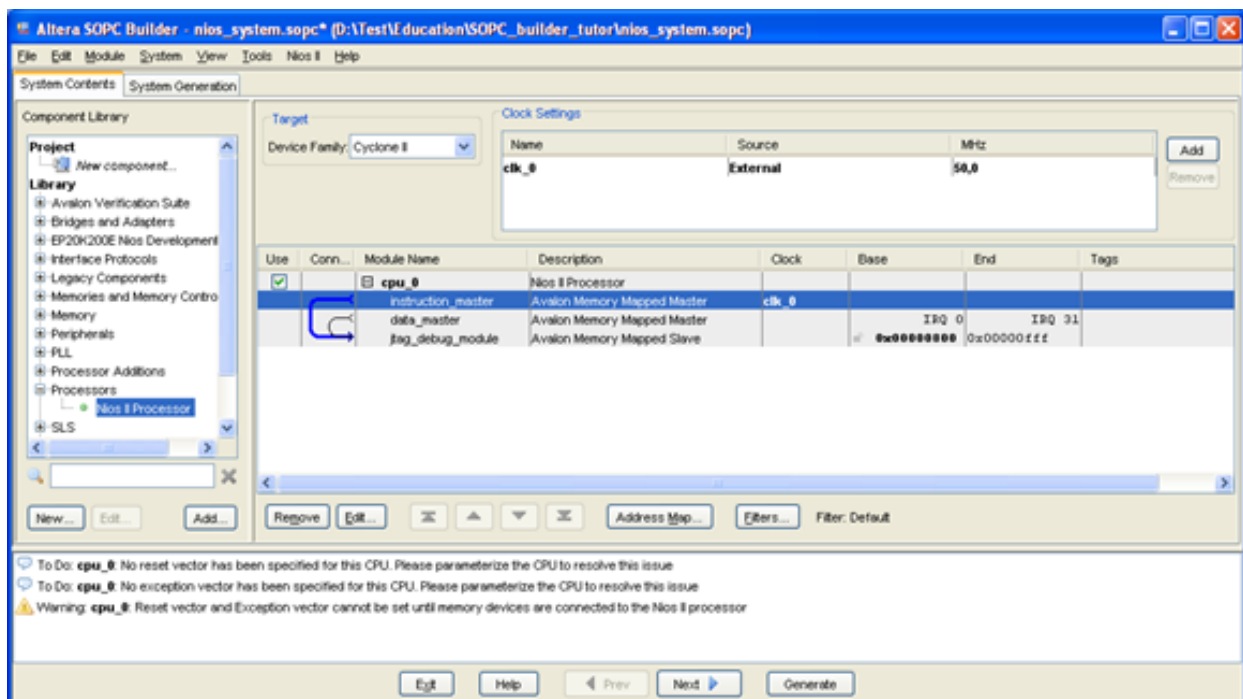


Рисунок 8.18 – Зовнішній вигляд діалогового вікна утиліти SOPC Builder

9. Підключіть до процесорного ядра внутрішній блок пам'яті. Для цього в розділі **Memories and Memory Controllers** розкрийте папку **On-Chip** і ви-

діліть компонент **On-Chip Memory (RAM or ROM)**. Натисніть кнопку **Add....**. У діалоговому вікні встановіть наступні параметри: тип пам'яті RAM (Writable), обсяг пам'яті – 4096 байт, розрядність шини даних – 32 (як показано на рис. 8.19). Інші налаштування залиште без змін. Натисніть кнопку **Finish**.

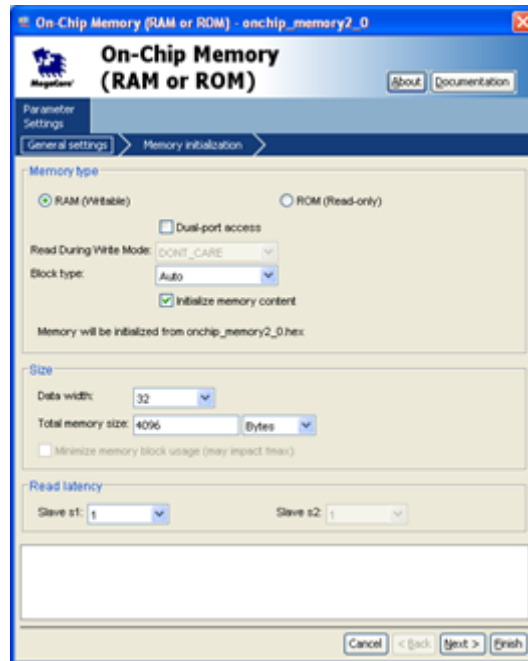


Рисунок 8.19 – Налаштування внутрішнього блоку пам'яті

10. Зовнішній вигляд діалогового вікна утиліти SOPC Builder повинен виглядати так, як показано на рис. 8.20.

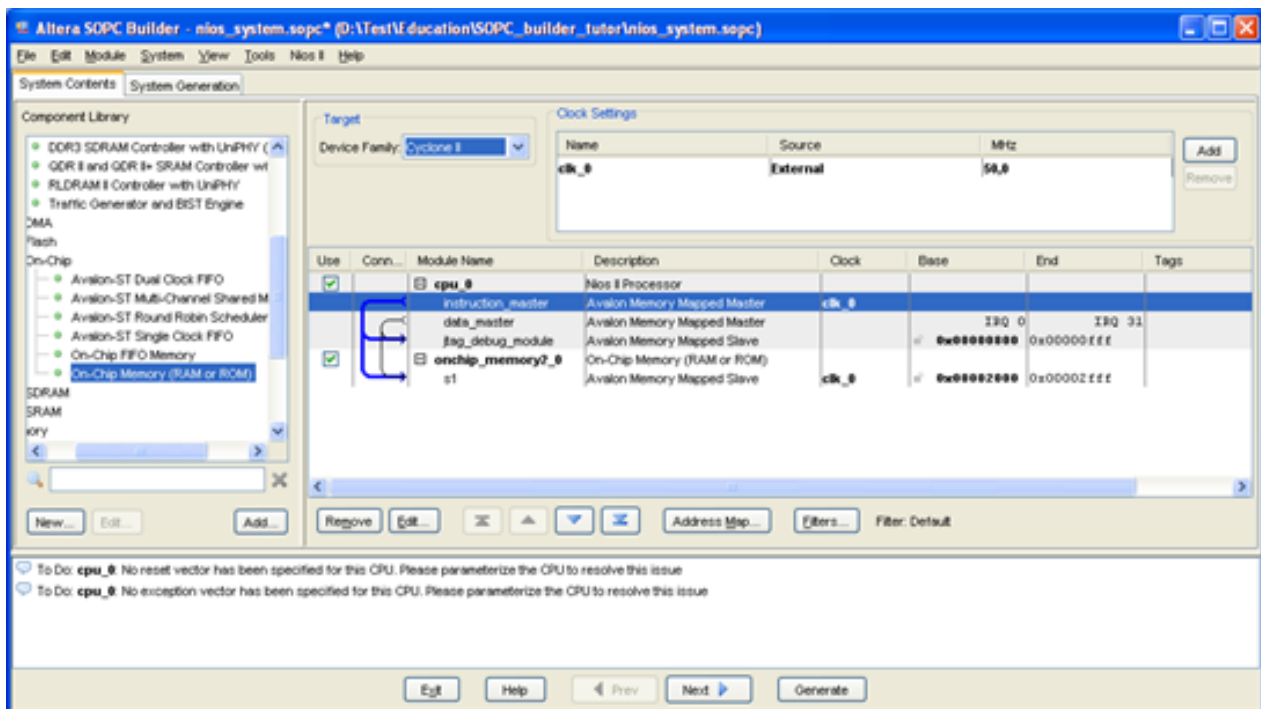


Рисунок 8.20 – Зовнішній вигляд діалогового вікна утиліти SOPC Builder

11. Підключіть до процесорного ядра контролер порту вводу/виводу для перемикачів. Для цього в переліку пристроїв в розділі **Peripherals** розкрийте папку **Microcontroller Peripherals** і виділіть компонент **PIO (Parallel I/O)**. Натисніть кнопку **Add...** У діалоговому вікні встановіть розрядність порту рівну 8 біт і напрямок передачі даних – **Input ports only** (як показано на рис.8.21). Натисніть кнопку **Finish**.



Рисунок 8.21 – Налаштування порту вводу/виводу для перемикачів

12. Аналогічним способом підключіть до процесорного ядра контролер порту вводу/виводу для світлодіодів. Для цього в переліку пристроїв в розділі **Peripherals** розкрийте папку **Microcontroller Peripherals** і виділіть компонент **PIO (Parallel I/O)**. Натисніть кнопку **Add...** У діалоговому вікні встановіть розрядність порту рівну 8 біт і напрямок передачі даних – **Output ports only**. Натисніть кнопку **Finish**.

13. Зовнішній вигляд діалогового вікна утиліти SOPC Builder показано на рис. 8.22.

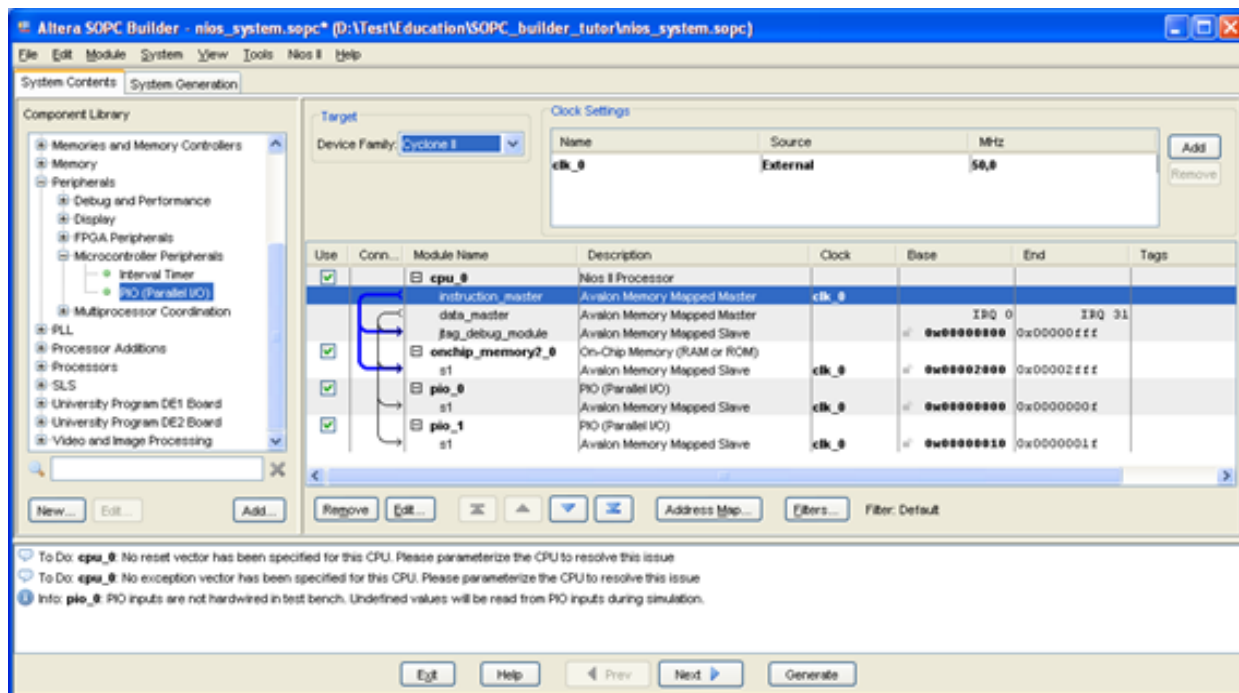


Рисунок 8.22 – Зовнішній вигляд діалогового вікна утиліти SOPC Builder

14. Підключіть до процесорного ядра контролер послідовного каналу JTAG UART. Для цього в переліку пристроїв в розділі **Interface Protocols** розкрийте папку **Serial** і виділіть компонент **JTAG UART**. Натисніть кнопку **Add...**. У діалоговому вікні залиште всі значення без змін (як показано на рис.8.23). Натисніть кнопку **Finish**.

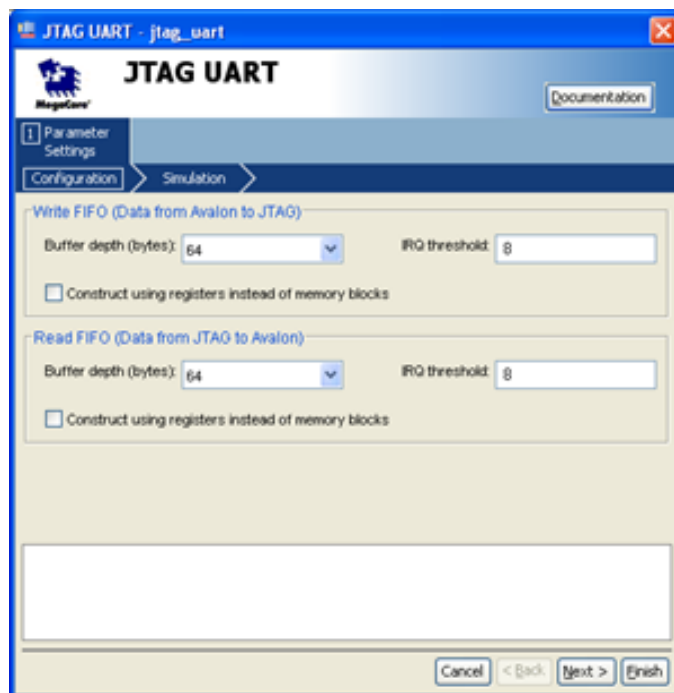


Рисунок 8.23 – Налаштування контролера послідовного каналу

15. Процесорний модуль зібраний. Для зміни імені компонента, що входить до складу процесорного модуля, потрібно виділити компонент, натиснути праву кнопку миші і вибрати команду **Rename** з контекстного меню. За заданими іменами відбувається звернення до компонентів з програми.

16. Перейменуйте порт вводу/виводу **pio_0** на **Switches**, а порт вводу/виводу **pio_1** на **LEDs**. Зовнішній вигляд діалогового вікна утиліти SOPC Builder повинен виглядати так, як показано на рис. 8.24.

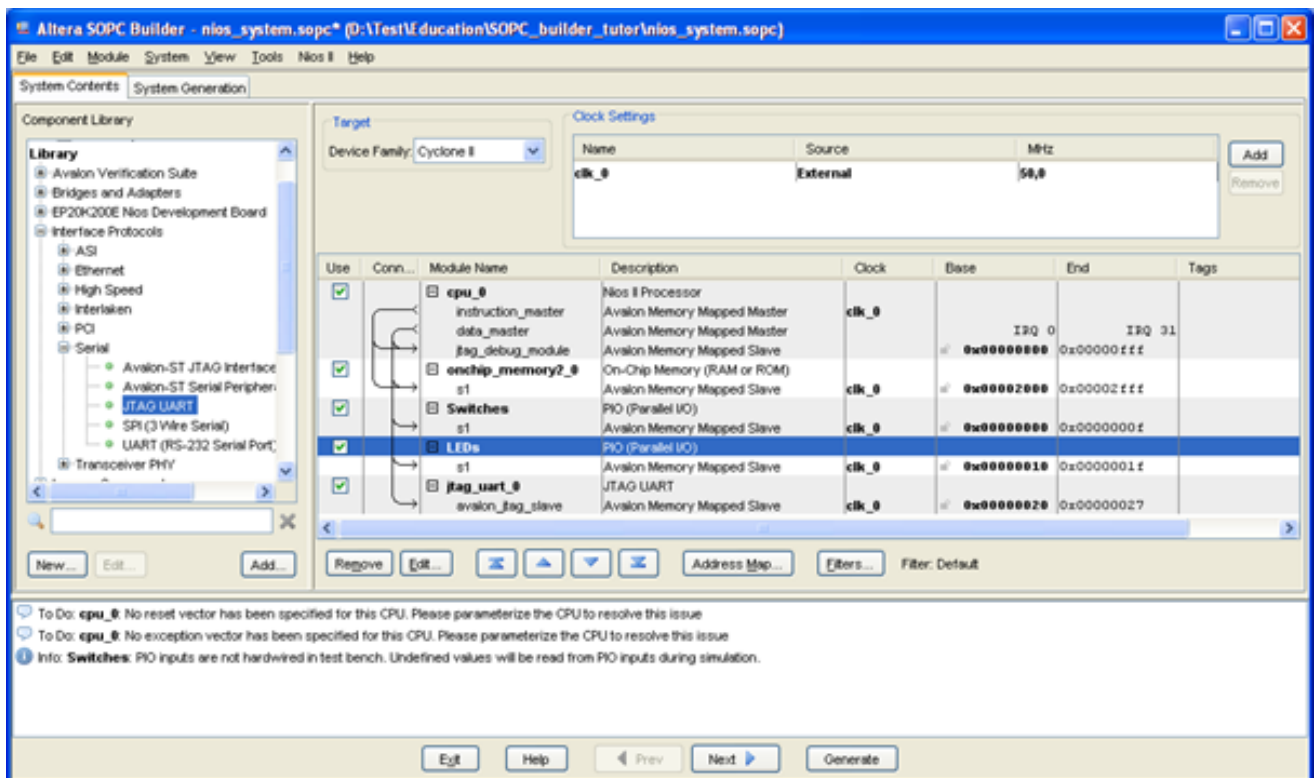


Рисунок 8.24 – Зовнішній вигляд діалогового вікна утиліти SOPC Builder

17. Для присвоєння базових адрес пристроям, які входять до складу процесорної системи, в меню **System** утиліти SOPC Builder виберіть команду **Auto-Assign Base Addresses**. Зверніть увагу на зміну базових адрес компонентів, що входять до складу процесорного модуля.

18. Якщо все виконано правильно, то процесорна система повинна виглядати так, як показано на рис. 8.25.

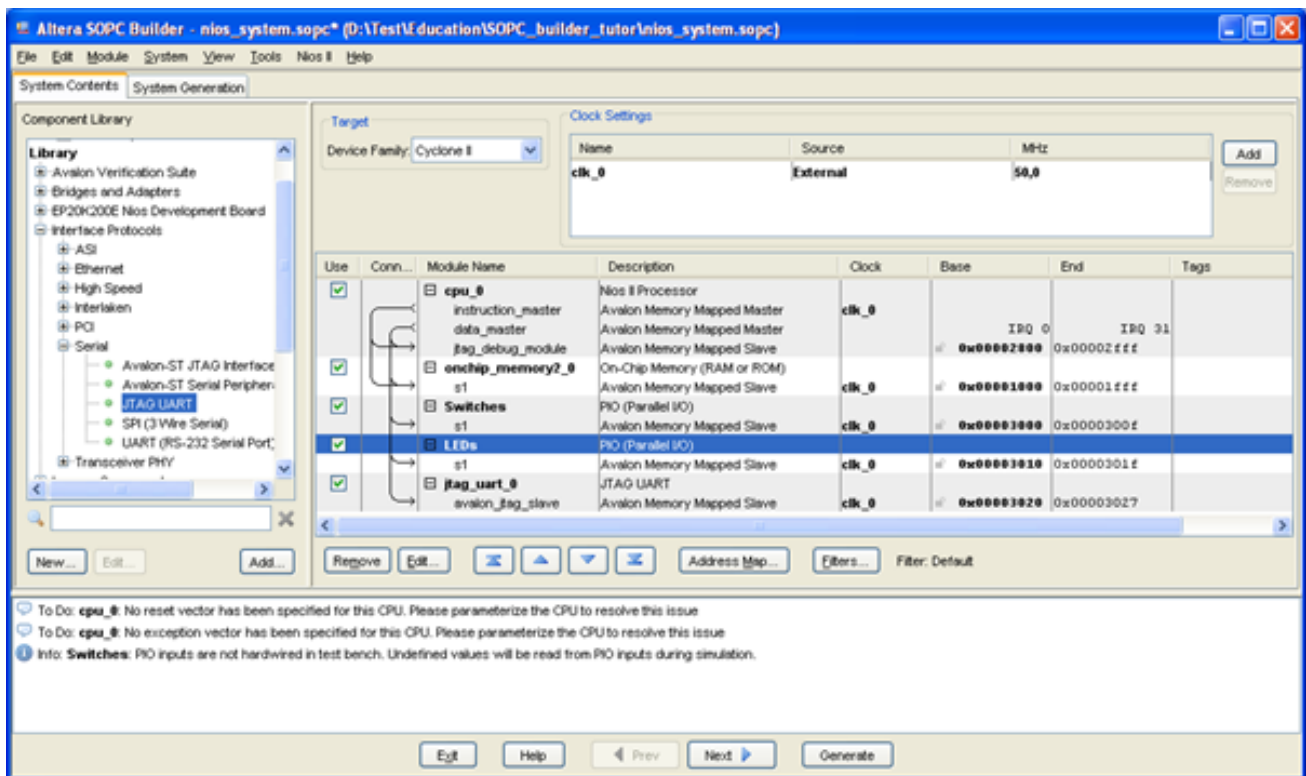


Рисунок 8.25 – Зовнішній вигляд вікна процесорної системи

19. Вкажіть розміщення стартової адреси програми і таблиці векторів переривання. Для цього викличте діалогове вікно налаштування процесорного модуля **cpu_0** (подвійним натисканням лівої кнопки миші на його імені). З меню, що випадає, для **Reset Vector** виберіть значення **onchip_memory2_0**. І для **Exception Vector** – значення **onchip_memory2_0** (рис. 8.26). Натисніть кнопку **Finish**.

20. Перейдіть на сторінку **System Generation** утиліти SOPC Builder. Перевірте, щоб опція **Simulation. Create project simulator files** залишилася не вибраною. Збережіть всі налаштування процесорної системи (меню **File** → **Save**). Натисніть кнопку **Generate**.

21. Після отримання повідомлення про успішне завершення створення процесорної системи, закрийте утиліту SOPC Builder.

22. Створіть в САПР Quartus II файл верхнього рівня ієрархії. Це може бути графічний файл або файл на мові HDL. У ньому повинні бути вказані зовнішні сигнали, які подаються на процесорний модуль (тактова частота, сигнал скидання, вхідні сигнали з перемикачів і вихідні сигнали на світлодіоди).

23. Додайте в файл створений процесорний модуль (рис. 8.27). Він знаходиться в робочій директорії проекту. Даний приклад – створення графічного файлу.

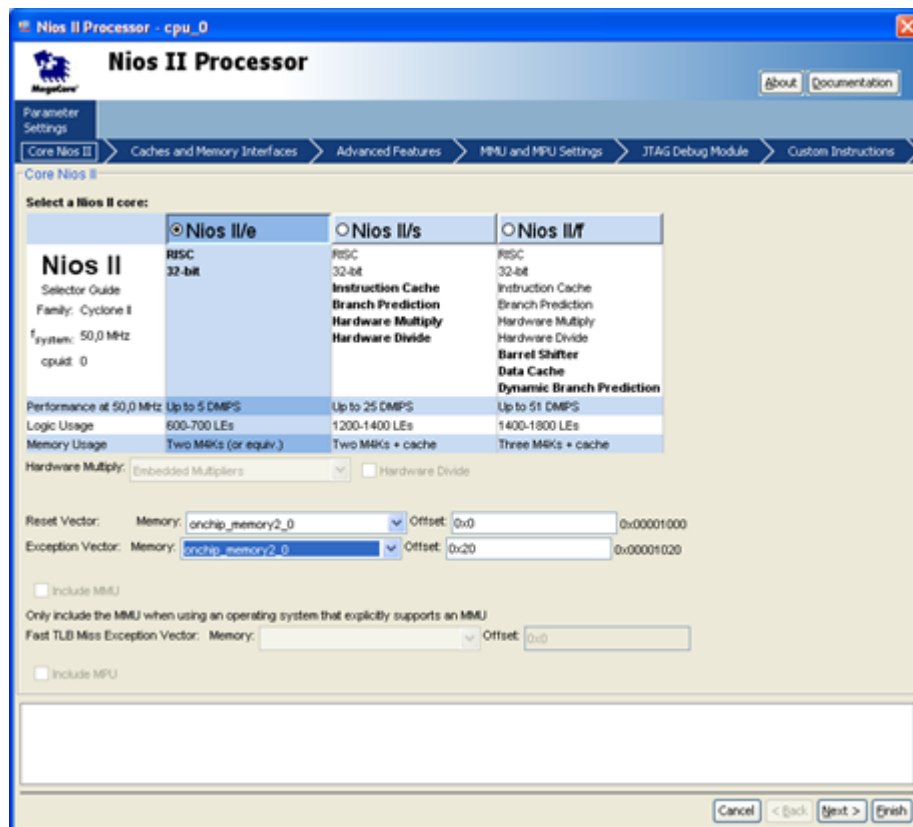


Рисунок 8.26 – Налаштування векторів переривання процесора

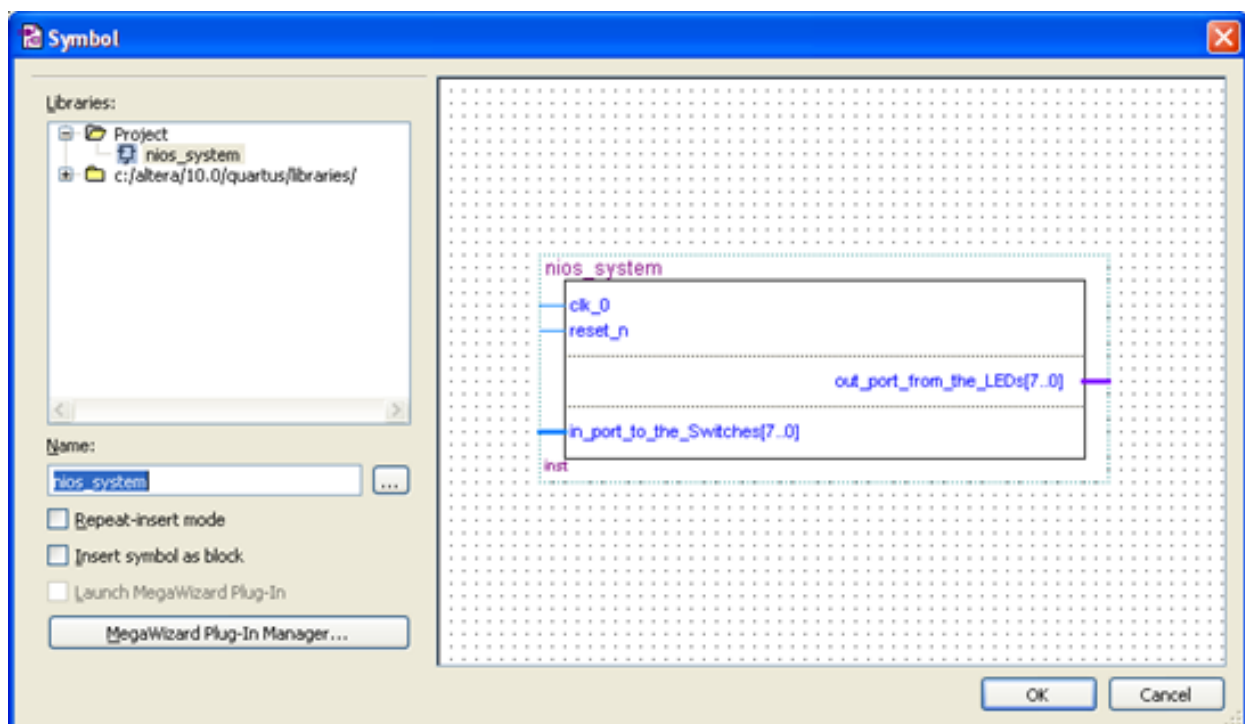


Рисунок 8.27 – Додавання в проект процесорного модуля **nios_system**

Для того, щоб скористатися файлом з призначеннями контактів вводу/виводу, що надходить разом з відлагоджувальним модулем DE2, імена зов-

нішніх сигналів у Вашому файлі повинні збігатися з іменами сигналів, зазначеними у файлі **DE2_pin_assignments.csv** (як показано на рис. 8.28). Збережіть файл (меню **File** → **Save**). Збережіть проект (меню **File** → **Save Project**)

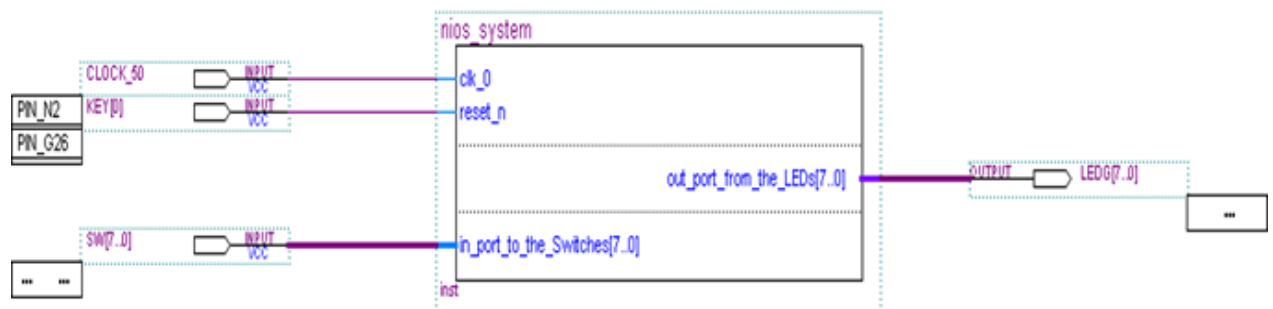


Рисунок 8.28 – Приклад файлу верхнього рівня ієрархії

24. Підключіть файл з призначеннями контактів вводу/виводу. Для цього в меню **Assignments** виберіть команду **Import Assignments**. У діалоговому вікні (рис. 8.29) вкажіть розташування файлу **DE2_pin_assignments.csv**. Натисніть **OK**.

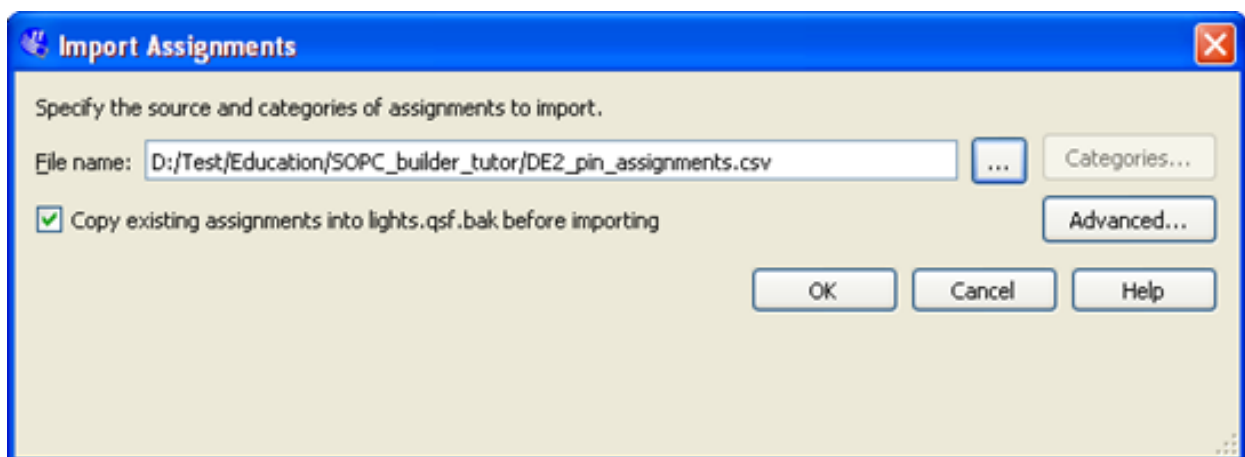


Рисунок 8.29 – Підключення файлу з призначеннями

25. Виконайте повну компіляцію проекту.

8.4 ПЛІС типу «програмована система на кристалі»

Основні відомості та визначення

Рівень інтеграції сучасних ВІС досягнув десятків мільйонів логічних елементів «2І-НІ», «2АБО-НІ», їх тактові частоти становлять 600...1000 МГц і більше. На таких кристалах можливе розмістити складну систему високої швидкодії (процесорну систему, пам'ять, інтерфейсні схеми та інше). Їх поява затирає межу між елементною базою і апаратурою.

Це стало поштовхом для того, що останнє десятиліття стрімко розвинувся новий напрямок проектування електронної апаратури, а саме, проектування апаратури у вигляді систем на кристалі (СнК, або SOC, про що говорилось раніше). У сучасній технічній літературі сформувався таке визначення СнК: система на кристалі – це НВІС, яка вміщує функціональні вузли закінченого пристрою, інтегрованого на кристалі, для автономного використання в електронній апаратурі. Типовим застосуванням таких мікросхем є різноманітні вбудовані системи. Для зручності застосування виробляються мікросхеми СнК, у яких остаточна структура системи та алгоритм її дії визначається користувачем. Це так звані програмовані системи на кристалі, ПСнК.

ПСнК може містити як цифрові, так і аналогові вузли. Основним цифровим вузлом зазвичай є процесор чи процесор обробки цифрових сигналів, що виконує програмне опрацювання цифрових даних та сигналів. Для виконання спеціалізованих функцій ПСнК має різні цифрові вузли та аналогові схеми, орієнтовані на конкретну галузь застосування ПСнК. Залежно від застосування ПСнК периферійними вузлами є різні таймери, лічильники, логічні елементи, ЦАП, АЦП, джерело опорної частоти, наприклад кварцові генератори та кола з оберненим зв'язком (PLL, phase-locked loops – цифрова система фазового автопідстроювання частоти) та інше. До складу ПСнК можуть входити або під'єднуватися до них у вигляді зовнішніх блоків різноманітні типи модулів пам'яті (SRAM, DRAM, Flash, ROM, EEPROM). Взаємодія із зовнішнім середовищем відбувається з використанням послідовних чи паралельних портів, а також комунікаційних інтерфейсів: USB, FireWire, Ethernet, UART, SPI, входів та виходів цифро-аналогових и аналого-цифрових перетворювачів, регуляторів напруги та стабілізаторів живлення.

Конфігурація вузлів ПСнК визначається їх функціональним призначенням. Організація зв'язків між вузлами системи може бути цілком різноманітною, включаючи використання стандартизованих шин (типу AMBA) чи спеціалізованих локальних інтерфейсів. Сьогодні системи на кристалі займають одну із найбільших частин ринку мікроелектроніки. Це зумовлено великим інтересом інженерів-розробників апаратури до використання ПСнК і виробників елементної бази ПСнК, зокрема Texas Instruments, Analog Devices, Altera, Atmel, Xilinx, Cirrus Logic, Cypress, NXP, RDC, Sharp, Marvell, NetSilicon та інших.

Переваги систем на кристалі:

- можливість отримання більше високих технічних показників (продуктивність, енергоспоживання, масогабаритні характеристики;

- більш низька вартість за крупносерійному виробництві;
- малі затрати на розробку і створення опитних зразків;
- можливість багатократної корекції проекту, менша ймовірність переробки плати;
- використання добре переварених серійних виробів;
- більш простий процес тестування і налаштування (можливість реалізації проекту і його налаштування частинами);
- вище надійність, ніж у наборів мікросхем із такою ж функціональністю;
- менша кількість складових спрощує монтаж готового виробу.

Але все ж таки, створення однієї дуже великої і складної системи на кристалі може виявитися дорожчим процесом, ніж серія малих через складність розробки, налагодження та зменшення частки придатних виробів.

Термін SOPC (System On Programmable Chip), тобто «система на програмованому кристалі», був уперше введений фірмою *Altera*. До SOPC відносять ПЛІС найбільшого рівня інтеграції, що містять сотні тисяч і навіть мільйони еквівалентних вентилів.

Ще додається можливість швидкої реконфігурації SOPC схем безпосередньо у працюючому пристрої, швидке відпрацювання прототипів і утворення принципово новітніх структур, які володіють багатофункціональністю і можливістю динамічного реконфігурування апаратних засобів.

Для зменшення працездатності розробки пристроїв створені бібліотеки схемних рішень – бібліотеки функціональних елементів подібні до тих, що використовуються при розробці спеціалізованих HBIC на основі БМК. Елементи бібліотек мають різні форми подання:

- soft-ядра або віртуальні компоненти. Це файли, які можливо інтегрувати до опису пристрою, що розробляється, на мовах HDL;
- firm-ядра, це варіант близький до першого, але його використовують для блоків, швидкодія яких особливе важлива. У цих ядрах схема задається більш жорстко і деякі міжз'єднання вже визначені.
- hard-ядра – це області кристала з наперед визначеними фіксованими функціями, їх виготовлено методами замовного проектування і ці ядра неможливо перепрограмувати.

Soft-ядра і firm-ядра ще називають "ядра (блоки) - інтелектуальної власності" (Intellectual Property, IP-core). Це програми завантаження ПЛІС для реалізації певних пристроїв, і вони придбаються окремо від кристала у якості самостійних продуктів високої вартості.

Hard-ядра отримують разом з мікросхемою, як її частину, і вони не є самостійним товаром.

У залежності від характеру, ядр що застосовуються, SOPC поділяють на два типи – однорідні і блокові.

Однорідні програмовані системи на кристалі. В однорідних програмованих системах на кристалі (у англomовній літературі вони позначаються назвою *generic*) застосовуються *soft-ядра* або *firm-ядра*. Різні функціональні блоки формуються на основі ідентичних схемотехнічних засобів внаслідок їх програмування, використовуючи при розробці так звані «одиниці інтелектуальної власності» IP (*Intellectual Properties*), тобто заздалегідь реалізовані параметризовані мегафункції. Тільки незапрограмований кристал є однорідним. Усі частини кристалу містять однотипні програмовані схемні ресурси.

Однак визначення «однорідні» належить основній частині кристала NBIC, окремі її частини допоміжного характеру можуть містити «неоднорідності», наприклад, вбудовану пам'ять, блоки керування синхросигналами типу PLL або DLL та інші деякі пристрої, що не виводять SOPC із класу однорідних. Данні, що завантажуються до пам'яті конфігурації для формування потрібного блоку (*soft-ядра* або *firm-ядра*) поставляються багатьма фірмами, але мають достатньо високу ціну.

Приклади IP рішень :

- декодери для цифрового телебачення;
- алгоритми кабельних, безпроводних мереж;
- IP рішення для ЦОС;
- перетворювачі Фур'є;
- фільтри;
- пам'ять і її компоненти;
- математичні функції – суматор, дільник, віднімач;
- процесори PicoBlaze і MicroBlaze з архітектурою Soft Processor (фірма Xilinx) і розрядністю 8...32;
- ядро процесора NIOS – це 16/32 – розрядний RISC – процесор зі змінюваною конвеєрною архітектурою (фірма Altera).

В однорідних SOPC блоки, що реалізуються, можна розташовувати у різних областях кристалу за допомогою їх програмного конфігурування.

Для проектування, розробки та комплексного налагоджування пристроїв на ПЛІС, в тому числі систем-на-кристалі компанії виробники мікросхем пропонують системи автоматизації проектування (САПР). САПР це інтегрована середовище розробки, яка реалізує всі етапи створення цифрового пристрою на базі ПЛІС, включаючи розробку проекту, синтез та моделювання, трасування, завантаження в кристал, тестування та налагоджування, в тому числі внутрісистемне. Зазвичай інтегровані програмні середовища підтримують всі сімейства мікросхем відповідного виробника. Так компанія Xilinx поставляє свої продукти разом з САПР Xilinx ISE (*Integrated Synthesis Environment*) [11].

Процесорні IP-ядра. Найбільш широко застосовують програмовані процесорні ядра, що випускаються відомими компаніями Xilinx та Altera.

Процесорне ядро Nios II (NII) компанії Altera – це тридцяти двох розрядне програмоване процесорне ядро є розвитком попередньої шістнадцятирозряд-

дної версії програмованого процесору Nios. Для взаємодії процесорів між собою та периферією на кристалі застосовується достатньо продуктивне шинне рішення Avalon. Основні особливості шини Avalon – простота архітектури, відповідно чого для реалізації її логіки задіяні мінімальні ресурси кристалу. Шина повністю синхронна, не підтримує передачі даних пакетами і можливість роботи в режимі «multi-master». В одному такті роботи один з процесорів працює в режимі «master» («Головний» процесор), інші пристрої системи – в режимі «slave» («Підлеглий» процесор). За одну транзакцію шина Avalon передає один байт, слово або подвійне слово (8, 16, або 32 біта) між одним з пристроїв «slave» і пристроєм «master».

Програмований процесор Nios II підходить для реалізації широкого кола обчислювальних додатків, від цифрових сигнальних процесорів (DSP, Digital Signal Processor) до систем управління.

Nios II має три непараметризовані модифікації.

Nios II/e (Economy) – 6-CPI процесор (CPI, Cycles Per Instruction, кількість тактів, необхідних для виконання однієї інструкції), мінімізований, неконвеєрний оптимізований за розміром, з програмно-реалізованим множенням;

Nios II/s (Standart) – 5-ступеневий конвеєр, репрезентує баланс між розміром та функціональністю, з апаратно реалізованим множенням, кешем команд;

Nios II/f (Fast) – потужний 6-ступеневий конвеєр, оптимізований для досягнення максимальної продуктивності, з динамічним розгалуженням, кешем команд та даних, опціонально ділильним пристроєм, обробкою зовнішніх переривань, опціонально пристроєм управління пам'яттю (MMU, Memory Management Unit) або пристроєм захисту пам'яті (MPU, Memory Protection Unit) [20].

Розробникам систем-на-кристалі на базі ядер Nios II потрібна система автоматизації проектування EDA (Electronic Design Automation) SOPC Builder (System on a Programmable Chip Builder). Пакет SOPC Builder – спеціалізоване програмне забезпечення компанії Altera, призначене для автоматичного проектування, безпосередньо орієнтованого на розробку систем на кристалі. Він включає набір різноманітних HDL-модулів: контролери пам'яті, інтерфейси, периферійне обладнання, тощо, на базі яких конфігурується система-на кристалі. Модулі процесорного ядра Nios II та шини Avalon мають засоби взаємодії з програмним середовищем SOPC Builder для автоматичної генерації ядра системи, шини і периферійних пристроїв.

Всі спеціалізовані програмні засоби автоматичного проектування, такі як SOPC Builder, утиліти, додаткові модулі та IP вузли, в тому числі модуль Avalon та процесорне ядро Nios потребують використання системи автоматизації проектування Quartus II, яку компанія Altera поставляє з усіма своїми продуктами. САПР Quartus II, забезпечує повний цикл проектування та розробки електронних пристроїв на кристалах ПЛІС, підтримуючи всі можливості продуктів компанії.

Процесорне ядро MicroBlaze фірми Xilinx – це тридцяти двох розрядний RISC програмований процесор з гарвардською архітектурою, за якої процесор

має розділені пам'ять команд і пам'ять даних. Щодо продуктивності, MicroBlaze може формувати нову командну інструкцію в кожному такті і підтримувати таку продуктивність у більшості випадків. Для реалізації вводу/виводу застосовується загальна шина CoreConnect On-Chip Peripheral Bus, яка може функціонувати в режимах «master» або «slave» та забезпечувати безпосереднє підключення великої кількості периферії та IP-блоків як компанії Xilinx, так і сторонніх виробників. Кожний процесор MicroBlaze має засоби Fast Simplex Link (FSL) для реалізації ефективного з'єднання точка-точка (point-to-point) для додавання спеціально створених співпроцесорів для прискорення обчислень.

Мультипроцесорні системи з ядром MicroBlaze конфігуруються за застосування інтегрованого пакету компанії Xilinx Embedded Development Kit (EDK). Цей пакет включає програмні засоби, документацію та IP-ядра і застосовується для розробки вбудованих систем на основі кристалів ПЛІС компанії Xilinx з вбудованими апаратними ядрами процесора PowerPC та програмованими процесорами MicroBlaze.

Процесорне ядро PicoBlaze – це восьмирозрядний RISC мікроконтролер, оптимізований для всіх сімейств мікросхем Xilinx. Компанія безкоштовно постачає процесорні ядра у вигляді вихідного коду на мові VHDL разом із спеціальним середовищем для розробки, але надана ліцензія дозволяє їх використання тільки з пристроями Xilinx. В якості прикладу, версія процесорного ядра PicoBlaze KCPSM3 займає всього 96 логічних блоків ПЛІС Spartan-3. Це 12,5% площини кристалу XC3S50, та 0,3% – кристалу XC3S5000. У типовій реалізації один блок оперативної пам'яті вміщує 1024 програмних інструкції, які автоматично завантажуються під час конфігурації процесорного ядра на кристалі. Навіть така проста конфігурація дозволяє досягати швидкості від 44 до 100 MIPS (мільйон команд за секунду) в залежності від сімейства ПЛІС.

В однорідних SOPC можливо конфігурувати на кристалі області одного або декількох процесорів, пам'яті і багатьох інших периферійних схем. Розробник сам розміщує на кристалі необхідні soft-ядра і firm-ядра. Можливо об'єднати на одному кристалі віртуальні компоненти від різних розробників. Однорідні SOPC характеризуються великою гнучкістю і універсальністю застосування. Виникає тільки проблема їхнього придбання внаслідок великої вартості. В однорідних SOPC не можна досягти максимальної швидкодії ядр.

Їх орієнтовано головним чином на використання у сферах телекомунікації, мережевих додатках, обробки зображень та мови і інших мультимедійних застосувань.

Блокові програмовані системи на кристалі

У блокових SOPC є окремі спеціалізовані області кристала, апаратні ядра (hard-ядра), які виконують жорстко визначені функції. SOPC блокового типу містять програмовані і фіксовані області. У фіксованих областях створено блоки з визначеними функціями. В інших областях кристалу розміщується програмована користувачем частина типу FPGA.

Характер і складність апаратних ядр протягом часу змінюється. Спочатку апаратні ядра виконували відносно прості функції, зараз основним апаратним ядром служить мікропроцесор або мікроконтролер. Апаратні ядра забезпечують роботу на максимально можливих частотах.

Hard-ядра реалізують блоки, які утворені методами проектування замовних схем. Такі блоки у порівнянні з soft-ядрами займають на кристалі значно меншу площу (у декілька разів), тому що вони не мають засобів конфігурування і оптимізовані для виконання конкретної функції.

Якщо прийняти площу, яку займає цифровий пристрій, утворений згідно методу «на стандартних чарунках», за одиницю, то площа такого же пристрою виготовленого згідно технології LPGA (БМК з лазерним програмуванням) становить у середньому 3,3 одиниці, для технології MPGA (БМК з масковим програмуванням) – 1,6 одиниць, а для НВІС програмованої логіки – 1020 одиниць. Згідно з цих причин швидкодія hard-ядр на 20...50% вища ніж швидкодія soft-ядр.

Визначеність hard-ядр знижує універсальність мікросхеми (зменшує її функціональну гнучкість) і зменшує коло споживачів. Hard-ядра фіксовані на площі кристалу, що утруднює розміщення і розведення міжз'єднань, для областей які конфігуруються. Це перешкоджає максимальному використанню програмованих логічних ресурсів і досягнення їх максимальної швидкодії.

Для hard-ядр обирають тільки широко поширені функціональні блоки. У першу чергу до них належать блоки ОЗП невеликої ємності, у яких можливо змінювати організацію пам'яті, режими роботи (синхронний або асинхронний), реалізовувати буфери FIFO і двопортову пам'ять. Апаратне ядро ОЗП ємністю 256...512 біт займає площу на кристалі, в десять разів меншу, ніж площа, яка потрібна була б для синтезу такого ж ОЗП засобами звичайної програмованої логіки. Крім того, при цьому в кілька разів підвищується швидкодія такого ОЗП. Для підвищення універсальності і розширення ринку збуту ядра ОЗП виконуються невеликої ємності.

Наступними блоками є апаратні перемножувачі для ЦОС. Так, перемножувач двох 8-бітних слів займає площу, рівну 1/5 площі, що вимагається для реалізації такого ж помножувача за допомогою логічних блоків FPGA.

Але найефективнішим напрямком успішного застосування апаратних ядр є інтерфейсні вузли – схеми інтерфейсу JTAG, контролери шини PCI (потрібні у деяких додатках, що вимагають максимальної швидкодії), UART, SPI, CAN, Ethernet і тощо. І нарешті у складі апаратних ядр з'явилися мікропроцесори і мікроконтролери. Сферою застосування блокових SOPC є системи найвищої швидкодії.

Блокові SOPC можна поділити на дві великі групи:

- блокові SOPC, що не мають процесорних ядр;
- блокові SOPC, що містять процесорні ядра.

Перша група ППСнК спеціалізовані і орієнтовані на ті чи інші конкретні застосування.

Друга група мікросхем є дійсно універсальною. Вона містить повний набір блоків, що використовуються у мікропроцесорній системі. Це насамперед цифрові блоки, а деякі ПСнК включають цифрову частину і аналогові блоки для введення, попередньої обробки і наступного оцифровування аналогових сигналів.

Системи на кристалі з блоковою структурою можна класифікувати за різними ознаками:

- за типом процесорного ядра: ARM, MIPS, PowerPC, x86 та інші;
- за продуктивністю ядра та частоті системної шини;
- за набором інтерфейсів;
- за вартістю кристала та вартістю мінімальної кількості компонентів для його підключення;
- за призначенням кристала з точки зору виробника.

Інженер-розробник повинен враховувати:

- доступність ППСнК;
- цикл виробництва кристала, що планується;
- експлуатаційні характеристики розробляемого виробу;
- технологію друкованих плат і способи монтажу компонентів;
- повноту документації і технічної підтримки і тощо.

У мікросхемах SOPC з hard-ядрами процесорів (фірми Atmel, Altera, Xilinx, Cypress Semiconductor та інші) використовувалися 8- та 32-розрядні процесорні ядра. В перших SOPC це були широко визнані 8-розрядні разрядные CISC-процесори 8051 фірми Intel і RISC-процесори AVR фірми Atmel. Через деякий час розробники отримали 32-розрядні ядра, які були пристосовані до базового технологічного процесу виробництва мікросхем FPGA – схеми RISC-процесорів ARM, MIPS и PowerPC.

Перші процесорні hard-ядра з'явилися у відносно нескладних мікросхемах SOPC. У нинішній час hard-ядра процесорів реалізуються у широкому діапазоні складності, вартості та інше. Застосовуються як спеціально розроблені ядра, так узяті з процесорів та мікроконтролерів, що вже вживають.

Серед 8-розрядних визнані ядра Intel 8051 (CISC-процесор з повним набором команд), AVR (фірма Atmel), M8C (фірма Cypress Microsystems). Серед 32-розрядних процесорних ядер найбільш відомі ARM (ARM Holdings), PowerPC (IBM та інші), MIPS (MIPS Technology), ARC і Tensilica.

Ядра ARM9T. Ядра ARM9T — скалярні RISC-процесори, орієнтовані на архітектурне сімейство ARM9 Thumb системи команд ARM ISA (Instruction Set Architecture). Більшість команд є обумовленими (Conditional), що виконуються за певних умов, які визначаються самою командою. Це зменшує кількість розгалужень шляхів виконання програми і за тим підвищує продуктивність процесора. Процесор має п'ятиступеневий конвеєр, більшість операцій виконується за один такт, тактова частота може змінюватися від нуля до 200 МГц, кеші команд і даних мають об'єми 16 Кбайт, використовуються з системною шиною AMBA. (0,18 мкм).

Ядра PowerPC 405C. Назва походить від Performance Optimization With Enhanced RISC. Ядро PowerPC 405C — скалярний RISC-процесор з кеш-пам'яттю команд та даних і тактовою частотою до 380 МГц (для версії з проектними нормами 0,18 мкм і напругою живлення 1,8 В) з гарвардською архітектурою. Це об'єднана розробка фірм Apple, IBM, Motorola. Ядро характеризується розвиненою системою команд у межах концепції RISC і можливостями, якими володіють процесори більш високого рівня. Процесор ґрунтується на ISA PowerPC і є представником класичних RISC з архітектурою Load/Store. Процесор має п'ятиступеневий конвеєр, більшість операцій виконується за один такт (але не команди множення і ділення) і підтримує шину CoreConnect от фірми IBM. Реалізовані черги команд для двох виконавчих блоків (ALU і MAC), конфігуровані розміри кешів (від їх відсутності до розмірів 8, 16 або 32 Кбайта), механізми віртуальної пам'яті, два пристрої для виконання математичних операцій (ALU і MAC), блок JTAG.

Ядра MIPS32 4Кс. Ядро MIPS (Microprocessor without Interlocked Pipeline Stages, що означає "мікропроцесор без затримок очікування конвеєра") фірми MIPS Technologies ліцензовано багатьма фірмами. Містить кеші команд і даних, має п'ятиступеневий конвеєр, апаратні засоби для виконання операцій множення ті ділення, множення з накопичуванням. Використовуються з системною шиною AMBA. Робоча частота ядра 200 МГц (для технології 0,25 мкм). Передбачено додавання кешей команд та даних, блока тестування з інтерфейсом JTAG. Процесорне ядро MIPS32 4Кс орієнтовано для застосування у SOPC у якості як hard-ядр, так і soft-ядр.

Системні шини. Для ефективної взаємодії процесорних ядер з головною пам'яттю, периферією і пристроями, які релізовані у FPGA, потрібні системи шин високої продуктивності. Тому розробники орієнтувалися на дві шинні системи: AMBA от фірми ARM і CoreConnect от фірми IBM.

У шинній системі AMBA існує 3 шини: дві системні, що мають назву АНВ (AMBA High-Speed Bus) і АСВ (AMBA System Bus) та периферійна АРВ (AMBA Peripheral Bus). Системні шини є швидкісними, вони мають окремі лінії для адреси та даних. Більш пізня розробка шини АНВ має удосконалену архітектуру, більшу швидкодію та підтримує передавання пакетів даних.

Шинна система CoreConnect складається з локальної процесорної шини PLB (Processor Local Bus) і периферійної шини OPB (On-Chip Peripheral Bus). Шини не мультиплексуються, одночасно підтримують читання та запису (мають окремі шини даних для цих режимів), налаштовуються версії відмінних розрядностей (від 16 до 128). Частота тактування PLB для відмінних розрядностей становить 66, 133 і 183 МГц.

Сучасні архітектури SOPC оцінюються не по кількості еквівалентних логічних елементів (логічних вентилів) або тригерів: популярність SOPC залежить від функціональності, доступності ресурсів, продуктивності та ефективності системи взагалі. Найбільшу важливість отримує оптимальний набір функціональних блоків, можливості масштабування та конфігурування.

Для переважної більшості розробників створення системи на одному кристалі здійснено тільки на основі ПЛІС, тому що замовлене проектування по економічним міркуванням можливо тільки при масовому виробництві.

За своїми архітектурними ознаками SOPC, як правило, відносяться до структур комбінованого типу, у яких поєднуються ознаки FPGA і CPLD з перевагою ознак FPGA.

Сімейство Cyclone V.

Для прикладу розглянемо ПСНК сімейства Cyclone V фірми Altera. ПЛІС сімейства Cyclone розроблені для застосування у якості компонентів недорогих систем, що працюють з великими об'ємами даних, наприклад, Інтернет додатків та вбудованих систем у промисловості і автомобілебудуванні. ПЛІС цього сімейства містить запам'ятовуючий пристрій даних конфігурації, які завантажуються кожен раз при вмиканні джерела живлення. Останні покоління сімейства – це Cyclone V і Cyclone 10.

Мікросхеми сімейства Cyclone V включають ПСНК з блоковою структурою типу FPGA і ПСНК з вбудованими процесорними блоками типу Cyclone V SoC FPGA.

Сімейство ПЛІС Cyclone V виготовлено по 28-нм технології. Основою масиву програмованої логіки є адаптивні логічні модулі, АЛМ (Adaptive Logic Module, ALM), які застосовують у більш дорогих серіях Arria і Stratix (рис. 8.30).

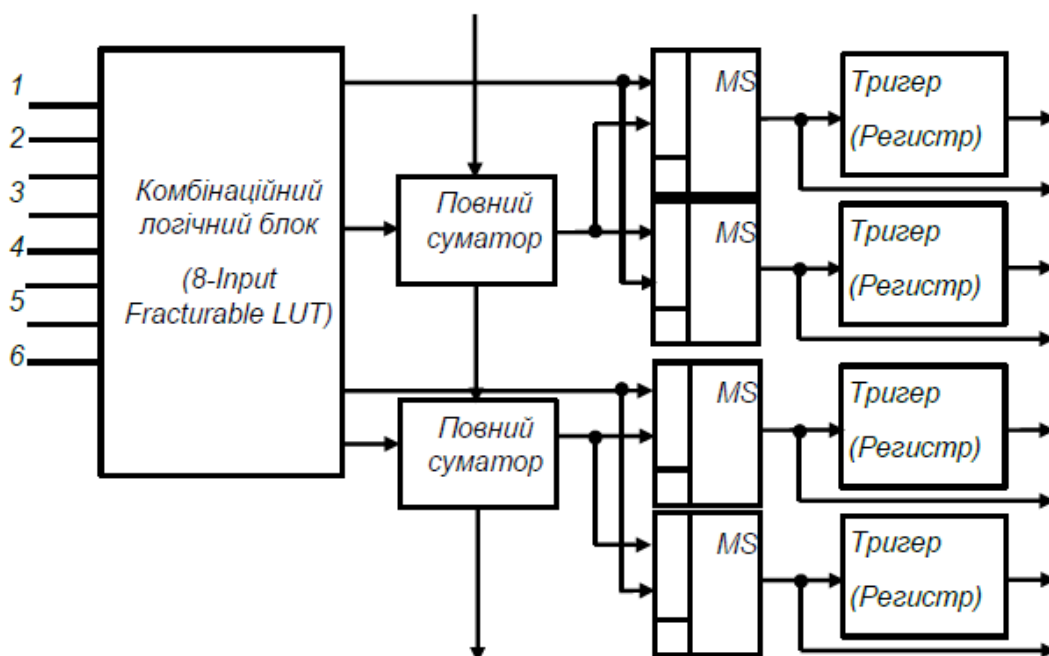


Рисунок 8.30 – Схема адаптивного логічного модуля

Окрім цього мікросхеми сімейства Cyclone V містять, DSP-блоки, змінної точності, вбудоване ОЗП, високошвидкісні трансивери, апаратні блоки (конт-

ролери PCI Express і зовнішньої синхронної пам'яті) та засоби захисту проекту від несакційного копіювання та модифікації (рис. 8.31).

Для підвищення функціональних можливостей, продуктивності, енергозбереження, надійності і полегшення проектування систем на кристалі розміщено апаратну процесорну систему (Hard Processor System, HPS), основою якого є одно- або двоядерний процесор ARM Cortex-A9.

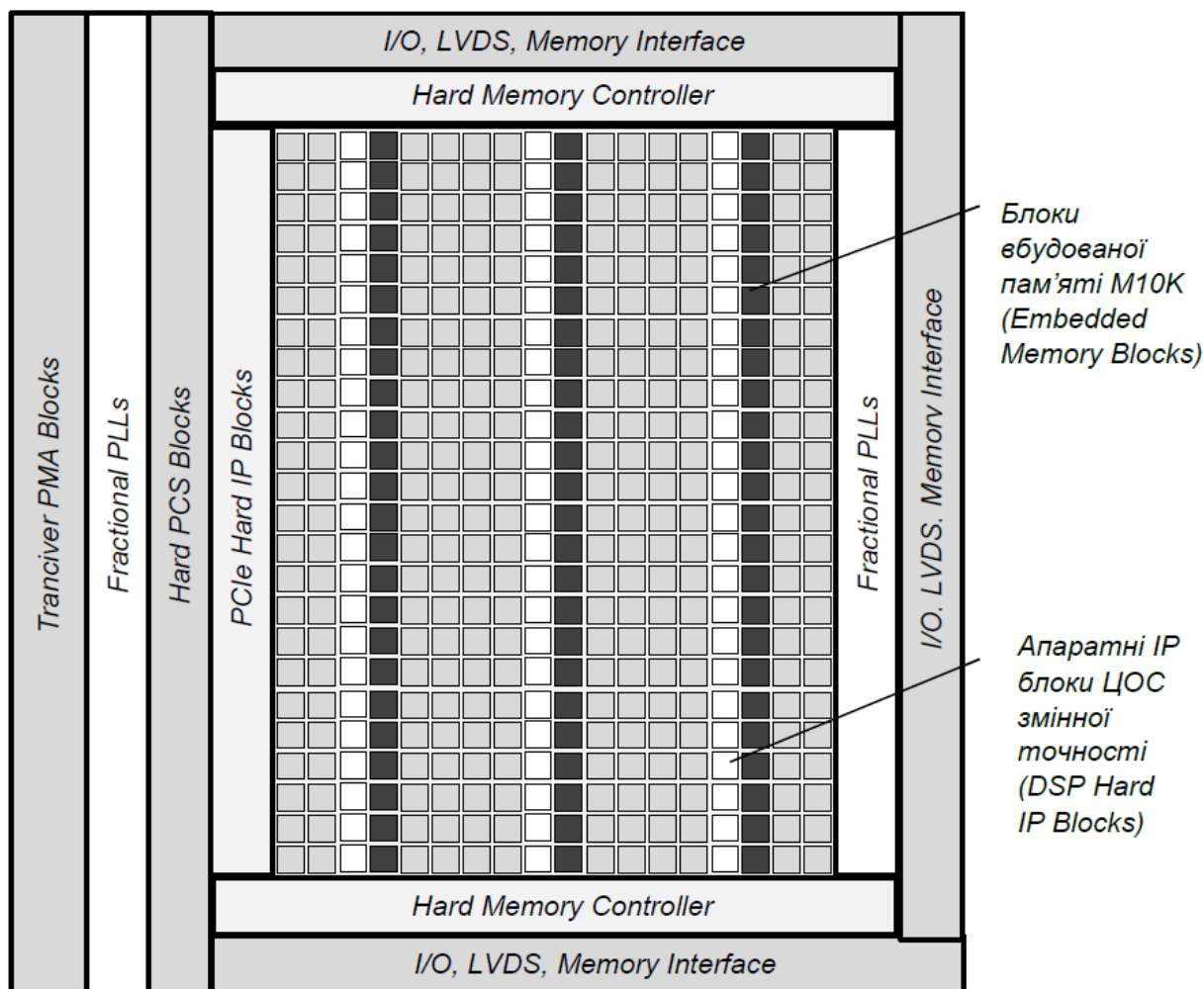


Рисунок 8.31 – Архітектура кристалу *FPGA Cyclone*

До ПСНК з вбудованими процесорними системами Cyclone V SoC FPGA (рис. 8.32) належать наступні підсімейства:

- Cyclone V SE – містить апаратний процесорний блок (одно- або двоядерний);
- Cyclone V SX – містить двоядерний апаратний процесорний блок і вбудовані трансивери з максимальною швидкістю 3,125 Гбіт/с;
- Cyclone V ST – містить двоядерний апаратний процесорний блок і вбудовані трансивери з максимальною швидкістю 5 Гбіт/с.

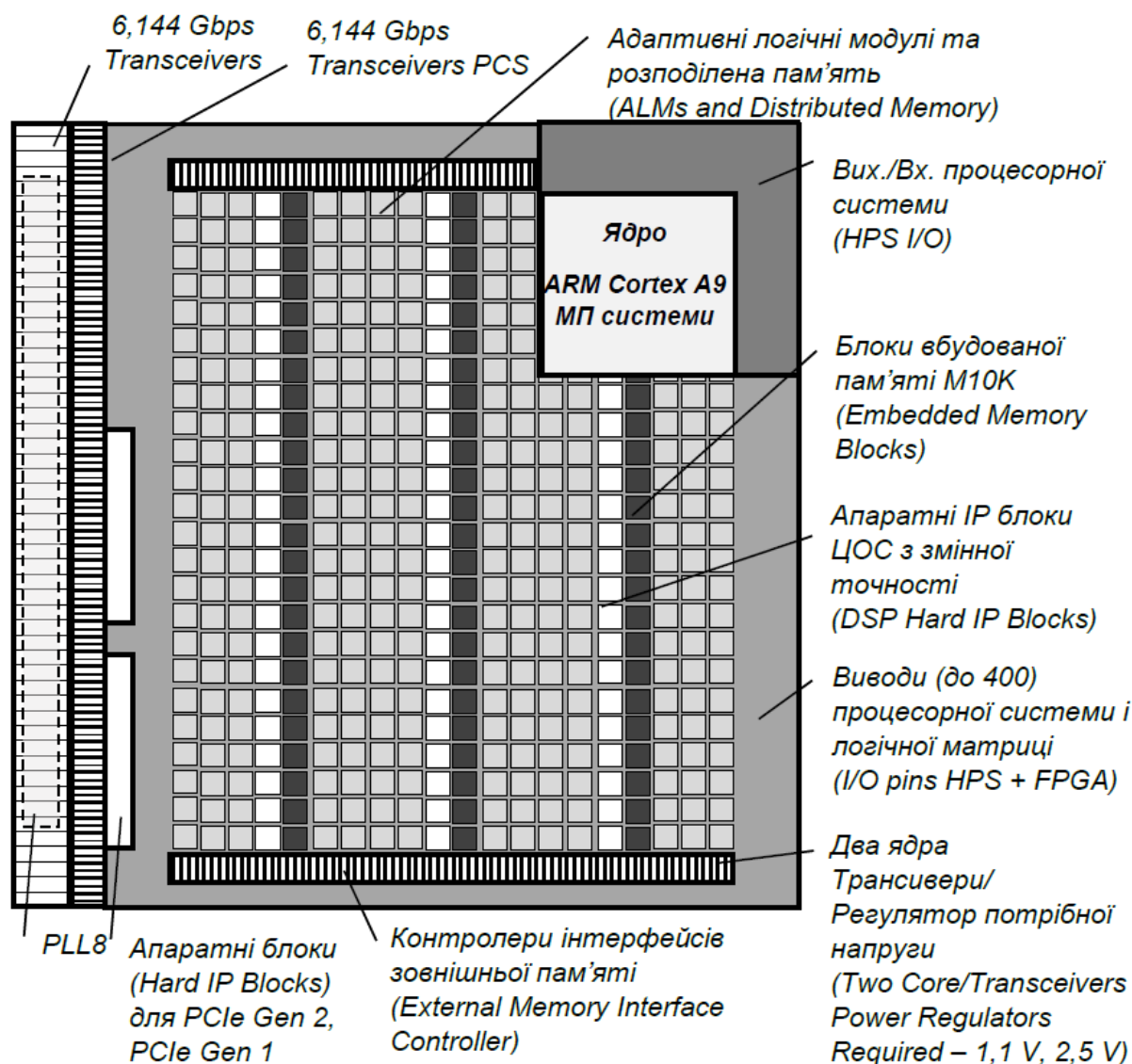


Рисунок 8.32 – Архітектура кристалу Cyclone V SoC FPGA

Незважаючи на низьку вартість, мікросхеми містять необхідний набір функціональних блоків для реалізації на їх основі цілої цифрової системи. Щоб наочно уявити обчислювальну потужність цих НВІС, перерахуємо весь набір ресурсів, що надаються розробникові цифрової системи мікросхемою Cyclone VSTD6:

- 41509 адаптивних логічних модулів;
- 110000 еквівалентних логічних елементів;
- 166036 тригерів;
- 514 блоків вбудованого ОЗУ;
- вбудоване ОЗУ об'ємом 5140 Кбіт;
- 512 математичних блоків обробки цифрових сигналів;
- 224 помножувача 18×18 ;
- двоядерний процесорний блок ARM Cortex-A9;
- 9 трансиверів зі швидкістю передачі 9 Гбіт / с;

- контролер зовнішньої пам'яті;
- 2 контролера PCI Express.

Процесорна апаратна система (Hard Processor System, HPS) побудована на базі одно- або двоядерних процесорів ARM Cortex-A9 MPCore з максимальною частотою до 925 МГц.

Вбудовані апаратні периферійні пристрої знімають потребу у необхідності використовувати для виконання цих функцій програмовану логіку зберігаючи більше ресурсів матриці FPGA для виконання спеціальних додатків користувача і зниження енергоспоживання.

Апаратні багатопортові контролери пам'яті, відділяючи процесор і FPGA логіку, підтримують різні типи ОЗП (DDR2, DDR3 і LPDDR2) та мають інтегровані пристрої корекції помилок кодів (error correction code, ECC) для високої надійності та безпеки критично важливих застосувань.

Широкосмугові міжз'єднання між HPS та матрицею FPGA забезпечують продуктивність обміну даними не досягну за вживанням двох кристалів окремо для процесора і для логічної матриці. Ця інтеграція забезпечує:

- пропускну здібність понад 100 Гбіт/с;
- когерентність даних інтегрованих на кристалі;
- значне енергозбереження в результаті ліквідації зовнішніх шляхів обміну даними між процесором і FPGA.

Гнучкість масиву програмованої логіки дозволяє розробляти різноманітні системи застосовуючи IP користувача або стандартні попередньо налаштовані IP від фірми Intel або від інших розробників. Це дозволяє:

- швидко адаптуватися до зміни проекту або зміни інтерфейсу і стандартів протоколу;
- додавати користувачем відповідне апаратне забезпечення масиву програмованої логіки для прискорення алгоритмів критичних до часу виконання;
- швидко запустити процесор без попередньої колосальної діяльності з програмування, тестування та інше, що зменшує одноразові витрати у порівнянні з мікросхемами ASIC.

Інтегрування на одному кристалі багатой кількості апаратних IP блоків знижує загальну вартість системи, споживану потужність і час розробки, підвищує продуктивність, надійність та гнучкість, системи що проектується. Мікросхеми Intel® SoC FPGAs призначені:

- для збереження гнучкості завантаження процесора і програмованої логічної матриці послідовністю конфігурування, системного скидання процесора, незалежності інтерфейсів пам'яті, які відповідають реалізації на двох кристалах;
- збереження цілості і надійності даних за допомогою блоків ECC;
- захист пам'яті DRAM відокремленням процесора і матриці з інтегрованими пристроями захисту пам'яті;
- вмикання налаштування системного рівня з адаптивним налаштуванням для усього пристрою.

Сімейство Cyclone 10.

Сімейство Cyclone 10 включає два типи мікросхем, що мають відносно невелику вартість. Мікросхеми Cyclone 10 GX орієнтовано на розробку високопродуктивних пристроїв: системи промислового зору, робототехніка, автомобільні системи, системи що працюють з Інтернетом. Інша модель Intel Cyclone 10 LP призначена для недорогих застосувань з малим споживанням потужності, наприклад, обробка невеликих потоків даних, комплекси давачів, контролери електроприводів та інше.

ПЛІС Cyclone 10 GX виготовлено за технологією 20 нм, забезпечують високу продуктивність (до 134 Гфлопс) і розширені можливості введення-виведення. Мікросхеми містять до 220 тис. логічних елементів, до 80 тис. адаптивних логічних модулів з 8-вхідними таблицями відповідності (LUT), до 192 DSP-блоків змінної точності. Підключення до інших пристроїв можливо з використанням LVDS- інтерфейсів з швидкістю передачі 1,4 Гбіт/с, мережевого порту 10G або шини PCI Express x4. В мікросхемах Cyclone 10 GX вбудовано 72 розрядний інтерфейс зовнішньої пам'яті (DDR3, DDR3L, LPDDR3), який працює на швидкості до 1 866 Мбіт/с. Напруга живлення ядра становить 0,9 В.

Мікросхеми Intel Cyclone 10 LP виготовлено за технологією 60 нм і оптимізована для низького енергоспоживання. Вона містить до 120 тис. логічних елементів, до 288 помножувачів 18×18 та до 230 каналів з LVDS- інтерфейсами. Потрібна напруга живлення 1,0 або 1,2 В.

Сімейство ПСНК Arria.

Перші HBIC сімейства Arria з'явилися в 2007 р і реалізовувалися на базі технології 90 нм, наступні HBIC реалізовані вже за технологією 28 нм та 20 нм. Це сімейство відносяться до ПЛІС середнього діапазону і призначені для вирішення телекомунікаційних задач з підвищеними вимогами за вартістю і енергоспоживанням. Крім цього, починаючи з сімейства Arria V, випущеного в 2011 р, у мікросхеми вбудовано апаратний процесорний блок, основою якого є двоядерний процесор ARM Cortex A9, і блоки тактування. Ці мікросхеми мають той же набір блоків, що і ПЛІС сімейства Stratix.

Останньою розробкою у цьому сімействі є мікросхеми середнього класу Arria 10, що виробляються з 20-нм технології і забезпечують баланс між продуктивністю та енергоспоживанням. Вони призначені для розробок, що потребують високої продуктивності, за жорсткими обмеженнями вартості та енергоспоживання. Основні сфери використання Arria 10 – телекомунікаційна апаратура і високопродуктивні обчислювальні системи.

ПЛІС Arria 10 включають до 1,15 млн еквівалентних логічних елементів, а також більше 3 300 апаратних помножувачів 18×19 . На кристалах розміщено апаратні контролери з підтримкою швидкісних інтерфейсів зовнішньої пам'яті (у тому числі DDR4 з швидкістю передачі 2 666 Мбіт/с та з кубом гібридної пам'яті з швидкістю передачі до 15 Гбіт/с), також контролери шини PCI Express Gen3 x8. Мікросхеми Arria 10 містять двоядерний процесор ARM Cortex A9. Пропускна здібність каналів зв'язку Arria 10 підвищилася у чотири рази у порівнянні з Arria 5 завдяки трансиверам зі швидкістю передавання даних до 25,78 Гбіт/с.

Сімейство Arria 10 включає три типи мікросхем.

ПЛІС Arria 10 GT – містять до 78 трансиверів зі швидкістю передавання даних до 25,78 Гбіт/с і до 1,15 млн. еквівалентних логічних елементів;

Arria 10 GX – ПЛІС, що містять 78 трансиверів зі швидкістю передавання даних до 17,4 Гбіт/с і до 1,15 млн. еквівалентних логічних елементів;

Arria 10 SX – системи на кристалі на базі двоядерного процесора ARM Cortex A9 (рис. 8.33), що містять 48 трансиверів зі швидкістю передавання даних до 17,4 Гбіт/с і до 660 тис. еквівалентних логічних елементів. Процесор працює на частоті до 1,5 ГГц.

Сімейство Arria 10 містить удосконалені апаратні DSP-блоки змінної розрядності, які забезпечують швидкість обробки даних до 1,5 Тфлопс.

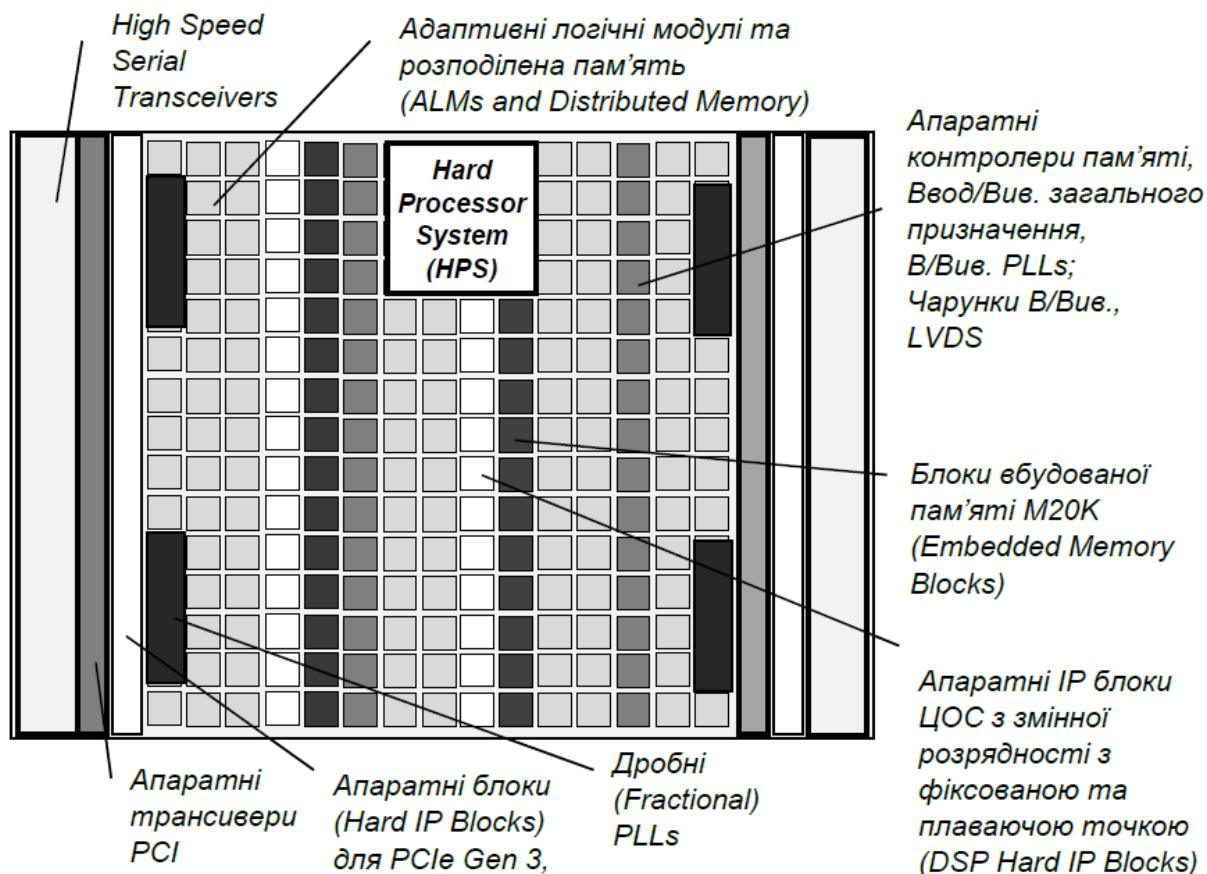


Рисунок 8.33 – Архітектура кристалу ПЛІС сімейства Arria® 10 SoC FPGA

Мікросхеми сімейства Stratix 10.

Сімейство Stratix 10 є найбільш продуктивним серед ПСНК фірми Intel Altera і містить чотири варіанти мікросхем типу система у корпусі (System in Package, SiP) – три різновиди мікросхем ПЛІС FPGA і мікросхему програмована система-на-кристалі SOPC FPGA, які призначені для використання у відповідних сферах сучасної радіоелектроніки: комунікація, дротовий та бездротовий зв'язок, обчислення, збереження інформації, медицина, випробування тощо.

Stratix 10 GX – ПЛІС для високошвидкісних застосувань з продуктивністю до 10 Тфлопс з підтримкою трансиверів зі швидкістю передавання даних до 28,3 Гбіт/с;

Stratix 10 SX – ПСнК на базі 64-розрядного процесора ARM Cortex-A53 плюс підтримка усіх функцій мікросхем Stratix 10 GX;

Stratix 10 TX – ПСнК на базі чотирьохядерного 64-розрядного процесора ARM Cortex-A53 з підтримкою H- і E-трансиверів, які забезпечують два режими роботи (56 Гбіт/с у режимі PAM 4 або 30 Гбіт/с у режимі NRZ), плюс підтримка усіх функцій мікросхем Stratix 10 GX и Stratix 10 SX;

Stratix 10 MX – ПСнК на базі чотирьохядерного 64-розрядного процесора ARM Cortex-A53 з підтримкою H- і E-трансиверів, які об'єднані у одному корпусі з високопродуктивною пам'яттю (HBM2) DRAM об'ємом до 16 Гбайт.

ПЛІС FPGA та ПСнК сімейства Stratix 10 створені за високопродуктивною архітектурою HyperFlex™ компанії Altera із застосуванням 14-нанометрового технологічного процесу Intel Tri-Gate, містить до 5,5 млн. еквівалентних логічних елементів, до 144 високошвидкісних трансиверів (до 56 Гбіт/с) та забезпечує продуктивність більше 10 Тфлопс у процесі цифрової обробки сигналів одинарної точності, що забезпечує у двічі вищу продуктивність у порівнянні з попередніми поколіннями.

Архітектура HyperFlex використовує тригери в усіх вузлах міжз'єднань логічної матриці, а не тільки у логічних елементах. Це дозволяє запобігти критичних затримок проходження та обробки сигналів і швидко узгоджувати частотно-часові характеристики системи, що розробляється. Зростання продуктивності логічної матриці у приблизно у два рази також дозволило підвищити ефективність використання системних ресурсів і знизити споживану потужність FPGA.

ПЛІС FPGA та SoCs сімейства Stratix 10 підтримують робочу частоту, що перевищує 1 ГГц і мають у два рази кращу продуктивність у порівнянні з існуючих 28 нм FPGA ПЛІС старших моделей. В системах з вимогами обмеження потужності ПЛІС сімейства Stratix 10 дозволяють розробникам скоротити споживання енергії на 70%, за умови збереження продуктивності пристрою на рівні попереднього покоління.

ПЛІС FPGA та SoCs сімейства Stratix 10 відрізняються найвищим рівнем продуктивності в промисловості та системної інтеграції і містять:

- більше 5,5 млн. логічних елементів на одному кристалі;
- високошвидкісні прийомопередавачі (до 144 одиниць) з пропускну здібністю до 56 Гбіт/с, що у чотири рази вище, ніж у попередніх пристроях;
- вбудовані апаратні блоки цифрової обробки сигналів (DSP), які мають продуктивність понад 10 трлн. операцій з плаваючою точкою у секунду (10 TeraFLOPs);
- апаратний 64-бітний 4-ядерний процесор ARM Cortex-A53 з робочою частотою до 1,5 ГГц (Stratix 10 SX SoCs);
- схеми підтримки швидкісного послідовного інтерфейсу пам'яті з пропускну здібністю понад 2,5 Тбіт/с;
- схеми паралельного інтерфейсу пам'яті з пропускну здібністю понад 2,5 Тбіт/с для підтримки ОЗП типу DDR4 SDRAM з частотою 2 666 Мбіт/с;

- інтегрований пристрій менеджера безпеки (Secure Device Manager, SDM) для гнучкості оновлення коду конфігурації, забезпечення повнофункціонального захисту даних та IP-ядр;
- функція заборони фізичного копіювання (Physically Unclonable Function, PUF);
- систему виявлення та знешкодження поодиноких збоїв (Single Event Upset, SEU), які викликані іонізуючим випромінюванням;
- сумісність коду з кодом SoCs попередніх генерацій;
- сумісність призначення виводів з мікросхемами ПЛІС FPGA та SoCs сімейства Arria 10, що забезпечує легку взаємозаміну мікросхем у пристроях.

До складу ПЧНК Intel® Stratix® 10 SoCs 9 (рис. 8.34) входить апаратна процесорна система (Hard Processor System, HPS) на базі 64-розрядного чотирьох ядерного процесора ARM Cortex-A53, що працює на частоті до 1,5 ГГц і логічна матриця HyperFlex.

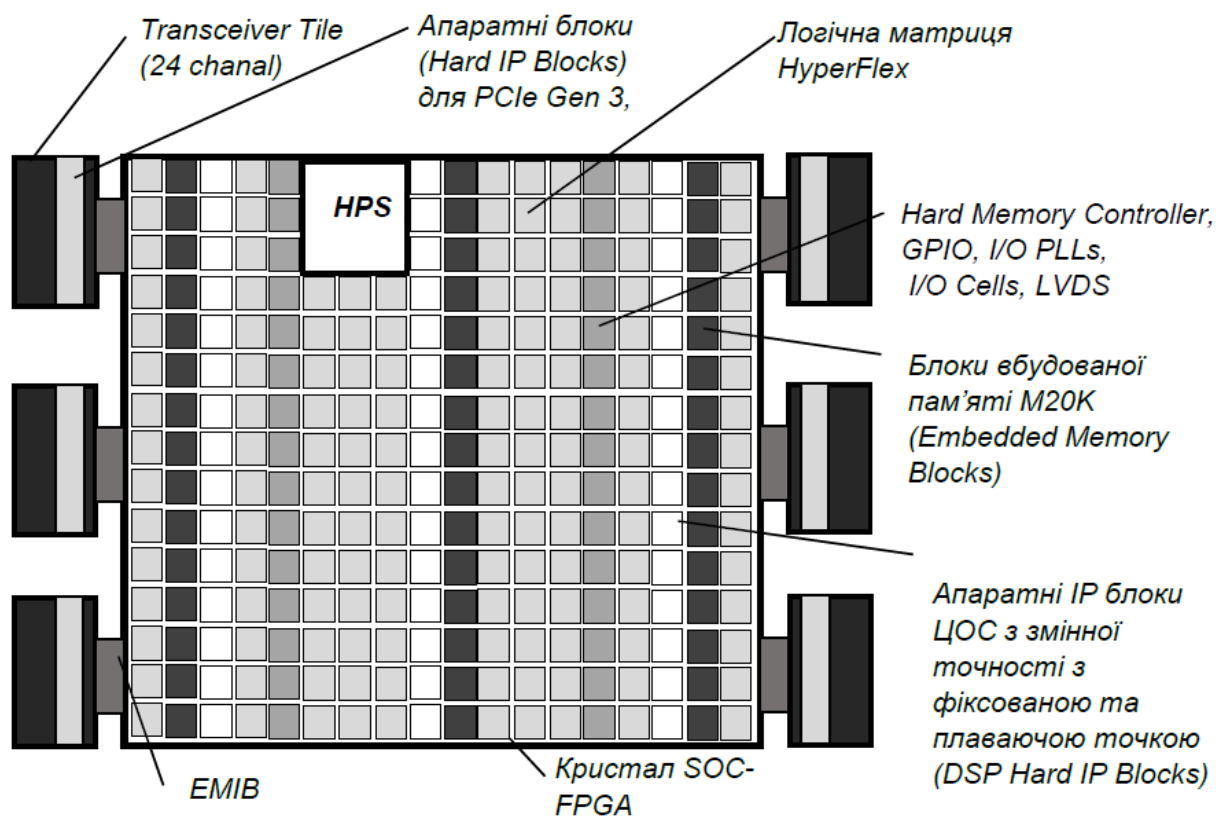


Рисунок 8.34 – Архітектура кристала ПЧНК типу Intel® Stratix® 10 SoCs

Процесор Cortex-A53 також підтримує 32 розрядний режим роботи. Блок керування конфігурацією та безпекою (Secure Device Manager, SDM) забезпечує багатобічний захист HBIC Stratix 10 від поодиноких збоїв (Single Event Upset, SEU), які викликані іонізуючим випромінюванням. Завдяки цьому Stratix 10 є оптимальним рішенням для систем військового призначення, безпеки «хмарних сховищ» та обчислювальних центрів, інших систем, що потребують високих вимог до багаторівневого захисту.

Об'єднання логічної матриці HyperFlex™ FPGA і процесора ARM Cortex-A53 на кристалі забезпечує ідеальне рішення для бездротового зв'язку G5, програмного забезпечення для сфери радіоприйому, безпечних обчислень у військових застосуваннях, для функції віртуалізації у мережах (network function virtualization, NFV), прискорення продуктивності центрів даних.

Всі HBIC сімейства Stratix 10 реалізовані за технологією інтеграції гетерогенних 3D система у корпусі (System in Package, SiP) забезпечує високу масштабованість, гнучкість використання і швидкість для створення нових рішень у сфері швидкісного передавання даних. Процес побудови гетерогенної 3D системи у корпусі ґрунтується на об'єднанні окремих кристалів з програмованою логічною матрицею по технології EMIB (Embedded Multi-die Interconnect Bridge), що розроблена фірмою Intel. У залежності від сфери застосування пристрою, що розробляється, у корпусі можливе розташування кристалу FPGA та кристалів, на яких розміщені швидкісні трансивери з відмінними засобами модуляції та з підтримкою нових стандартів передавання даних (PCIe Gen4, Multi-Port Ethernet), схеми пам'яті SRAM і DRAM, кристали з аналоговими схемами, мікросхеми ASIC, мікросхеми процесорів ПЧНК, кристали, які виконані з іншими технологічними нормами. Ці можливості ілюструють рис. 8.35, 8.36 та рис. 8.37.

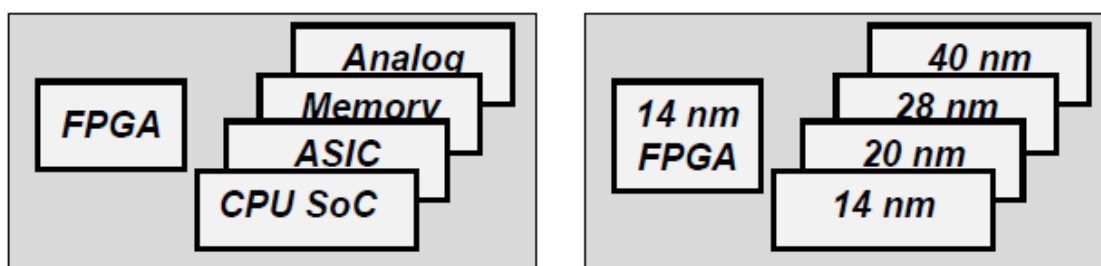


Рисунок 8.35 – Суміщення в одному корпусі кристалів відмінного функціонального призначення та з різними технологічними нормами

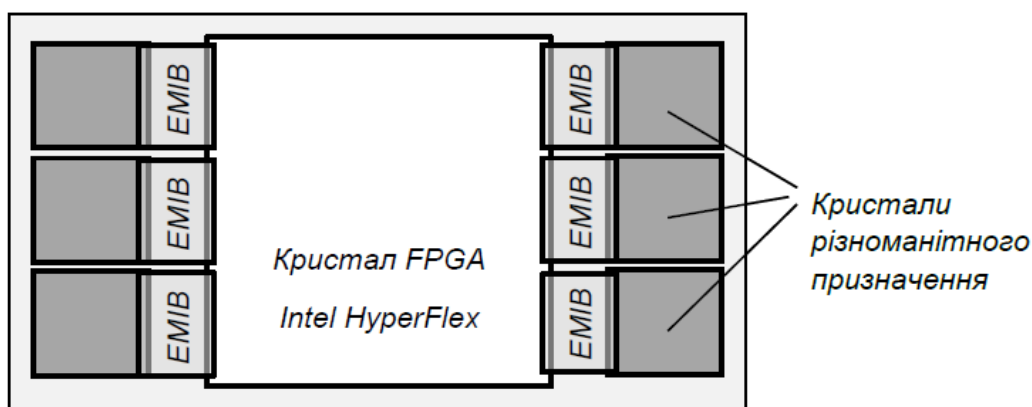


Рисунок 8.36 – Архітектура мікросхем Stratix 10 для різноманітних сфер призначення, що виконані за гетерогенної технологією Intel 3D SiP

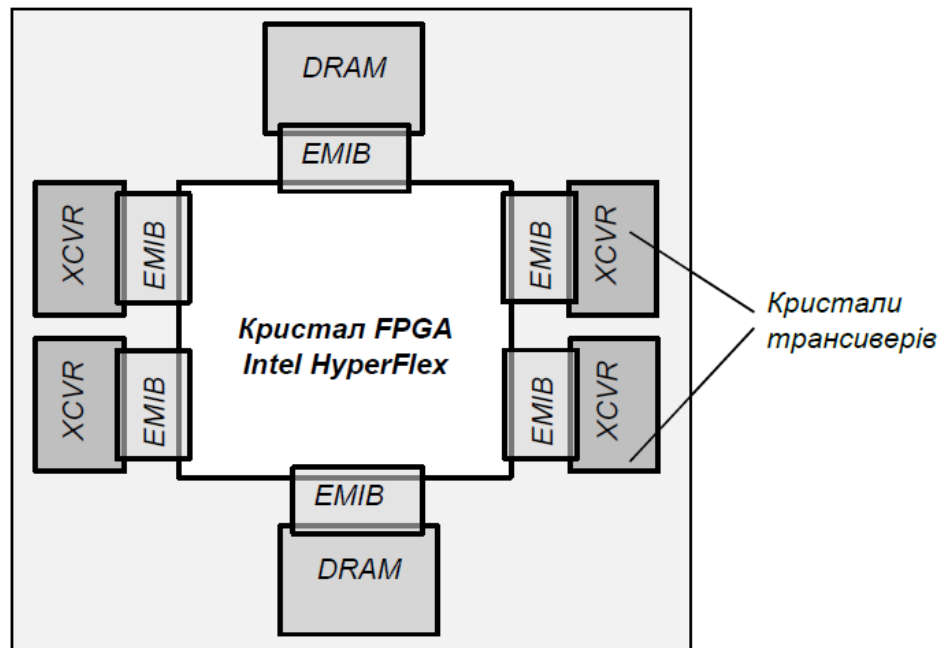


Рисунок 8.37 – Архітектура мікросхем *Stratix 10* для сфери телекомунікації, що виконані за гетерогенною технологією *Intel 3D SiP*

Технологія EMIB забезпечує значно більшу швидкодію та високу цілісність сигналів за меншої складності і вартості системи у порівнянні з альтернативними технологіями. Зв'язок між швидкодіючими пристроями відбувається через EMIB, не друковану плату (рис. 8.38, рис. 8.39).

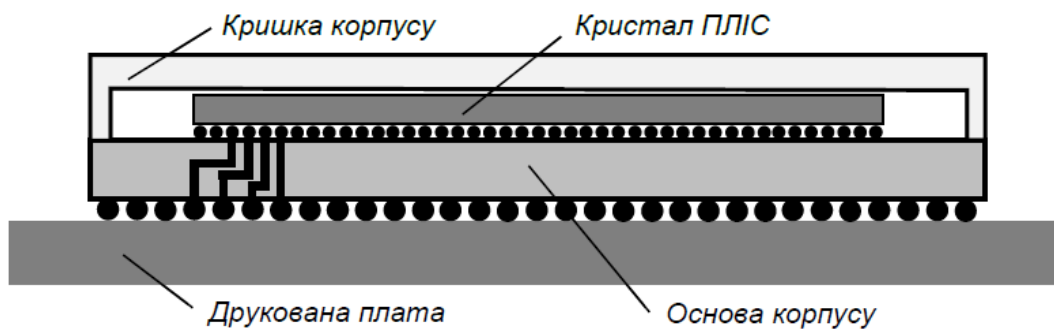


Рисунок 8.38 – Звичайний корпус ПЛІС

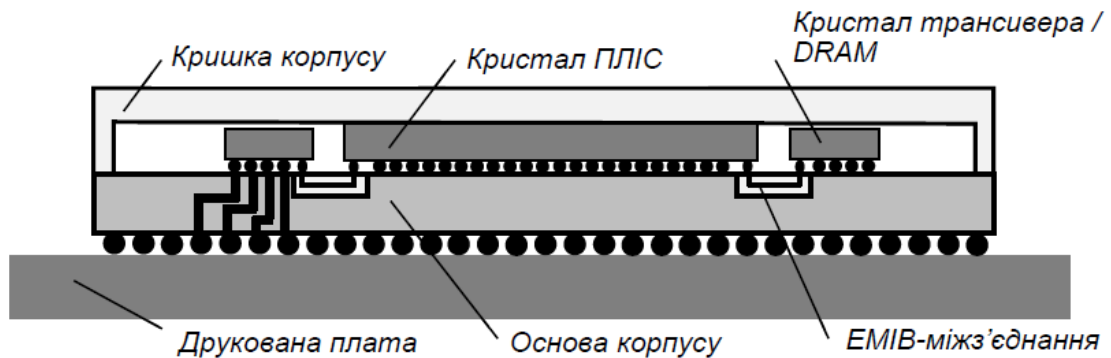


Рисунок 8.39 – Корпус ПЛІС *Stratix 10* з використанням технології *EMIB*

За рахунок цього здійснюється значне скорочення довжини ліній зв'язку, зменшення паразитних ємностей та індуктивностей та часу затримки поширення сигналів, підвищення однорідності ліній зв'язку між кристалами, зменшенню джитера та збільшенню когерентності сигналів. В результаті значно полегшується проектування друкованої плати і тестування розробляємої системи.

8.5 Контрольні запитання і завдання

1. Які існують шляхи вирішення задачі побудови системи керування?
2. Що являє собою клас пристроїв ASSP (Application-specific standard product)?
3. Які існують шляхи проектування та розробки системи керування?
4. Що являє собою узагальнена структура логічного елемента (LE) ПЛІС.
5. На яких елементах можна реалізувати логічну функцію?
6. Що означає LUT?
7. Для чого використовується логічний елемент (LE) з каналом перенесення?
8. Що представляє собою логічна структура елемента (LE) Cyclone IV?
9. Що таке Nios II?
10. Для чого використовується утиліта SOPC Builder?
11. Які компоненти повинен містити створюваний процесорний модуль?
12. Приведіть структуру логічного елемента (LE) зі зворотним зв'язком.
13. Приведіть приклад архітектури процесорного ядра з Nios II?
14. В яких режимах працює LUT в Cyclone IV?
15. Що таке LAB і його склад в Cyclone IV?
16. Яка структура з'єднань LAB в комутаційному полі ПЛІС?
17. Наведіть приклад сигналів управління.
18. Які функції карти ресурсів ПЛІС?
19. Які функції утиліти JTAG Debug?

20. Які функції утиліти Chip Planer?
21. Які існують пакети інструментів для розробника?
22. Який склад середовища розробки Quartus?
23. . Які чинники сприяли появі ПЛІС "SoC"?
24. Що таке *IP* - ядра і які функції вони реалізують?
25. Що таке *hard* - ядра?
26. Порівняйте між собою *IP* - ядра і *hard* - ядра.
27. Які ПЛІС відносяться до "SoC" з однорідною структурою?
28. Які особливості ПЛІС "SoC" зі вбудованими блоками?
29. Які функції реалізують вбудовані блоки?
30. Охарактеризуйте особливості ПЛІС різних фірм - виробників.
31. Які САПР використовують при проектуванні пристроїв на ПЛІС?
32. Назвіть основні сфери застосування ПЛІС.

9. МЕРЕЖИВІ ПРОТОКОЛИ ІНДУСТРІЇ 4.0

Невід'ємною частиною систем автоматизованого управління є мережі зв'язку, що поєднують між собою окремі компоненти комплексу засобів контролю та керування на виробництві.

Четверта промислова революція, більш відома як *Індустрія 4.0*, отримала свою назву від ініціативи 2011 року, очолюваної бізнесменами, політиками і вченими, які визначили її як засіб підвищення конкурентоспроможності промисловості Німеччини через посилену інтеграцію “кіберфізичних систем”, або CPS, в заводські процеси.

Цьому прикладу наслідували й інші країни, що активно освоюють нові технології. А термін *Індустрія 4.0* стали використовувати як синонім четвертої промислової революції. Суть її в тому, що сьогодні матеріальний світ з'єднується з віртуальним, в результаті чого народжуються нові CPS комплекси, об'єднані в одну цифрову екосистему. Роботизоване виробництво і “розумні” заводи – один з компонентів трансформованої галузі.

Було сформульовано кілька основних принципів побудови Індустрії 4.0, дотримуючись яких виробники можуть впроваджувати сценарії четвертої промислової революції на своїх підприємствах.

- Перший – це сумісність, що означає здатність машин, пристроїв, сенсорів і людей взаємодіяти і спілкуватися один з одним через Інтернет речей (IoT).

- Це веде до наступного принципу – прозорості, яка з'являється в результаті такої взаємодії. У віртуальному світі створюється цифрова копія реальних об'єктів, систем функцій, яка точно повторює все, що відбувається з її фізичним клоном. В результаті накопичується максимально повна інформація про всі процеси, які відбуваються з обладнанням, продуктами, виробництвом в цілому і так далі. Для цього потрібно забезпечити можливість збору всіх цих даних з сенсорів і сенсорів та обліку контексту, в якому вони генеруються.

- Технічна підтримка – третій принцип Індустрії 4.0. Суть його в тому, що комп'ютерні системи допомагають людям приймати рішення завдяки збору, аналізу та візуалізації всієї тієї інформації, про яку йдеться вище. Ця підтримка також може полягати в повному заміщенні людей машинами при виконанні небезпечних або рутинних операцій.

- Четвертий принцип – децентралізація управлінських рішень, делегування деяких з них кіберфізичним системам. Ідея в тому, щоб автоматизація була настільки повною, наскільки це взагалі можливо: всюди, де машина може ефективно працювати без втручання людей, рано чи пізно має відбутися заміщення людської праці. Співробітникам при цьому відводиться роль контролерів, які можуть підключитися в екстрених і нестандартних ситуаціях.

В результаті переходу промисловості на ці принципи відбувається також зміна в бізнес-моделях, при цьому зберігається принцип економії: роботизоване виробництво більш енергоефективно, супроводжується меншою кількістю відходів і браку.

9.1. Розподілені системи автоматизації

З ростом кількості сенсорів, збільшенням площі території, на якій розташована автоматизована система і ускладненням алгоритмів управління стає більш ефективним застосування *розподілених* систем. Розподілені системи складаються з безлічі територіально рознесених контролерів і модулів вводу-виводу. При такому підході структура розподіленої системи і структура алгоритму її роботи стають подібні до структури самого об'єкта автоматизації, а функції збору, обробки даних, управління та обчислення виявляються розподіленими серед безлічі контролерів. Кожен контролер працює зі своєю групою пристроїв введення-виведення і обслуговує певну частину об'єкта управління. Зокрема, технологічне обладнання, як правило, випускається з уже вбудованими ПЛК. Тенденція децентралізації управління та наближення контролерів до об'єктів управління є загальною для всіх систем автоматизації і частково навіяна успіхами об'єктно-орієнтованого програмування. Крім того, розподілену систему управління (PCY, DCS – Distributed Control System) можна визначити як систему, що складається з безлічі пристроїв, рознесених в просторі, кожне з яких не залежить від інших, але взаємодіє з ними для виконання спільного завдання. У граничному випадку елементи системи можуть перебувати на різних континентах земної кулі, а зв'язок між ними може виконуватися через Інтернет. Як "безлічі пристроїв" можуть виступати будь-які мікропроцесорні пристрої, наприклад, ПЛК або рознесені в просторі модулі введення-виведення одного контролера. Однак в останньому випадку тільки збір даних можна розглядати як розподілений, в той час як функція управління є зосередженою в одному контролері (рис.9.1).

Максимальні переваги розподіленої системи досягаються, коли контролери працюють автономно, а обмін інформацією між ними зведений до мінімуму.

Розподілена система має наступні характеристики, що відрізняють її від зосередженої:

- більшу швидкодію завдяки розподілу завдань між паралельно працюючими процесорами;
- підвищену надійність (відмова одного з контролерів не впливає на працездатність інших);
- велику стійкість до збоїв;
- більш просте нарощування або застосування реконфігурованих систем;
- спрощену процедуру модернізації;
- велику простоту проектування, налаштування, діагностики та обслуговування завдяки відповідності архітектури системи архітектурі об'єкта управління, а також відносній простоті кожного з модулів системи;
- поліпшену стійкість і точність завдяки зменшенню довжини ліній передачі аналогових сигналів від сенсорів до пристроїв введення;
- менший обсяг кабельної продукції, знижені вимоги до кабелю і більш низька його вартість;

- менші витрати на монтаж і обслуговування кабельного обладнання.

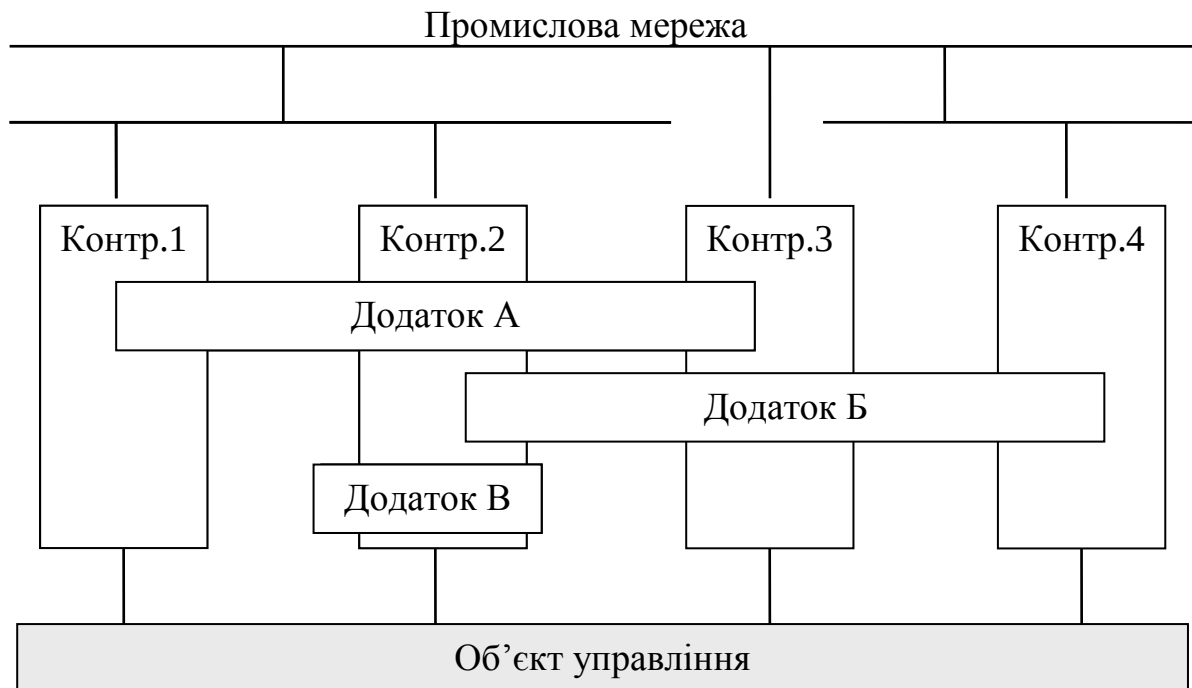


Рисунок 9.1 – Модель розподіленої системи автоматизації відповідно до стандарту МЕК 61499

Розподілена система пом'якшує також вимоги до операційних систем (ОС) реального часу, оскільки завдання розподілені між паралельно працюючими контролерами, на кожному з яких встановлена окрема ОС.

Архітектура системи із загальною шиною

Архітектура розподіленої системи промислової автоматизації на основі загальної шини показана на рис. 9.2. Для того, щоб отримати дані з модуля або контролера, комп'ютер (або контролер) посилає в шину його адресу і команду запиту даних. Мікропроцесор, що входить до складу кожного модуля або контролера, звіряє адресу на шині з його власною адресою, записаним в ПЗП, і, якщо адреси співпадають, виконує наступну за адресою команду. Команда дозволяє вважати дані, що надходять на вхід пристрою, або встановити необхідні дані на його виході.

Розподілена система із загальною шиною породжує дві нові проблеми в порівнянні з топологією "точка-точка" (коли з'єднуються тільки два пристрої): необхідність адресації пристроїв і необхідність очікування в черзі. Додавання адреси в комунікаційний пакет знижує швидкість обміну при коротких повідомленнях, а обмін з загальною шиною призводить до того, що кожен пристрій для передачі повідомлення повинно чекати, коли шина стане вільною. Це уповільнює швидкість обміну між пристроями в порівнянні з топологією "точка-точка". Затримка в мережах з великою кількістю пристроїв стає істотним обмеженням на застосування топології із загальною шиною в деяких додатках,

зокрема, в разі ПД-регулювання, коли затримка в мережі обмежує тактову частоту роботи контуру регулювання. Для таких випадків використовують локальні мережі або локальні технологічні контролери.

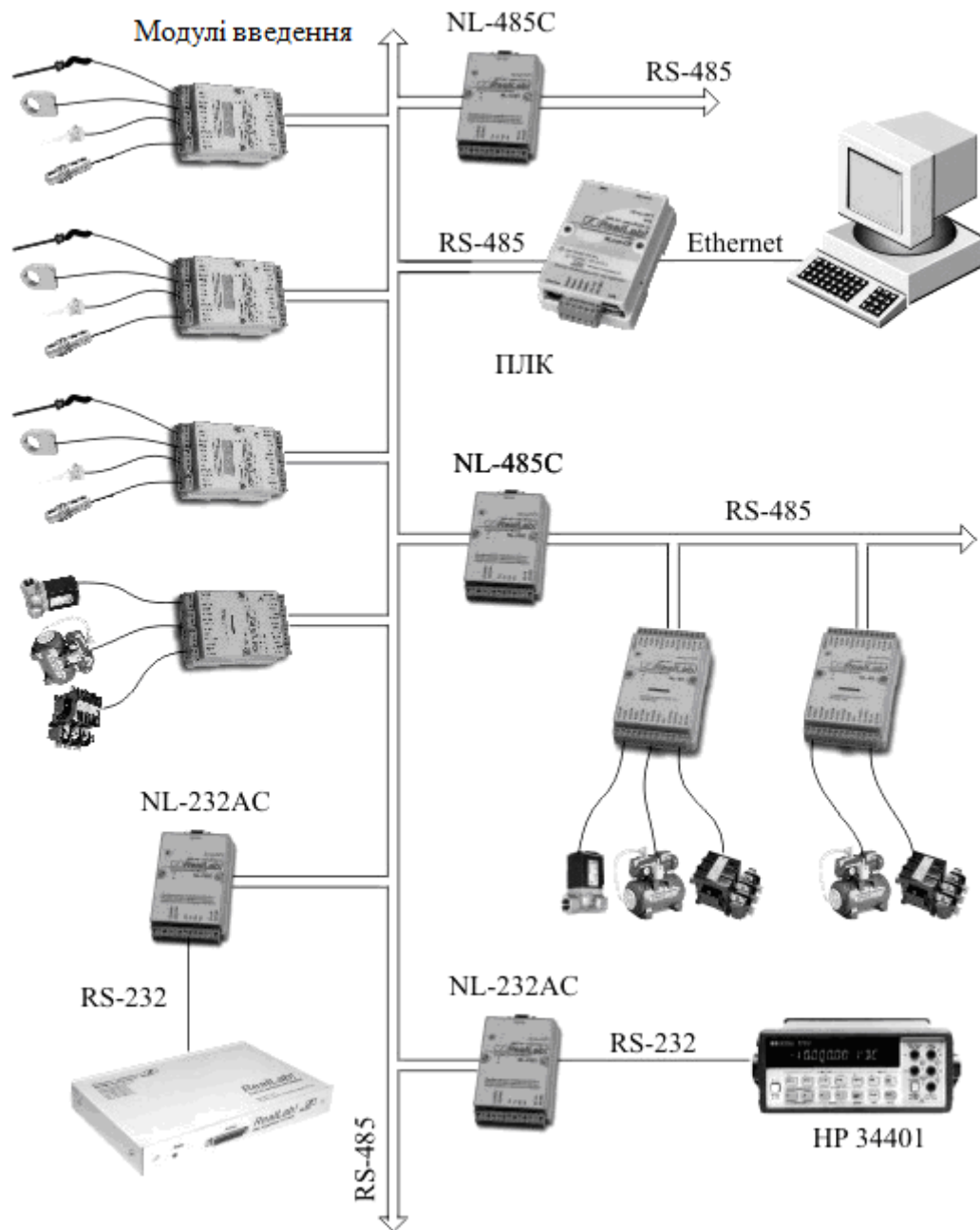


Рисунок 9.2 – Приклад архітектури розподіленої системи збору даних і управління на модулях вводу-виводу

Розподілені системи дозволяють вирішити також наступну проблему. З ростом кількості сенсорів в системі, збільшується число і сумарна довжина проводів, що з'єднують сенсори з пристроєм введення. Це призводить не тільки до зростання вартості кабельного обладнання, але і до проблем, пов'язаних з електромагнітними наведеннями, особливо якщо сенсори розподілені по ве-

ликій площі. У розподіленій системі модулі введення-виведення виготовляються з невеликою кількістю входів (зазвичай від 1 до 16), а самі модулі розташовуються поблизу місця установки сенсорів. Збільшення кількості сенсорів (входів) досягається шляхом нарощування числа модулів і об'єднання їх за допомогою загальної шини. Це скорочує загальну довжину проводів в системі, а також довжину проводів з аналоговими сигналами.

Зв'язок окремих пристроїв в розподіленій системі може здійснюватися за допомогою будь-якої промислової мережі. Найбільш поширені мережі Profibus, що пов'язано з популярністю виробів фірми Siemens, а також мережі Modbus з фізичною шиною RS-485 завдяки поширеності модулів і контролерів фірм ICP DAS, Advantech та інших. За останні роки стрімко зросла кількість використовуваних мереж Ethernet (точніше, Industrial Ethernet) в якості промислових мереж при швидкості передачі 10, 100 і 1000 Мбіт/с.

Деякі модулі введення-виведення, що входять до складу розподілених систем, дозволяють по команді з комп'ютера виконувати функції автоматичного регулювання. Для цього в них посилають значення уставки і параметри ПД-регулятора (пропорційний, диференційний та інтегральний коефіцієнти), потім команду запуску процесу регулювання. Наявність ПД-регулятора в модулях розподіленої системи дозволяє здійснити локальне регулювання (наприклад, підтримання стабільної температури в камері тепла і холоду), розвантаживши загальну шину для виконання інших завдань.

Розподілені системи будуються, як правило, з комерційно доступних компонентів (ПЛК, модулів введення-виведення, сенсорів, виконавчих пристроїв). Однак для однотипних тиражованих систем може бути вигідно будувати спеціалізовані системи, що складаються з повністю замовних (знову спроектованих) апаратних і програмних засобів. Кордон доцільності такого підходу визначається обсягом випуску виробів.

9.2 Багаторівнева мережива архітектура

Промислова мережа може бути підключена не тільки до одного комп'ютера, як показано на рис. 9.2 , а й до мережі комп'ютерів, наприклад, до локальної мережі Ethernet або глобальної мережі Internet . Така архітектура автоматизованої системи зручна при колективній роботі з системою автоматизації або для зв'язку технологічного рівня АСУ з управлінським. Наприклад, при використанні системи з багаторівневою архітектурою (рис. 9.3), температуру технологічного процесу формування тонких плівок металізації мікросхем, одночасно можуть спостерігати черговий оператор, головний інженер і директор підприємства.

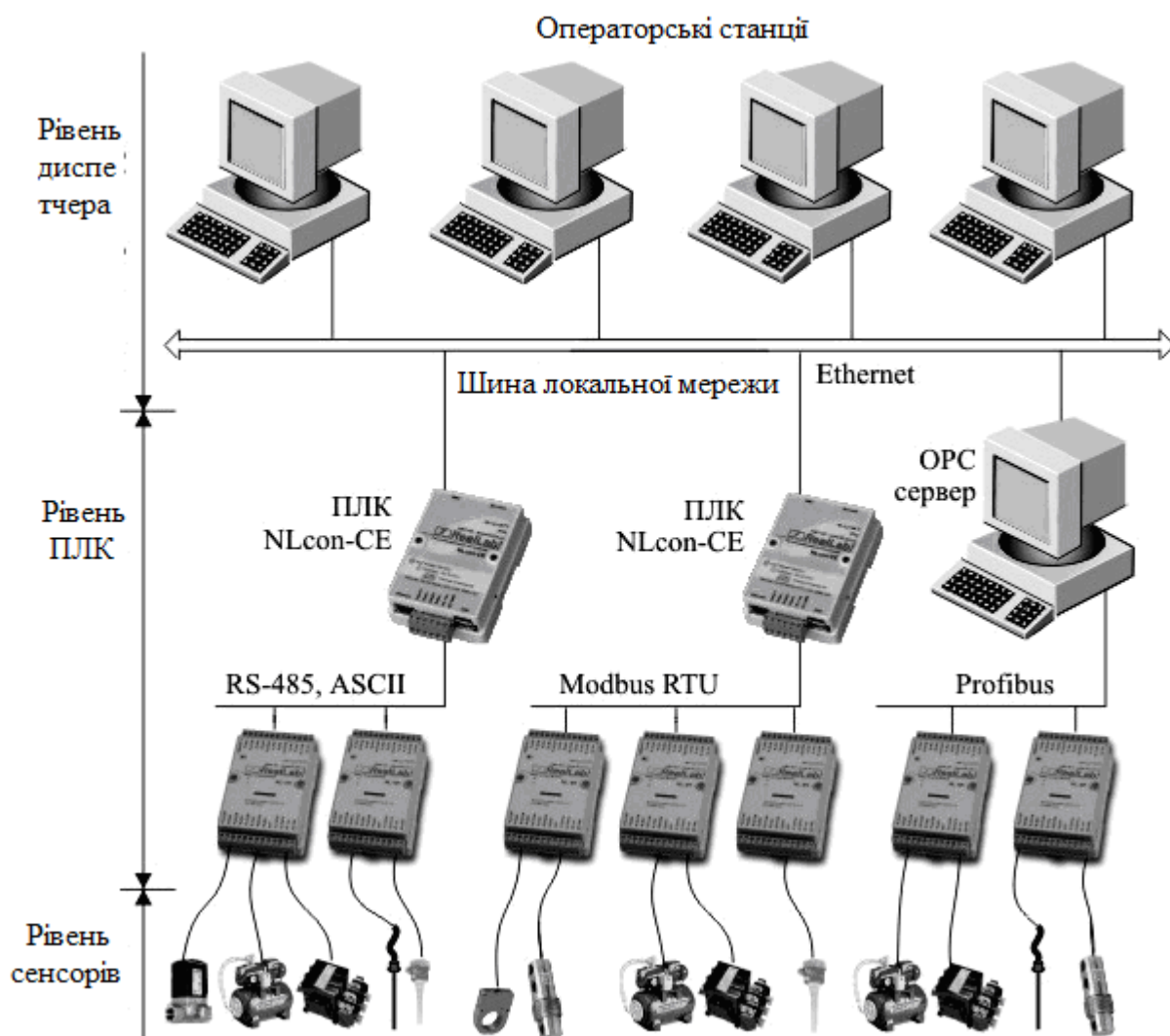


Рисунок 9.3 – Типова сучасна розподілена система автоматизації, що включає три рівні ієрархії

Доступ будь-якого комп'ютера мережі до пристроїв введення-виведення або контролерам здійснюється за допомогою OPC-сервера (*Open Platform Communications* – сімейство програмних технологій, що надають єдиний інтерфейс для управління об'єктами автоматизації і технологічними процесами). OPC-сервери можуть розташовуватися на декількох комп'ютерах або контролерах і доступ до будь-якого з них може здійснюватися з будь-якого комп'ютера мережі. Архітектура такої системи є досить загальною і широко використовується як для лабораторної автоматизації, так і для автоматизації технологічних процесів. Окремі промислові мережі можуть мати різні протоколи і містити обладнання різних виробників, а також різне фізичне середовище передачі даних – оптоволокну, мідні дроти, радіо ефір (через радіо, або GSM-модеми) та ін. Зазвичай OPC сервер працює тільки з одним або декількома портами вводу-виводу комп'ютера, до кожного з яких підключена одна промислова мережа, тому кількість OPC серверів в системі менше або дорівнює кількості промислових мереж. Завдяки наявності мережі Ethernet в систему легко можуть

бути включені сервери баз даних, комунікаційні сервери, веб-сервери, принтери, плоттери, факси, технологічне обладнання та інші пристрої з Ethernet - інтерфейсом.

Основою програмного забезпечення, встановленого на комп'ютерах мережі, є пакети SCADA – програмні засоби диспетчерського управління та збору даних. У контролерах виконуються завантажувальні модулі програм, які генеруються засобами візуального програмування ПЛК на мовах стандарту МЕК.

Системи управління з багаторівневою архітектурою зазвичай будуються за об'єктним принципом, коли структура системи вибирається подібній структурі об'єкта автоматизації, а кожна підсистема є локальною, тобто всі зворотні зв'язки замкнуті в межах цієї підсистеми. Кожна локальна підсистема виконує окрему функцію, що задається логікою функціонування всієї системи. Об'єктний принцип побудови дозволяє спростити проектування багаторівневої системи і забезпечити її структурну (архітектурну) надійність.

Аналіз складних систем управління дозволяє виділити в них кілька однорідних рівнів ієрархії (рис.9.4). Тут WAN – Wide Area Network – глобальна мережа, LAN – Local Area Network – локальна мережа.

Нижчий (нульовий) рівень включає в себе сенсори і виконавчі пристрої (актюатори): сенсори температури, тиску, кінцеві вимикачі, дискретні сенсори наявності напруги, вимірювальні трансформатори, реле-пускаті, контактори, електромагнітні клапани, електроприводи та ін. Сенсори і актюатори можуть мати інтерфейси типу AS-інтерфейс (ASI), 1-Wire або CAN, HART та ін. Морально застарілими інтерфейсами нульового рівня є аналогові інтерфейси 4...20 мА, 0...20 мА, 0...5 В. На даний час намітилася стійка тенденція до використання інтелектуальних сенсорів, які мають цифровий інтерфейс, вбудований мікроконтролер, пам'ять, засоби ідентифікації і виконують автоматичне калібрування і компенсацію характеристик датчика. Інтелектуальні сенсори в межах мережі повинні мати властивість взаємозамінності, зокрема, на рівні інтерфейсів.

Перший рівень складається з програмованих логічних контролерів і модулів аналого-цифрового і дискретного введення-виведення. Обмін інформацією між пристроями, що входять до складу автоматизованої системи (комп'ютерами, контролерами, датчиками, виконавчими пристроями), відбувається в загальному випадку через *промислову мережу* (Fieldbus, "польову шину"). типу Modbus RTU, Modbus TCP, Profibus та ін. Іноді модулі введення-виведення виділяють в окремий рівень ієрархії.

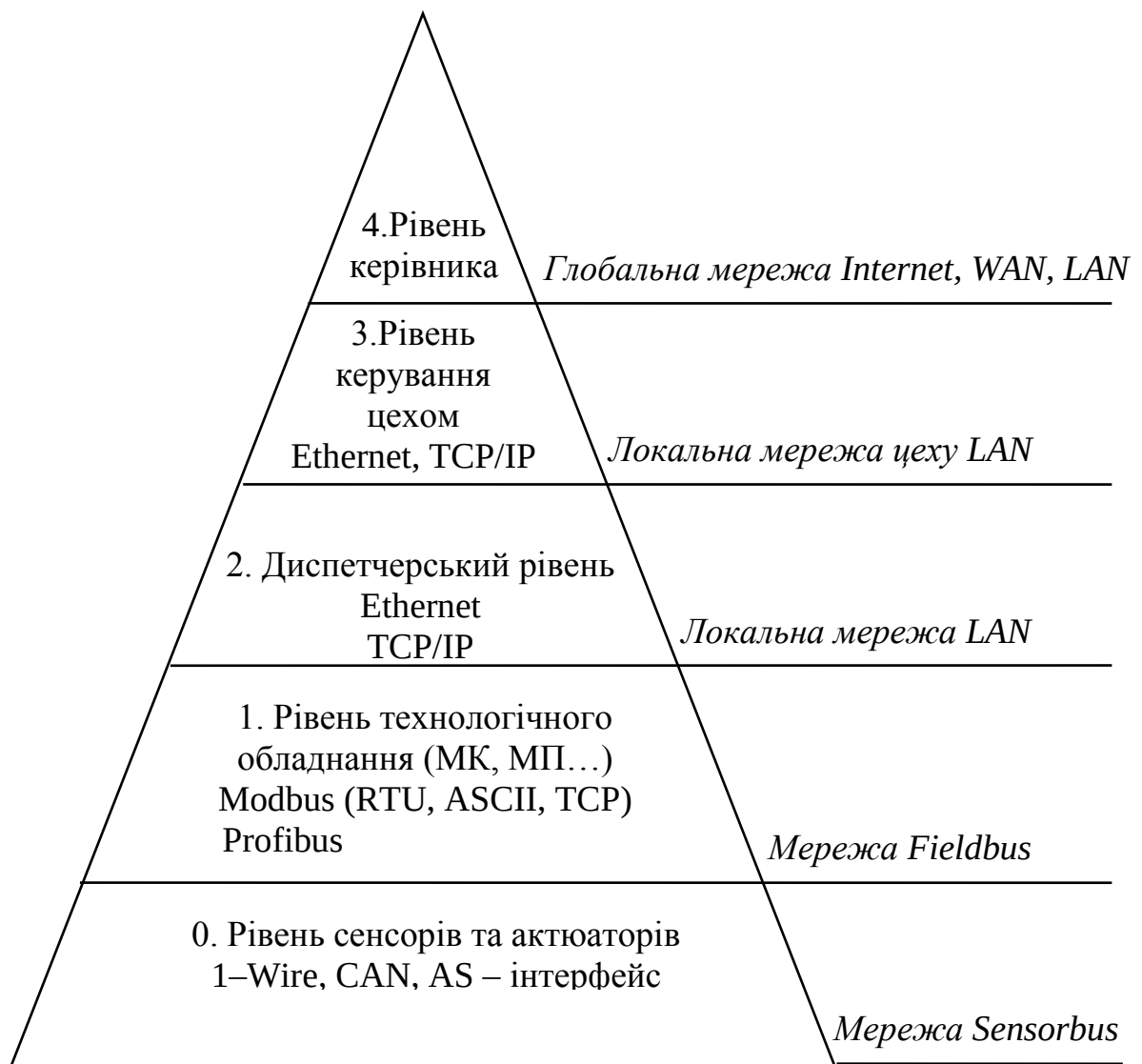


Рисунок 9.4 – Рівні ієрархії сучасної АСУ

Промислові мережі відрізняються від офісних наступними властивостями:

- спеціальним конструктивним виконанням, що забезпечує захист від пилу, вологи, вібрації, ударів;
- широким температурним діапазоном (зазвичай від -40 до +70 град);
- підвищену міцність кабелю, ізоляції, з'єднувачів, елементів кріплення;
- підвищеною стійкістю до дії електромагнітних завад;
- можливістю резервування для підвищення надійності;
- підвищеною надійністю передачі даних;
- можливістю самовідновлення після збою;
- детермінованістю (визначеністю) часу доставки повідомлень;
- можливістю роботи в реальному часі (з малої, постійної і відомою величиною затримки);

- роботою з довгими лініями зв'язку (від сотень метрів до кількох кілометрів).

Промислові мережі зазвичай не виходять за межі одного підприємства. Однак з появою Ethernet і Internet для промислових мереж стали застосовувати ту ж класифікацію, що і для офісних:

- LAN (Local Area Network) – мережі, розташовані на обмеженій території (в цеху, офісі, в межах заводу);
- MAN (Metropolitan Area Networks) – мережі міст ;
- WAN (Wide Area Network) – глобальна мережа, що охоплює кілька міст або континентів. Зазвичай для цього використовують Internet -технологію.

В даний час налічується більше 50 типів промислових мереж (Modbus, Profibus, DeviceNet, CANopen, LonWorks, ControlNet, SDS, Seriplex, ArcNet, BACnet, FDDI, FIP, FF, ASI, Ethernet, WorldFIP, Foundation Fieldbus, Interbus, BitBus та ін.). Однак широко поширеними є тільки частина з них. Переважна більшість АСУ ТП використовують мережі Modbus і Profibus. В останні роки зріс інтерес до мереж на основі CANopen і DeviceNet.

9.3. Загальні відомості про промислові мережі

Промисловою мережею називають комплекс обладнання і програмного забезпечення, які забезпечують обмін інформацією (комунікацію) між декількома пристроями. Промислова мережа є основою для побудови розподілених систем збору даних і управління.

Оскільки в промисловій автоматизації мереживі інтерфейси можуть бути невід'ємною частиною з'єднання пристроїв, а програмне забезпечення прикладного рівня моделі OSI (*Open systems interconnection basic reference model – Базова Еталонна Модель Взаємодії Відкритих Систем*) виповнюється на основному процесорі промислового контролера, то відокремити мереживу частину від пристроїв, що об'єднуються в мережу, іноді фізично неможливо. З іншого боку, зміну однієї мережі на іншу часто можна виконати за допомогою заміни електричного кабелю, ПО і адаптера, або введенням перетворювача інтерфейсу, тому часто один і той же тип ПЛК може використовуватися в мережах різних типів.

З'єднання промислової мережі з її компонентами (пристроями, вузлами мережі) виконується за допомогою інтерфейсів. Інтерфейсом мережі називають логічну і (або) фізичну межу між пристроєм і середовищем передачі інформації. Зазвичай цією межею є набір електронних компонентів і пов'язаного з ними програмного забезпечення. При істотних модифікаціях внутрішньої структури пристрою або програмного забезпечення інтерфейс залишається без змін, що є однією з ознак, що дозволяють виділити інтерфейс в складі обладнання.

Найбільш важливими параметрами інтерфейсу є пропускну здатність і максимальна довжина шини зв'язку. Промислові інтерфейси зазвичай забезпечують гальванічну розв'язку між сполученими пристроями. Найбільш поширені

в промисловій автоматизації послідовні інтерфейси RS-485, RS-232, RS-422, Ethernet, CAN, HART, AS-інтерфейс.

Для обміну інформацією взаємодіючі пристрої повинні мати однаковий *протокол обміну*. У простій формі протокол - це набір правил, які керують обміном інформацією. Він визначає синтаксис і семантику повідомлень, операції управління, синхронізацію і стану при комунікації. Протокол може бути реалізований апаратно, програмно або програмно-апаратно. Назва мережі зазвичай збігається з назвою протоколу, що пояснюється його визначальною роллю при створення мережі. В Україні використовуються мереживі протоколи, описані в серії стандартів.

Зазвичай мережа використовує кілька протоколів, що утворюють *стек протоколів* – набір пов'язаних комунікаційних протоколів, які функціонують спільно і використовують деякі або всі сім рівнів моделі OSI. Для більшості мереж стек протоколів реалізований за допомогою спеціалізованих мікросхем або вбудований в універсальний мікропроцесор.

Взаємодія пристроїв в промислових мережах виконується відповідно до моделі *клієнт-сервер*. У моделі клієнт-сервер взаємодіють два об'єкти. Сервером є об'єкт, який надає сервіс, тобто він виконує деякі дії за запитом клієнта. Мережа може містити кілька серверів і кілька клієнтів. Кожен клієнт може посылати запити більш ніж на один сервер, кожен сервер може відповідати на запити декількох клієнтів. Ця модель зручна для передачі даних, які з'являються періодично або в заздалегідь певний час, як, наприклад, значення температури в періодичному технологічному процесі. Однак ця модель незручна для передачі випадково виникаючих подій (наприклад, події, що складається з випадкового спрацьовування датчика рівня), оскільки для отримання цієї події клієнт повинен періодично, з високою частотою, запитувати стан датчика і аналізувати його, перевантажуючи мережу зайвим трафіком.

У будь-якої моделі взаємодії можна виділити головний пристрій, який управляє іншим (підлеглим) пристроєм. Пристрій, який відповідає на запити майстра, називають веденим або підлеглим (*Slave*). Ведений пристрій ніколи не починає комунікацію першим. Він чекає запиту від ведучого і тільки відповідає на запити.

У мережі може бути один або декілька ведучих пристроїв. Такі мережі називається, відповідно, *одномайстерними* або *багатомайстерними*. У *багатомайстерній* мережі виникає проблема вирішення конфліктів між пристроями, що намагаються одночасно отримати доступ до середовища передачі інформації. Конфлікти можуть бути вирішені шляхом передачі маркера, як, наприклад, в мережі Profibus, методом побітного порівняння ідентифікатора (використовується в CAN), методом прослуховування мережі (використовується в Ethernet) і методом запобігання колізій (використовується в бездротових мережах).

У всіх мережах застосовується "*широкомовлення*" без певного адреси, тобто всім учасникам мережі. Такий режим використовується зазвичай для

синхронізації процесів в мережі, наприклад, для одночасного запуску процесу введення даних усіма пристроями або для синхронізації годинника.

Деякі мережі використовують *багатоабонентський режим*, коли одне й те саме повідомлення надсилається кільком пристроям.

Передача інформації в мережі виконується через *канал* між передавальним і приймальним пристроєм. Канал включає в себе лінію зв'язку і приймально-передавальні пристрої.

У розподілених системах на основі промислових мереж може бути п'ять типів даних: *сигнали, команди, стан, події, запити*.

Сигнали – це результати вимірювань, отримані від сенсорів і вимірювальних перетворювачів. Їх "час життя" дуже короткий, тому часто потрібно отримати тільки останні дані і в максимально короткий термін.

Команди – це повідомлення, які викликають деякі дії, наприклад, закриття клапана або включення ПД-регулятора. Більшість систем повинні обробляти потоки команд, які передаються адресату з високою надійністю і їх не можна передати повторно.

Стан показує поточний або майбутній стан системи, в яке вона повинна перейти. Вимога до часу його доставки може бути не такі жорстким, як для команд; Неприйняття стан може бути послано повторно.

Подія настає зазвичай при досягненні поточним параметром граничного значення. Наприклад, подією може бути вихід температури за технологічно допустиму межу. За появою події повинні слідувати відповідні дії, тому для подій особливо важливо вимога гарантованого часу доставки.

Запит – це команда, що посилається для того, щоб отримати відповідь. Прикладом може бути запит серверу, який видає на нього відповідь.

Нижче при описі мереж буде використовуватися поняття *фрейму*. Під фреймом розуміють набір даних, що передаються по мережі і мають строго обумовлену структуру (формат).

Мережі можуть мати топологію *зірки, кільця, шини* або *змішану*. "Зірка" в промислової автоматизації використовується рідко. Шинна топологія є загальноприйнятною, що є однією з причин застосування терміну "промислова шина" замість "промислова мережа". До загальної шини в різних місцях може бути підключено будь-яку кількість пристроїв.

Основними параметрами промислових мереж є *продуктивність* і *надійність*. Продуктивність мережі характеризується *часом реакції* і *пропускною спроможністю*.

Час реакції мережі визначається як інтервал часу між запитом ведучого пристрою і відповіддю веденого за умови, що ведене пристрій має дуже малу затримку вироблення відповіді на запит.

Пропускна здатність мережі визначає кількість інформації, яку переносить мережею в одиницю часу. Вимірюється в *біт/с* і залежить від швидкодії приймачів і середовища передачі.

Важливою характеристикою промислових мереж є надійність доставки даних. Надійність характеризується *коефіцієнтом готовності, ймовірністю*

доставки даних, передбачуваністю часу доставки, безпекою, стійкістю до відмов.

Коефіцієнт готовності дорівнює відношенню часу напрацювання до відмови від суми часу напрацювання до відмови і часу відновлення після відмови.

Ймовірність доставки даних визначається завадостійкістю каналу передачі і детермінованістю доступу до каналу. У бездротових мережах ймовірність втрати пакетів при передачі набагато вище, ніж в дротяних. У мережах з випадковим методом доступу до каналу існує ймовірність того, що дані ніколи не будуть доставлені абоненту.

Час доставки даних в офісних мережах Ethernet є випадковою величиною, проте в промисловому Ethernet ця проблема вирішена застосуванням комутаторів.

Безпека – це здатність мережі захистити передані дані від несанкціонованого доступу.

Відмовостійкість – це здатність мережі продовжувати функціонування при відмові деяких елементів. При цьому характеристики системи можуть погіршитися, але вона не втрачає працездатності.

Останнім часом з'явився термін "якість обслуговування" (QoS – "Quality of Service"). QoS визначає ймовірність того, що мережа буде передавати заданий потік даних між двома вузлами відповідно до потреб додатка.

9.4. Модель взаємодії відкритих систем OSI

Оскільки основною функцією мережі є з'єднання між собою різного устаткування, проблема відкритості, зокрема, стандартизації, для мереж набуває особливого значення. У зв'язку з цим на початку 80-х років міжнародна організація по стандартизації ISO (International Standardization Organization) спільно з рядом інших організацій була сформульована і прийнята модель взаємодії відкритих систем OS, яка зіграла і грає до сих пір важливу роль в розвитку мереж.

Повний опис моделі OSI займає більше 1000 сторінок тексту. Це пов'язано з тим, що взаємодія пристроїв є складним завданням. Для вирішення таких завдань зазвичай використовується декомпозиція складної задачі на більш прості. Декомпозиція виконується таким чином, щоб кількість і складність зв'язків, а також потік даних між підзадачами були мінімальними. У моделі OSI було використано 7 підзадач (рівнів), причому декомпозиція виконана таким чином, що взаємодія здійснюється тільки між сусідніми рівнями.

Такий підхід забезпечив можливість вирішення завдання взаємодії систем для кожного рівня окремо, в тому числі незалежними групами розробників. Зокрема, для взаємодії пристроїв в мережі необхідно узгодити між собою електричні рівні сигналів, затримки і тривалості імпульсів, типи з'єднувачів, способи кодування інформації, способи забезпечення достовірності передачі, форми і формати адресації, формати даних, користувач може отримати доступ до ме-

режі, способів буферизації даних, способів поділу їх на пакети і відновлення цілісності повідомлень та ін. Сім рівнів моделі OSI представлені в табл. 9.1.

Таблиця 9.1 – Еталонна модель OSI

Рівень	Назва рівня	Назва протоколу	Приклади	Одиниця обміну
7	прикладний	прикладний протокол	FTP, HTTP, SMTP	APDU, повідомлення
6	рівень представлення	протокол рівня представлення	SSL	PPDU
5	сеансовий	сеансовий протокол		SPDU
4	транспортний	транспортний протокол	TCP, UDP, SPX	TPDU
3	мережевий	мережевий	IP, IPX	пакет
2	канальний (передачі даних)	протокол канального рівня		кадр
1	фізичний	протокол фізичного рівня		біт

Модель OSI не включає засоби взаємодії між собою додатків, розташованих на різних комп'ютерах мережі. Тому прикладний рівень не можна плутати з рівнем взаємодії додатків, який в модель OSI не входить.

Якщо додаток звертається із запитом до прикладного рівня, то на підставі цього запиту програмне забезпечення прикладного рівня формує повідомлення, що складається з заголовка і поля даних, і передає його вниз, на рівень представлення. Протокол представницького рівня виконує необхідні дії, що містяться в заголовку прикладного рівня і додає до повідомлення свою службову інформацію – заголовок представницького рівня, в якому містяться інструкції для відповідного рівня одержувача повідомлення. Сформований таким чином повідомлення з уже двома заголовками передається вниз сеансовому рівню, який також додає до нього свій заголовок. Таким чином, дійшовши до фізичного рівня, повідомлення обростає сім'ю заголовками, після чого воно передається по мережі адресату. Коли повідомлення досягне адресата, воно проходить весь стек протоколів в зворотному порядку, від фізичного рівня до прикладного. На кожному рівні виконуються відповідні функції, що містяться в заголовку кожного рівня.

На фізичному рівні (Physical layer) відбувається передача даних по таких лініях зв'язку, як кручена пара, коаксіальний кабель, оптоволоконний кабель або радіоканал. Основна функція фізичного рівня – забезпечити прийом логічної одиниці (а не нуля) на стороні одержувача в тому випадку, якщо на стороні відправника передана одиниця. На фізичному рівні використовуються

поняття: тривалість імпульсу і паузи, тривалості фронтів імпульсів, кількість проводів і цоколювка з'єднувачів, завадозахищеність, хвильовий опір, смуга пропускання, вид кодування інформації, синхронізація, модуляція, визначення початку кадру, рівень логічної одиниці і логічного нуля.

Канальний рівень. На фізичному рівні не враховується, що канал передачі може бути зайнятий іншими пристроями, підключеними до мережі. Перевірка доступності каналу передачі здійснюється на канальному рівні (Data Link Layer). Канальний рівень розбиває передані дані на кадри і виконує функції виявлення і корекції помилок за допомогою контрольної суми, що поміщається в початок або кінець кожного кадру. Кадр вважається отриманим, якщо контрольна сума, обчислена на стороні одержувача, збігається з контрольною сумою, приписаної до переданого кадру. При виявленні помилки канальний рівень може виконати повторну передачу пошкодженого кадру. Функція виправлення помилок не є обов'язковою вимогою стандарту.

Основне призначення рівня мережі (Network Layer) – визначення маршрутів пересилки пакетів від джерела до приймача повідомлень. Маршрути можуть бути жорстко задані або змінюватися динамічно в залежності від поточної завантаженості мережі. Мережевий рівень відповідає також за об'єднання кількох різних мереж в єдину транспортну систему, причому ці мережі можуть використовувати різні принципи передачі повідомлень, методи адресації і володіти довільною структурою зв'язків.

Транспортний рівень (Transport Layer) забезпечує доставку повідомлень з необхідним рівнем якості (надійності). У моделі OSI виділяють на вибір користувача п'ять класів сервісу, які відрізняються терміновістю доставки повідомлень, можливістю відновлення перерваного зв'язку, наявністю засобів мультиплексування декількох з'єднань між різними прикладними протоколами через загальний транспортний протокол, здатністю до виявлення і виправлення помилок передачі (таких як спотворення, втрата або дублювання пакетів).

На транспортному рівні виконується розбивка повідомлень сеансового рівня на пакети, їх нумерація, буферизація прийнятих пакетів, упорядкування, управління потоками.

Сеансовий рівень (Session Layer) управляє діалогом (сеансом зв'язку): відстежує черговість передачі повідомлень учасниками мережі; вставляє мітки в довгі повідомлення, щоб в разі втрати зв'язку виконає повторну передачу тільки загубленої частини повідомлення; встановлює спосіб обміну (двобічний або напівдуплексний).

Цей рівень рідко використовується на практиці, а його функції часто об'єднуються з функціями прикладного рівня.

Рівень представлення (Presentation Layer) оперує з формою подання переданої по мережі інформації, не змінюючи її змісту. На цьому рівні може здаватися тип кодування символів(наприклад, ASCII або ANSI), може змінюватися синтаксис переданих повідомлень, задаватися тип шифрування і дешифрування даних для забезпечення їх секретності. Прикладом такого протоколу є Secure Socket Layer (SSL), що забезпечує секретність повідомлень для протоколу TCP IP .

Прикладний рівень (Application Layer) являє собою набір популярних протоколів, за допомогою яких здійснюється доступ до файлів (наприклад, за допомогою протоколу FTP) або веб-сторінок (з допомогою HTTP), до електронної пошти (SMTP).

Модель OSI буде використана нижче при розгляді ряду промислових мереж.

9.5. Інтерфейси RS-485, RS-422 і RS-232

Інтерфейси RS-485 і RS-422 (Recommended Standard) описані в стандартах ANSI EIA/TIA. Інтерфейс RS-485 є найбільш поширеним в промисловій автоматизації. Його використовують промислові мережі Modbus, Profibus DP, ARCNET, BitBus, WorldFip, LAN, Interbus і безліч нестандартних мереж. Пов'язано це з тим, що за всіма основними показниками даний інтерфейс є найкращим з усіх можливих при сучасному рівні розвитку технології (табл.9.2). Основними його перевагами є:

- двосторонній обмін даними за все по одній парі проводів;
- робота з декількома трансиверами, підключеними до однієї і тієї ж лінії, тобто можливість організації мережі;
- велика довжина лінії зв'язку;
- досить висока швидкість передачі.

Таблиця 9.2 – Стандартні параметри інтерфейсів RS-422 і RS-485

Стандартні параметри інтерфейсів	RS-422	RS-485
Допустима кількість передавачів / приймачів	1/10	32/32
Максимальна довжина кабелю	1200 м	1200 м
Максимальна швидкість зв'язку	10 Мбіт/с	10 Мбіт/с
Діапазон напруги "1" передавача	+2...+10 В	+1.5...+6 В
Діапазон напруги "0" передавача	-2...-10 В	-1.5...-6 В
Діапазон синфазної напруги передавача	-3...+3 В	-1...+3 В
Допустимий діапазон напруги приймача	-7...+7 В	-7...+12 В
Граничний діапазон чутливості приймача	± 200 мВ	± 200 мВ
Струм короткого замикання драйвера	150 мА	250 мА
Допустимий опір навантаження передавача	100 Ом	54 Ом

Вхідний опір приймача	4 кОм	12 кОм
Максимальний час наростання сигналу передавача	10% біта	30% біта

Принципи побудови.

Диференційна передача сигналу. В основі побудови інтерфейсу RS -485 лежить диференційний спосіб передачі сигналу, коли напруга, що відповідає рівню логічної одиниці або нуля, відраховується не від «землі», а вимірюється як різниця потенціалів між двома передавальними лініями: U_{ex1} і U_{ex2} (рис. 9.5) . При цьому напруга кожної лінії щодо "землі" не повинна виходити за діапазон $-7... +12$ В.

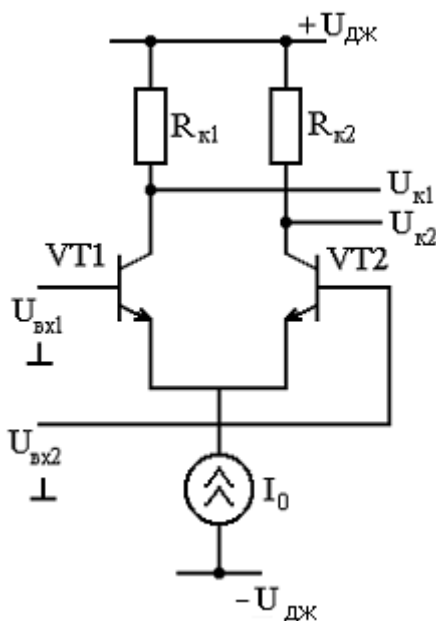


Рисунок 9.5 – Диференціальний підсилювальний каскад

Приймачі сигналу є диференціальними, тобто сприймають тільки різниця між напругою на лінії U_{ex1} і U_{ex2} . При різниці напруги понад 200 мВ, до +12 В вважається, що на лінії встановлено значення логічної одиниці, при напрузі менш –200 мВ, до –7 В - логічного нуля. Диференціальне напруга на виході передавача відповідно до стандарту повинно бути не менше 1,5 В, тому при порозі спрацьовування приймача 200 мВ перешкода (в тому числі падіння напруги на омичному опорі лінії) може мати розмах 1,3 В над рівнем 200 мВ. Такий великий запас необхідний для роботи на довгих лініях з великим провідникові. Фактично, саме цей запас по напрузі і визначає максимальну довжину лінії зв'язку (1200 м) при низьких швидкостях передачі (менше 100 Кбіт/с).

Завдяки симетрії ліній щодо "землі" в них наводяться перешкоди, близькі за формою і величиною. Оскільки у приймачі з диференціальним входом сигнал на виході залежить від різниці напруги на лініях, напруга перешкоди виявляється рівною нулю. У реальних умовах, коли існує невелика асиметрія ліній і

навантажень, перешкода пригнічується не повністю, але послаблюється істотно.

Для мінімізації чутливості лінії передачі до електромагнітної наводкою використовується кручена пара проводів. Токи, що наводяться в сусідніх витках внаслідок явища електромагнітної індукції, виявляються спрямованими назустріч один одному і взаємно компенсуються. Ступінь компенсації визначається якістю виготовлення кабелю і кількістю витків на одиницю довжини.

На рис. 9.6 показаний приклад поєднання двох перетворювачів інтерфейсів RS-485 з метою збільшення дальності зв'язку двох пристроїв.

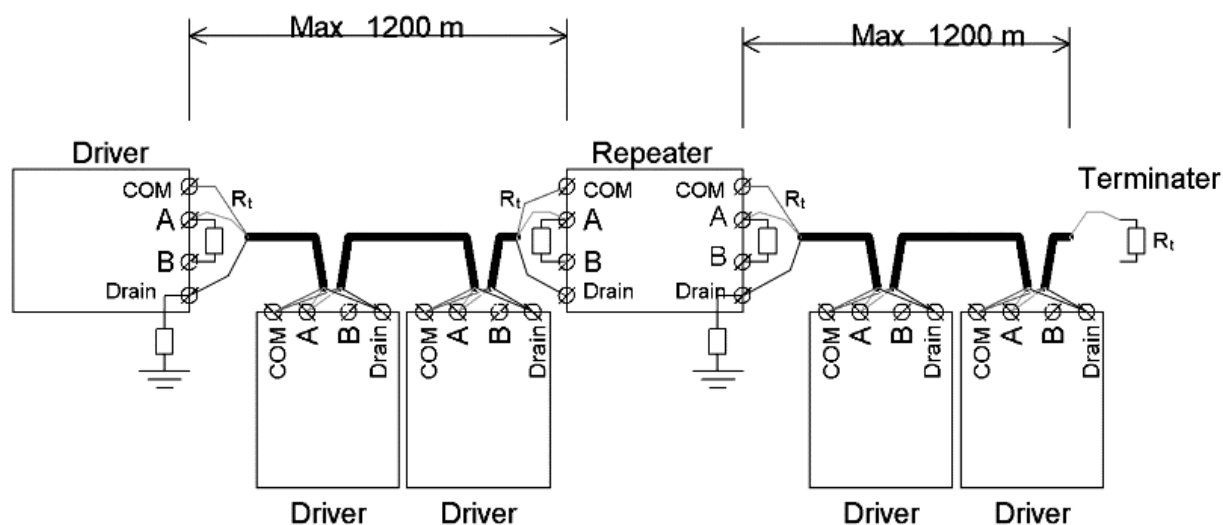


Рисунок 9.6 – Поєднання перетворювачів інтерфейсів RS-485 з метою збільшення дальності зв'язку двох пристроїв

Третій стан виходів

Другою особливістю передавача D (Driver) інтерфейсу RS -485 є можливість переведення вихідних каскадів в "третій" стан сигналом *Driver Enable*. Наявність третього стану дозволяє здійснити напівдуплексний обмін між будь-якими двома пристроями, підключеними до лінії, всього по двох проводах.

Перехід передавача інтерфейсу в третій стан здійснюється зазвичай сигналом RTS (Request To Send) COM-порту.

9.6 Інтерфейси RS-232 і RS-422

Інтерфейс RS-422 використовується набагато рідше, ніж RS-485 та, як правило, не для створення мережі, а для з'єднання двох пристроїв на великій відстані (до 1200 м), оскільки інтерфейс RS-232 працездатний тільки на відстані до 15 м. Кожен передавач RS-422 може бути навантажений на 10 приймачів. Інтерфейс працездатний при напрузі до ± 7 В.

Для сполучення пристрої, що мають відмінні інтерфейси застосовуються перетворювачі інтерфейсів (рис. 9.7)

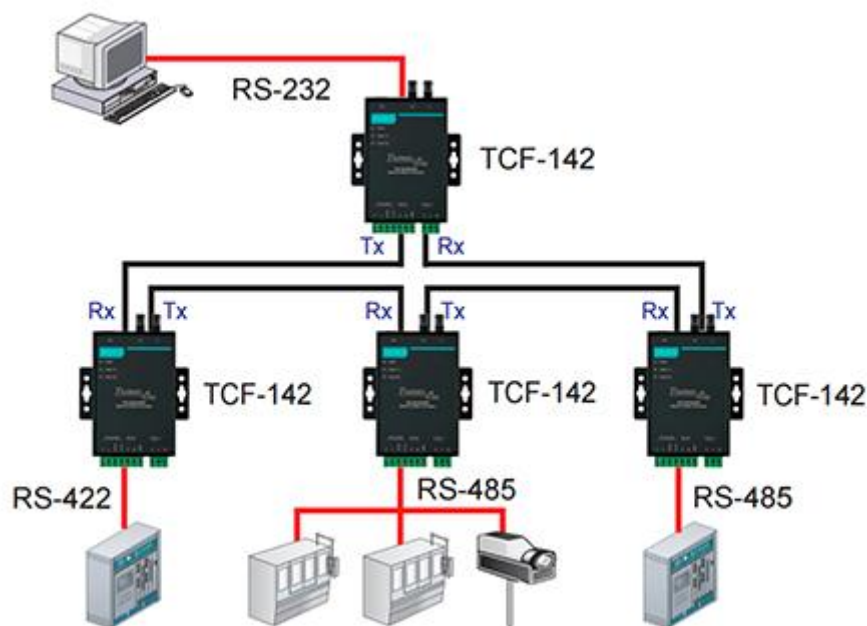


Рисунок 9.7 – Застосування перетворювачів інтерфейсів в мережах управління

9.7 Інтерфейс "струмова петля"

Інтерфейс струмова петля використовується для передачі інформації з 1950-х років. Спочатку в ньому використовувався струм 60 мА; пізніше, з 1962 року, набув поширення інтерфейс з струмом 20 мА, переважно в телетайпних апаратах. У 1980-х роках почала широко застосовуватися "струмова петля" 4...20 мА в різноманітному технологічному обладнанні, датчиках і виконавчих пристроях засобів автоматики. Популярність струмової петлі почала падати після появи стандарту на інтерфейс RS-485 (1983 р) і в даний час в новому обладнанні вона практично не застосовується.

У передавачі струмової петлі використовується не джерело напруги, як в інтерфейсі RS-485, а джерело струму. За визначенням, величина струму, що надходить з джерела, не залежить від параметрів навантаження. Таким чином, в колі протікає струм, що не залежить від опору кабелю, опору навантаження, індуктивної перешкоди, а також від напруги живлення джерела струму. Ця властивість струмової петлі є основним, що визначає всі варіанти її застосування.

Як лінія передачі зазвичай використовується екранована кручена пара, яка спільно з диференціальним приймачем дозволяє послабити індуктивну і синфазну перешкоду.

На приймальному кінці струм петлі перетворюється в напругу за допомогою каліброваного опору навантаження. При струмі 20 мА для отримання стандартного напруги 2,5 В, 5 В або 10 В використовують резистор опором 125 Ом, 250 Ом або 500 Ом відповідно.

Основним недоліком струмової петлі є її відносно низька швидкодія, що обмежується швидкістю заряду ємності кабелю від джерела струму. Наприклад, при типовій погонній ємності кабелю 75 пФ/м і довжині 1 км ємність кабелю складе 75 нФ. Для заряду такої ємності від джерела струму 20 мА до напруги 5 В необхідний час 19 мкс, що відповідає швидкості передачі близько 9 Кбіт/с.

Другим недоліком струмової петлі, що обмежує її практичне застосування, є відсутність стандарту на конструктивне виконання роз'ємів і електричні параметри, хоча фактично стали загальноприйнятими діапазони сигналів 0...20 мА і 4...20 мА; набагато рідше використовують 0 ... 60 мА. У перспективних розробках рекомендується використовувати тільки діапазон 4 ... 20 мА.

Інтерфейс струмова петля поширений в двох версіях: цифровий і аналоговий.

Аналогова "струмова петля"

Аналогова версія струмової петлі використовується, як правило, для передачі сигналів від різноманітних сенсорів до контролера або від контролера до виконавчих пристроїв. Застосування струмової петлі в даному випадку дає дві переваги. По-перше, приведення діапазону зміни вимірюваної величини до стандартного діапазону забезпечує взаємозамінність компонентів. По-друге, стає можливим передати сигнал на велику відстань з високою точністю (похибка струмової петлі може бути знижена до $\pm 0,05\%$). Крім того, стандарт струмова петля підтримується переважною більшістю виробників засобів промислової автоматизації.

У варіанті 4...20 мА в якості початку відліку прийнятий струм 4 мА. Це дозволяє проводити діагностику цілісності кабелю (кабель має розрив, якщо струм дорівнює нулю) на відміну від варіанту 0...20 мА, де величина 0 мА може означати не тільки нульову величину сигналу, але і обрив кабелю. Другою перевагою рівня відліку 4 мА є можливість подачі енергії датчику для його живлення.

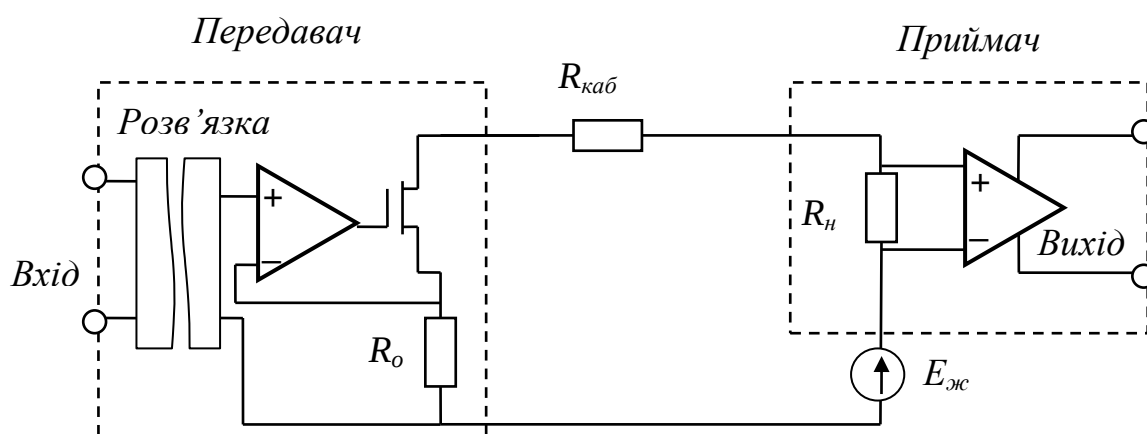


Рисунок 9.8 – Варіант побудови аналогової струмової петлі з зовнішнім джерелом живлення

На рис. 9.8 показано варіант побудови аналогової струмової петлі. Можливі варіанти з використанням вбудованого, або зовнішнього джерела живлення. Вбудоване джерело більш зручне при монтажі системи, а зовнішнє – тим, що його можна вибрати з будь-якими параметрами в залежності від поставленого завдання.

Перевагою схеми з операційним підсилювачем є можливість калібрування передавача без підключеного до нього кабелю і приймача, оскільки додана ними похибка дуже мала.

Напруга джерела живлення вибирається такою, щоб забезпечити роботу транзистора передавача в активному (ненасиченому) режимі і компенсувати падіння напруги на проводах кабелю і внутрішніх опорах передавача та приймача.

У схемі також використовується гальванічна розв'язка на вході передавача, що необхідно для виключення паразитних зв'язків між входом і виходом.

Цифрова струмова петля використовується зазвичай в версії 0...20 мА, оскільки вона реалізується набагато простіше, ніж 4...20 мА (рис. 9.9). Оскільки при цифровій передачі даних точність передачі логічних рівнів значної ролі не грає, можна використовувати джерело струму з невеликим внутрішнім опором і низькою точністю. Так при стандартному значенні напруги живлення 24 В і падінні напруги на вході приймача 0,8 В для отримання струму 20 мА опір R_o має дорівнювати приблизно 1,2 кОм. Опір кабелю перетином 0,35 кв. мм і довжиною 1 км дорівнює 97 Ом, що складе всього 10% від загального опору петлі і їм можна знехтувати. Падіння напруги на діоді оптрона становить 3,3% від напруги джерела живлення, і його впливом на струм в петлі також можна знехтувати. Тому з достатньою для практики точністю можна вважати, що передавач в цій схемі є джерелом струму.

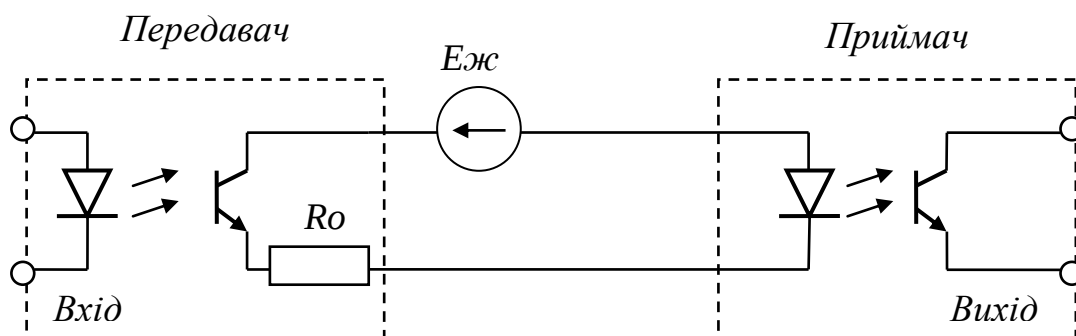


Рисунок 9.9 – Принцип реалізації цифрової струмової петлі

Як аналогова, так і цифрова струмова петля може використовуватися для передачі інформації декільком приймачів одночасно (рис. 9.10). Внаслідок низької швидкості передачі інформації через струмову петлю узгодження довгої лінії з передавачем і приймачем не потрібно.

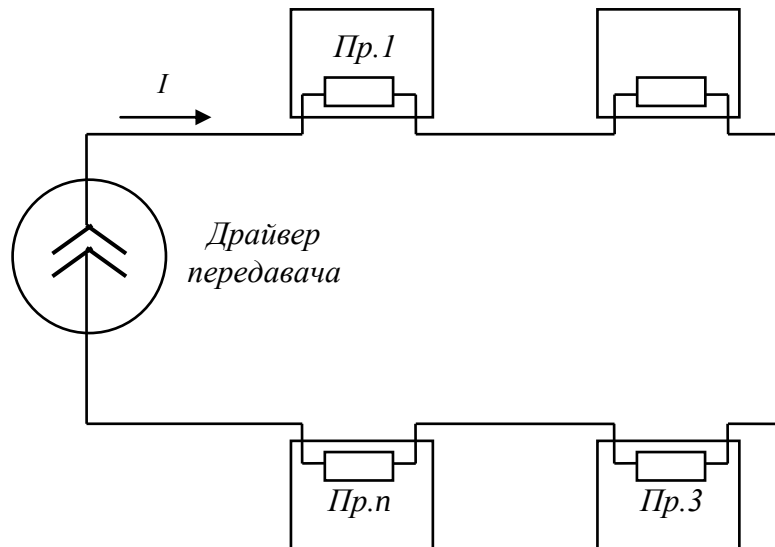


Рисунок 9.10 – Струмова петля може бути використана для передачі інформації декільком приймачам

Струмова петля знайшла своє "друге народження" у протоколі HART.

9.8 Магістральний протокол HART

HART-протокол (Highway Addressable Remote Transducer – магістральний адресуємий віддалений перетворювач) є відкритим стандартом на метод мереживного обміну, який включає в себе не тільки протокол взаємодії пристроїв, але і вимоги до апаратури каналу зв'язку. Стандарт HART був розроблений в 1980 році фірмою Rosemount, яка пізніше зробила його відкритим. В даний час стандарт підтримується міжнародною організацією HART Communication Foundation (HCF), що налічує 190 членів. HART застосовується для зв'язку контролера з сенсорами і вимірювальними перетворювачами, електромагнітними клапанами, локальними контролерами та іншим обладнанням.

Незважаючи на свою низьку швидкодію (1200 біт/с) і ненадійний аналоговий спосіб передачі даних, а також появу більш досконалих технологій, пристрої з HART-протоколом розробляються досі і обсяг цього сегмента ринку продовжує зростати. Застосовується HART-протокол на каналному і фізичному рівні.

Принципи побудови. При створенні HART-протоколу переслідувалася мета зробити його сумісним з широко поширеним в той час стандартом струмова петля та додати можливості, необхідні для управління інтелектуальними пристроями. Тому аналогова струмова петля 4...20 мА була модернізована таким чином, що отримала можливість напівдуплексного цифрового обміну даними. Для цього аналоговий сигнал підсумовується з цифровим сигналом і отримана таким чином сума передається за допомогою джерела струму 4...20 мА

по лінії зв'язку. Завдяки значній відмінності діапазонів частот аналогового (0...10 Гц) і цифрового (1200 Гц і 2200 Гц) сигналів вони легко можуть бути розділені фільтрами низьких і високих частот в приймачі. При передачі цифрового двійкового сигналу логічна одиниця кодується синусоїдальним сигналом з частотою 1200 Гц, нуль – 2200 Гц. При зміні частоти фаза коливань залишається безперервною. Такий спосіб формування сигналу називається частотною маніпуляцією з безперервною фазою. Вибір частот відповідає американським стандартам BELL 202 на телефонні канали зв'язку.

Принцип взаємодії пристроїв на фізичному рівні моделі OSI показаний на рис. 9.11. Опір навантаження вибирається так само, як і для струмової петлі (стандартом передбачена величина 230...1100 Ом) і служить для перетворення струму 4...20 мА в напругу. Акт взаємодії пристроїв ініціює контролер. Аналоговий сигнал зазвичай містить інформацію про вимірювану величину, а цифровий – інформацію про одиниці і діапазоні вимірювання, про вихід величини за межі динамічного діапазону, про тип датчика, ім'я виробника і т. і.). Аналоговий і цифровий сигнал підсумовуються і подаються в лінію зв'язку в формі струму. На стороні контролера струм перетвориться в напругу резистором навантаження. Отриманий сигнал подається на фільтр нижніх частот з частотою зрізу 10 Гц і на фільтр верхніх частот з частотою зрізу 400...800 Гц. На виході фільтрів виділяються цифровий і аналоговий сигнал. При використанні фільтрів другого порядку похибка, яку вносить цифровий сигнал в аналоговий, становить всього 0,01% від 20 мА.

Як і в звичайній струмовій петлі, джерело струму в HART може бути зовнішнім, або вбудованим.

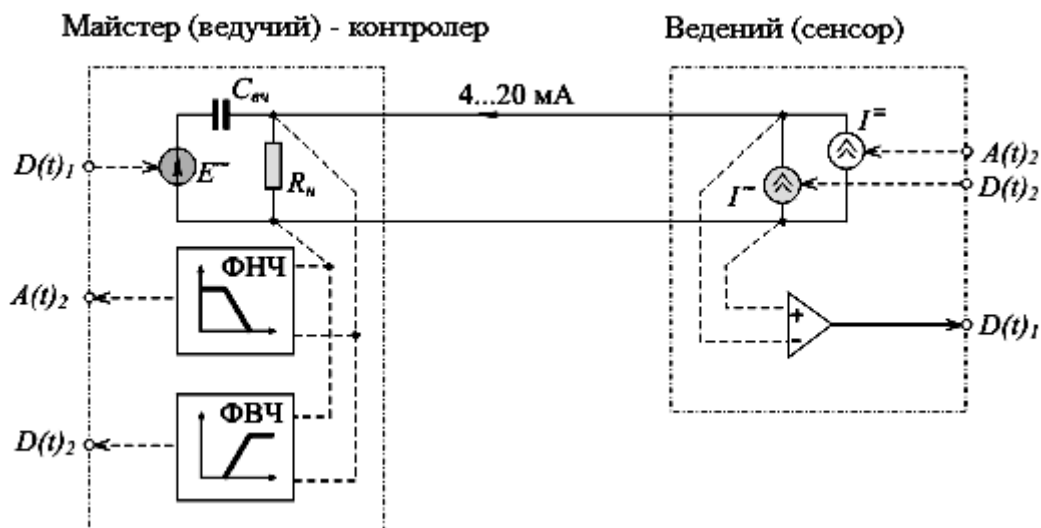


Рисунок 9.11 – Принцип роботи HART-протоколу на фізичному рівні

В окремому випадку HART - протокол може використовувати тільки цифровий сигнал, без аналогового, або тільки аналоговий сигнал 4...20 мА, без цифрового.

У разі, коли веденим пристроєм є не датчик, а виконавчий пристрій (наприклад електромагнітний клапан), аналоговий сигнал у формі струму повинен передаватися від ведучого пристрою до веденого і джерело струму повинно перебувати в ведучому пристрої. Оскільки HART пристрої містять мікроконтролер і МОН-ключі, необхідне для цього реконфігурування передавача і приймача виконується шляхом подачі відповідної команди.

HART – пристрої завжди містять мікроконтролер (рис. 9.12) з UART і ППЗП (перепрограмований постійний запам'ятовуючий пристрій). Цифровий сигнал, сформований мікроконтролером, перетворюється в UART в безперервну послідовність біт, що складається з двійкових слів довжиною 11 біт кожне. Кожне слово починається зі стартового біта (логічний нуль), за яким слідує байт переданих даних, потім біт паритету і стоповий біт. Сформована таким чином послідовність нулів і одиниць передається в модем, що виконує частотну маніпуляцію (ЧМ). Отриманий ЧМ сигнал передається в інтерфейсний блок для формування напруги, що подається в лінію зв'язку (нагадаємо, що від контролера до датчика передається сигнал у формі напруги, а назад – у формі струму).

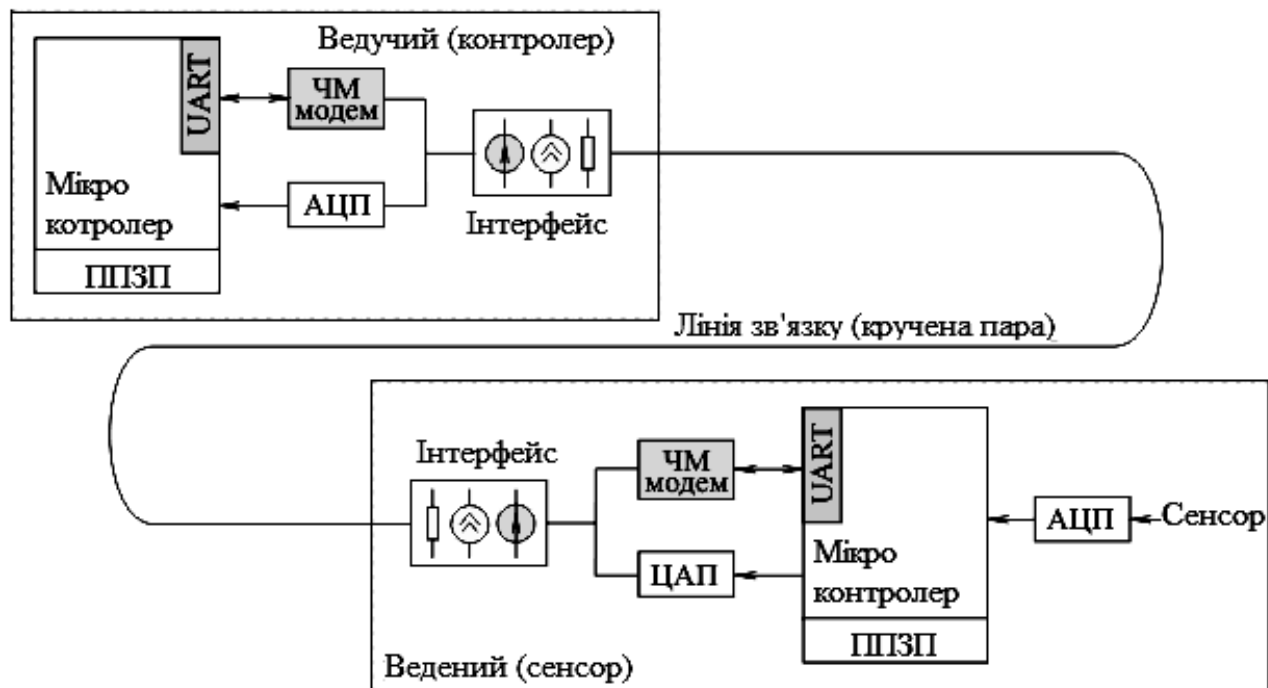


Рисунок 9.12 – Проходження аналогових і цифрових сигналів через пристрої з HART протоколом

На стороні датчика сигнал з лінії приймається інтерфейсним блоком, перетворюється ЧМ модемом в послідовність бітів, з якої контролер виділяє байти даних і біти паритету. Мікроконтролер перевіряє відповідність біта паритету переданому байту для кожного переданого слова, поки не виявить ознака кінця повідомлення.

9.9 CAN протокол

CAN (Controller Area Network – "область, охоплена мережею контролерів") являє собою комплекс стандартів для побудови розподілених промислових мереж, який використовує послідовну передачу даних в реальному часі з дуже високим ступенем надійності і захищеності. Центральне місце в CAN займає протокол канального рівня моделі OSI. Спочатку CAN був розроблений для автомобільної промисловості, але в даний час швидко впроваджується в область промислової автоматизації. Це добре продуманий, сучасний і багатобічний мережевий протокол. Початок розвитку CAN було покладено компанією Bosch в 1983 р, перші мікросхеми CAN контроллеров були випущені фірмами Intel і Philips в 1987 році, в даний час контролери та трансивери CAN випускаються багатьма фірмами, в тому числі Analog Devices, Atmel, Cast.

В Україні інтерес до CAN за останні роки сильно зріс. Серед протоколів прикладного рівня для роботи з CAN найбільше поширення отримали CANopen і DeviceNet.

В даний час CAN підтримується 11-ю стандартами ISO. Для його втілення в життя різні фірми розробили кілька протоколів: CANopen (організації CiA), SDS (Honeywell Micro Switch Division), CAN Kingdom (Kvaser), DeviceNet (Allen-Bradley, став Європейським стандартом в 2002 р) і ряд інших.

CAN характеризується наступними основними властивостями:

- кожному з повідомлень (а не пристрою) встановлюється свій пріоритет;
- гарантована величина паузи між двома актами обміну;
- гнучкість конфігурації і можливість модернізації системи;
- широкомовний прийом повідомлень з синхронізацією часу;
- несуперечливість даних на рівні всієї системи;
- допустимість кількох провідних пристроїв в мережі (багатомайстерня мережа);
- здатність до виявлення помилок і сигналізації про їх наявності;
- автоматичний повтор передачі повідомлень, доставлених з помилкою, відразу, як тільки мережа стане вільною;
- автоматичне розпізнавання збоїв і відмов з можливістю автоматичного відключення модулів.

До недоліків можна віднести порівняно високу вартість CAN-пристроїв, відсутність єдиного протоколу прикладного рівня, а також надмірну складність і заплутаність протоколів канального і прикладного рівня, викладених в стандартах організації CAN in Automation.

Фізичний рівень моделі OSI забезпечує надійну передачу бітів, ігноруючи зміст інформації, що передається. Основними поняттями фізичного рівня є лінії передачі (в більшості випадків це кручена пара, хоча допускається використовувати плоский кабель або один провід і "корпусні землі", оптоволокну, радіоканал), часові діаграми, система синхронізації, формат даних, забезпечен-

ня достовірності передачі (контрольна сума, методи кодування, виявлення і відновлення помилок). Характеристики передавача і приймача стандартом не встановлюються, оскільки вони можуть бути обрані для кожного конкретного випадку виходячи з вимог застосування.

9.10 Мережа Profibus

Слово PROFIBUS отримано від зобов'язань щодо зменшення PROcess FieLd BUS, що приблизно перекладається як "промислова шина для технологічних процесів". Стандарт Profibus був спочатку прийнятий в Німеччині в 1987 році, потім, в 1996 році, він став міжнародним (EN 50170 і EN 50254).

Мережа Profibus (як і інші описані тут промислові мережі, крім Industrial Ethernet) використовує тільки перший і другий рівні моделі OSI. Один з варіантів мережі, Profibus FMS, використовує також рівень 7 (табл. 9.3).

Таблиця 9.3 – Мережа Profibus відповідно до моделі OSI

№	Назва рівня	Profibus DP	Profibus FMS	Profibus PA
7	прикладний	немає	Fieldbus Message Specification (FMS)	немає
6	представлення	немає		
5	сеансовий			
4	транспортний			
3	мережа			
2	канальний (передачі даних)	FDL	FDL	IEC 1158-2
1	фізичний	RS-485, оптоволоконний інтерфейс	RS-485, оптоволоконний інтерфейс	Інтерфейс IEC 1158-2

Profibus має три модифікації: Profibus DP, Profibus FMS і Profibus PA. Profibus DP (Profibus for Decentralized Peripherals – "Profibus для децентралізованої периферії") використовує рівні 1 і 2 моделі OSI, а також призначений для користувача інтерфейс, який в модель OSI не входить. Безпосередній доступ з призначеного для користувача додатки до канального рівня здійснюється за допомогою DDLM (Direct Data Link Mapper – "прямий перетворювач для канального рівня"). Інтерфейс забезпечує функції, необхідні для зв'язку з пристроями введення-виведення і контролерами. Profibus DP на відміну від FMS і PA побудований таким чином, щоб забезпечити найбільш швидкий обмін даними з пристроями, підключеними до мережі.

Profibus FMS (Profibus з FMS протоколом) використовує рівень 7 моделі OSI і застосовується для обміну даними з контролерами і комп'ютерами на реєстровому рівні. Profibus FMS надає велику гнучкість при передачі великих обсягів даних, але програє протоколу DP в популярності внаслідок своєї складності.

Profibus FMS і DP використовують один і той же фізичний рівень, заснований на інтерфейсі RS-485 і можуть працювати в загальній мережі.

Profibus PA (Profibus for Process Automation – "для автоматизації технологічних процесів") використовує фізичний рівень на основі стандарту IEC 1158-2, який забезпечує живлення пристроїв через шину і не сумісний з RS-485. Особливістю Profibus PA є можливість роботи у *вибухонебезпечній зоні*.

В останні роки з'явився стандарт PROFINet, який заснований на Industrial Ethernet і технологіях COM, DCOM. Він легко забезпечує зв'язок промислової мережі Profibus з офісною мережею Ethernet.

Profibus є багатомайстерною мережею (з декількома ведучими пристроями). Як ведені пристрої виступають зазвичай пристрої введення-виведення, клапани, вимірювальні перетворювачі. Вони не можуть самостійно отримати доступ до шини і тільки відповідають на запити майстра.

На фізичному рівні Profibus DP і FMS використовують стандарт RS-485 при швидкості передачі до 12 Мбіт/с і з розмірами сегментів мережі до 32 пристроїв. Кількість пристроїв можна збільшити за допомогою повторювачів інтерфейсу.

Особливі вимоги встановлені до кабелю мережі. Він повинен мати хвильовий опір від 135 до 165 Ом при погонній ємності не більше 35 пФ/м, площа поперечного перерізу провідників більш 0,34 кв. мм. і погонний опір не більше 110 Ом/км. Кабель повинен мати одну або дві кручені пари з мідним екраном у вигляді сітки або фольги.

Стандартом для шини Profibus рекомендується роз'єм D-sub (DB-9) з 9-ю контактами. При необхідності мати ступінь захисту IP65/67 рекомендується використовувати циліндричний роз'єм типу M12 (IEC 947-5-2), HAN-BRID або гібридний роз'єм фірми Siemens.

По обидва боки лінії передачі підключаються погоджують резистори, які конструктивно встановлені в усі мережеві роз'єми і підключаються за допомогою мікроперемикачів. При швидкостях передачі більше 1,5 Мбіт/с для узгодження лінії додатково використовуються пласкі (друковані) котушки індуктивності.

Для збільшення дальності передачі в Profibus передбачена можливість роботи з оптоволоконним кабелем. При використанні скляного оптоволокна дальність зв'язку може бути збільшена до 15 км. Оптоволоконні інтерфейси виконуються у вигляді змінних модулів для контролерів.

Profibus PA використовує фізичний рівень, що відповідає стандарту IEC 1158-2. Дані передаються за допомогою рівнів струму +9 мА і -9 мА (струмова петля). Використовується манчестерський код (логічний нуль відповідає зміні негативного струму на позитивний, а логічна одиниця – позитивного на нега-

тивний). Швидкість передачі становить 31,25 Кбіт/с, в якості лінії передачі використовується кручена пара в екрані або без нього, один сегмент мережі може містити до 32 пристроїв. Максимальна довжина кабелю досягає 1,9 км. У кожному сегменті по обидва боки кабелю підключені RC-ланцюжка, що складаються з послідовно з'єднаних конденсатора ємністю 1 мкФ і резистора опором 100 Ом. Завдяки низькій енергії сигналу, що передається Profibus PA є іскробезпечним електричним колом і може бути використаний у вибухонебезпечних зонах.

Для передачі даних використовується NRZ-кодування і 11-бітний формат, ідентичний формату HART-протоколу

Кілька років тому для Profibus була розроблена модифікація інтерфейсу RS-485 для вибухонебезпечних зон – RS-485-IS (Intrinsically Safe – внутрішньо безпечний). Її відмінністю є наявність резисторів, що обмежують струм в лінії до значень, встановлених стандартом на іскробезпечні електричні кола.

Канальний рівень Profibus DP

Канальний рівень моделі OSI в Profibus називається FDL – рівнем (Fieldbus Data Link – "промисловий канал зв'язку"). Об'єкт MAC (Medium Access Control – "управління доступом до каналу") на каналному рівні визначає процедуру передачі даних пристроями, включаючи управління правами на передачу даних через мережу. Протокол каналного рівня забезпечує виконання наступних важливих вимог:

- в процесі комунікації між ведучими пристроями необхідно забезпечити виконання кожним з них свого завдання протягом заздалегідь визначеного інтервалу часу;
- взаємодія ведучих пристроїв (контролерів) з відомими має відбуватися максимально швидко.

9.11 Протокол Modbus

Протокол Modbus і мережа Modbus є найпоширенішими в світі. Незважаючи на свій вік (стандартом де-факто Modbus став ще в 1979 році), Modbus не тільки не застарів, але, навпаки, істотно зросла кількість нових розробок і обсяг організаційної підтримки цього протоколу. Мільйони Modbus-пристроїв по всьому світу продовжують успішно працювати, а остання версія опису протоколу з'явилася в грудні 2006 р.

Одним з переваг Modbus є відсутність необхідності в спеціальних інтерфейсних контролерів (Profibus і CAN вимагають для своєї реалізації замовні мікросхеми), простота програмної реалізації і елегантність принципів функціонування. Все це знижує витрати на освоєння стандарту як системними інтеграторами, так і розробниками контролерного обладнання. Високий ступінь відкритості протоколу забезпечується також повністю безкоштовними текстами стандартів (www.modbus.org).

Популярність протоколу в даний час пояснюється, перш за все, сумісністю з великою кількістю обладнання, яке має протокол Modbus. Крім того, Modbus має високу достовірність передачі даних, пов'язану із застосуванням надійного методу контролю помилок. Modbus дозволяє уніфікувати команди обміну завдяки стандартизації номерів (адрес) регістрів і функцій їх читання-запису.

Основним недоліком Modbus є мережевий обмін по типу ведучий/ведений, що не дозволяє веденим пристроям передавати дані по мірі їх появи і тому вимагає інтенсивного опитування ведених пристроїв провідним.

Різновидами Modbus є протоколи Modbus Plus – багатомайстерний протокол з кільцевої передачею маркера і Modbus TCP, розрахований на використання в мережах Ethernet і інтернет.

Протокол Modbus має два режими передачі: RTU (Remote Terminal Unit – «віддалене термінальне пристрій») і ASCII. Стандарт передбачає, що режим RTU в протоколі Modbus повинен бути присутнім обов'язково, а режим ASCII є опціональним. Користувач може вибирати будь-який з них, але всі модулі, включені в мережу Modbus, повинні мати один і той же режим передачі.

Ми розглянемо тільки протокол Modbus RTU, оскільки Modbus ASCII в Росії практично не використовується. Відзначимо, що Modbus ASCII не можна плутати з приватно-фірмовим протоколом DCON, який використовується в модулях фірм Advantech і ICP DAS і не відповідає стандарту Modbus.

Стандарт Modbus передбачає застосування фізичного інтерфейсу RS-485, RS-422 або RS-232. Найбільш поширеним для організації промислової мережі є 2-провідний інтерфейс RS-485. Для з'єднань точка-точка може бути використаний інтерфейс RS-232 або RS-422.

У стандарті Modbus є *обов'язкові* вимоги, *що рекомендуються* і *опціональні* (необов'язкові). Існує три ступеня відповідності стандарту: «повністю відповідає» - коли протокол відповідає всім обов'язковим і всім рекомендованим вимогам, «умовно відповідає» - коли протокол відповідає тільки обов'язковим вимогам і не відповідає рекомендованим, і «не відповідає».

Модель OSI протоколу Modbus містить три рівні: фізичний, канальний і прикладної (Табл. 9.3).

Таблиця 9.3 – Модель OSI для Modbus

номер	Назва рівня	Реалізація
7	прикладний	MODBUS Application Protocol
6	рівень представлення	немає
5	сеансовий	немає
4	транспортний	немає
3	мережа	немає
2	канальний (передачі даних)	протокол ведучий/ведений Режими RTU і ASCII
1	фізичний	RS-485 або RS-232

Фізичний рівень Modbus

У нових розробках на основі Modbus стандарт рекомендує використовувати інтерфейс RS-485 з двох провідною лінією передачі, але допускається застосування чотирьох ліній і інтерфейсу RS-232.

Modbus-шина повинна складатися з одного магістрального кабелю, від якого можуть бути зроблені відводи. Магістральний кабель Modbus повинен містити 3 провідника в загальному екрані, два з яких представляють собою виту пару, а третій з'єднує загальні ("земляні") висновки всіх інтерфейсів RS-485 в мережі. Загальний провід і екран повинні бути заземлені *в одній точці* , бажано близько провідного пристрою.

Пристрої можуть підключатися до кабелю трьома способами:

- безпосередньо до магістрального кабелю;
- через пасивний перехід (трійник);
- через активний перехід (містить повторювач інтерфейсу).

В документації на пристрій і на трійник повинні бути вказані найменування підключаються ланцюгів.

На кожному кінці магістрального кабелю повинні бути встановлені резистори для узгодження лінії передачі, як це потрібно для інтерфейсу RS-485 (див. Вище). На відміну від фізичного інтерфейсу RS-485, в якому термінальні резистори на низьких швидкостях обміну можна не використовувати, стандарт на протокол Modbus формально вимагає застосування термінальних резисторів для всіх швидкостей обміну. Їх номінал може бути рівним 150 Ом і потужність 0,5 Вт.

Modbus-пристрій обов'язково має підтримувати швидкості обміну 9600 біт / с і 19200 біт / с, з них 19200 біт / с встановлюється "за замовчуванням". Допускаються також швидкості 1200, 2400, 4800, ..., 38400 біт/с, 65 Кбіт/с, 115 Кбіт/с.

Швидкість передачі має бути витримана в передавачі з похибкою не гірше 1%, а приймач повинен приймати дані при відхиленні швидкості передачі до 2%.

Сегмент мережі, що не містить повторювачів інтерфейсу, повинен допускати підключення до 32 пристроїв, проте їх кількість може бути збільшена, якщо це допустимо виходячи з навантажувальної здатності передавачів і вхідного опору приймачів, які повинні бути приведені в документації на інтерфейси. Вказівка цих параметрів в документації є обов'язковою вимогою стандарту.

Максимальна довжина магістрального кабелю при швидкості передачі 9600 біт / с і перетині жив більш 0,13 кв. мм (AWG26) становить 1 км. Відводи від магістрального кабелю не повинні бути довше 20 м. При використанні багато портового пасивного розгалуджувача з N відводами довжина кожного відведення не повинна перевищувати значення $40\text{м}/N$.

Канальний рівень Modbus

Протокол Modbus передбачає, що тільки один пристрій – майстер (контролер) і до 247 ведених (модулів введення-виведення) можуть бути об'єднані в

промислову мережу. Обмін даними завжди ініціюється провідним. Ведені пристрої ніколи не починають передачу даних, поки не отримають запит від ведучого. Ведені пристрої також не можуть обмінюватися даними один з одним. Тому в будь-який момент часу в мережі Modbus може відбуватися тільки один акт обміну.

Адреси з 1 по 247 є адресами Modbus пристроїв в мережі, а з 248 по 255 зарезервовані. Провідний пристрій не повинно мати адреси і в мережі не повинно бути двох пристроїв з однаковими адресами.

Контролер може посилати запити до всіх пристроїв одночасно ("широкомовний режим") або тільки одному. Для широкомовного режиму зарезервовано адресу "0" (при використанні в команді цієї адреси вона приймається всіма пристроями мережі).

9.12 Промисловий Ethernet

Ethernet з'явився більше 30 років тому. В даний час під Ethernet розуміють сімейство продуктів для локальних мереж, які відповідають стандарту IEEE 802.3. Промислового застосування стандарту довгий час заважав метод випадкового доступу до мережі, що не гарантував доставку повідомлення в короткий і заздалегідь відоме час. Однак це проблема була вирішена застосуванням комутаторів. В даний час ринок промислового Ethernet зростає зі швидкістю 51% на рік, він став промисловим стандартом і є великий вибір обладнання, що задовольняє промисловим вимогам. Недоліком промислового Ethernet є відносно висока ціна: Ethernet модулі введення-виведення в середньому в 2 рази дорожче аналогічних Modbus-пристроїв.

Впровадженню Ethernet в промисловість сприяли такі його якості:

- висока швидкість передачі (до 10 Гбіт / с) і відповідність вимогам жорсткого реального часу при високій швидкодії (наприклад, при керуванні рухом);
- простота інтеграції з Internet і Intranet, в тому числі по протоколам прикладного рівня SNMP (Simple Network Management Protocol), FTP, MIME (Multipurpose Internet Mail Extensions), HTTP;
- простота інтеграції з офісними мережами;
- наявність великої кількості фахівців з обслуговування Ethernet;
- по-справжньому відкриті рішення;
- можливість організації багатомайстерних мереж;
- необмежені можливості по організації мереж найрізноманітніших топологій;
- широке застосування в офісних мережах, що забезпечило економічну ефективність технічної підтримки стандарту з боку міжнародних організацій по стандартизації;
- поява недорогих комутаторів, які вирішили проблему недетермінованості Ethernet.

Існують 4 стандартні швидкості передачі даних в мережах Ethernet по оптоволоконному кабелю і кручений парі проводів: 10 Мбіт/с, 100 Мбіт/с, 1 Гбіт/с і 10 Гбіт/с. Ethernet має кілька модифікацій, структура найменувань яких має наступний вигляд: <швидкість передачі> Base <додаткові позначення>. Швидкість вказується цифрою в Мбіт/с або в Гбіт/с, в останньому випадку до цифри додається буква G. Букви після "Base" означають тип кабелю (T – "Twisted pair", "кручена пара", F – "Fiber optic" – "оптоволоконний", S – "Short wavelength optic" – "оптичний короткохвильовий", L – "Long wavelength" – "довгохвильовий", C – "short Copper cable"). Символ "X" означає наявність блоку кодування на фізичному рівні.

Таблиця 9. 4 – Рівні моделі OSI для мережі Ethernet

Номер рівня	OSI модель	мережа Ethernet	варіанти реалізації
7	прикладний		
6	рівень представлення	прикладний	HTTP, FTP, SMTP, DNS NFS, XDR, RPC
5	сеансовий		
4	транспортний	транспортний	TCP UDP
3	Мережевий	Інтернет-протокол	IP
2	Канальний (передачі даних)	LLC або інші клієнти MAC MAC	Ethernet IEEE 802.3
1	фізичний	PHY	Вита пара, оптичний кабель, коаксіальний кабель

У табл. 9.4 приведена типова модель OSI для локальної мережі LAN, в якій Ethernet, як стандарт IEEE 802.3, охоплює тільки фізичний і канальний рівень. На інших рівнях представлені протоколи інтернету. Тут LLC - "Logical Link Control" ("управління логічними зв'язками") - клієнт підрівні MAC; MAC ("Medium Access Control") - підрівень управління доступом до лінії передачі; PHY (від "PHYsical") - фізичний рівень (лінія передачі).

Рівні 1 і 2 Ethernet-протоколу зазвичай втілюються апаратно, інші - програмно.

Відмінні особливості

При використанні стандартного Ethernet час реакції мережі часто перевищує 100 мс. У локальних сегментах мережі з малою кількістю пристроїв воно знижується до 20 мс при використанні протоколу TCP і до 10 мс - для UDP. Використання прямої MAC – адресації в локальних сегментах мережі може зменшити час реакції до 1 мс. Однак цей час в Ethernet є випадковою величиною. Тому основною відмінністю промислового Ethernet від офісного є наявність

комутаторів, які перетворюють Ethernet в мережу з детермінованим поведінкою.

Для промислових застосувань не годяться кабелі і роз'єми, що використовуються в офісних мережах. Тому ряд виробників представляють на ринку роз'єми RJ-45 зі ступенем захисту IP67 від впливів навколишнього середовища.

Надійність промислового Ethernet забезпечується резервуванням кабельних ліній і мережевих карт, а також спеціальним програмним забезпеченням для резервування.

Для підвищення ступеня захисту системи управління від несанкціонованого доступу магістральну мережу Industrial Ethernet відокремлюють від офісної брандмауером *.

Відмінними ознаками промислового Ethernet є:

- відсутність колізій і детермінованість поведінки завдяки застосуванню комутаторів;
- індустриальні кліматичні умови;
- стійкість до вібрацій;
- відсутність вентиляторів в обладнанні;
- підвищені вимоги до електромагнітної сумісності;
- компактність, кріплення на DIN-рейку;
- зручне підключення кабелів;
- діагностична індикація на панелі приладу;
- електроживлення від джерела напруги в діапазоні від 10 до 30 В;
- можливість резервування;
- роз'єми і обладнання зі ступенем захисту до IP67;
- захист від електростатичних зарядів, електромагнітних імпульсів, від перевищення напруги живлення;
- повнодуплексна передача.

Недоліком Ethernet є значний рівень "накладних витрат" в протоколах TCP/IP, які розраховані на пересилку даних великих обсягів. Якщо ж пристрій пересилає всього 8 байт, що типово для АСУ ТП, то корисна інформація в протоколі становить всього 11%, оскільки кожен пакет TCP / IP міститься 68 байт службової інформації. Однак це часто можна компенсувати великою пропускну здатністю мережі.

Стандарт Profinet

Profinet (Profibus + Ethernet) є одним із сучасних стандартів (IEC 61158 та IEC 61784), який забезпечує просту інтеграцію пристроїв Profibus, ASI і Interbus з промисловим Ethernet. Протокол ProfiNet розроблений Profibus International на базі Ethernet як сумісний з протоколом Profibus, підтримуваним компанією Siemens.

Бездротові локальні мережі

Існує багато об'єктів автоматизації, де складно обійтися без бездротових мереж або де їх застосування явно бажано:

- сенсори і виконавчі пристрої *на рухомих частинах* конвеєрів, вітряних млинів, ліфтів, міксерів, візків для переміщення вантажів по цеху, на крилах і лопатях літаків, на підшипниках двигунів, на роботах, в пересувних лабораторіях, сенсори на тілі людини і тварин; сенсори вібрації на контейнерах для перевезення вантажів;

- об'єкти, в яких небажано свердлими стіни або псувати дизайн: офісні будівлі, в яких встановлюється пожежна та охоронна сигналізація, сенсори для систем обігріву і кондиціонування повітря, для моніторингу механічної напруги в конструкціях будівель; в системах "розумного будинку" (керування освітленням, кондиціонуванням-обігрівом, охоронними датчиками, побутовими приладами та ін.);

- епізодичне програмування і діагностика ПЛК, коли прокладати постійні кабелі не вигідно; дистанційне зчитування показань лічильників, самописців;

- об'єкти з агресивними середовищами, вібрацією; об'єкти, що знаходяться під високою напругою або в місцях, не зручних для прокладки кабелю;

- відстеження траєкторії руху транспорту, охорона кордонів держави, моніторинг напруженості автомобільного трафіку в містах та умов на дорогах, моніторинг лісу, моря, сільськогосподарських культур, моніторинг шкідливих викидів в екології;

- будь-які об'єкти, для яких відомо, що вартість кабелів, кабельних каналів, опор або траншей, а також робіт з монтажу та обслуговування істотно перевищує витрати на заміну бездротової системи, за умови відсутності жорстких вимог до надійності доставки повідомлень в реальному часі;

- об'єкти у вибухонебезпечних зонах.

У більшості застосувань бездротові мережі дозволяють досягти наступних переваг в порівнянні з провідними мережами:

- істотно знизити вартість установки сенсорів;
- виключити необхідність профілактичного обслуговування кабелів;
- виключити дорогі місця розгалужень кабелю;
- зменшити кількість кабелів;
- зменшити трудовитрати і час на монтаж і обслуговування системи;
- знизити вартість системи за рахунок виключення кабелів;
- знизити вимоги до навчання персоналу монтажної організації;
- прискорити налагодження системи і пошук несправностей;
- забезпечити зручну модернізацію системи.

Оскільки реконфігурація системи і її монтаж стають набагато простішими, бездротові мережі можна використовувати і в традиційних областях застосування кабельних зв'язків, коли вартість кабелю і монтажу буде вищою, ніж установка бездротової системи.

Бездротові мережі діляться на наступні класи:

- стільникові мережі WWAN (Wireless Wide Area Network);
- бездротові LAN (WLAN - Wireless LAN);

- бездротові мережі сенсорів.

У промисловій автоматизації найбільшого поширення набули три типи бездротових мереж: Bluetooth на основі стандарту IEEE 802.15.1, ZigBee на основі IEEE 802.15.4 і Wi-Fi на основі IEEE 802.11. Фізичні рівні моделі OSI для цих мереж засновані на відповідних стандартах IEEE, а протоколи верхніх рівнів розроблені і підтримуються організаціями Bluetooth, ZigBee і Wi-Fi відповідно. Тому в назві мереж зазвичай вказують посилання на стандарт. Всі три мережі використовують неліцензованому ISM (Industrial, Scientific, and Medical) діапазон 2,4 ГГц.

9.13 Контрольні запитання і завдання

1. Які основні принципи побудови Індустрії 4.0?
2. Що потребує застосування розподілених систем?
3. Характеристики розподілених систем?
4. Наведіть приклад моделі розподіленої системи автоматизації.
5. Приклад архітектури розподіленої системи?
6. Яку проблему вирішують розподілені системи?
7. Наведіть приклад багаторівневої розподіленої системи
8. Що представляють собою рівні ієрархії сучасної АСУ?
9. Відмінності промислової мережі?
10. Класифікація промислових мереж?
11. Загальні відомості про промислові мережі?
12. Приведіть типи даних в промислових мережах.
13. Основні параметри характеристики промислових мереж?
14. Сім рівнів моделі OSI?
15. Яке призначення фізичного рівня?
16. Що таке прикладний рівень моделі OSI?
17. Яке призначення сеансового рівня?
18. Яке призначення канального рівня?
19. Яке призначення транспортного рівня?
20. Яке призначення рівня мережі та рівня представлення?
21. Які основні характеристики інтерфейсів RS-422 і RS-485?
22. Які переваги надає диференційний спосіб передачі сигналу?
23. Що дає наявність третього стану в RS-422 і RS-485?
24. Інтерфейс струмова петля. В чому його відмінність?
25. Аналогова версія струмової петлі.
26. Принцип реалізації цифрової струмової петлі.
27. Магістральний протокол HART. Характеристика та особливості.
28. В чому особливості протоколів Modbus, Profibus, DeviceNet, CANopen, Foundation Fieldbus?
29. Бездротові локальні мережі: в чому полягає їх необхідність?
30. Що є відмінними ознаками промислового Ethernet?

РЕКОМЕНДОВАНА ЛІТЕРАТУРА

1. Гвоздева, В. А. Основы построения автоматизированных информационных систем / В.А. Гвоздева, И.Ю. Лаврентьева. - Москва: СИНТЕГ, 2007. - 320 с.
2. STM32 F4 series of high-performance [Электронный ресурс]
3. Trevor Martin: The Insider's Guide To The STM32 ARM Based Microcontroller; Published – Coventry, UK: Hitex (UK) Ltd., 2009. – 103 p., ISBN: 0-9549988 8.
4. Ключев, А. С. Проектирование систем автоматизации технологических процессов / А.С. Ключев, Б.В. Глазов, А.Х. Дубровский. - М.: Энергия, 2015. - 512 с.
5. Фельдбаум, А. А. Вычислительные устройства в автоматических системах / А.А. Фельдбаум. - М.: Государственное издательство физико-математической литературы, 2017. - 800 с.
6. Грушвицкий Р. И., Мурсаев А. Х., Угрюмов Е. П. Проектирование систем на микросхемах с программируемой структурой. — 2-е изд., перераб. и доп. – СПб.: БХВ– Петербург, 2006. — 736 с.: ил. с. 144–160.
7. Стемпковский Системная среда САПР СБИС / Стемпковский. - М.: Наука, 2017. - 256 с.
8. Мельцер, М. И. Разработка алгоритмов АСУП / М.И. Мельцер. - М.: Статистика, 2014. - 240 с.
9. Передача данных в системах контроля и управления: практическое руководство / Дж. Парк, С. Маккей, Э. Райт ; – М.: ООО «Группа ИДТ», 2007. - 480 с.
10. Kuphaldt T.R. Lessons In Industrial Instrumentation. New York: Autoedition, 2017. — 3283 p.
11. Lamb F. Industrial Automation: Hands On. McGraw-Hill, New York, 2013, 352 p.
12. Clarke G., Reynders D., Wright E. Practical Modern SCADA Protocols: DNP3, 60870.5 and Related Systems. Oxford: IDC Technologies, 2004. — 548 p.
13. Проектування комп'ютерних систем на основі мікросхем програмованої логіки : монографія / С. А. Іванець, Ю. О. Зубань, В. В. Казимир, В. В. Литвинов. – Суми : Сумський державний університет, 2013. – 313 с.
14. Грушвицкий Р.И., Шашкин П.М. Проектирование в условиях временных ограничений: компиляция проектов//Компоненты и технологии, 2011. – № 1– 3.
15. [www/Siemens.de](http://www.Siemens.de) [Электронный ресурс]
16. [www/ti.com](http://www.ti.com) [Электронный ресурс]
17. [www/Cypress.com](http://www.Cypress.com) [Электронный ресурс]

Навчальне видання

БОНДАРЕНКО Ігор Миколайович, БОРОДІН Олександр Васильович,
КАРНАУШЕНКО Володимир Петрович

МІКРОПРОЦЕСОРНІ СИСТЕМИ КОНТРОЛЮ ТА КЕРУВАННЯ

Навчальний посібник

Відповідальний випусковий

І.М. Бондаренко

Редактор

О.Ф. Мусієздов

Комп'ютерна верстка

Л.Д. Медведєва

План 2019, (друге півріччя) поз. 8
Підп. до друку 26.06.19. Формат 60х 84 ¹/₁₆. Спосіб друку – ризографія.
Умов. друк. арк. 14,1. Облік.вид.арк. 13,9. Тираж 100 прим.
Зам. № 1 – 8 Ціна договірна.

ХНУРЕ, 61166, Харків, пр. Науки, 14

Віддруковано у редакційно–видавничому відділі ХНУРЕ

Харків, пр. Науки 14