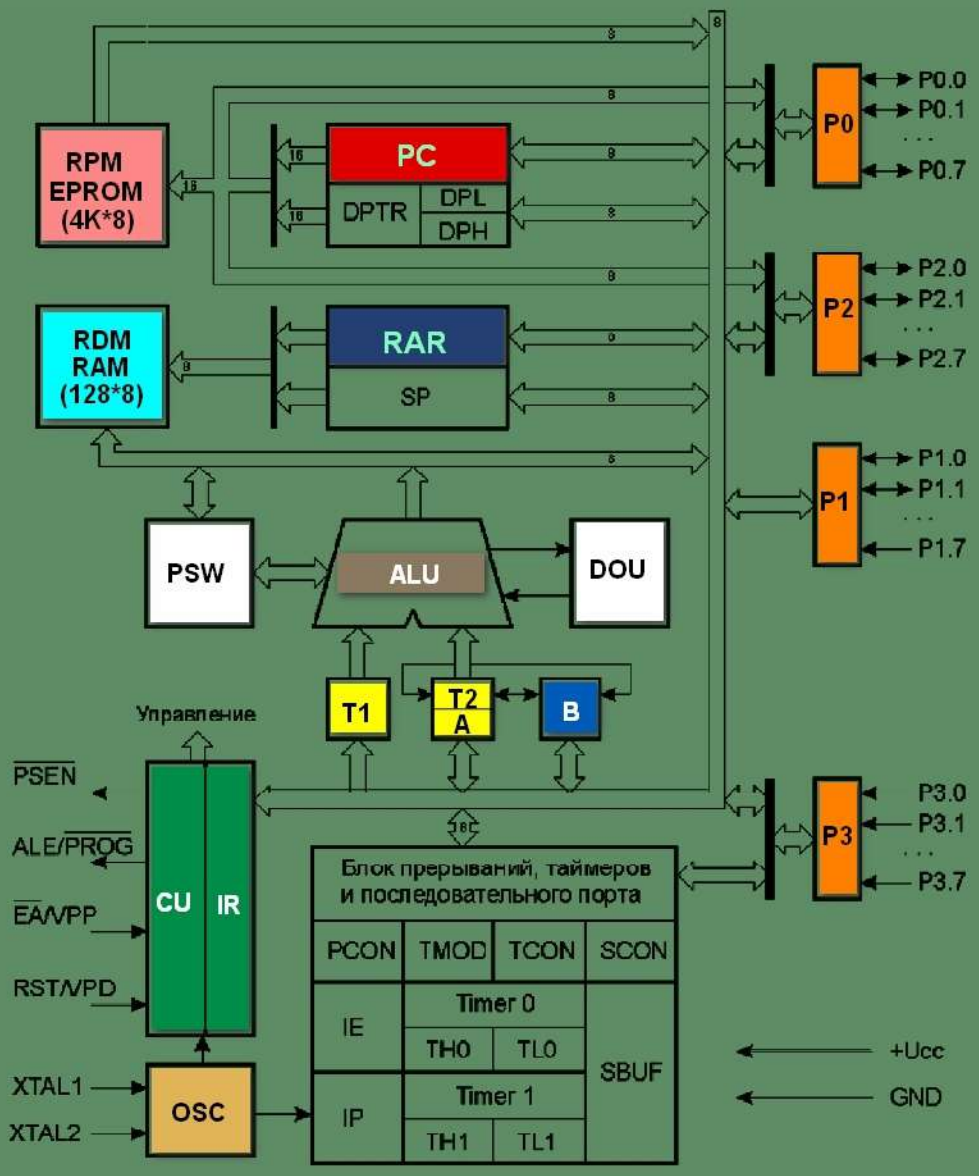
А.О. Новацький

ПРОЕКТУВАННЯ МИКРОПРОЦЕСОРНИХ СИСТЕМ НА БАЗІ МІКРОКОНТРОЛЕРІВ СІМЕЙСТВА MCS-51

Навчальний посібник



А.О. Новацький

ПРОЕКТУВАННЯ МІКРОПРОЦЕСОРНИХ СИСТЕМ НА БАЗІ МІКРОКОНТРОЛЕРІВ СІМЕЙСТВА MCS-51

✦

Навчальний посібник

для студентів вищих навчальних закладів, які навчаються за напрямом підготовки «Системна інженерія».

Київ
НТУУ «КПІ»
2016

Проектування мікропроцесорних систем: Проектування мікропроцесорних систем на базі мікроконтролерів сімейства MCS–51: Периферійні модулі мікроконтролерів сімейства MCS–51: Навчальний посібник для студентів напряму підготовки 6.050201 «Системна інженерія» кафедри Автоматики та управління у технічних системах / Автор: А.О. Новацький–К: НТУУ „КПІ”, 2016– 391с.

Відповідальний за випуск: к. т. н., доц. Л.Ю. Юрчук

Рецензенти: доктор фіз. мат. наук, професор А.Ю.Дорошенко,

к. т. н., доц. О.А.Чемерис

Надано гриф «Затверджено Вченою радою НТУУ «КПІ» як навчальний посібник для студентів, які навчаються за спеціальністю «Автоматизація та комп`ютерно–інтегровані технології»

(Протокол № 5 від 11 квітня 2016 року)

Навчальний посібник охоплює теоретичний матеріал та відповідні приклади, які необхідні для вивчення базової частини дисципліни «Проектування мікропроцесорних систем». Навчальний посібник складається із одинадцяти розділів: склад та основні характеристики мікроконтролерів сімейства МК–51; структура типового мікроконтролера МК–51; модуль таймерів/лічильників; паралельні порти; підсистема переривань; послідовний асинхронний інтерфейс (УАПП); організація пам’яті; тактування мікроконтролера; режим зниженого енергоспоживання; скидання мікроконтролера; послідовний синхронний інтерфейс SMBus (I^2C); послідовний периферійний інтерфейс SPI; послідовний периферійний інтерфейс CAN; аналоговий компаратор; аналого–цифровой перетворювач та цифро–аналоговий перетворювач.

В кожному розділі розглядаються основні архітектурні особливості мікроконтролерів МК–51 на апаратному рівні, наводяться приклади використання засобів цих мікроконтролерів.. Робота може бути корисною студентам відповідних спеціальностей при вивченні дисциплін, пов’язаних із проектуванням та використанням мікропроцесорних систем, а також при виконанні бакалаврських робіт, курсових та дипломних проектів, в яких використовуються мікропроцесорні пристрої. Останнє було враховано при оформленні роботи, яке виконано згідно вимог до конструкторської документації.

ЗМІСТ

ВСТУП

Перші мікроконтролери сімейства MCS–51 (МК51) було розроблено фірмою Intel у 1980 році для використання у вбудованих системах. Архітектура сімейства MCS–51 була настільки вдалою, що стала стандартом на світовому ринку 8–розрядних мікроконтролерів.

MCS–51 прийшли на заміну випущеним у 1976 році MCS–48 і на відміну від останніх мали зменшений час виконання команд (в 2,5 – 10 разів в залежності від умов експлуатації), збільшений обсяг вбудованої пам'яті, додаткові пристрої периферії, додаткові команди для програмування. За рахунок даних покращень, мікроконтролери стали зручнішими в програмуванні, дешевші в експлуатації.

Конструктивно, MCS–51 є однокристальним мікроконтролером гарвардської архітектури, який виконано за n–МОН або КМОН–технологією. Базовий мікроконтролер MCS–51 містить у собі 8–бітний мікропроцесор і8051 з підтримкою булевих операцій над окремими бітами, до 4096 байт вбудованої пам'яті програм (доступної тільки для читання), до 256 байт вбудованої пам'яті даних (доступної для читання і запису), підтримка адресного простору у 64 Кбайт для пам'яті програм і 64 Кбайт для зовнішньої пам'яті даних, два 16–бітні таймери/лічильники, двосторонній УАПП, генератор тактової частоти. Для підготовки програмного забезпечення мікроконтролерів MCS–51 використовуються в основному мови "ASM–51" і "C", для яких існують ряд компіляторів, бібліотек, стандартних підпрограм і програмних емуляторів, які вироблено різними закордонними та вітчизняними фірмами.

В даний час серед всіх 8–розрядних мікроконтролерів – сімейство MCS–51 широко використовується і має велику кількість різновидів і кількость компаній, що випускають його модифікації. Воно отримало свою назву від першого представника цього сімейства – мікроконтролера 8051, який випущено в 1980 році. Вдалий набір периферійних пристроїв, можливість гнучкого

вибору зовнішньої або внутрішньої програмної пам'яті і прийнятна ціна забезпечили цьому мікроконтролеру успіх на ринку.

Важливу роль у досягненні такої високої популярності сімейства МК51 зіграла відкрита політика фірми Intel, родоначальниці архітектури, яку було спрямовано на широке поширення ліцензій на ядро 8051 серед великої кількості провідних напівпровідникових компаній світу. Фірма Intel досить давно не займається випуском 8–розрядних мікропроцесорів. Велика кількість виробників придбали ліцензію на випуск мікроконтролерів з ядром MCS–51: Philips, Siemens, Dallas, Atmel, Winbond та інші. Всі компанії прагнуть одночасно зберегти сумісність зі стандартною архітектурою MCS–51 і привнести в неї які–небудь покращення, які дозволяють продукції виділятися на фоні продуктів конкуруючих фірм.

В результаті на сьогоднішній день існує більше 200 модифікацій мікроконтролерів сімейства 8051. Ці модифікації містять у собі кристали з найширшим спектром периферії: від простих 20–контактних пристроїв з одним таймером і 1К програмної пам'яті до найскладніших 100–контактних кристалів з 10–розрядними АЦП, масивами таймерів–лічильників, апаратними 16–розрядними множниками і 64К програмної пам'яті на кристалі.

Мікроконтролери фірми Atmel – найбільш популярні серед розробників з країн СНД. Цьому сприяє їх невисока ціна і наявність в гамі продукції великої кількості мікроконтролерів з флеш–пам’яттю програм, що особливо важливо для дрібносерійного виробництва. Поряд з повними стандартними версіями контролерів, фірма Atmel пропонує більш дешеві кристали з меншою кількістю виводів (20 замість 40 стандартних).

Однією із найавторитетніших компаній на ринку мікроконтролерів є Siemens. В результаті реорганізації підрозділ по випуску напівпровідників отримав свою власну назву – Infineon. Фірма Infineon випускає, разом з іншими

цікавими мікросхемами, мікроконтролери сімейства C500. Архітектура C500 базується на MCS–52, але має більш розвинуту периферію.

Велику номенклатуру мікроконтролерів з ядром MCS–51 випускає фірма Philips. Продукція фірми орієнтована, в першу чергу, на ринок побутової електроніки. Більшість кристалів мають інтерфейс I^2C. I^2C – це послідовна шина, яку запропоновано фірмою Philips в якості стандарту обміну в недорогих низькошвидкісних системах.

Фірма Analog Devices вийшли на ринок 8–розрядних мікроконтролерів, запропонувавши сімейство MicroConverter. У цих виробах об'єднано на одному кристалі найважливіші компоненти систем збору і обробки інформації – процесорне ядро MCS–52 і блок введення–виведення аналогової інформації, який включає багатоканальні ЦАП і АЦП.

Особливістю мікроконтролерів фірми Dallas Semiconductor є змінена часова сітка. Машинний цикл в даних мікропроцесорах займає 4 такти замість 12 у класичного 80C51. При цьому збережено повну сумісність зі стандартною архітектурою на рівні інструкцій. Мікроконтролери Dallas знаходять застосування в завданнях, які потребують підвищеної швидкості обробки.

Сьогодні фірма Silicon Laboratories динамічно розвиває перелік мікроконтролерів з ядром МК51. Фірмі вдалося створити велику кількість мікроконтролерів, що відрізняються продуктивністю, об'ємом Flash–пам'яті програм, вбудованої оперативної пам'яті, характеристиками периферійних вузлів, типом корпусу та іншими параметрами.

Всі мікроконтролери сімейства MCS–51 мають спільну систему команд. Наявність додаткового вбудованого периферійного обладнання впливає тільки на кількість регістрів спеціального призначення.

Поява на ринку 16–та 32–розрядних мікроконтролерів та цифрових сигнальних процесорів, які суттєво переважають 8–розрядні за продуктивністю та потужністю, не витіснило останні з ринку. Більш того, за кількістю

модифікацій 8–розрядні мікроконтролери значно перевершують всі інші групи. Головна причина криється в тому, що основна область застосування 8–розрядних мікроконтролерів – пристрої інтелектуального керування промислової автоматики та побутової апаратури. Специфіка алгоритмів керування цих пристроїв не вимагає виконання розрахунків високої точності в жорстких умовах реального часу. Основна частина операцій керування полягає у перетворенні логічної інформації, і 8–розрядні мікроконтролери з успіхом реалізують ці завдання.

За рахунок вдалої реалізації мікроконтролера, велика кількість наявних на ринку мікроконтролерів має і8051 сумісні процесори, тому ядро MCS–51 є одним з перших кроків до вивчення сучасних мікроконтролерів у програмах курсів у вищих навчальних закладах. Незважаючи на солідний вік сімейства і появу на світовому ринку однокристальних мікроконтролерів інших виробників, мікроконтролери MCS–51 ще достатньо довго будуть використовуватися в простих вбудованих системах керування.

СПИСОК СКОРОЧЕНЬ

АЛП – арифметико–логічний пристрій

АЦП – аналого–цифровий перетворювач

БРА – буферний регістр адреси

ВІС – велика інтегральна схема

ДК – двійковий (машинний) код

ДНЛ – диференціальна нелінійність

ДША – дешифратор адреси

ЕОМ – електронно–обчислювальна машина

ЗПД – зовнішня пам’ять даних

ЗПП – зовнішня пам'ять програм

КМОН – комплементарний метал–оксид–напівпровідник

КОП – код операції команди

КЦ – командний цикл

ЛК – лічильник команд

МА – мова асемблера

МЗР – молодший значущий розряд

МК – мікроконтролер

МОН – метал–оксид–напівпровідник

МП – мікропроцесор

МПС – мікропроцесорна система

НОЗП – над оперативний запам’ятовуючий пристрій

ОЗП – оперативний запам’ятовуючий пристрій

ПВВ – порти введення/виведення

ПЗП – постійний запам’ятовуючий пристрій

ПЛМ – програмована логічна матриця

ПП – пам’ять програм

ППЗП – перепрограмовуваний постійний запам'ятовуючий пристрій

ППІ – паралельний програмований інтерфейс

ППР – периферійний пристрій

РА (RAR) – регістр адреси

РВВ – розширювач введення/виведення

РЗП – регістр загального призначення

РКС – регістр керуючого слова

РКСТ – регістр керування–статусу таймерів/лічильників

РМП – регістр масок переривань

РП (IP) – регістр пріоритетів переривань

РПД – резидентна пам’ять даних

РПП – резидентна пам'ять програм

РПС (SP) – регістр–покажчик стеку

РРТЛ – регістр режимів таймерів/лічильників

РСП (PSW) – регістр стану програми (прапорців)

РСФ – регістр спеціальних функцій

РКПП -– регістр керування приймачем–передавачем

РШД – резидентна шина даних

СЗР – старший значущий розряд

ССП – слово стану програми

Т/Л (Timer/Counter – T/C) – таймер–лічильник

УАПП – універсальний асинхронний приймач–передавач

УСАПП – універсальний синхронно/асинхронний приймач–передавач

ФАПЧ – фазове автопідстроювання частоти

ЦАП – цифро–аналоговий перетворювач

ША – шина адреси

ШД – шина даних

ШІМ – широтно–імпульсна модуляція

BCD (binary–coded decimal) – двійково–десятковий код

BSP – Bit Stream Processor

BTL – Bit Timing Logic

CAN – Controller Area Network

CRC – Cyclic Redundancy Check Register

DLC – Data Length Code

DPH – старший байт регістра DPTR

DPL – молодший байт регістра DPTR

DPTR – регістр–покажчик даних

DRAM – (Dynamic Random Access Memory) – динамічна оперативна пам’ять з довільним доступом

EEPROM (Electrically Erasable Programmable Read–Only Memory) – енергонезалежна пам'ять, яку можна електрично стерти та перепрограмувати

EML– Error Management Logic

EPROM (Erasable Programmable Read–Only Memory) – клас напівпровідникових запам'ятовуючих пристроїв, постійна пам'ять, яка допускає перезапис (програмування) і для запису інформації в яку використовується електронний пристрій–програматор

FSM – Finite State Machine

I^2C (InterIC, або IIC) – двонаправлена двопровідна шина для так званого міжмікросхемного застосування

JTAG (Joint Test Action Group) – інтерфейс, який призначено для підключення складних цифрових мікросхем або пристроїв рівня друкованої плати до стандартної апаратури тестування і налагодження

PC – програмний лічильник

PROM – Programmable Read Only Memory

PSW – регістр стану програми

RAM (Random Access Memory) – пам’ять з довільним доступом

ROM (Read Only Memory) – пам’ять постійного зберігання (пам’ять тільки для читання)

SAM (Sequental Access Memory) – пам’ять з послідовним доступом

SBUF – буфер послідовного порту

SCL (Serial CLock) – послідовна лінія тактування

SP – регістр–покажчик стеку

SPI (Serial Peripheral Interface) – послідовний периферійний інтерфейс

SRAM (Static Random Access Memory) – статична оперативна пам’ять з довільним доступом

TTCAN – Time Triggered CAN

TWI (Two Wire (Serial) Interface) – двопровідний інтерфейс

UV–EPROM – Ultraviolet Erasable Programmable Read Only Memory

1 СКЛАД ТА ОСНОВНІ ХАРАКТЕРИСТИКИ МІКРОКОНТОЛЕРІВ СІМЕЙСТВА МК–51

1.1 Загальні відомості

Фірма Intel досить давно не займається випуском 8–розрядних мікропроцесорів. Велика кількість виробників придбали ліцензію на випуск мікроконтролерів з ядром MCS–51: Philips, Siemens, Dallas, Atmel, Winbond та інші. Всі компанії прагнуть одночасно зберегти сумісність зі стандартною архітектурою MCS–51 і привнести в неї які–небудь покращення, дозволяючи продукції виділятися на фоні продуктів конкуруючих фірм.

1.2 Мікроконтролери фірми Atmel

Мікроконтролери фірми Atmel – одні з найбільш популярних серед розробників з країн СНД. Цьому сприяє їх невисока ціна і наявність в гамі продукції великої кількості мікроконтролерів з флеш–пам'яттю програм, що особливо важливо для дрібносерійного виробництва. Поряд з повними стандартними версіями контролерів, фірма Atmel пропонує більш дешеві кристали з меншою кількістю виводів (20 замість 40 стандартних).

В таблиці 1.1 приведено характеристики деяких мікроконтролерів фірми Atmel з флеш–пам'яттю програм. Фірма випускає також одноразово програмовані (ОП) версії кристалів, але вони не користуються такою популярністю.

1.3 Мікроконтролери фірми Temic

Фірма Temic нещодавно стала підрозділом Atmel із назвою Atmel Wireless. На час об'єднання у цієї фірми був багатолітній досвід розробки и випуску мікроконтролерів з ядром MCS–52.

Таблиця 1.1 – Мікроконтролери фірми Atmel

Тип	Напруга живлення, В	Тактова частота, МГц	I/O	FLASH	EEPROM	SRAM	Інтерфейси	Аналогові інтерфейси	Таймери	Корпус
Мікроконтролери з Flash–пам'яттю програм										
AT89C2051	2.7–6.0	24	15	2K	–	128	UART	Comp	2x16–bit	PDIP20, SOIC20
AT89C4051	3.0–5.0	24	15	4K	–	128	UART	Comp	2x16–bit	PDIP20, SOIC20
AT89C51	5.0	24	32	4K	–	128	UART	–	2x16–bit	DIP40, PLCC44, PQFP44
AT89LV51	2.7	12	32	4K	–	128	UART	–	2x16–bit	DIP40, PLCC44, PQFP44
AT89C52	5.0	24	32	8K	–	256	UART	–	3x16–bit	DIP40, PLCC44, PQFP44
AT89LV52	2.7	12	32	8K	–	256	UART	–	3x16–bit	DIP40, PLCC44, PQFP44
AT89C55WD	5.0	33	32	20K	–	256	UART	–	3x16–bit	DIP40, PLCC44, PQFP44
AT89LV55	2.7	12	32	20K	–	256	UART	–	3x16–bit	DIP40, PLCC44, PQFP44
AT89C51RC	5.0	33	32	32K	–	512	UART	–	3x16–bit PCA	DIP40, PLCC44, PQFP44
Мікроконтролери з інтерфейсом USB, які програмуються внутрішньосхемно										
AT89C51SND1	2.7–3.3	20	44	64K	–	2304	UART, IDE, USB, SPI, I^2S, MP3 Decoder	10–bit ADC	2x16–bit WDT	TQFP80, PLCC84
AT89C51SND2	2.7–3.3	20	44	64K	–	2304	UART, IDE, USB, SPI, I^2S, MP3 Decoder	10–bit ADC, 2x20bit DAC	2x16–bit WDT	CTBGA100
AT89C5131	3.0–3.6	40	34 18	32K	4Kb	1280	UART, USB, SPI	–	3x16–bit WDT, PCA	PLCC52, VQFP64, MLF48, SO28

Продовження таблиці 1.1

AT89C5131A–L	3.0–3.6	48	34 18	32K	4Kb	1280	UART, USB, SPI, I^2C	–	3x16–bit WDT, PCA	PLCC52, VQFP64, MLF48, SO28
AT89C5130A–M	2.7–5.5	48	34	16K	–	1280 + DPRAM 1280	UART, USB, SPI	–	2x16–bit WDT	PLCC52, VQFP64, QFN32
AT89C5131A–M	3.0–3.6	40	34 18	32K	256	1024	USB, SPI	–	2x16–bit WDT	LCC52, VQFP64, MLF48, SO28
AT89C5132	2.7–3.3	40	44 38	64K	–	2304	UART, USB, SPI, I^2S	2ch 10bit	3x16–bit WDT	TQFP80, TQFP64
AT85C5122	3.6–5.5	16	46 13	32K CRAM		768	UART, USB, SPI, Smart Card	–	2x16–bit WDTA	VQFP64, PLCC28
AT89C5122	3.6–5.5	16	46 13	32K		768	UART, USB, SPI, Smart Card	–	2x16–bit WDTA	VQFP64, PLCC28
Мікроконтролери, які програмуються внутрішньосхемно										
T85C5121	3.6–5.5	16	14 30	16K CRAM		768	UART, SPI, Smart Card	–	2x16–bit WDTA	SSOP24, PLCC52
T89C5121	3.6–5.5	16	14 30	16K		768	UART, SPI, Smart Card	–	2x16–bit WDTA	SSOP24, PLCC52
AT89S51	4.0–5.5	33	32	4K	–	128	UART	–	2x16–bit WDT	DIP40, PLCC44, PQFP44
AT89LS51	2.7–4.0	16	32	4K	–	128	UART	–	2x16–bit WDT	DIP40, PLCC44, PQFP44
AT89S2051	2.7–5.5	24	15	2K	–	256	UART	–	2x16–bit	PDIP20, SO20
AT89S4051	2.7–5.5	24	15	4K	–	256	UART	–	2x16–bit	PDIP20, SO20
AT89S52	4.0–5.5	33	32	8K	–	256	UART	–	3x16–bit WDT	DIP40, PLCC44, PQFP44
AT89LS52	2.7–4.0	33	32	8K	–	256	UART	–	3x16–bit WDT	PDIP40, PDIP42, PLCC44, TQFP44
AT89S8252	4.0–6.0	24	32	8K	2K	256	UART, SPI	–	3x16–bit WDT	DIP40, PLCC44, PQFP44
AT89S8253	2.7–5.5	24	32	12K	2K	256	UART, SPI	–	3x16–bit WDT	DIP40, PLCC44, PQFP44, PDIP42

Продовження таблиці 1.1

AT89LS8252	2.7–6.0	12	32	8K	2K	256	UART, SPI	–	3x16–bit WDT	DIP40, PLCC44, PQFP44
AT89S53	4.0–6.0	24	32	12K	–	256	UART, SPI	–	3x16–bit WDT	DIP40, PLCC44, PQFP44
AT89LS53	2.7–6.0	12	32	12K	–	256	UART, SPI	–	3x16–bit WDT	DIP40, PLCC44, PQFP44
T89C51RB2M	4.5–5.5	40	32	16K	–	1280	UART, SPI	–	3x16–bit PCA, WDT	DIP40, PLCC44, VQFP44
T89C51RB2L	2.7–3.3	40	32	16K	–	1280	UART, SPI	–	3x16–bit PCA, WDT	DIP40, PLCC44, VQFP44
T89C51RC2	4.5–5.5	40	32	32K	–	1280	UART, SPI	–	3x16–bit PCA, WDT	DIP40, PLCC44, VQFP44
T89C5115	4.5–5.5	40	20	16K	2K	512	UART	10–bit ADC	3x16–bit PCA, WDT	SOIC28, PLCC28, VQFP32
T89C51IC2	2.7–3.6 4.5–5.5	40	32	32K	–	1280	UART, SPI, I^2C	–	3x16–bit PCA, WDT	PLCC44, VQFP44
AT89C51ID2	2.7–3.6 3.0–5.5	40	32 48	64K	2K	2K	UART, SPI, I^2C	–	3x16–bit PCA WDT	PLCC44, VQFP44 PLCC68, VQFP64
T89C51AC2	4.5–5.5	40	34	32K	2K	1280	UART	10–bit ADC	3x16–bit PCA, WDT	VQFP44, PLCC44, CA–BGA64
AT89C51AC3	3.0–5.5	60	36	64K	2K	2304	UART, SPI	10–bit ADC	3x16–bit PCA, WDT	VQFP44, PLCC44, VQFP64, PLCC52
T89C51RD2	3.0–5.5 4.5–5.5	40	32 48	64K	2K	1280	UART	–	3x16–bit PCA, WDT	PDIL40, PLCC44, VQFP44, PLCC68, VQFP64
AT89C51RD2	2.7–5.5	40	32 48	64K	–	1280	UART	–	3x16–bit PCA, WDT	PLCC44, VQFP44
AT89C51ED2	2.7–5.5	40	32 48	64K	2K	2048	UART	–	3x16–bit PCA, WDT	PDIL40, PLCC44, VQFP44, PLCC68, VQFP64

Продовження таблиці 1.1

AT89C51IC2	2.7–5.5	40	34 48	32K	–	1280	UART, TWI	–	3x16–bit PCA, WDT	LQFP44, PLCC44
AT89C51RB2	2.7–5.5	40	32 48	16K	–	1280	UART	–	3x16–bit PCA, WDT	PDIP40, LQFP44, PLCC44
T89C51CC01	4.5–5.5	40	34	32K	2K	1280	UART, CAN	10–bit ADC	3x16–bit PCA, WDT	TQFP44, PLCC44, CA–BGA64
T89C51CC02	2.7–3.3 4.5–5.5	40	20	16K	2K	512	UART, CAN	10–bit ADC	3x16–bit PCA, WDT	PLCC28, SOIC28, TSSOP28, SOIC24
AT89C51CC03	2.7–3.3 4.5–5.5	40	34	64K	2K	2304	UART, CAN	10–bit ADC	3x16–bit PCA, WDT	VQFP44, PLCC44, CA–BGA64
AT89LP213	2.4–5.5	20	14	2K	–	128	TWI, SPI	–	2x16–bit WDT	TSSOP14, PDIP14
AT89LP214	2.4–5.5	20	12	2K	–	128	UART, TWI, SPI	–	2x16–bit WDT	TSSOP14, PDIP14
AT89LP216	2.4–5.5	20	14	2K	–	128	UART, TWI, SPI	–	2x16–bit WDT	SOIC16, PDIP16, TSSOP16
AT89LP2052	2.4–5.5	20	15	2K	2K	256K	UART, SPI	10–bit ADC	2x16–bit PWM, WDT	PDIP20, TSSOP20, SOIC20
AT89LP4052	2.4–5.5	20	15	4K	2K	256K	UART, SPI	10–bit ADC	2x16–bit PWM, WDT	PDIP20, TSSOP20, SOIC20

Перш ніж перейти до розгляду продукції фірми Temic, коротко опишемо МК типу MCS–52 та їх основні відмінності від MCS–51. Сімейство MCS–52 являє собою вдосконалене сімейство MCS–51 з розширеними функціональними можливостями та підвищеною продуктивністю. МК типу MCS–52 програмно сумісні з сімейством MCS–51. В MCS–52 збільшено пам'ять програм на кристалі (в деяких до 64 Кбайт), введено додаткові регістри спеціальних функцій і нові режими роботи, підвищено захищеність програм від нелегального копіювання, лінії порту 1 використовуються в альтернативних режимах, розширено і виконано більш гнучкою систему переривань, розширено об’єм внутрішньої пам’яті даних. Сімейство MCS–52 навряд можна розглядати

як самостійне – для цих МК справедливо абсолютно все, що було раніше сказано для MCS–51. Але в той же час ці МК створюють «підсімейство» в сімействі MCS–51, оскільки між собою вони різняться лише наявністю чи відсутністю внутрішньої пам'яті програм і об'ємом цієї пам'яті, а їх нові (у зрівнянні з MCS–51) функціональні можливості досить великі.

Технічні дані деяких мікроконтролерів фірми Temic наведено у таблиці 1.2.

Таблиця 1.2 – Мікроконтролери фірми Atmel Wireless (Temic)

Назва	Корпуса	Пам'ять програм	Тактові частоти, МГц	Особливості
T89C51AC2	PLCC44, VQFP44, CA–BGA64	Флеш 32K	20, 40	Розширена пам'ять XRAM – 1K. EEPROM даних – 2K. Два DPTR. Режим X2 (6 тактів на машинний цикл). АЦП 8x10 біт,
T89C51RB2	DIP40, PLCC44, VQFP44,	Флеш 16K	30, 40	Розширена пам'ять XRAM – 1K. Інтерфейс SPI. Режим X2 для кожного периферійного блоку. Інтерфейс клавіатури.
T89C51RC2	DIP40, PLCC44, VQFP44,	Флеш 32K	30, 40	Розширена пам'ять XRAM – 1K. Інтерфейс SPI. Режим X2 для кожного периферійного блоку. Інтерфейс клавіатури.
T89C51RD2	DIP40, PLCC44, VQFP44, PLCC68, VQFP64	Флеш 64K	25, 40	Найбільша пам'ять програм в сімействі. Розширена пам'ять XRAM – 1K. Два DPTR.
T89C51IC2	PLCC44, VQFP44	Флеш 32K	30, 40	Розширена пам'ять XRAM – 1K. Інтерфейс SPI, i^2c. Інтерфейс клавіатури.

Всі перелічені в таблиці мікроконтролери мають можливість програмуватись внутрішньосхемно через UART. ОЗП даних в порівнянні зі стандартною архітектурою MCS–52 розширено на 1 Кбайт і містить 1280 байт. Флеш–пам'ять програм більше, ніж у кристалів Atmel. Мікроконтролери мають режим заборони видачі сигналу ALE для зменшення радіозавад. До складу периферійних пристроїв входить вартовий таймер.

1.4 Мікроконтролери фірми Siemens (Infineon)

Однією із найавторитетніших компаній на ринку мікроконтролерів є Siemens. В результаті реорганізації підрозділ по випуску напівпровідників отримав свою власну назву – Infineon. Фірма Infineon випускає, разом з іншими цікавими мікросхемами, мікроконтролери сімейства C500. Архітектура C500 базується на MCS–52, але має більш розвинуту периферію. Характеристики мікроконтролерів сімейства C500 приведено в таблиці 1.3. Специфіка мікроконтролерів Infineon визначається тим, що фірма Siemens спеціалізується на випуску апаратури промислової автоматики. 8–розрядне сімейство призначається, в першу чергу, для використання в інтелектуальних датчиках і нескладних приладах автоматики, які не потребують частого оновлення програмного забезпечення. Для таких пристроїв характерно використання в якості пам’яті програм ПЗП, що програмується масового або однократно (Mask ROM і OTP ROM). Більшість кристалів мають версії з обома типами пам’яті або без внутрішньої пам’яті програм. Велика кількість мікроконтролерів мають в своєму складі багатоканальні широтно–імульсні модулятори (ШІМ) і аналого–цифрові перетворювачі.

1.5 Мікроконтролери фірми Philips

Велику номенклатуру мікроконтролерів з ядром MCS–51 випускає фірма Philips. Продукція фірми орієнтована, в першу чергу, на ринок побутової електроніки. Більшість кристалів мають інтерфейс I^2C. I^2C – це послідовна шина, яку запропонована фірмою Philips в якості стандарту обміну в недорогих низькошвидкісних системах.

Характеристики деяких мікроконтролерів Philips наведено в таблиці 1.4. Літера x в назві мікросхеми означає, що даний кристал може поставлятися з різними типами пам'яті програм:

0 – без внутрішньої пам'яті;

3 – з масковим ПЗП (запрограмованим на замовлення);

7 – з ППЗП (EPROM);

9 – з флеш–пам'яттю.

Таблиця 1.3 – Мікроконтролери фірми Insineon (Siemens)

Назва	Корпуса	Пам’ять	ОЗП, байт	ШІМ, кан.	АЦП, кан. х біт	Такт. част., МГц	Особливості
C501G	DIP40, PLCC44, MQFP44	8K OTP	256	–	–	24	Контролер загального користування
C504	MQFP44	Немає або 16K ROM/OTP	512	6	8x10	24	Модуль керування двигуном пост. струму
C513AO	PLCC44	Немає або 16K ROM/OTP	512	–	–	16	Низьке енергоспоживання і електромагнітне випромінювання
C508	SDIP64, MQFP64	32K ROM/OTP	1280	6	8x10	20	Модуль керування двигуном пост. току, множник тактової частоти
C505L	MQFP80	32K OTP	512	4	8x10	20	Драйвер ЖКІ, годинник реального часу
C505A	MQFP44	32K OTP	1280	4	8x10	20	Низьке енергоспоживання і електромагнітне випромінювання
C505CA	MQFP44	32K ROM/OTP або 16K ROM	1280	4	8x10	20	Низьке енергоспоживання і електромагнітне випромінювання
C515	PLCC68	Немає	256	4	8x8	24	Сумісний з SAB 80C515
C515C	MQFP80	Немає або 64K ROM/OTP	2304	4	8x10	10	Низьке енергоспоживання і електромагнітне випромінювання
C517A	PLCC84	Немає	2304	21	12x8	18	Сумісний з SAB 80C517A
C509	MQFP100	Немає	3328	29	12x10	16	Завантажувач для зовнішньої флеш–пам’яті програм
C541	PLCC44	8K OTP	256	–	–	12	USB – функція

Фірма Philips, так само, як і фірма Atmel, випускає мікроконтролери в корпусах з малим числом виводів, деякі з яких також представлено в таблиці 1.4.

Таблиця 1.4 – Мікроконтролери фірми Philips

Назва	Корпус	Пам'ять програм	Тактова частота, МГц	Особливості
P8xC552 x=0,3,7	PLCC68, QFP80	8K	16 МГц, 24 МГц	АЦП 8x10 біт, I^2C
P8xC562 x=0,3	PLCC68	8K	16 МГц, 24 МГц	АЦП 8x8 біт, I^2C
P8xC557 x=0,3,9	QFP80	32K	16 МГц	ОЗУ 1К, АЦП 8x10 біт, ШІМ, I^2C, множник тактової частоти
P80C31/ P80C32	DIP40, PLCC44, QFP44	–	16 МГц, 33 МГц	ОЗП128/256 б. Режим заборони видачі ALE
P8xC51FA/FB/FC x=0,3,7	DIP40, PLCC44, QFP44	8/16/32K	16 МГц, 33 МГц	ОЗП 256 б. Додатковий таймер.
P8xC51RA+/RB+/RC+ x=0,3,7	DIP40, PLCC44, QFP44	8/16/32K	16 МГц, 33 МГц	ОЗП 512 б. Додатковий таймер, вартовий таймер.
P8xC51RD+ x=0,3,7	DIP40, PLCC44, QFP44	64K	16 МГц, 33 МГц	ОЗУ 1024 б. Додатковий таймер, вартовий таймер.
P8xC748 x=7,9	DIP24, SOIC24, PLCC28	2K	16 МГц	Зменшений корпус, ОЗП 64 б.
P8xC752 x=7,9	DIP24, SOIC24, PLCC28	2K	16 МГц	Зменшений корпус, ОЗП 64 б, АЦП 5x8 біт, i^2c.

1.6 Мікроконтролери фірми Analog Devices

Мікроконтролери фірми Analog Devices не є стандартними виробами, однак кількість проданих мікросхем та їх зростаюча популярність не дозволяють залишити без уваги продукцію цієї фірми в даному огляді. Технічні дані деяких мікроконтролерів наведено в таблиці 1.5.

Analog Devices є одним з найбільших світових виробників АЦП і ЦАП; дослідницькі центри фірми вносять величезний вклад у розвиток цієї галузі. Багато рішень від Analog Devices стали стандартами де–факто, аналоги випускаються багатьма фірмами.

Analog Devices вийшли на ринок 8–розрядних мікроконтролерів, запропонувавши сімейство MicroConverter. У цих виробах об'єднано на одному кристалі найважливіші компоненти систем збору і обробки інформації – процесорне ядро MCS–52 і блок введення–виведення аналогової інформації, що включає багатоканальні ЦАП і АЦП. Ідея такого об'єднання не нова, однак

фірмі Analog Devices вдалося розмістити на кристалі з процесором дійсно високоякісний АЦП, з роздільною здатністю 12 біт і більше, з функціями калібрування і вимірювання температури.

Таблиця 1.5 – Мікроконтролери фірми Analog Devices

Тип	ADC, каналів/біт	DAC, каналів/біт	Flash програм	EEPROM даних	RAM	Корпус	Додатково
ADuC812	8/12	2/12	8K	640b	256b	PQFP52, CSP56	5 µs ADC Conversion
ADuC814	6/12	2/12	8K	640b	256b	TSSOP28	Small, Low–Cost, Low–Power
ADuC816	2/16	1/12	8K	640b	256b	PQFP52, CSP56	Programmable Gain Input
ADuC824	1/24+1/16	1/12	8K	640b	256b	PQFP52, CSP56	Pin–Compatible Upgrade to ADuC816
ADuC831	8/12	2/12 + 2 PWM	62K	4K	256b+2K	PQFP52, CSP56	"Big Memory" Upgrade to ADuC812
ADuC832	8/12/	2/12–Bit + 2 PWM	62K	4K	256b+2K	PQFP52, CSP56	"Big Memory" Upgrade to ADuC812 plus PLL
ADuC834	1/24+1/16	Single 12–Bit + Dual PWM	62K	4K	256b+2K	PQFP52, CSP56	"Big Memory" Upgrade to ADuC824
ADuC836	2/16	1/12 + 2 PWM	62K	4K	256+2K	PQFP52, CSP56	"Big Memory" Upgrade to ADuC816
ADuC841	8/12	2/12 + 2 PWM	62K	4K	256+2K	PQFP52, CSP56	"Big Memory" Upgrade to ADuC812 Fast 8052 Core
ADuC842	8/12	2/12 + 2 PWM	62K	4K	256b+2K	PQFP52, CSP56	"Big Memory" Upgrade to ADuC812 plus PLL Fast 8052 Core
ADuC844	1/24+1/16	Single 12–Bit + Dual PWM	62K	4K	256b+2K	PQFP52, CSP56	"Big Memory" Upgrade to ADuC824 Fast 8052 Core
ADuC845	10/24+1/24	Single 12–Bit + Dual 16–bit + Dual PWM 16bit	62K	4K	256b+2K	PQFP52, CSP56	"Big Memory" Upgrade to ADuC824 Fast 8052 Core
ADuC846	2/16	1/12 + 2 PWM	62K	4K	256b+2K	PQFP52, CSP56	"Big Memory" Upgrade to ADuC816 Fast 8052 Core
ADuC847	10/24	Dual 16–bit + Dual PWM 16bit	62K	4K	256b+2K	PQFP52, CSP56	"Big Memory" Upgrade to ADuC816 Fast 8052 Core

1.7 Мікроконтролери фірми Dallas Semiconductor

Особливістю мікроконтролерів фірми Dallas Semiconductor (таблиця 1.6) є змінена часова сітка. Машинний цикл в даних мікроконтролерах займає 4 такти

замість 12 у класичного 80C51. При цьому збережено повну сумісність зі стандартною архітектурою на рівні інструкцій. Мікроконтролери Dallas знаходять застосування в завданнях, які потребують підвищеної швидкості обробки. Характеристики мікроконтролерів фірми наведено в таблиці 1.6. Багато мікроконтролерів мають розширений ОЗП, звернення до якого відбувається так само, як до зовнішньої пам'яті. У регістрову модель мікропроцесора включений другий покажчик зовнішньої пам'яті DPTR. Характерною особливістю периферії мікроконтролерів Dallas є наявність на одному кристалі двох послідовних портів.

Таблиця 1.6 – Мікроконтролери фірми Dallas

Назва	Корпуса	Пам’ять програм	Додатковий ОЗП	Кількість послідовних портів	Тактова частота	Особливості
DS80C310	DIP40, PLCC44, TQFP44	–	–	1	33 МГц	
DS80C320/323	DIP40, PLCC44, TQFP44	–	–	2	33 МГц, 18 МГц	
DS80C390	PLCC68, TQFP64	До 4K ОЗП	до 4K	2	40 МГц	Інтерфейс CAN 2.0B, математичний співпроцесор, перерозподіл ОЗП між областями даних і команд
DS80CH11	TQFP128	–	256b	1	33 МГц	ШІМ, АЦП 8x10 біт
DS8xC520	DIP40, PLCC44, TQFP44	16K ROM або EPROM	1K	2	33 МГц	
DS8xC530	PLCC52, TQFP52	16K ROM або EPROM	1K	2	33 МГц	Годинник реального часу

1.8 Склад та основні характеристики типового мікроконтролера фірми Analog Devices ADuC847

ADuC847 є покращеною модифікацією мікроконтролера ADuC834 і функціонально близький до мікроконтролера ADuC845. ADuC847 має поліпшену (12.58MIPs) версію ядра МК 8052. У порівнянні з ADuC845 ADuC847 містить більшу кількість каналів аналогового введення, але в ньому

відсутні ЦАП і додатковий АЦП. Виріб має всі функції ADuC834, але стандартне 12–циклове ядро замінено одноцикловим з продуктивністю 12.58MIPs.

ADuC847 є закінченим контролером для інтелектуальних датчиків, що включає в себе сигма–дельта АЦП високого розширення, гнучкий вхідний мультиплексор на 10/8 каналів, швидкий 8–розрядний мікроконтролер і вбудовану Flash/EEPROM – пам'ять програм і даних.

До складу АЦП включено вхідний мультиплексор, датчик температури і підсилювач з програмованим коефіцієнтом передачі (PGA), який дозволяє працювати з сигналами низького рівня, які знімаються безпосередньо з датчиків. АЦП з вбудованим фільтром і програмованим вихідним потоком даних призначено для вимірювання низькочастотних сигналів у широкому динамічному діапазоні напруг таких, як сигнали з зважувальних пристроїв, сигнали з датчиків деформації, з датчиків тиску або температури.

Пристрій працює від кварцового резонатора 32кГц, а висока частота 12.58МГц виробляється системою фазового автопідстроювання частоти (ФАПЧ). Висока частота проходить через програмно–керований дільник, з якого знімається частота для роботи мікропроцесорного ядра. Мікропроцесорне ядро є оптимізованим одноцикловим ядром 8052, що дає продуктивність до 12.58MIPS при виконанні команд, які сумісні з МК 8051.

Пристрій містить 62Кбайт внутрішньої Flash–пам'яті програм, 4Кбайт внутрішньої EEPROM–пам'яті даних і 2304 байт внутрішньої пам'яті даних з довільним доступом (RAM).

«Зашите» на етапі виробництва програмне забезпечення дозволяє робити завантаження програм в пристрій через послідовний порт (UART), а також виконувати налагодження прикладних програм системи через єдиний зовнішній вхід ЕА. Розробка виробів на основі ADuC847 підтримується недорогою системою QuickStart TM (апаратура і програми).

У класичному МК 8052 всі таймери інкрементуються на +1 під час кожного машинного циклу. Оскільки в ADuC847 один машинний цикл дорівнює одному періоду тактової частоти, таймери будуть інкрементуватися з частотою, яка, дорівнює тактовій частоті ядра.

Частота вихідного сигналу ALE мікроконтролера ADuC834 становить 1/6 тактової частоти ядра. У ядрі ADuC847 сигнал ALE виглядає наступним чином.

У разі одноциклової команди вихід ALE знаходиться у високому логічному стані протягом першої половини машинного циклу і в низькому – протягом другої половини. Тут частота на виході ALE дорівнює тактовій частоті ядра. Для команд, що містять два або більше циклів, вихід ALE знаходиться у високому логічному стані протягом першої половини машинного циклу і в низькому – протягом другої половини.

ADuC847 не підтримує доступ до зовнішньої пам'яті програм. При зверненні до зовнішньої пам'яті даних з довільним доступом буде потрібно програмувати регістр EWAIT, щоб при виконанні команди MOVX були виконані додаткові машинні цикли. Це необхідно виконати через відмінності у часі доступу до внутрішньої і зовнішньої RAM.

Більш детальну інформацію про МК–р ADuC847 наведено у [13].

1.9 Склад та основні характеристики мікроконтролерів фірми Silicon Laboratories (Cygnal)

Фірма Silicon Laboratories динамічно розвиває перелік випускаємих мікроконтролерів з ядром МК51. Фірмі вдалося створити велику кількість типів мікроконтролерів, що відрізняються продуктивністю, об'ємом Flash–пам'яті програм, вбудованої оперативної пам'яті, характеристиками аналого–цифрових вузлів, типом корпусу та іншими параметрами. Частину мікроконтролерів, що випускаються, наведено в таблиці 1.7. Вони умовно розділені на 11 сімейств.

Це дозволяє використовувати при розробці програмного забезпечення стандартні x51 орієнтовані асемблери і компілятори мов високого рівня (С–51,

PL/M–51, FORTRAN–51, PASCAL–51 і т. ін.), а також бібліотеки підпрограм, які створено різними колективами за десятиліття існування x51–сумісних мікроконтролерів. Мікроконтролерне ядро (CIP–51) містить повний набір периферійних вузлів, який є стандартним для сучасних мікроконтролерів. Залежно від типу сімейства, ядро може містити 3, 4 або 5 таймерів–лічильників, один або два послідовних порти UART, як мінімум 256 байт вбудованої оперативної пам'яті, 128–байт регістрів спеціальних функцій SFR (Special Function Register). Ядро також забезпечує керування лініями портів введення/виведення, яких мікроконтролери різних сімейств можуть мати від 1 до 8 (тобто від 8 до 64 ліній введення/виведення). Також, перевагою цього ядра є вбудована апаратура налагодження. Ядро забезпечує безпосереднє керування аналоговими і цифровими підсистемами мікроконтролера через відповідні регістри SFR. Таким чином, ядро CIP–51 з одного боку забезпечує повну сумісність зі стандартним x51 сумісним ядром, з іншого, має значно ширші апаратні можливості за рахунок поповнення вбудованою цифровою і аналоговою периферією. Структурну схему ядра CIP–51 показано на рисунку 1.1.

Безсумнівно, найважливішою перевагою ядра CIP–51 є вдосконалена конвеєрна архітектура, яка дозволяє значно збільшити продуктивність у порівнянні зі стандартною x51 (8051) архітектурою. У мікроконтролерах зі стандартною архітектурою 8051 всі інструкції, за винятком MUL і DIV, виконувалися за 12 або 24 машинних тактів. При цьому максимальна тактова частота для більшості x51 сумісних мікроконтролерів становила 12–24 МГц, і лише деякі мікроконтролери могли працювати на вищих частотах.

Таблиця 1.7 – Мікроконтролери фірми Silicon Laboratories

Сімейство	Тип	Основні параметри					
		Продуктив–ність MIPS	Flash ROM, K	RAM, byte	ADC, bit	Корпус (кількість виводів)	Примітка
C8051F0xx	C8051F000	20	32	256	12	64	
	C8051F001	20	32	256	12	48	
	C8051F002	20	32	256	12	32	
	C8051F005	25	32	2.25K	12	64	
	C8051F006	25	32	2.25K	12	48	
	C8051F007	25	32	2.25K	12	32	
	C8051F010	20	32	256	10	64	
	C8051F011	20	32	256	10	48	
	C8051F012	20	32	256	10	32	
	C8051F015	25	32	2.25K	10	64	
	C8051F016	25	32	2.25K	10	48	
	C8051F017	25	32	2.25K	10	32	
C8051F018/9	C8051F018	25	16	1.25K	10	48	
	C8051F019	25	16	1.25K	10	32	
C8051F02x	C8051F020	25	64	4.25K	12+8	100	
	C8051F021	25	64	4.25K	12+8	64	
	C8051F022	25	64	4.25K	10+8	100	
	C8051F023	25	64	4.25K	10+8	64	
C8051F04x	C8051F040	25	64	4.352K	12+8	100	CAN 2.0B
	C8051F041	25	64	4.352K	12+8	64	CAN 2.0B
	C8051F042	25	64	4.352K	10+8	100	CAN 2.0B
	C8051F043	25	64	4.352K	10+8	64	CAN 2.0B
	C8051F044	25	64	4.352K	10+8	100	CAN 2.0B
	C8051F045	25	64	4.352K	10+8	64	CAN 2.0B
	C8051F046	25	32	4.352K	10+8	100	CAN 2.0B
	C8051F047	25	32	4.352K	10+8	64	CAN 2.0B
C8051F06x	C8051F060	25	64	4.352K	16+10	100	CAN+DMA
	C8051F061	25	64	4.352K	16+10	64	CAN+DMA
	C8051F062	25	64	4.352K	16+10	100	CAN+DMA
	C8051F063	25	64	4.352K	16+10	64	CAN+DMA
C8051F12x	C8051F120	100	128	8.25K	12+8	100	
	C8051F121	100	128	8.25K	12+8	64	
	C8051F122	100	128	8.25K	10+8	100	
	C8051F123	100	128	8.25K	10+8	64	
	C8051F124	50	128	8.25K	12+8	100	
	C8051F125	50	128	8.25K	12+8	64	
	C8051F126	50	128	8.25K	10+8	100	
	C8051F127	50	128	8.25K	10+8	64	
C8051F13x	C8051F130	100	128	8.25K	10	100	
	C8051F131	100	128	8.25K	10	64	
	C8051F132	100	64	8.25K	10	100	
	C8051F133	100	64	8.25K	10	64	

Продовження таблиці 1.7

Сімейство	Тип	Основні параметри					
		Продуктив–ність MIPS	Flash ROM, K	RAM, byte	ADC, bit	Корпус (кількість виводів)	Примітка
C8051F2xx	C8051F206	25	8	1.25K	12	48	
	C8051F220	25	8	256	8	48	
	C8051F221	25	8	256	8	32	
	C8051F226	25	8	1.25K	8	48	
	C8051F230	25	8	256	–	48	
	C8051F231	25	8	256	–	32	
	C8051F236	25	8	1.25K	–	48	
C8051F30x	C8051F300	25	8	256	8	11	2% Osc
	C8051F300P	25	8	256	8	DIP14	2% Osc
	C8051F301	25	8	256	–	11	2% Osc
	C8051F302	25	8	256	8	11	
	C8051F303	25	8	256	–	11	
	C8051F304	25	4	256	–	11	
	C8051F305	25	3	256	–	11	
C8051F31x	C8051F310	25	16	1.28K	10	32	2% Osc
	C8051F311	25	16	1.28K	10	28	2% Osc
C8051F32x	C8051F320	25	16	2.3K	10	32	Osc + USB
	C8051F321	25	16	2.3K	10	28	Osc + USB
C8051F33x	C8051F330	25	8	768	10	20	Osc
	C8051F330P	25	8	768	10	DIP14	Osc
	C8051F331	25	8	768	–	20	Osc

Модернізоване ядро CIP–51 виконує 70% інструкцій за один або два машинних такти, і взагалі не має інструкцій, що виконуються більш ніж за вісім машинних тактів. При цьому, практично всі мікроконтролери фірми Silicon Laboratories (27 типів, 72%) можуть працювати при частоті тактового генератора 25МГц, шість мікроконтролерів працюють на частотах до 20МГц (16%), і чотири нових мікроконтролера сьомого сімейства (C8051F12x) функціонують на частотах до 100МГц. При цьому, відповідно розвивається пікова (гранична) продуктивність 25, 20 і 100 мільйонів інструкцій у секунду – MIPS (Million Instructions Per Second).

Як приклад на рисунку 1.2 показано співвідношення граничної продуктивності деяких поширених мікроконтролерів в порівнянні з граничною продуктивністю мікроконтролерів фірми Silicon Laboratories при тактовій частоті 25МГц.

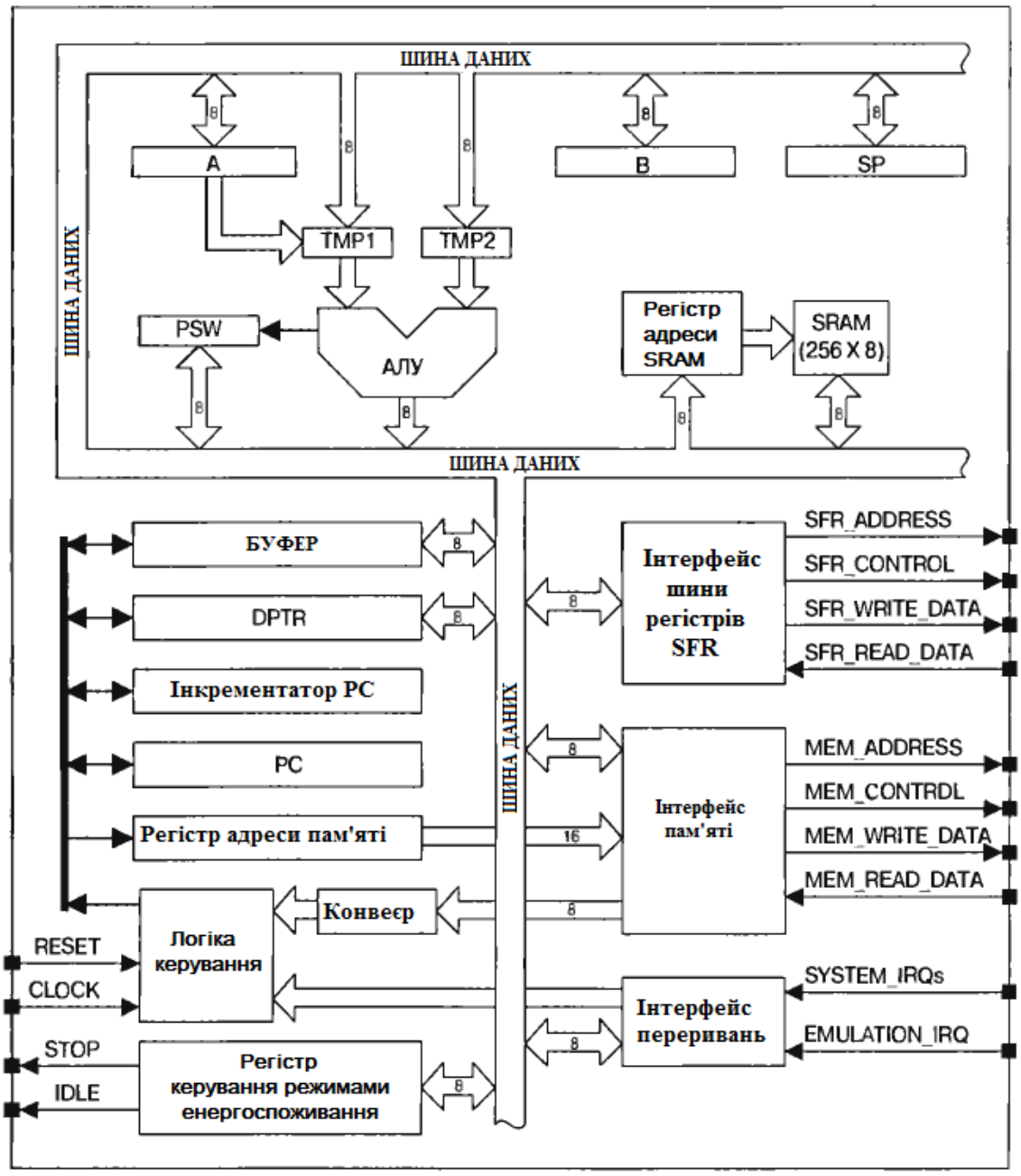


Рисунок 1.1 – Структура ядра CIP–51

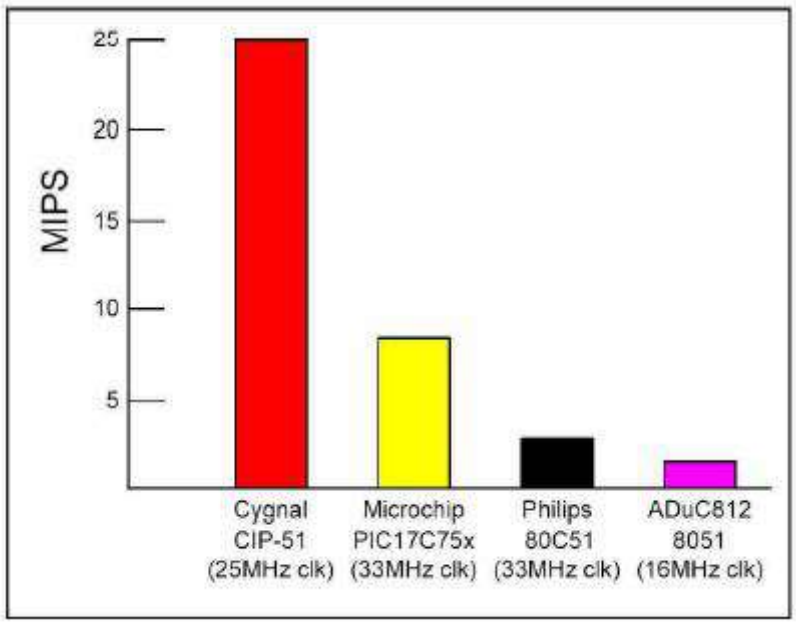


Рисунок 1.2 – Співвідношення граничної продуктивності деяких мікроконтролерів

Ще однією важливою перевагою мікропроцесорного ядра CIP–51 є наявність вбудованої апаратної підсистеми налагодження програмного забезпечення. Зв'язок з підсистемами мікроконтролера здійснюється через послідовний інтерфейс JTAG, який відповідає стандарту IEEE 1149.1. При цьому, забезпечуються як режим внутрішньосистемного програмування – ISP (In System Programarable), так і власне режим відлагодження. При програмуванні можливий запис, як усього масиву програми, так і модифікація окремих байтів. Природно, що вміст пам'яті програм може також читатися і звірятися з оригіналом. Вміст будь–якого байта пам`яті програм може читатися або змінюватися з використанням інструкцій MOVC або MOVX, що також дозволяє здійснювати незалежне зберігання даних і оперативно їх модифікувати під керування програми.

2 СТРУКТУРА ТИПОВОГО МІКРОКОНТРОЛЕРА МК51

2.1 Загальні відомості

Один з типових МК, наприклад AT89C51, складається з таких основних функціональних вузлів (рисунок 2.1): блока керування і синхронізації; блока арифметико–логічного пристрою АЛП (англ. Arifmetic–Logic Unit); резидентної пам'яті даних РПД (англ. Resident Data Memory) об'ємом 128 байт; резидентної пам'яті програм РПП (англ. Resident Program Memory) об'ємом 4 Кбайт; блока переривань (англ. Interrupt System); таймерів (англ. Timer); послідовного порту (англ. Serial Port); чотирьох паралельних портів введення/виведення (англ. Parallel Port), які програмуються; схеми десяткової корекції вмісту акумулятора (СДКА); внутрішнього генератора тактових імпульсів ГТІ (англ. Oscillator); резидентної шини даних РШД (англ. Resident Data Bus) і групи регістрів:

- *А* – акумулятор;
- *В* – регістр розширення акумулятора;
- *Т1, Т2* – регістри тимчасового збереження операндів;
- *РСП (PSW)* – регістр стану програми (прапорців);
- *РК (IR)* – регістр команд;
- *ЛК (PC)* – лічильник команд (програмний лічильник);
- *РПД (DPTR)* – регістр–покажчик даних, що складається з 2–х частин: молодшої – DPL і старшої – DPH;
- *РПС (SP)* – регістр–покажчик стеку;
- *РА (RAR)* – регістр адреси;
- *РРТЛ (TMOD)* – регістр режимів таймерів/лічильників;
- *РКСТ (TCON)* – регістр керування–статусу таймерів/лічильників;
- *РКПП (SCON)* – регістр керування приймачем–передавачем послідовного порту;

– *SBUF (буфер ПРМ* і *буфер ПД)* – буфери приймача і передавача послідовного порту;

– *РМП (IE)* – регістр масок переривань;

– *РП (IP)* – регістр пріоритетів переривань;

– *РКП (PCON)* – регістр керування потужністю споживання енергії від джерела живлення.

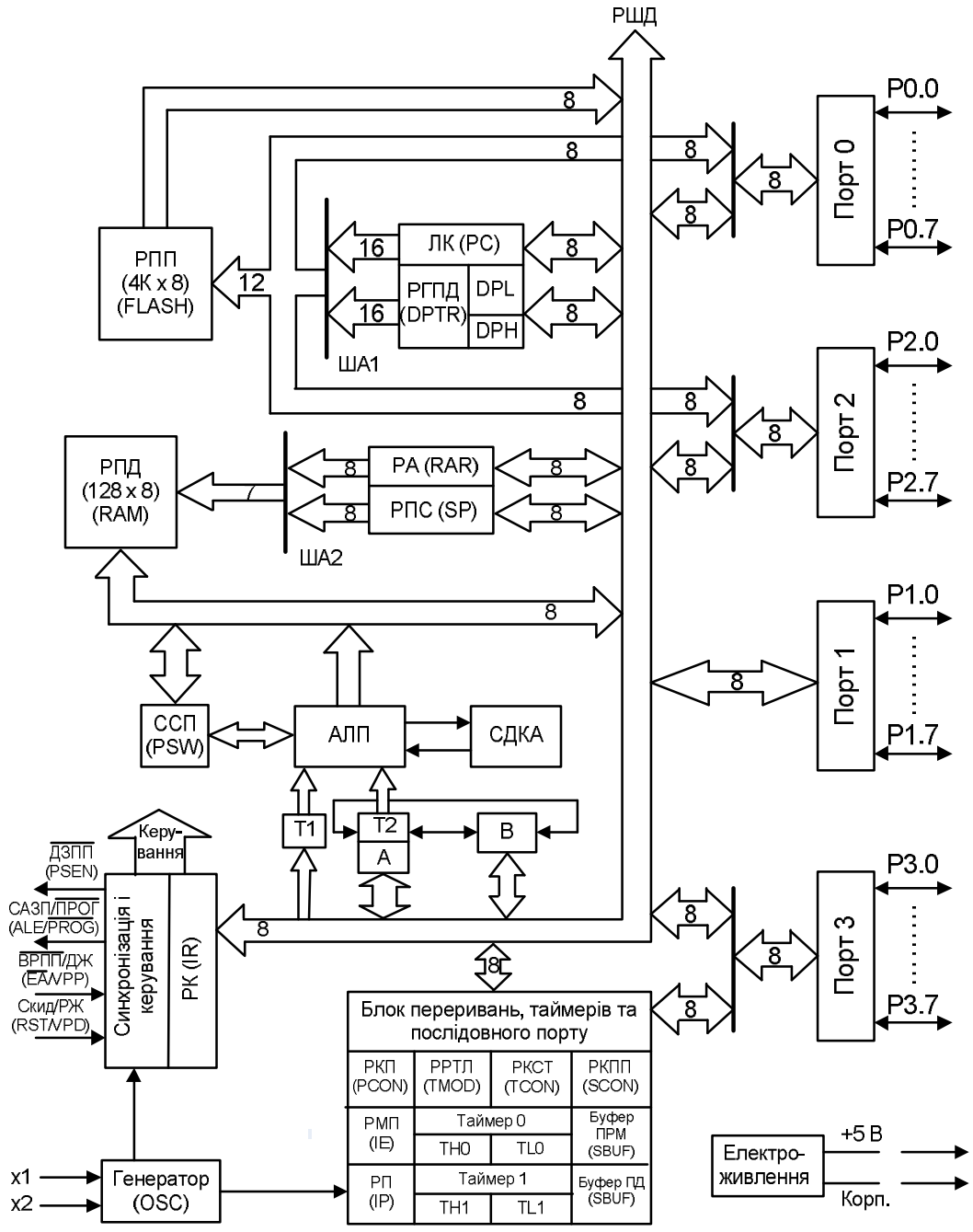


Рисунок 2.1 – Структурна схема МК AT89C51

2.2 Блок керування та синхронізації мікроконтролера

Блок керування та синхронізації призначено для формування синхронізуючих і керуючих сигналів, що забезпечують координацію спільної роботи блоків МК у всіх допустимих режимах їх роботи.

До складу блока керування входять: пристрій формування часових інтервалів, логіка введення/виведення, регістр команд, дешифратор команд, програмована логічна матриця (ПЛМ) і логіка керування мікроконтролером.

Пристрій формування часових інтервалів призначено для формування і видачі внутрішніх синхросигналів станів, фаз і циклів (рисунок 2.2).

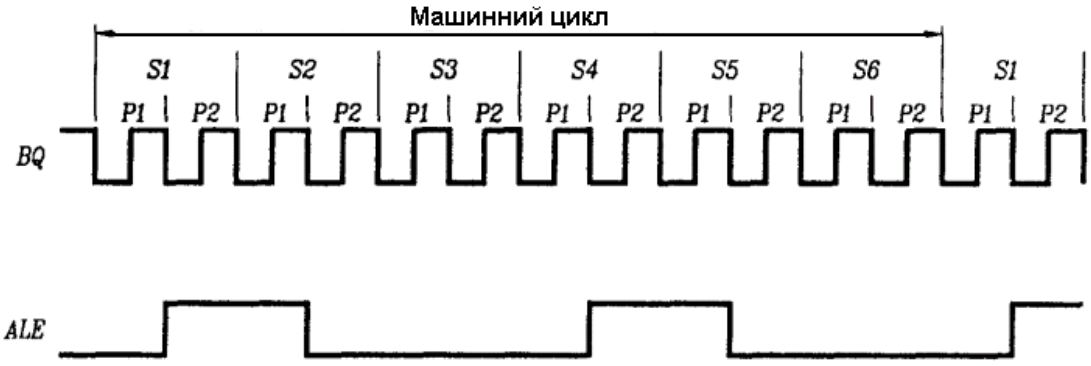


Рисунок 2.2 – Діаграма формування машинних циклів МК

Рисунок 2.2 ілюструє формування машинних циклів в МК. Всі машинні цикли однакові, складаються з 12 періодів тактового сигналу BQ, починаються фазою S1 P1 і закінчуються фазою S6 P2. Двічі за один машинний цикл формується сигнал дозволу фіксації адреси ALE (англ. Address Latch Enable), що видається на однойменний вивід мікроконтролера. Якщо, наприклад, зовнішня частота f_{BQ} = 12 МГц, то тривалість машинного циклу $T_{МЦ}$ = 1 мкс.

Кількість машинних циклів (англ. machine cycle) визначає тривалість виконання команд. Практично всі команди МК виконуються за один або два машинних цикли, крім команд множення MUL A, B і ділення DIV A, B, тривалість виконання яких складає чотири машинних цикли. Машинний цикл

має фіксовану тривалість і містить шість станів (англ. state) S1…S6, кожен з яких складається з двох часових інтервалів, які визначаються фазами (англ. phase) P1 і P2. Тривалість фази дорівнює періоду зовнішнього сигналу BQ, що є первинним сигналом синхронізації МК. Сигнал BQ формується або вмонтованим тактовим генератором МК (при підключенні до її виводів 18 (BQ2) і 19 (BQ1) кварцового резонатора або LC–ланцюжка), або зовнішнім джерелом тактових сигналів.

Схему підключення кварцового резонатора до виводів BQ2 і BQ1 показано на рисунку 2.3.

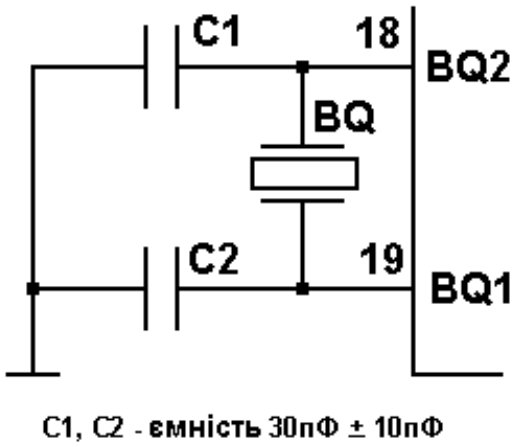


Рисунок 2.3 – Схема підключення кварцового резонатора

В *регістр команд (РК)* з пам'яті програм пересилається код операції чергової команди, що виконується. Дешифратор команд декодує код операції та ідентифікує тип команди, що підлягає виконанню.

Після цього з програмованої логічної матриці (ПЛМ) викликається послідовність керуючих сигналів для виконання команди.

2.3 Блок арифметико–логічного пристрою

Блок арифметико–логічного пристрою (АЛП) являє собою паралельний 8–розрядний пристрій, що забезпечує виконання арифметичних і логічних операцій, а також операцій логічного зсуву, скидання, встановлення і т.ін.

Блок АЛП складається з регістрів тимчасового збереження операндів Т1, Т2, ПЗП констант, суматора, додаткового регістра (регістра В), акумулятора, регістра стану програми.

Регістри тимчасового збереження операндів Т1, Т2 – 8–розрядні, призначені для прийому і збереження операндів на час виконання операцій над ними, програмно не доступні.

ПЗП констант забезпечує створення коду коригування при двійково–десятковому представленні даних, коду маски при бітових операціях і коду констант.

Паралельний 8–розрядний суматор являє собою схему комбінаційного типу з послідовним перенесенням, яку призначено для виконання арифметичних операцій додавання, віднімання і логічних операцій додавання, множення, порівняння тощо.

Регістр В – 8–розрядний регістр, який використовується під час операцій множення і ділення. Для інших інструкцій він може розглядатися як додатковий над оперативний запам`ятовуючий пристрій, об`ємом 1 байт.

Акумулятор являє собою 8–розрядний регістр, призначений для прийому і збереження результату, отриманого при виконанні арифметично–логічних операцій або операцій пересилання.

Регістр стану програми (PSW) призначений для збереження інформації про стан АЛП при виконанні програми. Позначення розрядів регістра PSW і призначення його розрядів наведено відповідно в таблицях 2.1 і 2.2.

Таблиця 2.1 – Позначення розрядів регістра PSW

Біти	7	6	5	4	3	2	1	0
Позначення	CY	AC	F0	RS1	RS0	OV	–	P

Таблиця 2.2 – Призначення окремих розрядів регістра PSW

<table>
<tr><th>Біти</th><th colspan="2">Найменування</th><th>Призначення бітів</th><th>Доступ до біта</th></tr>
<tr><td>7</td><td colspan="2">CY</td><td>Прапорець перенесення. Змінюється під час виконання деяких арифметичних і логічних інструкцій</td><td>апаратно або програмно</td></tr>
<tr><td>6</td><td colspan="2">AC</td><td>Прапорець додаткового перенесення. Апаратно встановлюється/скидається під час виконання інструкцій додавання або віднімання для показу перенесення або позики в біті 3 при утворенні молодшої тетради результату (D0…D3)</td><td>апаратно або програмно</td></tr>
<tr><td>5</td><td colspan="2">F0</td><td>Прапорець 0. Прапорець, який визначається користувачем</td><td>програмно</td></tr>
<tr><td>4</td><td colspan="2">RS1</td><td>Покажчик банку робочих регістрів РПД</td><td>програмно</td></tr>
<tr><td>3</td><td colspan="2">RS0</td><td>Покажчик банку робочих регістрів РПД</td><td>програмно</td></tr>
<tr><td></td><td>RS1</td><td>RS0</td><td rowspan="5"></td><td rowspan="5"></td></tr>
<tr><td></td><td>0</td><td>0</td></tr>
<tr><td></td><td>0</td><td>1</td></tr>
<tr><td></td><td>1</td><td>0</td></tr>
<tr><td></td><td>1</td><td>1</td></tr>
<tr><td colspan="3"></td><td>Банк 0 з адресами: 00H…07H
Банк 1 з адресами: 08H…0FH
Банк 2 з адресами: 10H…17H
Банк 3 з адресами: 18H…1FH</td><td></td></tr>
</table>

Продовження таблиці 2.2

2	OV	*Прапорець переповнення*. Апаратно встановлюється/скидається під час виконання арифметичних інструкцій для показу стану переповнення розрядної сітки чисел зі знаком	апаратно або програмно
1	–	*Резервний*. Містить тригер, доступний для запису ("0" і "1") і читання, який можна використовувати	програмно
0	P	*Біт парності*. Апаратно скидається/ встановлюється в кожному циклі інструкцій для показу парної/непарної кількості розрядів акумулятора, що знаходяться в стані "1"	апаратно або програмно

2.4 Резидентна пам'ять даних

Пам'ять даних ділиться на резидентну (внутрішню) – РПД, і зовнішню – ЗПД. Резидентна пам'ять даних РПД призначена для прийому, збереження і видачі інформації, яка використовується в процесі виконання програми. До складу РПД входить ОЗП (SRAM) ємністю 128 байт (рисунок 2.4) і дешифратор адреси. Керують роботою РПД два регістри (рисунок 2.1): РА (RAR) – регістр адреси; РПС (SP) – покажчик стеку.

Регістр адреси ОЗП (РА) призначено для прийому і збереження адреси комірки пам'яті, яка обирається за допомогою дешифратора і може містити як біт, так і байт інформації.

ОЗП являє собою 128 8–розрядних регістрів, які призначено для прийому, збереження і видачі цифрової інформації. 16 із цих регістрів допускають побітову адресацію.

На рисунку 2.4 наведено розподіл адресного простору РПД і область бітів, що адресуються прямо.

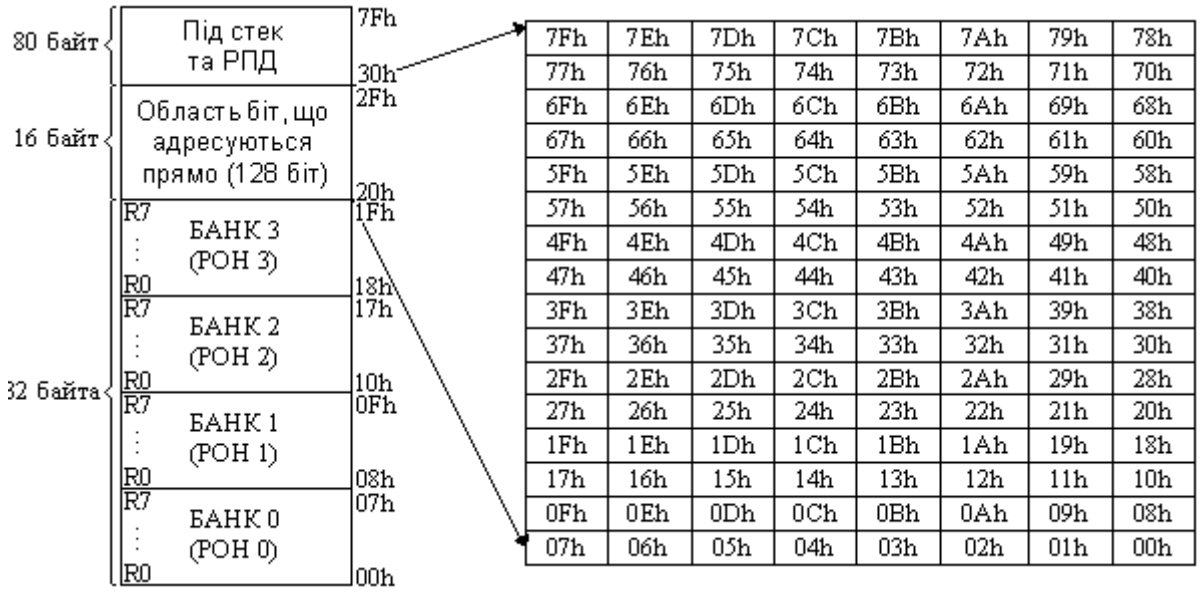


Рисунок 2.4 – Розподіл адресного простору РПД і область бітів, які адресуються прямо

Покажчик стеку являє собою 8–розрядний регістр, який призначено для прийому і збереження адреси комірки стеку. При виконанні команд LCALL, ACALL вміст покажчика стеку збільшується на 2. При виконанні команд RET, RETI вміст покажчика стеку зменшується на 2. При виконанні команди PUSH direct вміст покажчика стеку збільшується на 1. При виконанні команди POP direct вміст покажчика стеку зменшується на 1. Після скидання в покажчику стеку встановлюється адреса 07H, що відповідає початку стеку з адресою 08H.

Більш докладно організацію пам'яті даних мікропроцесорних систем, що використовуються у цьому МК, розглянуто в окремому розділі.

2.5 Резидентна пам'ять програм

Резидентну пам'ять програм РПП призначено для збереження програм і має окремий від пам'яті даних адресний простір об'ємом до 64 Кбайт, причому, наприклад, для мікросхеми AT89C51 частина пам'яті програм з адресами 0000H…0FFFH розташована на кристалі МК. Пам'ять програм, що розташована на кристалі (РПП), складається з 12–розрядного дешифратора та FLASH–ПЗП ємністю 4Кх8 біт. Запис програм у ПЗП відбувається під час розробки системи.

Якщо на вивід МК «відключення резидентної пам`яті програм» (ВРПП, DEMA) подається напруга живлення U_{CC} (логічна 1), то звернення до зовнішньої пам'яті програм відбувається автоматично при видачі лічильником команд адреси, що перевищує 0FFFH. Якщо адреса знаходиться в межах 0000H…0FFFH, звернення відбувається до пам'яті програм, яка розташована на кристалі (резидентної пам'яті програм).

Якщо на вивід «ВРПП» МК подається сигнал "логічний 0", то внутрішня пам'ять програм відключається, і, починаючи з адреси 0000H, всі звернення виконуються до зовнішньої пам'яті програм.

Для формування поточної 16–розрядної адреси пам'яті програм служить лічильник команд (програмний лічильник) – ЛК (PC). 12 молодших розрядів цього регістра використовуються при адресації комірок РПП об'ємом 2^{12} = 4 Кбайт.

Більш докладно організацію пам'яті програм мікропроцесорних систем, в яких застосовується даний МК, розглянуто в окремому розділі.

2.6 Блок переривань

МК, наприклад AT89C51, має систему переривань із п'ятьма векторами (адресами підпрограм обробки переривань) і двома рівнями пріоритетів. Джерелами переривань виступають: два зовнішніх переривання, що надходять через порт 3; два переривання від переповнення таймерів/лічильників T/CNT0 і

T/CNT1; переривання за завершенням передачі або прийому даних при обміні через послідовний порт.

Для програмування і керування роботою підсистеми переривань використовують два регістри: РП (IP) – регістр пріоритетів переривань та РМП (IE) – регістр масок переривань, а також чотири молодших біти регістра керування/статусу таймерів/лічильників (РКСТ) (таблиці 2.3, 2.4 та 2.8).

Таблиця 2.3 – Позначення розрядів регістра IP

Біти	7	6	5	4	3	2	1	0
Позначення	X	X	X	PS	PT1	PX1	PT0	PX0

Регістр пріоритетів переривань (IP) призначено для встановлення рівня пріоритету переривання для кожного з п'яти джерел переривань. Позначення розрядів регістра IP показано в таблиці 2.3, а їхнє призначення описано нижче.

PX0 (IP0) – встановлення рівня пріоритету переривання від зовнішнього джерела $\overline{\text{INT0}}$.

PT0 (IP1) – встановлення рівня пріоритету переривання від таймера/лічильника T/C0.

PX1 (IP2) – встановлення рівня пріоритету переривання від зовнішнього джерела $\overline{\text{INT1}}$.

PT1 (IP3) – встановлення рівня пріоритету переривання від таймера/лічильника T/C1.

PS (IP4) – встановлення рівня пріоритету переривання від послідовного порту.

X (IP7, IP6, IP5) – резервний розряд.

Наявність у розряді IP сигналу "логічна 1" встановлює для відповідного джерела високий рівень пріоритету, а наявність у розряді IP сигналу "логічний 0" – низький рівень пріоритету. При читанні резервних розрядів

відповідні лінії шини даних не визначені. Користувач не повинен записувати "1" у резервні розряди, тому що вони зарезервовані для подальшого розширення сімейства МК–51.

Регістр дозволу переривань (IE) призначено для дозволу або заборони переривань від відповідних джерел. Позначення розрядів регістра IE показано в таблиці 2.4, а їхнє призначення описано нижче.

Таблиця 2.4 – Позначення розрядів регістра IE

Біти	7	6	5	4	3	2	1	0
Позначення	EA	X	X	ES	ET1	EX1	ET0	EX0

EA (IE7) – керування всіма джерелами переривань одночасно. Якщо EA = 0, то переривання заборонено. Якщо EA = 1, то використовуються прапорці індивідуального дозволу переривань: EX0, ET0, EX1, ET1, ES.

X (IE6, IE5) – резервний розряд.

ES (IE4) – прапорець керування перериванням від послідовного порту: ES = 1 – дозвіл, ES = 0 – заборона.

ET1 (IE3) – прапорець керування перериванням від таймера/лічильника T/C1: ET1 = 1 – дозвіл, ET1 = 0 – заборона.

EX1 (IE2) – керування перериванням від зовнішнього джерела $\overline{\text{INT1}}$: EX1 = 1 – дозвіл, EX1 = 0 – заборона.

ET0 (IE1) – керування перериванням від таймера/лічильника T/C0: ET0 = 1 – дозвіл, ET0 = 0 – заборона.

EX0 (IE0) – керування перериванням від зовнішнього джерела $\overline{\text{INT0}}$: EX0 = 1 – дозвіл, EX0 = 0 – заборона.

При зчитуванні резервних розрядів значення сигналів на шині даних не визначено. Користувач не повинен записувати сигнал "логічна 1" у резервні

розряди, тому що вони зарезервовані для подальшого розширення сімейства МК–51.

Блок переривань містить також *схему логічної обробки переривань*, яка здійснює пріоритетний вибір запиту переривання, скидання його прапорця й ініціює формування апаратно реалізованої команди переходу на підпрограму обслуговування переривання LCALL.

Схема формування вектора переривання формує двобайтові адреси підпрограм обслуговування переривання в залежності від джерела переривання, які наведено в таблиці 2.5.

Таблиця 2.5 – Джерела переривань і адреси підпрограм, що їх обслуговують

Джерело переривання	Вектор переривання
Зовнішнє переривання $\overline{\text{INT0}}$	0003H
Таймер/лічильник Т/С0	000BH
Зовнішнє переривання $\overline{\text{INT1}}$	0013H
Таймер/лічильник Т/С1	001BH
Послідовний порт	0023H

Більш докладно підсистему переривань розглянуто в окремому розділі.

2.7 Блок таймерів/лічильників

Таймери/лічильники (Т/Л) призначено для підрахунку зовнішніх подій, для отримання програмно керованих часових затримок і виконання функцій мікроконтролера, які задають часові інтервали.

До складу блоку Т/Л входять:

- два 16–розрядних регістри Т/Л0 (Т/С0) і Т/Л1 (Т/С1);
- 8–розрядний регістр режимів Т/Л (TMOD);
- 8–розрядний регістр керування (TCON);

– схема інкременту;

– схема фіксації $\overline{\text{INT0}}$, $\overline{\text{INT1}}$, T0, T1;

– схема керування прапорцями;

– логіка керування Т/Л.

Два 16–розрядних регістри T/C0 і *T/C1* виконують функцію зберігання вмісту лічби. Кожен із них складається з пари восьмирозрядних регістрів, відповідно TH0, TL0 і TH1, TL1. Причому регістри TH0, TH1 – старші, а регістри TL0, TL1 – молодші 8 розрядів. Кожен із восьмирозрядних регістрів має свою адресу і може бути використаний як РЗП, якщо Т/Л не використовуються (біт TR0 для Т/Л0 і біт TR1 для Т/Л1 у регістрі керування TCON дорівнюють "0", таблиця 2.8).

Стартове значення регістрів Т/Л задається програмно. В процесі лічби вміст регістрів Т/Л інкрементується. Ознакою закінчення лічби, як правило, виступає переповнення регістра Т/Л, тобто перехід його вмісту зі стану "всі одиниці" у стан "усі нулі". Всі регістри TH0, TH1, TL0, TL1 доступні для читання, і, при необхідності, контроль досягнення необхідного значення може виконуватися програмно.

Регістр режимів Т/Л (TMOD) призначено для прийому і збереження коду, що визначає:

– один із 4–х можливих режимів роботи кожного Т/Л;

– роботу в якості таймерів або лічильників;

– керування Т/Л від зовнішнього виводу.

Позначення розрядів регістра TMOD наведено в таблиці 2.6, а призначення розрядів – у таблиці 2.7.

При роботі в якості таймера вміст регістра Т/Л інкрементується в кожному машинному циклі, тобто Т/Л являється лічильником машинних циклів МК. Оскільки машинний цикл складається з 12 періодів частоти синхронізації МК: f_{BQ}, то частота лічби в даному випадку дорівнює $f_{BQ}/12$.

Таблиця 2.6 – Позначення розрядів регістра TMOD

Біти	7	6	5	4	3	2	1	0
Позначення	GATE1	$C/\overline{T}1$	M1.1	M0.1	GATE0	$C/\overline{T}0$	M1.0	M0.0

Таблиця 2.7 – Призначення розрядів регістра TMOD

<table>
<tr><th>Біти</th><th>Найменування</th><th colspan="3">Призначення бітів</th><th>Примітка</th></tr>
<tr><td rowspan="6">0–1
4–5</td><td rowspan="6">M0–M1</td><td colspan="3">Визначають один із 4–х режимів роботи, окремо для Т/Л1 і Т/Л0.</td><td rowspan="6">Усі біти встановлюються програмно;
біти 0–3 визначають режим роботи Т/Л0,
біти 4–7 визначають режим роботи Т/Л1</td></tr>
<tr><td>M1</td><td>M0</td><td>Режим</td></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>2</td></tr>
<tr><td>1</td><td>1</td><td>3</td></tr>
<tr><td>2, 6</td><td>$C/\overline{T}0$
$C/\overline{T}1$</td><td colspan="3">Визначають роботу в якості:
$C/\overline{T}0$, $C/\overline{T}1 = 0$ – таймера
$C/\overline{T}0$, $C/\overline{T}1 = 1$ – лічильника зовнішніх подій</td><td></td></tr>
<tr><td>3, 7</td><td>GATE</td><td colspan="3">Дозволяє керувати таймером від зовнішнього виводу ($\overline{INT0}$ – для Т/Л0, $\overline{INT1}$ – для Т/Л1).
GATE = 0 – керування заборонено
GATE = 1 – керування дозволено</td><td></td></tr>
</table>

При роботі Т/Л в якості лічильника зовнішніх подій вміст регістра Т/Л інкрементується у відповідь на перехід із стану "логічна 1" у стан "логічний 0" сигналу на вході МК, який лічить (вивід Т0 для Т/Л0 і вивід Т1 для Т/Л1).

Входи МК, які лічать, апаратно перевіряються у фазі S5 P2 кожного машинного циклу.

Коли перевірки показують високий рівень сигналу на вході МК, який лічить, в одному машинному циклі і низький рівень в іншому машинному циклі, регістр Т/Л інкрементується. Нове (інкрементоване) значення заноситься в регістр Т/Л у фазі S3 P1 машинного циклу, що безпосередньо іде за тим, у якому було виявлено перехід сигналу із стану "логічна 1" у стан "логічний 0" на вході МК, який лічить. Оскільки для розпізнавання такого переходу потрібно два машинних цикли (24 періоди частоти синхронізації МК: f_{BQ}), то максимальна частота лічби Т/Л в режимі лічильника зовнішніх подій дорівнює $f_{BQ}/24$. Щоб рівень сигналу на вході МК, який лічить, було гарантовано зафіксовано, він повинен залишатися незмінним протягом як мінімум одного машинного циклу.

Регістр керування (TCON) призначено для прийому і збереження коду керуючого слова. Позначення розрядів регістра TCON наведено в таблиці 2.8, а призначення розрядів – у таблиці 2.9.

Таблиця 2.8 – Позначення розрядів регістра TCON

Біти	7	6	5	4	3	2	1	0
Позначення	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

Прапорці переповнення TF0 і TF1 встановлюються апаратно при переповненні відповідних Т/Л (перехід Т/Л із стану "всі одиниці" у стан "усі нулі"). Якщо при цьому переривання від відповідного Т/Л дозволено, то встановлення прапорця TF викликає переривання. Прапорці TF0 і TF1 скидаються апаратно при передачі керування підпрограмі обробки відповідного переривання.

Таблиця 2.9 – Призначення розрядів регістра TCON

Біти	Найменування	Призначення бітів	Примітка
6 4	TR1 TR0	Біти включення Т/Л, окремо для Т/Л0 і Т/Л1. TR = 0 – відключено, TR = 1 – включено	Біти встановлюються і скидаються програмно. Доступні для читання
7 5	TF1 TF0	Прапорці переповнення Т/Л	Біти скидаються і встановлюються апаратно і програмно. Доступні для читання
2 0	IT1 IT0	Біти, що визначають вид переривання за входами INT1, INT0. IT = 0 – переривання за рівнем (низьким), IT = 1 – переривання за фронтом (перехід із "1" в "0")	Біти встановлюються і скидаються програмно. Доступні для читання
3 1	IE1 IE0	Прапорці запиту зовнішніх переривань за входами INT1, INT0	Біти скидаються і встановлюються апаратно і програмно. Доступні для читання.
Біти 4, 5 відносяться до Т/Л0; біти 6, 7 – до Т/Л1. *Біти 0, 1 визначають зовнішні переривання за входом INT0, біти 2, 3 – за входом INT1.*			

Прапорці TF0 і TF1 програмно доступні і можуть бути встановлено/скинуто програмою. Використовуючи цей механізм, переривання

за TF0 і TF1 можуть бути викликано (встановленням TF) і скасовано (скиданням TF) програмою.

Прапорці IE0 і IE1 встановлюються апаратно від зовнішніх переривань (відповідно входи МК: INT0 і INT1) або програмно та ініціюють виклик підпрограми обробки відповідного переривання.

Скидання цих прапорців виконується апаратно при обслуговуванні переривання тільки в тому випадку, коли переривання було викликано за фронтом сигналу. Якщо переривання було викликано рівнем сигналу на вході INT0 (INT1), то скидання прапорця IE повинна виконувати підпрограма обслуговування переривання, впливаючи на джерело переривання для зняття ним запиту.

Схему інкременту призначено:

– для збільшення на 1 у кожному машинному циклі вмісту регістрів Т/Л0, Т/Л1, для яких встановлено режим таймера і дозволено лічбу;

– для збільшення на 1 вмісту регістрів Т/Л0, Т/Л1, для яких встановлено режим лічильника зовнішніх подій, дозволено лічбу і на відповідному вході МК (Т0 для Т/Л0 і Т1 для Т/Л1) зафіксовано імпульс, який треба підрахувати.

Схема фіксації INT0, INT1, T0, T1 являє собою чотири тригери. У кожному машинному циклі в момент S5 P2 у них запам'ятовується інформація з виводів МК: INT0, INT1, T0 та T1.

Схема керування прапорцями встановлює і скидає прапорці переповнення Т/Л і прапорці запитів зовнішніх переривань.

Логіка керування Т/Л синхронізує роботу регістрів Т/Л0 і Т/Л1 відповідно до запрограмованих режимів роботи і синхронізує роботу блока Т/Л з роботою МК.

Більш докладно режими роботи й особливості застосування таймерів/лічильників розглянуто в окремому розділі.

2.8 Блок послідовного порту (інтерфейсу)

Блок послідовного інтерфейсу призначено для організації введення/виведення послідовних даних.

До складу блоку входять: буфер інтерфейсу, логіка керування інтерфейсом, регістр керування, буфер передавача, буфер приймача, приймач–передавач послідовного порту.

Буфер інтерфейсу забезпечує побайтовий обмін інформацією між внутрішньою (резидентною) шиною даних і шиною інтерфейсу.

Логіку керування інтерфейсом призначено для формування сигналів керування, що забезпечують чотири режими роботи послідовного інтерфейсу.

Регістр керування (SCON) призначено для прийому і зберігання восьмибітного коду, який керує послідовним інтерфейсом. Позначення розрядів регістра SCON наведено в таблиці 2.10. Всі розряди регістра SCON програмно доступні для запису ("логічний 0" або "логічна 1") і читання.

Таблиця 2.10 – Позначення розрядів регістра SCON

Біти	7	6	5	4	3	2	1	0
Позначення	SM0	SM1	SM2	REN	TB8	RB8	TI	RI

Розряди SM0, SM1 визначають режим роботи інтерфейсу, як зазначено в таблиці 2.11.

Інші біти регістра мають таке призначення:

– *SM2* – дозвіл багатопроцесорної роботи. У режимах 2 і 3 при SM2 = 1 прапорець RI не активізується, якщо дев'ятий прийнятий біт даних дорівнює "0". У режимі 1 при SM2 = 1 прапорець RI не активізується, якщо прийнятий стоп–біт не дорівнює "1". В режимі 0 біт SM2 треба скинути у "0";

Таблиця 2.11 – Вплив розрядів SM0, SM1 регістра SCON на режим роботи інтерфейсу

SM0	SM1	Режим	Найменування	Швидкість передачі
0	0	0	Регістр зсуву	$f_{BQ}/12$
0	1	1	8–бітовий універсальний асинхронний приймач/ передавач (УАПП)	змінна, задається Т/Л1
1	0	2	9–бітовий УАПП	$f_{BQ}/64$ або $f_{BQ}/32$
1	1	3	9–бітовий УАПП	змінна, задається Т/Л1

– *REN* – дозвіл прийому послідовних даних. Встановлюється і скидається програмою відповідно для дозволу і заборони прийому;
– *TB8* – дев'ятий біт даних, що передаються у режимах 2 і 3. Встановлюється і скидається програмою;
– *RB8* – дев'ятий біт прийнятих даних у режимах 2 і 3. У режимі 1, якщо SM2 = 0, RB8 є прийнятим стоп–бітом. У режимі 0 біт RB8 не використовується;
– *TI* – прапорець переривання передавача. Встановлюється апаратно наприкінці часу видачі 8–го біта в режимі 0 або на початку стоп–біта в інших режимах. Скидається програмно;
– *RI* – прапорець переривання приймача. Встановлюється апаратно наприкінці часу прийому 8–го біта в режимі 0 або через половину інтервалу стоп–біта в режимах 1, 2, 3 при SM2 = 0. При SM2 = 1 див. опис для біта SM2.

Буфер передавача призначено для прийому з шини інтерфейсу паралельних даних і видачі їх на передавач послідовного порту.

Буфер приймача служить для прийому даних у паралельній формі від приймача послідовного інтерфейсу.

Буфер приймача і буфер передавача при програмному доступі мають однакове ім'я (SBUF) і адресу (99H). Якщо команда використовує SBUF як регістр – джерело, то звернення відбувається до буфера приймача. Якщо команда використовує SBUF як регістр призначення, то звернення відбувається до буфера передавача.

В усіх режимах роботи послідовного порту передача ініціюється будь–якою командою, що використовує SBUF як регістр призначення.

Приймач/передавач послідовного порту призначено для прийому послідовного потоку символів зі входу послідовного порту, виділення даних і видачі їх у буфер приймача, а також для прийому паралельних даних із буфера передавача, перетворення їх у послідовний потік символів і видачі його на вихід послідовного порту.

Більш докладно режими роботи й особливості застосування послідовного інтерфейсу розглянуто в окремому розділі.

2.9 Паралельні порти введення/виведення

МК AT89C51 містить 4 паралельних 8–розрядних порти введення/ виведення дискретної інформації, які програмуються: P0, P1, P2, P3.

Порти P0, P1, P2, P3 є двонаправленими портами введення/виведення і призначені для забезпечення обміну інформацією МК із зовнішніми пристроями, створюючи 32 лінії введення/виведення. Кожен з портів містить фіксатор–защіпку, що являє собою восьмирозрядний регістр, який має байтову і бітову адресацію для встановлення/скидання його розрядів за допомогою відповідних команд.

Фізичні адреси фіксаторів P0, P1, P2, P3 становлять для:

P0 – 80H, при бітовій адресації 80H…87H;

P1 – 90H, при бітовій адресації 90H…97H;

P2 – A0H, при бітовій адресації A0H…A7H;

P3 – B0H, при бітовій адресації B0H…B7H.

Крім роботи в якості звичайних портів введення/виведення лінії портів P0…P3 можуть виконувати ряд додаткових функцій, які описано нижче.

Через порт P0:

– виводиться молодший байт адреси A0…A7 при роботі з зовнішньою пам'яттю програм і зовнішньою пам'яттю даних;

– видається з МК і приймається в МК байт даних при роботі з зовнішньою пам'яттю (при цьому обмін байтом даних і виведення молодшого байта адреси зовнішньої пам'яті мультиплексовано в часі);

– задаються дані при програмуванні внутрішнього ПЗП, і читається вміст внутрішньої пам'яті програм.

Через порт P1:

– задається молодший байт адреси при програмуванні і читанні внутрішньої пам'яті програм.

Через порт P2:

– виводиться старший байт адреси A8…A15 при роботі з зовнішньою пам'яттю програм і зовнішньою пам'яттю даних (для зовнішньої пам'яті даних – тільки при використанні команд MOVX A, @ DPTR і MOVX @ DPTR, A, які формують 16–розрядну адресу);

– задаються старші розряди A8…A11 адреси при програмуванні і читанні внутрішньої пам'яті програм.

Кожна лінія порту P3 має індивідуальну альтернативну функцію:

P3.0 – RxD, вхід послідовного порту, який призначено для введення послідовних даних у приймач послідовного порту;

P3.1 – TxD, вихід послідовного порту, який призначено для виведення послідовних даних із передавача послідовного порту;

P3.2 – $\overline{\text{INT0}}$, використовується як вхід зовнішнього запиту переривання;

P3.3 – $\overline{\text{INT1}}$, використовується як вхід зовнішнього запиту переривання;

P3.4 – T0, використовується як вхід лічильника зовнішніх подій Т/Л0;

P3.5 – T1, використовується як вхід лічильника зовнішніх подій Т/Л1;

P3.6 – $\overline{WR}$, строб запису у зовнішню пам'ять даних, вихідний сигнал, який супроводжує виведення даних через порт P0 при використанні команд MOVX @ Ri, A і MOVX @ DPTR, A;

P3.7 – $\overline{RD}$, строб читання із зовнішньої пам'яті даних, вихідний сигнал, який супроводжує введення даних через порт P0 при використанні команд MOVX A, @ Ri і MOVX A, @ DPTR.

Альтернативна функція будь–якої з ліній порту P3 реалізується тільки в тому випадку, якщо у відповідному цій лінії фіксаторі–защіпці міститься сигнал "логічна. 1". Інакше на лінії порту P3 буде присутній сигнал "логічний 0".

2.10 Схема десяткової корекції акумулятора

АЛП МК дозволяє виконувати додавання двійково–десяткових даних в упакованому форматі (в 1 байт “упаковано” 2 десяткові цифри). При виконанні операції додавання таких чисел використовується команда “Додавання”, що підсумовує операнди за правилами двійкової арифметики і вміщує результат в акумулятор. Для виправлення можливої помилки (корекції вмісту акумулятора) застосовують команду десяткової корекції DA A, яка апаратно реалізується схемою десяткової корекції акумулятора.

2.11 Внутрішній тактовий генератор (OSC)

МК містить внутрішній тактовий генератор (рисунок 2.5), в якому BQ1 і BQ2 є відповідно входом і виходом підсилювача–інвертора, і який може бути ввімкнено у режим генератора при підключенні до виводів BQ1 і BQ2 кварцового резонатора або LC–ланцюжка (рисунок 2.6).

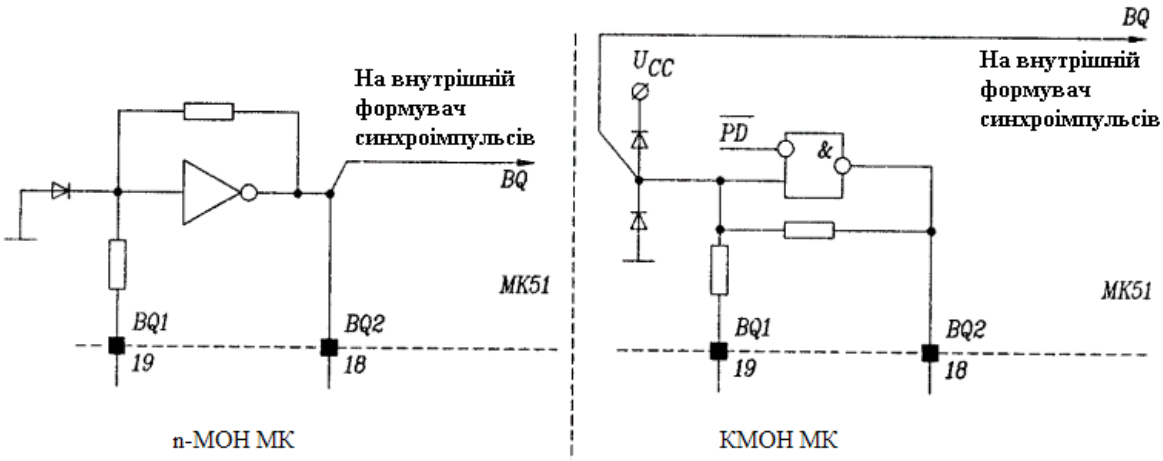


Рисунок 2.5 – Внутрішній тактовий генератор

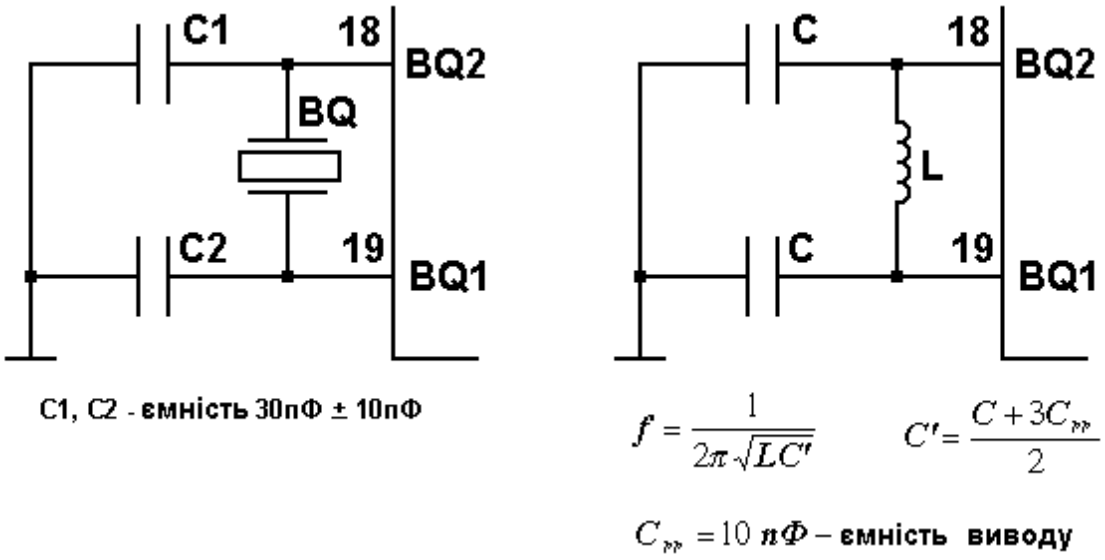


Рисунок 2.6 – Підключення до виводів BQ1 і BQ2 резонатора і LC–ланцюжка

2.12 Резидентна шина даних

Мікроконтролер містить 8–розрядну внутрішню (резидентну) шину даних (РШД), через яку здійснюється обмін інформацією між різними частинами МК.

2.13 Регістри

Акумулятор – 8–розрядний регістр, який призначено для прийому і зберігання результату, який отримано при виконанні арифметичних і логічних операцій або операцій пересилання.

Регістр B – 8–розрядний регістр, який використовується при виконанні операцій множення і ділення. В інших випадках він може розглядатися як додатковий регістр загального призначення (частина над оперативного ОЗП).

Регістри T1, T2 – 8–розрядні регістри тимчасового зберігання, які призначено для прийому і зберігання операндів на час виконання операцій над ними в АЛП. Програмно недоступні.

Регістр стану програми (PSW) служить для зберігання інформації про результат виконання операції в АЛП. Його називають також регістром прапорців (ознак). Позначення окремих розрядів регістра PSW і призначення розрядів приведено відповідно в таблицях 2.1, 2.2.

Прапорець перенесення CY може встановлюватися і скидатися як апаратно, так і програмно. Апаратно він встановлюється, якщо при виконанні арифметичних операцій формується перенесення/позика у старшому (сьомому) розряді 8–бітних операндів. При виконанні операцій множення і ділення прапорець CY скидається. Крім того, прапорець CY виконує функції “булевого акумулятора” в командах, що працюють з бітами.

Прапорець допоміжного перенесення AC встановлюється/скидається апаратно або програмно. Апаратно встановлюється при виконанні операцій додавання і віднімання при виникненні перенесення/позики в 3–му розряді при утворенні молодшої тетради результату. Частіше всього використовується схемою десяткової корекції акумулятора (СДКА) при виконанні команди DA A.

Прапорець користувача F0 встановлюється/скидається програмно і може використовуватися програмістом на свій розсуд.

Прапорці–покажчики поточного банку РЗП встановлюються/ скидаються програмно і вказують, який із 4–х банків РЗП РПД (рисунок2.2) в даний момент часу є робочим (поточним).

Прапорець переповнення OV встановлюється/скидається програмно або апаратно. Апаратно встановлюється тоді, коли при виконанні операції додавання/віднімання над числами зі знаком результат не вкладається в діапазон: –128 ... +127 і старший, знаковий біт спотворюється. При виконанні операції ділення прапорець OV апаратно скидається, а у випадку ділення на нуль встановлюється. При множенні прапорець OV апаратно встановлюється, якщо результат більше 255.

Прапорець парності (паритету) P встановлюється/скидається апаратно. Він доповнює вміст акумулятора до парного числа одиниць. У 9–розрядному слові, що складається з 8 розрядів акумулятора і біта P, завжди міститься парне число одиничних бітів.

Всі сім названих прапорців програмно доступні для читання.

Регістр команд РК (IR) призначено для зберігання коду операції (КОП) поточної команди, що виконується.

Лічильник команд ЛК (PC) містить 16–розрядну адресу комірки пам'яті програм. До складу лічильника команд входять 16–розрядні буфер PC та регістр PC, схема інкременту та регістр адреси пам'яті.

Буфер PC здійснює зв'язок між 8–розрядною РШД і 16–розрядним регістром PC, у якому зберігається поточна 16–розрядна адреса пам'яті програм.

Схема інкременту збільшує поточне значення 16–розрядної адреси пам'яті програм на одиницю.

Регістр адреси пам'яті призначено для запису і зберігання 16–розрядної адреси пам'яті програм або 8/16–розрядної адреси зовнішньої пам'яті даних.

Регістр–покажчик даних РГПД (DPTR) призначено для зберігання 16–розрядної адреси зовнішньої пам'яті даних. Складається з двох 8–розрядних регістрів DPH і DPL, що входять у блок регістрів спеціальних функцій [2,3]. Вони програмно доступні і можуть використовуватися в якості двох незалежних РЗП, якщо немає необхідності у зберіганні 16–розрядної адреси зовнішньої пам'яті даних.

Покажчик стеку (SP) адресує комірки спеціальної області пам'яті даних (РПД), яка називається стеком. SP адресує “верхівку” стеку – останню комірку стекової пам'яті, у яку записано інформацію. Покажчик стеку являє собою 8–розрядний регістр, вміст якого при виконанні команд LCALL, ACALL збільшується на 2. При виконанні команд RET, RETI вміст покажчика стеку зменшується на 2. При виконанні команди PUSH direct вміст SP збільшується на 1, а при виконанні команди POP direct – зменшується на 1.

Регістр адреси РА (RAR) – регістр комірки РШД, що адресується. Програмно не доступний.

Регістри РРТЛ (TMOD) і *РКСТ (TCON)* служать для програмування і керування роботою таймерів/лічильників і системи переривань. Формати, позначення і призначення їхніх окремих розрядів приведено в таблицях 2.6… 2.9.

Регістр РКПП (SCON), буфери ПД і *ПРМ (SBUF)* призначено для програмування і керування роботою послідовного інтерфейсу. Формати, позначення і призначення їх окремих розрядів приведено в таблицях 2.10, 2.11.

Регістри РМП (IE) і *РП (IR)* програмують і керують роботою системи переривань МК. Формати, позначення і призначення РМП і РП наведено в таблицях 2.3, 2.4.

Регістр керування потужністю РКП (PCON) служить для програмного керування споживанням енергії від джерела живлення, а також швидкістю

передачі за послідовним каналом. Формат, позначення і призначення його окремих розрядів наведено в таблицях 2.12, 2.13.

Таблиця 2.12 – Позначення розрядів регістра PCON

Біти	7	6	5	4	3	2	1	0
Позначення	SMOD	–	–	–	GF1	GF0	PD	IDL

Таблиця 2.13 – Призначення розрядів регістра PCON

Біти	Найменування	Призначення бітів	Примітка
7	SMOD	Біт подвоєння швидкості передачі через послідовний інтерфейс: при встановленні в "1" – швидкість передачі подвоюється	При роботі послідовного порту
6	–	Резервний	
5	–	Резервний	
4	–	Резервний	
3	GF1	Прапорець загального призначення	
2	GF0	Прапорець загального призначення	
1	PD	Біт вмикання режиму мікроспоживання: "1" – режим мікроспоживання ввімкнено	Якщо в PD і IDL одночасно записано "1", перевагу має PD
0	IDL	Біт холостого ходу: "1" – режим холостого ходу ввімкнено	

Більш докладно застосування цього регістра буде розглянуто в окремих розділах.

ПИТАННЯ ДЛЯ САМОКОНТРОЛЮ

1) Назвіть основні функціональні вузли МК.
2) Як співвідносяться між собою командний та машинний цикл?
3) Яке призначення пристрою формування часових інтервалів?
4) Опишіть структуру блоку арифметично–логічного пристрою.
5) Охарактеризуйте розподіл резидентної пам'яті даних та резидентної пам'яті програм.
6) Опишіть структуру системи переривань.
7) Назвіть складові блоку таймерів–лічильників та їх призначення.
8) Опишіть призначення та головні компоненти блоку послідовного інтерфейсу.
9) Назвіть призначення та додаткові функції паралельних портів введення/виведення.
10) Назвіть основні регістри МК–51 та коротко охарактеризуйте їх особливості застосування.
11) Що означає поняття «верхівка стеку» ?
12) Яке призначення має схема десяткової корекції акумулятора ?
13) Як можна подвоїти швидкість обміну послідовним портом?
14) Назвіть та опишіть альтернативні функції порту Р3.
15) Опишіть призначення окремих прапорців регістра ознак.
16) Опишіть призначення регістра команд.
17) Назвіть кількість РОН, які має РПД.
18) Назвіть довжину резидентної шини даних.
19) Поясніть призначення регістрів тимчасового зберігання операндів.
20) Назвіть об`єм пам`яті даних мікроконтролера AT89C51.
21) Назвіть об`єм пам`яті програм мікроконтролера AT89C51.
22) Скільки ліній введення/виведення має мікроконтролер AT89C51?

3 АРХІТЕКТУРА МОДУЛЯ ТАЙМЕРІВ/ЛІЧИЛЬНИКІВ

3.1 Способи формування інтервалів часу та лічба зовнішніх подій у мікропроцесорних системах та їх порівняльна характеристика

У мікропроцесорних системах (МПС) існують наступні способи формування часових інтервалів (часових затримок, одиночних імпульсів, послідовностей імпульсів і т.ін.):

- апаратний:
- програмний;
- апаратно–програмний.

Перший спосіб використовує спеціалізовані електронні пристрої, наприклад, лінії затримки, одновібратори, мультивібратори і т.ін.

У другому способі, використовуючи систему команд конкретного МП чи МК і з огляду на те, що виконання кожної команди займає деякий час, можна розробити підпрограми, які формують часові затримки, одиничні імпульси, послідовності імпульсів і т.ін.

Апаратно–програмний спосіб реалізується застосуванням таймерів, що програмуються.

Восьми– та шістнадцятирозрядні МП не містять вбудованих таймерів і використовують зовнішні мікросхеми, наприклад, КР580ВИ53.

Мікроконтролер типу МК–51 може містити два або більше внутрішніх 16–розрядних таймери, які програмуються.

Крім формування часових інтервалів, таймери можуть здійснювати лічбу імпульсів, які ідентифікують настання зовнішніх подій, тобто працювати як лічильники зовнішніх подій з апаратним чи програмним запуском.

3.2 Місце та склад модуля у структурі мікроконтролера

Таймери–лічильники: Т/Л (Timer/Counter – Т/С) – призначено для лічби

зовнішніх подій, для одержання програмно керованих часових затримок і виконання функцій мікроконтролера, які задають часові інтервали.

До складу модуля Т/Л входять (рисунок 2.1):

– два 16–розрядних регістри Т/Л0 і Т/Л1;

– восьмирозрядний регістр режимів Т/Л (TMOD);

– восьмирозрядний регістр керування/статусу (TCON);

– схема інкременту;

– схема фіксації Т0, Т1;

– схема керування прапорцями;

– логіка керування Т/Л.

Два 16–розрядні регістри Т/С0 і Т/С1 виконують функцію зберігання вмісту лічби. Кожен з них складається з пари восьмирозрядних регістрів, відповідно: TH0, TL0 і TH1, TL1. Причому регістри TH0, TH1 – старші, а регістри TL0, TL1 – молодші 8 розрядів. Кожен із восьмирозрядних регістрів має свою адресу і може бути використано як регістр загального призначення (РЗП), якщо Т/Л не використовуються (біт TL0 для Т/С0 і біт TL1 для Т/С1 у регістрі керування TCON дорівнюють "0").

Код величини початкової лічби заноситься в регістри Т/С програмно. В процесі лічби вміст регістрів Т/С інкрементується. Ознакою закінчення лічби, як правило, являється переповнення регістра Т/С, тобто перехід його вмісту зі стану "всі одиниці" у стан "усі нулі". Всі регістри TH0, TH1, TL0, TL1 доступні для читання, і, при необхідності, контроль досягнення необхідної величини лічби може виконуватися програмно.

Керуючі регістри TMOD і TCON будуть розглянуті нижче в окремому підрозділі 3.3.

Схему інкременту призначено:

– для збільшення на 1 у кожному машинному циклі вмісту регістрів Т/С0, Т/С1, для яких встановлено режим таймера і дозволено лічбу;

– для збільшення на 1 вмісту регістрів Т/С0, Т/С1, для яких встановлено режим лічильника зовнішніх подій, дозволено лічбу і на відповідному вході МК (Т0 для Т/С0 і Т1 для Т/С1) зафіксовано імпульс, який треба підрахувати.

Схема фіксації Т0, Т1 являє собою два тригери. У кожному машинному циклі в момент S5 P2 в них запам'ятовується інформація з виводів МК: Т0, Т1.

Схема керування прапорцями виробляє і скидає прапорці переповнення Т/С.

Логіка керування Т/С синхронізує роботу регістрів Т/С0 і Т/С1 відповідно до запрограмованих режимів роботи і синхронізує роботу блоку Т/С з роботою МК.

Більш докладно режими роботи та особливості застосування таймерів/лічильників розглянуто нижче.

3.3 Програмування модуля таймерів/лічильників

Для програмування модуля Т/Л призначено:

– восьмирозрядний регістр режимів Т/Л (TMOD);

– чотири старших біти восьмирозрядного регістра керування/статусу (TCON).

Регістр режимів Т/Л (TMOD) призначено для прийому і збереження коду, що визначає:

– один із 4–х можливих режимів роботи кожного Т/Л;

– роботу в якості таймерів або лічильників;

– керування Т/Л від зовнішнього виводу.

Позначення розрядів регістра TMOD наведено в таблиці 3.1, а призначення розрядів – у таблиці 3.2.

При роботі в якості таймера вміст регістра Т/С інкрементується в кожному машинному циклі, тобто Т/Л являється лічильником машинних циклів

МК. Оскільки машинний цикл складається з 12 періодів частоти синхронізації МК: f_{BQ}, то частота лічби в даному випадку дорівнює $f_{BQ}/12$.

При роботі Т/Л в якості лічильника зовнішніх подій вміст регістра Т/Л інкрементується у відповідь на перехід із стану "логічна 1" у стан "логічного 0" сигналу на вході МК, який лічить (вивід Т0 для Т/Л0 і вивід Т1 для Т/Л1). Входи, які лічать, апаратно перевіряються у фазі S5 P2 кожного машинного циклу. Коли перевірки показують високий рівень на вході, який лічить, в одному машинному циклі і низький рівень в іншому машинному циклі, регістр Т/С інкрементується.

Таблиця 3.1 – Позначення розрядів регістра TMOD

Біти	7	6	5	4	3	2	1	0
Позначення	GATE1	C/$\overline{T}$1	M1.1	M0.1	GATE0	C/$\overline{T}$0	M1.0	M0.0

Таблиця 3.2 – Призначення розрядів регістра TMOD

<table>
<tr><th>Біти</th><th>Найменування</th><th colspan="3">Призначення бітів</th><th>Примітка</th></tr>
<tr><td rowspan="6">0–1
4–5</td><td rowspan="6">M0…M1</td><td colspan="3">Визначають один із 4–х режимів роботи, окремо для Т/Л1 і Т/Л0.</td><td rowspan="6">Усі біти встановлюються програмно;
біти 0…3 визначають режим роботи Т/Л0,
біти 4…7 визначають режим роботи Т/Л1</td></tr>
<tr><td>M1</td><td>M0</td><td>Режим</td></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>2</td></tr>
<tr><td>1</td><td>1</td><td>3</td></tr>
<tr><td>2, 6</td><td>C/$\overline{T}$0
C/$\overline{T}$1</td><td colspan="3">Визначають роботу в якості:
C/T0, C/T1 = 0 – таймера;
C/T0, C/T1 = 1 – лічильника зовнішніх подій</td></tr>
</table>

Продовження таблиці 3.2

Біти	Найменування	Призначення бітів	Примітка
3, 7	GATE	Дозволяє керувати таймером від зовнішнього виводу ($\overline{INT0}$ – для Т/Л0, $\overline{INT1}$ – для Т/Л1). GATE = 0 – керування заборонено; GATE = 1 – керування дозволено	

Нове (інкрементоване) значення заноситься в регістр Т/С у фазі S3 P1 машинного циклу, що безпосередньо іде за тим, у якому було виявлено перехід із стану "логічна 1" у стан "логічний 0" на вході, який лічить. Оскільки для розпізнавання такого переходу потрібно два машинних цикли (24 періоди частоти синхронізації МК: f_{BQ}), то максимальна частота лічби Т/Л в режимі лічильника зовнішніх подій дорівнює $f_{BQ}/24$.

Щоб рівень сигналу на лічильному вході було гарантовано зафіксовано, він повинен залишатися незмінним протягом як мінімум одного машинного циклу.

Регістр керування/статусу (TCON) призначено для прийому і збереження коду керуючого слова. Позначення розрядів регістра TCON наведено в таблиці 3.3, а призначення розрядів – у таблиці 3.4.

Таблиця 3.3 – Позначення розрядів регістра TCON

Біти	7	6	5	4	3	2	1	0
Позначення	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

Прапорці переповнення TF0 і TF1 встановлюються апаратно при переповненні відповідних Т/Л (перехід Т/Л із стану "всі одиниці" у стан "всі нулі"). Якщо при цьому переривання від відповідного Т/Л дозволено, то

встановлення прапорця TF викликає переривання. Прапорці TF0 і TF1 скидаються апаратно при передачі керування підпрограмі обробки відповідного переривання.

Прапорці TF0 і TF1 програмно доступні і можуть бути встановлено/скинуто програмою. Використовуючи цей механізм, переривання за TF0 і TF1 можуть бути викликано встановленням TF і скасовано скиданням TF програмою.

Біти 0…3 регістра TCON використовуються при програмуванні системи переривань мікроконтролера і до програмування модуля таймерів/лічильників не мають відношення.

Таблиця 3.4 – Призначення розрядів регістра TCON

Біти	Найменування	Призначення бітів	Примітка
6 4	TR1 TR0	Біти включення Т/Л, окремо для Т/Л0 і Т/Л1. TR = 0 – вимкнено, TR = 1 – увімкнено.	Біти встановлюються і скидаються програмно. Доступні для читання.
7 5	TF1 TF0	Прапорці переповнення Т/Л.	Біти скидаються і встановлюються апаратно і програмно. Доступні для читання.

Продовження таблиці 3.4

Біти	Найменування	Призначення бітів	Примітка
2 0	IT1 IT0	Біти, що визначають вид переривання за входами INT1, INT0. IT = 0 – переривання за рівнем (низьким), IT = 1 – переривання за фронтом (перехід із "1" в "0")	Біти встановлюються і скидаються програмно. Доступні для читання
3 1	IE1 IE0	Прапорці запиту зовнішніх переривань за входами INT1, INT0	Біти скидаються і встановлюються апаратно і програмно. Доступні для читання
			Біти 4, 5 відносяться до Т/Л0; біти 6, 7 – до Т/Л1. Біти 0, 1 визначають зовнішні переривання за входом INT0, біти 2, 3 – за входом INT1

3.4 Робота таймерів/лічильників за структурною схемою

3.4.1 Загальна характеристика

Мікроконтролер містить два 16–розрядних таймери/лічильники зовнішніх подій Т/С, які програмуються та позначаються Т/С0 і Т/С1. Таймери/лічильники можуть програмуватися на режим таймера або лічильника зовнішніх подій. В режимі таймера вміст Т/С інкрементується в кожному машинному циклі, тобто з інтервалом $12 \cdot T_{BQ}$, де T_{BQ} – період частоти зовнішнього кварцового резонатора. В режимі лічильника зовнішніх подій Т/С

інкрементуються під дією переходу з рівня “логічна 1” на рівень “логічний 0” зовнішнього сигналу на входах T0, T1 (лінії P3.4, P3.5). Для розпізнавання перепаду з 1 в 0 потрібно не менше двох машинних циклів. У фазі S5 P2 першого МЦ на вході Ti (i = 0, 1) фіксується значення “логічна 1”. У фазі S5 P2 наступного МЦ розпізнається нульове значення сигналу на вході Ti (тобто виявляється перепад із 1 в 0). У черговому МЦ здійснюється інкремент вмісту T/C.

Таким чином, мінімальний період і максимальна частота імпульсів на вході Ti, які ідентифікують появу зовнішніх подій, визначаються з виразів:

$$T_{зовн.\ под.\ \min} = 24 \cdot T_{BQ},$$
$$f_{зовн.\ под.\ \max} = \frac{f_{BQ}}{24}. \qquad (3.1)$$

Для того, щоб при визначенні перепаду з 1 у 0 значення “логічна 1” і “логічний 0” на вході Ti було зафіксовано, тривалість імпульсів і пауз повинні відповідати співвідношенням:

$$t_{імп.\ зовн.\ под.} \geq T_{МЦ} = 12 \cdot T_{BQ},$$
$$t_{паузи\ зовн.\ под.} \geq T_{МЦ} = 12 \cdot T_{BQ}. \qquad (3.2)$$

Логічно кожен T/C розбивається на 2 частини по 8 біт, які позначаються TH0, TL0 для T/C0 і TH1, TL1 для T/C1. Останні можна використовувати як 8–розрядні регістри загального призначення, якщо попередньо зупинити T/C, скинувши біти TR0 = TR1 = 0 у регістрі РКСТ (TCON).

Початкові значення $N_{поч}$ в регістри THi, TLi (i = 0, 1) при програмуванні T/C записуються довільно.

Керувати лічбою (дозволяти/забороняти) можна програмно та апаратно.

Регістри THi і TLi програмно доступні для запису та читання.

При переповненні Т/С в процесі лічби встановлюються прапорці TFi (i = 0, 1) в регістрі РКСТ (TCON). Це може викликати переривання основної програми або ці прапорці можуть опитуватись програмно.

Для програмування і керування роботою Т/С призначено два регістри: TMOD і TCON. Позначення і призначення їхніх окремих розрядів наведено в таблицях 3.1…3.4.

3.4.2 Робота таймерів/лічильників у окремих режимах

Таймери/лічильники можуть працювати в одному із 4–х режимів роботи: 0, 1, 2 і 3.

Режим 0. Встановлення бітів M0 = 0, M1 = 0 в регістрі TMOD налаштовує обидва таймери на роботу в режимі 0. Спрощену структуру Т/С в цьому режимі наведено на рисунку 3.1.

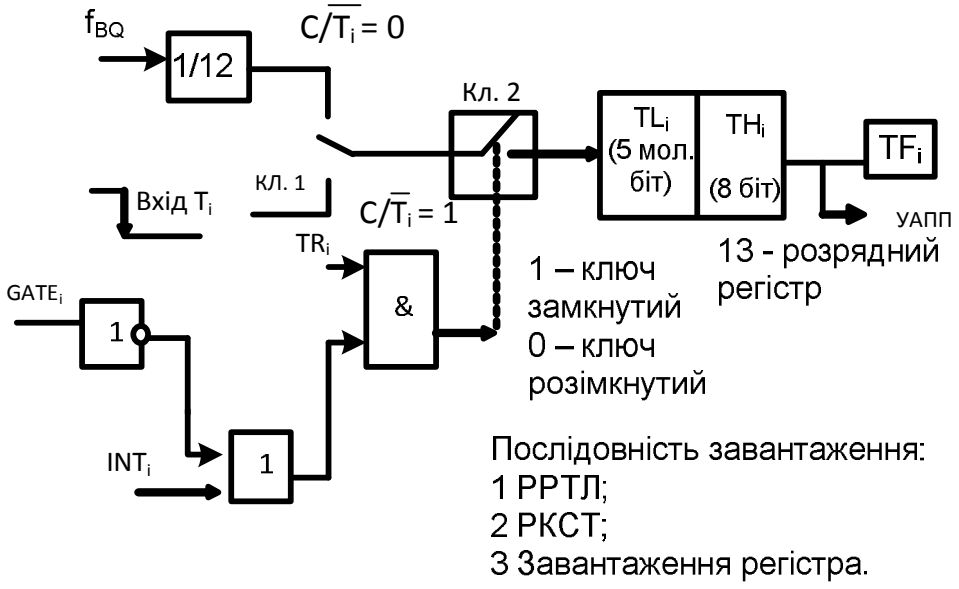


Рисунок 3.1 – Спрощена структурна схема і–го (i = 0, 1) таймера/лічильника в режимі 0

В цьому режимі кожен Т/Л працює як 13–розрядний двійковий лічильник. П’ять молодших розрядів виконують функцію попереднього дільника вхідної частоти на $2^5 = 32$. Значущими в цьому режимі є 5 молодших розрядів регістрів TLi і 8 розрядів THi. Обидва Т/Л можуть працювати як таймери або лічильники зовнішніх подій в залежності від значення розрядів $C/\overline{T}_i$ в регістрі TMOD.

Існують такі можливості зупинки (заборони) і включення (дозволу) лічби:

Програмно:

– скидання/встановлення бітів TRi в TCON при GATEi = 0 або INTi = 1 забороняє/дозволяє лічбу;

– встановлення/скидання біта GATEi в регістрі TMOD при TRi = 1 і INTi = 0 забороняє/дозволяє лічбу.

Апаратно: нульовий/одиничний сигнал на вході INTi при GATEi = 1 і TRi = 1 забороняє/дозволяє лічбу.

В обох випадках (при програмному та апаратному керуванні) при зупинці в Т/Сі зберігається поточне значення, а після увімкнення лічба продовжується з цього ж значення, якщо під час зупинки Т/Сі не перезавантажувався програмно.

Використання $C/\overline{T}_i$ *для вимірювання тривалості одиничного імпульсу* $t_{імп.\ вх}$ *на вході INTi*. Програмно встановлюється TRi = 1, GATEi = 1, $C/\overline{T}_i = 0$. Забороняється переривання від надходження активного логічного нульового сигналу на вході INTi. Завантажується в Т/Сі початкове значення $N_{поч}$. До надходження імпульсу на вході INTi зберігається значення “логічного 0” і Т/Сі зупинено (лічбу заборонено).

При надходженні одиничного імпульсу лічбу дозволено на час, який дорівнює тривалості цього імпульсу. Після закінчення імпульсу лічба знову апаратно забороняється і Т/Сі зупиняється. За різницею чисел в таймері після і до лічби та відомій частоті переключення Т/Сі: $f_{T/Ci} = f_{BQ}/12$ визначається $t_{імп.\ вх}$.

При переповненні T/Ci (перехід вмісту регістра T/C із стану “всі одиниці” у стан “всі нулі”) встановлюються прапорці TFi в регістрі TCON.

На виході першого T/C в момент переповнення формується короткий імпульс, який поступає в блок синхронізації послідовного інтерфейсу (УАПП).

Режим 1. Встановлення бітів M0 = 1, M1 = 0 в регістрі TMOD налаштовує обидва таймери на роботу в режимі 1. Кожен T/Ci працює як 16–розрядний лічильник. Решта (програмування, робота і застосування T/Ci) відбувається аналогічно режиму 0.

Режим 2. Встановлення бітів M0 = 0, M1 = 1 в регістрі TMOD визначає роботу T/Ci у режимі 2. Спрощену структуру T/C1 у цьому режимі наведено на рисунку 3.2.

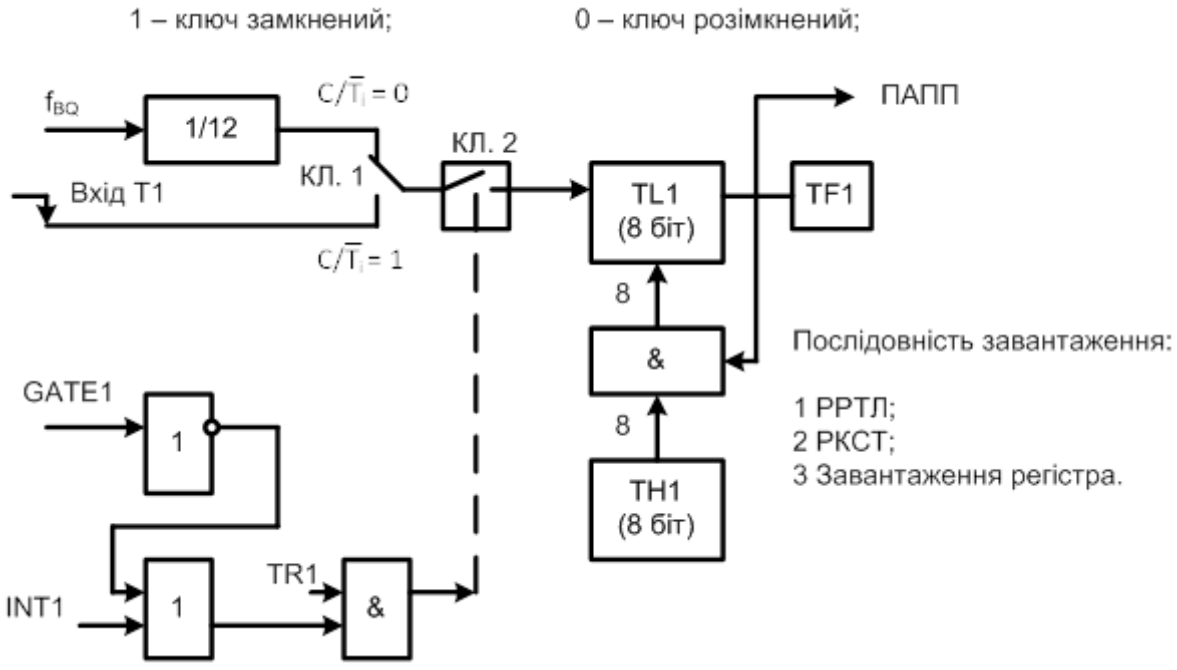


Рисунок 3.2 – Спрощена структурна схема таймера/лічильника 1 в режимі 2

Кожен T/Ci може працювати як 8–розрядний таймер/лічильник, який автоматично перезавантажується. У якості лічильника використовується регістр TLi, а THi містить програмно встановлене початкове значення з якого ведеться

лічба. При кожному черговому переповненні TLi встановлюється прапорець переповнення TFi і автоматично початкове значення перезавантажується з THi в TLi.

На виході першого таймера/лічильника: T/C_1 в цьому режимі формується послідовність прямокутних імпульсів, яка надходить в блок послідовного інтерфейсу і може використовуватися для синхронізації роботи останнього.

Режим 3. Встановлення бітів M0 = 1, M1 = 1 визначає режим 3. Спрощену структуру T/C0 у цьому режимі показано на рисунку 3.3.

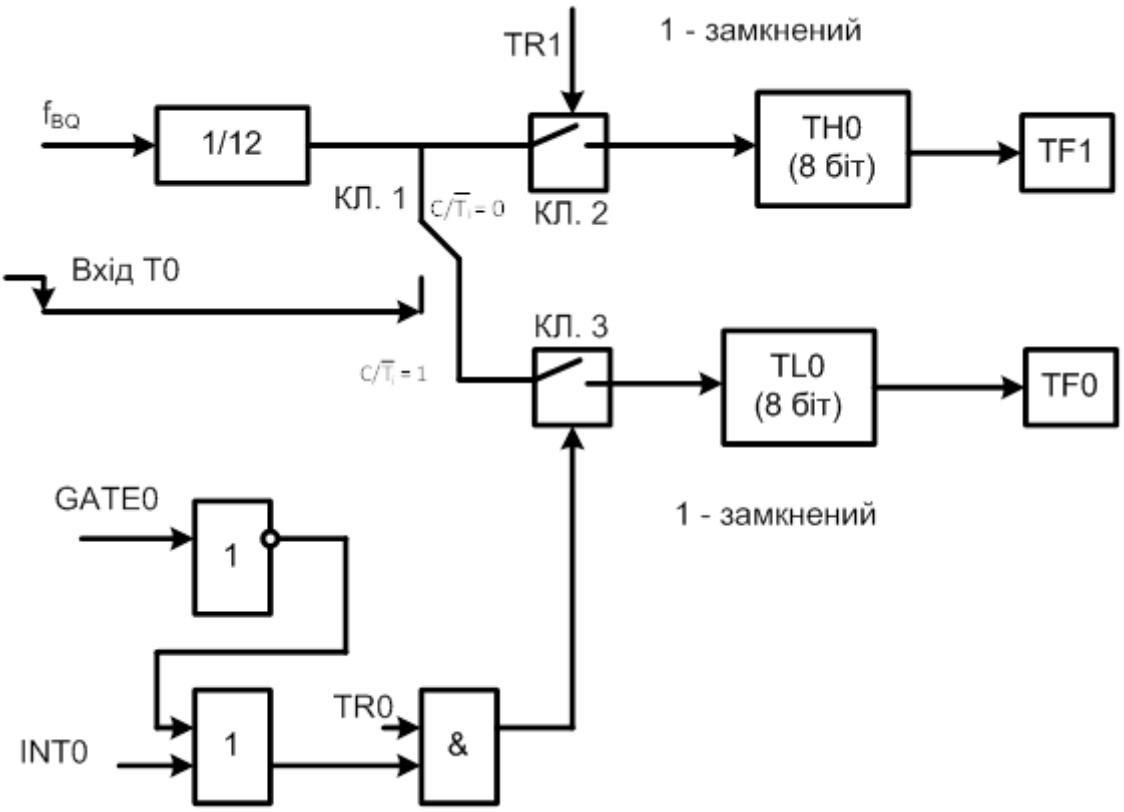


Рисунок 3.3 – Спрощена структурна схема таймера/лічильника 0 в режимі 3

Режим 3 має ряд особливостей. Якщо в цьому режимі запрограмувати обидва таймери, то T/C1 зупиняється, а T/C0 працює як два незалежних 8–розрядних регістри TH0 і TL0.

Пристрій на основі регістра TL0 може працювати в режимі таймера і в режимі лічильника зовнішніх подій. За ним зберігаються усі біти керування T/C0, він реагує на сигнали на входах T0 і INT0. При переповненні TL0 встановлюється прапорець TF0.

Пристрій на основі регістра TH0 може працювати тільки в режимі таймера. Для керування він використовує частину бітів, що керують роботою T/C1, який у режимі 3 зупинено. TR1 керує вмиканням/вимиканням TH0. При переповненні TH0 встановлюється прапорець TF1.

Друга особливість режиму 3 полягає в тому, що T/C0 може програмуватися в режим 3, а T/C1 – в режим 0, 1 чи 2. Оскільки біт TR1 керує роботою TH0, то T/C1 у режимах 0, 1, 2 при GATE1 = 0 завжди включено, а при GATE1 = 1 – виключено. Прапорець переповнення TF1 використовується TH0, тому при переповненні в режимах 0 і 1 T/C1 обнуляється, а в режимі 2 – перезавантажується, не встановлюючи прапорець переповнення. Інші керуючі біти і сигнали T/C1 використовуються аналогічно режимам 0, 1 та 2.

T/C1 апаратно пов’язано з блоком синхронізації послідовного інтерфейсу. При роботі в режимах 0, 1, 2 при переповненні T/C1 завжди формується тактовий імпульс, який надходить в інтерфейс. Тому режим 3 для T/C0 зручно застосовувати коли:

– потрібна робота двох 8–розрядних таймерів (TH0, TL0) і формування тактових імпульсів для послідовного інтерфейсу (T/C1 у режимі 2);

– потрібна робота 8–розрядного таймера (TH0), 8–розрядного лічильника зовнішніх подій (TL0) і формування синхроімпульсів для послідовного порту (T/C1 в режимі 2).

3.5 Розвиток архітектури модуля таймерів/лічильників у сучасних мікроконтролерах сімейства МК–51

3.5.1 Загальна інформація

В будь–якому мікроконтролері сімейства МК–51 наявні як мінімум по 2 таймери/лічильники. Їх характеристики та програмування було розглянуто вище. Тепер зупинимось на подальшому розвитку модуля таймерів/лічильників у сучасних МК сімейства МК–51. В таблиці 3.5 наведено наявність таймерів/лічильників у деяких мікроконтролерах сімейства Atmel.

У сучасних мікроконтролерах третім таймером/лічильником є таймер 2, який буде розглянуто нижче. Також новими для мікроконтролерів є модуль PCA (Programmable Counter Array), вартовий таймер (Watchdog Timer) і апаратний вартовий таймер (HWDT або WDTA) [11, 13].

Таблиця 3.5 – Таймери/лічильники загального призначення деяких МК виробництва Atmel

Таймер/лічильник	AT89LV51	AT89C52	AT89C51RC	AT89C51SND2	AT89C5131	AT89C5122
Таймер/лічильник T0 (8–розрядний)	♦	♦	♦	♦	♦	♦
Таймер/лічильник T1 (16–розрядний)	♦	♦	♦	♦	♦	♦
Таймер/лічильник T2 (16–розрядний)	–	♦	♦	–	♦	–
PCA	–	–	♦	–	♦	–
WDT	–	–	–	♦	♦	–
WDTA	–	–	–	–	–	♦

3.5.2 Таймер/лічильник Т2

3.5.2.1 Загальна характеристика

Таймер/лічильник Т2 (далі Т/Л Т2) представляє собою додатковий пристрій, який включено в клони мікроконтролера 8052 та сучасних модифікацій МК–51. Т/Л Т2 має цілий ряд особливостей і додаткових можливостей в порівнянні з Т/Л Т1 та Т0 в класичних мікроконтролерах 8051. Т/Л Т2 програмується групою спеціальних регістрів:

- T2CON – регістр керування і контролю. Регістр може адресуватися побітово;
- RCAP2H – старший байт, вміст якого використовується при роботі в режимі автоперезавантаження;
- RCAP2L – молодший байт, вміст якого використовується при роботі в режимі автоперезавантаження;
- TH1 – старший байт Т/Л Т2;
- TL1 – молодший байт Т/Л Т2.

Т/Л Т2, як і Т/Л Т1 та Т0, може працювати в якості 16–розрядного таймера, в режимі автоперезавантаження, в режимі генератора синхронізації при послідовному обміні даними, а також має додатковий режим, який називається режимом захоплення (capture mode). Для керування Т/Л Т2 використовується регістр спеціальних функцій T2CON, позначення бітів якого наведено на рисунку 3.4.

T2CON

TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/$\overline{T2}$	CP/RL2
7	6	5	4	3	2	1	0

Рисунок 3.4 – Позначення бітів регістра T2CON

Призначення бітів регістра T2CON наступні:

- TF2 – встановлення цього біта свідчить про переповнення Т/Л Т2. Якщо дозволено переривання від Т/Л Т2, то встановлення цього біта викликає переривання;
- EXF2 – встановлюється при виникненні переповнення, чи, якщо сигнал на виводі T2EX (P1.1) переходить із високого в низький рівень (перепад з 1 в 0). Перепад сигналу фіксується тільки при встановленні біта EXEN2;
- RCLK – якщо даний біт встановлено, то Т/Л Т2 використовується в якості генератора синхронізації послідовного порту при прийомі даних. Якщо біт дорівнює 0, то для синхронізації використовується Т/Л Т1;
- TCLK – якщо даний біт встановлено, то Т/Л Т2 використовується в якості генератора синхронізації послідовного порту при передачі даних. Якщо біт дорівнює 0, то для синхронізації використовується Т/Л Т1;
- EXEN2 – при встановленні біта перепад сигналу із високого в низький рівень на виводі T2EX (P1.1) ініціює режим захоплення, або викликає перезапуск Т/Л Т2;
- TR2 – встановлення цього біта в 1 викликає запуск Т/Л Т2. Скидання в 0 зупиняє роботу Т/Л Т2;
- C/$\overline{T2}$ – якщо даний біт скинуто, то Т/Л Т2 функціонує в режимі інтервального таймера. Якщо даний біт встановлено, то Т/Л Т2 інкрементується кожен раз при перепаді з 1 в 0 на виводі Т/Л Т2 (P1.0) (працює в якості лічильника зовнішніх подій);
- CP/RL2 – при скинутому біті, переповнення Т/Л Т2 виникає при роботі в режимі автоперезавантаження або при перепаді з 1 в 0 на виводі T2EX (біт EXEN2 необхідно встановити в 1). Якщо даний біт

встановлено, то Т/Л Т2 працює в режимі захоплення при виникненні перепаду з 1 в 0 на виводі T2EX (біт EXEN2 необхідно встановити в 1).

Програмування Т/Л Т2 в різних режимах відображає таблиця 3.6.

Таблиця 3.6 – Режими роботи таймера/лічильника Т2

RCLK і TCLK	CP/RL2	TR2	Режим
0 і 0	0	1	16–бітний таймер/лічильник із перезавантаженням
0 і 0	1	1	16–бітний таймер/лічильник із захопленням інформації
Хоча б один встановлено в 1	Будь–яке	1	Генератор синхронізації для приймача або передавача послідовного порту
Будь–яке	Будь–яке	0	Таймер відключено

3.5.2.2 Режим автоперезавантаження

Якщо Т/Л Т2 працює в режимі 16–бітного таймера/лічильника з автоперезавантаженням, то його можна налаштувати на підрахунок вгору або вниз (збільшення або зменшення вмісту регістрів TL2, TH2). Цей режим викликається встановленням в 1 біта DCEN (Down Counter Enable), який розташовано в регістрі T2MOD (таблиця 3.7). При скиданні біта DCEN Т2 за замовчуванням рахує вгору. Коли біт DCEN встановлено, Т/Л Т2 може рахувати вгору або вниз в залежності від значення сигналу на виводі P1.1.

Таблиця 3.7 – Регістр режиму Т/Л Т2 T2MOD

Адреса	Ім’я	Функція
T2MOD.7		Не використовується
T2MOD.6		Не використовується
T2MOD.5		Не використовується
T2MOD.4		Не використовується
T2MOD.3		Не використовується
T2MOD.2		Не використовується
T2MOD.1	T2OE	При встановленні біта на виводі P1.1 формується послідовність прямокутних імпульсів зі шпаруватістю 2
T2MOD.0	DCEN	При вставленні цього біта Т/Л Т2 конфігурується на підрахунок як вгору, так і вниз, в залежності від сигналу на виводі P1.1: 0 – вниз, 1 – вгору.
Біти регістра T2MOD не адресуються безпосередньо командами роботи з бітами. При скиданні мікроконтролера біти T2MOD.0 і T2MOD.1 скидаються в 0, значення решти бітів не визначено.		

3.5.2.3 Режим захоплення таймера/лічильника Т2

Суть цього режиму полягає в тому, що при встановленні прапорця EXEN2 регістра T2CON таймер може реагувати на перепад з 1 в 0 на виводі T2EX (P1.1). У момент фіксації перепаду поточні значення регістрів TH2 і TL2 запам'ятовуються в регістрах RCAP2H і RCAP2L відповідно. У цей же момент встановлюється прапорець EXF2, що може викликати переривання від Т/Л Т2. Слід мати на увазі, що навіть при встановленому режимі захоплення встановлення прапорця TF2, що сигналізує про переповнення таймера, також викликає переривання. У таких випадках програмний код обробника переривання Т/Л Т2 повинен враховувати обидві можливі причини виникнення

переривання і обробляти їх відповідним чином. Режим дуже ефективний при вимірах часових параметрів вхідних сигналів. При цьому прапорець EXF2, так само як і TF2, повинен скидатися програмно.

3.5.3 Таймер/лічильник PCA (Programmable Counter Array)

3.5.3.1 Загальна характеристика

Programmable Counter Array (група програмованих лічильників) має 16–бітний таймер/лічильник, який складається із регістрів CH і CL (відповідно старший і молодший байти), а також групу 16–бітних програмованих модулів порівняння/захоплення. В таблиці 3.8 приведено зв'язок таймера та модулів PCA із зовнішніми виводами мікроконтролерів.

Регістри PCA можна прочитати та завантажити в будь–який час. Читання повного 16–бітного значення таймера/лічильника PCA (далі будемо називати Т/Л PCA) потребує використання одного із PCA–модулів в режимі захоплення.

Спрощену структуру 16–бітного таймера/лічильника PCA представлено на рисунку 3.5.

Таблиця 3.8 – Зв'язок модулів PCA і зовнішніх виводів мікроконтролерів

Модулі PCA	Зовнішній вивід
16–бітний лічильник	P1.2/ECI
16–бітний модуль 0 порівняння/захоплення	P1.3/CEX0
16–бітний модуль 1 порівняння/захоплення	P1.3/CEX1
16–бітний модуль 2 порівняння/захоплення	P1.3/CEX2
16–бітний модуль 3 порівняння/захоплення	P1.3/CEX3
16–бітний модуль 4 порівняння/захоплення	P1.3/CEX4

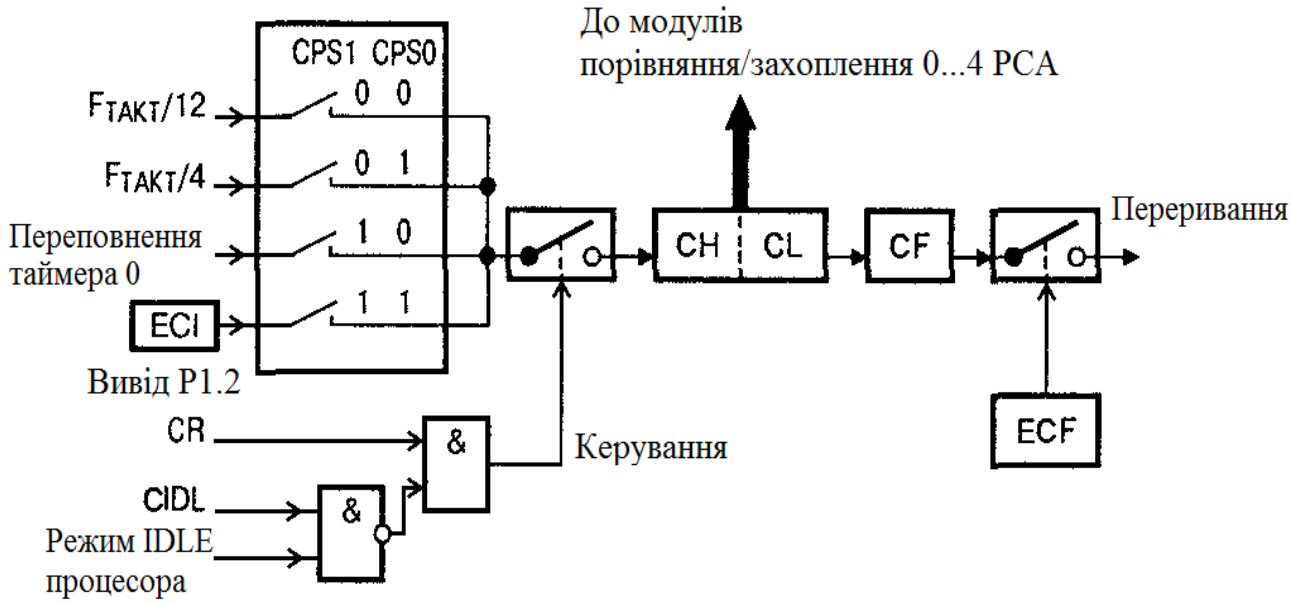


Рисунок 3.5 – Спрощена структура 16–бітного таймера/лічильника PCA

Користувач може обрати один з чотирьох варіантів вхідних сигналів Т/Л PCA:

- частоту тактового генератора, яку поділено на 12 (Fтакт/12). При цьому таймер PCA інкрементується один раз в кожному машинному циклі. Якщо тактова частота дорівнює 16 МГц – таймер інкрементується кожні 750 нс;
- частоту тактового генератора, яку поділено на 4 (Fтакт/4).

  Т/Л PCA інкрементується тричі в кожному машинному циклі, тобто кожні 250 нс при тактовій частоті генератора 16 МГц;
- переповнення таймера/лічильника Т/С0.

  Т/Л PCA інкрементується кожен раз, коли відбувається переповнення Т/С0. Цей режим дає можливість програмно задавати частоту вхідного сигналу Т/Л PCA;
- вхідний сигнал на лінії P1.2 (ECI). Т/Л PCA інкрементується з кожним переходом із 1 в 0 на вході ECI (вивід P1.2 мікроконтролера).

Максимально допустима частота сигналу на цьому вході дорівнює частоті тактового генератора, яку поділено на 8.

Вибір вхідного сигналу здійснюється бітами CPS0 і CPS1 в регістрі CMOD. В цьому регістрі також знаходиться біт ECF, який дозволяє переривання при переповненні Т/Л PCA. Крім того, встановлення в 1 біта CIDL дає можливість відключати Т/Л PCA в режимі Idle, що дає зниження енергоспоживання в згаданому режимі на 30%. В таблиці 3.9 приведено імена і призначення бітів регістра CMOD.

Ще два біти, що наведено у структурі таймера/лічильнику PCA (рисунок 3.5), знаходяться в регістрі CCON (опис наведено у таблиці 3.10). Біт CF встановлюється апаратно при переповненні таймера/лічильника. Встановлення або скидання біта CR відповідно вмикає або вимикає лічильник при умові, що біт CIDL=0 або відсутній режим IDLE.

3.5.3.2 Модуль порівняння/захоплення

Кожній із п’яти модулів порівняння/захоплення може функціонувати в наступних режимах:

- 16–бітний регістр–защіпка з керуванням за фронтом вхідного імпульсу;
- 16–бітний регістр–защіпка з керуванням за спадом вхідного імпульсу;
- 16–бітний програмований таймер/лічильник;
- 16–бітний високошвидкісний вихід;
- 8–бітний широтно–імпульсній модулятор (ШІМ).

Додатково модуль 4 може використовуватися як вартовий таймер. Кожен модуль можна запрограмувати на виконання будь–якої із функцій незалежно від інших.

Кожен із модулів має регістр CCAPMn (n=0…4), за допомогою якого відбувається вибір режиму його функціонування (опис наведено у таблиці 3.11).

Таблиця 3.9 – Регістр режиму таймера/лічильника PCA–CMOD

Символ	Позиція	Ім’я та призначення
CIDL	CMOD.7	Встановлення цього біта в 1 завершує лічбу Т/Л PCA в режимі Idle. При нульовому значенні CIDL лічба в режимі Idle не завершується
WDTE	CMOD.6	Біт керування вартовим таймером. Встановлення його в 1 дозволяє функціонування вартового таймера, скидання в 0 – забороняє
–	CMOD.5	Зарезервовано для подальшого використання
–	CMOD.4	Зарезервовано для подальшого використання
–	CMOD.3	Зарезервовано для подальшого використання
CPS1	CMOD.2	Вибір джерела тактування Т/Л PCA, старший біт (рисунок 3.5)
CPS0	CMOD.1	Вибір джерела тактування Т/Л PCA, молодший біт (рисунок 3.5).
ECF	CMOD.0	Біт дозволу переривання за переповненням таймера/лічильника PCA. При ECF=1 дозволено переривання при встановленні біта CF регістра CCON, при ECF=0 – заборонено переривання.
Адреса регістра CMOD – 0D9H, значення при скиданні – 00xxx000b		

Таблиця 3.10 – Регістр керування PCA–CCON

Символ	Позиція	Ім’я та призначення
CF	CMOD.7	Прапорець переповнення Т/Л PCA. Встановлюється апаратно і викликає переривання при ECF=1. Може бути встановлено також і програмно. Скидається тільки програмний шляхом
CR	CMOD.6	Біт керування ввімкненням лічильника PCA. Встановлення його в 1 дозволяє функціонування лічильника, скидання в 0 – забороняє при умові, що біт CIDL=0 або відсутній режим IDLE
–	CMOD.5	Зарезервовано для подальшого використання.
CCF4	CMOD.4	Прапорець порівняння/захоплення модуля 4 PCA. Очищується програмно
CCF3	CMOD.3	Прапорець порівняння/захоплення модуля 3 PCA
CCF2	CMOD.2	Прапорець порівняння/захоплення модуля 2 PCA
CCF1	CMOD.1	Прапорець порівняння/захоплення модуля 1 PCA
CCF0	CMOD.0	Прапорець порівняння/захоплення модуля 0 PCA
Адреса регістра CCON – 0D8H, значення при скиданні – 00x00000b		

Біт ECCFn регістр CCAPMn дозволяє переривання від модуля PCA, якщо його прапорець переривання (CCFn) було встановлено. Ці прапорці (CCF0…CCF4) знаходяться в регістрі CCON і встановлюються при описаних вище умовах функціонування модуля в режимах захоплення, програмованого таймера або в режимі високошвидкісного виходу. Крім того, кожен модуль має пару власних 8–бітних регістрів порівняння/защіпки (CCAPnH і CCAPnL). В цих регістрах запам’ятовуються значення таймера PCA в момент приходу

зовнішнього сигналу на захоплення інформації або містяться дані для порівняння з показами таймера/лічильника. В ШІМ–режимі старший байт CCAPnH керує шпаруватістю вихідних імпульсів.

Таблиця 3.11 – Регістр модуля порівняння–защіпки n: CCAPMn

Символ	Позиція	Ім’я і призначення
–	CCAPMn.7	Зарезервовано для подальшого використання
ECOMn	CCAPMn.6	Прапорець дозволу компаратора PCA (функція порівняння)
CAPPn	CCAPMn.5	Біт дозволу захоплення за додатним фронтом (перепад з 0 в 1)
CAPNn	CCAPMn.4	Біт дозволу захоплення за від’ємним фронтом (за спадом, перепадом з 1 в 0)
MATn	CCAPMn.3	При встановлені цього біта в 1 при рівності лічильника PCA і відповідного регістра порівняння/защіпки встановлюється прапорець переривання CCFn
TOGn	CCAPMn.2	При встановленні цього біта в 1 при рівності лічильника PCA і відповідного регістра порівняння/защіпки змінюється рівень (з 0 на 1 або навпаки) на відповідному виводі CEXn
PWMn	CCAPMn.1	При встановленні цього біта на вивід CEXn видається широтно–імпульсно–модульований сигнал (ШІМ– сигнал)
ECCFn	CCAPMn.0	Дозвіл переривання за встановленням прапорця CCFn регістра CCON
Адреса регістра з 0DAH (CCAPM0) по 0DEH (CCAPM4), значення при скиданні – x0000000b		

3.5.3.3 Режим захоплення PCA

Захоплення інформації можна здійснювати як додатним, так і від'ємним перепадом вхідного імпульсу, що робить PCA доволі гнучким інструментом для вимірювання періоду слідування сигналів, ширини імпульсу, шпаруватості і фазових затримок між імпульсами, які можуть приходити за п'ятьма входами. Встановлення бітів CAPPn або CAPNn в регістрі CCAPMn вибирає, яким перепадом буде відбуватися захоплення інформації – фронтом або спадом імпульсу в модулі n. Спрощену структуру модуля захоплення показано на рисунку 3.6.

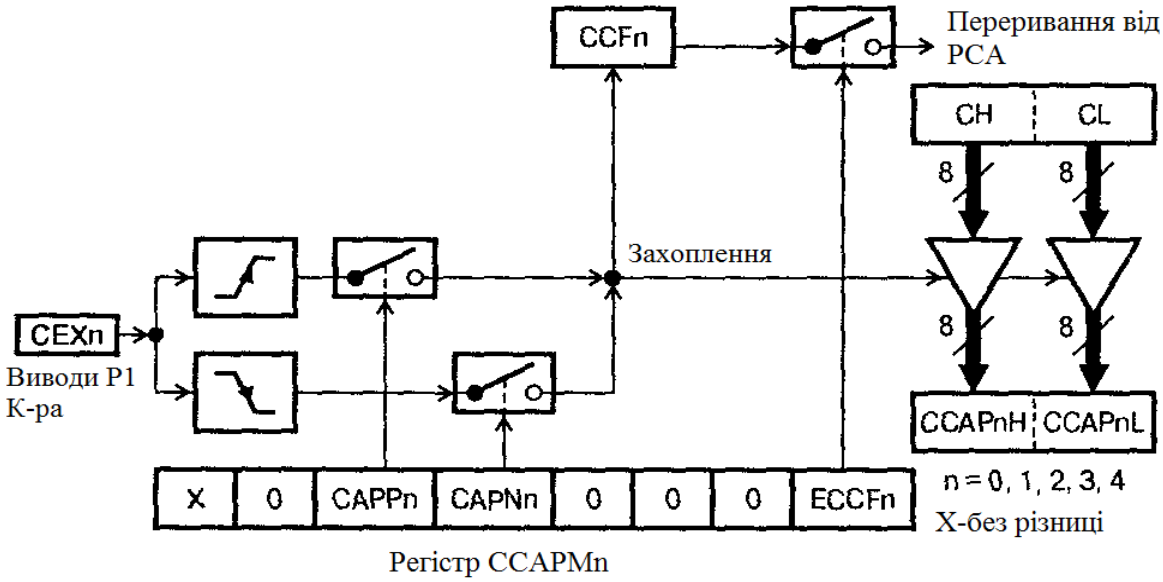


Рисунок 3.6 – Спрощена структура модуля захоплення

Керування захопленням інформації відбувається імпульсами на зовнішніх виводах CEX0…CEX4 (таблиця 3.8). Коли надійно виявлено перепад сигналу (фронт або спад), відбувається апаратне завантаження 16–бітного значення лічильника PCA (CH, CL) в регістри CCAPnH, CCAPnL відповідного модуля/защіпки. Результуючі значення цих регістрів відображають значення лічильника PCA в момент виявлення перепаду сигналу на зовнішньому виводі CEXn.

При захоплені встановлюється в одиницю зв’язаний з відповідним модулем прапорець CCFn в регістрі CCON. Якщо при цьому встановлено прапорець ECCFn в регістрі CCAPMn, то формується виклик переривання від PCA (відповідно, якщо воно не замасковано). Так як апаратно цей прапорець не скидається, після того, як підпрограму обробки переривання викликано, його потрібно скинути програмним шляхом в ході виконання цієї підпрограми.

Також в ході цієї підпрограми 16–бітне значення, яке записано в CCAPnH і CCAPnL повинно бути переписано в ОЗП мікроконтролера до того, як прийде новий фронт або спад, які викликають наступне захоплення інформації в згаданих регістрах. Якщо це не буде виконано, нова інформація записується поверх наявної, і остання зникає.

Час, необхідний для вищезгаданих дій, обмежено тактовою частотою мікроконтролера. Запам’ятовування двох 8–бітних регістрів і скидання прапорця вимагає 9 машинних циклів (сюди входять і виклик підпрограми переривання). Таким чином, для тактової частоти 12 МГц захоплення не може відбуватися частіше, ніж один раз в 10 мкс.

В мікроконтролерах сімейства Silicon Labs також присутній цей режим. Його реалізовано аналогічно.

3.5.3.4 Режим програмованого таймера

В більшості випадків програмований таймер використовується для виклику підпрограми обробки переривання, яка повинна виконуватися з постійним заданим інтервалом. Спрощену структуру таймера PCA в режимі програмованого таймера наведено на рисунку 3.7. Користувач здійснює попереднє завантаження 16–бітного значення в регістри CCAPnH, CCAPnL. В режимі, коли на вхід модуля подано сигнал з частотою, яка дорівнює частоті задаючого генератора, яку поділено на 4 (рисунок 3.5), порівняння виконується

тричі протягом кожного машинного циклу. Для дозволу порівняння необхідно встановити в 1 біт ECOMn регістра CCAPMn (таблиця 3.11).

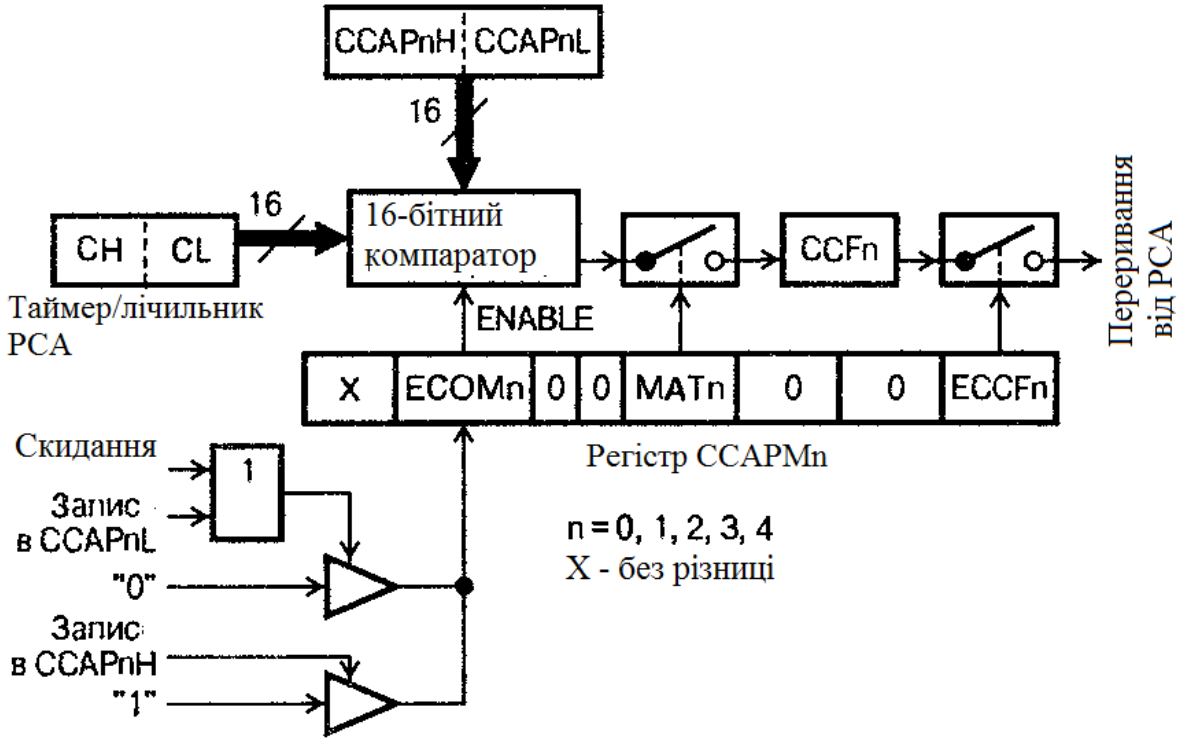


Рисунок 3.7 – Спрощена структура таймера PCA в режимі програмованого таймера

Для функціонування модуля в цьому режимі також необхідно встановити в 1 біт MATn в CCAPMn. Коли наступає збіг інформації в CCAPnH, CCAPnL з вмістом CH, CL, виробляється сигнал збігу і встановлюється прапорець CCFn. Якщо при цьому встановлено біт ECCFn, то буде викликано підпрограму обробки переривання (відповідно, якщо переривання в PCA не заборонено). Прапорець CCFn треба скинути програмно до того, як відбудеться новий збіг інформації в таймері і регістрах модуля порівняння/захоплення. Якщо підпрограма обробки переривання заносить нове значення в CCAPnH, CCAPnL, прапорець ECOMn автоматично скидається, як тільки інформація заноситься в CCAPnL. Запис CCAPnH знову встановлює цей біт в 1, дозволяючи, таким чином, роботу компаратора. За цією причиною програма користувача повинна спочатку заносити інформацію в CCAPnL, а потім в CCAPnH, а не навпаки,

щоб виключити виклик переривання в той момент, коли нове значення інформації остаточно не занесено в CCAPnL та CCAPnH.

В мікроконтролерах сімейства Silicon Labs також присутній цей режим. Його реалізовано аналогічно.

3.5.3.5 Режим високошвидкісного виходу

В режимі високошвидкісного виходу відбувається перемикання виводу CEXn з 1 в 0 і навпаки в момент збігу інформації в регістрах таймера PCA CH і CL і в регістрах модуля порівняння/захоплення CCAPnH, CCAPnL. При реалізації цього режиму вивід CEXn попередньо встановлюється в 0 або в 1. Користувач обирає, яким перепадом на ньому буде супроводжуватися рівність інформації в CH, CL з CCAPnH, CCAPnL. Крім формування перепаду на виводі CEXn, користувач може також викликати підпрограму переривання для виконання певних функцій мікроконтролера при вищезгаданому збігу значень в регістрах. З цією ціллю треба встановити прапорець ECCFn, що дозволяє переривання від PCA. Спрощену структуру таймера PCA в режимі високошвидкісного виходу показано на рисунку 3.8.

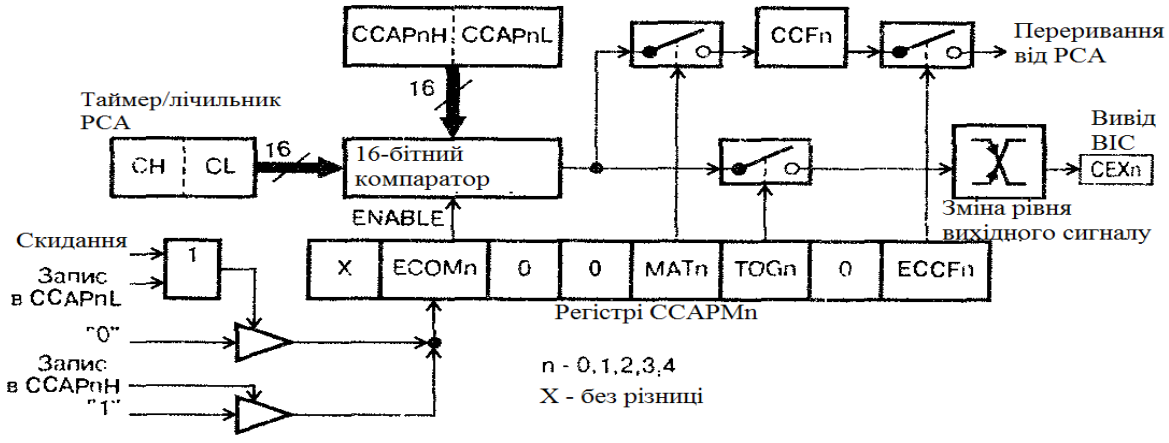


Рисунок 3.8 – Спрощена структура таймера PCA в режимі високошвидкісного виходу

В цьому режимі можливе більш швидке реагування на рівність інформації в таймері PCA і в модулі порівняння/захоплення, ніж при програмному

керуванні зовнішнім виводом в ході виконання підпрограми переривання, яка визначає факт збігу інформації. Іншими словами, в цьому режимі на роботу не впливає швидкість відпрацювання підпрограми переривання (режим програмованого таймера). Ця підпрограма потрібна в тому випадку, коли користувач хоче змінити інформацію в CCAPnH, CCAPnL. Якщо цю інформацію не міняти, то наступний сигнал з'явиться тільки тоді, коли таймер PCA дорахує до свого максимуму і почне лічбу спочатку. При частоті сигналу 16МГц частота повторення сигналів в режимі швидкісного виходу дорівнює 30,5Гц. В мікроконтролерах сімейства Silicon Labs також присутній цей режим. Його реалізовано аналогічно.

3.5.3.6 Режим вартового таймера PCA

Функцію вартового таймера може виконувати тільки четвертий модуль порівняння/захоплення (рисунок 3.9). Якщо у вартовому таймері немає необхідності, модуль можна використовувати аналогічно іншим модулям.

Вартовий таймер представляє собою об'єкт, який викликає автоматичне скидання мікроконтролера, якщо система не надішле йому керуючий сигнал підтвердження нормального функціонування. Такі таймери використовують пристрої, які працюють з електричними шумами, стрибками напруги живлення, електростатичним розрядом і т. ін., або які потребують підвищеної можливості до самовідновлення системи.

В цьому режимі в той момент, коли вміст таймера PCA дорівнює значенню в CCAP4H, CCAP4L, генерується внутрішній сигнал скидання. Для задання цього режиму потрібно встановити в 1 біт WDTE в регістрі CMOD. При цьому модуль 4 порівняння/захоплення треба налаштувати або на функціонування в режимі програмованого таймера, або в режимі високошвидкісного виходу. Спрощену структуру таймера PCA в режимі вартового таймера показано на рисунку 3.9.

В мікроконтролерах сімейства Silicon Labs також присутній цей режим. Його реалізовано аналогічно.

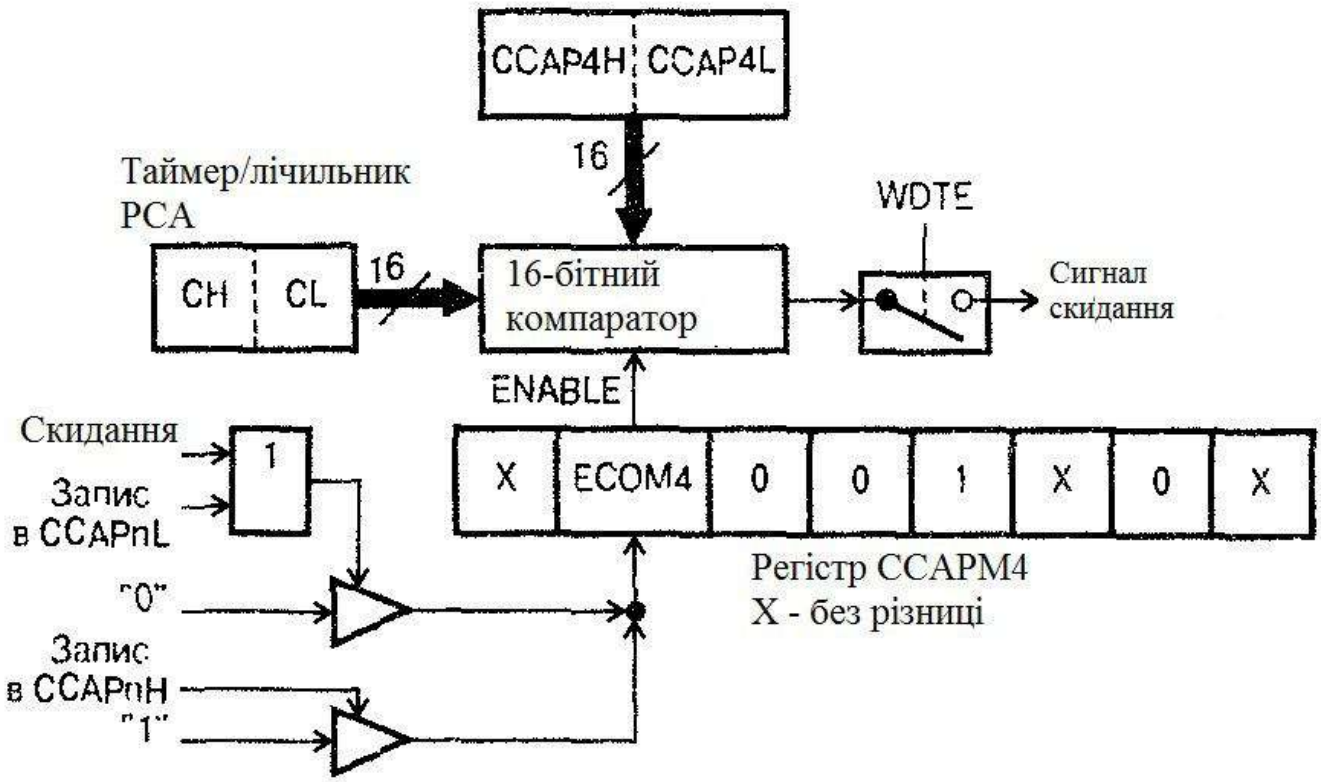


Рисунок 3.9 – Спрощена структура таймера PCA в режимі вартового таймера

Скидання при працюючому вартовому таймері не відбувається в наступних трьох випадках:

- при періодичній зміні значень в регістрах CCAP4H, CCAP4L таким чином, щоб знову записані значення до нового запису не могли співпасти зі значеннями в CH, CL;
- при періодичній зміні CH, CL з тим же результатом;
- при скиданні в 0 біта WDTE і повторним встановленням його в 1 після того, як значення CH, CL стали дорівнювати значенням в регістрах CCAP4H, CCAP4L.

Перші два випадки більше підходять для таких систем, які критичні до зависання, так як в цих випадках вартовий таймер не відключається. При цьому

другий варіант не рекомендується, якщо Т/Л РСА використовується ще будь–яким модулем порівняння/захоплення. Третій випадок найбільше підходить для практичної реалізації в системах, у яких зависання не носять критичної загрози.

3.5.3.7 Режим широтно–імпульсного модулятора

Будь–який із п’яти модулів РСА може бути запрограмовано в режим широтно–імпульсного модулятора (ШІМ). Вихід ШІМ може бути використано для перетворення цифрових даних в аналоговий сигнал з мінімальними апаратними витратами. Частота модуляції залежить від швидкості лічби таймера РСА. З 16–ти мегагерцовим кварцовим резонатором максимальна частота сигналу ШІМ не перевищує 15,6 кГц.

На рисунку 3.10 показано спрощену структуру таймера РСА в режимі ШІМ.

Для роботи в режимі ШІМ біти ECOMn і PWMn в регістрі CCAPMn повинно бути встановлено в одиницю. РСА виробляє 8–бітний ШІМ–сигнал шляхом порівняння вмісту CCAPnL і CL. Якщо CL<CCAPnL, то на зовнішньому виводі відповідного модуля порівняння/захоплення буде сигнал з нульовим рівнем, якщо CL>CCAPnL, то з одиничним.

Значення, яке записано в CCAPnL, задає шпаруватість вихідного сигналу. Для зміни значення CCAPnL без збоїв користувач повинен попередньо занести потрібне значення в CCAPnH. Це значення апаратно заноситься в CCAPnL в момент, коли CL міняє своє значення з FFH на 0, що відповідає початку нового циклу формування вихідного сигналу.

В CCAPnH можна занести будь–яке ціле число від 0 до 255, при цьому коефіцієнт заповнення змінюється від 100 до 0,4 %. Коефіцієнт заповнення розраховується за формулою:

$$K_{\text{зап}} = \frac{t_{\text{імп}}}{T} \cdot 100\% \quad , \tag{3.3}$$

де $t_{імп}$ – тривалість вихідного імпульсу; T – період слідування вихідних імпульсів.

Нульове значення $К_{зап}$ забезпечується шляхом прямого запису інформації на вивід відповідного порту. Значення CCAPnH, при якому досягається потрібний $К_{зап}$, визначається відношенням

$$CCAPnH=256\ (1 – К_{зап}), \qquad (3.4)$$

де CCAPnH – найближче ціле число до значення в правій частині виразу, $К_{зап}$ представляє собою величину в діапазоні від 0,004 до 1.

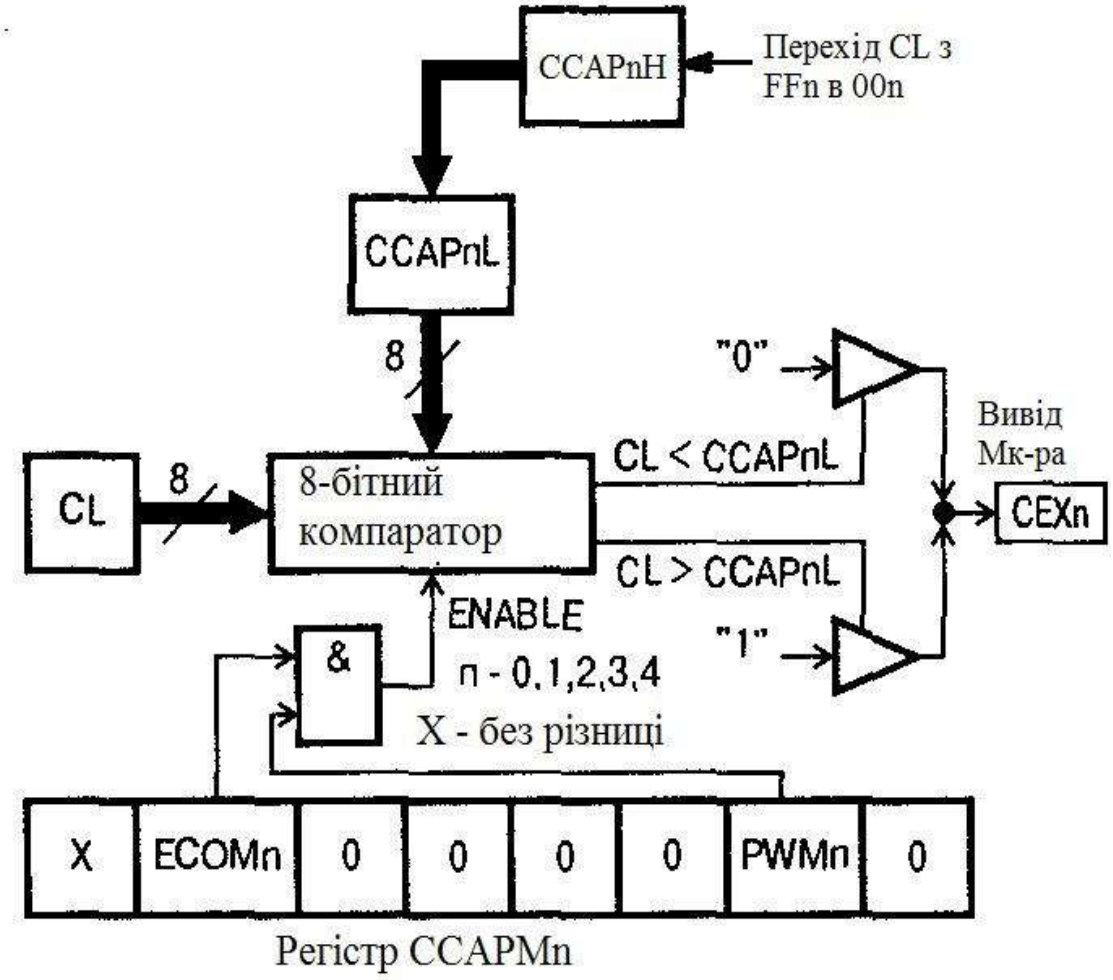


Рисунок 3.10 – Спрощена структура таймера PCA в режимі ШІМ

На рисунку 3.11 наведено залежності сигналів на виході мікроконтролера при роботі PCA в режимі ШІМ при різних значеннях інформації в CCAPnH.

В мікроконтролерах сімейства Silicon Labs також присутній режим широтно–імпульсної модуляції. Його реалізовано аналогічно.

$К_{зап}$ у відсотках	$К_{зап}$ у відносних одиницях	CCAPnH
100%	1	00
90%	0,9	26
50%	0,5	128
10%	0,1	230
0,4%	0,004	255

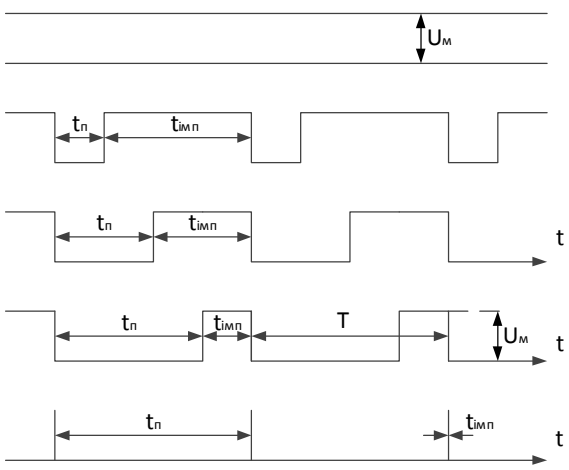


Рисунок 3.11 – Форми вихідного ШІМ–сигналу

3.5.4 Вартовий таймер

Більшість сучасних мікроконтролерів сім'ї МК51 мають вартовий таймер. Вартовий таймер – це схема, яка автоматично скидає мікроконтролер, якщо не отримує від керуючої системи сигналу, який підтверджує, що не відбулося ніякого збою. Такий пристрій використовується в системах, де є електричні завади або збої за живленням і де потрібно забезпечувати високу надійність.

Розглянемо роботу вартого таймера на прикладі мікроконтролера ADuC847. Призначення вартового таймера полягає в тому, щоб виробити скидання пристрою або його переривання, якщо ADuC847 входить в стан зависання в рамках відповідного інтервалу часу, можливо, через помилку програмування або дії електричної або радіочастотної завади. Роботу таймера може бути заборонено шляхом скидання біта WDE (дозвіл вартового таймера) в регістрі керування таймером (WDCON). В дозволеному стані схема таймера виробляє системне скидання або переривання (в залежності від значення біта

WDIR), якщо програма користувача не перевстановила біт вартового таймера (WDE) протягом визначеного часового інтервалу. Вартовий таймер представляє собою 16–бітний двійковий лічильник, який тактується власною частотою 32,768кГц. Часовий інтервал контролю може змінюватися за допомогою бітів PRE0…PRE3 в регістрі WDCON. Функціями керування і стану вартового таймера можна керувати за допомогою регістра керування таймером (WDCON) [13].

ПИТАННЯ ДЛЯ САМОКОНТРОЛЮ

1) Охарактеризуйте способи формування часових інтервалів у МПС.
2) Опишіть складові та роботу модуля Т/Л за структурною схемою.
3) Якою є роль регістрів TMOD і TCON у програмуванні МК–51?
4) Скільки МЦ потрібно для розпізнавання переходу 1 в 0? Чому?
5) Дайте характеристику кожному з режимів роботи Т/Л.
6) Назвіть варіанти вхідних сигналів Т/Л PCA.
7) Дайте визначення поняттю «вартовий таймер».
8) Опишіть роботу модуля PCA в режимі захоплення.
9) Опишіть роботу модуля PCA в режимі програмованого таймера.
10) Опишіть роботу модуля PCA в режимі високошвидкісного виходу.
11) Опишіть роботу модуля PCA в режимі вартового таймера PCA.
12) Опишіть роботу модуля PCA в режимі широтно–імпульсного модулятора.
13) Дайте характеристику таймера/лічильника Т2.
14) Чим відрізняється режим «таймера» від режиму «підрахунку зовнішніх подій»?
15) Як називається цифровий електронний пристрій, який складає основу модуля таймерів/лічильників?
16) Як за допомогою таймерів/лічильників мікроконтролера AT89C51 можна вимірювати тривалість додатних зовнішніх імпульсів?
17) Як за допомогою таймерів/лічильників мікроконтролера AT89C51 можна сформувати частоту синхронізації для послідовного порту?
18) Коли відбувається переповнення таймерів/лічильників?

4 АРХІТЕКТУРА ПАРАЛЕЛЬНИХ ПОРТІВ

4.1 Місце паралельних портів у структурі мікроконтролера

Існує два способи обміну даними між зовнішніми пристроями (ЗПР) і мікропроцесорною системою (МПС):

- паралельний, коли одночасно передаються всі біти або декілька біт слова даних;
- послідовний, коли біти слова даних пересилаються по черзі, починаючи, наприклад, з його молодшого розряду.

ЗПР зв'язуються з МПС лініями зв'язку, довжина яких при паралельному обміні обмежена і складає кілька метрів.

Мікроконтролер типу МК–51, наприклад, AT89C51, містить 4 паралельних 8–розрядних порти введення/виведення дискретної інформації: P0, P1, P2, P3 (рисунок 2.1). Ці порти можна запрограмувати.

Порти P0, P1, P2, P3 є двонаправленими портами введення/виведення і призначені для забезпечення обміну інформацією МК–ра із зовнішніми пристроями, створюючи 32 лінії введення/виведення. Кожен з портів містить фіксатор–защіпку, що являє собою восьмирозрядний регістр, який має байтову і бітову адресацію для встановлення/скидання його розрядів за допомогою відповідних команд.

Фізичні адреси фіксаторів P0, P1, P2, P3 становлять для:

- P0: 80H, при бітовій адресації: 80H … 87H;
- P1: 90H, при бітовій адресації: 90H … 97H;
- P2: A0H, при бітовій адресації: A0H … A7H;
- P3: B0H, при бітовій адресації: B0H … B7H.

Крім роботи в якості звичайних портів введення/виведення лінії портів P0…P3 можуть виконувати ряд додаткових функцій, які описано нижче.

Через порт P0:

– виводиться молодший байт адреси A0…A7 при роботі з зовнішньою пам'яттю програм і зовнішньою пам'яттю даних;

– видається з МК–ра і приймається в МК–р байт даних при роботі з зовнішньою пам'яттю (при цьому обмін байтом даних і виведення молодшого байта адреси зовнішньої пам'яті мультиплексовано в часі);

– задаються дані при програмуванні внутрішнього ПЗП, і читається вміст внутрішньої пам'яті програм.

Через порт P1:

– задається молодший байт адреси при програмуванні внутрішнього ПЗП і при читанні внутрішньої пам'яті програм.

Через порт P2:

– виводиться старший байт адреси A8…A15 при роботі з зовнішньою пам'яттю програм і зовнішньою пам'яттю даних (для зовнішньої пам'яті даних – тільки при використанні команд MOVX A, @ DPTR і MOVX @ DPTR, A, що формують 16–розрядну адресу);

– задаються старші розряди A8…A11 адреси при програмуванні внутрішнього ПЗП і при читанні внутрішньої пам'яті програм.

Кожна лінія порту P3 має індивідуальну альтернативну функцію:

P3.0 – RxD, вхід послідовного порту, який призначено для введення послідовних даних у приймач послідовного порту;

P3.1 – TxD, вихід послідовного порту, який призначено для виведення послідовних даних із передавача послідовного порту;

P3.2 – $\overline{\text{INT0}}$, який використовується як вхід зовнішнього запиту переривання;

P3.3 – $\overline{\text{INT1}}$, який використовується як вхід зовнішнього запиту переривання;

P3.4 – T0, який використовується як вхід лічильника зовнішніх подій Т/Л0;

P3.5 – T1, який використовується як вхід лічильника зовнішніх подій Т/Л1;

P3.6 – $\overline{WR}$, строб запису в зовнішню пам'ять даних, вихідний сигнал, який супроводжує виведення даних через порт P0 при використанні команд MOVX @ Ri, A і MOVX @ DPTR, A;

P3.7 – $\overline{RD}$, строб читання із зовнішньої пам'яті даних, вихідний сигнал, який супроводжує введення даних через порт P0 при використанні команд MOVX A, @ Ri і MOVX A, @ DPTR.

Альтернативна функція будь–якої з ліній порту P3 реалізується тільки в тому випадку, якщо у відповідному цій лінії фіксаторі–защіпці міститься сигнал "логічна 1". Інакше на лінії порту P3 буде присутній сигнал "логічний 0".

4.2 Використання паралельних портів введення/виведення

4.2.1 Загальні відомості

Порти P0…P3 можуть використовуватись:

- як 8–розрядні паралельні порти введення/виведення інформації;
- як 32 однорозрядні лінії введення/виведення;
- при роботі з зовнішньою пам’яттю програм і даних;
- в режимі альтернативних функцій (8 ліній порту P3);
- при програмуванні та перевірці РПП.

Сигналом “RESET” в регістри–защіпки всіх портів автоматично записуються одиниці, що налаштовує усі 32 лінії портів на введення інформації, а вісім ліній порту P3, крім цього, на режим “альтернативних” функцій.

4.2.2 Особливості роботи порту P0

На рисунку 4.1 наведено спрощену структурну схему і–го розряду порту P0, яка пояснює його роботу в двох основних режимах:

– робота з зовнішньою пам’яттю програм або даних (ЗПП або ЗПД);

– робота в якості порту введення/виведення.

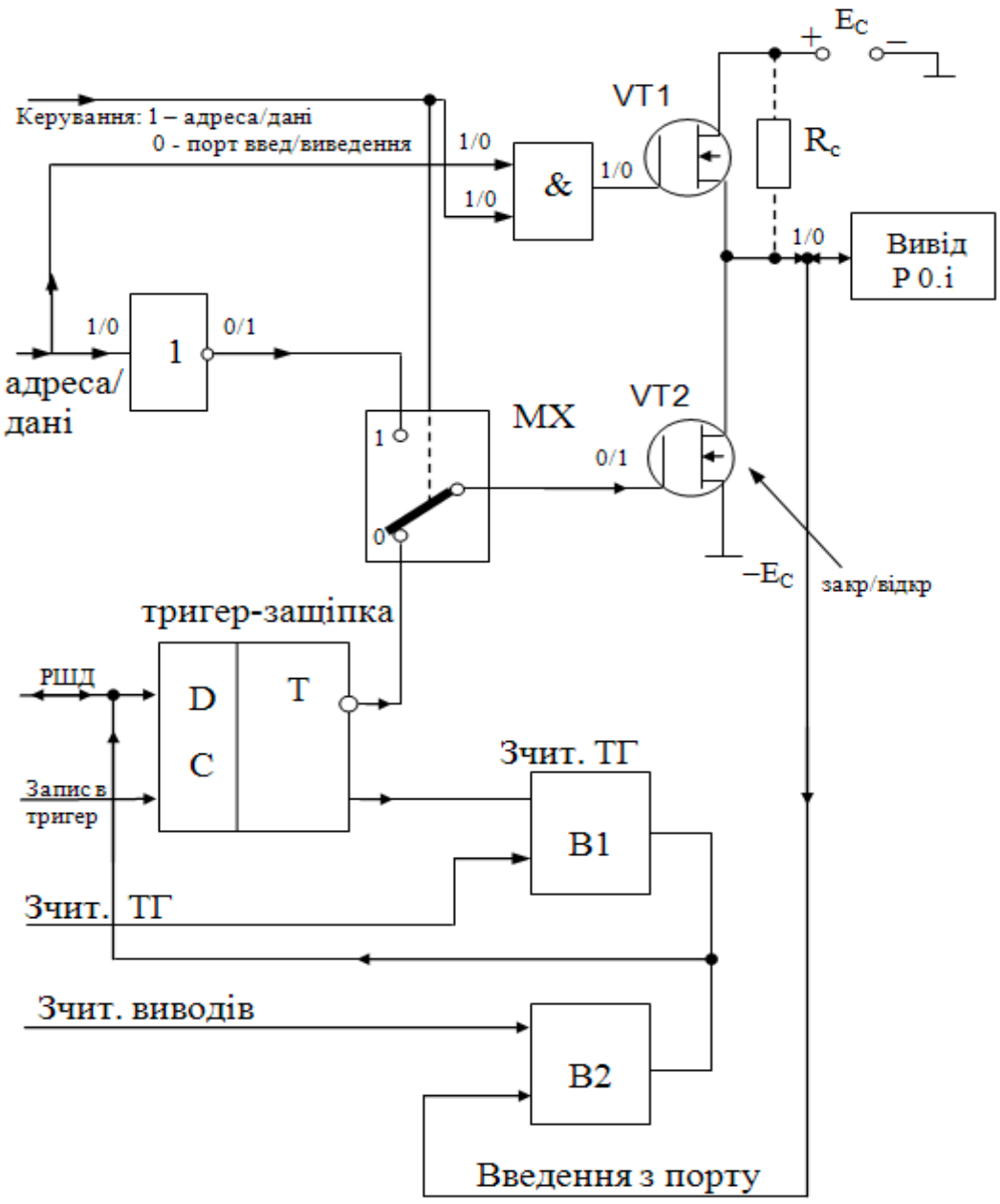


Рисунок 4.1 – Спрощена структурна схема і–го виводу порту P0

В залежності від типу команд, які виконуються в даний час (робота з портами або із зовнішньою пам’яттю (ЗП)), внутрішній сигнал «керування» приймає значення: “логічний 0” – Р0 використовується як порт введення/виведення; “логічна 1” – порт використовується для роботи з ЗП.

4.2.2.1 Особливості роботи Р0 із ЗП (ЗПП або ЗПД)

При роботі із ЗП сигнал “керування” дорівнює “логічній 1”. Мультиплексор МХ знаходиться у верхньому положенні. На вхід інвертора та на верхній вхід кон’юнктора надходить значення і–го розряду молодшого байта адреси ЗП: 1/0. Після інвертора воно приймає значення 0/1. У зв’язку з тим, що на другому нижньому вході кон’юнктора присутній сигнал “логічна 1”, вихідний сигнал кон’юнктора повторює значення і–го розряду адреси: 1/0. Таким чином, на затворах транзисторів VT1 і VT2 присутні дві протилежні комбінації логічних сигналів:

1/0 на затворі VT1
0/1 на затворі VT2
} при значенні і–го біта адреси 1/0.

В схемі використано МОН–польові транзистори з каналом типу n. Якщо на затворі такого транзистора присутній сигнал “логічна 1”, то транзистор відкрито, а якщо сигнал “логічний 0” – закрито. Отже, якщо в і–му біті адреси видається сигнал “логічна 1”, то VT1 – відкрито, а VT2 – закрито, і з і–го виводу порту 0 знімається також сигнал “логічна 1”. Якщо ж у і–му біті адреси присутній сигнал “логічний 0”, то VT1 – закрито, а VT2 – відкрито. З виходу при цьому знімається сигнал “логічний 0”. Сказане ілюструє таблиця 4.1.

Виводи порту Р0 при роботі з ЗП використовуються в режимі мультиплексування: спочатку видається молодший байт адреси ЗП, а потім через виводи видаються або приймаються дані (видаються при роботі з ЗПД, а приймаються при роботі з ЗПП або ЗПД). Необхідно звернути увагу на те, що

при зчитуванні ЗПП або ЗПД на нижній транзистор VT2 апаратно автоматично подається сигнал “логічний 0”, VT2 закривається і не шунтує і–й вивід P0, через який відбувається зчитування з пам’яті. Необхідно відзначити також, що при виконанні команд роботи із зовнішньою пам'яттю в тригери–защіпки порту P0 автоматично записуються сигнали “логічна 1”.

Таблиця 4.1 – Значення вихідних сигналів транзисторів

Значення і–го біта молодшого байта адреси ЗП	Сигнали на затворах транзисторів		Стан транзисторів		Значення вихідного сигналу
	VT1	VT2	VT1	VT2	
0	0	1	закрито	відкрито	0
1	1	0	відкрито	закрито	1

4.2.2.2 Особливості роботи P0 в якості порту введення/виведення

Сигнал “керування” дорівнює нулю (рисунок 4.1). МХ знаходиться в нижньому положенні. На затворі VT1 постійно присутній сигнал “логічний 0” і він закритий.

4.2.2.2.1 Виведення даних через P0

Через РШД внутрішнім сигналом “запис в тригер” значення і–го біта, який виводиться, запам’ятовується в тригері–защіпці. При виведенні значення “логічного 0” сигналом високого рівня, що знімається з виходу $\overline{Q}$ тригера–защіпки, транзистор VT2 відкривається (VT1 закритий постійно), і з і–го виводу порту P0 видається значення “логічного 0”. При виведенні значення “логічної 1” сигналом низького рівня, що знімається з виходу $\overline{Q}$ тригера–защіпки, транзистор VT2 закривається (VT1 закритий постійно), а і–й вивід P0 виявляється “обірваним” – знаходиться в z–стані. Тому для виведення сигналу “логічної 1” через і–й вивід, коли P0 працює як порт виведення, необхідно

включати зовнішній резистор стоку R_c між $+E_C$ та і–м виводом порту Р0 (рисунок 4.1). Такий резистор називається «таким, що підтягує вивід до джерела $+E_C$».

4.2.2.2.2 Введення даних через Р0

При введенні даних відповідний тригер–защіпку треба встановити у 1. У протилежному випадку, якщо його скинуто в 0, VT2 – відкритий і шунтує і–й вивід, тобто постійно буде вводитись значення “логічний 0”.

В тригерах–защіпках значення “логічна 1” можна встановити різними засобами:

- сигналом “RESET” автоматично у всі тригери–защіпки записується сигнал “логічна 1”;
- при виконанні команд, які працюють із ЗП (ЗПП чи ЗПД), автоматично в тригери–защіпки записується сигнал “логічна 1”;
- командами, які працюють із портом Р0 в цілому;
- командами, які працюють з окремими бітами Р0.

При виконанні команд “введення” формується внутрішній сигнал “зчитування виводів”, який через буфер В2 передає значення і–го виводу Р0 на РШД.

4.2.2.2.3 Спеціальний режим використання порту Р0: режим “зчитування – модифікація – запис”

В цьому режимі порт Р0 працює в тих випадках, коли при виконанні команд порт є одночасно операндом і місцем призначення результату. При цьому інформація зчитується не з зовнішніх виводів, а з тригерів–защіпок. Це відбувається через буфер В1 (рисунок 4.1) при формуванні внутрішнього сигналу “зчитування ТГ”. В середині мікроконтролера відбувається

модифікація вмісту і–го розряду тригера–защіпки відповідно до команди, що виконується, а потім запис нового значення знову в тригер–защіпку.

Якщо зчитувати інформацію не з тригерів–защіпок, а з зовнішніх виводів, то можлива помилка. Наприклад, якщо одиничний вихідний сигнал керує якимось потужним виконавчим елементом, то цей сигнал може падати за рівнем і при зчитуванні з виводу P0, а не із защіпки, може сприйматися як сигнал “логічний 0” замість сигналу “логічна 1”.

4.2.3 Особливості роботи порту P1

Порт P1 не використовується при роботі з зовнішньою пам'яттю, тому на відміну від портів P0 і P2 є “чистим” портом введення/виведення.

Структуру P1 наведено в [2,3] і вона дуже подібна на розглянуту вище схему порту P0 (рисунок 4.1).

Порт P1 містить 8 тригерів–защіпок, буфери введення/виведення, декілька МОН–транзисторів, один із яких виконує функцію зовнішнього (“підтягуючого”) резистора Rc, групу додаткових логічних елементів для підвищення швидкодії.

Порт P1 може використовуватися в режимі “зчитування – модифікація – запис”, який описано при розгляді структури і особливостей функціонування порту P0.

Крім застосування в якості порту введення/виведення порт P1 застосовується при програмуванні та перевірці РПП [2,3].

Кожна з восьми ліній порту P1 може програмуватись незалежно одна від одної на введення або виведення інформації.

4.2.4 Особливості роботи порту P2

Структуру порту P2 наведено в [2,3] і виконано аналогічно за схемами портів P0, P1. Крім введення/виведення порт P2 використовується для видачі старшого байта адреси при роботі з зовнішньою пам’яттю, тому його схема

також, як і схема порту P0, містить мультиплексор. Порт P2 бере участь у програмуванні та перевірці РПП і також може використовуватися у режимі “зчитування – модифікація – запис”, який описано вище. Кожна з восьми ліній порту P2 може програмуватись незалежно одна від одної на введення або виведення інформації.

4.2.5 Особливості роботи порту P3

Спрощену структуру i–го виводу (i = 0, 1, …, 7) порту P3 наведено на рисунку 4.2.

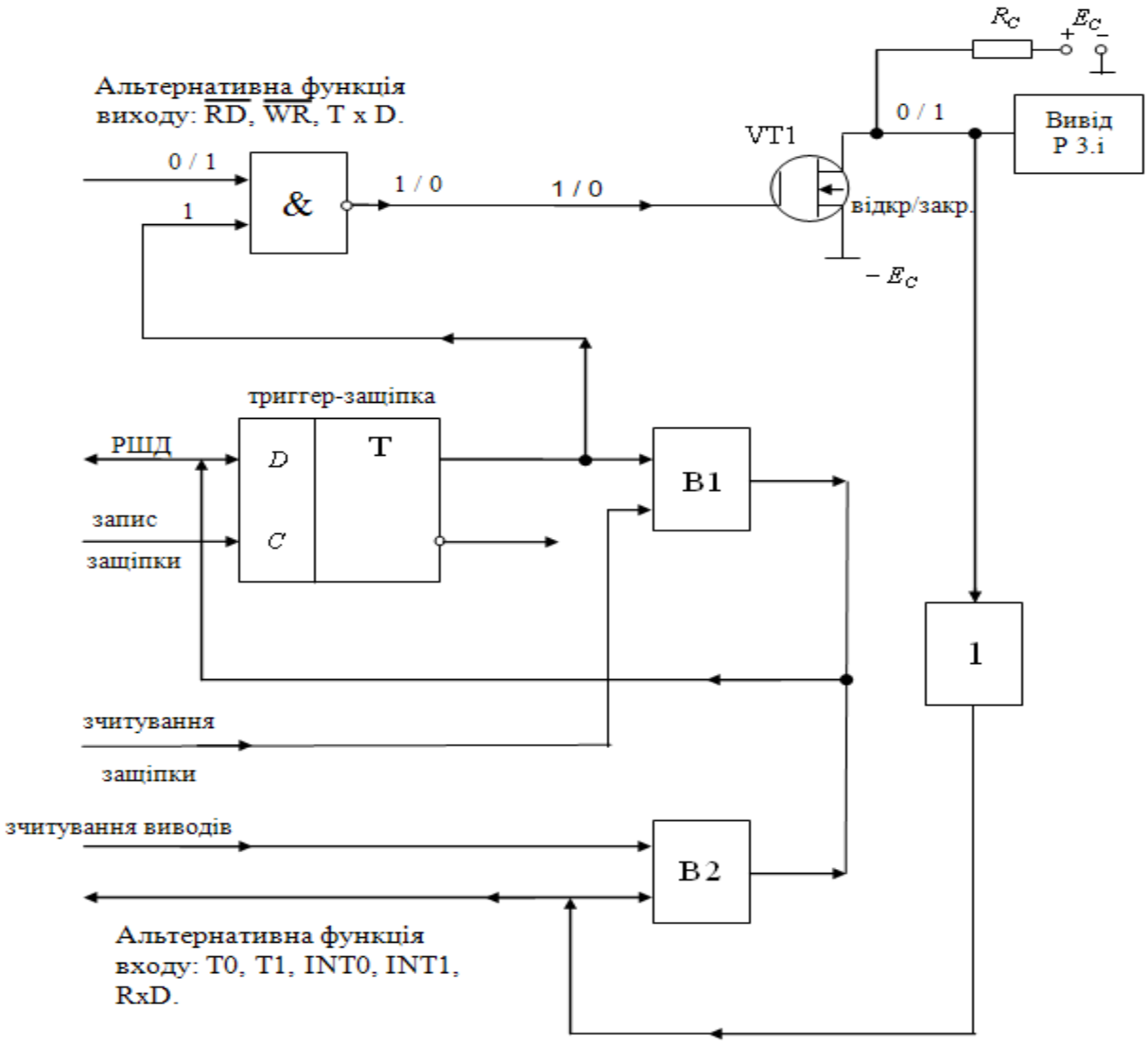


Рисунок 4.2 – Спрощена структурна схема порту P3 (i–го виводу)

Р3 може працювати як двонаправлений 8–розрядний порт введення/виведення інформації або виконувати альтернативні функції.

4.2.5.1 Робота Р3 в режимі “альтернативних функцій”

Р3 може виконувати три альтернативні функції виходу і п'ять альтернативних функцій входу (рисунок 4.2).

4.2.5.1.1 Виконання портом Р3 альтернативних функцій виходу

Порт Р3 може виконувати три альтернативні функції виходу. При цьому використовуються такі його виводи:

- *P3.1 (TxD)* – вихід послідовного порту, який призначено для виведення послідовних даних із передавача послідовного порту;
- *P3.6 (WR)* – вихід, з якого знімається сигнал $\overline{WR}$ – строб запису у ЗПД при виконанні команд MOVX @ Ri, A; MOVX @DPTR, A;
- *P3.7 (RD)* – вихід, з якого знімає сигнал $\overline{RD}$ – строб зчитування з ЗПД при виконанні команд MOVX A, @Ri; MOVX A, @DPTR.

Щоб одна з 3–х ліній Р3 могла виконувати названі функції, у відповідний тригер–защіпку регістра порту треба записати сигнал “логічна 1”, який з одиничного виходу тригера подається на нижній вхід кон’юнктора (рисунок 4.2).

Якщо значення альтернативної функції виходу 0/1, то після елемента І–НЕ воно перетвориться в 1/0 і поступає на затвор МОН–транзистора VT1 з n–каналом. Якщо на вході транзистора присутній сигнал “логічна 1”, то він відкритий, а якщо сигнал “логічний 0” – то закритий.

Таким чином, з виходу транзистора і відповідного виходу порту знімається значення 0/1, яке дорівнює значенню альтернативної функції.

4.2.5.1.2 Виконання портом Р3 альтернативних функцій входу

Порт Р3 може виконувати п’ять альтернативних функцій входу. При цьому використовуються такі його виводи:

- *P3.0 (RxD)* – вхід послідовного порту, який призначено для введення послідовних даних у приймач послідовного порту;
- *P3.2 (INT0)* – використовується як вхід зовнішнього запиту на переривання;
- *P3.3 (INT1)* – використовується як вхід зовнішнього запиту на переривання;
- *P3.4 (T0)* – вхід лічильника зовнішніх подій Т/С0;
- *P3.5 (T0)* – вхід лічильника зовнішніх подій Т/С1.

В схемах вказаних п’яти ліній порту Р3 сигнал “альтернативна функція виходу” приймає одиничне значення. В тригери–защіпки порту треба записати сигнал “логічна 1”. Сигналом “логічний 0” на затворі вихідний транзистор VT1 закрито і він не шунтує вхід порту (рисунок 4.2).

4.2.5.2 Робота Р3 у якості порту виведення

У трьох розрядах порту Р3: Р3.1, Р3.6 і Р3.7, які використовуються у якості альтернативних функцій виходу, внутрішній сигнал “альтернативна функція виходу” дорівнює “логічній 1”, якщо в даний момент не виконується команда MOVX і немає виведення даних із передавача послідовного порту. В інших п'ятьох розрядах Р3 вказаний сигнал постійно приймає значення “логічна 1”. У цьому випадку сигналом на виході порту керує відповідний тригер–защіпка. Якщо тригер встановлено в 1, то Р3.i = 1, а якщо тригер знаходиться в 0, то Р3.i = 0.

4.2.5.3 Робота Р3 в якості порту введення

В цьому випадку у відповідний тригер–защіпку треба записати сигнал “логічна 1”, тоді вихідний транзистор VT1 буде закрито і він не шунтує і–й вивід порту. Внутрішнім сигналом “зчитування виводів” (введення) через буфер В2 сигнал, який вводиться, передається на резидентну шину даних (РШД).

Аналогічно іншим портам Р3 може використовуватись в режимі “зчитування – модифікація – запис” (описано при розгляді роботи порту Р0).

Варто підкреслити, що кожен біт порту Р3 може програмуватися незалежно один від одного і використовуватися, як вхід чи вихід, або виконувати одну з альтернативних функцій.

4.3 Розширення резидентної (внутрішньої) системи введення/виведення (РСВВ/ВИВ)

Для сполучення розглянутого МК–ра з об'єктом, який має велике число входів/виходів, можна розширити резидентну систему введення/виведення, підключивши до мікроконтролера необхідну кількість зовнішніх портів. Таке розширення може бути виконано, наприклад, такими способами:

– з використанням стандартного розширювача введення/виведення (РВВ), наприклад, КР580ВР43;

– із застосуванням інтерфейсних великих інтегральних схем (ВІС) типу КР580ВВ55, КР580ВВ51.

Нижче зупинимося на другому способі.

У наведеному на рисунку 4.3 прикладі до МК–ра підключено 64 мікросхеми КР580ВВ55, що є паралельними програмованими інтерфейсами (ППІ).

Кожен ППІ містить три 8–розрядних порти введення–виведення інформації: порт А (РА7…РА0); порт В (РВ7…РВ0) і порт С (РС7…РС0). Це

дозволяє організувати (64х3) = 192 8–розрядних порти введення/виведення. Для вибору (адресації) усередині ППІ одного з 3–х портів або регістра керуючого слова (РКС) використовуються два його входи: А1, А0. Переведення одного з обраних ППІ в активний стан здійснюється нульовим сигналом на вході CS (вибір кристала).

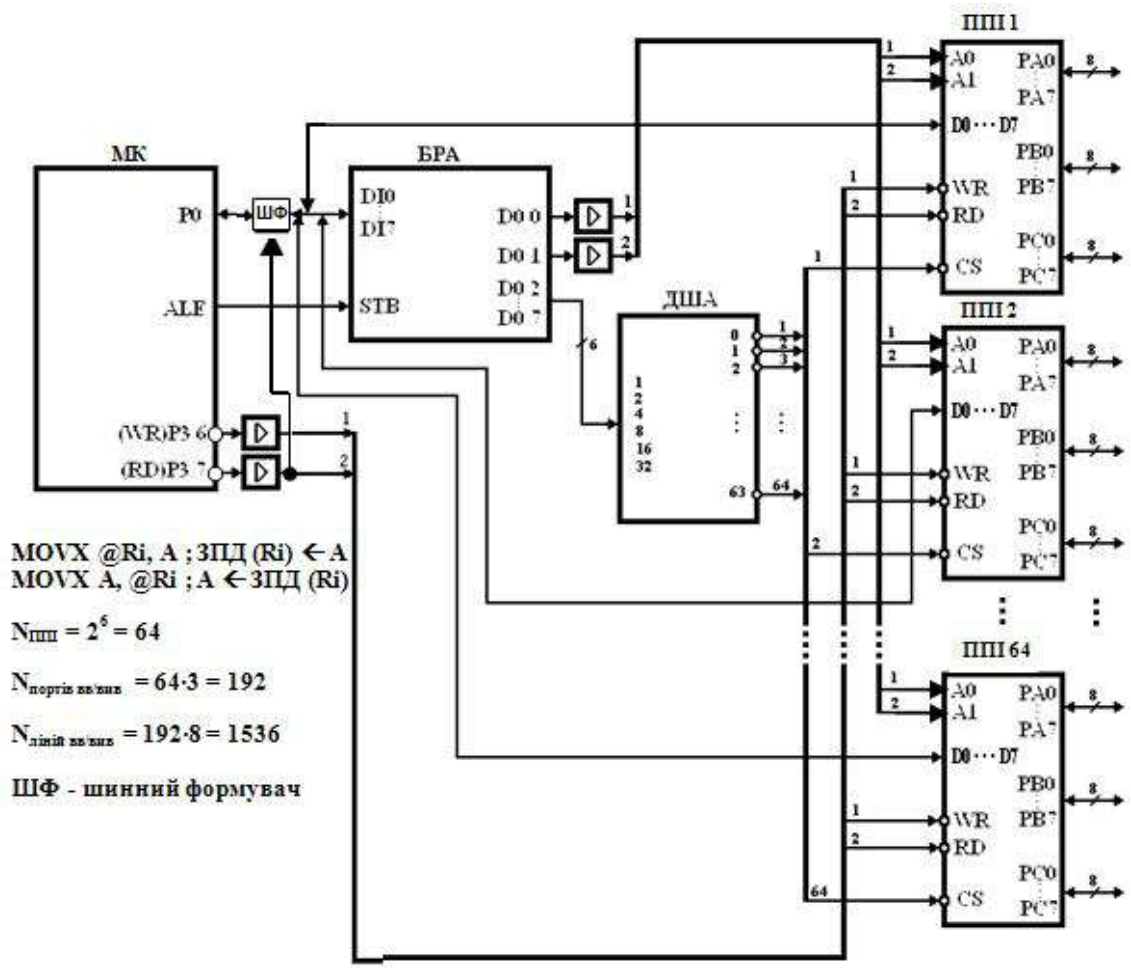


Рисунок 4.3 – Розширення РС ВВ/ВИВ за допомогою мікросхем КР580ВВ55

Звернення до інтерфейсів виконується командами роботи з ЗПД:

MOVX @Ri, A; ЗПД (Ri) ← A – при виведенні;

MOVX A, @Ri; A ← ЗПД (Ri) – при введенні даних.

При цьому порт P0 МК–ра використовується у режимі “мультиплексування”.

Спочатку на виходи P0 видається вміст регістра Ri (i = 0,1), який запам'ятовується (“защіпується”) у зовнішньому буферному 8–розрядному регістрі адреси (БРА). Потім через лінії P0 здійснюється обмін інформацією між МК і обраним портом ППІ. Два молодших розряди адреси з виходу БРА через підсилювачі надходять на входи A1, A0 ППІ і вибирають у ньому необхідний порт або РКС (при програмуванні ППІ). Інші шість старших розрядів з виходу БРА подаються на входи дешифратора адреси (ДША), який перетворює шестирозрядний паралельний двійковий код (ДК) у шестидесятичотирьохпозиційний унітарний код з активним нульовим рівнем. В залежності від комбінації ДК на входах ДША активний сигнал низького рівня з'являється тільки на одному з виходів ДША, номер якого відповідає десятковому еквіваленту вхідного ДК. Цей сигнал подається на вхід CS відповідного ППІ і підключає його виходи D0…D7 до ліній порту P0. Інші, не обранні ППІ, відключаються від P0 (переводяться у 3–й, високоімпедансний стан). Шинний формувач служить для підсилення сигналів, що знімаються з виходів порту P0.

При використанні команд MOVX @DPTR, A; MOVX A, @DPTR і деякому ускладненні схеми число ППІ, які підключаються, а отже число портів введення/виведення, може бути значно збільшено.

З виходів P3.6 та P3.7 МК–ра знімаються керуючі сигнали, які визначають напрямок обміну інформацією між МК–ром та обраним портом або РКС ППІ, який є активним (його обрано сигналом $\overline{CS}$).

Якщо МК–р записує дані у ППІ, то WR (P3.6) = 0, а RD (P3.7) = 1, а якщо читає дані з ППІ, то WR = 1, а RD = 0.

Обмін інформацією між ППІ та МК–ром здійснюється через порт P0, який працює у режимі “мультиплексування”: спочатку через P0 видається молодший байт адреси ЗПД, потім здійснюється обмін (читання або запис).

4.4 Розвиток архітектури паралельних портів в сучасних мікроконтролерах сімейства МК–51

4.4.1 Паралельні порти на прикладі сімейства мікроконтролерів SiLab (Cygnal)

Мікроконтролери сімейств МК–51 мають різну кількість стандартних 8–ми розрядних портів введення/виведення (від 1 до 8). Нагадаємо, що стандартний мікроконтролер 8051 має чотири порти (Ports 0,1,2 і 3). Порти, що не мають зовнішніх виводів, можуть бути використані, як регістри загального призначення. Порти введення/виведення, наприклад, SiLab (Cygnal) поводять себе так само, як порти стандартного 8051, але мають додаткові функціональні якості.

Кожна лінія порту введення/виведення може бути налаштована як звичайна вхідна або вихідна лінія, як лінія з "третім" високоімпедансним станом або як лінія з "відкритим стоком" через відповідні регістри (Port Configuration Registers) PRT0CF, PRT1CF, PRT2CF, PRT3CF. Плюс до цього, підтяжки рівнів "Pull–Up", які зазвичай використовуються в стандартних 8051 мікроконтролерах, можуть бути програмно заборонені з метою додаткового енергозбереження.

Безсумнівним досягненням архітектури мікроконтролерів фірми Cygnal є наявність функціонального вузла "Crossbar", що виконує роль програмно–керованого комутаційного вузла, який перемикає периферійні внутрішні ресурси на лінії введення/виведення портів 0,1 і 2. Іншими словами, вузол "Crossbar" – це комутатор внутрішніх ресурсів. Це нововведення значно підвищує гнучкість розробки мікроконтролерної системи. Внутрішні периферійні вузли, такі як таймери, послідовні інтерфейси, переривання, входи запуску АЦП, виходи компараторів та інші цифрові сигнали, а в деяких сімействах і аналогові входи АЦП, можуть бути програмно призначені на

виводи вищевказаних портів. Налаштування портів здійснюється шляхом програмування спеціальних регістрів XBR0, XBR1 і XBR2 і так званої таблиці пріоритетного декодування (Priority Decode Table). Рисунок 4.4 ілюструє функціональне призначення Crossbar (на прикладі двох сімейств C8051F0xx і C8051F018–9).

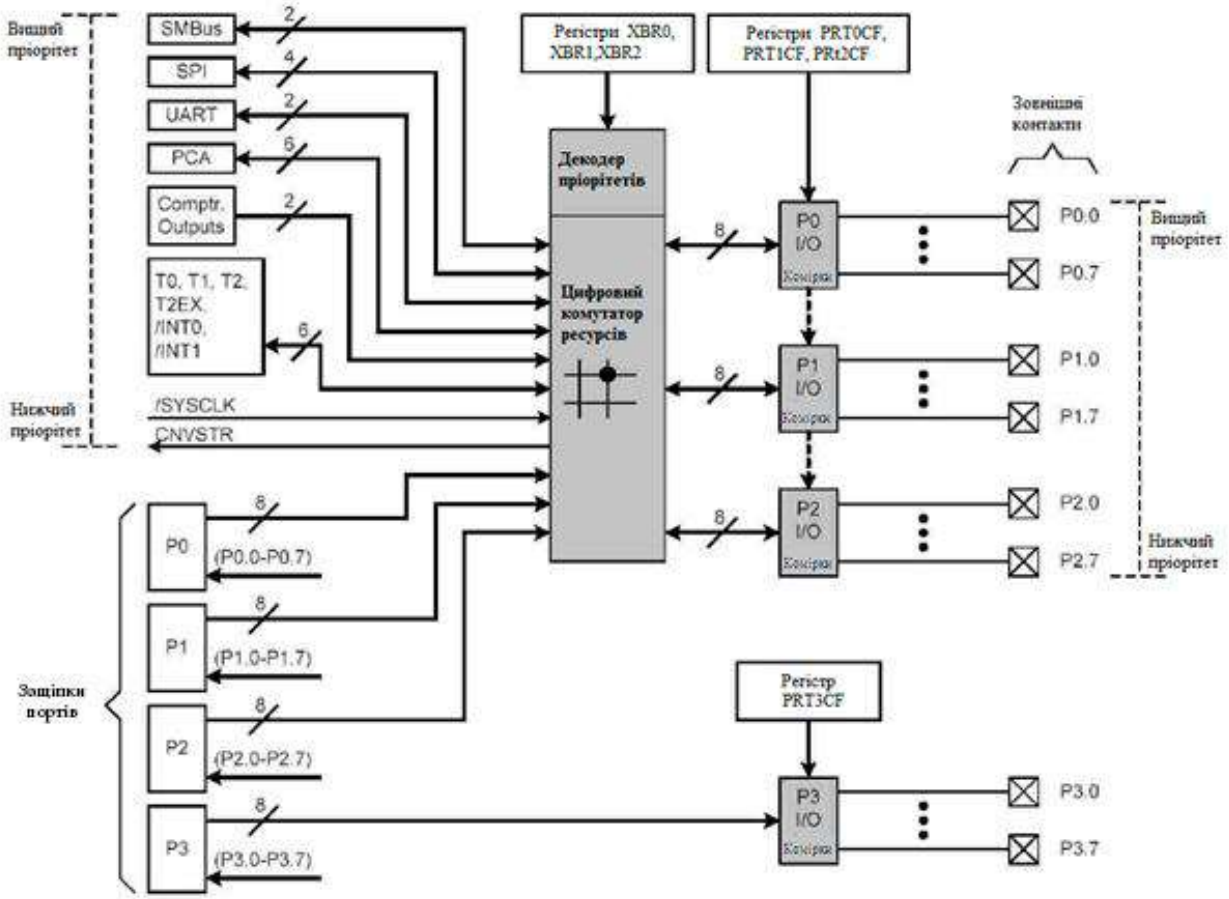


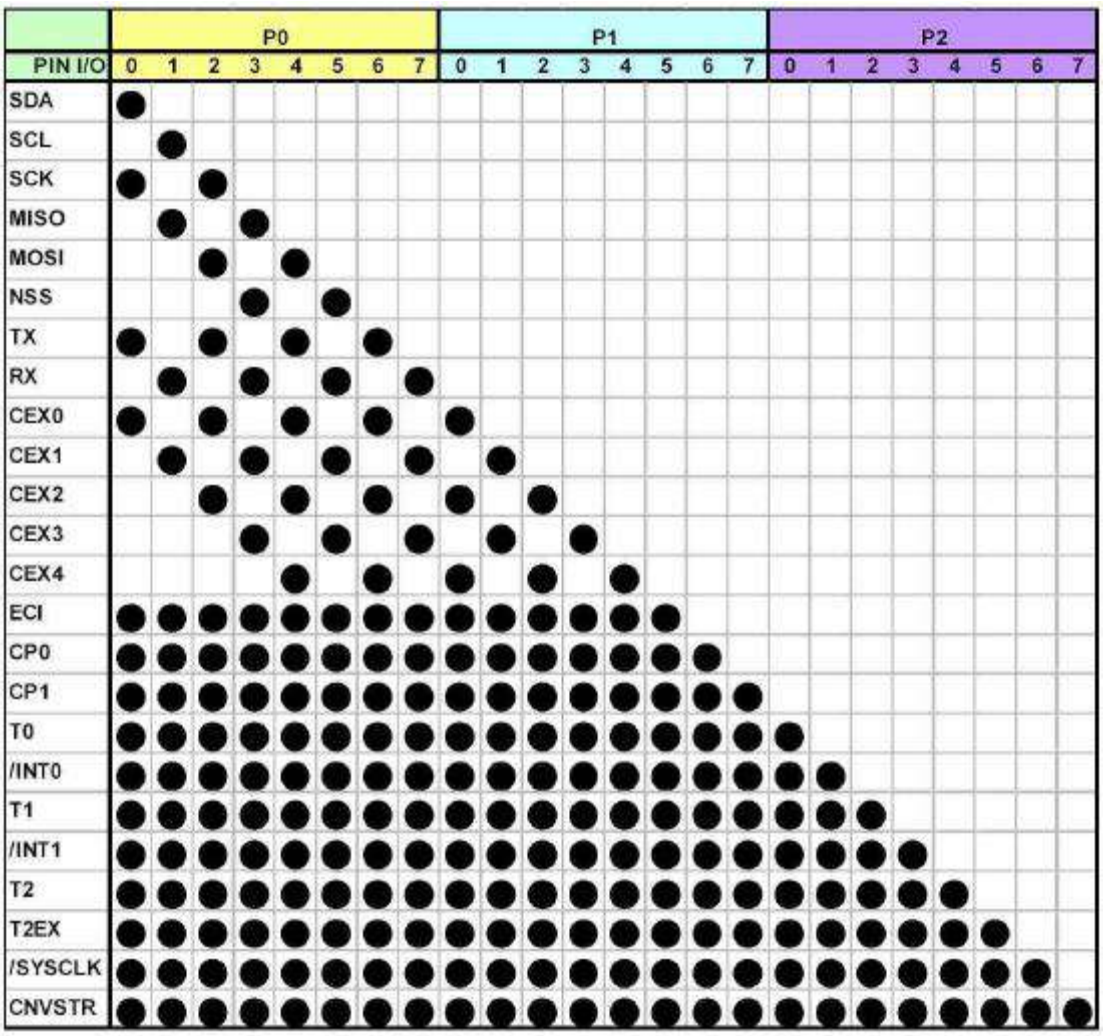
Рисунок 4.4 – Crossbar – комутатор ресурсів

Головним завданням розробників комутатора ресурсів CrossBar було створення гнучкої і зручної архітектури, що дозволяє проектувальнику контролерів оперативно вибирати і перерозподіляти значну кількість внутрішніх аналогових і цифрових ресурсів на обмежене число зовнішніх виводів мікроконтролера.

Таблиця пріоритетного декодування комутатора ресурсів індивідуальна для кожного сімейства мікроконтролерів через різний склад внутрішньої

периферії. Однак, методика вибору периферії однакова для всіх сімейств. В таблиці 4.2, як приклад, показано таблицю пріоритетного дешифратора комутатора ресурсів сімейств C8051F0xx і C8051F018–9.

Таблиця 4.2 – Приклад пріоритетного дешифратора комутатора ресурсів



В таблиці по вертикалі перераховано виводи всіх периферійних пристроїв, які можуть бути підключені до входів/виходів портів P0...P2. Виводи периферійних пристроїв перераховано в порядку зменшення пріоритетів. Наприклад, найвищий пріоритет має інтерфейс SMBus, що має сигнали SDA і SCL. Потім йде інтерфейс SPI, що має сигнали SCK, MISO, MOSI і NSS. Далі йдуть послідовний порт UART з сигналами TX і RX, потім програмований таймер/лічильник PCA, що має виходи CEX0, ..., CEX4 і т. ін.

Як вже було сказано вище, вибір периферійних пристроїв здійснюється шляхом програмування спеціальних регістрів XBR0, XBR1 і XBR2. Ці регістри містять біт (або біти), що визначають активність кожного з наведених в таблиці 4.2 периферійних пристроїв. Наприклад, включення до складу або виключення зі складу периферійних пристроїв інтерфейсів SMBus, SPI і UART здійснюється встановленням або скиданням одного біта для кожного інтерфейсу. Для програмованого масиву – лічильника PCA є три біти, що визначають кількість комутатора каналів, які підключаються.

Методика визначення активної периферії і співвіднесення її до конкретних портів введення/виведення наступні:

1. Проектувальник в регістрах XBR0, XBR1 і XBR2 встановлює в логічну одиницю біти, що відповідають за активність тих чи інших периферійних пристроїв. Біти периферійних пристроїв, які не використовуються, скидаються в логічний 0.

2. Призначення виводів активних периферійних пристроїв завжди починається з молодших бітів порту P0. Далі, при повному заповненні порту P0, починається заповнення порту P1, а, якщо його не вистачить, то і P2. Заповнення відбувається у відповідності з пріоритетом зі зсувом в бік вільних молодших виводів. Наприклад, якщо пристрій SMBus активний, то його сигнали займають молодші розряди порту P0: SDA – P0.0; SCL – P0.1. Якщо ж пристрій SMBus пасивний (вимкнений), а активний наступний пристрій – SPI, то його сигнали займають молодші розряди: SCK – P0.0; MISO – P0.1; MOSI–P0.2; NSS – P0.3. Якщо і пристрій SPI пасивний, а активний пристрій UART, то його сигнали займуть молодші розряди: TX – P0.0; RX – P0.1.

3. Наступні активні пристрої завжди займають наступні молодші вільні виводи портів P0 ... P2. Наприклад, якщо першим активним пристроєм був інтерфейс SPI, і його сигнали зайняли молодші розряди P0: SCK – P0.0; MISO –

P0.1; MOSI–P0.2; NSS – P0.3, а наступний активний пристрій – PCA із запрограмованим одним каналом, його канал буде призначено на перший наступний вільний вивід: CEX0 – P0.4.

4. Виводи, що залишилися вільними після призначення активних периферійних пристроїв, можуть бути використані як програмовані входи/виходи загального призначення.

5. Після призначення активних периферійних пристроїв необхідно дозволити роботу (встановити активність) Crossbar шляхом встановлення біта XBARE в регістрі XBR2. Поки Crossbar не є активним, всі лінії портів залишаються в режимі введення незалежно від призначень регістрів XBRn.

Початкові характеристики портів введення/виведення встановлюються в регістрах конфігурації портів (Port Configuration Registers): PRTOCF, PRT1CF, PRT2CF і PRT3CF. Кожен вивід може бути призначено як вивід "з відкритим стоком" або як тристабільний вивід. Таке призначення потрібно навіть для ліній введення/виведення інтерфейсів, які програмуються регістрами XBRn. Слід пам'ятати, що тільки лінії SDA і SCL інтерфейсу SMBus і лінія RX в режимі 0 інтерфейсу UART автоматично встановлюються як лінії "з відкритим стоком". Біт WEAKPUD в регістрі XBR2 відповідає за "слабку підтяжку" вихідного рівня. Якщо цей біт скинуто в 0, то "слабка підтяжка" виходів здійснюється для всіх ліній портів, встановлених в режим "з відкритим стоком". З іншого боку, цей біт не діє на лінії введення/виведення, які встановлено як тристабільні. Крім того, "слабка підтяжка" вимикається для виводів, що у стані логічного нуля, для запобігання непотрібних втрат потужності. На рисунку 4.5 показано функціональну схему однієї лінії порту.

Як видно з цієї функціональної схеми, так звана "слабка підтяжка" (weakpud) являє собою відкритий польовий транзистор VT3 з високим опором каналу у відкритому стані.

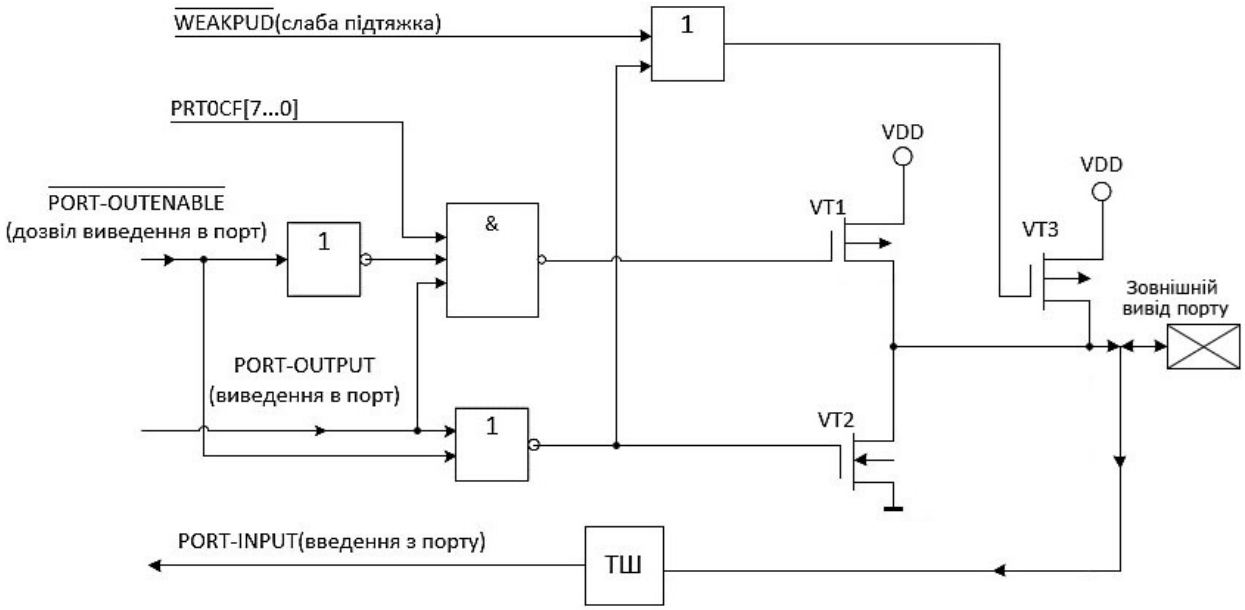


Рисунок 4.5 – Функціональна схема лінії порту

Мікроконтролери фірми Silicon Laboratories можуть мати від 1 до 5 8–ми розрядних портів введення / виведення (в залежності від типу корпусу), які можуть використовуватися як лінії введення / виведення периферійних вбудованих пристроїв або як порти введення / виведення загального призначення. Звернення до кожного такого порту здійснюється через відповідний регістр SFR (P0, P1, ...), який доступний як побайтно, так і побітно. При записі в порт, відповідні значення запам'ятовуються і виводяться на відповідні виводи. При читанні, логічні рівні портів можуть бути прочитані незалежно від призначень, зроблених у регістрах XBRn. Винятком є дії з інструкціями, які виконують "читання – модифікацію – запис" *(read–modify–write):* ANL, ORL, XRL, JBC, CPL, INC, DEC, DJNZ. При виконанні цих інструкцій стан регістра порту (а не стан вивода) читається, модифікується і записується назад в регістр.

У деяких мікроконтролерах, що мають корпуси з обмеженою кількістю виводів, деякі з портів можуть не мати своїх виводів. Наприклад, порти P2 і P3 не мають виводів в мікроконтролерах F001/06/11/16, порти P1, P2 і P3 не мають

виводів в мікроконтролерах F002/07/12/17. Однак ці порти залишаються доступними для ядра CIP–51. У цьому випадку, розробнику слід дбати самому про коректне написання програми при використанні, наприклад, команд MOV, CLR або SET, при яких адресується безпосередній біт порту.

4.4.2 Програмування паралельних портів

4.4.2.1 XBR0 – Регістр 0 комутатора ресурсів CrossBar

Назва регістра:	XBR0 – Port I/O CrossBar Register 0		
SFR–адреса:	0xE1	Значення після скидання:	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
CP0OEN	ECIE	PCA0ME			UARTEN	SPI0OEN	SMB0EN
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Біт 7: CP0OEN – Compator 0 Output Enable Bit – біт дозволу з'єднання виходу компаратора 0 на зовнішній вивід (1 – дозволено).

Біт 6: ECIE – PCA0 Counter Input Enable Bit– біт дозволу з'єднання входу 16–бітного модулю порівняння/захоплення 0 на зовнішній вивід (1 – дозволено).

Біти 5–3: PCA0ME – PCA0 Module I / O Enables Bits – біти дозволу з'єднання виходів модулів PCA на зовнішні виводи:

000 – заборонено всі;

001 – дозвіл тільки для одного першого;

010 – дозвіл тільки двох перших;

011 – дозвіл тільки трьох перших;

100 – дозвіл тільки чотирьох перших;

101 – дозвіл всі п'яти модулів.

Біт 2: UARTEN – UART I / O Enable Bit – біт дозволу інтерфейсу UART.

Біт 1: SPI0OEN – SPI Bus I / O Enable Bit – біт дозволу інтерфейсу SPI.

Біт 0: SMB0OEN – SMBus I / O Enable Bit – біт дозволу інтерфейсу

SMBus.

4.4.2.2 XBR1 – Регістр 1 комутатора ресурсів CrossBar

Назва регістра:	XBR1 – Port I/O CrossBar Register 1		
SFR–адреса:	0xE2	Значення після скидання:	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
SYSCKE	T2EXE	T2E	INT1E	TIE	INT0E	T0E	CP1OEN
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Біт 7: SYSCLK Output Enable Bit – біт дозволу виходу SYSCLK.

Біт 6: T2EXE– Timer 2 EX Enable Bit – біт дозволу входу T2EX таймера 2.

Біт 5: T2E – Timer 2 Enable Bit – біт дозволу входу T2 таймера 2.

Біт 4: INT1E – /INT1 Enable Bit – біт дозволу входу /INT1.

Біт 3: T1E – Timer 1 Enable Bit – біт дозволу входу таймера 1.

Біт 2: INT0E – /INT0 Enable Bit – біт дозволу входу /INT0.

Біт 1: T0E – Timer 0 Enable Bit – біт дозволу входу таймера 0.

Біт 0: CP1OEN – Comparator 1 Output Enable Bit – біт дозволу виходу компаратора 1.

Встановлення будь–якого з бітів дозволяє відповідний ресурс, скидання– забороняє.

4.4.2.3 XBR2 – Регістр 2 комутатора ресурсів CrossBar

Назва регістра:	XBR2 – Port I/O CrossBar Register 2		
SFR–адреса:	0xE3	Значення після скидання:	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
WEAKPUD	XBARE	–	–	–	–	–	CNVSTE
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Біт 7: WEAKPUD – Port I / O Weak Pull–up Disable Bit – біт дозволу (0) / заборони (1) підтяжки виводів (підключення резисторів, які з'єднано з напругою живлення) за винятком ліній введення / виведення, які не сконфігуровано з «відкритим стоком").

Біт 6: XBARE – CrossBar Enable Bit – біт дозволу (1) комутатора ресурсів CrossBar.

Біти 5–1 не використовуються, читаються як нулі, при запису значення ігноруються.

Біт 0: CNVSTE – ADC Convert Start Input Enable Bit – біт дозволу (1) входу запуску ADC.

4.4.2.4 P0: Регістр порту 0

Назва регістра:	P0 – Port0 Register		
SFR–адреса:	0x80	Значення після скидання:	11111111b (0xFF)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
P0.7	P0.6	P0.5	P0.4	P0.3	P0.2	P0.1	P0.0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Біти 7…0: P0. [7…0]

При запису:

0 – логічний низький рівень.

1 – логічний високий рівень або високоімпедансний (3–й) стан, якщо відповідний біт в регістрі PRT0CF.n дорівнює 0.

При читанні:

0 – якщо відповідний вивід скинуто в логічний 0.

1 – якщо відповідний вивід встановлено в логічну 1.

4.4.2.5 PRT0CF – Регістр конфігурації порту 0

Назва регістра:	PRT0CF – Port0 Configuration Register		
SFR–адреса:	0xA4	Значення після скидання:	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Біти 7…0: PRT0CF. [7…0] – Output Configuration Bits – біти конфігурації порту 0.

0 – відповідний вивід порту 0 встановлено в режим з "відкритим стоком".

1 – відповідний вивід порту 0 встановлено в звичайний режим.

Примітка: Якщо до виводів порту 0 приєднані лінії сигналів SDA, SCL і RX, вони завжди працюють в режимі з "відкритим стоком" незалежно від значень, встановлених в регістрі PRT0CF.

4.4.2.6 P1– Регістр порту 1

Назва регістра:	P1 – Port1 Register		
SFR–адреса:	0x90	Значення після скидання:	11111111b (0xFF)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
P1.7	P1.6	P1.5	P1.4	P1.3	P1.2	P1.1	P1.0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Біти 7…0: P1. [7…0]

При запису:

0 – логічний низький рівень.

1 – логічний високий рівень або високоімпедансний (3–) стан, якщо відповідний біт в регістрі PRT1CF.n дорівнює 0.

При читанні:

0 – якщо відповідний вивід скинуто в логічний 0.

1 – якщо відповідний вивід встановлено в логічну 1.

4.4.2.7 PRT1CF Регістр конфігурації порту 1

Назва регістра:	PRT1CF – Port1 Configuration Register		
SFR–адреса:	0xA5	Значення після скидання:	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Біти 7…0: PRT1CF. [7…0] – Output Configuration Bits – біти конфігурації порту 1.

0 – відповідний вивід порту 1 встановлено в режим з "відкритим стоком".

1 – відповідний вивід порту 1 встановлено в звичайний режим.

4.4.2.8 PRT11F – Регістр прапорців переривань порту 1

Назва регістра:	PRT11F – Port1 Interrupt Flag Register		
SFR–адреса:	0xAD	Значення після скидання:	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
IE7	IE6	IE5	IE4	–	–	–	–
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Біт 7: IE7 – External Interrupt 7 Pending Flag – прапорець переривання (1), що генерується в разі виявлення на вході P1.7 перепаду з високого логічного рівня в низький (задній фронт імпульсу). Якщо фронт не виявлено, прапорець дорівнює 0.

Біт 6: IE6 – External Interrupt 6 Pending Flag – прапорець переривання (1), що генерується в разі виявлення на вході P1.6 перепаду з високого логічного рівня в низький (задній фронт імпульсу). Якщо фронт не виявлено, прапорець

дорівнює 0.

Біт 5: IE5 – External Interrupt 5 Pending Flag – прапорець переривання (1), що генерується в разі виявлення на вході P1.5 перепаду з високого логічного рівня в низький (задній фронт імпульсу). Якщо фронт не виявлено, прапорець дорівнює 0.

Біт 4: IE4 – External Interrupt 4 Pending Flag – прапорець переривання (1), що генерується в разі виявлення на вході P1.4 перепаду з високого логічного рівня в низький (задній фронт імпульсу). Якщо фронт не виявлено, прапорець дорівнює 0.

Біти 3…0 не використовуються, читаються як 0000b, при записі значення ігнорується.

4.4.2.9 P2 – Регістр порту 2

Назва регістра:	P2 – Port2 Register		
SFR–адреса:	0xA0	Значення після скидання:	11111111b (0xFF)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
P2.7	P2.6	P2.5	P2.4	P2.3	P2.2	P2.1	P2.0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Біти 7…0: P2. [7…0]

При запису:

0 – логічний низький рівень.

1 – логічний високий рівень або високоімпедансний (3–й) стан, якщо відповідний біт в регістрі PRT2CF.n дорівнює 0.

При читанні:

0 – якщо відповідний вивід скинуто в логічний 0.

1 – якщо відповідний вивід встановлено в логічну 1.

4.4.2.10 PRT2CF – Регістр конфігурації порту 2

Назва регістра:	PRT2CF – Port2 Configuration Register		
SFR–адреса:	0xA6	Значення після скидання:	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Біти 7…0: PRT2CF. [7…0] – Output Configuration Bits – біти конфігурації порту 2.

0 – відповідний вивід порту 2 встановлено в режим з "відкритим стоком".

1 – відповідний вивід порту 2 встановлено в звичайний режим.

4.4.2.11 P3 – Регістр порту 3

Назва регістра:	P3 – Port3 Register		
SFR–адреса:	0xB0	Значення після скидання:	11111111b (0xFF)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
P3.7	P3.6	P3.5	P3.4	P3.3	P3.2	P3.1	P3.0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Біти 7…0: P3. [7…0]

При запису:

0 – логічний низький рівень.

1 – логічний високий рівень або високоімпедансний (3–) стан, якщо відповідний біт в регістрі PRT3CF.n дорівнює 0.

При читанні:

0 – якщо відповідний вивід скинуто в логічний 0.

1 – якщо відповідний вивід встановлено в логічну 1.

4.4.2.12 PRT3CF – Регістр конфігурації порту 3

Назва регістра:	PRT3CF – Port3 Configuration Register		
SFR–адреса:	0xA7	Значення після скидання:	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Біти 7…0: PRT3CF. [7…0] – Output Configuration Bits – біти конфігурації порту 3.

0 – відповідний вивід порту 3 встановлено в режим з "відкритим стоком".

1 – відповідний вивід порту 3 встановлено в звичайний режим.

ПИТАННЯ ДЛЯ САМОКОНТРОЛЮ

1) Охарактеризуйте способи обміну даними між зовнішніми пристроями і мікропроцесорною системою.
2) За відповідними схемами опишіть:
    a. Роботу P0 із ЗПП,
    b. Роботу P0 в якості порту введення/виведення.
3) Назвіть спільні та відмінні риси в конфігуруванні та роботі портів P1 і P2.
4) Опишіть основні та альтернативні функції порту P3.
5) Для чого використовуються паралельні порти введення/виведення?
6) Опишіть роботу P3 у якості порту введення/виведення.
7) Назвіть способи розширення резидентної (внутрішньої) системи введення/виведення.
8) Опишіть призначення ліній A0 та A1 паралельного інтерфейсу ВВ55.
9) Опишіть призначення лінії CS паралельного інтерфейсу ВВ55.
10) Як використовується порт P0 при роботі із зовнішньою пам`яттю даних?
11) Як програмуються паралельні порти на введення або виведення інформації?
12) Опишіть використання вузла "Crossbar" мікроконтролерів SiLab (Cygnal).
13) Як запрограмувати лінії портів мікроконтролерів SiLab (Cygnal) як лінії з "відкритим стоком"?
14) Як запрограмувати лінії портів мікроконтролерів SiLab (Cygnal) як лінії з "третім" високоімпедансним станом?
15) Опишіть призначення регістрів XBR0, XBR1 і XBR2.
16) Опишіть призначення регістрів PRTOCF, PRT1CF, PRT2CF і PRT3CF.

5 АРХІТЕКТУРА ПІДСИСТЕМИ ПЕРЕРИВАНЬ

5.1 Загальні відомості

Підсистема переривань у мікропроцесорних системах (МПС) призначена для прийому, пріоритетної обробки і обслуговування запитів переривань. В загальному випадку запити переривань можуть формуватися за командами програм, внутрішніми та зовнішніми джерелами. Прийом та обробку програмних і внутрішніх переривань виконують внутрішні засоби процесора. При проектуванні підсистеми переривань для обробки програмних і внутрішніх переривань процесорів, виконаних на однокристальних МП або МК необхідно розробити підпрограми обслуговування переривань. Для обробки зовнішніх переривань потрібно розробити зовнішні апаратні засоби. Джерелами зовнішніх переривань в МПС є пристрої введення/виведення; пристрої, що задають час; аварійні ситуації (наприклад, збій енергоживлення та подібні). Вихідною інформацією при проектуванні системи переривань є число запитів переривань, які обслуговуються, розподілення їх пріоритетів, час на обслуговування і т. ін. При проектуванні підсистеми переривань від зовнішніх чинників вирішують задачі обробки запитів переривань і формування сигналу переривання для процесора; відмови та дозволу прийому запитів переривань під час виконання програм; ідентифікації джерел запитів переривань; передачі керування підпрограмам обслуговування запитів переривань; збереження у стеку поточного стану процесора при переході до підпрограм обслуговування запитів переривань і його відновлення після завершення підпрограм; обробки пріоритетів запитів переривань; сполучення засобів підтримки системи переривань з процесором.

Засоби, які реалізують обробку переривань в мікропроцесорних системах, називаються підсистемою переривань.

Переривання МП–в та МК–в можуть програмно маскуватися. Для цього використовуються прапорці (тригери), які приймають два значення:

- нуль, переривання заборонено;
- одиниця, переривання дозволено.

Існують два види прапорців маскування переривань:

- прапорець глобального (загального) маскування переривань від всіх периферійних пристроїв (модулів);
- прапорці маскування переривань від окремих периферійних модулів мікроконтролера.

Для кожного можливого переривання при проектуванні МПС розроблюється відповідна підпрограма, яка зберігається в пам'яті МПС. Для того, щоб викликати потрібну підпрограму обробки переривання, треба визначити її адресу. Як це зробити, залежить від типу конкретного МП–ра або МК–ра, який використовується в МПС.

При одночасному надходженні переривань від декількох джерел діє система пріоритетів, особливості якої залежать від типу конкретного МП–ра або МК–ра.

Загальний підхід призначення пріоритетів переривань полягає в тому, що, як правило, існує дві ступені розподілу пріоритетів. Перша ступінь реалізована апаратно і дозволяє при одночасному запиті від декількох джерел обрати одне із більш високим пріоритетом.

Друга ступінь реалізується програмно. При цьому для кожного джерела переривання в мікроконтролері використовується окремий прапорець (тригер), який програмно або встановлюється в 1, що задає цьому перериванню високий пріоритет, або скидається в 0, що задає низький пріоритет. Очевидно, що переривання із більш високим програмно заданим пріоритетом перерве підпрограму обслуговування переривання із більш низьким програмно встановленим пріоритетом.

5.2 Підсистема переривань мікроконтролера AT89C51

5.2.1 Загальні відомості

МК AT89C51 має підсистему переривань із п'ятьма векторами (адресами підпрограм обробки переривань) і двома рівнями пріоритетів. Джерелами переривань являються: два зовнішніх переривання, що надходять через порт 3; два переривання від переповнення таймерів–лічильників T/C0 і T/C1 та переривання за завершенням передачі або прийому даних при обміні через послідовний порт.

Підсистема переривань включає два керуючих регістри, за допомогою яких вона програмується, а також *схему логічної обробки прапорців переривань*, яка здійснює пріоритетний вибір запиту переривання, скидання його прапорця та ініціює формування апаратно реалізованої команди переходу на підпрограму обслуговування переривання LCALL.

Схема обробки вектора переривання формує двобайтові адреси підпрограм обслуговування переривання в залежності від джерела переривання.

5.2.2 Структура підсистеми переривань

На рисунку 5.1 зображено всі можливі джерела переривання і порядок їхньої обробки.

Кожне з зовнішніх переривань $\overline{INT}0$, $\overline{INT}1$ може бути активовано за рівнем ("0") або за фронтом (перехід з "1" в "0") сигналів на лініях МК–ра: P3.2, P3.3, що визначається станом бітів IT0 та IT1 регістра TCON. При надходженні запиту зовнішнього переривання $\overline{INT}x$ (x = 0, 1) встановлюється прапорець IEx (x = 0, 1) регістра TCON.

Встановлення прапорців IEx у регістрі TCON викликає відповідне переривання, якщо воно не замасковано.

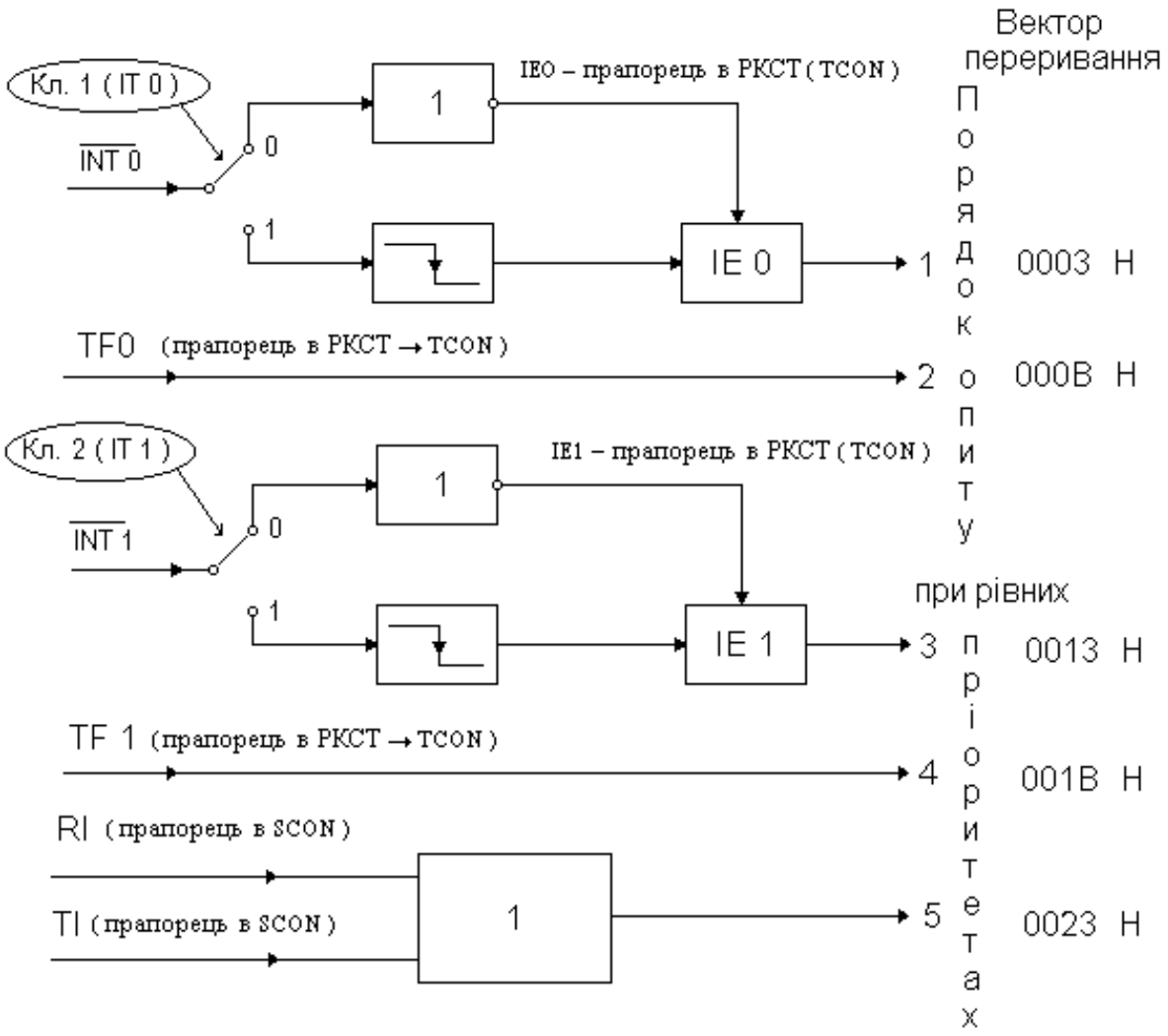


Рисунок 5.1 – Спрощена схема підсистеми переривань

Скидання прапорця IEx здійснюється наступним чином: при перериванні за фронтом IEx скидається апаратно (автоматично внутрішніми засобами МК–ра) при зверненні до відповідної підпрограми обробки переривання; при перериванні за нульовим рівнем прапорець очищається при знятті запиту зовнішнього переривання, тобто в IEx відслідковується стан виводу $\overline{INTx}$.

Щоб зовнішнє переривання за рівнем було розпізнано, необхідно, щоб низький рівень на виводі $\overline{INTx}$ утримувався протягом не менше 12 періодів сигналу тактової частоти МК–ра. Це пояснюється тим, що перевірка виводів МК–ра $\overline{INT0}$, $\overline{INT1}$ виконується його внутрішніми апаратними засобами один

раз у кожному машинному циклі. У випадку зовнішнього переривання за фронтом прапорець IEx буде встановлено, якщо дві послідовні перевірки входу $\overline{INTx}$ покажуть в одному машинному циклі "1", а в наступному "0".

Тому, якщо зовнішнє переривання активується за переходом зі стану високого рівня у стан низького рівня, то мінімум одному машинному циклу низького рівня на виводі $\overline{INTx}$ повинний передувати мінімум один машинний цикл високого рівня Якщо зовнішнє переривання активується за рівнем, запит повинен утримуватися до початку обслуговуючої підпрограми і зніматися до завершення цієї підпрограми для запобігання повторного обслуговування.

Переривання від таймерів/лічильників викликаються встановленням прапорців TF0 і TF1 регістра TCON, які встановлюються при переповненні відповідних регістрів таймерів/лічильників. Скидання прапорців TF0 і TF1 здійснюється внутрішньою апаратурою МК–ра при переході до підпрограми обслуговування переривання.

Переривання від послідовного порту викликається встановленням прапорця переривання приймача RI або прапорця переривання передавача TI у регістрі SCON. На відміну від всіх інших прапорців, RI і TI скидаються тільки програмним шляхом, як правило, в межах підпрограми обробки переривання, де визначається, якому з прапорців RI або TI відповідає переривання.

Кожне з перерахованих джерел переривань може бути індивідуально дозволено або заборонено встановленням або скиданням відповідного біта у регістрі дозволу переривань IE. Регістр IE містить також біт EA, скидання якого у "0" забороняє відразу всі переривання. Необхідною умовою переривання є його дозвіл у регістрі IE.

Усі біти, що викликають переривання (IE0, IE1, TF0, TF1, RI, TI), можуть бути програмно встановлені або скинуті з тим же результатом, що й у випадку їхнього апаратного встановлення або скидання. Тобто переривання можуть програмно викликатися або переривання, які очікують обслуговування, можуть

програмно ліквідуватися. Крім того, переривання за $\overline{INT0}$, $\overline{INT1}$ можуть викликатися програмним встановленням P3.2 = 0 і P3.3 = 0, як показано в наведеному нижче прикладі:

```
MAIN:   MOV IE, # 00000101B     ;дозвіл переривання від INT0, INT1.
        MOV IP, # 04H           ;присвоєння INT1 старшого пріоритету.
        SETB EA                 ;загальний дозвіл переривання.
        MOV P3; # 11110011B     ;імітація зовнішніх переривань.
SUBR:   ORG013H          ;перехід до підпрограми обслуговування INT1.
```

У запропонованому прикладі запити переривання $\overline{INT0}$ і $\overline{INT1}$, що мають різний пріоритет, надходять одночасно. При цьому обслуговується переривання $\overline{INT1}$ з вищим пріоритетом.

У випадку, коли переривання за $\overline{INTx}$ (x = 0, 1) викликається низьким рівнем сигналу на відповідному вході МК–ра, прапорець IEx (x = 0, 1) при переході до підпрограми обробки переривання автоматично скидається, а потім, якщо відповідний вивід МК–ра P3.2 або P3.3 все ще знаходиться у стані логічного "0", знову встановлюється. Тому, у випадку, коли переривання за входами $\overline{INT0}$, $\overline{INT1}$ викликається рівнем, програмне встановлення в "1" прапорців IE0, IE1 викличе переривання, після чого відповідний прапорець IEx (x = 0, 1) буде автоматично скинуто при переході до підпрограми обробки переривання.

Прапорці IE0, IE1, TF0, TF1, RI, TI встановлюються незалежно від того, дозволено або ні відповідне переривання у регістрі IE.

Структура пріоритетів переривань є двоступінчатою. Кожному джерелу переривання може бути індивідуально привласнено один з двох рівнів пріоритету: високий або низький. Це виконується встановленням (високий рівень пріоритету) або скиданням (низький рівень пріоритету) відповідного біта у регістрі пріоритетів переривань IP. Підпрограма обробки переривання з

низьким рівнем пріоритету може бути перервана запитом переривання з високим рівнем пріоритету, але не може бути перервана іншим запитом переривання з низьким рівнем пріоритету. Підпрограма обробки переривання з високим рівнем пріоритету не може бути перервана ніяким іншим запитом переривання від жодного з джерел. Якщо два запити з різними рівнями пріоритету прийнято одночасно, спочатку буде обслуговано запит з високим рівнем пріоритету. Якщо одночасно прийнято запити з однаковим програмно встановленим рівнем пріоритету, то їх обробка буде здійснюватися у порядку, що задається послідовністю внутрішнього опиту прапорців переривань. Таким чином, у межах одного пріоритетного рівня існує ще одна структура пріоритетів:

Джерело	Пріоритет у середині рівня
1. IE0	(вищий)
2. TF0	
3. IE1	
4. TF1	
5. RI + TI	(нижчий)

Необхідно окремо підкреслити, що структура "Пріоритет в середині рівня" працює тільки у тих випадках, коли визначається послідовність обслуговування запитів на переривання, які прийнято одночасно і при цьому мають однаковий рівень пріоритету, який задано програмно.

Обробка переривань і час відгуку. Програмно встановлені рівні на виводах $\overline{INT0}$ і $\overline{INT1}$ встановлюють прапорці переривання IE0 і IE1 у фазі S5 P2 кожного машинного циклу. У фазі S5 P2 встановлюються прапорці переривань послідовного порту RI і TI. Прапорці TF0 і TF1 таймерів/лічильників встановлюються у фазі S5 P2 машинного циклу, в якому відбувається переповнення T/C. Аналіз (опитування) прапорців виконується внутрішніми

засобами МК–ра в наступному після встановлення (защіпання) прапорців машинному циклі (цикл опитування прапорця). І тільки після виконання останнього циклу поточної команди здійснюється апаратний виклик відповідної підпрограми обслуговування, який еквівалентний команді LCALL [2,3].

Апаратно реалізована команда LCALL завантажує вміст лічильника команд PC у стек (при цьому PSW у стек не записується), після чого записує у PC адресу відповідної підпрограми обробки переривання (таблиця 5.1).

Таблиця 5.1 – Адреси підпрограм (вектор переривання)

Джерело переривання	Адреса підпрограм (вектор переривання)
IE0	0003 H
TF0	000B H
IE1	0013 H
TF1	001B H
TI + RI	0023 H

При виконанні апаратно реалізованої команди LCALL у комірку стеку з молодшою адресою завантажуються розряди 0…7 лічильника команд, а у наступну комірку стеку – розряди 8…15 лічильника команд.

Підпрограма обслуговування переривання продовжується до виконання команди RETI. Команда RETI відновлює стан логіки обробки переривань і завантажує у лічильник команд PC 2 байти адреси повернення з двох верхніх комірок стеку. Відновлення стану логіки обробки переривань полягає в наступному: при переході за вектором на підпрограму обробки переривання автоматично до виконання команди RETI незалежно від стану бітів регістра IE забороняються всі переривання з рівнем пріоритету, що дорівнює рівню

пріоритету переривання, яке обслуговується. Тобто вкладені переривання з рівними рівнями пріоритету неможливі. Команда RETI знімає цю заборону.

При використанні команди RET відновлюється тільки стан лічильника команд, тобто відбувається повернення у перервану програму. Стан логіки обробки переривань команда RET не змінює, тобто логіка керування обслуговуванням переривань, як і раніше, вважає, що продовжує обслуговуватися переривання, підпрограма обробки якого була закінчена командою RET. Це може привести до того, що наступні переривання будуть заборонені, якщо команда RET завершувала підпрограму із високим програмно встановленим рівнем пріоритету.

5.2.3 Керуючі регістри та алгоритм обробки переривань

Для програмування і керування роботою підсистеми переривань служать два регістри: РМП (IE) – регістр масок переривань і РП (IP) – регістр пріоритетів переривань, а також чотири молодших біти регістра РКСТ (таблиці 5.2, 5.3, 3.3, 3.4).

Регістр пріоритетів переривань (IP) призначений для встановлення рівня пріоритету переривання для кожного з п'ятьох джерел переривань. Позначення розрядів регістра IP показано в таблиці 5.2, а їхнє призначення зазначене нижче.

Таблиця 5.2 – Позначення розрядів регістра IP

Біти	7	6	5	4	3	2	1	0
Позначення	X	X	X	PS	PT1	PX1	PT0	PX0

PX0 (IP0) – встановлення рівня пріоритету переривання від зовнішнього джерела $\overline{INT0}$.

PT0 (IP1) – встановлення рівня пріоритету переривання від T/C0.

PX1 (IP2) – встановлення рівня пріоритету переривання від зовнішнього джерела $\overline{INT1}$.

PT1 (IP3) – встановлення рівня пріоритету переривання від T/C1.

PS (IP4) – встановлення рівня пріоритету переривання від послідовного порту.

X (IP7, IP6, IP5) – резервний розряд.

Наявність у розряді IP сигналу "логічна 1" встановлює для відповідного джерела високий рівень пріоритету, а наявність у розряді IP сигналу "логічний 0" – низький рівень пріоритету. При читанні резервних розрядів відповідні лінії шини даних не визначено. Користувач не повинен записувати "1" у резервні розряди, тому що вони зарезервовані для подальшого розширення сімейства МК–51.

Регістр дозволу переривань (IE) призначено для дозволу або заборони переривань від відповідних джерел. Позначення розрядів регістра IE показано в таблиці 5.3, а їхнє призначення зазначено нижче.

Таблиця 5.3 – Позначення розрядів регістра IE

Біти	7	6	5	4	3	2	1	0
Позначення	EA	X	X	ES	ET1	EX1	ET0	EX0

EA (IE7) – керування всіма джерелами переривань одночасно. Якщо EA = 0, то переривання заборонено. Якщо EA = 1, то переривання можуть бути дозволені індивідуальними дозволами EX0, ET0, EX1, ET1, ES.

X (IE6, IE5) – резервний розряд.

ES (IE4) – керування перериванням від послідовного порту: ES = 1 – дозвіл, ES = 0 – заборона.

ET1 (IE3) – керування перериванням від T/C1: ET1 = 1 – дозвіл, ET1 = 0 – заборона.

EX1 (IE2) – керування перериванням від зовнішнього джерела $\overline{INT1}$: EX1 = 1 – дозвіл, EX1 = 0 – заборона.

ET0 (IE1) – керування перериванням від T/C0: ET0 = 1 – дозвіл, ET0 = 0 – заборона.

EX0 (IE0) – керування перериванням від зовнішнього джерела $\overline{INT0}$: EX0 = 1 – дозвіл, EX0 = 0 – заборона.

При зчитуванні резервних розрядів відповідні лінії шини не визначено. Користувач не повинен записувати сигнал "логічна 1" у резервні розряди, тому що вони зарезервовані для подальшого розширення сімейства МК–51.

Керуючі біти IT0, IT1, які визначають вид переривання за входами INT0, INT1, та прапорці зовнішніх переривань IE0, IE1 знаходяться у молодшій тетраді регістра TCON, який розглянуто у розділі 3 (таблиці 3.3, 3.4).

Прапорці IE0 і IE1 встановлюються апаратно від зовнішніх переривань (відповідно входи МК: INT0 і INT1) або програмно й ініціюють виклик підпрограми обробки відповідного переривання. Скидання цих прапорців виконується апаратно при обслуговуванні переривання тільки в тому випадку, коли переривання було викликано за фронтом сигналу. Якщо переривання було викликано рівнем сигналу на вході INT0 (INT1), то скидання прапорця IE повинна виконувати підпрограма обслуговування переривання, впливаючи на джерело переривання для зняття ним запиту.

Алгоритм обробки переривання при виявленні запиту переривання подано на рисунку 5.2.

5.3 Розвиток підсистеми переривань в сучасних мікроконтролерах сімейства МК–51

5.3.1 Підсистема переривань мікроконтролерів SiLab

Розширений контролер переривань мікроконтролерів SiLab (ядро CIP–51) може обслужити досить велику кількість джерел переривання (середнє – 22

джерела переривань), яке залежить від складу вбудованої периферії сімейства. Внутрішня периферія може генерувати, наприклад, 12 переривань.

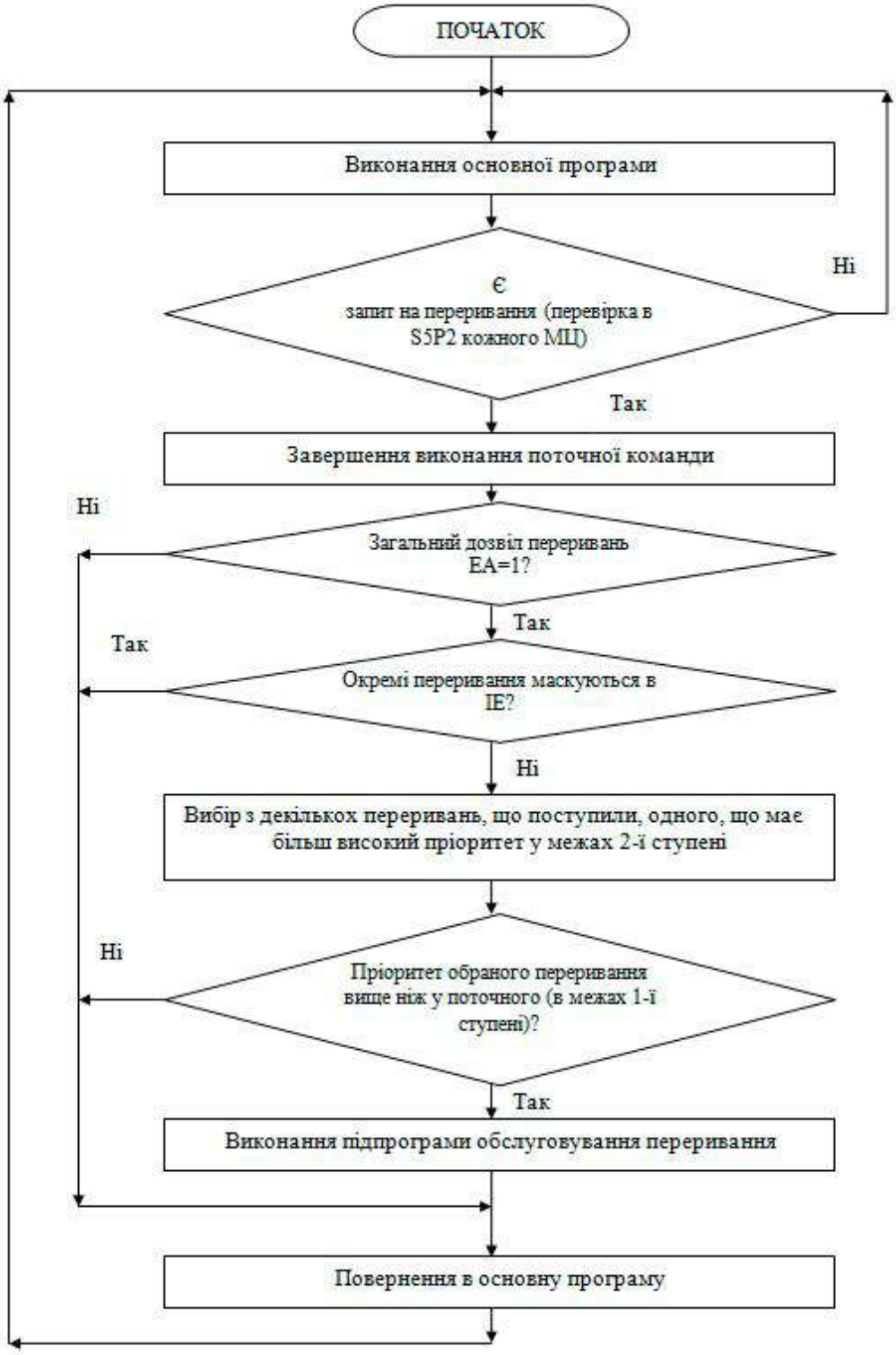


Рисунок 5.2 – Схема алгоритму обробки переривання

Ще, наприклад, 10 зовнішніх джерел переривань можуть бути пов'язані з лініями портів введення/виведення. Навіть найбільш "слабкі" з мікроконтролерів вигідно відрізняються від стандартного ядра 8051, який може обслуговувати всього 5 джерел. Наявність великої кількості джерел переривання дозволяє підвищити загальну продуктивність системи за рахунок обслуговування переривань від багаточисельних аналогових вузлів та звільнення потужності процесора для виконання основного завдання. Очевидно, що ця якість дуже корисна для систем керування і контролю реального часу. Джерела переривань можуть мати два рівні пріоритетів. Кожне джерело переривань має один або кілька прапорців (бітів) в регістрі спеціальних функцій. При дотриманні сконфігурованих умов у внутрішній периферії або на вхідних лініях зовнішніх входів переривань прапорець, який відповідає джерелу переривань, встановлюється в стан логічної одиниці. Встановлення відповідного прапорця викликає власне переривання, якщо воно дозволено для відповідного джерела. Це, в свою чергу, викличе генерацію інструкції LCALL та перехід на заздалегідь визначену адресу підпрограми обробки переривання відповідного джерела – ISR (Interrupt Service Routine) відразу ж після завершення поточної інструкції. Кожна ISR повинна завершитися інструкцією RETI, яка повертає програму на інструкцію, наступну безпосередньо за інструкцією, що виконується в момент генерації переривання. Якщо переривання для відповідного джерела заборонено, відповідний прапорець переривання ігнорується, а виконання програми триває в нормальному режимі. Очевидно, що кожне джерело переривання може бути дозволено (заборонено) шляхом встановлення/скидання відповідного біта в регістрі спеціальних функцій SFR (IE–EIE2). Всі переривання також можуть бути або дозволені, або заборонені відповідним встановленням біта EA (IE.7). Деякі з прапорців переривань скидаються в 0 ("знімаються",

"стираються") при виконанні підпрограми автоматично. Інші прапорці необхідно знімати програмно.

Два зовнішніх джерела переривань (/INT0 і /INT1) можуть бути налаштовані на сприйняття нульового рівня (потенціалу) або перепаду з високого на низький рівень (задній фронт імпульсу) шляхом встановлення бітів IT0 (TCON.0) і IT1 (TCON.2). При цьому їм відповідають прапорці IE0 (TCON.l) і IE1 (TCON.3). Якщо ці переривання налаштовано на сприйняття фронту, відповідні прапорці стираються автоматично, як тільки ядро мікроконтролера згенерує перехід на відповідну підпрограму обслуговування переривання. Якщо ж переривання налаштовано на сприйняття рівня, то відповідний прапорець повторює стан на відповідному вході мікроконтролера. Необхідно враховувати, що якщо вхідний активний (низький) потенціал переривання не буде знято до завершення виконання процедури переривання, ця процедура буде повторена знову. Ще чотири зовнішніх переривання (External Interrupts 4...7) налаштовано на сприйняття заднього фронту імпульсу (від'ємного перепаду рівнів з високого в низький). Відповідні прапорці цих переривань знаходяться в регістрі переривання першого порту (Port 1 Interrupt Flag Register).

Кожне з джерел переривань може бути індивідуально запрограмовано на один з двох пріоритетних рівнів: низький або високий. Підпрограми обробки переривань низького пріоритету можуть бути перервані джерелами переривань з високим пріоритетом. Очевидно, що підпрограми обробки переривань з високим пріоритетом не можуть бути перервані. Кожне джерело переривань має індивідуальний біт пріоритету в SFR (IP–EIP2), який використовується для встановлення рівня пріоритету. Після скидання мікроконтролера всі пріоритети встановлюються низькими. Якщо ядро одночасно сприйняло два переривання з різними пріоритетами, переривання з високим пріоритетом обробляється першим. Якщо одночасно сприйняті переривання з однаковим пріоритетом,

вступає в силу фіксований рівень пріоритетів (див. опис відповідних регістрів SFR). Час реакції на переривання залежить від стану процесора в момент виникнення переривання. Перевірка наявності переривань відбувається в кожному такті генератора, а час виконання інструкції LCALL дорівнює чотирьом тактам, отже, мінімально можливий час реакції на переривання становить 5 тактів. Якщо переривання виявлено в момент виконання інструкції RETI, то після її завершення виконується наступна команда основної програми, а лише потім генерується інструкція LCALL. Отже, максимальний час реакції на переривання буде в тому випадку, коли переривання виявлено при виконанні інструкції RETI, а слідом за нею йде сама "довга" (за часом) інструкція DIV. У цьому випадку час реакції складе 18 тактів: 1 на виявлення переривання, 5 тактів на виконання інструкції RETI, 8 тактів на виконання інструкції DIV і 4 такти на виконання LCALL. Якщо під час виконання підпрограми обробки переривання прийшло наступне переривання з рівним пріоритетом, це нове переривання буде обслуговано тільки після завершення виконання поточної підпрограми, включаючи інструкцію повернення RETI, і наступної інструкції основної програми.

Як приклад нижче в таблиці 5.4 наведено переривання мікроконтролерів Silicon Labs F02x.

Таблиця 5.4 – Переривання мікроконтролерів Silicon Labs F02x

1	Скидання
2	Зовнішнє переривання 0
3	Переповнення таймера 0
4	Зовнішні переривання 1
5	Переповнення таймера 1
6	Послідовний інтерфейс UART0
7	Переповнення таймера 2
8	Послідовний інтерфейс SPI
9	Послідовний інтерфейс SMBus
10	Віконний компаратор перетворювача ADC0
11	Програмований масив лічильника PCA
12	Компаратор 0, падаючий фронт
13	Компаратор 0, наростаючий фронт
14	Компаратор 1, падаючий фронт
15	Компаратор 1, наростаючий фронт
16	Переповнення таймера 3
17	Кінець перетворення перетворювача ADC0
18	Переповнення таймера 4
19	Кінець перетворення перетворювача ADC1
20	Зовнішнє переривання 6
21	Зовнішнє переривання 7
22	Послідовний інтерфейс UART1
23	Готовність зовнішнього генератора OSC

ПИТАННЯ ДЛЯ САМОКОНТРОЛЮ

1) Опишіть призначення підсистеми переривань у МПС та дайте їх загальну характеристику.
2) Назвіть можливі джерела переривань і порядок їх обробки.
3) Як виконується маскування окремих переривань?
4) Опишіть дві ступені розподілу пріоритетів переривань.
5) Опишіть спрощену схему підсистеми переривань мікроконтролера AT89C51.
6) Назвіть адреси підпрограм обробки переривань мікроконтролера AT89C51.
7) Як виконується апаратно реалізована команда LCALL?
8) Чому підпрограма обслуговування переривання повинна закінчуватись командою RETI, а не RET?
9) Поясніть позначення та призначення окремих розрядів регістра IP.
10) Поясніть позначення та призначення окремих розрядів регістра IE.
11) Які регістри використовуються в програмуванні підсистеми переривань мікроконтролера AT89C51?
12) Опишіть алгоритм обробки переривання.
13) Як можуть формуватись запити переривань?
14) Наведіть приклади джерел зовнішніх та внутрішніх переривань в МПС.
15) Де зберігається поточний стан процесора при переході до підпрограм обслуговування запитів переривань?
16) Як може бути активовано зовнішнє переривання на лініях МК–ра?
17) В чому полягає відновлення стану логіки обробки переривань при поверненні з підпрограм?
18) Назвіть та поясніть джерела переривань мікроконтролерів Silicon Labs F02x.

6 АРХІТЕКТУРА ПОСЛІДОВНОГО ІНТЕРФЕЙСУ

6.1 Місце послідовного інтерфейсу у структурі мікроконтролера

6.1.1 Принцип роботи послідовного інтерфейсу

Існує два способи обміну даними між зовнішніми пристроями (ЗПР) і мікропроцесорною системою (МПС):

- паралельний, коли одночасно передаються всі біти або декілька біт слова даних;
- послідовний, коли біти слова даних пересилаються по черзі, починаючи, наприклад, з його молодшого розряду.

ЗПР зв'язуються з МПС лініями зв'язку, довжина яких при паралельному обміні обмежена і складає кілька метрів.

При послідовному обміні даними обмежень на довжину ліній зв'язку не накладається. Ця обставина, а також бажання використовувати для дистанційного обміну інформацією між ЗПР і МПС існуючі канали зв'язку, обумовили широке поширення послідовного обміну даними між ЗПР і МПС чи між декількома МПС.

Обмін інформацією в МП–рі або МК–рі здійснюється в паралельній формі. Тому при послідовному обміні даними необхідно: при передачі даних від МПС до ЗПР – перетворити дані з паралельної форми в послідовну, а при прийомі інформації від ЗПР та введенні її у МПС – перетворити з послідовної форми в паралельну.

Процес перетворення даних з паралельної форми в послідовну показано на рисунку 6.1.

Для перетворення даних з паралельної форми в послідовну, інформація завантажується в регістр зсуву. Вміст регістра зсуву послідовно зсувається на один розряд при надходженні кожного тактового імпульсу від генератора тактових імпульсів (ГТІ).

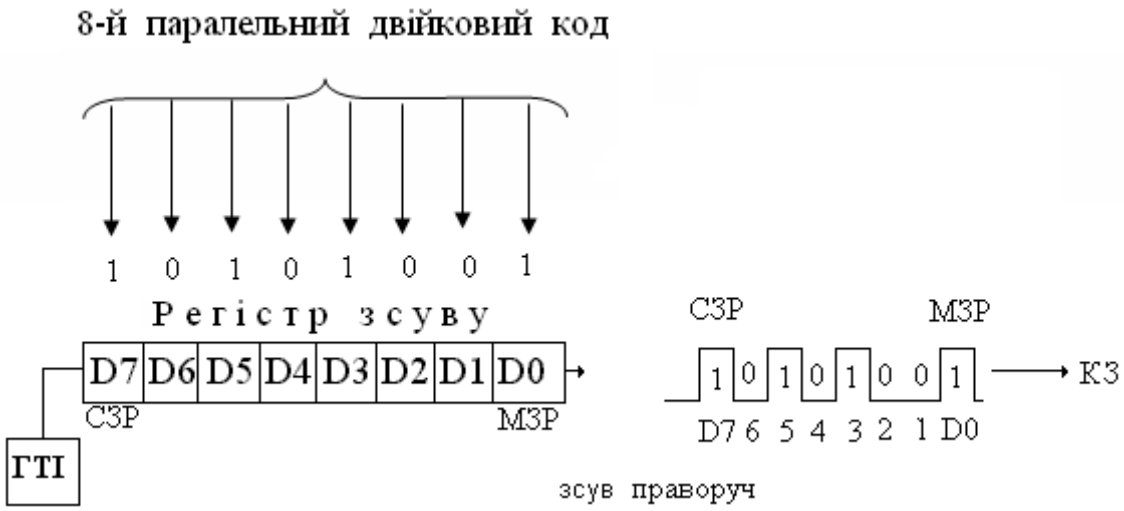


Рисунок 6.1 – Перетворення даних з паралельної форми в послідовну

Дані на виході такого регістра будуть мати послідовну форму. Часто, при послідовній передачі в канал зв’язку (КЗ) першим передається молодший значущий розряд (МЗР) слова даних, останнім – старший значущий розряд (СЗР).

Для зворотного перетворення даних з послідовної форми в паралельну необхідно виконати дії, зворотні відносно описаного. Дані, що надходять з каналу зв'язку в послідовній формі, вводяться біт за бітом у регістр зсуву. Після заповнення регістра зсуву, інформація з нього в паралельній формі передається в МП–р.

Подібні перетворення виконуються в МК–рі при обміні даними через його послідовний інтерфейс (порт) (рисунок 6.2).

6.1.2 Місце послідовного порту в структурі мікроконтролера AT89C51

Мікроконтролер типу МК–51, наприклад, AT89C51, містить один модуль (блок) послідовного введення/виведення інформації (послідовний інтерфейс) (рисунок 6.2).

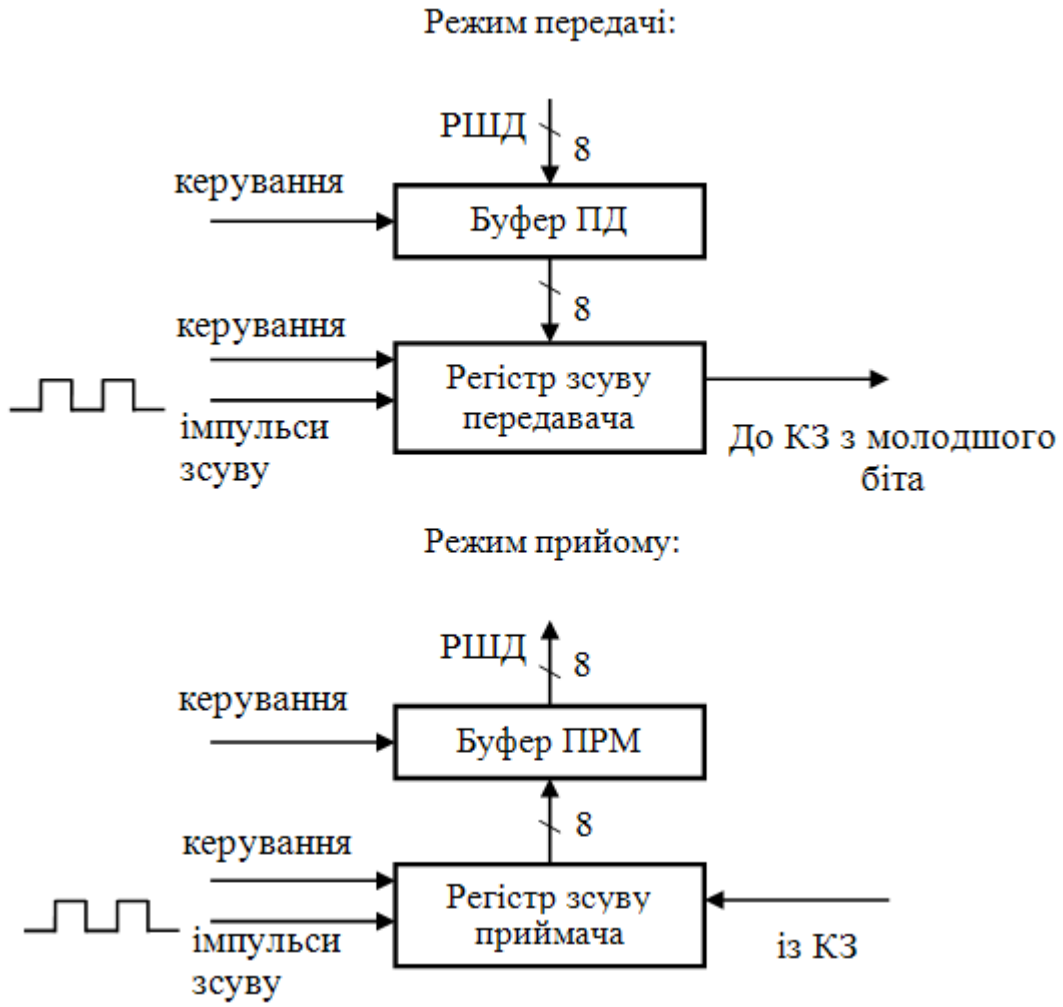


Рисунок 6.2 – Спрощена структура послідовного порту мікроконтролера

До складу інтерфейсу входять: буфер інтерфейсу, логіка керування інтерфейсом, регістр керування, буфер передавача, буфер приймача, приймач–передавач послідовного порту.

Буфер інтерфейсу забезпечує побайтовий обмін інформацією між внутрішньою (резидентною) шиною даних і шиною інтерфейсу.

Логіка керування інтерфейсом призначена для формування сигналів керування, які забезпечують чотири режими роботи послідовного інтерфейсу.

Регістр керування (SCON) призначено для прийому і зберігання коду восьмибітного слова, яке керує послідовним інтерфейсом. Позначення та

призначення розрядів регістра SCON наведено у розділі 2. Всі розряди регістра SCON програмно доступні для запису і читання.

Буфер передавача призначено для прийому з шини мікроконтролера паралельних даних і їх видачі на передавач послідовного порту.

Буфер приймача служить для прийому даних у паралельній формі від приймача послідовного інтерфейсу.

Буфер приймача і буфер передавача при програмному доступі мають однакове логічне ім'я (SBUF) і адресу (99H). Якщо команда використовує SBUF як регістр джерела, то звернення відбувається до буфера приймача. Якщо команда використовує SBUF як регістр призначення, то звернення відбувається до буфера передавача.

В усіх режимах роботи послідовного порту передача ініціюється будь–якою командою, яка використовує SBUF як регістр призначення.

Приймач–передавач послідовного порту призначено для прийому послідовного потоку символів зі входу послідовного порту, виділення даних і видачі їх у буфер приймача, а також для прийому паралельних даних із буфера передавача, перетворення їх у послідовний потік символів і видачі його на вихід послідовного порту.

Послідовний порт мікроконтролера може використовуватися, або в якості регістра зсуву для розширення можливостей введення/виведення, або в якості універсального асинхронного приймача–передавача (УАПП) з фіксованою, або змінною швидкістю послідовного обміну і можливістю дуплексного обміну (тобто через послідовний порт можна одночасно приймати і передавати дані).

Принцип роботи послідовного порту пояснює його спрощена структура (рисунок 6.2), яка включає наступні вузли: регістр зсуву, що передає (регістр зсуву ПД); регістр зсуву, що приймає (регістр зсуву ПРМ); буферний регістр приймача–передавача (SBUF), що містить буфер передавача (ПД) і буфер приймача (ПРМ).

Зв'язок із ЗПР здійснюється через дві лінії порту Р3, які виконують альтернативну функцію:

– P3.0: RxD, вхід послідовного порту, який призначено для введення послідовних даних у приймач послідовного порту;
– P3.1: TxD, вихід послідовного порту, який призначеной для виведення послідовних даних із передавача послідовного порту.

6.2 Програмування послідовного порту

Для програмування послідовного порту призначені:

– регістр керування приймачем–передавачем (РКПП): SCON;
– старший біт регістра керування потужністю (РКП): PCON.7 (SMOD).

Регістр керування (SCON) призначено для прийому і зберігання коду восьмибітового слова, яке керує послідовним інтерфейсом. Позначення розрядів регістра SCON наведене в таблиці 6.1. Всі розряди регістра SCON програмно доступні для запису і читання.

Розряди SM0, SM1 визначають режим роботи інтерфейсу, як зазначено в таблиці 6.2.

Таблиця 6.1 – Позначення розрядів регістра SCON

Біти	7	6	5	4	3	2	1	0
Позначення	SM0	SM1	SM2	REN	TB8	RB8	TI	RI

Інші біти регістра мають наступне призначення:

SM2 – дозвіл багатопроцесорної роботи. У режимах 2 і 3 при SM2 = 1 прапорець RI не встановлюється, якщо дев'ятий прийнятий біт даних дорівнює "0". У режимі 1 при SM2 = 1 прапорець RI не встановлюється, якщо не прийнято стоп–біт, який дорівнює "1". В режимі 0 біт SM2 повинен бути скинутий у "0".

Таблиця 6.2 – Вплив розрядів SM0, SM1 регістра SCON на режим роботи інтерфейсу

SM0	SM1	Режим	Найменування	Швидкість передачі
0	0	0	Регістр зсуву	$f_{BQ}/12$
0	1	1	8–бітовий універсальний асинхронний приймач–передавач (УАПП)	змінна, задається Т/Л1
1	0	2	9–бітовий УАПП	$f_{BQ}/64$ або $f_{BQ}/32$
1	1	3	9–бітовий УАПП	змінна, задається Т/Л1

REN – дозвіл прийому послідовних даних. Встановлюється і скидається програмою відповідно для дозволу і заборони прийому;

TB8 – дев'ятий біт даних, які передаються, у режимах 2 і 3. Встановлюється і скидається програмою;

RB8 – дев'ятий біт прийнятих даних у режимах 2 і 3. У режимі 1, якщо SM2 = 0, RB8 є прийнятим стоп–бітом. У режимі 0 біт RB8 не використовується;

TI – прапорець переривання передавача. Встановлюється апаратно наприкінці видачі 8–го біта в режимі 0 або на початку стоп–біта в інших режимах. Скидається програмно;

RI – прапорець переривання приймача. Встановлюється апаратно наприкінці прийому 8–го біта в режимі 0 або через половину інтервалу стоп–біта в режимах 1, 2, 3 при SM2 = 0. При SM2 = 1 див. опис для біта SM2.

Для програмування послідовного порту призначено також старший біт регістра PCON: PCON.7 (SMOD), за допомогою якого можна у два рази змінювати швидкість передачі даних послідовним портом.

6.3 Режими роботи інтерфейсу

6.3.1 Загальні відомості

Послідовний порт працює в одному з двох режимів: передачі та прийому.

При передачі байт за резидентною шиною даних (РШД) записується у буфер передавача. Імпульсами зсуву, які може формувати таймер/лічильник1, дані у послідовному двійковому коді, починаючи з молодшого значущого розряду, передаються у канал зв'язку (КЗ).

При прийомі під дією імпульсів зсуву, які можуть бути також сформовані в Т/Л1, дані з каналу зв'язку у послідовному двійковому коді, починаючи з молодшого розряду, заповнюють регістр зсуву приймача. Після виконання необхідних перевірок прийнятий байт переписується у буфер приймача, звідки він може бути прочитаний відповідною командою.

Отже, при передачі проводиться перетворення паралельного ДК у послідовний, а при прийомі – навпаки.

Послідовний порт може приймати черговий байт навіть якщо вже прийнятий до цього байт не був прочитаний із регістра приймача. Однак, якщо до закінчення прийому байт, що знаходиться у регістрі приймача, не буде прочитано, попередній прийнятий байт втрачається. Програмний доступ до регістрів приймача і передавача здійснюється зверненням до регістра спеціальних функцій SBUF. При записі в SBUF байт завантажується в регістр передавача, а при читанні SBUF байт читається з регістра приймача.

Прийом і видача байта даних починається з молодшого розряду і закінчується старшим розрядом. Для дозволу прийому необхідно встановити в 1 розряд REN регістра керування SCON (регістр SCON описано вище).

Послідовний порт може бути запрограмовано на один з чотирьох режимів прийому/передачі шляхом програмування розрядів SM0 і SM1 регістра SCON. В усіх чотирьох режимах передача ініціюється будь–якою командою, що використовує SBUF в якості регістра призначення (виконує операцію "Запис у

SBUF"). Прийом у режимі 0 ініціюється одночасним виконанням умов REN = 1 і RI = 0 (REN і RI – розряди регістра керування SCON). В інших режимах прийом ініціюється приходом старт–біта (нульовий рівень) при REN = 1.

6.3.2 Робота послідовного порту в режимі 0

6.3.2.1 Загальні відомості

У режимі 0 послідовний порт працює як восьмирозрядний регістр зсуву. При цьому 8 біт інформації у послідовному коді приймаються і передаються через двонаправлений вивід RxD. На виводі TxD формується сигнал синхронізації зсувів. Швидкість (частота) прийому/передачі в режимі 0 постійна і складає: $f_{BQ}/12$, де f_{BQ} – частота синхронізації мікроконтролера.

Часові діаграми, що ілюструють роботу послідовного порту у режимі 0, показано на рисунках 6.3, 6.4. Всі зображені на цих рисунках сигнали, за винятком RxD і TxD, є внутрішніми сигналами мікроконтролера.

6.3.2.2 Передача в режимі 0

Передача починається будь–якою командою, яка використовує SBUF в якості регістра призначення (виконує операцію "запис у SBUF") (рисунок 6.3).

При виконанні такої команди у фазі S6 P2 виробляється внутрішній імпульс ЗАПИС У SBUF, за яким призначений до передачі байт записується у регістр зсуву передавача і запускається блок керування передачею.

Внутрішня система тактування мікроконтролера організована так, що між сигналом ЗАПИС У SBUF і початком передачі проходить один повний машинний цикл, після чого виробляється внутрішній сигнал ПОСИЛКА, що дозволяє видачу вмісту регістра зсуву передавача на вихід RxD (вивід P3.0 мікроконтролера) та імпульсів синхронізації зсуву на вихід TxD (вивід P3.1 мікроконтролера).

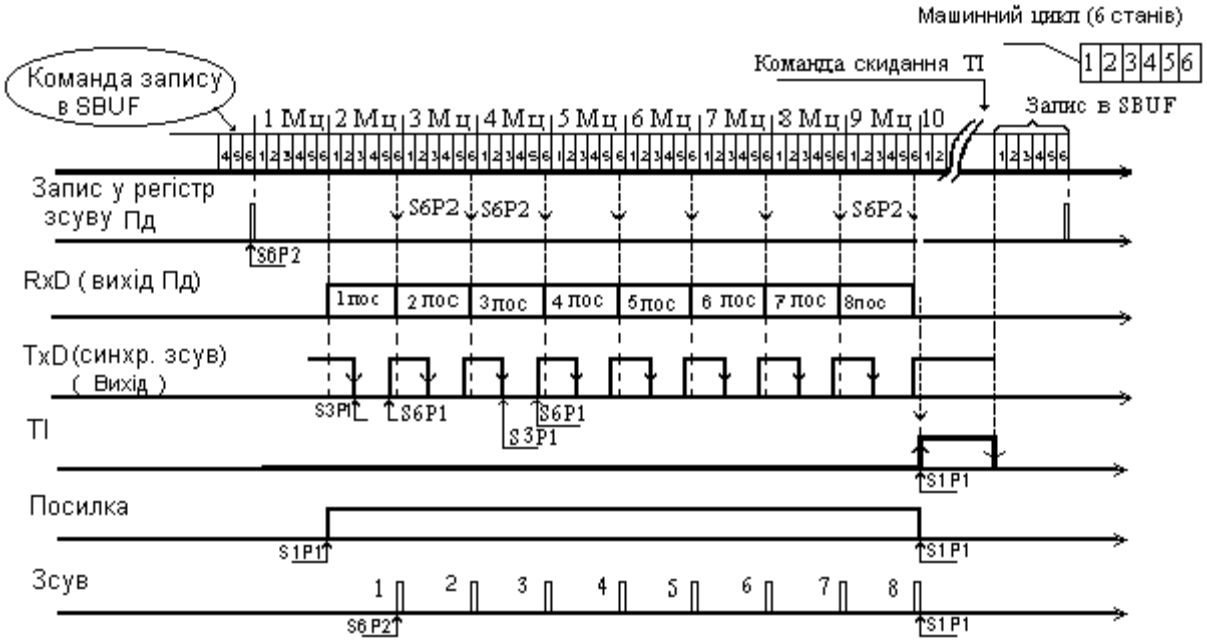


Рисунок 6.3 – Передача в режимі 0

Сигнал СИНХР ЗСУВ має низький рівень у станах S3, S4 і S5 кожного машинного циклу і високий рівень у станах S6, S1 і S2. У фазі S6 P2 кожного машинного циклу, в якому сигнал ПОСИЛКА активний, формується внутрішній імпульс ЗСУВ, за яким вміст регістра зсуву передавача зсувається на одну позицію і на вихід RxD виставляється черговий біт повідомлення, що передається. Усього формується вісім імпульсів ЗСУВ, після чого блок керування передачею знімає сигнал ПОСИЛКА і встановлює прапорець переривання передавача TI (1–й розряд у регістрі SCON). Обидві ці дії виконуються у фазі S1 P1 10–го машинного циклу після сигналу ЗАПИС У SBUF. За перериванням (TI = 1) переривається основна програма, викликається підпрограма, що скидає прапорець TI і записує черговий байт, що передається, у SBUF.

6.3.2.3 Прийом у режимі 0

Прийом починається при одночасному виконанні двох умов: REN = 1 і RI = 0 (рисунок 6.4).

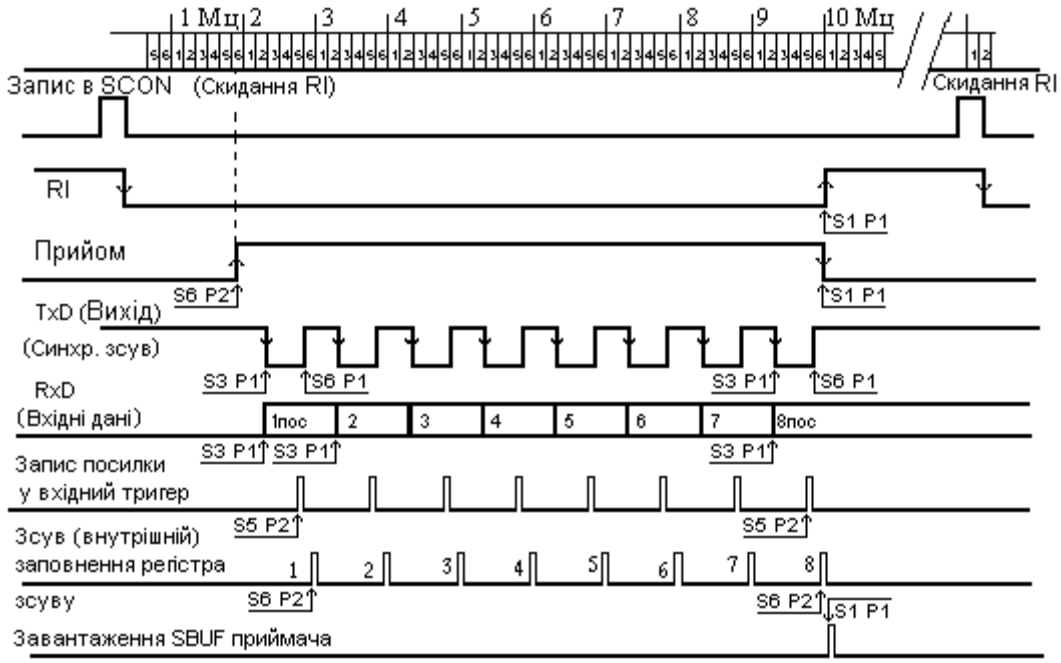


Рисунок 6.4 – Прийом у режимі 0

У фазі S6 P2 наступного машинного циклу блок керування прийомом виробляє внутрішній сигнал ПРИЙОМ, що дозволяє видачу імпульсів СИНХР ЗСУВ на вихід мікроконтролера TxD. Імпульси СИНХР ЗСУВ змінюють свій стан у фазах S3 P1 і S6 P1 і синхронізують моменти надходження повідомлень на вхід RxD (ПРМ) від зовнішніх пристроїв. Біти повідомлень, що приймаються, через вхід RxD надходять на регістр зсуву приймача. Стан входу RxD перевіряється у фазі S5 P2. У фазі S6 P2 кожного машинного циклу, в якому сигнал ПРИЙОМ активний, формується внутрішній імпульс ЗСУВ і вміст регістра зсуву приймача зсувається вліво на одну позицію. Значення, яке при цьому записується в його крайній правий розряд, є значенням сигналу на

вході RxD, який отримується у фазі S5 P2 цього ж машинного циклу. Всього формується вісім імпульсів ЗСУВ, після чого блок керування прийомом формує сигнал завантаження вмісту регістра зсуву приймача у SBUF. У фазі S1 P1 10–го машинного циклу після запису в SCON, який скинув RI у 0, сигнал ПРИЙОМ скидається і встановлюється прапорець переривання приймача RI (0–й біт у регістрі SCON). Далі за RI = 1 виконується переривання основної програми, викликається підпрограма, яка скидає прапорець RI і читає прийнятий байт із буфера ПРМ.

6.3.3 Робота послідовного порту в режимі 1

6.3.3.1 Загальні відомості

У режимі 1 прийом/передача даних здійснюється у форматі восьмирозрядного УАПП. Через TxD передаються, а через RxD приймаються 10 біт: старт–біт (логічний 0), 8 біт даних і стоп–біт (логічна 1). Під час прийому стоп–біт заноситься у біт RB8 регістра SCON. Швидкість (частота) прийому/передачі визначається частотою переповнень таймера/лічильника 1: $F_{OV\ T/C1}$ та в середині УАПП формується за схемою, яку наведено на рисунку 6.5.

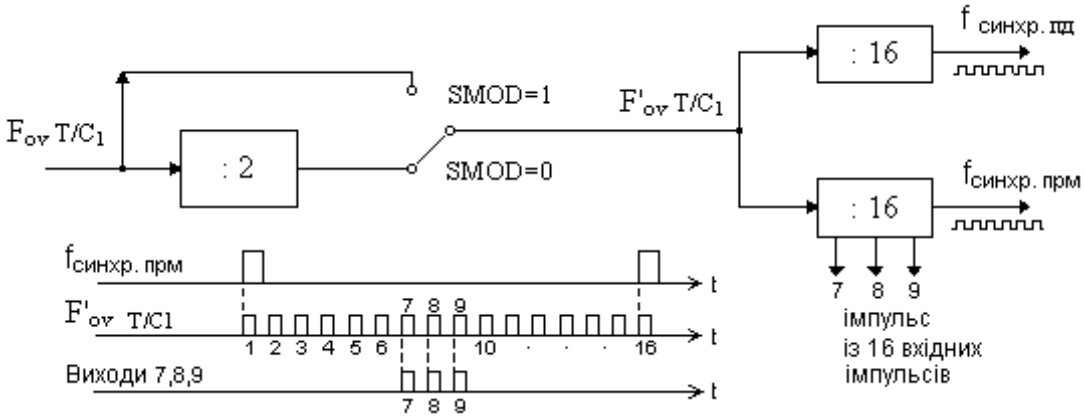


Рисунок 6.5 – Схема формування синхрочастот передачі і прийому в середині МК–51 для послідовного порту, що працює в режимах 1 та 3

В залежності від значення біта SMOD регістра PCON частота, що надходить на вхід дільників на 16, $F'_{OV\ T/C1} = F_{OV\ T/C1}$ при SMOD = 1 і $F'_{OV\ T/C1} = \frac{F_{OV\ T/C1}}{2}$ при SMOD = 0. На виходах дільників на 16 формуються частоти синхронізації передавача $f_{СИНХР.\ ПД}$ і приймача $f_{СИНХР.\ ПРМ}$. На виходах 7, 8, 9 дільника на 16, що формує $f_{СИНХР.\ ПРМ}$, виробляються 3 коротких імпульси, коли на вхід лічильника (дільника) надходять відповідно 7–, 8– і 9–й імпульси частоти $F'_{OV\ T/C1}$, починаючи з початку циклу лічби. Ці імпульси використовуються логічною схемою ідентифікації значення чергової прийнятої посилки (0/1) за мажоритарним принципом.

6.3.3.2 Передача в режимі 1

Передача (рисунок 6.6) ініціюється будь–якою командою, що використовує SBUF в якості регістра призначення, у який виконується запис.

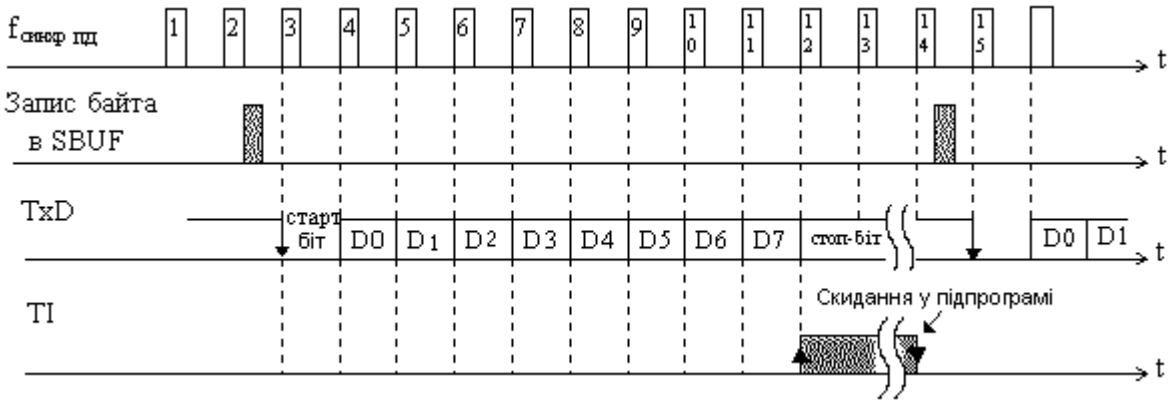


Рисунок 6.6 – Передача в режимі 1

Внутрішній імпульс мікроконтролера ЗАПИС У SBUF, що виробляється при цьому, завантажує призначений до передачі байт у молодші 8 розрядів регістра зсуву передавача та ініціює початок роботи блока керування передачею.

В режимі 1 регістр зсуву передавача має 9 розрядів і в його 9–й розряд за імпульсом ЗАПИС У SBUF заноситься "1" (стоп–біт).

Реально передача починається у фазі S1 P1 машинного циклу, який слідує за найближчим після імпульсу ЗАПИС У SBUF переповненням дільника на 16 у ланцюзі формування сигналу $f_{СИНХР.\ ПД}$ (рисунок 6.5). Таким чином, початок передачі синхронізовано дільником на 16, а не імпульсом ЗАПИС У SBUF. Період сигналу $f_{СИНХР.\ ПД}$ (синхронізація передавача) визначає час, протягом якого біт, що видається, присутній на виході TxD (час передачі біта).

Передача починається встановленням активного рівня внутрішнього сигналу мікроконтролера ПОСИЛКА, поява якого викликає видачу на вихід TxD рівня старт–біта (нуль). Після цього через час передачі одного біта стає активним внутрішній сигнал мікроконтролера ДАНІ, що дозволяє видачу вмісту регістра зсуву передавача на вихід TxD (вивід P3.0 мікроконтролера). При появі активного сигналу ДАНІ старт–біт на виході TxD замінюється бітом D0 регістра зсуву передавача. По закінченні часу передачі біта D0 формується перший внутрішній імпульс мікроконтролера ЗСУВ, за яким вміст регістра зсуву передавача зсувається на один розряд, і біт D0 на виході TxD замінюється бітом D1. Усього формується 9 імпульсів ЗСУВ, у результаті чого на вихід TxD видаються 8 біт даних і стоп–біт. По закінченні видачі всіх біт повідомлення блок керування передачею встановлює прапорець переривання передавача TI і знімає сигнали ПОСИЛКА і ДАНІ. За перериванням відбувається скидання прапорця TI, запис нового байта і т. ін.

6.3.3.3 Прийом у режимі 1

Прийом (рисунок 6.7) починається при виявленні переходу сигналу на вході RxD з "1" в "0". Для того, щоб виявити такий перехід, вхід RxD апаратно опитується з частотою $F'_{OV\ T/C1}$ (рисунок 6.5), тобто 16 разів на одну посилку.

Коли перехід сигналу на вході RxD із "1" в "0" виявлено, негайно скидається лічильник (дільник на 16) у ланцюзі формування сигналу $f_{СИНХР.\ ПРМ}$ (рисунок 6.5), у результаті чого відбувається поєднання моментів переповнення цього лічильника (імпульси $f_{СИНХР.\ ПРМ}$ на рисунку 6.5) із границями зміни бітів прийнятого повідомлення на вході RxD.

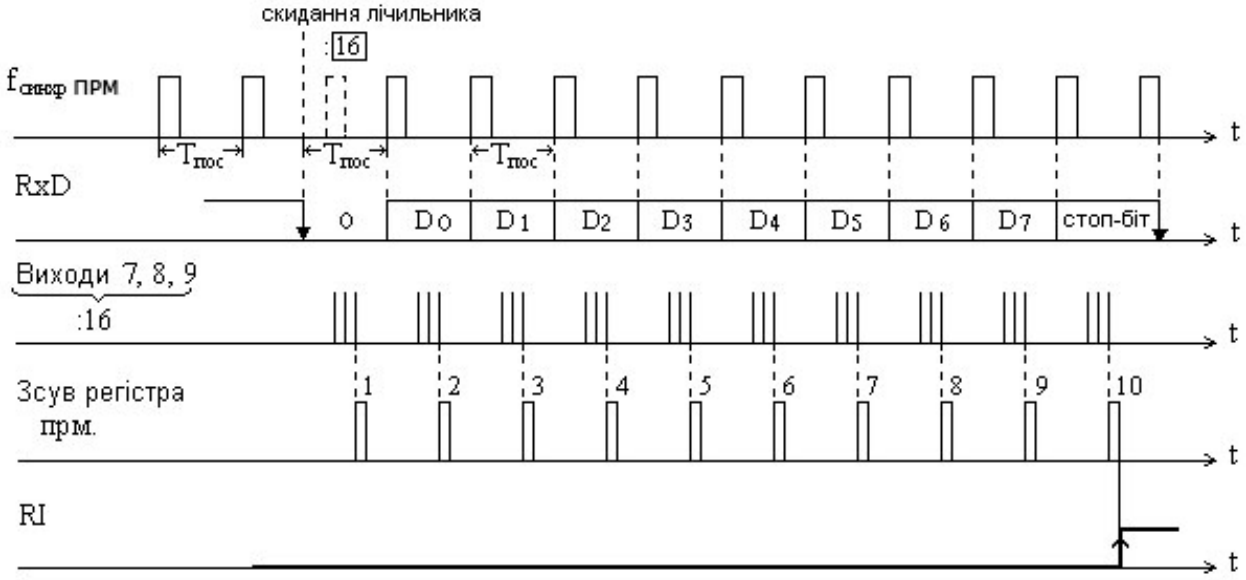


Рисунок 6.7 – Прийом у режимі 1

Шістнадцять станів лічильника (дільника) поділяють час, протягом якого кожний біт прийнятого повідомлення присутній на вході RxD, на 16 фаз, з 1–ї по 16–у для кожного біта. У фазах 7, 8 і 9 спеціальний пристрій мікроконтролера: біт–детектор, зчитує із входу RxD три значення прийнятого біта, та за мажоритарним принципом "2 або 3 із 3–х" вибирає одне з них і подає його на вхід регістра зсуву приймача. Блок керування прийомом при цьому формує внутрішній імпульс мікроконтролера ЗСУВ, у результаті чого вміст регістра зсуву приймача зсувається на один розряд і прийнятий біт заноситься у регістр зсуву приймача. Усього формується 10 імпульсів ЗСУВ, а регістр зсуву приймача у режимі 1 є 9–розрядним. Тому після 10–го імпульсу ЗСУВ у регістрі зсуву приймача знаходяться біти даних D0…D7 і стоп–біт. Після 10–го

імпульсу ЗСУВ блок керування прийомом завантажує дані з регістра зсуву приймача у SBUF, завантажує стоп–біт з регістра зсуву приймача у розряд RB8 регістра SCON і встановлює прапорець переривання приймача RI. Сигнал завантаження SBUF, RB8 і встановлення RI виробляється блоком керування прийомом тільки в тому випадку, якщо в момент генерації останнього імпульсу ЗСУВ виконуються наступні умови:

– RI = 0;

– або SM2 = 0;

– або SM2 = 1 та прийнятий стоп–біт дорівнює "1".

Якщо хоча б одна з цих умов не виконується, прийняте повідомлення безповоротно втрачається, а прапорець RI не встановлюється. Якщо наведені вище умови виконано, стоп–біт записується у RB8, вісім біт даних надходять у SBUF і встановлюється прапорець RI. У цей час, незалежно від виконання наведених вище умов, послідовний порт знову починає відслідковувати наявність переходу сигналу з "1" у "0" на вході RxD і прийом нового байта. До закінчення цього процесу попередній байт повинний бути прочитаний із буфера ПРМ, інакше буде накладання нового прийнятого байта на старий.

Якщо мажоритарний відбір при прийомі першого біта повідомлення (старт–біт) показує не нульове значення біта, всі пристрої блоку прийому скидаються, і починається відслідковування наступного переходу сигналу з "1" в "0" на вході RxD. Таким чином, забезпечується захист від помилкових старт–бітів.

6.3.3.4 Робота послідовного порту в режимах 2 і 3

Режими 2 і 3 – це режими 9–розрядного УАПП з постійною (режим 2) та змінною (режим 3) швидкостями обміну. У цих режимах 11 біт передаються/приймаються відповідно через виводи TxD/RxD у наступній послідовності: старт–біт, 9 біт даних, стоп–біт. 9–й біт даних при передачі

визначаються вмістом розряду TB8 регістра SCON. При прийомі 9–й біт даних заноситься у біт RB8 регістра SCON.

Швидкість (частота) прийому/передачі у режимі 2 (рисунок 6.8) програмно налаштовується на одну з двох можливих величин: $f_{BQ}/32$ і $f_{BQ}/64$, де f_{BQ} – частота синхронізації мікроконтролера.

У режимі 3 швидкість (частота) прийому/передачі визначається частотою переповнень таймера/лічильника1: $F_{OV\ T/C1}$ (рисунок 6.5).

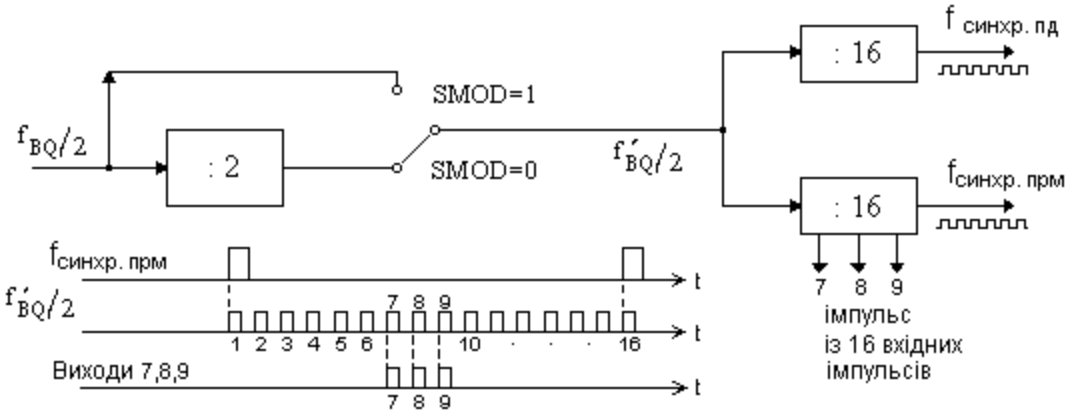


Рисунок 6.8 – Схема формування синхрочастот передачі і прийому всередині МК–51 для послідовного порту, що працює у режимі 2

Розходження у швидкості (частоті) прийому/передачі є єдиною відмінністю між режимом 2 і режимом 3. В усьому іншому ці два режими є цілком ідентичними.

Часові діаграми, що ілюструють роботу послідовного порту в режимах 2 і 3, наведено у [2,3].

Робота УАПП у режимах 2 і 3 дуже схожа на режим 1. Але при цьому існує ряд відмінностей:

– у форматі даних, якими відбувається обмін. Після восьми інформаційних, перед стоп–бітом, присутній 9–й біт, що програмується. При

передачі значення 9–го біта визначається значенням розряду TB8 регістра SCON. При прийомі 9–й біт фіксується у розряді RB8 регістра SCON;

– якщо біт SM2 регістра SCON встановлено в одиницю, то повідомлення, у якому 9–й біт дорівнює нулю, бракується (втрачається). Тобто прапорець RI не встановлюється, і переривання основної програми при прийомі не відбувається;

– на часових діаграмах роботи передавача у режимах 2 і 3 у порівнянні з режимом 1 буде додано ще один біт (TB8) і цикл передачі подовжується на один такт (період частоти синхронізації передавача);

– на часових діаграмах роботи приймача у режимах 2, 3 буде додано ще один прийнятий біт (RB8) перед стоп–бітом. Крім того, детектування стоп–біта не відбувається, і прапорець RI встановлюється після 10–го зсуву, тобто після фіксації RB8

6.4 Швидкість передачі–прийому даних через послідовний порт

Швидкість (частота передачі бітів) через послідовний інтерфейс $V_{пд}$ в залежності від режиму роботи послідовного порту визначається або частотою синхронізації мікроконтролера f_{BQ} (режими 0 і 2), або частотою переповнення таймера/лічильника1 $F_{OV\ T/C1}$ (режими 1 і 3).

У режимі 0 швидкість послідовного обміну максимальна. Вона постійна і складає: $$V_{пд} = f_{BQ}/12 \ \ [біт/с]. \qquad (6.1)$$

При необхідності працювати в асинхронному режимі зі зміненою у 2 рази швидкістю використовується режим 2. У цьому режимі швидкість послідовної передачі залежить від стану біта SMOD регістра SCON і частоти f_{BQ}:

$$V_{пд} = (2^{SMOD}/64) * f_{BQ} \ \ [біт/с]. \qquad (6.2)$$

Тобто при SMOD = 0, $V_{пд} = f_{BQ}/64$, а при SMOD = 1, $V_{пд} = f_{BQ}/32$. За сигналом “скидання” біт SMOD скидається в нуль. Для встановлення в

одиницю біта SMOD використовуються команди з адресацією байтів, наприклад, команда MOV 87H, #80H.

У режимах 1, 3 є можливість змінити швидкість асинхронної послідовної передачі у більш широкому діапазоні:

$$V_{пд} = (2^{SMOD}/32) * F_{OV\ T/C1}\ \ [біт/с], \qquad (6.3)$$

де $F_{OV\ T/C1}$ – частота переповнення Т/Л1.

Для використання Т/Л1 в якості джерела для завдання швидкості обміну необхідно:

– заборонити переривання від Т/Л1;

– запрограмувати роботу Т/Л1 в якості таймера або в якості лічильника, встановивши при цьому для нього один з режимів 0, 1 або 2;

– увімкнути Т/Л1 на лічбу.

Звичайно для синхронізації послідовного порту таймер Т/Л1 програмується у режим автозавантаження (режим 2). У цьому випадку швидкість послідовного обміну визначається за формулою:

$$V_{пд} = (2^{SMOD} * f_{BQ})/(32 * 12 * [256–(TH1)])\ \ [біт/с], \qquad (6.4)$$

де (TH1) – вміст регістра TH1 у десятковому коді.

Якщо необхідний послідовний обмін з дуже низькою швидкістю, то можна використовувати Т/Л1 у режимі 16–розрядного таймера (режим 1), дозволивши при цьому переривання від Т/Л1 з метою перезавантаження TL1/TH1 у підпрограмі обслуговування переривання. У таблиці 6.3 наведено ряд стандартних швидкостей послідовного обміну і те, як вони можуть бути реалізовані в МК–рі.

Таблиця 6.3 – Формування стандартних швидкостей обміну через послідовний порт

Режим роботи послідовного порту	Швидкість прийому/ передачі, Кбод	f_{BQ}, МГц	SMOD	Розряди TMOD			TH1	Примітка
				C/T	M1	M0		
Режим 0	Макс.: 1000	12	X	X	X	X	X	
Режим 2	Макс.: 375	12	1	X	X	X	X	
Режим 1, 3	62,5	12	1	0	1	0	FFH	
	19,2	11,059	1	0	1	0	FDH	
	9,6	11,059	0	0	1	0	FDH	
	4,8	11,059	0	0	1	0	FAH	
	2,4	11,059	0	0	1	0	E4H	
	1,2	11,059	0	0	1	0	E8H	
	0,1375	11,986	0	0	1	0	18H	
	0,110	6	0	0	1	0	72H	
	0,110	12	0	0	0	1	FEH	TL1 = EBH

У таблиці 6.4 наведено зведену інформацію з усіх чотирьох режимів роботи послідовного порту МК–ра сімейства МК–51, наприклад, AT89C51.

Таблиця 6.4 – Зведена інформація з усіх режимів роботи послідовного порту

Режим обміну	Вид обміну	Розряди регістра SCON							Швидкість передачі	Примітка
		SM0	SM1	SM2	REN	TB8	RB8	ПРАПОРЕЦЬ		
0	ПД	0	0	0	–	–	–	TI	$f_{BQ}/12$	
	ПРМ				1			RI		Для ініціалізації прийому встановити: RI=0
1	ПД	0	1	–	–	–	–	TI	$\frac{2^{SMOD} \cdot F_{OV}}{32}$	
	ПРМ			0	1		стоп–біт	RI		
				1			1			
							0	–		Прийнятий байт губиться
2	ПД	1	0	–	–	9–й біт даних	–	TI	$\frac{2^{SMOD} \cdot f_{BQ}}{32}$	
	ПРМ			0	1		9–й біт даних	RI		
				1			1			
							0	–		Прийнятий байт губиться
3	ПРД	1	1	–	–	9–й біт даних	–	TI	$\frac{2^{SMOD} \cdot F_{OV}}{32}$	
	ПРМ			0	1		9–й біт даних	RI		
				1			1			
							0	–		Прийнятий байт губиться

6.5 Приклад програмування асинхронного послідовного порту мікроконтролера сімейства МК–51

Вихідні дані для програмування:

– швидкість обміну – 110 біт/с (бод);

– f_{BQ} = 6 МГц;

– режим роботи послідовного порту – 3;

– режим роботи Т/Л1 – 2;

– вид обміну – під керуванням мікропроцесора (програмно–керуючий);

– біт SMOD регістра PCON сигналом "RESET" встановлено у 0.

Нижче наведено приклад програми ініціалізації послідовного порту і фрагменти програмно–керуючого обміну:

```
                        ; ініціалізація послідовного порту
                        ; для роботи зі швидкістю 110 бод
                        ; на частоті тактового сигналу 6 МГц;
INT1:  CLR  TCON.6      ; зупинка таймера Т/Л1;
       CLR  IE.3        ; заборона переривань від Т/Л1;
       CLR  IE.4        ; заборона переривань від УАПП;
       MOV  TH1,#72H    ; значення, що автоматично завантажується,
                        ; для отримання швидкості 110 бод;
MOV  SCON,#11011000B    ; встановлення режиму 9–розрядного УАПП;
MOV  TMOD,#00100000B    ; встановлення режиму автозавантаження
                        ; таймера 1;
       SETB  TCON.6     ; запуск таймера1;
                        ; прийом символу від зовнішнього пристрою;
```

```
CIN:  JNB  RI,CIN        ; очікування завершення прийому;
      MOV  A,SBUF        ; читання отриманого символу;
      CLR  RI            ; скидання прапорця прийому;
                         ; передача символу на зовнішній пристрій;
COUT: JNB  TI,COUT       ; очікування закінчення передачі
                         ; попереднього символу;
      CLR  TI            ; скидання прапорця передачі;
      MOV  SBUF,A        ; видача наступного символу.
```

6.6 Особливості міжконтролерного обміну інформацією в локальних керуючих мережах при використанні послідовного порту

Режими 2 і 3 послідовного порту дозволяють організувати роботу МК–рів у багатопроцесорних системах, які використовують для обміну інформацією між МК–ми, наприклад, моноканал, що розділяється (коаксіальний кабель, звита пара, оптоволокно). У цих режимах приймається дев'ять біт даних і 9–й прийнятий біт записується у біт RB8 регістра SCON. При цьому, якщо біт SM2 регістра SCON встановлено в "1", то після прийому останньої посилки прапорець переривання приймача RI буде встановлено тільки в тому випадку, якщо RB8 = 1. Цю особливість роботи послідовного порту у режимах 2 і 3 можна використовувати для організації міжконтролерного обміну наступним чином.

Коли ведучий МК–р хоче передати блок даних одному з ведених МК–в, він видає в моноканал посилку з адресою веденого МК, котрому буде передаватися блок даних. Адресна відправка відрізняється від відправки з даними тим, що в адресній відправці 9–й біт даних дорівнює "1", а у відправці з даними – "0". Таким чином, при SM2 = 1 жоден з ведених МК–в не буде реагувати на відправку з даними, але усі ведені зреагують на адресну відправку.

Проаналізувавши отриману адресу, той МК–р, що адресується, скидає свій біт SM2, а інші залишають його без зміни і знову переходять до виконання перерваної програми. Після цього ведучий МК–р може починати видачу в моноканал блоку даних, на відправки якого буде реагувати тільки МК–р, у котрого SM2 = 0.

Біт SM2 ніяк не бере участі у роботі послідовного порту в режимі 0. У режимі 1 біт SM2 може використовуватися для контролю правильності прийнятого стоп–біта: в режимі 1, якщо SM2 = 1, прапорець переривання приймача RI не буде встановлено, якщо прийнятий стоп–біт не дорівнює "1".

6.7 Розвиток архітектури послідовних портів у сучасних мікроконтролерах сімейства МК–51

Всі мікроконтролери МК–51 мають асинхронний інтерфейс для послідовного обміну інформацією з іншими мікроконтролерами або персональними комп'ютерами.

Стандартну архітектуру асинхронного послідовного порту було розглянуто вище. Всі мікроконтролери даного сімейства мають у своєму складі модуль УАПП (UART). До складу сучасних мікроконтролерів даного сімейства також входять модулі синхронного послідовного обміну інформацією: SPI та I^2C. Розглянемо роботу кожного інтерфейсу більш детально.

6.7.1 Послідовний периферійний інтерфейс (SPI)

6.7.1.1 Загальна характеристика

Послідовний синхронний периферійний інтерфейс SPI (Serial Peripheral Interface) мікроконтролерів сімейства Cygnal представляє собою повнодуплексний чотирьохпровідний інтерфейс з шинною конфігурацією підключення вузлів (пристроїв). SPI–інтерфейс дозволяє підключати до одного ведучого або головного (Master) вузлу кілька ведених (Slave) вузлів через спільну шину. Окремий сигнал NSS (Slave–Select signal), що надсилається

ведучим пристроєм використовується при виборі веденого пристрою при здійсненні з ним обміну даними. Крім того, можлива також побудова системи з багатьма ведучими вузлами. При цьому передбачено виявлення конфліктів при одночасній передачі від кількох ведучих вузлів.

Інтерфейс SPI може бути запрограмований для роботи в якості ведучого (Master) або веденого (Slave). Якщо інтерфейс запрограмований як ведучий, він може працювати на максимальній швидкості передачі даних (bits / sec), що дорівнює половині тактової частоти. Якщо інтерфейс запрограмований для роботи в якості веденого, його максимальна швидкість в дуплексному режимі дорівнює одній десятій тактової частоти. Мається на увазі, що джерелом синхронізації в обох випадках є системний генератор тактової частоти. Якщо ведучий інтерфейс виробляє SCK, NSS і послідовні дані асинхронно, максимальна швидкість передачі повинна бути менше однієї десятої тактової частоти.

Існує ще один особливий режим, коли ведучий повинен тільки передавати дані веденому (напівдуплексний режим), і не повинен приймати дані від нього. У цьому випадку максимальна швидкість передачі, при синхронному режимі роботи складає одну четверту від системної тактової частоти.

6.7.1.2 Робота SPI–інтерфейсу за функціональною схемою

Типову структурну схему SPI–мережі наведено на рисунку 6.9, а функціональну схему SPI–інтерфейсу показано на рисунку 6.10.

Інтерфейс SPI має чотири сигнальні лінії: MOSI, MISO, SCK та NSS.

Лінія MOSI (Master–Out, Slave–In) – вихідна лінія даних ведучого інтерфейсу і вхідна лінія даних веденого інтерфейсу. З назви випливає, що лінію призначено для передачі даних від ведучого (Master) інтерфейсу (або вузла мережі) до веденого (Slave) інтерфейсу (або вузла мережі).

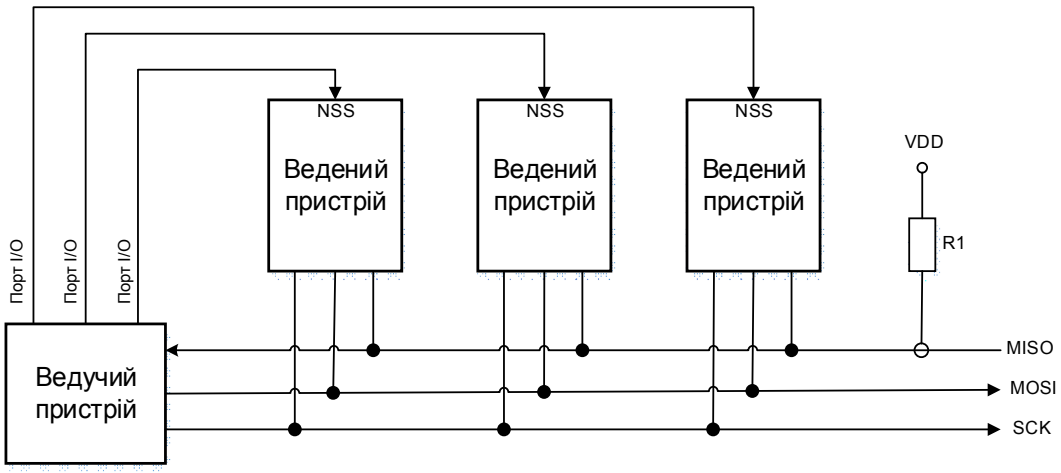


Рисунок 6.9 – Структурна схема SPI–мережі

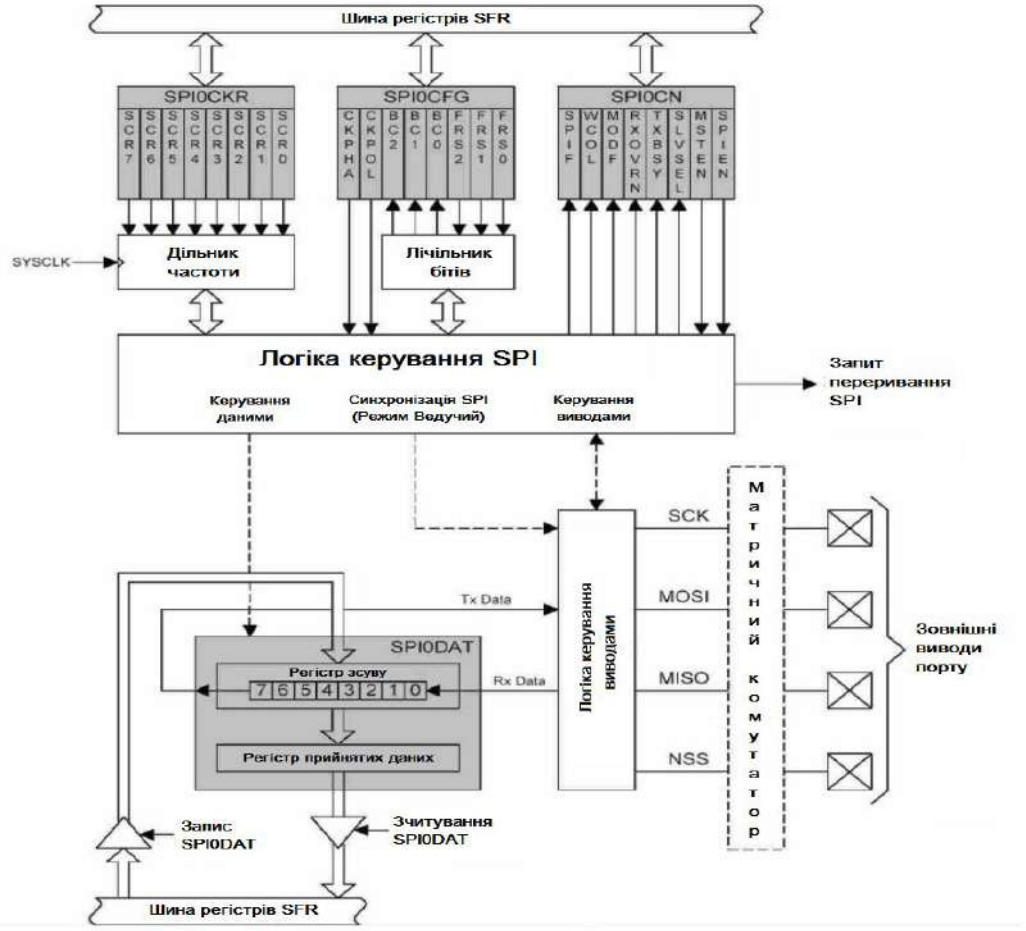


Рисунок 6.10 – Функціональна схема SPI–інтерфейсу

Лінія MISO (Master–In, Slave–Out) – вхідна лінія даних ведучого інтерфейсу і вихідна лінія даних веденого інтерфейсу. Лінію призначено для передачі даних від веденого інтерфейсу до ведучого. Дані передаються байтами, бітами, починаючи зі старшого біта. Слід пам'ятати, що вихід MISO веденого інтерфейсу знаходиться у високоімпедансному стані, якщо ведений інтерфейс не обрано за лінією NSS.

Лінію NSS (Slave Select) – лінію вибірки веденого, призначено для вибірки ведучим веденого інтерфейсу низьким логічним потенціалом.

Лінія SCK (Serial Clock) – вихідна лінія тактових імпульсів ведучого вузла і вхідна лінія тактових імпульсів веденого вузла. Лінія SCK використовується для синхронізації передачі даних між ведучим і веденим інтерфейсами за лініями MOSI та MISO.

У мережі на базі SPI–інтерфейсів одночасно тільки один інтерфейс може бути ведучим. Інтерфейс налаштовується на режим ведучого встановленням прапорця MSTEN (Master Enable flag) – біта SPI0CN.1 регістра SPI0CN. Якщо інтерфейс налаштовано на режим ведучого, то запис байта даних у регістр даних SPI0DAT приводить до початку передачі. Ведучий інтерфейс негайно побітно зсуває дані і видає їх на лінію MOSI в супроводі тактових імпульсів на лінії SCK. Після завершення передачі встановлюється прапорець SPIF (SPI0CN.7). Якщо дозволено переривання, то виконується відповідна підпрограма обробки цього переривання. Крім того, інтерфейс може бути запрограмований на видачу від одного до восьми бітів для здійснення зв'язку з SPI–приладами, що мають різну довжину слова. Довжину передачі (кількість переданих бітів) може бути задано бітами FRS [2:0] в регістрі конфігурації SPI0CFG (SPI Configuration Register).

З'єднання двох мікроконтролерів (ведучий–ведений) за інтерфейсом SPI приведено на рисунку 6.11.

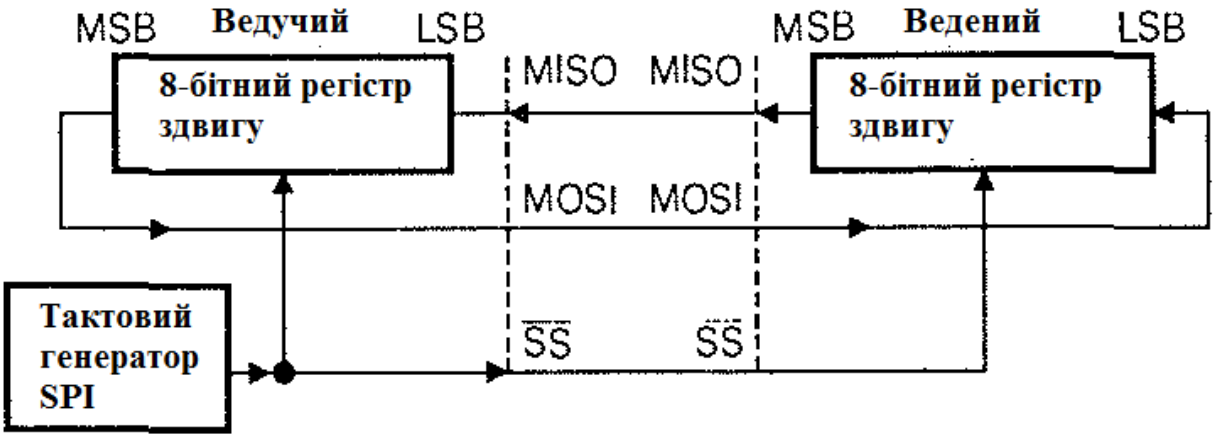


Рисунок 6.11 – Схема з'єднання вузлів SPI двох різних пристроїв

Вище вже зазначалося, що інтерфейс може працювати в дуплексному режимі. Це означає, що можлива одночасна передача даних за лінією MOSI від ведучого до веденого, і лінією MISO від веденого до ведучого. Дані, які отримано від веденого інтерфейсу, замінюють дані в регістрі даних ведучого інтерфейсу. Цей регістр двічі буферизований на введення, але не на виведення. Тобто якщо в регістр даних SPI0DAT здійснюється спроба запису даних під час передачі попереднього байта, встановлюється прапорець WCOL (SPI0CN.6) і спроба запису ігнорується.

Таким чином, поточна передача даних триває безперервно. Зчитування з регістра даних SPI0DAT призводить до зчитування приймального буфера. Якщо прийом не закінчено, встановлюється прапорець RXOVRN (SPI0CN.4). Нові дані не передаються в регістр зчитування, доки попередній прийнятий байт не буде прочитано. Очевидно, що при затримці зчитування прийнятих байтів може відбутися втрата даних. Якщо SPI–інтерфейс не налаштовано, як ведучий (Master), він буде працювати в режимі веденого (Slave).

Крім того, підтримується режим мережі з багатьма ведучими. Прапорець помилки режиму MODF (SPI0CN.5–Mode Fault flag) встановлюється в логічну одиницю, якщо інтерфейс визначено як ведучий (MSTEN = 1) і вивід NSS переведено в низький логічний рівень, тобто SPI–інтерфейс намагаються

використовувати в якості веденого. Якщо встановлено прапорець MODF, біти MSTEN та SPIEN в регістрі керування SPI апаратно автоматично скидаються, переводячи інтерфейс в автономний стан. Таким чином, у системі з багатьма ведучими, ядро може визначити чи вільна шина шляхом опитування прапорця SLVSEL (SPI0CN.2) перед тим, як встановити прапорець MSTEN (тобто визначити інтерфейсу режим ведучого) і ініціалізувати обмін.

6.7.1.3 Часові діаграми роботи інтерфейсу

Часові діаграми роботи SPI–інтерфейсу показано на рисунку 6.12.

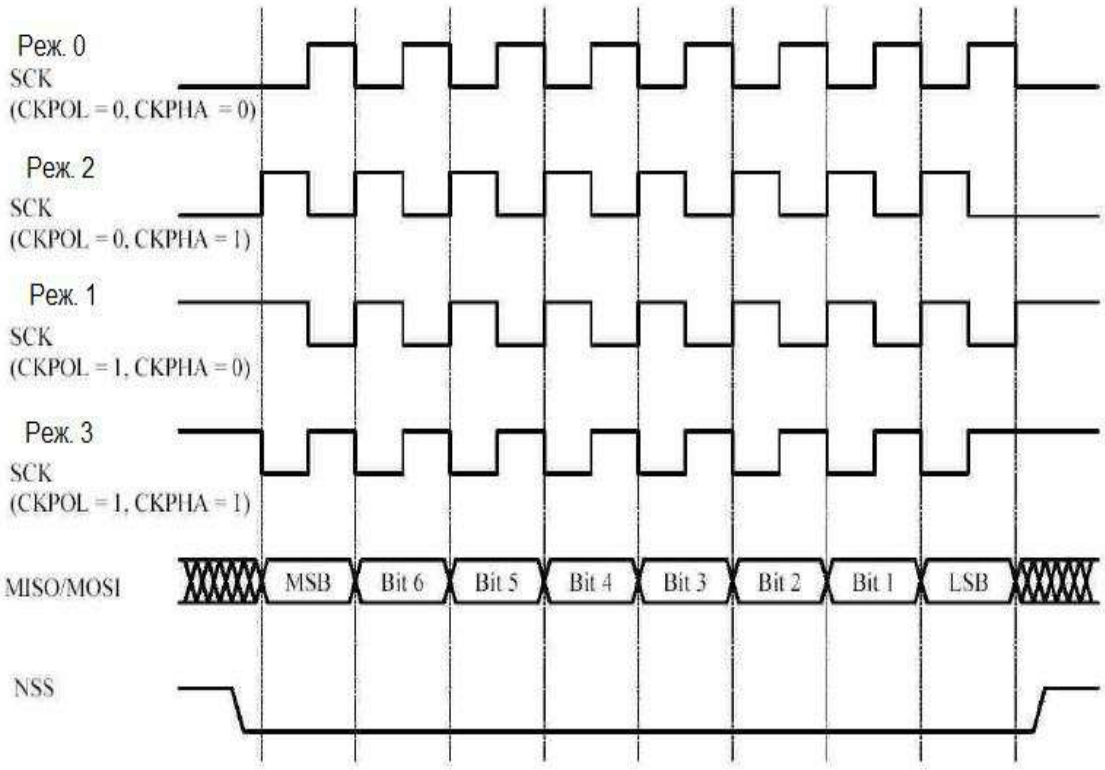


Рисунок 6.12 – Часові діаграми роботи SPI–інтерфейсу

Можливі чотири комбінації фаз тактових імпульсів і їх полярностей залежно від комбінації керуючих бітів у регістрі конфігурації SPI0CFG (SPI Configuration Register). Біт CKPHA (SPI0CFG.7) вибирає одну з двох фаз тактових імпульсів, тобто фронт, за яким здійснюється запис даних. Інший біт

CKPOL (SPI0CFG.6) визначає активну полярність (високий чи низький рівень). Очевидно, що і ведучий, і ведений вузли повинні мати однакові налаштування фази і полярності. Ще одна важлива особливість налаштування полягає в тому, що інтерфейс SPI повинен бути заборонений шляхом скидання бітів SPIEN (SPI0CN.0) на час налаштування фази і полярності тактових імпульсів.

Перед входами восьмирозрядних регістрів зсуву ведучого та веденого мікроконтролерів (рисунок 6.11) знаходяться два синхронних тригери (буфери) (на рисунку не показані), в які першим перепадом сигналу на лінії SCK записується значення сигналу, яке присутнє на їх інформаційному вході.

Другим перепадом імпульсів на лінії SCK відбувається зсув інформації вліво відповідно до рисунку 6.11. При цьому стан буферів переписується в молодші розряди регістрів зсуву, а черговий вихідний біт з регістрів зсуву виставляється на лінії MOSI/MISO.

Вказаний зсув у часі між моментом видачі чергового розряду в лінію зв’язку між мікроконтролерами і моментом фіксації цього біта в буфері дозволяє компенсувати часові затримки при передачі сигналів між мікроконтролерами.

Далі аналогічно здійснюється обмін між ведучим та веденим всіма наступними бітами. З кожним непарним/парним перепадом сигналу SCK відбувається фіксація чергового біта в буфері, а з кожним парним/непарним – зсув інформації вліво.

Крім описаних SFR–регістрів, при налаштуванні SPI–інтерфейсу використовуються регістр налаштування швидкості передачі (SPI0CKR).

6.7.1.4 Програмування інтерфейсу

6.7.1.4.1 SPI0CFG – Регістр конфігурації інтерфейсу SPI

Назва регістра:	SPI0CFG – SPI Configuration Register		
SFR–адреса:	0x9A	Значення після скидання:	00000111b (0x07)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
CKPHA	CKPOL	BC2	BC1	BC0	SPIFRS2	SPIFRS1	SPIFRS0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Біт 7: CKPHA – SPI Clock Phase – біт керування фазою тактування.

0 – Дані дійсні за переднім фронтом тактових імпульсів SCK.

1 – Дані дійсні за заднім фронтом тактових імпульсів SCK.

Біт 6: CKPOL – SPI Clock Polarity – біт керування полярністю тактових імпульсів.

0 – Не активний стан лінії SCK – логічний 0.

1 – Не активний стан лінії SCK – логічна 1.

Біти 5…3: BC2…0 – SPI Bit Count – індикатор поточного біта, який передається.

000 – передається молодший 0–й біт.

001 – передається 1–й біт.

...

111 – передається старший 7 біт.

Біти 2…0: SPIFRS2…0 – SPI Frame Size – розмір кадру (довжини слова).

000 – 1 біт.

001 – 2 біти.

...

111 – 8 біт.

6.7.1.4.2 SPI0CN – Регістр керування інтерфейсом SPI

Назва регістра:	SPI0CN– SPI Control Register		
SFR–адреса:	0x8F	Значення після скидання:	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
SPIF	WCOL	MODF	RXOVRN	TXBSY	SLVSEL	MSTEN	SPIEN
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Біт 7:

SPIF – SPI Interrupt Flag – прапорець переривання від інтерфейсу SPI. Прапорець встановлюється апаратно після завершення передачі даних. Якщо дозволено переривання, генерується переривання від модуля SPI. Біт повинен скидатися програмно.

Біт 6:

WCOL – Write Collision Flag – прапорець помилки запису. Прапорець встановлюється апаратно (і генерується переривання, якщо воно дозволено) при записі байта в регістр під час не завершеної передачі. Біт повинен скидатися програмно.

Біт 5:

MODF – Mode Fault Flag – прапорець помилки режиму. Прапорець встановлюється апаратно (і генерується переривання, якщо воно дозволено) коли в режимі ведучого (MSTEN = 1) виявлено NSS = 0. Біт повинен скидатися програмно.

Біт 4:

RXOVRN – Receive Overrun Flag – прапорець переповнення прийому. Біт встановлюється апаратно (і генерується переривання, якщо воно дозволено) у випадку, якщо при прийомі останнього біта в буферному регістрі знаходиться

не прочитаний байт даних, який отримано раніше. Біт повинен скидатися програмно.

Біт 3:

TXBSY – Transmit Busy Flag – біт зайнятості передавача. Біт встановлюється апаратно при поточній передачі і знімається апаратно після її завершення.

Біт 2:

SLVSEL – Slave Selected Flag – прапорець вибору веденого. Біт апаратно встановлюється в 1, якщо NSS = 0, і апаратно скидатися при NSS = 1.

Біт 1:

MSTEN – Master Mode Enable – біт дозволу (1) режиму ведучого.

Біт 0:

SPIEN – SPI Enable – біт дозволу (1) інтерфейсу SPI.

6.7.1.4.3 SPI0CKR – Регістр керування швидкістю інтерфейсу SPI

Назва регістра:	SPI0CKR– SPI Clock Rate Register		
SFR–адреса:	0x9D	Значення після скидання:	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
SCR7	SCR6	SCR5	SCR4	SCR3	SCR2	SCR1	SCR0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Біти 7…0: SPI0CKR– SPI Clock Rate – байт визначення швидкості (частоти SCK):

$F_{SCK} = 0.5*F_{SYSCLK}/(SPI0CKR +1)$, для $0 \leq SPI0CKR \leq 255$.

6.7.1.4.4 SPI0DAT – Регістр даних інтерфейсу SPI

Назва регістра:	SPI0CKR– SPI Data Register		
SFR–адреса:	0x9B	Значення після скидання:	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Біти 7…0: SPI0 Transmitt and Receive Data – регістр даних

6.7.2 Інтерфейс I^2C

6.7.2.1 Загальна характеристика

Інтерфейс I^2C являється двопровідним послідовним синхронним інтерфейсом, який розроблено фірмою Philips Corporation. Він призначений для зв'язку між інтегральними мікросхемами або модулями. Існує ціла група I^2C–сумісних пристроїв для різних додатків: цифро–аналогові й аналого–цифрові перетворювачі, мікросхеми пам'яті, мікроконтролери, що містять модуль I^2C і т.ін.

Шина інтерфейсу I^2C складається із двох ліній:

– двонаправленої лінії даних (SDA);

– лінії тактових (синхро) імпульсів (SCL).

На рисунку 6.13 приведено структурну схему типової мережі, що використовує для обміну даними інтерфейс (шину) I^2C.

Лінії SDA і SCL шини з'єднано з додатним полюсом напруги живлення (+Vcc) через резистори, що підтягують, R_1 та R_2.

Передавач генерує та передає повідомлення, а приймач його приймає.

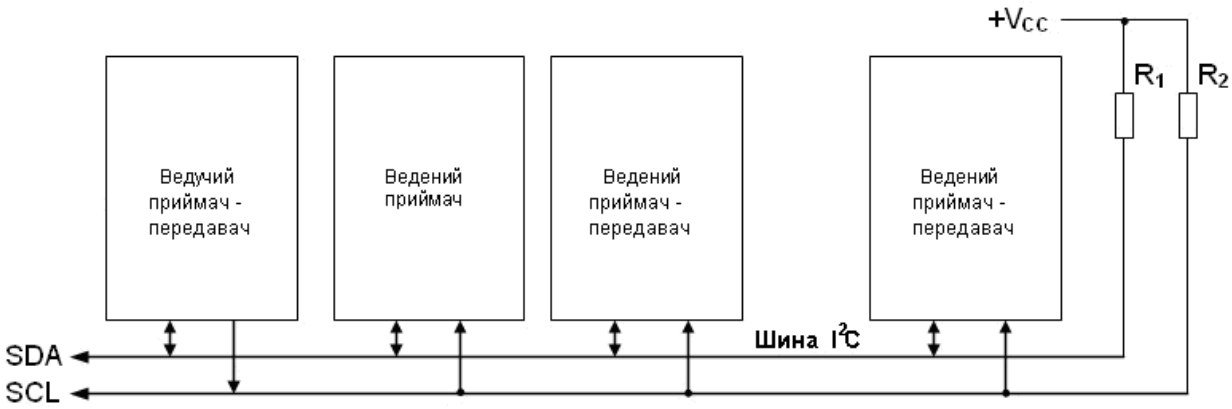


Рисунок 6.13 – Структурна схема мережі з інтерфейсом I^2C

Один із двох пристроїв, що беруть участь в обміні, являється ведучим (master), а інший – веденим (slave). Ведучий пристрій керує роботою шини та формує тактові сигнали (синхросигнали) SCL.

Кожний пристрій, що використовує для обміну інтерфейс I^2C, має свою адресу. Коли ведучий пристрій бажає ініціювати обмін даними, він передає на лінію SDA адресу пристрою, з яким буде виконуватися обмін (передача/прийом). Всі ведені пристрої стежать за адресою, що виставляється на шину, і порівнюють її із власною адресою. Після адреси ведучий передає біт напрямку $R/\overline{W}$, що визначає чи буде ведучий читати дані від веденого ($R/\overline{W}$ = 1) або буде передавати дані веденому ($R/\overline{W}$ = 0). Ведений приймач після одержання адреси або даних видає на шину SDA біт підтвердження (логічний нуль). Інтерфейс I^2C може використовувати два формати адреси:

– 7–ми бітну адресу;

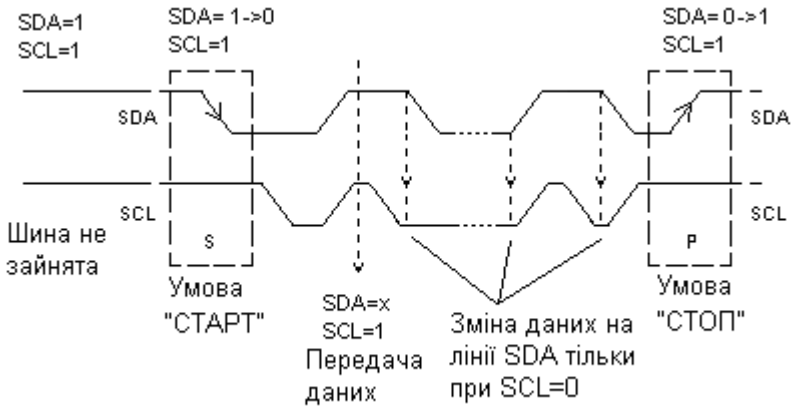
– 10–ти бітну адресу.

На рисунку 6.14 наведено формат 7–ми бітної адреси.

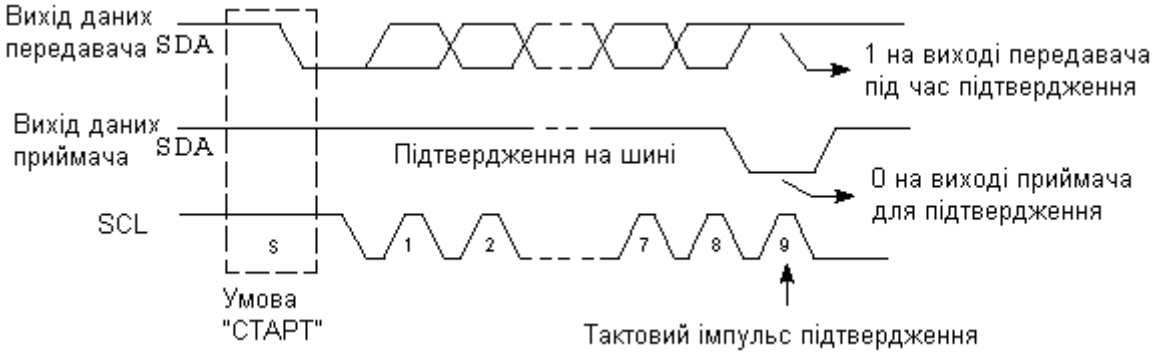
S	A6	A5	A4	A3	A2	A1	A0	R/$\overline{W}$	$\overline{ACK}$

Рисунок 6.14 – Формат 7–бітної адреси (S–Старт, A0…A6–адреса, $R/\overline{W}$ – біт «читання/запис», біт $\overline{ACK}$ – підтвердження)

На рисунку 6.15 наведено часові діаграми, що відображають стани на шині (рисунок 6.15, а) та пояснюють формування сигналу підтвердження (рисунок 6.15, б).



а



б

Рисунок 6.15 – Стани на шині: старт – s, стоп – p, передача даних (а); формування сигналу підтвердження (б)

Наведені діаграми відображають наступні коректні стани сигналів на шині під час обміну даними:

– шина не зайнята: на обох лініях одиниця (SDA = SCL = 1);

– початок обміну даними: зміна сигналу на лінії даних SDA з одиниці в нуль при одиничному значенні сигналу на лінії SCL, що визначає умову початку обміну (умова «СТАРТ» – S);

– припинення передачі: зміна сигналу на лінії даних з нуля в одиницю при одиничному значенні сигналу на лінії SCL, що визначає умову закінчення обміну (умова «СТОП» – Р);

– коректність даних: при одиничному значенні сигналу на лінії SCL стан лінії даних не повинен змінюватися, щоб не сформувати невірну умову СТАРТ або СТОП. Дані можна змінювати, якщо на лінії SCL присутній низький рівень сигналу (логічний нуль). На один біт інформації на лінії SDA міститься один тактовий імпульс на лінії SCL. Кожний цикл обміну даними починається умовою «СТАРТ» і закінчується умовою «СТОП». Кількість інформаційних бітів даних, які передаються між цими станами, необмежена. Дані передаються байтами. Приймач підтверджує одержання чергового байта, посилаючи біт підтвердження (логічний нуль) після прийому кожного байта;

– біт підтвердження: передається після прийому кожного байта даних або адреси. Активний передавач (ведений або ведучий) після передачі чергового байта формує на лінії SDA сигнал високого рівня. Ведучий пристрій (приймач або передавач) формує на лінії SCL тактовий імпульс, а приймач (ведучий або ведений) видає на лінію даних SDA сигнал підтвердження низького рівня.

Приймач, що генерує біт підтвердження, підключає лінію SDA до низького рівня і утримує її в цьому стані доти, поки тактовий імпульс лінії SCL не переключиться в стан низького рівня. Для припинення обміну приймач повинен залишити останній прийнятий байт без підтвердження, що автоматично викликає формування активним передавачем умови «СТОП».

Можливі чотири режими (типу) обміну даними для інтерфейсу I^2C (рисунок 6.15):

– ведучий передавач: на вихід SDA передавача виводяться дані, а на лінію SCL видаються синхроімпульси. Перший переданий байт містить адресу

веденого приймача (7 біт) і біт напрямку обміну даними $R/\overline{W} = 0$, що говорить про те, що буде проводитися запис (передача). Дані передаються послідовно по 8 біт. Після передачі чергового байта (адреса або дані), ведучий передавач очікує від веденого приймача біт підтвердження $\overline{ACK}$. Для задавання початку й кінця сеансу обміну даними ведучий передавач формує умови «СТАРТ» і «СТОП»;

– ведучий приймач: спочатку сеансу обміну ведучий приймач передає на лінію SDA адресу веденого передавача (7 біт) і біт напрямку обміну $R/\overline{W} = 1$, що говорить про те, що ведучий буде здійснювати прийом. Ведучий приймач формує імпульси синхронізації, що передаються лінією SCL. Після прийому адреси ведений передавач виставляє на лінію SDA сигнал підтвердження $\overline{ACK}$, а потім передає дані. Дані від веденого передавача передаються послідовно по 8 біт лінією SDA. Після прийому чергового байта ведучий приймач виставляє на лінію SDA сигнал підтвердження $\overline{ACK}$. Умови СТАРТ і СТОП формуються ведучим пристроєм для вказівки початку й кінця сеансу обміну послідовними даними;

– ведений приймач: ведучий передавач видає на лінію SDA адресу веденого приймача й біт напрямку $R/\overline{W} = 0$, що говорить про те, що буде виконуватися запис (передача). На лінію SCL ведучий передавач видає синхроімпульси. Після одержання від веденого приймача сигналу підтвердження $\overline{ACK}$ ведучий передавач послідовно передає на лінію SDA дані. Ведений приймач після прийому чергового байта даних передає сигнал підтвердження $\overline{ACK}$, що надходить до ведучого передавача лінією SDA. Умови СТАРТ і СТОП формуються ведучим передавачем;

– ведений передавач: перший байт на шині SDA приймається й обробляється веденим передавачем так само, як і в режимі веденого приймача. При цьому біт напрямку $R/\overline{W} = 1$, що говорить про те, що ведучий буде здійснювати прийом. Дані послідовно передаються лінією SDA від веденого

передавача, у той час, як синхроімпульси передаються лінією SCL від ведучого приймача. Після передачі кожного байта ведений передавач аналізує наявність на лінії SDA біта підтвердження $\overline{ACK}$, що передає ведучий приймач. Умови СТАРТ і СТОП формує ведучий приймач.

У підпорядкованому режимі апаратні засоби інтерфейсу I^2C здійснюють пошук своєї власної підпорядкованої адреси або адреси загального виклику. Якщо визначається одна із цих адрес, запитується переривання й виконуються відповідні дії. Якщо модуль I^2C хоче захопити шину й стати ведучим, то він чекає, поки шина звільниться (SDA = SCL = 1). Можливе функціонування в якості веденого при цьому не переривається.

Два й більше пристрої I^2C можуть спробувати стати ведучими й одночасно згенерувати умову «СТАРТ». У цьому випадку здійснюється арбітраж шини в моменти, коли шина SCL перебуває у високому стані. Якщо один ведучий передає на лінію даних низький рівень, а інший – високий, то останній відключається від лінії, тому що стан шини SDA (низький) не відповідає високому стану внутрішньої шини даних пристрою, що бажає стати ведучим. Якщо арбітраж шини втрачено у головному режимі, то відповідний пристрій I^2C перемикається в підпорядкований режим і може визначати свою власну підпорядковану адресу.

На рисунку 6.16 наведено часові діаграми, що відображають обмін даними шиною I^2C.

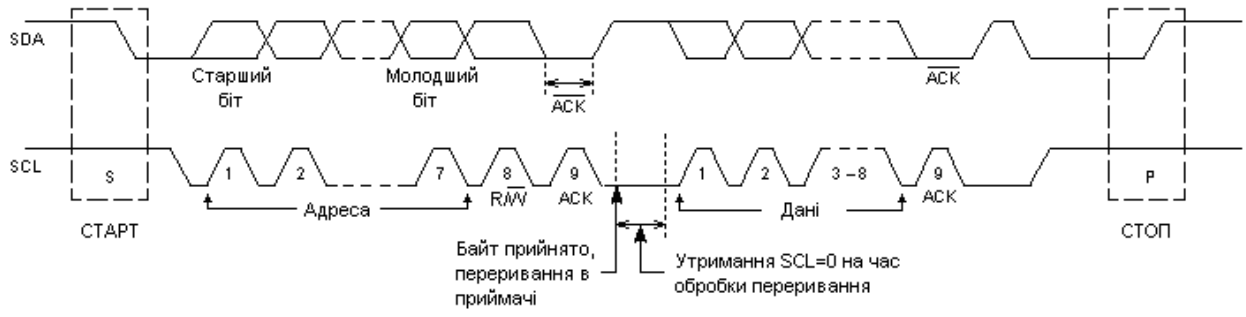


Рисунок 6.16 – Приклад обміну даними шиною I^2C

6.7.2.2 Інтерфейс I^2C у складі мікроконтролерів фірми Cygnal

6.7.2.2.1 Загальна характеристика

Багато мікроконтролерів фірми Cygnal оснащено послідовним інтерфейсом введення/виведення SMBus, який відповідає специфікації версії 1.1 (System Management Bus Specification, v1.1) [10]. Цей інтерфейс є двопровідною двонаправленою послідовною шиною, що сумісна з послідовним інтерфейсом I^2C. Зчитування та запис інформації проводиться байтами під автономним керуванням вбудованого контролера послідовної передачі даних. Дані можуть передаватися зі швидкістю до 1/178 від системної тактової частоти. При цьому швидкість передачі може бути вище, ніж передбачено специфікацією. Реальна швидкість передачі залежить від типів пристроїв, які підключено до шини (точніше від їх граничної швидкодії), і звичайно може бути набагато меншою.

Шина SMBus використовує механізм синхронізації, аналогічний інтерфейсу I^2C. Головний вузол передає тактові імпульси, а ведений або працює на запропонованій головним вузлом швидкості, або пригальмовує швидкість, переводячи лінію SCL в низький логічний рівень, знижуючи, таким чином, частоту тактових імпульсів.

На шині SMBus можливі кілька типів стану тайм–ауту.

Тайм–аут переведенням тактової лінії в нуль (SCL Low Timeout). Якщо лінія SCL переведена низькошвидкісним веденим вузлом в стан низького логічного рівня, ніяка подальша передача неможлива. Очевидно, що головний вузол у цій ситуації нічого не може зробити. Для виключення цієї ситуації передбачено, що кожен вузол визначає тривалість перебування лінії SCL в стані низького рівня, і якщо ця тривалість перевищує 25 ms, стан вважається тайм–аутом. Кожен вузол, який визначив стан тайм–ауту на шині, зобов'язаний скинути свій інтерфейс не пізніше 10 ms. Інтерфейс SMBus не має спеціального вузла моніторингу лінії SCL, однак будь–який з таймерів загального

призначення в режимі 16–бітного регістра з автозавантаженням може бути використаний для цього, а таймер 4 багатьох мікроконтролерів спеціально пристосований для таких цілей.

Тайм–аут тактової лінії в одиниці (SCL High Timeout). Цей тайм–аут можна вважати функціонально нормальним (тобто не помилковим), тому що він свідчить про те, що шина вільна. Цей тайм–аут визначається, якщо обидві лінії SCL і SDA, перебувають у стані логічної одиниці більш ніж 50 mks. Якщо біт FTE в регістрі SMB0CN встановлено, та час підтримки високого рівня на лінії SCL перевищує межу, яка визначається значенням регістра SMB0CR, то відбувається умова тайм–ауту.

Інтерфейс SMBus програмується за допомогою 5 регістрів спеціальних функцій: SMB0CN – регістр керування; SMB0CR – регістр завдання швидкості; SMB0ADR – регістр адреси; SMB0DAT – регістр даних і SMB0STA – регістр стану.

6.7.2.2.2 Опис роботи за функціональною схемою

Функціональну схему контролера SMBus0 приведено на рисунку 6.17.

Центральним вузлом контролера є регістр даних, що зсувається, SMB0DAT (SMBus0 Data Register). Він має функції паралельного запису та зчитування. Послідовний вхід даних цього регістра через фільтр підключено до лінії SDA, що назначається за допомогою матричного комутатора на один із виводів портів МК, дозволяючи здійснювати прийом даних. Послідовний вихід даних того ж регістра через керуючий мультиплексор, інвертор та електронний ключ також під’єднується до лінії SDA, що дозволяє реалізувати передачу даних. Другим вузлом пристрою є логіка керування SMBus. Цей вузол передусім відповідає за формування та видачу на вивід SCL (ведучий режим) та прийом (ведений режим) імпульсів синхронізації.

Ця функція підтримується підключенням входу вузла через фільтр та виходу через інвертор та електронний ключ до лінії SCL, яка назначається за допомогою матричного комутатора на один з виводів портів МК.

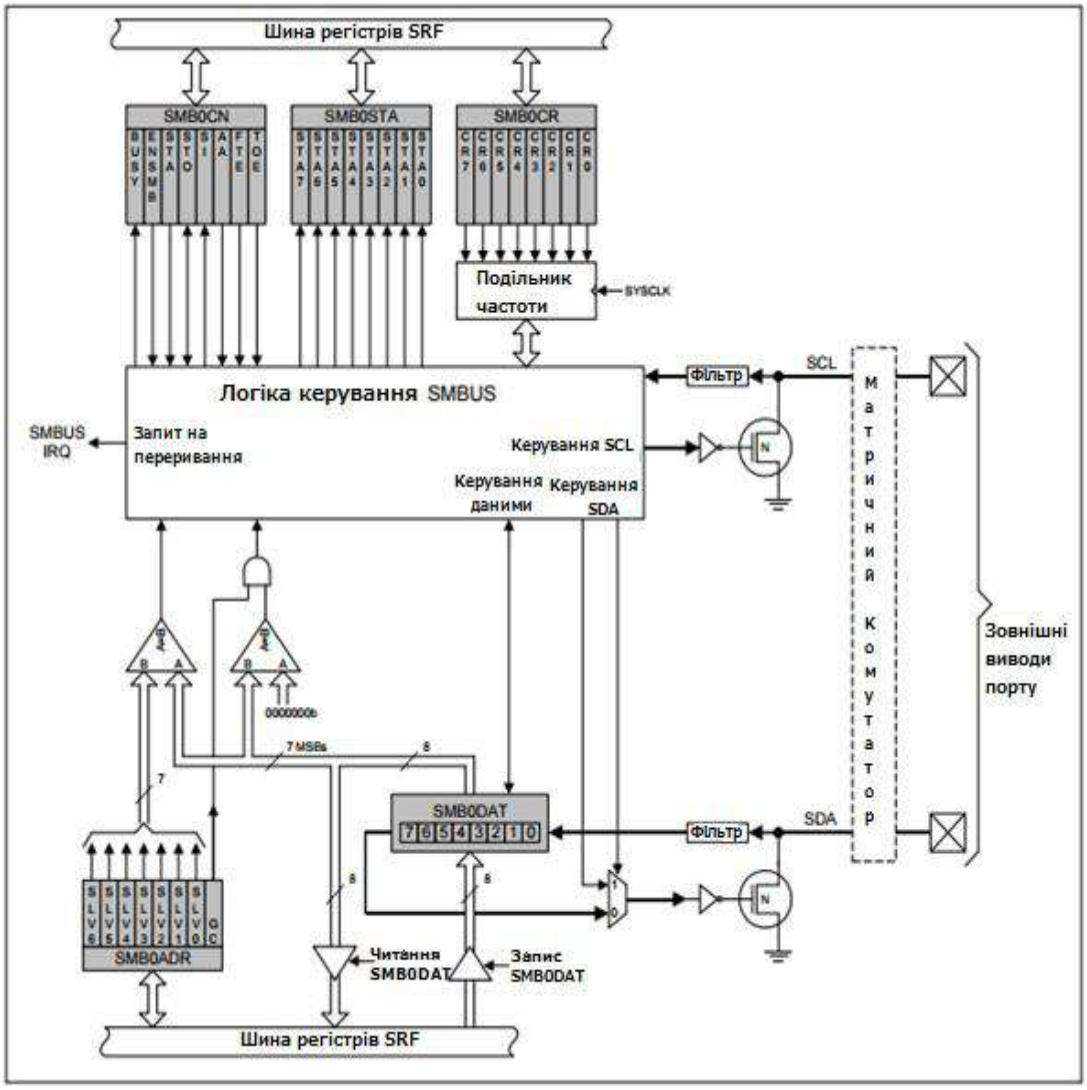


Рисунок 6.17 – Функціональна схема контролера послідовного інтерфейсу SMBus0

Цей самий вузол підтримує відповідні протоколи в різноманітних режимах та ситуаціях, що потребують втручання процесора, та формує запит переривання від SMBus: IRQ.

Окрім цього, схема модуля включає до себе компаратор адреси веденого та подільник частоти синхронізації МК–ра SYSCLK для керування швидкістю

передачі / прийому. Ця швидкість задається значенням, що міститься в регістрі частоти синхронізації SMBOCR (SMBus0 Clock Rate Register):

$$CR = SYSCLK/ 2 (256 – SMB0CR).$$

Регістр адреси веденого SMB0ADR (SMBus0 Address Register) дозволяє задати 7–бітну адресу веденого SLV6…SLV0 (SMB0ADR.7…SMB0ADR.1) (SMBus0 Slave Address) та дати дозвіл на прийом загальних викликів з адресою 0000000b через встановлення в 1 біта дозволу GC (SMB0ADR.O) (General Call Address Enable).

Керування контролером SMBus виконується за допомогою бітів керування та прапорців, які розміщено в регістрі керування SMB0CN (Control Register). Кожен раз, коли встановлюється прапорець переривання від SMBus – SI (SMB0CN.3) (Serial Interrupt Flag), регістр стану контролера SMB0STA отримує код стану контролера. Цей код ідентифікує один з 28 станів контролера та є дійсним, доки прапорець SI знаходиться у стані логічної одиниці. Тому програмне скидання цього прапорця має виконуватися після переходу до підпрограми обробки відповідного стану. Для полегшення цього переходу код стану зсунуто у регістрі на три розряди вліво. Отже, три молодші біти завжди дорівнюють нулю, тому всі можливі коди стану кратні восьми. Це дозволяє використовувати код стану безпосередньо як індекс переходу до потрібної 8–байтової секції програмного коду обробки переривання. Такий перехід можна реалізувати за допомогою команди JMP @A+DPTR. Коди станів контролера SMBus у різних режимах наведено у таблиці 6.5.

Типову конфігурацію шини SMBus зображено на рисунку 6.13. До шини можна підключати пристрої з напругою живлення 3…5 В. Двонаправлені лінії SDA та SCL мають бути з’єднані з джерелом живлення через резистори для реалізації монтажного АБО для нульових вихідних рівнів. Це дозволяє підключати до ліній пристрої, що мають вихід з відкритим стоком або колектором.

Таблиця 6.5 – Коди станів SMBus (регістр SMB0STA)

Режим	**Код стану**	**Стан SMBus**	**Типові дії**
Ведучий: перед./прийом	08H	Стан START передано	Завантажити до регістра SMB0DAT адресу веденого + R/$\overline{W}$. Скинути STA
	10H	Стан START передано повторно	Завантажити до регістра SMB0DAT адресу веденого + R/$\overline{W}$. Скинути STA
Ведучий: передача	18H	Адреса веденого + $\overline{W}$ передано. $\overline{ACK}$ прийнято	Завантажити до регістра SMB0DAT байт даних для передачі
	20H	Адреса веденого + $\overline{W}$ передано. NACK прийнято	Повторити запит підтвердження. Встановити біти STO та STA
	28H	Байт даних передано. $\overline{ACK}$ прийнято	Завантажити до регістра SMB0DAT наступний байт, або встановити біт STO, або скинути біт STO, потім встановити біт STA для повторення стану START
	30H	Байт даних передано. NACK прийнято	Передати повторно, або встановити біт STO
	38H	Втрата керування (арбітражу)	Зберегти поточні дані
Ведучий: прийом	40H	Адреса веденого + R передано. $\overline{ACK}$ прийнято	Якщо приймається лише один байт, то очистити прапорець AA (надіслати NACK після прийому байта). Очікувати на прийом даних
	48 H	Адреса ведучого + R передано. NACK прийнято	Повторити запит на підтвердження. Встановити біти STO та STA
	50H	Байт даних прийнято. $\overline{ACK}$ передано	Прочитати регістр SMB0DAT. Чекати на наступний байт. Якщо наступний байт останній, очистити прапорець AA
	58H	Байт даних прийнято. NACK передано	Встановити біт STO
Ведений : прийом	60H	Прийнято власну адресу веденого + $\overline{W}$. $\overline{ACK}$ передано	Очікувати на дані
	68H	Втрата керування під час передачі адреси веденого + R/$\overline{W}$ від ведучого. Прийнято власну адресу веденого + $\overline{W}$. $\overline{ACK}$ передано	Зберегти поточні дані для повторення спроби передачі після звільнення шини. Очікувати на дані
	70H	Прийнято адресу загального виклику + $\overline{W}$. $\overline{ACK}$ передано	Очікувати на дані
	78H	Втрата керування під час передачі адреси веденого + R/$\overline{W}$ від ведучого. Прийнято адресу загального виклику. $\overline{ACK}$передано	Зберегти поточні дані для повторення спроби передачі після звільнення шини. Очікувати на дані

Продовження таблиці 6.5

Режим	Код стану	Стан SMBus	Типові дії
	80H	Прийнято байт даних. $\overline{ACK}$ передано	Читати регістр SMB0DAT. Чекати на наступний байт або на умову STOP
	88H	Прийнято байт даних. NACK передано	Встановити біт STO для скидання SMBus
	90H	Прийнято байт даних після загального виклику. $\overline{ACK}$ передано	Читати регістр SMB0DAT. Чекати на наступний байт або на умову STOP
	98H	Прийнято байт даних після загального виклику. NACK передано	Встановити біт STO для скидання SMBus
	A0H	STOP або повторний START прийнято	Немає необхідності в діях
Ведений: передавач	A8H	Прийнято власну адресу веденого + R. $\overline{ACK}$ передано	Завантажити до регістра SMB0DAT байт даних для передачі
	B0H	Втрата керування під час передачі адреси веденого + R/$\overline{W}$ від ведучого. Прийнято власну адресу веденого + R. $\overline{ACK}$ передано	Зберегти поточні дані для повторення спроби передачі після звільнення шини. Завантажити до регістра SMB0DAT байт даних для передачі
	B8H	Байт даних передано. $\overline{ACK}$ прийнято	Завантажити до регістра SMB0DAT байт даних для передачі
	C0H	Байт даних передано. NACK прийнято	Чекати на STOP
	C8H	Останній байт даних передано (прапорець AA = 0). $\overline{ACK}$ прийнято	Встановити біт STO для скидання SMBus
Ведении ПД/ПРМ	D0H	Переповнення таймера контролю високого рівня сигналу SCL, час на SMB0CR минув	Встановити біт STO для скидання SMBus
Усі режими	00H	Помилка шини (не коректний START або STOP)	Встановити біт STO для скидання SMBus
	F8H	Режим очікування	Прапорець SI не встановлено

Номінал резисторів та кількість пристроїв, що можуть підключатися до ліній, визначаються лише вимогами до довжини фронту імпульсу, що зростає та спадає. Фронт імпульсу, що зростає, не має перевищувати 300 наносекунд, імпульсу, що спадає – 1000 наносекунд.

Будь–який з пристроїв, який під’єднаний до шини SMBus, може конкурувати за роль ведучого, пославши до шини умову «START». Спеціальна арбітражна схема контролера забезпечує процес вибору ведучого серед декількох контролерів, що надіслали умову «START». Таким чином, у кожному циклі обміну інформацією на шині наявний лише один ведучий пристрій. Саме цьому пристрою належить право вибрати ведений пристрій, що адресується, та напрям передачі даних: від ведучого до веденого – ЗАПИС (WRITE), або від веденого до ведучого – ЧИТАННЯ (READ).

Часові діаграми протоколу передачі даних показано на рисунку 6.16. Типовий цикл передачі (транзакція) містить відсилання умови «START», адресного байта (7–бітний код адреси веденого SLA6...SLA0 та біт напрямку R/$\overline{W}$), передача/прийом одного чи більше байтів даних та передачу умови «STOP». Кожен байт, який прийнятий ведучим або веденим пристроєм, має бути затверджений відправленням біта $\overline{ACK}$ (acknowledged – підтвердження) у вигляді сигналу низького рівня на лінії SDA під час імпульсу додатної полярності на лінії SCL. Якщо пристрій, що приймає, не підтверджує прийом байта (тобто рівень сигналу на лінії SDA залишається високим), пристрій, що передає, сприймає його як сигнал NACK (not acknowledged – не підтверджено). Одиничний стан біта напрямку R/$\overline{W}$ відповідає прийому, нульовий стан – передачі.

Усі транзакції ініціюються ведучим пристроєм, він посилає умову «START», потім адресу веденого з бітом напрямку. Якщо напрям передачі – ЗАПИС, то ведучий посилає черговий байт, кожен раз дочекавшись підтвердження від веденого. Якщо напрямок передачі – ЧИТАННЯ, то ведучий очікує черговий байт від веденого, підтверджуючи прийом чергового байта бітом підтвердження. Після завершенні транзакцій ведучий генерує умову «STOP» і звільняє шину.

Арбітраж шини SMBus виконується наступним чином. Ведучий пристрій має право призначити передачу, тільки якщо шина вільна. Шина вважається вільною тільки після виконання умови «STOP», а саме якщо лінії SCL та SDA залишаються у стані високого потенціалу протягом певного часу. Для контролерів, що ми розглядаємо, цей час становить 50 мкс. У разі, коли два або більше ведучих пристрої намагаються почати передачу даних одночасно, використовується схема арбітражу, яка змусить за певними умовами будь–який ведучий пристрій звільнити шину. Ведучі пристрої продовжують передавати до тих пір, поки один з них не спробує передати на лінію SDA сигнал високого рівня, в той час як інші ведучі видають на цю лінію сигнал низького рівня. Ведучий пристрій, що намагається передати на лінію SDA сигнал високого рівня, визначить, що замість сигналу високого рівня на лінії SDA присутній сигнал низького рівня, і звільнить шину. Пристрій, що виграв арбітраж продовжує передавати свої дані без будь–якої перерви. Пристрій, що втратив арбітраж, стає веденим і приймає залишок переданих даних. Дана схема арбітражу не є руйнуючою: один з пристроїв завжди виграє і ніякі дані не втрачаються.

Шина SMBus підтримує механізм синхронізації, що дозволяє підключати до однієї шини пристрої з різною швидкодією. Для цього використовується розширення тривалості низького рівня сигналу SCL. Ведений пристрій може просто утримувати лінію SCL в стані низького рівня, не даючи можливості ведучому передати черговий синхроімпульс, тим самим знижуючи частоту синхронізації. Якщо ведений пристрій продовжує утримувати лінію SCL в стані тривалого часу, то подальший зв'язок стає неможливим. Для вирішення цієї проблеми будь–який пристрій, який підключений до шини, має ідентифікувати, як помилку часу, перевищення тривалості низького рівня імпульсу синхронізації на 25–мс. Будь–який з пристроїв, що виявив помилку часу, повинен скинути свої модулі не пізніше 10 мс після її виявлення.

Кожний контролер SMBus може бути налаштований як ведучий або ведений пристрій. Отже, в кожний момент часу контролер може працювати в одному з чотирьох режимів: режим передачі ведучим пристроєм, режим прийому ведучим пристроєм, режим передачі веденим пристроєм, режим прийому веденим пристроєм. Послідовні формати передачі/прийому даних шиною SMBus в цих режимах наведено на рисунку 6.18. На рисунку вказано моменти встановлення прапорця запиту переривання SI.

Ці переривання можуть оброблятися МК–м, як апаратні векторні переривання, або програмно в режимі «поллінгу» (опитуванням стану прапорця SI). У будь–якому випадку ідентифікація стану контролера виконується за допомогою коду стану в регістрі SMB0STA (таблиця 6.5).

У режимі передачі ведучим пристроєм (рисунок 6.18, а) контролер передає умову «START», потім байт, що містить адресу веденого (або адресу загального виклику 0000000b), і біт напрямку, який дорівнює 0 та відповідає операції ЗАПИС (W – WRITE). Отримавши підтвердження прийому адреси від веденого $\bar{A}$ ($\overline{ACK}$), ведучий посилає байти даних, щоразу чекаючи підтвердження $\bar{A}$. Для вказівки кінця транзакції ведучий посилає стан «STOP» P.

У режимі прийому ведучим пристроєм (рисунок 6.18, б) контролер передає умову «START», потім байт, що містить адресу веденого, і біт напрямку, що дорівнює 1 та відповідає операції ЧИТАННЯ (R – READ). Отримавши підтвердження прийому адреси від веденого $\bar{A}$ ($\overline{ACK}$), ведучий чекає і приймає байти даних, щоразу відповідаючи посилкою підтвердження $\bar{A}$. Після отримання останнього байта ведучий відповідає сигналом не підтвердження N (NACK), що є вказівкою веденому пристрою припинити передачу. Транзакція завершується посилкою ведучим умови «STOP» P.

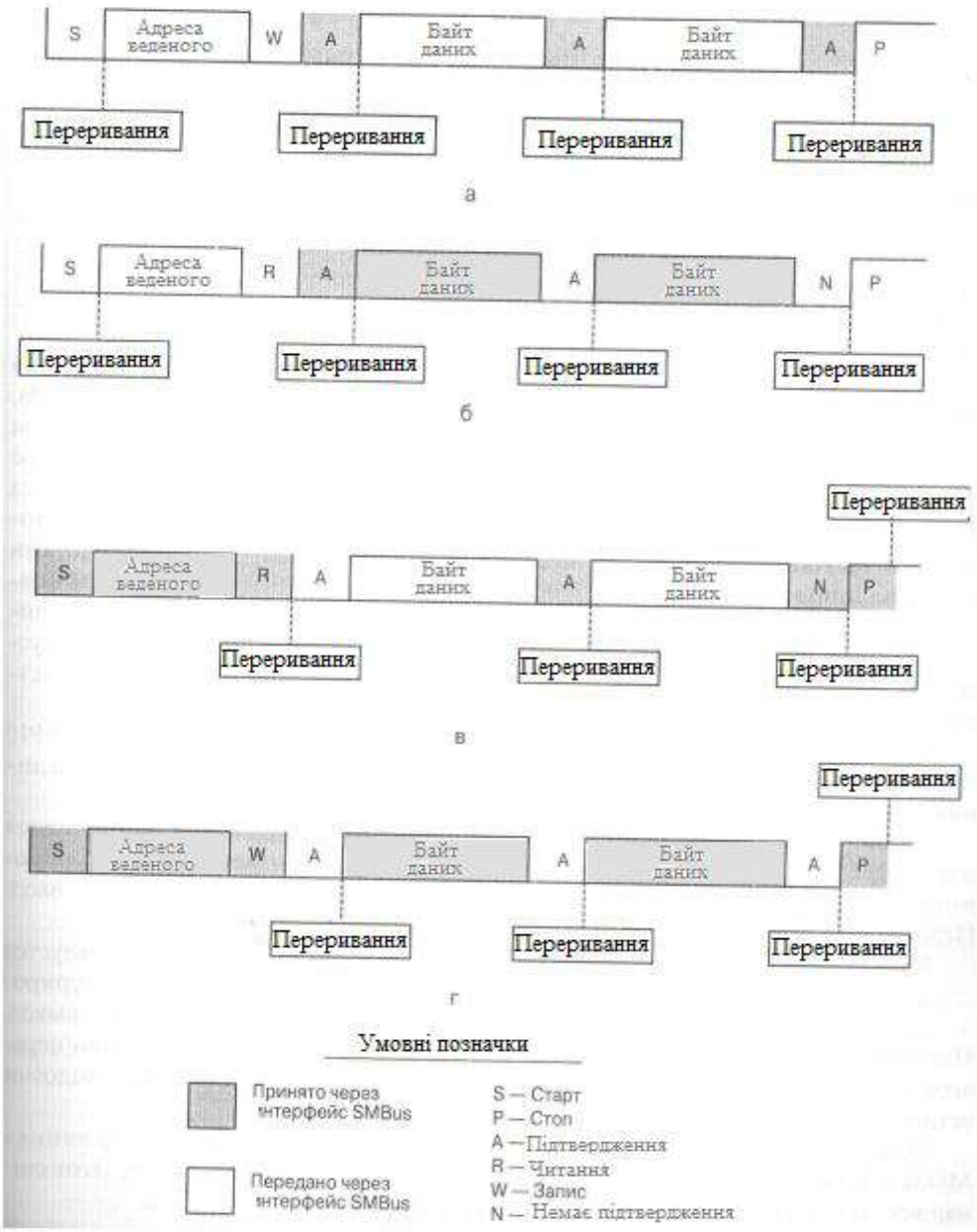


Рисунок 6.18– Послідовні формати передачі/прийому даних SMBus:

а – формат передачі ведучим пристроєм;

б – формат прийому ведучим пристроєм;

в – формат передачі веденим пристроєм;

г – формат прийому веденим пристроєм.

У режимі передачі веденим пристроєм (рисунок 6.18, в) контролер приймає від ведучого умову «START», потім байт, що містить адресу веденого, і біт напрямку, який дорівнює 1 та відповідає операції ЧИТАННЯ (R – READ).

Якщо адреса веденого збігається з власною адресою в регістрі SMB0ADR, контролер посилає ведучому підтвердження $\bar{A}$ ($\overline{ACK}$), а потім байти даних, щоразу чекаючи від керуючого підтвердження $\bar{A}$. Для вказівки кінця транзакції керуючий посилає стан не підтвердження N (NACK) та умову «STOP» P.

У режимі прийому веденим пристроєм (рисунок 6.18, г) контролер приймає від ведучого умову «START», потім байт, що містить адресу веденого (або адресу загального виклику 0000000b), і біт напрямку, шо дорівнює 0 та відповідає операції ЗАПИС (W – WRITE). Якщо адреса веденого збігається з власною адресою в регістрі SMB0ADR або з адресою загального виклику (при встановленому біті дозволу загального виклику GC в регістрі SMB0ADR), контролер посилає ведучому підтвердження $\bar{A}$ ($\overline{ACK}$). Далі він приймає байти даних, щоразу посилаючи підтвердження $\bar{A}$. Транзакція закінчується після отримання від ведучого умови «STOP» P.

Контроль тривалості фази низького рівня імпульсу синхронізації SCL на шині здійснюється за допомогою таймера 4, якщо встановлено біт TOE (SMBOCN.0) (SMBus Timeout Enable Bit) і роботу таймера 4 дозволено. У цьому випадку фаза високого рівня синхроімпульсу SCL призводить до перезавантаження таймера, а фаза низького рівня дозволяє відлік часу. Якщо таймер налаштовано на тривалість відліку до переповнення 25 мс, то встановлення прапорця запиту переривання від таймера 3 свідчитиме про помилку часу на шині. У цьому випадку підпрограма обробки переривання від таймера 3 повинна забезпечувати скидання модуля SMBus.

6.7.2.2.3 Програмування модуля SMBus

6.7.2.2.3.1 SMB0CN – регістр керування модулем

Назва регістра:	SMB0CN – SMBus Control Register		
SFR–адреса:	0xA7	Значення після скидання:	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BUSY	ENSMB	STA	STO	SI	AA	FTE	TOE
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Біт 7: BUSY – Busy Status Flag – прапорець зайнятості шини SMBus (логічна1).

Біт 6: ENSMB – SMBus Enable – біт дозволу (включення) модуля SMBus (логічна1).

Біт 5: STA – SMBus Start Flag – прапорець початку передачі (запуску модуля SMBus), логічна1). Якщо SMBus–інтерфейс працює в режимі ведучого, формування умови «START» починається, якщо шина вільна. Якщо шину зайнято, формування умови «START» відкладається до умови «STOP». Прапорець STO повинен бути очищений до встановлення STA.

Біт 4: STO – SMBus Stop Flag – прапорець завершення передачі. Встановлення прапорця STO призводить до передачі умови «STOP». Якщо умову «STOP» прийнято, прапорець STO апаратно скидається.

Біт 3: SI – SMBus Serial Interrupt Flag – прапорець переривання від послідовного інтерфейсу SMBus. Цей прапорець встановлюється апаратно, якщо має місце одна з 27 ситуацій, перерахованих в таблиці 6.5 (крім коду 0xF8). Якщо дозволено переривання, встановлення цього прапорця

викличе відповідну підпрограму обробки цього переривання. Прапорець скидається тільки програмно.

Біт 2: AA – SMBus Assert Acknowledge Flag – прапорець призначення підтвердження. Цей біт визначає тип біта підтвердження, що передається під час такту підтвердження на лінії SCL:

0 – Стан "Not Acknowledge" – під час тактового сигналу «підтвердження» SDA = 1;

1 – Стан "Acknowledge" – під час тактового сигналу «підтвердження»

SDA = 0.

Біт1: FTE: Біт дозволу таймера звільнення шини SMBus.

0: не використовується тайм–аут високого рівня на лінії SCL.

1: якщо час утримання високого рівня на лінії SCL перевищує межу, яку визначає значення регістра SMB0CR, то відбувається умова тайм–ауту.

Біт 0: TOE: Біт дозволу тайм–ауту SMBus.

0: не використовується тайм–аут низького рівня на лінії SCL.

1: якщо час утримання низького рівня на лінії SCL перевищує межу, яка визначається таймером 4. Якщо його включено, то відбувається умова тайм–ауту.

Регістр керування SMB0CN використовується для керування модулем SMBus і його налаштування. Всі біти цього регістра можуть бути прочитані і записані програмно. Два з керуючих бітів також встановлюються модулем SMBus0 апаратно. Прапорець переривання від послідовного порту (SI, SMB0CN.3) встановлюється в 1 апаратно при виникненні переривання від модуля SMBus. Він може бути змінений також програмно. Прапорець STOP

(STO, SMB0CN.4) скидається в 0 апаратно при виявленні на шині умови STOP. Встановлення в 1 прапорця ENSMB включає модуль SMBus. Скидання в 0 прапорця ENSMB відключає модульSMBus і видаляє його з шини. Скидання прапорця ENSMB і потім повторне його встановлення в 1 призведуть до скидання модуля SMBus. Однак, прапорець ENSMB не слід використовувати для тимчасового видалення пристрою з шини, тому що інформацію про стан шини буде втрачено. Замість цього для тимчасового видалення пристрою з шини слід використовувати прапорець підтвердження AA (опис прапорця AA наведено нижче).

Встановлення в 1 прапорця запуску (STA, SMB0CN.5) переведе модуль SMBus в режим ведучого. Якщо шина вільна, модуль SMBus згенерує умову «START». Якщо шина зайнята, то модуль SMBus буде очікувати умову «STOP», що свідчить про звільнення шини, і потім згенерує умову «START» через 5мкс після затримки, що визначається значенням регістра SMB0CR. (Згідно з протоколом SMBus, модуль SMBus також буде вважати шину вільною, якщо шина простоює протягом 50мкс і умову «STOP» не виявлено). Якщо біт STA встановлюється в 1 в той час, коли модуль SMBus знаходиться в режимі ведучого і вже передані один або декілька байт, то модуль згенерує подію «повторний START». Щоб гарантувати правильне функціонування, прапорець STO слід скинути 0 до встановлення в 1 біта STA.

Якщо прапорець закінчення передачі (STO, SMB0CN.4) встановлюється в 1 в той час, коли модуль SMBus знаходиться в режимі ведучого, то модуль SMBus згенерує на шині умову «STOP». В режимі веденого прапорець STO можна використовувати для відновлення зі стану збою. У цьому випадку умова «STOP» не генерується, але модуль SMBus веде себе так, як ніби умову «STOP» вже отримано, і переходить в режим «не адресованого» веденого приймача. Слід мати на увазі, що цей умовний біт STOP не викличе звільнення шини. Шина буде залишатися зайнятої до тих пір, поки на ній не з'явиться умова

«STOP», або поки не станеться умова тайм–ауту звільнення шини. При виявленні на шині умови «STOP» модуль SMBus автоматично скидає в 0 прапорець STO.

Прапорець переривань від послідовного порту (SI, SMB0CN.3) встановлюється апаратно в 1, якщо інтерфейс SMBus переходить до одного з 27 можливих станів. Якщо переривання для модуля SMBus дозволено, то при встановленні в 1 прапорця SI генерується запит на переривання. Прапорець SI повинен бути скинутий програмно.

Важлива примітка: Якщо прапорець SI встановлений в 1, в той час, коли на лінії SCL утримується низький рівень сигналу, то період тактового імпульсу буде «розтягуватися» (на ділянці з низьким рівнем сигналу) і передача послідовних даних по шині припиниться до тих пір, поки не буде скинутий в 0 прапорець SI. На тривалість високого рівня сигналу на лінії SCL встановлення прапорця SI не впливає.

Прапорець призначення підтвердження AA (SMB0CN.2) використовується для завдання рівня сигналу на лінії SDA під час тактового імпульсу підтвердження на лінії SCL. Встановлення в 1 прапорця AA призведе до передачі біта підтвердження ACK (низький рівень сигналу на лінії SDA) під час тактового імпульсу підтвердження на лінії SCL, якщо пристрій розпізнав свою адресу. Скидання в 0 прапорця AA призведе до передачі біта «немає підтвердження» NACK (високий рівень сигналу на лінії SDA) під час тактового імпульсу підтвердження на лінії SCL. Після передачі байта в режимі веденого ведений пристрій можна тимчасово видалити з шини шляхом скидання в 0 прапорця AA. Власна адреса веденого і адреса загального виклику будуть ігноруватися. Для відновлення роботи на шині необхідно встановити в 1 прапорець AA, щоб дозволити веденому розпізнавати свою адресу.

Встановлення в 1 біта дозволу таймера звільнення шини SMBus (FTE, SMB0CN.1) включить таймер відліку часу очікування звільнення шини, який

визначається значенням регістра SMB0CR. Якщо на лінії SCL утримується високий рівень сигналу, то таймер відраховує тайм–аут, який визначається регістром SMB0CR. Переповнення таймера означає закінчення тайм–ауту звільнення шини: якщо модуль SMBus0 чекає моменту для генерації біта START, то він згенерує його після закінчення даного тайм–ауту. Період звільнення шини повинен бути не більше 50мкс.

Коли біт (TOE, SMB0CN.0) встановлено в 1, таймер 4 використовується для відліку часу очікування низького рівня сигналу на лінії SCL. Якщо таймер 4 включено, то він буде перезавантажуватися, коли на лінії SCL присутній сигнал високого рівня, і буде відраховувати тайм–аут, коли на лінії SCL присутній сигнал низького рівня. Якщо Таймер 4 включено і налаштовано на переповнення через 25мс і біт TE встановлено в 1, то переповнення таймера 4 означає закінчення часу очікування низького рівня сигналу на лінії SCL. В цьому випадку для скидання модуля SMBus0 можна використовувати процедуру обробки переривання від таймера 4.

6.7.2.2.3.2 SMB0CR – регістр встановлення тактової частоти модуля SMBus

Назва регістра:	SMB0CR – SMBus Clock Rate Register		
SFR–адреса:	0xCF	Значення після скидання:	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Встановленням бітів цього регістра програмується тактова частота імпульсів SCL, які видаються на лінію SCL в режимі ведучого. Значення з регістра завантажується в спеціальний 8–бітний таймер. Таймер лічить у прямому напрямі і при переповненні (переході із стану 0xFF в стан 0x00)

відбувається спрацьовування SCL–логіки (стан сигналу на лінії SCL змінюється на протилежний).

Значення SMB0CR обмежується наступним рівнянням: SMB0CR<((288–0.85*SYSCLK)/1.125), де SMB0CR–8–розрядне значення (без знака) регістра SMB0CR; SYSCLK–системна тактова частота в [Гц]. Тривалість утримання низького і високого рівнів тактового сигналу на лінії SCL визначається наступними рівняннями:

T_{LOW}=(256–SMB0CR)/SYSCLK, T_{HIGH}=(258–SMB0CR)/SYSCLK+625нс. Значення регістра SMB0CR визначає також тайм–аут звільнення шини у відповідності з наступним рівнянням:

T_{BFT} = 10 * [(256 – SMB0CR) + 1] / SYSCLK.

6.7.2.2.3.3 SMB0DAT – регістр даних модуля SMBus

Назва регістра:	SMB0DAT – SMBus Data Register		
SFR–адреса:	0xC2	Значення після скидання:	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Регістр призначено для читання прийнятих даних (байтів) або для запису даних, які передаються. Операції читання і запису можливі тільки тоді, коли встановлено прапорець SI (SMB0CN.3). В інший час регістр використовується інтерфейсом для зсуву і його дані не достовірні.

Дані завжди зсуваються старшими розрядами вперед. Після прийому байта перший біт прийнятих даних займає старший розряд регістра SMB0DAT. Коли дані висуваються з регістра, вони одночасно з`являються на шині. Тому регістр SMB0DAT завжди зберігає останній байт даних, який присутній на шині. Таким чином, у випадку втрати арбітражу перехід від ведучого

передавача до веденого приймача відбувається з коректними даними у регістрі SMB0DAT.

6.7.2.2.3.4 SMB0ADR – регістр адреси модуля SMBus

Назва регістра:	SMB0ADR – SMBus Address Register		
SFR–адреса:	0xC3	Значення після скидання:	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
SLV6	SLV5	SLV4	SLV3	SLV2	SLV1	SLV0	GC
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Біти 7…1: SLV6…0 – SMBus Slave Address – 7–бітова адреса веденого. Ці біти програмуються, тобто адреса розпізнається з прийнятої передачі, тільки якщо інтерфейс працює в режимі веденого. У режимі ведучого вміст регістра ігнорується.

Біт 0: GC – General Call Address Enable – біт дозволу загального виклику (передачі усім). Якщо біт 0 встановлено в одиницю, дозволяється розпізнавання адреси 0x00 – загальної передачі (загального виклику).

Регістр адреси SMB0ADR зберігає адресу веденого пристрою для інтерфейсу SMBus. У веденому режимі сім старших значущих бітів утворюють 7–бітну адресу веденого. Молодший значущий біт, біт 0, використовується для дозволу розпізнавання адреси загального виклику (0x00). Якщо біт 0 встановлено у 1, адреса загального виклику буде розпізнаватися. У іншому випадку, адреса загального виклику буде ігноруватися. Вміст цього регістра також ігнорується, якщо модуль SMBus працює у ведучому режимі.

6.7.2.2.3.5 SMB0STA – регістр статусу SMBus

Назва регістра:	SMB0ADR – SMBus Status Register		
SFR–адреса:	0xC1	Значення після скидання:	11111000b (0xF8)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
STA7	STA6	STA5	STA4	STA3	STA2	STA1	STA0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Біти 7…3: STA7…3 – SMBus Status Code – код статусу інтерфейсу SMBus. Біти містять 28 статусних кодів інтерфейсу. Коди дійсні, коли прапорець SI встановлено у 1. Будь–яка спроба запису в цей регістр може призвести до непередбачуваних наслідків!

Біти 2…0: STA2…0 –завжди читаються як 000b, якщо SI встановлено в 1.

Регістр стану SMB0STA містить 8–бітний код стану, якій показує поточний стан модуля SMBus. Існують 28 можливих станів модуля SMBus, кожному з яких відповідає унікальний код стану. П`ять старших значущих бітів коду стану можуть мати різні значення, а три молодших значущих біти для коректних кодів стану завжди дорівнюють нулю, коли SI = 1. Тому всі можливі коди стану кратні восьми. Це дозволяє застосовувати у програмі код стану у якості індексу, який використовується для переходу на відповідну восьми байтну процедуру обслуговування переривання, або для переходу на більш складну процедуру обслуговування переривання.

Для потреб програми користувача вміст регістра SMB0STA визначено тільки тоді, коли прапорець SI встановлено у 1. Програма ніколи не повинна записувати дані у регістр SMB0STA. Це приведе до не визначеного результату. У таблиці 6.5 наведено всі 28 станів модуля SMBus разом з відповідними кодами.

6.7.2.2.4 Взаємодія прикладної програми з модулем SMBus

Взаємодія прикладної програми з модулем SMBus базується на використанні переривання від модуля, яке виникає після кожної події, що відбулася на шині (прийом байта, формування станів СТАРТ/СТОП і т. ін.). Відповідно, під час передачі даних шиною програма може виконувати інші задачі. Якщо переривання від модуля SMBus з якихось причин використовувати не можна, його можна заборонити. У цьому випадку програма повинна буде постійно слідкувати за станом прапорця SI для реагування на події, які відбуваються на шині.

Встановлення прапорця SI регістра SMB0CN означає, що модуль SMBus закінчив виконання чергової операції і очікує реакції програми.

У старших п'яти розрядах регістра SMB0STA при цьому формується відповідне значення, яке характеризує поточний стан шини SMBus. Відповідно, програма повинна проаналізувати це значення і задати подальше поводження модуля SMBus, маніпулюючи вмістом регістрів SMB0CN і SMB0DAT.

На рисунку 6.19 показано взаємодію прикладної програми з модулем SMBus на найпростішому прикладі передачі одного байта даних від ведучого до веденого.

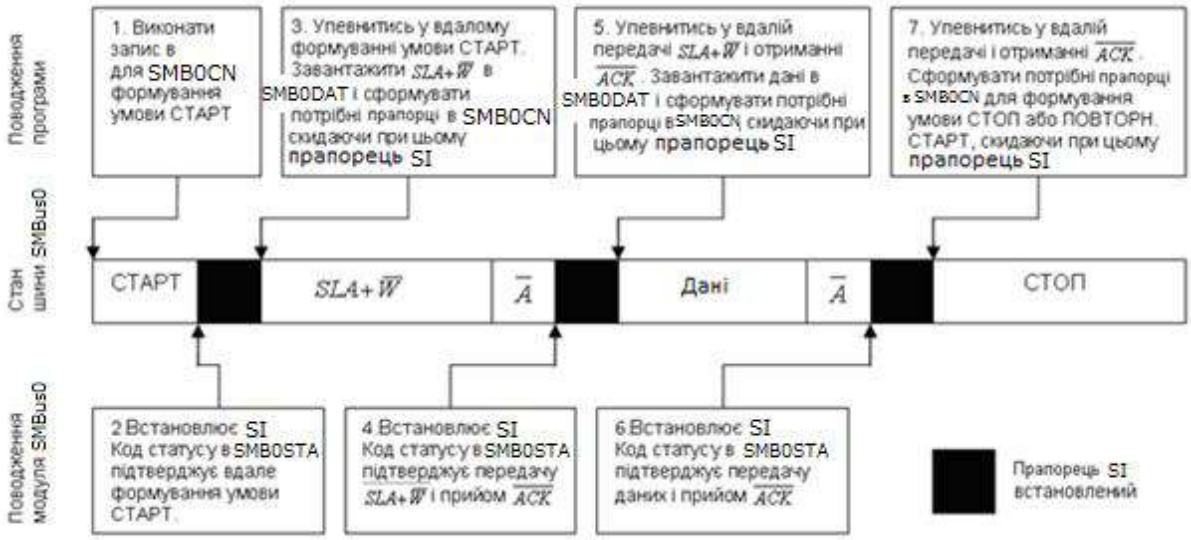


Рисунок 6.19 – Приклад взаємодії програми з модулем SMBus0

Передача даних відбувається у наступній послідовності:

- першою операцією при передачі даних шиною SMBus є формування стану СТАРТ. Для цього варто записати певне значення в регістр SMB0CN, відповідно до якого модуль SMBus сформує на шині стан СТАРТ. При цьому для скидання прапорця SI розряд, який відповідає прапорцю SI, повинен бути встановлений в «1». Формування стану СТАРТ почнеться відразу ж після скидання прапорця SI;
- після формування стану СТАРТ встановлюється прапорець SI. Число, яке перебуває у регістрі стану SMB0STA, відображає результат виконання цієї операції;
- необхідно впевнитися в успішному формуванні стану СТАРТ, перевіривши вміст регістра SMB0STA. Якщо код статусу відповідає очікуваному, варто завантажити в регістр SMB0DAT вміст пакета $SLA + \overline{W}$ і сформувати в регістрі SMB0CN команду на передачу

пакета (не забуваючи скинути при цьому прапорець SI). Передача адресного пакета почнеться відразу ж після скидання прапорця;

- по закінченні передачі адресного пакета встановлюється прапорець SI. Код статусу, що перебуває в регістрі SMB0STA, говорить про успішну передачу пакета, а також про одержання підтвердження від веденого пристрою;
- необхідно впевнитися в успішній передачі пакета та одержанні підтвердження, перевіривши вміст регістра SMB0STA. Якщо код статусу відповідає очікуваному, слід завантажити дані в регістр SMB0DAT, а потім сформувати в регістрі SMB0CN команду на передачу пакета (не забуваючи скинути при цьому прапорець SI). Передача пакета даних почнеться відразу ж після скидання прапорця;
- по закінченні передачі пакета даних встановлюється прапорець SI. Код статусу, що перебуває в регістрі SMB0STA, говорить про успішну передачу пакета, а також про одержання підтвердження від веденого пристрою;
- необхідно впевнитися в успішній передачі пакета та одержанні підтвердження, перевіривши вміст регістра SMB0STA. Якщо код статусу відповідає очікуваному, слід записати в регістр SMB0CN значення (не забуваючи скинути при цьому прапорець SI), відповідно до якого модуль SMBus сформує на шині стан СТОП або ПОВСТАРТ. Формування стану СТОП або ПОВСТАРТ почнеться відразу ж після скидання прапорця SI.

Зі сказаного видно, що будь–який етап взаємодії прикладної програми з модулем SMBus складається із наступних трьох частин:

1) Після завершення модулем виконання чергової операції, він встановлює прапорець SI регістра SMB0CN, записує у регістр SMB0STA код

статусу і очікує реакції програми. Доки прапорець встановлено в «1», на лінії SCL утримується НИЗЬКИЙ рівень.

2) Після встановлення прапорця SI користувач повинен занести в регістри модуля значення, які відповідають наступному етапу обміну.

3) Після оновлення вмісту регістрів модуля, користувач повинен сформувати в регістрі SMB0CN команду для виконання наступного етапу обміну. При завантаженні в регістр нового значення необхідно скинути прапорець SI записом у нього логічної 1. Після скидання прапорця модуль почне виконання операції, обумовленої вмістом регістра SMB0CN.

Можливі значення кодів статусу, про які згадується в прикладі, наведено в таблиці 6.5.

6.7.3 Послідовні порти в мікроконтролерах деяких виробників

Нижче в таблицях 6.6…6.10, як приклад, наведено деякі мікроконтролери різних виробників, які містять відповідну кількість послівних портів. З повним списком ви можете ознайомитися в довідковій літературі.

Таблиця 6.6 – Мікроконтролери фірми Philips

Позначення	Послідовні канали
На базі 6–тактного ядра	
8xC524	UART, I2C
8xC576	UART
8xC748	–
8xC751	I2C
На базі ядра 51MX	
87C51MB2	2 UART, SPI
87C51MC2	2 UART, SPI

Продовження таблиці 6.6

Сімейство SOC5XA	
P51XAG1x	2 UART
P51XAC37	UART, SPI, CAN 2.0
P51XAH40	4 UART

Таблиця 6.7 – Мікроконтролери фірми ATMEL

Позначення	Послідовні канали
AT89LS8252	UART, SPI
AT89S51	UART
AT89S4D12	SPI
T89C51IC2	UART, SPI, I2C
AT89C52	UART

Таблиця 6.8 – Мікроконтролери фірми Siemens

Позначення	Послідовні канали
C501G	UART
C505C	UART, CAN
C509	2xUART
C517A	2xUART

Таблиця 6.9 – Мікроконтролери фірми Dallas Semiconductor

Позначення	Послідовні канали
DS80C310	UART
DS80C320	2xUART
DS80C520	2xUART
DS80C350	2xUART

Таблиця 6.10 – Мікроконтролери фірми Silicon Laboratories

Сімейство	Послідовні канали
C8051F0xx	SPI, SMBus (I^2C), UART
C8051F3xx	SPI, SMBus (I^2C), UART
C8051F2xx	SPI, UART
C8051F02x	SPI, SMBus (I^2C), 2xUART
C8051F04x	SPI, SMBus (I^2C), 2xUART
C8051F06x	SPI, SMBus (I^2C), 2xUART
C8051F12x	SPI, SMBus (I^2C), 2xUART
C8051F13x	SPI, SMBus (I^2C), 2xUART

ПИТАННЯ ДЛЯ САМОКОНТРОЛЮ

1) Опишіть особливості паралельного та послідовного обміну у мікропроцесорних системах.
2) Опишіть принцип перетворення даних із паралельної форми у послідовну і навпаки.
3) Опишіть структуру послідовного порту мікроконтролера AT89C51.
4) Які регістри використовуються для програмування послідовного порту мікроконтролера AT89C51?
5) Через які лінії мікроконтролера AT89C51 здійснюється обмін у послідовному форматі?
6) Опишіть призначення окремих розрядів регістра SCON для програмування послідовного порту мікроконтролера AT89C51.
7) Як можна змінити швидкість обміну через послідовний порт мікроконтролера AT89C51 в два рази?
8) Опишіть роботу послідовного порту мікроконтролера AT89C51 в різних режимах за часовим діаграмами.
9) Як формується синхрочастота для послідовного порту в різних режимах?
10) Як розраховується швидкість обміну у різних режимах?
11) Дайте характеристику інтерфейсу SPI.
12) Опишіть роботу інтерфейсу SPI.
13) Опишіть програмування інтерфейсу SPI.
14) Дайте характеристику інтерфейсу I^2C.
15) Опишіть роботу інтерфейсу I^2C.
16) Опишіть програмування інтерфейсу I^2C.
17) Опишіть взаємодію прикладної програми з модулем SMBus.

7 ОРГАНІЗАЦІЯ ПАМ’ЯТІ МІКРОКОНТРОЛЕРІВ СІМЕЙСТВА МК–51

7.1 Призначення та місце модуля пам’яті у мікропроцесорних системах

При вивченні модульної структури мікропроцесорної системи (МПС) відзначалося, що одним з основних її модулів є пам'ять (рисунок 7.1). У пам'яті, яку також називають запам'ятовуючим пристроєм (ЗП), зберігаються програми та дані, що використовуються для керування роботою МПС.

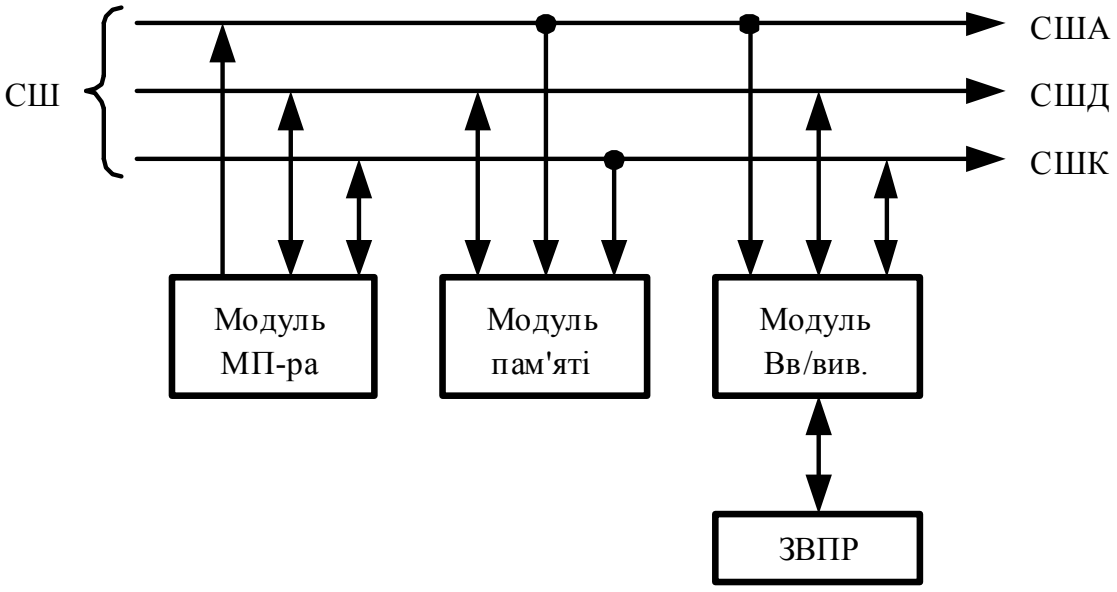


Рисунок 7.1 – Модульна структура МПС

Пам'ять поділяється на:

– оперативний запам'ятовуючий пристрій (ОЗП);

– постійний запам'ятовуючий пристрій (ПЗП).

Керуючі програми повинні розміщуватися в енергонезалежному ПЗП, щоб зберігатися при вимиканні живлення, тому його іноді називають пам'яттю програм. В ОЗП зберігаються дані, що приймають участь у виконанні операцій, тому його називають пам'яттю даних. Такий поділ є дуже умовним, тому що в

процесі роботи системи (великої, складної) в ОЗП із зовнішнього носія інформації може завантажуватися керуюча програма.

Під час роботи системи ПЗП допускає головним чином читання записаної в нього інформації, а в ОЗП можна спочатку записати поточні дані, а потім їх прочитати.

7.2 Класифікація пам’яті

Пам'ять часто розділяють на основну (або внутрішню) і зовнішню. Кожен МП або МК може адресувати деяке число комірок пам'яті, що утворюють єдиний адресний прості даного МП–ра або МК–ра. Цей простір (ці комірки пам'яті) утворює основну (внутрішню) пам'ять МПС. Будь–яку комірку основної (внутрішньої) пам'яті можна адресувати в конкретній системі для читання або для запису інформації (наприклад, якщо США має 16 ліній, то основна пам'ять МП–ра має 2^{16}=65536 комірок).

Зовнішню пам'ять МПС утворюють пристрої тривалого зберігання інформації, наприклад: магнітні стрічки, диски, дискети і т. ін.

Основна (внутрішня) пам'ять виконується, як правило, на напівпровідникових ВІС, її і будемо розглядати в даному розділі.

Основна пам'ять МП–в і МК–в у свою чергу поділяється на: внутрішню (резидентну) пам'ять (РП) і зовнішню пам'ять (ЗП) по відношенню до МП–ра або МК–ра.

Наприклад, внутрішню пам'ять восьми і шістнадцятирозрядних мікропроцесорів утворюють регістри загального призначення (РЗП), що є над оперативним ОЗП (НОЗП), тому що характеризуються мінімальним часом доступу.

До складу МК–в крім РЗП входить ОЗП невеликого об`єму (резидентна пам’ять даних – РПД), наприклад, 128 байт (МК–р AT89C51). Крім ОЗП деякі

МК–ри мають резидентну (внутрішню) пам'ять програм (РПП), наприклад, об`ємом 4 Кбайт (МК–р AT89C51).

Крім наявності РПД і РПП МК–ри можуть адресувати зовнішню основну пам'ять програм (ЗПП) і зовнішню основну пам'ять даних (ЗПД). Наприклад, в МК–рі типу МК–51 AT89C51 загальний об`єм пам'яті програм (РПП і ЗПП) складає 64 Кбайт, загальний об`єм пам'яті даних (РПД і ЗПД) дорівнює 128 байт + 64 Кбайт. При цьому РПП і ЗПП фізично розділені, а логічно сполучені. РПД і ЗПД розділені логічно і фізично.

У залежності від способу звернення до окремих комірок пам'яті розрізняють:

– пам'ять з довільним доступом;

– пам'ять з послідовним доступом.

При довільному доступі (вибірці) допускається будь–який порядок надходження адрес, що лежать в адресному просторі даного процесора.

При адресації комірок пам'яті за другим типом зміна адрес можлива тільки у визначеному порядку – зростання чи зменшення адрес. Прикладом такого доступу є запис і читання інформації на магнітну стрічку. Очевидно, що основна пам'ять МПС є ЗП із довільним доступом.

В залежності від того, як впливає наявність джерела живлення на збереження інформації, пам’ять поділяється на:

– енергозалежну;

– енергонезалежну.

В енергозалежній пам’яті дані при відсутності джерела живлення руйнуються, а в енергонезалежній – ні. Прикладом енергозалежної пам’яті являється статичний оперативний запам’ятовуючий пристрій (SRAM), а прикладом енергонезалежної – FLASH та EEPROM пам’ять.

Основна пам'ять МПС будується, як правило, на МОН–(КМОН) транзисторах і поділяється на статичну і динамічну.

ОЗП динамічного типу часто виконуються на МОН–(КМОН)–транзисторах. Особливістю цих транзисторів є дуже високий вхідний опір, що дозволяє використовувати як елемент пам'яті конденсатор, функції якого виконує вхідна паразитна ємність МОН–транзистора. В результаті виходить так звана динамічна пам'ять.

У статичних ОЗП кожен елемент пам'яті, об’ємом 1 біт, являє собою симетричний тригер, який побудовано, наприклад, на МОН–транзисторах з перехресними зв'язками (рисунок 7.2).

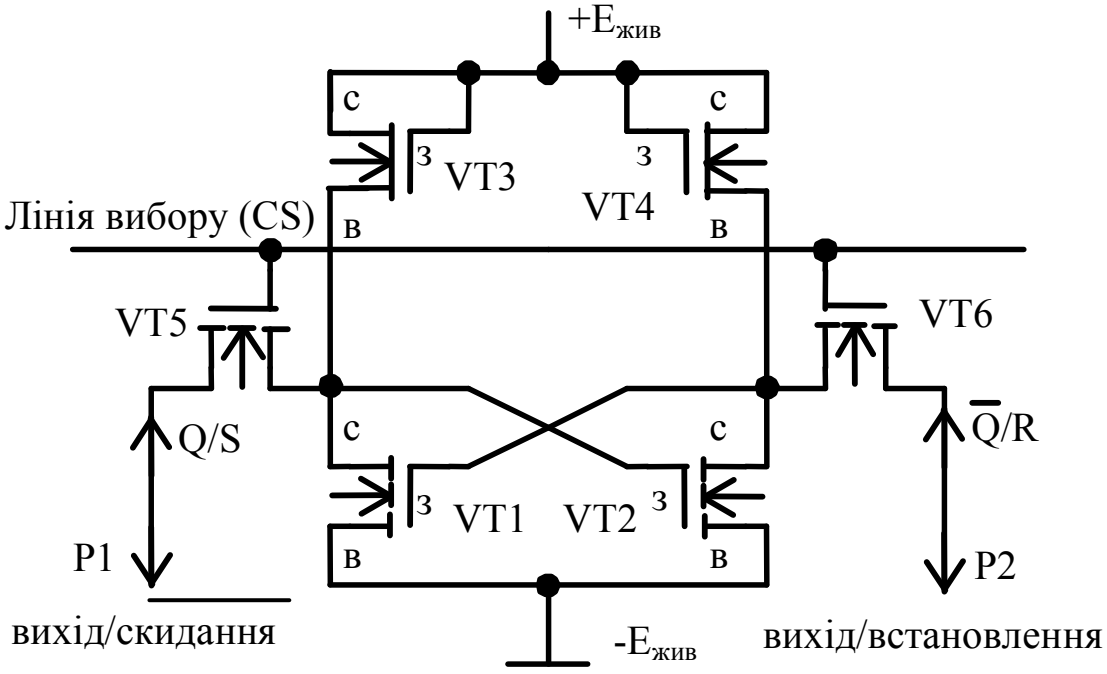


Рисунок 7.2 – Комірка статичної пам'яті

Кожен тригер зберігає один біт інформації. Тригери мають роздільні входи для встановлення їх у 0 чи 1, а також виводи для з'єднання їх з лініями шини даних. Запам'ятовуючі елементи (тригери) на поверхні напівпровідникового кристала великої інтегральної схеми (ВІС) пам'яті з'єднуються необхідним чином для утворення байта, слова і т. ін.

Зазвичай, один тригер складається із шести МОН–транзисторів. Збережена інформація визначається станами транзисторів VT1 і VT2: коли один з них у провідному стані, то інший виключений, і навпаки. Стану, коли VT2 проводить струм, а VT1 закритий, умовно присвоюється значення 1, протилежний стан вважається нульовим. Транзистори VT3 і VT4 виконують функції навантажувальних резисторів для VT1, VT2, а VT5 і VT6 діють як керуючі виводи доступу до комірки пам'яті.

При зверненні до пам'яті, спочатку вибирається елемент за допомогою задання високого рівня напруги на лінії вибору – CS. Транзистори VT5 і VT6 включаються, і лінія P1 підключається до затвору VT2, а лінія P2 – до затвору VT1. Для запису в цей елемент одиниці на лінії P1 встановлюється високий рівень, а на лінії P2 низький. При цьому VT2 відкривається, а VT1 закривається. При записі нуля значення на лініях P1 і P2 змінюються на протилежні. У будь–якому випадку подальші стани VT1 та VT2 не змінюються до наступної операції запису.

Зчитування здійснюється подачею напруги на лінію вибору – CS: стан VT2 передається на лінію "Скидання", а стан VT1 – на лінію "Встановлення".

У зв'язку з необхідністю регенерації, динамічна пам'ять набагато повільніша, ніж статична. Але при великих об`ємах ОЗП вигідніше використовувати саме динамічну пам'ять, тому що щільність упакування значно вище, а питома вартість збереження, енергоспоживання і тепловиділення істотно нижча, ніж у статичній пам'яті. Тому в сучасних ПК ОЗП побудовано саме на динамічній пам'яті. Статична пам'ять застосовується для побудови "кешу" (cache). Він є додатковим і швидкодіючим сховищем копій блоків інформації основної пам'яті, до яких, ймовірно, найближчим часом буде звернення. Кеш прозорий для програміста, він не являє собою додаткової області, що адресується.

В мікропроцесорних системах головним чином використовуються статичні ОЗП та EEPROM–пам’ять даних.

7.3 Основні характеристики пам’яті

Основними характеристиками (параметрами) пам’яті є:

- об’єм;
- розрядність;
- швидкодія;
- потужність, що споживається;
- габарити;
- вартість;
- технологія виготовлення та програмування.

Об’єм пам'яті вимірюється кількістю комірок, що вона містить. Розрядність відображає кількість біт інформації, що містяться в одній комірці пам'яті. З метою зменшення потужності, що споживається від джерел живлення, зменшення габаритів та вартості, сучасні ВІС пам'яті виконуються за МОН–(КМОН)–технологією. Основним параметром, який характеризує швидкодію пам'яті, є час доступу до пам'яті.

Розрізняють час доступу при читанні та записі. Час, необхідний для виведення інформації з пам'яті на шину даних після одержання адреси потрібної комірки, називається часом доступу при читанні. Час доступу при записі – час, необхідний для запису даних в область, що адресується. Час доступу сучасних напівпровідникових ВІС складає одиниці–десятки наносекунд.

7.4 Призначення та організація стеку

Стеком називається спеціальна область пам'яті (ОЗП), що головним чином створюється і використовується при організації і виконанні підпрограм. Стек в МПС поділяється на:

– апаратний;

– програмний.

Властивості стеку залежать від типу МП–ра або МК–ра. В МПС, в яких стек реалізовано апаратно, відсутні команди запису в стек або читання зі стеку (PUSH/POP) і стек використовується тільки для зберігання адрес повернення із підпрограм. Більшість МПС мають програмний стек, який виконує функцію апаратного стеку, а також дає можливість користуватися стеком командами PUSH/POP. Для адресації комірок стеку МП–ри та МК–ри містять спеціальні регістри–покажчики стеку (SP). До виконання команди роботи зі стеком SP – адресує "верхівку" стеку – останню комірку стекової пам'яті, у яку записана інформація, або першу вільну комірку пам’яті (залежить від типу МП–ра або МК–ра). Для запису даних у стек існує команда з мнемонікою PUSH, а для читання – POP. Механізм виконання цих команд в "інтелоподібних" МП дуже схожий. По–перше, стек заповнюється в бік менших адрес; по–друге, обмін зі стеком відбувається словами (2 байти), причому при записі спочатку записується старший байт слова, а потім – молодший, а при читанні – навпаки; по–третє, виконується правило обміну зі стеком: першим увійшов (записався) у стек – останнім вийшов (був прочитаний) зі стеку, тобто стек це набір даних типу FILO ("first in last out").

Виконання команд PUSH і POP в МК–рах типу МК–51 відрізняється від описаного: по–перше, стек заповнюється в бік зростання адреси, по–друге, обмін зі стеком відбувається байтами. Принцип "першим увійшов, останнім вийшов" зберігається.

У 8–розрядних МП–х для стеку виділяється спеціальна частина загального адресного простору способом "карти пам'яті". У 16–розрядних МП для стеку використовується окремий сегмент, максимальний об`єм якого складає 64 байти. У 8–розрядних МК–в, наприклад, AT89C51, стек

розташовується всередині резидентної пам'яті даних, об`єм якої складає 128 байт.

7.5 Місце модуля пам’яті у структурі мікроконтролера

Мікроконтролери сімейства МК–51, наприклад AT89C51, включає два види пам’яті (рисунок 2.1):

– резидентну пам'ять даних (РПД);

– резидентну пам'ять програм (РПП).

7.5.1 Резидентна пам'ять даних

Пам'ять даних ділиться на внутрішню (резидентну) – РПД, і зовнішню – ЗПД. Резидентна пам'ять даних (РПД) призначена для прийому, збереження і видачі інформації, яка використовується в процесі виконання програми. До складу вузла РПД входить статичний оперативний запам’ятовуючий пристрій (СОЗП – SRAM) ємністю 128 байт і дешифратор адреси. Керують роботою РПД два регістри: РА (RAR) – регістр адреси; РПС (SP) – покажчик стеку.

Регістр адреси РА призначено для прийому і збереження адреси комірки пам'яті, яка обирається за допомогою дешифратора і може містити як біт, так і байт інформації.

СОЗП являє собою 128 8–розрядних регістрів, які призначено для прийому, збереження і видачі різної інформації. 16 із цих регістрів допускають побітову адресацію.

РПС (SP) – регістр–покажчик стеку являє собою 8–розрядний регістр, який призначено для прийому і збереження адреси комірки стеку. При виконанні команд LCALL, ACALL вміст покажчика стеку збільшується на 2. При виконанні команд RET, RETI вміст покажчика стеку зменшується на 2. При виконанні команди PUSH direct вміст покажчика стеку збільшується на 1. При виконанні команди POP direct вміст покажчика стеку зменшується на 1.

Після скидання в покажчику стеку встановлюється адреса 07Н, що відповідає початку стеку з адресою 08Н.

7.5.2 Резидентна пам'ять програм

Пам'ять програм (ПП) призначена для збереження програм і має окремий від пам'яті даних адресний простір об'ємом до 64 Кбайт, причому, для мікросхеми AT89C51 частина пам'яті програм з адресами 0000Н–0FFFH розташована на кристалі МК. Для мікросхеми AT89C51 резидентна пам'ять програм, що розташована на кристалі (РПП), складається з 12–розрядного дешифратора та FLASH–ПЗП ємністю 4Кх8 біт. Запис програм у ПП відбувається під час розробки системи.

Якщо на вивід МК «ВРПП (DEMA)» подається напруга живлення U_{CC} (логічна 1), то звернення до зовнішньої пам'яті програм відбувається автоматично при видачі лічильником команд адреси, яка перевищує 0FFFH. Якщо адреса знаходиться в межах 0000Н–0FFFH, звернення відбувається до пам'яті програм, яка розташована на кристалі (резидентної пам'яті програм).

Якщо на вивід МК «ВРПП» подається сигнал "логічний 0", то внутрішня пам'ять програм відключається, і, починаючи з адреси 0000Н, всі звернення виконуються до зовнішньої пам'яті програм.

Для формування поточної 16–розрядної адреси пам'яті програм служить лічильник команд (програмний лічильник) – ЛК (РС). 12 молодших розрядів цього регістра використовуються при адресації комірок РПП об'ємом 2^{12} = 4 Кбайт.

7.6 Організація памяті типового мікроконтролера сімейства МК–51

7.6.1 Загальні відомості

Усі МК–ри сімейства МК–51 мають декілька адресованих просторів пам’яті, які функціонально і логічно розділено за рахунок різниці в механізмах адресації і сигналах керування записом і зчитуванням:

– пам’ять програм: резидентна пам`ять програм (РПП) та зовнішня пам`ять програм (ЗПП);

– пам`ять даних: внутрішня (резидентна) пам’ять даних (РПД) та зовнішня пам’ять даних (ЗПД).

Структуру адресного простору МК–ра, наприклад, AT89C51, показано на рисунку 7.3. Зліва наводяться адреси відповідних областей пам’яті.

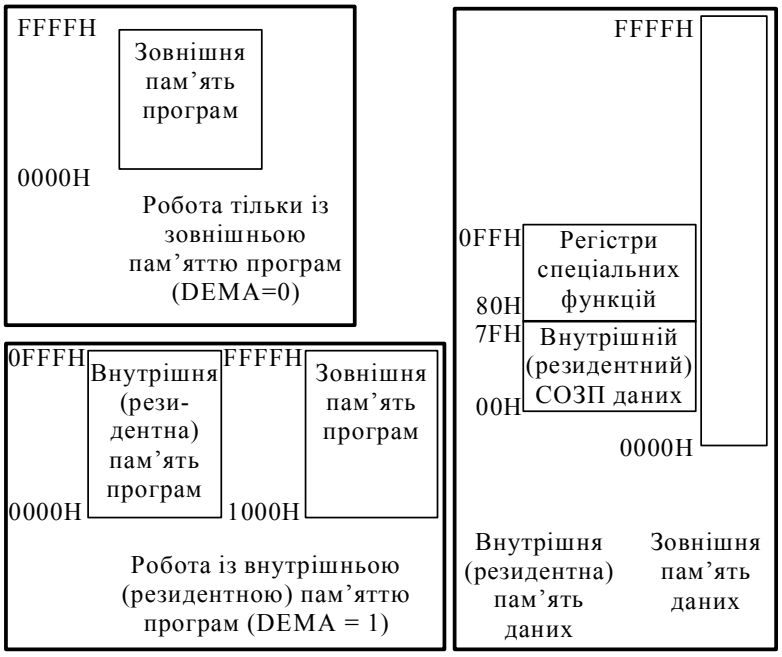


Рисунок 7.3 – Простір пам’яті МК–ра AT89C51

7.6.2 Пам'ять програм

Пам’ять програм (ПП) має 16–бітову адресну шину і доступна тільки для зчитування. МК–р не має команд і сигналів керування, які призначено для запису в пам’ять програм. Пам’ять програм має байтову організацію і загальний об’єм до 64 Кбайт. МК–р AT89C51 має розташовану на кристалі внутрішню пам’ять програм ємністю 4 Кбайт, яка може бути розширена до 64 Кбайт за рахунок підключення мікросхем зовнішньої пам’яті програм. Внутрішня пам’ять програм являє собою ПЗП, який формується при виготовленні МК–ра.

З точки зору програміста може бути використаний тільки один вид пам’яті програм об’ємом 64 Кбайти. Той факт, що в ряді МК–в він утворюється комбінацією масивів, які знаходяться на кристалі і поза у співвідношенні 4 К/60 К для програміста невідчутний, тому що МК–р автоматично вибирає байт із відповідного масиву згідно з його адресою.

Сигналом, який стробує вибірку і введення байту із зовнішньої пам’яті програм у МК–р є сигнал "дозвіл ЗПП" – $\overline{ДЗПП}$ ($\overline{PME}$). Для МК–ра, що містить внутрішню пам’ять програм, $\overline{PME}$ формується тільки в тому випадку, якщо адреса в лічильнику команд перевищує максимальну адресу внутрішньої пам’яті програм 0FFFH (тобто для вибірок із внутрішньої пам’яті програм не формується $\overline{PME}$).

Для МК–ра, що не мають внутрішньої пам’яті програм, сигнал $\overline{PME}$ формується при будь–якому зверненні до пам’яті програм.

МК–ри сімейства МК–51 мають зовнішній вивід "відключення РПП" – ВРПП (DEMA), за допомогою якого можна заборонити роботу внутрішньої пам’яті програм, для чого необхідно подати на вивід DEMA сигнал "логічний 0". При цьому внутрішня пам’ять програм відключається і, починаючи з нульової адреси, всі звернення відбуваються до зовнішньої пам’яті програм із формуванням сигналу $\overline{PME}$. У випадку, якщо DEMA = 1, працюють як внутрішня (резидентна), так і зовнішня пам’ять програм.

Для МК–в, що не мають внутрішньої (резидентної) пам’яті програм, для нормальної роботи завжди необхідно задавати DEMA = 0.

Таким чином, доступ до зовнішньої пам’яті програм здійснюється в двох випадках:

- при дії сигналу DEMA = 0 незалежно від адреси звернення;
- у будь–якому іншому випадку, якщо програмний лічильник (PC) містить число, більше ніж 0FFFH.

При вибірці з зовнішньої пам’яті програм завжди використовується 16–бітова адреса, молодший байт якої видається через порт P0, а старший байт – через порт P2 МК–ра. Байт із зовнішньої пам’яті програм вводиться в МК–р через порт P0, який у цьому випадку використовується як шина адрес/даних у режимі мультиплексування.

На рисунку 7.4 показано молодші адреси блоків пам’яті програм, які, як правило, виділяються для підпрограм обробки переривань і підпрограми початку роботи МК–ра після скидання.

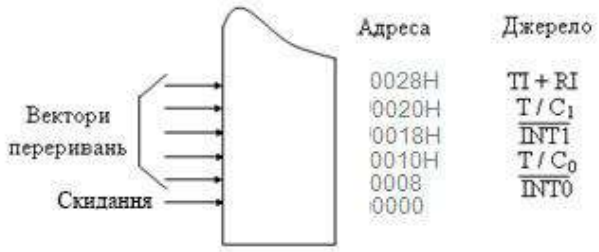


Рисунок 7.4 – Молодші адреси пам’яті програм

7.6.3 Сторінкова адресація зовнішньої пам’яті програм

На рисунку 7.5 наведено функціональну схему, яка реалізує сторінкову адресацію зовнішньої пам’яті програм (ЗПП) мікроконтролера AT89C51.

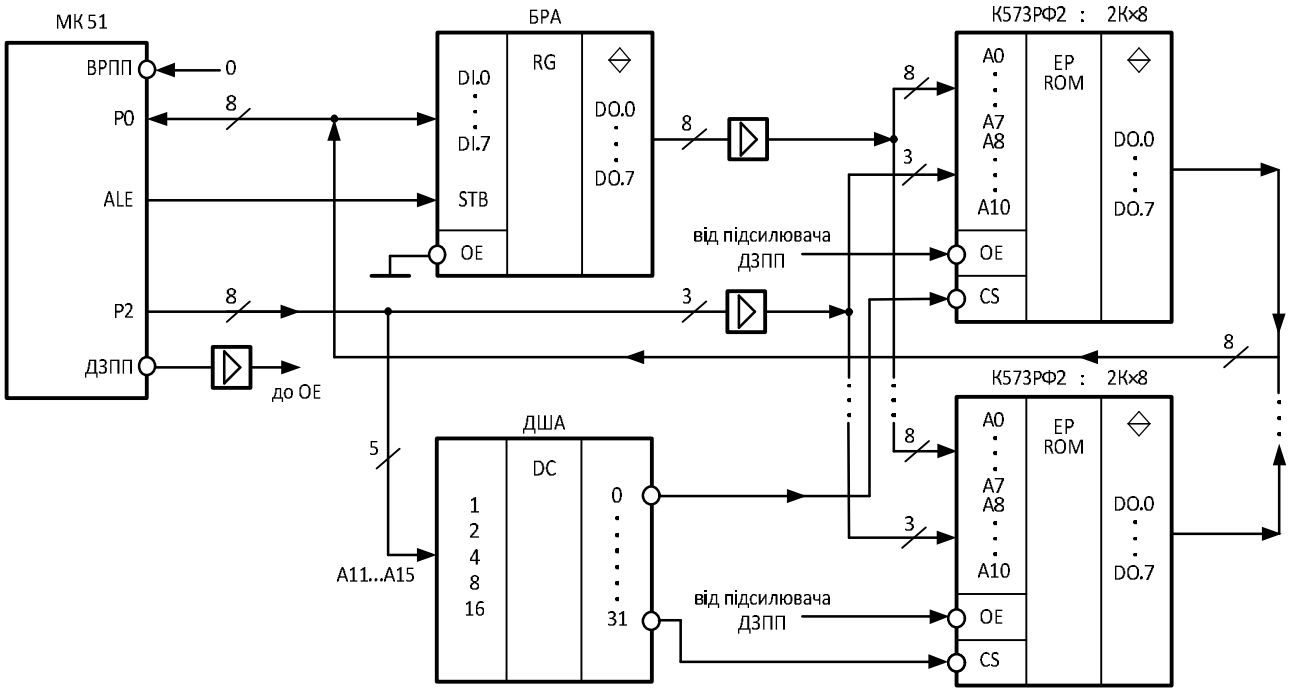


Рисунок 7.5 – Сторінкова адресація зовнішньої пам’яті програм

Пам’ять, об’ємом 65536 (64К) байт, поділено на 32 сторінки, кожна з яких має об’єм 2048 (2К) байт. Сторінки реалізовано на інтегральній мікросхемі пам’яті типу EEPROM: К573РФ2.

При адресації комірок ЗПП на лінії порту P0 спочатку видається молодший байт адреси, а потім ці лінії звільняються для введення в мікроконтролер інформації, яка знімається з виходів D0…D7 мікросхем К573РФ2.

Паралельний буферний регістр адреси (БРА) запам’ятовує молодший байт адреси ЗПП, виконуючи таким чином демультиплексування восьми ліній порту P0.

Молодший байт адреси записується у БРА сигналом ALE, який знімається з відповідного виходу мікроконтролера.

Інші три старші розряди адреси: A8…A10 для мікросхеми К573РФ2 знімаються з виходів порту P2. Старший байт адреси ЗПП постійно присутній на виходах порту P2 протягом всього циклу зчитування інформації із ЗПП.

Для вибору однієї з 32–х мікросхем ЗПП використовуються дешифратор адреси (ДША), на вхід якого подається п’ятирозрядний паралельний двійковий код (ДК), який відповідає розрядам А11…А15 адреси ЗПП та знімається з виходів порту Р2.

ДША має 32 виходи, на одному з яких знімається активний сигнал – логічний 0. На інших виходах при цьому присутні логічні одиниці. Тобто дешифратор перетворює 5–розрядний паралельний вхідний двійковий код у 32–х позиційний унітарний код з активним нульовим рівнем.

Підсилювачі сигналів призначено для підвищення навантажувальної здатності виходів: ДЗПП, порта Р2 мікроконтролера ти виходів DO.0…DO.7 БРА, тому що вони навантажуються на входи 32–х мікросхем К573РФ2, які разом споживають більший струм, ніж спроможні видати на вихід наведені вище виводи.

7.6.4 Пам'ять даних

Пам’ять даних МК–ра, наприклад, AT89C51 поділяється на:

– внутрішню (резидентну) пам’ять даних ;

зовнішню пам’ять даних (ЗПД).

На рисунку 2.4 наведено розподіл адресного простору РПД і області бітів, що адресуються прямо.

Внутрішня пам’ять даних МК–ра складається з двох областей: 128 байт статичної оперативної пам’яті (РПД) з адресами 0–7FH і області регістрів спеціальних функцій, які задаються адресами 80H–FFH. Розподіл простору внутрішньої пам’яті даних показано на рисунку 7.6.

Фізично внутрішній статичний ОЗП даних і область регістрів спеціальних функцій є окремими пристроями.

Як видно з рисунку 7.6, мікроконтролер містить набір регістрів спеціальних функцій, частина яких дозволяє побітову адресацію (рисунок 7.7).

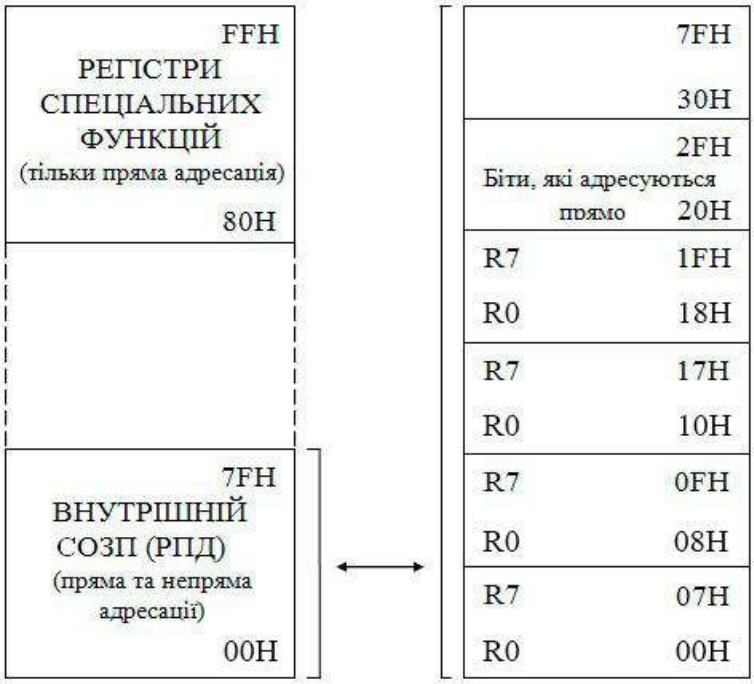


Рисунок 7.6 – Адресний простір внутрішньої статичної пам’яті даних (SRAM) та регістрів спеціальних функцій (РСФ)

Регістри спеціальних функцій (SFR, SPECIAL FUNCTION REGISTERS) та їхні адреси наведено в таблиці 7.1.

Нижче описуються функції цих регістрів.

Акумулятор А. ACC – один з найважливіших регістрів мікроконтролера. Команди, які призначено для роботи з акумулятором, використовують його ім'я "A", наприклад, MOV A, P2. Ім'я "ACC" використовується, приміром, при побітовій адресації акумулятора. Так, наприклад, символічне ім’я п’ятого біта акумулятора при використанні мови асемблера ASM51 буде таким: ACC.5.

Регістр B. Використовується під час операцій множення та ділення. Для інших команд регістр B може розглядатись як додатковий регістр внутрішньої надоперативної пам’яті даних (НОЗП).

Таблиця 7.1 – Регістри спеціальних функцій

Позначення	Найменування	Адреса
* ACC	Акумулятор	0E0H
* B	Регістр B	0F0H
* PSW	Регістр стану програми	0D0H
SP	Регістр–покажчик стеку	81H
DPTR	Регістр–покажчик даних (2 байти):	
DPL	Молодший байт	82H
DPH	Старший байт	83H
* P0	Порт 0	80H
* P1	Порт 1	90H
* P2	Порт 2	0A0H
* P3	Порт 3	0B0H
* IP	Регістр пріоритетів переривань	0B8H
* IE	Регістр дозволу (масок) переривань	0A8H
TMOD	Регістр режимів таймерів/лічильників	89H
* TCON	Регістр керування–статусу таймерів/лічильників	88H
TH0	Таймер/лічильник 0 (старший байт)	8CH
TL0	Таймер/лічильник 0 (молодший байт)	8AH
TH1	Таймер/лічильник 1 (старший байт)	8DH
TL1	Таймер/лічильник 1 (молодший байт)	8BH
* SCON	Регістр керування послідовним портом	98H
SBUF	Буфер послідовного порту	99H
PCON	Регістр керування енергоспоживанням	87H
* – регістри, які допускають побітову адресацію.		

Регістр–покажчик стеку SP. 8–бітовий регістр, вміст якого інкрементується перед записом даних у стек при виконанні команд PUSH і CALL. При початковому скиданні в покажчик стеку заноситься значення 07H, а область стеку в ОЗП даних починається з адреси 08H. При необхідності, шляхом перевизначення покажчика стеку, область стеку може бути розташована в будь–якому місці внутрішнього ОЗП даних мікроконтролера.

Регістр стану програми PSW. Регістр PSW містить інформацію про стан програми. Формат регістра PSW (слова стану програми – ССП) наведено у таблиці 7.2.

Регістр–покажчик даних DPTR. Складається зі старшого байта (DPH) та молодшого байта (DPL). Містить 16–бітову адресу при зверненні до зовнішньої пам’яті. Може використовуватися як 16–бітовий регістр або як два незалежних 8–бітових регістри.

Порт 0 – Порт 3. Регістрами спеціальних функцій P0, P1, P2, P3 є регістри–"защіпки" відповідно портів P0, P1, P2, P3.

Буфер послідовного порту SBUF. Являє собою два окремих регістри: буфер передавача та буфер приймача. Коли дані записуються в SBUF, вони надходять у буфер передавача, причому запис байта в SBUF автоматично ініціює його передачу через послідовний порт. Коли дані зчитуються з SBUF, вони вибираються з буфера приймача.

Регістри таймерів. Регістрові пари (TH0, TL0) та (TH1, TL1) утворюють 16–бітові регістри–лічильники відповідно таймера/лічильника0 і таймера/лічильника1.

Регістри керування. Регістри спеціальних функцій *IP, IE, TMOD, TCON, SCON і PCON* містять біти керування та біти стану системи переривань, таймерів/лічильників, послідовного порту, схеми керування споживанням енергії від джерела живлення.

Таблиця 7.2 – Формат регістра ССП

Позиція	Символ	Ім’я та призначення
PSW.7	C	*Прапорець перенесення*. Встановлюється та скидається апаратними засобами при виконанні арифметичних і логічних операцій. Програмно доступний.
PSW.6	AC	*Прапорець допоміжного перенесення*. Встановлюється та скидається апаратними засобами при виконанні команд додавання і віднімання та сигналізує про перенесення (переповнення) або позику в біті 3 (вважаючи молодший біт нульовим). Програмно доступний.
PSW.5	F0	*Прапорець F0*. Може бути встановлений в 0 чи 1 або перевірений програмою як прапорець, який специфікується користувачем.
PSW.4	RS1	*Вибір банку регістрів*. Встановлюється та скидається програмою для вибору робочого банку регістрів (див. рисунок 7).
PSW.3	RS0	
PSW.2	OV	*Прапорець переповнення*. Встановлюється та скидається апаратно при виконанні арифметичних операцій. Програмно доступний.
PSW.1	–	*Не використовується*.
PSW.0	P	*Прапорець парності*. Встановлюється і скидається апаратно в кожному циклі команди та фіксує непарне/парне (1/0) число одиничних біт в акумуляторі, тобто виконує контроль за парністю.

Всі 21 РСФ допускають пряму байтову адресацію, а 11 РСФ, які у таблиці 7.1 відмічено зірочками, допускають також пряму бітову адресацію (рисунок 7.7).

Прямі адреси байтів (регістрів)	ст. біт (D7)						мл. біт (D0)		Ідентифікатори регістрів, що адресуються прямо
0F0H	F7	F6	F5	F4	F3	F2	F1	F0	B
0E0H	E7	E6	E5	E4	E3	E2	E1	E0	ACC
	CY	AC	F0	RS1	RS0	OV		P	
0D0H	D7	D6	D5	D4	D3	D2	D1	D0	PSW
			PT2	PS	PT1	PX1	PT0	PX0	
0B8H	–	–	BD	BC	BB	BA	B9	B8	IP
0B0H	B7	B6	B5	B4	B3	B2	B1	B0	P3
	EA		ET2	ES	ET1	EX1	ET0	EX0	
0A8H	AF	–	AD	AC	AB	AA	A9	A8	IE
0A0H	A7	A6	A5	A4	A3	A2	A1	A0	P2
	SM0	SM1	SM2	REN	TB8	RB8	TI	RI	
98H	9F	9E	9D	9C	9B	9A	99	98	SCON
90H	97	96	95	94	93	92	91	90	P1
	TF1	TR1	TE0	TR0	IE1	IT1	IE0	IT0	
88H	8F	8E	8D	8C	8B	8A	89	88	TCON
80H	87	86	85	84	83	82	81	80	P0

Рисунок 7.7 – Адреси бітів регістрів спеціальних функцій

Всі комірки внутрішньої пам’яті даних (РПД) можуть адресуватись з використанням прямого і непрямого методів адресації (режими адресації описані в [2,3]. Крім того, внутрішній СОЗП (SRAM) даних має такі особливості (рисунок 7.6).

Молодші 32 байти внутрішнього статичного ОЗП (СОЗП) даних згруповані в 4 банки по 8 регістрів у кожному (БАНК0…БАНК3). Команди програми можуть звертатися до регістрів, використовуючи їх імена R0…R7. Два біти регістра PSW (покажчики банку робочих регістрів RS0 і RS1) визначають банки, із регістрами яких проводяться маніпуляції. Наявність такого механізму роботи з комірками СОЗП дозволяє заощаджувати пам’ять програм, тому що команди, які оперують із регістрами R0–R7, коротші за команди, що використовують пряму адресацію.

Наступні 16 байт, які розташовані після двох банків регістрів внутрішнього ОЗП даних (адреси 20H…2FH) утворюють область комірок, до яких можлива побітова адресація. Набір команд МК–ра сімейства МК–51 містить значну кількість інструкцій, які дозволяють працювати з окремими бітами, використовуючи при цьому пряму адресацію. 128 біт, що складають вказану область внутрішнього ОЗП даних, мають адреси 00H…FH і призначені для використання з такими інструкціями. Бітова адресація ОЗП показана на рисунку 7.8, де в квадратах, що символізують біти, зазначені їхні адреси.

Звернення до внутрішнього СОЗП даних завжди здійснюється з використанням 8–розрядної адреси. При вмиканні живлення вміст СОЗП буде мати випадкове значення.

Зовнішня пам’ять даних формується додатковими мікросхемами пам’яті, що підключаються до МК–ра, і може мати ємність до 64 Кбайт. Простори внутрішньої та зовнішньої пам’яті даних не перетинаються, тому що доступ до них здійснюється за допомогою різних команд. Для роботи з зовнішньою

пам’яттю даних існують спеціальні команди MOVX, які не впливають на внутрішню пам’ять даних МК–ра.

Адреса байта	ст. біт (D7)							мол. біт (D0)	
2FH	7F	7E	7D	7C	7B	7A	79	78	
2EH	77	76	75	74	73	72	71	70	
2DH	6F	6E	6D	6C	6B	6A	69	68	
2CH	67	66	65	64	63	62	61	60	
2BH	5F	5E	5D	5C	5B	5A	59	58	
2AH	57	56	55	54	53	52	51	50	
29H	4F	4E	4D	4C	4B	4A	49	48	
28H	47	46	45	44	43	42	41	40	
27H	3F	3E	3D	3C	3B	3A	39	38	
26H	37	36	35	34	33	42	31	30	
25H	2F	2E	2D	2C	2B	2A	29	28	
24H	27	26	25	24	23	22	21	20	
23H	1F	1E	1D	1C	1B	1A	19	18	
22H	17	16	15	14	13	12	11	10	
21H	0F	0E	0D	0C	0B	0A	9	8	
20H	7	6	5	4	3	2	1	0	
1FH	**БАНК 3**								R7
18H									R0
17H	**БАНК 2**								R7
10H									R0
0FH	**БАНК 1**								R7
08H									R0
07H	**БАНК 0**								R7
00H									R0

Рисунок 7.8 – Організація побітової адресації ОЗП

Таким чином, у системі можуть одночасно бути присутні внутрішня пам'ять даних з адресами 00H…FFH і зовнішня пам’ять даних з адресами 0000H…FFFFH. Звернення до комірок зовнішньої пам’яті даних здійснюється тільки з використанням непрямої адресації за допомогою регістрів R0 і R1 активного банка регістрів внутрішнього СОЗП (команди типу MOVX @Ri) або за допомогою регістра спеціальних функцій DPTR (команди типу

MOVX @DPTR). Відповідно, в першому випадку буде формуватися 8–розрядна, а в другому випадку 16–розрядна адреси зовнішньої пам'яті даних.

При зверненнях до зовнішньої пам'яті даних адреса виводиться через порт P0 (молодший байт) і порт P2 (старший байт) МК–ра. Обмін байтом даних (запис і зчитування) проводиться через порт P0 МК–ра, тобто порт P0 використовується як шина адреси/даних у режимі мультиплексування.

Зчитування даних із зовнішньої пам'яті даних у МК–р проводиться за допомогою керуючого вихідного сигналу $\overline{RD}$, а запис даних із МК–ра у зовнішню пам'ять даних за допомогою керуючого вихідного сигналу $\overline{WR}$.

Кожен тип зовнішньої пам'яті (пам'ять програм, пам'ять даних) може бути доданий незалежно від іншого і кожен з них використовує ті ж шини адреси/даних, але інші сигнали керування.

7.6.5 Приклади підключення до мікроконтролера AT89C51 ЗПД та ЗПП

На рисунках 7.9, 7.10 наведені функціональні електричні схеми підключення до МК–ра AT89C51 відповідно ЗПД об'ємом 2 Кбайт (мікросхема К537РУ8) та ЗПП об'ємом 8 Кбайт (мікросхема К573РФ6).

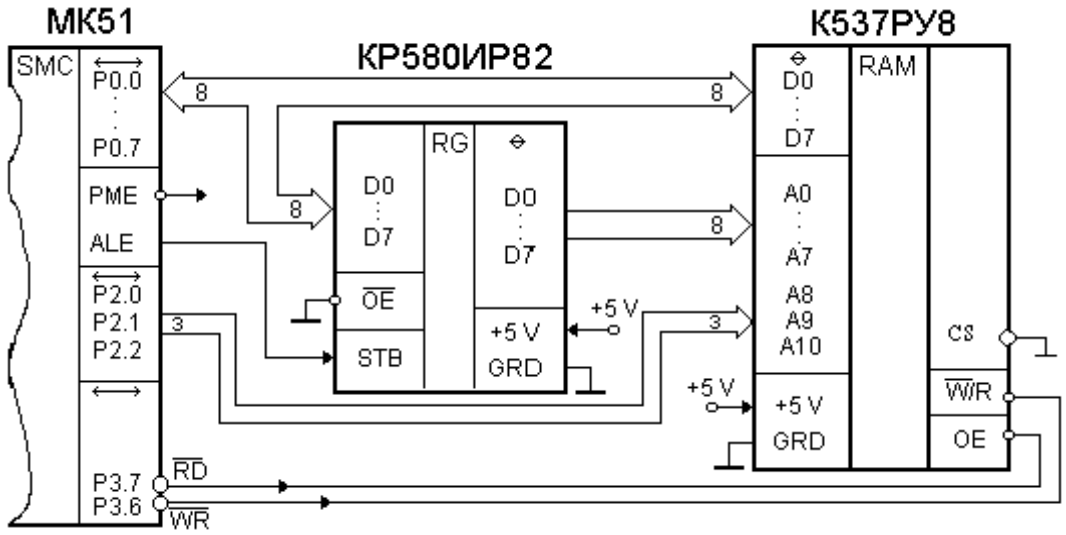


Рисунок 7.9 – Підключення ЗПД (2 Кбайт) до МК–51

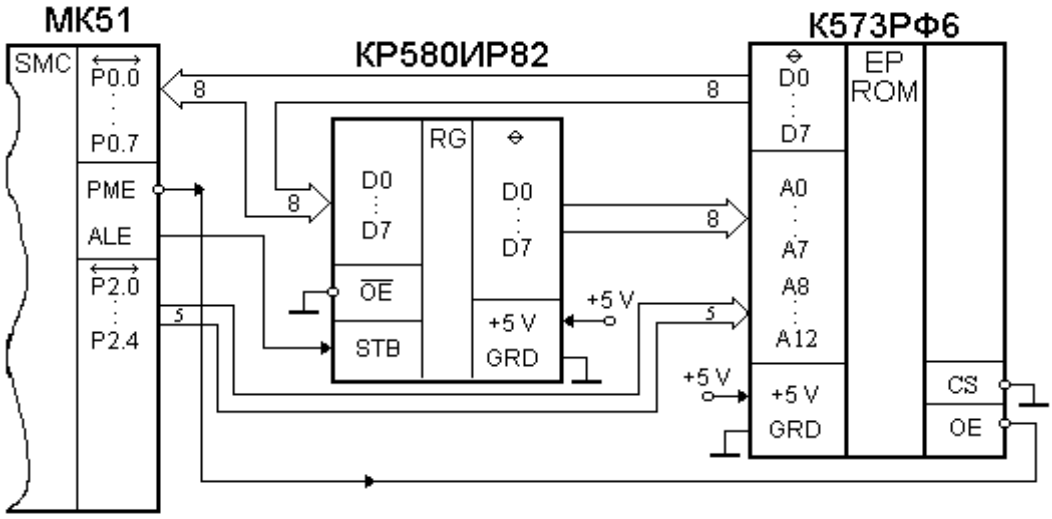


Рисунок 7.10 – Підключення ЗПП (8 Кбайт) до МК–51

Буферний регістр адреси, БРА, (мікросхема КР580НР82) призначений для демультиплексування восьми ліній порту Р0 мікроконтролера. Як зазначалося вище, при роботі з ЗПД або ЗПП спочатку через лінії порту Р0 видається молодший байт адреси ЗПД або ЗПП. При цьому на лінії ALE мікроконтролера формується імпульс–строб, який надходить на вхід STB БРА та защіпає в ньому молодший байт адреси. Після цього лінії порту Р0 звільняються для роботи із ЗПД (читання або запис даних) або ЗПП (читання байтів програми).

Оскільки об’єм ЗПД складає 2 Кбайт, а ЗПП – 8 Кбайт, то додаткові старші біти адреси: 3 для ЗПД та 5 для ЗПП постійно присутні на лініях Р2.0…Р2.4 протягом всього циклу взаємодії з ЗПД або ЗПП.

Вихідні буфери шини даних ЗПД та ЗПП (D0…D7) мають 3 стани: два активних – логічна 1 або логічний 0, та 3–й високоімпедансний стан. Для того, щоб перевести ці буфери у 3–й відключений від виходів стан, потрібно подати на вхід CS логічну 1. В активний стан виходи даних мікросхем переводяться:

– для ЗПД сигналом $\overline{RD}$ (логічний нуль), який формується на виході Р3.7 МК–ра при виконанні команд читання з ЗПД (при цьому CS = 0);

– для ЗПП сигналом $\overline{RME}$ (логічний нуль), який формується на виході PME МК–ра при зверненні до ЗПП (при цьому CS = 0).

Тобто, щоб зробити мікросхеми ЗПД та ЗПП активними, потрібно подати на вхід CS логічний нуль.

На керуючий вхід $\overline{W} / R$ мікросхеми ЗПД подаються:

– сигнал $\overline{WR}$ (логічний нуль), який формується на лінії P3.6 при виконанні команд запису даних у ЗПД;

– логічна одиниця, яка знімається з виходу P3.7 при відсутності запису даних у ЗПД.

7.7 Особливий режим роботи пам’яті мікроконтролера

Пам'ять програм МК–51 заповнюється один раз на етапі розробки МК – ра, і не може бути змінена в завершеній системі.

Тому розглянутий МК–51 не є машиною класичної «фон–нейманівської» архітектури. МК містить пам'ять даних і пам'ять програм, які логічно (програмно) розділені. Отже, оперативна пам'ять даних (резидентна або зовнішня) не може бути використана для зберігання кодів програми, тому що вибірка команд в МК51 виконується тільки з пам'яті програм (внутрішньої і/або зовнішньої). Це пояснюється тим, що у більшості застосувань мікроконтролера потрібна наявність однієї незмінної прикладної програми, яка записується у ПЗП на етапі його розробки, а ОЗП використовується для тимчасового зберігання змінних.

Однак, на етапі розробки і налагодження прикладних програм машина «фон–нейманівського» типу виявляється дуже зручною, тому що дозволяє розробнику оперативно змінювати коди прикладної програми, розташованої в ОЗП.

Для цього МПС з мікроконтролером може бути модифікована з метою суміщення адресних просторів ЗПП і ЗПД шляхом використання зовнішньої логіки, як показано на рисунку 7.11. ЗПП і ЗПД об'єднані у спільному зовнішньому ОЗП, об'ємом 64 КБайт.

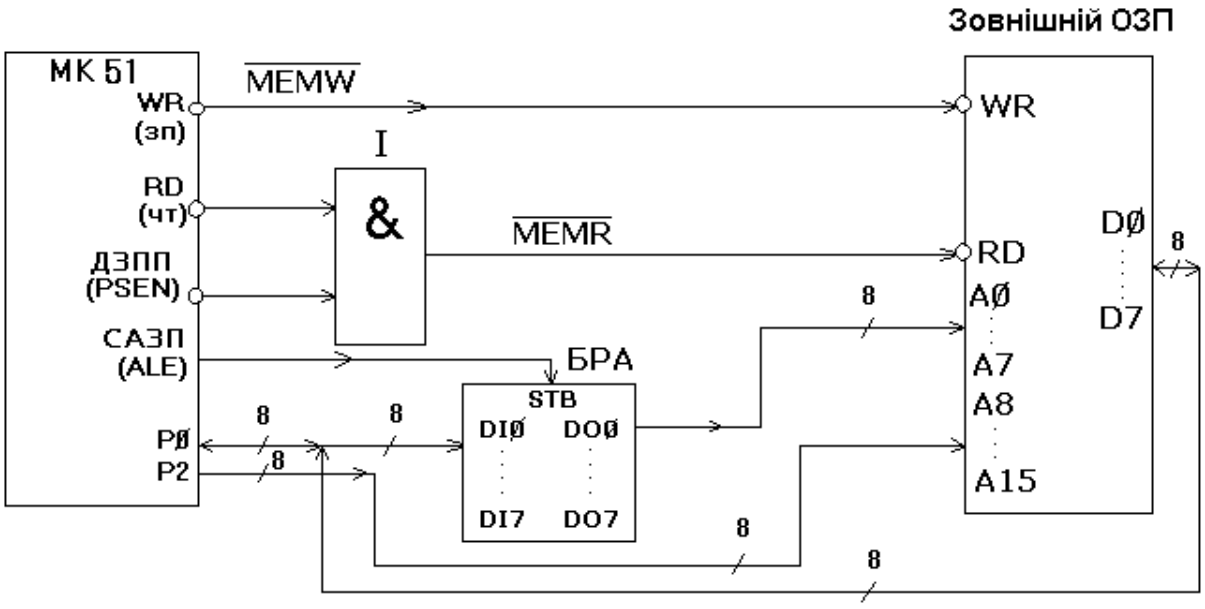


Рисунок 7.11 – Суміщення адресних просторів ЗПД і ЗПП в єдиному зовнішньому ОЗП об'ємом 64 КБайт

Запис даних в ОЗП стробується сигналом $\overline{MEMW}$, що формується на виході WR (P3.6) при виконанні команд запису в ЗПД.

Читання ОЗП стробується сигналом $\overline{MEMR}$, який формується в одному з двох випадків:

– або при читанні ЗПД (супроводжується сигналом $\overline{RD}$ (P3.7) при виконанні команд читання ЗПД);

– або при читанні команд програми з ЗПП (супроводжується сигналом $\overline{PSEN}$, що формується при зверненні до ЗПП).

Логічний елемент І (кон'юнктор) реалізує перемикальну функцію:

$$F = A \wedge B = \overline{\overline{A} \vee \overline{B}}, \qquad (7.1)$$

де змінна А відповідає логічному сигналу на виводі RD (P3.7), а змінна В – на виводі PSEN.

Порт P0 використовується у режимі «мультиплексування». Спочатку через нього виводиться молодший байт адреси зовнішнього ОЗП (ЗПД або

ЗПП), що запам'ятовується у зовнішньому буферному регістрі адреси (БРА). Потім через лінії Р0 здійснюється обмін даними між МК–м і ОЗП. Старший байт адреси ЗПП або ЗПД постійно є присутнім на виході порту Р2 протягом усього циклу роботи з зовнішньою пам'яттю.

При використанні описаного особливого режиму роботи пам'яті варто враховувати, що в МК–рі на фізичному (апаратному) рівні використовується п'ять видів пам'яті: РПД, РПП, ЗПД, ЗПП і блок регістрів спеціальних функцій. Внаслідок цього, версія прикладної програми, що налагоджується у зовнішній суміщеній пам'яті програм і даних (у зовнішньому ОЗП), буде відрізнятися від остаточної версії робочої програми, яка завантажується в пам'ять програм.

Описаний спосіб організації керування зовнішньою пам'яттю може бути використаний у тих застосуваннях МК–51, де потрібні оперативне перезавантаження або модифікація прикладних програм за допомогою пристроїв введення/виведення (ПВВ) як в ЕОМ класичної архітектури.

7.8 Паралельне/послідовне програмування FLASH–пам’яті

Розглянемо приклад програмування FLASH–пам’яті на основі мікроконтролера Aduc847. Aduc847 – мікроконтролер виробника Analog Devices, який побудовано на ядрі МК–51.

У нормальному режимі 62КБ Flash–пам'яті програм можна програмувати двома способами:

– послідовним програмуванням. У складі мікропроцесора ADuC847 є стандартний послідовний порт (UART). Режим послідовного завантаження вмикається автоматично при подачі живлення або після виконання зовнішнього скидання, якщо контакт PSEN підключено через зовнішній резистор 1кОм до землі. Потрапивши в режим послідовного завантаження, мікроконтролер починає виконувати ПЗ (програмне забезпечення) з прихованої частини ПЗП. У цей час

користувач може завантажувати свій код в масив пам'яті програм 62КБ;

– паралельним програмуванням. Режим паралельного програмування повністю сумісний зі стандартними пристроями програматорів EEPROM або Flash–пам'яті від сторонніх виробників. Схему зовнішньої конфігурації контактів, які необхідні для підтримки паралельного програмування, показано на рисунку 7.12.

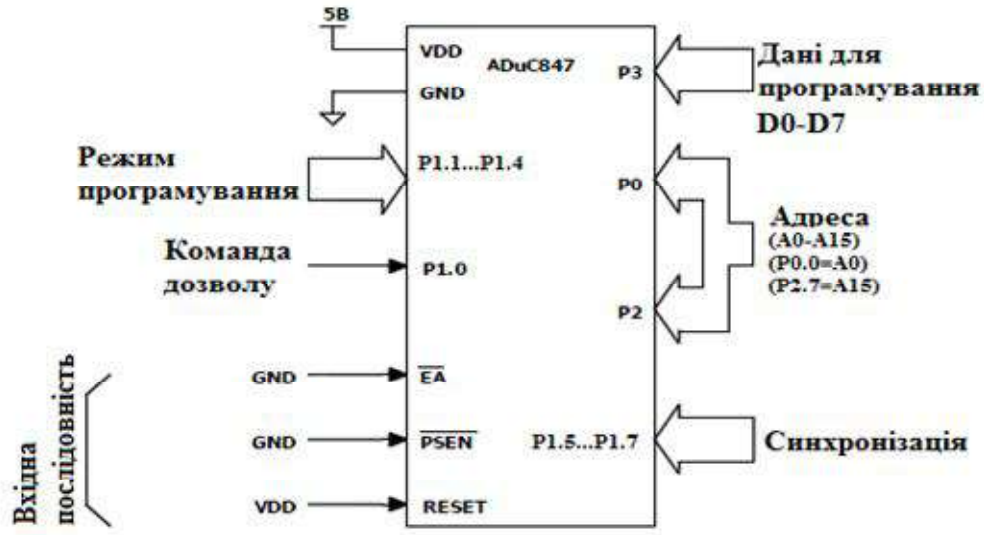


Рисунок 7.12 – Паралельне програмування Flash–пам’яті

У цьому режимі порти P0 і P2 використовуються в якості зовнішнього інтерфейсу шини адреси, а порт P3 – шини даних. Лінія P1.0 використовується як строб дозволу запису даних. Лінії P1.1, P1.2, P1.3 і P1.4 використовуються як порти загального призначення, які встановлюють пристрій у різні режими стирання та програмування під час виконання процесу паралельного програмування.

7.8.1 Режим завантаження Flash–пам’яті користувачем (ULOAD)

На рисунку 7.13 показано, що користувачеві для розміщення його прикладних програм доступний один блок Flash–пам'яті розміром 62 КБайт. У

такому режимі роботи Flash / EE пам'ять являє собою ПЗП виконуваного коду, доступну лише для читання.

Рисунок 7.13 – Карта розподілу Flash–пам'яті програм в нормальному режимі роботи

У режимі ULOAD (завантаження користувачем ПЗ) Flash / EE пам'ять можна використовувати і для запису. Як показано на рисунку 7.14, в режимі ULOAD нижні 56 КБайт пам'яті програм можна використовувати для запису ПЗ користувача. У режимі ULOAD, використовуючи будь–який протокол обміну, можна модифікувати програмний код користувача, який розташовано в цій області пам'яті.

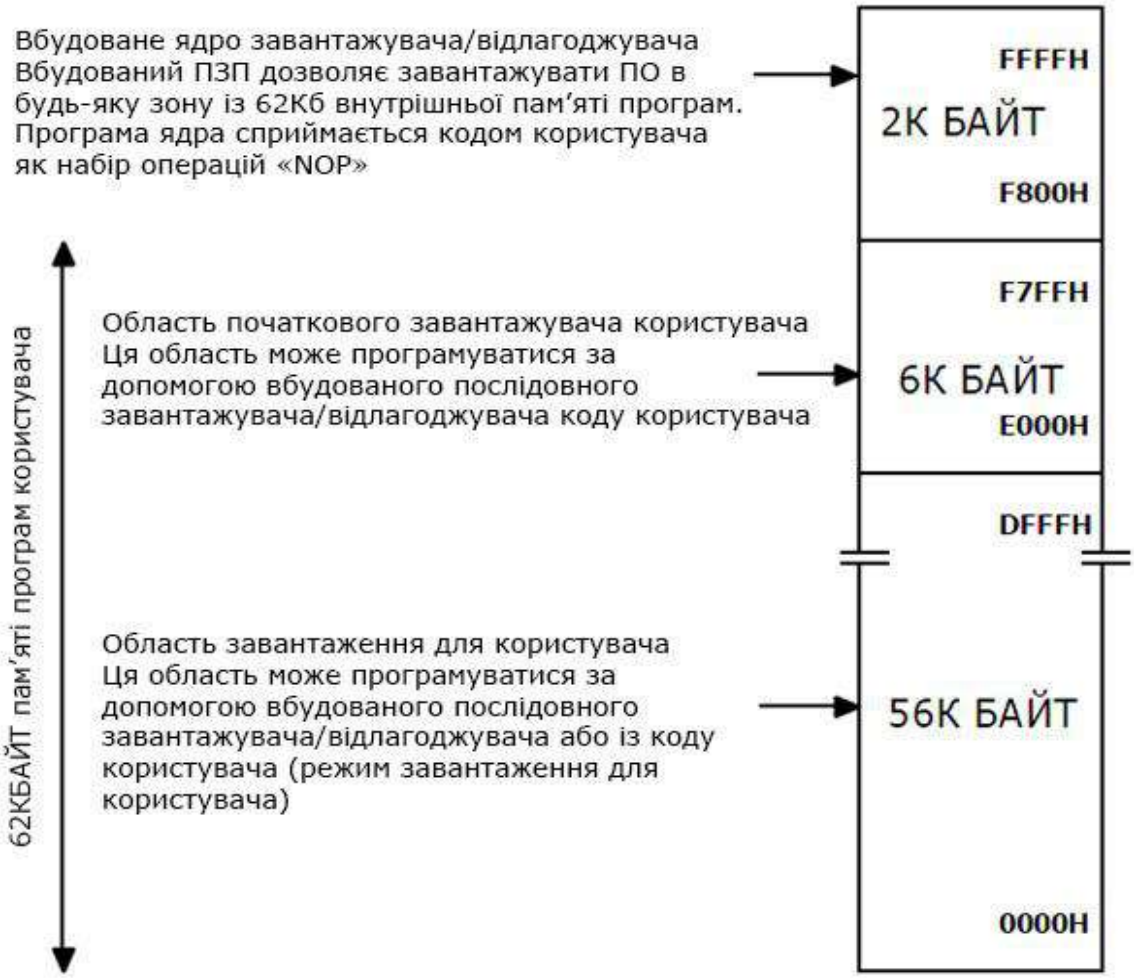


Рисунок 7.14 – Карта Flash–пам'яті в режимі завантаження ПЗ

Крім того, в режимі ULOAD в 56 КБайт Flash–пам'яті можна зберігати дані. Це особливо корисно в додатках, пов'язаних з реєстрацією даних, причому ADuC847 може надати для запису даних 60 КБайт Flash–пам'яті (оскільки існує ще 4 КБайт спеціальної додаткової Flash–пам'яті даних).

Верхні 6 КБайт з 62 КБайт Flash–пам'яті програм програмуються тільки послідовно або паралельно. Це означає, що цей простір пам'яті може бути використано тільки як ПЗП програм користувача. Тому цю область пам'яті неможливо довільно стерти або випадково перепрограмувати. З цієї причини в цій області пам'яті (6 КБайт) зазвичай розміщують початковий завантажувач.

При послідовному програмуванні існує варіант запуску початкового завантажувача, який визначається фразою: «Після сигналу СКИДАННЯ завжди починати роботу з адреси E000h». При завантаженні рекомендується користуватися початковим завантажувачем, оскільки при цьому гарантується, що з появою сигналу СКИДАННЯ завантаження завжди буде виконуватися коректно.

7.9 Програмування FLASH–пам'яті за допомогою інтерфейсу JTAG

Flash–пам'ять програм різних мікроконтролерів також має різний об'єм і розташування, що показано на відповідних картах пам'яті. Основна Flash–пам'ять програм розташована з адреси 0x0000. Чиста Flash–пам'ять (після стирання) містить код 0xFF у всіх байтах. При запису кожного байта, його стирання перед записом здійснюється автоматично. Очікування або визначення завершення операції запису / стирання також не потрібно. Пам'ять розрахована мінімум на 20000 циклів запису / стирання.

Взагалі кажучи, термін "програмна пам'ять" має на увазі пам'ять, доступну тільки для читання. Однак, ядро CIP–51 дозволяє і писати в цю пам'ять при встановленні біта PSCTL.0 (Program Store Write Enable Bit) регістра PSCTL [13]. Запис здійснюється інструкцією MOVX. Цей механізм дозволяє оновлювати програмний код і використовувати пам'ять програм для запису даних, що рідко змінюються, і різних параметрів, які налаштовують.

Програмування Flash–пам'яті здійснюється через послідовний інтерфейс JTAG, який керується утилітами фірми Cygnal. Нагадаємо, що операція запису може тільки очистити будь–який біт, тобто записати логічний 0. Записати одиницю неможливо, тому відновлення одиниць можливо тільки операцією стирання. З цього випливає, що перш ніж записати новий байт в пам'ять, старий байт необхідно стерти. Flash–пам'ять, як правило, має секторну організацію. Стирання проводиться секторами, при цьому всі байти сектора

встановлюються в 0xFF. Природно, що при такій операції стирання, записувати також треба цілий сектор, що буває не дуже зручно.

Однак є й інший, більш зручний шлях. Якщо встановити біт PSEE (Program Store Erase Enable) (PSCTL.l) і біт PSWE (PSCTL.0) в регістрі PSCTL в логічну одиницю, то стає можливим використання інструкції MOVX для запису тільки одного байта (з попереднім автоматичним стиранням), а не цілого сектора. Наведемо типову послідовність алгоритму програмування.

1) Заборонити переривання.
2) Дозволити операції запису / стирання Flash–пам’яті в регістрі FLSCL встановленням біта FLSCL.0.
3) Встановити біт PSEE (PSCTL.l) для дозволу стирання сектора Flash–пам'яті.
4) Встановити біт PSWE (PSCTL.0) для дозволу операції запису у Flash–пам’ять.
5) Використовувати команду MOVX для запису байта даних в будь–яку комірку у середині 512–байтного сектора, який таким чином перед записом буде стертий. Повторювати цю операцію для будь–якої кількості байт від одного до цілого сектора.
6) Скинути біт PSEE (PSCTL.l) для заборони стирання сектора Flash–пам'яті.
7) Скинути біт PSWE (PSCTL.0) для заборони операцій запису у Flash–пам’ять.
8) Дозволити переривання.

Час запису / стирання автоматично контролюється апаратно на основі даних, встановлених у регістрі FLSCL (Flash Memory Timing Prescaler). Чотири біта регістра FLSCL визначають часовий інтервал операцій запису / стирання.

Ядро CIP–51 має багаторівневий захист Flash–пам'яті. Біти PSCTL.0 (Program Store Write Enable) і PSCTL.l (Program Store Erase Enable) регістра PSCTL захищають Flash–пам'ять від випадкової модифікації програмою. Ці біти повинні бути попередньо встановлені в 1 для того, щоб дозволити програмі модифікувати Flash–пам'ять.

Є також кілька байтів захисту, кожен біт яких захищає блок пам'яті програм певного розміру від операцій читання без JTAG–інтерфейсу. Таким чином, одні біти захищають пам'ять від запису / стирання, інші – від читання. Блок Flash–пам'яті, що містить байти захисту, дозволений для запису, але заборонений для читання програмним забезпеченням.

Отже, байт FRLB (FLASH Read Lock Byte) містить біти захисту пам'яті від читання через JTAG–інтерфейс. Логічний 0 у відповідному біті закриває відповідний блок пам'яті і робить його недоступним для читання через JTAG–інтерфейс. Запис в біт логічної одиниці дозволяє читання відповідного блоку.

Байт FWELB (FLASH Write / Erase Lock Byte) містить біти захисту пам'яті від запису та стирання через JTAG–інтерфейс. Логічний 0 блокує запис, логічна 1 – дозволяє запис та стирання.

Ще один тип захисту міститься в регістрі FLACL (FLASH Access Limit Register). Цей регістр містить старший байт 16–бітної адреси, нижче якої програмне читання заборонено. Всі спроби читання цих байтів повертають код 0x00. Таким чином, можливий багатоступеневий захист пам'яті, оскільки блоки, що містять байти захисту можуть бути, в свою чергу закриті. Повне стирання всього захисту можливо тільки при повному стиранні всієї програмної Flash–пам'яті через JTAG–програматор і неможливо з програми користувача.

7.10 Пам’ять даних EEPROM у мікроконтролерах сімейства МК–51

EEPROM (англ. Electrically Erasable Programmable Read–Only Memory це енергонезалежний електрично стираємий програмований ПЗП, ЕСППЗП).

Пам'ять такого типу може стиратися і заповнюватися даними кілька десятків тисяч разів. Використовується в твердотільних накопичувачах та мікроконтролерах. Однією з різновидів EEPROM є флеш–пам'ять.

Розглянемо EEPROM на базі мікроконтролера ADuC824.

ADuC824 має 640 байт EEPROM–пам’яті даних. Вона знаходиться в своєму власному адресному просторі, не пересікаючись ні з адресним простором пам’яті програм, ні з простором статичної пам’яті даних. Сказане означає, що доступ до EEPROM–пам’яті відрізняється від доступу до регістрів, статичної оперативної пам’яті і пам’яті програм.

У мікроконтролера ADuC824 EEPROM–пам’ять організовано в вигляді 160 4–байтних сторінок.

Взаємодія з цією областю пам’яті відбувається через SFR–регістри. Для збереження вмісту 4–байтної сторінки послідовного звернення використовується 4 регістри даних EDATA1…4. Для збереження 8–бітної адреси сторінки, до якої буде здійснено доступ, використовується регістр EADRL. ECON – 8–бітний регістр керування, в який можна записати одну з п’яти команд доступу до EEPROM–даних, забезпечуючи виконання різних операцій: читання, запису, стирання, верифікації.

Об’єми пам’яті в деяких мікроконтролерах різних виробників представлено в таблицях 7.3…7.5.

Таблиця 7.3 – Об’єми пам’яті в мікроконтролерах ATMEL

Тип	ПЗП, КБайт	ОЗП, байт
AT89C51–12AC	4	128
AT89C51–20AC	4	128
AT89C52–12PI	8	256
AT89C52–16JI	8	256
AT89C1051–12PC	1	64
AT89C2051–12PC	2	128
AT89C55–12AC	20	256
AT89S53–16AA	12	256
AT89S8252–16AA	8	256+2048

Таблиця 7.4 – Об’єми пам’яті в мікроконтролерах Philips

Тип	ПЗП, КБайт	ОЗП, байт
8xC51	4	128
8xC52	8	256
8xC54	16	256
8xC58	32	256
8xC51RA+	8	512
8xC51RB+	16	512
8xC51RC+	32	512
8xC51RD+	64	1024
8xC748	2	64
8xC750	1	64

Таблиця 7.5 – Об’єми пам’яті в мікроконтролерах Silicon Labs

Лінійка мікроконтролерів	ПЗП, КБайт	ОЗП, байт
C8051F0xx	32	256/2304
C8051F018, C8051F019	16	1298
C8051F02x	64	4352
C8051F04x	64	4352
C8051F2xx	8	1280/ 256
C8051F30x	8	256
C8051F12x	128	8448
C8051F06x	64	4352
C8051F31x	16	1280
C8051F32x	16	2304
C8051F33x	8	768

ПИТАННЯ ДЛЯ САМОКОНТРОЛЮ

1) Опишіть призначення та місце модуля пам’яті у мікропроцесорних системах.
2) В чому полягає різниця між зовнішньою та резидентною пам’яттю?
3) Опишіть основні характеристики пам’яті.
4) Опишіть призначення та організацію стеку.
5) Чим відрізняються апаратний і програмний стек?
6) Чим відрізняються енергонезалежна та енергозалежна пам`ять?
7) Опишіть сторінкову організацію пам`яті.
8) Як адресуються комірки зовнішньої пам`яті мікроконтролера AT89C51?
9) Опишіть регістри спеціальних функцій мікроконтролера AT89C51.
10) Які регістри спеціальних функцій допускають бітову адресацію?
11) Опишіть організацію пам`яті програм мікроконтролера AT89C51.
12) Опишіть організацію пам`яті даних мікроконтролера AT89C51.
13) Опишіть приклад підключення мікросхеми ЗПД до МК AT89C51.
14) Опишіть приклад підключення мікросхеми ЗПП до МК AT89C51.
15) Назвіть об’єми пам’яті в деяких мікроконтролерах Silicon Labs.
16) Опишіть схему суміщення адресних просторів ЗПД і ЗПП в єдиному зовнішньому ОЗП, об'ємом 64 КБайт.
17) Назвіть та опишіть способи програмування FLASH– та EEPROM–пам’яті.
18) Як програмується поточний банк РОНів РПД?
19) Як можна відключити РПП в МПС на основі мікроконтролера AT89C51?
20) Назвіть адреси підпрограм обробки переривань в МПС на основі мікроконтролера AT89C51.

8 ТАКТУВАННЯ, РЕЖИМ ЗНИЖЕНОГО ЕНЕРГОСПОЖИВАННЯ ТА СКИДАННЯ

8.1 Тактування мікроконтролера

8.1.1 Блок керування та синхронізації

Блок керування та синхронізації призначений для формування синхронізуючих і керуючих сигналів, які забезпечують координацію спільної роботи блоків МК–ра в усіх допустимих режимах роботи.

До складу блока керування входять: пристрій формування часових інтервалів, логіка введення/виведення, регістр команд, дешифратор команд, логічна матриця, що програмується (ПЛМ) і логіка керування контролером.

Пристрій формування часових інтервалів призначений для формування та видачі внутрішніх синхросигналів станів, фаз і циклів. Кількість *машинних циклів* (англ. machine cycle) визначається тривалістю виконання команд. Практично всі команди МК–ра, наприклад AT89C51, виконуються за один або два машинних цикли, крім команд множення MUL AB і ділення DIV AB, тривалість виконання яких складає чотири машинних цикли. Машинний цикл має фіксовану тривалість і містить шість *станів* (англ. states) S1…S6, кожен з який складається з двох часових інтервалів, визначених фазами (англ. phase) P1 і P2 (рисунок 8.1).

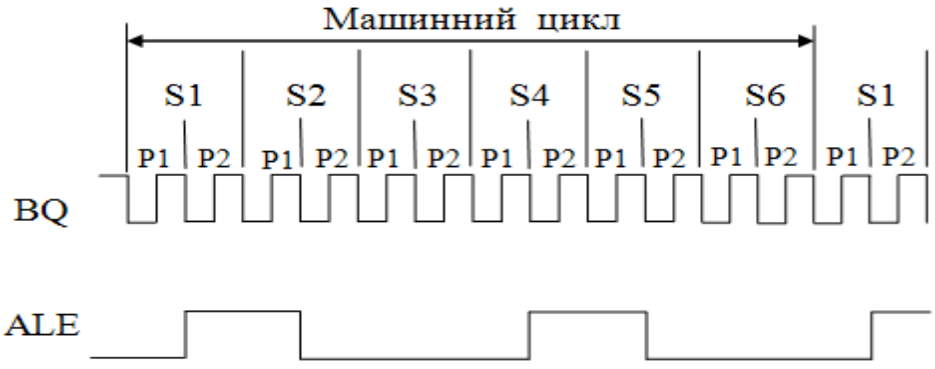


Рисунок 8.1 – Діаграма формування машинних циклів МК–ра

Тривалість фази дорівнює періоду сигналу f_{BQ}, який є первинним сигналом синхронізації МК–ра. Всі машинні цикли однакові, складаються з 12 періодів сигналу f_{BQ}, починаються фазою S1 P1 і закінчуються фазою S6 P2. Двічі за один машинний цикл формується сигнал дозволу фіксації адреси ALE (англ. Address Latch Enable), який видається на однойменний вивід. Якщо, наприклад, зовнішня частота $f_{BQ} = 12$ МГц, то тривалість машинного циклу $T_{МЦ} = 1$ мкс.

Сигнал f_{BQ} формується внутрішнім тактовим генератором МК–ра (при підключенні до її виводів 18 (BQ2) та 19 (BQ1) кварцового резонатора чи LC–ланцюжка) або зовнішнім джерелом тактових сигналів.

Схема підключення кварцового резонатора до виводів BQ2 і BQ1 показана на рисунку 8.2. Дві ємності C1 і C2 призначені для підвищення стабільності роботи генератора тактових імпульсів.

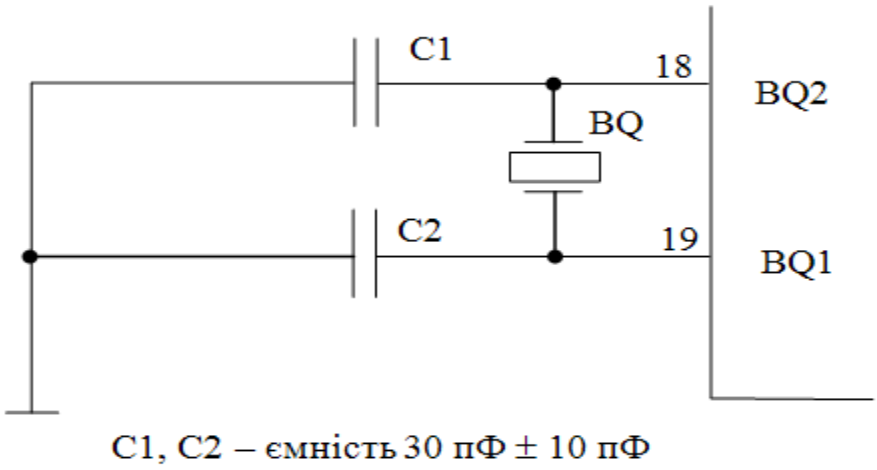


Рисунок 8.2 – Схема підключення кварцового резонатора

8.1.2 Внутрішній тактовий генератор

МК–р містить внутрішній тактовий генератор (OSC) (рисунок 8.3), в якому BQ1 і BQ2 є відповідно входом і виходом підсилювача–інвертора, який може бути включений у режимі генератора при підключенні до виводів BQ1 і BQ2 резонатора або LC–ланцюжка (рисунок 8.4).

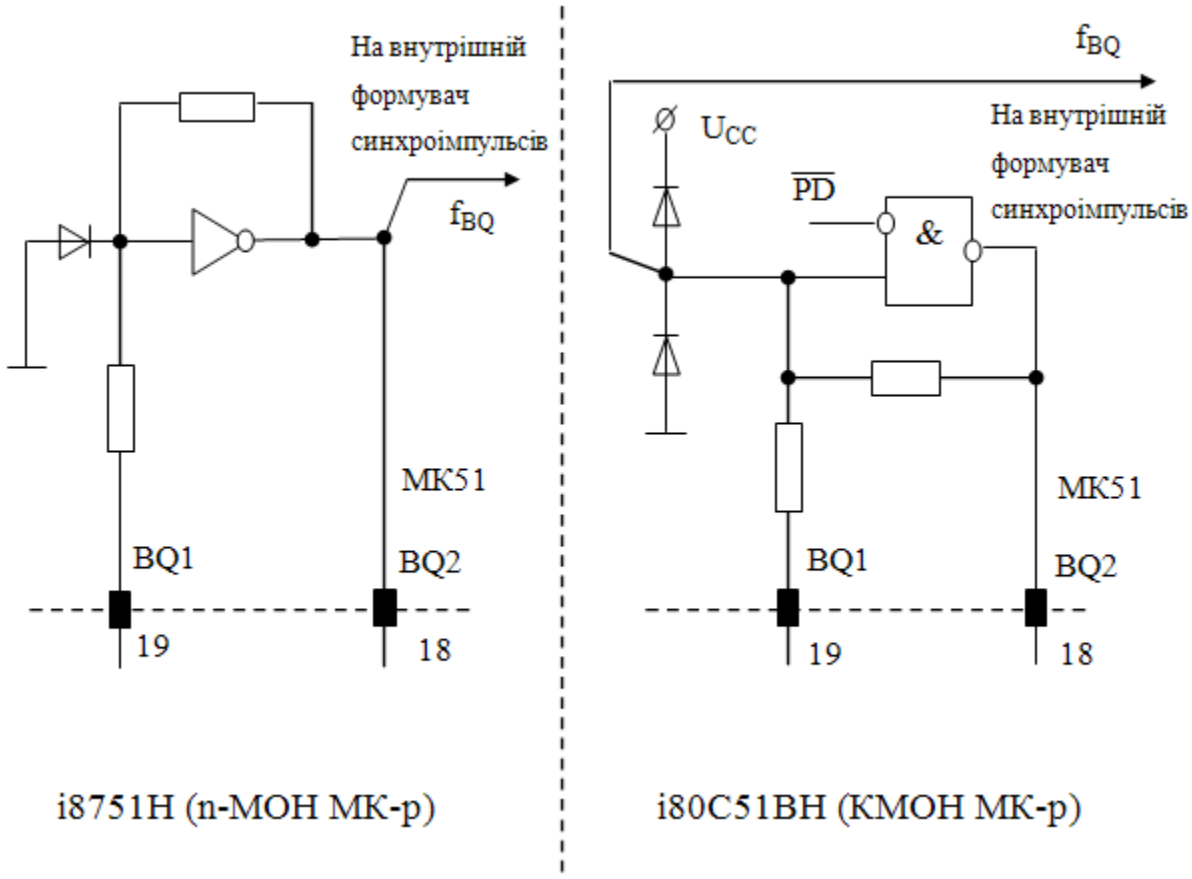
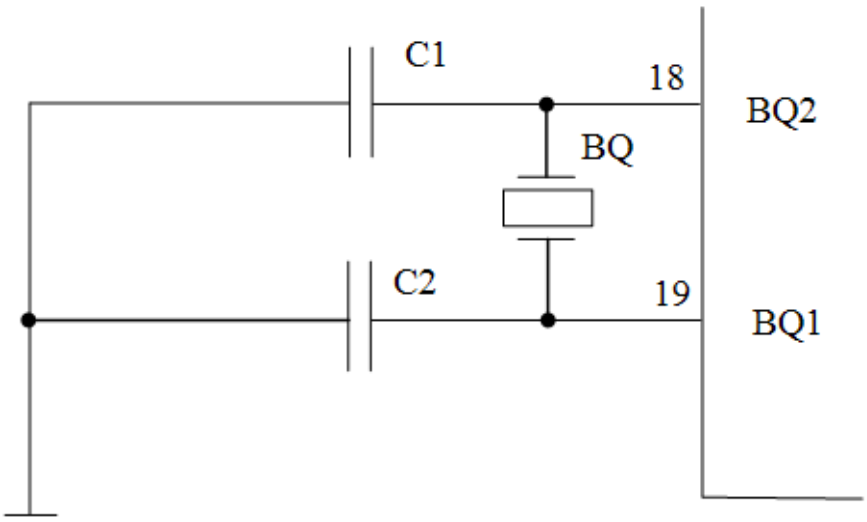


Рисунок 8.3 – Внутрішній тактовий генератор

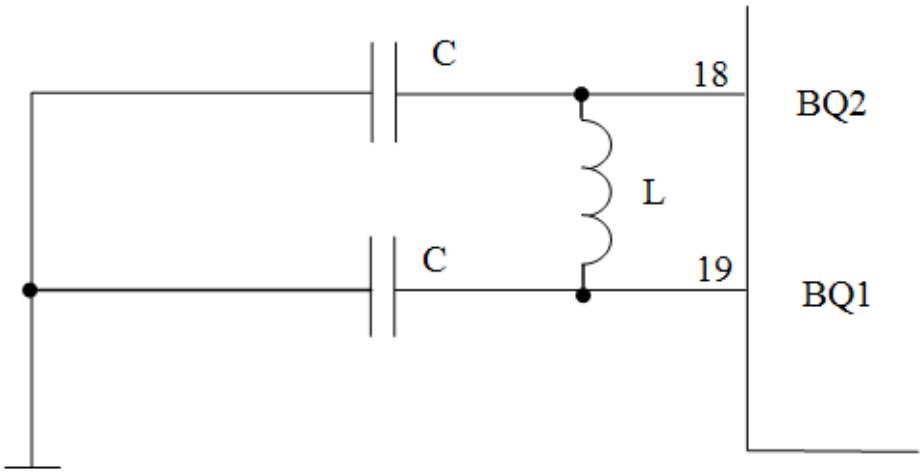
8.1.3 Розвиток архітектури тактового генератора сучасного МК

Розглянемо характеристики та принцип роботи тактового генератора на прикладі мікроконтролера фірми Silicon Laboratories серії C8051Fxxx. Даний мікроконтролер має вбудований багатофункціональний генератор, який починає функціонувати відразу після скидання мікроконтролера на частоті 24 МГц. При необхідності, можна перемкнути тактування мікроконтролера в режим генератора із зовнішнім елементом формування імпульсів, в якості якого може використовуватися кварцовий або керамічний резонатор, конденсатор, RC ланцюжок або зовнішнє джерело тактових імпульсів.



C1, C2 – ємність 30 пФ ± 10 пФ

а)



$$f = \frac{1}{2\pi\sqrt{LC'}} \qquad C' = \frac{C + 3C_в}{2}$$

$C_в$ = 10 пФ – ємність виводу

б)

Рисунок 8.4 – Підключення до виводів BQ1 і BQ2 резонатора і LC–ланцюжка

Перемикання може здійснюватися «на льоту», при цьому робота мікроконтролера не порушується. Це дозволяє працювати на низьких частотах (при малому споживанні) і перемикати тактування на високі частоти тільки в разі потреби, наприклад, для обробки даних від зовнішніх датчиків. Вбудований тактовий генератор можна також перемикати програмно у більшості мікроконтролерів до частоти 16 МГц, а у деяких мікроконтролерах до 24 МГц.

Багатофункціональний генератор складається з внутрішнього і зовнішнього генераторів. Після скидання завжди вмикається внутрішній генератор на частоті 24МГц. Внутрішній генератор може бути дозволений або заборонений або його частота може бути змінена шляхом встановлення відповідних значень регістра OSCICN (Internal Oscillator Control Register).

Під час скидання обидва (внутрішній і зовнішній) генератори загальмовані. Ядро може почати працювати на одній з частот внутрішнього або зовнішнього генераторів, яка визначається бітом CLKSL в регістрі OSCICN. Зовнішній генератор передбачає наявність зовнішнього резонатора (кварцового або п’єзокерамічного), конденсатора або RC–ланцюжка, який під'єднано між виводами XTAL1 і XTAL2. Зовнішній генератор повинен бути також налаштований з використанням регістра OSCXCN. Крім того, може бути використаний зовнішній автономний генератор, вихідний сигнал якого може бути подано на вивід XTAL1. Обидва виводи XTAL1 і XTAL2 розраховані на 3.6V і не можуть працювати з 5V логікою! Очевидно, що перш ніж переключити ядро на роботу від зовнішнього генератора, його необхідно дозволити і запустити.

Деякі очевидні рекомендації. Ланцюги підключення кварцових резонаторів дуже чутливі до всяких завад, тому вони повинні бути мінімальної довжини, тобто резонатор повинен розташовуватися якомога ближче до мікроконтролера. При використанні найбільш поширеного кварцового

резонатора 11.0592МГц, який зазвичай застосовується з іншими x51 сумісними мікроконтролерами, необхідно обидва виводи резонатора зашунтувати на землю конденсаторами з ємністю від 7 pF до 33 pF. Слід відзначити, що на запуск і стабілізацію режиму генератора потрібен певний час, мінімум – 1 ms, як наслідок, після встановлення біта дозволу генерації необхідно зробити відповідну затримку перед встановленням біта XTLVLD. Відсутність затримки може призвести до непередбачуваних результатів.

В якості формувача часових інтервалів може бути використано звичайний RC–ланцюжок. При цьому слід дотримуватися певних рекомендацій. Конденсатор повинен бути не більше 100 pF, але й не менше ~ 7 pF. Також необхідно встановити біти керування частотою зовнішнього генератора XFCN (External Oscillator Frequency Control) в регістрі OSCXCN. Розрахунок частоти генератора можна зробити за формулою:

$$F = 1{,}23 * 10^3/RC .$$

Припустимо, ми встановили конденсатор C = 50pF і резистор R = 246КОм, тоді частота генератора складе:

$f = 1.23 * 10^3/RC = 1.23 * 10^3 / [246 * 10^3 * 50* 10^{-9}] = 10^5$ Hz = 100 kHz

При цьому необхідно підрахувати величину XFCN:

XFCN > $\log_2$ (f/25kHz) = $\log_2$ (100kHz/25kHz) = I$\log_2$ (4) = 2, тобто необхідно встановити код 010.

Вбудований генератор може працювати не тільки з RC–ланцюжком, а й з одним тільки конденсатором, який також повинен бути не більше 100 pF. Визначення частоти здійснюється за формулою: f = KF / (C * VDD), проте коефіцієнт KF вибирається з спеціальної таблиці 8.1.

Таблиця 8.1– Вибір коефіцієнта KF

XFCN	Резонатор (XOSCMD = 11x)	RC (XOSCMD = 10x)	C (XOSCMD = 10x)
000	f ≤32кГц	f ≤25 кГц	KF = 0.87
001	32 кГц < f ≤ 84 кГц	25 кГц < f ≤50 кГц	KF = 2.6
010	84 кГц < f ≤ 225 кГц	50 кГц < f ≤100 кГц	KF = 7.7
011	225 кГц < f ≤ 590 кГц	100 кГц < f ≤200 кГц	KF = 22
100	590 кГц < f ≤ 1,5 МГц	200 кГц < f ≤400 кГц	KF = 65
101	1,5 МГц < f ≤ 4 МГц	400 кГц < f ≤800 кГц	KF = 180
110	4 МГц < f ≤ 10 МГц	800 кГц < f ≤1.6 МГц	KF = 664
111	10 МГц < f ≤ 30 МГц	1.6 МГц < f ≤3.2 МГц	KF = 1590

8.2 Режим зниженого енергоспоживання

8.2.1 Загальні відомості

Мікроконтролери сімейства МК–51 окрім основного (номінального) режиму роботи можуть працювати у режимах зниженого енергоспоживання, які для деяких МК–в сімейства, наприклад, i8751H, i80C31BH, i80C51BH фірми Intel, умовно звуться режимами холостого ходу та мікроспоживання.

Програмування режимів холостого ходу і зниженого енергоспоживання проводиться за допомогою регістра PCON.

Конструкція регістра керування енергоспоживанням (PCON) визначається технологією виготовлення МК–ра: n–MOH або KMOH.

Для варіанта виготовлення за технологією n–MOH (наприклад, i8751H) регістр PCON має всього 1 біт, що керує швидкістю передачі послідовного порту SMOD.

Для варіанта виготовлення за технологією KMOH (наприклад, i80C51BH) позначення розрядів регістра PCON наведено в таблиці 8.2, а призначення розрядів – у таблиці 8.3.

Таблиця 8.2 – Позначення розрядів регістра PCON

Біти	7	6	5	4	3	2	1	0
Позначення	SMOD	–	–	–	GF1	GF0	PD	IDL

Для n–МОН і КМОН МК–в позначення і призначення розряду SMOD ідентичні.

Таблиця 8.3 – Призначення розрядів регістра PCON

Біти	Найменування	Призначення бітів	Примітка
7	SMOD	Біт подвоєння швидкості передачі: при встановленні у стан «логічна 1» – швидкість передачі подвоюється	При роботі послідовного порту
6	–	Резервний	
5	–	Резервний	
4	–	Резервний	
3	GF1	Прапорець загального призначення	
2	GF0	Прапорець загального призначення	
1	PD	Біт вмикання режиму мікроспоживання: при встановленні у стан «логічна 1» – режим мікроспоживання	Якщо в PD і IDL одночасно записано сигнал «логічна 1» – перевагу має PD
0	IDL	Біт холостого ходу: при встановленні у стан «логічна 1» – режим холостого ходу	

Всі біти регістра PCON програмно доступні для запису («логічний 0» і «логічна 1») і читання.

Функції біта SMOD докладно розглянуто при описі роботи послідовного порту.

Біти PCON із номерами 4…6 зарезервовано для розширення сімейства МК–51. При зчитуванні значення цих розрядів не визначено. Програміст не повинен записувати сигнал «логічна 1» у ці біти, тому що вони можуть використовуватися в майбутніх розробках МК–ра сімейства МК–51 для завдання нових функцій. У цьому випадку пасивне значення бітів 4…6 буде «логічний 0», а активне – «логічна 1».

Біти GF1 і GF0 користувач може задіяти за своїм розсудом.

У МК–рах сімейства МК–51, які виконано за КМОН–технологією, є два режими зменшеного енергоспоживання: режим холостого ходу і режим мікро споживання. Джерелом живлення в цих режимах є вивід U_{CC}. Струм споживання від U_{CC} у нормальному режимі складає 18 мА, у режимі холостого ходу – 4,2 мА, а в режимі зниженого споживання – 50 мкА.

8.2.2 Режим холостого ходу

Інструкція, яка встановлює PCON.0 = 1, є останньою інструкцією перед переходом у режим холостого ходу. У цьому режимі блокуються функціональні вузли центрального процесора, що приводить до зменшення енергоспоживання. При цьому зберігаються стани покажчика стеку, програмного лічильника, PSW, акумулятора і всіх інших регістрів, включаючи регістри портів, а також внутрішнього СОЗП даних.

Для закінчення режиму холостого ходу є два варіанти. Активізація будь–якого дозволеного переривання автоматично призведе до встановлення PCON.0 = 0 (закінчення режиму холостого ходу). Після виконання команди RETI (вихід із підпрограми обслуговування переривання) буде виконано

команду, яка є наступною за командою, що перевела МК–р у режим холостого ходу.

Біти GF0 і GF1 зручно використовувати для індикації режиму, у якому була викликана підпрограма опрацювання переривання (відбулося це при нормальній роботі МК–ра чи в режимі холостого ходу). Наприклад, команда, яка викликає режим холостого ходу, може також встановлювати один або декілька прапорців (GF0, GF1 або яких–небудь інших). Підпрограма обробки переривання, виконуючи перевірку цих прапорців, може визначити передісторію свого виклику.

Іншим варіантом закінчення режиму холостого ходу є апаратне скидання за входом RST тривалістю не менше двох машинних циклів.

Активний сигнал скидання на виводі RST асинхронно скидає біт IDL (PCON.0). Оскільки тактовий генератор працює, МК–р відразу після скидання IDL починає виконувати програму з команди, яка є наступною за командою, що викликала режим холостого ходу. Між скиданням біта IDL і моментом, коли включиться внутрішній алгоритм скидання, може пройти до двох машинних циклів виконання програми. Внутрішні апаратні засоби МК–ра блокують доступ до внутрішньої пам’яті даних протягом зазначеного часу, але не блокують доступ до портів. Якщо при цьому зміна інформації на портах не бажана, то необхідно стежити, щоб за командою, яка встановлює біт IDL, не йшла безпосередньо команда, яка записує інформацію в порт або в зовнішню пам’ять даних.

8.2.3 Режим мікроспоживання

Інструкція, що встановлює PCON.1 = 1, є останньою виконаною командою перед переходом у режим мікроспоживання. У цьому режимі тактовий генератор вимикається, припиняючи тим самим роботу усіх вузлів

МК–ра (зберігається тільки вміст СОЗП). Єдиним виходом із цього стану є апаратне скидання RST.

У режимі мікроспоживання напруга U_{CC} може бути зменшена до 2 В і повинна бути відновлена до номінальної величини перед виходом із цього режиму.

Сигнал RST необхідно утримувати в активному стані не менше 10 мс при f_{BQ} = 1 МГц (час відновлення роботи тактового генератора).

При записаних IDL = 1 і PD = 1 перевагу має біт PD.

8.2.4 Режим зниженого споживання для мікроконтролерів n–МОН типу

Під час нормальної роботи внутрішній СОЗП живиться від джерела U_{CC}. Проте для МК–в n–МОН типу сімейства МК–51 у випадку, якщо напруга на виводі RST перевищує U_{CC}, вона стає джерелом живлення для СОЗП. Це реалізовано за допомогою двох внутрішніх діодів, із катодів яких береться живлення СОЗП, а аноди підключені відповідно до входу RST/ VPD–виводу резервного живлення МК–ра (рисунок 8.5).

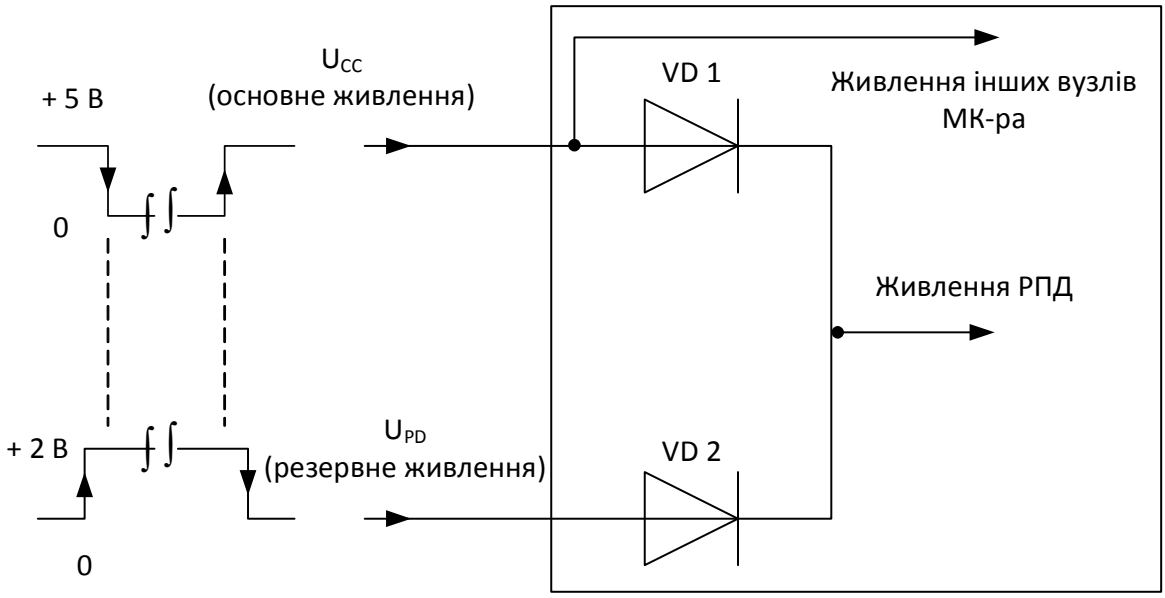


Рисунок 8.5 – Перехід на джерело резервного живлення в МК–х n–МОН типу

Необхідно підкреслити, що в КМОН МК–рах подібний режим відсутній. Даний режим доцільно використовувати при не миттєвих відмовах основного блоку електроживлення МК–ра. У цьому випадку можна забезпечити збереження вмісту РПД (СОЗП) за допомогою малопотужного (батарейного) аварійного джерела живлення $U_{PD} = 2$ В, яке підключається до виводу RST/VPD. Для цього система контролю основного електроживлення при його зниженні формує сигнал зовнішнього переривання МК–ру, що викликає відповідну підпрограму, яка виконує такі дії:

– перезавантажує у РПД основні параметри перерваного перенесення;

– видає сигнал, який дозволяє підключення до виводу RST/VPD аварійного джерела живлення. Напруга $U_{PD} = 2$ В, тому сигнал «СКИДАННЯ» на МК–р не діє, бо мінімальне значення сигналу «RESET» дорівнює 2,5 В.

Ці процедури МК–р повинен встигнути виконати до того, як напруга основного джерела живлення впаде нижче робочої границі. Після відновлення номінального значення напруги в основному ланцюзі живлення джерело аварійного живлення відключається і виконується системне скидання.

8.2.5 Розвиток режимів зниженого енергоспоживання

8.2.4.1 Режим зупинки тактового генератора

Статичне ядро даних МК дозволяє знижувати тактову частоту аж до 0, тобто, до повної зупинки мікроконтролера. При зупинці вміст ОЗП і регістрів SFR не змінюється. Такий режим дозволяє за рахунок зниження тактової частоти знижувати енергоспоживання до заданого рівня. Нижчий рівень споживання досягається шляхом переведення мікроконтролера в режим вимкнення.

8.2.4.2 Режим холостого ходу (режим Idle)

Окрім основного режиму виконання програми RUN, МК типу C051Fxxx

може знаходитись в одному з двох режимів зменшеного енергоспоживання IDLE (холостий хід) чи STOP (зупинка), як це зображено на рисунку 8.6.

Перехід МК з режиму RUN в режим IDLE відбувається після встановлення в 1 біта IDLE (біт 0) в регістрі керування енергоспоживанням PCON (Power Control). В цьому режимі процесор зупинено, вміст пам'яті даних зберігається, однак генератор продовжує працювати, продовжують працювати й периферійні пристрої, які викликані до введення цього режиму, а також контролер переривань. Зупинка процесора дозволяє знизити споживання енергії в цьому режимі до рівня близького 10% від рівня режиму RUN.

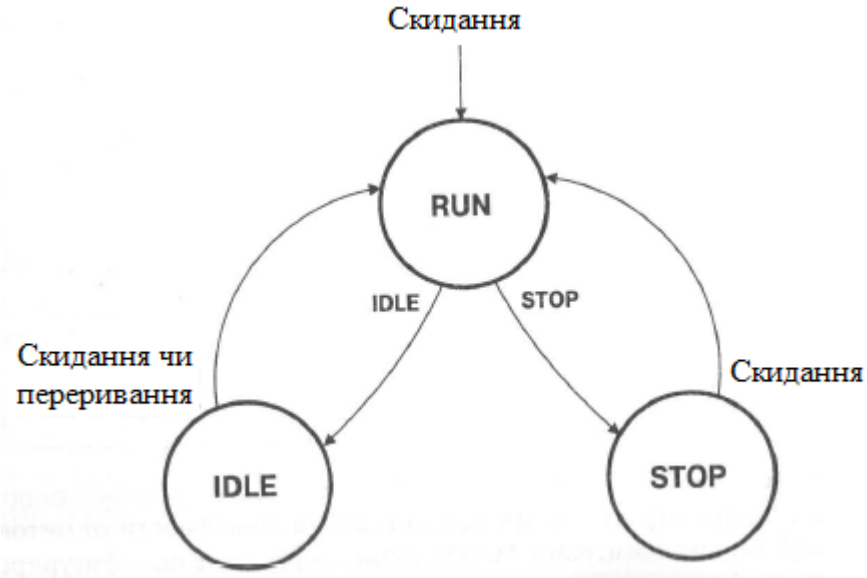


Рисунок 8.6 – Режими зменшеного споживання енергії

Вихід з режиму IDLE можливий чи через переривання, чи в результаті скидання (рисунок 8.6). Якщо вихід з режиму IDLE відбувся в результаті переривання, то біт IDLE в регістрі PCON скидається та процесор відновляє виконання команд. Передусім буде виконано підпрограму обробки переривання, яке відновило роботу процесора, а потім керування буде надано команді, що йде безпосередньо за командою, що визвала встановлення біта IDLE в регістрі PCON та введення режиму IDLE. Якщо вихід з режиму IDLE відбувся в результаті внутрішнього чи зовнішнього скидання, то процесор

виконує всю послідовність апаратного скидання та починає виконання програми з адреси 0000H. Одним з джерел внутрішнього скидання може бути вартовий таймер WDT, якщо він був дозволений до введення режиму IDLE. В цьому випадку після закінчення циклу відліку вартовий таймер викличе скидання та повернення процесора в режим виконання програми RUN. Це застерігає процесор від переходу в режим IDLE в результаті випадкового запису у відповідний біт регістра PCON при дії завад. Якщо за умов роботи системи, яка проектується, МК повинен знаходитись в режимі зменшеного енергоспоживання IDLE значний час, то до введення цього режиму слід заборонити роботу вартового таймера. Це дозволяє знизити споживання енергії в цьому режимі до рівня близького 1% від рівня режиму RUN.

Коли режим холостого ходу переривається апаратним скиданням, виконання поточної інструкції супроводжується виконанням алгоритму внутрішнього скидання, що вимагає для нормального завершення двох машинних циклів. В цей час допустиме звернення до внутрішнього ОЗП, але доступ до портів блоковано. Тому відразу після інструкції, що переводить мікроконтролер у режим холостого ходу, не повинні йти команди звернення до зовнішньої пам'яті або портів введення/виведення.

8.2.4.3 Режим STOP

Перехід МК з режиму RUN в режим STOP відбувається після встановлення в 1 біта STOP (біт 1) в регістрі керування енергоспоживанням PCON. В цьому режимі процесор зупинено, вміст пам’яті даних зберігається, внутрішній генератор зупинено, периферійні пристрої переводяться у вимкнений стан. Кожен з аналогових периферійних пристроїв може бути вимкнено індивідуально до введення режиму STOP. Усе це дозволяє знизити споживання енергії в цьому режимі до рівня близького приблизно 0.1% від рівня режиму RUN.

Вихід з режиму STOP можливий тільки в результаті зовнішнього чи внутрішнього скидання. При цьому процесор виконує всю послідовність апаратного скидання й починає виконання програми з адреси 0000H.

Якщо роботу детектора порушення синхронизації MCD дозволено, то через 100 мкс після переходу в режим STOP відбудеться скидання МК через відсутність синхроімпульсів й він повернеться в режим RUN. Якщо за умов роботи системи, яка проектується, МК має знаходитись в режимі зменшеного енергоспоживання STOP значний час, то до введення цього режиму слід заборонити роботу детектора порушення синхронизації.

8.3 Скидання мікроконтролера

8.3.1 Мікроконтролер AT89C51

Скидання МК–ра здійснюється шляхом подачі на вхід RST сигналу «логічна 1» на час, не менший двох машинних циклів (24 періодів кварцового резонатора) при працюючому внутрішньому тактовому генераторі.

При подачі сигналу скидання на вхід RST внутрішній алгоритм скидання МК–ра проводить такі дії:

– встановлює у стан «логічного 0» лічильник команд PC і всі регістри спеціальних функцій, крім регістрів–защіпок портів P0…P3, регістра–покажчика стеку SP і регістра SBUF;

– регістр–покажчик стеку набуває значення 07H;

– забороняє всі джерела переривань, роботу таймерів/лічильників і послідовного порту;

– вибирає БАНК 0 СОЗП, підготовлює порти P0…P3 для прийому даних;

– у регістрах спеціальних функцій PCON, IP і IE резервні біти набувають випадкових значень, а всі інші біти скидаються в нуль;

– у регістрах SBUF встановлюються випадкові значення;

– встановлює фіксатори–защіпки портів P0…P3 у стан «логічна 1» (відбувається налаштування портів на введення інформації).

Узагальнені дані щодо стану регістрів після скидання МК–ра зазначено в таблиці 8.5.

Таблиця 8.5 – Стан регістрів після скидання

Регістр	Інформація
PC	0000H
ACC	00H
B	00H
PSW	00H
SP	07H
DPTR	0000H
P0…P3	FFH
IP	XXX00000B
IE	0XX00000B
TMOD	00H
TCON	00H
TH0	00H
TL0	00H
TH1	00H
TL1	00H
SCON	00H
SBUF	Невизначено
PCON	0XXX0000B *для KMON типу* 0XXXXXXXB *для n – MON типу*

Сигнал скидання на вході RST не впливає на внутрішній СОЗП даних. Вміст комірок внутрішнього СОЗП даних набуває випадкових значень лише після ввімкнення живлення.

Для n–МОН МК–в автоматичне скидання при ввімкнені живлення U_{CC} може бути реалізовано підключенням входу RST до U_{CC} через конденсатор ємністю 10 мкФ і до шини 0 В через резистор 8,2 кОм.

На рисунку 8.7 показано схему підключення МК–ра для реалізації автоматичного скидання при вмиканні живлення.

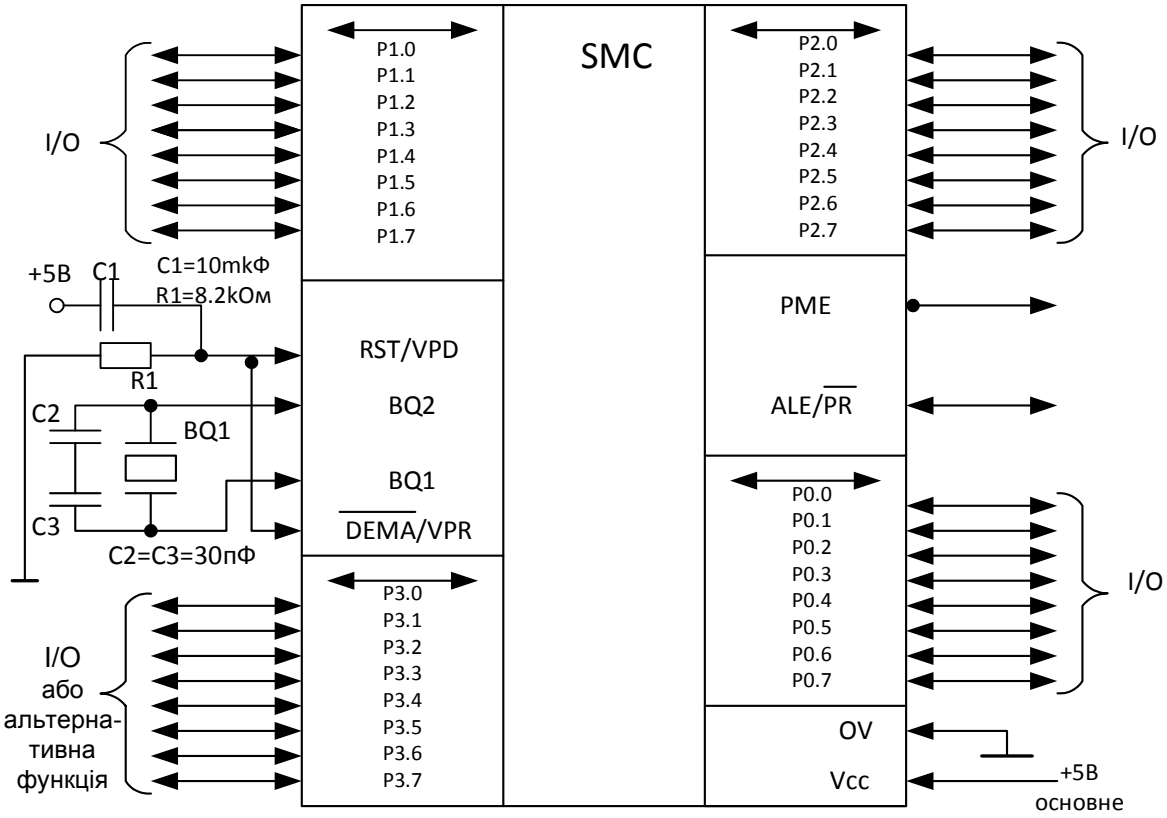


Рисунок 8.7 – Схема включення МК–ра для здійснення автоматичного скидання при вмиканні живлення

Для КМОН МК–в цей резистор не потрібний, проте, його наявність не принесе шкоди (КМОН МК–ри містять внутрішній резистор, підключений між RST і виводом 0 В). Якщо використовується тільки внутрішній резистор, то ємність конденсатора може бути зменшено до 1 мкФ.

Робота ланцюжка R1C1 полягає у наступному. У вихідному стані, коли $U_{CC} = 0$, ємність C1 розряджена. При ввімкнені живлення ємність C1 не може зарядитися миттєво, тому пропускає напругу живлення на вхід RST. Тобто на вході RST присутня логічна 1, що ініціює початок процесу «СКИДАННЯ».

Потім ємність C1 заряджається від джерела живлення через резистор R1, а напруга на вході RST зменшується. В середині МК–ра на лінії RST стоїть тригер Шмітта, який спрацьовує при зменшенні вхідної напруги до відповідного рівня і вимикає внутрішній сигнал «СКИДАННЯ».

Щоб при ввімкнені живлення процедура скидання була гарантовано виконана, вивід RST повинен утримуватись у стані високого рівня протягом часу, достатнього для запуску тактового генератора МК–ра плюс додатково як мінімум два машинних цикли. Час запуску тактового генератора МК–ра залежить від його частоти роботи і для 10 МГц кварцового резонатора складає в середньому 1 мс, а для 1 МГц кварцового резонатора – 10 мс.

В схемі скидання, яку показано на рисунку 8.7, при швидкому зменшенні напруги живлення на вході RST з`являється від'ємна напруги, яка є безпечною для мікросхеми завдяки наявності в структурі МК–ра внутрішньої схеми захисту.

Виводи портів знаходяться у випадковому стані до моменту запуску тактового генератора МК–ра, і тільки після цього внутрішній сигнал скидання записує сигнал «логічна 1» у фіксатори–защіпки портів, налаштовуючи їх на режим введення.

Ввімкнення живлення без забезпечення гарантованого скидання може призвести до того, що МК–р почне виконання програми з деякої випадкової адреси. Це пояснюється тим, що лічильник команд PC не буде скинутий у стан 0000H.

На рисунку 8.8 наведено схему, яка буде формувати сигнал скидання у двох випадках:

– автоматично при ввімкнені джерела живлення (кнопка КН1 замкнена, а кнопка КН2 – розімкнена);

– при короткочасному натисканні на кнопку КН2 (зовнішнє скидання) при замкненій кнопці КН1.

Діод VD1 потрібний для того, щоб після зняття напруги живлення перед повторною подачею напруги живлення забезпечити швидкий розряд ємності С1 через малий внутрішній опір джерела,.

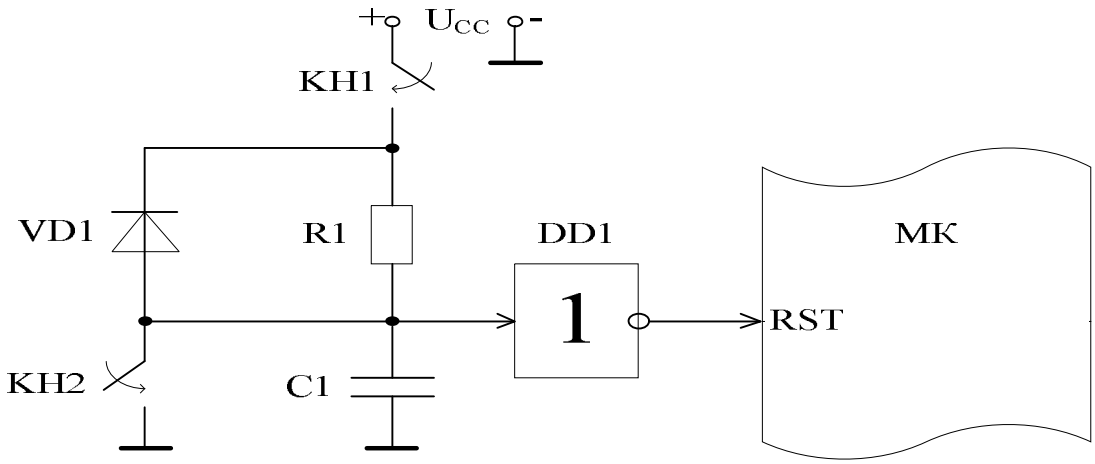


Рисунок 8.8 – Схема формування сигналу «СКИДАННЯ»

Схема працює наступним чином: у вихідному стані $U_{CC} = 0$, ємність розряджена через діод VD1 та малий внутрішній опір джерела живлення. Після замкнення КН1 (КН2 розімкнена) ємність не може миттєво зарядитися, тому на вході інвертора присутній «логічний 0», а на його виході – «логічна 1». Починається скидання МК–ра. Потім ємність заряджається, на вході інвертора напруга збільшується, а на виході – зменшується. Подальшу роботу схеми описано вище. Коли короткочасно натиснути, а потім відтиснути КН2 (КН1 замкнена), то ємність швидко розряджається, а після відпускання КН2 описаний процес повторюється.

8.3.2 Розвиток видів скидання

Підсистема скидання, наприклад, в мікроконтролерах фірми Silicon Laboratories забезпечує можливість скидання мікроконтролерного ядра в залежності від семи різних причин: від вбудованого монітору живлення, який виробляє сигнал скидання при включенні або значному зниженні рівня напруги живлення; від вартового таймера WDT (Watchdog Timer), при відсутності тактових імпульсів на детекторі тактових імпульсів; за сигналом від компаратора 0; при наявності програмного скидання; за сигналом з входів CNVSTR або $\overline{RST}$. Вхід скидання $\overline{RST}$ дозволяє здійснювати зовнішнє скидання. Кожну з перелічених причин скидання, крім входу скидання $\overline{RST}$ і монітора живлення, можна заборонити програмно. Вартовий таймер WDT при скиданні системи стає активним, що потрібно враховувати при розробці програмного забезпечення (його потрібно або заборонити на стадії ініціалізації, або враховувати час його спрацьовування).

При виникненні стану «Скидання» ядро зупиняє виконання програми, переводить всі виводи мікроконтролера в початковий стан, встановлює в початковий стан регістри спеціальних функцій, відновлює початковий стан таймерів і переривань, скидає лічильник адреси в стан 0x0000 і починає виконання програми з цієї адреси. В усі порти введення/виведення записується код 0xFF. Якщо стан «Скидання» виник від монітору живлення та запису 1 в біт PORSF регістра RSTSRC, вивід $\overline{RST}$ переводиться в стан 0 і утримується в цьому стані певний час, необхідний для скидання. Після скидання внутрішній генератор запускається на частоті 2МГц. Вартовий таймер WDT встановлюється на максимально можливий час.

Функціональну схему підсистеми скидання в мікроконтролерах сімейства C8051Fxxx приведено на рисунку 8.9.

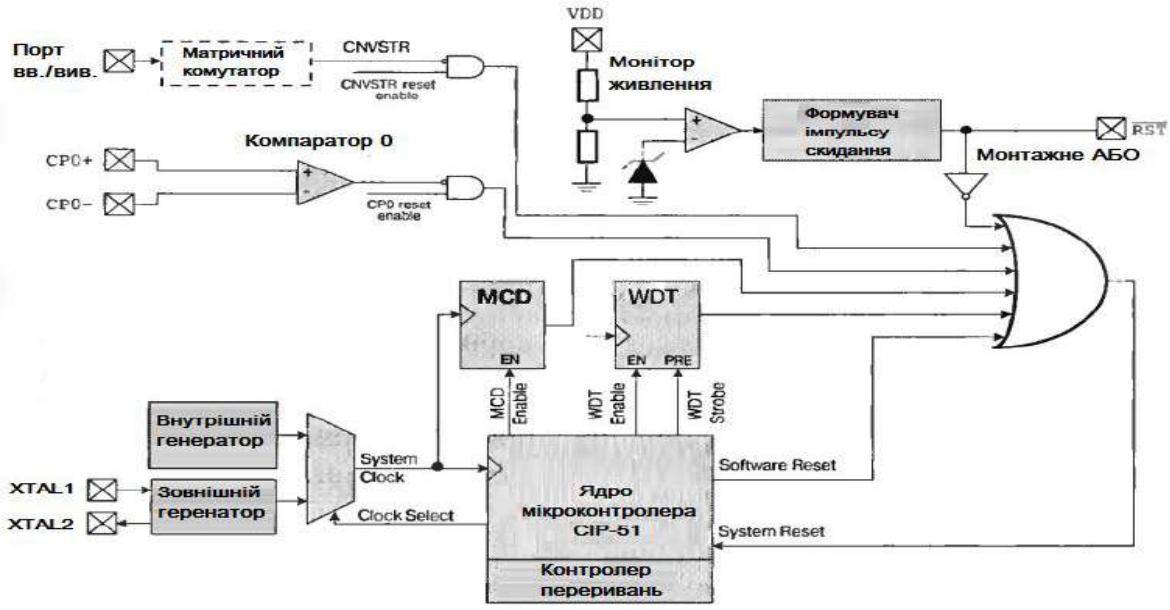


Рисунок 8.9 – Підсистема скидання мікроконтролера CIP–51

Як ми вже відзначали вище, існує сім способів виклику скидання. Розглянемо їх докладніше:

1. При перевищенні певного рівня напруги живлення VRS_T, зазначеного в електричних характеристиках кожного сімейства, виробляється сигнал «СКИДАННЯ ПІСЛЯ ВВІМКНЕННЯ ЖИВЛЕННЯ» (Power–on Reset) RST. Під час цього режиму скидання прапорець RSTSRC регістра PORSF (RSTSRC.1) апаратно встановлюється в логічну одиницю. Всі інші прапорці в регістрі RSTSRC не визначені. Прапорець PORSF дорівнює нулю при скиданні від інших джерел. Оскільки всі способи скидання змушують мікроконтролер починати виконувати програму з нульової стартової адреси, то тільки опитуванням прапорця PORSF мікроконтролер може визначити, що скидання було викликано перевищенням напруги живлення. Це необхідно для визначення стану пам'яті. Зрозуміло, що якщо був встановлений прапорець PORSF – внутрішня пам'ять даних містить довільну інформацію.

2. Іншим способом скидання може бути програмне встановлення прапорця PORSF (RSTSRC.1) в логічну одиницю.

3. У разі навіть короткочасного зниження напруги живлення нижче рівня VRS_T, монітором виробляється сигнал «СКИДАННЯ ВІД ПОРУШЕННЯ ЖИВЛЕННЯ» (Power–fail Reset) з встановленням прапорця PORSF.

4. Четвертою причиною може служити зовнішній сигнал скидання, який подається нульовим рівнем на вхід $\overline{RST}$. Тривалість імпульсу скидання повинна бути більше 12 тактів. Після виходу з цього режиму встановлюється прапорець PINRSF (RSTSRC.0). Нагадаємо, що вивід $\overline{RST}$ сумісний з 5В–рівнями.

5. Ще одним способом виклику режиму скидання є відсутність тактових імпульсів (Missing Clock Detector Reset) протягом більш ніж 100мкс. Після скидання встановлюється прапорець MCDRSF (RSTSRC.2), який дозволяє визначити причину скидання.

6. Ще одним джерелом скидання є компаратор 0, який може бути запрограмований на вироблення сигналу скидання низького рівня і запис високого рівня в прапорець CORSEF (RSTSRC.5). Звичайно, компаратор 0 повинен бути попередньо дозволений бітом CPTOCN.7. При цьому, якщо на вході (CPO+), який не інвертує, напруга нижче, ніж на вході (CPO–), який інвертує, генерується сигнал скидання (Comparator 0 Reset). Слід пам'ятати, що при цьому режимі скидання, імпульс на виводі $\overline{RST}$ не генерується. Слід зауважити, що цей сигнал скидання виробляється як при активному тактовому генераторі, так і при відсутності тактових імпульсів.

7. Наступним джерелом скидання є зовнішній сигнал CNVSTR, який може бути запрограмований, як вхід скидання, чутливий до нульового потенціалу, шляхом запису 1 в прапорець CNVRSEF (RSTSRC.6). Цей сигнал може бути поданий на один з входів перших трьох портів P0, P1 і P2 через комутатор ресурсів Crossbar. Після цього типу скидання (External

CNVSTR Pin Reset) встановлюється прапорець CNVRSEF (RSTSRC.6). При цьому режимі скидання імпульс на виводі $\overline{RST}$ також не генерується.

8. Скидання від вартового таймера. МК містить програмований вартовий таймер (Watchdog Timer – WDT), що працює незалежно від системного тактового сигналу. WDT переводить МК в стан скидання в разі свого переповнення. Щоб запобігти скиданню, WDT повинен перезапускатися з прикладної програми до того, як відбудеться його переповнення. Якщо в системі відбувається програмний / апаратний збій, який не дозволяє програмі перезапустити WDT, то WDT переповниться і викличе скидання. Це запобігає виходу системи з під контролю.

Після кожного скидання WDT автоматично включається і запускається за замовчуванням з максимальним тайм–аутом. При необхідності WDT можна програмно відключити або заблокувати, запобігши його випадкове відключення. Після блокування WDT його не можна відключити до наступного системного скидання. Стан виводу /RST не впливає на скидання цього типу.

WDT складається з 21–розрядної таймера, що працює з тактовою частотою, яка програмується. Цей таймер вимірює період між операціями запису певних значень в його регістр керування. Якщо цей період перевищує встановлену межу, то генерується скидання від WDT. WDT може бути програмно дозволений або заборонений, крім цього можна заблокувати функцію відключення WDT. Керування WDT здійснюється за допомогою регістра керування WDT (WDTCN) [10, 11].

ПИТАННЯ ДЛЯ САМОКОНТРОЛЮ

1) Для чого призначений блок керування та синхронізації мікропроцесора?
2) В чому полягає робота в режимі зниженого енергоспоживання?
3) Поясніть роботу в режимі холостого ходу.
4) Які особливості роботи мікроконтролера в режимі мікроспоживання?
5) Як здійснюється скидання мікроконтролера AT89C51?
6) Чим визначається тривалість однієї фази МК AT89C51?
7) Опишіть підключення кварцового резонатора до генератора МК AT89C51.
8) Опишіть формат та призначення розрядів регістра PCON.
9) Опишіть режими IDLE та STOP мікроконтролера типу C051Fxxx.
10) Опишіть стан мікроконтролера AT89C51 після скидання.
11) Опишіть структуру підсистеми скидання МК CIP–51.
12) Опишіть особливості скидання від вартового таймеру.
13) Опишіть особливості скидання від компаратора 0.
14) Опишіть особливості скидання від зовнішнього сигналу CNVSTR.

9 АНАЛОГО–ЦИФРОВИЙ ПЕРЕТВОРЮВАЧ

9.1 Загальні відомості

Мікроконтролери сімейства SYGNAL мають один або декілька аналого–цифрових перетворювача, наприклад, 12–розрядний (ADC0) та 8–розрядний (ADC1).

9.2 АЦП ADC0

9.2.1 Опис функціональної схеми

Розглянемо архітектуру 12–розрядного АЦП: ADC0, який входить, наприклад, до складу сімейства мікроконтролерів C8051F0xx.

Перетворювач ADC0 – це 12–бітний аналого–цифровий перетворювач, що забезпечує максимальну частоту вибірки до 100 КГц. Функціональну схему перетворювача наведено на рисунку 9.1.

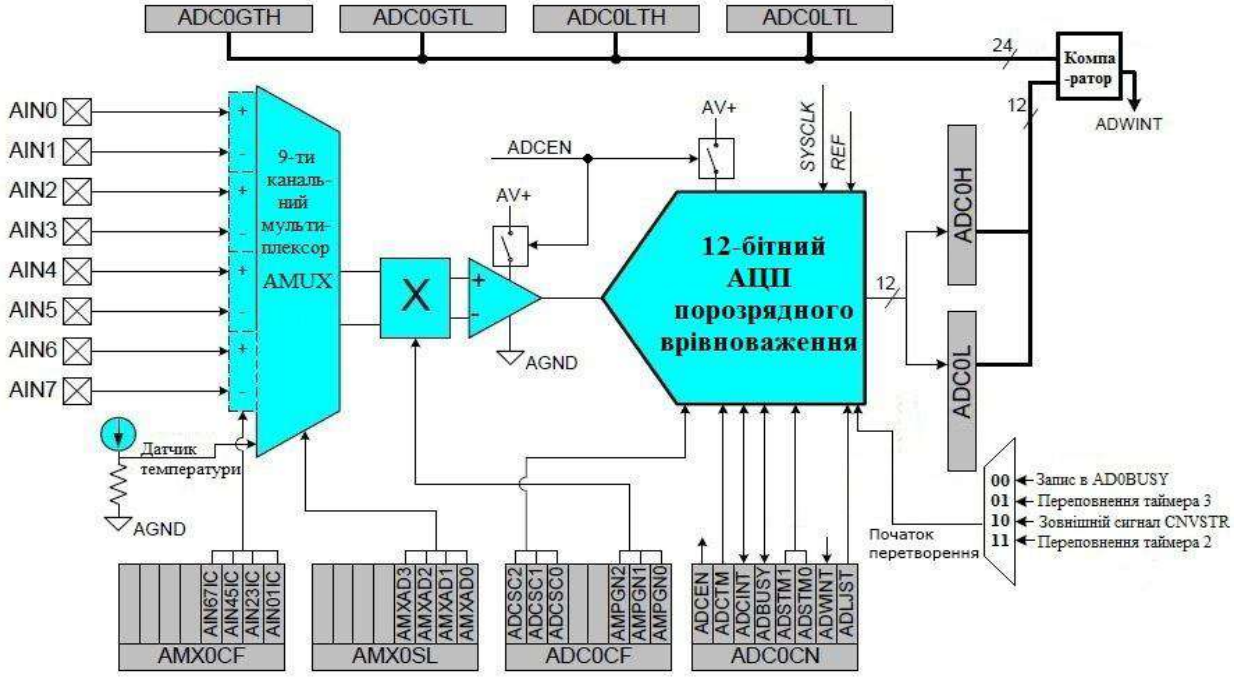


Рисунок 9.1 – Функціональна схема аналого–цифрового перетворювача ADC0

Центральним вузлом пристрою є 12–бітний АЦП послідовного наближення (SAR – Successive–Approximation –Register ADC0), вихідний результат якого записується у регістри ADC0H (ADC0 Data Word MSB Register) та ADC0L (ADC0 Data Word LSB Register). Аналоговий сигнал подається на вхід АЦП через 9–канальний аналоговий мультиплексор та підсилювач з програмованим коефіцієнтом підсилення (PGA0 – Programmable Gain Amplifier) із вбудованою системою вибірки–зберігання (track–and–hold), яка запам`ятовує поточне значення вхідного аналогового сигналу в момент часової вибірки.

Окрім цього, АЦП забезпечений так званим віконним детектором (Window Detector), що дозволяє апаратно визначити, чи знаходиться результат перетворення у межах «вікна», утвореного двома границями. До складу віконного детектора входять цифровий компаратор та дві пари регістрів границь: верхньої ADC0GTH (ADC0 Greater Than Data High Byte Register), ADC0GTL (ADC0 Greater Than Data Low Byte Register) та нижньої ADC0LTH (ADC0 Less Than Data High Byte Register), ADC0LTL (ADC0 Less Than Data Low Byte Register). Завантажуючи певні значення в ці регістри, можна змусити цифровий компаратор генерувати прапорець запиту на переривання від віконного детектора AD0WINT (ADC0CN.1), коли результат перетворення знаходиться як у межах, так і за межами «вікна», утвореного двома границями.

Також, до складу ADC0 входить вбудований аналоговий датчик температури. Залежність вихідної напруги датчика від температури має лінійний характер (рисунок 9.2).

Для забезпечення роботи перетворювача на його керуючі входи поступають імпульси системної тактової частоти SYSCLK та опорна напруга REF. Підсистеми АЦП: ADC0, PGA та пристрій вибірки–зберігання працюють в активному режимі лише тоді, коли біт ADC0EN регістра керування ADC0 (ADC0CN, рисунок 9.7) встановлено в 1 та знаходяться у режимі зниженого енергоспоживання, коли цей біт скинуто в 0.

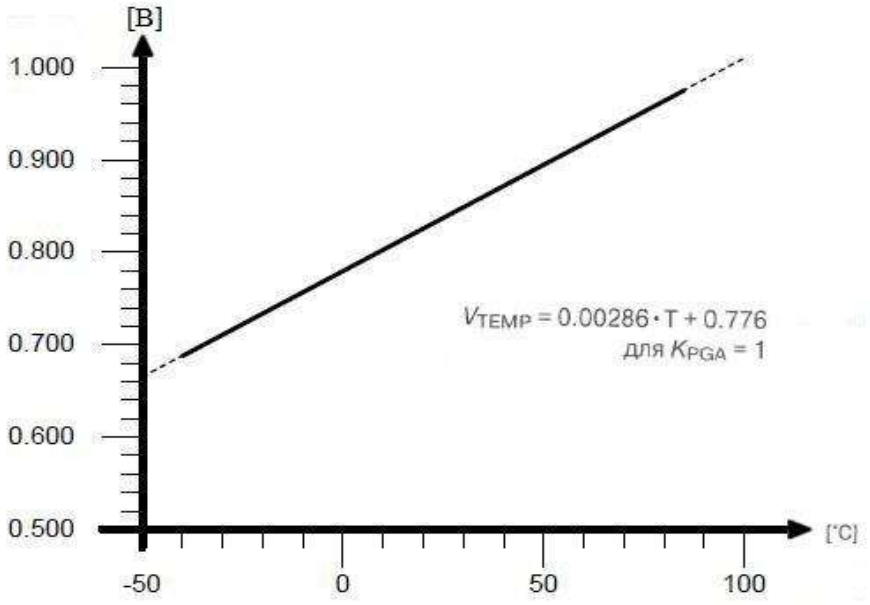


Рисунок 9.2 – Передатна характеристика температурного датчика

9.2.2 Аналоговий мультиплексор та програмований підсилювач

Аналоговий мультиплексор AMUX (рисунок 9.1) має 9 аналогових входів, 8 з яких під'єднано до зовнішніх виводів МК – аналоговим входам AIN0…AIN7 (Analog Input), а 9–й під'єднано до датчика температури. За допомогою регістра конфігурації аналогового мультиплексора AMX0CF (Configuration Register) (рисунок 9.4) окремі одиночні (SE – Single Ended) аналогові входи можуть бути об'єднано в диференціальні (DIFF–Differential) пари, як це показано на рисунку 9.1. За допомогою другого керуючого регістра AMX0SL (рисунок 9.5) (Channel Select Register) можна обирати поточний канал для вимірів (підрозділ 9.2.4). Ця властивість допомагає вибирати найбільш прийнятну технологію для кожного з вхідних каналів чи навіть запрограмувати зміну режимів «на льоту». За замовчуванням після скидання всі канали конфігуруються як одиночні.

Регістр конфігурації перетворювача ADC0CF задає коефіцієнт підсилення програмного підсилювача PGA0 та частоту синхронізації перетворювача. Вибір коефіцієнта підсилення (1, 2, 4, 8, 16 або 0.5) виконується за допомогою трьох молодших бітів цього регістра: AMP0GN[2:0] (ADC0 Internal Amplifier

Gain – PGA). Після скидання МК стан цих бітів 000, що відповідає коефіцієнту підсилення 1.

Три старших біти цього регістра (біти 7…5, ADCSC2…0 (ADC SAR Conversion Clock Period)) програмують вхідну частоту для АЦП, тобто опосередковано визначають час перетворення. Код, який встановлюється у цих бітах, задає коефіцієнт ділення подільника, на вхід якого поступає системна тактова частота. Нижче наведено вплив цих бітів на вхідну частоту АЦП: CLK_{SAR0} :

000: Частота дискретизації CLK_{SAR0} = системній тактовій частоті ;

001: Частота дискретизації CLK_{SAR0} = 1/2 системної тактової частоти;

010: Частота дискретизації CLK_{SAR0} = 1/4 системної тактової частоти;

011: Частота дискретизації CLK_{SAR0} = 1/8 системної тактової частоти;

1xx: Частота дискретизації CLK_{SAR0} = 1/16 системної тактової частоти;

(Зауваження: Частота дискретизації має бути ≤ 2MHz).

9.2.3 Режими роботи ADC0 за часовими діаграмами

На рисунку 9.3 наведено часові діаграми роботи ADC0.

Керування роботою перетворювача здійснюється за допомогою бітів та прапорців регістра керування ADC0CN (ADC0 Control Register). Встановлення у 1 біта дозволу роботи перетворювача AD0EN (ADC0CN.7) (ADC0 Enable Bit) переводить його в робочий режим. Вибір джерела, що викликає запуск перетворювача, здійснюється за допомогою бітів режиму запуска ADSTM[1:0] (ADC0CN [3:2]) (ADC0 Start of Conversion Mode Select).

В залежності від стану цих бітів може бути обране відповідне джерело (рисунок 9.1):

– запис 1 в біт зайнятості перетворювача ADBUSY (ADC0CN.4) (ADC0 BUSY Bit);
– переповнення таймера 3;
– наростаючий фронт сигналу на зовнішньому вході запуску перетворювача CNVSTR (Start of Conversion);
– переповнення таймера 2.

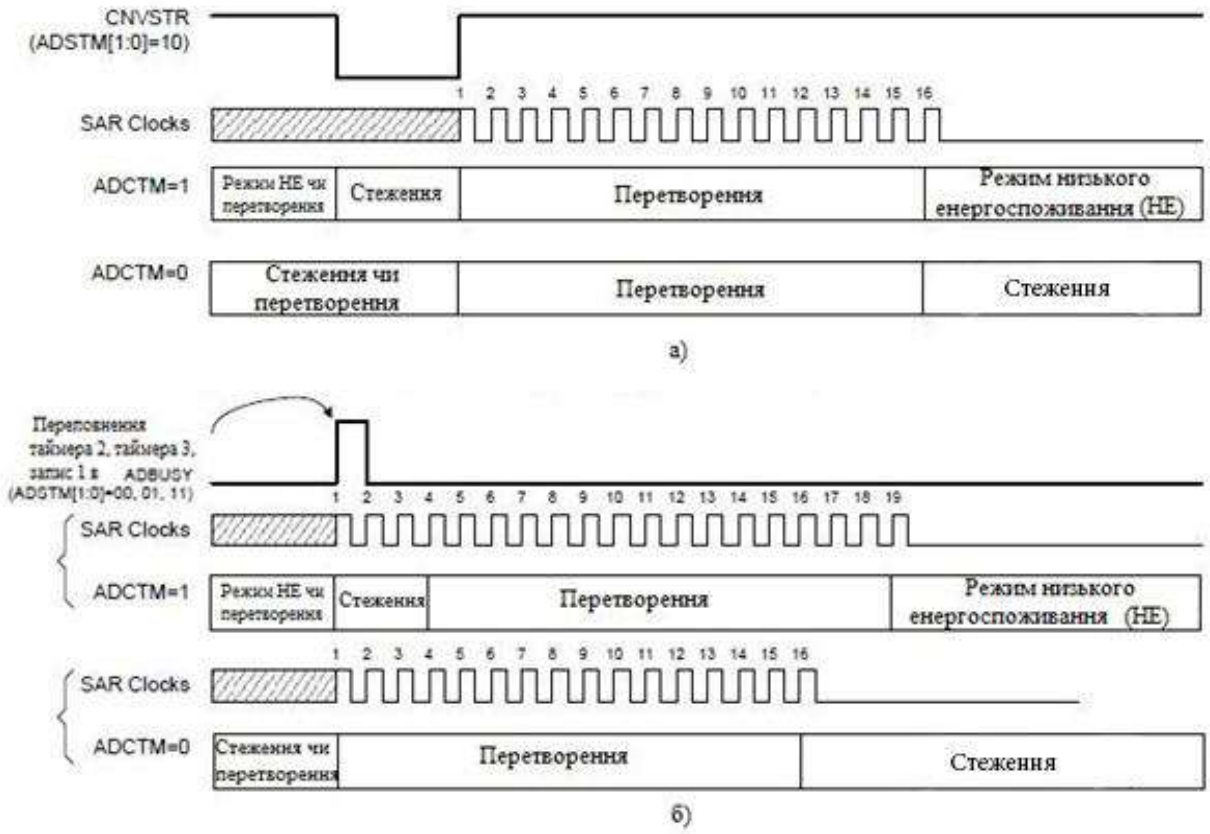


Рисунок 9.3 – Часові діаграми роботи АЦП ADC0:
а – запуск від зовнішнього джерела; б – запуск від внутрішнього джерела

Під час процесу перетворення біт зайнятості ADBUSY зберігає стан логічної 1, а в кінці перетворення скидається, одночасно встановлюючи прапорець запиту на переривання від перетворювача за завершенням перетворення ADCINT (ADC0CN.5) (ADC0 Conversion Complete Interrupt Flag).

В цей час результат перетворення записується в парі регістрів ADC0H та ADC0L. 12–бітний результат перетворення розміщується в цих регістрах зміщеним вправо (ADLJST = 0) чи вліво (ADLJST = 1), в залежності від стану біта вибору вирівнювання вліво ADLJST (ADC0CN.0) (ADC0 Left Justify Select).

Біт режиму стеження ADCTM (ADC0CN.6) (ADC0 Track Mode Bit) керує режимом стеження. Коли цей біт знаходиться у вихідному нульовому стані, за час, поки перетворювач чекає запуску, відбувається стеження за вхідною напругою. У цьому випадку процес перетворення починається відразу після активації будь–якого із джерел запуску та триває 15 періодів тактової частоти SAR Clocks, як це показано на рисунку 9.3.

Якщо ADCTM = 1, то перетворювач працює у режимі низького електроживлення (HE). В цьому випадку після запуску перед перетворенням деякий період ведеться стеження. Цей період у випадку використання сигналу зовнішнього запуску CNVSTR має тривалість, яка дорівнює тривалості низького рівня цього сигналу (рисунок 9.3, а), а у випадку використання зовнішніх джерел запуску триває 3 періоди тактової частоти SAR Clocks (рисунок 9.3, б). Цей режим може бути використано для перемикання аналогового мультиплексора та зміни коефіцієнта підсилення підсилювача: як правило, трьох періодів тактової частоти достатньо для затухання перехідних процесів на вході перетворювача, викликаних перемиканням.

Важливою особливістю схеми ADC0 являється присутність програмованого «віконного детектора» ADC0. Віконний детектор має 2 регістри, що програмуються користувачем, в які він може записати дві 12–бітні (в загальному випадку) величини – дві границі: регістри верхньої границі – ADC0GHT та ADC0GTL: регістри нижньої границі – ADC0LTH та ADC0LTL. Інтервал величин (кодів) між цими границями визначає «вікно» вихідних кодів. Вузол віконного детектора постійно відстежує вихідний код ADC та зрівнює

його із заданими в регістрах величинами порогів, і якщо вихідний код виходить за межі заданих порогів (інакше кажучи, за межі «вікна»), вузол може інформувати про це систему. Інформування може бути здійснене шляхом подачі відповідного перериванням чи програмним опитом прапорця ADWINT в регістрі ADC0CN.

9.2.4 Програмування ADC0

При програмуванні ADC0 використовуються наступні регістри:

– AMX0CF;
– AMX0SL;
– ADC0CF;
– ADC0CN;
– ADC0GTH, ADC0GTL;
– ADC0LTH, ADC0LTL;
– ADC0H, ADC0L.

9.2.4.1 Регістр конфігурації аналогового мультиплексора AMX0CF

Нижче наведено формат (рисунок 9.4) та опис окремих розрядів регістра AMX0CF.

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Значення при скиданні
-	-	-	-	AIN67IC	AIN45IC	AIN23IC	AIN01IC	00000000
Біт 7	Біт 6	Біт 5	Біт 4	Біт 3	Біт 2	Біт 1	Біт 0	Адреса SFR 0xBA

Рисунок 9.4 – Регістр конфігурації аналогового мультиплексора AMX0CF (AMUX Configuration Register)

Біт 7…4: Не використовується. Зчитування = 0000b; Запис = не впливає;

Біт 3: AIN67IC: AIN6, AIN7. Вхідна пара аналогових входів;

0: AIN6 та AIN7 незалежні односторонні аналогові входи;

1: AIN6 та AIN7: диференціальні аналогові входи: + та – (відповідно);

Біт 2: AIN45IC: AIN4, AIN5. Вхідна пара аналогових входів;

0: AIN4 та AIN5 незалежні односторонні аналогові входи;

1: AIN4 та AIN5: диференціальні аналогові входи: + та – (відповідно);

Біт 1: AIN23IC: AIN2, AIN3. Вхідна пара аналогових входів;

0: AIN2 та AIN3 незалежні односторонні аналогові входи;

1: AIN2 та AIN3: диференціальні аналогові входи: + та – (відповідно);

Біт 0: AIN01IC: AIN0, AIN1. Вхідна пара аналогових входів;

0: AIN0 та AIN1 незалежні односторонні аналогові входи;

1: AIN0 та AIN1: диференціальні аналогові входи: + та – (відповідно).

9.2.4.2 Регістр вибору каналу аналогового мультиплексора AMX0SL

Нижче наведено формат (рисунок 9.5) та опис окремих розрядів регістра AMX0SL.

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Значення при скиданні
-	-	-	-	AMXAD3	AMXAD2	AMXAD1	AMXAD0	00000000
Біт 7	**Біт 6**	**Біт 5**	**Біт 4**	**Біт 3**	**Біт 2**	**Біт 1**	**Біт 0**	Адреса SFR 0xBB

Рисунок 9.5 – Регістр вибору каналу аналогового мультиплексора AMX0SL (AMUX Channel Select Register)

Біт 7…4: Не використовується. Зчитування = 0000b; Запис = не впливає;

Біт 3…0: AMXAD3…0: Біти адреси AMUX:

0000…1111: Входи ADC обираються за таблицею 9.1.

Таблиця 9.1 – Вибір каналів аналогового мультиплексора

AMX0CF 0-3 \ AMX0SL 0-3	0000	0001	0010	0011	0100	0101	0110	0111	1xxx
0000	AIN0	AIN1	AIN2	AIN3	AIN4	AIN5	AIN6	AIN7	Датчик температури
0001	+(AIN0) -(AIN1)		AIN2	AIN3	AIN4	AIN5	AIN6	AIN7	Датчик температури
0010	AIN0	AIN1	+(AIN2) -(AIN3)		AIN4	AIN5	AIN6	AIN7	Датчик температури
0011	+(AIN0) -(AIN1)		+(AIN2) -(AIN3)		AIN4	AIN5	AIN6	AIN7	Датчик температури
0100	AIN0	AIN1	AIN2	AIN3	+(AIN4) -(AIN5)		AIN6	AIN7	Датчик температури
0101	+(AIN0) -(AIN1)		AIN2	AIN3	+(AIN4) -(AIN5)		AIN6	AIN7	Датчик температури
0110	AIN0	AIN1	+(AIN2) -(AIN3)		+(AIN4) -(AIN5)		AIN6	AIN7	Датчик температури
0111	+(AIN0) -(AIN1)		+(AIN2) -(AIN3)		+(AIN4) -(AIN5)		AIN6	AIN7	Датчик температури
1000	AIN0	AIN1	AIN2	AIN3	AIN4	AIN5	+(AIN6) -(AIN7)		Датчик температури
1001	+(AIN0) -(AIN1)		AIN2	AIN3	AIN4	AIN5	+(AIN6) -(AIN7)		Датчик температури
1010	AIN0	AIN1	+(AIN2) -(AIN3)		AIN4	AIN5	+(AIN6) -(AIN7)		Датчик температури
1011	+(AIN0) -(AIN1)		+(AIN2) -(AIN3)		AIN4	AIN5	+(AIN6) -(AIN7)		Датчик температури
1100	AIN0	AIN1	AIN2	AIN3	+(AIN4) -(AIN5)		+(AIN6) -(AIN7)		Датчик температури
1101	+(AIN0) -(AIN1)		AIN2	AIN3	+(AIN4) -(AIN5)		+(AIN6) -(AIN7)		Датчик температури
1110	AIN0	AIN1	+(AIN2) -(AIN3)		+(AIN4) -(AIN5)		+(AIN6) -(AIN7)		Датчик температури
1111	+(AIN0) -(AIN1)		+(AIN2) -(AIN3)		+(AIN4) -(AIN5)		+(AIN6) -(AIN7)		Датчик температури

9.2.4.3 Регістр конфігурації аналого–цифрового перетворювача ADC0CF

Нижче наведено формат (рисунок 9.6) та опис окремих розрядів регістра ADC0CF.

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	Значення при скиданні 01100000 Адреса SFR 0xBC
ADCSC2	ADCSC1	ADCSC0	-	-	AMPGN2	AMPGN1	AMPGN0	
Біт 7	Біт 6	Біт 5	Біт 4	Біт 3	Біт 2	Біт 1	Біт 0	

Рисунок 9.6 – Регістр конфігурації аналого–цифрового перетворювача ADC0CF (ADC Configuration Register)

Біти 7…5: ADCSC2…0: ADC SAR Conversion Clock Period.

Ці біти програмують вхідну частоту для АЦП, тобто опосередковано визначають час перетворення. Код, який встановлюється у цих бітах, задає

коефіцієнт ділення подільника, на вхід якого поступає системна тактова частота. Нижче наведено вплив цих бітів на вхідну частоту АЦП: CLK_{SAR0} :

000: Частота дискретизації CLK_{SAR0} = системній тактовій частоті ;

001: Частота дискретизації CLK_{SAR0} = 1/2 системної тактової частоти;

010: Частота дискретизації CLK_{SAR0} = 1/4 системної тактової частоти;

011: Частота дискретизації CLK_{SAR0} = 1/8 системної тактової частоти;

1xx: Частота дискретизації CLK_{SAR0} = 1/16 системної тактової частоти;

(Зауваження: Частота дискретизації має бути ≤ 2MHz).

Біти 4…3: Не використовуються. Зчитування = 00b; Запис = не впливає

Біти 2…0: AMPGN2…0. Ці біти визначають коефіцієнт підсилення вхідного підсилювача АЦП:

000: Коефіцієнт підсилення = 1;

001: Коефіцієнт підсилення = 2;

010: Коефіцієнт підсилення = 4;

011: Коефіцієнт підсилення = 8;

10x: Коефіцієнт підсилення = 16;

11x: Коефіцієнт підсилення = 0.5.

В деяких мікроконтролерах сімейства SYGNAL, наприклад, C8051F040/1/2/3 для програмування вхідної частоти для АЦП використовуються п`ять старших розрядів регістра: біти 7…3: AD0SC4…0– біти встановлення періоду сигналу дискретизації АЦП0.

В цьому випадку частота сигналу дискретизації АЦП0 визначається частотою системного тактового сигналу у відповідності з наступним рівнянням:

AD0SC = (SYSCLK / CLKSAR0) – 1,

де AD0SC – 5–розрядне значення, що задається бітами AD0SC4…0,

CLKSAR0 – необхідна частота сигналу дискретизації АЦП0.

9.2.4.4 Регістр керування АЦП ADC0CN

Нижче наведено формат (рисунок 9.7) та опис окремих розрядів регістра ADC0CN.

Назва регістра:	ADC0CN - ADC Control Register		
SFR-адреса	0xE8	Значення після скидання:	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
ADCEN	ADCTM	ADCINT	ADBUSY	ADSTM1	ADSTM0	ADWINT	ADLIST
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Рисунок 9.7 – ADC0CN: регістр керування ADC0 (ADC0 Start of Conversion Mode Select)

Біт 7: ADCEN: біт вмикання ADC0.

0: ADC0 вимкнено. ADC0 в режимі низького електроживлення.

1: ADC0 ввімкнено. АЦП готовий для перетворення даних.

Біт 6: ADCTM: біт режиму стеження ADC0.

0: Якщо ADC0 ввімкнено, стеження ведеться на протязі всього періоду до початку перетворення.

1: Стеження керується бітами ADSTM1…0:

00: Стеження починається із запису 1 в ADBUSY і триває 3 періоди тактової частоти SAR Clock;

01: Стеження починається з переповнення таймера 3 і триває 3 періоди тактової частоти SAR Clocks;

10: ADC0 веде стеження лише коли матричний комутатор назначає наростаючий вхід сигналу на відповідний вхід CNVSTR (Start of Conversion;

11: Стеження починається із переповнення таймера 2 і триває 3 періоди тактової частоти SAR Clock.

Біт 5: ADCINT: прапорець переривання від завершення перетворення даних (має скидатися програмно).

0: ADC0 не завершив перетворення з того часу, коли востаннє було скинуто цей біт.

1: ADC0 завершив процес перетворення.

Біт 4: ADBUSY: біт зайнятості.

Зчитування

0: ADC0 завершив перетворення чи після скидання не було перетворень. Спадаючий фронт ADBUSY генерує переривання, якщо воно дозволене.

1: ADC0 перетворює дані.

Запис

0: ігнорується.

1: ADC0 розпочинає перетворення, якщо ADSTM1…0 = 00b.

Біти 3…2 : ADSTM1…0: біти програмування режиму запуску перетворення ADC0:

00: Перетворення розпочинається після запису 1 в ADBUSY;

01: Перетворення розпочинається після кожного переповнення таймера 3;

10: Перетворення розпочинається після кожного перепаду рівня з низького в високий на вході CNVSTR;

11: Перетворення розпочинається після кожного переповнення таймера 2.

Біт 1: ADWINT: прапорець переривання, якщо вихідний код виходить за межі заданих порогів (інакше кажучи, за межі «вікна»), має скидатися програмно.

0: Сигнал перетворення знаходиться в межах вікна.

1: Сигнал перетворення знаходиться за межами вікна.

Біт 0: ADLJLIST: біт вирівнювання результату перетворення за лівим краєм.

0: Дані в ADC0H: ADC0L зміщені вправо.

1: Дані в ADC0H: ADC0L зміщені вліво.

9.2.4.5 Регістри верхньої границі вікна ADC0GTH, ADC0GTL

Нижче наведено формат (рисунок 9.8, 9.9) та опис окремих розрядів пари регістрів верхньої границі вікна ADC0GTH, ADC0GTL

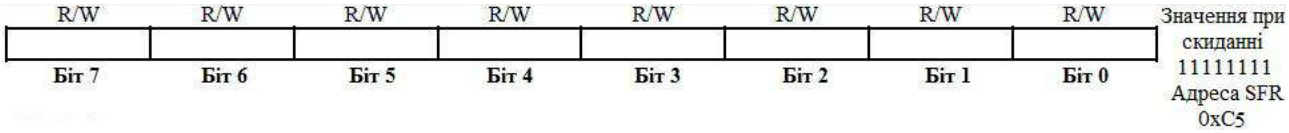


Рисунок 9.8 – Регістр старшого байта верхньої границі вікна ADC0GTH

Біти 7…0: Старший байт верхньої границі вікна ADC Greater–Than Data High Byte Register.

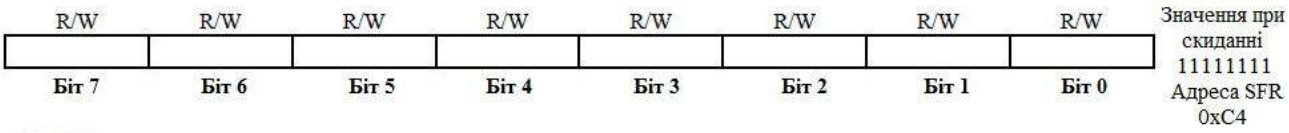


Рисунок 9.9 – Регістр молодшого байта верхньої границі вікна ADC0GTL

Біти 7…0: Молодший байт верхньої границі вікна ADC Greater–Than Data Low Byte Register.

9.2.4.6 Регістр нижньої границі вікна ADC0LTH, ADC0LTL

Нижче наведено формат (рисунок 9.10, 9.11) та опис окремих розрядів регістрів нижньої границі вікна ADC0LTH, ADC0LTL

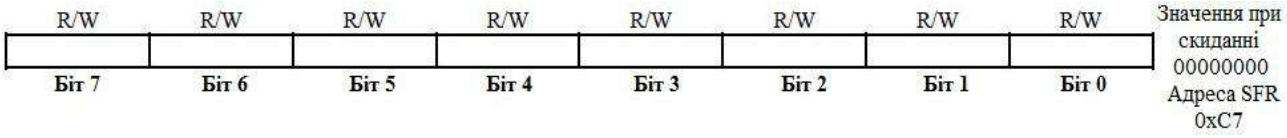


Рисунок 9.10 – Регістр старшого байта нижньої границі вікна ADC0LTH

Біти 7…0: Старший байт нижньої границі вікна ADC Less–Than Data High Byte Register.

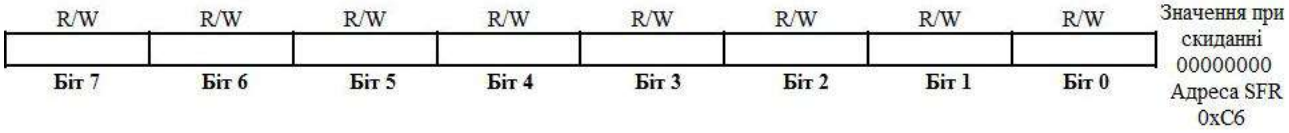


Рисунок 9.11 – Регістр молодшого байта нижньої границі вікна ADC0LTL

Біти 7…0: Молодший байт нижньої границі вікна ADC Less–Than Data Low Byte Register.

9.2.4.7 Регістри ADC0H (ADC0 Data Word MSB Register) та ADC0L (ADC0 Data Word LSB Register)

Нижче наведено формат (рисунок 9.12, 9.13) та опис окремих розрядів регістрів ADC0H, ADC0L

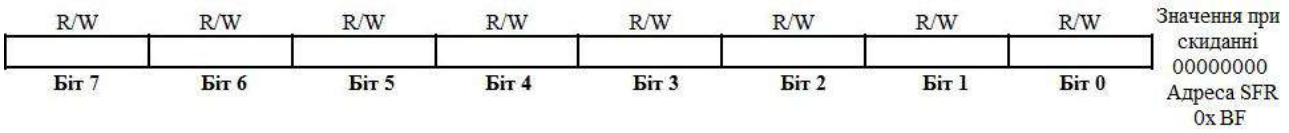


Рисунок 9.12 – ADC0H (ADC0 Data Word MSB Register)

Біти 7…0: Біти ADC (Data Words Bits).

Старший байт вихідних даних ADC.

Для ADLJST = 1: Старші 8 біт 12–бітного вихідного коду ADC.

Для ADLJST = 0: Біти 7…4 – розширення біта 3 зі знаком. Біти 3…0 – старші 4 біти 12–бітного вихідного коду ADC.

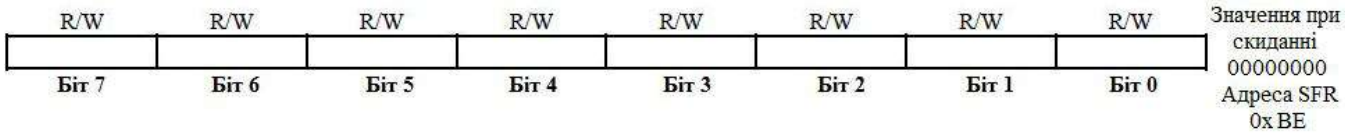


Рисунок 9.13 – ADC0L (ADC0 Data Word LSB Register)

Біти 7…0: Біти ADC (Data Words Bits).

Молодший байт вихідних даних ADC.

Для ADLJST = 1: Біти 7…4 – чотири молодших біти 12–бітного вихідного коду ADC. Біти 3..0 завжди зчитуються як 0.

Для ADLJST = 0: Біти 7..0 – молодші 8 біт 12–бітного вихідного коду ADC.

9.2.5 Результат перетворення АЦП

12–розрядний результат перетворення АЦП0 виходить наступним чином: якщо AD0LJST = 0: ADC0H[3:0]: ADC0L[7:0], ADC0H[7:4] = 0000b (у випадку перетворення диференціального сигналу біти ADC0H[7:4] будуть знаковим розширенням біта ADC0H.3. Якщо AD0LJST = 1 у випадку одиночного сигналу: ADC0H[7:0]: ADC0L[7:4], , ADC0L[3:0] = 0000b.

9.2.5.1 Перетворення в одиночному режимі

Коли вхід AIN0 працює в одиночному режимі (AMX0CF=0x00, AMX0SL=0x00), порядок запису результату перетворення наведено в таблиці 9.2.

Таблиця 9.2 – Зв'язок між вхідними сигналами та вихідними кодами в одиночному режимі

AIN0 – AGND (Вольти)	ADC0H:ADC0L (AD0LJST = 0)	ADC0H:ADC0L (AD0LJST = 1)
V_{REF} * (4095/4096)	0x0FFF	0xFFF0
$V_{REF}/2$	0x0800	0x8000
V_{REF} * (2047/4096)	0x07FF	0x7FF0
0	0x0000	0x0000

9.2.5.2 Перетворення у диференціальному режимі

Порядок запису результату перетворення, входи AIN0…AIN1 працюють в диференціальному режимі (AMX0CF=0x01, AMX0SL=0x00), наведено в таблиці 9.3.

9.2.5.3 Зв'язок між вхідною напругою та вихідним кодом

Якщо біт AD0LJST = 0, вихідний код АЦП зв'язаний із вхідною напругою U_{in} виразом:

$$\text{Код АЦП} = U_{in} \times (K_П / U_{REF}) \times 2^n, \text{ де}$$

U_{REF} – значення опорної напруги;

$K_П$ – коефіцієнт підсилення;

n = 12, якщо входи працюють в одиночному режимі;

n = 11, якщо входи працюють в диференціальному режимі.

Таблиця 9.3 – Зв’язок між вхідними сигналами та вихідними кодами в диференціальному режимі

AIN0 – AGND (Вольти)	ADC0H:ADC0L (AD0LJST = 0)	ADC0H:ADC0L (AD0LJST = 1)
V_{REF} * (2047/2048)	0x07FF	0x7FF0
VREF/2	0x0400	0x4000
V_{REF} * (1/2048)	0x0001	0x0010
0	0x0000	0x0000
– V_{REF} * (1/2048)	0xFFFF(–1d)	0xFFF0
– V_{REF}/2	0xFC00(–1024d)	0xC000
– V_{REF}	0xF800(–2048d)	0x8000

9.3 Перетворювач ADC1

9.3.1 Опис функціональної схеми

Розглянемо архітектуру 8–розрядного АЦП: ADC1, який входить, наприклад, до складу сімейства мікроконтролерів C8051F02x.

Перетворювач ADC1 являє собою 8–бітний АЦП, що забезпечує максимальну частоту вибірки до 500 КГц. Функціональну схему ADC1 наведено на рисунку 9.14. Центральним вузлом цього пристрою є 8–бітний АЦП порозрядного врівноваження, вихідний результат якого записується в регістр ADC1 (ADC1 Data Word Register). Аналоговий сигнал подається на вхід АЦП через 8–канальний аналоговий мультиплескор (Analog Multiplexor – AMUX1) та підсилювач з програмованим коефіцієнтом підсилення (Programmable Gain Amplifier – PGA1) із вбудованою системою вибірки–зберігання (track–and–hold), яка забезпечує фіксацію перетворюваного сигналу під час перетворення. Для забезпечення роботи перетворювача на його

службові входи поступають імпульси системної тактової частоти SYSCLK та опорна напруга REF.

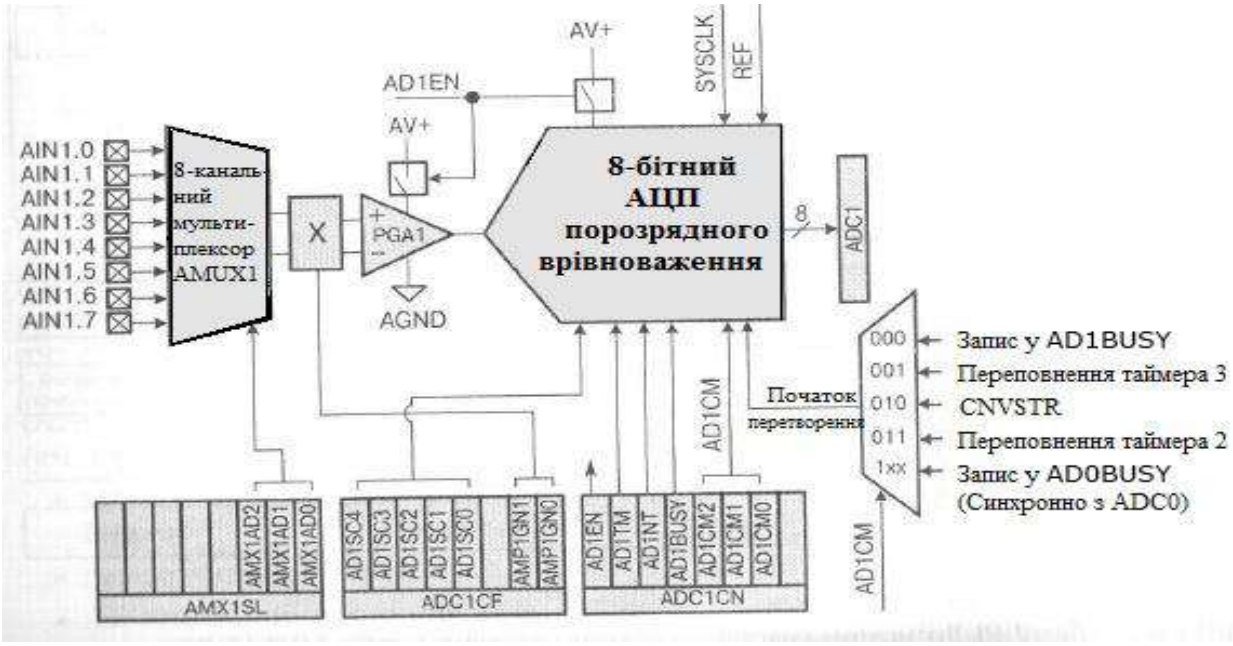


Рисунок 9.14 – Функціональна схема аналого–цифрового перетворювача ADC1

9.3.2 Аналоговий мультиплексор та програмований підсилювач

Аналоговий мультиплескор AMUX1 має 8 аналогових входів AIN1.0...AIN1.7, які підключено до виводів порту 1 МК: P1.0…P1.7. Виводи порту 1, які мають використовуватися як входи ADC1, необхідно налаштувати як аналогові, записавши 0 у відповідні біти регістра вибору режиму порту 1 P1MDIN (Port 1 Input Mode Register). Виводи, які налаштовано як аналогові, пропускаються при призначенні ліній портів для інших периферійних вузлів за допомогою матричного комутатора.

За допомогою керуючого регістра AMX1SL (AMUX1 Channel Select Register) можна обрати поточний канал для вимірів. Для цього достатньо записати у молодші 3 біти цього регістра: AMX1AD2…AMX1AD0 (AMX1SL[2:0]) (AMX1 Adress Bits) двійковий код відповідного каналу.

Регістр конфігурації перетворювача ADC1CF (ADC1 Configuration Register) задає коефіцієнт підсилення підсилювача PGA1 та частоту

синхронізації перетворювача. Вибір коефіцієнта підсилення (0.5, 1, 2 чи 4) здійснюється за допомогою 2–молодших бітів цього регістра: AMP1GN1… AMP1GN0 (ADC1 Internel Amplifier Gain – PGA). Після скидання МК значення цих бітів дорівнює 00, що відповідає коефіцієнту підсилення 0.5. Значення п’яти старших бітів цього регістра: AD1SC4… AD1SC0 (ADC1 SAR Conversion Clock Period Bits) для бажаної частоти синхронізації перетворювача CLK_{SAR1} (максимальне значення 6 МГц) можна вирахувати за формулою

$$AD1SC\ [4:0] = SYSCLK/\ CLK_{SAR1} - 1.$$

9.3.3 Робота ADC1 за часовими діаграмами

Часові діаграми роботи АЦП наведено на рисунку 9.15.

Керування роботою перетворювача здійснюється за допомогою бітів та прапорців регістра керування ADC1CN (ADC1 Control Register). Встановлення в 1 біта дозволу роботи перетворювача AD1EN (ADC1CN.7) (ADC1 Enable Bit) переводить його в робочий режим. Вибір джерела, що запускає перетворювач, здійснюється за допомогою бітів режиму запуску AD1CM2… AD1CM0 (ADC1CN.[3:1]) ADC1 Start of Conversion Mode Select). В залежності від стану цих бітів може бути обрано одне з наступних джерел:

- запис 1 в біт зайнятості AD1BUSY (ADC1CN.4);
- переповнення таймера 3;
- наростаючий фронт сигналу на зовнішньому вході запуску перетворювача CNVSTR, який призначається за допомогою матричного комутатора на одну з ліній введення/виведення МК;
- переповнення таймера 2;
- запис 1 в біт зайнятості AD0BUSY (ADC0CN.4) – запуск синхронно з ADC0.

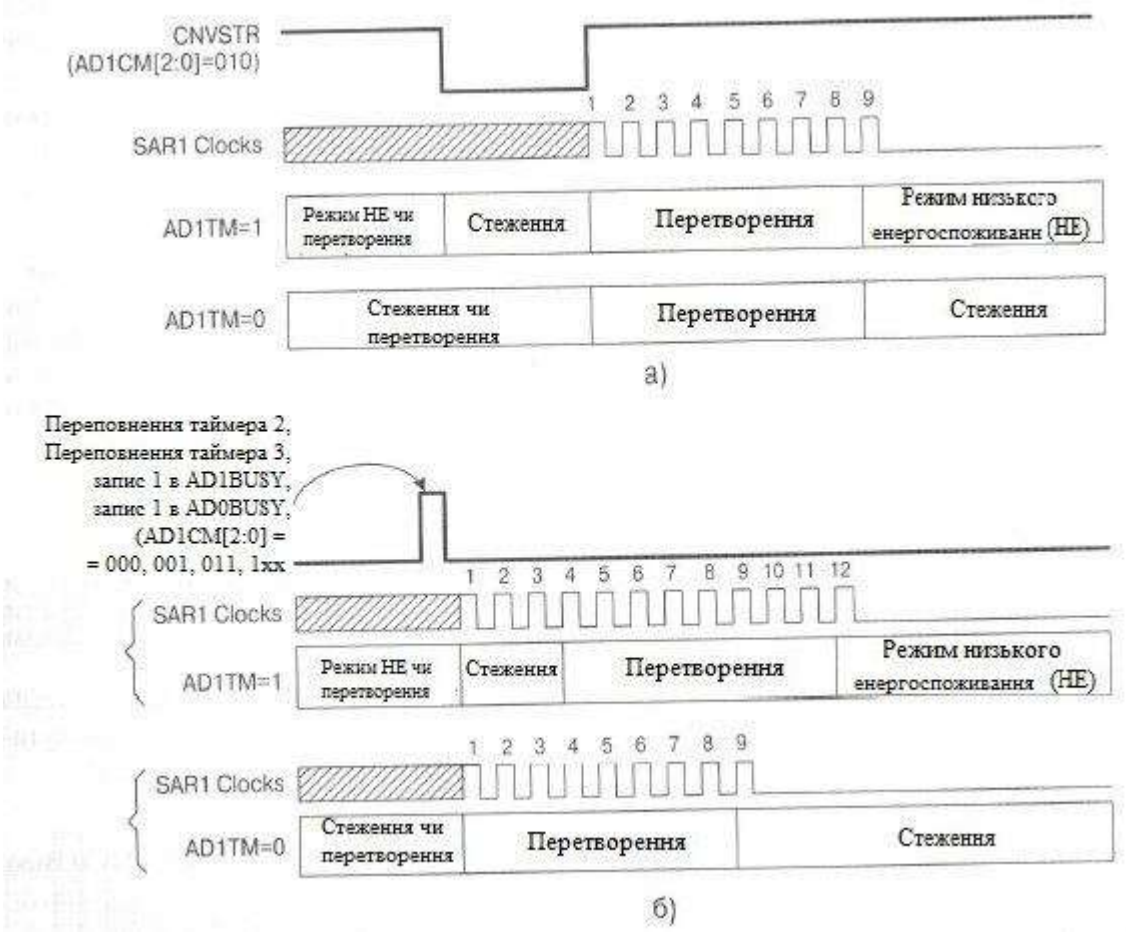


Рисунок 9.15 – Часові діаграми роботи АЦП ADC1:

а – запуск від зовнішнього джерела;

б – запуск від внутрішнього джерела

Під час процесу перетворення біт зайнятості AD1BUSY зберігає стан логічної 1, а в кінці перетворення скидається, одночасно встановлюючи прапорець запиту на переривання від перетворювача за завершенням перетворення AD1INT (ADC1CN.5) (ADC1 Conversion Complete Interrupt Flag). В цей час результат перетворення записується в регістрі ADC1.

Біт режиму стеження AD1TM (ADC1CN.6) (ADC1 Track Mode Bit) керує режимом стеження. Коли цей біт знаходиться у вихідному нульовому стані, за час, доки перетворювач чекає запуску, відбувається стеження за вхідною напругою. У цьому випадку процес перетворення починається відразу після активації будь–якого із джерел запуску та триває 8 періодів тактової частоти

SAR Clocks, як це показано на рисунку 9.15. Якщо AD1TM = 1, то перетворювач працює у режимі низького енергоспоживання (НЕ). В цьому випадку перед перетворенням деякий період ведеться стеження, який у випадку використання сигналу зовнішнього запуску CNVSTR має тривалість, яка дорівнює тривалості низького рівня цього сигналу (рисунок 9.15, а), у випадку використання зовнішніх джерел запуску триває 3 періоди тактової частоти SAR Clocks (рисунок 9.15, б). Цей режим може бути використано для перемикання аналогового мультиплексора та зміни коефіцієнта підсилення підсилювача: як правило, трьох періодів тактової частоти достатньо для затухання перехідних процесів на вході перетворювача, викликаних перемиканням.

9.3.4 Програмування ADC1

При програмуванні ADC1 використовуються наступні регістри:

– ADC1CF;
– AMX1SL;
– ADC1CN;
– ADC1.

9.3.4.1 Регістр конфігурації АЦП ADC1CF

Нижче наведено формат (рисунок 9.16) та опис окремих розрядів регістра ADC1CF.

Назва регістра	AMX1CF - AMUX1 Configuration Register							
SFR адреса:	0xAB			Значення після скидання:		11111000b (0xF8)		
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
AD1SC4	AD1SC3	AD1SC2	AD1SC1	AD1SC0	-	AMP1GN 1	AMP1GN 0	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	

Рисунок 9.16 – Регістр конфігурації АЦП ADC1CF

Біти 7…3: ADISC4…0 – визначають період перетворення за приблизною формулою:

AD1SC = SYSCLK/ CLK_{SAR1} – 1, де

AD1SC – код періоду перетворення аналого–цифрового перетворювача ADC1;

SYSCLK – системна тактова частота;

CLK_{SAR1} – частота аналого–цифрового перетворювача ADC1.

Біт 2: не використовується; при зчитуванні повертає 0b, при запису значення ігнорується;

Біти 1…0: AMP1GN1…AMP1GN0 – визначають коефіцієнт підсилення вхідного підсилювача PGA1:

00 – коефіцієнт підсилення дорівнює 0.5;

01 – коефіцієнт підсилення дорівнює 1;

10 – коефіцієнт підсилення дорівнює 2;

11 – коефіцієнт підсилення дорівнює 4.

9.3.4.2 Регістр вибору каналу аналогового мультиплексора AMX1SL

Нижче наведено формат (рисунок 9.17) та опис окремих розрядів регістра AMX1SL.

Назва регістра:	AMX1SL - AMUX1 Channel Select Register		
SFR-адреса:	0xAC	Значення після скидання:	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
-	-	-	-	-	AMX1AD2	AMX1AD1	AMX1AD0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Рисунок 9.17 – Регістр вибору каналів аналогового мультиплексора AMX1SL

Біти 7…3: не використовується, при зчитуванні повертає 00000b, при запису значення ігнорується.

Біти 2…0: AMX1AD2… AMX1AD 0 (AMUX1 Adress Bits) – біти вибору каналу мультиплексора:

000 – обрано канал AIN1.0;

001 – обрано канал AIN1.1;

010 – обрано канал AIN1.2;

011 – обрано канал AIN1.3;

100 – обрано канал AIN1.4;

101 – обрано канал AIN1.5;

110 – обрано канал AIN1.6;

111 – обрано канал AIN1.7.

9.3.4.3 Регістр керування АЦП ADC1CN

Нижче наведено формат (рисунок 9.18) та опис окремих розрядів регістра ADC1CN.

Назва регістра:	ADC1CN - ADC1 Control Register		
SFR-адреса:	0xAA	Значення після скидання:	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
AD1EN	AD1TM	AD1INT	AD1BUSY	AD1CM2	AD1CM1	AD1CM0	-
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Рисунок 9.18 – Регістр керування ADC1CN

Біт 7: AD1EN: біт вмикання ADC1.

0: ADC1 вимкнено. ADC1 знаходиться в режимі низького енергоспоживання.

1: ADC1 ввімкнено. ADC1 готовий для перетворення даних.

Біт 6: AD1TM: біт режиму стеження ADC1.

0: Якщо ADC1 знаходиться у вихідному нульовому стані, за час, доки перетворювач чекає запуску, відбувається стеження за вхідною напругою.

1: Якщо AD1TM = 1, то перетворювач працює у режимі низького енергоспоживання (НЕ). В цьому випадку перед перетворенням деякий період ведеться стеження, який у випадку використання сигналу зовнішнього запуску CNVSTR має тривалість, яка дорівнює тривалості низького рівня цього сигналу (рисунок 9.15, а), у випадку використання зовнішніх джерел запуску триває 3 періоди тактової частоти SAR Clocks (рисунок 9.15, б). Стеження керується бітами AD1CTM2… AD1CTM0.

Біт 5: AD1INT: прапорець переривання після завершення аналого–цифрового перетворення (має скидатися програмно).

0: ADC1 не завершував перетворення з того часу, коли востаннє біт було скинуто;

1: ADC1 завершив процес перетворення.

Біт 4: AD1BUSY: біт занятості.

Зчитування

0: ADC1 завершив перетворення чи після скидання не було перетворень. Падаючий фронт ADBUSY генерує переривання, якщо воно дозволено.

1: ADC1 перетворює дані.

Запис

0: немає ефекту.

1: ADC1 розпочинає перетворення, якщо ADCTM2…0 = 000b

Біт 3 …1 : AD1CM2…AD1CM0: біти режиму запуску перетворення ADC1.

Якщо AD1TM = 0, то:

000: Перетворення розпочинається після запису 1 в ADBUSY;

001: Перетворення розпочинається після кожного переповнення таймеру 3;

010: Перетворення розпочинається після кожного перепаду рівня з низького в високий на вході CNVSTR;

011: Перетворення розпочинається після кожного переповнення таймеру 2;

1xx: Перетворення розпочинається при встановленні біта AD0BUSY, тобто синхронний запуск ADC0 та ADC1.

Якщо AD1TM = 1, то:

000: Стеження розпочинається після запису 1 в ADBUSY і триває 3 періоди тактової частоти SAR Clocks, з подальшим перетворенням;

001: Стеження розпочинається після кожного переповнення таймеру 3 і триває 3 періоди тактової частоти SAR Clocks, з подальшим перетворенням;

010: Стеження розпочинається при низькому рівні на вході CNVSTR, перетворення розпочинається після кожного перепаду рівня з низького в високий на вході CNVSTR;

011: Стеження розпочинається після кожного переповнення таймеру 2 і триває 3 періоди тактової частоти SAR Clocks, з подальшим перетворенням;

1xx: Стеження розпочинається при встановленні біта AD0BUSY, тобто синхронний запуск ADC0 та ADC1 і триває 3 періоди тактової частоти SAR Clocks, з подальшим перетворенням;

Біт 0: не використовується, зчитується як 0, значення при записі ігнорується.

9.3.4.4 Регістр даних ADC1

Нижче наведено формат (рисунок 9.19) та опис окремих розрядів регістра ADC1.

Назва регістра	ADC1 - ADC1 Data Word Register		
SFR-адреса:	0x9C	Значення після скидання	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Рисунок 9.19 – Регістр даних ADC1

Біти 7…0: Біти ADC1 (Data Words Bits).

Байт вихідних даних ADC1.

ПИТАННЯ ДЛЯ САМОКОНТРОЛЮ

1) Опишіть функціональну схему 12–розрядного аналого–цифрового перетворювача.
2) В чому полягає суть програмованого підсилювача?
3) Опишіть принцип роботи АЦП.
4) Як працює аналоговий мультиплексом?
5) Для чого в модуль АЦП входить пристрій вибірки–зберігання?
6) Опишіть роботу ADC0 за часовими діаграмами.
7) Назвіть та опишіть способи запуску ADC0.
8) Як зберігається результат перетворення в ADC0?
9) Як здійснюється інформування про вихід вихідного коду ADC0 за межі «вікна»?
10) Опишіть призначення окремих розрядів регістра конфігурації аналогового мультиплексора AMX0CF.
11) Опишіть призначення окремих розрядів регістра вибору каналу аналогового мультиплексора AMX0SL.
12) Опишіть призначення окремих розрядів регістра конфігурації аналого–цифрового перетворювача ADC0CF.
13) Опишіть призначення окремих розрядів регістра керування ADC0CN.
14) Опишіть зв’язок між вхідними сигналами та вихідними кодами в одиночному та диференціальному режимах ADC0.
15) Опишіть функціональну схему 8–розрядного аналого–цифрового перетворювача.
16) Як програмується частота синхронізації аналого–цифрового перетворювача ADC1?
17) Опишіть призначення окремих розрядів керуючих регістрів ADC1.

10 МОДУЛЬ ЦИФРО–АНАЛОГОВОГО ПЕРЕТВОРЮВАЧА. МОДУЛЬ АНАЛОГОВОГО КОМПАРАТОРА

10.1 Модуль цифро–аналогового перетворювача (ЦАП)

10.1.1 Загальні відомості

Розглянемо архітектуру модуля ЦАП у складі мікроконтролерів сімейства МК51 на прикладі мікроконтролера фірми Cygnal C8051F020. Схема МК C8051F020 містить два ідентичних 12–бітних цифро–аналогових перетворювача (ЦАП): DAC0 і DAC1 з частотою перетворення до 1 МГц. Нижче наведено опис тільки перетворювача DAC0. Замінивши в описі індекс 0 на 1, можна легко отримати опис для перетворювача DAC1.

10.1.2 Принцип роботи ЦАП

Нижче буде розглянуто принцип роботи та розрахунку ЦАП на основі резисторної матриці R–2R з підсумовуванням напруг, який використовується в модулі ЦАП мікроконтролерів сімействах Cygnal. ЦАП з підсумовуванням напруг використовує режим роботи підсумовуючого елемента, близький до холостого ходу (операційний підсилювач підсумовує напруги, рисунок 10.1).

ЦАП, з підсумовуванням напруг, використовує зворотне включення входу і виходу матриці R–2R. На входи $a_0, a_1, a_2 ... a_{n-1}$ надходять цифрові сигнали, які відповідають значенню і–го розряду вхідного двійкового коду (i=0,1,…,n–1). Якщо на вході і–го розряду присутня логічна одиниця, то відповідний ключ $\text{Кл}_\text{і}$ переключається у верхнє положення та опорна напруга $U_{\text{ОП}}$ через резистори матриці R–2R з визначеним коефіцієнтом ділення подається на вхід операційного підсилювача (ОП) DA1, який не інвертує, де відбувається підсумовування напруг.

Якщо на вхід і–го розряду надходить логічний нуль, то ключ переключається в нижнє положення, і дана гілка матриці R–2R підключається до спільної шини.

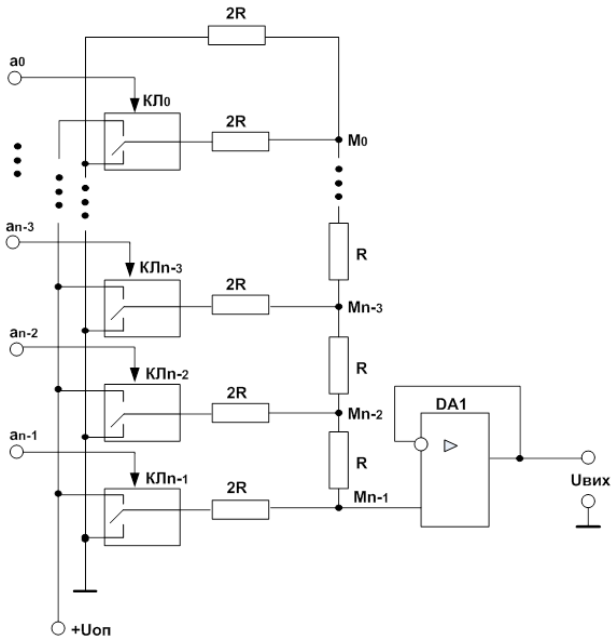


Рисунок 10.1 – n–розрядний ЦАП з матрицею R–2R та з підсумовуванням напруг

Оскільки матриця резисторів є лінійним ланцюгом, її роботу можна проаналізувати методом суперпозиції, тобто внесок у вихідну напругу від кожного джерела (розряду) розрахувати незалежно один від одного. Внески від кожного розряду підсумовуються на вході ОП, який не інвертує, і на виході отримуємо результат у вигляді напруги.

10.1.3 Розрахунок цифро–аналогових перетворювачів на матриці R–2R з підсумовуванням напруг

Розглянемо роботу ЦАП, якщо в старшому розряді вхідного ДК присутня логічна одиниця, а в інших розрядах – логічні нулі. Отже, ключ $Кл_{n-1}$ знаходиться у верхньому положенні і підключає гілку резисторної матриці

(РМ) з резистором 2R до джерела опорної напруги $U_{\text{ОП}}$. Інші ключі знаходяться в нижньому положенні і підключають інші гілки РМ (резистори 2R) до спільної шини. Еквівалентну схему ЦАП для цього випадку наведено на рисунку 10.2, а.

Очевидно, що еквівалентний опір РМ вище вузла M_{n-1} дорівнює 2R. Так як вхідний опір ОП великий і останній працює в режимі, близькому до холостого ходу, то струм, який створюється джерелом $U_{\text{ОП}}$, протікає через два однакових резистори 2R, які утворюють дільник напруги $U_{\text{ОП}}$. У цьому випадку напруга на виході дільника визначається з виразу:

$$U_{діл} = U_{\text{Н}} = \frac{U_{\text{ОП}} \cdot 2R}{2R + 2R} = \frac{U_{\text{ОП}}}{2}. \qquad (10.1)$$

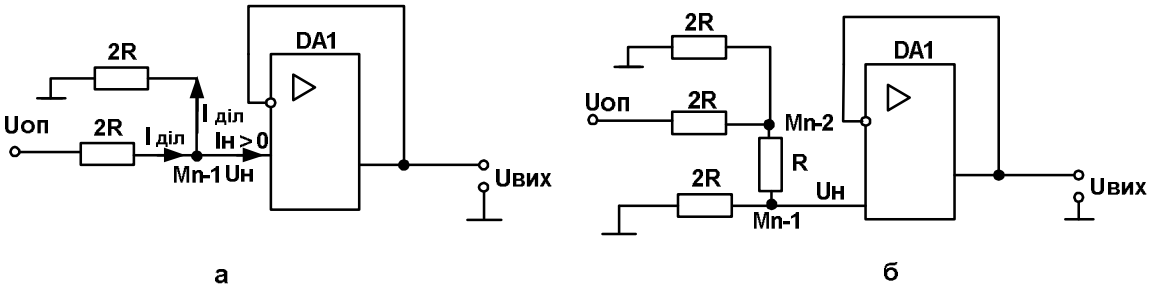


Рисунок 10.2 – Еквівалентні схеми ЦАП: а – при перетворенні коду 100…0В; б – при перетворенні коду 010…0В

Розглянемо роботу ЦАП, якщо на вхід схеми надходить комбінація ДК: 010…0В. У цьому випадку ключ $Кл_{n-2}$ увімкнений у верхнє положення, а інші ключі – у нижнє. Еквівалентна схема ЦАП прийме вигляд, який представлено на рисунку 10.2, б.

Розглядаючи резистори R і 2R, які розташовані нижче вузла M_{n-2}, як включені послідовно $(R_{\text{ВХ.DA1}} \to \infty)$, замінюємо їх еквівалентним опором:

$$\text{R} + 2\text{R} = 3\text{R}. \qquad (10.2)$$

Тоді напруга в точці M_{n-2} визначається виразом:

$$U_{M_{n-2}} = \frac{U_{\text{оп}} \cdot 2\text{R} \| 3\text{R}}{2\text{R} + 2\text{R} \| 3\text{R}} = \frac{U_{\text{оп}} \cdot 6/5 \cdot R}{2\text{R} + 6/5 \cdot R} = \frac{U_{\text{оп}} \cdot 3}{8}. \qquad (10.3)$$

Знаючи напругу в точці M_{n-2}, можна визначити сигнал у вузлі M_{n-1}

$$U_{M_{n-1}} = U_{\text{н}} = \frac{U_{M_{n-2}} \cdot 2\text{R}}{\text{R} + 2\text{R}} = \frac{U_{\text{оп}}}{4}. \qquad (10.4)$$

Аналогічним чином можна довести, що при подачі на вхід ЦАП ДК: 001…0В, напруга на вході ОП, який не інвертує, буде дорівнювати:

$$U_{\text{Н}} = \frac{U_{\text{ОП}}}{8}. \qquad (10.5)$$

І, нарешті, при надходженні коду: 00...01 В напруга

$$U_{\text{Н}} = \frac{U_{\text{ОП}}}{2^n}. \qquad (10.6)$$

Оскільки коефіцієнт передачі розглянутого підсумовуючого операційного підсилювача $K_{\text{U.ІМС ОП}} = 1$, то вираз для визначення сумарної вихідної напруги від дії одиниць у всіх розрядах вхідного ДК прийме вигляд:

$$U_{\text{ВИХ max}} = U_{\text{ОП}}\left(\frac{1}{2} + \frac{1}{4} + \frac{1}{8} + ... + \frac{1}{2^n}\right) = \frac{\text{U}_{\text{ОП}}}{2^n}\sum_{i=0}^{n-1} 2^i. \qquad (10.7)$$

Якщо позначити значення і–х розрядів вхідного ДК як a_i, де a_i дорівнює 0 чи 1, то останній вираз перетвориться до вигляду:

$$U_{\text{ВИХ}} = \frac{\text{U}_{\text{ОП}}}{2^n}\sum_{i=0}^{n-1} a_i \cdot 2^i. \qquad (10.8)$$

Співмножник $\sum_{\text{i}=0}^{\text{n}-1} \text{a}_\text{i} \cdot 2^\text{i}$ є десятковим еквівалентом вхідного двійкового коду (представляє десяткове значення вхідного цифрового коду).

Розглянутий перетворювач називають таким, що помножує, тому що вихідна напруга пропорційна добутку значення опорного сигналу $U_{\text{ОП}}$ на значення вхідного цифрового коду.

Коефіцієнт передачі, тобто розрахункове збільшення вихідної напруги при зміні вхідного коду на одиницю молодшого розряду (ціна молодшого значущого розряду (МЗР)) складає:

$$K_{\text{ЦАП}} = \frac{U_{\text{ОП}}}{2^{\text{n}}} \left[\frac{\text{В}}{\text{МЗР}}\right]. \qquad (10.9)$$

Для мікроконтролера C8051F020 вирази (10.7), (10.8) на (10.9) мають вигляд :

$$U_{\text{вих}.max} = \frac{U_{\text{оп}}}{2^{12}} \sum_{i=0}^{11} 2^i = \frac{U_{\text{оп}}}{2^{12}} \cdot 1023 \qquad (10.10)$$

$$U_{\text{вих}} = \frac{U_{\text{оп}}}{2^{12}} \sum_{i=0}^{11} a_i 2^i \qquad (10.11)$$

$$K_{\text{ЦАП}} = \frac{U_{\text{оп}}}{2^{12}} \left[\frac{\text{В}}{\text{МЗР}}\right] \qquad (10.12)$$

10.1.4 Опис функціональної схеми

Функціональну схему ЦАП DAC0 приведено на рисунку 10.3.

Власне, крім перетворювача DAC0, коефіцієнт передачі якого визначається напругою джерела опорної напруги V_{REF}, схема містить пару регістрів даних перетворювача DAC0H (DAC0 High Byte Register) і DAC0L (DAC0 Low Byte Register), пару регістр–защіпка, мультиплексор вхідних даних, мультиплексор синхронізації і вихідний повторювач, вихід якого підключений безпосередньо до аналогового виходу МК DAC0.

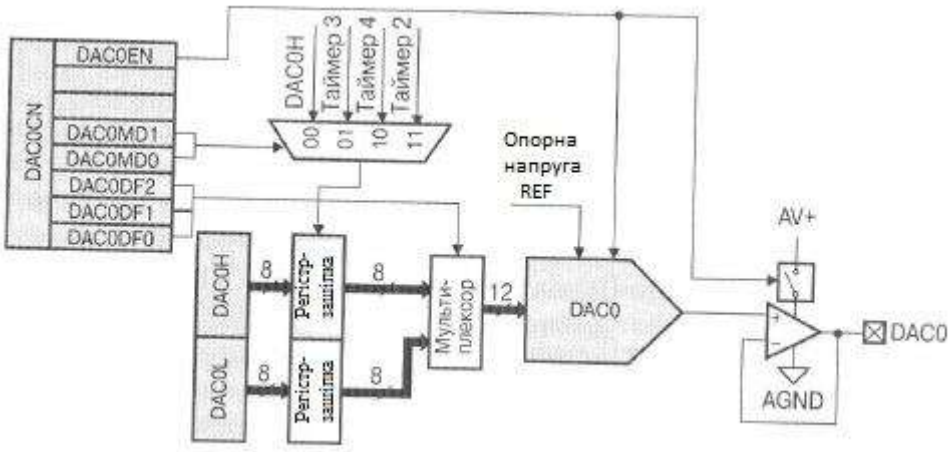


Рисунок 10.3 – Функціональна схема цифро–аналогового перетворювача DAC0

Керування роботою перетворювача відбувається за допомогою керуючого регістра DAC0CN (DAC0 Control Register, рисунок 10.7). Старший біт цього регістра – біт дозволу роботи перетворювача DAC0EN– DAC0CN.7(DAC0 Enable Bit). Якщо DAC0EN = 0, перетворювач вимкнений, струм споживання становить менше 1мкА, а його вихід DAC0 знаходиться у високоімпедансному стані. Якщо DAC0EN = 1, то перетворювач увімкнений, а його вихід активний.

Три молодших біти цього регістра – біти вибору формату даних DAC0DF2… DAC0DF0–DAC0CN[2:0] (DAC0 Data Format Bits). Це дозволяє керуючи мультиплексором даних вибрати один з 5 можливих варіантів розміщення 12–бітного вхідного слова перетворювача (dddd dddd dddd) в 16–бітному просторі пари регістрів даних DAC0H і DAC0L:

DAC0H	DAC0L
0000 dddd	dddd dddd;
000d dddd	dddd ddd0;
00dd dddd	dddd dd00;
0ddd dddd	dddd d000;
dddd dddd	dddd 0000.

Дана властивість дозволяє масштабувати коефіцієнт передачі перетворювача без зміни вхідного слова, а також використовувати як перетворювач з розширенням від 4 до 8 біт, завантажуючи вхідне дане тільки в старший регістр DAC0H і зберігаючи в регістрі DAC0L постійне значення (найчастіше 00H).

Момент передачі даних із регістрів DAC0H і DAC0L в регістри–защіпки визначає момент зміни аналогової напруги на виході DAC0 і залежить від режиму роботи перетворювача. Режим задається двома бітами режиму DAC0MD1…AC0MD0–DAC0CN[4:3] (DAC0 Mode Bits), які керують роботою мультиплексора синхронізації (рисунок 10.3). За допомогою цього мультиплексора в якості джерела синхронізації можна вибрати момент запису в регістр DAC0H або моменти переповнення одного з таймерів (3,4 або 2). В першому випадку зміна напруги на виході DAC0 буде відбуватися в момент запису значення в регістр DAC0H. Для коректної роботи 12–бітного перетворювача в цьому режимі потрібно спочатку проводити запис в молодший регістр DAC0L, а потім в старший – DAC0H. Режим синхронізації від таймерів використовується для точної синхронізації при формуванні аналогового сигналу, щоб моменти зміни вихідної напруги перетворювача на залежали від змінного часу реакції процесу на переривання. В цьому випадку зміна коду в регістрах–защіпках перетворювача відбувається точно в момент переповнення таймера, а в підпрограмі обробки переривання від таймера потрібно завантажити нове значення в регістри DAC0H і DAC0L.

10.1.5 Схема формування опорної напруги

Функціональну схему формування опорної напруги для АЦП і ЦАП, наприклад в МК C8051F020, наведено на рисунку 10.4.

Три зовнішніх виводи V_{REF0}, V_{REF1} і V_{REFD} дозволяють окремо для перетворювачів ADC0, ADC1 і пари DAC0, DAC1 вибирати в якості джерела

опорної напруги як зовнішню (як правило, на основі параметричного стабілізатора), так і внутрішню схему генератора опорної напруги. Окрім того, ADC0 може використовувати в якості опорної вихідну напругу перетворювача DAC0, а ADC1 може використовувати як опорну напругу живлення аналогової частини МК: AV+. Вибір відповідного джерела проводиться за допомогою внутрішніх мультиплексорів опорної напруги (рисунок 10.4).

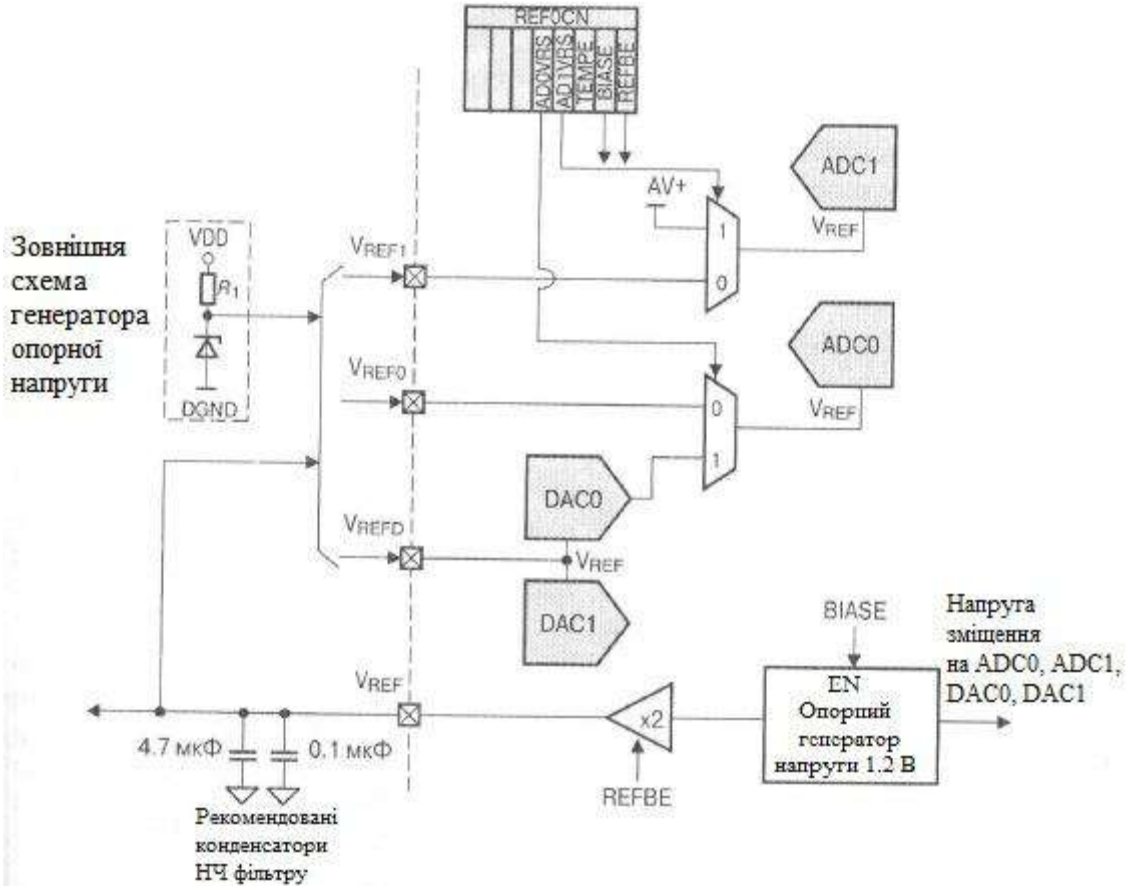


Рисунок 10.4 – Функціональна схема формування опорної напруги

Схема внутрішнього джерела опорної напруги містить опорний генератор напруги 1,2 В з коефіцієнтом температурної нестабільності $15 \cdot 10^{-6}\ {}^{1}/_{\circ C}$ і буферний підсилювач, який масштабує, з коефіцієнтом підсилення 2. Вихід останнього підключений до зовнішнього виходу V_{REF}. При використанні внутрішнього генератора рекомендується до цього виходу підключити

фільтруючі конденсатори, так як це показано на рисунку 10.4. Номінальна напруга внутрішнього джерела, таким чином, складе 2,4 В.

Керування роботою схеми виконується за допомогою бітів регістра керування опорною напругою REF0CN (Reference Control Register). Біт дозволу генератора зміщення BIASE–REF0CN.1 (ADC/DAC Bias Generator Enable Bit) повинен бути рівним 1, якщо використовується хоча б один з перетворювачів. Біт дозволу буферного підсилювача внутрішнього генератора REFBE–REF0CN.0 (Internal Reference Buffer Enable Bit) повинен бути встановлений в 1, якщо хоча б один з перетворювачів використовує внутрішній генератор опорної напруги (встановлена одна або кілька перемичок на відповідних зовнішніх виходах). Керуючі біти вибору джерела опорної напруги ADC0VRS–REF0CN.4 (ADC0 Voltage Reference Select) та ADC1VRS–REF0CN.3 (ADC0 Voltage Reference Select) керують роботою мультиплексорів опорної напруги перетворювачів ADC0 і ADC1 (рисунок 10.4).

Крім того, в цьому ж керуючому регістрі знаходиться і біт дозволу температурного датчика TEMPE (REF0CN.2) (Temperature Sensor Enable Bit), який керує ввімкненням вбудованого в модуль АЦП температурного датчика.

10.1.6 Програмування модуля ЦАП

10.1.6.1 Регістр даних

Регістр даних складається з двох байтів: старшого DAC0H (рисунок 10.5) та молодшого DAC0L (рисунок 10.6).

Назва регістра	DAC0H –DAC0 High Byte Register		
SFR – адреса	0xD3	Значення після скидання	00000000b (0x00)

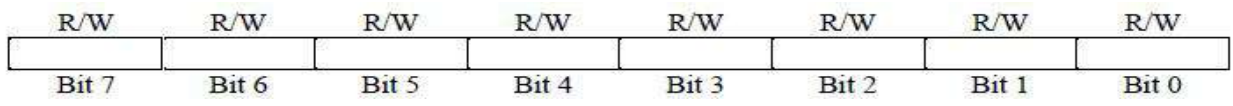


Рисунок 10.5 – DAC0H – старший байт регістра даних DAC0

Назва регістра	DAC0L –DAC0 Low Byte Register		
SFR – адреса	0xD2	Значення після скидання	00000000b (0x00)

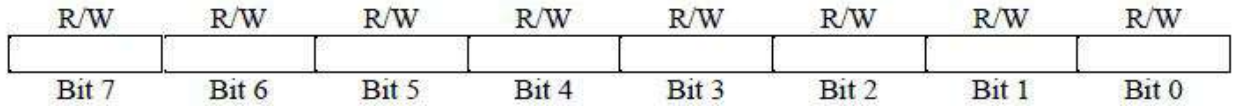


Рисунок 10.6 – DAC0L – молодший байт регістра даних DAC0

10.1.6.2 Регістр керування

Старший біт цього регістра (рисунок 10.7) – біт дозволу роботи перетворювача DAC0EN–DAC0CN.7 (DAC0 Enable Bit). Якщо DAC0EN = 0, перетворювач вимкнений, струм споживання становить менше 1мкА, а його вихід DAC0 знаходиться у високоімпедансному стані. Якщо DAC0EN = 1, то перетворювач увімкнений, а його вихід активний.

Назва регістра	DAC0CN –DAC0 Control Register		
SFR– адреса	0xD4	Значення після скидання	00000000b (0x00)

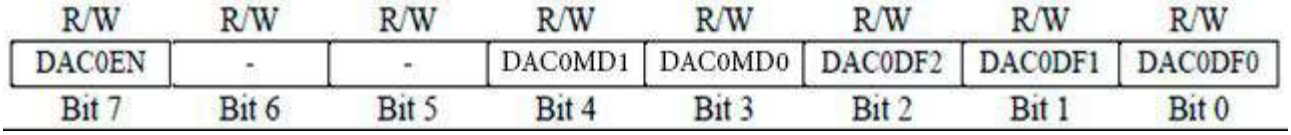


Рисунок 10.7 – DAC0CN – регістр керування DAC0

Три молодших біти цього регістра – біти вибору формату даних DAC0DF2… DAC0DF0, DAC0CN[2:0] (DAC0 Data Format Bits). Вони керують мультиплексором даних та дозволяють вибрати один з 5 можливих варіантів розміщення 12–бітного вхідного слова перетворювача (dddd dddd dddd) в 16–бітному просторі пари регістрів даних DAC0H і DAC0L:

DAC0H DAC0L

0000 dddd dddd dddd;

000d dddd dddd ddd0;

00dd dddd dddd dd00;

0ddd dddd dddd d000;

dddd dddd dddd 0000.

Якщо код в бітах DAC0DF2… DAC0DF0 дорівнює нулю (000b), дані вводяться без зсуву. Якщо код = 001b – дані записуються зі зсувом на 1 розряд вліво (в сторону більших розрядів). Якщо код = 010b зсув складає 2 розряди, якщо код = 011b – зсув складає 3 розряди, у всіх інших випадках зсув складає 4 розряди.

Дана властивість дозволяє масштабувати коефіцієнт передачі перетворювача без зміни вхідного слова, а також використовувати як перетворювач з розширенням від 4 до 8 біт, завантажуючи вхідне дане тільки в старший регістр DAC0H і зберігаючи в регістрі DAC0L постійне значення (найчастіше 00H).

DAC0MD1… DAC0MD0–DAC0CN[4:3]: біти режиму DAC0.

00 – зміна напруги на виході DAC0 буде відбуватися в момент запису значення в регістр DAC0H;

01 – зміна напруги на виході DAC0 буде відбуватися при переповненні таймера 3;

10 – зміна напруги на виході DAC0 буде відбуватися при переповненні таймера 4;

11 – зміна напруги на виході DAC0 буде відбуватися при переповненні таймера 2.

Біти 6…5 не використовуються, при читанні повертають 00b, при запису значення ігнорується.

10.1.6.3 Регістр керування опорною напругою

Нижче наведено формат (рисунок 10.8) та опис окремих розрядів регістра REF0CN.

Назва регістра	REF0CN – Reference Control Register		
SFR– адреса	0xD1	Значення після скидання	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
-	-	-	AD0VRS	AD1VRS	TEMPE	BIASE	REFBE
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Рисунок 10.8 – REF0CN – регістр керування опорною напругою

Біти 7…5: не використовуються, при зчитуванні повертається 00000b, при запису значенні ігнорується;

Біт 4: AD0VRS (ADC0 Voltage Reference Select)–вибір джерела опорної напруги ADC0: 0–ADC0 використовує опорну напругу з входу VREF0; 1–ADC0 використовує опорну напругу з виходу DAC0;

Біт 3: AD1VRS (ADC1 Voltage Reference Select)–вибір джерела опорної напруги ADC1: 0–ADC1 використовує опорну напругу з входу VREF1; 1–ADC1 використовує в якості опорної напруги живлення AV+.

Біт 2: TEMPE – біт дозволу температурного датчика (1 – ввімкнений; 0 – вимкнений);

Біт 1: BIASE – біт дозволу ланцюгів опорних напруг BIAS, необхідних для роботи аналогово–цифрових і цифро–аналогових перетворювачів (ADC & DAC): 1– ввімкнено;

Біт 0: REFBE – біт керування джерелом опорної напруги. Якщо біт = 0, використовується зовнішня опорна напруга, яка подається на вивід V_{REF}. Якщо біт = 1, використовується внутрішня опорна напруга = 2.4В.

10.2 Аналогові компаратори

10.2.1 Загальні відомості

Кожен МК C8051F020, наприклад, містить по два аналогових компаратори. Функціональну схему аналогових компараторів наведено на рисунку 10.9.

Входи кожного з компараторів підключені до виводів МК: CP0+, CP0– входи компаратора 0 і CP1 +, CP1– входи компаратора 1. Напруга на входах компаратора не повинна виходити за допустимий діапазон: –0,25 ... (AV +) + 0,25 В. Виходи компараторів через схему синхронізації можуть бути призначені на лінії портів введення / виведення МК за допомогою матричного комутатора. Крім того, вихід компаратора CP0 додатково підключено до схеми скидання МК і може використовуватися як джерело скидання.

Керування роботою компараторів здійснюється за допомогою бітів і прапорців керуючих регістрів компараторів: CPT0CN і CPT1CN (Comparator Control Register). Структура і функції цих регістрів абсолютно ідентичні, тому нижче дається опис бітів і прапорців тільки для компаратора 0.

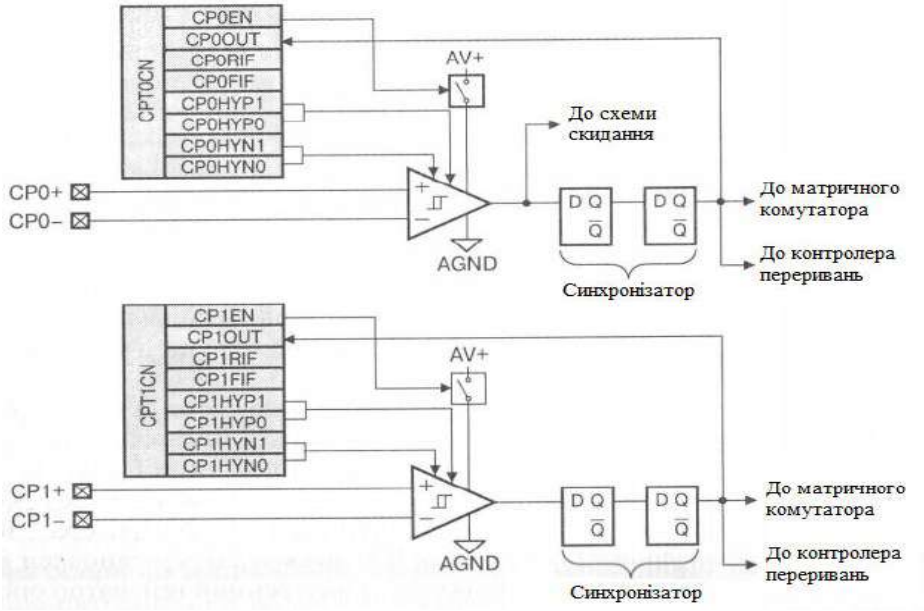


Рисунок 10.9 – Функціональна схема аналогових компараторів

Якщо біт дозволу роботи компаратора 0 CP0EN–CPT0CN.7 (Comparator 0 Enable Bit) знаходиться в стані 0, то компаратор вимкнений, струм споживання падає нижче 1 мкА, а на його виході, якщо він призначений на один з виходів МК, напруга близька до нульової. Встановлення в 1 біта дозволу CP0EN переводить компаратор в активний робочий стан. Співвідношення між вхідними напругами компаратора можна програмно проконтролювати за прапорцями стану виходу CP0OUT–CPT0CN.6 (Comparator 0 Output Stage Flag): якщо VCP0+ < VCP0– , то CP0OUT = 0, якщо CPO+ > CP0–, то CP0OUT = 1. Зміна напруги на виході компаратора може викликати переривання від компаратора: за наростаючим фронтом вихідного сигналу: прапорець CPT0RIF–CPT0CN.5 (Comparator 0 Rising–Edge Interrupt Flag) дорівнює 1, або за спадаючим фронтом: прапорець CPT0FIF–CPT0CN.4 (Comparator 0 Falling–Edge Interrupt Flag) дорівнює 1. Якщо переривання дозволені, то встановлення цих прапорців викликає звернення процесора до підпрограми обробки переривання від компаратора 0. Зазначені прапорці апаратно не скидаються, тому повинні бути скинуті в підпрограмі обробки переривання

За допомогою керуючих бітів регістра CPT0CN можна запрограмувати додатний та від`ємний гістерезис компаратора. Компаратор, що має можливість програмувати гістерезис, зручний для побудови схем поліпшення форми сигналу. Вплив гістерезису на вихідний сигнал компаратора зрозуміло з часових діаграм роботи компаратора (рисунок 10.10). Наявність гістерезису важлива також при реалізації на компараторі пристрою порівняння у релейних системах автоматичного керування .

У регістрі CPT0CN є два біти керування додатним гістерезисом CP0HYP 1… CP0HYP0–CPT0CN [3: 2] (Comparator 0 Positive Hysteresis Control Bits) і два біти керування від`ємним гістерезисом CP0HYN1… CP0HYN0–CPT0CN [1: 0] (Comparator 0 Negative Hysteresis Control Bits). Комбінація цих бітів 00b відповідає гістерезису 0 мВ, 01b: 2 мВ, 10b: 4 мВ, 11b: 10 мВ.

Рисунок 10.10 – Часові діаграми роботи компаратора

Таким чином, на кристалі МК є потужне процесорне ядро, значні за обсягом запам'ятовуючі пристрої програм і даних, розвинена система введення / виведення і потужний за своїми функціональними можливостями набір цифрових і прецизійних аналогових периферійних вузлів. Це дозволяє на базі одного МК практично без застосування додаткових компонентів реалізовувати складні системи керування різними об'єктами відповідно до концепції «Система на кристалі» ("System–on–Chip").

10.2.2 Програмування аналогових компараторів

10.2.2.1 CPT0CN – регістр керування компаратором 0

Нижче наведено формат (рисунок 10.11) та опис окремих розрядів регістра CPT0CN

Назва регістра:	CPT0CN – Comparator 0 Control Register							
SFR– адреса:	0x9E			Значення після скидання:			00000000b(0x00)	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
CP0EN	CP0OUT	CP0RIF	CP0FIF	CP0HYP1	CP0HYP0	CP0HYN1	CP0HYN0	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	

Рисунок 10.11 – Формат та опис регістра CPT0CN

Біт 7: CP0EN – біт дозволу (увімкнення) компаратора 0 (1 – ввімкнений, 0 – вимкнений)

Біт 6: CP0OUT – прапорець стану виходу компаратора 0. Якщо CP0+ < CP0–, прапорець дорівнює 0, інакше 1.

Біт 5: CP0RIF – прапорець дозволу переривання за переднім (наростаючим) фронтом вихідного імпульсу (1 – дозволено).

Біт 4: CP0FIF – прапорець дозволу переривання за заднім (спадаючим) фронтом вихідного імпульсу (1 – дозволено).

Біти 3…2:

CP0HYP1…0 – біти встановлення додатного гістерезису:

00 – додатний гістерезис заборонено;

01 – додатний гістерезис = 2 мВ;

10 – додатний гістерезис = 4 мВ;

11 – додатний гістерезис = 10 мВ.

Біти 1…0:

CP0HYN1…0 – біти встановлення від'ємного гістерезису:

00 – від'ємний гістерезис заборонено;

01 – від'ємний гістерезис = 2 мВ;

10 – від'ємний гістерезис = 4 мВ;

11 – від'ємний гістерезис = 10 мВ.

10.2.2.2 CPT1CN – регістр керування компаратором 1

Нижче наведено формат (рисунок 10.12) та опис регістра CPT1CN**.**

Назва регістра:	CPT1CN – Comparator 1 Control Register							
SFR–адреса:	0x9F		Значення після скидання:				00000000b(0x00)	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
CP1EN	CP1OUT	CP1RIF	CP1FIF	CP1HYP1	CP1HYP0	CP1HYN1	CP1HYN0	
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	

Рисунок 10.12 – Формат та опис регістра CPT1CN

Призначення бітів повністю співпадає з призначенням бітів, які описано в підрозділі 10.2.2.1, але по відношенню до компаратора 1.

ПИТАННЯ ДЛЯ САМОКОНТРОЛЮ

1) Скільки модулів ЦАП має типовий мікроконтролер фірми Cygnal C8051F020?
2) Опишіть принцип роботи ЦАП на основі резисторної матриці R–2R з підсумовуванням напруг.
3) Як розрахувати вихідну напругу та коефіцієнт передачі ЦАП мікроконтролера фірми Cygnal C8051F020?
4) Опишіть функціональну схему модуля ЦАП DAC0.
5) Опишіть 5 можливих варіантів розміщення 12–бітного вхідного слова перетворювача DAC0.
6) Чим визначається момент зміни аналогової напруги на виході DAC0?
7) Опишіть схему формування опорної напруги для ЦАП та АЦП.
8) Опишіть формат та призначення окремих розрядів регістра DAC0CN.
9) Опишіть формат та призначення окремих розрядів регістра REF0CN.
10) Скільки аналогових компараторів має мікроконтролер фірми Cygnal C8051F020?
11) Як змінюється вихідна напруга аналогового компаратора в залежності від співвідношення сигналів на його входах?
12) Як може бути викликано підпрограму обробки переривання від зміни вихідного сигналу аналогового компаратора?
13) Опишіть призначення та програмування додатного та від`ємного гістерезису аналогового компаратора.
14) Опишіть формат та призначення окремих розрядів регістра CPT0CN.

11 CAN–МОДУЛЬ

11.1 Основні характеристики CAN – протоколу

11.1.1 Загальна характеристика

До основних характеристик CAN–протоколу можна віднести:

- пріоритетність повідомлень;
- гарантований час відгуку;
- гнучкість конфігурації;
- груповий прийом із синхронізацією часу;
- система несуперечності даних;
- робота у мультимайстерному режимі;
- виявлення помилок та сигналізація про їх наявність;
- автоматична ретрансляція пошкоджених повідомлень, як тільки шина знову стане вільною;
- можливість відрізняти нерегулярні помилки та постійні відмови вузлів, а також автономне вимикання дефектних вузлів.

11.2 Структура повідомлень CAN–мережі

11.2.1 Загальні відомості

Повідомлення, що передаються CAN–шиною, називаються кадрами (фреймами). Формати кадрів наведено на рисунках 11.1…11.5. Залежно від ініціатора передачі і її мети існують чотири типи кадрів:

1) Data Frame – кадр даних;
2) Remote Frame – кадр запиту даних;
3) Error Frame – кадр помилки;
4) Overload Frame – кадр перевантаження.

Є два формати повідомлень, які передаються, що відрізняються довжиною поля ідентифікатора:

– кадри даних з 11–розрядним ідентифікатором, які називаються стандартними кадрами;
– кадри даних, що містять 29–розрядні ідентифікатори та називаються розширеними кадрами.

Кадр даних передає дані від передавача до приймача. Кадр віддаленого запиту даних передається вузлом, щоб запросити передачу кадру даних від іншого вузла з тим же самим ідентифікатором. Кадр помилки передається кожним вузлом при виявленні помилки на шині. Кадр перевантаження використовується, щоб забезпечити додаткову затримку між попереднім і наступним кадром даних або кадром віддаленого запиту даних. Кадри даних і кадри віддалених запитів даних можуть використовуватися і у стандартному і у розширеному форматі. Вони відокремлюються від попередніх кадрів міжкадровим простором.

11.2.2 Кадр даних (Data frame)

Кадр даних складається із семи різних полів: "початок кадру (повідомлення)" (start of frame), "поле арбітражу" (arbitration field), "поле керування" (control field), "поле даних" (data field), "поле CRC" (CRC field), "поле підтвердження" (ACK field), "кінець кадру (повідомлення)" (end of frame). Поле даних може мати нульову довжину. На рисунках 11.1, 11.2 наведено структури відповідно стандартного та розширеного повідомлень.

Нижче наведений опис окремих полів цих повідомлень.

11.2.2.1 Початок кадру (стандартний або розширений формат) (Start of frame)

Початок кадру відмічає початок кадру даних або кадру віддаленого запиту даних. Це поле складається з одиночного нульового біта. Вузлу дозволено почати передачу, коли шина вільна (див. міжкадровий простір). Всі

вузли повинні синхронізуватися за фронтом, який викликаний передачею поля "початок кадру" (див. апаратна синхронізація) вузлом, що почав передачу першим.

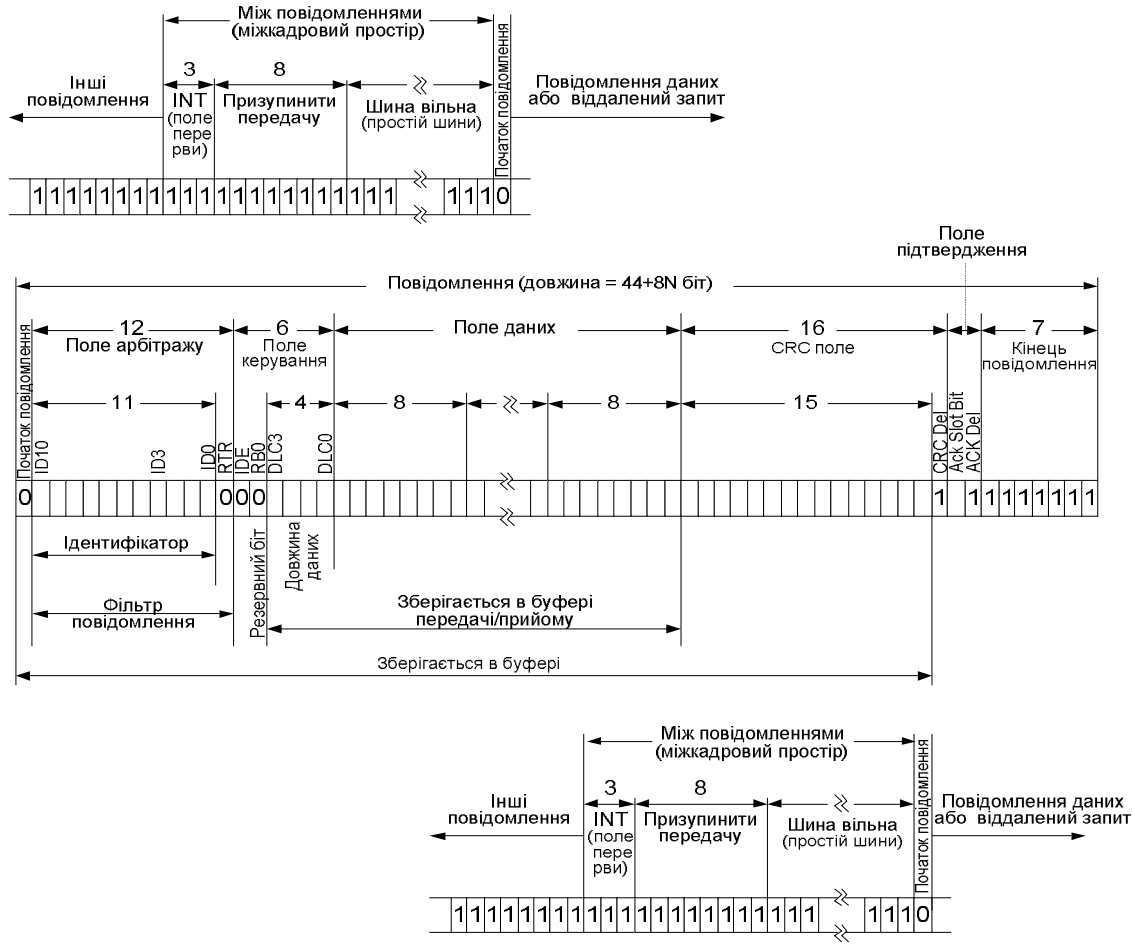


Рисунок 11.1 – Стандартне повідомлення

11.2.2.2 Поле арбітражу (Arbitration field)

Формат поля арбітражу відрізняється для стандартного та розширеного форматів:

– у стандартному форматі поле арбітражу складається з 11–розрядного ідентифікатора та RTR–біта. Біти ідентифікатора позначено ID10…ID0;

– у розширеному форматі поле арбітражу складається з 29–розрядного ідентифікатора, SRR–біта, IDE–біта, та RTR–біта. Біти ідентифікатора позначено ID28…ID0.

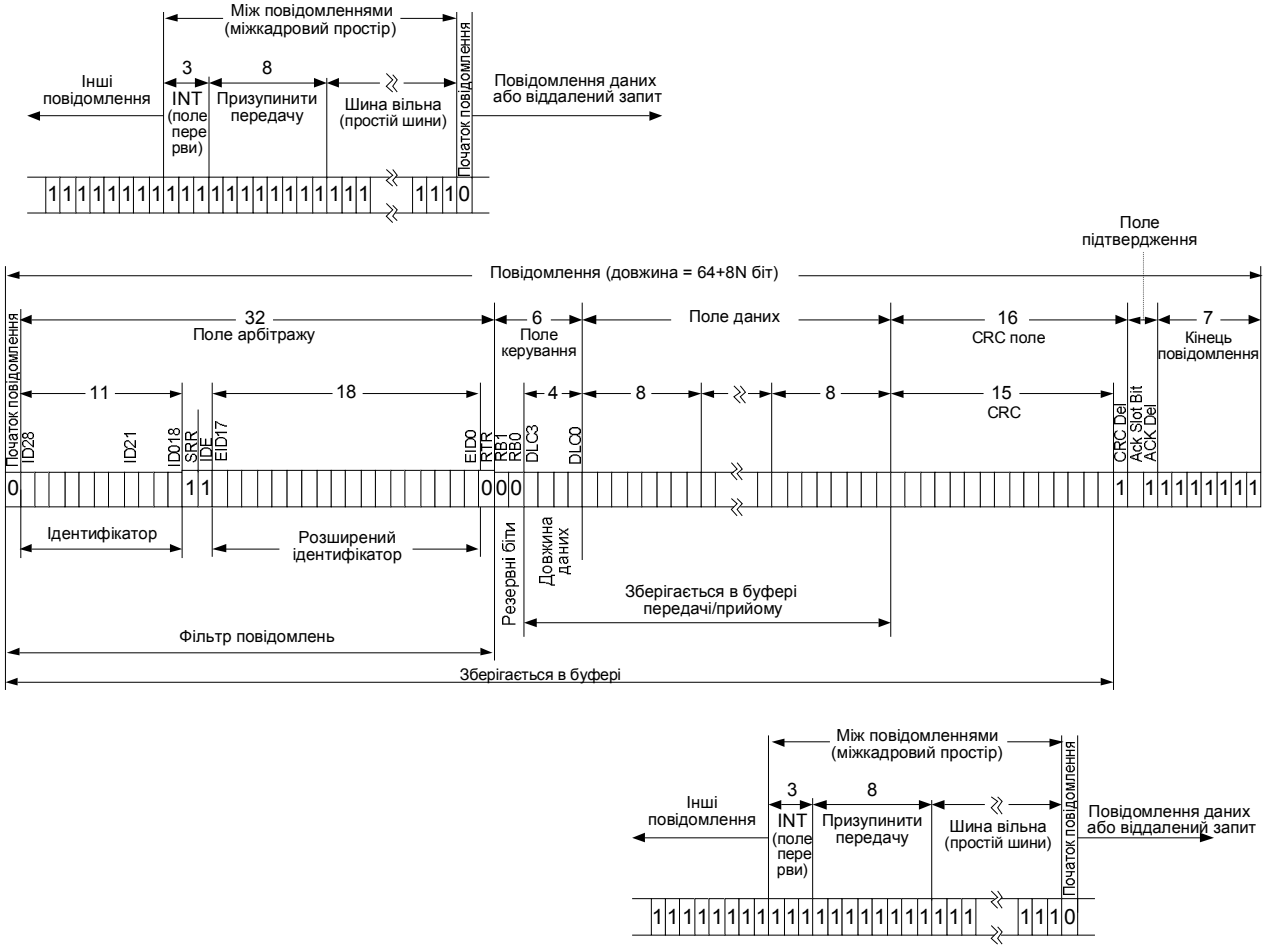


Рисунок 11.2 – Розширене повідомлення

11.2.2.3 Ідентифікатор (Identifier)

Ідентифікатор стандартного повідомлення.

Довжина ідентифікатора – 11 біт і відповідає BASE ID у розширеному форматі. Ці біти передаються в порядку ID10...ID0. Молодший біт – ID0. Сім старших бітів (ID10 … ID4) не повинні бути одиничними бітами. У стандартному кадрі ідентифікатор супроводжується RTR–бітом, який дорівнює нулю.

Ідентифікатор розширеного повідомлення.

На відміну від стандартного ідентифікатора, розширений ідентифікатор складається з 29 біт. Його формат містить дві секції:

1. Base ID – 11 біт
2. Extended ID – 18 біт

Base ID складається з 11 біт. Ця секція передається в порядку від ID28 до ID18. Це еквівалентно формату стандартного ідентифікатора. Base ID визначає базовий пріоритет розширеного кадру.

Extended ID складається з 18 біт. Ця секція передається в порядку від ID17 до ID0. У розширеному кадру ідентифікатор супроводжується RTR бітом, який дорівнює нулю.

11.2.2.4 Біт віддаленого запиту (стандартний і розширений формат) (біт RTR)

В кадрах даних RTR біт передається нульовим рівнем. У кадру віддаленого запиту даних RTR–біт повинен бути одиничним. У розширеному кадру спочатку передається Base ID, а наступними бітами передаються SRR та IDE. Extended ID передається після IDE–біта.

11.2.2.5 Біт SRR (розширений формат)

Замінник біта віддаленого запиту в стандартному повідомленні. SRR – одиничний біт. Він передається в розширених кадрах на позиції RTR біта. Таким чином, він заміняє RTR – біт стандартного кадру. Отже, при одночасній передачі стандартного кадру та розширеного кадру, Base ID якого збігається з ідентифікатором стандартного кадру, стандартний кадр має перевагу над розширеним кадром.

11.2.2.6 Біт IDE (розширений формат)

Біт розширеного ідентифікатора IDE належить:

- полю арбітражу для розширеного формату;
- полю керування для стандартного формату. IDE–біт у стандартному форматі передається нульовим рівнем, у той час як у розширеному форматі IDE біт передається одиничним рівнем.

11.2.2.7 Поле керування (стандартний формат і розширений формат) (Control field)

Поле керування складається із шести біт. Формат поля керування відрізняється для стандартного й розширеного формату. Кадри в стандартному форматі включають: код довжини даних (DLC), біт IDE, що передається нульовим рівнем (див. вище), та зарезервований біт RB0. Кадри в розширеному форматі включають код довжини даних і два зарезервованих біти RB1 й RB0. Зарезервовані біти передаються нульовим рівнем.

11.2.2.8 Код довжини даних (стандартний і розширений формати) (Data length code)

Число байт у полі даних визначається кодом довжини даних. Цей код довжини даних, розміром 4 біти, передається усередині поля керування (таблиця 11.1). Допустиме число байт даних, що передаються: {0,1,...,7,8}.
Інші величини використовуватися не можуть.

11.2.2.9 Поле даних (стандартний і розширений формати) (Data field)

Поле даних складається з даних, які будуть передані всередині кадру даних.

Воно може містити від 0 до 8 байт, кожен з яких містить 8 біт, які передаються, починаючи зі старшого значущого розряду (СЗР).

11.2.2.10 Поле CRC (стандартний і розширений формати) (CRC field)

Містить послідовність CRC та CRC–роздільник.

Таблиця 11.1 – Кодування довжини даних

Число байт даних	Код довжини даних			
	DLC3	DLC2	DLC1	DLC0
0	d	d	d	d
1	d	d	d	r
2	d	d	r	d
3	d	d	r	r
4	d	r	d	d
5	d	r	d	r
6	d	r	r	d
7	d	r	r	r
8	r	d	d	d
d – “домінантний” r – “рецесивний”				

11.2.2.10 Послідовність CRC (стандартний і розширений формати) (CRC Sequence)

Послідовність контролю кадру, яка отримується за правилом надлишкового циклічного коду, найкраще підходить для кадрів із числом бітів менше ніж 127 біт. При обчисленні CRC береться початковий поліном, що визначається як поліном, коефіцієнти якого задано послідовністю біт, які складаються з полів: "початок кадру", "поле арбітражу", "керуюче поле", "поле даних" (якщо є) і, для 15 молодших коефіцієнтів, 0. Цей поліном ділиться на поліном:

$$X^{15} + X^{14} + X^{10} + X^{8} + X^{7} + X^{4} + X^{3} + 1.$$

Залишок від цього поліноміального ділення і є послідовність CRC, яка передається шиною.

Приклад розрахунку поля CRC в кадру даних в якому передається 1 байт даних: 00001111, з ідентифікатором: 00000000001 (рисунки 11.3…11.5).

Довжина поля CRC дорівнює 15 бітам. В доцільності вибору такої довжини можна переконатися скориставшись наступною формулою:

$$k \geq \frac{d-1}{2}\log_2(m+k+1),$$

де m = 103 – довжина інформаційних символів, k – довжина перевірочних символів (поля CRC), d = 5 – мінімальна кодова відстань.

$$15 \geq \frac{5-1}{2}\log_2(103+15+1).$$

Рисунок 11.3 – Повідомлення, яке потрібно передати

Циклічний зсув здійснюється справа наліво, причому крайній лівий символ щоразу переноситься в кінець комбінації. Практично всі циклічні коди відносяться до систематичних кодів – у них контрольні та інформаційні розряди розташовані на чітко визначених місцях. Крім того, циклічні коди відносяться до числа блочних кодів. Кожен блок (одна літера є окремим випадком блоку) кодується самостійно. Ідея побудови циклічних кодів базується на використанні многочленів, що не приводяться в полі двійкових чисел. Многочлени, що не приводяться – це такі многочлени, які діляться без залишку тільки на себе і на одиницю.

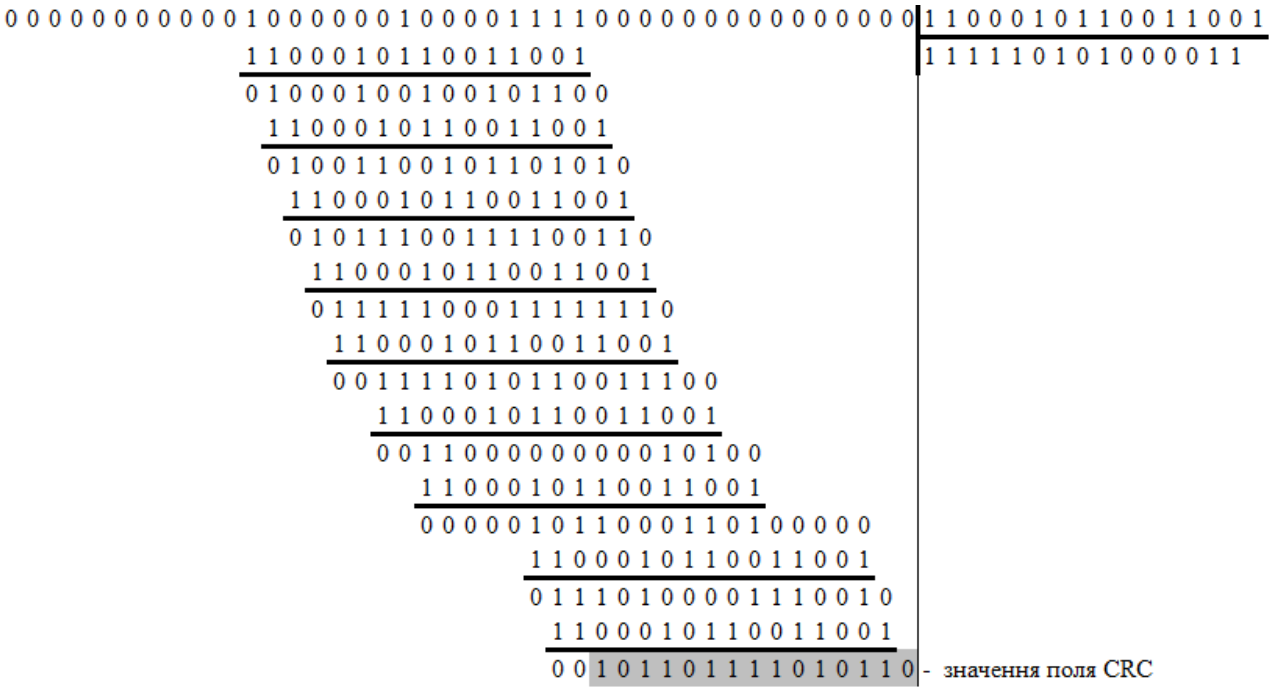


Рисунок 11.4 – Поліноміальне ділення

Початок повідомлення	Ідентифікатор	RTR	IDE	RB0	Довжина данних	Поле даних	CRC поле	CRC Del	ACK Slot Bit	ACK Del	Кінець кадру
0	0 0 0 0 0 0 0 0 0 0 0 1	0	0	0	0 0 0 1	0 0 0 0 1 1 1 1	1 0 1 1 0 1 1 1 1 0 1 0 1 1 0	1	1		1 1 1 1 1 1 1

Рисунок 11.5 – Повідомлення з розрахованим полем CRC

11.2.2.11 Роздільник CRC (стандартний формат, а також розширений формат) (CRC–Delimiter)

Послідовність CRC супроводжується роздільником CRC, що складається з одного одиничного біта.

Циклічні коди (CRC) названі так тому, що в них частина комбінацій коду або всі комбінації можуть бути отримані шляхом циклічного зсуву однієї або декількох комбінацій коду.

11.2.2.12 Поле підтвердження (стандартний формат і розширений формат) (ACK–field)

Поле підтвердження має довжину два біти й містить: біт "область підтвердження" (ACK Slot Bit) і роздільник підтвердження (ACK–delimiter). У поле підтвердження вузол, що передає, посилає два біти з одиничним рівнем. Приймач, що отримав повідомлення правильно, сповіщає про це передавачу, посилаючи біт з нульовим рівнем протягом прийому біта "область підтвердження".

Всі вузли, що отримали повідомлення з відповідною послідовністю CRC, сповіщають про це під час прийому біта "область підтвердження" шляхом заміни біта з одиничним рівнем на біт з нульовим рівнем.

Роздільник підтвердження – другий біт поля підтвердження, і він повинен бути представлений одиничним рівнем.

11.2.2.13 Кінець кадру (стандартний формат і розширений формат) (End of frame)

Кожен кадр даних і кадр віддаленого запиту даних закінчується послідовністю прапорців, яка складається із семи одиничних бітів.

11.2.3 Кадр віддаленого запиту даних (Remote frame)

Вузол, що діє як приймач деяких даних, може ініціювати передачу відповідних даних потрібними вузлами, посилаючи кадр віддаленого запиту даних (рисунок 11.6).

Кадр віддаленого запиту даних існує і у стандартному форматі, і у розширеному форматі. В обох випадках він складається із шести бітових полів: "початок кадру (повідомлення)" (Start of frame), "поле арбітражу" (Arbitration field), "поле керування " (Control field), "поле CRC" (CRC–field), "поле підтвердження" (ACK–field), "кінець кадру" (End of frame). На відміну від кадру даних, RTR біт кадру віддаленого запиту даних – одиничний. У цьому кадру

відсутнє поле даних. Значення коду довжини даних відповідає коду довжини даних кадру даних, який запитується. RTR–біт вказує, чи є переданий кадр кадром даних, чи кадром віддаленого запиту.

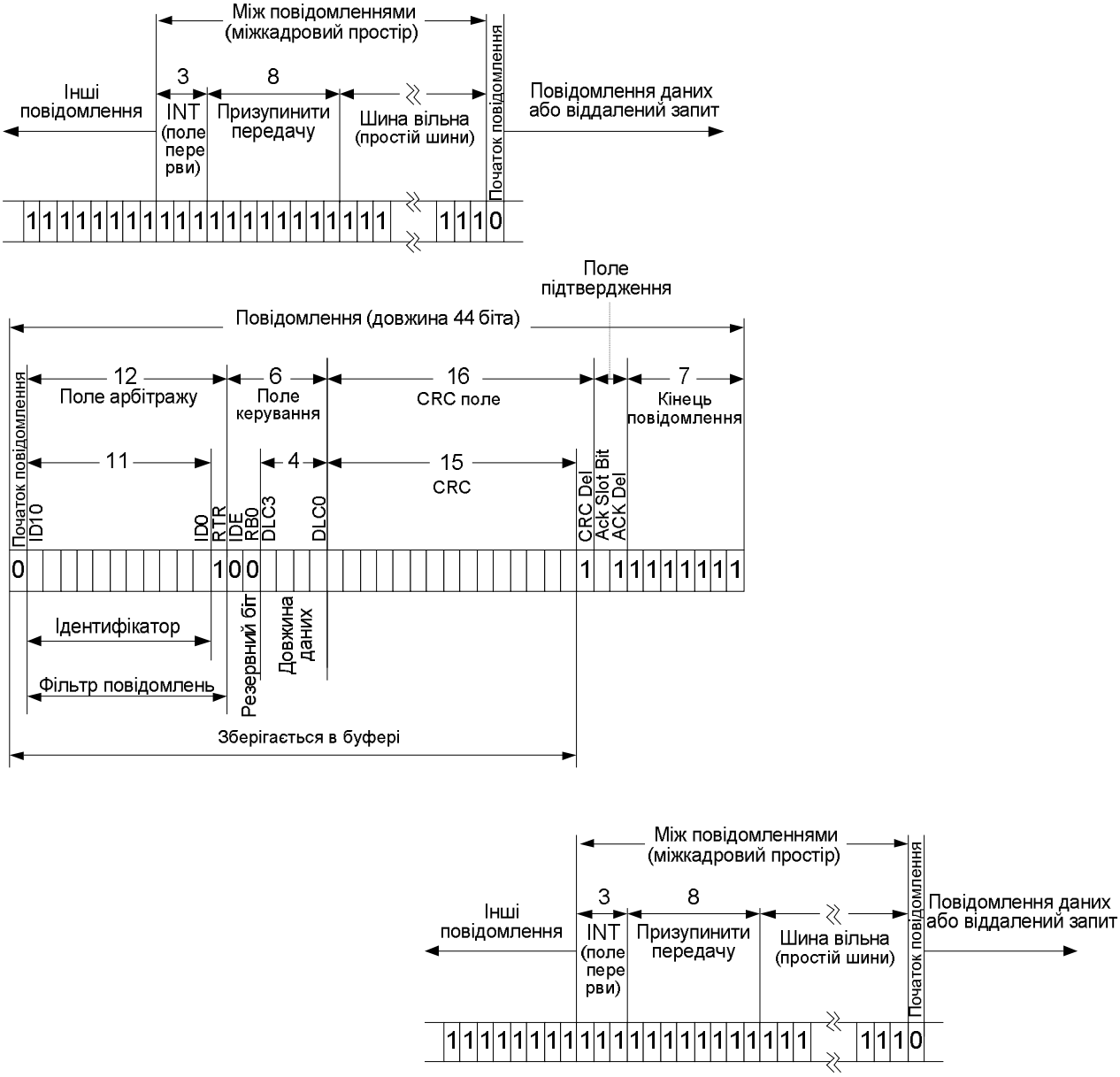


Рисунок 11.6 – Віддалений запит (стандартний)

В таблиці 11.2 показаний вплив бітів RTR, SRR и IDE на тип повідомлення, що передається.

Таблиця 11.2 – Вплив бітів RTR, SRR та IDE на тип кадру (повідомлення)

№ рядка	Значення біта			Тип кадру
	RTR в стандартному повідомленні (SRR в розширеному повідомленні)	IDE	RTR в розширеному повідомленні	
1	0	0	–	Стандартне повідомлення
2	1	1	0	Розширене повідомлення
3	1	0	–	Віддалений запит (стандартний)
4	1	1	1	Віддалений запит (розширений)

11.2.4 Кадр помилки (Error frame)

Кадр (повідомлення) помилки (рисунок 11.7) складається із двох різних полів. Перше поле є суперпозицією прапорців помилки, отриманих від вузла, який передає кадр помилки, та інших вузлів. Наступне поле – роздільник помилки (Error Delimiter).

Для правильного закінчення кадру помилки, вузол у стані "пасивної помилки" може мати потребу в доступі до шини, принаймні, на 3 бітових інтервали (якщо має місце локальна помилка приймача, що перебуває в стані "пасивної помилки"). Отже, шина не повинна бути завантажена на 100 %.

11.2.4.1 Прапорець помилки (Error flag)

Є два види прапорця помилки: прапорець активної помилки та прапорець пасивної помилки.

1. Прапорець активної помилки складається із шести послідовних нульових біт.

2. Прапорець пасивної помилки складається із шести послідовних одиничних біт, якщо вони не перезаписані нульовими бітами інших вузлів.

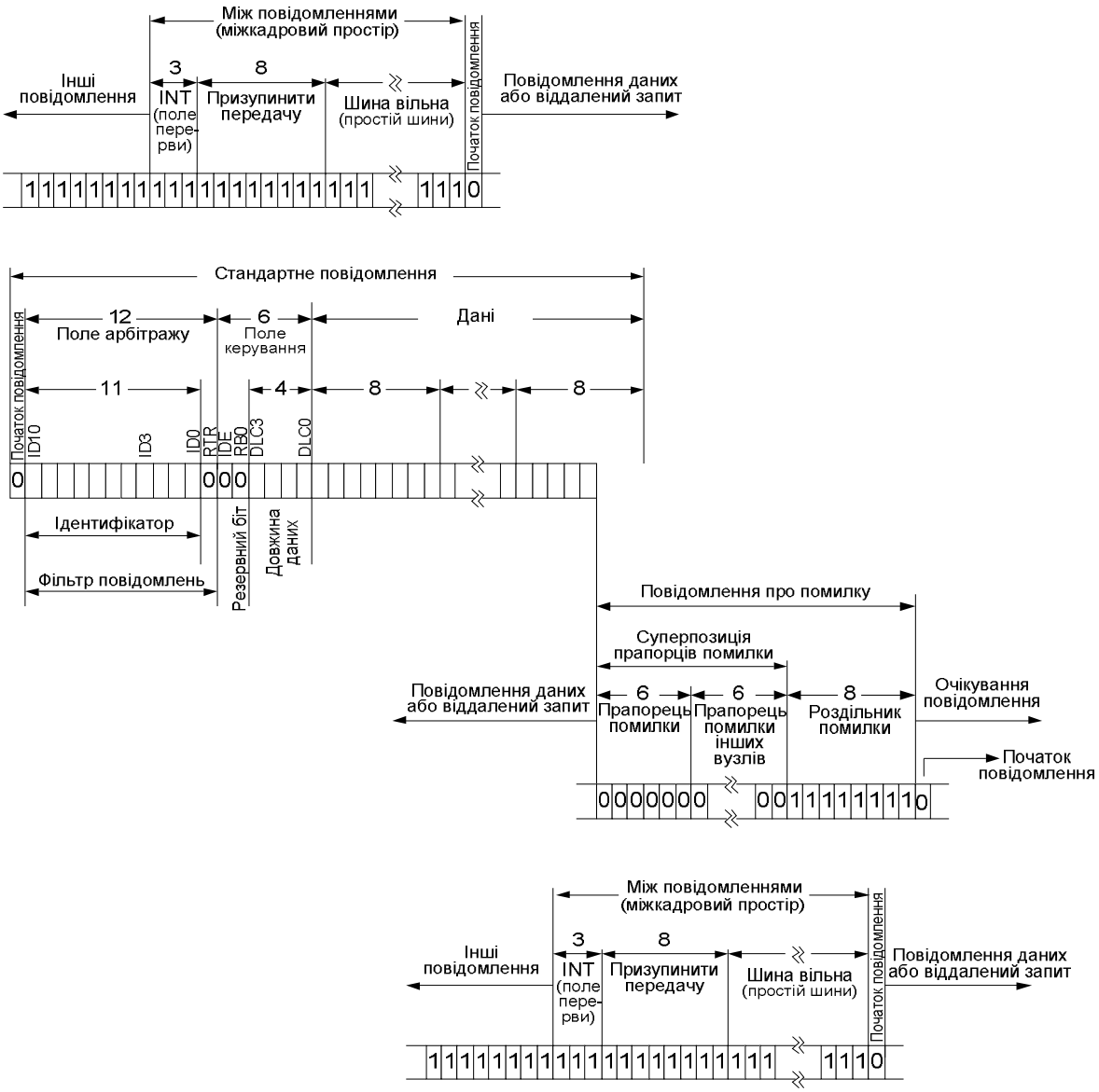


Рисунок 11.7 – Повідомлення про помилку

Вузол у стані "активної помилки", виявивши умову помилки, сповіщає про це передачею прапорця активної помилки. Форма прапорця помилки порушує

правило додаткового біта (біт–стаффінг), який застосовується до всіх полів від поля "початок кадру" до роздільника CRC, або руйнує фіксовану форму поля підтвердження або поля "кінець кадру". При відсутності помилки передавач вставляє один додатковий біт протилежної полярності після п'яти однакових, які ідуть підряд. Як наслідок порушення біт–стаффінгу, всі інші вузли виявляють умову помилки і у свою чергу починають передачу прапорця помилки. Таким чином, послідовність "домінантних" бітів, що фактично може з'явитися на шині, отримується суперпозицією різних прапорців помилки, які передані окремими вузлами. Сумарна довжина цієї послідовності змінюється від шести до дванадцяти біт. Вузол, у стані "пасивної помилки", виявивши умову помилки, робить спробу сповістити про це передачею прапорця пасивної помилки. Він очікує появи шести послідовних однакових одиничних біт, які починають прапорець пасивної помилки. Прапорець пасивної помилки завершується після виявлення цих 6 біт.

11.2.4.2 Роздільник помилки (Error Delimiter)

Роздільник помилки складається з восьми "рецесивних" бітів. Після передачі прапорця помилки кожен вузол посилає "рецесивні" біти та перевіряє шину, поки не виявляє "рецесивний" біт. Далі він починає передачу ще семи "рецесивних" бітів.

11.2.5 Кадр перевантаження (Overload Frame)

Кадр перевантаження містить два бітових поля: прапорець перевантаження та роздільник перевантаження (рисунок 11.8).

Є три види перевантаження, які приводять до передачі прапорця перевантаження:

– внутрішній стан приймача, який потребує затримки наступного кадру даних або кадру віддаленого запиту даних;

– виявлення "домінантного" біта при передачі першого та другого бітів поля перерви;
– якщо вузол виявляє "домінантний" біт на восьмому біті (останньому біті) роздільника помилки або роздільника перевантаження, це спричиняє передачу кадру перевантаження (а не кадру помилки). Лічильники помилок не будуть збільшені.

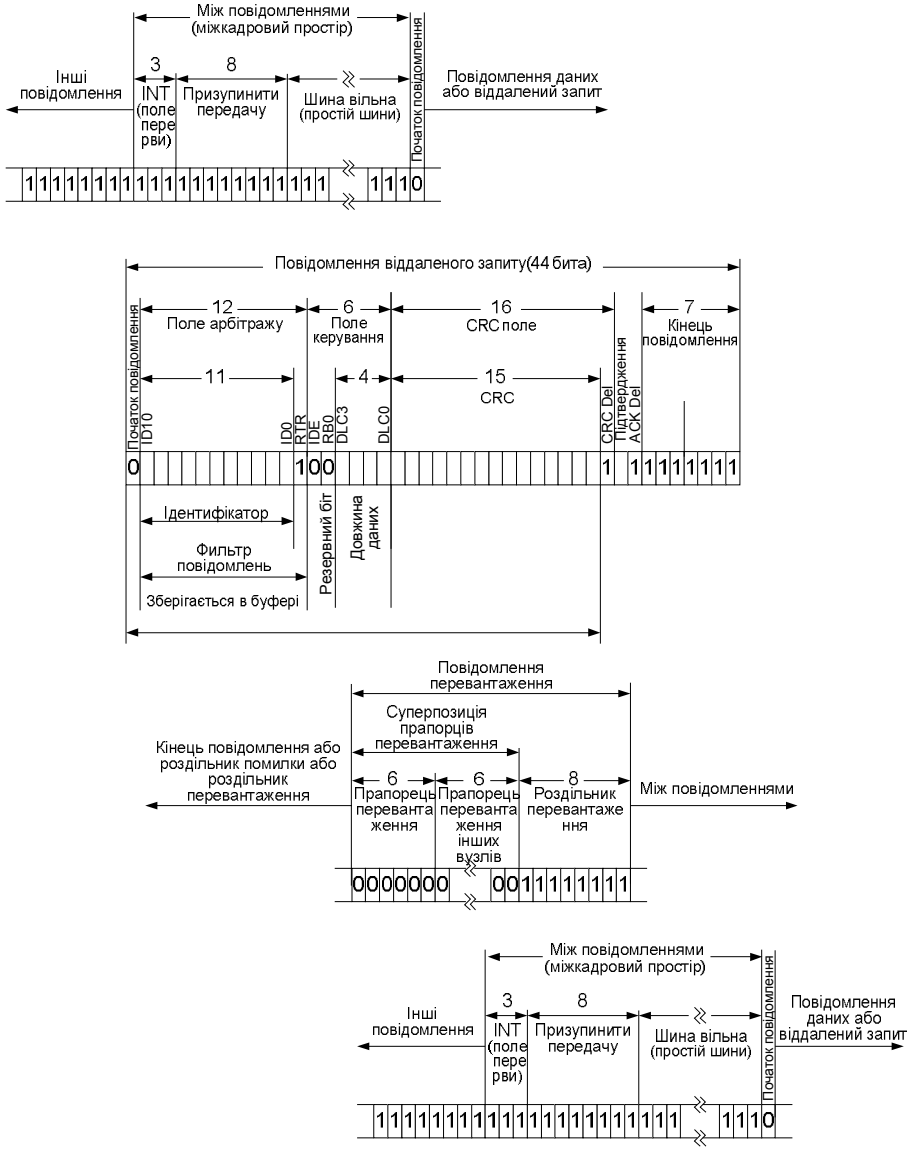


Рисунок 11.8 – Повідомлення перевантаження

Передачу кадру перевантаження, обумовленого 1–м видом перевантаження, дозволено починати в першому бітовому інтервалі передбаченого поля перерви, у той час як кадр перевантаження, обумовлений 2–м й 3–м видами перевантаження, починає передаватися на один біт пізніше виявлення "домінантного" біта. Для того щоб затримати наступний кадр даних або кадр віддаленого запиту даних, може бути сформовано не більше ніж два кадри перевантаження підряд.

Прапорець перевантаження (Overload Flag) складається із шести "домінантних" бітів. Повна форма відповідає прапорцю активної помилки. Форма прапорця перевантаження порушує фіксовану форму поля перерви. Всі інші вузли також виявляють умову перевантаження і у свою чергу починають передачу прапорця перевантаження. Таким чином, послідовність "домінантних" бітів, що фактично може з'явитися на шині, отримується суперпозицією прапорців перевантаження, які передані окремими вузлами. Сумарна довжина цієї послідовності змінюється від шести до дванадцяти біт. У випадку, якщо виявлено "домінантний" біт під час 3–го біта перерви, то цей біт інтерпретується як початок кадру.

Роздільник перевантаження (Overload Delimiter) складається з восьми "рецесивних" біт. Роздільник перевантаження має таку ж форму, як і роздільник помилки. Після передачі прапорця перевантаження вузол контролює шину, поки не виявить перехід від "домінантного" біта до "рецесивного" біта. У цей момент часу кожен вузол закінчує передачу прапорця перевантаження, і всі вузли починають передачу ще 7 "рецесивних" біт.

11.2.6 Міжкадровий простір (Interframe Space)

Кадри даних і кадри віддаленого запиту даних відокремлюються від попередніх кадрів (кадр даних, кадр віддаленого запиту даних, кадр помилки, кадр перевантаження) бітовим полем, яке називається міжкадровим простором

(рисунок 11.1, 11.2, 11.6, 11.7, 11.8). На відміну від них, кадрам перевантаження й кадрам помилки не передує міжкадровий простір (кратні кадри перевантаження також не відокремлюються інтервалами).

Для вузла, що був джерелом попереднього повідомлення, міжкадровий простір містить бітові поля: перерва та простій шини. Для вузла в стані "пасивної помилки", що був джерелом попереднього повідомлення, міжкадровий простір містить додаткове поле «призупинити передачу». Це поле має бути між полями «перерва» і «простій» шини.

11.2.6.1 Поле перерви (Intermission)

Складається із трьох "рецесивних" біт. Протягом перерви єдина дія, яку можна виконати – повідомлення перевантаження й ніякий вузол не може почати передачу кадру даних або кадру віддаленого запиту даних.

Зауваження: якщо вузол має повідомлення, яке потрібно передати, і він виявляє "домінантний" біт у третьому біті перерви, це інтерпретується, як біт "початок кадру", і, з наступного біта, він починає передавати повідомлення, з першого біта його ідентифікатора, не передаючи попередньо, біт "початок кадру".

11.2.6.2 Простій шини (Bus Idle)

Період простою шини може мати довільну довжину. Коли шина вільна, тоді вузол, якому потрібно передати інформацію, може звертатися до шини. Повідомлення, що чекає передачі під час передачі іншого повідомлення, починає передаватися в першому біті після перерви. Виявлення "домінантного" біта на шині інтерпретується як "початок кадру".

11.2.6.3 Призупинення передачі (Suspend Transmission)

Після того, як вузол у стані "пасивної помилки" передав повідомлення, він посилає вісім "рецесивних" бітів після поля перерви, перед передачею

наступного повідомлення або розпізнавання простою шини. Якщо тим часом починається передача, яка викликана іншим вузлом, вузол перетворюється в приймач цього повідомлення.

11.3 Бітова синхронізація

11.3.1 Номінальна швидкість передачі та номінальна тривалість біта

Номінальна швидкість передачі інформації визначається числом бітів у секунду, які передаються ідеальним передавачем при відсутності пере синхронізації.

Номінальний час передачі біта =

=1/ номінальна швидкість передачі інформації в бітах в секунду.

Номінальний час передачі біта можна розділити на кілька ділянок, що не перекриваються (рисунок 11.9):

- сегмент синхронізації (SYNC_SEG);
- сегмент часу розповсюдження (PROP_SEG);
- фазовий сегмент 1 (PHASE_SEG1);
- фазовий сегмент 2 (PHASE_SEG2).

Кожен сегмент складається із цілого числа відрізків часу, які називаються квантами часу TQ. Номінальний час передачі біта коливається від 8TQ до 25TQ (включно). Мінімальний номінальний час передачі біта дорівнює 1мкс, що відповідає швидкості передачі 1 Мбіт/с.

Тривалість одного кванта часу, як правило, визначається тактовою частотою мікроконтролера і значенням коефіцієнта ділення цієї частоти, що може програмно змінюватися:

$$T_{SCL} = (BRP[5:0] + 1)\cdot T_{CLKIO},$$

де T_{CLKIO} – період тактової частоти підсистеми введення/виведення МК–ра, BRP[5:0] – десятковий еквівалент шести біт регістра CANBT1[14].

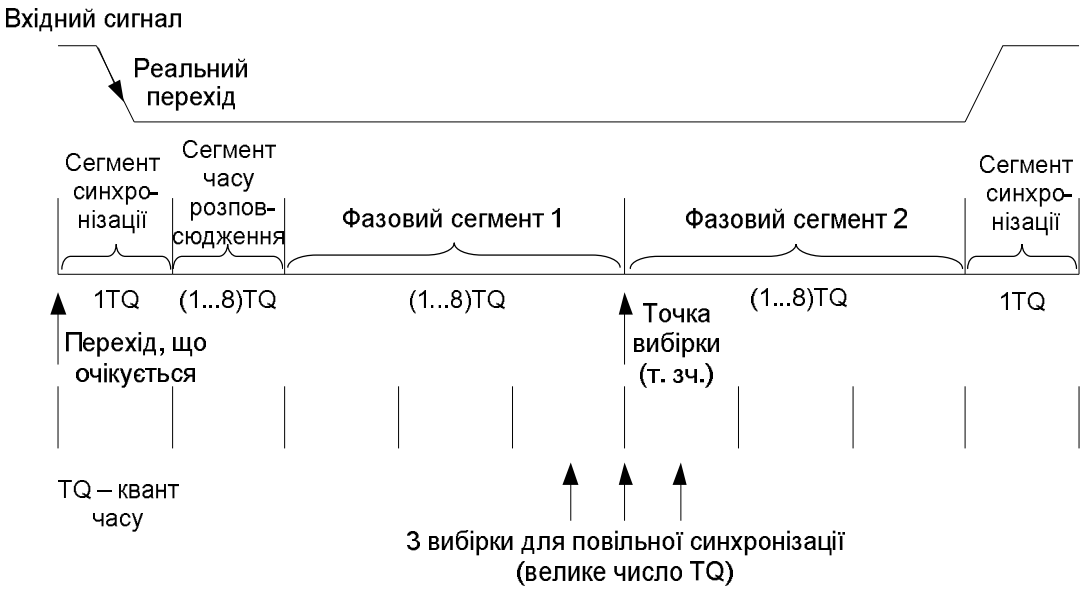


Рисунок 11.9 – Номінальний час біта

11.3.2 Сегмент синхронізації (SYNC_SEG)

Сегмент синхронізації (SYNC_SEG) використовується для синхронізації різних вузлів на шині. Передбачається, що фронт сигналу, що визначає початок передачі, повинен перебувати в межах цього сегмента.

11.3.3 Сегмент часу розповсюдження

Використовується, щоб компенсувати час фізичного запізнювання в мережі. Цей час дорівнює подвоєній сумі часу розповсюдження сигналу на лінії шини, затримки вхідного компаратора, і затримки вихідного формувача. Значення цього сегмента може програмуватися.

11.3.4 Фазові сегменти 1, 2

Ці сегменти використовуються, щоб компенсувати помилки фази при прийомі (фазові зсуви). Фазовий сегмент 1 може бути подовжений, а фазовий 2 –скорочений пересинхронізацією (ресинхронізацією).

Фазові сегменти призначені для оптимального розташування точки вибірки отриманого біта у межах часу біта, що передається. Точка вибірки розташовується між фазовим сегментом 1 і фазовим сегментом 2.

Фазовий сегмент 1 визначає точку вибірки в межах біта, що передається. Фазовий сегмент 2 забезпечує затримку до початку наступного біта. Тривалість обох цих сегментів може програмуватися.

11.3.5 Точка зчитування (вибірки) (Sample point)

Точка зчитування (вибірки) (Sample point) – це момент часу, при якому рівень на шині читається та інтерпретується, як значення відповідного біта. Вона розташована наприкінці фазового сегмента 1 (рисунок 11.9).

Якщо синхронізація біта повільна (містить велику кількість TQ), то можливо задати багаторазовий вибірковий контроль стану шини. В цьому випадку CAN–модуль робить вибірку три рази для кожного біта, який прийнятий, з періодичністю TQ/2. Значення, яке отримано за мажоритарним принципом (два або три рази), приймається як правильне. Увімкнення багаторазової вибірки здійснюється встановленням спеціального біта (біт SMP в регістрі CANBT3), який програмується.

11.3.6 Синхронізація

Щоб компенсувати зсув фази між частотами генераторів різних вузлів шини та при зміні ведучого при арбітражі, кожен CAN–модуль повинен синхронізуватися до переходу рівня вхідного сигналу з 1 на 0 (рисунок 11.9). Як тільки перехід виявлений, то логіка синхронізації порівнює розміщення

переходу з переходом, що очікується при прийомі, та виконує налаштування значень фазових сегментів 1 та 2. Є два механізми синхронізації:

- апаратна синхронізація;
- синхронізація з відновленням тактових інтервалів (пересинхронізація/ ресинхронізація).

11.3.6.1 Апаратна синхронізація

Апаратна синхронізація виконується при переході від “рецесивного” до “домінантного” біта протягом холостого ходу шини, вказуючи початок повідомлення. При апаратній (жорсткій) синхронізації тактові інтервали (тривалість сегментів) не змінюються протягом усього повідомлення.

Після апаратної синхронізації внутрішній бітовий час кожного вузла перезапускається з сегмента синхронізації SYNC_SEG (рисунок 11.9).

11.3.6.2 Синхронізація з відновленням тактових інтервалів (пересинхронізація, ресинхронізація)

Синхронізація з відновленням тактових інтервалів призначена для зменшення фазових спотворень при прийомі та виконується автоматичним подовженням фазового сегмента 1 або скороченням фазового сегмента 2. Максимальне значення зміни фазових сегментів коливається в межах від 1TQ до 4TQ. Синхронізація виконується тільки при переходах від “рецесивного” до “домінантного” біта протягом прийому кадру. Фіксоване значення максимального числа послідовних біт однакової полярності (“біт–стаффінг“) гарантує своєчасне відновлення синхронізації.

Фазове спотворення (помилка фази): (e) визначається положенням фронту вхідного сигналу (переходу з одиниці в нуль) відносно точки зчитування бітового інтервалу приймача, в якому з'явився перехід з «1» в «0» і вимірюється у квантах часу TQ. Значення (e) визначається в квантах TQ як різниця часу точки зчитування ($t_{т.зч.}$) бітового інтервалу приймача, протягом якого з'явився перехід вхідного сигналу з «1» в «0», та часу появи цього переходу ($t_{ч}$).

Знак помилки фази (e) визначається в такий спосіб:

- e = 0, якщо фронт вхідного сигналу (перехід з 1 в 0) перебуває в межах сегмента синхронізації SYNC_SEG (пересинхронізація не проводиться);
- e > 0, якщо фронт вхідного сигналу (перехід з 1 в 0) не перебуває в межах SYNC_SEG та знаходиться перед точкою зчитування бітового інтервалу приймача, протягом якого з'явився перехід вхідного сигналу з «1» в «0»;
- e < 0, якщо фронт сигналу (перехід з 1 в 0) не перебуває в межах SYNC_SEG та знаходиться після точки зчитування бітового інтервалу приймача, протягом якого з'явився перехід вхідного сигналу з «1» в «0».

Ефект від пере синхронізації – такий самий як і у випадку апаратної синхронізації, коли величина помилки фази (e) менше або дорівнює значенню ширини періоду пере синхронізації, що програмується.

Коли величина помилки фази (e) більша, ніж ширина переходу пере синхронізації, що програмується, тоді:

- якщо помилка фази (e) додатна, то фазовий сегмент 1 подовжується на ширину періоду пере синхронізації, що програмується;

– якщо помилка фази (e) від’ємна, то фазовий сегмент 2 скорочується на ширину періоду пере синхронізації, що програмується.

11.3.6.3 Правила синхронізації

Апаратна синхронізація та пересинхронізація – дві форми синхронізації. Вони задовольняють наступним умовам:

1. Допускається тільки один тип пере синхронізації в межах бітового інтервалу.
2. Фронт сигналу буде використовуватися для пере синхронізації тільки тоді, якщо значення, яке виявлене при попередній точці зчитування (попереднє значення на шині), відрізняється від значення на шині відразу після фронту.
3. Апаратна синхронізація відбувається щораз, коли є перехід від "рецесивного" біта до "домінантного" протягом простою шини.

На рисунках 11.10, 11.11 наведені приклади пере синхронізації.

11.3.6.4 Тривалість сегментів

SYNC_SEG – 1 квант часу.

PROP_SEG – програмується, може бути 1,2...8 квантів часу.

PHASE_SEG1 – програмується, може бути 1,2...8 квантів часу.

PHASE_SEG2 – програмується, може бути 1,2...8 квантів часу.

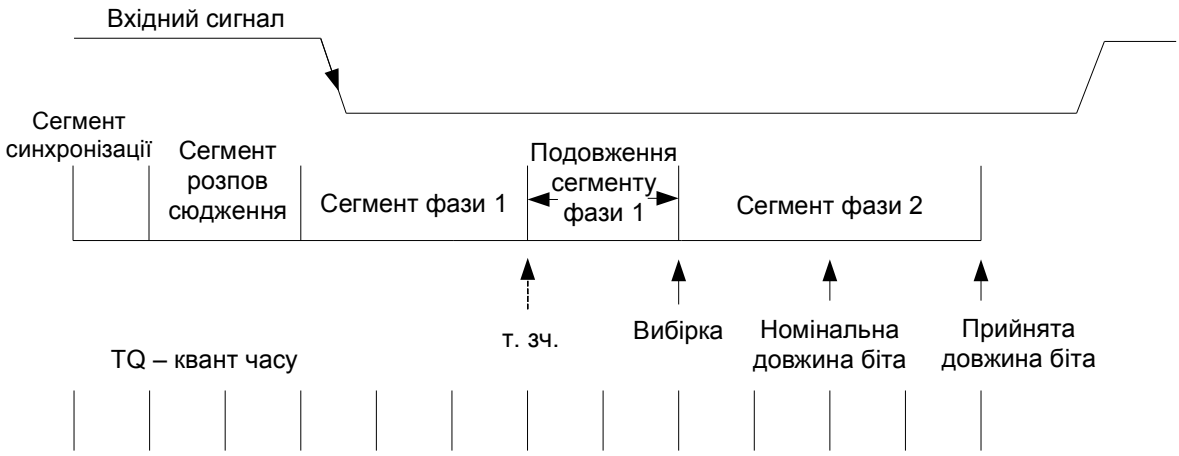


Рисунок 11.10 – Синхронізація з відновленням тактових інтервалів за рахунок автоматичного подовження фазового сегмента (сегмента фази) 1

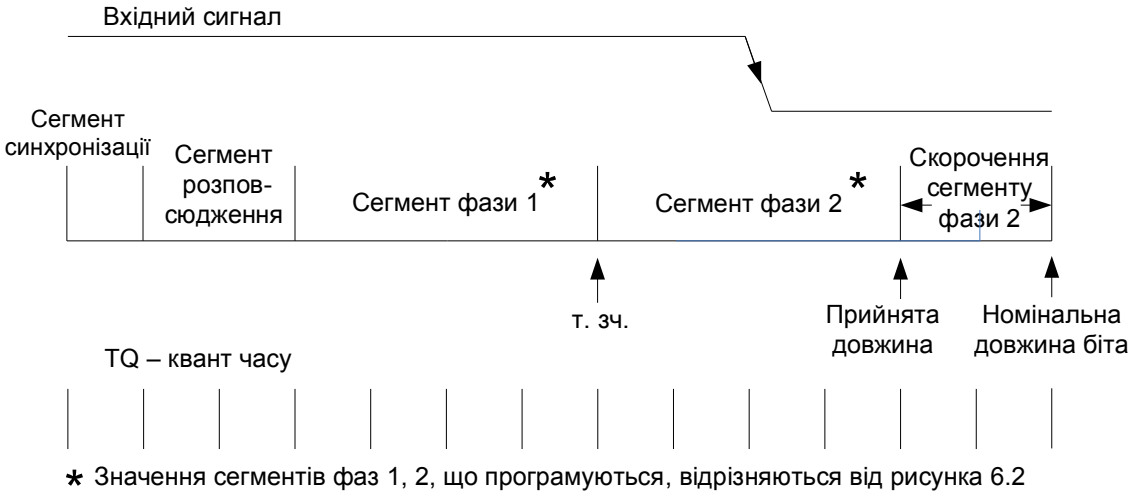


Рисунок 11.11 – Синхронізація з відновленням тактових інтервалів за рахунок автоматичного скорочення фазового сегмента (сегмента фази) 2

11.4 CAN–модуль мікроконтролера C8051F04x сімейства Silicon Labs

11.4.1 Загальна характеристика

Мікроконтролери, наприклад, C8051F40/1/2/3 сімейства Silicon Labs мають вбудований контролер локальної мережі (CAN), який дозволяє здійснювати послідовний обмін даними згідно CAN–протоколу. Даний CAN–контролер забезпечує роботу в CAN–мережі у відповідності зі специфікаціями

Bosch 2.0 A (базовий CAN (Controller Area Network)) і 2.0 B (розширений CAN).

Структурну схему CAN–контролера приведено на рисунку 11.12.

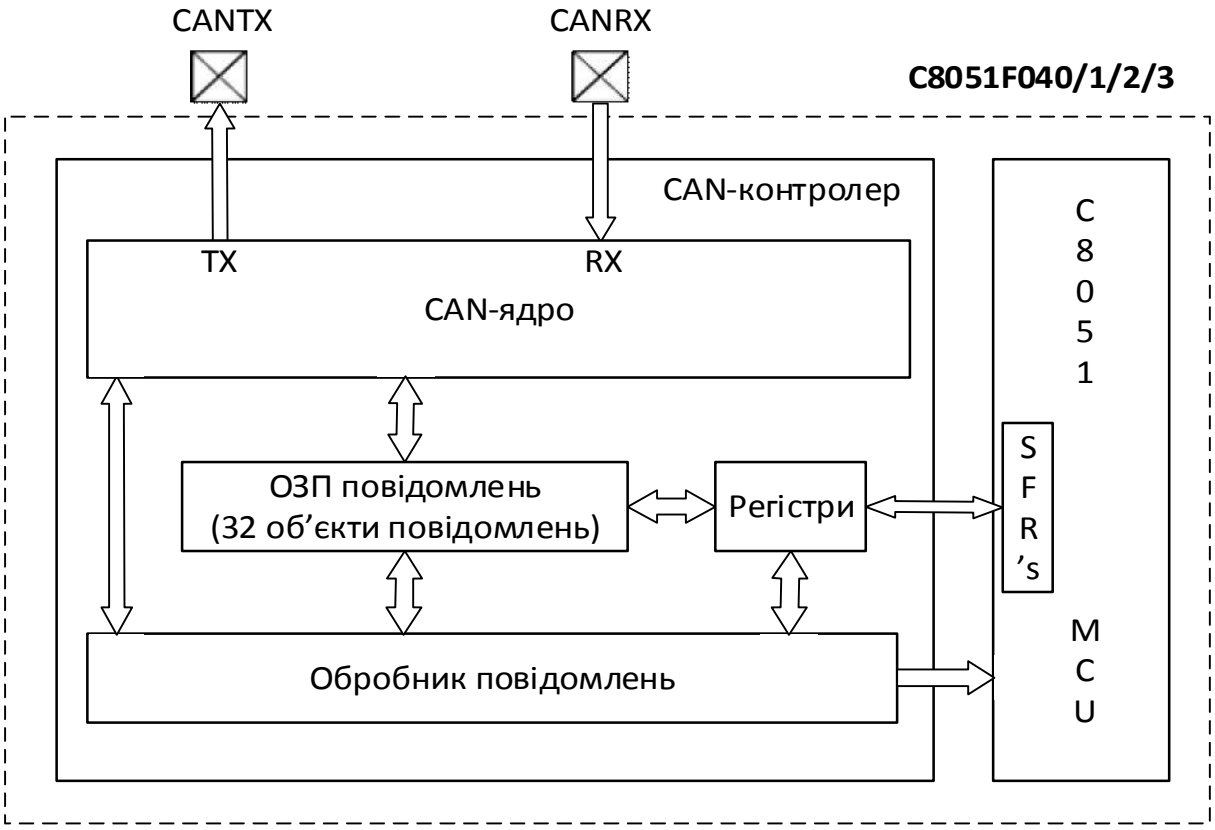


Рисунок 11.12 – Структурна схема CAN–контролера

CAN–контролер складається з CAN–ядра, ОЗП–повідомлень (окремого від ОЗП CIP–51), кінцевого автомата обробки повідомлень і регістрів керування. CAN–контролер Silicon Labs являє собою контролер CAN–протоколу і не має драйверів фізичного рівня (тобто трансиверів). Типову схему конфігурації CAN–мережі приведено на рисунку 11.13.

CAN–контролер Silicon Labs забезпечує асинхронний послідовний обмін даними зі швидкістю до 1Мбіт/сек., хоча ця швидкість може бути обмежена фізичним середовищем, яку обрано для передачі даних CAN–шиною.

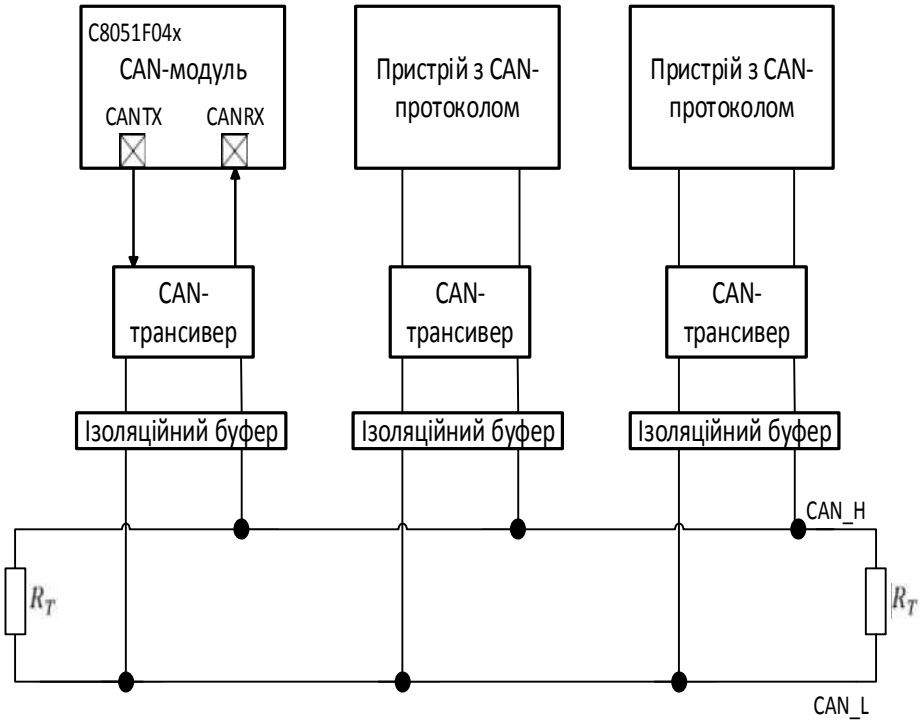


Рисунок 11.13 – Типова схема конфігурації CAN–мережі

Для кожного повідомлення, яке повинно бути передано або отримано, даний модуль містить один так званий об'єкт повідомлення MOb (рисунок 11.12). MOb – це дескриптор CAN–кадру. Він містить всю інформацію для роботи з CAN–кадром. Інакше кажучи, MOb був розроблений для опису CAN–кадру, як об'єкта.

CAN–контролер має 32 об`єкти повідомлень, які можуть бути налаштовані для передачі або прийому даних. Дані, що надходять, об`єкти повідомлень і маски ідентифікаторів зберігаються в ОЗП–повідомлень CAN–контролера. Всі функції протоколу, які пов'язано з передачею даних і фільтрацією повідомлень, що надходять, виконуються CAN–контролером, а не ядром CIP–51. Це забезпечує мінімальне навантаження на процесор при здійсненні взаємодії за CAN –протоколом. CIP–51 налаштовує CAN–контролер, приймає отримані дані і видає дані для передачі за допомогою регістрів спеціального призначення CAN–контролера.

11.4.2 Функціонування CAN – контролера

CAN–контролер, який реалізовано у МК C8051F04x сімейства Silicon Labs, являє собою повнофункціональний Bosch CAN–модуль і повністю відповідає специфікації CAN 2.0 B. На рисунку 11.12 наведено структурну схему CAN–контролера. CAN–ядро реалізує послідовну передачу/прийом (CANTX/CANRX) даних, перетворення форматів повідомлень (послідовний/паралельний), а також інші функції протоколу, такі як передачу даних і фільтрацію повідомлень, що надходять. ОЗП–повідомлень містить 32 об`єкти повідомлень, які можна передати в CAN–мережу, або заповнити даними з CAN–мережі. CAN–регістри і обробник повідомлень забезпечують інтерфейс обміну даними та повідомленнями між CAN–контролером і ядром CIP–51.

Функціонування та використання CAN–контролера мікроконтролерів сімейства C8051F04x, яке слід використовувати при налаштуванні CAN–контролера, докладно описано в [12]. В даному розділі описується, як забезпечити доступ до CAN–контролера.

Типова послідовність дій для ініціалізації CAN–контролера складається з наступних кроків:

Крок 1. Встановити в регістрі спеціального призначення SFRPAGE ядра CIP51 сторінку CAN0_PAGE.

Крок 2. Встановити у 1 біти INIT і CCE регістра керування CAN (CAN Control Register) [12].

Крок 3. Встановити часові параметри в регістрі BTR і в допоміжному регістрі BRP.

Крок 4. Ініціалізувати кожен об`єкт повідомлень або встановити його біт MsgVal в стан NOT VALID.

Крок 5. Скинути в 0 біт INIT.

Звернення до регістра керування CAN (CAN0CN), регістра тестування CAN (CAN0TST) і регістра стану CAN (CAN0STA) CAN–контролера можливо в режимах прямої або непрямої адресації за допомогою регістрів спеціального призначення (SFR) CIP–51. Всі інші CAN–регістри повинні адресуватися за допомогою методу непрямої адресації, який описано в підрозділі "Використання регістрів CAN0ADR, CAN0DATH і CAN0DATL для доступу до CAN–регістрів".

11.4.2.1 Тактування CAN–контролера

Частота тактування CAN–контролера (f_{CAN}) визначається системною тактовою частотою CIP–51 (SYSCLK). Між 8051 MCU та CAN–контролером знаходиться регістр CAN0CFG, який виконує функцію попереднього ділення системної частоти SYSCLK. Нижче наведено формат цього регістра та призначення його окремих розрядів. Для отримання тактової частоти модуля CAN_CLK частота після регістра CAN0CFG може ділитися за допомогою регістрів CAN–контролера: Bit Timing Register та BRP Extension Register. Нижче наведено формат регістра CAN0CFG (рисунок 11.14) та призначення його окремих розрядів (таблиця 11.3).

CAN clock configuration (налаштування генератора CAN)

SFR : CAN0CFG, SFR Адреса = 0x92; SFR Сторінка = 0x01.

Біт	7	6	5	4	3	2	1	0
Назва	–	–	–	–	–	–	SYSDIV[1:0]	
Тип	R	R	R	R	R	R	R/W	
Скид	0	0	0	0	0	0	0	0

Рисунок 11.14–Формат регістра CAN0CFG

Таблиця 11.3–Призначення окремих розрядів регістра CAN0CFG

Біт	Назва	Функція
7:2	Не використовуються	Зчитування=000000b; Запис – не важливо
1:0	SYSDIV[1:0]	Дільник системного тактового генератора Генератор CAN–контролера є похідним від системного генератора і має бути менше або дорівнювати 25 МГц. 00: Частота CAN = Системна частота/1 01: Частота CAN = Системна частота/2 10: Частота CAN = Системна частота/4 11: Частота CAN = Системна частота/8

Слід мати на увазі, що зазвичай потрібно використовувати зовнішній генератор (наприклад, генератор з кварцовим резонатором) для забезпечення високої точності часових параметрів, які визначаються специфікацією CAN–протоколу. Додаткову інформацію за цією темою наведено в [12].

11.4.2.2 Приклад розрахунку частоти тактування CAN–контролера для передачі даних зі швидкістю 1Мбіт/сек

Цей приклад показує, як налаштувати часові параметри CAN–контролера для обміну даними зі швидкістю 1Мбіт/сек.

В таблиці 11.4 приведено системні параметри, які необхідні для наведених нижче розрахунків.

Кожен біт, що передається за CAN–мережею, складається з 4–х сегментів (Sync_Seg, Prop_Seg, Phase_Seg1 та Phase_Seg2), як показано на рисунку 11.15.

Таблиця 11.4 – Основні системні параметри

Параметр	Значення	Пояснення
Системна тактова частота CIP–51 (SYSCLK)	22.1184 МГц	Зовнішній генератор функціонує в режимі з кварцовим резонатором. Кварцовий резонатор 22.1184 МГц підключений між виходами XTAL1 та XTAL2.
Частота тактування CAN–контролера (f_{sys})	22.1184 МГц	Визначається частотою SYSCLK.
Період тактування CAN–контролера (t_{sys})	45.211 нс	Визначається як $\frac{1}{f_{sys}}$
Період дискретизації CAN– контролера Примітка 3	45.211 нс	Дорівнює t_{sys} x (BRP +1) Примітки 1 та 2
Довжина CAN–шини	10 м	Затримка розповсюдження сигналу між CAN–вузлами складає 5 нс/м
Час затримки розповсюдження	400 нс	2 x (затримка петлі приймача–передавача + затримка лінії)

Примітка 1. Період дискретизації CAN–контролера (tq) являє собою мінімальний інтервал часу, який розпізнається CAN–контролером. Часові параметри часто визначаються в цілих одиницях періоду дискретизації.

Примітка 2. Попередній дільник швидкості (частоти) передачі даних (Baud Rate Prescaler – BRP) визначається як значення додаткового регістра плюс 1. Після скидання додатковий регістр BRP приймає значення 0x0000, а попередній дільник швидкості (частоти) передачі даних має значення 1.

Примітка 3. Визначається специфікацією ISO–11898 на приймачі–передавачі. CAN не визначає параметри фізичного рівня.

Рисунок 11.15 – Чотири сегменти бітового інтервалу CAN

Сума цих сегментів визначає бітовий інтервал CAN: $t_{біта}$=1/швидкість передачі в бітах за секунду.

В прикладі, який розглядається нижче, потрібно отримати швидкість передачі 1Мбіт/с, тому необхідний бітовий інтервал складає 1000 нс.

Ми будемо встановлювати довжину 4–х бітових сегментів таким чином, щоб їх сума як можна точніше відповідала необхідному бітовому інтервалу. Так як довжина кожного сегменту задається цілим числом періодів дискретизації, то найбільш точно потрібному значенню відповідає бітовий інтервал, що дорівнює 22 t_q (994.64 нс), якому відповідає швидкість передачі 1.00539 Мбіт/с. Prop_Seg не повинен бути менше часу розповсюдження (400 нс). Ми вибираємо його довжину 9 t_q (406.899 нс).

Інші кванти (t_q), що залишилися в бітовому інтервалі, діляться між сегментами Phase_Seg1 та Phase_Seg2:

Phase_Seg1 + Phase_Seg2 = Бітовий інтервал – (Sync_Reg + Prop_Seg)=

=22 *tq*–(1 *tq*+*9 tq*)=12 *tq*.

Якщо сума Phase_Seg1 + Phase_Seg2 парна, то Phase_Seg2 = Phase_Seg1. Інакше Phase_Seg2 = Phase_Seg1 + 1. Тривалість Phase_Seg2 повинна бути не менше *tq*.

Вибираємо Phase_Seg1 = 6 t_q та Phase_Seg2 = 6 t_q.

Ширина переходу пере синхронізації (Synchronization Jump Width – SJW) визначається із умови:

SJW = min (4, Phase_Seg1).

Цей параметр використовується для визначення значення, яке необхідно записати в регістр BTR, а також для визначення потрібної точності генератора. Так як в якості джерела системних тактових імпульсів ми використовуємо кварцовий генератор, то немає необхідності розраховувати точність генератора.

Значення ширини стрибка синхронізації (SJW), яке необхідно записати в регістр BTR, SJWp = SJW – 1 = min (4, 6) – 1 = 3.

Обчислимо значення, що записуються в регістр BTR:

TSEG1 = (Prop_Seg + Phase_Seg1 – 1) = 9 + 6 – 1 = 14,

TSEG2 = (Phase_Seg2 – 1) = 5.

Регістр BTR = (TSEG2 * 0x1000) + (TSEG1 * 0x0100) + (SJWp * 0x0040) + BRPE = 0x5EC0.

В додатковому регістрі BRP (BRP Extension Register) залишається значення 0x0000, що записується в нього після скидання.

Для ініціалізації CAN–регістрів часових параметрів необхідно виконати послідовність дій, що складається із наступних кроків:

Крок 1. Встановити в регістрі спеціального призначення SFRPAGE ядра CIP51 сторінку CAN0_PAGE.

Крок 2. Встановити в 1 біти INIT і CCE регістра керування CAN0CN.

Крок 3. Записати в регістр CAN0ADR значення 0x03, що вказує на індекс регістра BTR (таблиця 11.5).

Крок 4. Використовуючи метод непрямої адресації, щоб завантажити регістр BTR, записати значення 0x5EC0 в SFR–регістри [CAN0DATH:CAN0DATL].

Крок 5. Виконати інші дії для ініціалізації CAN–контролера.

11.4.2.3 CAN–регістри

CAN–регістри класифікуються наступним чином:

1. Регістри протоколу CAN–контролера: регістр керування, регістр переривання, регістр контролю помилок, регістр стану шини, регістр тестування.

2. Інтерфейсні регістри об`єктів повідомлень. Використовуються для налаштування 32 об`єктів повідомлень для передачі даних в об`єкти повідомлень і прийому даних з них. Процесорне ядро CIP–51 звертається до ОЗП–повідомлень за допомогою інтерфейсних регістрів об`єктів повідомлень. При запису номера об`єкта повідомлення в регістр запиту команди IF1 або IF2 вміст відповідних інтерфейсних регістрів (IF1 або IF2) буде передано в об`єкт повідомлень, що знаходиться в ОЗП CAN, або отримано з нього.

3. Регістри обробника повідомлень. Ці регістри доступні тільки для читання. Вони використовуються ядром CIP–51 для отримання інформації про об`єкти повідомлень (прапорці дійсності повідомлень, очікування запиту передачі, прапорці надходження нових даних, відкладених переривань (який з об`єктів повідомлень викликав переривання) та умова виникнення переривання).

4. Регістри спеціального призначення (SFR) ядра CIP–51. П'ять регістрів, які розташовано в пам'яті CIP–51, забезпечують прямий доступ до деяких регістрів протоколу CAN–контролера і індексний непрямий доступ до всіх CAN–регістрів.

11.4.2.4 Регістри протоколу CAN–контролера

Регістри протоколу CAN–контролера використовуються для налаштування CAN–контролера, обслуговування переривань, спостереження за станом шини та переведення контролера в тестові режими роботи. Доступ до регістрів протоколу CAN–контролера здійснюється через SFR–регістри CIP–51

за допомогою індексного методу. До деяких регістрів протоколу CAN–контролера можна звертатися безпосередньо шляхом адресації регістрів SFR в пам'яті CIP–51.

До регістрів протоколу CAN–контролера відносяться: регістр керування CAN (CAN0CN), регістр стану CAN (CAN0STA), регістр тестування CAN (CAN0TST), регістр лічильника помилок, регістр BTR і додатковий регістр BRP (регістр попереднього дільника швидкості передачі даних). До регістрів CAN0CN, CAN0STA і CAN0TST можна звертатися через SFR–регістри CIP–51. Доступ до всіх інших регістрів здійснюється опосередковано через регістри CAN0ADR, CAN0DATH і CAN0DATL за допомогою методу індексної адресації CAN.

Інформацію про функціонування регістрів керування CAN–протоколом та їх використання наведено в [12].

11.4.2.5 Інтерфейсні регістри об`єктів повідомлень

Є два набори інтерфейсних регістрів об`єктів повідомлень. Вони використовуються для налаштування 32 об`єктів повідомлень, які передають дані на CAN–шину і приймають дані з неї. Об`єкти повідомлень можна налаштувати для передачі або прийому даних, їм можна призначити ідентифікатори повідомлень, які використовуються для фільтрації повідомлень усіма вузлами CAN–мережі. Об`єкти повідомлень зберігаються в ОЗП–повідомлень. Доступ до них і їх налаштування здійснюються за допомогою інтерфейсних регістрів об`єктів повідомлень. Ці регістри доступні через регістри CIP–51: CAN0ADR, CAN0DATH і CAN0DATL за допомогою індексного методу непрямої адресації. Інформація про функціонування інтерфейсних регістрів об`єктів повідомлень та їх використання наведена в [12].

11.4.2.6 Регістри обробника повідомлень

Регістри обробника повідомлень доступні тільки для читання. Їх прапорці можна прочитати за допомогою індексного методу доступу через регістри: CAN0ADR, CAN0DATH і CAN0DATL. Регістри обробника повідомлень надають інформацію о перериваннях, помилках, запитах передачі/прийому і про оновлення даних. Інформацію про функціонування регістрів обробника повідомлень та їх використання наведено в [12].

11.4.2.7 Регістри спеціального призначення CIP–51

Периферійні модулі МК сімейства C8051F04x налаштовуються і керуються за допомогою регістрів спеціального призначення (SFR–регістрів). До більшості регістрів CAN–контролера не можна звертатися безпосередньо через SFR–регістри. Тільки три регістра CAN–контролера доступні безпосередньо через SFR–регістри. Доступ до всіх інших регістрів CAN–контролера здійснюється опосередковано за допомогою трьох регістрів SFR CIP–51: регістри даних CAN (CAN0DATH, рисунок 11.16 і CAN0DATL, рисунок 11.17) і регістр адреси CAN (CAN0ADR, рисунок 11.18). Таким чином, існує всього п'ять CAN–регістрів, які використовуються для налаштування і запуску CAN–контролера.

CAN0DAT регістри використовуються для читання/запису значень даних з/в CAN регістрів(–и), на які вказує порядковий номер, що міститься в регістрі CAN0ADR.

Назва регістра:		CAN0DATH – CAN Data Access Register High Byte					
SFR– адреса/сторінка:		0xD9/1	Значення після скидання:			00000000b (0x00)	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Рисунок 11.16 – CAN0DATH – Регістр доступу до старшого байта CAN–даних

Біти 7…0: CAN0DATH: Старший байт регістра даних CAN.

Назва регістра:	CAN0DATL – CAN Data Access Register Low Byte		
SFR– адреса/сторінка:	0xD8/1	Значення після скидання:	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Рисунок 11.17 – CAN0DATL – Регістр доступу до молодшого байта CAN–даних

Біти 7…0: CAN0DATL: Молодший байт регістра даних CAN.

Регістр CAN0ADR використовується для адресації регістрів даних [CAN0DATH:CAN0DATL] необхідного CAN–регістра.

Індексний номер потрібного CAN–регістра записується в регістр CAN0ADR. Після цього регістр CAN0DAT можна використовувати для запису/читання в/з CAN–регістр(–а).

Назва регістра:	CAN0ADR – CAN Address Index Register		
SFR– адреса/сторінка:	0xDA/1	Значення після скидання:	00000000b (0x00)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Рисунок 11.18 – CAN0ADR – Індексний регістр адреси CAN

Біти 7…0: CAN0ADR: Індексний номер регістра CAN–контролера.

Примітка. Якщо вміст регістра CAN0ADR відповідає діапазонам: 0x08… 0x12 і 0x20…0x2A (регістри IF1 та IF2), то цей регістр буде автоматично інкрементуватись при запису до регістра CAN0DATL.

11.4.2.8 Використання регістрів CAN0ADR, CAN0DATH і CAN0DATL для доступу до CAN–регістрів

Кожен регістр CAN–контролера має індексний номер (таблиця 11.5). Розмір адресного простору CAN–регістрів складає 128 слів (256 байт). CAN–регістр доступний через регістри даних CAN (CAN0DATH і CAN0DATL) тоді, коли індексний номер CAN–регістра поміщений у регістр адреси CAN (CAN0ADR). Наприклад, якщо потрібно завантажити в регістр BTR нове значення, необхідно в регістр CAN0ADR записати значення 0x03.

Після цього доступ до молодшого байта регістра BTR здійснюється через регістр CAN0DATL, а до старшого – через регістр CAN0DATH. Регістр CAN0DATL доступний в побітовому режимі адресації. Щоб завантажити значення 0x2304 в регістр BTR, необхідно: CAN0ADR = 0x03; // Завантажити індекс регістра BTR (таблиця 11.5), CAN0DATH = 0x23; // Завантажити старший байт значення в старший байт регістра даних, CAN0DATL = 0x04; // Завантажити молодший байт значення в молодший байт регістра даних.

Примітка: Доступ до регістрів CAN0CN, CAN0STA і CAN0TST можна здійснювати як за допомогою індексного методу, так і безпосередньо через SFR–регістри CIP–51. CAN0CN розташований за адресою 0xF8/SFR сторінка 1, CAN0TST розташований за адресою 0xDB/SFR сторінка 1, CAN0STA розташований за адресою 0xC0/SFR сторінка 1.

В цілях полегшення програмування об`єктів повідомлень регістр CAN0ADR наділений можливістю автоінкременту свого вмісту для індексів з діапазонів 0x08...0x12 (інтерфейсні регістри 1) і 0x20...0x2A (інтерфейсні регістри 2). Якщо регістр CAN0ADR містить індекс з цих діапазонів, то при

читанні/запису регістра CAN0DATL відбудеться автоінкремент вмісту регістра CAN0ADR на 1, після чого він буде вказувати на 16–розрядне слово наступного CAN–регістра. Це прискорює програмування часто змінюваних інтерфейсних регістрів під час налаштування об`єктів повідомлень.

Таблиця 11.5 – Індекси регістрів CAN

Індекси CAN–регістрів	Назва регістра	Значення після скидання	Примітка
0x00	CAN Control Register	0x0001	Доступний через SFR–регістр
0x01	Status Register	0x0000	Доступний через SFR–регістр
0x02	Error Counter[1]	0x0000	Доступний тільки для читання
0x03	Bit Timing Register[2]	0x2301	Запис дозволено бітом CCE регістра CAN0CN
0x04	Interrupt Register[1]	0x0000	Доступний тільки для читання
0x05	Test Register	0x0000	Біт 7 (RX), визначається шиною CAN
0x06	BRP Extension Register[2]	0x0000	Запис дозволено бітом TEST регістра CAN0CN
0x08	IF1 Command Request	0x0001	CAN0ADR автоінкрементується у індексному діапазоні IF1 (0x08…0x12) при запису в CAN0DATL
0x09	IF1 Command Mask	0x0000	CAN0ADR автоінкрементується при запису в CAN0DATL
0x0A	IF1 Mask 1	0xFFFF	CAN0ADR автоінкрементується при запису в CAN0DATL
0x0B	IF1 Mask 2	0xFFFF	CAN0ADR автоінкрементується при запису в CAN0DATL
0x0C	IF1 Arbitration 1	0x0000	CAN0ADR автоінкрементується при запису в CAN0DATL
0x0D	IF1 Arbitration 2	0x0000	CAN0ADR автоінкрементується при запису в CAN0DATL

Продовження таблиці 11.5

0x0E	IF1 Message Control	0x0000	CAN0ADR автоінкрементується при запису в CAN0DATL
0x0F	IF1 Data A 1	0x0000	CAN0ADR автоінкрементується при запису в CAN0DATL
0x10	IF1 Data A 2	0x0000	CAN0ADR автоінкрементується при запису в CAN0DATL
0x11	IF1 Data B 1	0x0000	CAN0ADR автоінкрементується при запису в CAN0DATL
0x12	IF1 Data B 2	0x0000	CAN0ADR автоінкрементується при запису в CAN0DATL
0x20	IF2 Command Request	0x0001	CAN0ADR автоінкрементується у індексному діапазоні IF2 (0x20…0x2A) при запису в CAN0DATL
0x21	IF2 Command Mask	0x0000	CAN0ADR автоінкрементується при запису в CAN0DATL
0x22	IF2 Mask 1	0xFFFF	CAN0ADR автоінкрементується при запису в CAN0DATL
0x23	IF2 Mask 2	0xFFFF	CAN0ADR автоінкрементується при запису в CAN0DATL
0x24	IF2 Arbitration 1	0x0000	CAN0ADR автоінкрементується при запису в CAN0DATL
0x25	IF2 Arbitration 2	0x0000	CAN0ADR автоінкрементується при запису в CAN0DATL
0x26	IF2 Message Control	0x0000	CAN0ADR автоінкрементується при запису в CAN0DATL
0x27	IF2 Data A 1	0x0000	CAN0ADR автоінкрементується при запису в CAN0DATL
0x28	IF2 Data A 2	0x0000	CAN0ADR автоінкрементується при запису в CAN0DATL
0x29	IF2 Data B 1	0x0000	CAN0ADR автоінкрементується при запису в CAN0DATL
0x2A	IF2 Data B 2	0x0000	CAN0ADR автоінкрементується при запису в CAN0DATL

Продовження таблиці 11.5

0x40	Transmission Request 1[1]	0x0000	Прапорці запитів передачі для об`єктів повідомлень (тільки читання)
0x41	Transmission Request 2[1]	0x0000	Прапорці запитів передачі для об`єктів повідомлень (тільки читання)
0x48	New Data 1[1]	0x0000	Прапорці нових даних для об`єктів повідомлень (тільки читання)
0x49	New Data 2[1]	0x0000	Прапорці нових даних для об`єктів повідомлень (тільки читання)
0x50	Interrupt Pending 1[1]	0x0000	Прапорці очікування переривань для об`єктів повідомлень (тільки читання)
0x51	Interrupt Pending 2 [1]	0x0000	Прапорці очікування переривань для об`єктів повідомлень (тільки читання)
0x58	Message Valid 1[1]	0x0000	Прапорці достовірності повідомлень для об`єктів повідомлень (тільки читання)
0x59	Message Valid 2[1]	0x0000	Прапорці достовірності повідомлень для об`єктів повідомлень (тільки читання)

Примітки:

1. Тільки для читання регістра.
2. Дозвіл запису: CCE=1.
3. При скиданні в CAN0TST може бути : r0000000b, де r відслідковує значення на лінії CAN RX.
4. Запис дозволено бітом Test.

11.4.3 Опис CAN–регістрів

Нижче розглядаються три групи CAN–регістрів: регістри протоколу CAN–контролера (CAN Protocol Related Registers): CAN Control Register, Status Register, Error Counter, Bit Timing Register, Test Register, BRP Extension Register; інтерфейсні регістри об`єктів повідомлень (Message Interface Registers Sets): IFx Command Request Registers, IFx Command Mask Registers, IFx Message Buffer Registers (IFx Mask Registers, IFx Arbitration Registers, IFx Message Control Registers, IFx Data A and Data B Registers) та регістри обробника повідомлень (Message Handler Registers): Interrupt Register, Transmission Request Registers, New Data Registers, Interrupt pending Registers, Message Valid 1 Register.

11.4.3.1 Регістри протоколу CAN–контролера (CAN Protocol Related Registers)

11.4.3.1.1 CAN Control Register

На рисунку 11.19 наведено формат регістра, а у таблиці 11.6 наведено опис окремих бітів.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
res	**res**	**res**	**res**	**res**	**res**	**res**	**res**	**Test**	**CCE**	**DAR**	**res**	**EIE**	**SIE**	**IE**	**Init**
r	r	r	r	r	r	r	r	rw	rw	rw	r	rw	rw	rw	rw

Рисунок 11.19 – CAN Control Register (регістр керування)

Таблиця 11.6 – Опис бітів CAN Control Register

Біт	Назва	Функція
7	Test	Вмикання тестового режиму: 1: Тестовий режим, 0: Звичайне функціонування.
6	CCE	Дозвіл зміни конфігурації (conf change enable): 1: CPU може записувати в Bit Timing Register (Init = 1), 0: CPU не може записувати в Bit Timing Register.
5	DAR	Відключення автоматичної ретрансляції: 1: Автоматична ретрансляція відключена, 0: Автоматична ретрансляція активна.
3	EIE	Дозвіл переривання за помилками:1: Зміна бітів BOff або EWarn в регістрі стану згенерує переривання.0: Жодних переривань за помилковими станами не буде.
2	SIE	Дозвіл переривань за зміною стану1: Переривання відбудеться після успішної передачі або у випадку виявлення помилки шини, 0: Жодних переривань за зміною стану не буде.
1	IE	Дозвіл переривань CAN–модуля: 1: Переривання встановлять IRQ_B в LOW. IRQ_B залишатиметься в стані LOW поки усі переривання, які очікують, не будуть опрацьовані, 0: Модульне переривання IRQ_B завжди в стані HIGH.

Продовження таблиці 11.6

Біт	Назва	Функція
0	Init	Ініціалізація: 1: Ініціалізація почалась, 0: Звичайне функціонування.

11.4.3.1.2 Status Register

На рисунку 11.20 наведено формат регістра, а у таблиці 11.7 наведено опис окремих бітів.

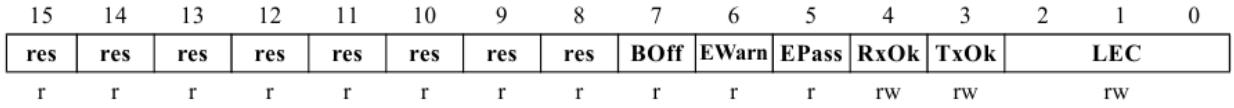


Рисунок 11.20 – CAN Status Register (регістр стану)

Таблиця 11.7 – Опис бітів CAN Status Register

Біт	Назва	Функція
7	BOff	Стан busoff (шина відключена): 1: Модуль CAN знаходиться в стані busoff, 0: Модуль CAN знаходиться не в стані busoff.
6	EWarn	Попередження про помилку: 1: Принаймні один з лічильників помилок в EML досяг попереджувальної межі в 96, 0: Усі лічильники менше попереджувальної межі в 96.

Продовження таблиці 11.7

Біт	Назва	Функція
5	EPass	1: Ядро CAN в стані error passive (пасивної помилки), 0: Ядро CAN в стані error active (активної помилки).
4	RxOk	Успішний прийом повідомлення: 1: Відколи цей біт був скинутий процесором останнього разу в 0, повідомлення було успішно прийнято (незалежно від результатів фільтрування), 0: Відколи цей біт було останнього разу скинуто процесором, жодне повідомлення не було успішно прийнято (цей біт ніколи не скидається ядром CAN).
3	TxOk	Успішна передача повідомлення: 1: Відколи цей біт було останнього разу скинуто процесором, повідомлення було успішно відправлено (без помилок і підтверджено хоча б одним приймачем), 0: Відколи цей біт було останнього разу скинуто процесором, жодне повідомлення не було успішно передано (цей біт ніколи не скидається ядром CAN).

Продовження таблиці 11.7

Біт	Назва	Функція
2…0	LEC	Тип останньої помилки в шині: 0: Жодних помилок, 1: Stuff Error – у отриманому повідомленні виявлено послідовність з більш ніж 5 однакових бітів, там де це не дозволено, 2: Form Error – частина отриманого кадру з визначеним форматом має невірний формат, 3: AckError – Повідомлення відправлене цим ядром CAN не було підтверджене жодним вузлом, 4: Bit1Error – під час передачі повідомлення (за виключенням поля арбітражу), пристрій хотів відправити рецесивний рівень (біт з логічним значенням ‘1’), але значення в шині було домінантне, 5: Bit0Error – під час передачі повідомлення (або біта підтвердження, або прапорця активної помилки, або прапорця перевантаження) пристрій хотів відправити домінантний рівень (біт з логічним значенням ‘0’), але значення в шині було рецесивне. Під час відновлення після busoff цей статус встановлюється кожного разу після детектування послідовності з 11 рецесивних бітів. Це дозволяє процесору монітори проходження послідовностей під час відновлення після busoff (це вказує на те, що шина не застрягла на домінантному значені або постійно відбуваються помилки), 6: CRCError – CRC–контрольна сума в отриманому повідомленні невірна, отримана та підрахована суми не співпадають, 7: не використовується – якщо LEC має значення ‘7’, на шині не було помічено жодних подій з моменту останнього запису процесором значення в LEC.

В полі LEC зберігається код, що вказує на тип останньої помилки, що виникла в шині. Значення поля буде скинуто в ‘0’ після трансферу (прийому або передачі) повідомлення без помилок. Код ‘7’, що не використовується, може бути записаний процесором для перевірки оновлення стану.

11.4.3.1.2.1 Переривання стану

Переривання стану генеруються бітами BOff і EWarn (Error Interrupt) або RxOk, TxOk, і LEC (Status Change Interrupt), припускаючи, що відповідні біти дозволу в CAN Control Register встановлені. Зміна біту EPass або запис до RxOk, TxOk або LEC ніколи не згенерує переривання стану.

Зчитування Status Register скине значення Status Interrupt (8000h) в Interrupt Register, якщо воно очікує обробки.

11.4.3.1.3 Error Counter

На рисунку 11.21 наведено формат регістра, а у таблиці 11.8 наведено опис окремих бітів.

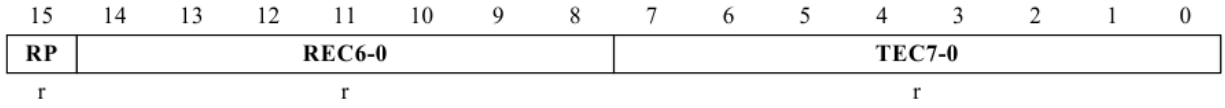


Рисунок 11.21 – Error Counter

Таблиця 11.8 – Опис бітів Error Counter

Біт	Назва	Функція
15	RP	1: Лічильник помилок прийому досяг значення ‘error passive’(≥128), 0: Лічильник помилок прийому нижче рівня ‘error passive’.

Продовження таблиці 11.8

Біт	Назва	Функція
14…8	REC6…0	Лічильник помилок прийому: Поточний стан лічильника, від 0 до 127.
7…0	TEC7…0	Лічильник помилок передачі: Поточний стан лічильника, від 0 до 255.

11.4.3.1.4 Bit Timing Register

На рисунку 11.22 наведено формат регістра, а у таблиці 11.9 наведено опис окремих бітів.

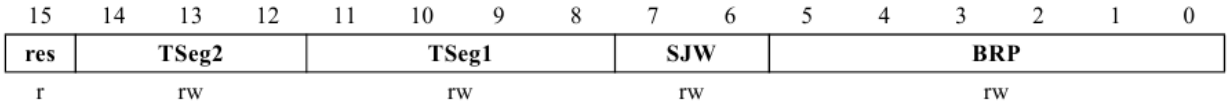


Рисунок 11.22 – Bit Timing Register

Таблиця 11.9 – Опис бітів Bit Timing Register

Біт	Назва	Функція
11…8	TSeg1	Сегмент часу до точки вибірки (sample point): дійсними значеннями для TSeg1 є: 0x01…0x0F. Апаратними засобами буде використовуватись значення, на 1 більше ніж записано тут.
14…12	TSeg2	Сегмент часу після точки вибірки (sample point): дійсними значеннями для TSeg2 є: 0x00…0x07. Апаратними засобами буде використовуватись значення, на 1 більше ніж записано тут.

Продовження таблиці 11.9

Біт	Назва	Функція
7…6	SJW	Ширина періоду пере синхронізації: дійсними значеннями для SJW є: 0x0…0x3. Апаратними засобами буде використовуватись значення, на 1 більше ніж записано тут.
5…0	BRP	Дільник швидкості передачі: дійсними значеннями для BRP є: 0x01…0x3F. Це значення, на яке ділиться частота осцилятора для генерація часового кванту біта. Час біта отримується множенням цього кванту. Апаратними засобами буде використовуватись значення, на 1 більше ніж записано тут.

Для тактової частоти модуля CAN_CLK 8 МГц значення в цьому регістрі після скидання 0x2301 налаштує швидкість передачі CAN–500 КБіт/с. Регістри доступні лише для запису, якщо біти CCE та Init в CAN Control Register встановлено в 1.

11.4.3.1.5 Test Register

На рисунку 11.23 наведено формат регістра, а у таблиці 11.10 наведено опис окремих бітів.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
res	res	res	res	res	res	res	res	Rx	Tx1	Tx0	LBack	Silent	Basic	res	res
r	r	r	r	r	r	r	r	r	rw	rw	rw	rw	rw	r	r

Рисунок 11.23 – Test Register

Таблиця 11.10 – Опис бітів Test Register

Біт	Назва	Функція
7	RX	Відслідковує поточний стан піну CAN_RX: 1: Шина в рецесивному стані (CAN_RX = ‘1’), 0: Шина в домінантному стані (CAN_RX = ‘0’).
6:5	TX1…0	Керування стану піну CAN_TX: 00: Значення скидання, CAN_TX контролюється ядром, 01: Sample point може контролюється на піні CAN_TX, 10: пін видає домінантне значення (‘0’). 11: пін видає рецесивне значення (‘1’).
4	LBack	Режим закільцьовування: 1: Режим закільцьовування ввімкнено, 0: Режим закільцьовування вимкнено.
3	Silent	Режим мовчання: 1: Модуль в режимі мовчання, 0: Модуль в звичайному режимі.
2	Basic	Базовий режим: 1: Регістри IF1 використовуються як буфер Tx, регістри IF2 використовуються як буфер Rx, 0: Базовий режим вимкнений.

Доступ для запису до Test Register вмикається встановленням біта Test в CAN Control Register. Різні тестові функції можуть об`єднані, але TX1…0: ‘00’ припиняє передачу повідомлень.

11.4.3.1.6 BRP Extension Register

На рисунку 11.24 наведено формат регістра, а у таблиці 11.11 наведено опис окремих бітів.

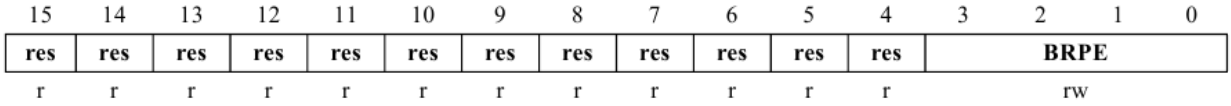


Рисунок 11.24 – BRP Extension Register

Таблиця 11.11 – Опис бітів BRP Extension Register

Біт	Назва	Функція
3…0	BRPE	Розширення дільника швидкості передачі Програмуючи BRPE, дільник швидкості передачі можна збільшити до 1023. Апаратними засобами буде використовуватись значення, на 1 більше ніж записано тут.

11.4.3.2 Інтерфейсні регістри об`єктів повідомлень (Message Interface Registers Sets)

11.4.3.2.1 IFx Command Request Registers

Передача повідомлення починається відразу після того, як процесор запише номер повідомлення в Command Request Register. Одночасно з цією операцією біт ‘Busy’ автоматично встановлюється в ‘1’ і сигнал CAN_WAIT_B притягується до низького рівня, щоб повідомити процесору, що передача

триває. Після очікування від 3 до 6 періодів CAN_CLK передача між інтерфейсним регістром та оперативною пам’яттю повідомлень завершена. Біт ‘Busy’ скидається в ‘0’ і сигнал CAN_WAIT_B повертається до високого рівня.

На рисунку 11.25 наведено формат регістрів, а у таблиці 11.12 наведено опис окремих бітів.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IF1 Command Request Register (addresses 0x11 & 0x10)	**Busy**	**res**	**res**	**res**	**res**	**res**	**res**	**res**	**res**	**res**	**Message Number**					
IF2 Command Request Register (addresses 0x41 & 0x40)	**Busy**	**res**	**res**	**res**	**res**	**res**	**res**	**res**	**res**	**res**	**Message Number**					
	r	r	r	r	r	r	r	r	r	r	rw					

Рисунок 11.25 – IFx Command Request Registers

Таблиця 11.12 – Опис бітів IFx Command Request Registers

Біт	Назва	Функція
15	Busy	1: Встановлюється в одиницю під час запису до IFx Command Request Register, 0: Скидається до нуля, коли операція запису/зчитування завершена.
5…0	Message number	0x01…0x20–Дійсний номер повідомлення, об’єкт повідомлення в оперативній пам’яті повідомлень обрано для обміну, 0x00 – Невірний номер повідомлення, інтерпретується як 0x20, 0x21…0x3F – Невірний номер повідомлення, інтерпретується як 0x01…0x1F.

Якщо буде записано невірний номер, він буде автоматично трансформований до правильного вигляду і об’єкт повідомлення буде переданий.

11.4.3.2.2 IFx Command Mask Registers

Керуючі біти IFx Command Mask Register визначають напрям передачі і визначають який з IFx Message Buffer Registers є джерелом або ціллю передачі.

На рисунку 11.26 наведено формат регістрів, а у таблиці 11.13 наведено опис окремих бітів.

	15 14 13 12 11 10 9 8	7	6	5	4	3	2	1	0
IF2 Command Mask Register (addresses 0x13 & 0x12)	**res**	**WR/RD**	**Mask**	**Arb**	**Control**	**ClrIntPnd**	**TxRqst/ NewDat**	**Data A**	**Data B**
IF2 Command Mask Register (addresses 0x43 & 0x42)	**res**	**WR/RD**	**Mask**	**Arb**	**Control**	**ClrIntPnd**	**TxRqst/ NewDat**	**Data A**	**Data B**
	r r r r r r r r	rw	rw	rw	rw	rw	rw	rw	rw

Рисунок 11.26 – IFx Command Mask Registers

Таблиця 11.13 – Опис бітів IFx Command Mask Registers

Біт	Назва	Функція
7	WR/RD	1: Запис – передача даних з обраного Message Buffer Register в об’єкт повідомлення за адресою в Command Request Register, 0: Зчитування – передача даних з об’єкта повідомлень за адресою з Command Request Register в обраний Message Buffer Register.
Решта бітів IFx Command Mask Registers мають різні функції в залежності від напрямку передачі: <u>Напрямок = запис</u>		

Продовження таблиці 11.13

Біт	Назва	Функція
6	Mask	Доступ до бітів маски: 1: Передача Identifier Mask + Mdir + MXtd до об'єкту повідомлення, 0: Біти маски незмінні.
5	Arb	Доступ до бітів арбітражу: 1: Передача Identifier + Dir + Xtd + MsgVal до об'єкту повідомлення, 0: Біти арбітражу незмінні.
4	Control	Доступу до бітів контролю: 1: Передача бітів контролю до об'єкту повідомлення, 0: Біти контролю незмінні.
3	ClrIntPnd	Очистити біт очікування переривання. Під час запису до об'єкту повідомлення цей біт ігнорується.
2	TxRqst/ NewDat	Доступ до біту запиту передачі: 1: Встановлює біт TxRqst, 0: Біт TxRqst залишається незмінним. Якщо передача запитується програмуванням біту TxRqst в IFx Command Mask Register, то біт TxRqst в IFx Message Control Register ігнорується.

Продовження таблиці 11.13

Біт	Назва	Функція
1	Data A	Доступ до бітів даних 0…3: 1: Передача бітів даних 0…3 до об'єкту повідомлення, 0: Біти даних 0…3 незмінні.
0	Data B	Доступ до бітів даних 4…7 1: Передача бітів даних 4…7 до об'єкту повідомлення. 0: Біти даних 4…7 незмінні.
Напрямок = зчитування		
6	Mask	Доступ до бітів маски: 1: Передача Identifier Mask + Mdir + MXtd до регістра IFx Message Buffer, 0: Біти маски незмінні.
5	Arb	Доступ до бітів арбітражу: 1: Передача Identifier + Dir + Xtd + MsgVal до регістра IFx Message Buffer, 0: Біти арбітражу незмінні.
4	Control	Доступу до бітів контролю: 1: Передача бітів контролю до IFx Message Buffer, 0: Біти контролю незмінні.

Продовження таблиці 11.13

Біт	Назва	Функція
3	ClrIntPnd	Очистити біт очікування переривання: 1: Очистити біт IntPnd в об'єкті повідомлення, 0: Біт IntPnd не змінюється.
2	TxRqst/ NewDat	Доступ до біта нових даних: 1: Очистити біт NewDat в об'єкті повідомлення, 0: Біт NewDat залишається незмінним. Доступ для читання об'єкту повідомлення може бути поєднаний зі скиданням бітів контролю IntPnd та NewDat. Значення цих бітів, що передається до IFx Message Control Register завжди відображає стан цих бітів до скидання.
1	Data A	Доступ до бітів даних 0…3 1: Передача бітів даних 0…3 до IFx Message Buffer Register. 0: Біти даних 0…3 незмінні.
0	Data B	Доступ до бітів даних 4…7 1: Передача бітів даних 4…7 до IFx Message Buffer Register. 0: Біти даних 4…7 незмінні.

11.4.3.2.3 IFx Message Buffer Registers

Буферні об`єкти повідомлень відображають об`єкти повідомлень у оперативній пам`яті повідомлень. Нижче описано функції цих регістрів.

11.4.3.2.3.1 IFx Mask Registers

На рисунку 11.27 наведено формат регістрів: IFx Mask Registers.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IF1 Mask 1 Register (addresses 0x15 & 0x14)	**Msk15-0**															
IF1 Mask 2 Register (addresses 0x17 & 0x16)	**MXtd**	**MDir**	**res**	**Msk28-16**												
IF2 Mask 1 Register (addresses 0x45 & 0x44)	**Msk15-0**															
IF2 Mask 2 Register (addresses 0x47 & 0x46)	**MXtd**	**MDir**	**res**	**Msk28-16**												
	rw	rw	r	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Рисунок 11.27 – IFx Mask Registers

11.4.3.2.3.2 IFx Arbitration Registers

На рисунку 11.28 наведено формат регістрів: IFx Arbitration Registers.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IF1 Arbitration 1 Register (addresses 0x19 & 0x18)	**ID5-0**															
IF1 Arbitration 2 Register (addresses 0x1B & 0x1A)	**MsgVal**	**Xtd**	**Dir**	**ID28-16**												
IF2 Arbitration 1 Register (addresses 0x49 & 0x48)	**ID15-0**															
IF2 Arbitration 2 Register (addresses 0x4B & 0x4A)	**MsgVal**	**Xtd**	**Dir**	**ID28-16**												
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Рисунок 11.28 – IFx Arbitration Registers

11.4.3.2.3.3 IFx Message Control Registers

На рисунку 11.29 наведено формат регістрів: IFx Message Control Registers.

	15	14	13	12	11	10	9	8	7	6	5	4	3 2 1 0
IF1 Message Control Register (addresses 0x1D & 0x1C)	**NewDat**	**MsgLst**	**IntPnd**	**UMask**	**TxIE**	**RxIE**	**RmtEn**	**TxRqst**	**EoB**	**res**	**res**	**res**	**DLC3-0**
IF2 Message Control Register (addresses 0x4D & 0x4C)	**NewDat**	**MsgLst**	**IntPnd**	**UMask**	**TxIE**	**RxIE**	**RmtEn**	**TxRqst**	**EoB**	**res**	**res**	**res**	**DLC3-0**
	rw	rw	rw	rw	rw	rw	rw	rw	rw	r	r	r	rw

Рисунок 11.29 – IFx Message Control Registers

11.4.3.2.3.4 IFx Data A та Data B Registers

Біти даних повідомлень CAN зберігаються в IFx Message Buffer Registers в наступному порядку (рисунок 11.30):

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IF1 Message Data A1 (addresses 0x1F & 0x1E)	**Data(1)**											**Data(0)**				
IF1 Message Data A2 (addresses 0x21 & 0x20)	**Data(3)**											**Data(2)**				
IF1 Message Data B1 (addresses 0x23 & 0x22)	**Data(5)**											**Data(4)**				
IF1 Message Data B2 (addresses 0x25 & 0x24)	**Data(7)**											**Data(6)**				
IF2 Message Data A1 (addresses 0x4F & 0x4E)	**Data(1)**											**Data(0)**				
IF2 Message Data A2 (addresses 0x51 & 0x50)	**Data(3)**											**Data(2)**				
IF2 Message Data B1 (addresses 0x53 & 0x52)	**Data(5)**											**Data(4)**				
IF2 Message Data B2 (addresses 0x55 & 0x54)	**Data(7)**											**Data(6)**				
	rw											rw				

Рисунок 11.30 – IFx Data A та Data B Registers

В кадрі даних CAN Data(0) є першим, Data(7) є останнім бітом, який буде передано чи прийнято. В послідовному потоці бітів CAN, MSB кожного байта буде передано першим.

11.4.3.2.4 Message Object (об’єкт повідомлення) в пам’яті повідомлень

В оперативній пам’яті повідомлень знаходяться 32 об’єкти повідомлень. Щоб уникнути конфлікту між доступом процесора до оперативної пам’яті повідомлень та відправленням і прийомом повідомлень, процесор не має прямого доступу до неї, доступ до об’єктів повідомлень здійснюється через IFx Interface Registers.

Структуру об’єкту повідомлення в оперативній пам’яті повідомлень наведено на рисунку 11.31. Функції окремих бітів описано в таблиці 11.14.

Message Object												
UMask	Msk28-0	MXtd	MDir	EoB	NewDat		MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
MsgVal	ID28-0	Xtd	Dir	DLC3-0	Data 0	Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7

Рисунок 11.31 – Message Object

Таблиця 11.14 – Опис бітів об’єкту повідомлення (Message Object)

Назва	Функція
MsgVal	Повідомлення дійсне: 1: Об’єкт повідомлення налаштовано і він має бути розглянутий обробником повідомлень, 0: Об’єкт повідомлення ігнорується обробником повідомлень. Під час ініціалізації процесор має скинути біт MsgVal усіх об’єктів повідомлень, що не використовуються, до того як він скине біт Init в CAN Control Register. Також цей біт має бути скинутий до ідентифікатору Id28…0, бітів контролю Xtd, Dir, або зміни довжини коду даних: DLC3…0, або якщо об’єкт повідомлення більше не потрібен.
UMask	Використання маску прийому: 1: Використовувати маску (Msk28…0, MXtd, та MDir) для фільтрації, 0: Маска ігнорується. Якщо цей біт встановлено в 1, біти маски об’єкта повідомлення мають бути запрограмовані під час ініціалізації об’єкту повідомлення до того як MsgVal буде встановлено в 1.
ID28…0	Ідентифікатор повідомлення: ID28 … ID0─29–ти бітний індикатор (“Extended Frame”), ID28…ID18─11–ти бітний індикатор (“Standard Frame”).

Продовження таблиці 11.14

Назва	Функція
Msk28…0	Маска ідентифікатора: 1: Відповідний біт ідентифікації використовується для фільтрації, 0: Значення відповідного біта в ідентифікаторі об'єкту повідомлення при фільтрації не аналізується.
Xtd	Біт розширеного ідентифікатору: 1: для цього об'єкту повідомлення буде використано 29–ти бітний ідентифікатор, 0: для цього об'єкту повідомлення буде використано 11–ти бітний ідентифікатор
MXtd	Маска біта розширеного ідентифікатору: 1: Біт розширеного ідентифікатору використовуватиметься при фільтрації, 0: Біт розширеного ідентифікатору не матиме впливу при фільтрації. Якщо для ідентифікації об'єктів повідомлень використовуються стандартні 11…бітні ідентифікатори, ідентифікатори отриманого кадру даних записуються в біти з ID28 по ID18. Для фільтрації будуть використані лише ці біти разом з бітам маски з Msk28 по Msk18.

Продовження таблиці 11.14

Назва	Функція
Dir	Напрямок повідомлення: 1: Напрямок – передача: При встановленні TxRqst відповідний об’єкт повідомлення передається як кадр даних., 0: Напрямок – прийом: При прийомі кадру віддаленого запиту з відповідним ідентифікатором у цьому об’єкті повідомлення встановлюється біт TxRqst, якщо біт RmtEn =1. При прийомі кадру даних з відповідним ідентифікатором повідомлення зберігається в цьому об’єкті повідомлення.
MDir	Маска біта напрямку повідомлення: 1: При прийомі відбувається фільтрація біта напрямку повідомлення (Dir), 0: При прийомі не відбувається фільтрація біта напрямку повідомлення (Dir). Регістри арбітражу: ID28…0, Xtd, та Dir використовуються для визначення ідентифікатору і типу вихідного повідомлення і використовуються разом з регістрами маски Msk28…0, MXtd, та MDir для фільтрації вхідних повідомлень. Отримане повідомлення зберігається в дійсному об’єкті повідомлення з відповідним ідентифікатором і напрямком – прийом (кадр даних) або напрямком – передача (віддалений кадр). Розширені фрейми можуть бути збережені лише в об’єктах повідомлень з Xtd = 1. Якщо отримане повідомлення (кадр даних або віддалений кадр) відповідає більш ніж одному дійсному об’єкту повідомлення, то він зберігається в тому об’єкті повідомлення, в яке має найменший номер.

Продовження таблиці 11.14

Назва	Функція
EoB	Кінець буфера: 1: Одиночний об’єкт повідомлення або останній об’єкт повідомлення з буферу FIFO, 0: Об’єкт повідомлення належить буферу FIFO і це не останній об’єкт повідомлення з цього буфера. Цей біт використовується для поєднання двох та більше об’єктів повідомлень (до 32) для побудови буфера FIFO. Для одиночного об’єкту повідомлення (того, що не належить буферу FIFO) цей біт має бути завжди встановлений в одиницю.
NewDat	Нові дані: 1: Обробник повідомлень або CPU записав нові дані в розділ даних цього об’єкту повідомлення, 0: Жодних нових даних не було записано в розділі даних цього об’єкту повідомлення обробником повідомлень з моменту останнього скидання цього прапорця процесором.
MsgLst	Втрата повідомлення (лише для дійсних об’єктів повідомлень з напрямком – прийом): 1: Обробник повідомлень записав нове повідомлення в цей об’єкт повідомлення, хоча NewDat = 1, процесор втратив попереднє повідомлення, 0: Жодних повідомлень не було втрачено з моменту останнього скидання цього прапорця процесором.

Продовження таблиці 11.14

Назва	Функція
RxIE	Дозвіл переривання за прийомом: 1: прапорець IntPnd буде встановлений після успішного прийому кадру, 0: прапорець IntPnd не буде змінений після успішного прийому кадру.
TxIE	Дозвіл переривання за передачею: 1: прапорець IntPnd буде встановлений після успішної передачі кадру, 0: прапорець IntPnd не буде змінений після успішної передачі кадру.
IntPnd	Переривання, що очікує: 1: Цей об’єкт повідомлення є джерелом переривання. Ідентифікатор переривання в регістрі переривань вкаже на цей об’єкт повідомлення, якщо немає джерел переривань з більш високим пріоритетом, 0: Цей об’єкт повідомлення не є джерелом переривання.

Продовження таблиці 11.14

Назва	Функція
TxRqst	Запит передачі: 1: Передача цього об'єкту повідомлення запитано, але ще не зроблено, 0: Цей об'єкт повідомлення не очікує на передачу.
DLC3…0	Код довжини даних: 0…8: Кадр даних містить 0…8 байтів даних, 9…15: Кадр даних містить 8 байтів даних. Це поле об'єкту повідомлення має бути визначено однаково в усіх відповідних об'єктах з таким самих ідентифікатором на усіх інших вузлах. Коли обробних повідомлень зберігає кадр даних, він запише в DLC значення з отриманого повідомлення.
Data 0	Перший байт даних CAN– кадру.
….	…….
Data 7	Восьмий байт даних CAN– кадру.

Примітка: Байт Data 0 – це перший байт даних, який зсувається в регістр зсуву CAN – ядром під час прийому, Data 7 – останній байт. Коли обробник повідомлень зберігає кадр даних, він запише усі вісім байт даних в об'єкт повідомлення. Якщо код довжини даних (DLC) менше 8, решта байтів об'єкту повідомлення буде перезаписана невизначеними значеннями (non specified values).

11.4.3.3 Регістри обробника повідомлень (Message Handler Registers)

Усі регістри обробника повідомлень доступні лише для читання. Їх вміст (біти TxRqst, NewDat, IntPnd, та MsgVal кожного об’єкту повідомлення та ідентифікатор переривань) є статусною інформацією, яка забезпечена обробником повідомлень.

11.4.3.3.1 Interrupt Register (регістр переривань)

На рисунку 11.32 наведено формат регістра, а у таблиці 11.15 наведено опис окремих бітів.

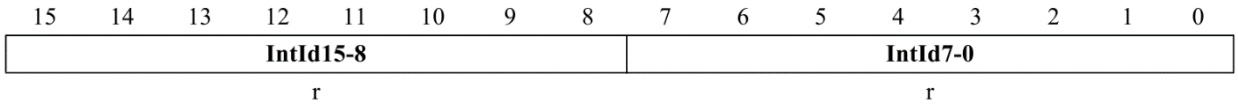


Рисунок 11.32 – Interrupt Register

Таблиця 11.15 – Опис бітів Interrupt Register

Назва	Функція
IntId15…0	Ідентифікатор переривань (число ідентифікує джерело переривань): 0x0000 – Немає переривань, які очікують, 0x0001…0x0020 – Номер об’єкта повідомлення, що є джерелом переривання, 0x0021…0x7FFF – Не використовуються, 0x8000 – Переривання стану (Status Interrupt), 0x8001–0xFFFF – Не використовуються.

Якщо обробки очікують декілька переривань, CAN Interrupt Register буде вказувати на джерело переривання з найвищим пріоритетом, незважаючи на хронологію їх виклику. Переривання продовжує очікувати обробки доки процесор не очистить його. Якщо значення IntId відмінне від 0x0000 і значення IE=1, лінія переривань IRQ_B активна. Лінія переривань залишатиметься активною доки значення IntId не буде встановлено в 0x0000 (причиною переривання є скидання) або IE не буде скинуто.

Переривання стану (Status Interrupt) має найвищий пріоритет. Пріоритет переривання об’єкта повідомлення зменшується зі зростанням номеру повідомлення в порівнянні з іншими перериваннями повідомлень.

11.4.3.3.2 Transmission Request Registers (регістр запиту передачі)

На рисунку 11.33 наведено формат регістра, а у таблиці 11.16 наведено опис окремих бітів.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Transmission Request 1 Register (addresses 0x81 & 0x80)	**TxRqst16-9**								**TxRqst8-1**							
Transmission Request 2 Register (addresses 0x83 & 0x82)	**TxRqst32-25**								**TxRqst24-17**							
	r								r							

Рисунок 11.33 – Transmission Request Registers

Ці регістри зберігають біти TxRqst усіх 32 об’єктів повідомлень. Зчитуючи біти TxRqst процесор може перевіряти для яких об’єктів повідомлень запитується передача. Біт TxRqst деякого конкретного об’єкту повідомлення може бути встановлений або скинутий процесором через IFx Message Interface Registers або через обробник повідомлень після отримання віддаленого кадру або після успішної передачі.

Таблиця 11.16 – Опис бітів Transmission Request Registers

Назва	Функція
TxRqst32…1	Біти запиту передачі: 1: Передача цього об’єкту повідомлення запитується і ще не завершена, 0: Об’єкт повідомлення не очікує на передачу.

11.4.3.3.3 New Data Registers (регістри нових даних)

На рисунку 11.34 наведено формат регістрів, а у таблиці 11.17 наведено опис окремих бітів.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
New Data 1 Register (addresses 0x91 & 0x90)	**NewDat16-9**								**NewDat8-1**							
New Data 2 Register (addresses 0x93 & 0x92)	**NewDat32-25**								**NewDat24-17**							
	r								r							

Рисунок 11.34 – New Data Registers

Ці регістри зберігають біти NewDat усіх 32 об’єктів повідомлень. Зчитуючи біти NewDat процесор може перевіряти для яких об’єктів повідомлень було оновлено розділ даних. Біт NewDat деякого конкретного об’єкта повідомлення може бути встановлений або скинутий процесором через IFx Message Interface Registers або через обробник повідомлень після отримання кадру даних або після успішної передачі.

Таблиця 11.17 – Опис бітів New Data Registers

Назва	Функція
NewDat 32…1	Біти нових даних: 1: Обробник повідомлень записав нові дані в розділ даних цього об’єкту повідомлення, 0: Жодних нових даних не було записано в розділ даних цього об’єкту повідомлення обробником повідомлень з моменту відколи цей прапорець було останнього разу скинутий. процесором.

11.4.3.3.4 Interrupt Pending Registers (регістри очікуючих переривань)

На рисунку 11.35 наведено формат регістрів, а у таблиці 11.18 наведено опис окремих бітів.

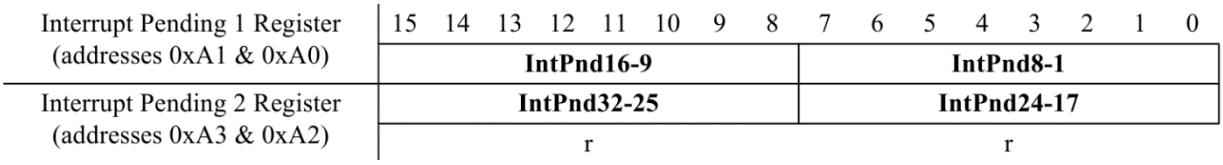


Рисунок 11.35 – Interrupt Pending Registers

Таблиця 11.18 – Опис бітів Interrupt Pending Registers

Назва	Функція
IntPnd 32…1	Біти переривань, що очікують: 1: Цей об’єкт повідомлення є джерелом переривання, 0: Цей об’єкт повідомлення не є джерелом переривання.

Ці регістри зберігають біти IntPnd усіх 32 об’єктів повідомлень. Зчитуючи біти IntPnd процесор може перевіряти для яких об’єктів повідомлень очікуються переривання. Біт IntPnd деякого конкретного об’єкту повідомлення може бути встановлений або скинутий процесором через IFx Message Interface Registers або через обробник повідомлень після отримання кадру чи після успішної передачі. Це також вплине на значення біта IntId в Interrupt Register.

11.4.3.3.4 Message Valid Register (регістри дійсності повідомлень)

На рисунку 11.36 наведено формат регістрів, а у таблиці 11.19 наведено опис окремих бітів.

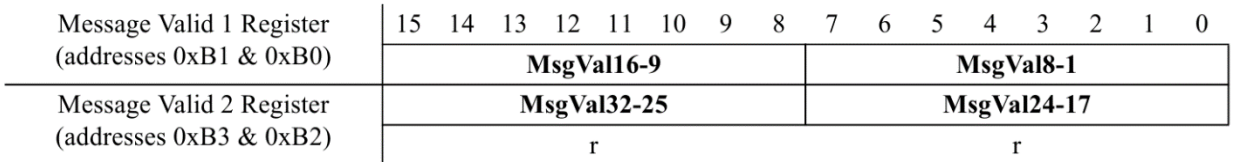


Рисунок 11.36 – Message Valid 1 Register

Таблиця 11.19 – Опис бітів Message Valid Register

Назва	Функція
MsgVal32…1	Біти дійсності повідомлень (усіх об’єктів повідомлень): 1: Цей об’єкт повідомлення налаштований і має бути розглянутий обробником повідомлень, 0: Цей об’єкт повідомлення ігнорується обробником повідомлень.

Ці регістри зберігають біти MsgVal усіх 32 об’єктів повідомлень. Зчитуючи біти MsgVal процесор може перевіряти які об’єкти повідомлень

дійсні. Біт MsgVal деякого конкретного об’єкта повідомлення може бути встановлений або скинутий процесором через IFx Message Interface Registers.

11.4.4 Схеми алгоритмів роботи

11.4.4.1 Алгоритм ініціалізації CAN модуля

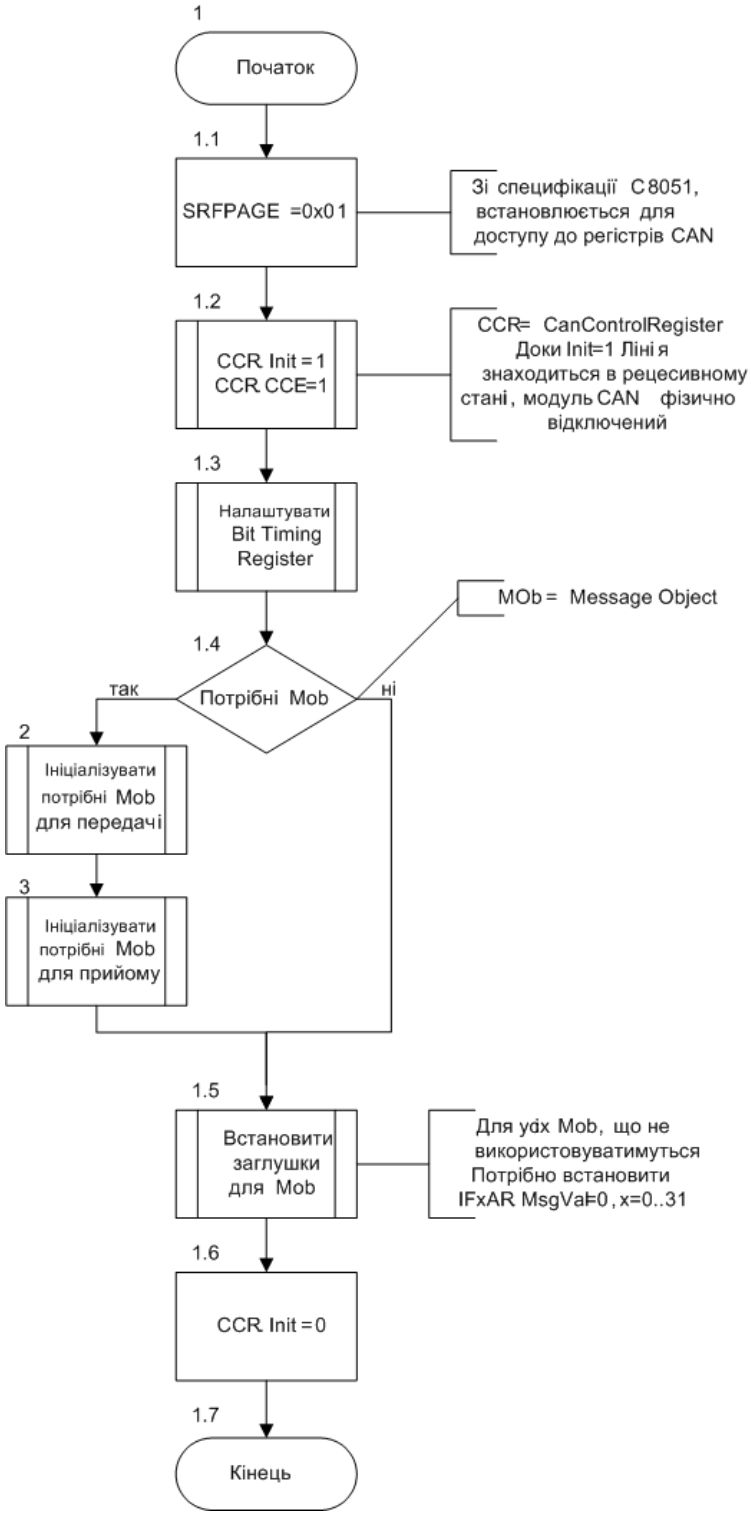


Рисунок 11.37 – Алгоритм ініціалізації CAN модуля

11.4.4.2 Алгоритм підготовки MOb для передачі кадру даних

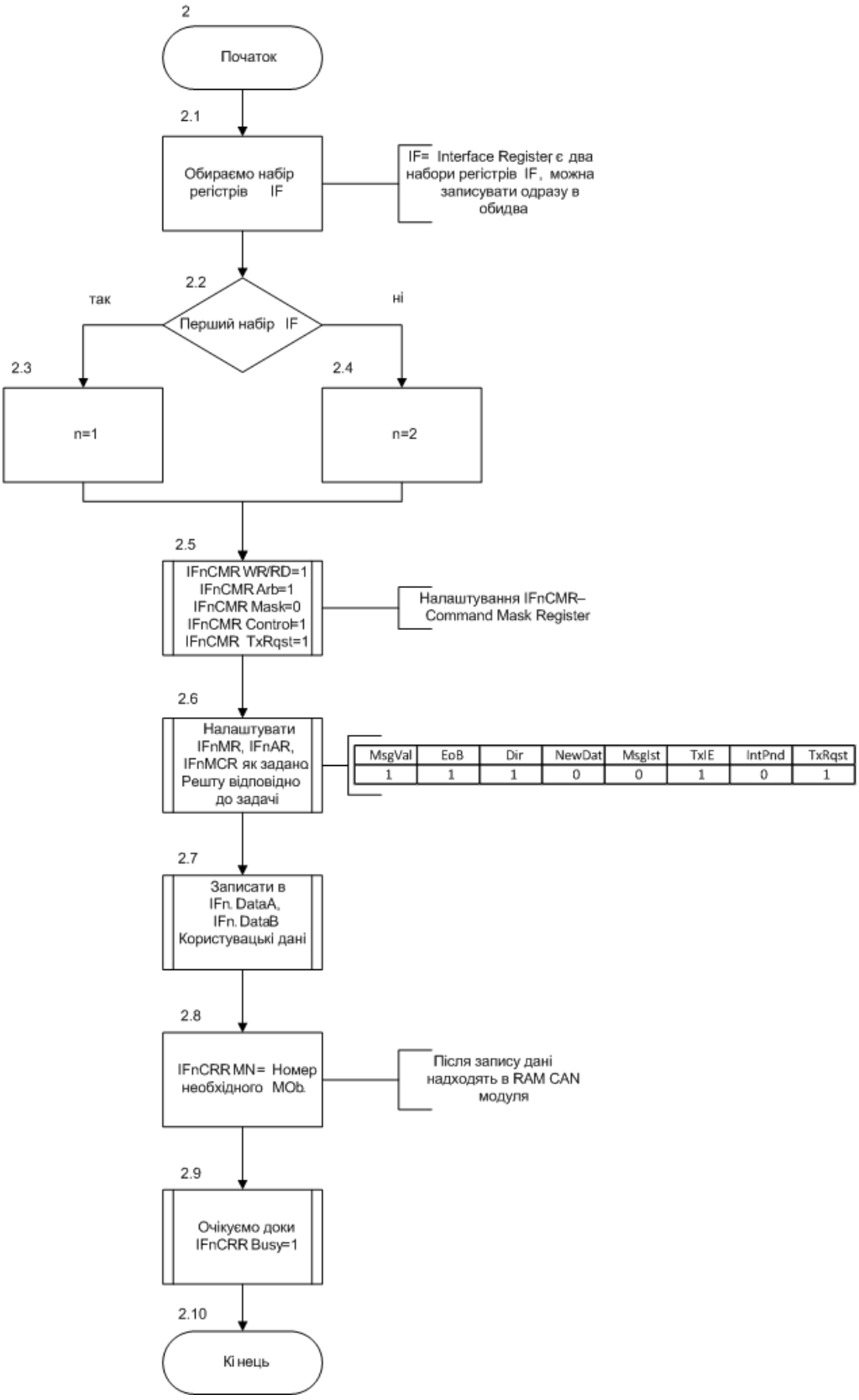


Рисунок 11.38 – Алгоритм підготовки MOb для передачі кадру даних

11.4.4.3 Алгоритм підготовки MOb для прийому кадру даних

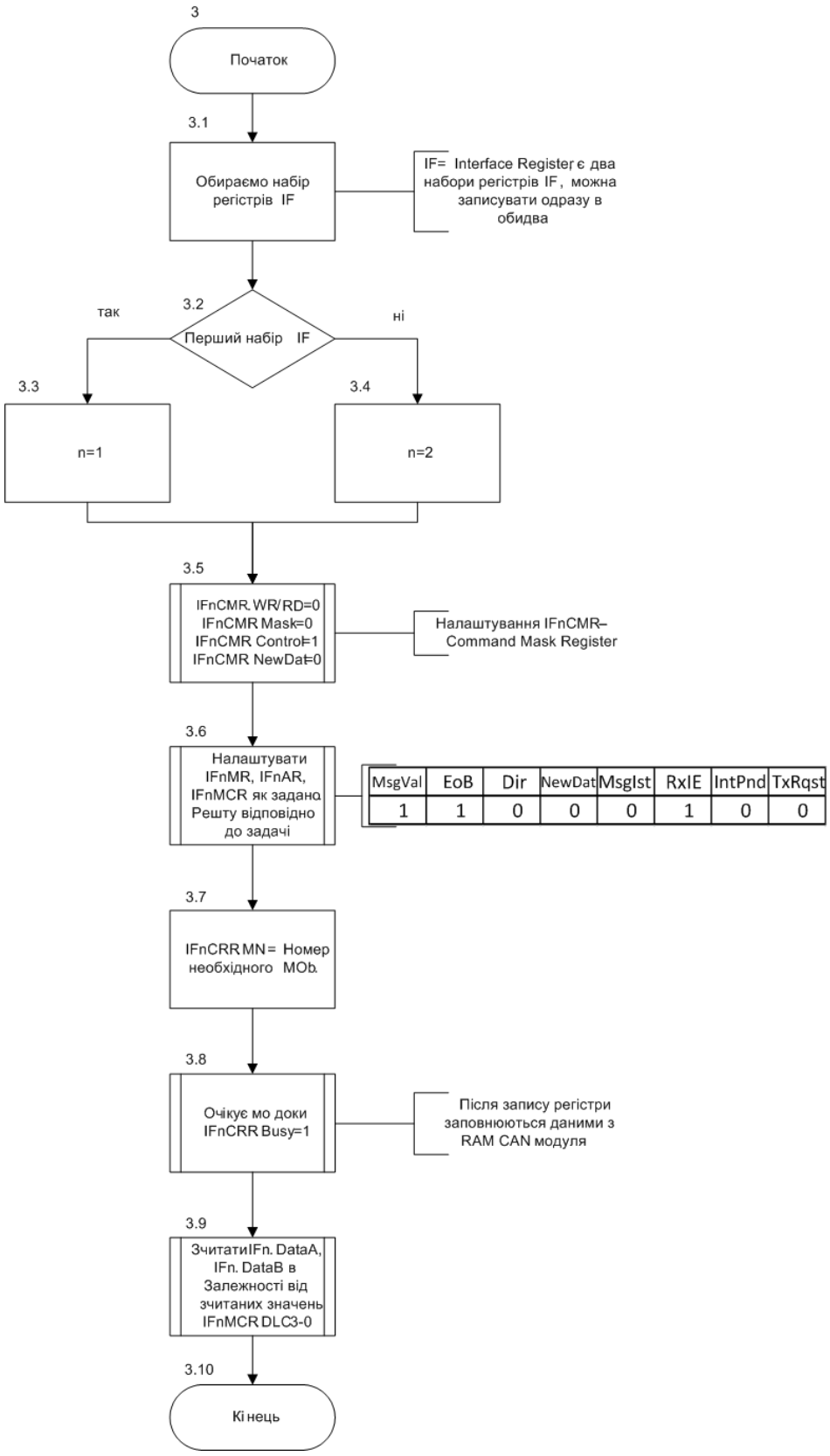


Рисунок 11.39 – Алгоритм підготовки MOb для прийому кадру даних

11.4.4.4 Загальний алгоритм роботи CAN–модуля

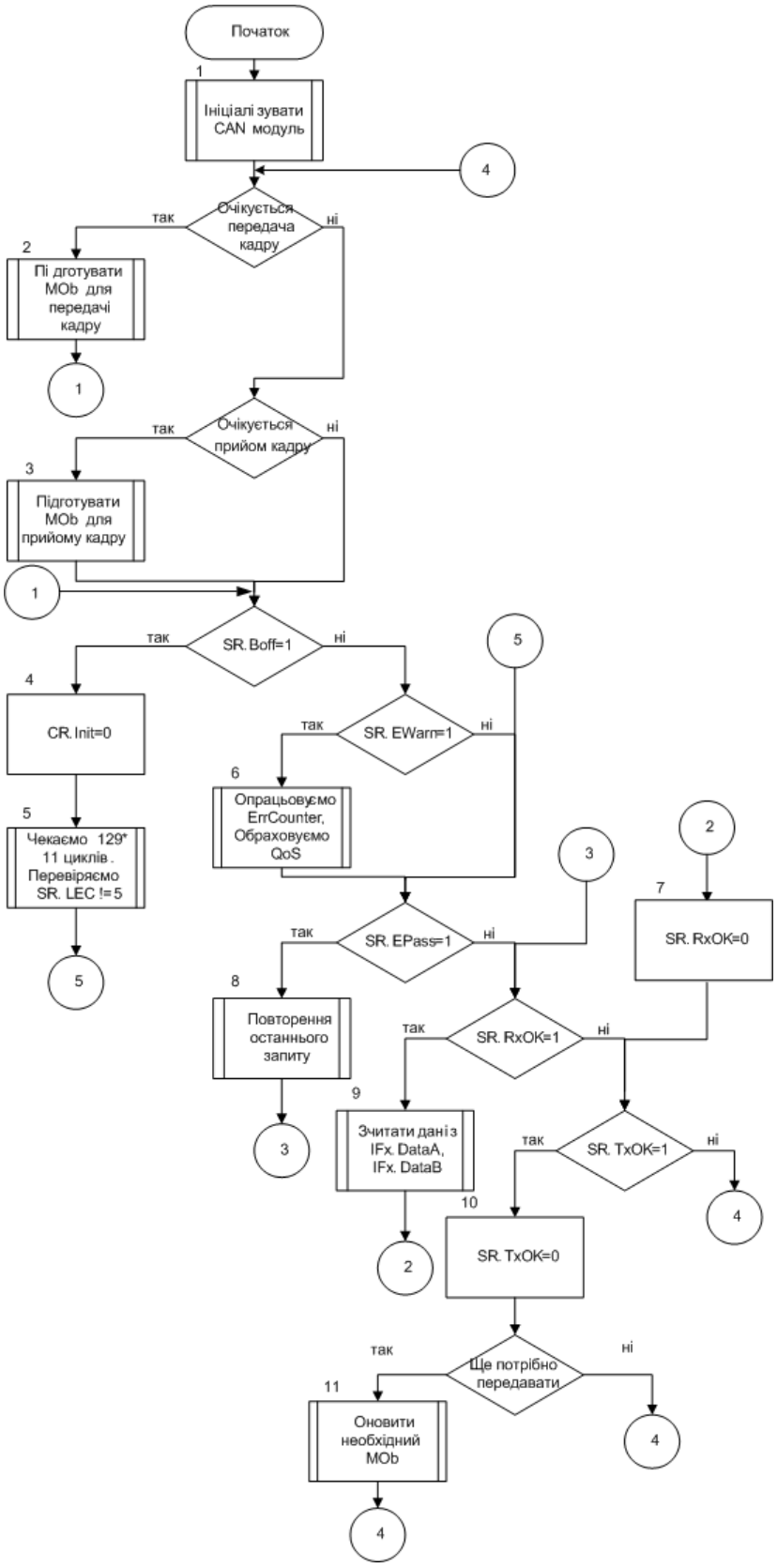


Рисунок 11.40 – Загальний алгоритм роботи CAN

ПИТАННЯ ДЛЯ САМОКОНТРОЛЮ

1) Назвіть основні характеристики CAN–протоколу.
2) Які типи кадрів передаються в CAN–мережах?
3) Опишіть формати повідомлень, які передаються CAN–мережею.
4) Як виконується контроль кадру, який формується за правилом надлишкового циклічного коду?
5) Опишіть ділянки номінального часу біта.
6) Як виконується апаратна синхронізація в CAN–мережах?
7) Як виконується синхронізація з відновленням тактових інтервалів?
8) Поясніть структурну схему мікроконтролера сімейства C8051F04x.
9) Поясніть типову схему конфігурації CAN–мережі.
10) Опишіть типову послідовність дій для ініціалізації CAN–контролера.
11) Які регістри використовуються для отримання тактової частоти CAN_CLK?
12) Назвіть та опишіть формати регістрів протоколу CAN–контролера.
13) Назвіть та опишіть формати інтерфейсних регістрів об`єктів повідомлень CAN–контролера.
14) Назвіть та опишіть формати регістрів обробника повідомлень.
15) Назвіть та опишіть формати регістрів спеціального призначення (SFR) ядра CIP–51, які використовуються при програмуванні CAN–контролера.
16) Поясніть використання регістрів CAN0ADR, CAN0DATH і CAN0DATL для доступу до CAN–регістрів.
17) Як здійснюється доступ до регістрів CAN0CN, CAN0STA і CAN0TST за допомогою індексного методу?
18) Скільки об`єктів повідомлень має CAN–процесор?
19) Назвіть максимальну швидкість обміну в CAN–мережі.

ПРЕДМЕТНИЙ ПОКАЖЧИК

СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

1. Гилмор Ч. Введение в микропроцессорную технику/ Пер. С англ..– М.: Энероатомидат, 1984.

2. В.В. Сташин и др.. Проектирование цифровых устройств на однокристальных микроконтроллерах. – М.: Энергоатомиздат, 1990.

3. Боборыкин А.В. и др. Однокристальные микро–ЭВМ. – М.: “МИКАП”, 1994.

4. А.О.Новацький, П.М.Повідайко. Організація та застосування одно кристальної мікроЕОМ МК51. – Навчальний посібник. – Житомир, 2001.

5. Мікроконтролери C8051 F045R– RUS.pdf

6. Мікроконтролери C8051 F045R.pdf

7. Мікроконтролери C8051 F020.pdf

8. Мікроконтролери C8051 F001.pdf

9. Мікроконтролери C8051 F045R–RUS.pdf

10. Гладштейн М. А. Микроконтроллеры смешанного сигнала C8051 Fxx фирмы Silicon Laboratories и их применение. Руководство пользователя. – М. Издательство Додэка, 2008.

11. О. Николайчук x51–совместимые микроконтроллеры фирмы Silicon Laboratories (Cygnal), М., ИД СКИМЕН, 2004.

12. Посібник користувача Bosch CAN user's Guide.

13. ADuC847–datasheet.

14. Проектування CAN–мережі: Навчальний посібник для студентів спеціальності 8.050201.01 «Комп’ютеризовані системи управління та автоматика» кафедри Автоматики та управління у технічних системах / Автор: А.О. Новацький К: НТУУ „КПІ”, 2011.