Мета викладання дисципліни «Проектування вбудованих комп'ютерних систем» на спеціальності 091503 «Спеціалізовані комп'ютерні системи» — надати студентам загальних принципів та деяких технічних особливостей розробки вбудованих систем керування обладнанням різноманітного призначення.

Навчальний посібник висвітлює матеріал курсу, розглядаючи як основу побудови таких систем розповсюджені мікроконтролери сімейства x51. Виклад теоретичного матеріалу доповнюється відомостями практичного характеру, необхідними для написання програмного забезпечення для систем, що розробляються.

Посібник призначений для студентів 5 курсу спеціальності 091503 «Спеціалізовані комп'ютерні системи», але може також стати в нагоді студентам інших спеціальностей, що займаються розробкою систем на базі мікроконтролерів.

Укладач:
Нєнов Олексій Леонідович

Рецензент:

Навчальний посібник розглянутий і рекомендований до видання на засіданні кафедри інформаційних систем і мереж
протокол № 4 від 28.11.2008 р.

Завідувач кафедри ІСМ, д. т. н., проф. Князєва Н. О.

Посібник затверджений методичною комісією факультету інформаційних технологій
протокол № ____ від _____ 200__ р.

Голова методичної комісії,

ЗМІСТ

Передмова

Дисципліна «Проектування вбудованих комп'ютерних систем» є однією із завершальних циклу дисциплін, що опановують студенти спеціальності 091503 «Спеціалізовані комп'ютерні системи». У рамках цього курсу розглядаються відомості, необхідні для побудови насамперед мікропроцесорних систем керування спеціалізованим устаткуванням.

Завдання проектування вбудованих комп'ютерних систем (ВКС) є комплексним, потребуючим від розроблювача специфічних знань із різних областей апаратної й програмної інженерії.

Насамперед, перед розроблювачем постає необхідність вибору основи побудови вбудованої комп'ютерної системи, у якості якої можуть виступати програмовані логічні інтегральні схеми (ПЛІС), програмовані логічні контролери, мікроконтролери загального або спеціалізованого призначення, мікропроцесори загального призначення або навіть готові мікро-ЕОМ у промисловому виконанні. Вибір основи ВКС у значній мірі визначає характер подальшого проектування. Кожен із зазначених засобів висуває специфічні вимоги до апаратного і програмного інтерфейсу з об'єктом керування, а також до системи керування верхнього рівня.

Таким чином, черговим етапом проектування є розробка схем апаратного сполучення елементів системи керування з врахуванням їх функціональних, електричних й інших параметрів. Тут необхідні знання і навички в області схемотехнічного проектування, а також знання стандартів розповсюджених інтерфейсів.

Далі, необхідно розробити алгоритм взаємодії елементів системи. Для цього зручно виконувати моделювання поводження об'єктів у термінах станів. На основі розробленої моделі виконуються написання й налагодження програм, керуючих кожним з об'єктів взаємодії. Все це вимагає знання архітектури конкретної платформи, мови програмування, засобів, наявних у розпорядженні програміста, і особливостей їхнього використання.

Нарешті, спроектована на схемотехнічному, логічному й програмному рівнях, система повинна бути конструктивно інтегрована в кероване устаткування. Це завдання вимагає наявності специфічних знань машинобудівного проектування, включаючи конструювання та перевірочні розрахунки.

У даному посібнику в якості основи побудови подібних систем розглядаються широко розповсюджені мікроконтролери сімейства x51. Приводяться відомості загального характеру, довідкові дані, необхідні для розуміння особливостей архітектури даного сімейства контролерів й їхнього подальшого використання.

Вступ

Поява першого мікропроцесора в 1971 році проголосила початок нової ери як в обчислювальній техніці, так і в області вбудованих систем керування устаткуванням – ери високопродуктивних надійних мікропроцесорних систем керування (СК), інтегрованих у різні машини, механізми, прилади, вироби (верстати, роботи, автомобілі, побутову техніку тощо).

Революція в керуючій електроніці супроводжується революцією й у силовій електроніці, що дозволяє створювати інтегрально-гібридні інтелектуальні електронні модулі, а також конструктивно інтегрувати в одному виробі:

- робочий орган механізму;
- силовий перетворювач;
- пристрій керування;
- джерело живлення;
- датчики і т. д.

Розвиток електромеханіки призвів до появи нового напряму – *мехатроніки*, що являє собою єдність механіки й електроніки на принципово новому рівні. Ця єдність являє собою ціле, яке більше, ніж проста сума складових, що народжує, зокрема, нові спеціальності. При цьому ріст керуючих обчислювальних ресурсів (на великих інтегральних мікросхемах) супроводжується значним розвитком надійності устаткування за рахунок зменшення монтажних з'єднань і розширенням функцій можливостей виробу. Серед цих можливостей – адаптація до умов експлуатації, інтерактивне навчання в процесі діалогу з оператором тощо.

Глава 1. Загальні відомості про вбудовані комп'ютерні системи

§ 1. Поняття вбудованої системи керування

Вбудована система керування (ВСК) – система керування, конструктивно інтегрована в устаткування. (Система керування (СУ) – пристрій або група пристроїв, призначених для маніпулювання поводженням деякого об'єкта керування). Приклад: система керування, вбудована в статистичний споживач частоти для асинхронних двигунів – мікроконтролери ЕОМ з набором інтерфейсів для керування інвертором, приводом у цілому, взаємодії з оператором (пульт керування). Обов'язковий для багатьох ВСК компонент – інтерфейс із системою керування більш високого рівня (промисловим контролером або промисловим комп'ютером). Наявність такого інтерфейсу дозволяє вирішувати завдання комплексної автоматизації групою одиниць технологічного устаткування, іншими словами, будувати розподілені системи керування.

Залежно від складності завдання, вбудовані системи керування можуть бути:

- одно- або багатопроцесорні;
- одно- або багаторівневі.

Нижній рівень – багаторівневої СК вирішує завдання безпосереднього керування окремими компонентами устаткування, а верхній рівень – завдання: 1) загального керування в реальному часі, 2) зв'язку з оператором, 3) зв'язку із системою керування більш високого рівня та ін.

Типові приклади систем з ВСК – верстати із числовим програмним керуванням, роботи, автомобільна і авіатехніка, побутова електроніка, системи зв'язку.

Для багаторівневих систем керування зараз простежується тенденція оформлення систем керування середнього рівня в самостійний конструктив для монтажу в стійки або панелі керування (настільні, настінні). Такі пристрої одержали назву індустріальних робочих

станцій, індустріальних комп'ютерів, панельних комп'ютерів, промислових комп'ютерів. Підключення комп'ютерів нижнього рівня до системи середнього рівня може здійснюватися:

- через стандартний послідовний або паралельний інтерфейс;
- шляхом їх приєднання до системної шини як спеціалізованих пристроїв з'єднання з об'єктом керування (модулі керування спеціальними операціями).

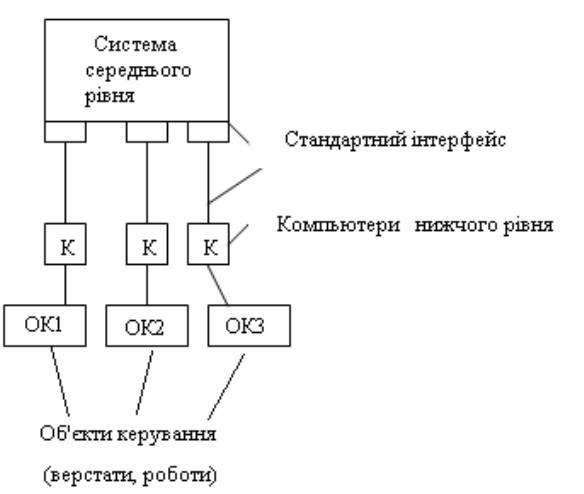

Рисунок 1 – Багаторівнева система керування

Індустріальні комп'ютери, як правило, мають великий набір стандартних операцій, що робить їх універсальними засобами керування (на середньому рівні) для широкого кола застосувань. Проте, на нижньому рівні керування в більшості випадків потрібна розробка спеціалізованих контролерів (вбудованих), що можуть виконувати лише фахівці в даній предметній області, які добре розуміються на особливостях об'єкта керування й можуть запропонувати та реалізувати весь комплекс необхідних алгоритмів керування об'єктом.

Важливою відмінністю промислового контролера від промислового комп'ютера є адаптація мови програмування під конкретну область застосування. Більшість промислових комп'ютерів Siemens, Auer Bredley, Fanuc, ABB та ін. є програмованими логічними контро-

лерами (PLC) і мають вбудовані інтерпретатори з мови релейної автоматики або Булевої алгебри. Ці мови дозволяють інженеру, не знайомому з програмуванням комп'ютерів загального призначення, виконати програмування подібної системи керування.

Таким чином, однорівневі системи керування виконуються переважно на базі однокристальних мікро-ЕОМ (мікроконтролерів), а багаторівневі припускають використання на середньому рівні багатоплатних миікро-ЕОМ типу промислового програмованого контролера або промислового комп'ютера.

До складу вбудованої системи керування крім мікропроцесора (однокристального мікроконтролера) входять додаткові елементи пам'яті, периферійні інтерфейси великих інтегральних мікросхем для органічної сполуки з датчиками, об'єктами керування, системами керування більш високого рівня.

З погляду виробників мікропроцесорної техніки всі завдання, які розв'язуються вбудованими системами, поділяються на два великих класи:

1. керування подіями в реальному часі;
2. керування потоками даних.

Будь-який клас завдань висуває специфічні вимоги до мікропроцесора (мікроконтролера). Це відбивається у наборі функцій, реалізованих на кристалі та у системі команд.

До першого класу (керування подіями в реальному часі) належать завдання, які вимагають швидкої реакції мікропроцесорної системи на зміну зовнішніх умов (спрацьовування датчиків, зміна параметрів). Це, наприклад, системи керування приводами, роботами, системи розподіленої автоматизації. Такі завдання, як правило, вимагають використання мікроконтролерів з великим обсягом інтегрованої периферії (включаючи пам'ять і пристрій виводу). Обсяг пам'яті може бути невеликий (до 32 Кб – достатньо для реалізації алгоритму керування). Наприклад, Intel MCS-51/151/251 (8 біт), MCS-96/196/296 (16 біт).

До другого класу (керування потоками даних) належать завдання, які вимагають швидкої обробки значних обсягів інформації.

Наприклад, у мікропроцесорній системі підтримки комп'ютерних мереж, системах керування літальними апаратами, рухомим складом, системах обробки відеозображень – мікропроцесори виконують безліч обчислювальних операцій, у тому числі із плаваючою точкою. Для рішення такі завдання потрібно високопродуктивний процесори (32/ 64-розрядний). Наприклад, в Intel 80C186, 386EX – 16/ 32-розрядні мікропроцесорні PC-подібної архітектури, i960 (побудованої за RISC-технологією).

Отже, нижній рівень керування будується, в основному, на базі однокристальних мікро-ЕОМ або мікроконтролерів. Можуть також застосовуватися закінчені одноплатні системи керування на їхній основі, які випускаються рядом фірм як контролери-прототипи.

§ 2. Засоби, що використовуються як основа ВКС

В якості основи побудови ВКС можуть використовуватися:

- мікропроцесори (загального призначення або спеціалізовані),
- мікроконтролери (однокристальні),
- одноплатні контролери,
- ПЛІС,
- схемні рішення,

а також, можливо, їхні комбінації.

Мікроконтролер – мікросхема, призначена для керування електронними пристроями.

Типовий мікроконтролер сполучає на одному кристалі функції процесора й периферійних пристроїв, а також містить оперативний запам'ятовувальний пристрій і постійний запам'ятовувальний пристрій. По суті, мікроконтролер – це однокристальний комп'ютер, здатний виконувати відносно прості завдання.

Використання однієї мікросхеми, замість цілого набору (як у випадку персонального комп'ютера) знижує розміри, енергоспоживання й вартість пристроїв, побудованих на базі мікроконтролера.

При проектуванні мікроконтролерів доводиться дотримувати баланс між розмірами й вартістю з однієї сторони й гнучкістю й продуктивністю з іншої. Величезна кількість типів мікроконтролерів,

які відрізняються архітектурою процесорного модуля, розміром і типом вбудованої пам'яті, набором периферійних пристроїв, типом корпуса й т. д.

У той час як 8-розрядні процесори загального призначення були витіснені більш продуктивними моделями, 8-розрядні мікроконтролери продовжують широко використовуватися. Це обумовлено тим, що існує велика кількість застосувань, у яких не потрібна висока продуктивність, але важлива низька вартість. Проте існують мікроконтролери, які мають великі обчислювальні можливості, наприклад цифрові сигнальні процесори (DSP).

Обмеження за ціною й енергоспоживанням стримують також ріст тактової частоти мікроконтролера. Оскільки різноманітних завдань безліч, то існують відповідно мікроконтролери з різними тактовими частотами й напругами живлення. У багатьох моделях мікроконтролерів використовується статична пам'ять (SRAM) для оперативного запам'ятовувального пристрою й внутрішніх регістрів, що дає можливість контролеру працювати на менших тактових частотах і навіть не втрачати дані при зупинці тактового генератора. Часто в мікроконтролері передбачені різні режими енергозбереження.

Крім оперативного запам'ятовувального пристрою, мікроконтролер має вбудовану енергетично незалежну (постійну) пам'ять для зберігання програми й даних, а також може бути оснащений шинами для підключення зовнішньої пам'яті. Найбільш дешеві типи пам'яті допускають лише однократний запис (для масового виробництва налагоджених мікроконтролерів). Може також використатися Flash-пам’ять. На відміну від процесорів загального призначення, у мікроконтролерах часто використовується гарвардська архітектура пам’яті (роздільні пам'ять і шини під програми й дані).

З периферії мікроконтролер може містити у собі:

- інтерфейси вводу-виводу, такі як UART, I2C, SPI, CAN, USB тощо;
- аналого-цифрові та цифро-аналогові перетворювачі;
- компаратори;
- широтно-імпульсні модулятори;

- таймери.

Програмування мікроконтролерів як правило здійснюється мовою Асемблера або С, хоча існують компілятори для інших мов. Нерідко використовуються також вбудовані інтерпретатори Бейсіка й Форту.

Для налагодження програм використовуються:

- програмні симулятори – спеціальні програми для персональних комп'ютерів, які імітують роботу мікроконтролера;
- внутрісхемні емулятори – електронні пристрої, які імітують мікроконтролер і які можна підключити замість нього до розроблювального вбудованого пристрою;
- інтерфейс JTAG – спеціальний апаратний інтерфейс, розроблений для тестування цифрових процесорів (IEEE 1149.1). Передбачає наявність порту тестування із чотирма або п'ятьма виділеними виводами мікросхеми, а також вбудованого в мікросхему автомата керування.

Програмована логічна інтегральна схема – електронний компонент, що використовується для створення цифрових інтегральних схем. Логіка роботи програмованої логічної інтегральної схеми не визначається при виготовленні, а задається за допомогою програмування (проектування). Для цього використовуються налагоджувальні середовища, які дозволяють задати бажану структуру цифрового пристрою у вигляді принципової електричної схеми або програми на спеціальних мовах Verilog, VHDL. Альтернативою програмованої логічної інтегральної схеми є замовлені великі інтегральні мікросхеми, які істотно дорожчі, а також комп'ютери (мікроконтролери), які через програмний спосіб реалізації алгоритмів повільніші, ніж програмовані логічні інтегральні схеми.

ПЛІС широко використовуються для побудови різних по складності й можливостям цифрових пристроїв. Це застосування, де необхідна велика кількість портів вводу-виводу, цифрова обробка сигналу, цифрова відео-аудіо апаратура, високошвидкісна передача даних, криптографія, проектування ASIC, в якості мостів (комутаторів) між системами з різною логікою й напругою живлення та ін.

§ 3. Відмінності між вбудованими системами керування та звичайними комп'ютерами

ВКС відрізняються від звичайних комп'ютерів (наприклад, персонального комп'ютера) такими характерними особливостями:

- вбудовані системи керування вирішують певні специфічні завдання;
- вбудовані системи керування будуються на основі різноманітного асортименту процесорів та архітектур;
- вбудовані системи керування чутливі до обмежень за вартістю, габаритами, відмовами живлення, тепловиділенню (мікросхема часто зовсім не охолоджуються, або, максимум, охолоджуються невеликим радіатором), а також за іншими параметрами;
- вбудовані системи керування в більшості своїй є системами реального часу (з відповідним програмним забезпеченням, зокрема операційною системою);
- вбудовані системи керування звичайно мають менші ресурси та вимагають менше ресурсів для свого функціонування, ніж звичайний комп'ютер (отже, вбудовані системи керування менш ресурсоемні);
- вбудовані системи керування часто зберігають весь об'єктний код (програмне забезпечення) в ROM (постійний запам'ятовувальний пристрій);
- проектування вбудованих систем керування часто вимагає використання специфічних інструментів і методів;
- вбудовані мікропроцесори часто забезпечуються схемою налагодження.

§ 4. Архітектура вбудованих комп’ютерних систем

На рисунках 2 і 3 показані схеми комп’ютерної системи загального призначення та ВКС відповідно.

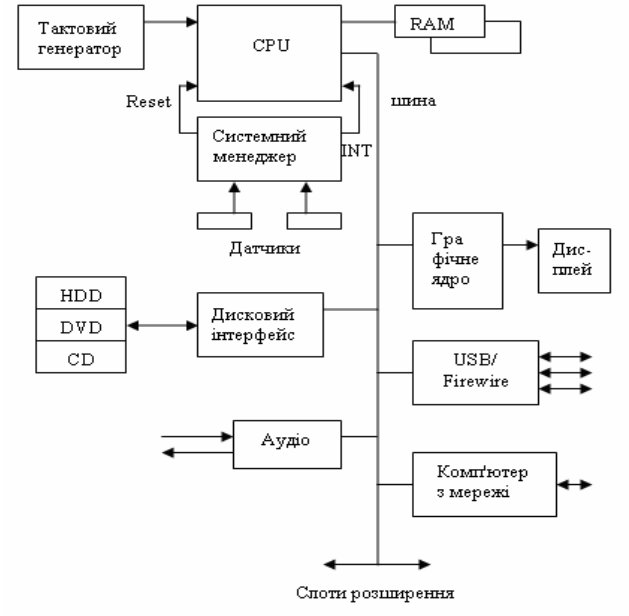

Рисунок 2 – Схема комп'ютера загального призначення

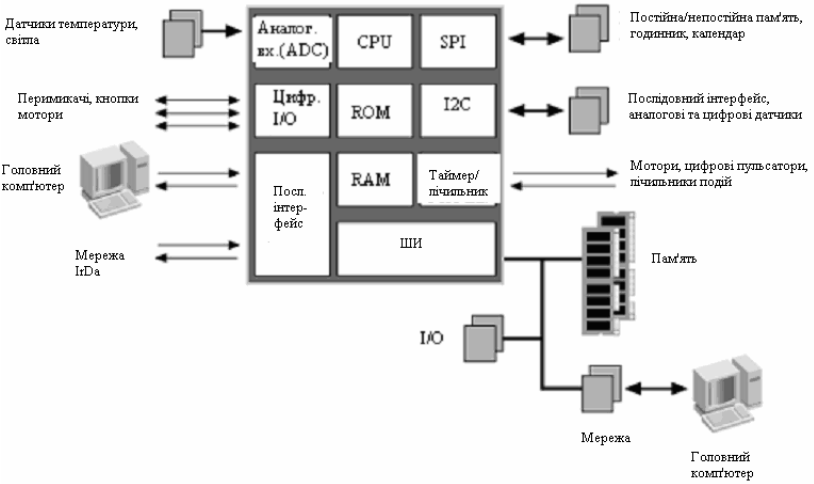

Рисунок 3 – Архітектура типової вбудованої системи

Як видно з рисунка 3, мікроконтролер містить процесор, невелику кількість внутрішньої пам'яті (постійний запам'ятовувальний пристрій та оперативний запам'ятовувальний пристрій), а також реалізовані у вигляді внутрішніх блоків підсистеми введення-виведення інформації. Ці підсистеми забезпечують додаткову функціональність і є загальними для багатьох процесорів. Найбільш загальним інтерфейсом вводу-виводу є цифровий ввід-вивід, що називається також ввід-вивід загального призначення (GPIO – General Purpose Input/Output). Ці порти можуть керуватися програмно (на рівні точка-точка). Цифрові входи можуть використовуватися для читання стану перемикачів, кнопок, цифрових пристроїв. Цифрові виходи можуть використовуватися для вмикання і вимикання зовнішніх пристроїв або зміни їхнього стану (наприклад, для активізації схеми керування тиском, вмикання і вимикання лампочки, активізації деяких інших пристроїв).

Контрольні питання до глави 1

1. Що таке "мехатроніка"?
2. У чому полягає специфіка вбудованих комп'ютерних систем у загальному класі комп'ютерних систем?
3. Яким чином можна класифікувати вбудовані комп’ютерні системи?
4. Які класи завдань вирішують вбудовані комп'ютерні системи?
5. Що таке ПЛІС? У чому її відмінності від мікроконтролерів і звичайних інтегральних схем?
6. Яка архітектура типової багаторівневої системи керування?
7. У чому відмінність програмувального мікроконтролера від комп’ютера загального призначення та від промислового комп'ютера?
8. Які принципові розходження в структурі універсальних і вбудованих мікропроцесорних систем?
9. Що входить до складу типової вбудованої системи керування?

Глава 2. Основи програмної архітектури МК сімейства x51

§ 5. Загальні відомості

Як уже згадувалося, часто як основа побудови вбудованих мікропроцесорних систем використовуються мікроконтролери сімейства x51. Це сімейство з'явилося наприкінці 80-х років минулого сторіччя в особі мікросхеми i8051 фірми Intel. Однак, незважаючи на солідний вік, воно продовжує розвиватися й залишається одним з найпоширеніших сімейств контролерів на сьогоднішній день. Причинами популярності сімейства є вдалість технологічних рішень, прийнятих при розробці даної мікросхеми, добре сполучення функціональності, продуктивності, порівняльної простоти виготовлення та використання.

Під *програмною архітектурою мікропроцесорної системи* розуміється сукупність внутрішніх і зовнішніх програмно доступних ресурсів, система команд, система переривань, функції вводу-виводу, протоколи обміну по магістралях.

§ 6. Пам’ять та регістри загального призначення

Істотною особливістю запам'ятовувального пристрою МК x51 є так звана *гарвардська архітектура* – програма й константи зберігаються в постійному запам'ятовувальному пристрої (ПЗП) обсягом 2-32 Кб, а дані – в оперативному запам'ятовувальному пристрої (ОЗП) обсягом 128-256 байт.

Для спілкування із зовнішніми запам'ятовувальними пристроями в мікроконтролері є два 16-розрядних регістри: **PC** (Program Counter – програмний лічильник) – використовується тільки для читання команд із ПЗП, і **DPTR** (Data PoinTeR – покажчик даних) – для читання даних з ПЗП та ОЗП, а також для запису в останнє. Отже, максимальний адресний простір, що може використовуватися мікроконтролером – 64 К.

Оскільки ОЗП мікроконтролера має невелику ємність, адресний простір в його 8-розрядній і становить 256 байт. Старші адреси цього адресного простору використовуються для звертання до функціональ-

них регістрів (внутрішніх пристроїв) мікроконтролера. Початкові осередки оперативної пам'яті (32 байта) використовуються під однобайтові регістри загального призначення (РЗП): R0, R1, R2,...R7. Фізичні адреси регістрів загального призначення залежать від вмісту 3-го та 4-го розрядів регістра прапорців PSW (Processor Status Word) (див. далі).

Група регістрів загального призначення в певному місці оперативної пам'яті називається банком. Існує 4 банки регістрів загального призначення (з номерами 0-3) в оперативній пам'яті. До якого з них буде відбуватися звернення, залежить від битів (прапорців) RS0 та RS1 (див. таблицю 1).

Таблиця 1.

Банки пам’яті РЗП МК 8051

RS0	RS1	R0	R1	R2	R3	R4	R5	R6	R7	
0	0	00h	01h	02h	03h	04h	05h	06h	07h	банк 0
0	1	08h	09h	0Ah	0Bh	0Ch	0Dh	0Eh	0Fh	банк 1
1	0	10h	11h	12h	13h	14h	15h	16h	17h	банк 2
1	1	18h	19h	1Ah	1Bh	1Ch	1Dh	1Eh	1Fh	банк 3

Комірки за адресами 20h...1Fh – програміст може використовувати за своїм розсудом, 80h-0FFh – для доступу до функціональних регістрів (SFR).

§ 7. Спеціальні функціональні регістри

Спеціальні функціональні регістри (SFR) являють собою комірки, пов'язані з особливими функціями мікроконтролера і його внутрішніх пристроїв. Більшість функціональних регістрів мікроконтролера – 8-розрядні. Доступ до них здійснюється по певних адресах (далі ці адреси вказані в дужках після мнемонічного позначення регістру). Розглянемо основні спеціальні функціональні регістри.

- Накопичувач (*акумулятор*) **A** або **ACC** (0E0h) – використовується для арифметичних і логічних операцій;
- додатковий регістр **B** (0Fh) – використовується разом з A в командах множення й ділення;

- слово стану **PSW** (0D0h) – містить прапорці:

CY	(0D7h)	перенесення 7 із (старшого) розряду
AC	(0D6h)	перенесення із розряду (середина байта)
F0	(0D5h)	прапорець для використання програмістом
RS1	(0D4h)	старший розряд номера банка
RS0	(0D3h)	молодший розряд номера банка
OV	(0D2h)	переповнення результату
	(0D1h)	прапорець без імені (може використовуватися програмістом)
P	(0D0h)	ознака парної кількості одиниць у коді результату

Із двох згаданих раніше регістрів програмний лічильник ніяк не відображається на адресний простір оперативного запам'ятовувального пристрою, щоб не зіпсувати його вміст:

- покажчик даних **DPTR** (складається з 2 байтів): старшого **DPH** (83h) і молодшого **DPL** (82h);
- покажчик стека SP (81h) – використовується для звернення до оперативної пам'яті у стековому режимі;
- регістри паралельних портів: **P0** (80h), **P1** (90h), **P2** (0A0h), **P3** (0B0h). Їхні назви збігаються з назвами виводів мікроконтролера;
- буфер послідовного порту **SBUF** (Serial BUFer) (99h) – використається для введення/виведення через послідовний порт;
- регістр керування послідовним портом **SCON** (Serial CONtrol) (98h);
- 16-розрядні лічильники (умовно позначаються T/C0 і T/C1) складаються зі старшого **TH0** (8Ch), **TH1** (8Dh) і молодшого байтів **TL0** (8Ah), **TL1** (8Bh);
- регістр керування лічильниками **TCON** (88h).

Для роботи мікроконтролера із зовнішніми об'єктами в реальному масштабі часу в його складі є система переривань. Для роботи із системою переривань використовуються:

- регістр дозволу переривань **IE** (Interrupt Enable) (0A8h);
- регістр пріоритетів переривань **IP** (Interrupt Priority) (0B8h).

Можливий доступ до окремих бітів функціональних регістрів за назвою регістра і номера біта. Деякі з цих бітів мають свої мнемонічні імена.

Початковий стан всіх функціональних регістрів мікроконтролера при його ініціалізації наведений в таблиці 2:

Таблиця 2.

Початковий стан функціональних регістрів МК 8051 при ініціалізації

Регістр	Значення
A, B, PSW, DPH, DPL	00h
SP	07h
P0, P1, P2, P3	FFh
SBUF	xxxxxxxxb (не визначене)
SCON, TH0, TH1, TL0, TL1, TCON	00h
IE	0x000000b
IP	xx000000b

Інші спеціальні функціональні регістри розглядаються далі у відповідних главах.

§ 8. Цикл виконання команди

Цикл виконання команди складається з таких кроків:

1. читання коду з ПЗП за адресою з PC (з наступним інкрементуванням PC);
2. читання операндів команди;
3. виконання команди;
4. збереження результатів (після цього PC вказує на наступну команду).

Цикл виконання різних команд займає від 1 до 10 тактів. Період такту дорівнює 12 коливанням кварцового резонатора (при Чрез = 12 МГц період такту дорівнює 1 мкс).

§ 9. Формати й способи адресації даних у МК х51

Мікроконтролер 8051 працює, в основному, з даними бітового й байтового формату (до ОЗП і функціональних регістрів може зверта-

тися побайтово або побітово). Деякі команди працюють із 2-байтними даними. Ряд команд обробляє біти байт незалежно один від одного.

У записі команд можуть випливати один або 2 операнди, назив-ні джерелом і приймачем. Виконання команди може впливати на прапорці PSW.

Види адресації даних

Види адресації даних визначають, де саме розташовані дані, що обробляються командою.

- *Безпосередня* – дані в команді (# число);
- *Регістрова* – дані в регістрі (з РЗП);

В пам'яті внутрішній або зовнішній:

- *Пряма* (direct) (позначається далі як src або dst);
- *Непряма* (indirect) ; (позначається @R0/1 або DPTR)
- *Індексна*
- *Стекова*

Як і в x86, одній мнемоніці може відповідати декілька різновидів машинних команд.

При *безпосередній адресації* кодуються тільки дані-джерело, наприклад

```
Mov     R0, #5  ; запис числа в регістр
```

У такому випадку ставиться знак # і число, відповідне даним, що обробляються.

Числа в асемблері x51, так само як і в x86, можуть записуватися з суфіксом **d** (десяткова система, припускається за умовчанням), **h** (16-кова система), **o** або **q** (вісімкова система), а також **b** (двійкова система). Після знака **;** в асемблері записується коментар.

При *регістровій адресації* (наприклад Mov A, R0) дані розташовані в регістрі, номер якого зберігається в КОП. Регістрова адресація для обох операндів одної команди не допускається.

При *прямій адресації,* наприклад

```
Mov     R0, 5   ; з 5-ї комірки пам'яті
```

або

```
Mov     5, 6    ; з 5-ї комірки в 6-у
```

в команді вказується ім'я операнда, що перетворюється на адресу елементу пам'яті (наприклад функціонального регістра або зарезервованої комірки даних). Допускається пряма адресація для обох операндів.

При *непрямій адресації* адреса даних зберігається в регістрі R0, R1 або DPTR. Перед ім'ям регістра ставиться знак @ (наприклад Mov A, @R0). Регістри R0 та R1 дозволяють адресувати тільки 256 комірок. Непряма адресація обох операндів недопустима.

При *індексній адресації* адреса обчислюється як сума акумулятора A і вказівника PC або DPTR. Використовується тільки для читання з ПЗП. Записується @A+PC або @A+DPTR (для Mov така адресація можлива тільки з мнемонікою MovC).

При *стековій адресації* (використовується лише в командах PUSH/POP і XCALL/RET(I)) адреса береться з регістру SP, після чого його значення змінюється на 1 (+1 перед записом і –1 після його читання). Стек таким чином росте «вгору» (прямо), а не «униз» (назад), як в архітектурі x86.

Способи адресації команд

Команди можуть займати в пам’яті від 1 до 3 байтів. Перший байт – код операції (КОП), за ним слідують адреси і (або) дані. Всього в МК 8051 використовується 255 команд. КОП 0A51 зарезервований.

При природному (лінійному) порядку команди виконуються в порядку їх знаходження в ПЗП. Команди умовного та безумовного переходу можуть порушувати природний порядок виконання.

По дальності переходи діляться на короткі (відносні) (short), довгі (long) і абсолютні (absolute).

Короткий перехід – суть додавання до PC (тобто адресі чергової команди) знакового байтового зсуву (±127).

У попередників 8051 адресний простір ПЗП складав 2 Кбайти. В МК 8051 з його 64 Кбайтами ПЗП двохкілобайтна ділянка стала називатися сторінкою. Сторінка адресується 11 бітами (2^{11} = 2 К); 8 молодших з них знаходяться в другому байті команди, а 3 старших – в КОП. 5 старших бітів PC містять номер сторінки (за [8051 IDE], биті КОП 0..4 для команди AJMP завжди дорівнюють 00001).

При *абсолютному переході* 11 молодших бітів PC замінюються 11-бітною адресою з команди (отже, це абсолютний перехід всередині сторінки).

При *довгому переході* всі 16 бітів PC ініціалізуються значенням з 2-байтної адресної частині команди (отже це фактично абсолютний перехід в межах всіх 64 Кб ПЗП).

Букви S, A та L в мнемоніках позначають, відповідно, короткий, абсолютний і довгий переходи.

Різновиди команд абсолютного переходу – команди виклику підпрограми (із запам'ятовуванням адреси повернення) і повернення з підпрограми.

Існує також команда безумовного переходу, що використовує індексну адресацію (докладніше див. далі).

Контрольні питання до глави 2

1. Що входить у поняття архітектури мікропроцесорної системи?
2. У чому полягає особливість гарвардської архітектури пам'яті?
3. Яке максимальний адресний простір мікроконтролера? Чим визначається його розмір?
4. Що таке банк регістрів загального призначення? Від чого залежить, до якого банку відбувається звернення?
5. Які групи можна виділити серед спеціальних функціональних регістрів?
6. Який склад і призначення прапорців МК x51?
7. Які кроки містить у собі цикл виконання команди?
8. Як співвідносяться частота кварцового резонатора контролера й період такту?
9. Які способи адресації даних використаються в архітектурі x51?
10. Які способи адресації команд використаються в архітектурі x51?

Глава 3. Команди та директиви асемблера МК сімейства x51

В цій главі розглянутий набір команд і директив типового асемблера МК сімейства x51, необхідних для написання програм. Матеріал глави можна використовувати як довідник по основних командах. В описі команд позначення src означає джерело даних, а dst – приймач даних.

§ 10. Команди пересилки

У сімействі МК x51 є команди пересилання байта, напівбайта, 2-бітових і бітових значень. При цьому самі дані, що пересилаються, не змінюються (змінюється тільки вміст приймача), але може здійснюватися обробка масиву даних (типу сортування). Далі при описі команд символ | розділяє варіанти операндів, з яких необхідно використати тільки один.

Байтові операнди:

- **MOV** A, #src | Rn | @Ri | src – пряма адресація
  Rn, A | #src | src
  @Ri, A | #src | src
  dst, A | #src | src | Rn | @Ri

2-байтові операнди:

DPTR, #src

Бітові операнди:

C, flag
Flag, C

Використання зовнішнього ОЗП:

- **MOVX** A, @Ri | @DPTR
  @Ri | @DPTR, A

Читання з ПЗП:

- **MOVC** A, @A+DPTR | @A+PC

Очищення акумулятора:

- **CLR** A

Звернення до стека у ОЗП:

- **PUSH** src
- **POP** dst

Обмін операндів:

- **XCH** A, Rn | @Ri | src

Обмін лише молодших напівбайтів операндів:

- **XCHD** A, @Ri

Бітові операції пересилки

(іноді виділяються в окрему групу)

- **CLR** C | flag
- **SETB** C | flag

§ 11. Команди порозрядної обробки (логічні та зсувні)

Мнемонічні (для архітектур x51 та x86) та математичні позначення логічних (булевих) команд наведені в таблиці 3:

Таблиця 3.

Мнемонічні та математичні позначення логічних команд

НЕ	І	АБО	ВИКЛЮЧАЮЧЕ АБО	
!	Λ	$\vee$	$\forall$	- математичні позначення
CPL	ANL	ORL	XLR	- x51-мнемоніка
NOT	AND	OR	XOR	- x86-мнемоніка

Інверсія

- **CPL** A | C | flag

«І»

- **ANL** A, #src | Rn | @Ri | src
  dst, A | #src ; 8 біт
  C, flag | / flag ; 1 біт (/flag – інверт. біт)

«АБО»

- **ORL** A, #src | Rn | @Ri | src
  dst, A | #src
  C, flag | / flag

«ВИКЛЮЧАЮЧЕ АБО»

- **XRL** A, #src | Rn | @Ri | src | @Ri
  dst, A | #src

Зсуви (циклічні)

Зсув вліво:

- **RL** A

Зсув вправо:

- **RR** A

Зсув через прапорець C:

- **RLC** A
- **RRC** A

Обмін старших і молодший напівбайтів A – еквівалентно циклічному зсуву на 4 біти (без C):

- **SWAP** A

(За допомогою RLC і RRC при C = 0 може здійснюватися множення або ділення на 2).

§ 12. Арифметичні команди

Арифметичні команди призначені для обробки однобайтних додатних цілих чисел.

Додавання байтових операндів, сума розташовується в накопичувачі (акумуляторі):

- **ADD** A, #src | Rn | @Ri | src

Додавання з переносом:

- **ADDC** A, #svc | Rn | @Ri | src

Збільшення на 1 (інкремент):

- **INC** A | Rn | @Ri | src | DPTR

Корекція суми двійково-десяткових:

- **DA** A

Віднімання з позикою:

- **SUBB** A, #src | Rn | @Ri | src

Зменшення на 1 (декремент)

- **DEC** A | Rn | @Ri | src – тільки 1 байт

Множення A та B с занесенням добутку в BA:

- **MUL** AB

(OV = 1, если B = 0)

Ділення A (ділене) на B (дільник) – результат: частка в A, залишок в B.

- **DIV** AB

(OV = 1, якщо дільник B = 0)

§ 13. Керуючі команди

Умовні відносні переходи

- **JZ/JNZ** rel
- **JC/JNC** rel
- **JB/JNB** flag, rel

Умовний перехід з очищенням біта flag

- **JBC** flag, rel;

Умовний перехід з порівнянням та модифікацією прапорця C (перехід відбувається, якщо операнди не рівні)

- **CJNE** A, src | #src, rel
  Rn | @Ri,#src, rel

Зменшення операнда на 1 і перехід, якщо він $\neq 0$:

- **DJNZ** Rn | dst, rel

Безумовні переходи різної дальності

- **SJMP** rel
- **AJMP** adr11
- **LJMP** adr16

Безумовний перехід з індексною адресацією

- **JMP** @A+DPTR

Виклики (запис PC в стек (2-байти, молодший байт за молодшою адресою) та перехід за вказаною адресою):

- **ACALL** adr11
- **LCALL** adr16

Повернення

- **RET** ; прочитати адресу із стека (2 байти) і перейти
- **RETI** ; повернення з обробника переривання

§ 14. Директиви асемблера x51

Директиви асемблера дозволяють керувати процесом трансляції програми, не впливаючи на склад її команд. Існує досить багато різних програм-асемблерів для мікроконтролерів сімейства x51 й їхніх емуляторів. У кожному з них склад і синтаксис директив може розрізнятися.

Розглянемо склад директив для варіанта асемблера фірми AceBus (асемблер у складі інтегрованого середи розробки AceBus 8051IDE).

- **ORG** *адреса* – встановлює під час трансляції програмний вказівник (PC) асемблера в певне положення (тобто, виконує запис в PC певного значення; при виконанні відповідне значення буде знаходитися в PC мікроконтролера).
- **DATAORG** *число* – запис у лічильник адрес даних асемблера певного значення; після цього резервування даних буде змінювати його значення щодо записаного значення.
- *ідентифікатор* **EQU** *значення* – встановлює відповідність між значенням і строковим ідентифікатором.
- *ідентифікатор* **SET** *значення* – аналог EQU, але з можливістю змінювати далі в програмі значення за допомогою SET.
- *ідентифікатор* **BIT** *адреса* – встановлює відповідний біт адреси із символьним ідентифікатором (наприклад, HoldPin BIT 80h).
- *ідентифікатор*: **DB** *значення* – резервування програмної пам'яті (ПЗП) побайтово з ініціалізацією, наприклад:

```
NUM:        DB          10
MSQ:        DB          'Hello', 13, 10
```

- *ідентифікатор*: **DW** *значення* – резервування ПЗП 2-байтовими комірками (під кожне значення виділяється 2 байти в порядку “молодший, старший”).
- *ідентифікатор* **DS** *число* – резервування простору пам'яті даних (внутрішнього або зовнішнього ОЗП) розміром у *число* байт.

Керування лістингом:

- **PAGE**
- **%TITLE**
- **%SUBTTL**, %**TOPMAR**, %**BOTMAR**
- **END** – кінець програми
- **RADIX** DEC | HEX | BIN – зміна основи системи числення за замовчуванням (споконвічно – десяткова)
- **INCLUDE** "*файл*" – включення тексту файлу в текстовий файл (для багатофайлових проектів).
- **BREAK** – команда симулятору призупинити виконання програми (у режимі симуляції виконання).

Специфікою багатьох інших асемблерів є необхідність починати запис директиви із крапки.

Контрольні питання до глави 3

1. Які групи команд є в МК сімейства x51? У яку з них входить найбільше число команд?
2. Які команди входять у групу команд пересилання?
3. За допомогою яких команд можна звертатися до зовнішньої пам'яті програм і даних?
4. Які команди входять у групу логічних команд?
5. Які команди входять у групу зсувних команд?
6. Які команди входять у групу арифметичних команд?
7. Які команди входять у групу керуючих команд? Що для них є загальним?
8. Яка мнемонічна команда має найбільше число варіантів операндів?
9. Чим директиви асемблера принципово відрізняються від команд асемблера?
10. Як можна згрупувати по категоріях директиви асемблера для МК сімейства x51?
11. Які директиви асемблера x51 використовуються для резервування та іменування областей пам'яті?

Глава 4. Структурна організація МК x51

§ 15. Логічна структура МК 8051

Структурна схема мікроконтролера 8051

Нижче (рисунок 4) наведена структурна схема мікроконтролера i8051, побудованого на основі високорівневої n-МОП технології. Аналогом цієї мікросхеми є вітчизняна мікросхема КМ1816ВЕ51.

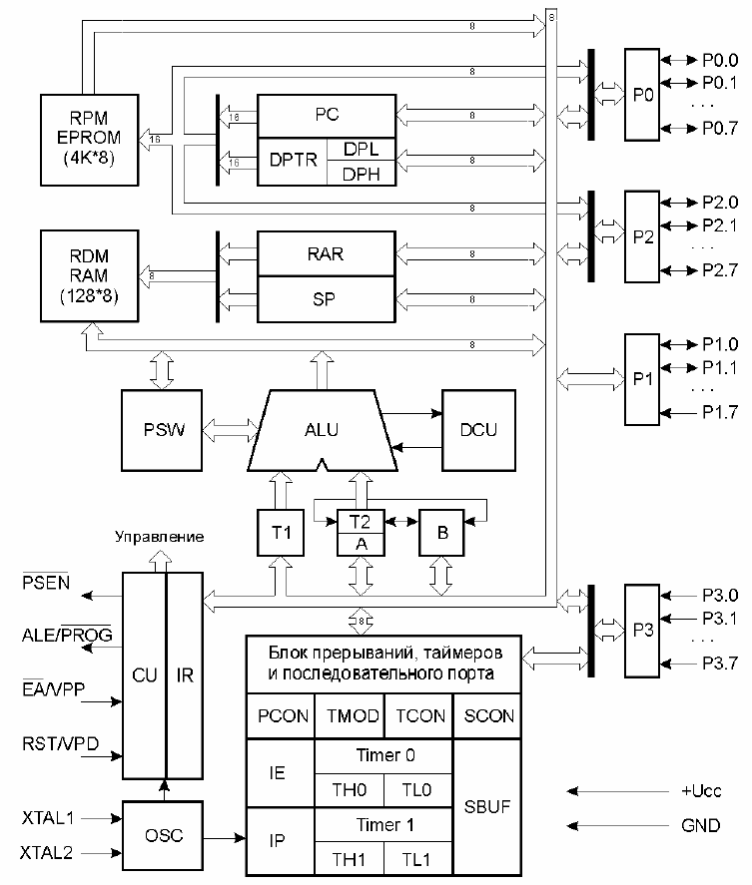

Рисунок 4 – Структурна схема мікроконтролера 8051

Охарактеризуємо коротко показані блоки, лінії та сигнали, які не описувалися вище.

Блоки:

- IR – регістр команд, зберігає код виконуваної команди;
- CU – пристрій керування;
- OSC – внутрішній генератор синхросигналів, на основі яких пристрій керування CU формує машинні такти (див. далі);
- RAR – програмно недоступний регістр-покажчик даних;
- RDM – резидентна (внутрішня) пам'ять даних (ОЗП, RAM);
- RPM – резидентна (внутрішня) пам'ять програм (ПЗП, ROM);
- P0, P1, P2, P3 – паралельні порти вводу-виводу:
- P0 – 8-бітний двонаправлений порт вводу-виводу інформації: при роботі із зовнішніми ОЗП й ПЗУ по лініях порту в режимі часового мультиплексування видається адреса зовнішньої пам'яті, після чого здійснюється передача або прийом даних.
- P1 – 8-бітний квазі-двонаправлений порт вводу-виводу: кожен розряд порту може бути запрограмований як на ввід, так і на вивід інформації, незалежно від стану інших розрядів;
- P2 – 8-бітний квазі-двонаправлений порт, аналогічний P1; крім того, виводи цього порту використовуються для видачі адресної інформації при звертанні до зовнішньої пам'яті програм або даних (якщо використовується 16-бітова адресація останньої). Виводи порту використовуються при програмуванні мікросхеми 8751 для введення в мікроконтролер старших розрядів адреси;
- P3 – 8-бітний квазі-двонаправлений порт, аналогічний P1; крім того, виводи цього порту можуть виконувати ряд альтернативних функцій, які використовуються при роботі таймерів, порту послідовного вводу-висновку, контролера переривань, і зовнішньої пам'яті програм і даних;
- T1, T2 – програмно недоступні регістри для тимчасового зберігання операндів;

- DCU – схема десяткової корекції;

Лінії та сигнали:

- $\overline{\text{PSEN}}$ – дозвіл зовнішньої пам'яті програм; видається тільки при звертанні до зовнішнього ПЗП;
- ALE – строб адреси зовнішньої пам'яті; на виході ALE звичайно формується безперервна послідовність прямокутних імпульсів із частотою резонатора Чрез / 6 і шпаруватістю 33 % (тобто, тривалість сигналу «1» удвічі перевищує тривалість «0»)
- $\overline{\text{PROG}}$ – сигнал програмування;
- $\overline{\text{EA}}$ – сигнал блокування роботи із внутрішньою пам'яттю (рівень 0 на цьому вході змушує мікроконтролер виконувати програму тільки із зовнішнього ПЗП, ігноруючи внутрішнє);
- VPP - напруга програмування;
- RST - вхід загального скидання мікроконтролера;
- VPD - вивід резервного живлення від зовнішнього джерела;
- XTAL1, XTAL2 - виводи для підключення кварцового резонатора до внутрішнього генератора синхросигналу;

Арифметико-логічний пристрій

8-бітний арифметико-логічний пристрій (ALU) може виконувати арифметичні операції додавання, вирахування, множення й ділення; логічні операції І, АБО, ВИКЛЮЧАЮЧЕ АБО, а також операції циклічного зсуву, скидання, інвертування і т. п. До входів підключені програмно-недоступні регістри Т1 й Т2, призначені для тимчасового зберігання операндів, схема десяткової корекції (DCU) і схема формування ознак результату операції (PSW).

Найпростіша операція бітового додавання використається в ALU для інкрементування вмісту регістрів, просування регістра-покажчика даних (RAR) і автоматичного обчислення наступної адреси резидентної пам'яті програм. Найпростіша операція вирахування використається в ALU для декрементування регістрів і порівняння змінних.

Найпростіші операції автоматично утворять “тандеми” для виконання таких операцій, як, наприклад, інкрементування 16-бітних регістрових пар. В ALU реалізується механізм каскадного виконання найпростіших операцій для реалізації складних команд. Так, наприклад, при виконанні однієї з команд умовної передачі керування по результаті порівняння в ALU тричі інкрементується лічильник команд (PC), двічі провадиться читання з RDM, виконується арифметичне порівняння двох змінних, формується 16-бітна адреса переходу й приймається рішення про те, робити або не робити перехід по програмі. Всі перераховані операції виконуються всього лише за 2 мкс.

Важливою особливістю ALU є його здатність оперувати не тільки байтами, але й бітами.

Окремі програмно-доступні біти можуть бути встановлені, скинуті, інвертовані, передані, перевірені й використані в логічних операціях. Ця здатність досить важлива, оскільки для керування об'єктами часто застосовуються алгоритми, що містять операції над вхідними й вихідними булевими змінними, реалізація яких засобами звичайних мікропроцесорів сполучена з певними труднощами. Таким чином, ALU може оперувати чотирма типами інформаційних об'єктів: булевими (1 біт), цифровими (4 біти), байтовими (8 біт) і адресними (16 біт). В ALU виконується 51 різна операція пересилання або перетворення цих даних. Оскільки використовується 11 режимів адресації (7 для даних й 4 для адрес), шляхом комбінування операції й режиму адресації базове число команд 111 розширюється до 255 з 256 можливих при однобайтовому коді операції.

Пристрій керування й синхронізації

Кварцовий резонатор, що підключає до зовнішніх виводів мікроконтролера, управляє роботою внутрішнього генератора OSC, що у свою чергу формує сигнали синхронізації. Пристрій керування (CU) на основі сигналів синхронізації формує машинний цикл фіксованої тривалості, рівним 12 періодам резонатора. Більшість команд мікроконтролера виконується за один машинний цикл. Деякі команди, що оперують із 2-байтовими словами або пов'язані зі звертанням до зовнішньої пам'яті, виконуються за два машинних цикли. Тільки

команди ділення й множення вимагають чотирьох машинних циклів. На основі цих особливостей роботи пристрою керування може провадитися розрахунок часу виконання прикладних програм.

На схемі мікроконтролера до пристрою керування примикає регістр команд (IR). У його функцію входить зберігання коду виконуваної команди.

Організація пам'яті МК 8051

Пам'ять програм (ПЗП)

Адресується незалежно від пам'яті даних (ОЗП). Об'єм вбудованої (резидентної) ПЗП – 4 Кб.

При використанні зовнішніх ПЗП використовується 16 біт адреси отже доступ до максимум 64 Кб ПЗП (усього).

Звертання до ПЗП здійснюється при виконанні програми (PC), а також при виконанні команди (MOVC) копіювання байта в акумулятор A (адресація за допомогою PC або DPTR).

Пам'ять даних (ОЗП)

Об’єм вбудованої пам'яті даних – 128 байт, зовнішньої – до 64 Кб (адресується за допомогою DPTR).

Перші 32 байта займають 4 банки РЗП, наступні байти – на розсуд розроблювача (стік, системна й користувальницька області даних). Програмний доступ до пам'яті даних здійснюється за допомогою прямої або непрямої (через R0 | R1) адресації. Доступ до зовнішнього ОЗП – тільки за допомогою непрямої адресації (через R0 | R1 або DPTR).

Частина пам'яті даних являє собою бітову область, до якої можна адресуватися побітово за допомогою спеціальних команд (MOV C, bit), або у вигляді *(адреса байта).(розряд)*, або у вигляді абсолютної бітової адреси (див. таблицю 4).

Таблиця 4.

Адреси бітів з індивідуальним доступом

Адреса байта	Адреси бітів (D7 … D0)
2Fh ⋮ 20h	7F … 78 ⋮ 07 … 00

Таким чином, якщо адреса 0 є операндом звичайної байтової команди MOV (наприклад, MOV A, 0), вона виступає адресою байта ОЗП, а якщо операндом бітової команди (наприклад MOV C, 0), – адресою біта ОЗП у байті 20h.

§ 16. Цоколевка та стандартна схема підключення МК x51

Цоколевка та стандартна схема включення МК сімейства x51 в ланцюг живлення наведені на рисунках 5а та 5б.

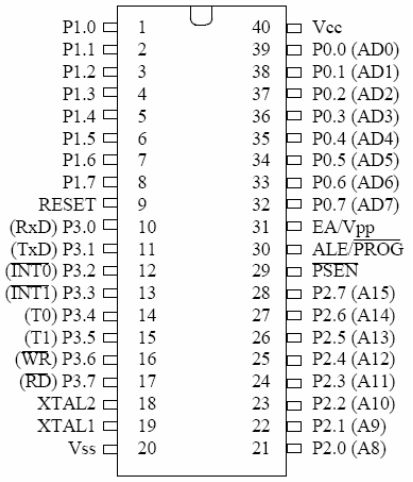

Рисунок 5а – Цоколевка МК 8051 з позначеннями виводів

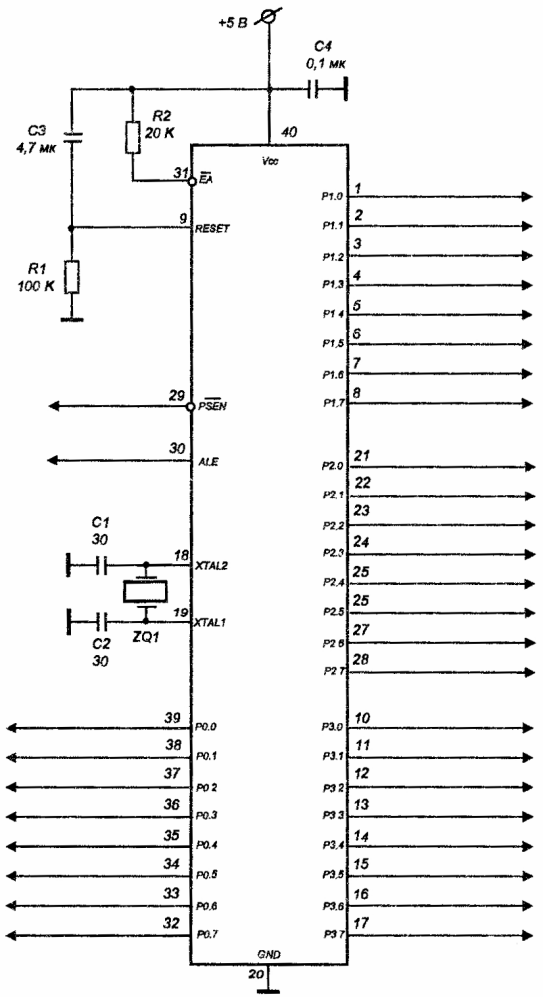

Рисунок 5б – Стандартна схема включення МК сімейства x51

Додаткові позначення на рисунках 5а й 5б:

- Vss - потенціал загального проводу ("землі")
- Vcc - основна напруга живлення +5 В

Конденсатори C1 і C2 (15...30 пф) уведені в схему для стабільності запуску резонатора ZQ1. Конденсатор C3 забезпечує подачу сигналу на вхід RST при включенні МК. При вимиканні він вирядже-

ний, і вхід RST виявляється під потенціалом, близьким до напруги живлення (Vcc), протягом десятків міилісекунд; у міру зарядки С3 цей потенціал знижується.

Контрольні питання до глави 4

1. Які блоки в структурі МК 8051 є програмно недоступними? Які функції вони виконують?
2. Які функції арифметико-логічного пристрою (ALU) МК 8051?
3. Які функції пристрою керування й синхронізації (CU) МК 8051?
4. Чим відрізняються друг від друга 4 паралельних порти МК 8051?
5. Який обсяг убудованої пам'яті даних і пам'яті програм МК 8051?
6. Яка адресація використовується при організації програмного доступу до внутрішньої й зовнішньої пам'яті даних МК 8051?
7. Що являє собою бітова область пам'яті даних?
8. Яке призначення основних виводів МК 8051? Які сигнали на них подаються або формуються?
9. Яке призначення додаткових елементів на стандартній схемі включення МК сімейства x51?

Глава 5. Сполучення МК x51 із програмно керованими мікросхемами

АЦП використовується для оцифровки й передачі в МК зовнішніх аналогових сигналів.

АЦП характеризуються принципом перетворення, швидкодією, розрядністю, точностними параметрами, живлячою напругою, діапазоном вхідних напруг, кількістю каналів та ін.

З погляду інтерфейсу АЦП діляться на паралельні й послідовні. Паралельні передають всі біти результату одночасно по індивідуальних лініях. Це значить, що з 12-розрядним АЦП МК повинен бути зв'язаний мінімум 12-ма провідниками.

Послідовні АЦП повільніше, але їхня швидкість може бути цілком достатньою для широкого класу задач (~10 мкс на весь цикл перетворення/передачі (порції даних, наприклад, обмірюваного значення з аналогового датчика)).

АЦП можуть містити внутрішні регістри різної розрядності (від 4 до 24), у які МК повинен попередньо занести певну інформацію (керуючого слова). Наприклад, для багатоканальних АЦП необхідно задати номер каналу.

Інші відмітні характеристики менш важливі, але їх також необхідно враховувати при розробці програмно-апаратного сполучення із МК.

§ 17. Сполучення з паралельним АЦП

Як приклад розглянемо сполучення МК із 12-розрядним паралельним АЦП AD7880 фірми Analog Devices. Він вимагає всього одної напруги живлення 5 В і споживає при цьому щодо невеликий струм (<10 мА).

Схема сполучення

Для зв'язку із МК необхідні 12 ліній даних і 4 лінії керування: CONVST, CS, RD (входи АЦП) і BUSY (вихід АЦП; у деяких випадках не використовується). Крім того, на вхід CLKIN АЦП необхідно подати тактову послідовність із частотою до 2,5 МГц і шпаруватістю

від 40 до 60 % (на жаль, вихід ALE МК не підходить по шпаруватості, отже необхідно використовувати окремий генератор).

Схема сполучення АЦП із мікроконтролером показана на рисунку 6.

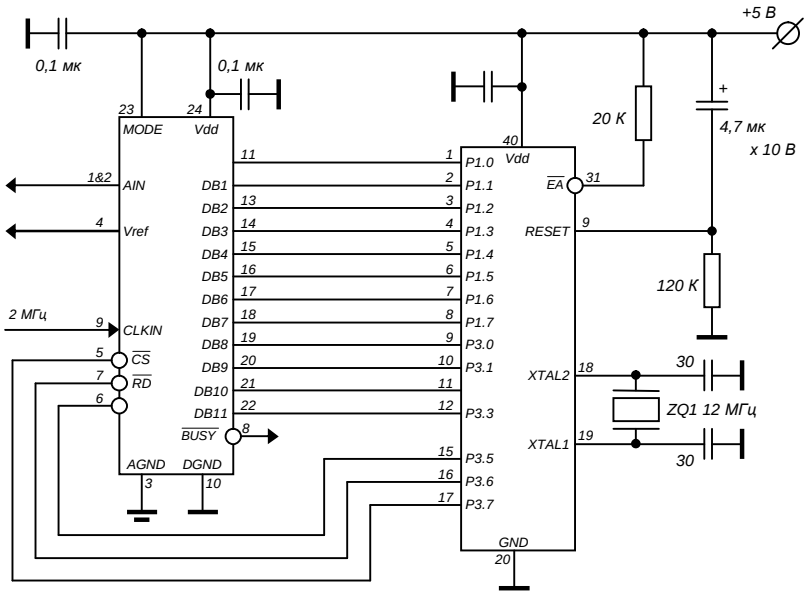

Рисунок 6 – Схема сполучення паралельного АЦП із мікроконтролером

Оскільки АЦП із МК повинні зв'язувати 15 або 16 провідників, ми змушені цілком задіяти два із чотирьох 8-розрядних портів вводу-виводу МК. Зручніше взяти лінії портів P1 і P3, оскільки в часто використовуваних апаратних засобах налагодження порти P0 і P2 використовуються для зв'язку з комп'ютером.

Алгоритм взаємодії з АЦП AD7880

10. У початковий момент часу на всіх ніжках, з'єднаних із входами керування АЦП, мікроконтролер повинен установити одиниці.

11. Для запуску перетворення на CONVST необхідно подати негативний імпульс. Перепад на ньому з 0 в 1 при одиничних рівнях сигналів на CS і RD запускає цикл перетворення.
12. Протягом приблизно 20 мкс АЦП (при тактовій частоті 2 МГц) оцифровывает сигнал і заносить його у свій вихідний регістр. У цей час він утримує нульовий рівень на своєму виході BUSY. Визначити, чи завершив АЦП перетворення, чи ні, можна або по стану сигналу BUSY, або просто відрахувати 20 мкс після подачі сигналу старту перетворення - за цей час воно завершиться.
13. Після завершення, коли BUSY повернеться в 1, дані можуть бути зчитані з АЦП. Із цією метою МК повинен установити в 0 спочатку сигнал на вході CS, а потім - на RD. Приблизно через 75 нс після установки RD в 0 на виходах даних DB0-DB11 АЦП з'явиться результат перетворення, що МК повинен зчитати.
14. Після читання МК повинен повернути спочатку RD, а потім CS у вихідний одиничний стан. На цьому цикл завершується.

Даний алгоритм можна проілюструвати такою часовою діаграмою:

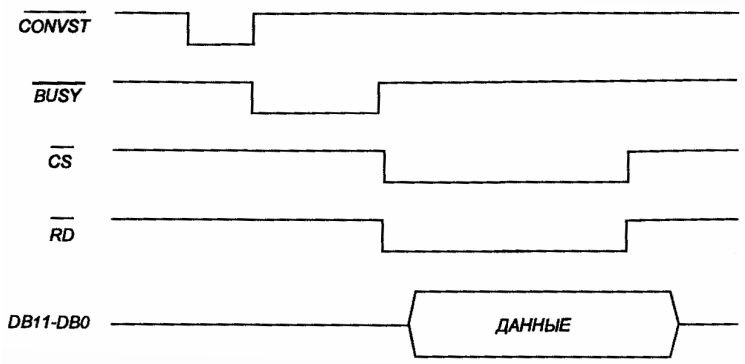

Рисунок 7 – Сигнали обміну МК із 12-розрядним паралельним АЦП AD7880

§ 18. Сполучення з послідовним АЦП

Сполучення МК із послідовним АЦП розглянемо на прикладі АЦП ADS7816 фірми Burr-Brown (8 виводів).

Схема сполучення

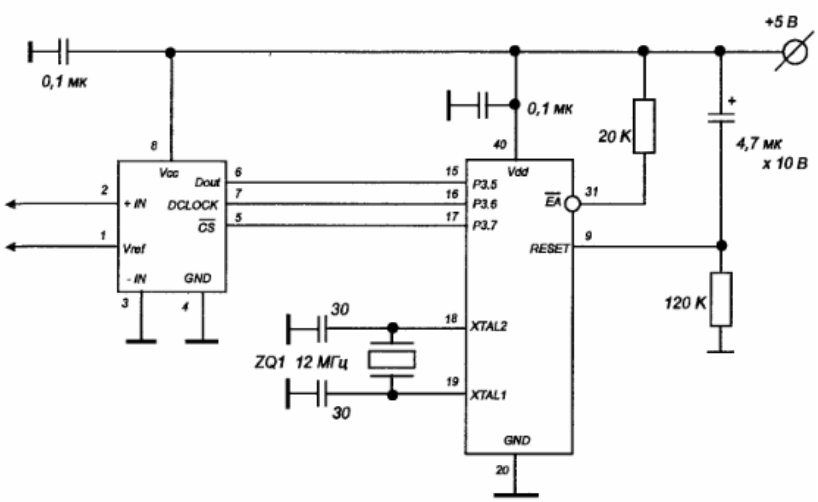

Рисунок 8 – Схема сполучення послідовного АЦП із мікроконтролером x51

Позначення на рисунку:

- +IN і -IN - аналогічні входи; -IN рекомендується з'єднати із загальним проводом;
- на V_{ref} подається опорна напруга;
- V_{cc} – живлення;
- GND - земля;
- на вхід $\overline{CS}$ подається сигнал старту перетворення;
- DCLOCK - вхід тактування АЦП (цього разу тактуємо за допомогою мікроконтролера);
- вихід Dout - послідовно переданий результат.

Алгоритм взаємодії з АЦП ADS7816

1. встановлюється ″1″ на $\overline{CS}$ й Dout, ″0″ – на DCLOCK;
2. запуск перетворювача – ″0″ на $\overline{CS}$;
3. підготовка до читання – 3 позитивних імпульси на DCLOCK;
4. зчитування старшого біта (DB11) з Dout;
5. зчитування наступного біта – молодшого;
6. наступний біт – позитивний імпульс на DCLOCK;
7. закінчення зчитування – ″1″ на $\overline{CS}$.

Часова діаграма взаємодії контролера з АЦП ADS7816 наведена на рисунку 9:

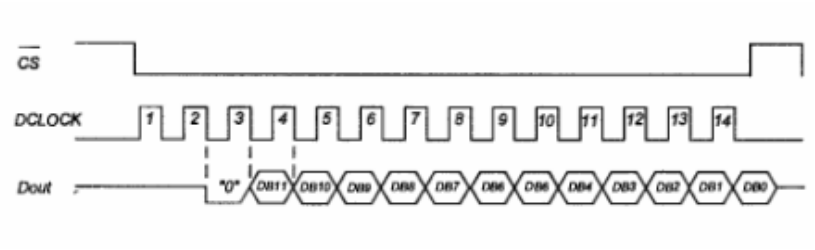

Рисунок 9 – Часова діаграма взаємодії контролера з АЦП ADS7816

§ 19. Робота МК x51 із зовнішньою пам'яттю даних (ОЗП)

Зазвичай в якості зовнішньої пам'яті даних (ОЗП) використовуються мікросхеми SRAM з байтовою організацією (x8) об'ємом 2 або 8 Кб (архітектури 2Кx8; 8Кx8). Такі мікросхеми мають 8 виводів даних (D0-D7) і 11 або 13 адресних входів (A0-A10/A12). Додатковий вхід $\overline{WE}$ визначає характер звернення (стан операції): 1 – читання; 0 – запис. Вхід $\overline{CE}$ активізує або деактивізує мікросхему (1 – вимк.; 0 – увімк.). Крім того, вхід $\overline{OE}$ дозволяє включати і відключати вихідні буфери мікросхеми від ліній даних D0-D7 (0 – приєднані – відкривається такт для виконання операції, 1 – від’єднанні – так званий «сірий» стан).

На рисунку 10 наведене зображення схемотехніки мікросхеми 6116 (КР537РУ10) ємністю 2 Кб з 11 адресними входами.

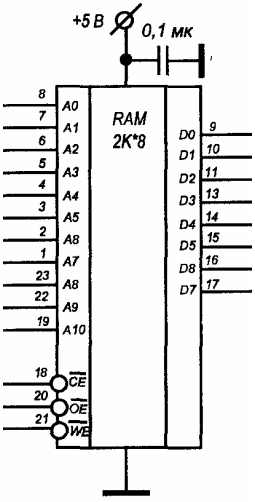

Рисунок 10 – Мікросхема 6116 (КР537РУ10)

Порядок використання мікросхем:

- при записі:
  1. встановлення адреси комірки – на адресних лініях A0-A10/A12;
  2. встановлення записуваних даних на лініях D0-D7;
  3. встановлення 0 на вході $\overline{WE}$ (запис);
  4. встановлення 0 на вході $\overline{CE}$ (характер звернення до даних мікросхеми);
  5. встановлення 0 на вході $\overline{OE}$ (мікросхема пам'яті зніме дані з D0-D7 і запише за адресою у комірці).
- при читанні:
  1. встановлення адреси A0-A13;
  2. встановлення «1» на $\overline{WE}$ (читання);
  3. після – подача 0 на $\overline{CE}$ – дані з'являться на лініях D0-D7.

4. подача 0 на $\overline{OE}$;

5. читання даних.

Часові діаграми:

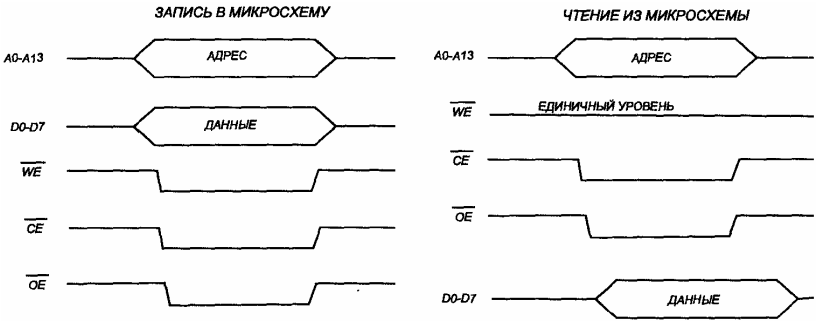

Рисунок 11 – Часові діаграми при доступі до зовнішньої пам'яті

З метою зменшення кількості задіяних виводів мікроконтролера (і збільшення незадіяних, а отже – і не залежних від роботи із зовнішньою ОЗП) лінії порту P0 мультиплексовані. У перший момент звернення до пам'яті по лініях P0 виводиться адреса A0-A7, одночасно з цим встановлюється в 1 сигнал на виході ALE. Через 2 періоди тактового генератора ALE скидається в 0, і через декілька мілісекунд адресна інформація пропадає з виходів P0, даючи можливість вести обмін даними, аби згодом мати можливість звертатися до адреси, використовуючи регістр-задвижку (8-бітовий) типа 555ИР22, інформація в якому фіксується по спаду на його вході STB (з виходу ALE).

У мікропроцесорних системах загального призначення часто використовують дешифратор для вибору певної мікросхеми пам'яті. На дешифратор подаються старші біти адреси, що не відносяться до адреси внутрішнього простору мікросхеми. Якщо мікросхем пам'яті не більше, ніж число адресних ліній мікроконтролера мінус число адресних ліній мікросхеми ОЗП, тоді можна не використовувати дешифратор, а вільні лінії підводити безпосередньо до мікросхем (входи $\overline{CE}$).

§ 20. Сполучення мікроконтролера з індикаторами різних типів

У вбудованих системах управління незрідка передбачається необхідність відображення різної інформації в ході роботи системи за допомогою індикаторів різних типів. Розглянемо варіанти сполучення мікроконтролера x51 з індикаторами різних типів.

Сполучення мікроконтролера з рідкокристалічним індикатором на основі контролера типу HT1610

Взаємодія МК з індикатором на основі власного контролера здійснюється порівняно просто завдяки відсутності необхідності на низькому рівні управляти кожним сегментом. Замість цього МК передає індикатору підготовлені коди символів, і останній самостійно управляє процесом відображення кожного з них.

Рідкокристалічний індикатор на основі контролера типа HT1610 є двосторонньою друкарською платою, з одного боку якої знаходиться 10-розрядний 7-сегментний ЖК-дисплей (див. рисунок 12), а з іншої – електронні компоненти: контролер, кварц, ємності. Такий індикатор часто використовується, наприклад, в телефонних апаратах.

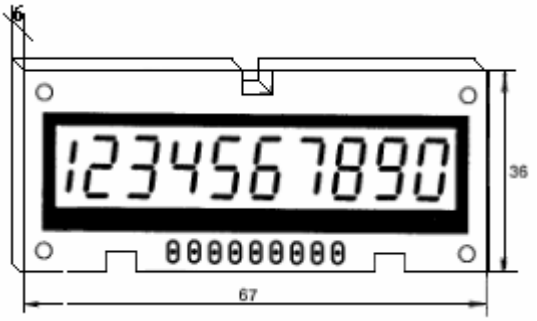

Рисунок 12 – Зовнішній вигляд 10-розрядного 7-сегментного рідкокристалічного індикатора

Схема з'єднання індикатора з мікроконтролером (живлення 1,5 В) наведена на рисунку 13.

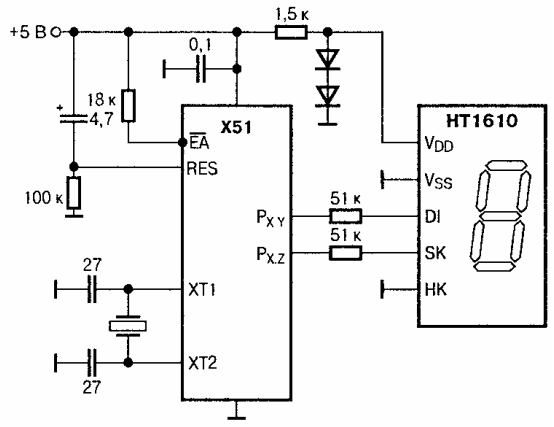

Рисунок 13 – З'єднання індикатора з мікроконтролером на основі контролера типа HT1610

Призначення виводів:

- V_{DD} – живлення (1,5 В);
- V_{SS} – виводиться на землю (0);
- HK – також виводиться на землю, що переводить мікроконтролер в режим зовнішнього управління;
- DI і SK – сигнальні лінії для передачі інформації: DI – сама числова інформація, SK – тактові імпульси (готовність даних).

Алгоритм взаємодії контролера HT1610 та мікроконтролера схематично наведен на рисунку 14:

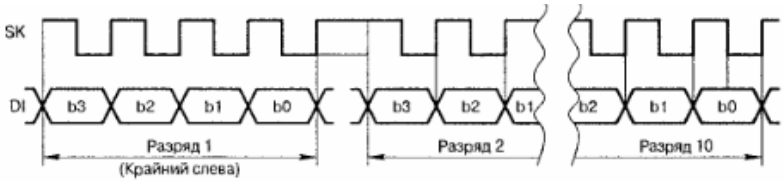

Рисунок 14 – Алгоритм взаємодії контролера HT1610 і мікроконтролера x51

Після закінчення передачі SK потрібно встановити в (1(, DI – неважливо (хоча краще подати «1», заздалегідь уникаючи можливого шунтування нулем на DI).

Відповідність переданих код знакам, що відображуються, наведена в таблиці 5:

Таблиця 5.

Відповідність кодів контролера HT1610 знакам, що відображуються

Біти 3 2 1 0	Знак
0 0 0 0	(пропуск)
0 0 0 1	1
0 0 1 0	2
0 0 1 1	3
0 1 0 0	4
0 1 0 1	5
0 1 1 0	6
0 1 1 1	7
1 0 0 0	8
1 0 0 1	9
1 0 1 0	0
1 0 1 1	F
1 1 0 0	⌋
1 1 0 1	L
1 1 1 0	P
1 1 1 1	–

Сполучення зі світлодіодним індикатором типа АЛС318

АЛС318 – багаторозрядний 7-сегментний світлодіодний індикатор з об'єднаними катодами. Анодами управляє дешифратор типа КР514ИД1 (на нього подається код знаку, а він активізує потрібні сегменти). Катодами управляє або другий дешифратор, або мікроконтролер. Розглянемо перший варіант, оскільки він вимагає використання меншого числа виводів мікроконтролера.

При використанні 9 розрядів необхідніший дешифратор «4 в 16» (наприклад, 155ИД3). Три мікросхеми DD2, DD3 і DD4 можна було б замінити однією ПЛІС, але ми розглянемо саме таку схему, тому що вона дозволяє краще зрозуміти призначення функціональних блоків.

Для роботи з індикатором використовується порт P1: 4 молодших біта – виведення коду цифри (код збігається із значенням) на входи індикатора (аноди сегментів). Ще один – десяткова точка. Катоди управляються дешифратором DD4. На інформаційні входи DD4 поступають сигнали з трьох старших бітів P1. Установка P1.4 в 1 засвічує десяткову точку розряду, катодний вивід якого встановлюється в 0 відповідним виходом DD4 (див. рисунок 15).

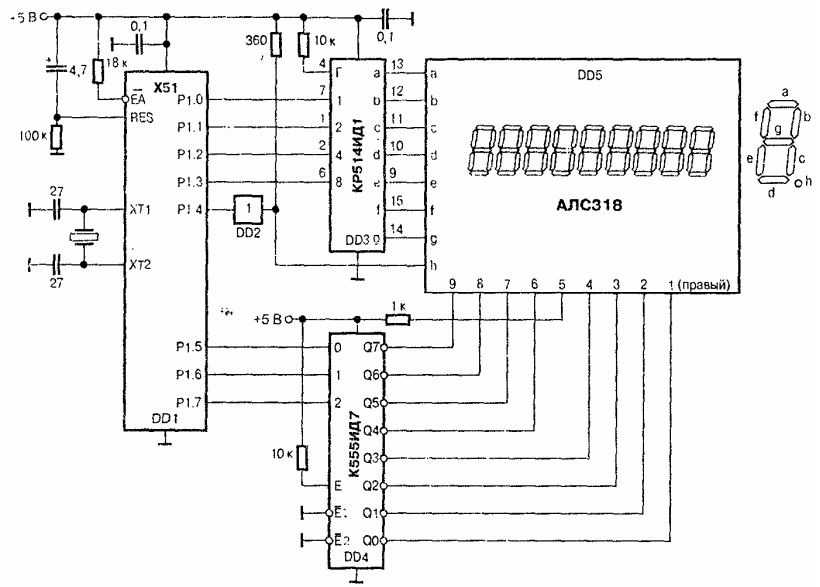

Рисунок 15 – Схема сполучення МК x51 зі світлодіодним індикатором АЛС318 із загальними катодами

Оскільки на аноди подається код 1 знаку, необхідно виводити всі розряди по черзі (система динамічної індикації), кілька разів обнуляючи відповідний катод розряду.

Специфіка індикатора із загальними анодами:

- катоди управляються дешифратором КР514ИД2 (а не ИД1);
- аноди підключаються до дешифратора DD4 через буферні транзистори.

Схема сполучення такого індикатора показана на рисунку 16.

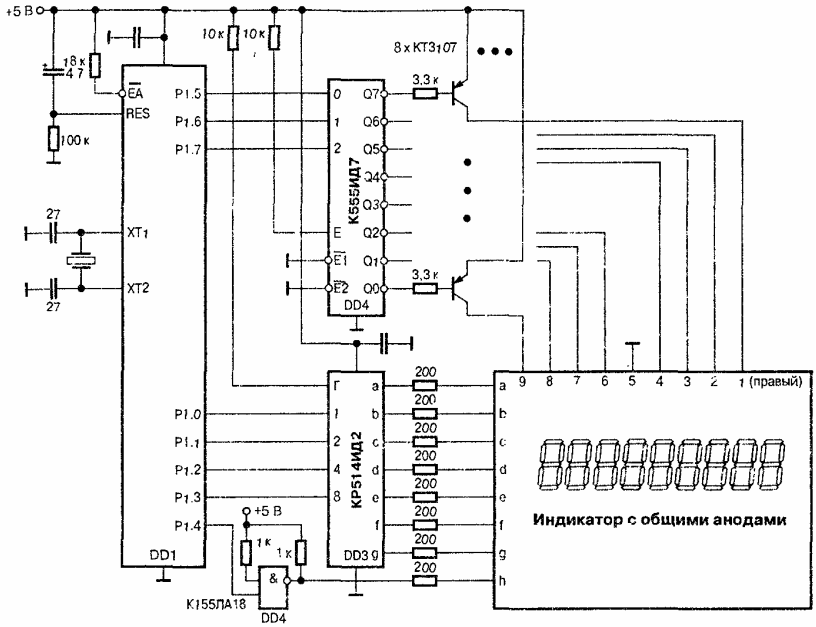

Рисунок 16 – Схема сполучення МК х51 зі світлодіодним індикатором АЛС318 із загальними анодами

При необхідності використання багаторозрядного табло можна набрати необхідне число розрядів з одиночних індикаторів, підібравши дешифратор відповідної розрядності (див. рисунок 17). Схема сполучення аналогічна багаторозрядному індикатору із загальним катодом.

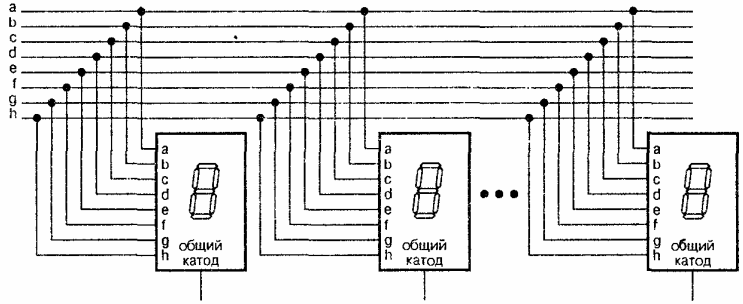

Рисунок 17 – Багаторозрядний індикатор, що складається з окремих однорозрядних

Окрім розглянутих типів індикаторів поширені також індикатори матричного типа, в яких кожен символ відображується за допомогою матриці 5х7 або 5х10 точок. Такі індикатори також можуть бути набірними з окремих блоків-символів, одно- або багаторядковими.

Контрольні питання до глави 5

1. Яке призначення АЦП в мікропроцесорних системах управління? Наведіть приклади його практичного використання.
2. На які класи поділяються АЦП з точки зору інтерфейсу? Які переваги і недоліки кожного класу?
3. Які лінії МК зручно використовувати для передачі сигналу з паралельного АЦП і чому?
4. По яких ознаках МК може визначити завершення циклу оцифрування даних паралельним АЦП?
5. Яким чином можна тактувати роботу послідовного АЦП при його взаємодії з МК х51?
6. Який порядок звернення до мікросхеми SRAM при записі даних з МК х51?
7. Який порядок звернення до мікросхеми SRAM при читанні даних в МК х51?
8. Для чого використовується мультиплексування порту Р0 при взаємодії МК х51 із зовнішньою пам'яттю даних?
9. У чому полягає особливість взаємодії МК х51 з індикатором на основі власного контролера?
10. У чому полягає особливість взаємодії МК х51 з індикатором без вбудованого контролера?

Глава 6. Таймери-лічильники МК x51

§ 21. Загальні відомості

Ми ознайомилися з трьома важливими аспектами архітектури МК x51:

- типовими схемами включення;
- організацією пам'яті і регістрів;
- системою команд.

Це вже дозволяє розробляти прості пристрої і програми для них.

Познайомимося тепер з організацією периферійних внутрішніх пристроїв і системою їх використання і обслуговування. Використання внутрішньої периферії часто полегшує вирішення певних завдань і дозволяє вирішувати нові. Почнемо з таймерів.

При розробці програм часто необхідно формувати затримки певної тривалості. До цих пір ми підбирали необхідне число команд з відомою тривалістю виконання, проте під час таких затримок мікроконтролера не можуть виконувати корисну роботу. Це критично, якщо затримки великі відносно часу всього циклу роботи мікроконтролера (умовно > 50%), а також, якщо під час затримок необхідно виконувати інші дії (виміри, відображення тощо).

Таймер-лічильник (T/C, Timer/Counter) може спрощено розглядатися як 16-розрядний лічильник (регістр T0 або T1), на вхід якого може подаватися сигнал з частотою (1/12) внутрішнього тактового генератора мікроконтролера або сигнал із зовнішньої лінії мікроконтролера. Забезпечується можливість за допомогою спеціальних команд запускати його на рахування імпульсів, зупиняти, прочитувати або записувати число, дізнаватися про досягнення лічильником максимального значення (0FFFFh) (тобто, про наявність переповнювання).

Перший програмний таймер-лічильник був випущений Intel в середині 70-х років у вигляді окремої мікросхеми 8253 (КР580ВІ53). Ця ж мікросхема була інтегрована в мікроконтролер 8051 в двох екземплярах (T/C0 та T/C1).

Якщо на вхід таймера-лічильника поступають сигнали ззовні, він виступає як лічильник зовнішніх імпульсів, а коли з тактового генератора (точніше, з дільника Д частоти на 12) – як таймер.

§ 22. Структура таймера-лічильника

Структура таймера-лічильника наведена на рисунку 18.

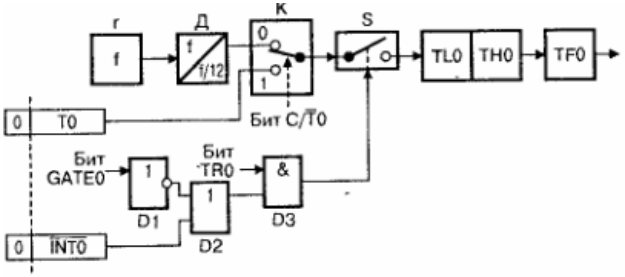

Рисунок 18 – Спрощена структура таймера/лічильника Т/С0

Входи Т0 (до Т/С0) і Т1 (до Т/С1) є лініями порту Р3 – Р3.4 і Р3.5; інші біти порту Р3 теж багатофункціональні (див. таблицю 6).

Таблиця 6.

Альтернативні функції ліній порту Р3

Лінія	Позначення	Функція
Р3.0	RxD	Лінії послідовного порту
Р3.1	TxD	
Р3.2	INT0	Входи зовнішніх переривань
Р3.3	INT1	
Р3.4	T0	Входи до таймерів
Р3.5	T1	
Р3.6	$\overline{WR}$	Виходи «запис» і «читання» при використанні зовнішнього ОЗП
Р3.7	$\overline{RD}$	

Альтернативні функції виконуються після запису «1» в регістр-задвижку відповідних ліній порту Р1. Певна лінія порту містить регістр-задвижку (SFR). Порт Р3 теж містить додаткові елементи для чтения/записи альтернативних сигналів. Одні команди мікроконтролера забезпечують читання виводів порту (тобто зовнішніх сигналів),

інші – читання SFR порту. Для порту P1,2 і 3 необхідно занести в SFR «1», аби використовувати відповідну лінію для введення інформації. До речі, наявність внутрішнього навантаження (опір) у вихідних каскадах портів P1-P3 забезпечує в режимі введення даних живлення пристроїв, що підключені до порту. Запис «1» в SFR P0 переводить лінії P0 в високоімпедансний стан.

Адреси регістрів Т/С:

TL0 – 8Ah

TH0 – 8Ch

TL1 – 8Bh

TH1 – 8Dh

TMOD – 89h } регістри керування

TCON – 88h

Склад бітів регістра TMOD проілюструємо рисунком:

$GATE_1$	C/T_1	$M1_1$	$M0_1$	$GATE_0$	C/T_0	$M1_0$	$M0_0$
7	6	5	4	3	2	1	0
для Т/С1				для Т/С0			

Рисунок 19 – Біти регістра TMOD

Отже, якщо необхідно змінити тривалість зовнішнього імпульсу, потрібно:

1. скинути біт С/Т для переводу в режим таймера;
2. очистити байтові регістри TL і TH;
3. встановити біт GATE «1» (при цьому забезпечується залежність від $\overline{INF}$) і біт TR в «1»;
4. подати вимірюваний імпульс позитивної полярності на вхід $\overline{INF}$.

Поки на $\overline{INF}$ буде 1, таймер-лічильник працюватиме і рахуватиме імпульси з дільника Д (1/12) (при 12 МГц МК – з частотою 1 МГц). З цього виходить, що коли $\overline{INF}$ обнулиться, то в TH-TL виявиться тривалість імпульсу в мікросекундах.

§ 23. Режими роботи таймерів-лічильників

Як випливає з опису керуючих бітів TMOD, в режимі 0 обидва Т/С працюють однаково; те ж справедливо для режимів 1 і 2. Режими роботи 3 для Т/С0 і Т/С1 різні. Розглянемо коротко роботу Т/С в кожному з режимів.

- **Режим 0.** Переведення будь-якого Т/С в режим 0 робить його схожим на таймер КМ1816ВЕ48 (восьмибітовий лічильник), до входу якого підключений п'ятибітовий переддільник частоти на 32. У цьому режимі таймерний регістр має розрядність 13 бітів. При переході із стану "всі одиниці" в стан "всі нулі" встановлюється прапор переривання від таймера TF1. Вхідний синхросигнал таймера 1 дозволений (поступає на вхід Т/С1), коли керуючий біт TR1 встановлено в 1 або керуючий біт GATE (блокування) встановлений в 0, або на зовнішній вивід запиту переривання INT1 поступає рівень 1. Відзначимо попутно, що установка біта GATE в 1 дозволяє використовувати таймер для виміру тривалості імпульсного сигналу переривання, що подається на вхід запиту.
- **Режим 1.** Робота будь-якого Т/С в цьому режимі така ж, як і в режимі 0, за винятком того, що таймерний регістр має розрядність 16 біт.
- **Режим 2.** У цьому режимі робота організована таким чином, що переповнювання (перехід із стану "всі одиниці" в стан, "всі нулі") восьмибітового лічильника TL1 приводить не лише до установки прапора TF1 (див. рисунок 20), але і автоматично перезавантажує в TL1 вміст старшого байта (TH1) таймерного регістра, яке заздалегідь було задано програмним шляхом. Перевантаження залишає вміст TH1 незмінним. У режимі 2 Т/С0 і Т/С1 також працюють абсолютно однаково.

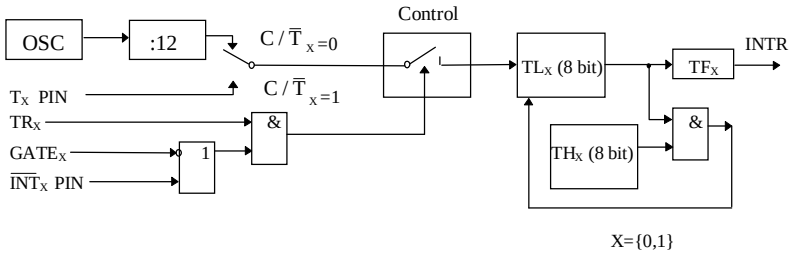

Рисунок 20 – логіка роботи таймера в режимі 2

- **Режим 3.** У режимі 3 T/C0 і T/C1 працюють по-різному. T/C1 зберігає незмінним свій поточний вміст. Іншими словами, ефект такий же, як і при скиданні керівника біта TR1 в 0. У режимі 3 TL0 і TH0 функціонують як два незалежні восьмибітові лічильники. Роботу TL0 визначають керуючі біти T/C0 (C/T, GATE TR0), вхідний сигнал INT0 і прапор переповнювання TF0. Роботу TH0, який може виконувати лише функції таймера (підрахунок машинних циклів МК), визначає керуючий біт TR1. При цьому TH0 використовує прапор переповнювання TF1. Режим 3 використовується в тих випадках, коли потрібна наявність додаткового восьмибітового таймера або лічильника подій. Можна вважати, що в цьому режимі МК 8051 має в своєму складі три таймер-лічильники. У разі ж, якщо T/C0 використовується в режимі 3, T/C1 може бути або вимкнений, або переведений в режим 0, 1 або 2, або може бути використаний послідовним портом як генератор частоти передачі.

Найчастіше на практиці використовуються режими 2 і 3.

У модернізованих моделях мікроконтролерів сімейства x51 може бути присутнім третій таймер лічильник T/C2 і (або) блок програмних лічильників PCA, які теж можуть бути використані для відліку часових інтервалів.

Контрольні питання до глави 6

1. Яке призначення таймерів-лічильників у складі МК х51?
2. Які комірки використовує таймер-лічильник для накопичення кількості підраховуваних імпульсів?
3. У яких випадках таймер-лічильник виступає як таймер, а в яких – як лічильник?
4. Які альтернативні функції покладені на лінії порту Р3?
5. Які дії необхідно виконати для виміру тривалості зовнішнього імпульсу за допомогою таймера-лічильника?
6. У яких режимах можуть функціонувати таймери-лічильники МК 8051? Які з є найбільш споживаними?

Глава 7. Система переривань МК x51

§ 24. Загальні відомості про систему переривань

Система переривань мікроконтролерів сімейства x51 де в чому схожа на типову систему переривань архітектури x86. Існує набір подій, що викликають переривання, наслідком чого є виклик відповідного обробника переривань.

На відміну від механізму переривань x86 (де адреси обробників беруться з таблиці векторів), в архітектурі x51 адреси обробників є зумовленими константами. При виникненні перериваючої події адреса обробника завантажується в PC, і мікроконтролер починає виконувати програму обробника.

Серед джерел переривань в мікроконтролерах можуть використовуватися наступні події: поява на певному вході імпульсу; досягнення внутрішнім лічильником максимального значення (точніше, переповнювання лічильника); прийом або передача байта вбудованим приймачем; зниження напруги живлення нижче заданого рівня.

Більшість мікроконтролерів сімейства x51 мають як мінімум 5 джерел переривань. Перерахуємо їх, вказуючи зумовлену адресу обробника:

- від T/C0 (установка прапора TF0) – 000Bh
- від T/C1 (установка прапора TF1) – 001Bh
- від INT0 (перепад з 1 в 0 і установка прапора IE0) – 0003h
- від INT1 (перепад з 1 в 0 і установка прапора IE1) – 0013h
- від вбудованого послідовного приймача (при завершенні операції прийому (прапор R1) або передачі байта (прапор T1) – 0023h

Підпрограма-обробник переривання повинна завершуватися командою RETI, що відрізняється від RET тим, що окрім повернення управління вона дозволяє переривання, які автоматично були заборонені після виклику підпрограми.

§ 25. Структура керуючих регістрів

Прапори TF, IE, TR, а також IT входять до складу регістра управління /статусу **TCON** (адреса 88h):

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

Рисунок 21 – Структура регістра управління /статусу TCON

TF – прапор переповнювання лічильника таймера, встановлюється апаратно; скидається апаратно при обслуговуванні переривання;

TR – біт управління T/C, скидання і установка здійснюються програмно для зупинки і пуску таймера-лічильника (відповідно);

IE – прапор фронту (зрізу) переривання INT, встановлюється апаратно при зрізі на INT; скидається апаратно при обслуговуванні (див. далі);

IT – біт управління типом переривання зі входу INT; скидання і установка здійснюються програмно для специфікації запиту INT, тобто вибору низького рівня або зрізу як джерела переривання INT.

Якщо IT = 0, то джерелом переривання INT є низький рівень на INT. При цьому скидання IE здійсниться лише тоді, коли підпрограма-обробник впливає на джерело сигналу на INT, що змусить його встановити його в ”1” (або коли саме джерело не ініціює 1 на INT). Наступне переривання станеться після скидання і нової установки IE. Якщо ж IT = 1, то джерелом переривання є перехід з 1 в 0 на вході INT. Поки прапор переривання не скинутий, мікроконтролер не «бачить» нових запитів від джерела переривань.

Структура регістра дозволу переривань **IE** (адреса 0A8h):

EA	-	-	ES	ET1	EX1	ET0	EX0
7	6	5	4	3	2	1	0

Рисунок 22 – Структура регістра дозволу переривань IE

EA – глобальне блокування переривань (0 – заборонені); скидання і установка здійснюються програмно;

ES – біт дозволу (1) / заборони (0) переривань від приймача;

ET – біт дозволу (1) / заборони (0) переривань від T/C;

EX – біт дозволу (1) / заборони (0) переривань від входу INT.

§ 26. Пріоритети переривань

Кожному з 5 переривань може бути призначений свій пріоритет (низький “0” або високий “1”).

Після старту мікроконтролера всі переривання мають низький (тобто, однаковий) пріоритет. Першими обслуговуються переривання, які виникли раніше (надходження сигналу переривання при виконанні обробника ігнорується).

Управління пріоритетами здійснюється за допомогою регістра **IP** (адреса 0B8h).

-	-	-	PS	PT1	PX1	PT0	PX0
7	6	5	4	3	2	1	0

Рисунок 23 – Структура регістра пріоритету переривань IP

PS – біт пріоритету переривань від приймача;

PT – біт пріоритету переривань від T/C;

PX – біт пріоритету зовнішніх переривань INT.

Запис 1 у відповідний біт встановлює високий пріоритет. Тоді відповідний обробник зможе перервати низькоприорітетний обробник. Поки виконується обробник, інші переривання виявляються заборонені (до його завершення командою RETI).

Контрольні питання до глави 7

1. Що відбувається в мікроконтролері x51 при виникненні перериваючої події?
2. Які сигнали в МК x51 використовуються як перериваючі події?
3. Коли вирішуються переривання, заборонені при активізації обробника переривання, що виникло раніше?

4. Яка роль регістра управління / статусу TCON в системі переривань МК x51?
5. Яка роль регістра дозволу переривань IE в системі переривань МК x51?
6. Скільки пріоритетів переривань є в МК 8051? Для чого використовуються ці пріоритети?
7. Яка роль регістра пріоритету переривань IP в системі переривань МК x51?

Глава 8. Послідовний порт (UART) МК x51

§ 27. Основи використання UART

Послідовний порт мікроконтролера x51 інакше називається «Універсальний асинхронний приймач-передавач» (УАПП, UART). Призначений він для прийому і передачі інформації, представленої послідовним кодом (молодші біти – першими) в повному дуплексному режимі.

Структурна схема UART приведена на рисунку 24.

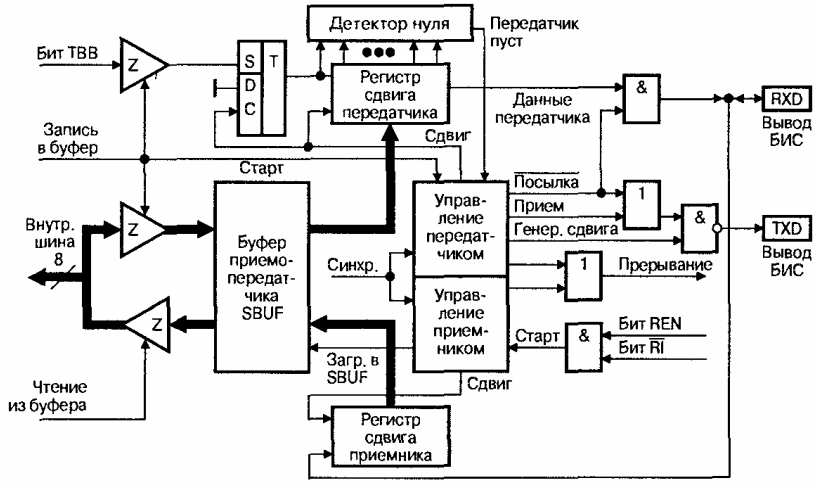

Рисунок 24 – Структурна схема UART мікроконтролера 8051

До складу UART входять:

- приймаючий зсувний регістр, оскільки інформацію необхідно приймати побітово, зсуваючи кожен біт,
- передавальний зсувний регістр, оскільки передавати також необхідно побітово,
- буферний регістр SBUF,

а також 2 службових регістра:

- регістр управління /статусу – SCON,

- біт SMOD регістра управління потужністю PCON (розглядається далі).

Запис байта в SBUF призводить до автоматичного копіювання його в зсувний регістр передавача і ініціалізує початок передачі байта. Біти, що висуваються із зсувного регістра передавача починають послідовно передаватися по лінії TxD мікроконтролера.

Аналогічно, байт, прийнятий ззовні (з лінії RxD, побітово) в приймаючий зсувний регістр, автоматично копіюється в SBUF. Наявність SBUF при прийомі дозволяється поєднати дві операції: 1) читання і аналізу попереднього байта з SBUF і 2) прийому чергового байта в приймаючий зсувний регістр (поки наступний не буде остаточно прийнятий).

UART може працювати в 4-х режимах:

- №0 – (так званий зсувний регістр розширення вводу-виводу) прийом і передача здійснюються через вивід входу приймача RxD 8-бітовими порціями; через вивід виходу передавача (TxD) видаються імпульси зсуву, що супроводжують будь-який біт. Частота передачі Чпер (як і максимальна частота прийому) дорівнює 1/12 частоти кварцевого резонатора (Чрез).
- №1 – 8-бітовий приймач-передавач із змінною швидкістю передачі) – прийом порцій даних здійснюється через вхід RxD, передача – через TxD по 10 біт: стартовий біт (молодший, 0), 8 бітів даних (1-8), і стоп-біт (старший, 9). При прийомі в біт RB8 регістра SCON заноситься (копіюється) стоп-біт. Частота прийому/передачі залежить від T/C1.
- №2 – (9-бітовий приймач-передавач з фіксованою швидкістю передачі) – прийом здійснюється через RxD, передача – через TxD порціями по 11 біт: старт-біт (0), 8 бітів даних (1-8), програмний біт (9) і стоп-біт (10). Сенс і призначення програмного біта визначаються програмою; часто це, наприклад, біт контролю парності з PSW.0. При прийомі цей біт поміщається в RB8 в SCON, а стоп-біт втрачається. Частота передачі визначається програмою і може дорівнювати 1/32 або

1/64 частоти кварцевого резонатора залежно від біта SMOD (біт подвоєної швидкості – при значенні ”1”).

- №3 – аналогічний режиму 2, лише Чпер залежить також від T/C1, а не лише від SMOD.

Передача ініціюється інструкцією розміщення даних в SBUF. Прийом – при виявленні перепаду з 1 в 0 на вході приймача (RxD), при цьому REN має дорівнювати ”1” (а в режимі №0 – RI має дорівнювати ”0”).

Регістр управління/статусу UART **SCON** (розташований за адресою 98h) призначений для управління режимом роботи UART і перериваннями. Його структура ілюструється рисунком 25.

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

Рисунок 25 – Структура регістра управління / статусу SCON

Призначення бітів регістра:

- Біти SM1-SM0 – задають номер режиму роботи UART; встановлюються та скидаються програмно.
- SM2 – біт управління спеціальним режимом роботи UART: заборони прийому повідомлення (10 біт), в якому 9-й біт дорівнює ”0” (при SM2 = 1).
- REN – біт дозволу прийому (при значенні ”1”).
- TB8 – передача біта 8; встановлюється та скидається апаратно для завдання 9-го біта при 9-бітовому режимі передачі (режим 2 і 3 UART).
- RB8 – прийом біта 8; встановлюється та скидається апаратно для фіксації біта 9 в режимі 9-бітового приймача (режим 2 і 3 UART).
- TI – прапор переривання передавача (встановлюється апаратно після закінчення передачі байта; скидається програмно після обслуговування переривання).

- RI – прапор переривання приймача (встановлюється апаратно після закінчення прийняття байта; скидається програмно після обслуговування переривання).

Швидкість (частота) прийому / передачі (як вже вказувалося) в різних режимах визначається по різному. У режимі ”0” вона залежить лише від частоти резонатора: Чпер = (1/12)Чрез. У режимі 1, 2, 3 вона також залежить від управління біта SMOD у складі PCON.

Структура регістра управління потужністю **PCON** (87h) показана на рисунку 26.

MOD	-	-	-	GF1	GF0	PD	IDL
7	6	5	4	3	2	1	0

Рисунок 26 – Структура регістра управління потужністю PCON

Призначення бітів регістра:

- При SMOD = 1 швидкість передачі удвічі вища, ніж при = 0.
- GF1, GF0 – прапори загального призначення (специфікуються користувачем).
- PD – біт зниженої потужності (при значенні ”1”).
- IDL – біт холостого ходу (при ”1” і PD = 0).

PCON вважається скинутим при записі в нього значення 0xxx0000b (де x – будь-який біт).

У режимі 2 частота передачі визначається по формулі

$$Ч_{пр/пер} = \frac{2^{SMOD} * Ч_{кварц.рез.}}{64}$$

З цього виходить, що при SMOD = 0: Чпер = Чрез / 64, а при SMOD = 0: Чпер = Чрез / 32.

У режимі 1 і 3 Чпер. залежить від Т/С1, точніше, від частоти його переповнювання $Ч_{OVLT1}$:

$$Ч_{пер} = \frac{2^{SMOD} * Ч_{OVLT1}}{32} \quad (1)$$

Переривання від Т/С1 при цьому має бути заблоковане.

Режим роботи Т/С ролі не грає. При роботі в режимі з автоперезавантаженням (TMOD = 0010xxxxb):

$$Ч_{пер.} = \frac{2^{SMOD} * Ч_{рез.}}{32*12*(256 - TH1)}$$

Чпер (частота роботи UART), таким чином, може вагатися в діапазоні від 110 Гц (режими 1 і 3, Т/С в режимі 1 (16-бітовий), перезавантажуване число = 0FEEEBh – перезавантаження програмним дорогою) до 1 МГц (режим 0) при Чрез = 12 МГц.

Частота прийому може не дорівнювати частоті передачі.

§ 28. Використання UART для організації взаємодії пристроїв

Особливості реалізації підпрограм для використання UART

Як вказувалося вище, прийом і передача даних за допомогою UART здійснюється з певною швидкістю, залежною як від мікроконтролера, так і від зовнішнього пристрою.

Для завдання і підтримки певної швидкості зручно використовувати таймер в режимі з автоперезавантаженням (режим №2).

Налаштування самого UART залежить від вибраного режиму його роботи. Найбільш простим є режим №1 (асинхронний обмін 10-бітовими порціями, що включають старт- і стоп-біти, з налаштованою швидкістю передачі.

Підпрограма налаштування мікроконтролера перед використанням UART в режимі №1

Основні кроки:

8. заборонити переривання (див. раніше);
9. налаштувати Т/С1 (або 0) на режим №2 відповідно до вибраної швидкості обміну (на режим №2 – 8-бітовий Т/С з автоперезавантаженням TL1 з TH1);

10. запустити T/C1.

Виберемо для прикладу швидкість обміну 9600 бит/с (Чпр/пер = 9,6 кГц).

Відповідно до специфікації SCON:

- для режиму №1 UART необхідно задати SM1-0 = 01b ;

- для режиму №1 UART SM2 = 0 (немає необхідності контролювати стоп-біт);

- при передачі REN можна скинути, при прийомі – необхідно встановити;

- у режимі №1 UART біт TB8 не використовується (біти RB8, T1 і R1 встановлюють апаратно).

Отже, в SCON необхідно занести 00001001b = 09h

Отже, текст підпрограми ініціації UART:

```
Serinit:  mov   IE, #0        ; заборона переривань
          mov   TMOD, #20h    ; режим №2 для T/C1
          mov   TH1, #RATE    ; константа для швидкості 9600 бод.
          mov   TL1, #RATE    ; потім перезавантажуватиметься з TH1
          anl   PCON, #7Fh    ; скидання біта SMOD
          mov   SCON, #50h    ; для режиму №1
          setb  TR1           ; старт T/C1
          (ret)
```

Розрахунок константи RATE для певної швидкості обміну

Відповідно до формули (1), в режимі №1

$$Ч_{пр/пер} = \frac{2^{SMOD} * Ч_{OVLT1}}{32}$$

При SMOD = 0

$$Ч_{пр/пер} = \frac{Ч_{OVLT1}}{16}$$

$Ч_{OVLT1}$ в режимі з автоперезавантаженням TL1 з TH1

$$Ч_{OVLT1} = \frac{Ч_{рез.}}{12*(100h - RATE)}$$
$$(256) \quad (TH1)$$

$$Ч_{пр/пер} = \frac{2^{SMOD} * Ч_{рез.}}{32*12*(256 - TH1)}$$

$$TH1 = 256 - \frac{2^{SMOD} * Ч_{рез.}}{32*12*Ч_{пр/пер}}$$

SMOD = 0, $Ч_{рез}$ = 12МГц, $Ч_{пр/пер}$ = 9600 Гц

$$TH1 = 256 - \frac{2^0 * 12000}{32*12*9.6} = 256 - \frac{1000}{32*9.6} = 252{,}745 = 253 = 0FCh$$

Обчислене значення можна описати в програмі як константу з ідентифікатором: RATE EQU 0FCh.

Підпрограма прийому байта

```
getch:    jnb   RI, getch     ; прочитування - лише при RI = 1
                              ; (сигнал закінчення прийому)
          mov   A, SBUF       ; читаємо в ACC прийнятий байт
          clr   RI            ;скидання RI після закінчення
                              ; обслуговування прийому
          ret
```

Підпрограма посилки байта

Основні кроки:

1. записати байт в SBUF
2. почекати установки TI (закінчення посилки)
3. скинути TI

```
putch:    mov   SBUF, A  ; вантажимо посиланий байт з ACC
send:     jnb   TI, send ; чекаємо, поки TI не дорівнюватиме "1"
          clr   TI
          ret
```

§ 29. Використання каналу RS-232 для послідовного прийому-передачі

У стандартній логіці значення ”1” представляється рівнем напруги від 2,4 В до 5 В, а ”0” – від 0 до 0,8 В. Прі передачі ж по каналу RS-232 ”1” і ”0” кодуються однаковими по величині (від 5 – 12 В), але різними по знаку сигналами [14, с. 308].

Отже, сполучаючи UART мікроконтролера x51 з каналом RS-232, необхідно передбачити певні засоби перетворення. Це можуть бути каскади з 3-4 транзисторів, 2-3 діодів і близько 10 резисторів, або готові перетворювачі, такі як MAX202E від Maxim або AD232 від Analog Devices. Останні містять перетворювач напруги з Останні містять перетворювач напруги з ±5 В в ±10 В (такий, як MAX680) і каскади (2 каскади, що підтримують 2 канала приймача, – ми скористаємося лише одним) перетворення стандартних логічних сигналів в сигнали RS232.

Частина схеми зв'язку МК 8051 з каналом RS232 наведена на рисунку 27.

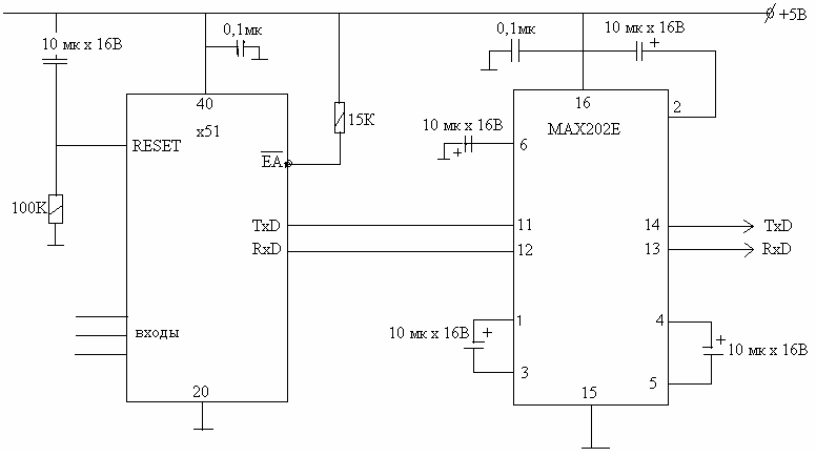

Рисунок 27 – Фрагмент схеми зв'язку МК 8051 з каналом RS232

§ 30. Приклад організації взаємодії МК 8051 і ПК x86 через UART-RS232

Один з пристроїв має бути ведучим (що віддає команди), а інше – веденим (що їх приймає і виконує). Ведучим виберемо персональний комп'ютер (тоді мікроконтролер нічого не робитиме самостійно без команди персонального комп'ютера).

Необхідно продумати алгоритм взаємодії, що враховує всі можливі варіанти поведінки системи. При цьому зручно виділити стани, в

яких можуть знаходитися пристрої, – учасники взаємодії, і змалювати їх у вигляді діаграми станів.

Приймемо, що в нашій системі дані поступають лише від мікроконтролера до персонального комп'ютера.

Стани мікроконтролера

Виділимо наступні стани мікроконтролера:

- *Wait* – стан чекання, в якому мікроконтролер (пристрій на основі мікроконтролера) опиняється відразу після включення. Перед безпосередньою взаємодією мікроконтролер повинен виконати ініціалізацію (1). Вона виконується по команді NUL від персонального комп'ютера. Окрім ініціалізації мікроконтролер повинен послати у відповідь сигнал ACK (якщо все пройшло успішно). При помилці мікроконтролер повинен послати сигнал NAK.
- *Ready* – стан готовності, в який мікроконтролер потрапляє зокрема після ініціалізації (і не лише). У даному стані мікроконтролер чекає запиту персонального комп'ютера на посилку деяких даних (назвемо цей запит XON). Після прийняття цього запиту мікроконтролер повинен почати передачу – в новому стані.
- *Sending* – посилка (відправка). У цьому стані мікроконтролер готує дані (повідомлення), що відправляються, а потім по частинах (байтами) посилає в персональний комп'ютер. Після закінчення передачі мікроконтролер переходить автоматично в особливий стан.
- *Sent* – в цьому стані мікроконтролер чекає підтвердження від персонального комп'ютера успішної передачі. У нас повідомлення складається з декількох (трьох) байтів. Тоді можливі три варіанти відповіді персонального комп'ютера:
  - XOFF – успішний прийом, більше повідомлень не потрібно;
  - XON – потрібні ще повідомлення;
  - NAK – помилка при прийомі, у тому числі невірні дані.

Після XOFF ми повертаємося в стан Ready, після XON – в стан Sending (передача чергової порції). Після NAK, швидше за все, потрібна повторна пересилка останній порції даних (знову стан Sending).

Діаграма станів мікроконтролера змальована на рисунку 28.

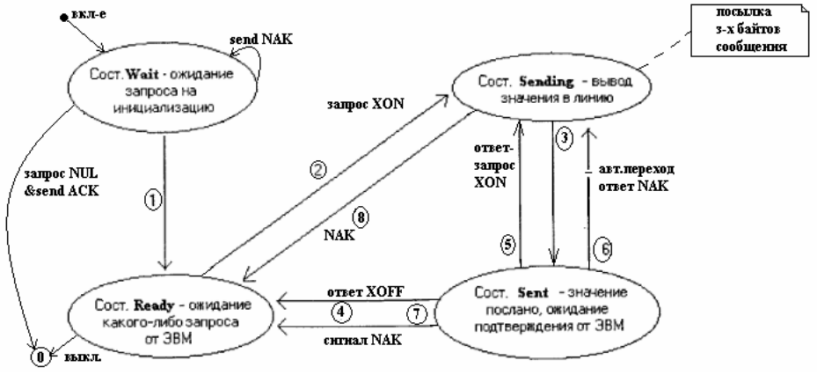

Рисунок 28 – Діаграма станів мікроконтролера

Також необхідно передбачити можливість виникнення помилки введення-виводу і можливість перервати обмін з мікроконтролером. Як варіант, в цих випадках персональний комп'ютер може послати запит NUL, у відповідь на який мікроконтролер знову виконає ініціалізацію (переходи (7) і (8)).

На додаток до діаграми вкажемо, що при здобутті невідомого сигналу мікроконтролер повинен відповісти сигналом NAK.

Як певний сигнал зручно розглядати прийом-передачу певного байта-коду, наприклад, символу або просто числа.

Стани персонального комп'ютера

Тепер розглянемо стани персонального комп'ютера:

- *Init* – стан підготовки пристрою до передачі даних; персональний комп'ютер потрапляє в нього після натиснення користувачем певної клавіші (по певній команді користувача) і посилає мікроконтролеру сигнал NUL. У відповідь персонального комп'ютера чекає від мікроконтролера сигнал ACK, якщо все OK, або NAK, якщо сталася помилка ініціалізації мі-

кроконтролера. У останньому випадку персональний комп'ютер повинен перейти в завершальний стан Done.

- *Ready* – стан підготовки персонального комп'ютера до прийому байтів повідомлення від мікроконтролера. Спочатку (і в більшості випадків) запит на посилку байта – сигнал XON. Можливий також запит NAK на повторну посилку останнього або декількох байтових повідомлень, якщо набуте невірного або неповного значення (якщо воно складається з декількох байтів). Далі персональний комп'ютер автоматично переходить в стан прийому.
- *Receiving* – в стані прийому персональний комп'ютер прочитує байти повідомлення (3 байти в прикладі). Оскільки байти поступають від мікроконтролера з певною швидкістю, персональний комп'ютер може чекати певний час чергового байта. Якщо цей час витік, виникає помилка введення-виводу (якщо не задавати тайм-аут, персональний комп'ютер може зависати, чекаючи байта, який він не встиг прийняти). Оскільки наше повідомлення складається з трьох байтів, введемо два тайм-аути: 1) чекання чергового повідомлення; 2) чекання чергового байта в повідомленні. Можливі причини виходу із стану Receiving:
    1. витікання тайм-ауту 2) – тоді необхідно сигналом NAK запитати повторну передачу повідомлення і перейти в стан Ready;
    2. помилка введення-виводу, зареєстрована за допомогою регістра стану COM-порта персонального комп'ютера, – тоді необхідно (як найкращий варіант) переініціалізувати і персональний комп'ютер, і мікроконтролер (стан Init);
    3. отримані всі (3) символи повідомлення – тоді необхідно перейти до аналізу повідомлення в наступному стані.
- *Received* – стан аналізу отриманого повідомлення – перевірка на приналежність байтів певним діапазонам (або наявність певних кодових байтів). При виявленні недопустимих байто-

вих значень персональний комп'ютер вимагає повторної посилки за допомогою сигналу NAK і повертається в стан Ready. Якщо ж всі байти допустимі, персональний комп'ютер підтверджує вдалий прийом сигналом XOFF і переходить в стан Done (або XON, якщо потрібне ще повідомлення, а потім – Receiving).

- *Done* – кінцевий стан персонального комп'ютера після успішного прийому повідомлення або невдалої ініціалізації. Для випадку успішного прийому в ньому може відображуватися або зберегатися отримане значення. Для іншого випадку, вочевидь, необхідно видати повідомлення про помилку, при цьому можна приймачем виводу передати код помилки замість значення, що виводиться, а її написати з врахуванням можливого отримання коду помилки.

Діаграма станів персонального комп'ютера приведена на рисунку 29.

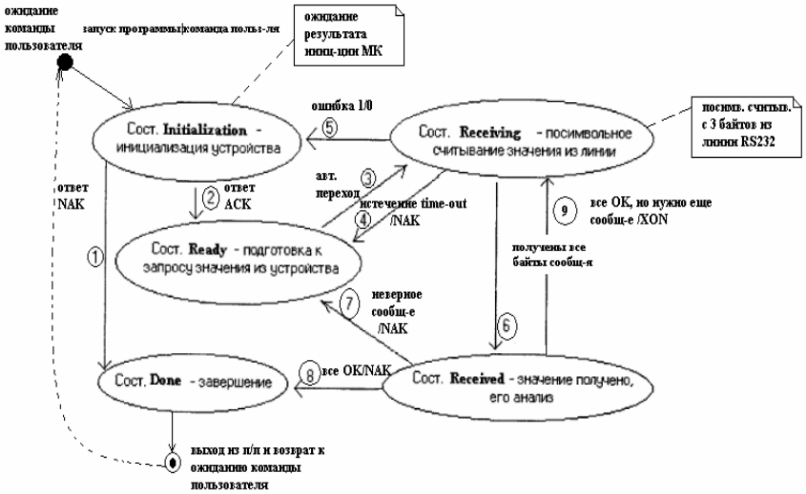

Рисунок 29 – Діаграма станів мікроконтролера

Особливості реалізації підпрограм для персонального комп'ютера

Спочатку необхідно визначити наявність COM-портів на персональному комп'ютері і визначити їх адреси. Вони зберігаються в

області даних BIOS по адресах: 40h:0, 2, 4 і 6 для портів COM1, 2, 3 і 4 відповідно.

Кожен COM-порт має базову адресу (наприклад, 3F8h) і декілька додаткових байтів з адресами: [база + 1] (3F9h), +2, +3, +4, +5 – кожен для певного призначення.

Далі необхідно налаштувати вибраний COM-порт на певну швидкість обміну і формат одиничного інформаційного пакету (наявність і кількість старт- і стоп-бітів, наявність біта парності). Перед налаштуванням бажано на всяк випадок деякий час почекати завершення обміну даними (за готовність COM-порта відповідає байт статусу [база+5]).

Відправка байта через COM-порт здійснюється записом за базовою адресою (перед цим необхідно почекати закінчення посилки попереднього байта, перевіряючи значення [база+5]).

Процедура прийому байта заснована на читанні байта з [база]. Заздалегідь необхідно переконатися у відсутності помилок введення-виводу, перевіряючи байт статусу [база+5]. При цьому можливі ситуації:

- попередній байт не був вчасно лічений;
- помилка парності;
- помилка синхронізації;
- у буфері ще немає символу.

§ 31. Робота UART в мультимікропроцесорних системах

Мультимікропроцесорні системи будуються, в основному, як системи децентралізованного управління устаткуванням. Такі системи використовуються для управління (регулювання) в топологічно розподілених об'єктах (прокатних станах, рухливому складі ЖД і метро, складальних конвеєрах, лініях гнучких автоматизованих виробництв (ГАП) та ін.). При цьому виникає завдання обміну інформацією між безліччю мікроконтролерів, об'єднаних в єдину локальну обчислювально-управляючу мережу. Як правило, локальна мережа на основі 8051 має магістральну архітектуру з моно каналом, що розділяється (загальною шиною) (коаксіальний кабель, вита пара,

оптоволокно), по якому здійснюється обмін інформацією між мікро-ЕОМ (див. рисунок 30).

Рисунок 30 – Структура типової розподіленої системи управління з моноканалом

У регістрі SCON (див. рисунок 25) є біт SM2, за допомогою якого можна в режимі 2 і 3 UART реалізувати міжпроцесорний обмін. У режимі 2 і 3 біт 9 переданого пакету потрапляє в RB8, а його значенням можна управляти (на відміну від режиму 1, при якому в RB8 потрапляє стоп-біт, завжди рівний ”1”).

Далі, установкою біта SM2 можна вирішувати переривання від приймача лише при RB8 = 1.

9-й біт, рівний ”1”, може означати, що даний байт є ідентифікатором мікроконтролера-одержувача. Це свого роду елемент простого протоколу інформаційного обміну між взаємодіючими пристроями.

Тоді при отриманні широкомовного повідомлення від провідної мікро-ЕОМ ведені підсистеми (мікроконтролери) отримають байт-ідентифікатор і можуть бути запрограмовані так, що викличуть відповідні обробники переривання. Обробник порівнює байт-ідентифікатор з власною мережевою адресою мікроконтролера, і якщо вони збігаються, мікроконтролер скидає біт SM2 і готується до прийому даних. Якщо не збігаються, обробник не скидає SM2, а просто повертає управління основній програмі, ігноруючи, таким чином, повідомлення, послане іншій системі управління нижнього рівня.

При SM = 1 (у ведених мікроконтролерів) інформаційні байти, в яких біт 9 рівний ”0”, переривань не викличуть, тобто просто ігноруватимуться.

Для автономних мікроконтролерів біт SM2 може бути використаний в режимі 1 роботи UART для контролю стоп-біта. В режимі 0 UART біт SM2 не використовується і має бути скинутий.

Контрольні питання до глави 8

1. Для чого призначений послідовний порт (UART) мікроконтролера x51?
2. Які зовнішні лінії МК x51 використовуються для послідовного обміну даними через UART?
3. У яких режимах може працювати UART МК x51? Чим відрізняються режими 0 і 1?
4. Чим відрізняються режими 1 і 2 роботи UART МК x51?
5. Чим відрізняються режими 2 і 3 роботи UART МК x51?
6. Яка операція починає процес передачі, а яка – процес прийому через UART МК x51?
7. Яка роль регістра управління/статусу SCON при використанні UART МК x51?
8. Яка роль регістра управління потужністю PCON при використанні UART МК x51?
9. Як визначається швидкість передачі в різних режимах роботи UART?
10. Які дії необхідно виконати для налаштування мікроконтролера перед використанням UART?
11. Як налаштувати UART на певну швидкість обміну?
12. У чому полягає особливість апаратного сполучення UART МК з каналом RS-232?
13. За яким загальним принципом виконується моделювання взаємодії пристроїв в термінах станів?

14. Які графічні елементи використовуються при побудові діаграми станів пристрою?
15. За яким принципом організовується обмін інформацією між мікро-ЕОМ з використанням UART в розподілених системах управління устаткуванням?

Глава 9. Особливі режими роботи мікроконтролера МК х51

§ 32. Режими роботи мікроконтролера із зниженим енергоспоживанням

У багатьох варіантах використання мікроконтролера енергоспоживання є одним з основних параметрів. У цих випадках доцільно використовувати k-МОН-версії мікроконтролера, оскільки саме в них зазвичай є можливості зниження енергоспоживання (у n-МОН-варіантах зустрічаються рідко).

k-МОН-версії мікроконтролера мають 2 режими зниженого енергоспоживання:

1. режим холостого ходу;
2. режим вимкненої напруги живлення (Power Down Mode) (вище названий режимом зниженої потужності).

У режимі холостого ходу (IDL = 1) генератор G мікроконтролера працює, подаючи тактові сигнали на схему переривань, UART і T/C (див. рисунок 31). Регістри і паралельні порти зберігають своє значення. Проте, синхросигнал генератора на CPU відключається.

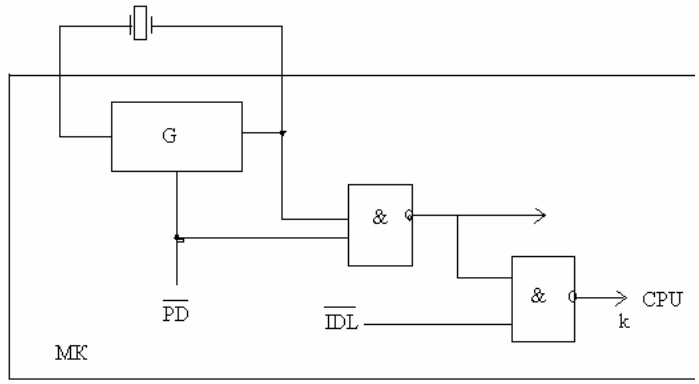

Рисунок 31 – Схема функціонування блоків МК х51 в режимі холостого ходу

На рисунку $\overline{PD}$ і $\overline{IDL}$ – біти Power Down і холостого ходу відповідно, вхідні до складу регістра PCON. У n-МОН-варіантах мікроконтролера PCON зазвичай містить лише біт SMOD. В будь-якому разі, в невживані (резервні) біти не рекомендується заносити 1.

У режимі Power Down (PD=1) генератор G відключається (зупиняється). Регістри і паралельні порти зберігають своє значення. У даному режимі можна відключити мікроконтролер від напруги живлення на вході Vcc (ніжка 40). Для збереження вмісту ОЗП його потрібно живити від резервного джерела через вхід RESET (ніжка 9).

Розглянемо режими зниженого енергоспоживання детальніше.

Режим холостого ходу

Відразу після встановлення біту IDL (PCON.0) в ”1” виконання програми припиняється. Оскільки T/C, UART і система переривань продовжують тактуватися, вони здатні відстежувати сигнали, що поступають. При використанні внутрішнього ОЗП паралельні порти зберігають значення відповідних SFR, а при використанні зовнішнього ОЗП P0 переходить у високоімпенданський стан. На виходах ALE і PSEN встановлюється ”1”. Припинення холостого ходу здійснюється одним з 2-х способів:

а) виклик будь-якого переривання – апаратне скидання PCON.0, потім виконання обробника, а після команди RETI – продовження перерваної програми; для індикації того, що переривання сталося саме під час холостого ходу, можуть використовуватися прапори GF0 і GF1 (прапори загального призначення в PCON) – їх (або один з них) можна встановити перед перекладом мікроконтролера в режим холостого ходу; обробник переривання може перевірити стан GFx;

б) апаратне скидання протягом двох машинних циклів (24 періоди коливань кварцевого резонатора) – автоматично стирає PCON.0, і CPU продовжує виконувати програму; апаратні особливості реалізації вимагають, щоб перші дві команди після холостого ходу не зверталися до ОЗП або паралельних портів (тобто такі команди не слід розташовувати за командою переведення в режим холостого ходу). Після

апаратного скидання вміст внутрішніх регістрів паралельних портів перевизначається.

Режим Power Down

Відразу після встановлення PCON.1 в ”1” генератор зупиняється. Як наслідок, припиняють роботу T/C, UART і схема переривань. За наявності резервного живлення вбудоване ОЗП і регістри SFR, у тому числі паралельні порти, зберігають своє значення. На виводах ALE і PSEN – ”0”.

Вихід з режиму Power Down здійснюється лише апаратним скиданням контролера. Він перевизначає все SFR, але не змінює вмісту вбудованого ОЗП.

У даному режимі Vcc можна знизити до 2 В (але не *до* переходу в даний режим!). Апаратне скидання не повинне подаватися раніше, ніж напруга на вході Vcc досягне свого нормального рівня і утримається протягом мінімум 10 мс (поки генератор G перестартує і стабілізується).

§ 33. Покроковий режим роботи мікроконтролера 8051

Як вже наголошувалося, наладка ВСУ може бути вельми складним завданням. Апаратні відладчики дороги, а програмні не завжди доступні або ефективні (важко емулювати роботу деяких зовнішніх пристроїв). На щастя, окрім «методу того, щоб уважно вдивлятися» є ще можливість покрокового виконання програми, що налагоджується.

В мікроконтролера 8048 є вивід, подачею на який спеціального сигналу можна перевести мікроконтролер в режим чекання. В 8051 такого спеціального виводу немає. Проте, можна запрограмувати один з виводів INT (наприклад, INT0 (P3.2)) на спрацьовування по рівню вхідного сигналу.

Пригадаємо деякі властивості переривань мікроконтролера 8051: запит на переривання (IRQ) не обслуговується, поки не завершиться обробка переривання з найбільшим пріоритетом і доки після RETI не виконується хоч би одна яка-небудь команда (простіше кажучи, поки не здійсниться вихід з обробника). Далі, обробник IRQ

ніколи не перериває сам себе (тобто, немає реєнтерабельних обробників).

Обробник IRQ від INT0, наприклад, необхідно завершити командами:

```
jnb   P3.2 $        ; чекати, поки INT0 = 0
jb    P3.2 $        ; чекати, поки INT0 = 1
reti                ; повернення з ISR
```

Вихід з обробника здійснюється лише після позитивного імпульсу на INT0. Після цього виконається перша команда, і відразу ж знову викличеться обробник від INT0, оскільки на нім – ”0” (низький рівень) (відповідний біт IT в TCON має бути рівний ”0” для вибору низького рівня як джерело переривань). Таким чином, при будь-якому імпульсі на INT0 мікроконтролер виконує лише одну команду.

Контрольні питання до глави 9

1. Які режими роботи із зниженим енергоспоживанням реалізовані в МК 8051? Чим вони відрізняються один від одного?
2. Яким чином здійснюється перехід МК в режим холостого ходу і вихід з нього?
3. Яким чином здійснюється перехід МК в режим Power Down і вихід з нього?
4. Що забезпечує збереження вмісту зовнішнього і внутрішнього ОЗП під час переходу МК в режим Power Down?
5. Для чого використовується покроковий режим роботи мікроконтролера x51?
6. Як у відсутність спеціального режиму чекання реалізувати покроковий режим роботи мікроконтролера x51?

Глава 10. Версії мікроконтролерів, оснащені ПЗП з ультрафіолетовим стиранням

Мікросхема 8051 і її вітчизняний аналог 1816ВЕ51 оснащені масочно програмованим внутрішнім ПЗП, що передбачає однократне програмування на заводі-виготовлювачі. Самостійно програмувати такі варіанти мікроконтролерів можна лише за наявності зовнішнього ПЗП.

Версії мікроконтролерів, оснащені ПЗП з ультрафіолетовим стиранням (УФ-ПЗП) можуть бути багато разів перепрограмовані, що значно підвищує зручність використання при розробці нестандартних програм. Від Intel – це версії 8751 (N-МОП) і 87С51 (ДО-МОН). Радянська мікросхема 1816ВЕ751 (аналог 8751) на практиці не зустрічається.

§ 34. Програмування мікроконтролера з УФ-ПЗП

Процедура програмування мікроконтролера з УФ-ПЗП декілька розрізняється залежно від базової технології його виробництва:

- n-МОП – при напрузі Uпрг на вході $\overline{EA}$/Vpp (ніжка 31) від 21 до 25 В; запис кожного байта – одним імпульсом 50-мілісекунди; загальний час програмування – 4 мин.
- k-МОН – Uпрг = 12,75 В; запис кожного байта – послідовністю з 25 100-микросекундных імпульсів; загальний час програмування – близько 13 секунд.

Підключення мікроконтролера N-МОП до програматора показано на рисунку 32:

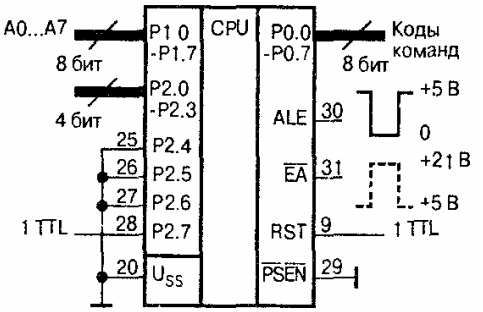

Рисунок 32 – Схема підключення мікроконтролера n-МОП до програматора

Мікроконтролер повинен працювати на зниженій частоті 4,6 МГц із-за необхідності мультиплексування на внутрішній шині адресної і кодової інформації.

Як видно на рисунку, адреси подаються на виводи P1-P2, а коди команд (завантажувані байти) – на P0. Виводи P2.4-6 і $\overline{PSEN}$ мають бути заземлені. На P2.7 і RST подається логічна «1». На $\overline{EA}$ / Vpp підтримується рівень +5 В. В момент завантаження байта $\overline{EA}$ / Vpp підключається до джерела U = +21 В, при цьому на ALE повинен утримуватися 0 протягом як мінімум 50 мс (потім – знову +5 В). Джерело +21 В має бути добре стабілізоване (зниження Uпрг може привести до збою програмування, а перевищення більш, ніж на 1 В – до пошкодження мікросхеми).

§ 35. Захист від зовнішнього читання УФ-ПЗП в МК х51

Розробниками з Intel для мікроконтролерів з УФ-ПЗП передбачений так званий біт захисту, що перешкоджає (при 1) доступ на читання до внутрішнього ПЗП ззовні.

При цьому пристрій і його програма виявляються захищені від несанкціонованого копіювання.

Запис біта захисту здійснюється подібно до запису програми, за виключенням:

- на P2.6 необхідно подати «1»;

- значення на P0, P1 і P2.0-3 можуть бути довільними.

Очищення біта захисту (штатним чином) здійснюється шляхом повного стирання вмісту УФ-ПЗП.

У новіших версіях мікроконтролерах система захисту була ускладнена з метою перешкодити вибірковому стиранню біта захисту вузьконаправленим ультрафіолетовим випромінювачем.

Якщо біт захисту дорівнює 0, вміст УФ-ПЗП може бути прочитаний з метою перевірки правильності (верифікації) завантаження програми.

Читання УФ-ПЗП може здійснюватися за допомогою програматора. Адреса подається як і при записі. На P2.7 необхідно подавати «0» в якості строб-сигнала читання.

§ 36. Реакція УФ-ПЗП на світло

Вікно мікроконтролера необхідно закривати непрозорою наклейкою. Це необхідно для захисту не лише ПЗП, але і інших внутрішніх елементів (у тому числі ОЗП), які можуть почати збої внаслідок іонізації кремнію при попаданні світла на кристал або кремнієву підкладку.

Стирання УФ-ПЗП здійснюється джерелом УФ-випромінювача з довжиною хвилі < 400 нм. При потужності 12 мВт/см^2 кварцевої лампи і відстані до вікна мікроконтролера 1.2 см достатньо 10-15 хвилин для надійного стирання інформації. Дуже тривале опромінення (> 20 хвилин) може вивести мікросхему з ладу.

У спектрі сонячного і люмінесцентного світла присутнє випромінювання з довжиною хвилі < 400 нм. Перебування відкритої мікросхеми на сонячному світлу більше тижня або під люмінесцентними лампами більше 3 років може привести до спотворення вмісту УФ-ПЗП.

Контрольні питання до глави 10

1. Якого роду інформація записується в УФ-ПЗП мікроконтролера?
2. Які параметри програмування УФ-ПЗП мікроконтролера, побудованого на основі технології n-МОН?
3. Які параметри програмування УФ-ПЗП мікроконтролера, побудованого на основі технології k-МОН?
4. Які засоби передбачені в МК з УФ-ПЗП для захисту від читання пам'яті програм ззовні?
5. Яким чином здійснюється стирання вмісту УФ-ПЗП?

Література

1. Белов А. В. Конструирование устройств на микроконтроллерах.– СПб.: Наука и техника, 2005.– 256 с.
2. Белов А. В. Самоучитель по микропроцессорной технике.– СПб.: Наука и техника, 2003.– 224 с.
3. Бродин В. Б., Калинин А. В. Системы на микроконтроллерах и БИС программируемой логики.– М.: ЭКОМ, 2002.– 400 с.
4. Зотов В. Ю. Проектирование встраиваемых микропроцессорных систем на основе ПЛИС фирмы XILINX.– М.: Горячая линия – Телеком, 2006.– 520 с.
5. Ибрагим К. Ф. Основы электронной техники: элементы, схемы, системы. 2-е изд. – М.: Мир, 2001.– 398 с.
6. Каспер Э. Программирование на языке Ассемблера для микроконтроллеров семейства i8051.– М.: Горячая линия – Телеком, 2004.– 191 с.
7. Новиков Ю. В. Основы цифровой схемотехники. Базовые элементы и схемы. Методы проектирования.– М.: Мир, 2001. – 379 с.
8. Петров И. В. Программируемые контроллеры. Стандартные языки и приемы прикладного проектирования / Под ред. В. П. Дьяконова.– М.: СОЛОН-Пресс, 2004. – 256 с.
9. Предко М. Руководство по микроконтроллерам. Том 1.– М.: Постмаркет, 2001.– 416 с.
10. Предко М. Руководство по микроконтроллерам. Том 2.– М.: Постмаркет, 2001.– 488 с.
11. Точчи Р., Уидмер Н. Цифровые системы. Теория и практика, 8-е издание.– М. : Издательский дом "Вильямс", 2004. – 1024 с.
12. Угрюмов Е. П. Цифровая схемотехника.– СПб.: БХВ – Санкт-Петербург, 2000.– 528 с.
13. Фрунзе А. В. Микроконтроллеры? Это же просто! Т. 1.– М.: ООО «ИД СКИМЕН», 2002.– 336 с.

14. Фрунзе А. В. Микроконтроллеры? Это же просто! Т. 2.– М.: ООО «ИД СКИМЕН», 2002.– 392 с.
15. Фрунзе А. В. Микроконтроллеры? Это же просто! Т. 3.– М.: ООО «ИД СКИМЕН», 2003.– 224 с.
16. Швец В. А., Шестакова В. В., Бурцева Н. В., Мелешко Т. В. Одноплатные микроконтроллеры. Проектирование и применение.– К.: МК-Пресс, 2005.– 304 с.
17. Berger A. S. Embedded Systems Design. An Introduction to Processes Tools and Techniques.– CMP Books, 2002.– 237 p.
18. Braunl T. Embedded Robotics, Mobile Robot Design and Applications with Embedded Systems.– Springer, 2006.– 458 p.
19. Catsoulis J. Designing embedded hardware.– O'Reilly, 2005.– 400 p.

Нєнов Олексій Леонідович

ПРОЕКТУВАННЯ ВБУДОВАНИХ КОМП'ЮТЕРНИХ СИСТЕМ

Підписано в друк XX.XX.2009 р. Формат 60×84 1/16.
Умовн. друк. арк. 3,1. Наклад прим.
Надруковано видавницький центром ОГАХ.
65082, Одеса, вул. Дворянська, 1/3.