Лекція 10

Розробка простих цифрових пристроїв

Вступ

Щоб створити складний пристрій, необхідно ще володіти прийомами системотехніки, тобто:

1. **вміти на підставі аналізу функцій, які повинен виконувати пристрій у цілому, спроектувати його структуру,**
2. **сформулювати принципи взаємодії вузлів,**
3. **чітко визначити всі завдання, які повинен вирішувати кожний з вузлів, виробити вимоги до окремих вузлів.**

І тільки потім, після всієї цієї попередньої роботи вже можна переходити до розробки вузлів, власне кажучи до схемотехніки.

Прийоми системотехніки сформулювати, формалізувати, описати, навіть перелічити набагато складніше, ніж прийоми схемотехніки.

Проектування складного, надійно працюючого цифрового пристрою з мінімальними апаратними витратами те саме що й мистецтво і вимагає від розроблювача певних здібностей, навіть таланту. Навчити цьому практично неможливо. Більше того, спроба навчити системотехніки може навіть принести шкоду, тому що обмежить творчі здатності розроблювача декількома твердими стандартними алгоритмами.

Вступ

Однак можна показати декілька найпростіших прикладів розробки, продемонструвати послідовність кроків, які необхідні при проектуванні, які можуть зустрітися в процесі створення пристроїв деяких розповсюджених типів.

Виходячи із цих прикладів, розроблювач може надалі спробувати за аналогією створити щось своє, більш-менш довершене, але обов'язково працездатне.

Розробка клавіатури

Різні клавіатури з великою кількістю клавіш (кнопок) широко використовуються в цифрових системах:

1. у комп'ютерах,
2. контролерах,
3. вимірювальних приладах,
4. у побутовій техніці.

Основне завдання будь-якої клавіатури досить просте: вона повинна при будь-якому натисканні на клавішу видавати код номера цієї клавіші й сигнал флажка натискання клавіші (строб цього коду). Одержавши цей сигнал стробу, зовнішній пристрій читає код натиснутої клавіші і починає необхідні дії.

Головне завдання при проектуванні клавіатури - в мінімізації апаратних витрат та у забезпеченні надійного спрацьовування в будь-якій ситуації. Існує багато схемотехнічних рішень цього завдання - від примітивних до найскладніших.

Розробка клавіатури

Клавіатури можуть бути механічними, квазісенсорними або сенсорними, клавіатури можуть мати жорстку логіку роботи або бути інтелектуальними, навіть допускати перепрограмування.

Ми будемо як приклад розглядати найпростішу механічну клавіатуру із жорсткою логікою роботи.

Опис вимог. Кількість клавіш повноцінної клавіатури комп'ютера перевищує сотню, тому ми будемо проектувати клавіатуру на максимальну кількість клавіш, рівну 128.
Природно, що клавіатура повинна мати захист від дребезгу механічних контактів і повинна коректно обробляти ситуацію одночасного натискання декількох клавіш.
Приймемо, наприклад, що при одночасному натисканні декількох клавіш клавіатура повинна видавати код тільки однієї з них. Приймемо також, що максимально можливий темп натискання клавіш на клавіатурі не повинен перевищувати 20 натискань у секунду (це досить багато).
Таким чином, основні вимоги до проектованого пристрою сформульовані. Почнемо розробку.

Розробка клавіатури

Прийом проектування. Дуже часто зручним і ефективним прийомом є розробка пристрою "з кінця".

Тобто проектування починається виходячи з необхідного результату, з тих сигналів, які пристрій повинен видавати назовні і приймати ззовні. І тільки наприкінці проектування розробляється та частина пристрою, що виконує необхідну функцію.

Такий підхід гарантує, що розроблений пристрій не буде надмірно надлишковим, не буде робити нічого зайвого, а також те, що він коректно буде взаємодіяти з іншими пристроями й системами.

Цей принцип проектування не універсальний, часом дотримуватись його протягом усього процесу розробки важко, але спробувати його застосувати до будь-якого пристрою ніколи не перешкодить.

Розробка клавіатури

У нашому випадку необхідно спочатку визначитися, що повинна видавати назовні клавіатура.

Приймемо, що наша клавіатура повинна видавати **7-розрядний двійковий номер натиснутої клавіші** (тому що $2^7 = 128$) і супроводжувати його **позитивним сигналом флажка** натискання. Сигнал флажка і код клавіші повинні зберігатися доти , доки натиснуто клавішу. За цей час (декілька мілісекунд) зовнішній пристрій повинен встигнути проаналізувати сигнал флажка і прочитати вихідний код клавіатури. Звичайно дана вимога не занадто жорстка.

Також необхідно визначитися, як клавіатура буде поводитися при **одночасному натисканні декількох клавіш**. Найбільш складні, інтелектуальні клавіатури видають послідовно коди всіх натиснутих клавіш, запам'ятовуючи їх у буферній пам'яті. Але ми приймемо, що клавіатура повинна видавати тільки код однієї з одночасно натиснутих клавіш (першої за встановленим порядком). Натискання всіх інших клавіш одночасно з цією просто ігноруються.

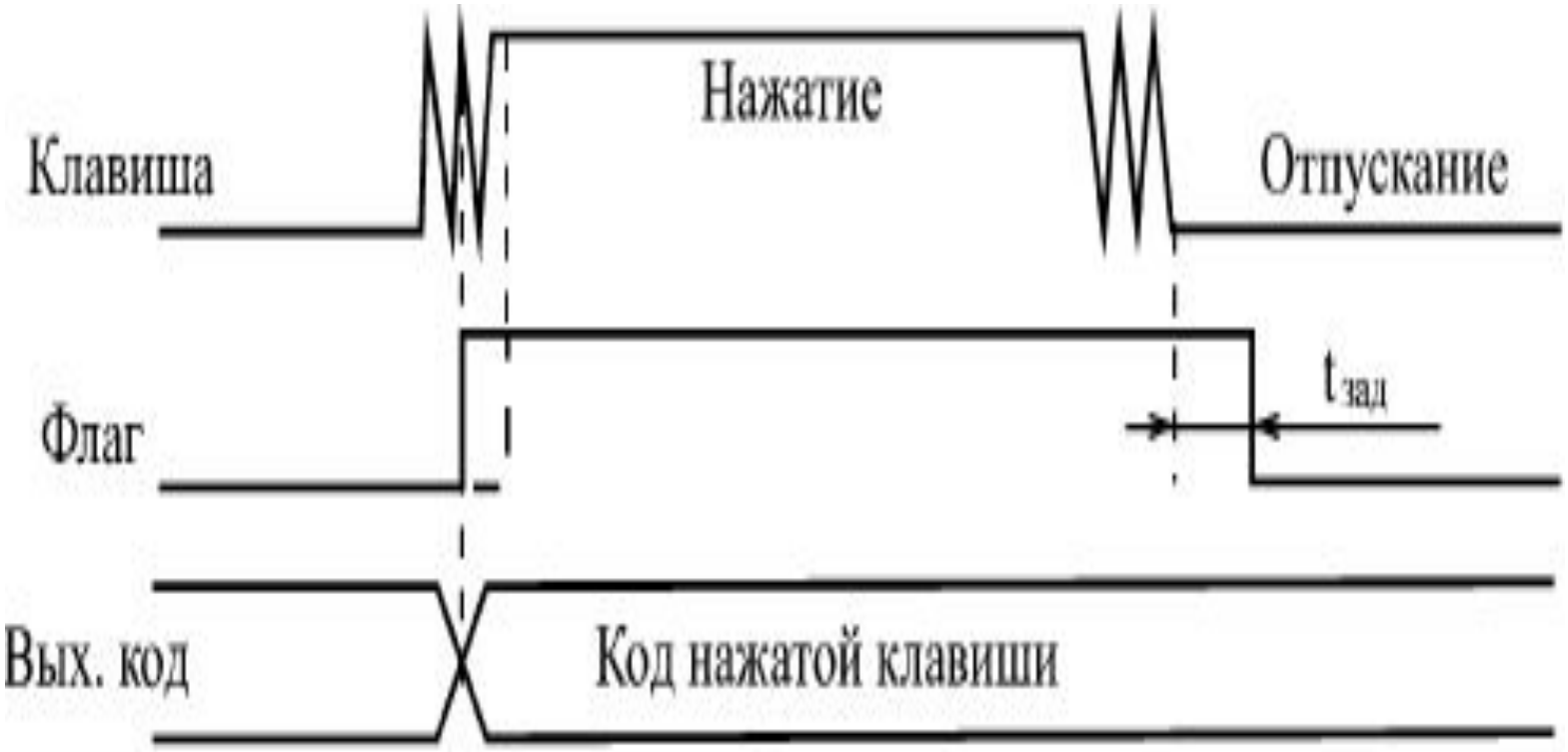
Розробка клавіатури

При проектуванні механічної клавіатури важливо вирішити, як буде оброблятися неминуче присутній дребезг механічних контактів клавіш. Його можна обробляти як усередині клавіатури, так і поза її (тобто перенести цю функцію на зовнішній пристрій).

Обидва ці підходи мають свої переваги. Але наша клавіатура буде обробляти дребезг контактів самостійно. Принцип обробки вибираємо дуже простій: перше зафіксоване замикання контактів клавіші вважається початком натискання, а кінець натискання визначаються тоді, коли контакти будуть розімкнуті протягом заданого інтервалу часу.

Розробка клавіатури

У результаті часова діаграма роботи розроблювальної клавіатури може бути спрощено представлена у вигляді мал. 1. Тут сигнал флажка починається при фіксації одиничного сигналу із клавіші (це може бути як під час дребезга, так і після його закінчення).



Розробка клавіатури

Подальша розробка неможлива без вибору принципу перетворення сигналів від натискання клавіш у код номера натиснутої клавіші.
Найпростішим шляхом побудови подібного перетворювача є використання пріоритетних шифраторів (мал. 2).
Кожна клавіша дає свій логічний сигнал, сигнали від усіх клавіш перетворюються шифратором у код номера клавіші.
Однак такий простий підхід добрий при невеликій кількості клавіш (до 8 або до 16), тому що при великій кількості входів пріоритетний шифратор виходить досить складним.
При малій кількості клавіш дребезг контактів звичайно усувається окремо для кожної клавіші за допомогою RS-тригера (як це показано на малюнку). Це рішення просте, але потребує більших апаратурних витрат.

Розробка клавіатури

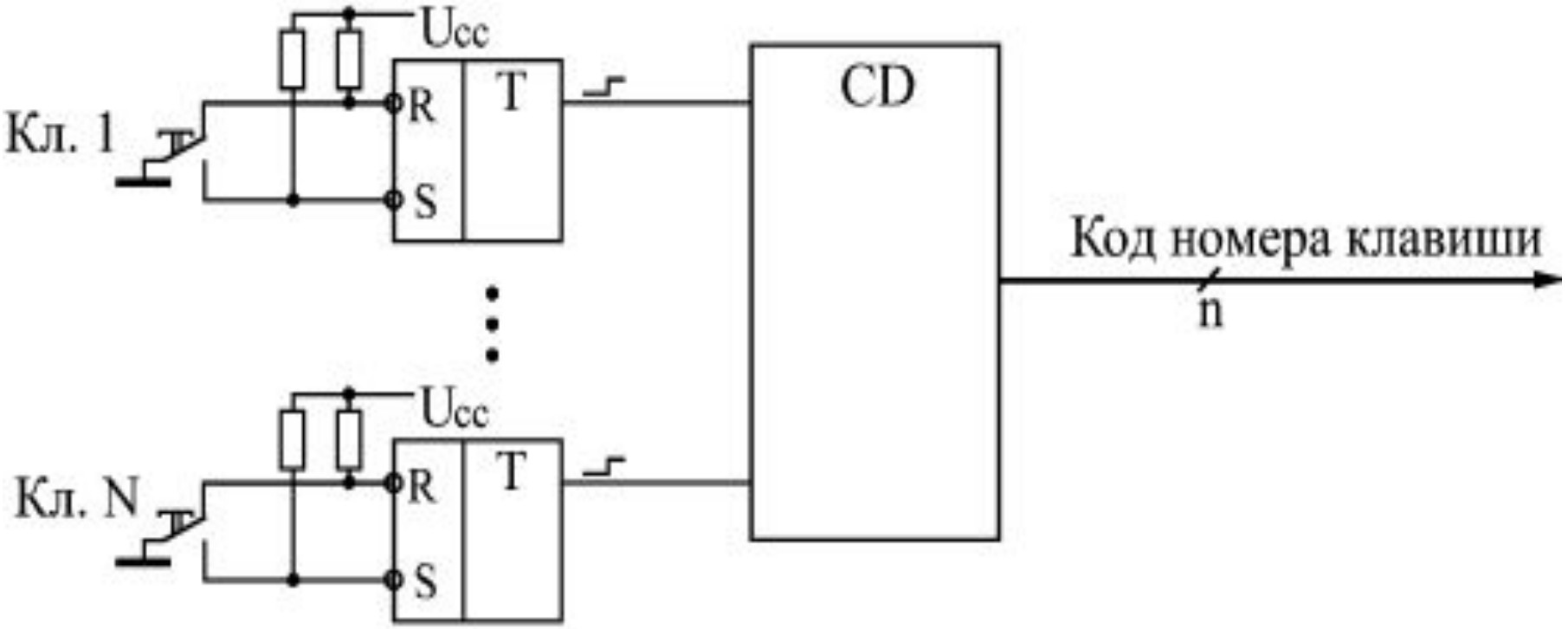


Рис. 2. Найпростіший перетворювач для клавіатури

Розробка клавіатури

Іншим шляхом побудови перетворювача є використання так званої комутаційної матриці, стан якої періодично опитується із частотою тактового генератора.
Комутаційна матриця являє собою дві групи перехресних провідників (рядка і стовпця), у всіх точках перетину яких знаходяться клавіші. У цьому випадку кожна клавіша не формує свого окремого логічного сигналу, а тільки комутирує (з'єднує) одну з рядків матриці з одним з її стовпців.
Найбільш універсальна схема перетворювача, легко нарощувана й досить проста, наведена на мал. 3.

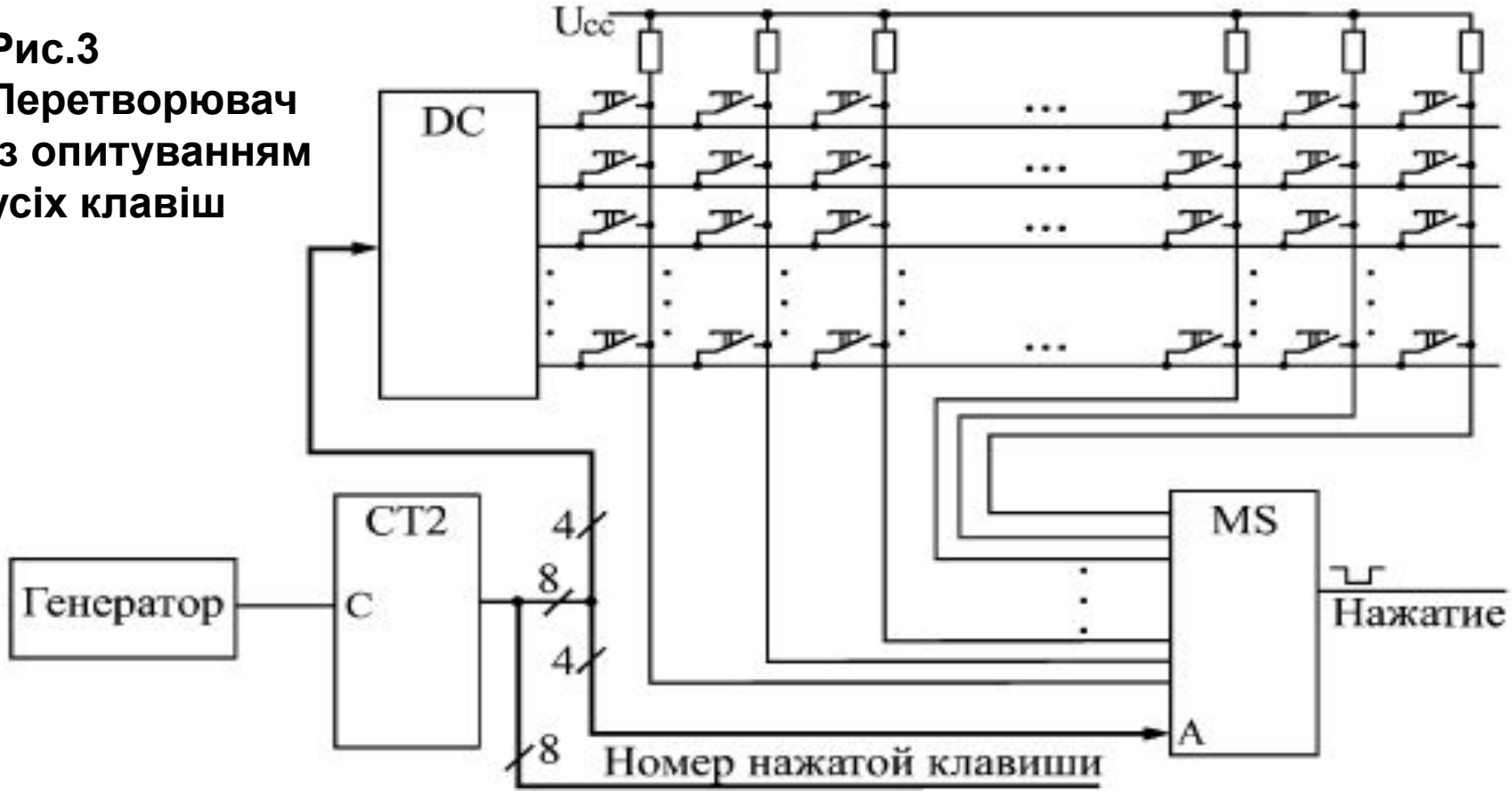
Опис роботи:

1. Для опитування комутаційної матриці застосовується лічильник, що тактується від генератора.
2. Старші розряди лічильника використовуються для вибору одного з рядків матриці за допомогою дешифратора (на обраний рядок надходить сигнал логічного нуля, на невибрану - сигнал логічної одиниці).
3. Молодші розряди лічильника використовуються для опитування стовпців матриці за допомогою мультиплексора. Сигнал з опитуваного стовпця подається на вихід мультиплексора.

Розробка клавіатури

4. Ознакою натискання клавіші є нульовий сигнал на виході мультиплексора.
5. У цей момент на виходах лічильника присутній код номера натиснутої клавіші. Така схема легко дозволяє будувати клавіатури на велику кількість клавіш (до 256, як на малюнку, і навіть більше), однак вона вимагає досить великого часу для повного опитування клавіатури (тому що кількість тактів опитування дорівнює повній кількості клавіш).

Рис.3 Перетворювач із опитуванням усіх клавіш

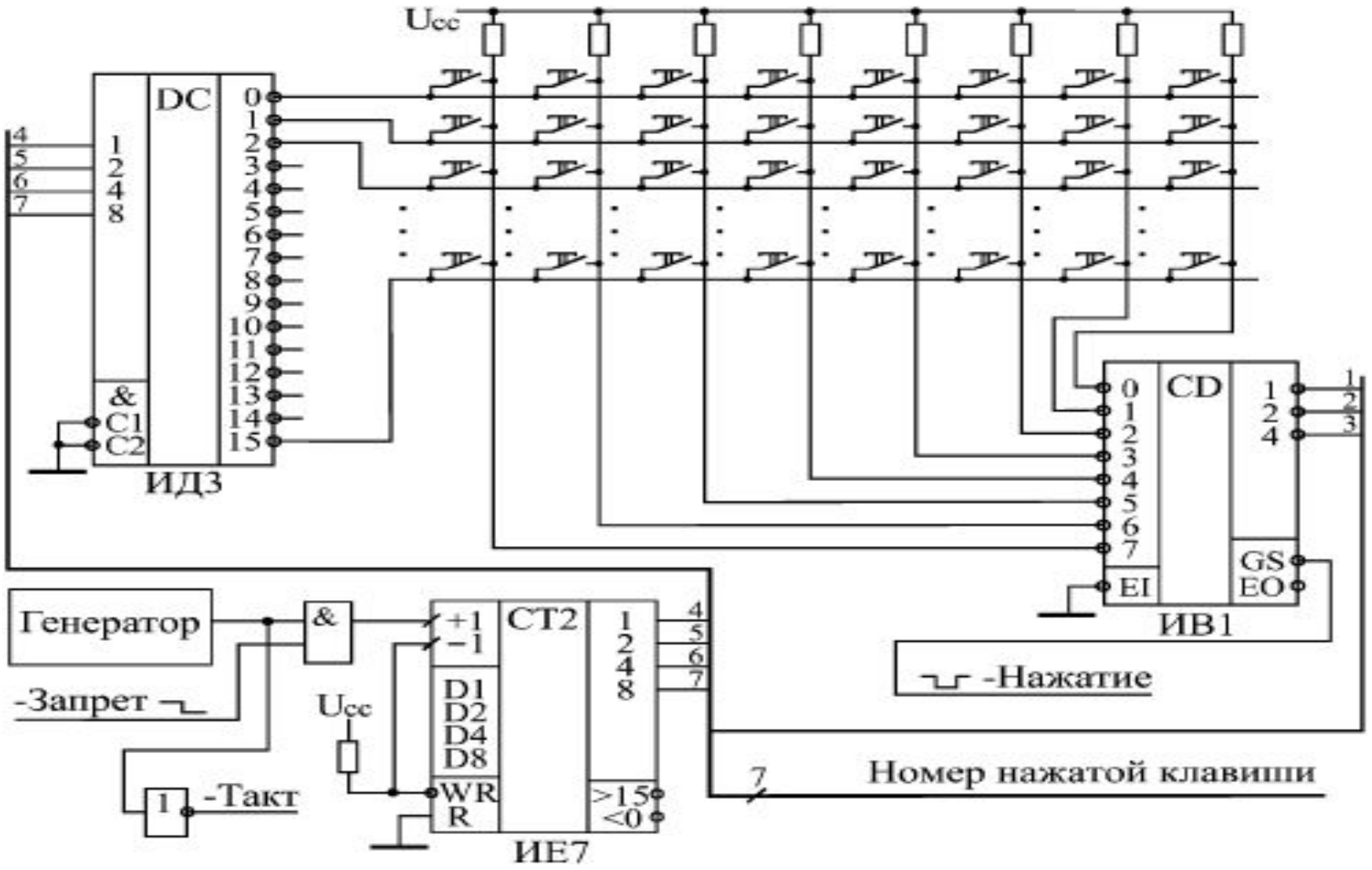


Розробка клавіатури

Суміщення двох розглянутих підходів (мал.2 і 3) дозволяє створювати досить великі клавіатури з малими апаратурними витратами й малим часом опитування.

При такому комбінованому методі (мал. 4) також використовується комутаційна матриця із клавішами на всіх перетинаннях рядків і стовпців, але опитуються не всі клавіші по черзі, а тільки рядки (або стовпці) матриці. Для опитування, як і в попередньому випадку , застосовуються генератор, лічильник і дешифратор. Положення ж натиснутої клавіші в рядку (або в стовпці) визначається за допомогою шифратора. Код натиснутої клавіші утворюється з вихідного коду лічильника (старші розряди) і коду з виходу шифратора (молодші розряди).

Розробка клавіатури (Рис.4 Перетворювач із опитуванням рядків клавіш)



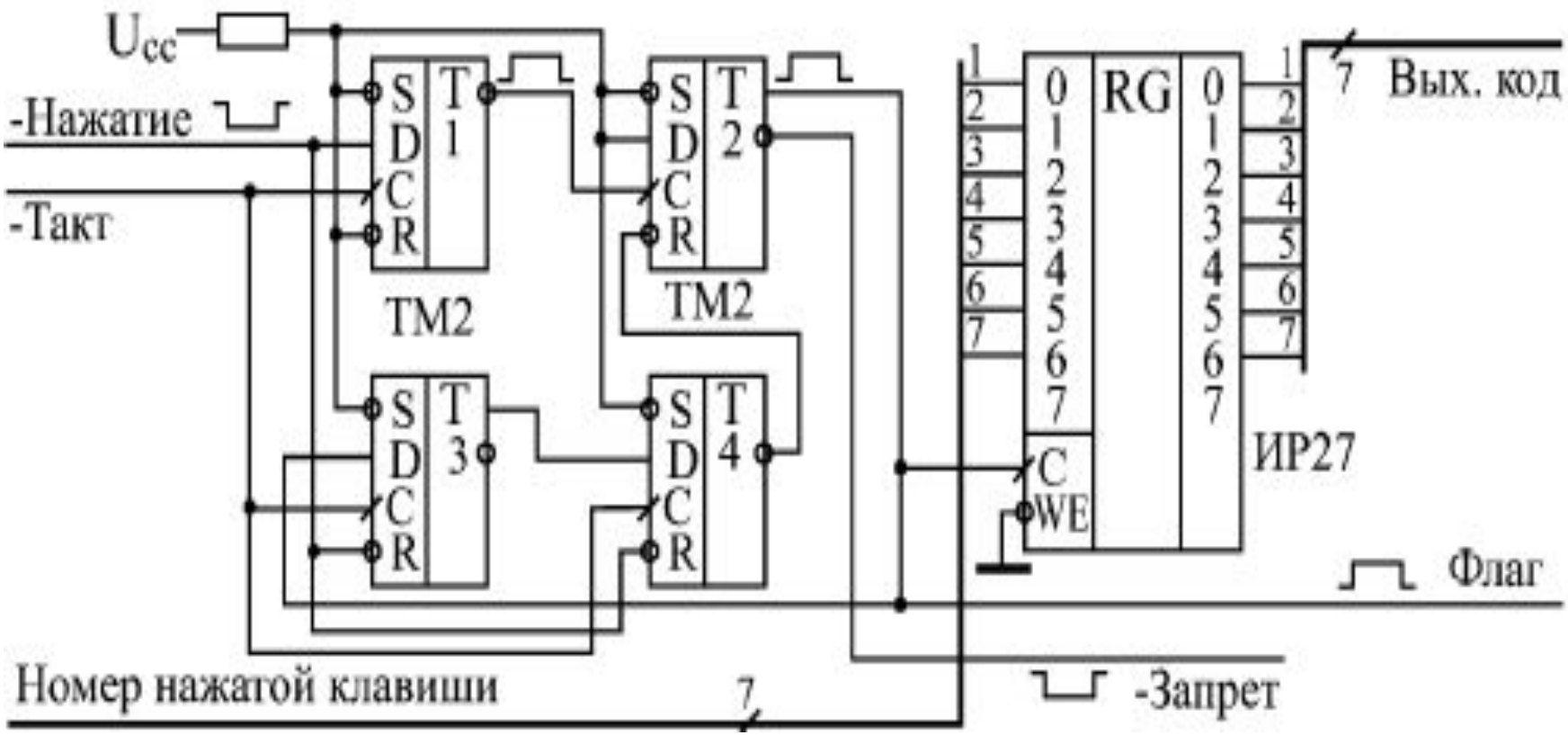
Оцінка частоти

Оцінимо, якою повинна бути частота тактового генератора.

Ми прийняли, що максимальна швидкість натискання рівна 20 раз за секунду. Виходить, що за 1/20 секунди треба встигнути опитати всю клавіатуру, тобто всі 16 рядків. Таким чином, мінімально припустима тактова частота становить 16•20 = 320 Гц.
Але треба закласти й запас на обробку дребезга контактів.
Тому приймемо тактову частоту опитування рівною 400 Гц. Вона може бути й більшою, але надмірно збільшувати її (наприклад, вище 1 кГц) не варто, тому що при швидкому перемиканні мікросхем збільшується споживаний схемою струм.

Зрозуміло, що генератор повинен бути не кварцевим, тому що кварцеві резонатори на низькі частоти не випускаються, а дільник частоти різко ускладнить схему. До того ж точна витримка частоти генератора в цьому випадку зовсім не потрібна.

Схема вироблення вихідних сигналів клавіатури



Розробка обчислювача контрольної суми

Різні контрольні суми широко застосовуються в цифрових пристроях і системах для контролю правильності зберігання або передачі масивів інформації.

Суть цього методу контролю проста: до збереженого або переданого інформаційного масиву приєднується невеликий контрольний код (звичайно від 1 розряду до 32 розрядів), у якому у згорнутому вигляді міститься інформація про весь масив.

При читанні або одержанні цього масиву ще раз обчислюється той же самий контрольний код по тому ж самому алгоритму . Якщо цей заново обчислений код дорівнює тому коду, який був приєднаний до масиву, то вважається , що масив збережений або переданий без помилок.

Логіка тут наступна: контрольний код (він же контрольна сума) є набагато меншим від контрольованого масиву, тому ймовірність спотворення контрольної суми набагато менше, чим імовірність спотворення масиву. Якщо ж спотворяться як масив, так і контрольна сума, то ймовірність того, що ці спотворення не будуть помічені при повторному підрахунку контрольної суми, украй мала.

Контрольні суми

Контрольні суми застосовуються:

1. при зберіганні даних у пам'яті (оперативній і постійній),
2.при зберіганні даних на магнітних носіях (дисках, стрічках),
3.у локальних і глобальних мережах передачі інформації.

У випадку захисту контрольною сумою збереженої інформації можна визначити , що даний масив (файл, сектор на диску) зіпсований і його не можна використовувати.

У випадку захисту контрольною сумою переданої по мережі інформації приймач може зажадати від передавача повторної передачі спотвореного масиву.

Контрольні суми

Існує безліч способів обчислення контрольної суми, що різняться ступенем складності обчислення й надійністю виявлення помилок. Але найбільше поширення одержав так званий "**циклічний метод контролю надлишковості"** або CRC (Cyclic Redundancy Check), при якому застосовується циклічна контрольна сума.

Обчислюється циклічна контрольна сума в такий спосіб.
Увесь масив інформації розглядається як одне N-розрядне двійкове число, де N - кількість біт у всіх байтах масиву.

Для обчислення контрольної суми це N-розрядне число ділиться на деяке постійне число (поліном), обране спеціальним чином (але ділиться не просто, а по модулю 2). Частка від цього ділення відкидається, а залишок саме й використовується в якості контрольної суми.

Даний метод виявляє одиночні помилки в масиві з імовірністю 100%, а будь-яку іншу кількість помилок з імовірністю, приблизно рівною $1-2^{-n}$, де n - кількість розрядів контрольної суми (це вірно тільки за умови, що N набагато більше n, що, втім, майже завжди виконується). Наприклад, при n = 8 дана ймовірність складе 0,996, для n = 16 вона буде рівна 0,999985, а для n = 32 вона буде 0,9999999997672. Інакше кажучи, майже всі помилки будуть виявлятися.

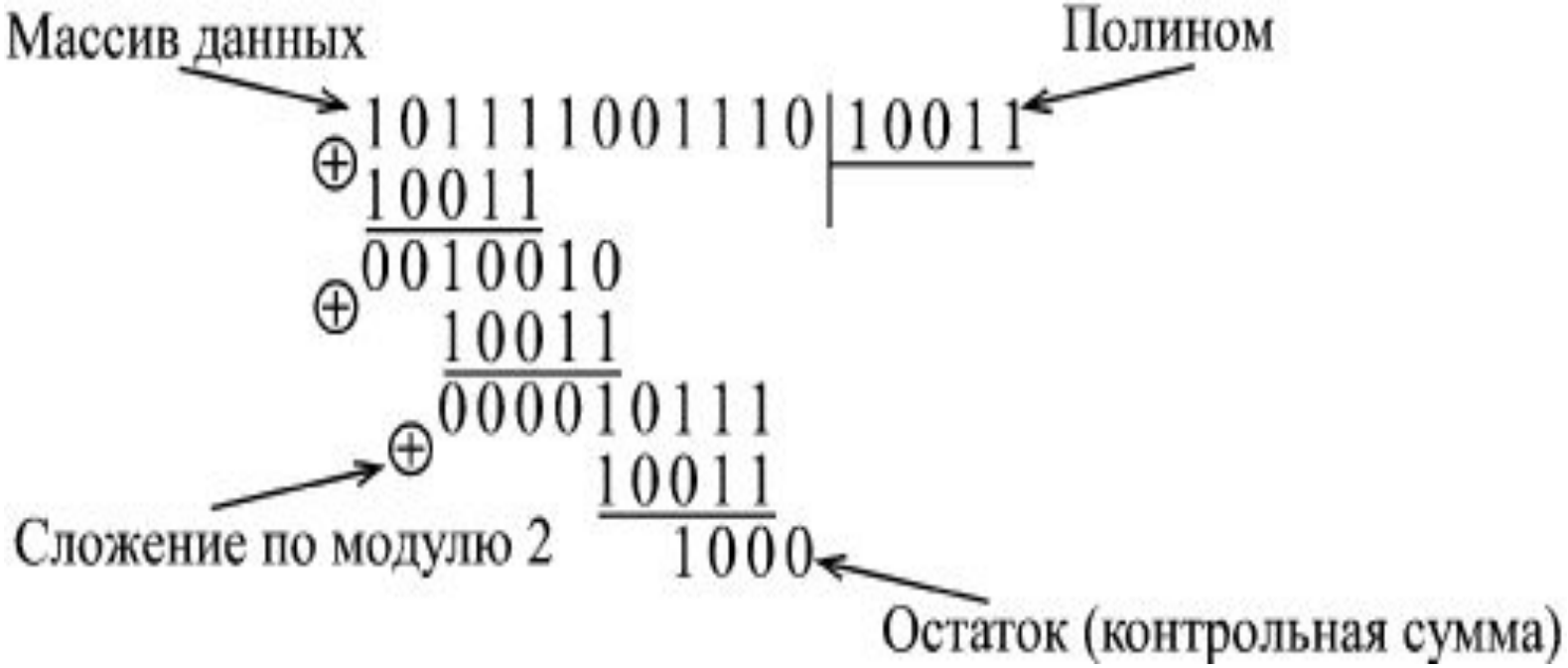
Що таке ділення по модулю 2 ?

Нехай масив (послідовність біт) має такий вигляд: 101111001110 (для простоти беремо невелику розрядність).

Число, на яке ділимо (називане звичайно утворюючим поліномом) візьмемо 10011. **Як воно вибирається?**
Воно повинне ділитися по модулю 2 без залишку тільки на одиницю й саме на себе (тобто це повинне бути просте число по значенню ділення по модулю 2).
Розрядність полінома береться на одиницю більшою, ніж необхідна розрядність контрольної суми (залишку від ділення).
Так, щоб одержати 8- розрядний залишок (8-розрядну контрольну суму), треба брати 9-розрядний поліном. У нашому випадку поліном 5-розрядний, отже, залишок буде 4- розрядний. Для одержання 8-розрядного залишку можна використовувати , наприклад, поліном 1 0001 1101 або 11D в 16-ричному коді.

Обчислення циклічної контрольної суми

Ділення по модулю 2 проводиться точно так само, як і звичне для нас ділення "у стовпчик" (мал. 6), але замість віднімання в цьому випадку використовується поразрядне додавання по модулю 2, тобто кожен результуючий біт являє собою функцію, Исключающее ИЛИ від відповідних бітів доданків. Частка від ділення нас не цікавить, а залишок , рівний у нашому прикладі 1000, і буде циклічною контрольною сумою.



Як практично реалізувати

Як практично реалізувати обчислення цього залишку (контрольної суми)?
Можна зробити це по приведеному тут принципу ділення в стовпчик (апаратно або програмно).
Але в кожному разі це досить громіздко й повільно. Прискорити процес обчислення можна, скориставшись табличним методом. Для цього складається таблиця чисел розміром 2^nxn, де n - розрядність контрольної суми. Принцип обчислення чисел у таблиці дуже простий (табл. 1).

Числа являють собою залишок ділення по модулю 2 числа з n кінцевими нулями (у нашому прикладі n = 8) і з n початковими розрядами, рівними номеру числа (його адресі) у таблиці. Ділення проводиться на обраний поліном (у нашому випадку - 9-розрядний). Таблиця обчислюється один раз і зберігається на диску або в ПЗУ.

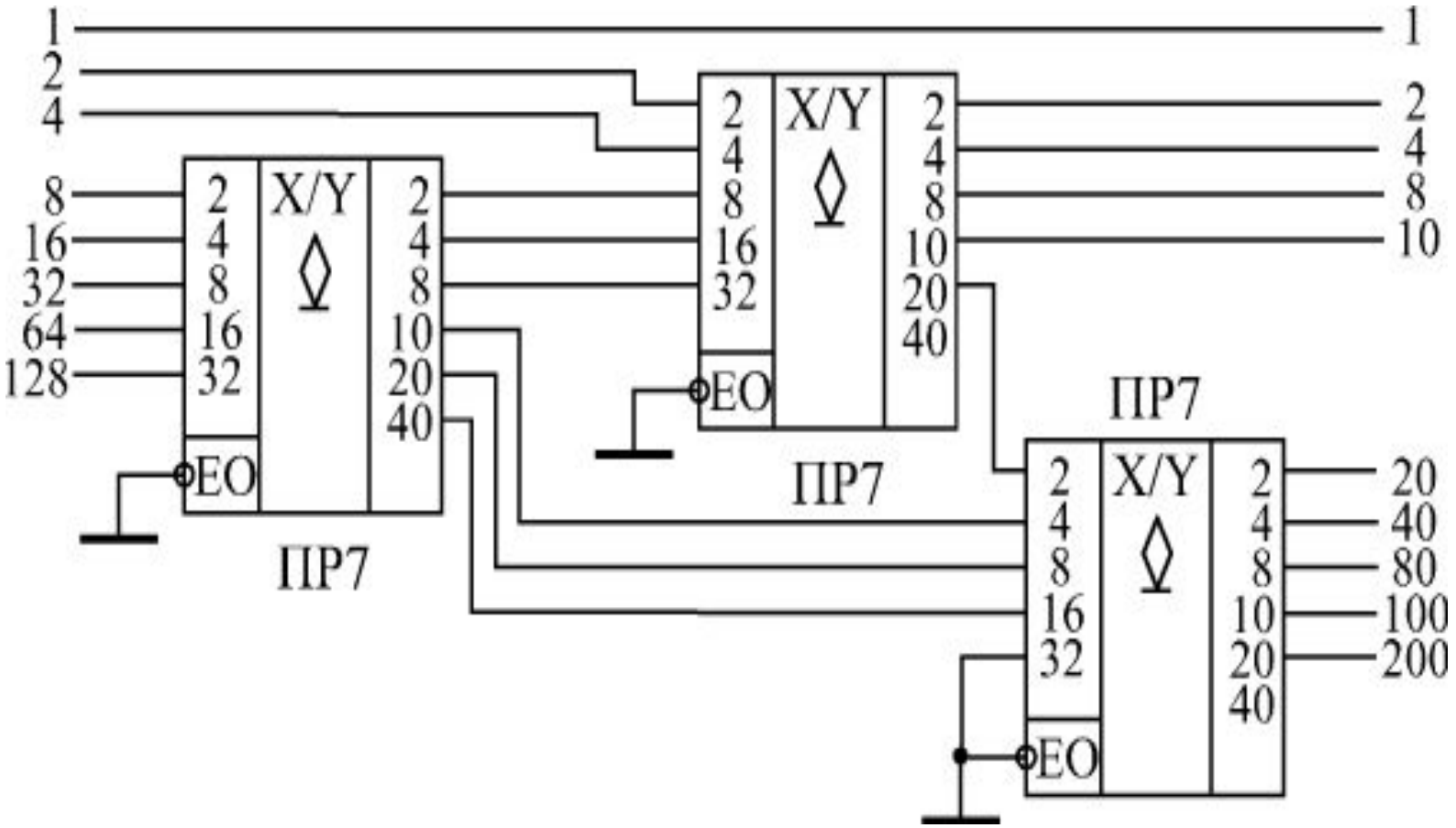
Табличний метод

Таблица 1. Табличный метод вычисления циклической контрольной суммы

Адрес в таблице	Данные в таблице (числа)
0	0
1	Остаток от деления числа 1 0000 0000 на полином
2	Остаток от деления числа 10 0000 0000 на полином
3	Остаток от деления числа 11 0000 0000 на полином
4	Остаток от деления числа 100 0000 0000 на полином
5	Остаток от деления числа 101 0000 0000 на полином
255	Остаток от деления числа 1111 1111 0000 0000 на полином

BCD

Перетворювач двійкового коду від 0 до 255 у двійково-десятковий код

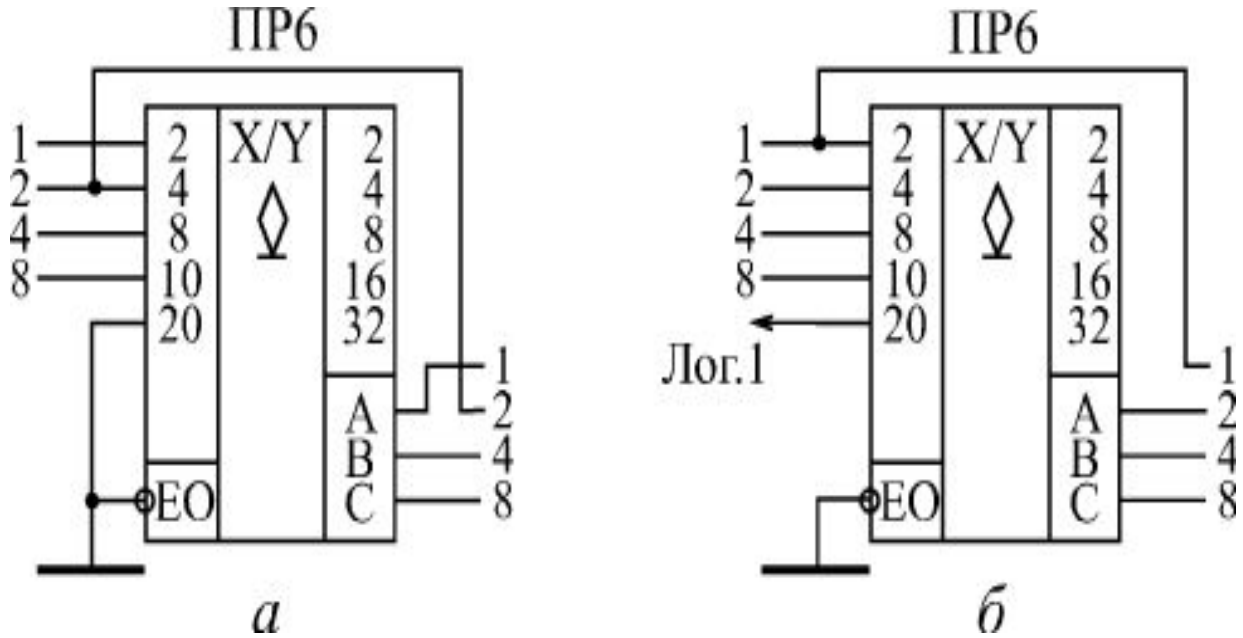


Перетворювачі

Наявність додаткових виходів A, B, C у мікросхеми ПР6 дозволяє перетворювати двійково-десятковий код від 0 до 9 у код доповнення до 9 або до 10 (див. рис).

Тобто сума вхідного й вихідного кодів у цьому випадку рівна, відповідно, 9 або 10.

Наприклад, при вхідному коді 6 на виході схеми а буде код 3, а на виході схеми б - код 4. У схемі б при вхідному коді 0 на виході також формується код 0.



Перетворювачі вхідного коду на додаток до 9 (а) і на додаток до 10 (б)

Одновібратори й генератори

Одновібратори й генератори взагалі -то не можна відносити до комбінаційних мікросхем. Вони займають проміжне положення між комбінаційними мікросхемами й мікросхемами із внутрішньою пам'яттю.

Їхні вихідні сигнали не визначаються однозначно вхідними сигналами, як у комбінаційних мікросхем. Але в той же час вони й не зберігають інформацію тривалий час .

Одновібратори (" мультивібратори, що чекають, “ ждущие") являють собою мікросхеми, які у відповідь на вхідний сигнал (логічний рівень або фронт) формують вихідний імпульс заданої тривалості. Тривалість визначається зовнішніми елементами, які задають затримку резисторами й конденсаторами.

Тобто можна вважати , що в одновібраторів є внутрішня пам'ять, але ця пам'ять зберігає інформацію про вхідний сигнал строго заданий час, а потім ця інформація зникає. На схемах одновібратори позначаються буквами G1.

Усі стандартні мікросхеми одновібраторів бувають двох основних типів:

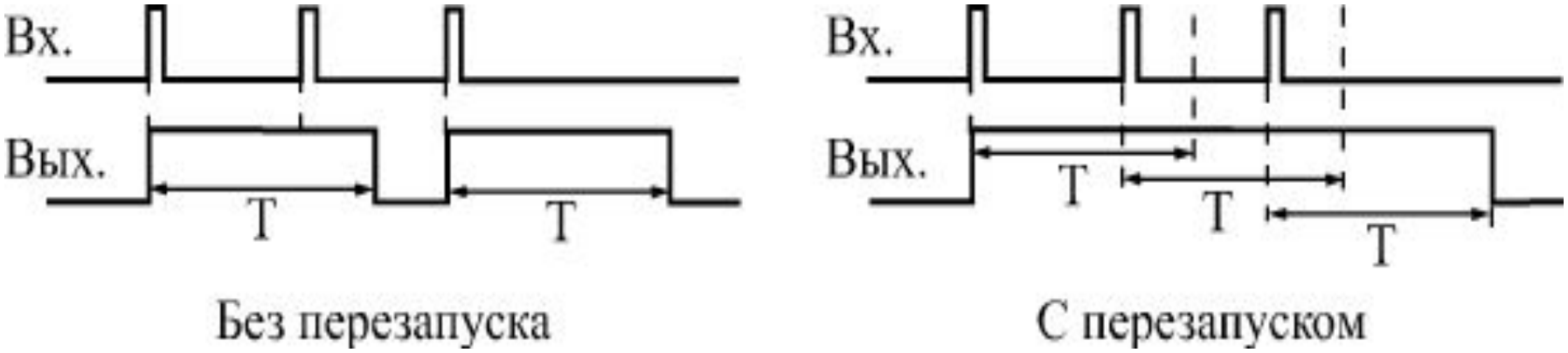
- Одновібратори без перезапуску (одиночний одновібратор)
- Одновібратори з перезапуском).

Одновібратори й генератори

Різниця між цими двома типами ілюструється на рис.

Одновібратор без перезапуску не реагує на вхідний сигнал до закінчення свого вихідного імпульсу.

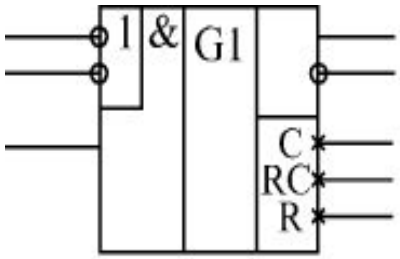
Одновібратор з перезапуском починає відлік нового часу затримки T з кожним новим вхідним сигналом незалежно від того, чи закінчився попередній час затримки. У випадку , коли період проходження вхідних сигналів менший від часу затримки T, вихідний імпульс одновібратора з перезапуском не переривається. Якщо період проходження вхідних імпульсів, що запускають, більший від затримки одновібратора T, то обидва типи одновібраторів працюють однаково.



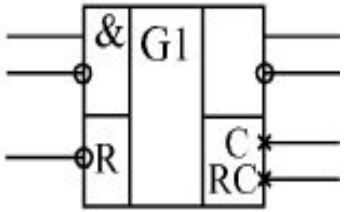
Принцип роботи одновібраторів без перезапуску й з перезапуском

Одновібратори й генератори

На рис наведені позначення мікросхем одновібраторів . Мікросхеми АГ3 і АГ4 відрізняються одна від іншої тільки тим, що АГ3 працює з перезапуском, а АГ4 - без перезапуску.

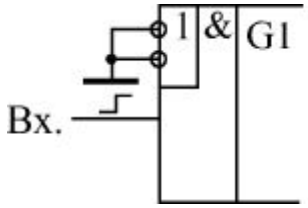


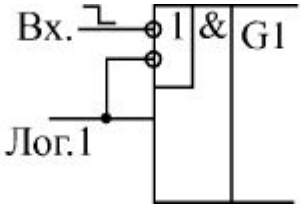
АГ1

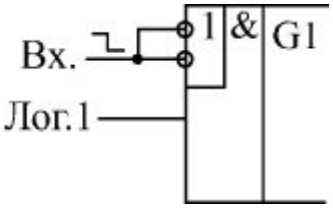


АГ3, АГ4

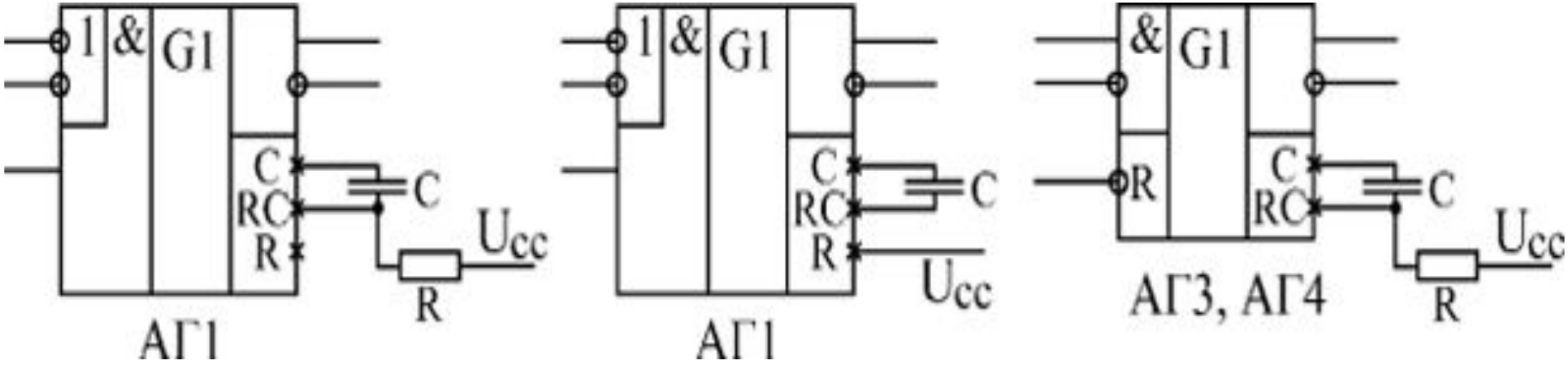
Варіанти запуску одновібратора АГ1







Одновібратори й генератори



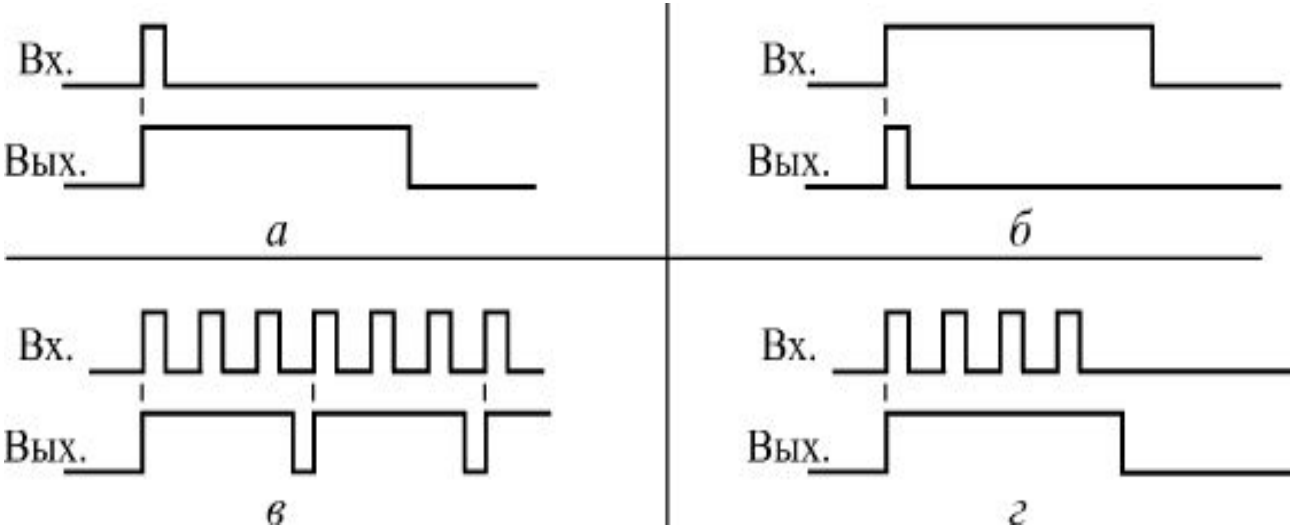
Стандартні схеми включення одновібраторів

Для одновібраторів АГ3 і АГ4 тривалість імпульсу можна оцінити по формулі: T = 0,32C(R + 0,7), де опір резистора виміряється в кілоомах. Опір резистора може знаходитися в межах від 5,1 Ком до 51 Ком, ємність конденсатора – будь-яка. Перезапуск одновібратора можливий тільки в тому випадку, коли інтервал між вхідними, що запускають імпульсами більше 0,224C (якщо ємність виміряється в нанофарадах, то часовий інтервал - у мікросекундах).

Застосування одновібраторів

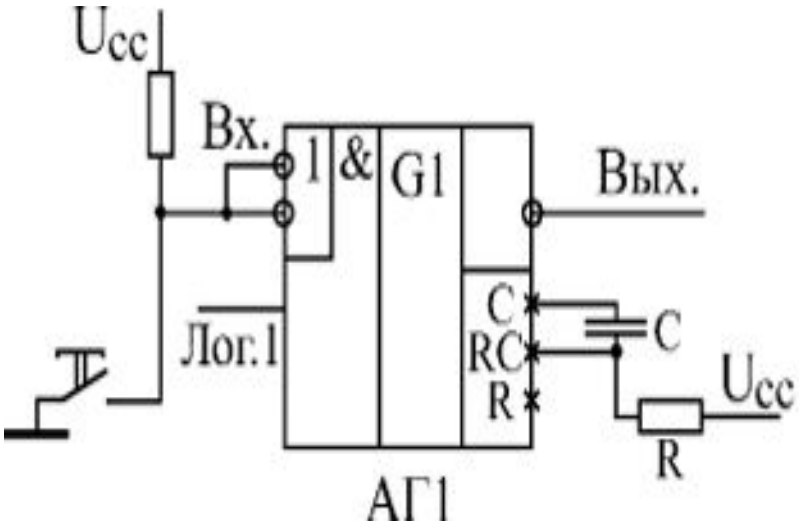
Найпоширеніші застосування одновібраторів наступні (рис):
1.збільшення тривалості вхідного імпульсу;
2.зменшення тривалості вхідного імпульсу;
3.ділення частоти вхідного сигналу в задане число разів;
4.формування сигналу послідовності, що огинає, вхідні імпульси.

Для збільшення або зменшення тривалості вхідного сигналу (а і б) треба усього лишень вибрати опір резистора і ємність конденсатора, виходячи з необхідної тривалості вихідного сигналу. У цьому випадку можна використовувати одновібратор будь-якого типу: як з перезапуском, так і без перезапуску.



Застосування одновібраторів

Ще одне важливе застосування одновібратора полягає в придушенні дребезга контактів кнопки. Одновібратор з великим часом затримки (порядку декількох десятих часток секунди) надійно пригнічує паразитні імпульси, що виникають через дребезг контактів, і формує ідеальні імпульси при будь-якому натисканні кнопки

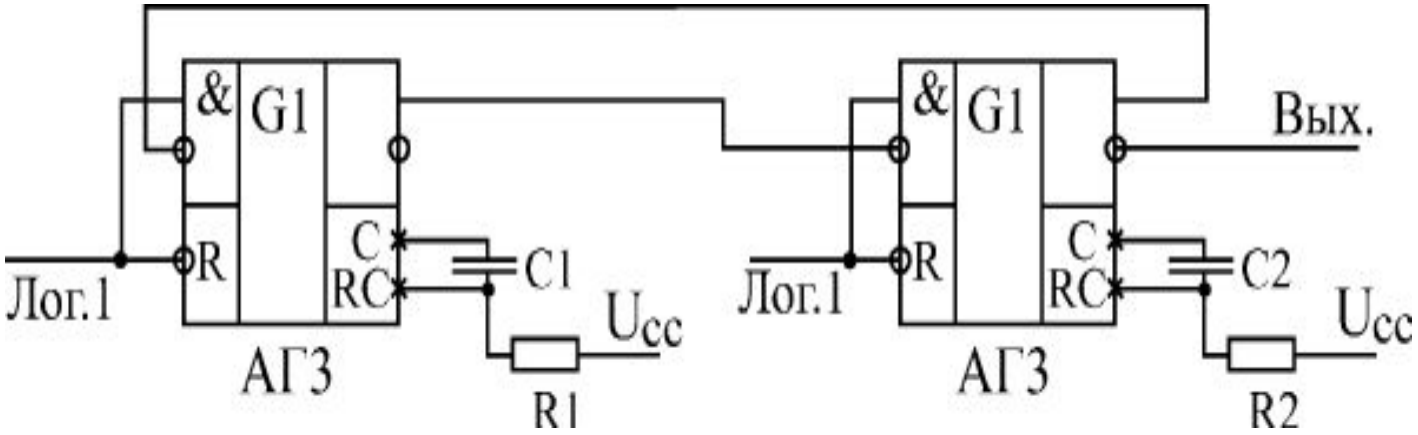


Можна використовувати як одновібратор з перезапуском, так і одновібратор без перезапуску (на малюнку). Можна також підібрати час затримки так, що одновібратор буде видавати один імпульс при натисканні кнопки, а інший імпульс - при відпусканню кнопки. Іноді це буває зручніше.

Застосування одновібраторів

Одновібратори можна також застосовувати для побудови **генераторів (мультивібраторів) прямокутних імпульсів** з різними значеннями тривалості імпульсів і паузи між ними.

При цьому два одновібратори замикаються в кільце так, що кожен з них запускає інший після закінчення свого вихідного імпульсу (рис.). Один одновібратор формує тривалість імпульсу, а інший визначає паузу між імпульсами. Змінюючи номінали резисторів і конденсаторів, можна одержати потрібні співвідношення імпульсу й паузи.



Генератор імпульсів на двох одновібраторах

Застосування одновібраторів

Таким чином, одновібратори досить легко дозволяють вирішувати самі різноманітні задачі. Однак, застосовуючи одновібратори, треба завжди пам'ятати, що тривалість їх вихідних імпульсів **не можна задати дуже точно** - адже одновібратор має аналогові ланки. На тривалість вихідного імпульсу одновібратора впливають номінали резисторів і конденсаторів, температура навколишнього середовища, старіння елементів, перешкоди в ланках живлення й інші фактори. Тому застосування одновібраторів потрібно по можливості обмежувати тільки тими випадками, коли час затримки можна задавати з не занадто високою точністю (погрішність не менш 20-30%).

Будь-яку функцію одновібратора може виконати синхронні пристрої (на основі кварцового генератора, тригерів, регістрів, лічильників), причому виконати набагато точніше й надійніше. І їм не потрібно ніяких додаткових елементів, що задають час (резисторів і конденсаторів).

Генератори з автопідстройкою частоти

Окрім одновібраторів, розробники використовують також спеціалізовані генератори ("мультивібратори", англ. "multivibrator"). Позначаються вони на схемах буквою G.

Наприклад, мікросхема ГГ1 являє собою два генератори в одному корпусі . Мікросхеми генераторів використовують досить рідко, частіше застосовують генератори на інверторах або на тригерах Шмітта.

Однак у деяких випадках генератори ГГ1 не можуть бути замінені нічим. Справа в тому, що вони допускають зміну частоти вихідних імпульсів за допомогою рівнів двох вхідних керуючих напруг. Тому вони називаються також "генератори, що керовані напругою“ . Ефект зміни частоти можна використовувати , наприклад, у системах автопідстроювання частоти (АПЧ) або в пристроях із частотною модуляцією (ЧМ).

Схема включення генератора ГГ1