ЦИФРОВА СХЕМОТЕХНІКА

ЗАНЯТТЯ 7. РЕГІСТРИ. ЇХ КЛАСИФІКАЦІЯ ТА ВИКОРИСТАННЯ

ЗНАЧЕННЯ ТА РОЛЬ РЕГІСТРІВ У ЦИФРОВІЙ СХЕМОТЕХНІЦІ

Регістр — це одна з основних складових будь-якої цифрової системи, яка виконує функцію **збереження та обробки бінарних даних**. Завдяки своїй простій, але гнучкій конструкції, регістри використовуються в різноманітних пристроях — від мікропроцесорів до систем керування.

Введення (запис) та зчитування (виведення) даних у регістрі залежать від способу підключення і характеристик цих з'єднань. Відповідно, існує чотири основні методи прийому і передачі інформації:

Послідовний вхід і послідовний вихід (SISO) — дані записуються і зчитуються по одному біту за такт.

Послідовний вхід і паралельний вихід (SIPO) — інформація вводиться послідовно, а зчитується одразу всіма бітами одночасно.

Паралельний вхід і послідовний вихід (PISO) — всі біти даних записуються одночасно, а зчитуються по одному за кожен тактовий імпульс.

Паралельний вхід і паралельний вихід (PIPO) — як введення, так і зчитування відбуваються одночасно для всіх бітів.

ОСНОВНІ ФУНКЦІЇ РЕГІСТРІВ

Регістри виконують різноманітні завдання, пов'язані зі збереженням, перетворенням і передачею даних. Основні функції регістрів включають:

1. Збереження даних — регістр тимчасово зберігає бінарну інформацію, наприклад, команди процесора або результати обчислень.

2. Передача даних — регістр забезпечує передачу інформації між різними блоками схеми у послідовному або паралельному режимі.

3. Зсув даних — регістр зсуву дозволяє зміщувати бітову послідовність вліво або вправо, що важливо для арифметичних операцій і кодування.

4. Перетворення форматів даних — універсальні регістри можуть виконувати серійно-паралельне або паралельно-серійне перетворення.

5. Буферизація сигналів — регістри використовуються для усунення часових затримок між різними частинами схеми.

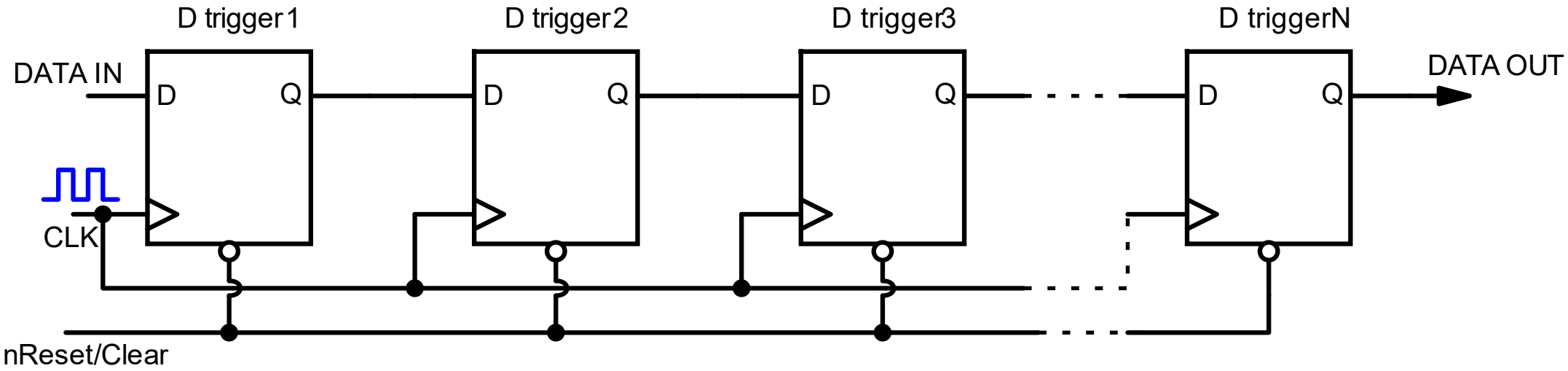
6. Синхронізація даних — регістри дозволяють узгоджувати швидкості роботи різних компонентів системи, особливо в схемах із тактовими сигналами.

7. Формування імпульсів — регістри можуть брати участь у створенні часових затримок або зміни тривалості сигналів.

SISO

SISO (Serial-In, Serial-Out). Вхідні дані подаються **послідовно**, один біт за такт. Вихідні дані зчитуються **послідовно**, після проходження через усі тригери. Цей регістр ще називають зсувним (Shift register).

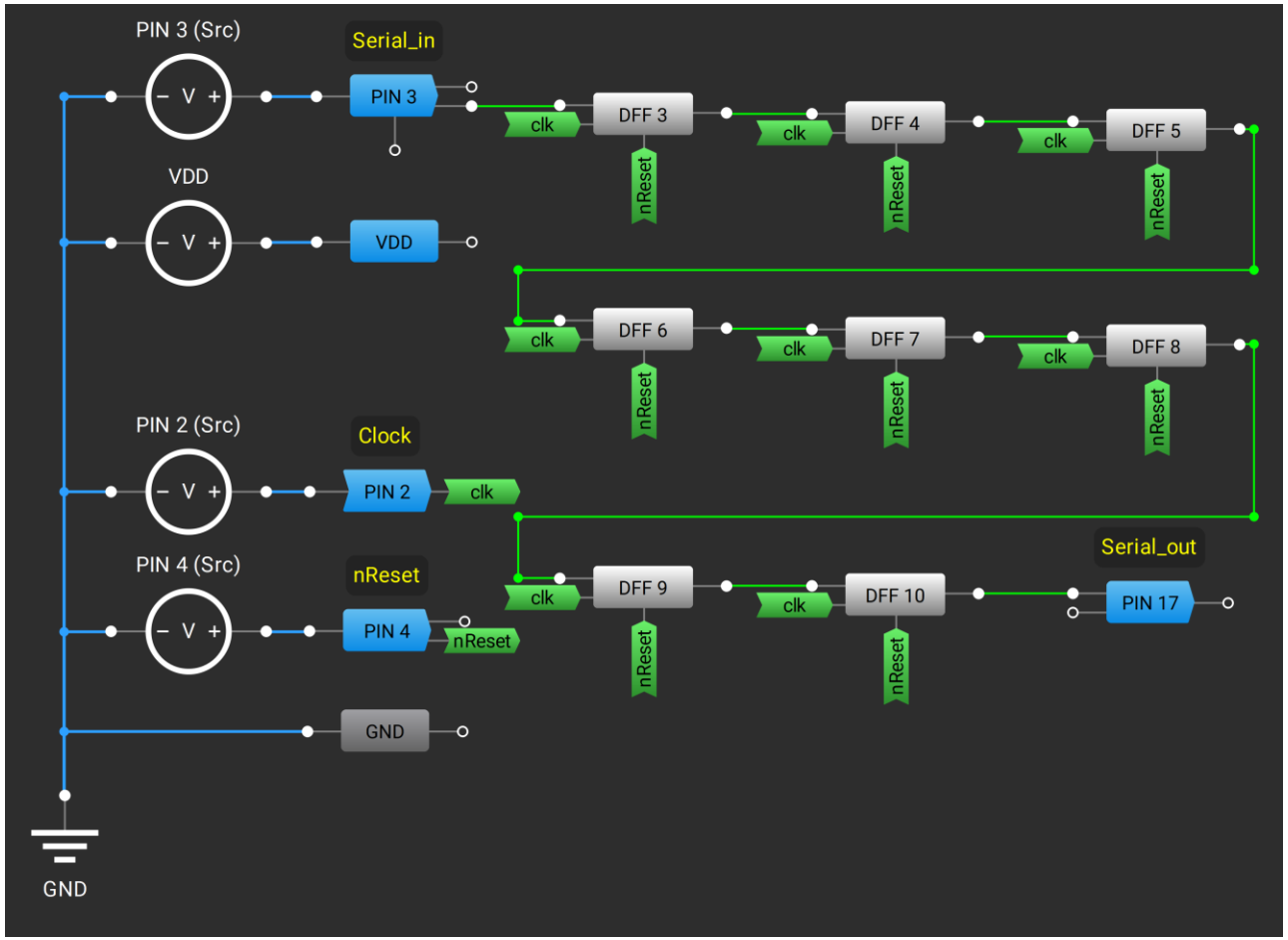
Зсувний регістр складається з **ланцюжка тригерів** (зазвичай D-тригерів), з’єднаних так, що вихід одного тригера подається на вхід наступного. Кожен тактовий сигнал **зсуває біти на одну позицію**, що дозволяє реалізовувати різні операції з даними.

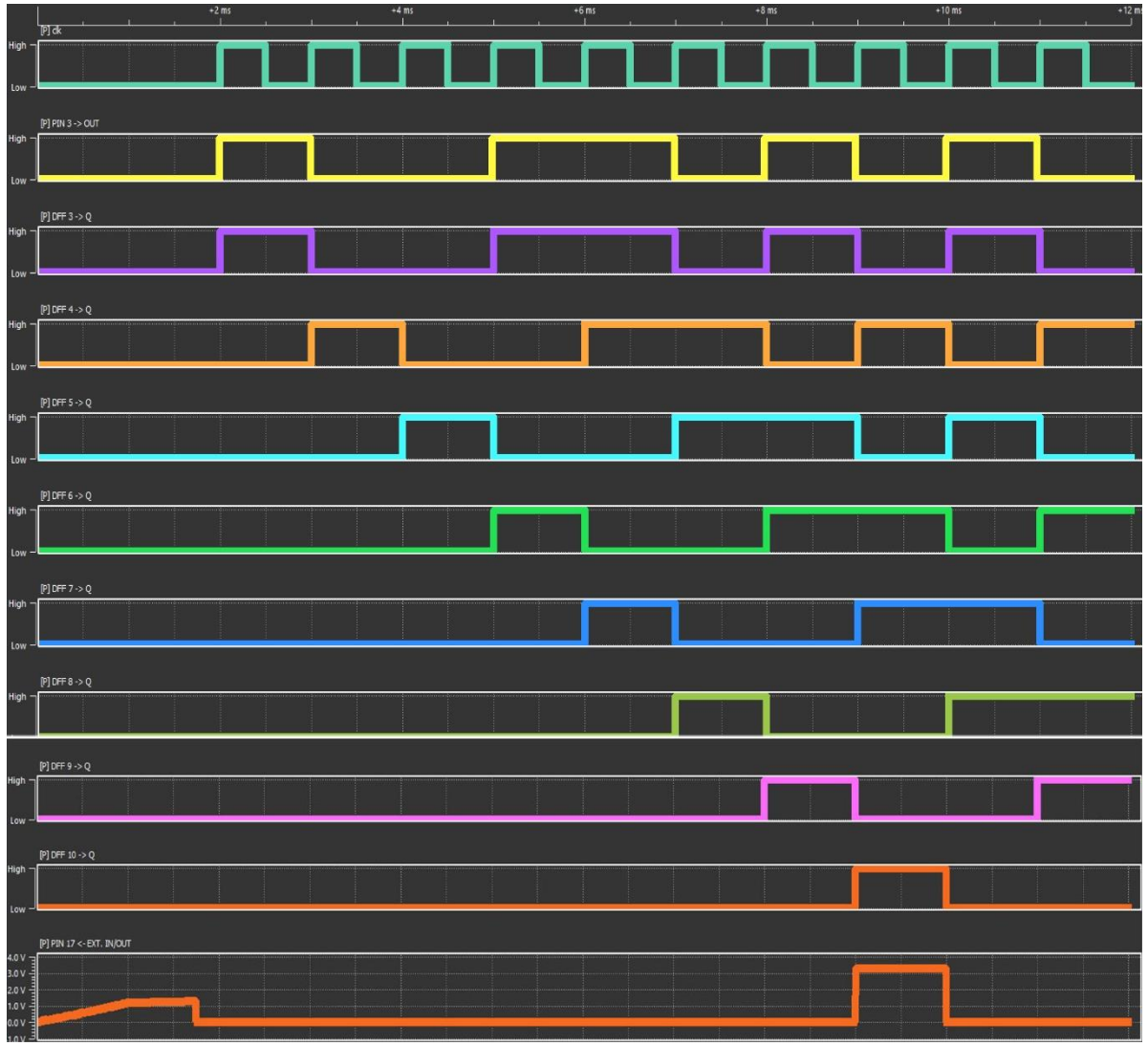


Serial input serial output(SISO)

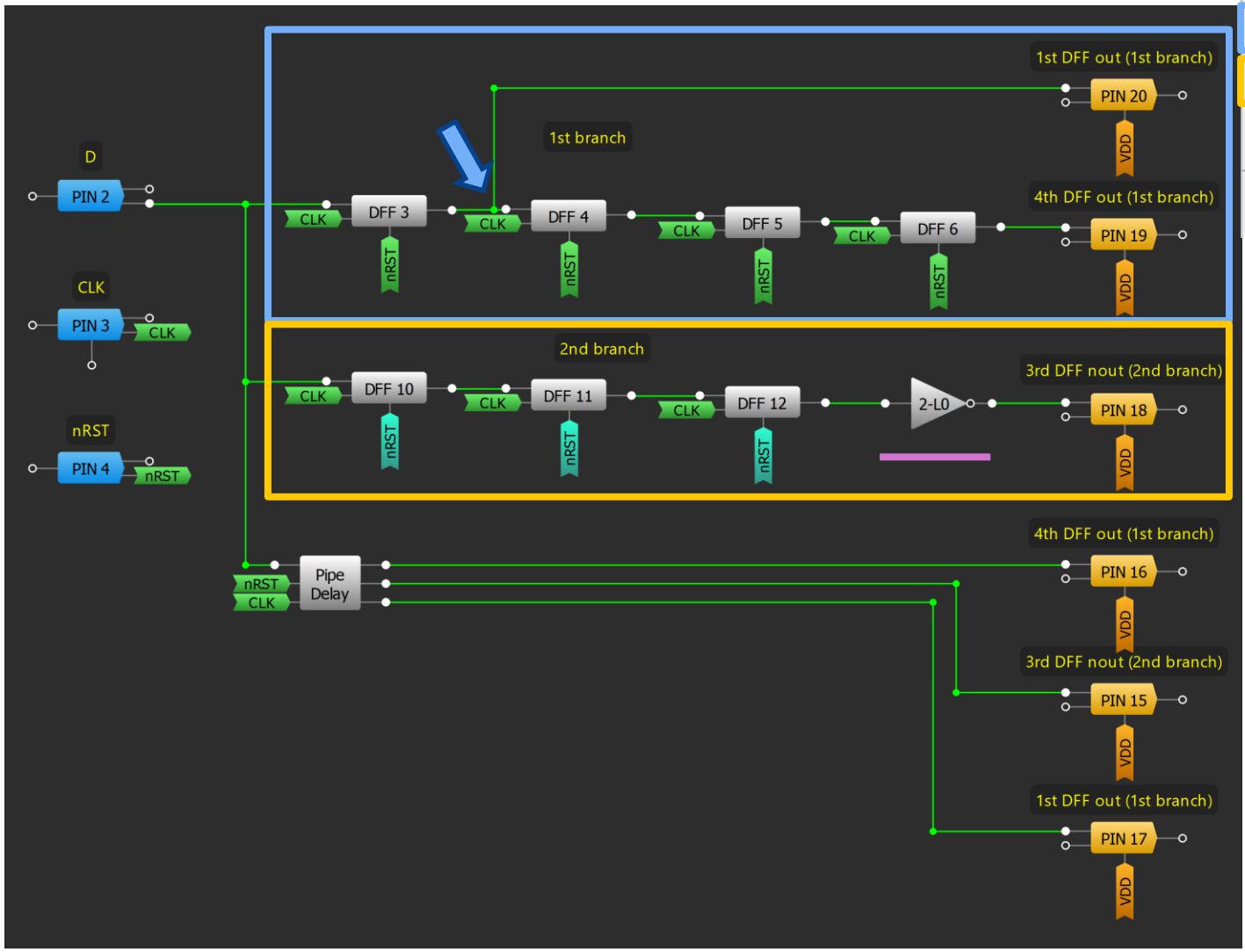
FIFO (First In, First Out) "перший прийшов — перший вийшов". FIFO-регістр — це пам'ять із черговим доступом, де дані зчитуються в тому ж порядку, в якому були записані.

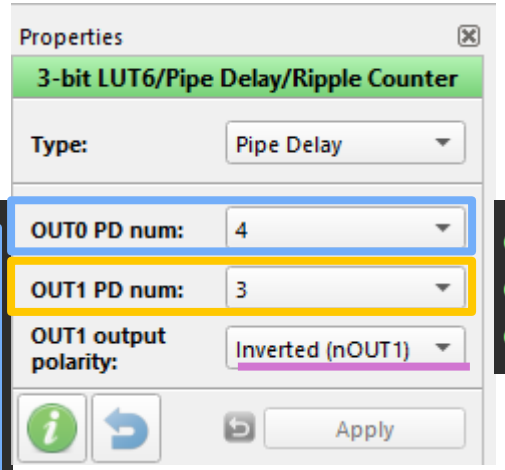
SISO

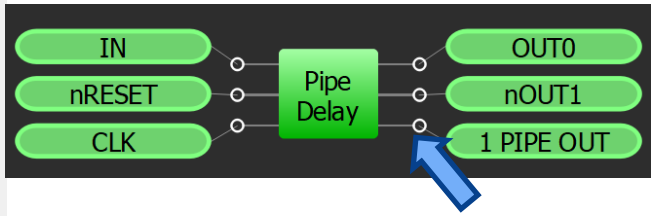

PIN 3 (Src)
Serial_in
PIN 3
VDD
VDD
DFF 3
DFF 4
DFF 5
DFF 6
DFF 7
DFF 8
DFF 9
DFF 10
clk
nReset
PIN 2 (Src)
Clock
PIN 2
PIN 4 (Src)
nReset
PIN 4
Serial_out
PIN 17
GND
GND


[P] clk
[P] PIN 3 -> OUT
[P] DFF 3 -> Q
[P] DFF 4 -> Q
[P] DFF 5 -> Q
[P] DFF 6 -> Q
[P] DFF 7 -> Q
[P] DFF 8 -> Q
[P] DFF 9 -> Q
[P] DFF 10 -> Q
[P] PIN 17 <- EXT. IN/OUT
High
Low

SHIFT REGISTER EXAMPLE







Pipe Delay структурно є регістр зсуву, котрий можна налаштувати максимум на 16 поліслovно ввімкнених DFF.
На рисунку показано Pipe Delay і DFF+INV, котрі можна використати для створення ідентичного функціоналу

OUT0 PD num – кількість DFF використаних в для першого виходу.

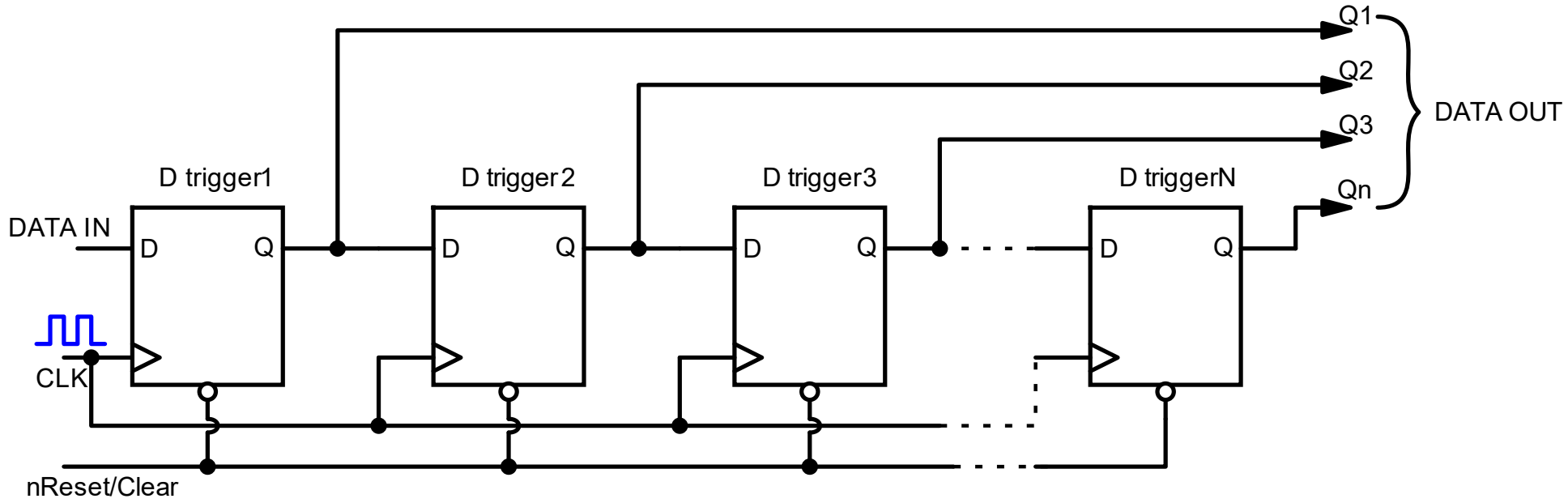
OUT1 PD num – кількість DFF використаних в для другого виходу. Другий вихід може бути інвертований (зручно для створення дільників частоти, машин станів, лічильників)

1 PIPE OUT – вихід першої DFF

SIPO

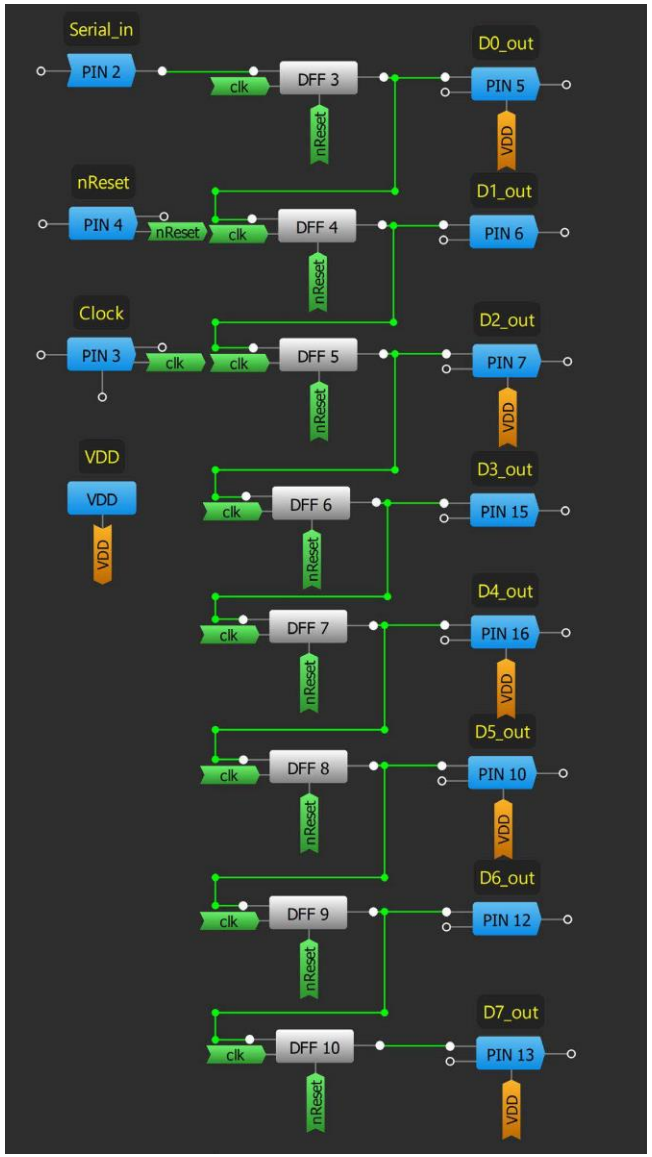
SIPO-регістр — це тип зсувного регістра, який приймає дані послідовно (біт за бітом) і виводить їх паралельно (всі біти одночасно). Він використовується для перетворення послідовних даних у паралельні, що дозволяє зменшити кількість ліній зв’язку між пристроями.

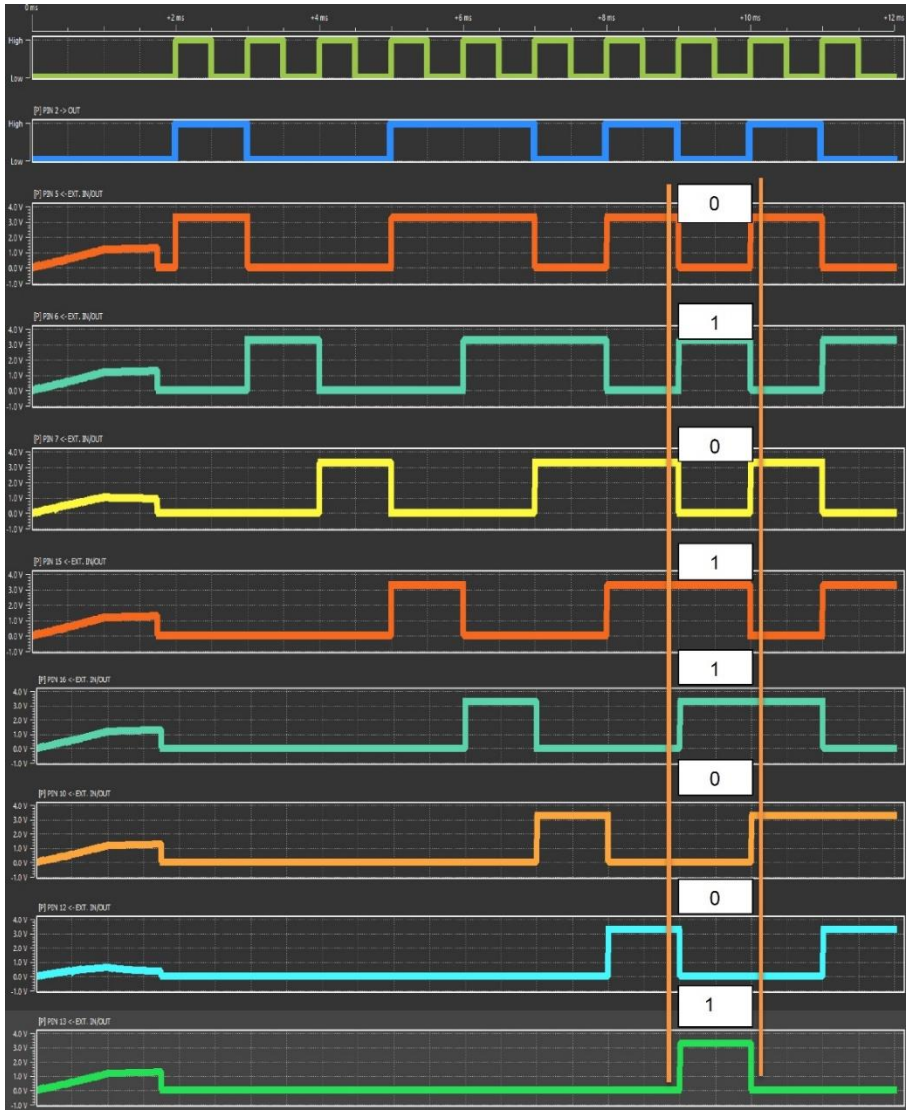
Основою SIPO-регістра є ланцюг D-тригерів, де кожен тригер зберігає один біт інформації. Під час подачі кожного тактового імпульсу біт із серійного входу передається в перший тригер, а дані в регістрі зсуваються на один біт вправо. Після завершення прийому всі біти одночасно зчитуються з паралельних виходів.



Serial input parallel output(SIPO)

SIPO





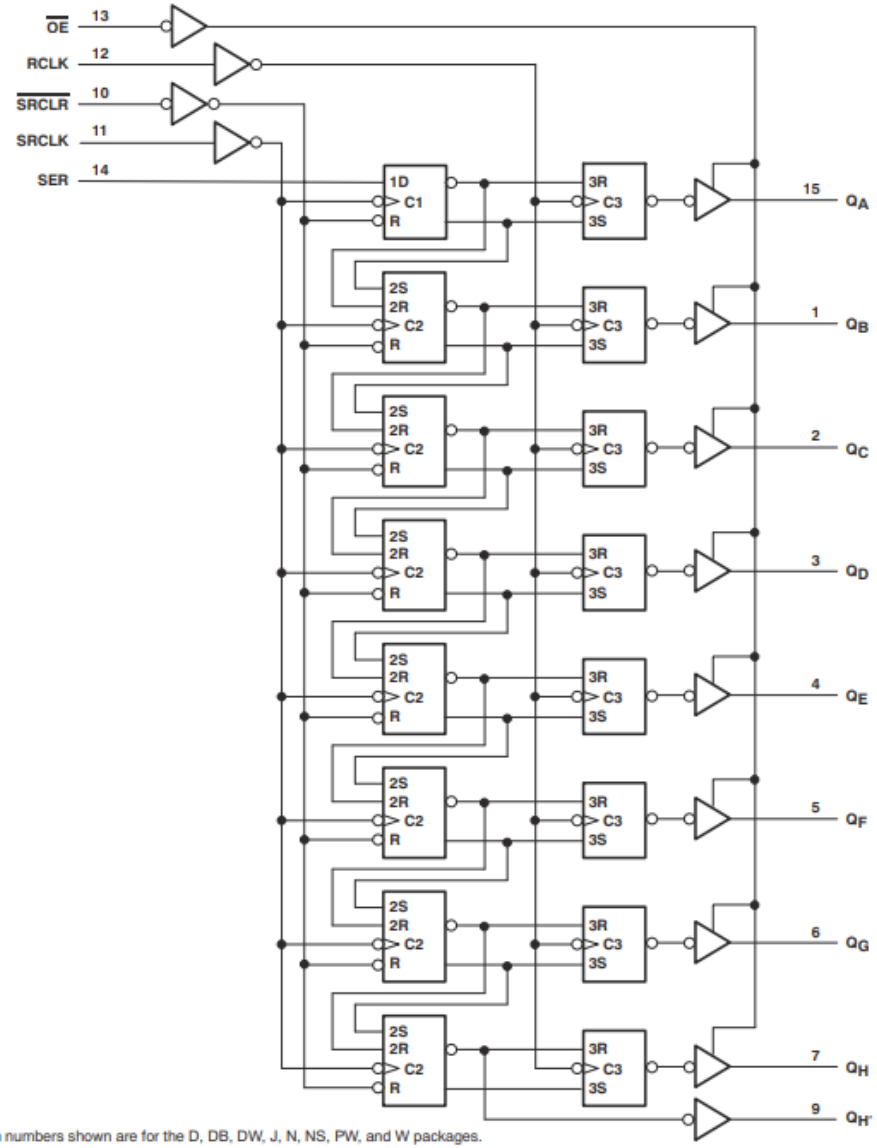
Реальне застосування SIPO-регістра:
- керування світлодіодними панелями — наприклад, 74HC595 (розглянемо далі) часто використовується для зменшення кількості керуючих ліній.
- обмін даними між мікроконтролерами через SPI або UART.
- буферизація серійного потоку даних перед їх одночасною обробкою.
- підключення дисплеїв, індикаторів, модулів пам’яті.

Переваги:
- зменшення кількості вхідних ліній
- зручність у передачі послідовних даних через одну лінію

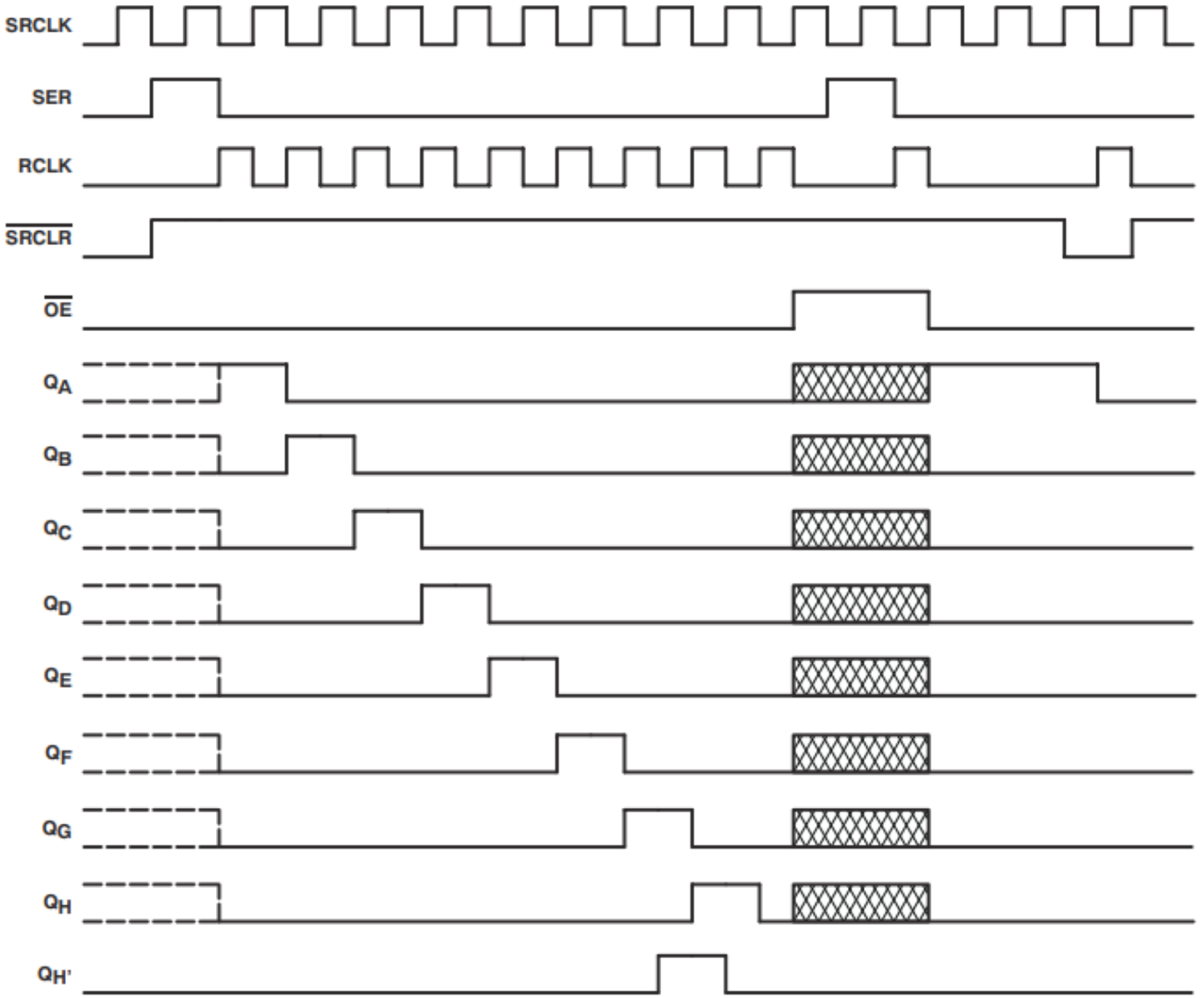
Недоліки:
- необхідність чекати завершення прийому всіх бітів перед зчитуванням.
- швидкість обмежена кількістю тактових імпульсів для повного запису.

https://www.ti.com/lit/ds/symlink/sn74hc595.pdf

SIPO



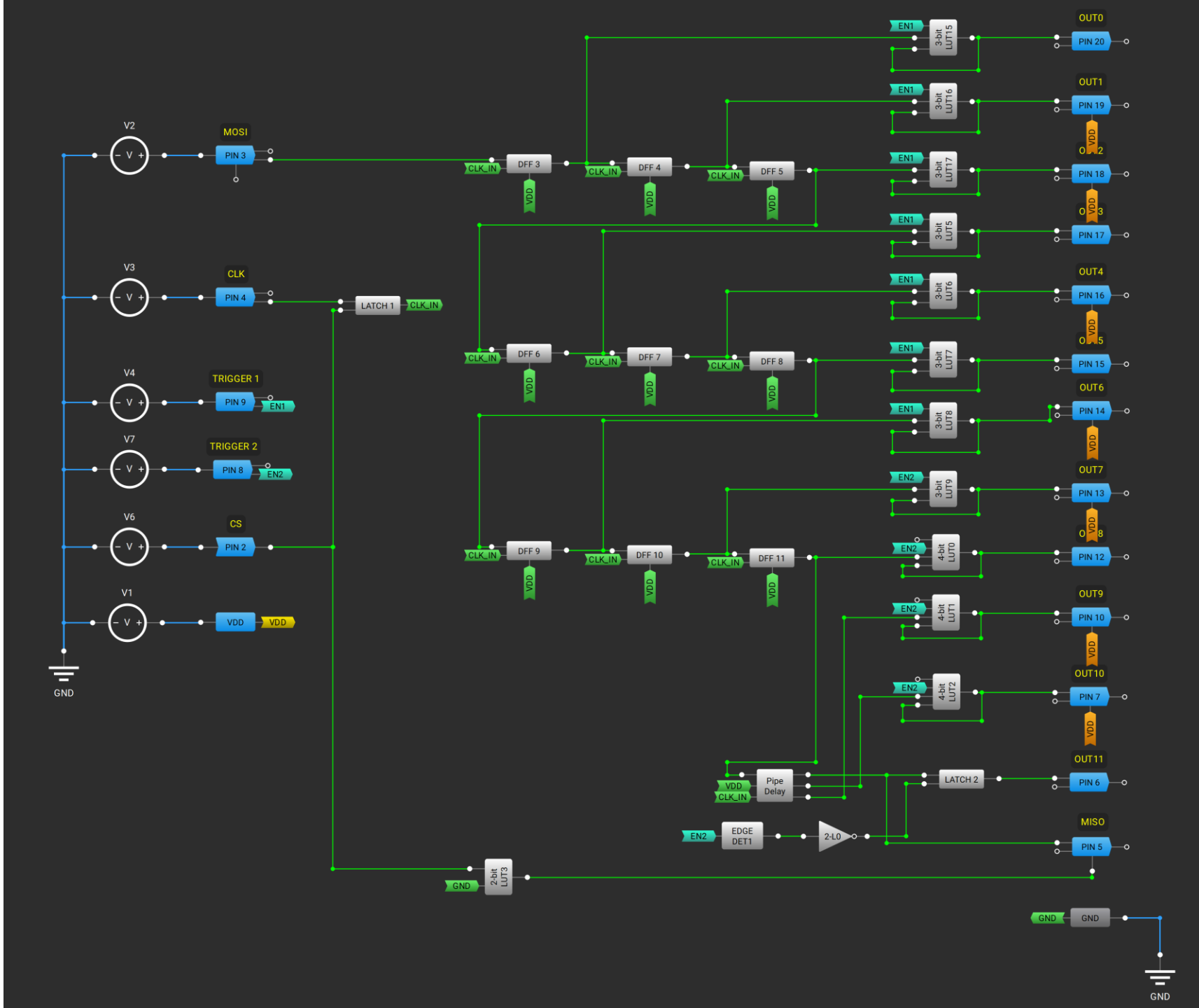
Pin numbers shown are for the D, DB, DW, J, N, NS, PW, and W packages.



SHADOW BUFFER

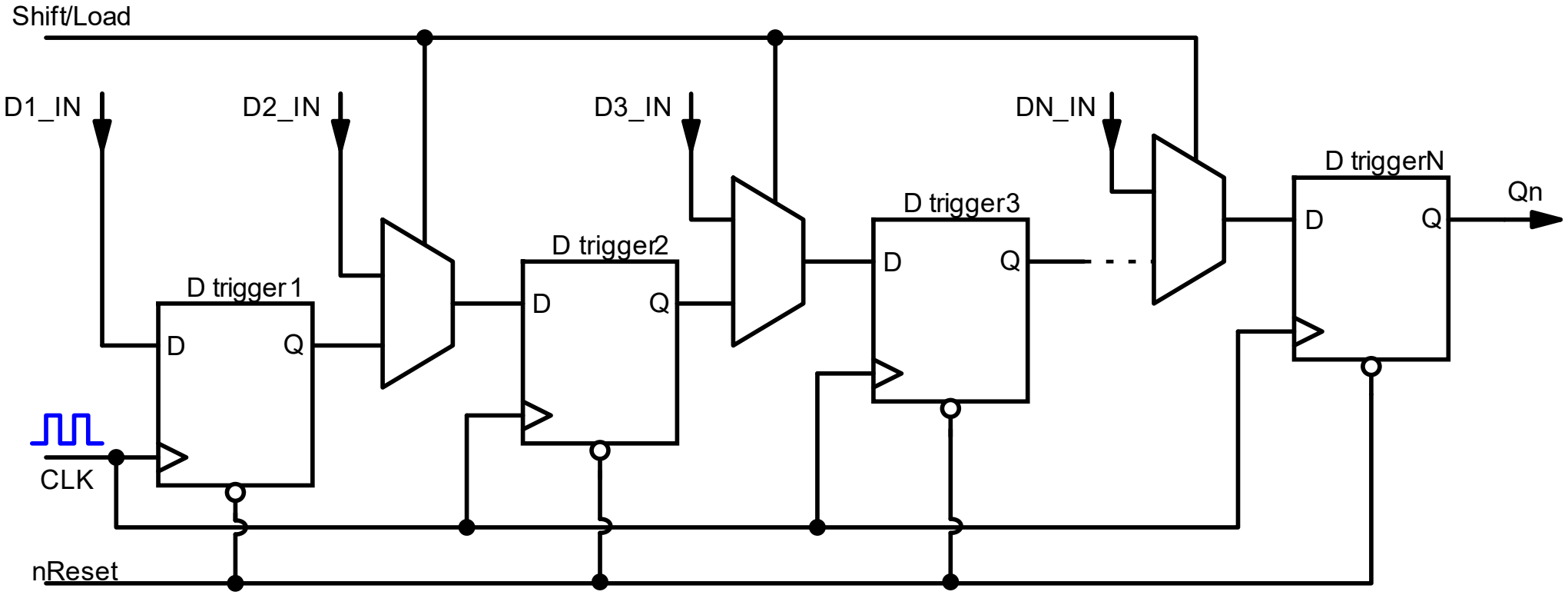
Оф топік.

Реальний кейс, котрий я виконав для одного замовника

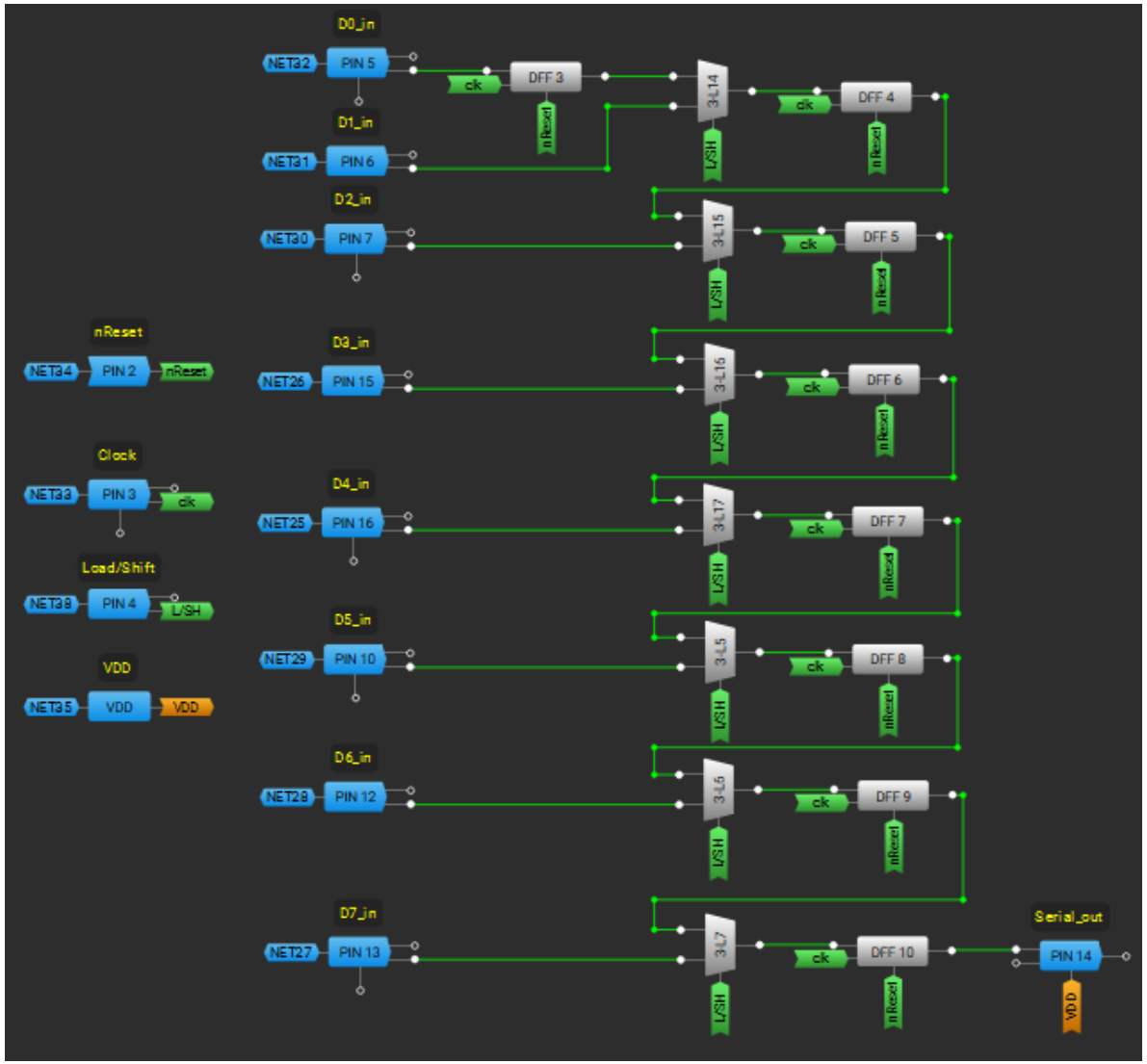


PISO

PISO-регістр — це тип регістра, який дозволяє **паралельно записувати дані**, а потім **послідовно передавати їх через один вихід**. Він використовує **паралельні входи** для зберігання всіх бітів, але передача їх здійснюється через **серійний вихід** по одному біту за раз. Це зручно для перетворення **паралельних даних у серійні**, що дозволяє економити кількість ліній для передачі даних.



PISO EXAMPLE

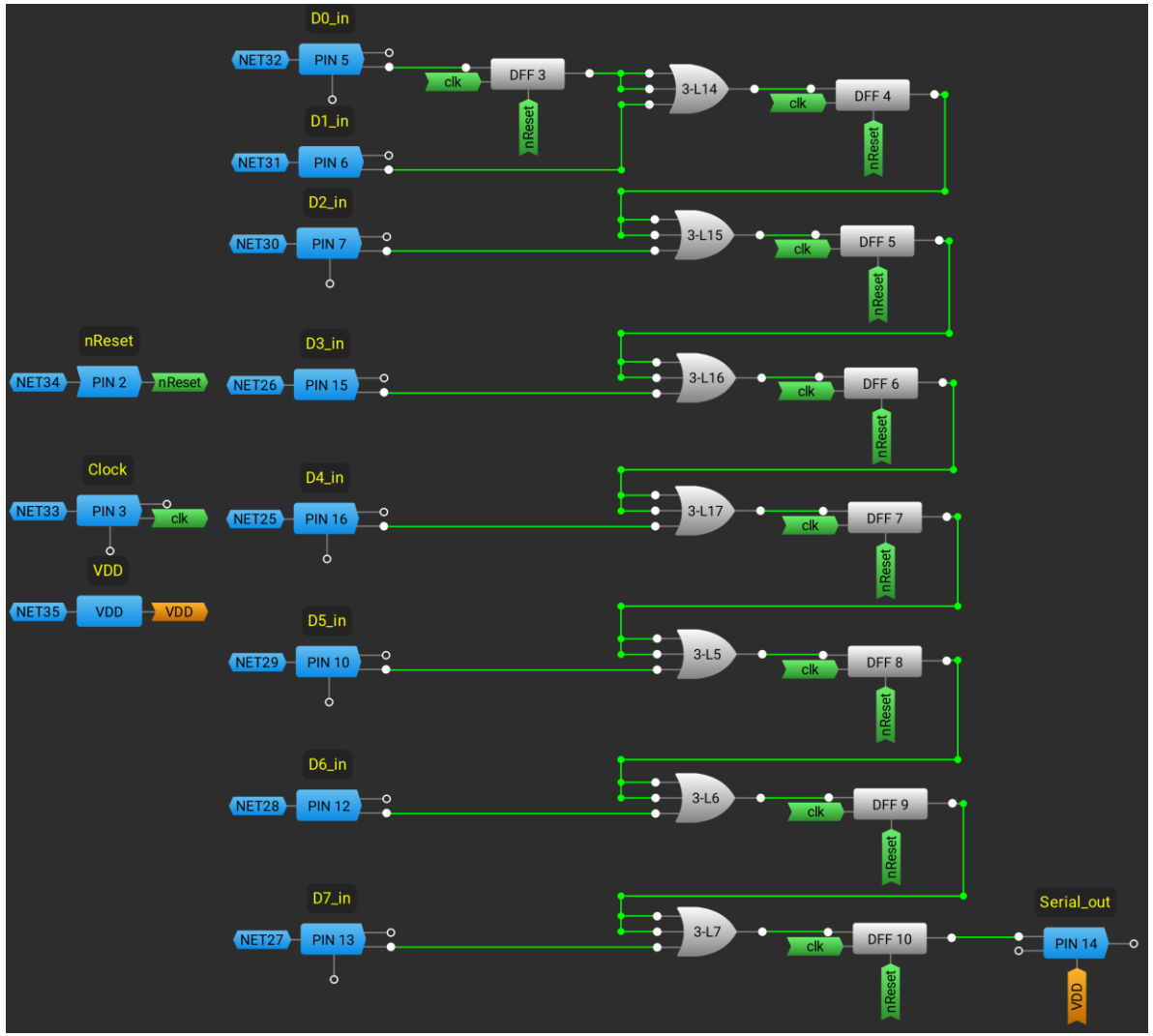


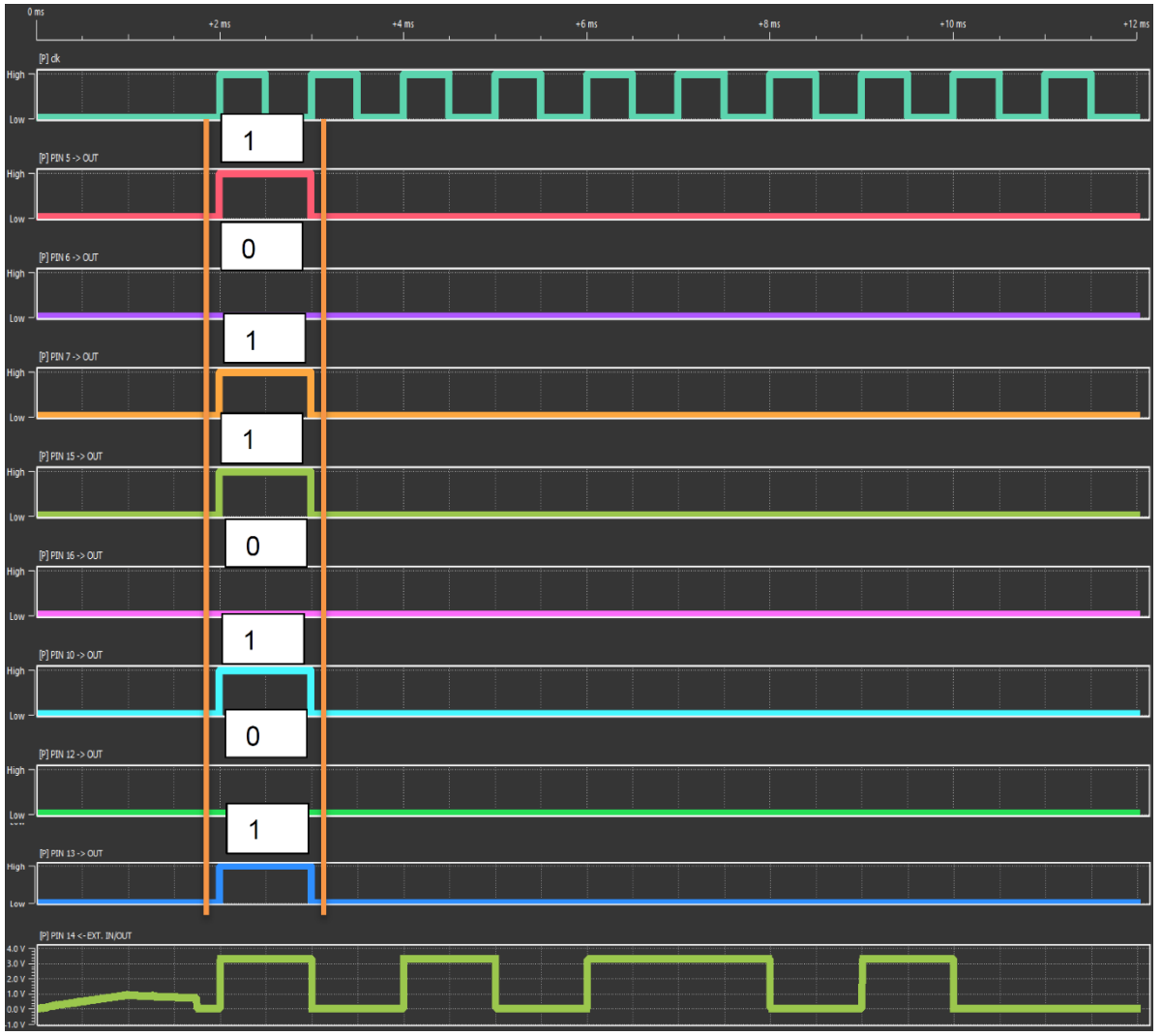
Застосування PISO-регістра
- перетворення паралельних даних у послідовні — використовується в цифрових системах передачі даних, де потрібно передавати великі обсяги інформації через один канал (наприклад, SPI або UART).
- керування пристроями через послідовний інтерфейс
- Шини даних між мікросхемами — в разі, коли є потреба передавати дані на великі відстані через одну серійну лінію.

Переваги:
- економія кількості ліній зв'язку — замість чотирьох ліній можна використовувати одну для передачі даних.
- швидка передача даних через серійний канал.
- зручність при підключенні до послідовних інтерфейсів (SPI, UART).

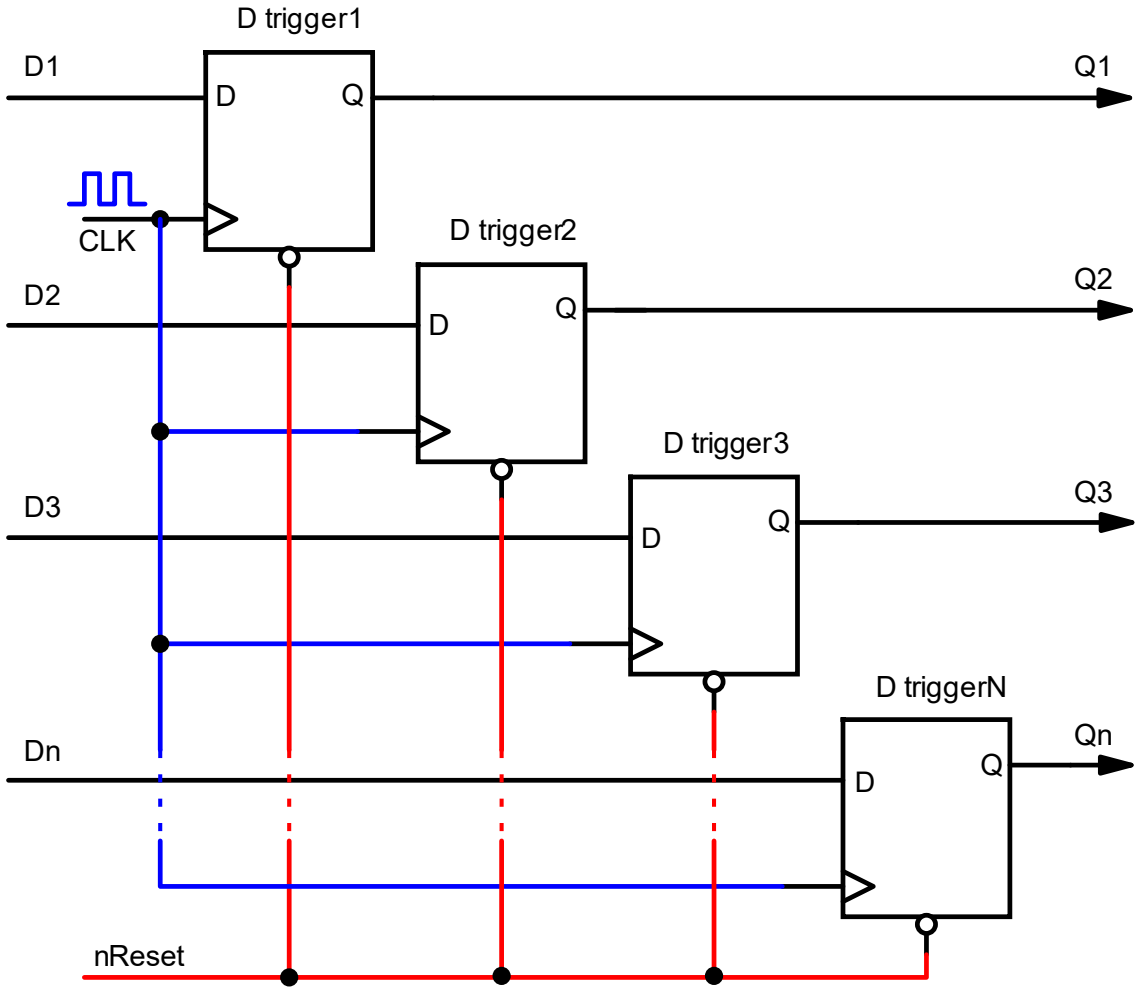
Недоліки:
- необхідність кількох тактових імпульсів для повного передачі, оскільки процес займає більше часу, ніж паралельний запис.
- швидкість обмежена тактовим сигналом, бо дані передаються по одному біту за раз.

PISO EXAMPLE





PIPO



PIPO-регістр — це тип регістра, який приймає і виводить дані **паралельно**, тобто всі біти одночасно. На відміну від SIPO або PISO регістрів, тут немає зсуву — **вхідні дані записуються в регістр за один тактовий імпульс** і одночасно з'являються на виходах.

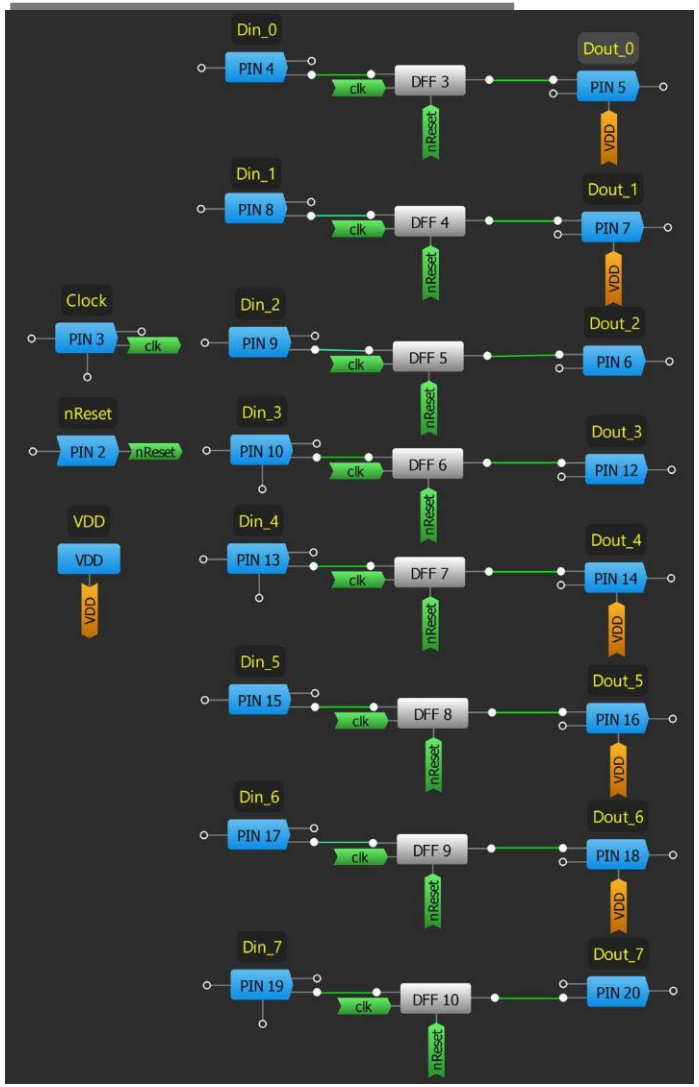
PIPO-регістри часто використовуються у випадках, коли важливо **швидко передавати або зберігати дані**, які обробляються **паралельно**:

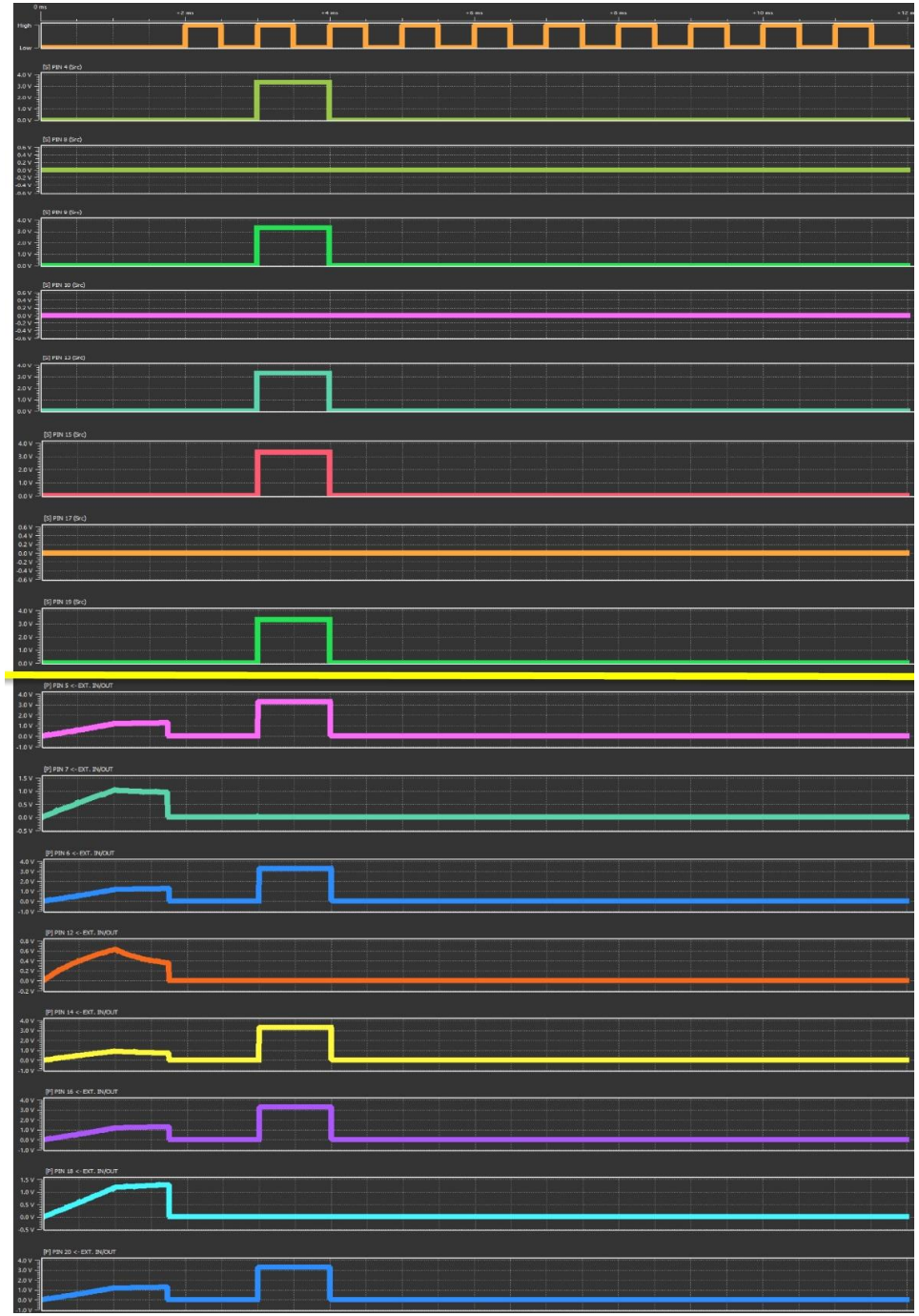
- буферизація даних та синхронізація між пристроями.
- швидке пересилання інформації в цифрових системах.
- тимчасове збереження даних перед їх подальшою обробкою.
- ALU для роботи з операндами.

Переваги: швидка передача даних (всі біти зчитуються і записуються одночасно), проста схема без зсуву.

Недоліки: більше ліній підключення потрібна окрема лінія для кожного біта, менша гнучкість у порівнянні з SIPO чи PISO.

PIPO

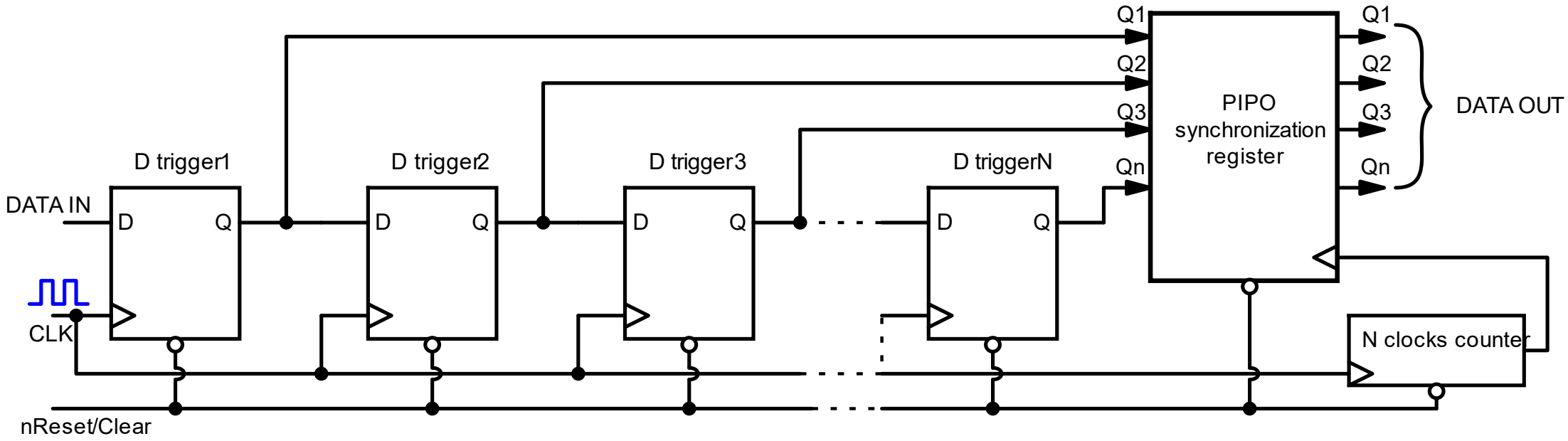




Inputs

Outputs

SIPO+PIPO (SIPO WITH SYNCHRONIZED OUTPUT)

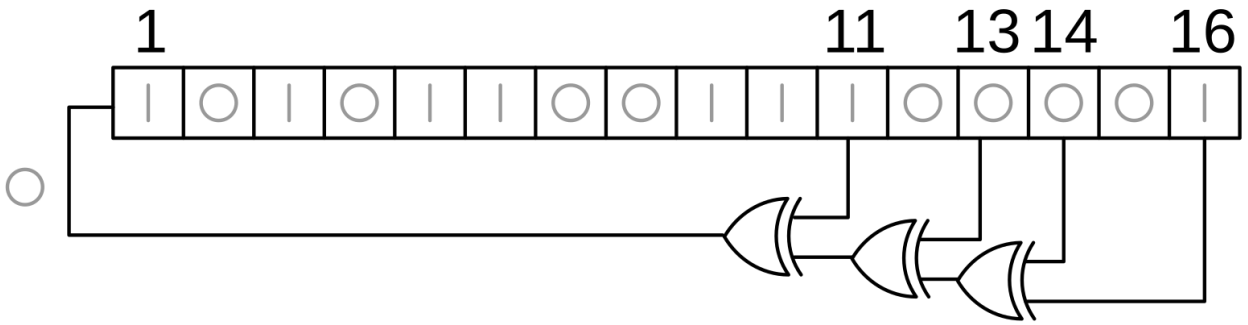


Serial input parallel output (SIPO) output is synchronized

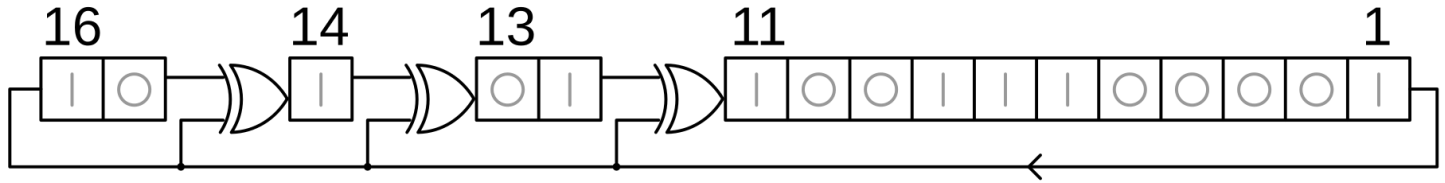
SIPO з синхронізаційним виходом може бути створений дованням PIPO.
PIPO усвою чергу записується наростаючим рівнем, що формується на виході N-бітного лічильника
Дані навиході PIPO завжди будуть змінюватись одночасно, кожних N синхроімпульсів, що приходять на вхід регістра

LFSR (LINEAR FEEDBACK SHIFT REGISTER)

LFSR (Linear Feedback Shift Register) — це лінійний регістр зсуву з зворотним зв'язком, який генерує послідовність бітів на основі двійкової арифметики і операції XOR. LFSR широко використовується для генерації **псевдовипадкових чисел**, тестування цифрових схем, кодування і шифрування даних, а також у системах передачі інформації.



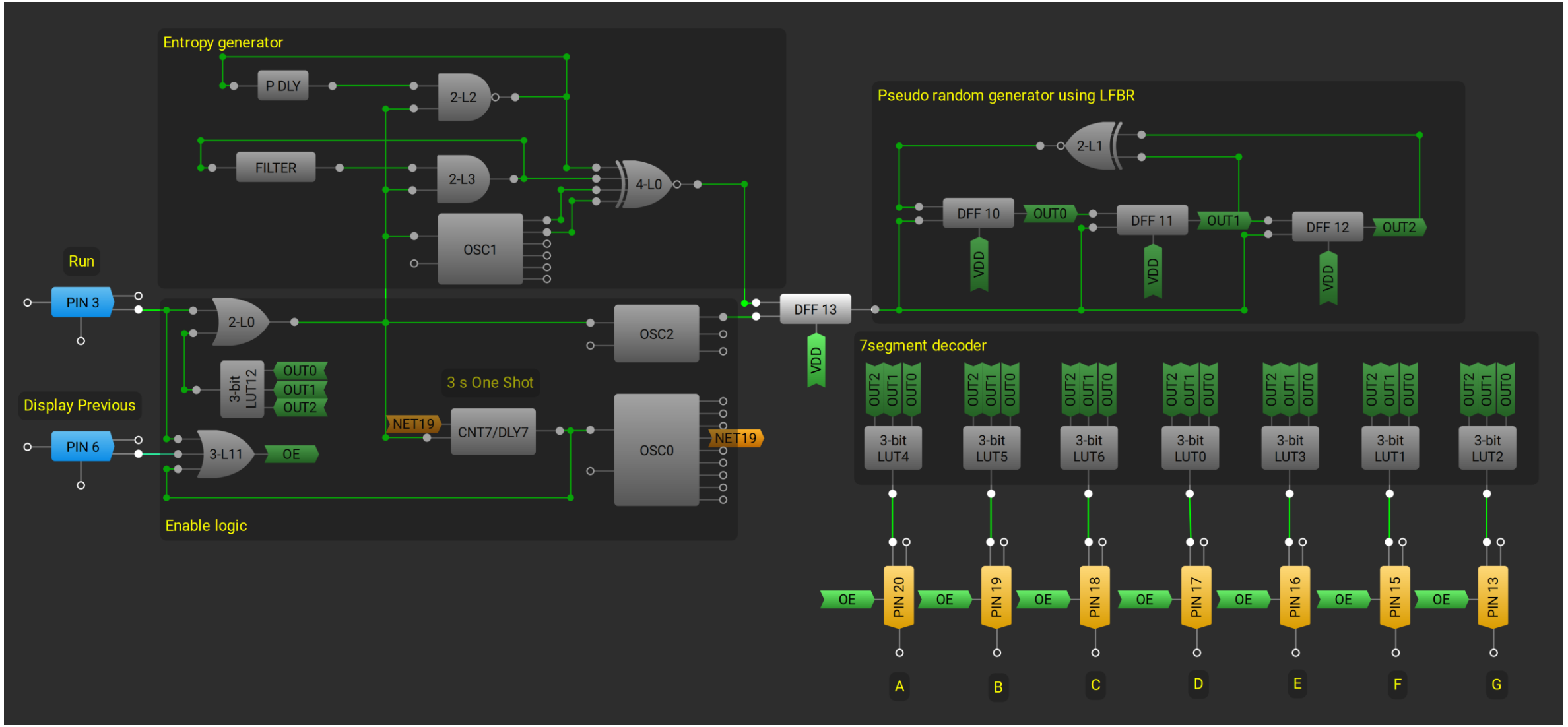
Fibonacci LFSR. Зворотний зв'язок подається **до входу регістра зсуву**. Вхідний біт формується як XOR певних бітів регістра.



Galois LFSR. Зворотний зв'язок застосовується **всередині регістра під час зсуву**.
XOR виконується безпосередньо з кожним бітом на основі зворотного зв'язку.

https://en.wikipedia.org/wiki/Linear-feedback_shift_register

PSEUDO RANDOM GENERATOR USING LFBR

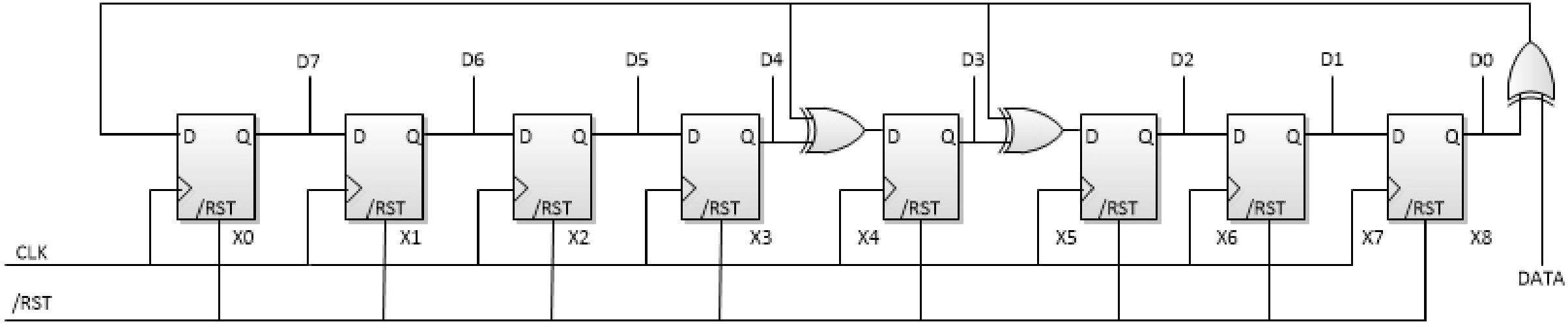


LOGIC DIAGRAM OF 1-WIRE CRC GENERATOR UNIT

CRC (Cyclic Redundancy Check) — це один із найпопулярніших і надійних методів виявлення помилок при передачі даних у цифрових системах. CRC широко використовується в комунікаційних протоколах (наприклад, Ethernet, USB) і файлових системах. Його основна ідея — створення контрольної суми на основі поліноміальних обчислень.

Регістри зсуву відіграють ключову роль у реалізації CRC. Завдяки їхній здатності зміщувати біти і виконувати побітові операції, вони забезпечують ефективне та швидке обчислення контрольної суми.

The generator polynomial for the CRC is: $x^8+x^5+x^4+1$



LOGIC DIAGRAM OF 1-WIRE CRC GENERATOR UNIT

Принцип роботи CRC за допомогою регістрів зсуву:

1.Ініціалізація регістра зсуву. Регістри зсуву заповнюються початковим значенням (наприклад, усі біти нулі або одиниці).

2.Подача вхідних даних**.** Дані передаються побітово в регістр зсуву.

3.Зсув і XOR. Під час кожного тактового імпульсу відбувається зсув на один біт, а найстарший біт регістра використовується для визначення необхідності виконання XOR з фіксованим поліномом.

4.Формування контрольної суми. Після обробки всіх бітів отримуємо значення CRC, яке можна передавати разом із даними.

5.Перевірка на приймальній стороні. Отримані дані разом із CRC знову пропускаються через той самий механізм, і якщо вихідний CRC дорівнює нулю — помилок немає.

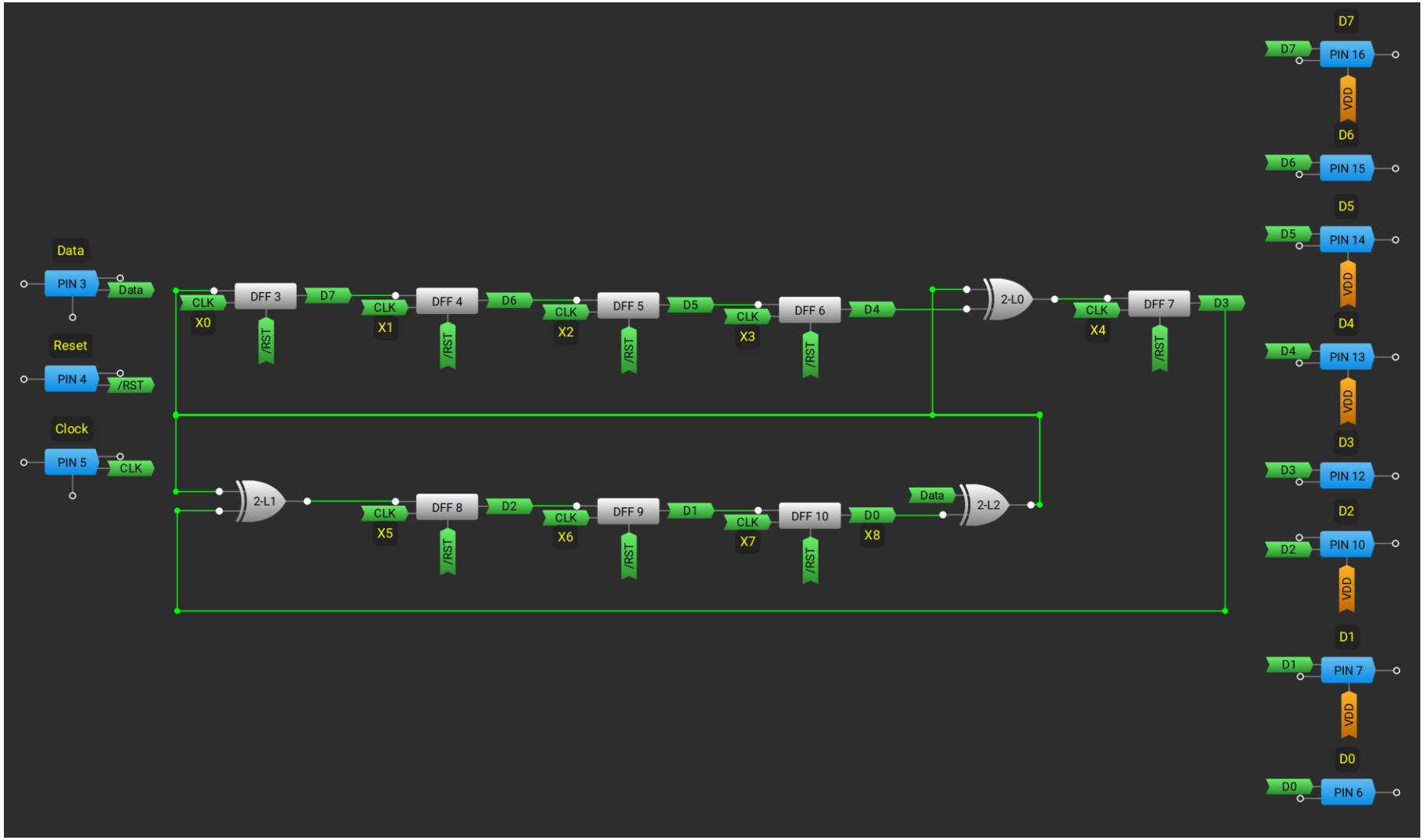
Переваги використання регістрів зсуву для CRC:

1.Швидкість і ефективність — простий апаратний підхід забезпечує швидке обчислення CRC.

2.Мінімум ресурсів — реалізація CRC на регістрах зсуву потребує невеликої кількості логічних елементів.

3.Гнучкість — можна легко налаштовувати під різні поліноми CRC (наприклад, CRC-8, CRC-16, CRC-32).

ПРИКЛАД РЕАЛІЗАЦІЇ CRC-8



Приклад реалізації CRC-8 з поліномом $x^8+x^5+x^4+1$, використовуючи Shift Register з паралельним виходом

ПИТАННЯ ОФ ТОПІК

1. POR
2. LDO
3. Потенціал, різниця потенціалів

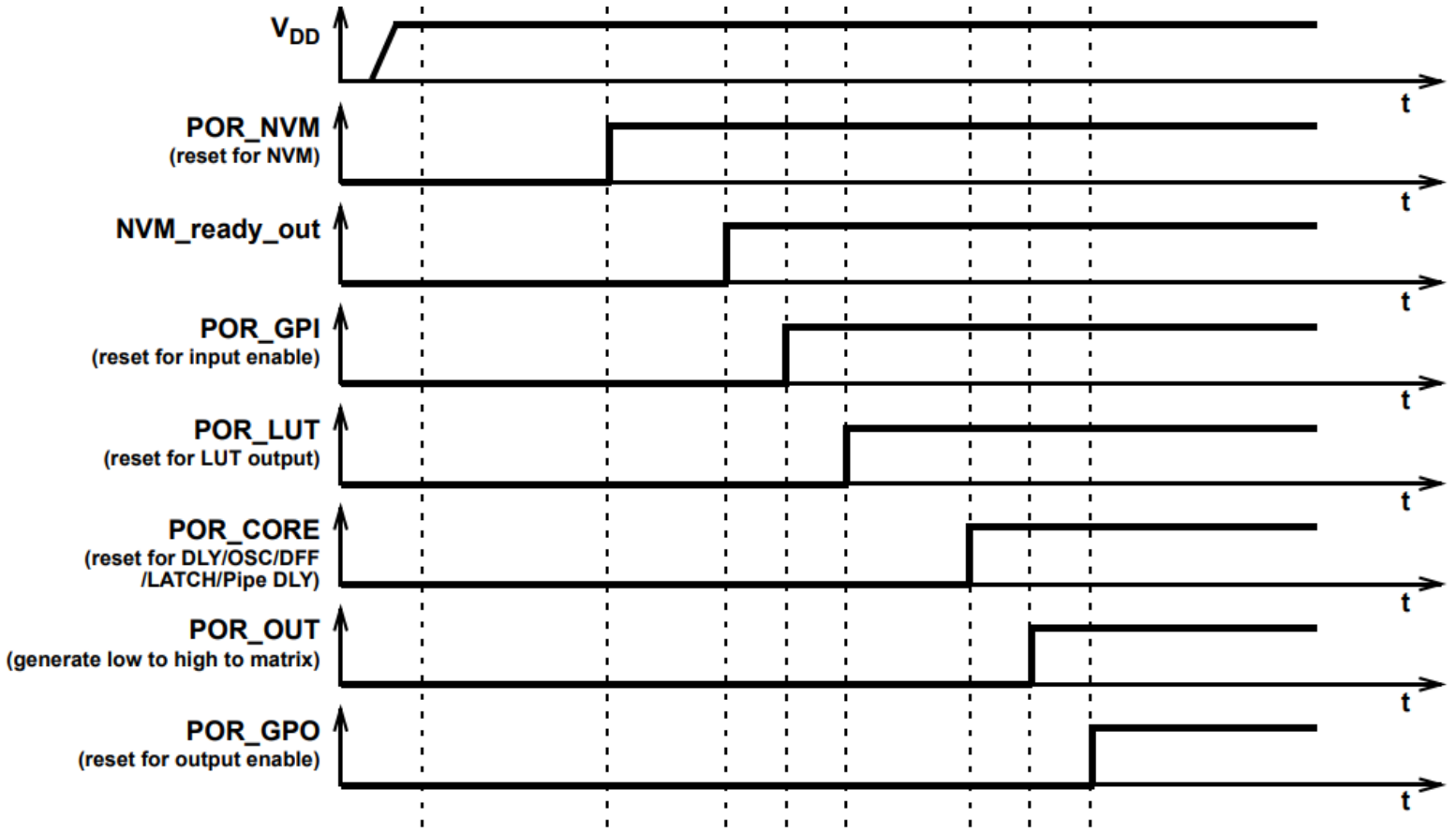
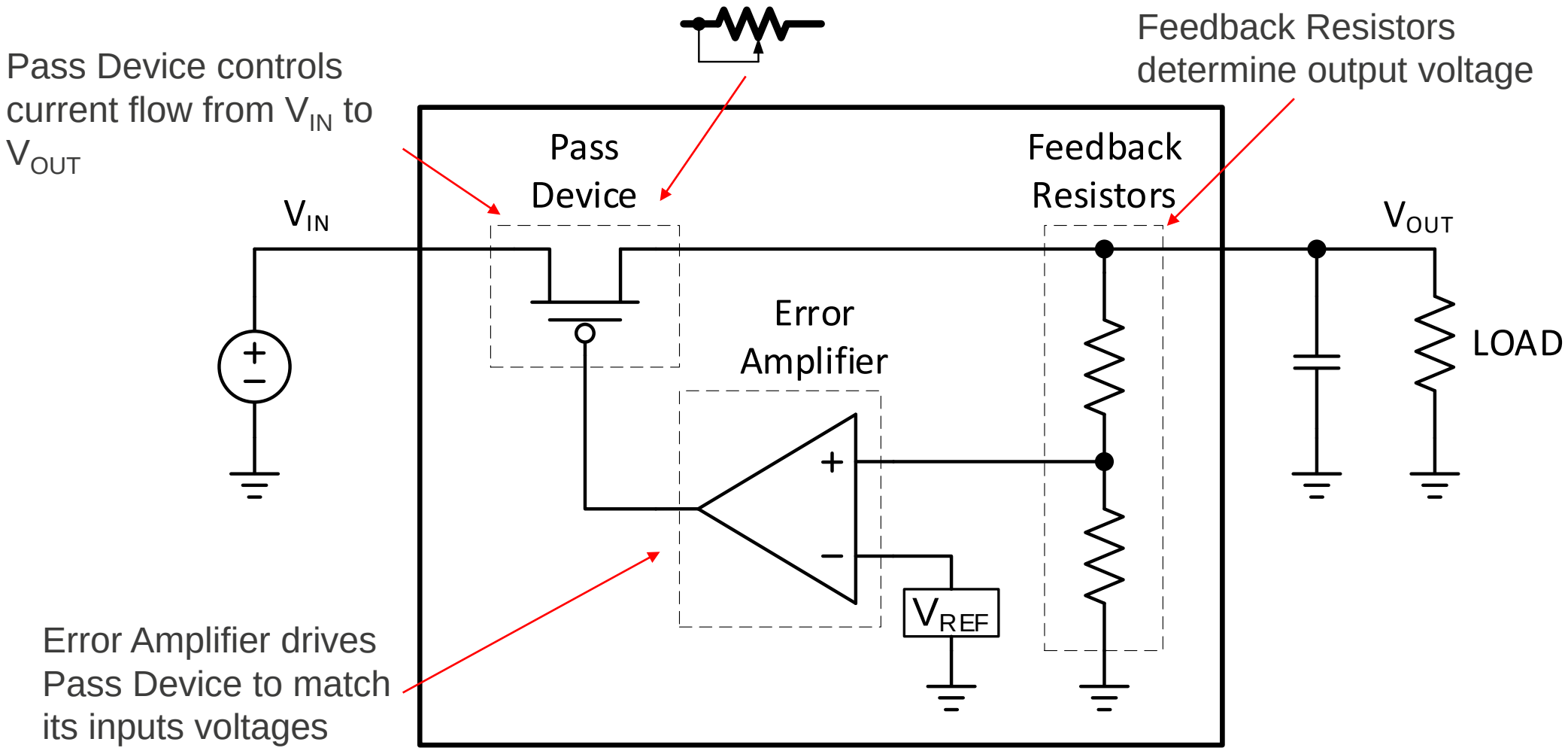


Figure 89: POR Sequence

WHAT IS LOW-DROPOUT REGULATOR (LDO)?

SIMPLIFIED LDO ARCHITECTURE



Renesas.com