Лабораторная работа №1	39	2022
Построение логических схем в среде моделирования	Ляпин Дмитрий Романович	

Цель работы: моделирование логических схем на элементах с памятью.

Инструментарий и требования к работе: работа выполняется в среде моделирования Logisim evolution.

Описание

Составить и описать принцип работы двух схем: счётчика и регистра сдвига с линейной обратной связью.

Вариант

1. Асинхронный вычитающий счётчик по модулю 15
2. РСЛОС в конфигурации Галуа (5, 3)

Счётчик

Принцип работы

Счётчик – схема, отсчитывающая входящие импульсы и выводящая количество в двоичном представлении на выходах. Поскольку схема, очевидно, конечна, она не может считать сколько угодно; поэтому выводится количество импульсов по модулю заданного числа, обычно этот модуль – степень двойки, но у нас задание – по модулю 15.

Счётчик может на каждый входящий импульс увеличивать записанное количество, например, по модулю 4:

$$0 \to 1 \to 2 \to 3 \to 0 \to 1 \to \cdots$$

– такой счётчик называется суммирующим; а может уменьшать, например:

$$0 \to 3 \to 2 \to 1 \to 0 \to 3 \to \cdots$$

– такой счётчик называется вычитающим.

В зависимости от принципа работы счётчики делятся на синхронные и асинхронные. В обоих через полтакта вычисляется новое значение, ещё через полтакта – перезаписывается. Разница в том, что в случае асинхронного новое значение вычисляется и перезаписывается для каждого бита отдельно, и, как следствие, с небольшой задержкой относительно друг друга, в случае же синхронного – вычисляется и перезаписывается одновременно.

Счётчик по модулю 2

Прежде всего, построим счётчик из одного бита, иначе говоря, схему, которая будет делить частоту тактирования на два.

Возьмём за основу D-триггер (схема на Рис.1, подсхема Dff в файле)

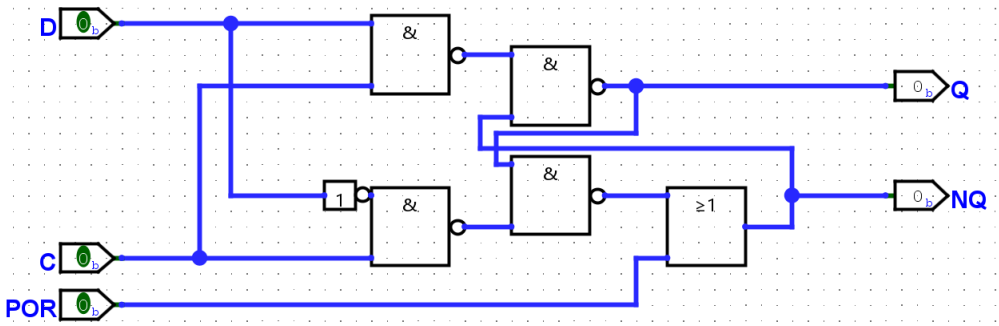

Рис.1 – D-триггер

Подача сигнала на POR (Power-On Reset) в начале симуляции устанавливает триггер в состояние 0. Когда C = 1, в триггер записывается значение со входа D, когда C = 0 – игнорируется.

C	D	Q(t)	Q(t+1)
0	X	0	0
0	X	1	1
1	0	X	0
1	1	X	1

Табл.1 – Таблица истинности D-триггера

Составим два D-триггера в счётчик. Схема на рис.2, подсхема T. Когда вход C (Clock) принимает значение 0, в **D-триггер 2** записывается инвертированное значение **D-триггера 1**. Когда вход C принимает значение 1, в **D-триггер 1** записывается значение **D-триггера 2**. Таким образом, каждый такт C значение **D-триггера 1** меняется: временная диаграмма на рис.3

При этом заметим поведение, которое нам ещё пригодится: если на POR подано значение 1, то триггер ведёт себя по-другому – выход Q дублирует значение C, выход NQ равен единице (Временная диаграмма на рис.4). Это поведение теряется, если POR подвести к обоим D-триггерам, но нам это не нужно.

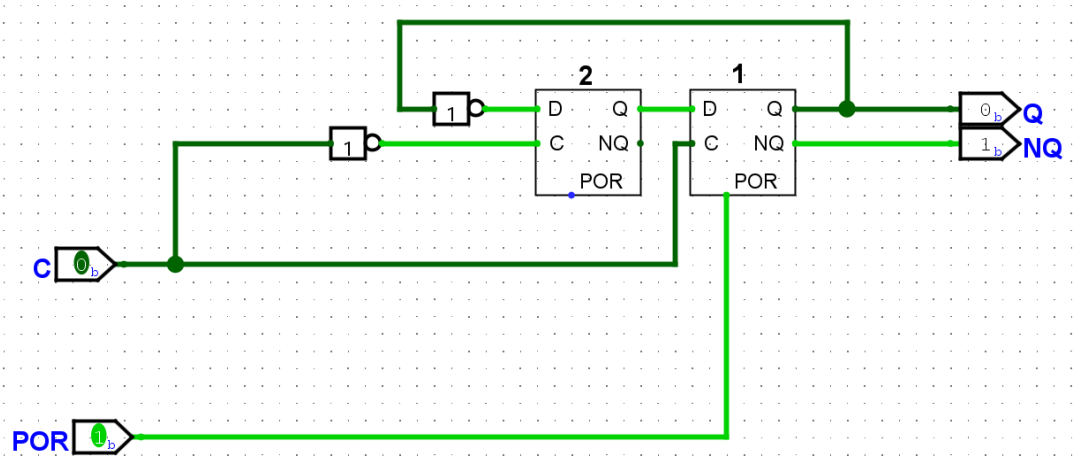

Рис.2 – Т-триггер

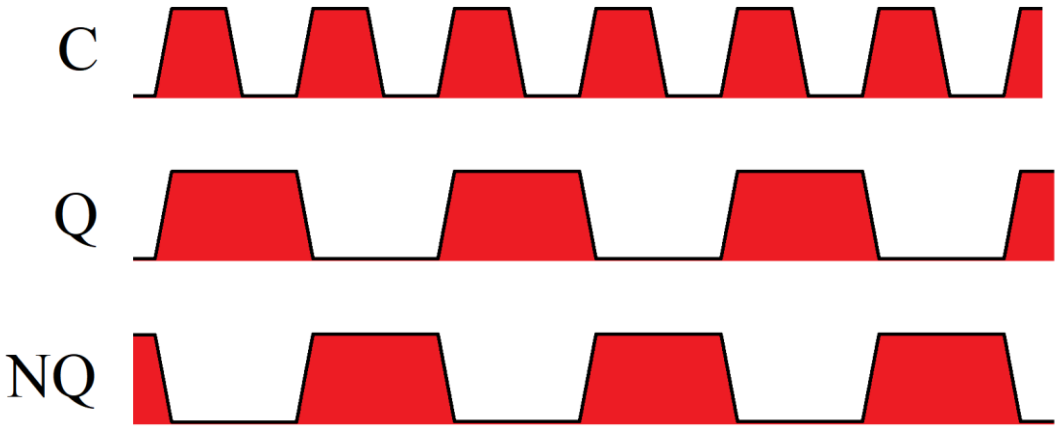

Рис.3 – Временная диаграмма Т-триггера при POR = 0

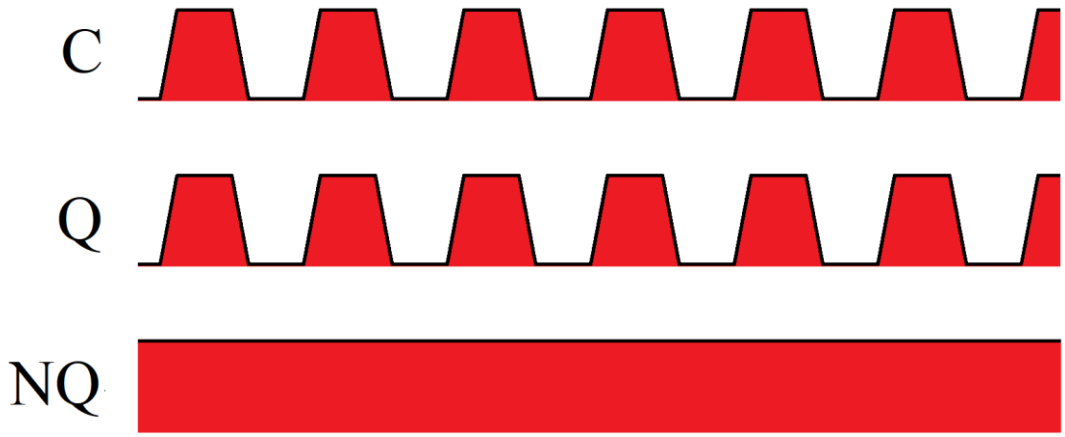

Рис.4 – Временная диаграмма Т-триггера при POR = 1

Счётчик по модулю 16

Соберём последовательно четыре Т-триггера в схему, соединяя выход Q одного со входом С другого. Получится счётчик по модулю 2^4, то есть, 16. Схема на рис.5

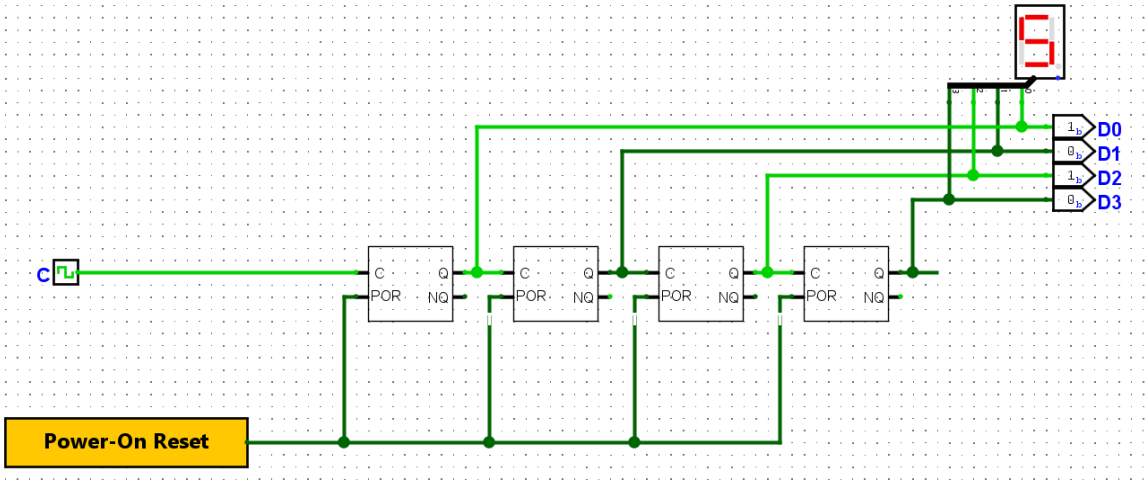

Рис.5 – Счётчик по модулю 16

Если при этом снимать значения с выходов Q, получится вычитающий счётчик, а если с NQ – суммирующий. Временная диаграмма полученного вычитающего счётчика по модулю 16 на рис.6.

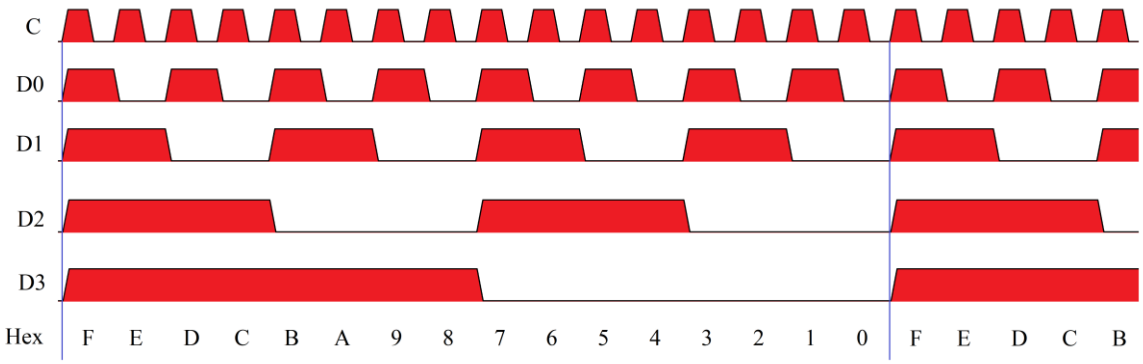

Рис.6 – Временная диаграмма счётчика по модулю 16

Счётчик по модулю 15

Теперь добавим немного, чтобы значение 15 (F) пропускалось. Но сделать это впрямую сложно – можно получить мерцание. Поэтому сделаем таким образом, чтобы значения 15 (F) и 14 (E) проходились за полтакта каждое. Этого мы сможем добиться, вспомнив описанное ранее и подав единицу на POR Т-триггера младшего бита. Таким образом «ускоряться» нам надо лишь тогда, когда остальные биты равны единице. Более того, F и E нужно интерпретировать одинаково как E, потому что F и E проходятся за один такт и должны означать E. Добавим это всё в схему: рис.7, схема main, временная диаграмма на рис.8

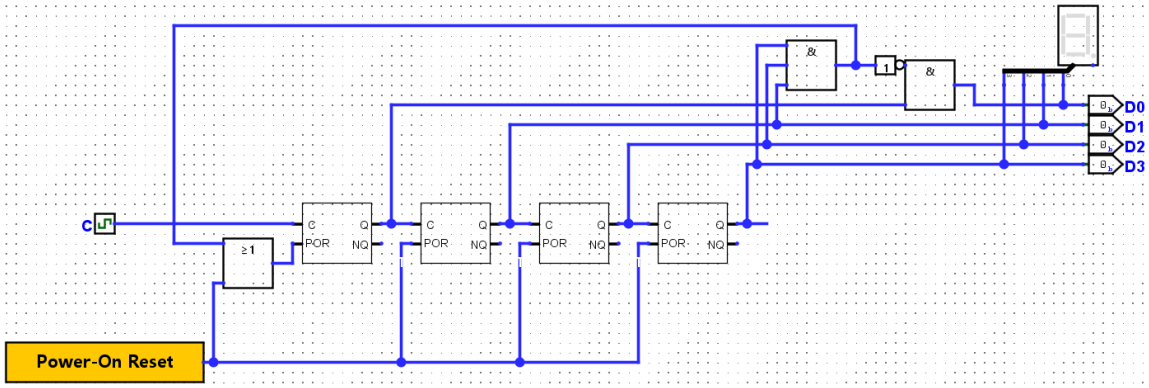

Рис.7 – Счётчик по модулю 15

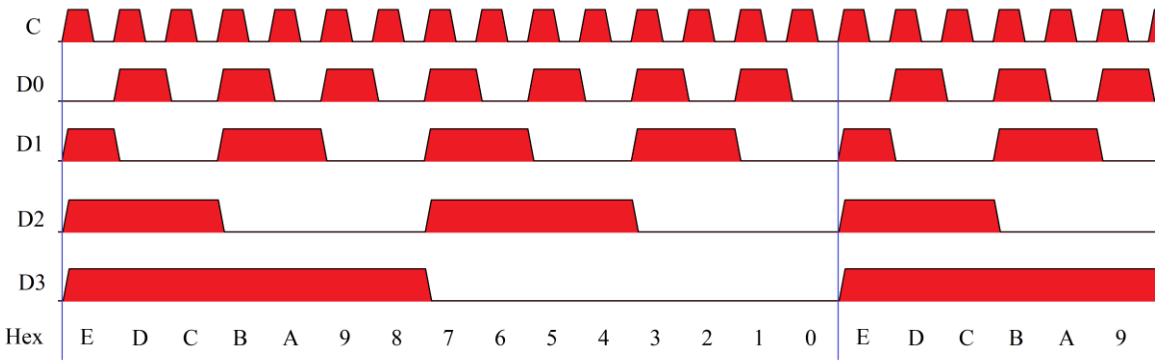

Рис.8 – Временная диаграмма счётчика по модулю 15

Регистр сдвига с линейной обратной связью

Принцип работы

Сдвиговый регистр представляет из себя последовательно в цикле соединённые ячейки памяти, по которым с частотой тактирования сдвигаются данные. «Обратная связь» предполагает, что в ячейку записывается не обязательно значение предыдущей ячейки, а функция от значений предыдущих ячеек. Линейность обратной связи означает, что эта функция линейна. На выход снимается бит с одной из ячеек – поскольку РС проходит большой цикл в околохаотическом порядке, получается генератор псевдослучайных чисел.

В случае конфигурации Фибоначчи в последнюю ячейку записывается xor от значений некоторых ячеек (обязательно включая первую, т.к. иначе нет никакого смысла в ячейках до первого ответвления), остальные сдвигаются без изменений. В случае конфигурации Галуа в некоторые ячейки вдвигается xor значения предыдущей и значения первой. В этой работе задание – конфигурация Галуа (5, 3). Обсудим, что значат эти цифры.

en.wikipedia.org/wiki/Linear-feedback_shift_register#Galois_LFSRs приводит пример 16-битного РСЛОС в конфигурации Галуа (рис.9)

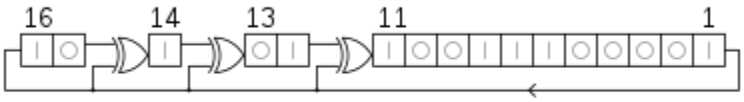

Рис.9 16-битный РСЛОС в конфигурации Галуа

Так как первый бит без изменений вдвигается в последний, без разницы, откуда снимать бит выхода – с начала или с конца. При этом РСЛОС Галуа характеризуется количеством бит, и тем, после (перед) какой ячейкой стоят xor gate. Нам дано (5, 3), из чего можно заключить, что это пять бит, и xor перед третьим (на схеме сверху отмечены именно ячейки, перед которыми стоят xor). Нумерация с единицы – видно оттуда же. Таким образом, выданную конфигурацию я интерпретирую так, как изображено на рис.10

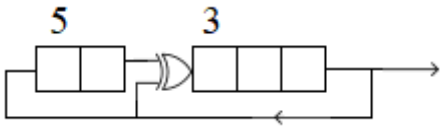

Рис.10 – Требуемая конфигурация.

Строение ячейки

Возьмём за основу D-триггер, уже описанный ранее в работе (рис. 1, табл. 1) и несколько его модернизируем (рис. 2). По аналогии с POR добавим два контакта, позволяющие установить значение 0 или 1 (S0 и S1 соответственно) в обход D и C. Они будут использоваться только в начале для задания стартового состояния.

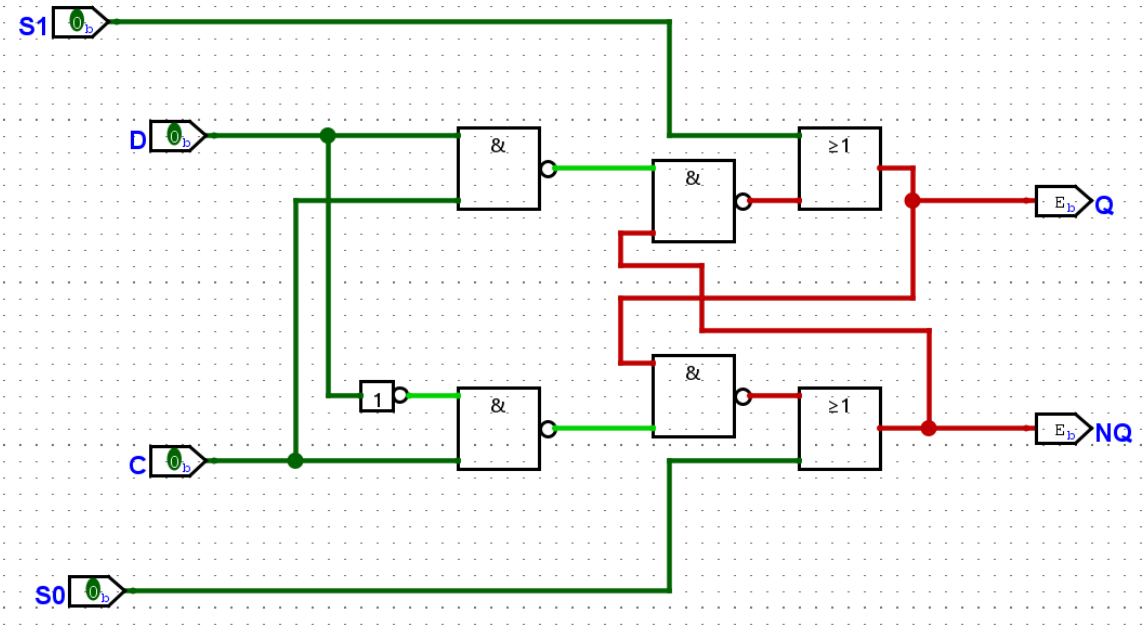

Рис.11 – Модифицированный D-trigger

Этот триггер расположен в подсхеме Dff.

Из двух таких собираем ячейку (Memory Cell, подсхема MC), которая будет хранить (с возможностью сдвига) данные. Схема на рис.12. При POR = 1 в обе ячейки записывается значение IV (initial value). В противном случае IV игнорируется. При C = 1 в левый D-триггер записывается значение со входа D, при C = 0 из левого записывается в правый. Таким образом, если соединить ячейки в линию (см. далее) при переходе $0 \rightarrow 1$ на C из всех правых D-триггеров

переписывается значение в левый D-триггер следующей ячейки, при переходе $1 \to 0$ на C – из левых в правые. Таким образом, за $0 \to 1 \to 0$ происходит сдвиг.

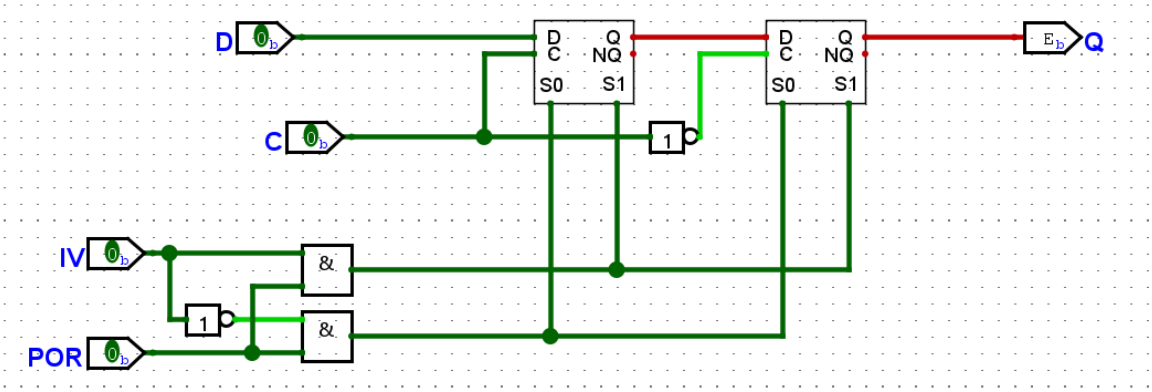

Рис.12 – Ячейка регистра

Собираем ячейки в линию (рис.13), вставляя xor gate в нужном месте в соответствии с конфигурацией. Подводим ко всем ячейкам общий Clock, общий POR, и некоторое стартовое значение. Для удобства в схеме его можно ввести одной константой. Снимаем бит на пине R.

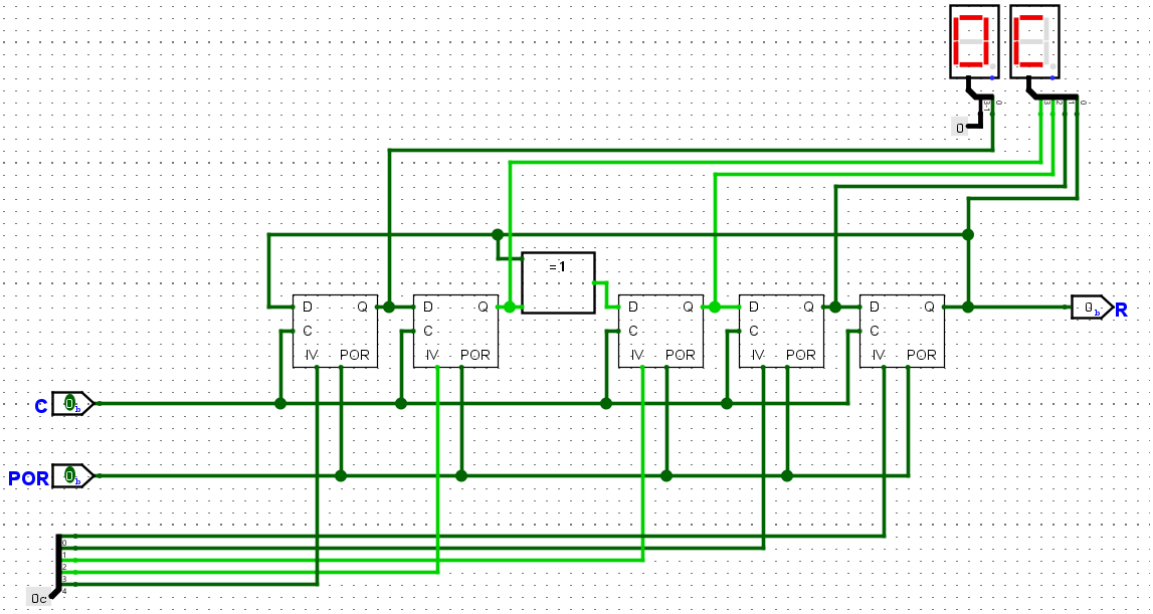

Рис.13 – Итоговый РСЛОС Галуа (5, 3)

Для удобства добавим ещё два hex индикатора – посмотрим, какие значения проходит регистр (табл. 2). Проходятся все 31 (25-1) значение, кроме нуля.

Состояние регистра		Выходной бит
0 0 0 0 1	01	1
1 0 1 0 0	14	0
0 1 0 1 0	0A	0
0 0 1 0 1	05	1
1 0 1 1 0	16	0
0 1 0 1 1	0B	1
1 0 0 0 1	11	1
1 1 1 0 0	1C	0
0 1 1 1 0	0E	0
0 0 1 1 1	07	1
1 0 1 1 1	17	1
1 1 1 1 1	1F	1
1 1 0 1 1	1B	1
1 1 0 0 1	19	1
1 1 0 0 0	18	0
0 1 1 0 0	0C	0
0 0 1 1 0	06	0
0 0 0 1 1	03	1
1 0 1 0 1	15	1
1 1 1 1 0	1E	0
0 1 1 1 1	0F	1
1 0 0 1 1	13	1
1 1 1 0 1	1D	1
1 1 0 1 0	1A	0
0 1 1 0 1	0D	1
1 0 0 1 0	12	0
0 1 0 0 1	09	1
1 0 0 0 0	10	0
0 1 0 0 0	08	0
0 0 1 0 0	04	0
0 0 0 1 0	02	0
0 0 0 0 1	01	1

Табл.2 – Состояния регистра