A Configurable Asynchronous Pseudorandom Bit Sequence Generator

Alex Chow, Bill Coates, David Hopkins
VLSI Research Group
Sun Microsystems Laboratories

Outline

- Pseudorandom bit sequences (PRBS's)
- Traditional PRBS generators
- An asynchronous implementation
  - Motivation
  - Design
  - Properties

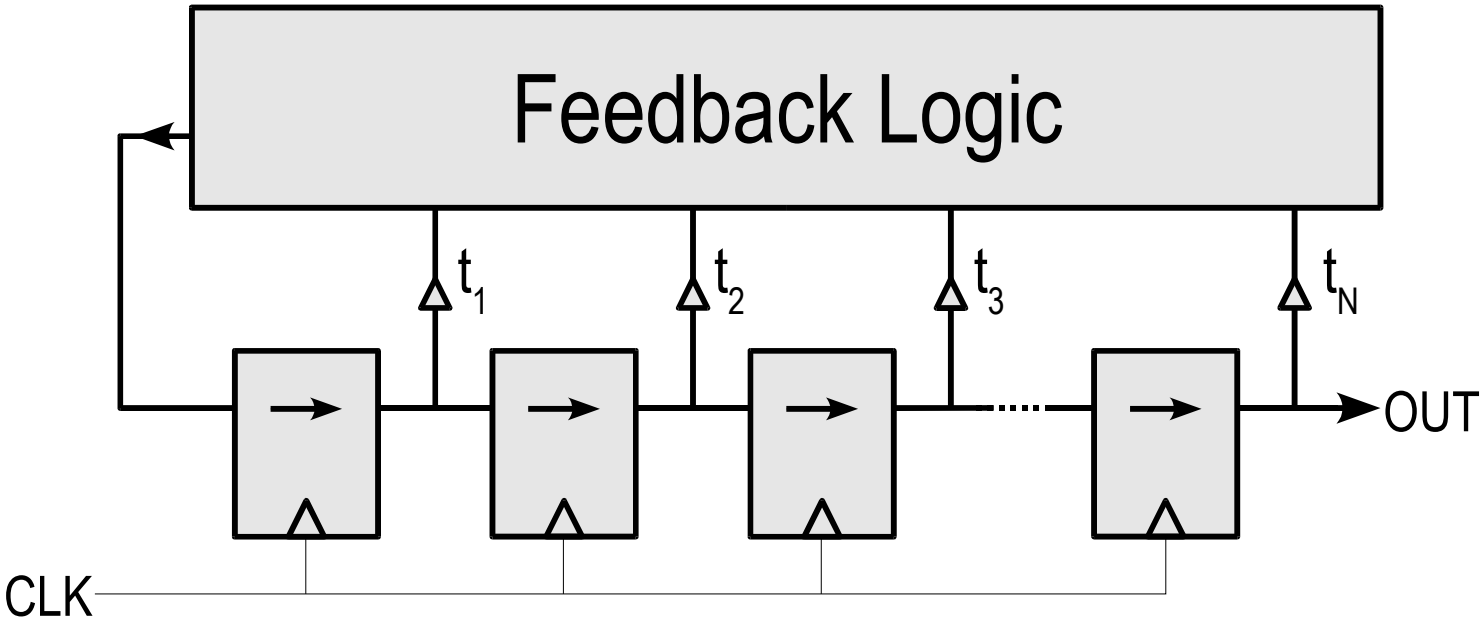
Pseudorandom Bit Sequence (PRBS)

- A statistically “random” bit sequence
  - Behaves like noise
  - Carries an equal number of 1's and 0's
  - Has near zero correlation with another PRBS
- Useful for
  - Testing I/O channels
  - Cryptography
  - Spread-spectrum communication systems

Generating a PRBS

Using a Linear Feedback Shift Register (LFSR)

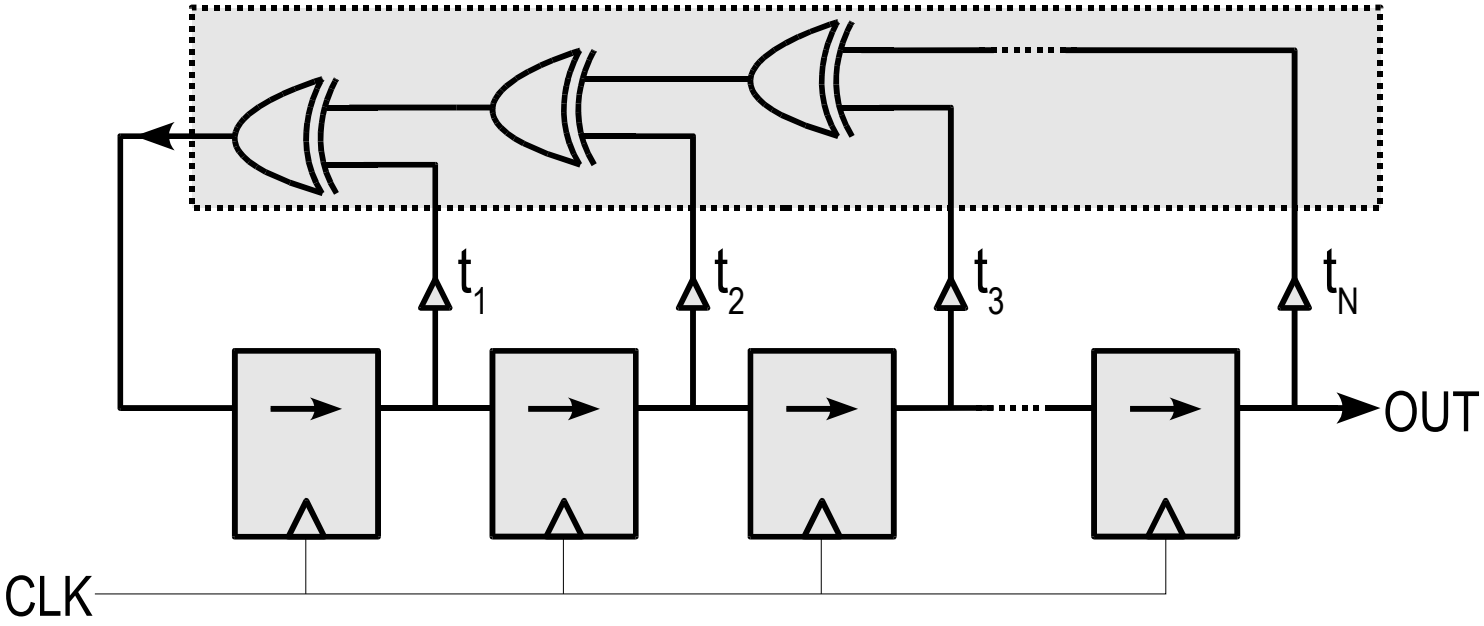
- Next input bit is a linear function of the current state



Generating a PRBS

Using a Linear Feedback Shift Register (LFSR)

- Next input bit is a linear function of the current state
- Feedback logic is a sum (XOR) of a subset of bits in the register



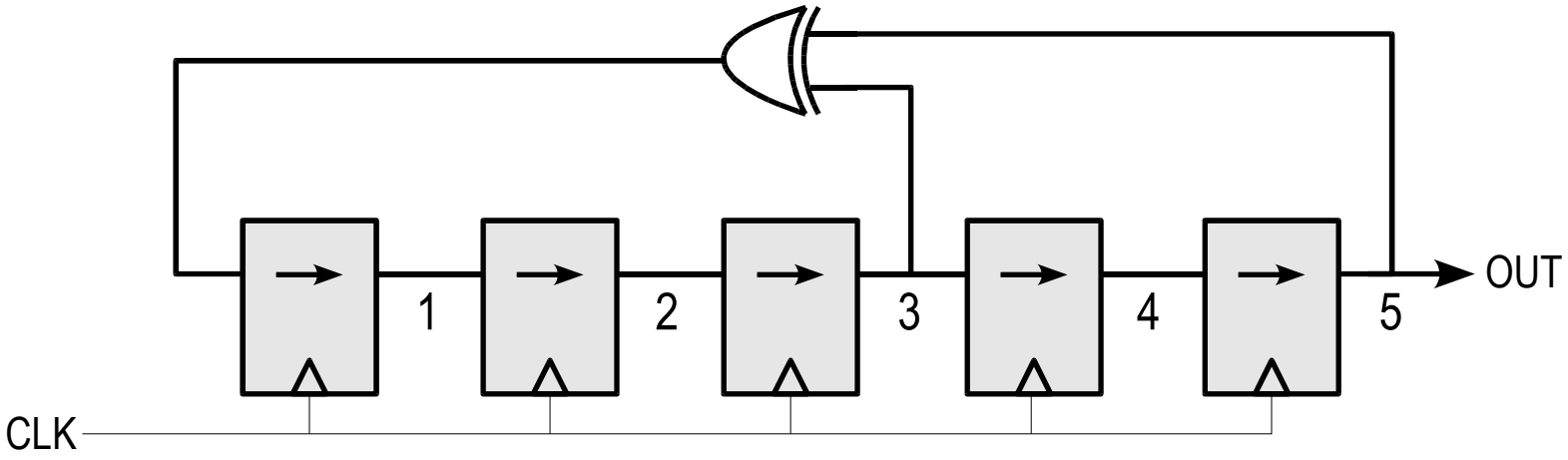
Feedback Tap Sequence

- Not every LFSR can generate a PRBS
  - Only ones with 'maximal' tap sequences do
  - We find these using finite-field mathematics
- A 'maximal' LFSR generates a PRBS known as a maximal-length sequence
  - An *N*-stage LFSR can produce a PRBS of length $2^N - 1$
  - The PRBS contains every *N*-bit pattern (except all 0's)

Feedback Tap Sequence

Notation

- We list the positions of all tapped bits
- Example: 5-stage LFSR with taps at positions 3, 5
  - > Notation: $N = 5$, Taps = [5,3]
  - > Produces a PRBS with length $2^5 - 1 = 31$



Exclusivity of an LFSR

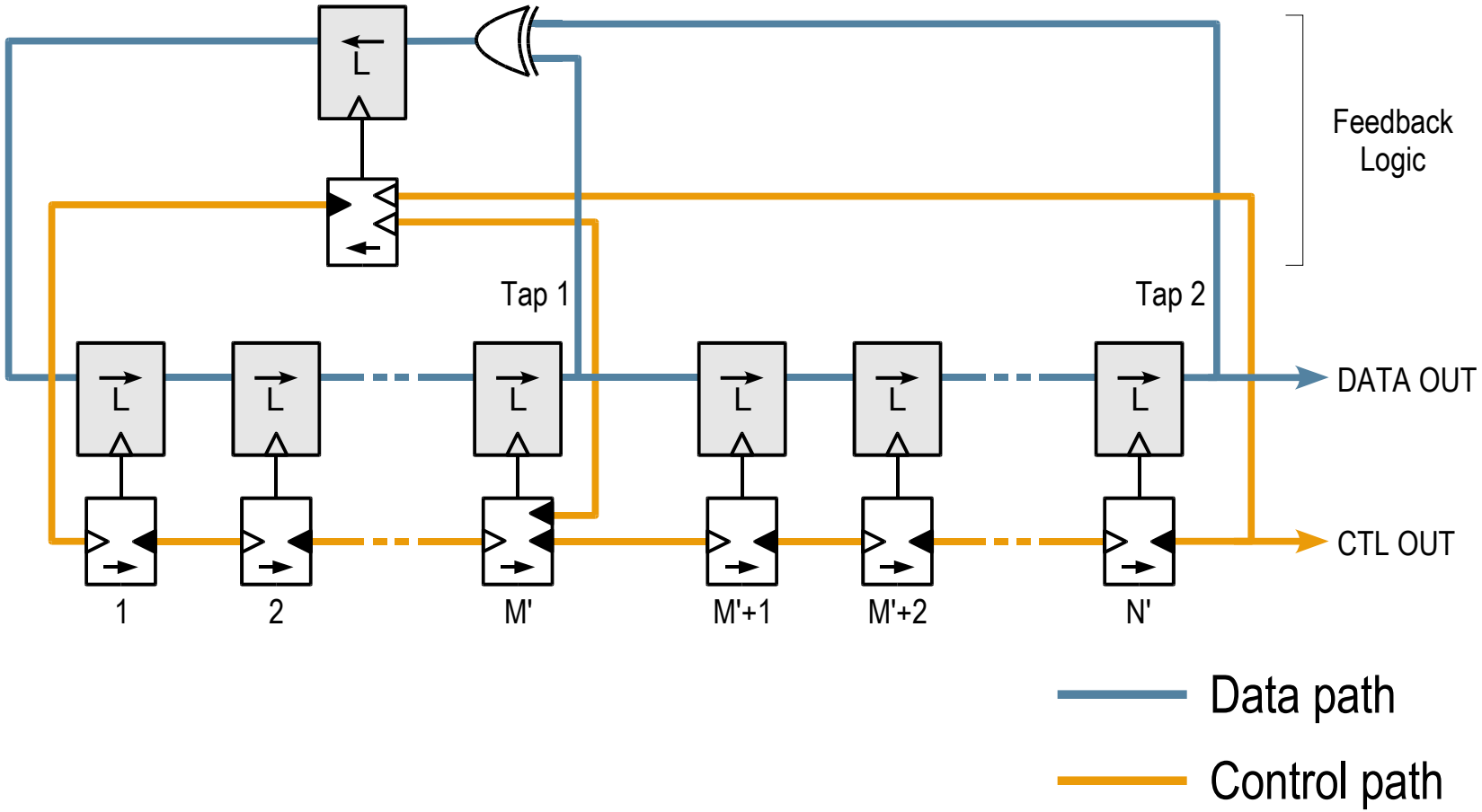
- Properties of the PRBS depend only on the LFSR structure, i.e.
  - Number of LFSR stages
  - Tap sequence (location and number of taps)
- Each LFSR circuit can produce only one PRBS

An Asynchronous PRBS Generator

- Motivation: logic flexibility
  - One circuit can produce many PRBS's
- Replace shift register with asynchronous FIFO rings
- We present a GasP implementation
  - Any other asynchronous control element will do

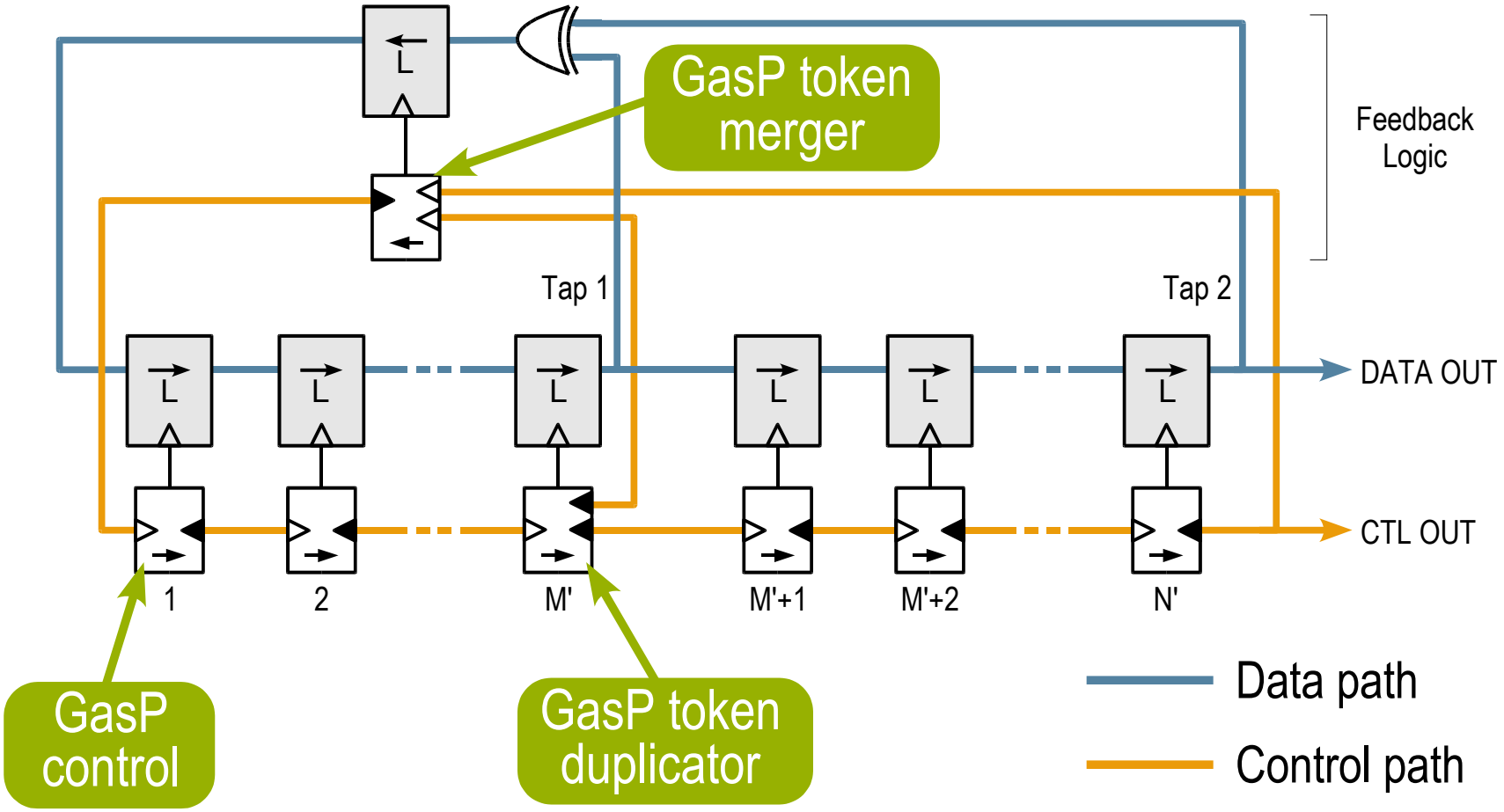
Two-Tap FIFO PRBS Generator

The Circuit



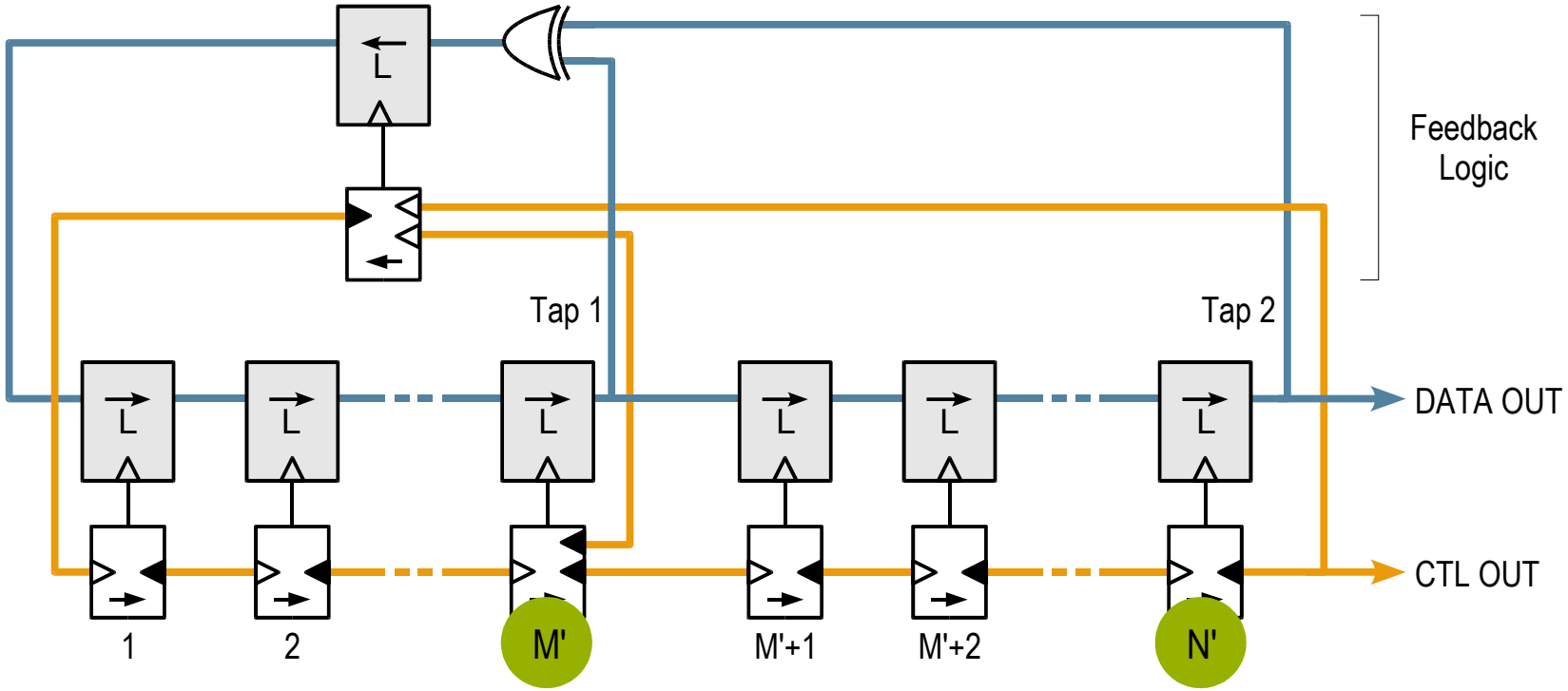
Two-Tap FIFO PRBS Generator

GasP Control Elements



Two-Tap FIFO PRBS Generator

Tap Notation



- Separate physical taps [*N'*,*M'*] from logical taps [*N*,*M*]

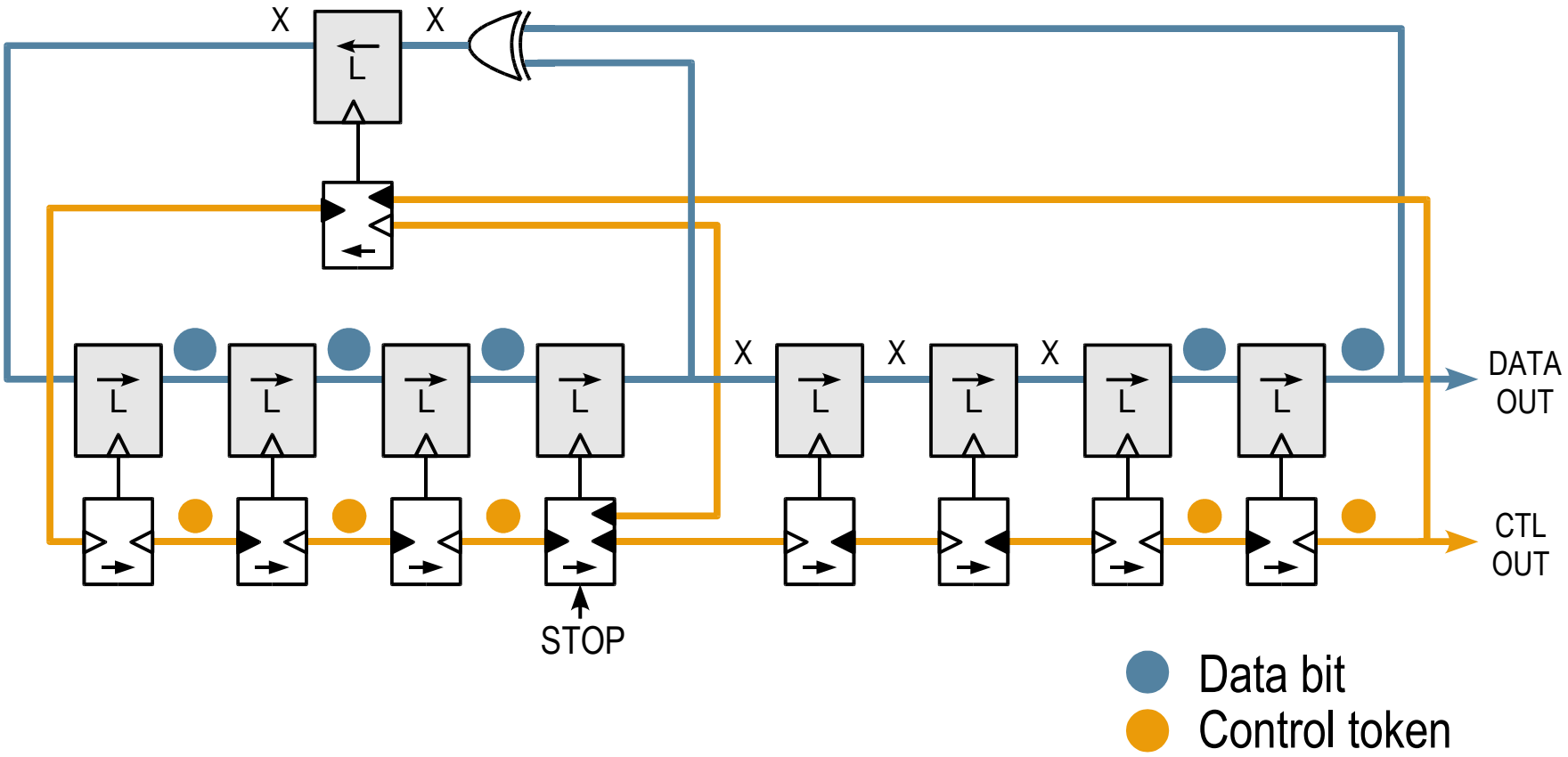
Maintaining Proper Bit Sequence

- Must enforce feedback computation formula
  - XOR must operate on bits specified by tap sequence
- Automatic for LFSR
  - Circuit hardware enforces formula by design
- Need explicit management for async FIFO
  - Not every stage contains valid data
  - Data bits may propagate autonomously
  - Must explicitly synchronize pairs of control tokens to guarantee correct data sequencing

Maintaining Proper Bit Sequence

Example

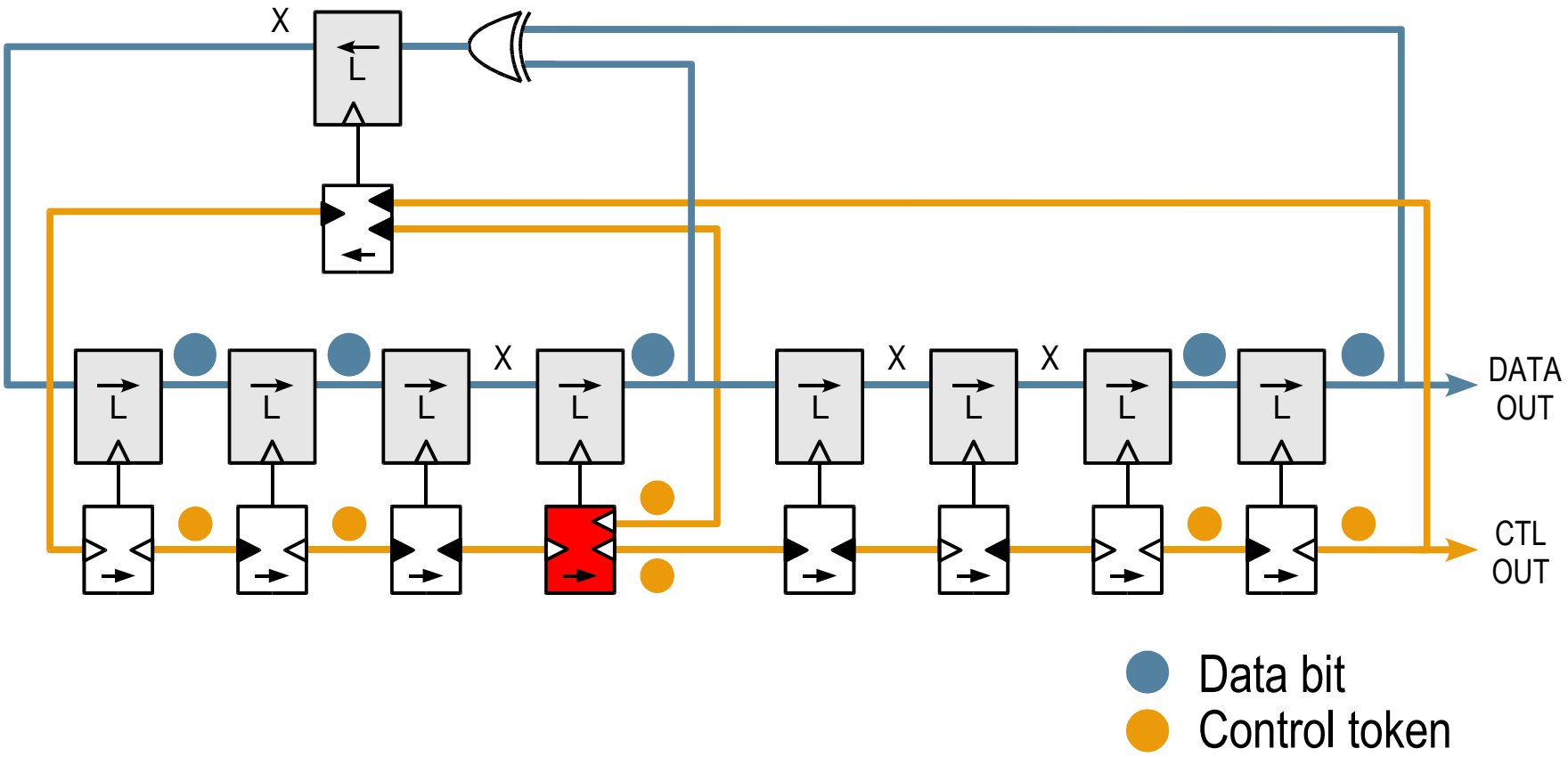
- Emulate 5-stage LFSR with taps = [5,3]



Maintaining Proper Bit Sequence

Example

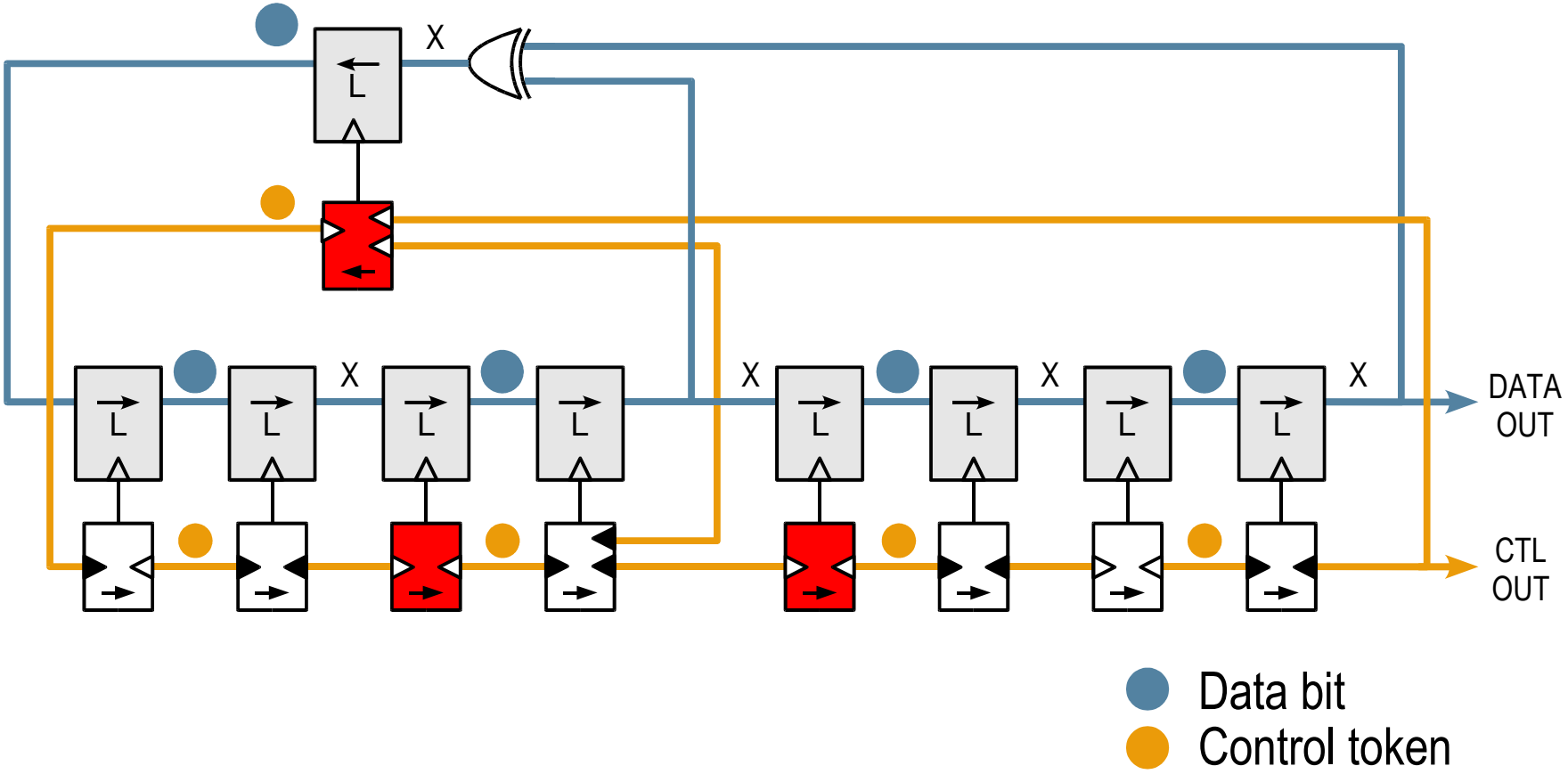
- Emulate 5-stage LFSR with taps = [5,3]



Maintaining Proper Bit Sequence

Example

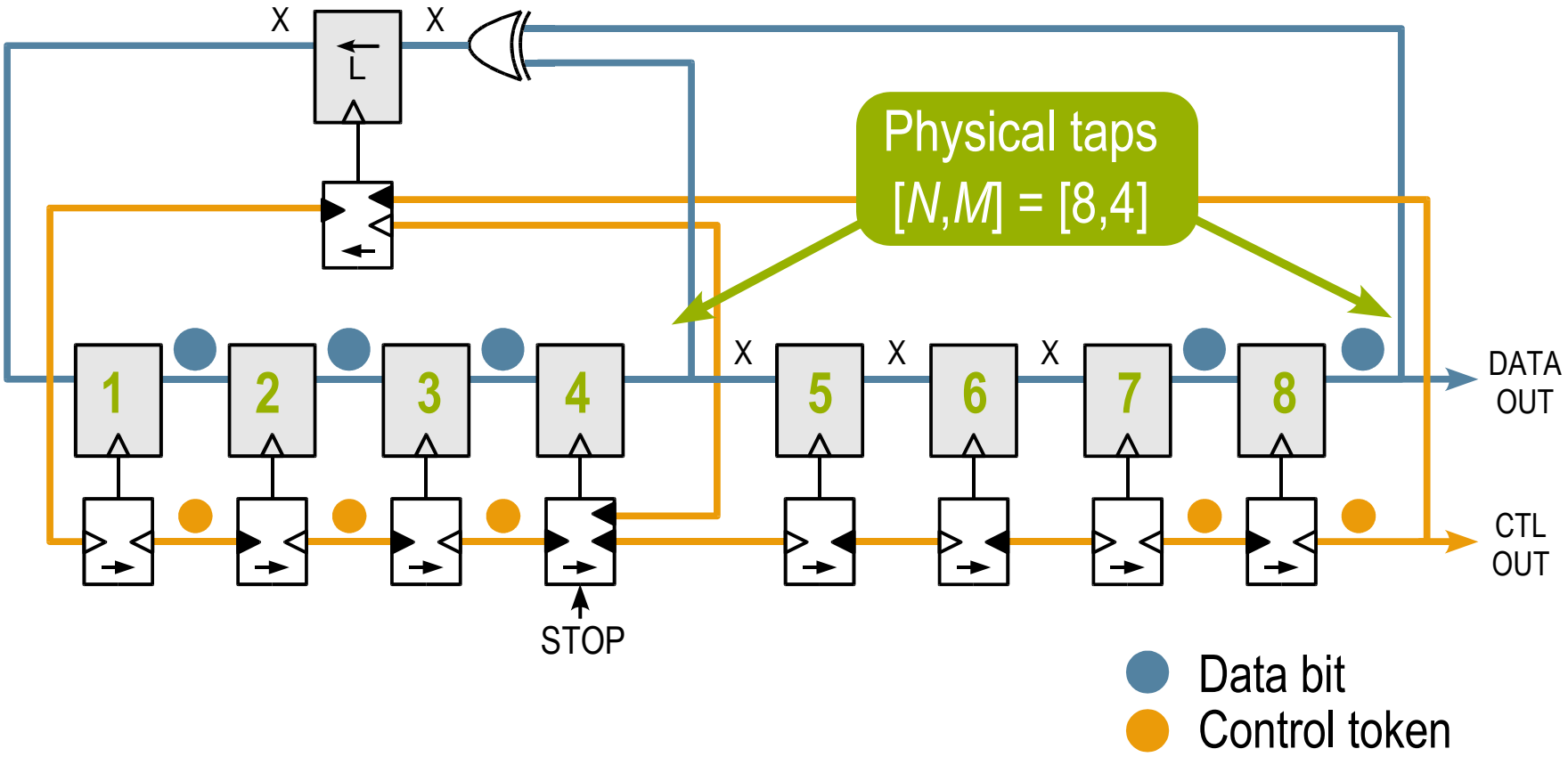
- Emulate 5-stage LFSR with taps = [5,3]



Maintaining Proper Bit Sequence

Example

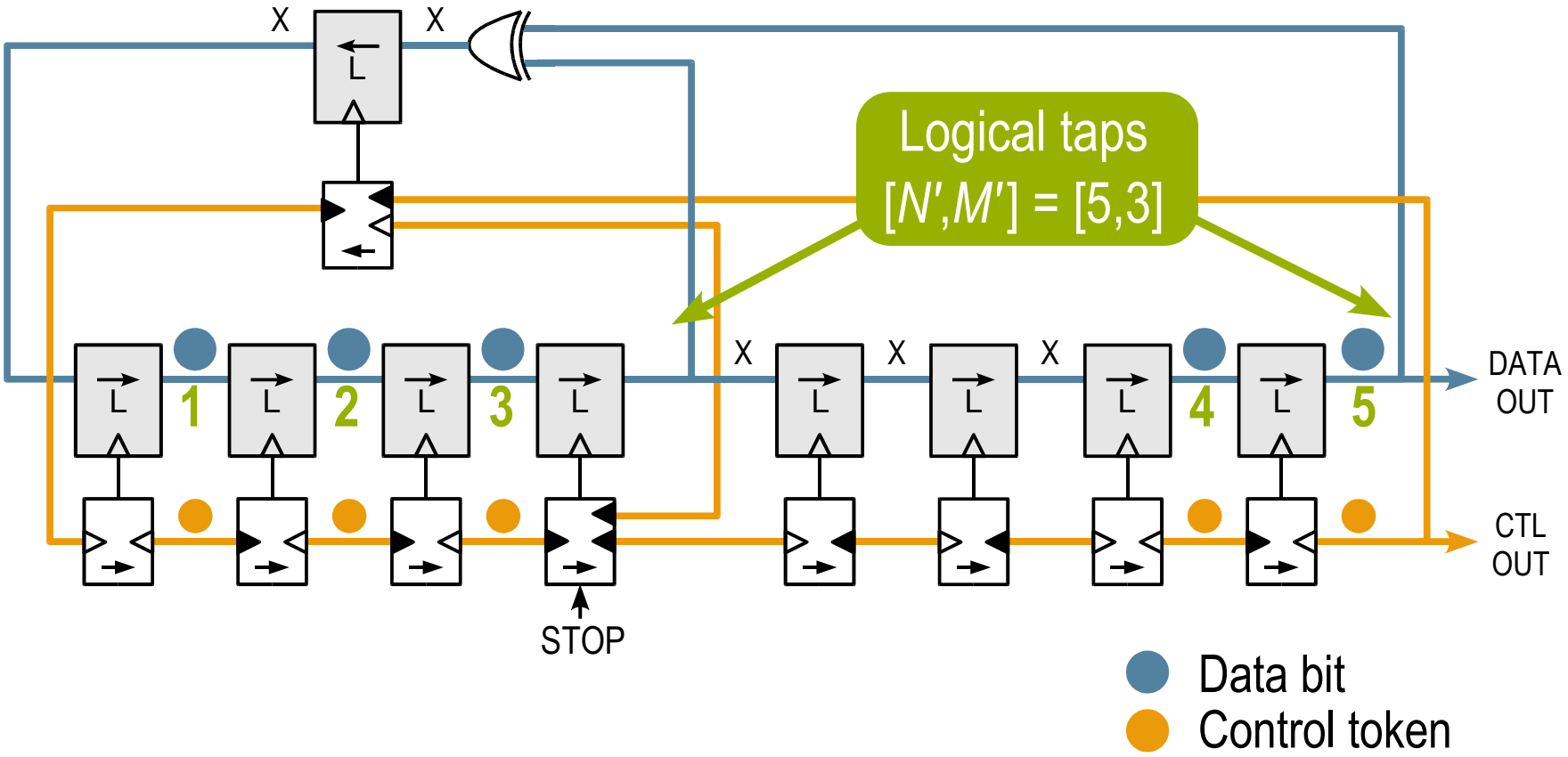
- Emulate 5-stage LFSR with taps = [5,3]



Maintaining Proper Bit Sequence

Example

- Emulate 5-stage LFSR with taps = [5,3]



Physical & Logical Correspondence

- Correspondence for LFSR is strict
  - Physical tap sequence = logical tap sequence
- Correspondence for async FIFO is soft
  - Logical tap sequence depends only on token distribution
  - Physical circuit structure need only enable logical tap sequence, not match it

Generality of an Async FIFO

- Async logic decouples logical bit distribution from physical circuit structure
- Different FIFO ring configurations can produce the same PRBS
- One FIFO can produce many different PRBS's

Generating Different PRBS's

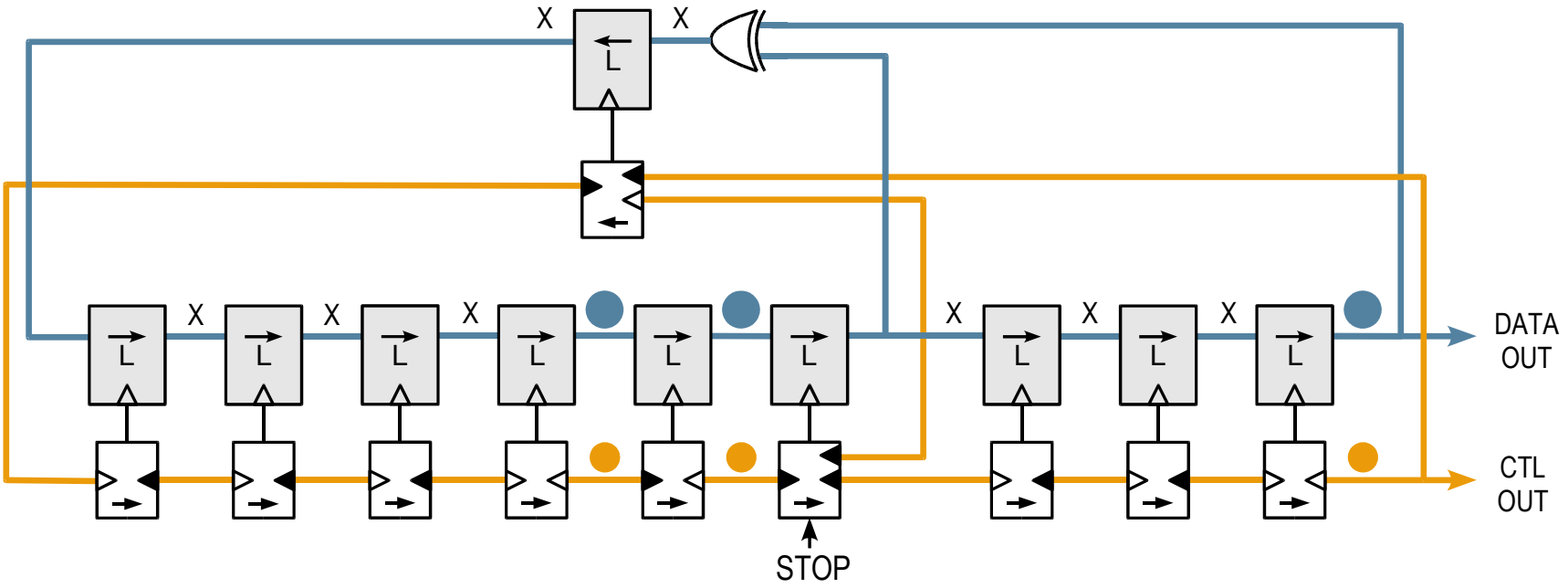
Example

- Construct a circuit to generate all two-tap maximal-length patterns of order $N \leq 7$

Order *N*	Logical Taps [*N*,*M*]	Min # Stages 1	Min # Stages 2
3	[3,2]	2	1
4	[4,3]	3	1
5	[5,3]	3	2
6	[6,5]	5	1
7	[7,4]	4	3
7	[7,6]	6	1

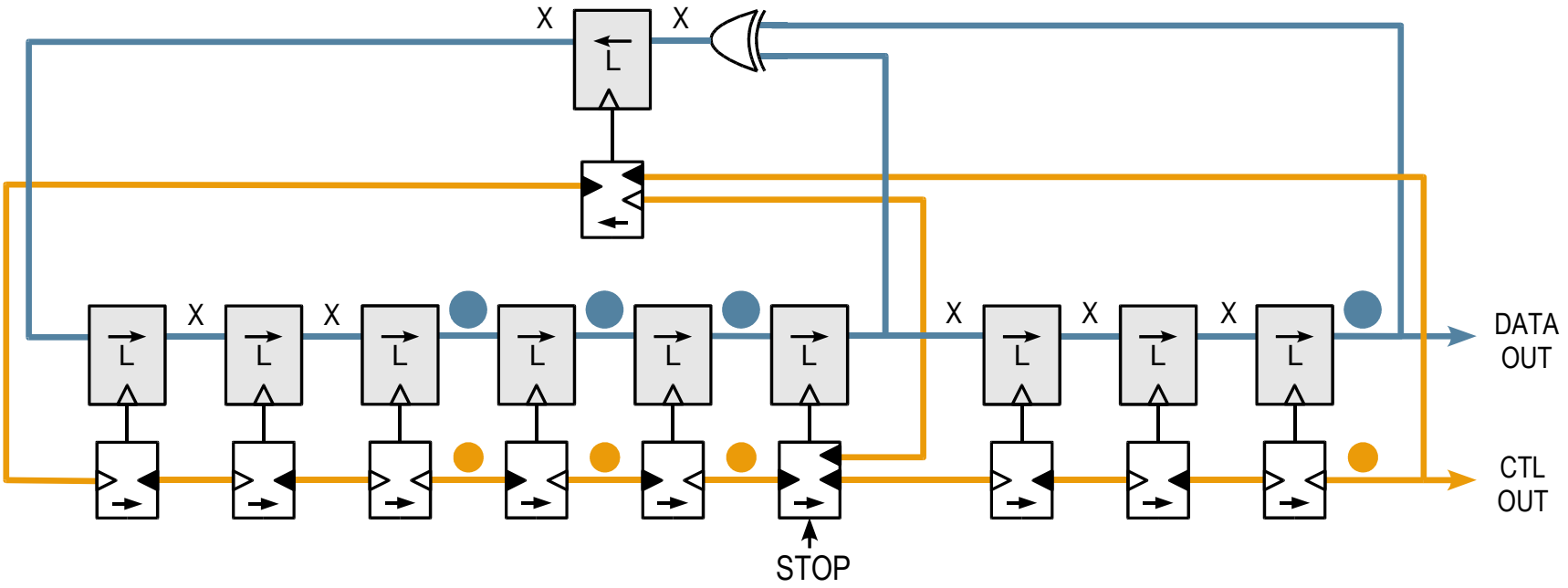
Generating All Patterns for $N \leq 7$

- N = 3
- Taps = [3,2]



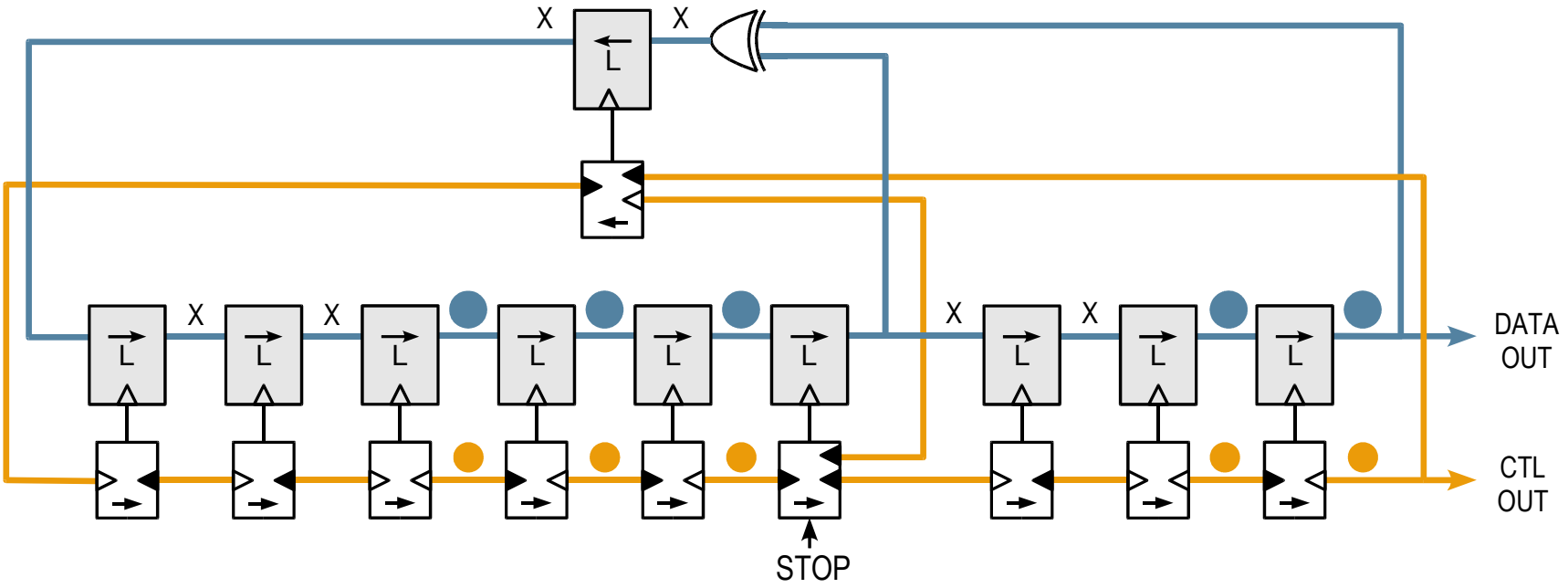
Generating All Patterns for $N \leq 7$

- N = 4
- Taps = [4,3]



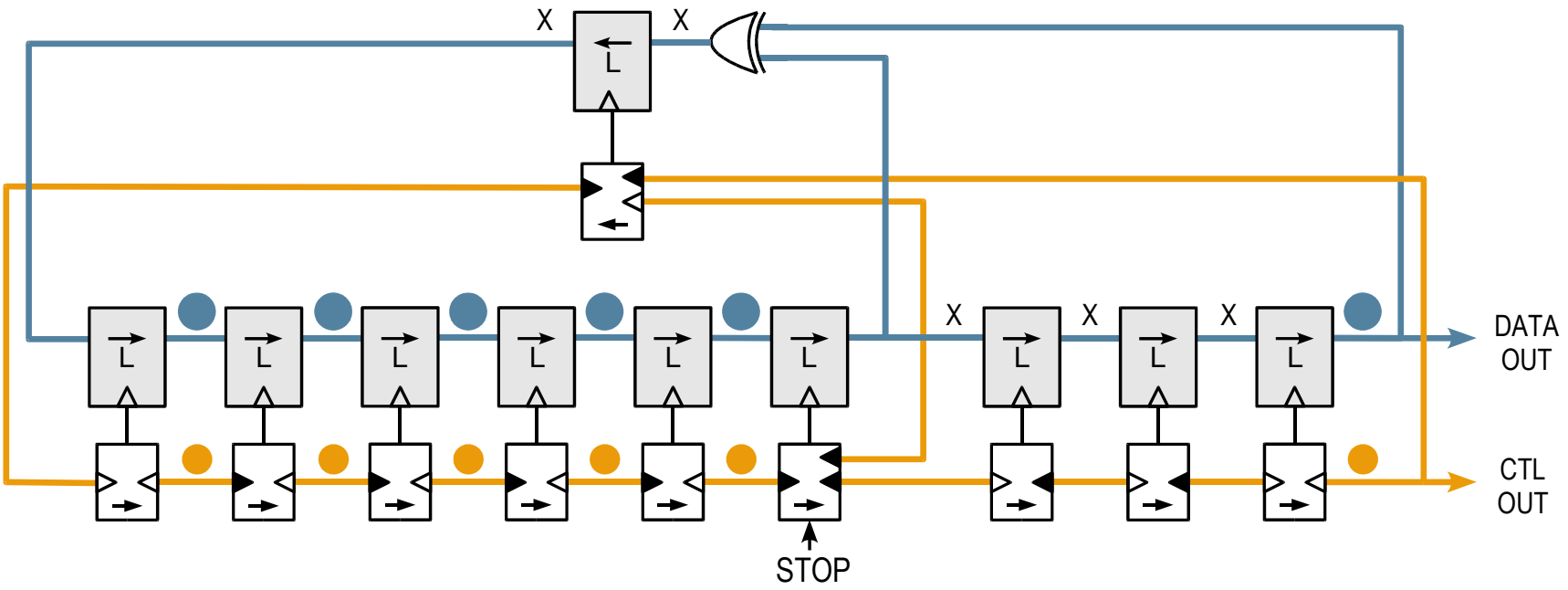
Generating All Patterns for $N \leq 7$

- N = 5
- Taps = [5,3]



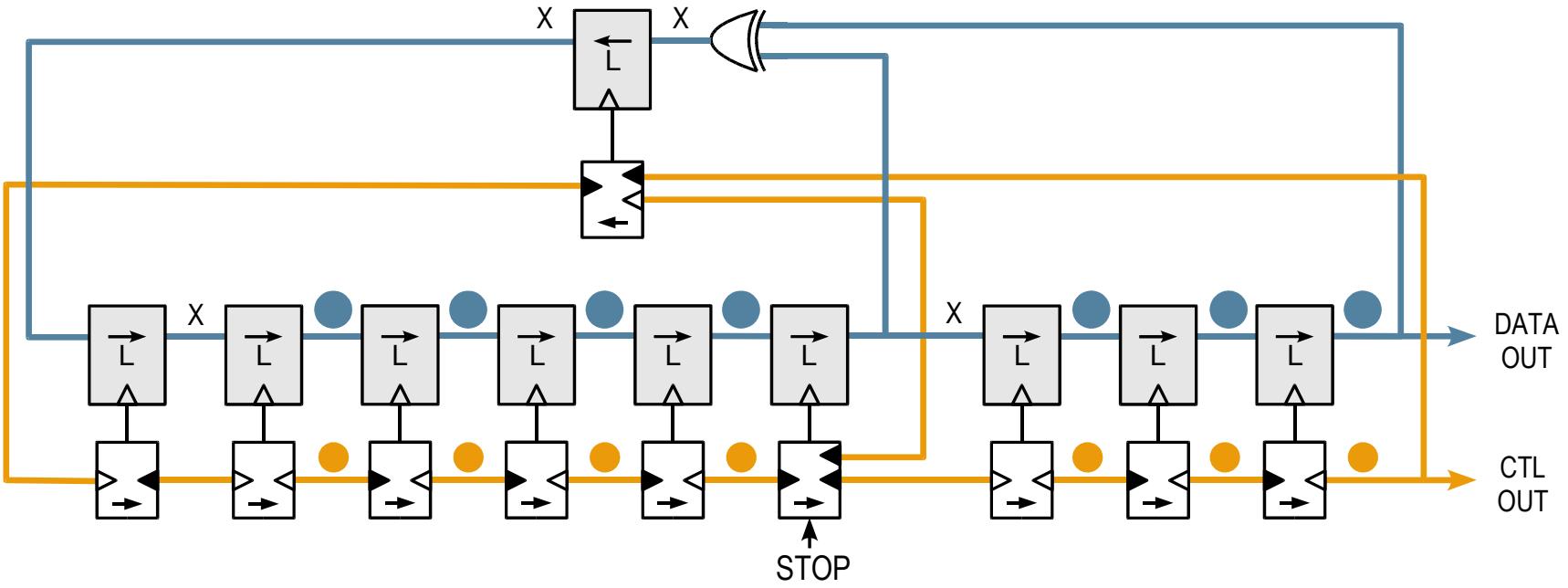
Generating All Patterns for $N \leq 7$

- N = 6
- Taps = [6,5]



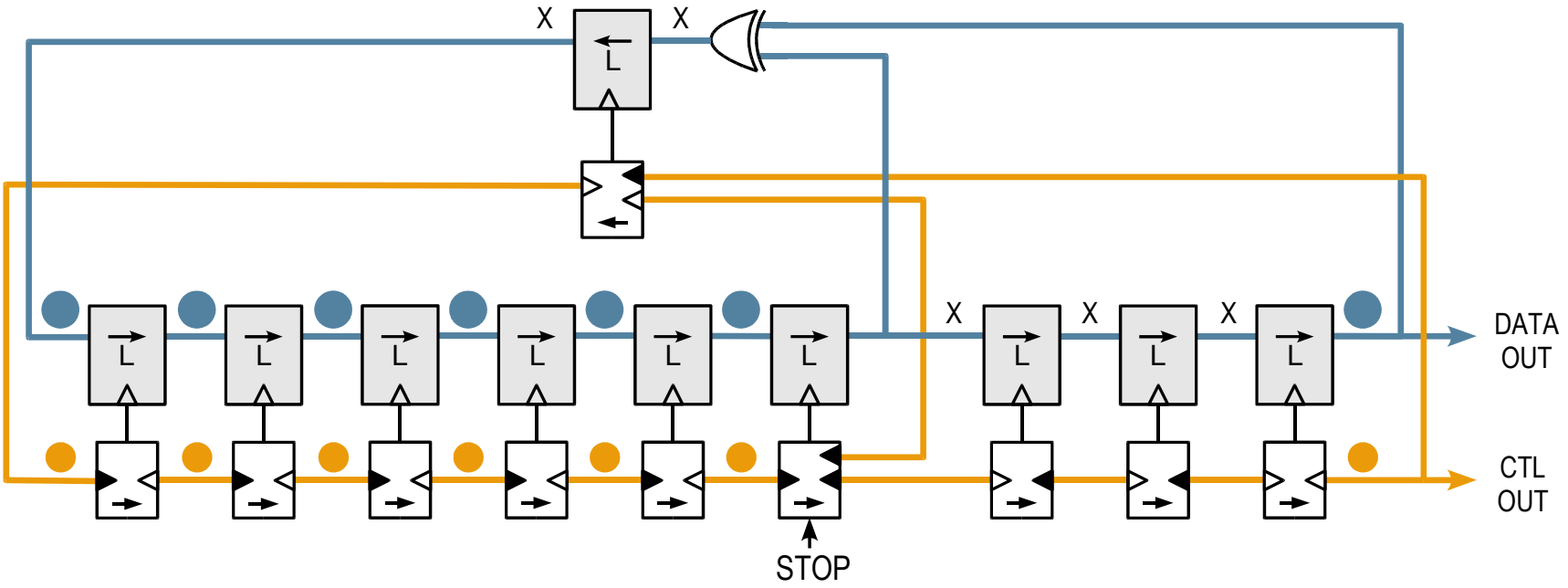
Generating All Patterns for $N \leq 7$

- N = 7
- Taps = [7,4]



Generating All Patterns for $N \leq 7$

- N = 7
- Taps = [7,6]



Throughput and I/O

- Throughput varies for different patterns
  - Because token occupancy is different for each pattern
- Use the generator synchronously
  - Take the output through async-to-clocked interfaces
  - Clock rate must be < minimum async throughput

Summary

- Async design decouples number and distribution of logical bits from physical shift register structure
- Exploit this orthogonality to let one FIFO circuit adopt multiple logical configurations
- PRBS generator serves as an illustrative example
  - > Can apply this idea to other circuits
  - > E.g. CRC, encoders, filters

Questions

Alex Chow
alex.chow@sun.com