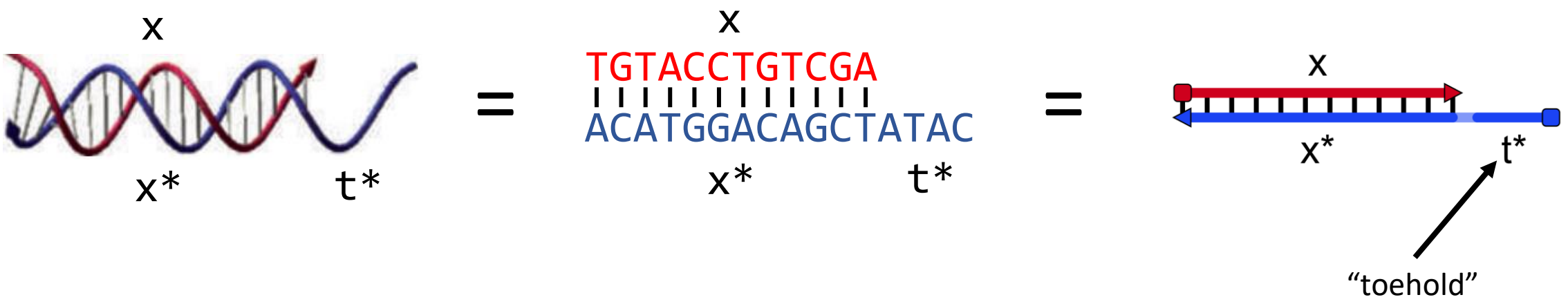
DNA strand displacement

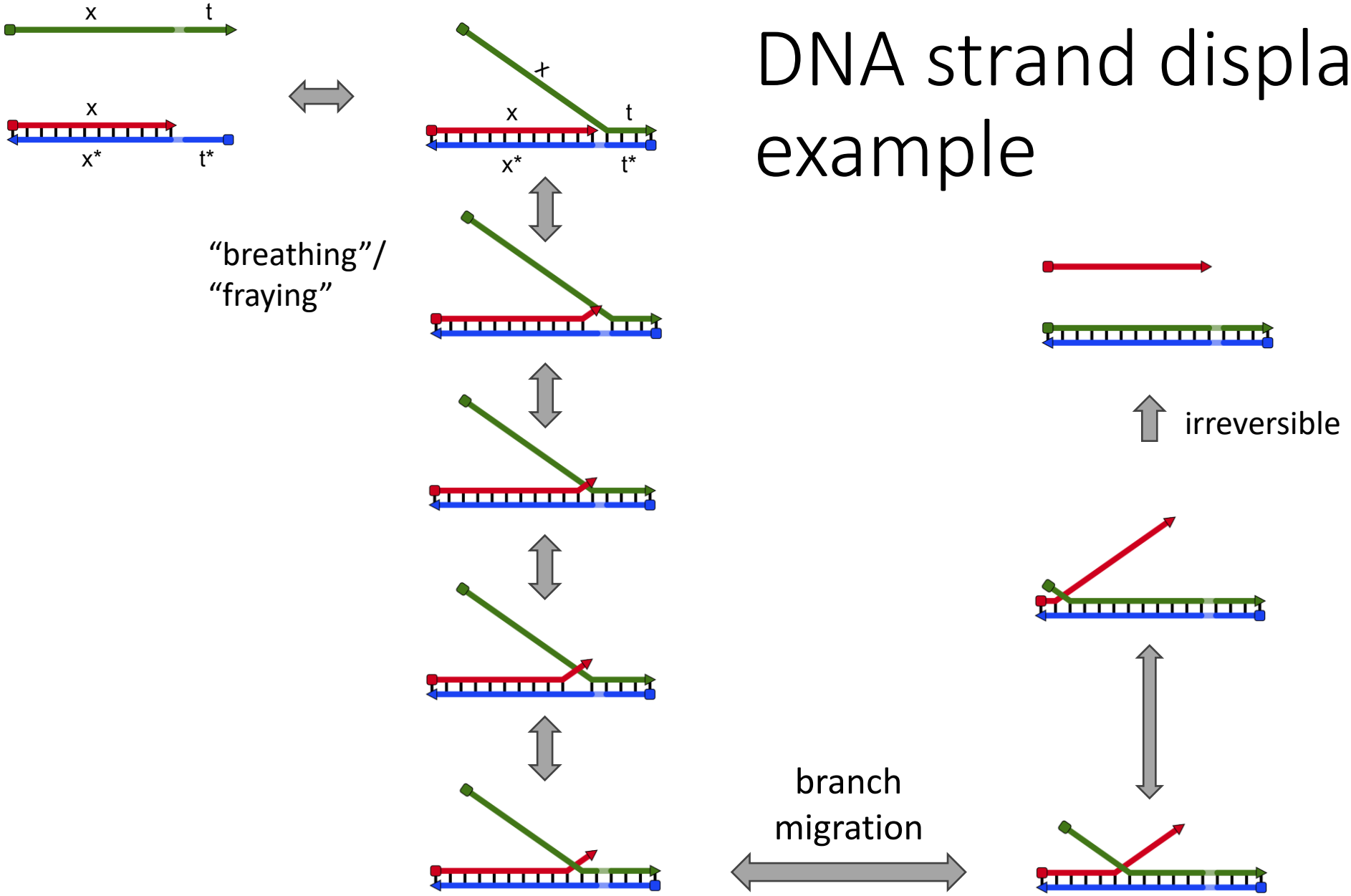
DNA reconfiguring itself without enzymes

ECS 232: Theory of Molecular Computation, UC Davis

DNA strands with “long” and “short” (toehold) binding domains

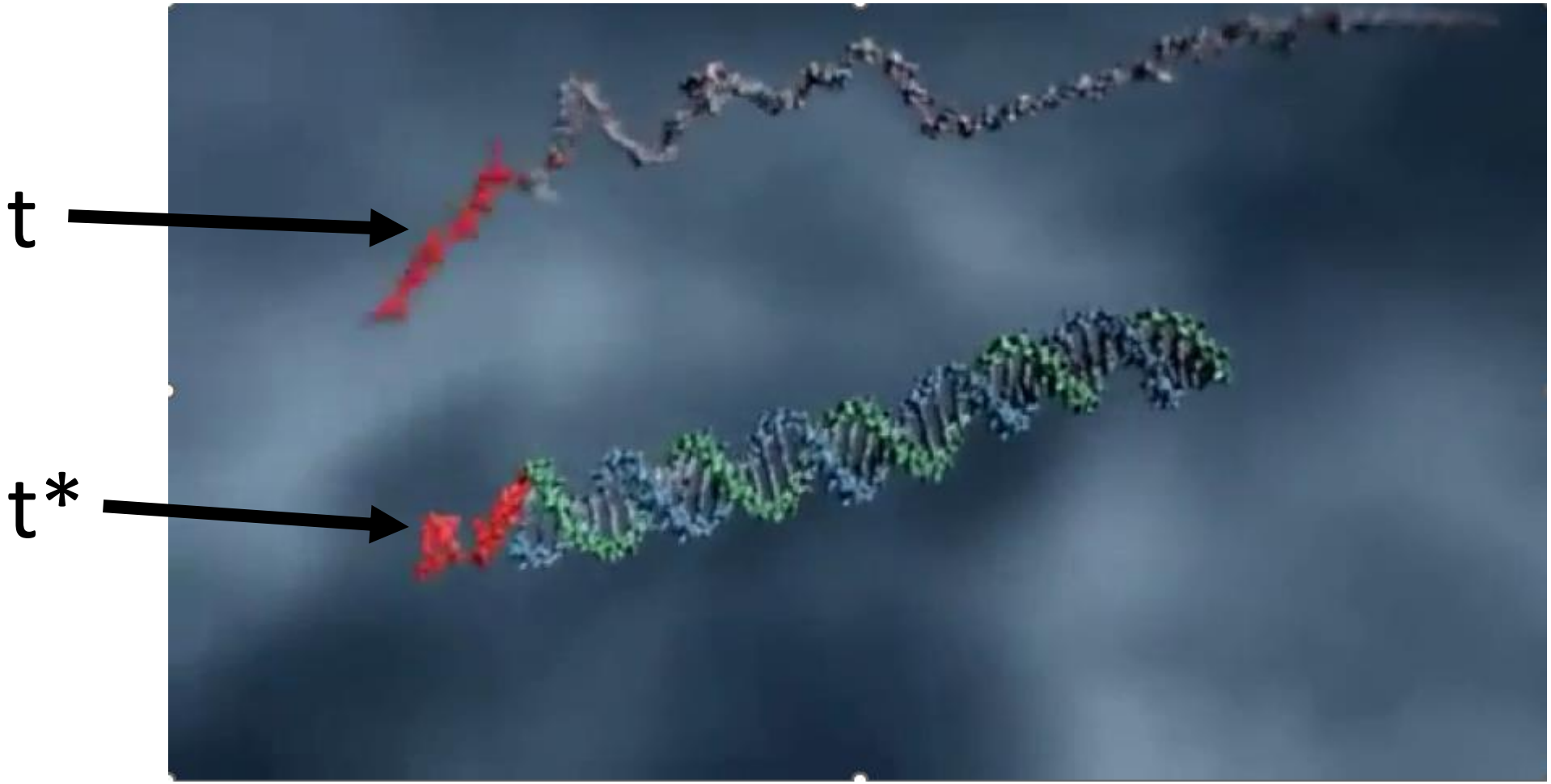


DNA strand displacement example



DNA strand displacement

https://www.microsoft.com/en-us/research/video/dna-strand-displacement/



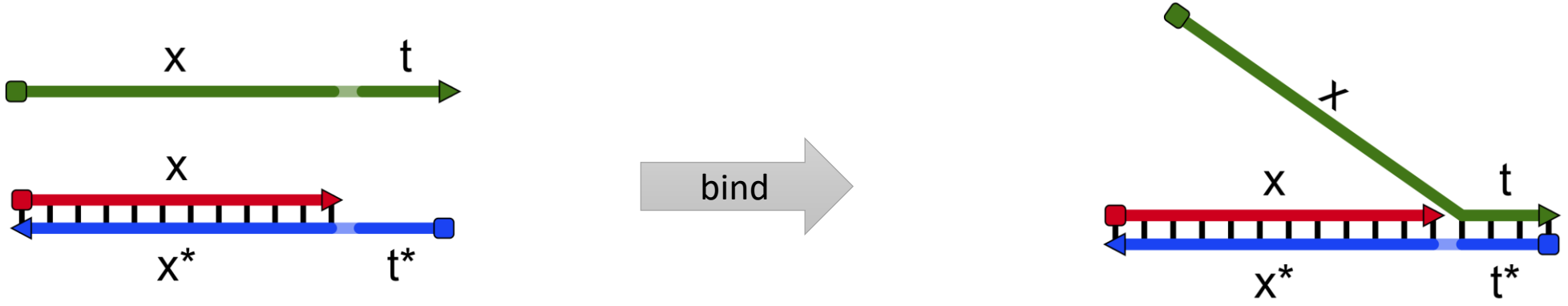
DNA strand displacement model

3 rules:

1. bind
2. release
3. displace

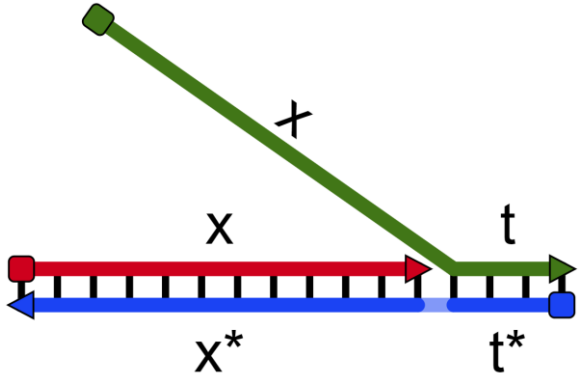
Bind rule

single-stranded complementary domains can bind

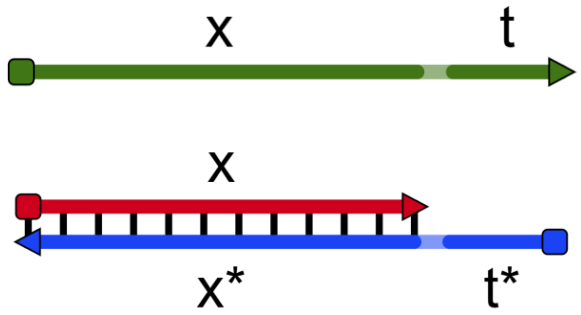


Release rule

double-stranded complementary domains can unbind
IF they are toehold-length (short, < 8 nt)

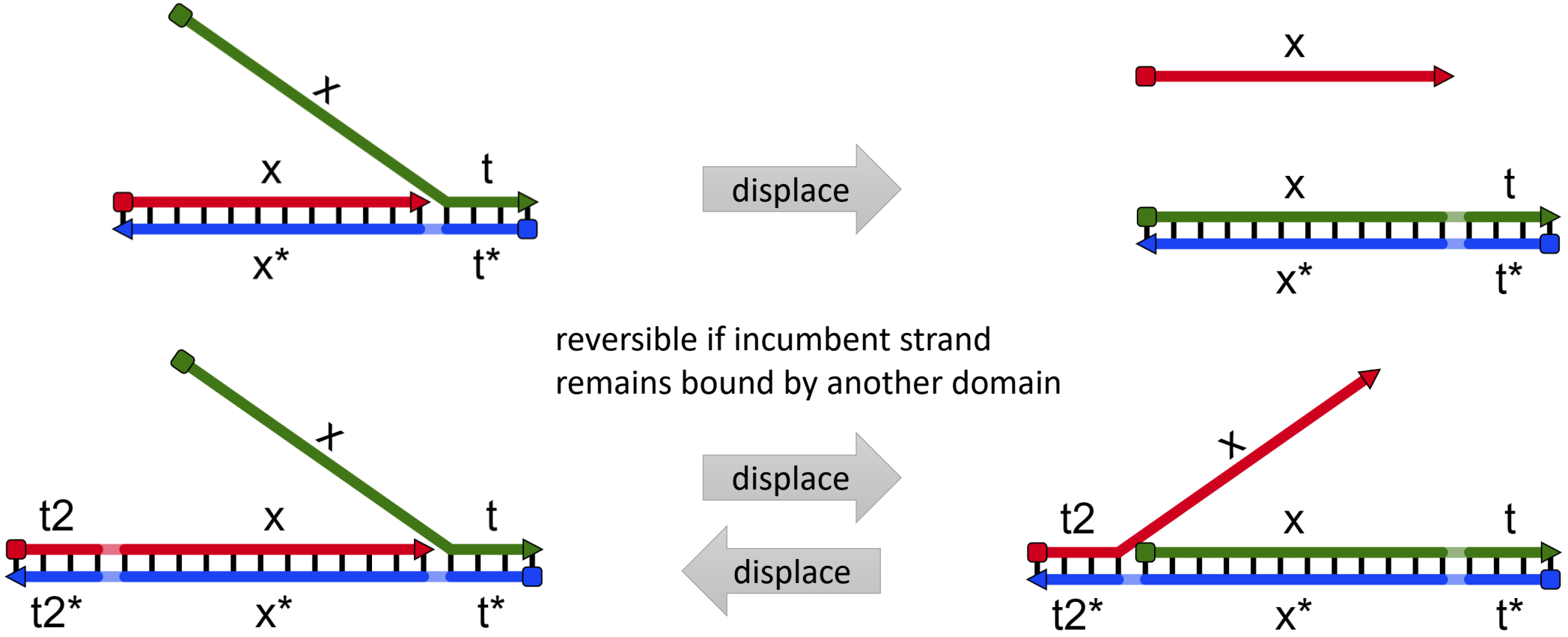


release



Displace rule

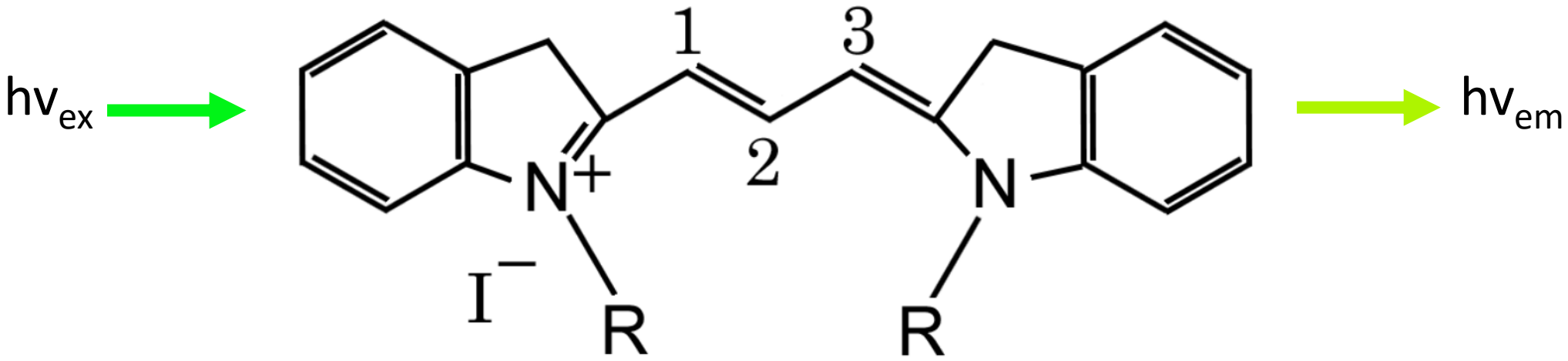
A domain (invader) can displace an identical domain (incumbent) of another strand,
IF neighboring domains are already bound



Readout

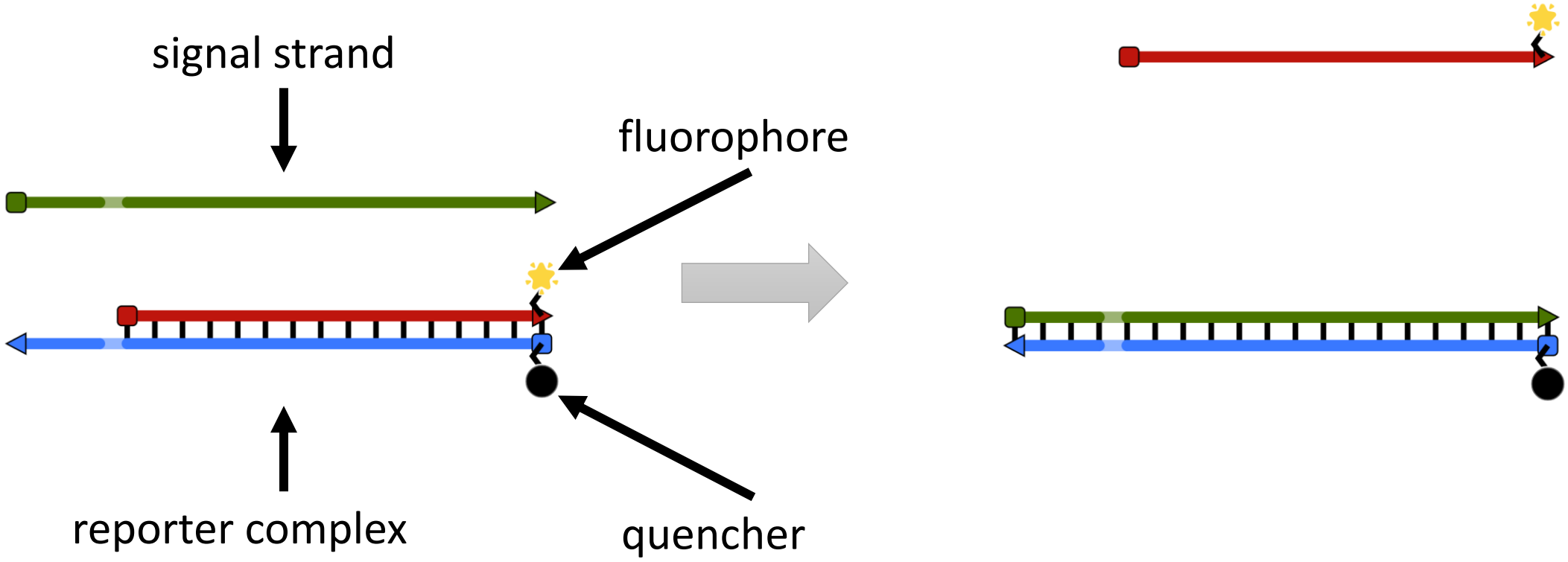
How do we read a “signal” in a DNA strand displacement system?

Fluorophores, when “excited” by light at one wavelength, emit light at a longer wavelength.

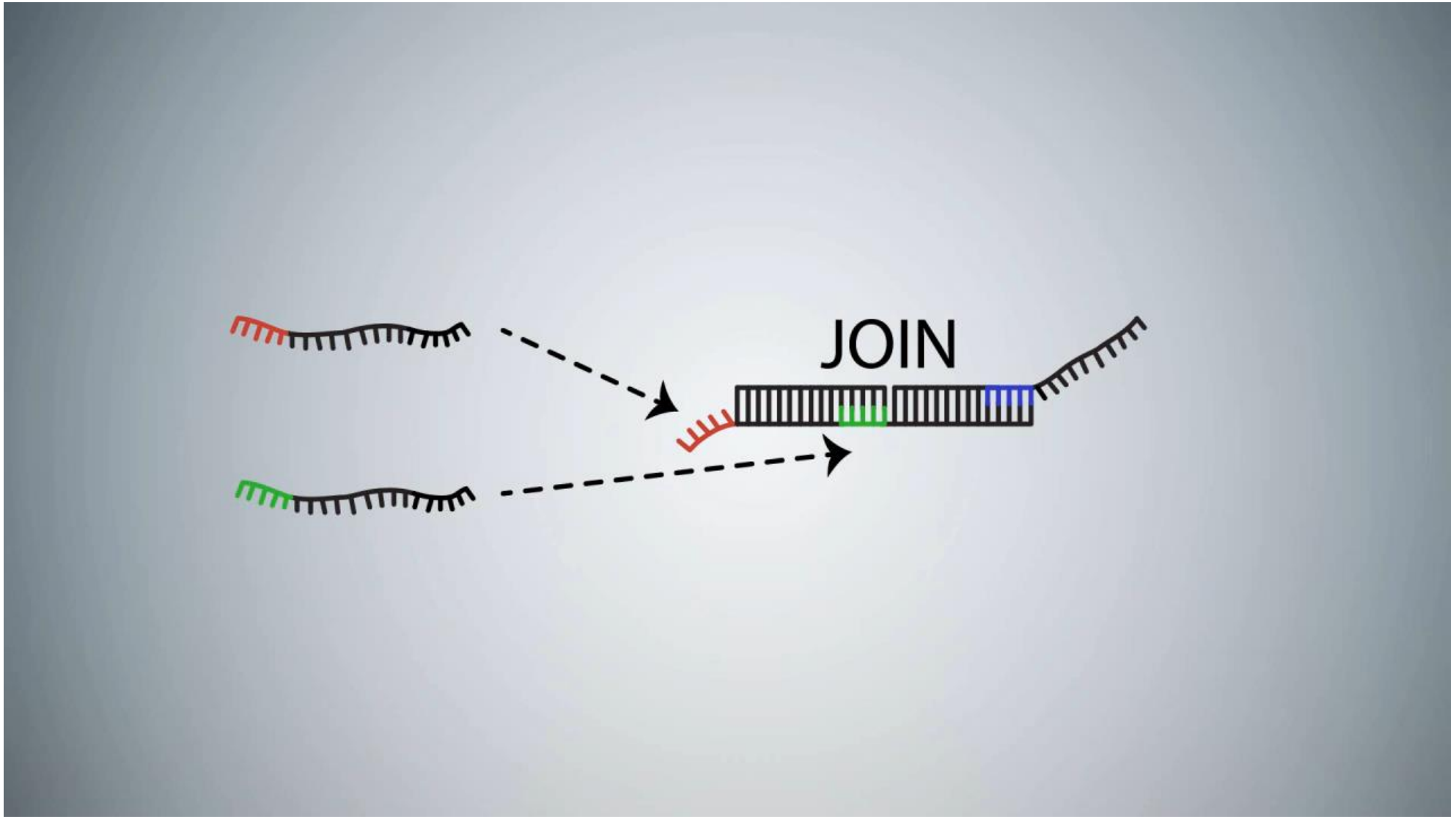


Reporter complexes

How do we read a “signal”?
“signal” = single strand is freed from a double-stranded complex.



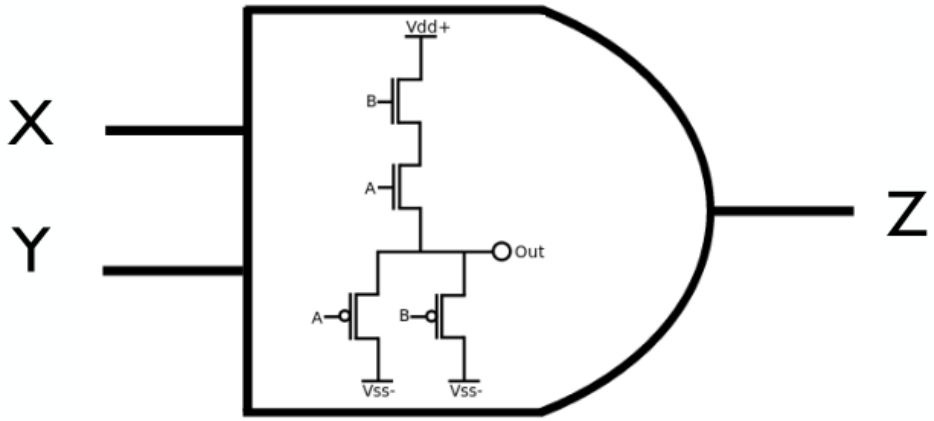
Reporter complex depiction



Boolean logic with DNA strand displacement

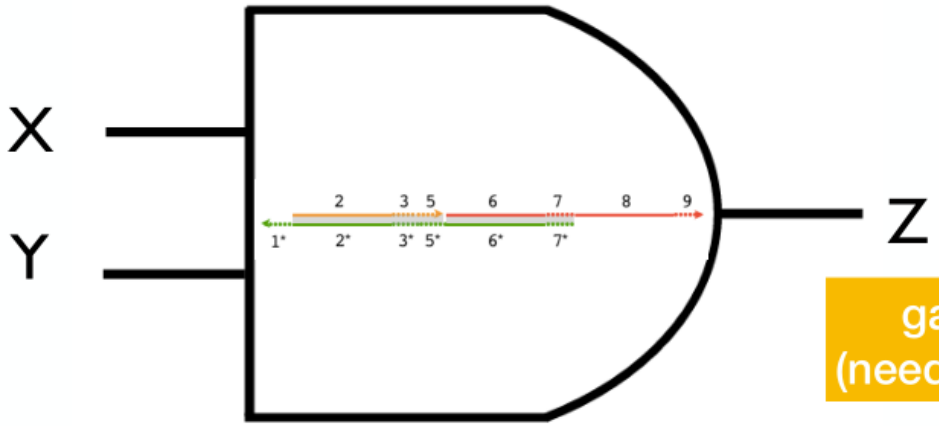
AND gate

voltages



release Z if and only if X and Y are present

strands



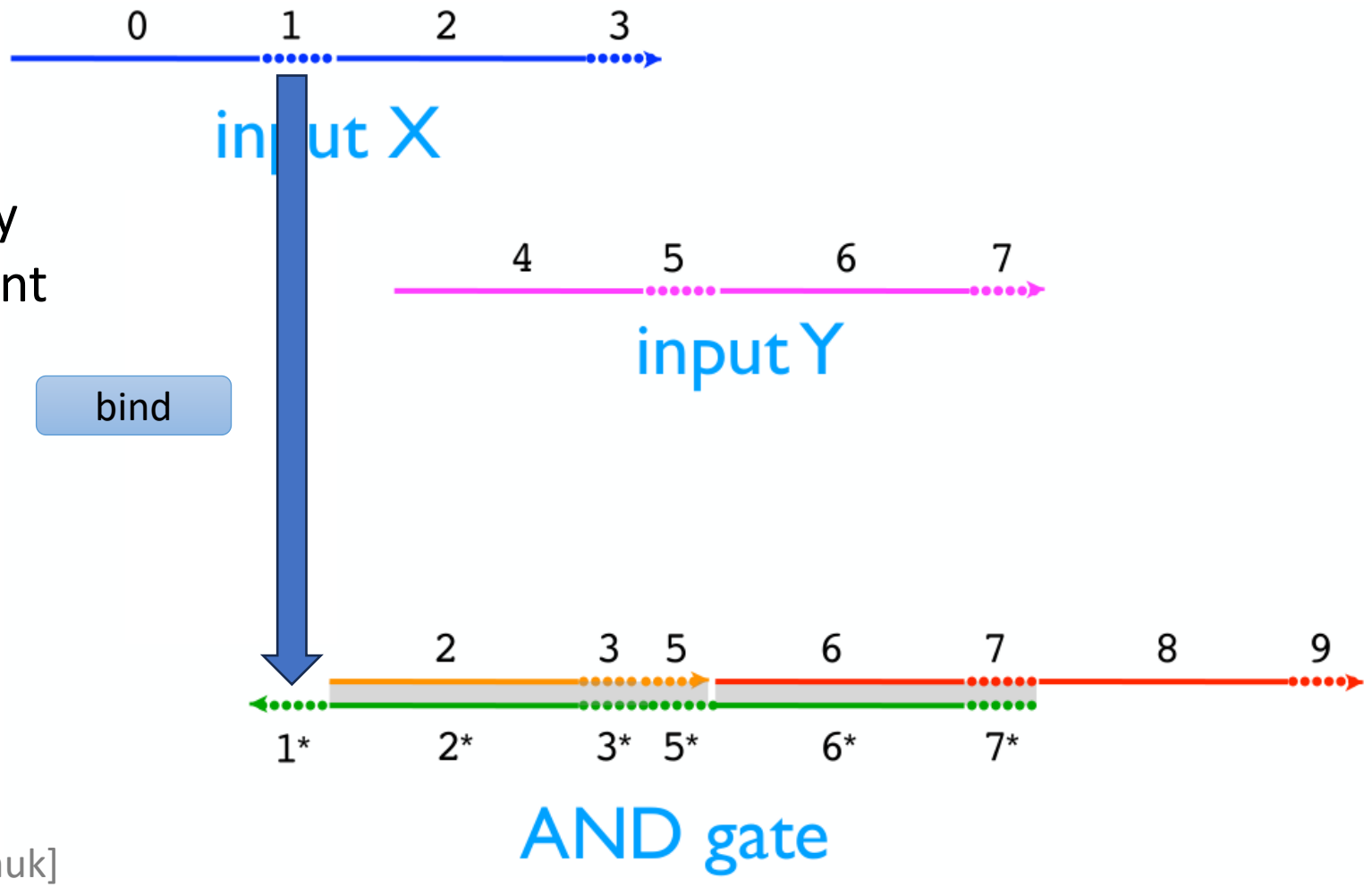
gates get consumed!
(need to have many copies)

gate=complex

[slides credit: Chris Thachuk]

Strand displacement cascade example: AND gate

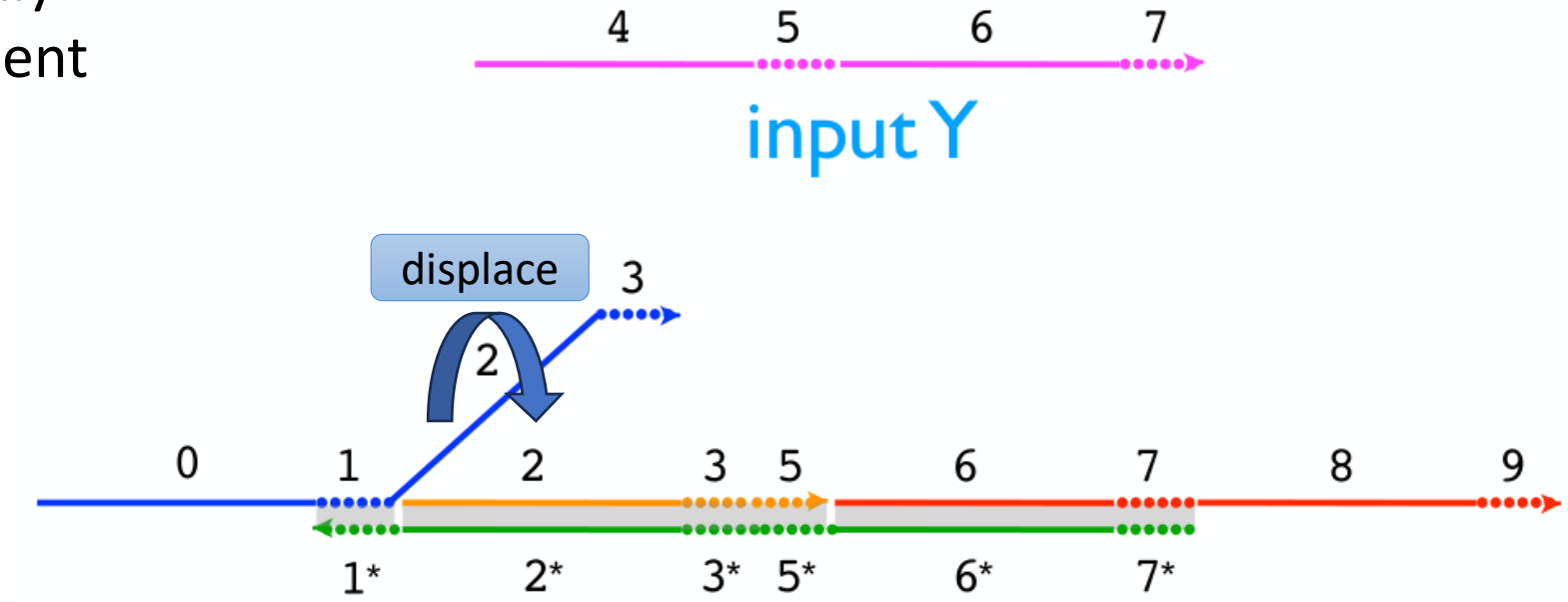
release Z if and only
if X and Y are present



[slides credit: Chris Thachuk]

Strand displacement cascade example: AND gate

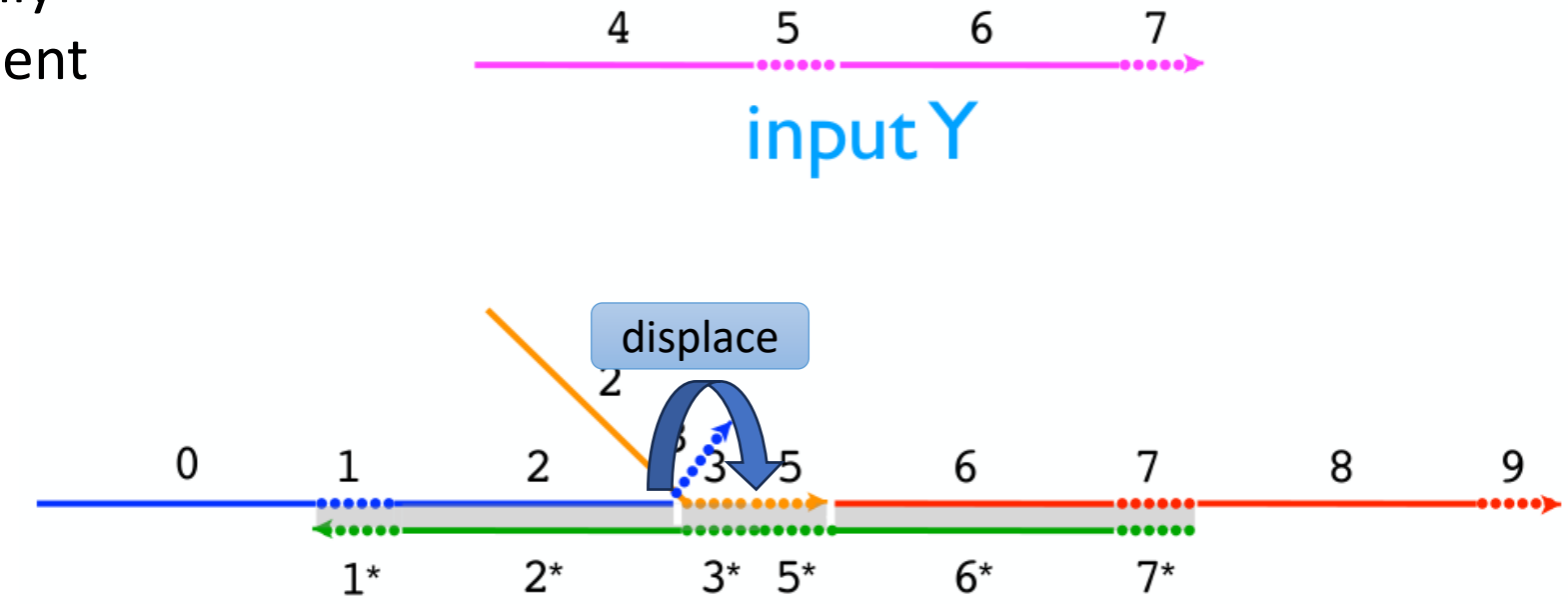
release Z if and only
if X and Y are present



[slides credit: Chris Thachuk]

Strand displacement cascade example: AND gate

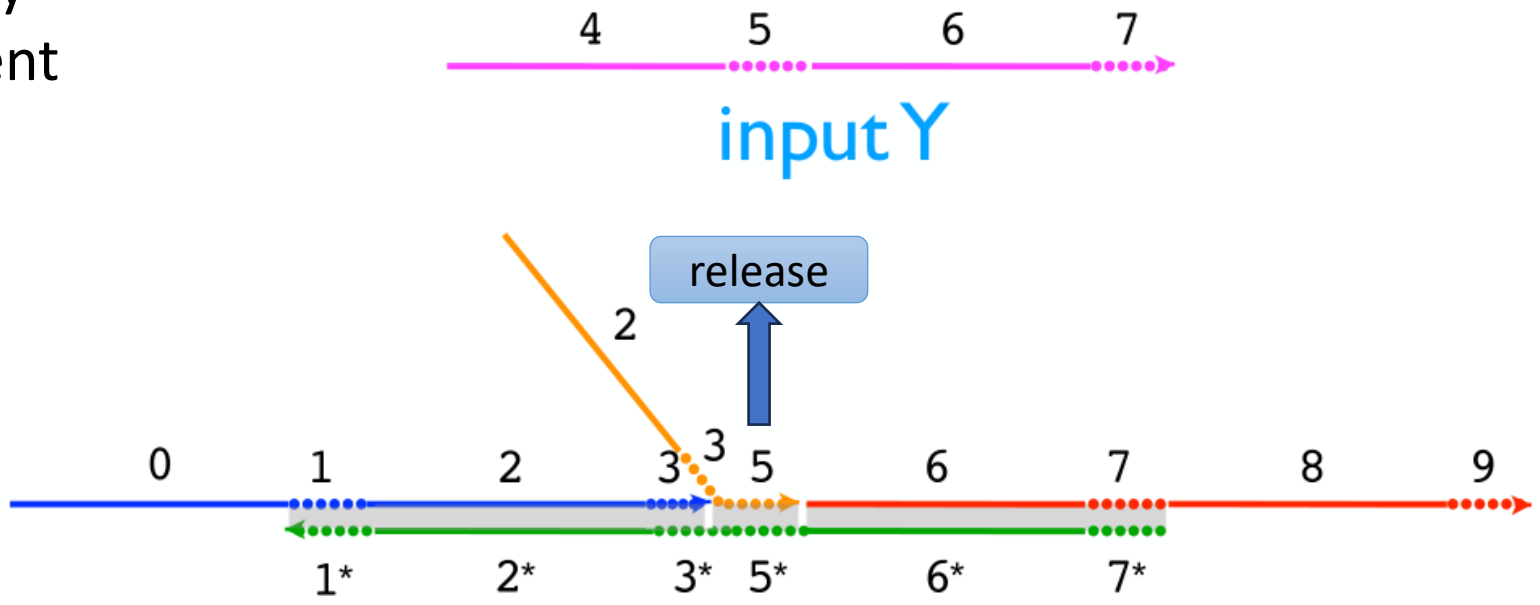
release Z if and only
if X and Y are present



[slides credit: Chris Thachuk]

Strand displacement cascade example: AND gate

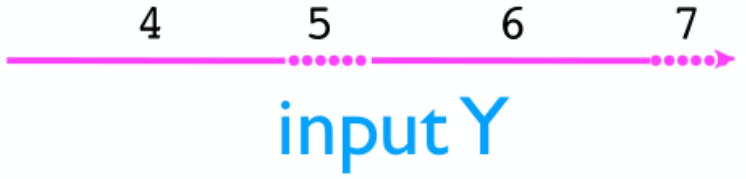
release Z if and only
if X and Y are present

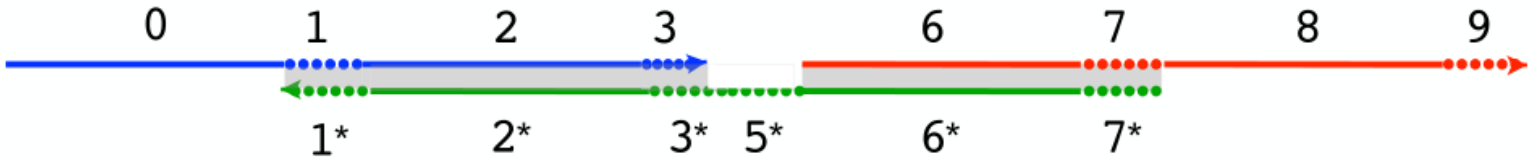


[slides credit: Chris Thachuk]

Strand displacement cascade example: AND gate

release Z if and only
if X and Y are present

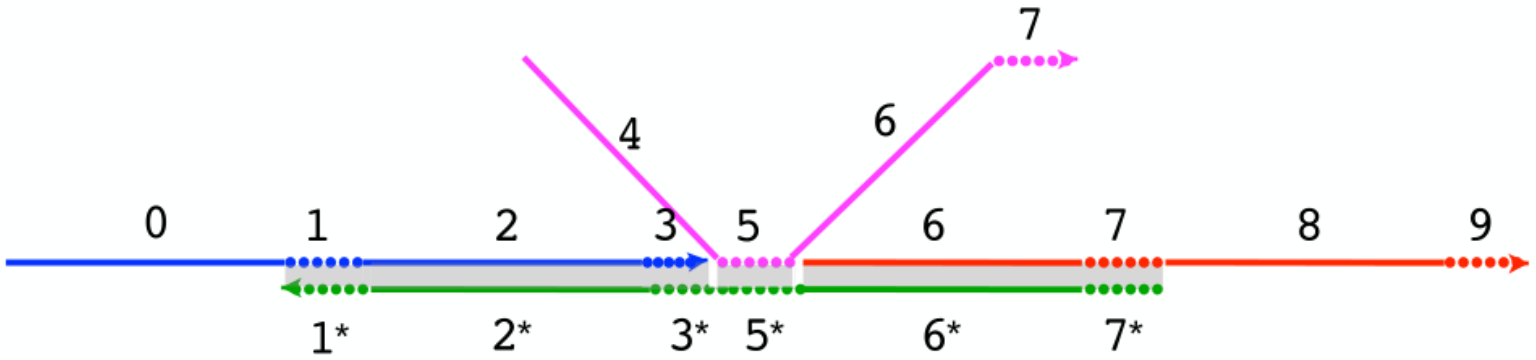




[slides credit: Chris Thachuk]

Strand displacement cascade example: AND gate

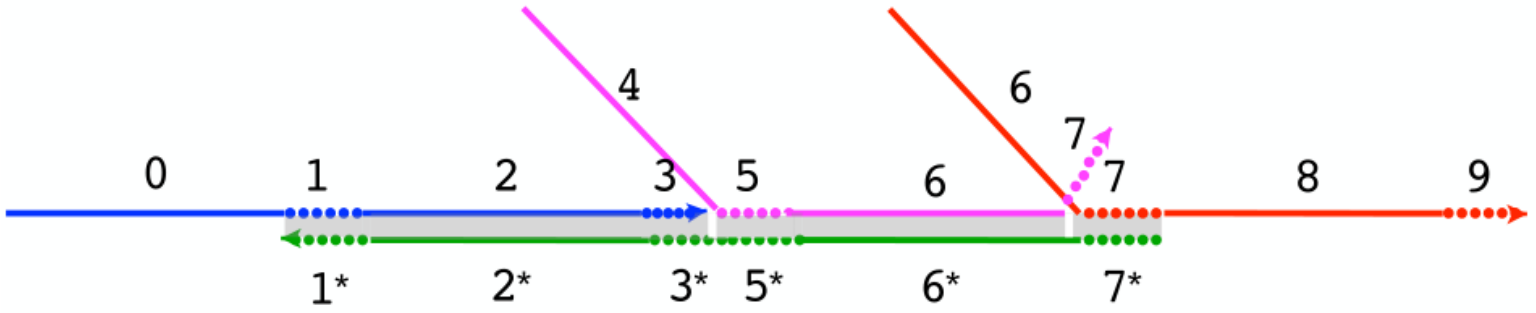
release Z if and only
if X and Y are present



[slides credit: Chris Thachuk]

Strand displacement cascade example: AND gate

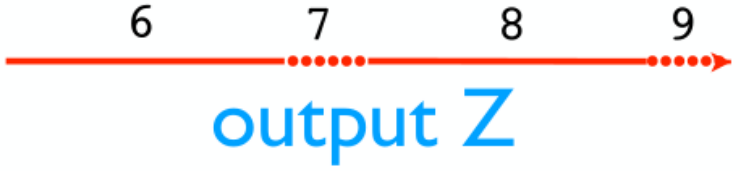
release Z if and only
if X and Y are present

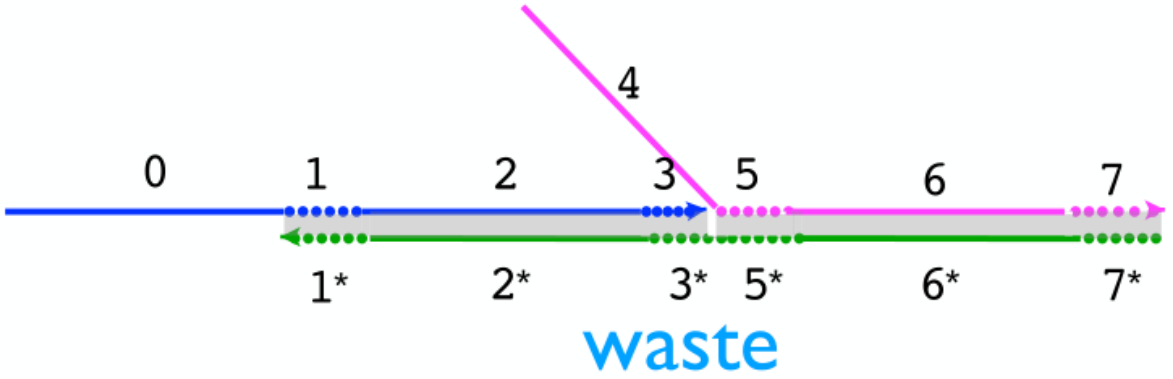


[slides credit: Chris Thachuk]

Strand displacement cascade example: AND gate

release Z if and only
if X and Y are present

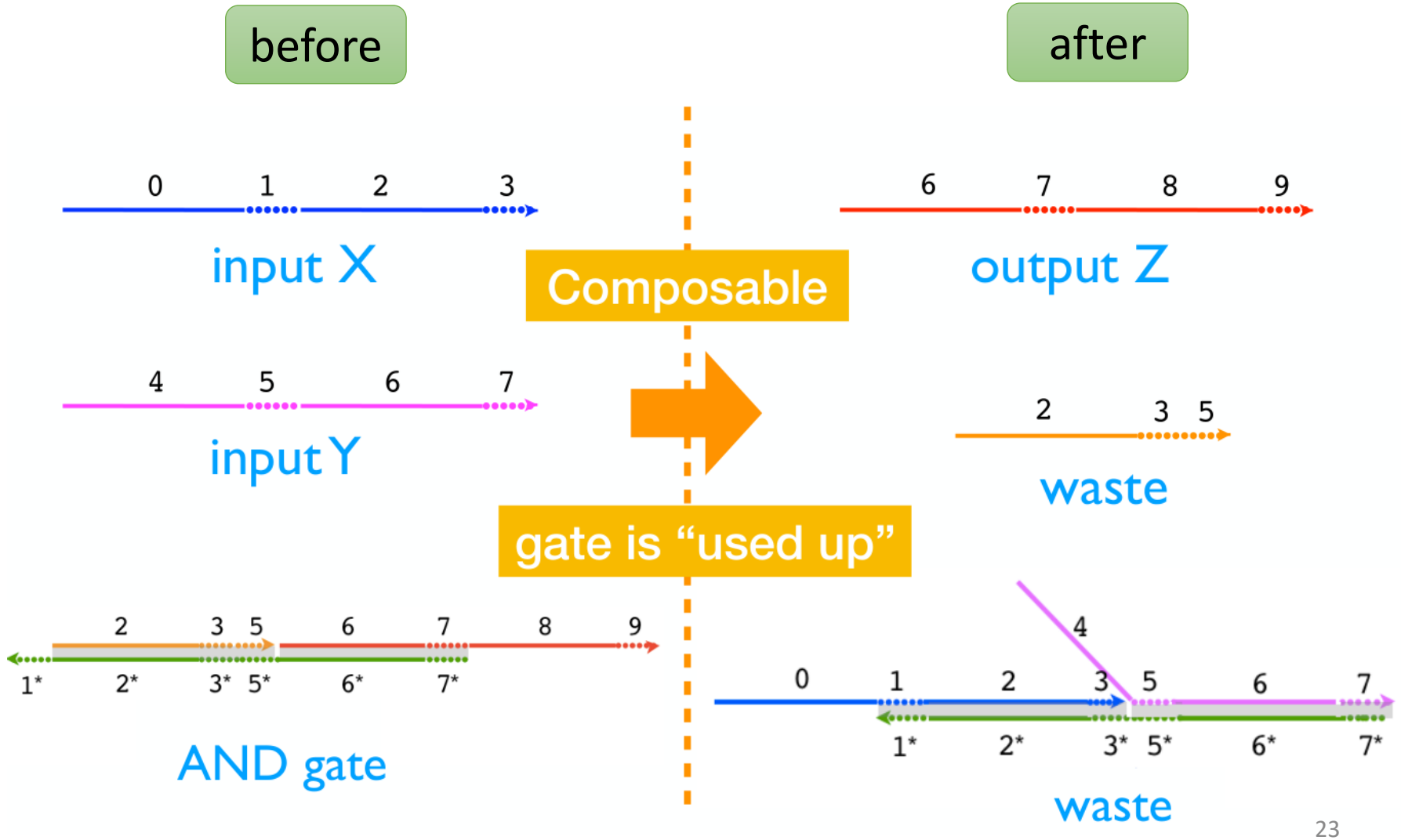




[slides credit: Chris Thachuk]

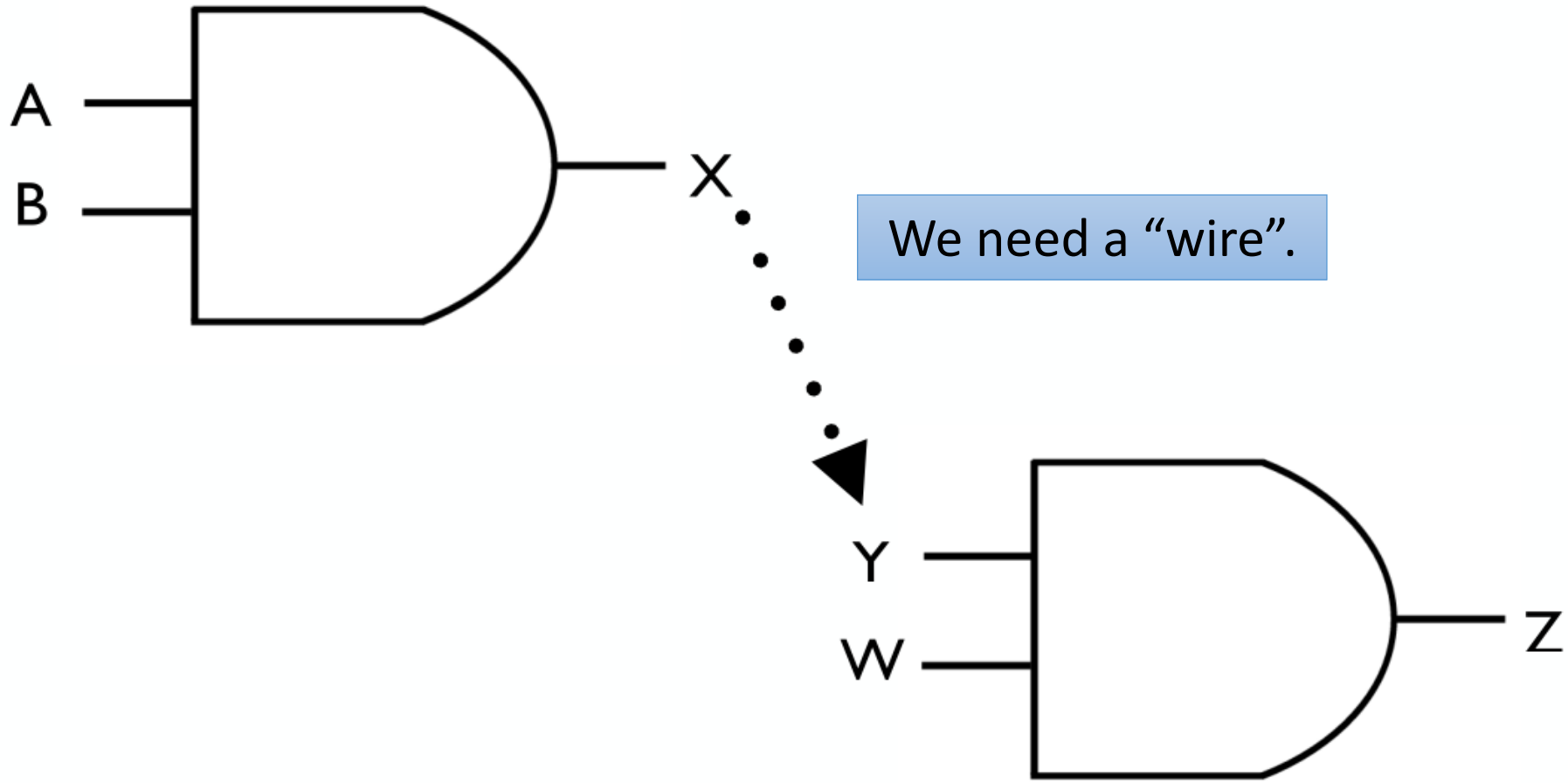
Strand displacement cascade example: AND gate

release Z if and only
if X and Y are present



[slides credit: Chris Thachuk]

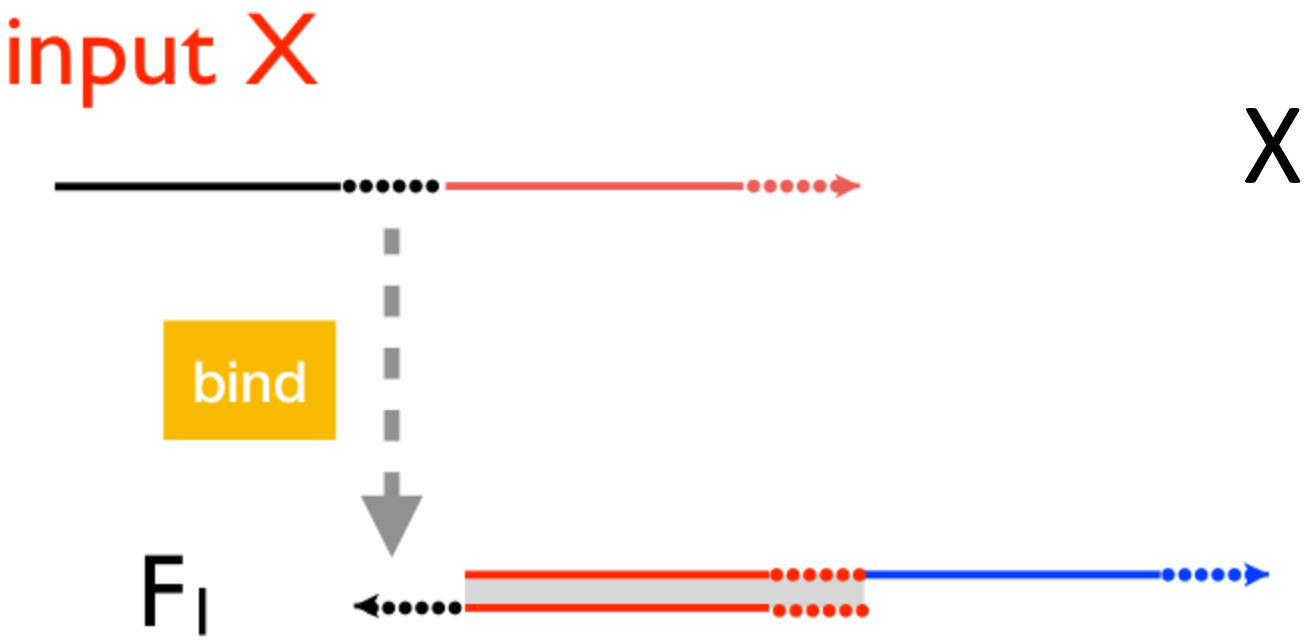
Composing AND gates

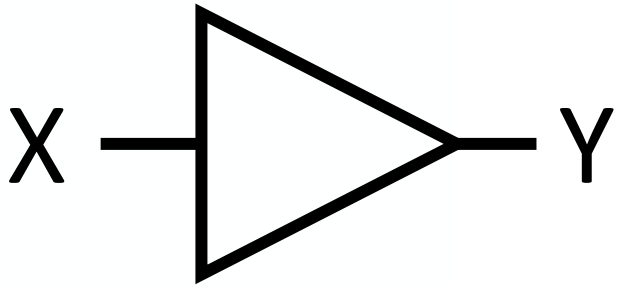


Translator gate (“wire”)

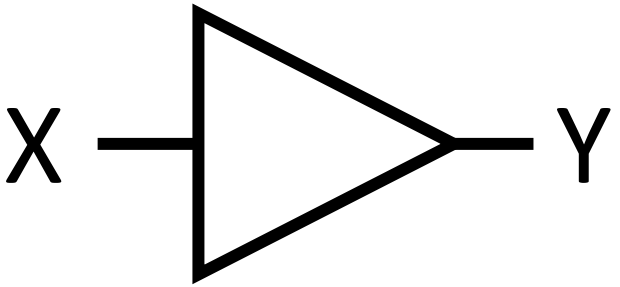
We need a “wire” to translate the signal:
X → Y
(with no shared DNA sequences between X and Y)

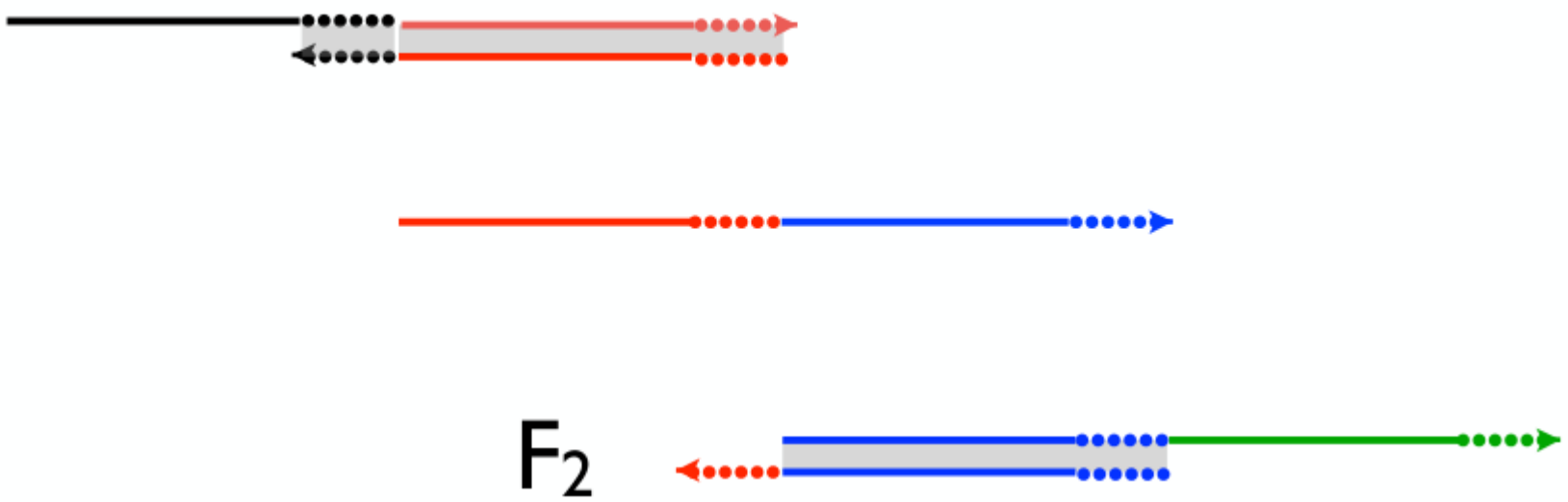
Translator gate (a “wire”)



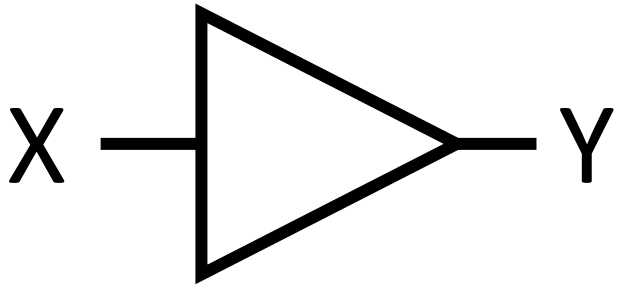


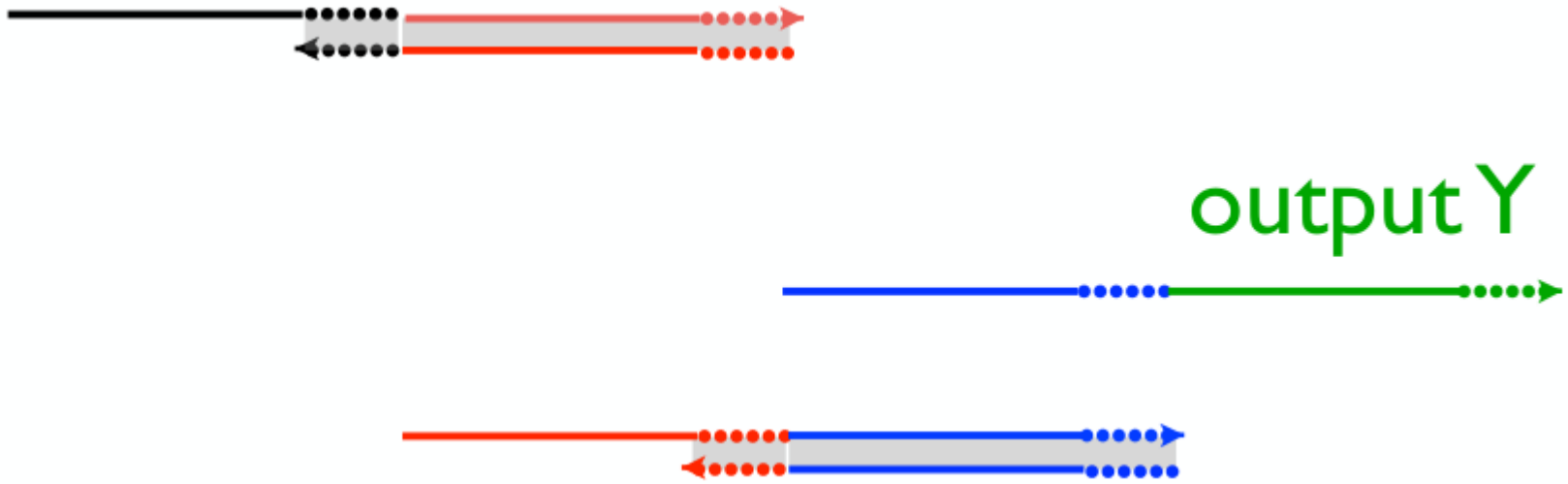
Translator gate (a “wire”)





Translator gate (a “wire”)

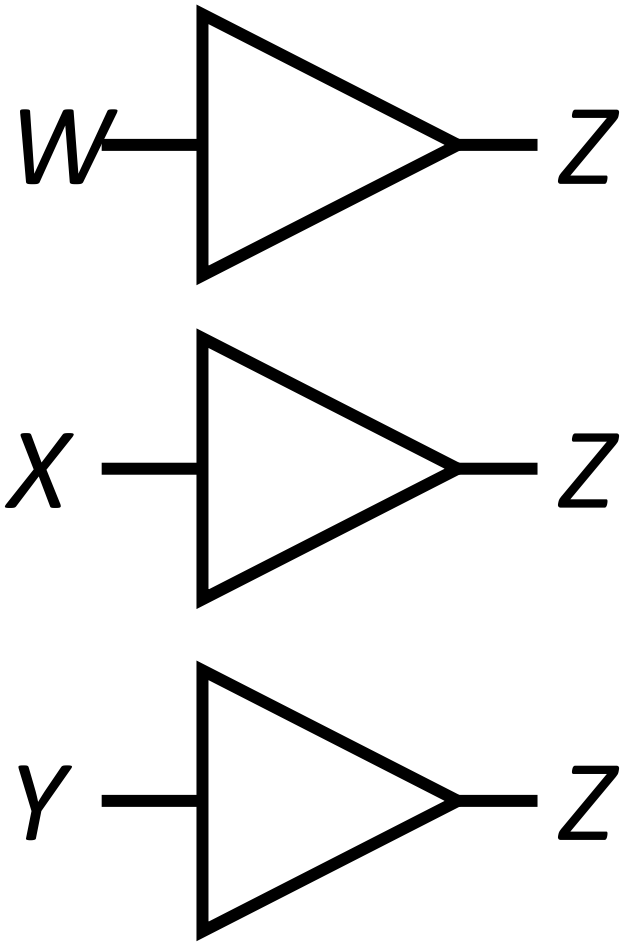




Strand displacement cascade example: OR gate

An OR gate can be implemented by multiple translators:

$Z \leftarrow W$ **OR** X **OR** Y



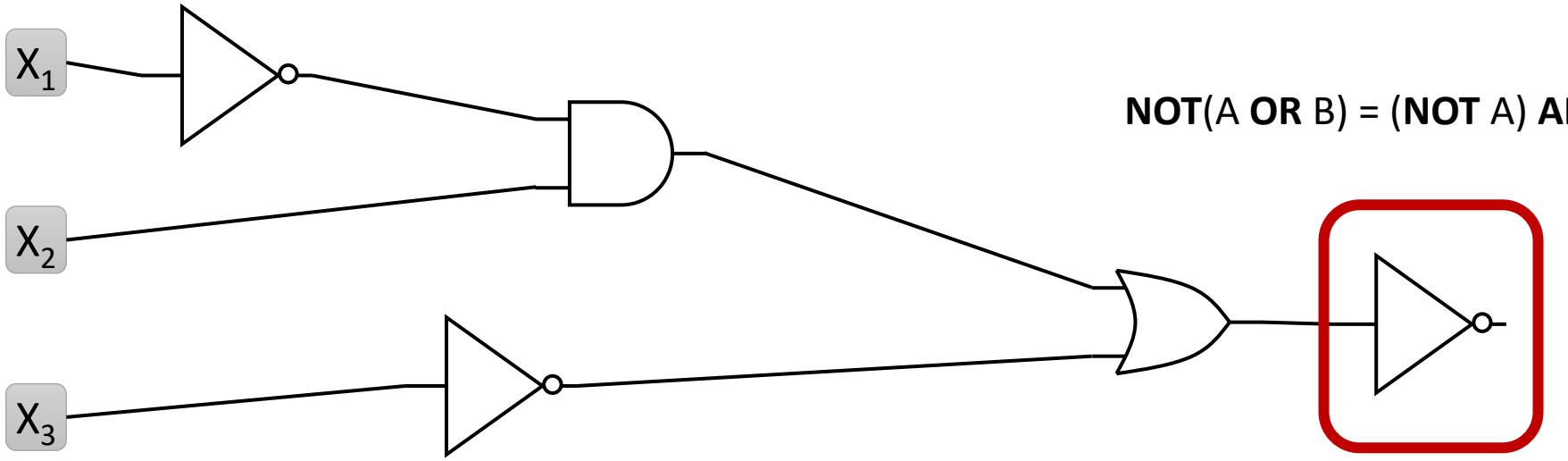
Strand displacement cascade example: Avoiding the need for NOT gates using dual-rail logic

NOT gates are tricky with molecular circuits:

How to make a molecule Y present
if and only if X is not present??

Instead we use dual-rail logic, using de Morgan's Laws to push all the NOT gates to the input.

(Then we can "manually" specify FALSE input values by the presence of a "negated" strand.)



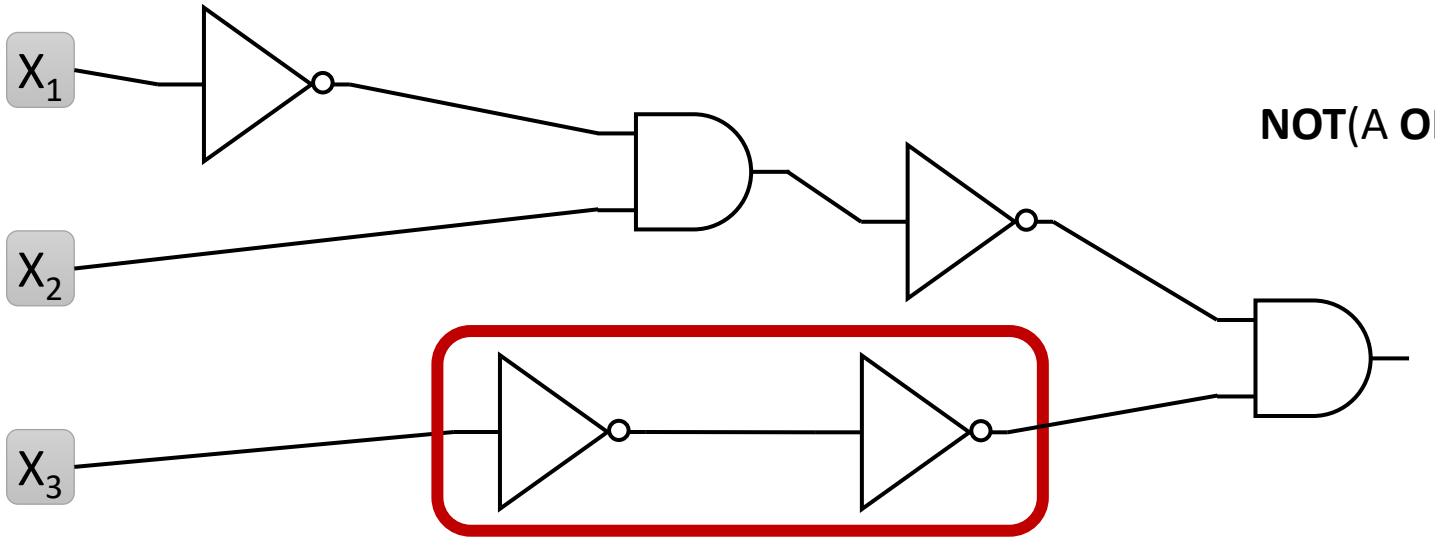
NOT(A **OR** B) = (**NOT** A) **AND** (**NOT** B)

Strand displacement cascade example: Avoiding the need for NOT gates using dual-rail logic

NOT gates are tricky with molecular circuits:

How to make a molecule Y present if and only if X is not present??

Instead we use “dual-rail” logic, using de Morgan’s Laws to push all the NOT gates to the input, so we can “manually” specify FALSE input values.



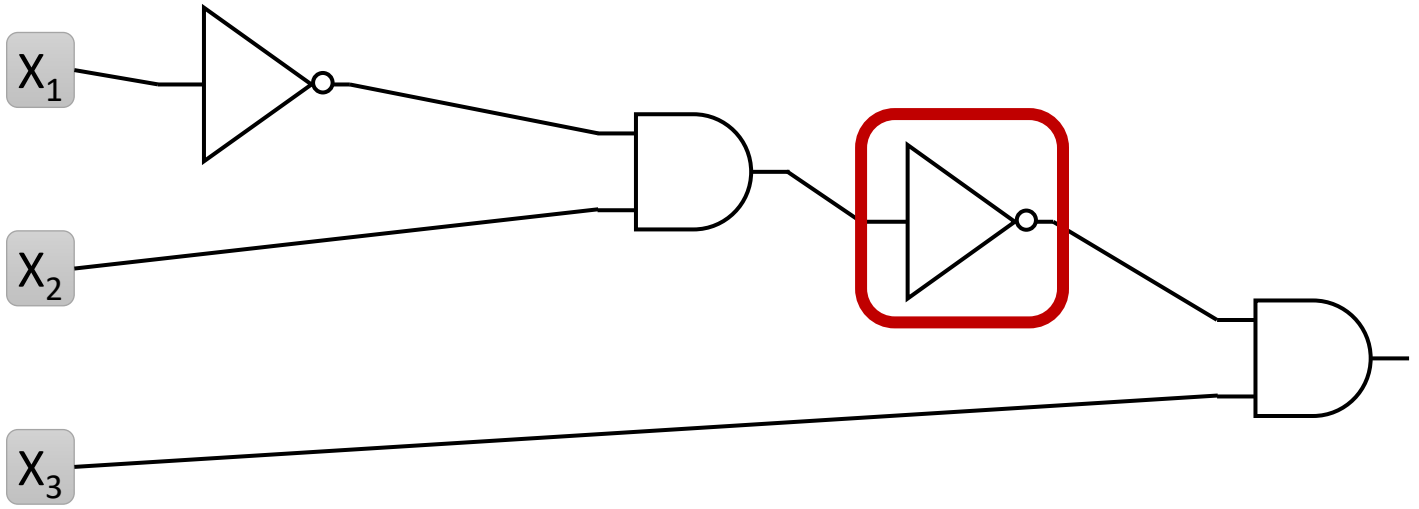
NOT(A **OR** B) = (**NOT** A) **AND** (**NOT** B)

Strand displacement cascade example: Avoiding the need for NOT gates using dual-rail logic

NOT gates are tricky with molecular circuits:

How to make a molecule Y present
if and only if X is not present??

Instead we use “dual-rail” logic, using de Morgan’s Laws to push all the NOT gates to the input, so we can “manually” specify FALSE input values.

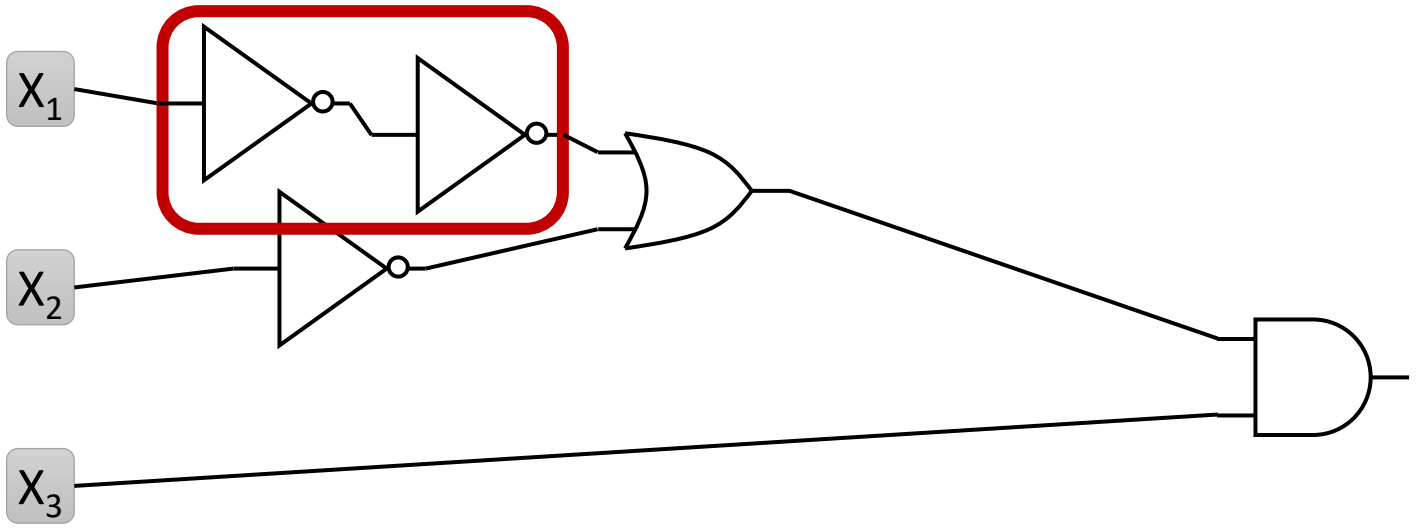


Strand displacement cascade example: Avoiding the need for NOT gates using dual-rail logic

NOT gates are tricky with molecular circuits:

How to make a molecule Y present
if and only if X is not present??

Instead we use “dual-rail” logic, using de Morgan’s Laws to push all the NOT gates to the input, so we can “manually” specify FALSE input values.

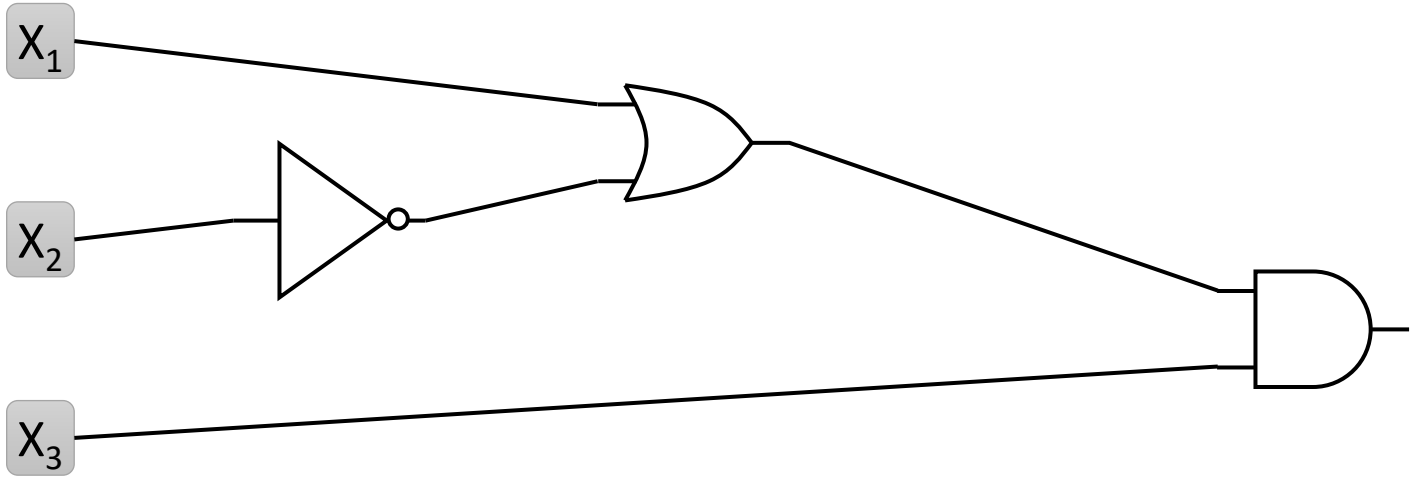


Strand displacement cascade example: Avoiding the need for NOT gates using dual-rail logic

NOT gates are tricky with molecular circuits:

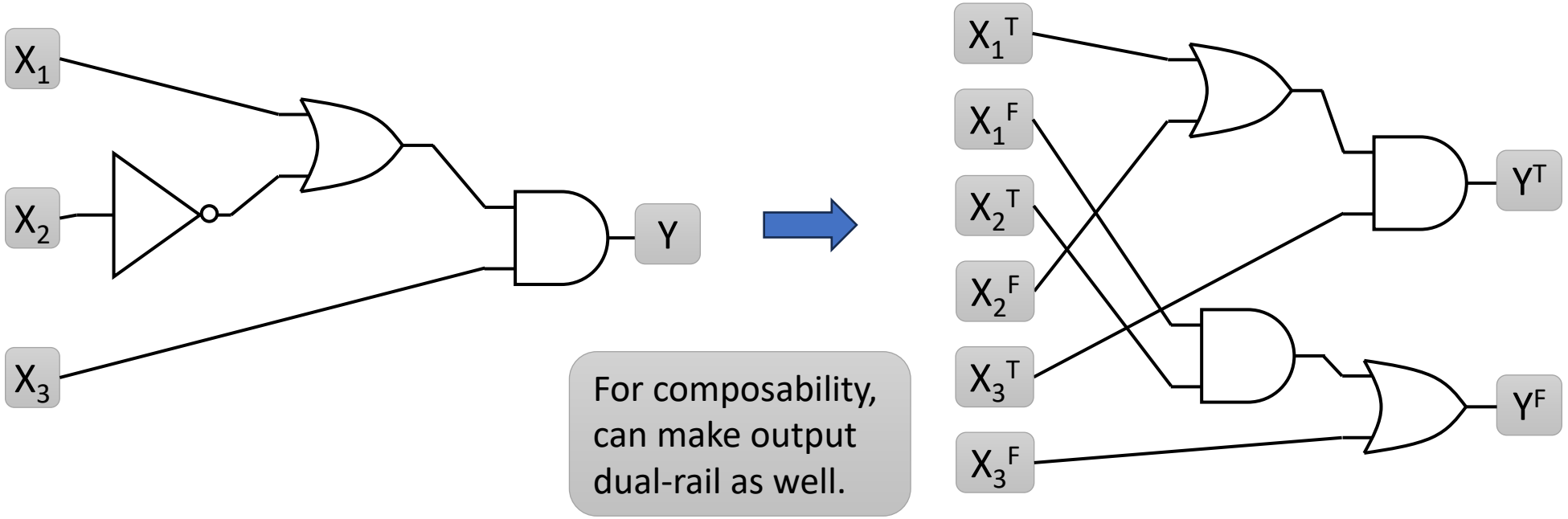
How to make a molecule Y present
if and only if X is not present??

Instead we use “dual-rail” logic, using de Morgan’s Laws to push all the NOT gates to the input, so we can “manually” specify FALSE input values.



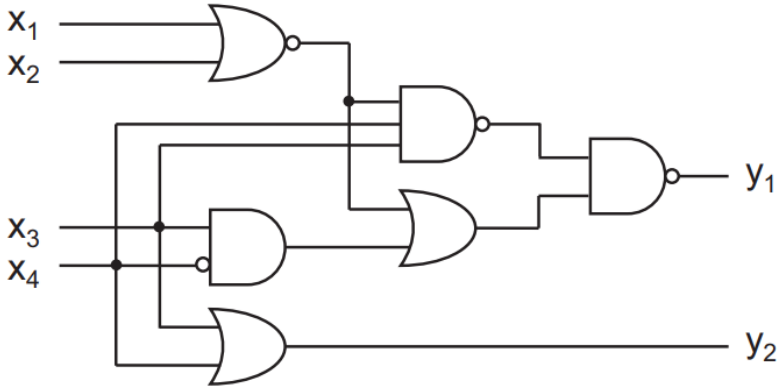
Strand displacement cascade example: Avoiding the need for NOT gates using dual-rail logic

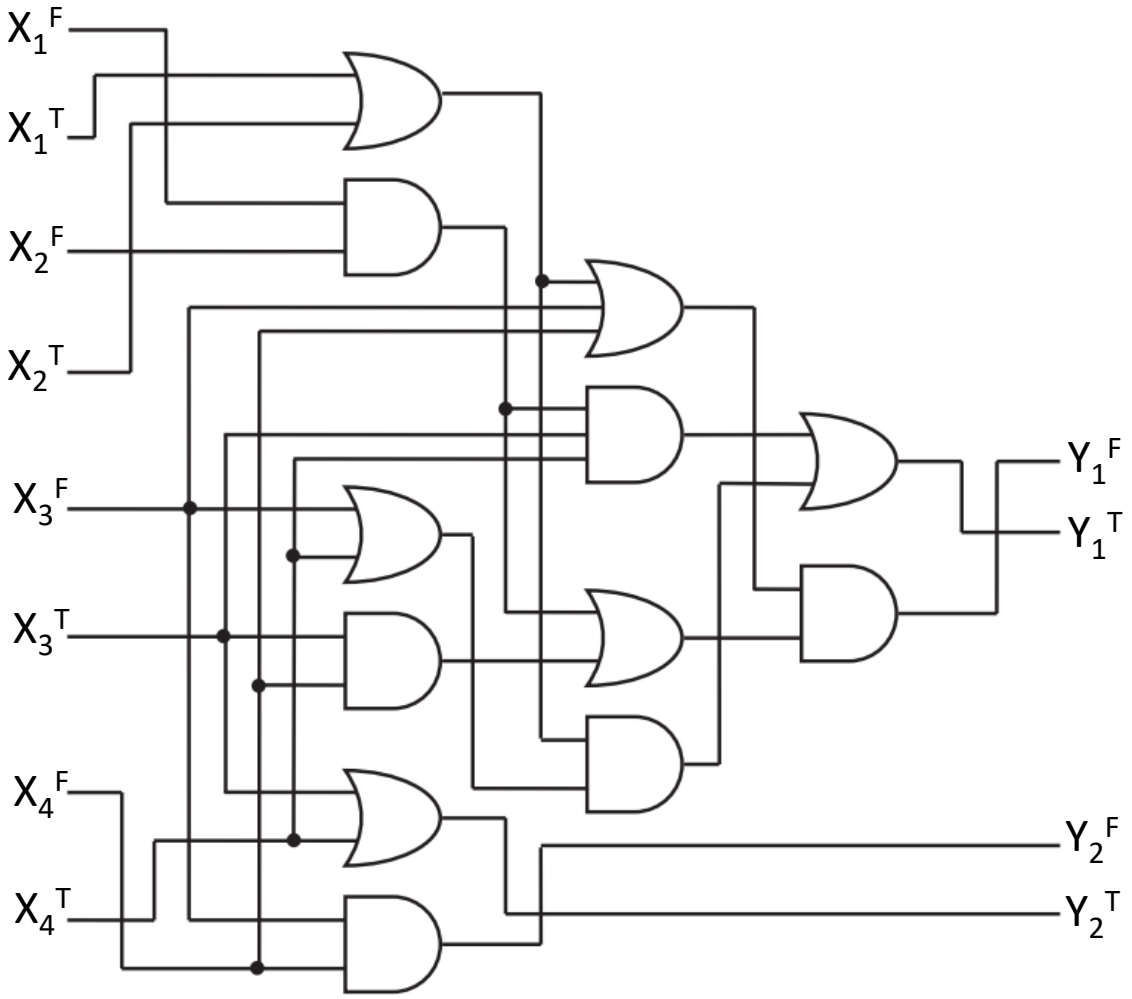
For each input X_i, there are two species X_i^T and X_i^F:
Give species X_i^F to specify that Boolean input X_i = *False*
Give species X_i^T to specify that Boolean input X_i = *True*.



Dual-rail logic computing square root of 4-bit number

$$y_2 y_1 = \left\lfloor \sqrt{x_4 x_3 x_2 x_1} \right\rfloor$$





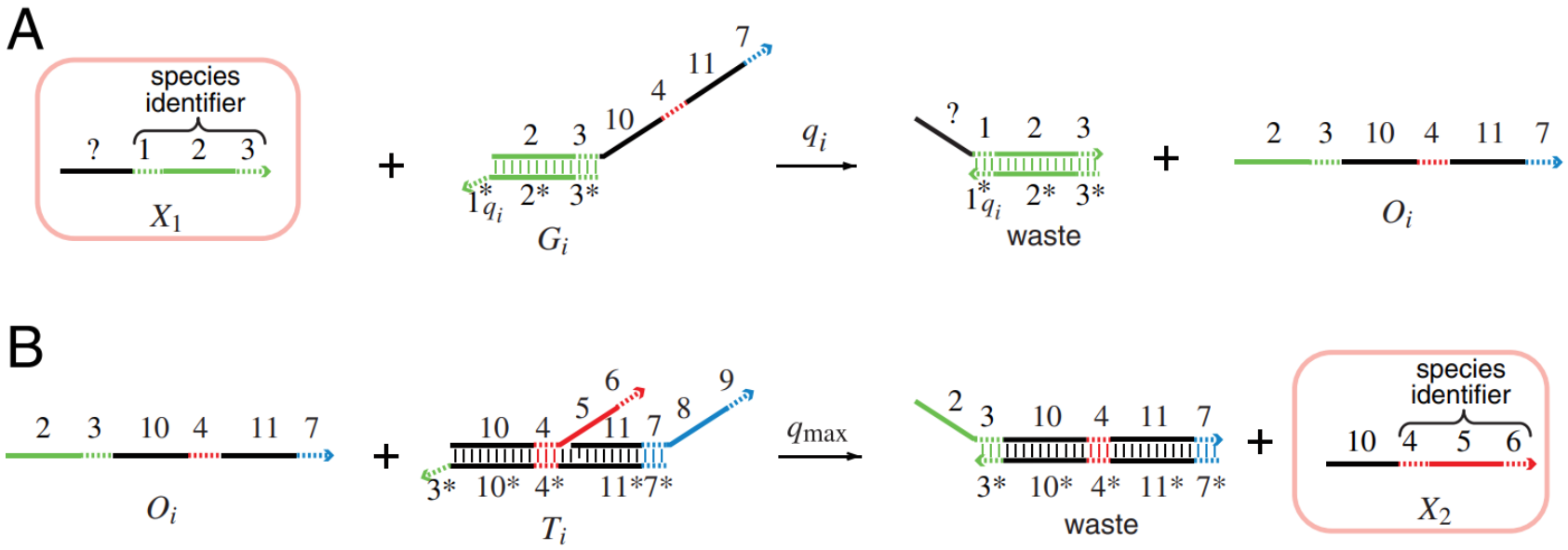
[Scaling up digital circuit computation with DNA strand displacement cascades. Lulu Qian and Erik Winfree, *Science* 2011]

Implementing CRNs with DNA

"Compiling" arbitrary chemical reaction networks into DNA strands that implement the reactions using DNA strand displacement

DNA strand displacement can implement any CRN

unimolecular reaction $X_1 \rightarrow X_2 + X_3$



reaction i: $X_1 \xrightarrow{k_i} X_2 + X_3$ **[1]**

⇓ implement

$X_1 + G_i \xrightarrow{q_i} O_i$ **[2]**

$O_i + T_i \xrightarrow{q_{max}} X_2 + X_3$ **[3]**

⇓ simplify

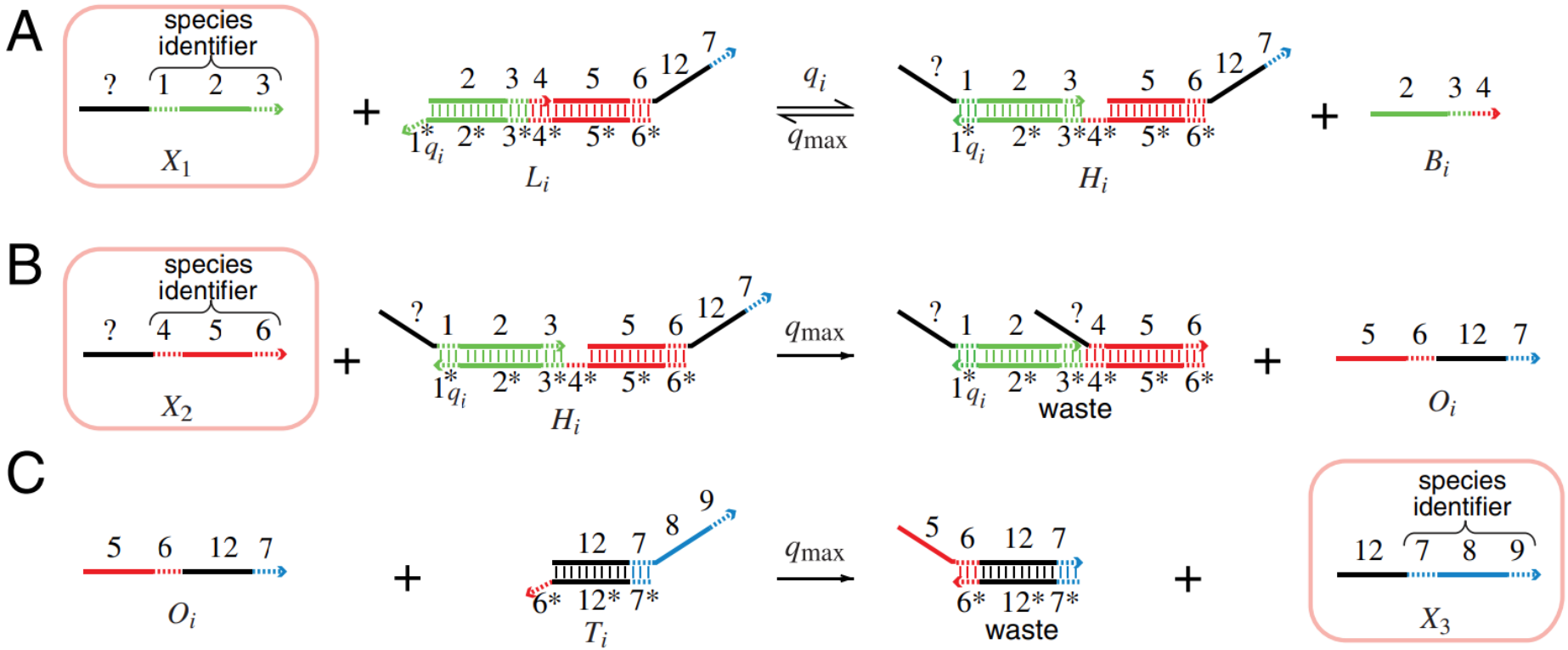
$X_1 \xrightarrow{q_i C_{max}} O_i$ **[4]**

$O_i \xrightarrow{q_{max} C_{max}} X_2 + X_3$ **[5]**

[DNA as a universal substrate for chemical kinetics, David Soloveichik, Georg Seelig, and Erik Winfree, *PNAS* 2010]

DNA strand displacement can implement any CRN

bimolecular reaction $X_1 + X_2 \rightarrow X_3$



reaction i : $X_1 + X_2 \xrightarrow{k_i} X_3$ [6]

⇓ implement

$X_1 + L_i \underset{q_{max}}{\overset{q_i}{\rightleftharpoons}} H_i + B_i$ [7]

$X_2 + H_i \xrightarrow{q_{max}} O_i$ [8]

$O_i + T_i \xrightarrow{q_{max}} X_3$ [9]

⇓ simplify

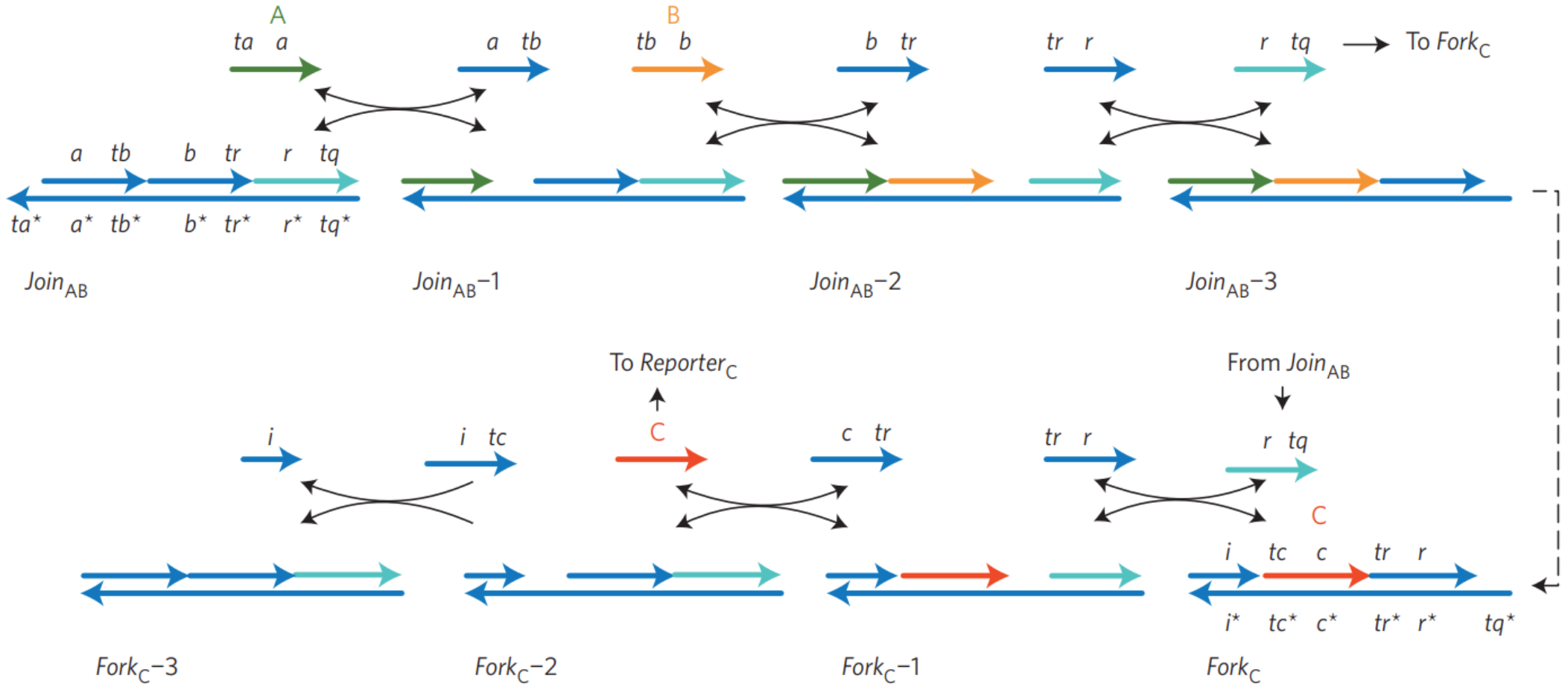
$X_1 \underset{q_{max}C_{max}}{\overset{q_iC_{max}}{\rightleftharpoons}} H_i$ [10]

$X_2 + H_i \xrightarrow{q_{max}} X_3$ [11]

[DNA as a universal substrate for chemical kinetics, David Soloveichik, Georg Seelig, and Erik Winfree, *PNAS* 2010]

“Two-domain” scheme for compiling CRN to DSD

reaction A+B → C

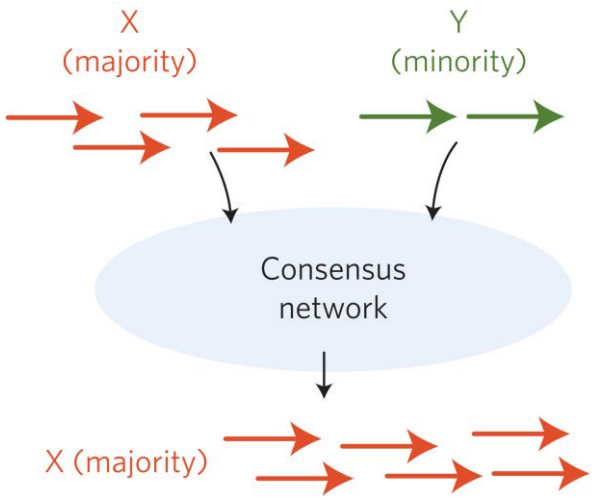


[Programmable chemical controllers made from DNA. Yuan-Jyue Chen, Neil Dalchau, Niranjan Srinivas, Andrew Phillips, Luca Cardelli, David Soloveichik, and Georg Seelig. *Nature Nanotechnology* 2013.]

Experimental implementations of CRN-to-DSD schemes

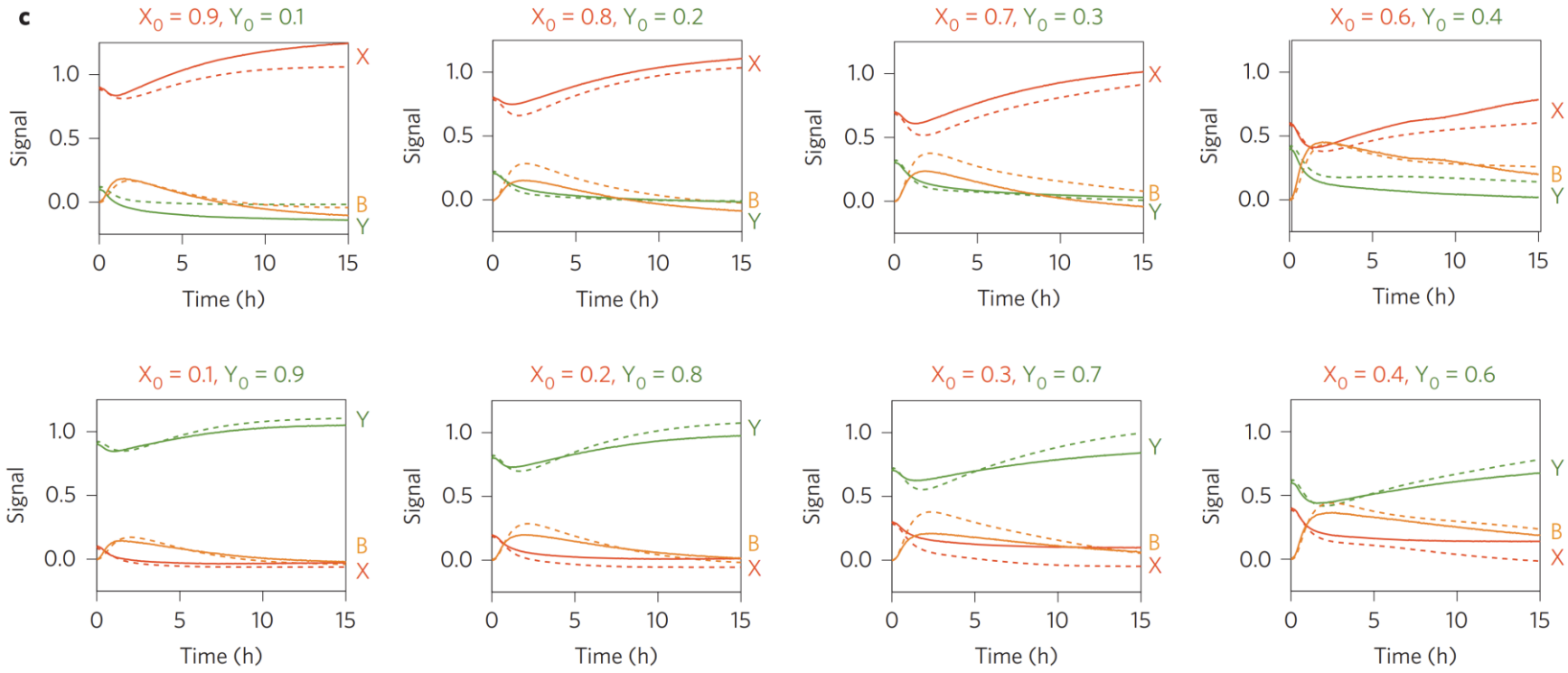
DSD computing approximate majority

Goal:



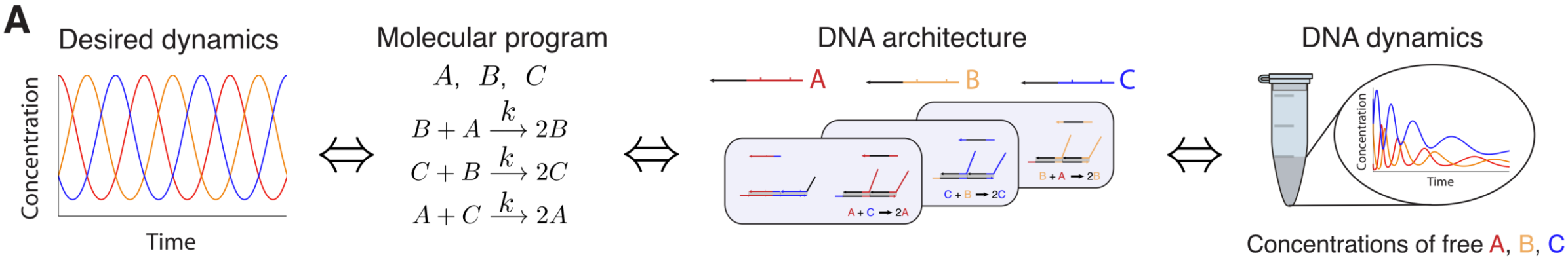
CRN:

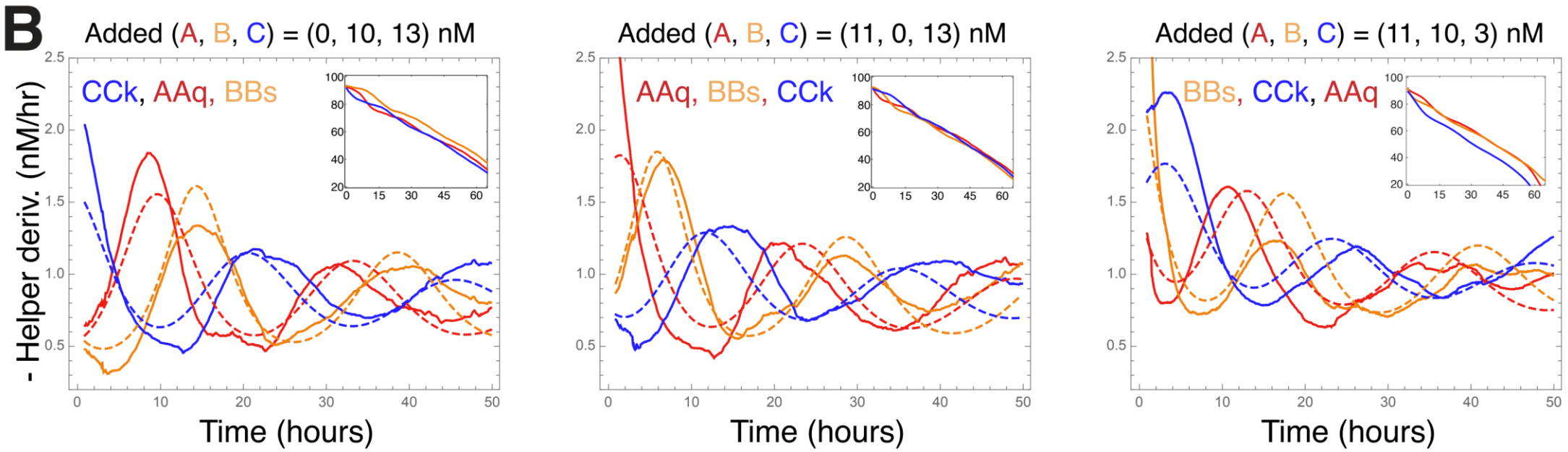
X+Y → 2B
X+B → 2X
Y+B → 2Y



[Programmable chemical controllers made from DNA. Yuan-Jyue Chen, Neil Dalchau, Niranjan Srinivas, Andrew Phillips, Luca Cardelli, David Soloveichik, and Georg Seelig. *Nature Nanotechnology* 2013.]

DSD implementing chemical “rock-paper-scissors” oscillator





[Enzyme-free nucleic acid dynamical systems. Niranjan Srinivas, James Parkin, Georg Seelig, Erik Winfree, David Soloveichik, *Science* 2017]