NVIDIA.

DEEP
LEARNING
INSTITUTE

MULTI-GPU PROGRAMMING FOR CUDA C++

Dr. Momme Allalen | LRZ | 30.11.2021

INTRODUCTION TO CUDA STREAMS

INTRODUCTION TO CUDA STREAMS

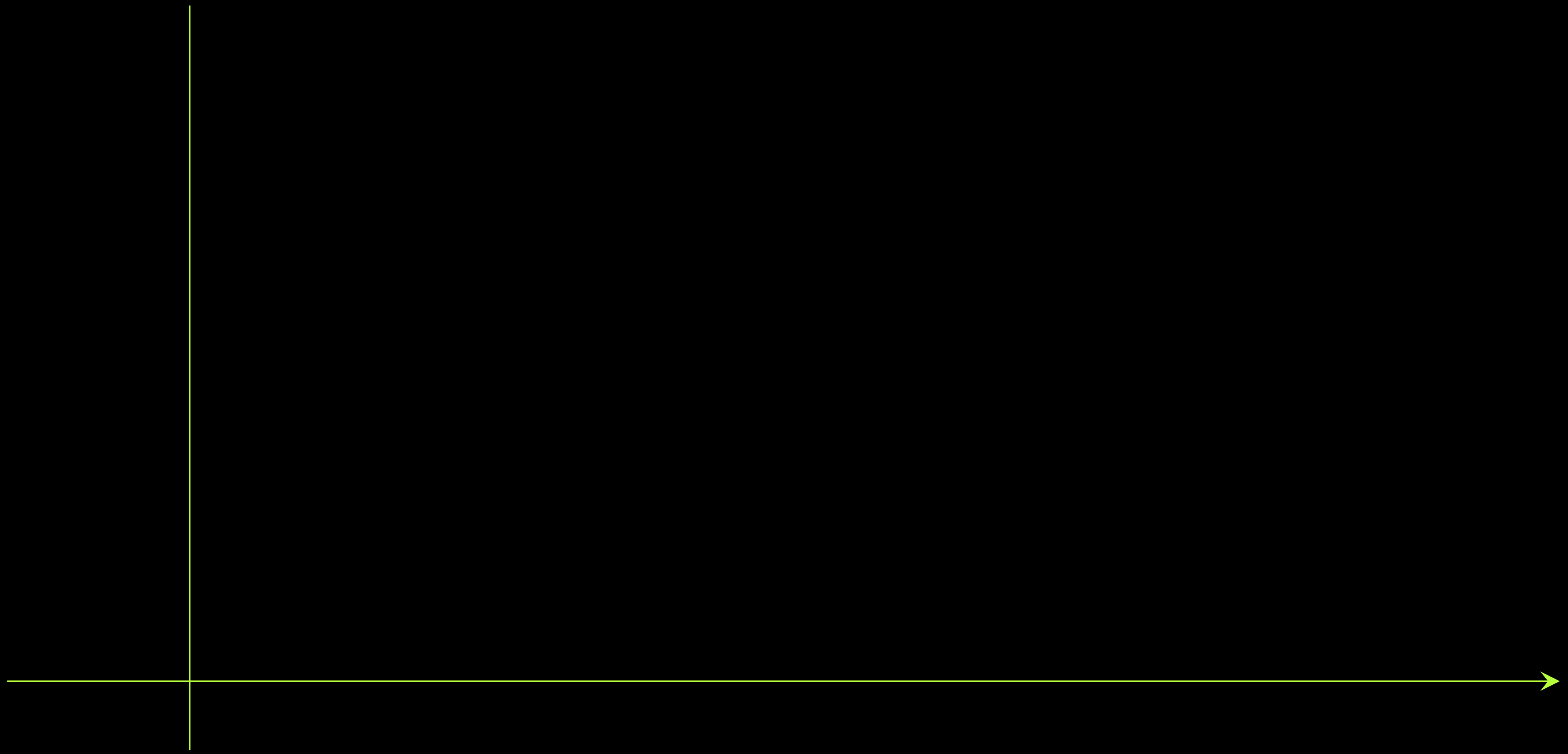
Stream Behavior

Default Stream Behavior

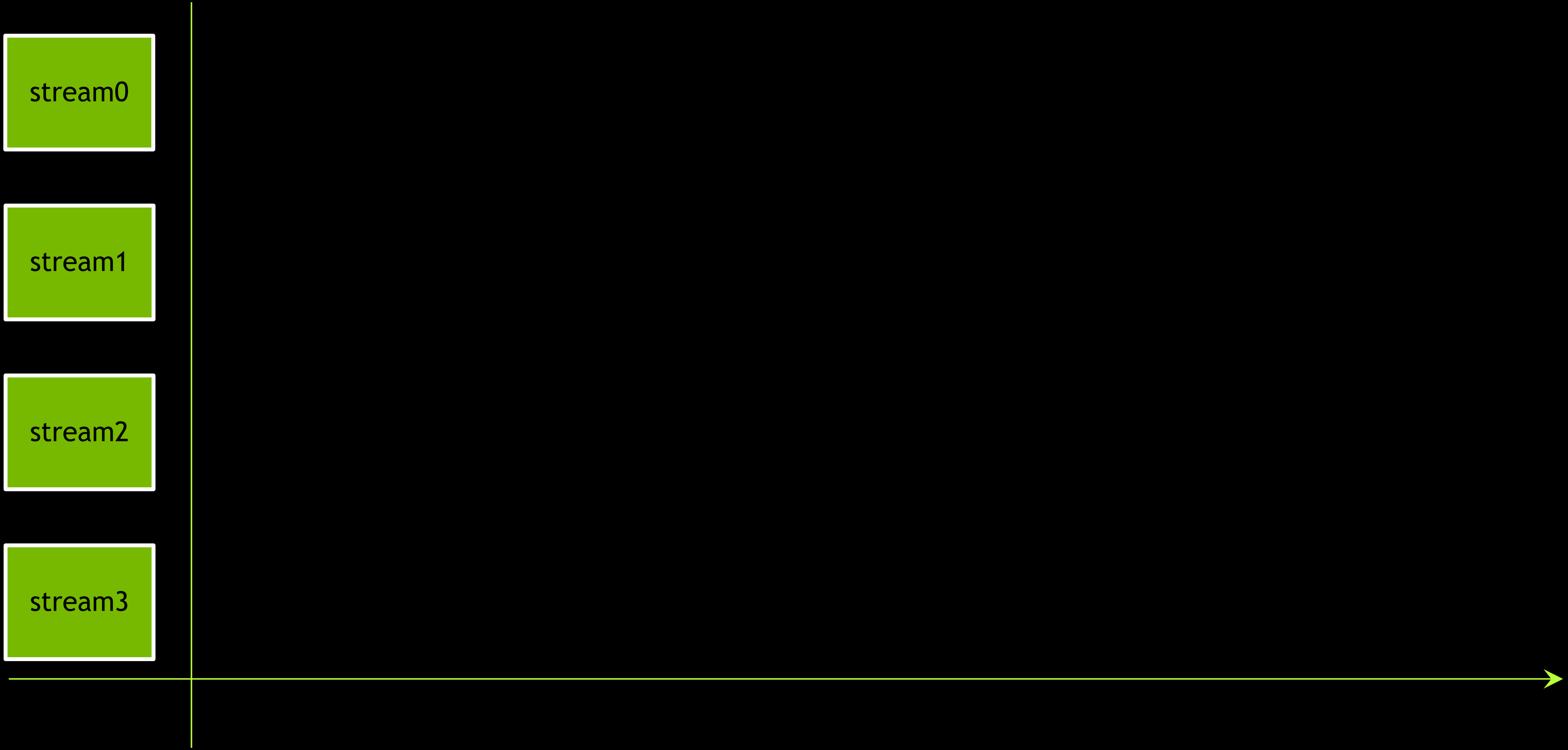
Streams in CUDA Programming

STREAM BEHAVIOR

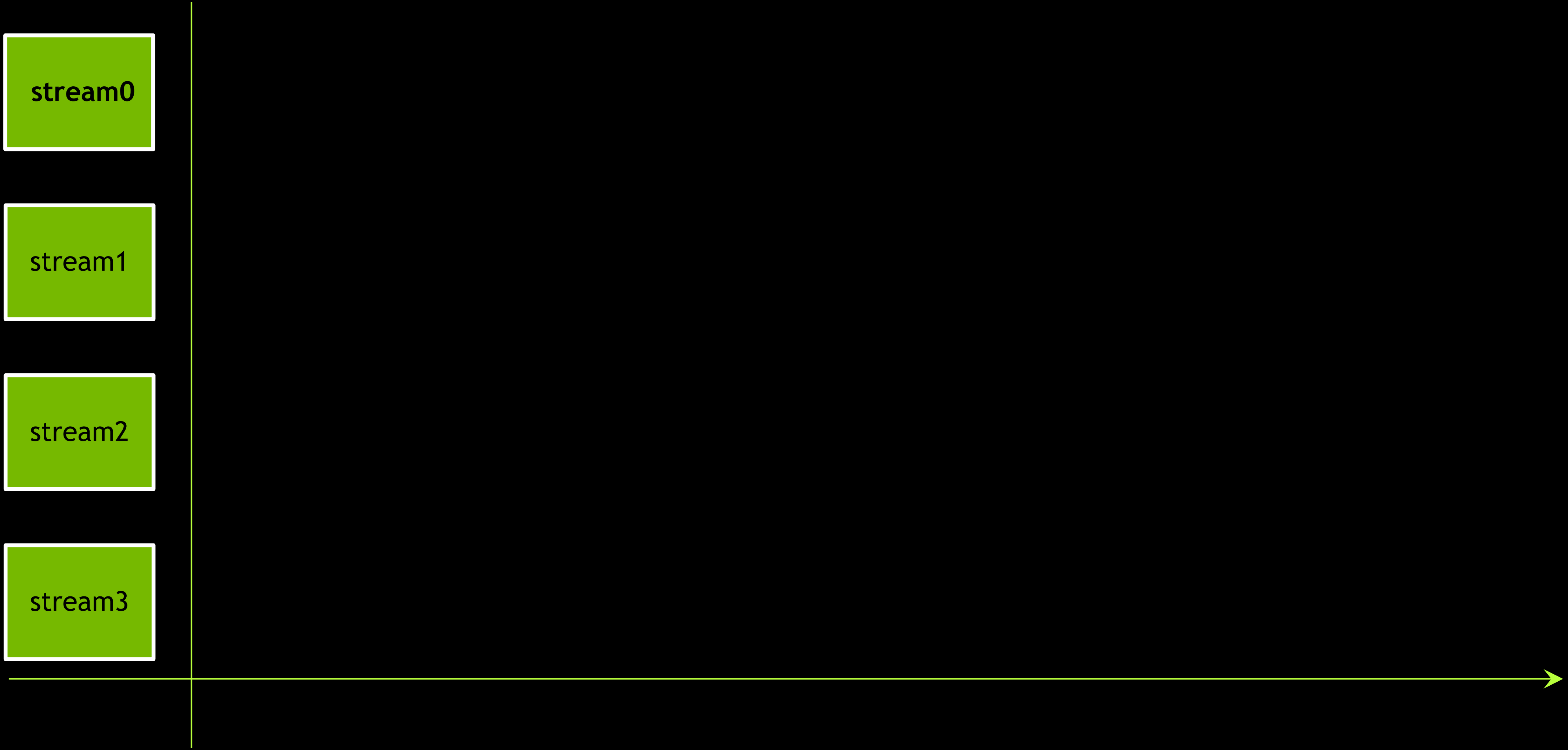
A **stream** is a series of operations that occur in issue order on the GPU



Multiple streams can be created and utilized by CUDA programmers



A special stream called the **default stream** (here labeled as stream0)



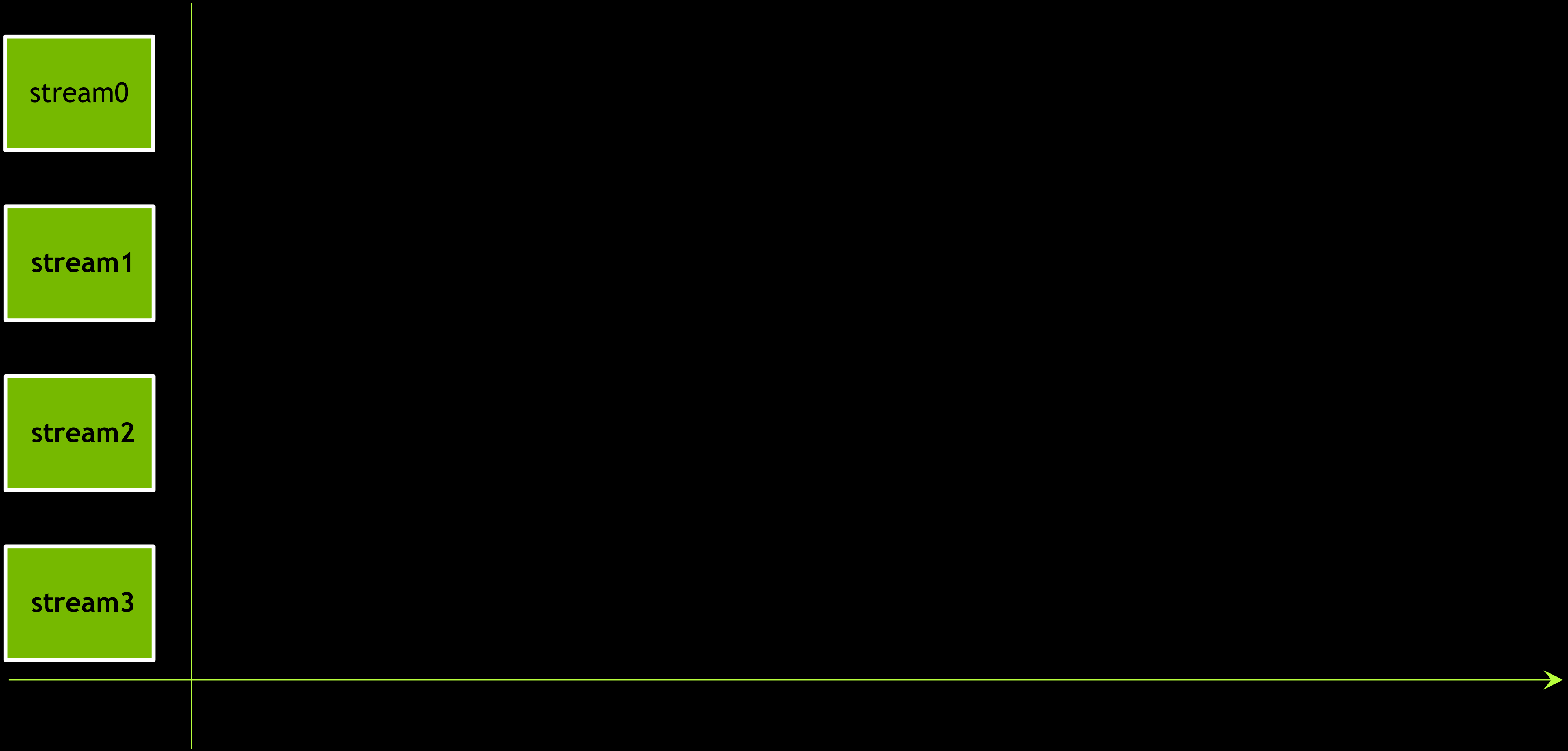
stream0

stream1

stream2

stream3

All other streams are referred to as **non-default streams** (here labelled streams 1-3)



stream0

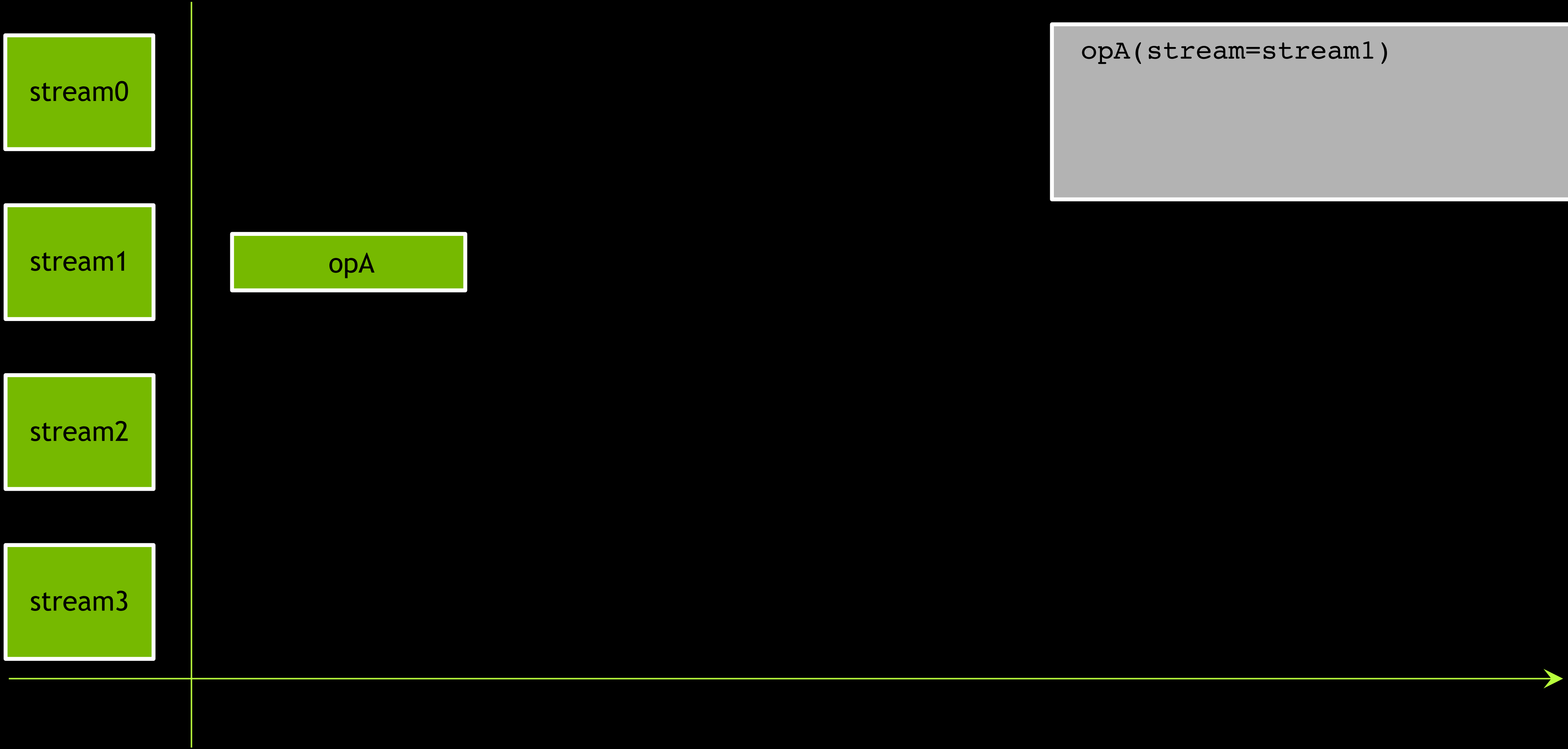
stream1

stream2

stream3

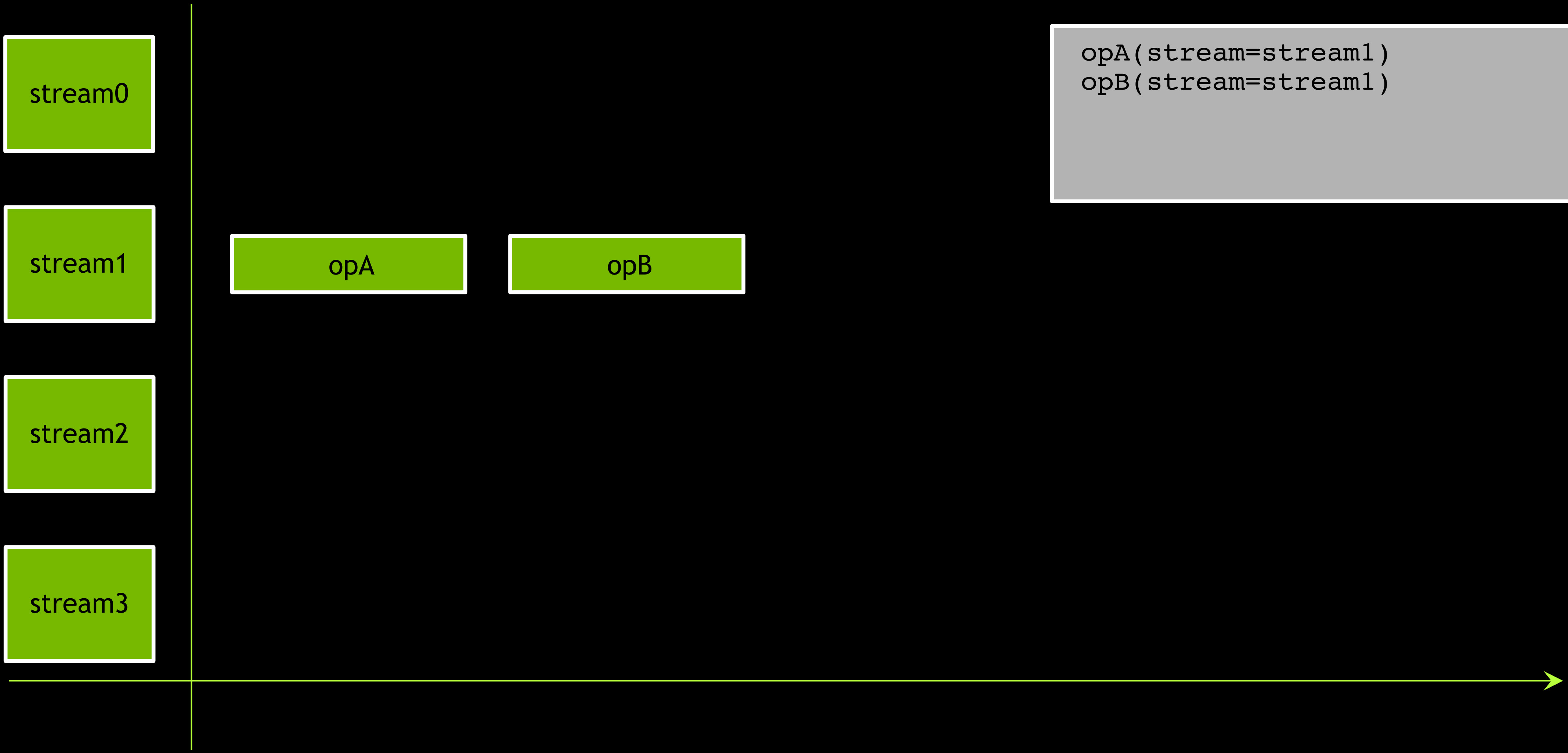
Operations in the same stream will execute in issue order

```
opA(stream=stream1)
```



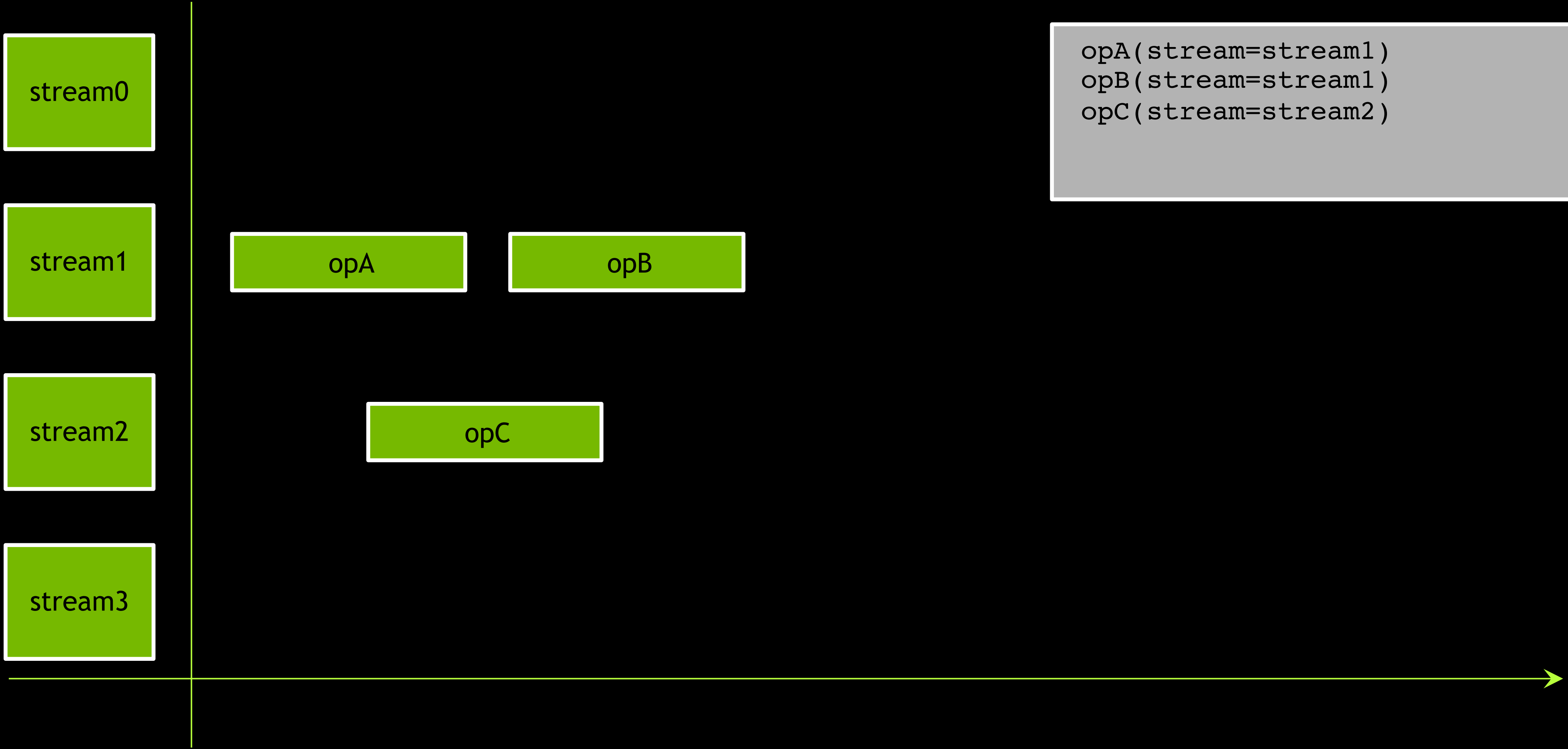
Operations in the same stream will execute in issue order

```
opA(stream=stream1)
opB(stream=stream1)
```



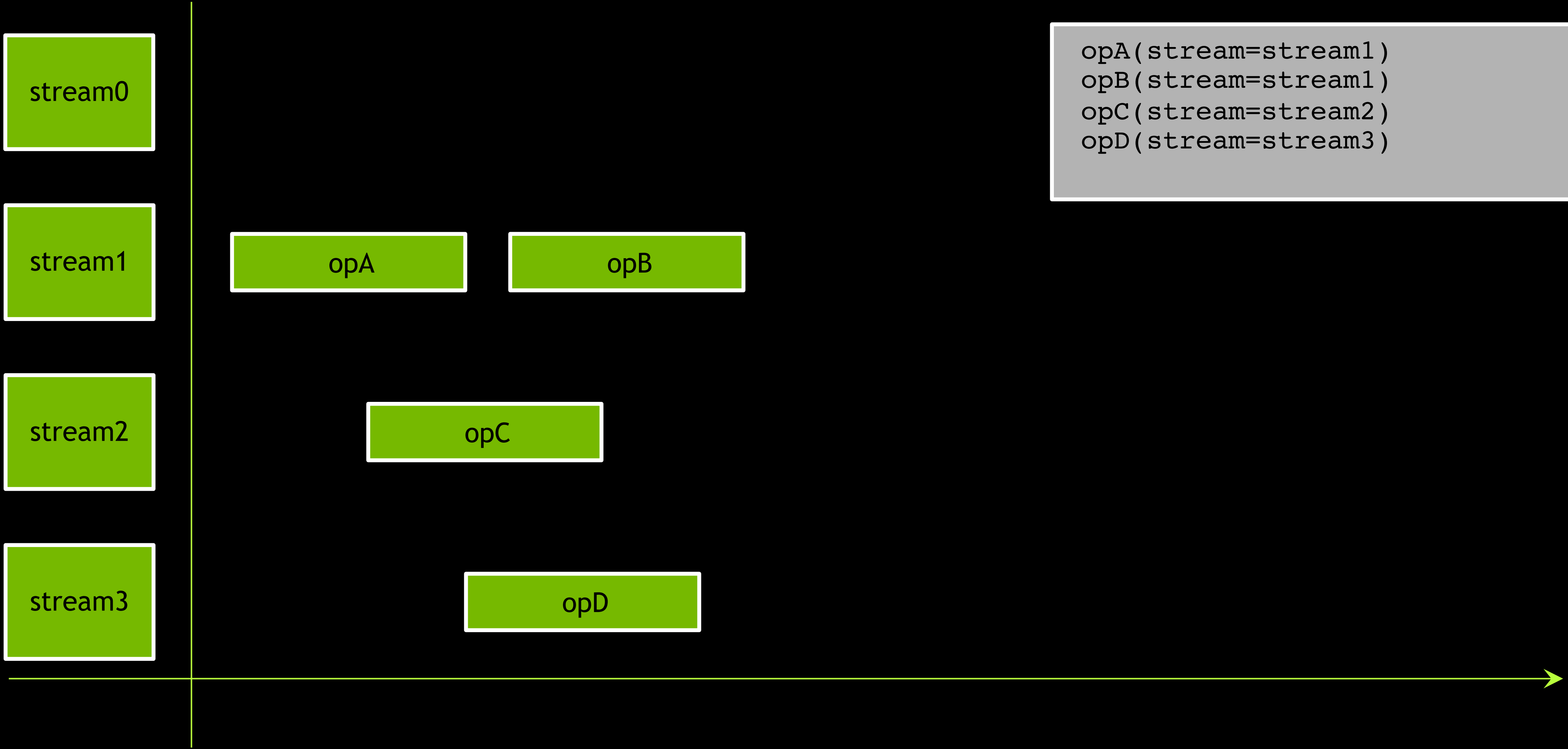
However, operations launched in **different non-default streams** have no fixed order of execution

```
opA(stream=stream1)
opB(stream=stream1)
opC(stream=stream2)
```



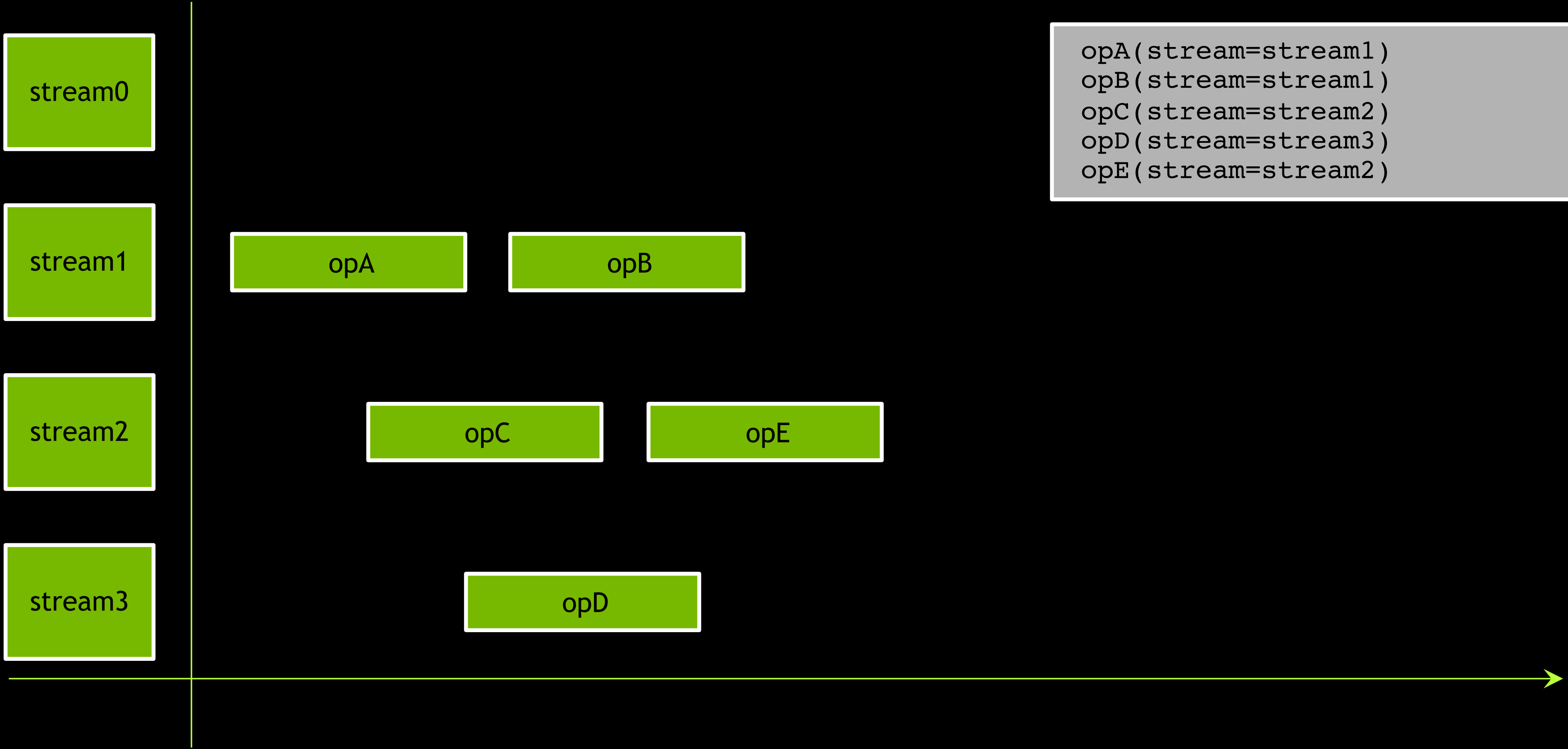
However, operations launched in **different non-default streams** have no fixed order of execution

```
opA(stream=stream1)
opB(stream=stream1)
opC(stream=stream2)
opD(stream=stream3)
```



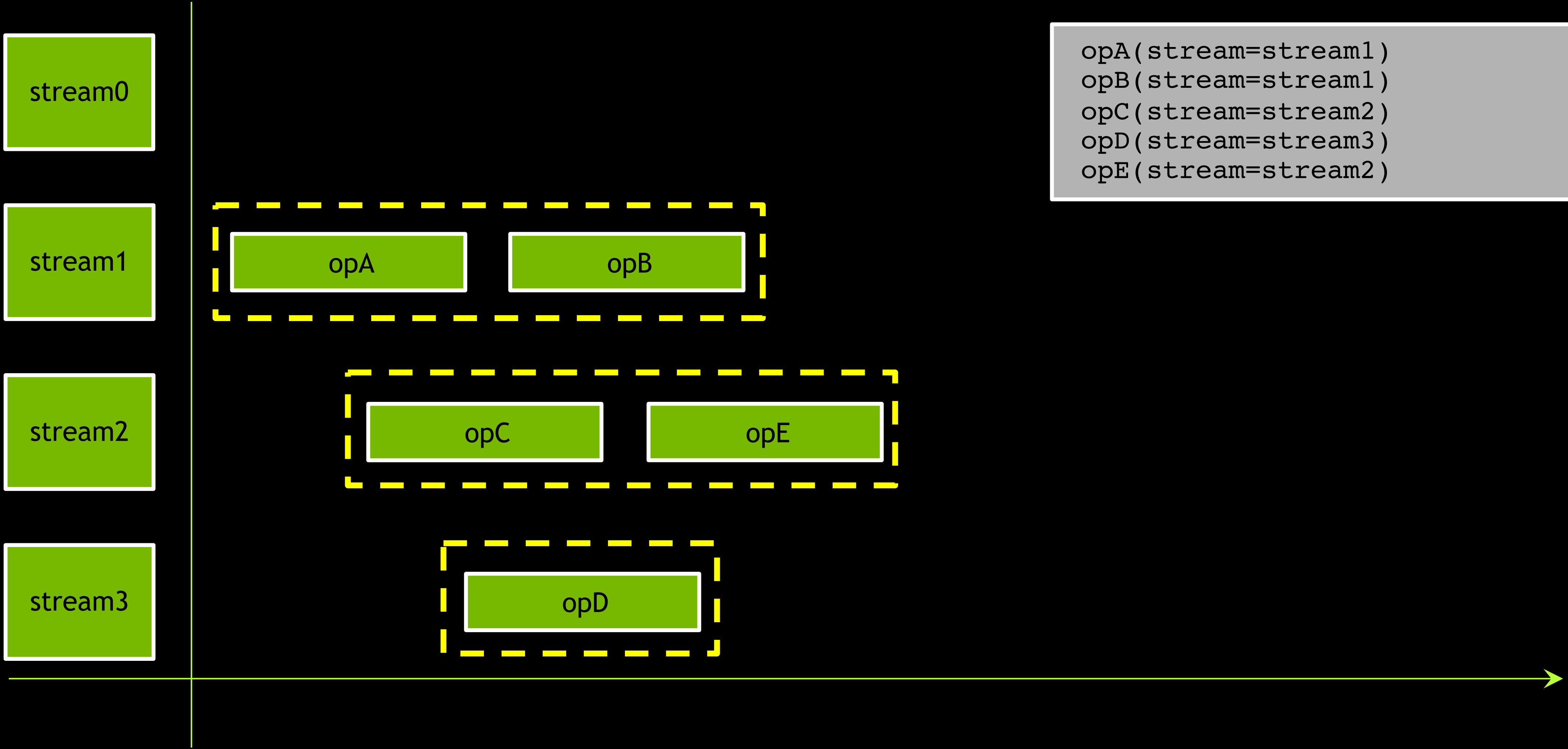
However, operations launched in **different non-default streams** have no fixed order of execution

```
opA(stream=stream1)
opB(stream=stream1)
opC(stream=stream2)
opD(stream=stream3)
opE(stream=stream2)
```



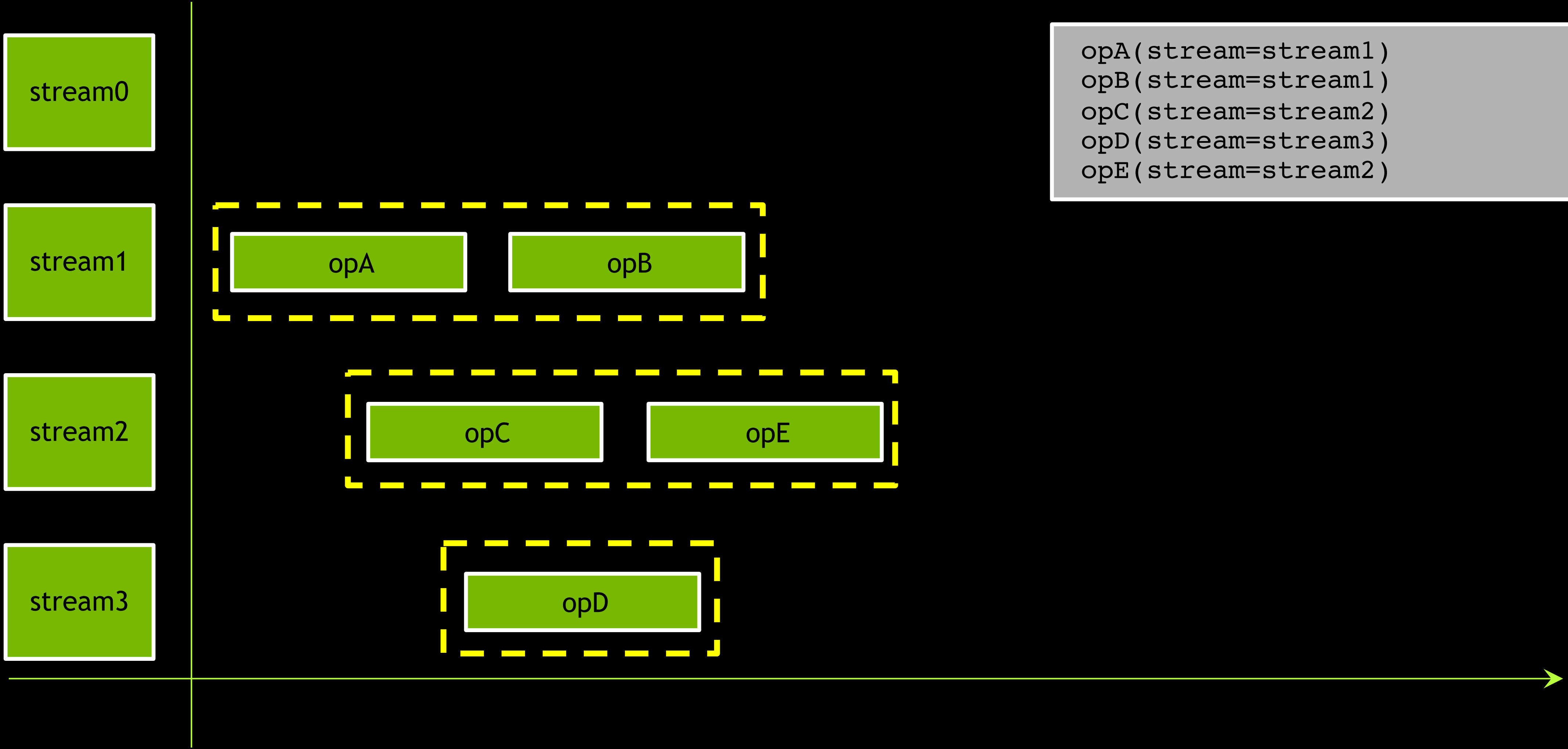
1. Operations issued into the same stream will execute in issue-order

```
opA(stream=stream1)
opB(stream=stream1)
opC(stream=stream2)
opD(stream=stream3)
opE(stream=stream2)
```



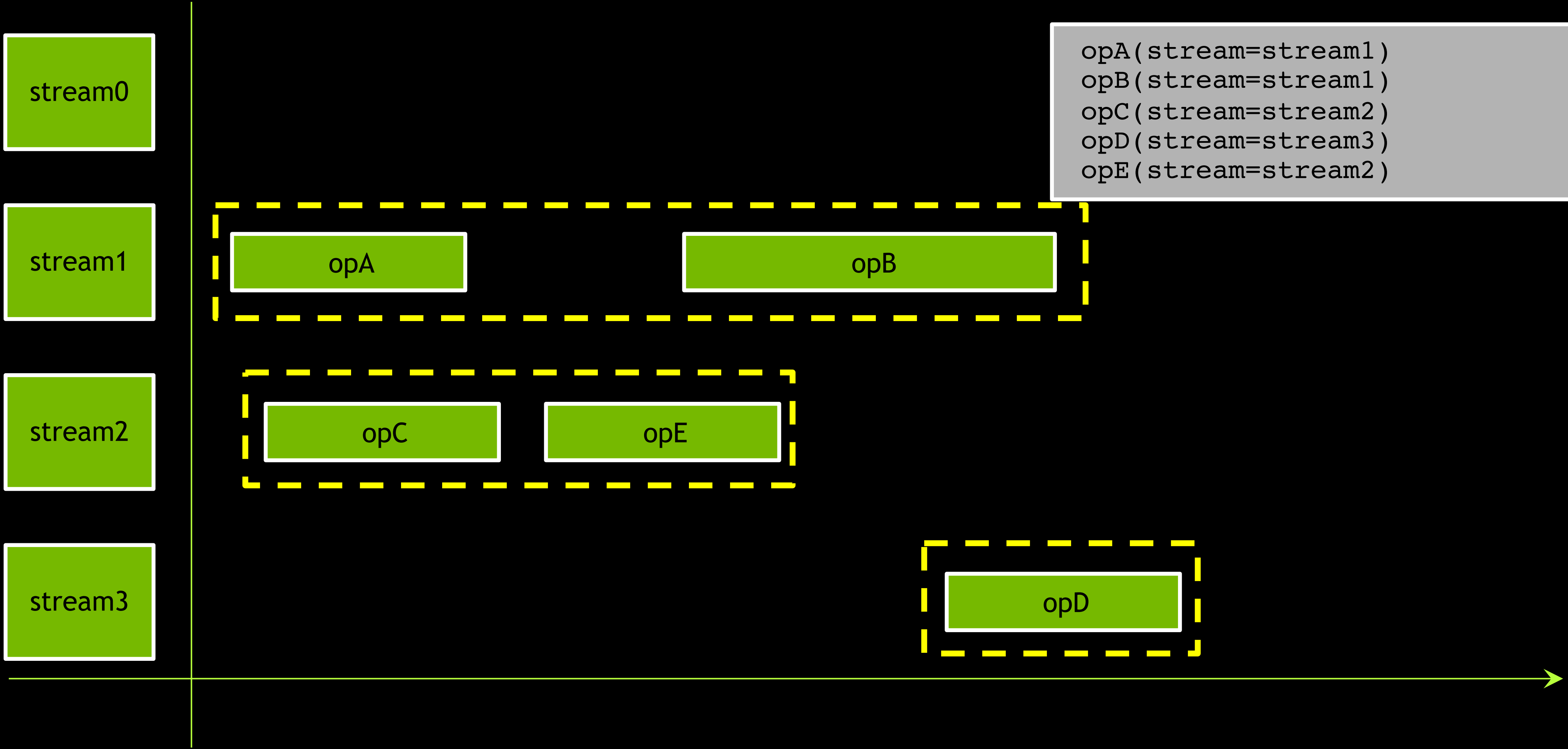
2. Operations in different non-default streams have no fixed order

```
opA(stream=stream1)
opB(stream=stream1)
opC(stream=stream2)
opD(stream=stream3)
opE(stream=stream2)
```



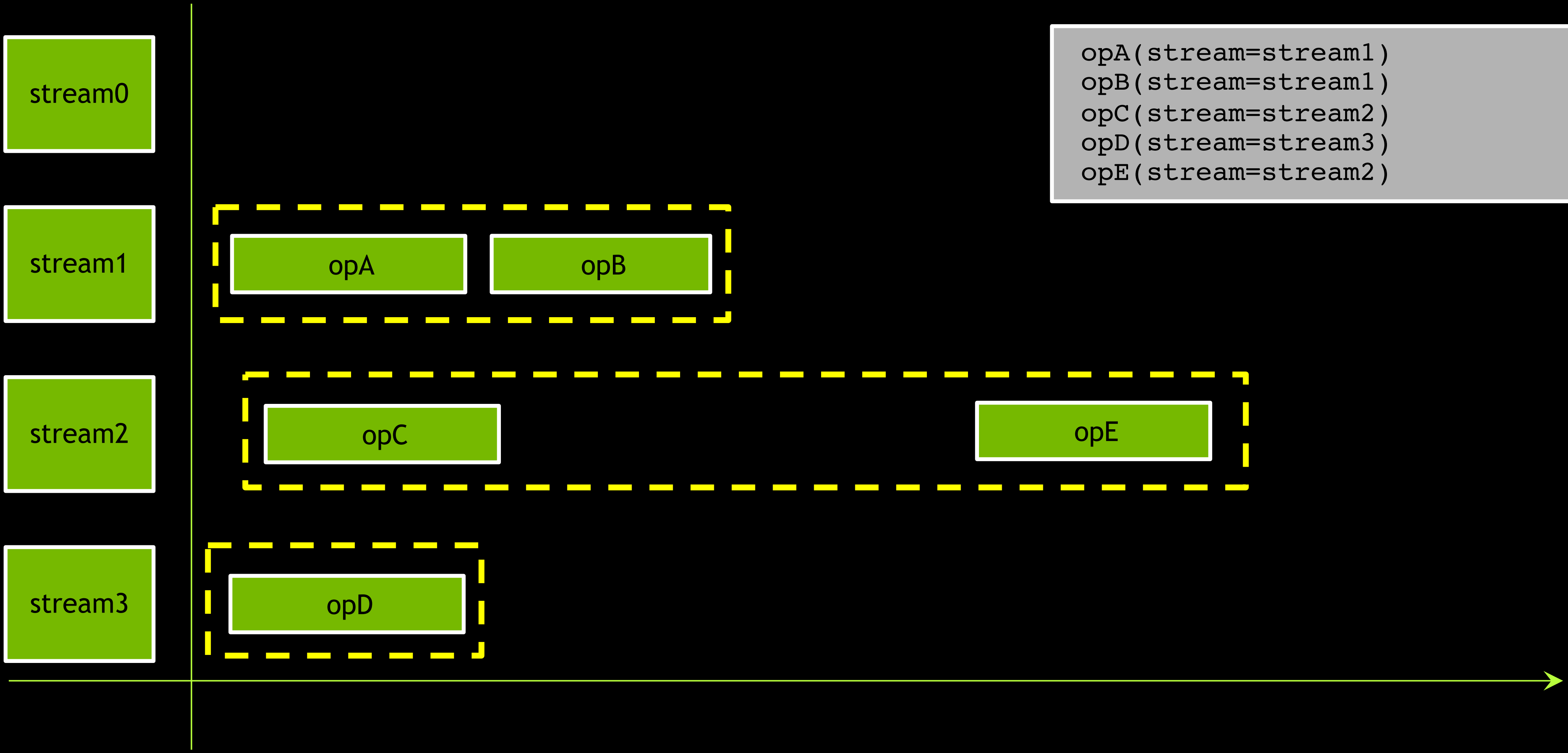
2. Operations in different non-default streams have no fixed order

```
opA(stream=stream1)
opB(stream=stream1)
opC(stream=stream2)
opD(stream=stream3)
opE(stream=stream2)
```



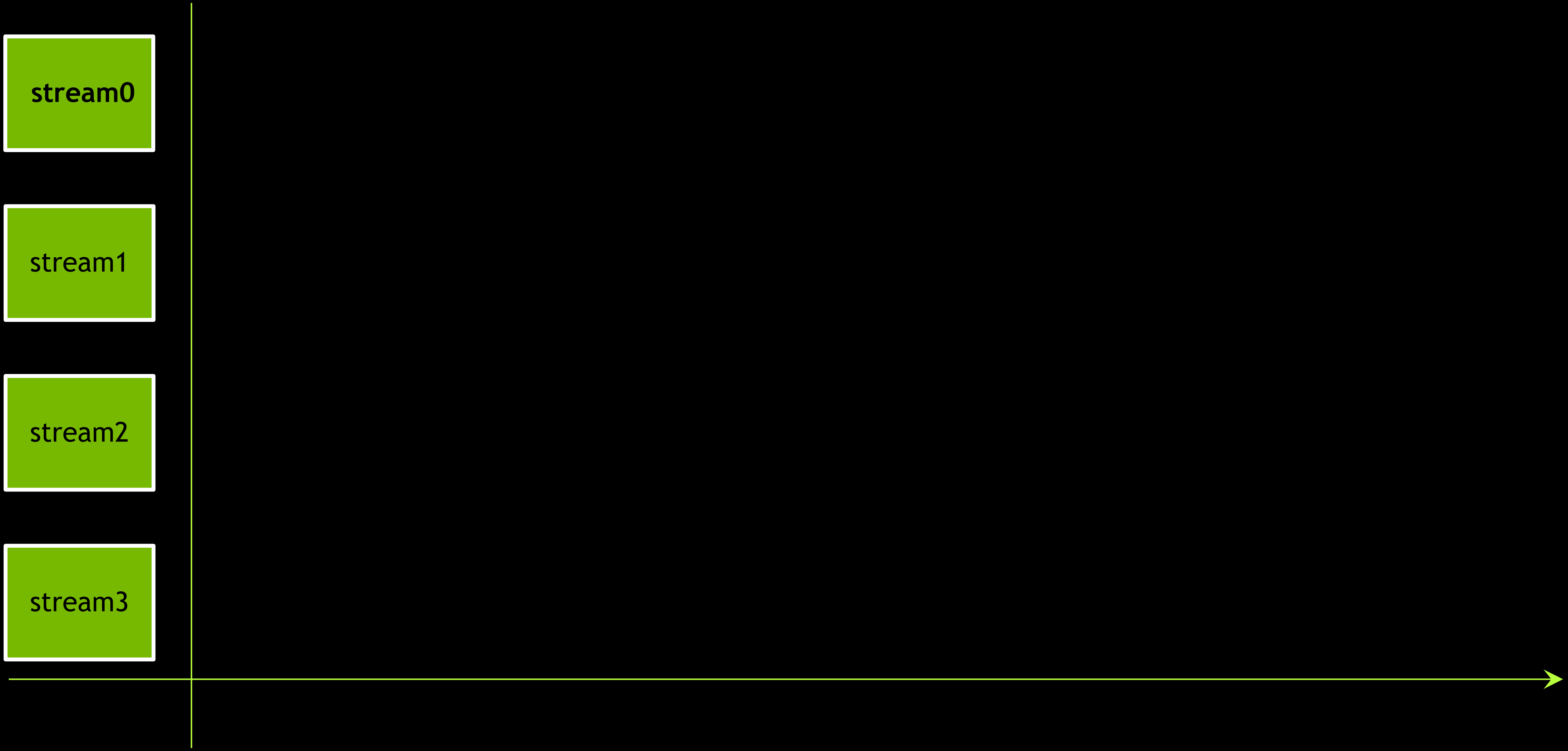
2. Operations in different non-default streams have no fixed order

```
opA(stream=stream1)
opB(stream=stream1)
opC(stream=stream2)
opD(stream=stream3)
opE(stream=stream2)
```



DEFAULT STREAM BEHAVIOR

The **default stream** is special



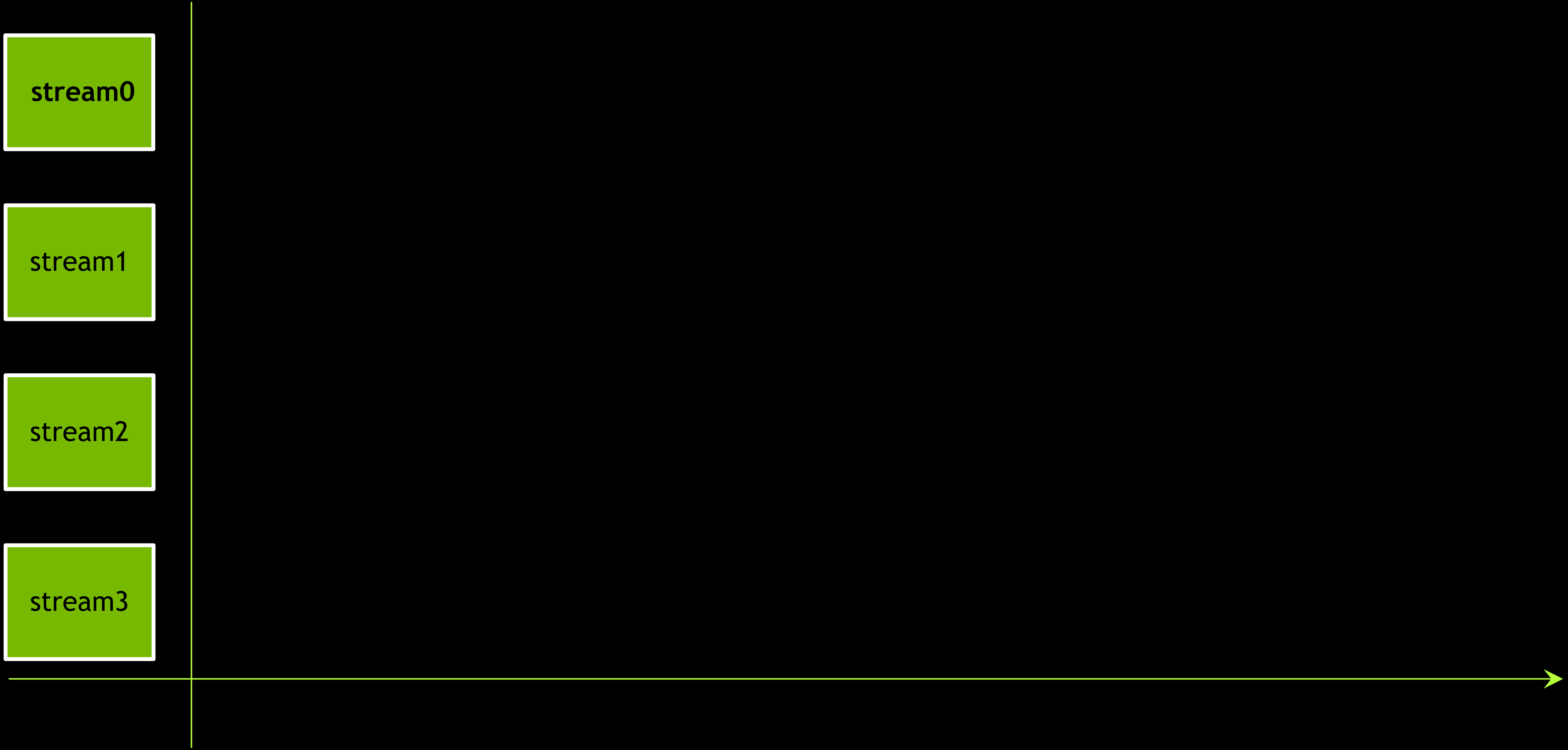
stream0

stream1

stream2

stream3

There can be no execution in any non-default streams at the same time as any execution in the default stream



stream0

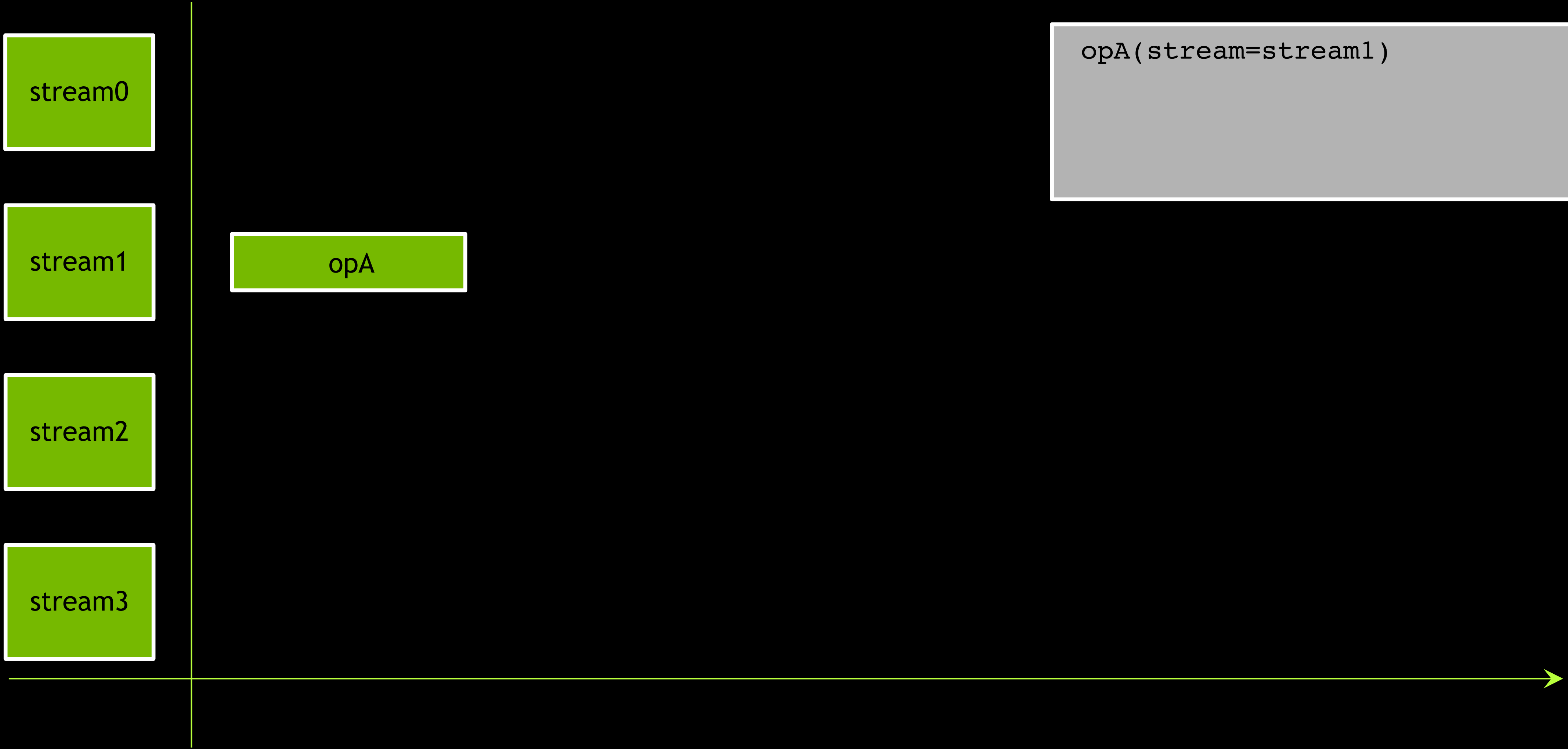
stream1

stream2

stream3

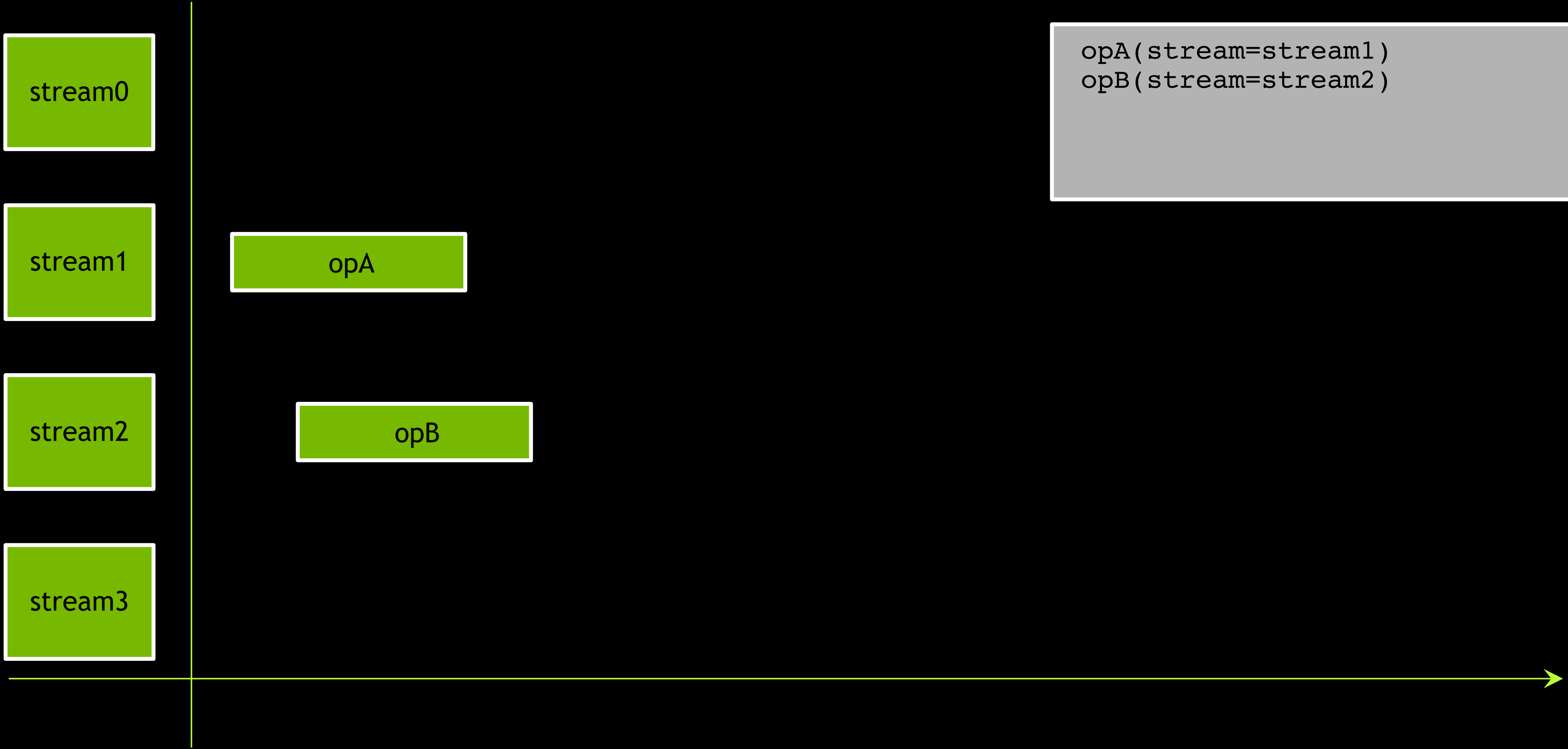
There can be no execution in any non-default streams at the same time as any execution in the default stream

```
opA(stream=stream1)
```



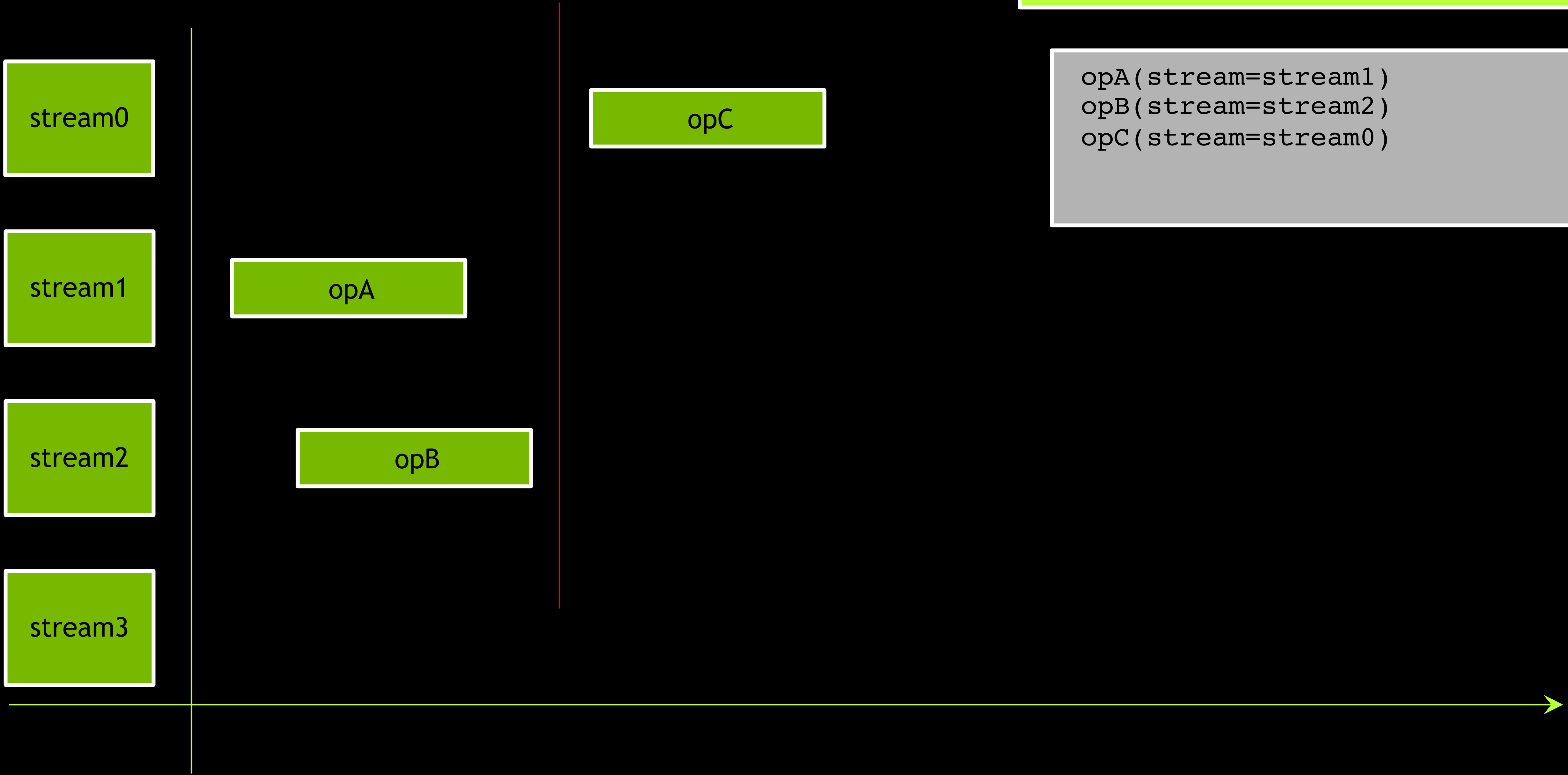
There can be no execution in any non-default streams at the same time as any execution in the default stream

```
opA(stream=stream1)
opB(stream=stream2)
```



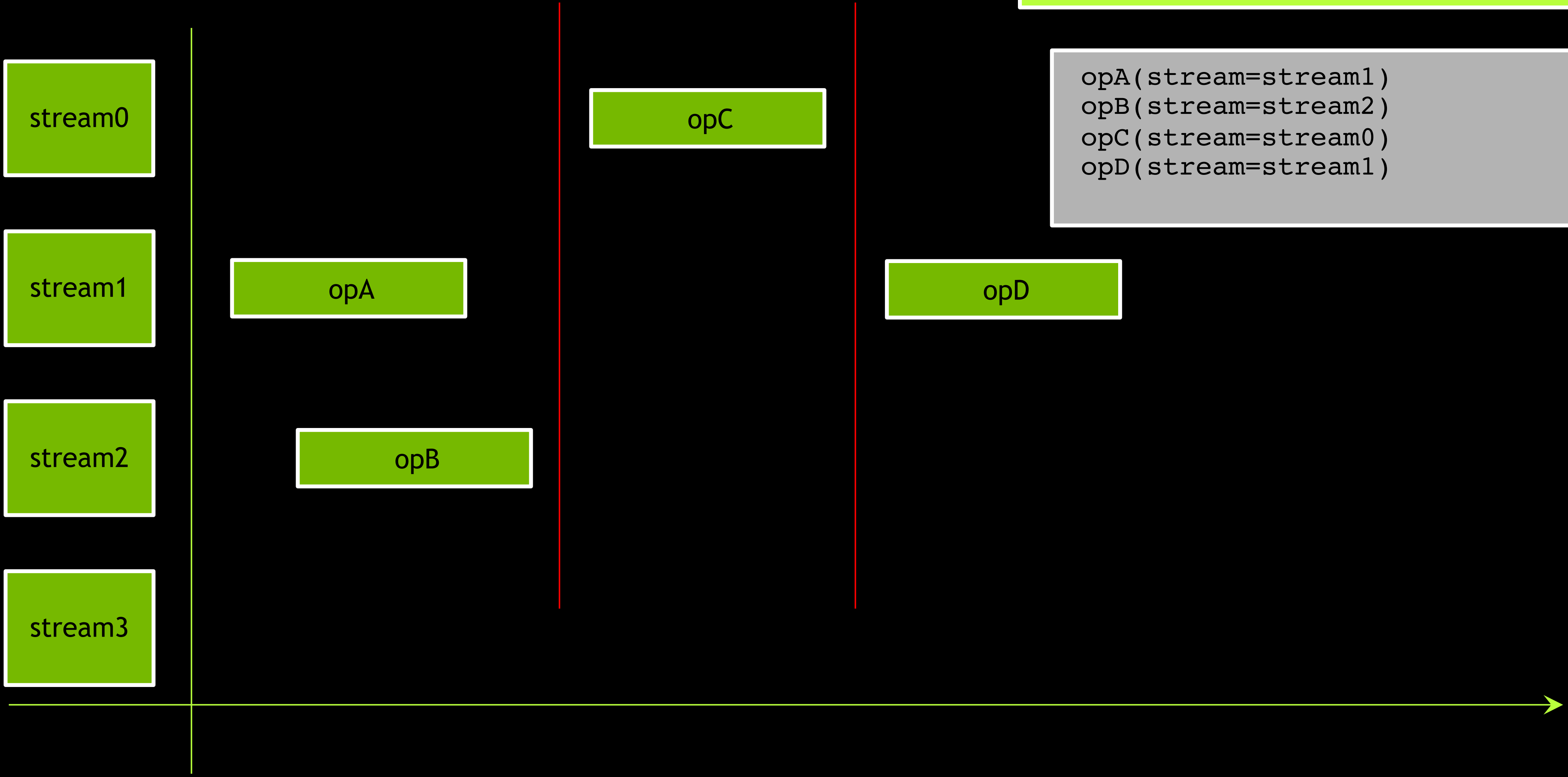
The default stream will both wait for all non-default stream execution to complete before beginning…

```
opA(stream=stream1)
opB(stream=stream2)
opC(stream=stream0)
```



…and must complete before any other non-default stream work can begin

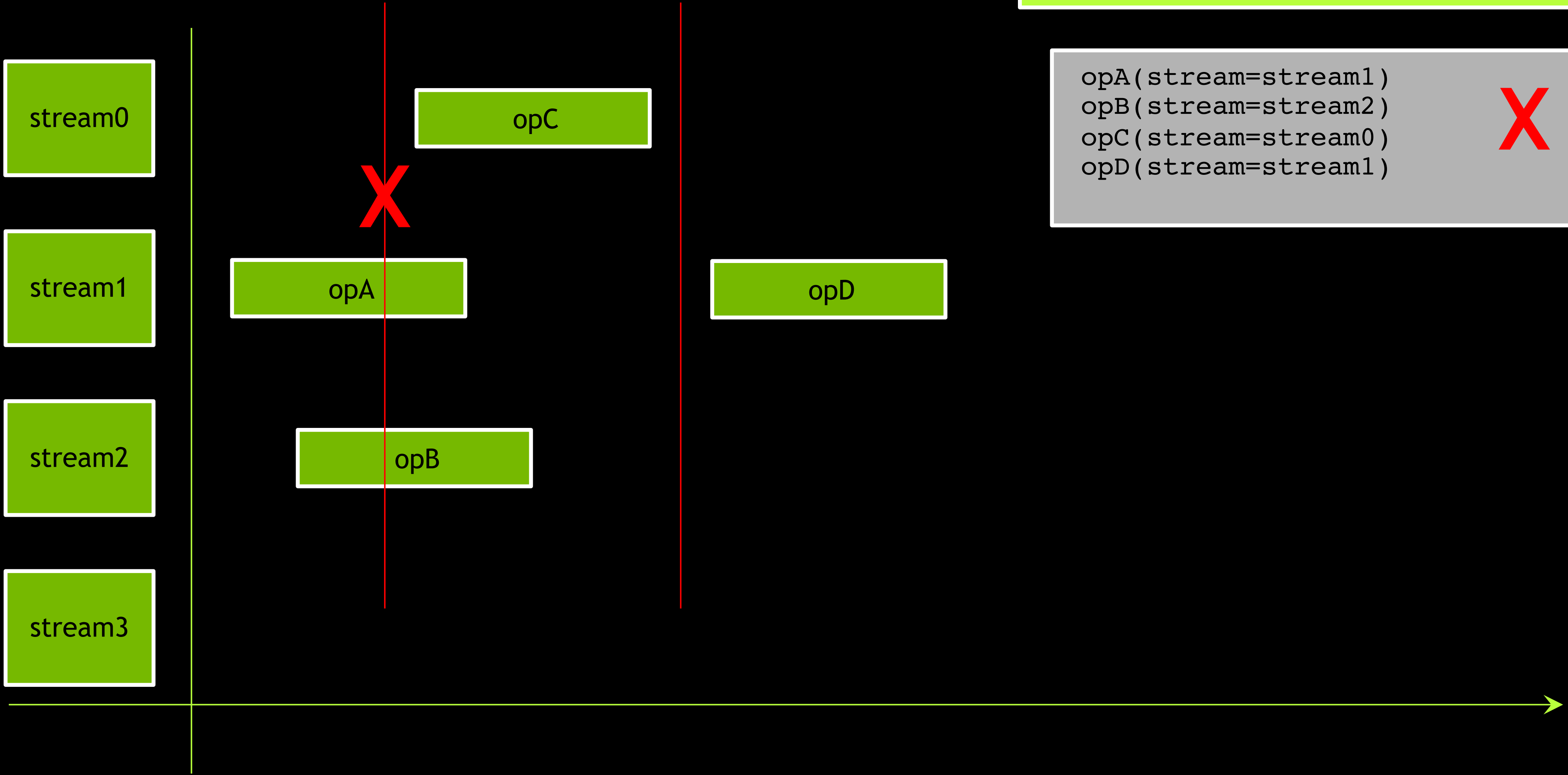
```
opA(stream=stream1)
opB(stream=stream2)
opC(stream=stream0)
opD(stream=stream1)
```



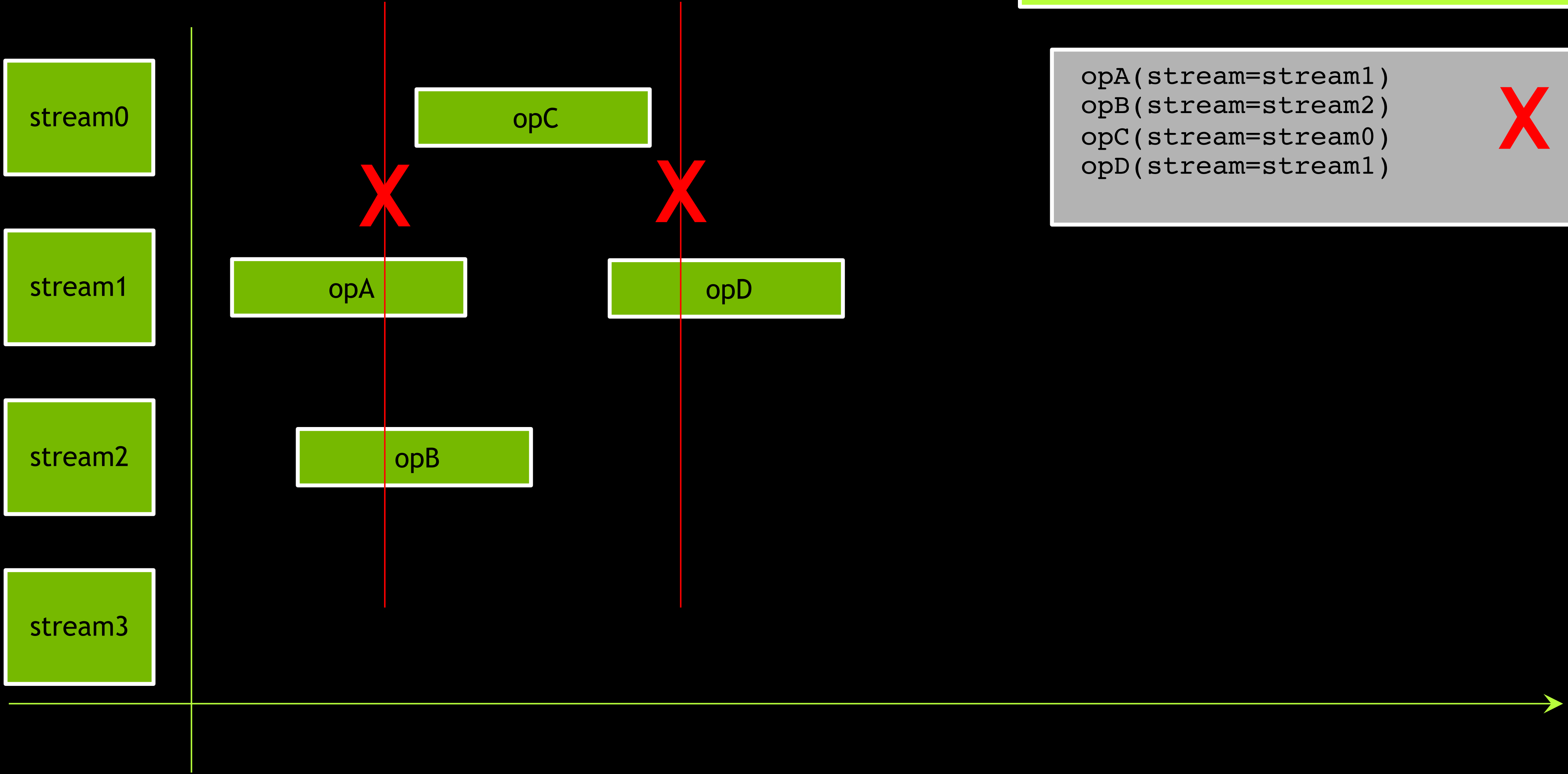
Default stream overlap with non-default streams cannot occur

```
opA(stream=stream1)
opB(stream=stream2)
opC(stream=stream0)
opD(stream=stream1)
```

X



Default stream overlap with non-default streams cannot occur



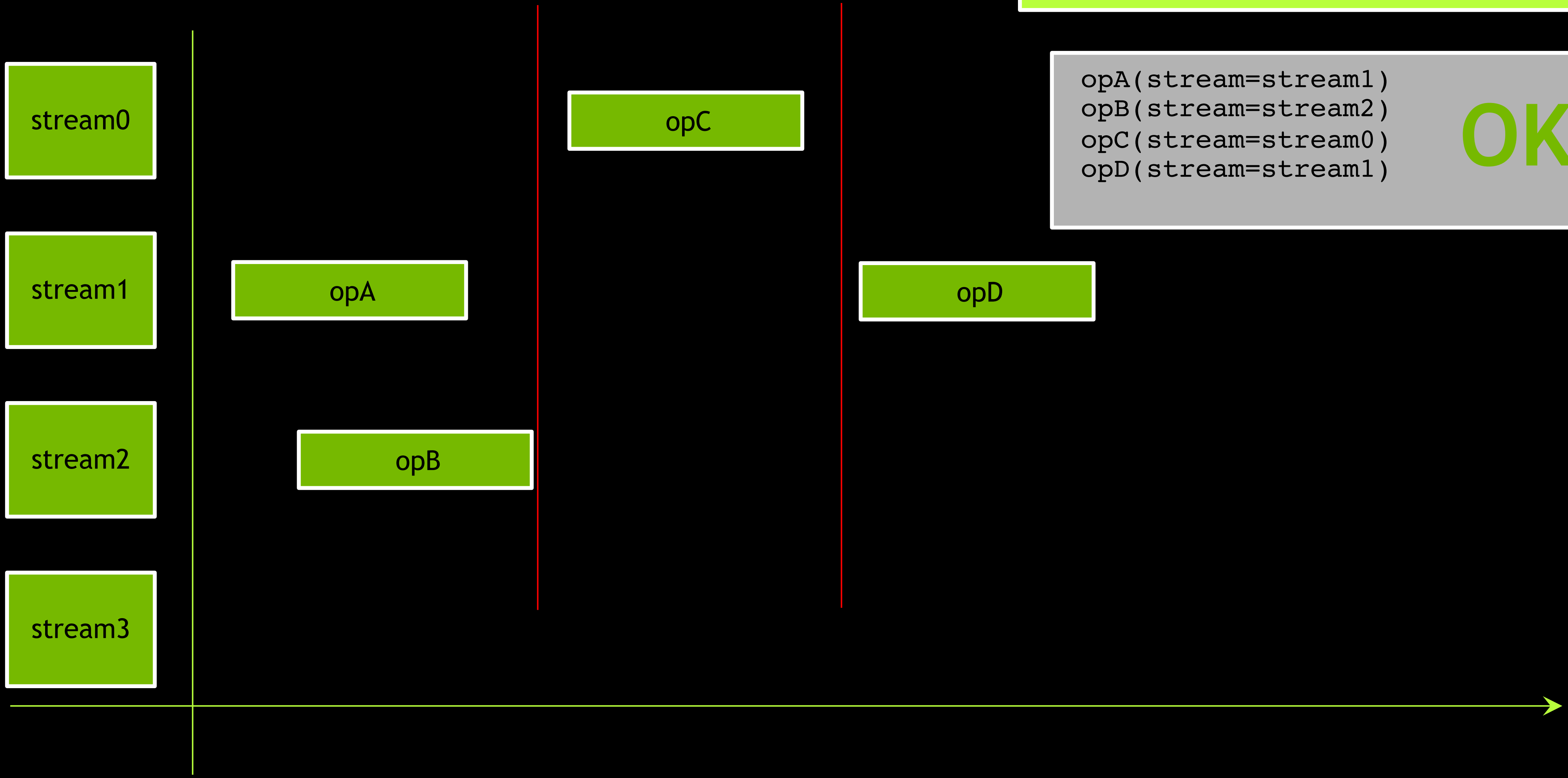
```
opA(stream=stream1)
opB(stream=stream2)
opC(stream=stream0)
opD(stream=stream1)
```

X

Default stream overlap with non-default streams cannot occur

```
opA(stream=stream1)
opB(stream=stream2)
opC(stream=stream0)
opD(stream=stream1)
```

OK



stream0

stream1

stream2

stream3

opA

opB

opC

opD

STREAMS IN CUDA PROGRAMMING

MANY CUDA RUNTIME FUNCTIONS EXPECT A STREAM ARGUMENT

MANY CUDA RUNTIME FUNCTIONS EXPECT A STREAM ARGUMENT

They all have a default value of 0, the default stream

MANY CUDA RUNTIME FUNCTIONS EXPECT A STREAM ARGUMENT

They all have a default value of 0, the default stream

Look for `cudaStream_t` in the CUDA Runtime API docs

MANY CUDA RUNTIME FUNCTIONS EXPECT A STREAM ARGUMENT

They all have a default value of 0, the default stream

Look for `cudaStream_t` in the CUDA Runtime API docs

We will be looking specifically at memory copies in non-default streams

KERNEL LAUNCHES ALWAYS TAKE PLACE IN STREAMS

KERNEL LAUNCHES ALWAYS TAKE PLACE IN STREAMS

When launched they have a default value of 0, the default stream

KERNEL LAUNCHES ALWAYS TAKE PLACE IN STREAMS

When launched they have a default value of 0, the default stream

They can be launched in a non-default stream using the 4th launch configuration argument

KERNEL LAUNCHES ALWAYS TAKE PLACE IN STREAMS

When launched they have a default value of 0, the default stream

They can be launched in a non-default stream using the 4th launch configuration argument

```
kernel<<<grid, block, shared_memory, stream>>>()
```

NVIDIA

DEEP
LEARNING
INSTITUTE

www.nvidia.com/dli