OpenCL Library of Stream Memory Components Targeting FPGAs

Jasmina Vasiljević and Paul Chow
University of Toronto, Canada

Paul Schumacher, Fernando Martinez Vallina,
Jeff Fitfield and Ralph Wittig
Xiliknx Inc., San Jose, CA, USA

FPT 2015

Introduction

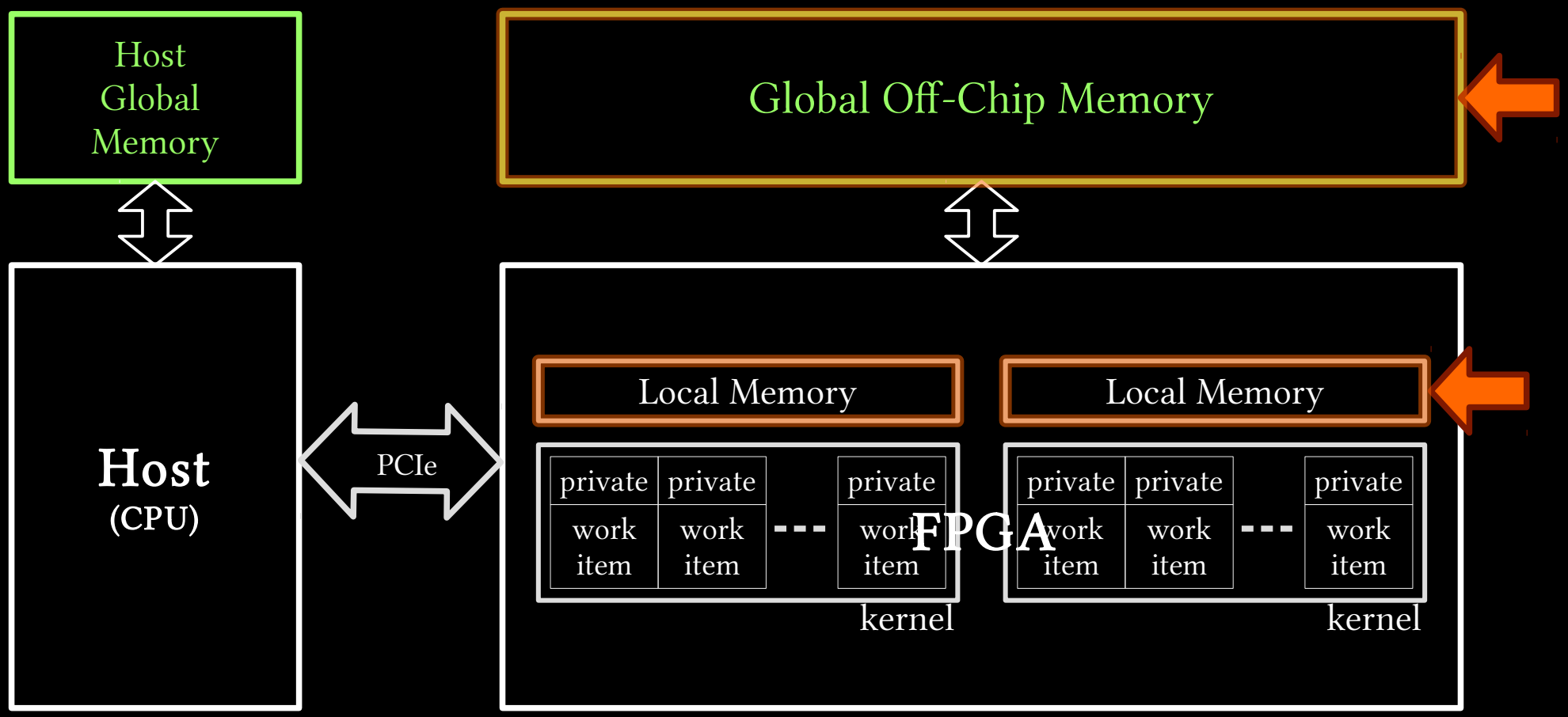
- Exciting time for FPGAs
  - Data center deployment
  - Stream applications
- Improvement in programming environment
  - OpenCL compilers for FPGAs
  - Xilinx SDAccel
- In a nutshell
  - A library of stream memory components
  - SDAccel environment

Introduction

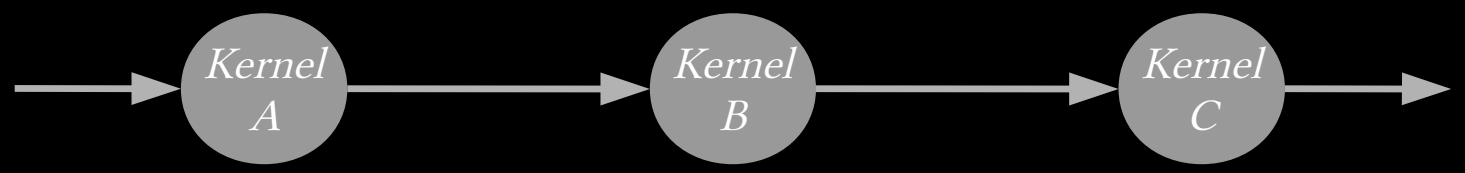
- *Stream applications on FPGAs*
  - Parallel processing pipelines & compute intensive
  - Use lots of memory
  - High throughput
  - *Steady and predictable data accesses*

- ***Library of commonly used memory building blocks***
  - *Stencil, tile, transpose*
  - *Pre-optimized data movement and buffering*
  - *Highly configurable*

OpenCL Memory Model

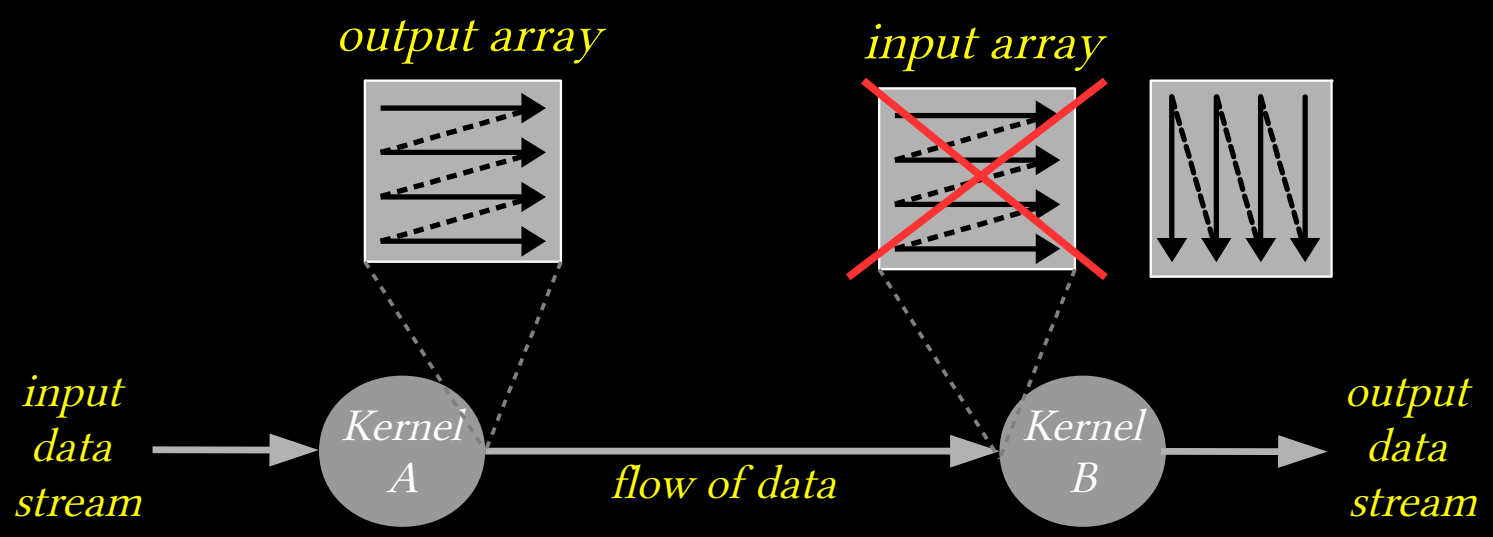


Stream Applications

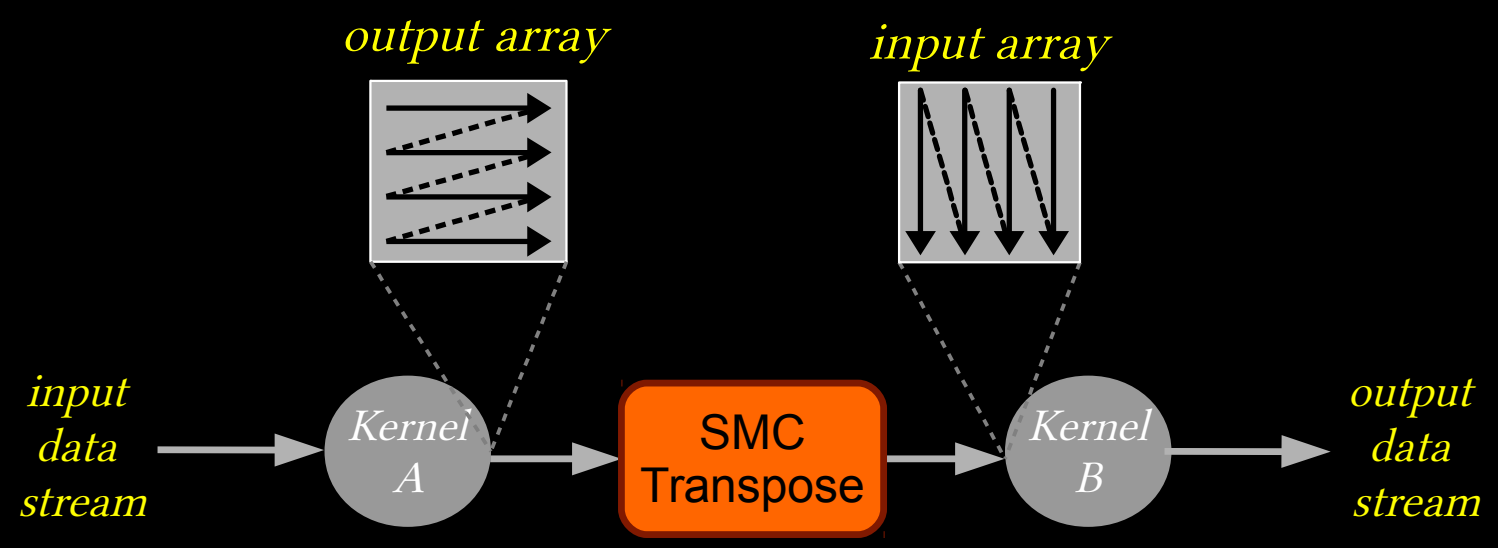


- DFG – Data Flow Graph
  - *Nodes* – compute kernels
  - *Edges* – flow of data

Stream Applications



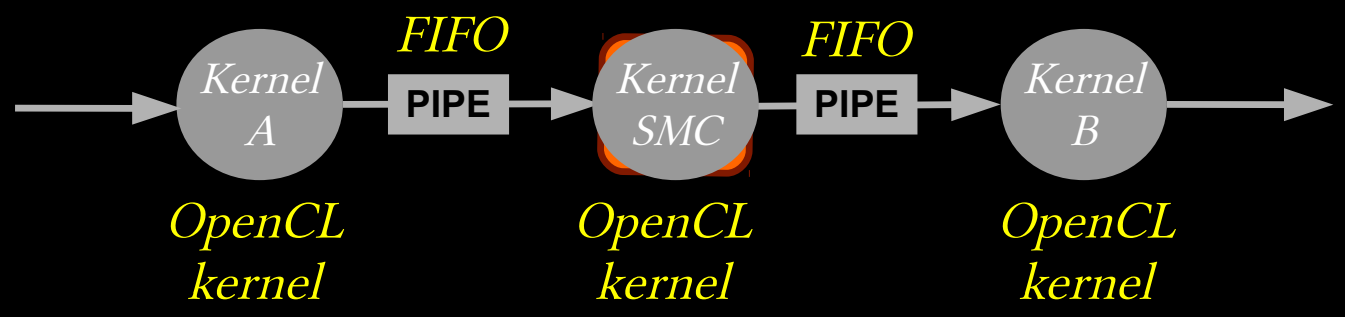
Stream Applications



- SMC – Stream Memory Component
  - Buffers and re-formats data

Stream Memory Components

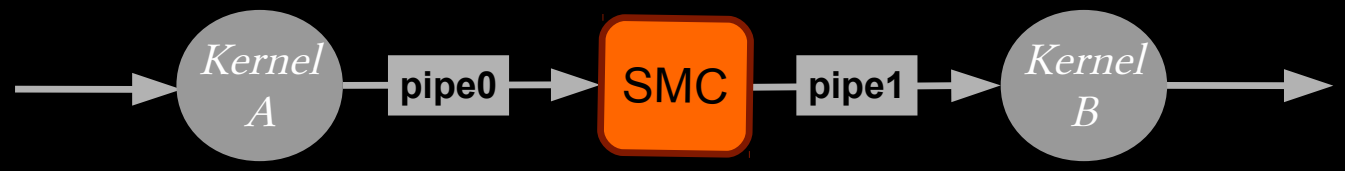
- OpenCL -



- SMC is implemented as an OpenCL kernel
- OpenCL ***pipe object***
  - a FIFO implemented in BRAMs

Stream Memory Components

- OpenCL -



Host Code

```
0. #include "smc.h"
1.
2. cl_mem pipe0;
3. cl_mem pipe1;
4.
5. clEnqueueNDRangeKernel(KernelA, ...);
6.
7. smcEnqueueTranspose(p0, p1, p2, ...);
8.
9. clEnqueueNDRangeKernel(KernelB, ...);
```

include the SMC header

create two pipes

launch Kernel A

launch SMC

launch Kernel B

The SMC Library

The SMC Library

- User Configurations -

1) *Common data transformations*

2) *Input/output configuration*

3) *Memory resources vs. performance trade-off*

The SMC Library

- User Configurations -

1) ***Common data transformations***

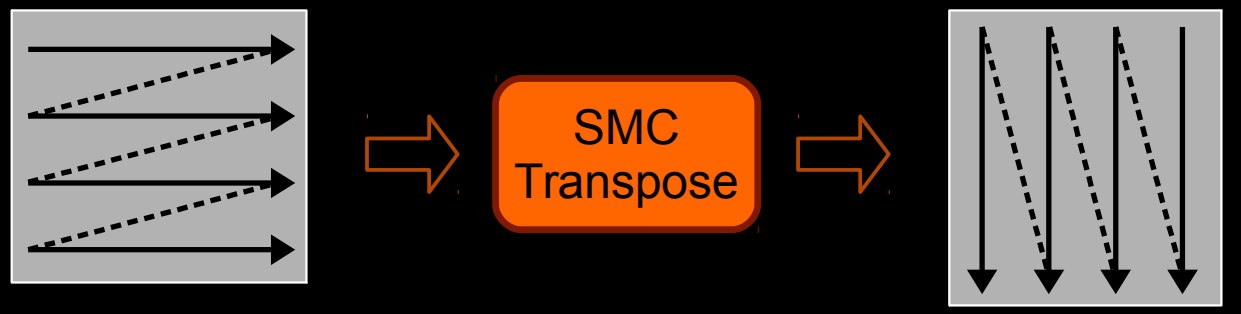
2) *Input/output configuration*

3) *Memory resources vs. performance trade-off*

Stream Memory Components

- Common Data Transformations -

1. **Transpose**
2.
3.

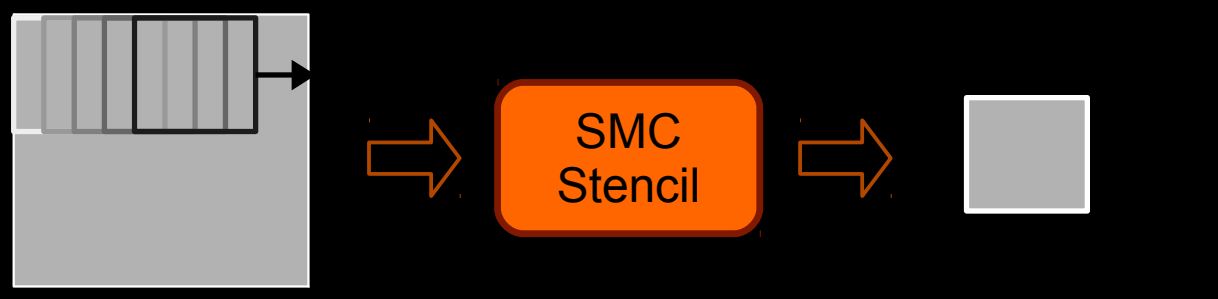


frame in *frame out*

Stream Memory Components

- Common Data Transformations -

1. Transpose
2. **Stencil**
3.

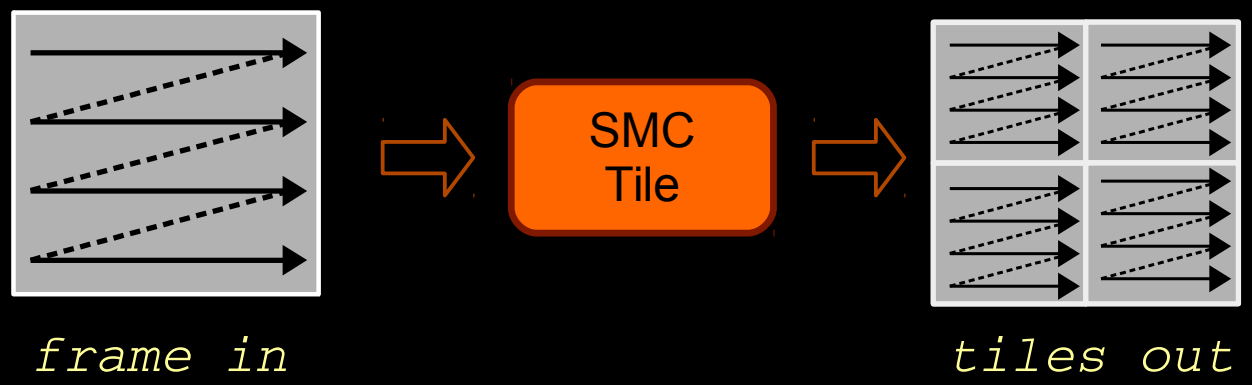


frame in *window out*

Stream Memory Components

- Common Data Transformations -

1. Transpose
2. Stencil
3. **Tile**



The SMC Library

- User Configurations -

1) *Common data transformations*

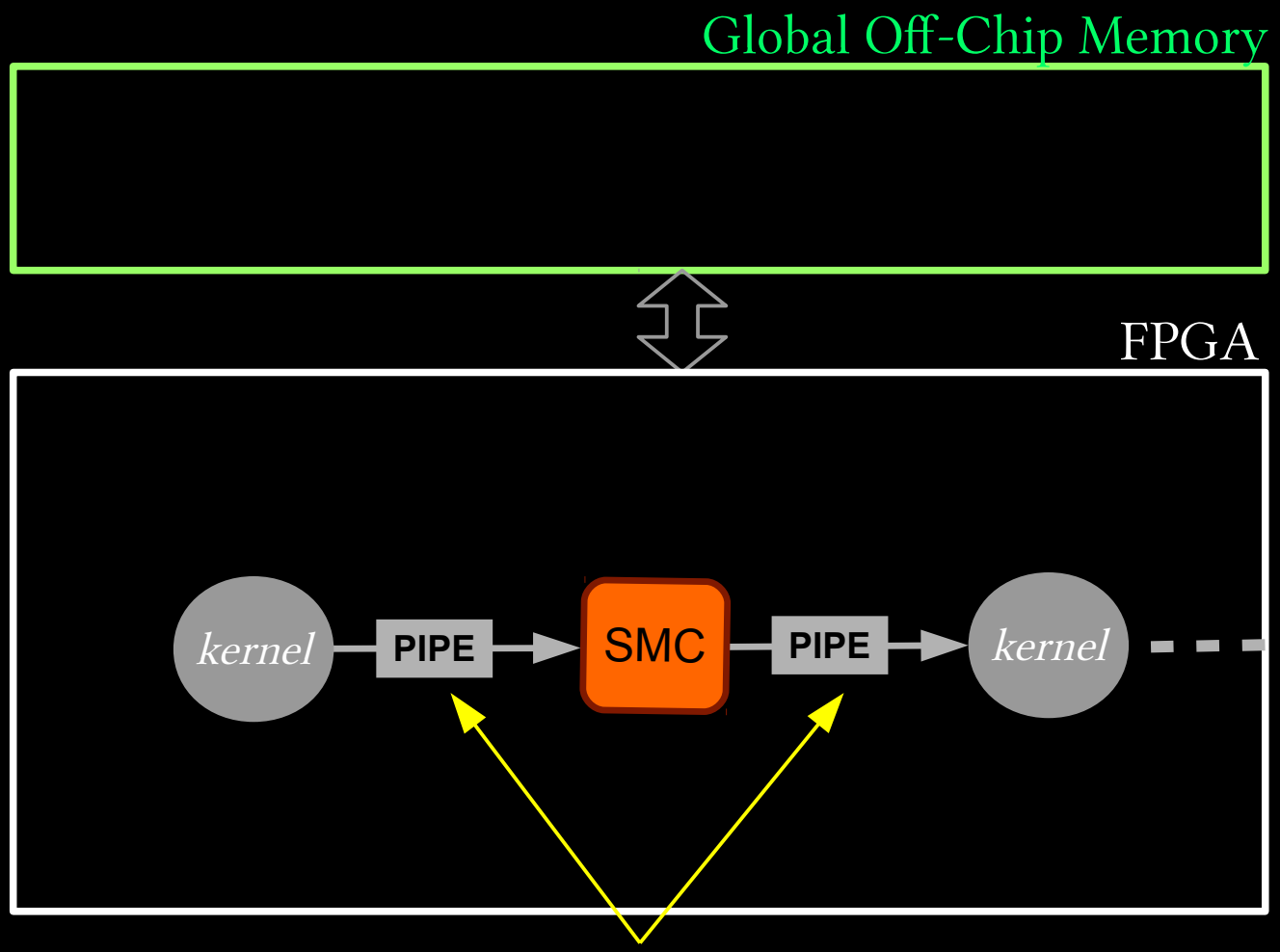
2) ***Input/output configuration***

3) *Memory resources vs. performance trade-off*

Stream Memory Components

- Input & Output Configuration -

1. **PIPE to PIPE**
2.
3.
4.

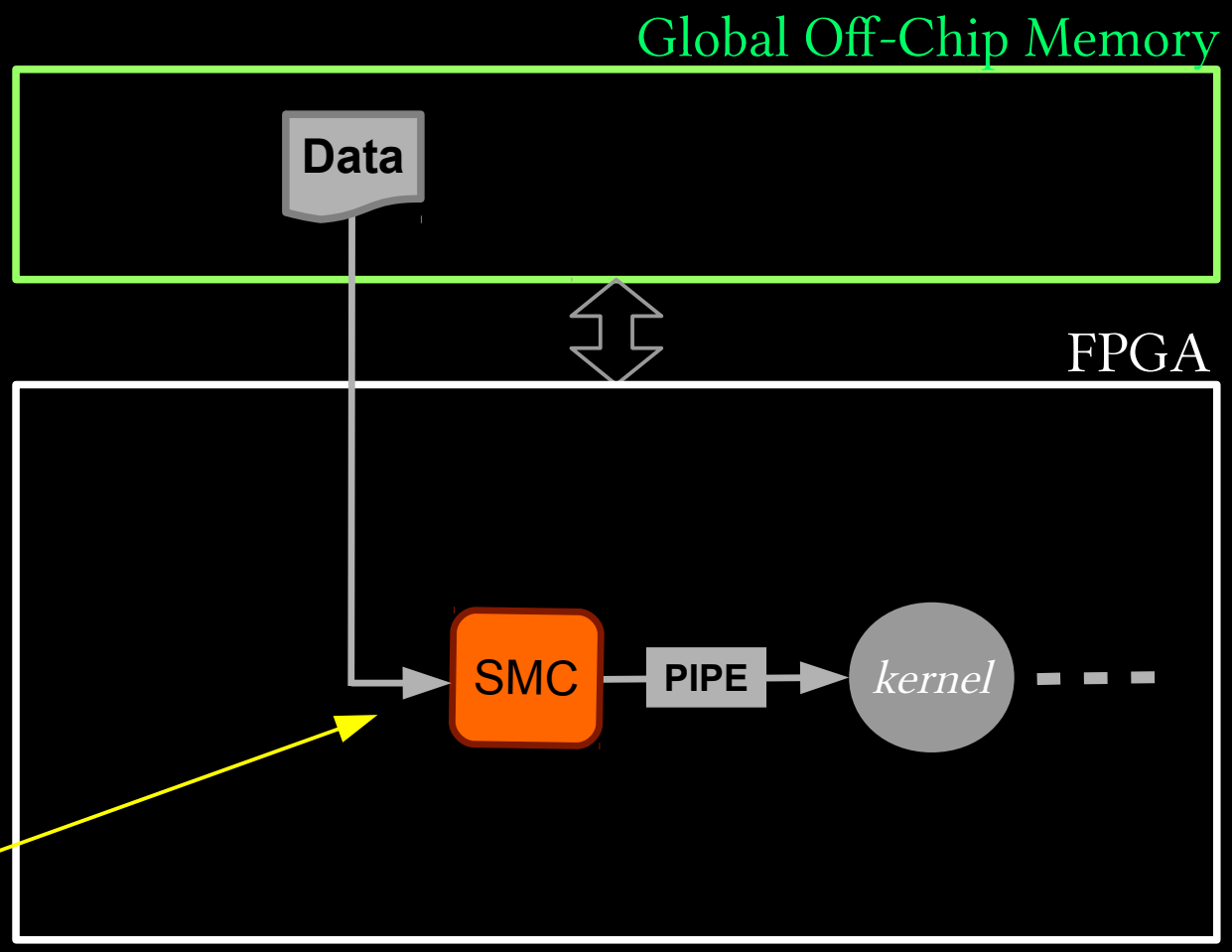


- low latency read/write
- BRAMs
- random access

Stream Memory Components

- Input & Output Configuration -

1. PIPE to PIPE
2. **GLOBAL to PIPE**
3.
4.

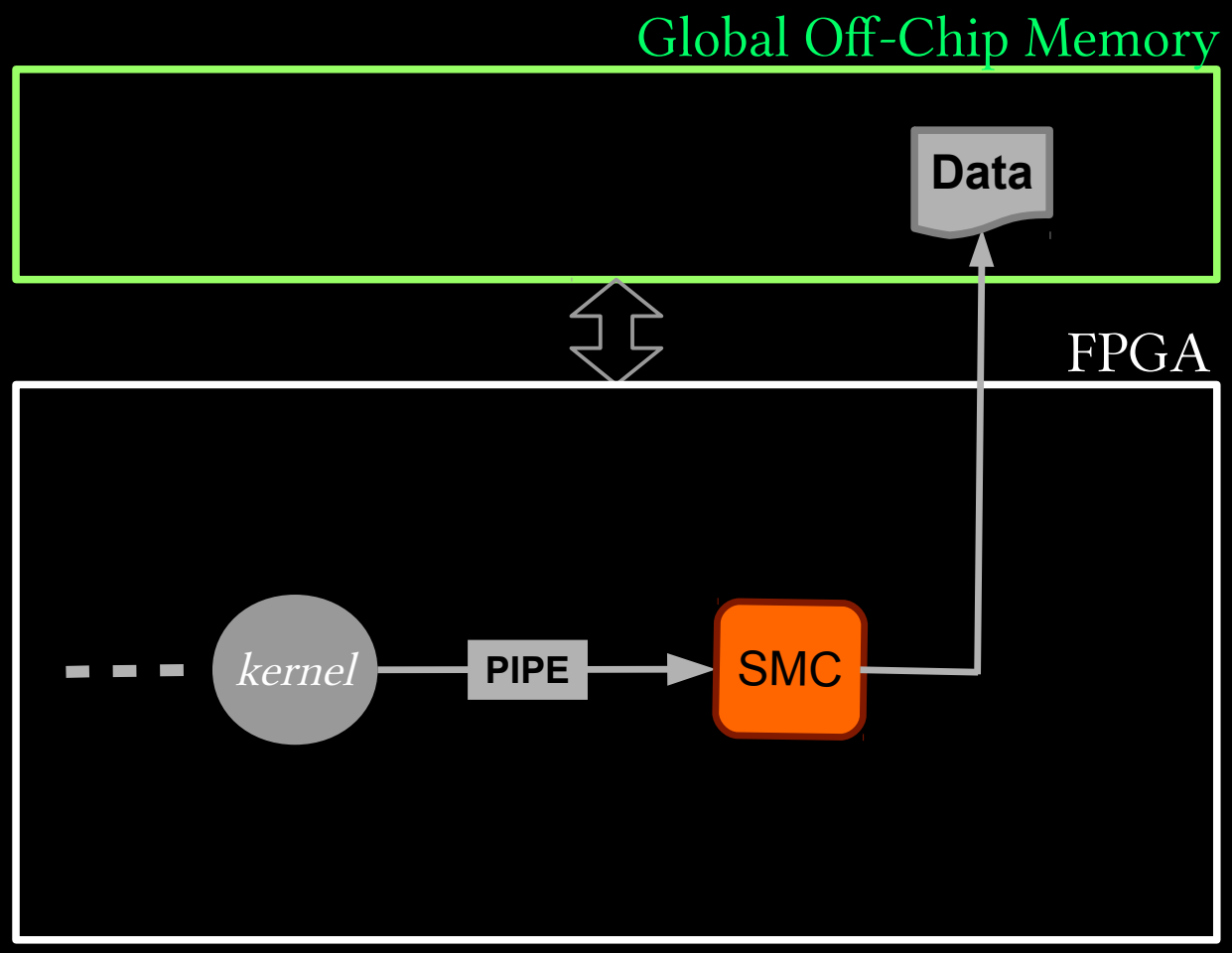


- high access latency
- data access pattern sensitive
- burst read/writes maximize performance

Stream Memory Components

- Input & Output Configuration -

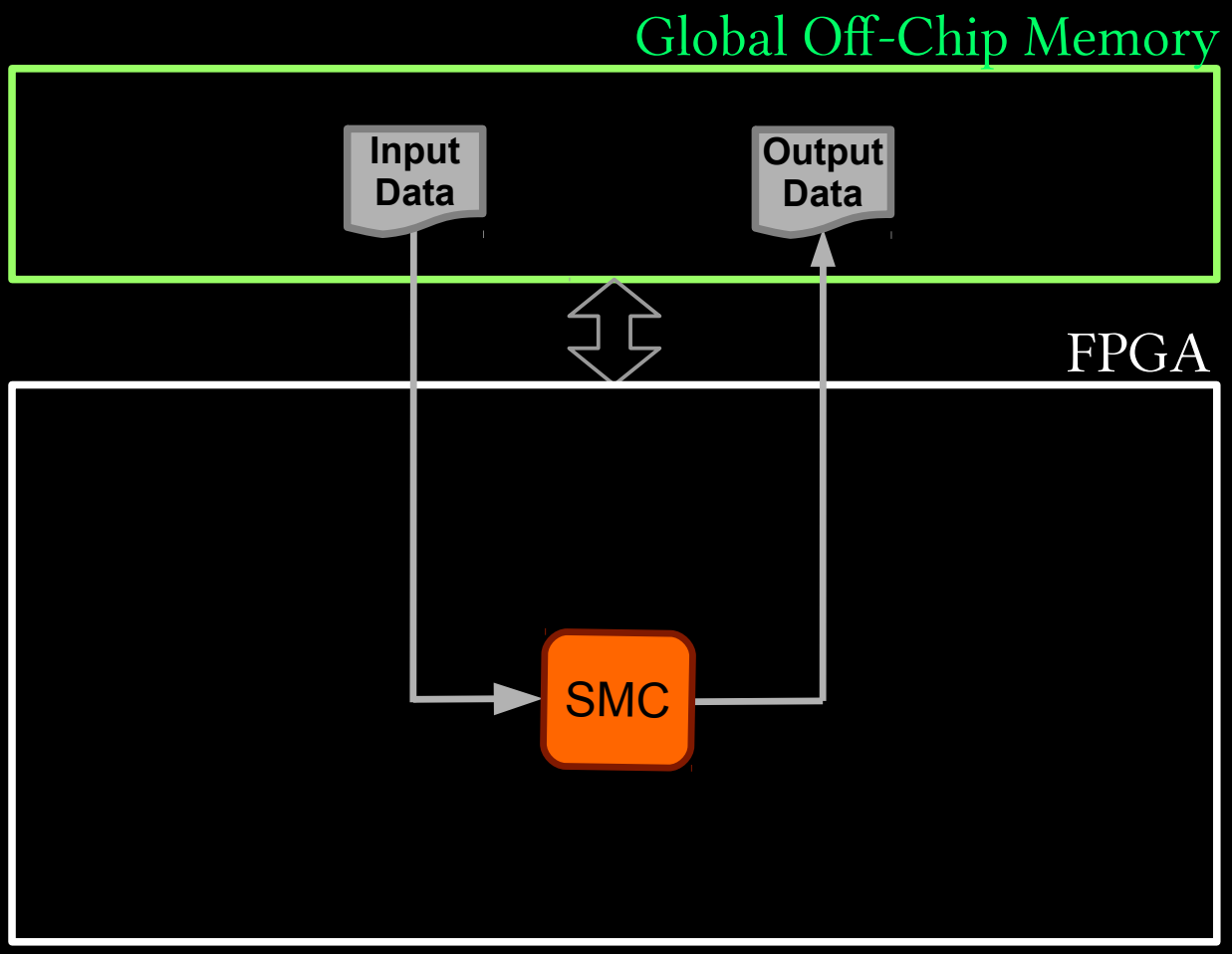
1. PIPE to PIPE
2. GLOBAL to PIPE
3. **PIPE to GLOBAL**
4.



Stream Memory Components

- Input & Output Configuration -

1. PIPE to PIPE
2. GLOBAL to PIPE
3. PIPE to GLOBAL
4. **GLOBAL to GLOBAL**



The SMC Library

- User Configurations -

1) *Common data transformations*

2) *Input/output configuration*

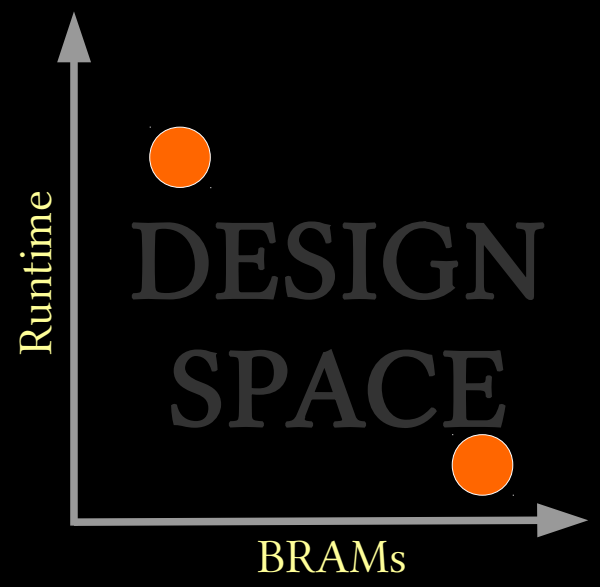
3) ***Memory resources vs. performance trade-off***

Stream Memory Components

- Memory Resources vs. Performance -

- Tradeoff between BRAMs used and performance of the SMC

- Two optimization approaches:
  1) *Adjusting the size of burst accesses from global memory*
  2) *Mapping to heterogeneous OpenCL memory regions*



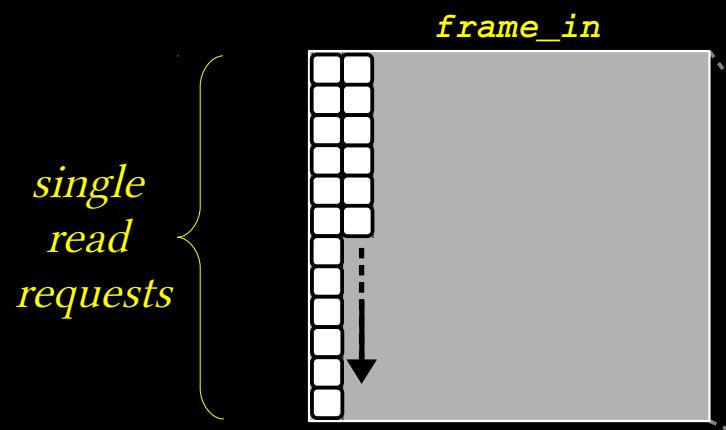
Host Code

```
smcEnqueueTranspose(...,128,...);
```

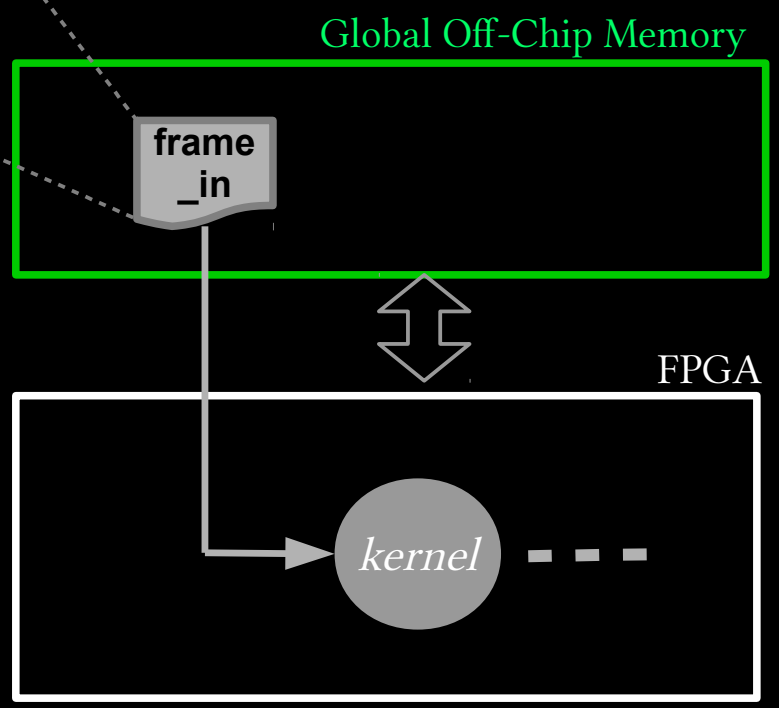
# of BRAMs

Stream Memory Components

- Memory Resources vs. Performance -



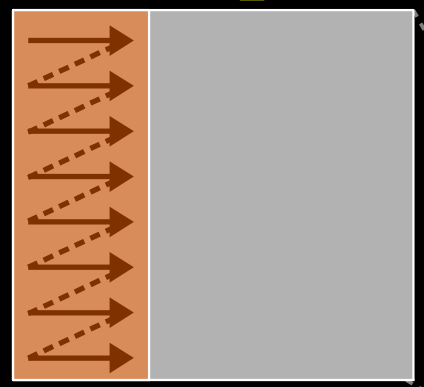

```
__kernel myKernel( __global int* frame_in
                   __global int* frame_out)

  for x=0 to H {
    for y=0 to W {

      frame_out[x][y] = frame_in[x][y] + C;

    }
  }
```



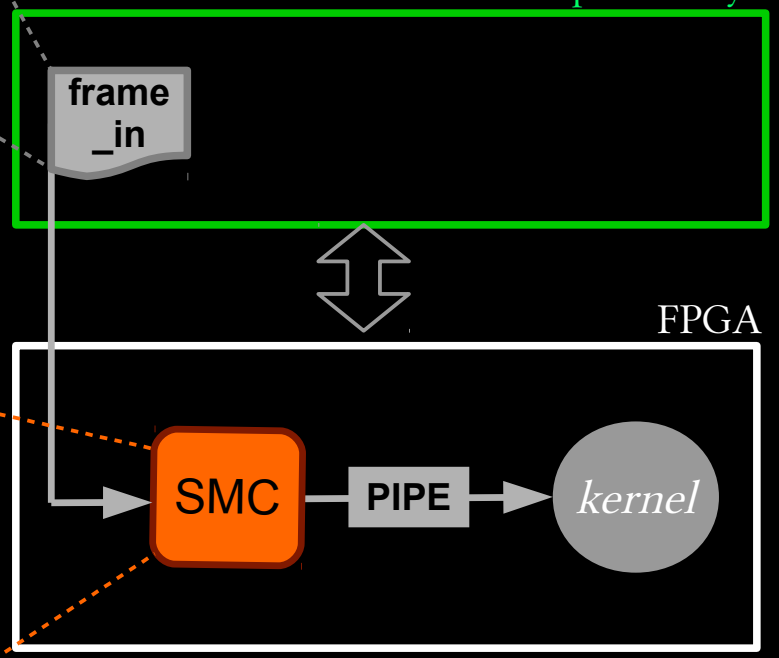
Stream Memory Components

- Memory Resources vs. Performance -

- 512-bit words
- burst reads

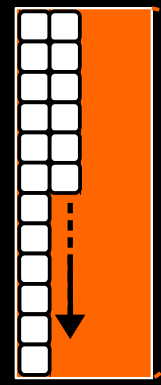



```
__kernel myKernel( __global int* output)

  for x=0 to H {
    for y=0 to W {

      read_pipe(smc_transpose, &temp);
      frame_out[x][y] = temp + C;

    }
  }
```



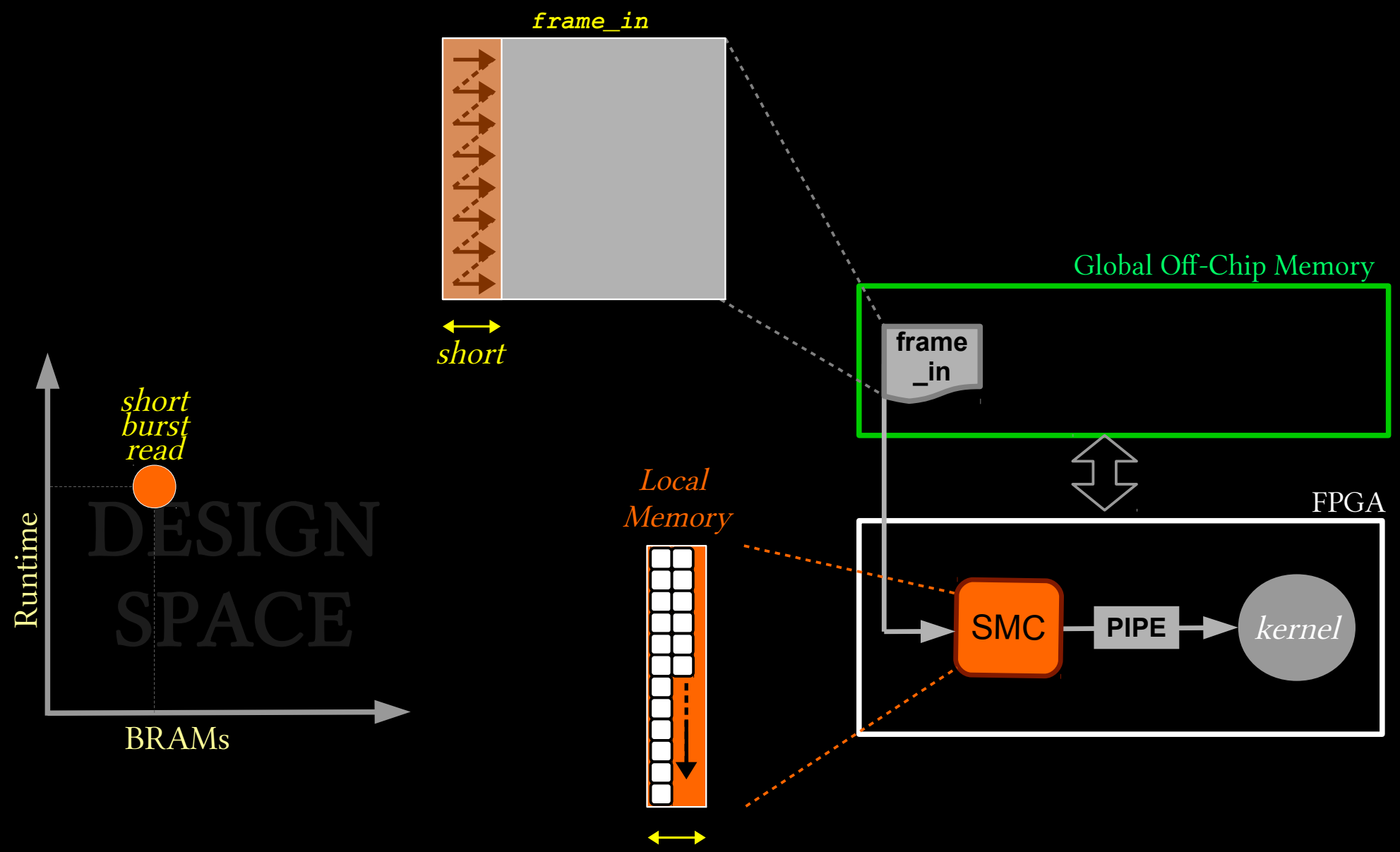
Local Memory

low-latency random access



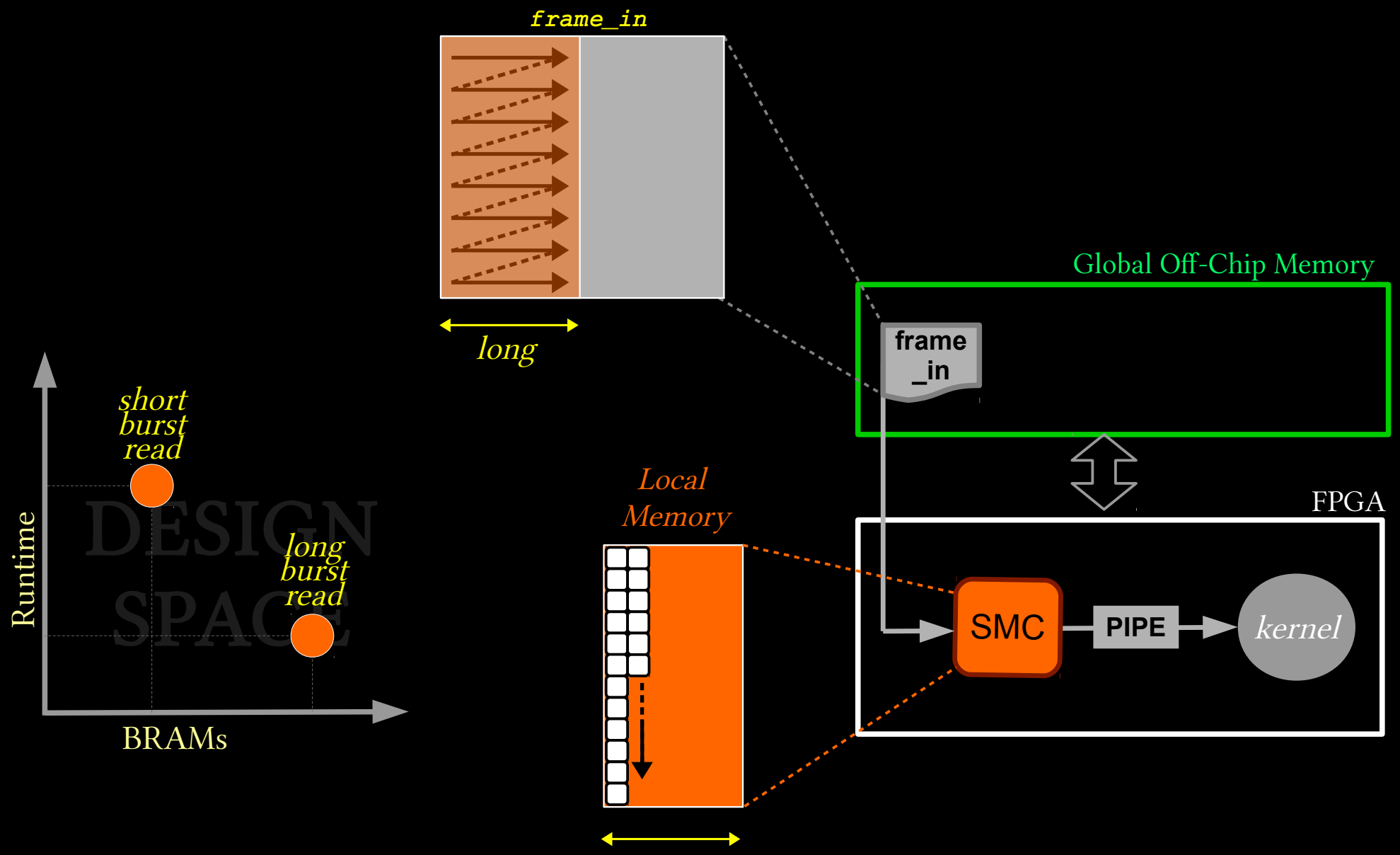
Stream Memory Components

- Memory Resources vs. Performance -



Stream Memory Components

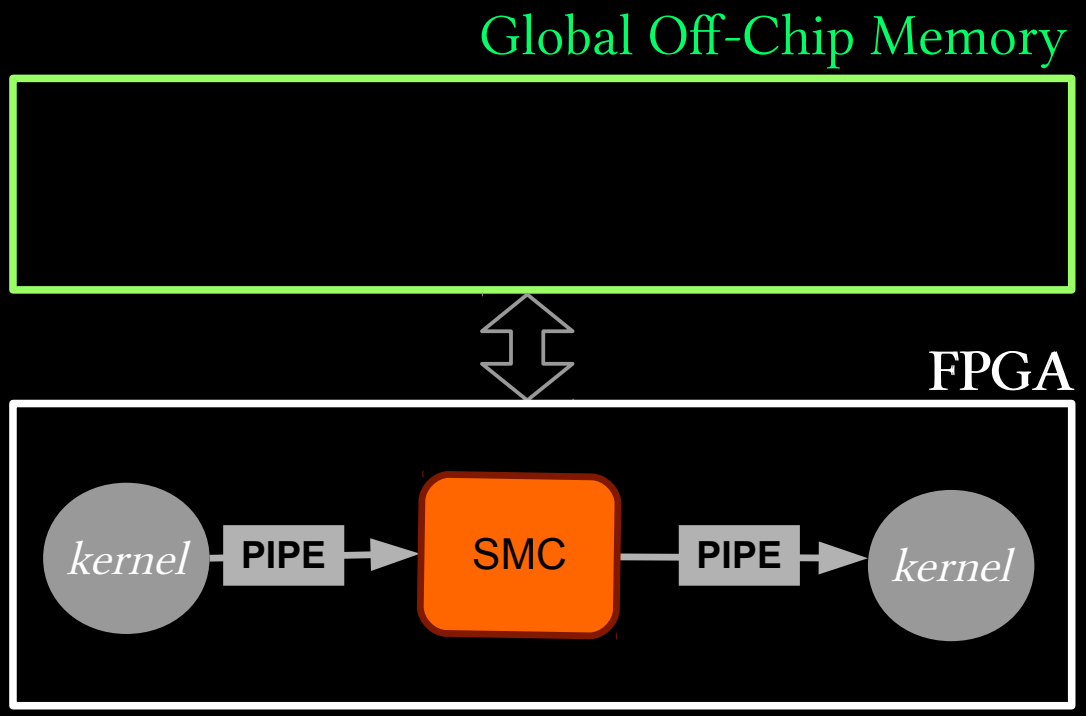
- Memory Resources vs. Performance -



Stream Memory Components

- Memory Resources vs. Performance -

- SMC needs to buffer data
- Buffering size:
  - SMC type
  - data size
  - performance



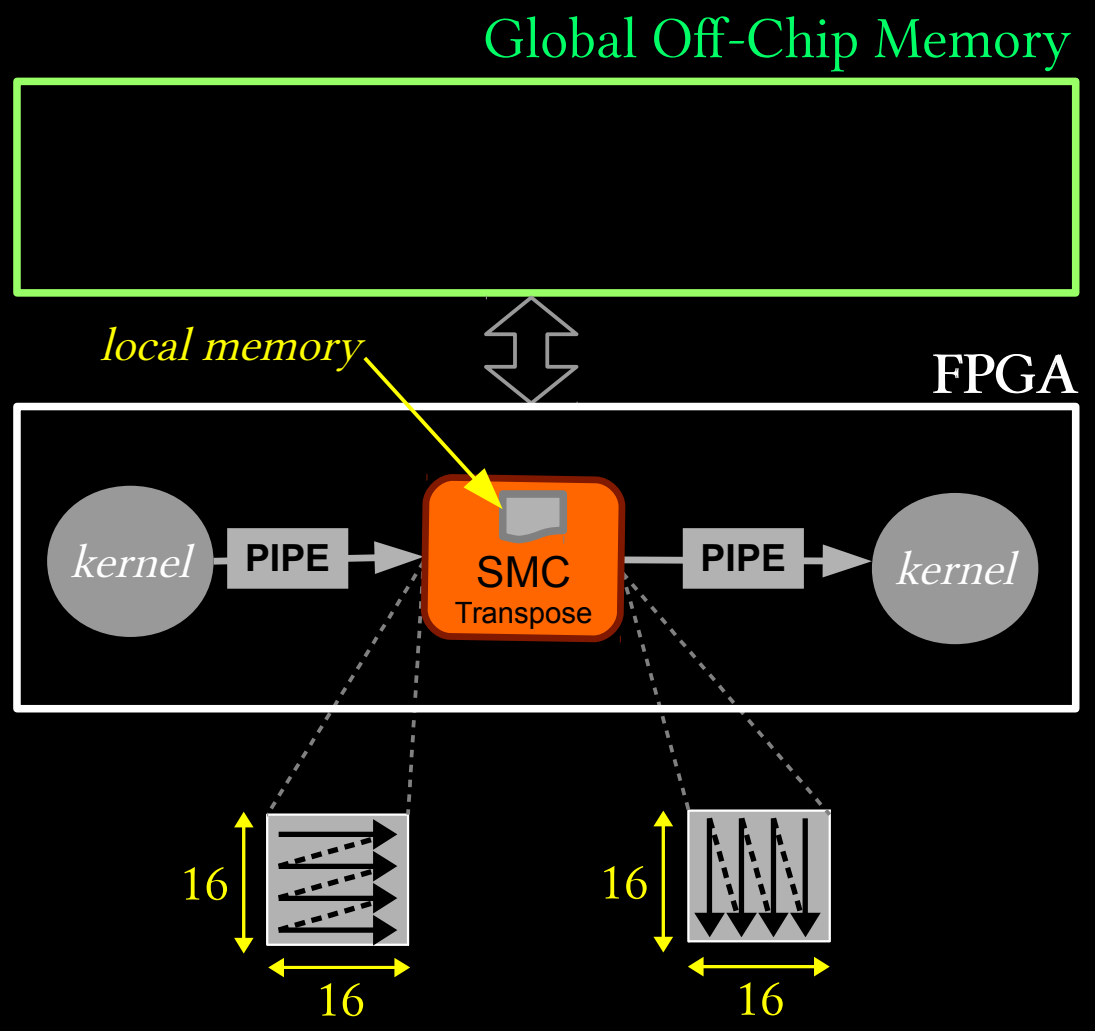
Stream Memory Components

- Memory Resources vs. Performance -

Example:

- SMC transpose
- PIPE to PIPE config
- Data size: 16x16

256 words
1 BRAM of storage



Host Code

```
smcEnqueueTranspose(...,16,16,...);
```

data size

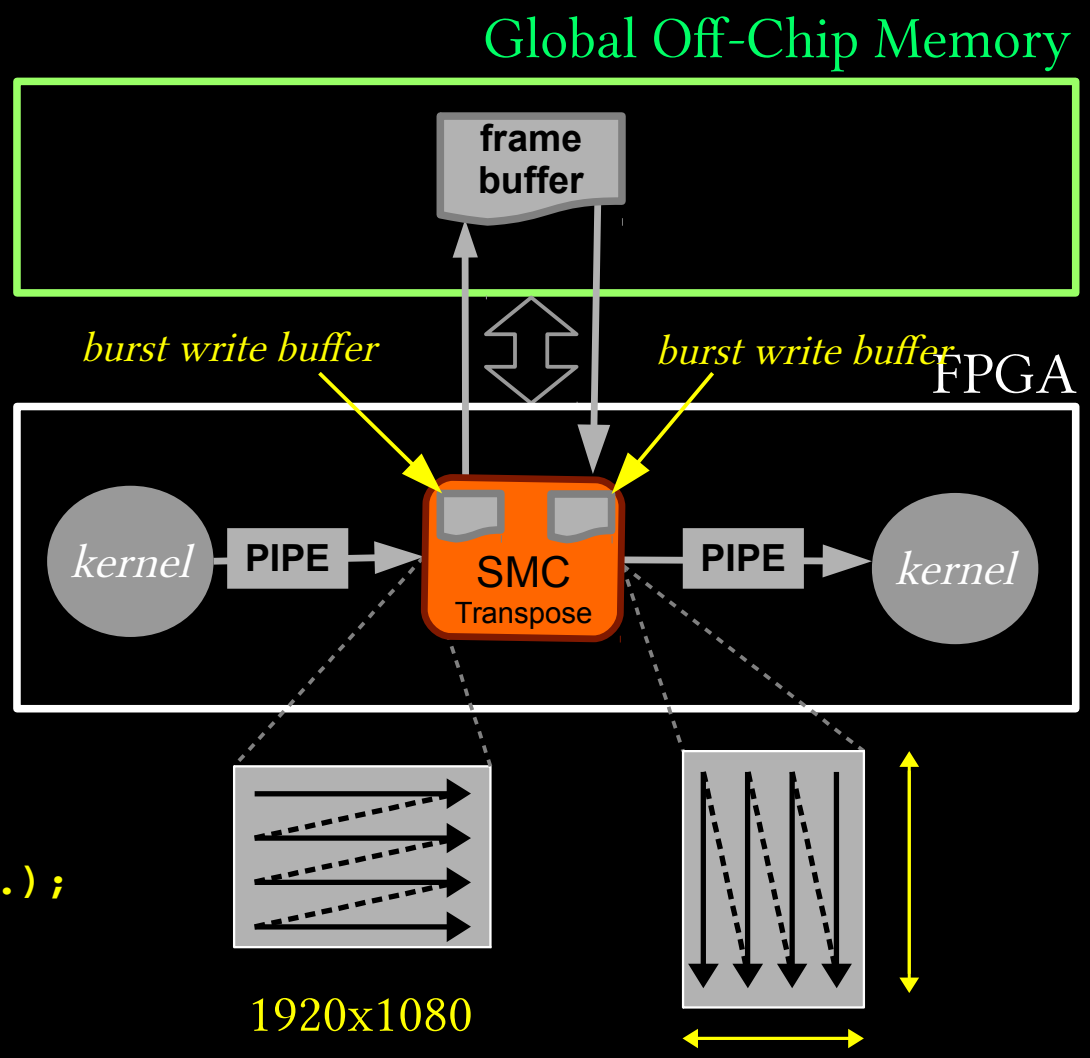
Stream Memory Components

- Memory Resources vs. Performance -

Example:

- SMC transpose
- PIPE to PIPE config
- Data size: HD video frame

2,073,600 words
3048 BRAM of storage



Host Code

```
smcEnqueueTranspose(...,1920,1080,...);
```

data size

Experimental Evaluation

- Memory Resources vs. Performance -

Transpose

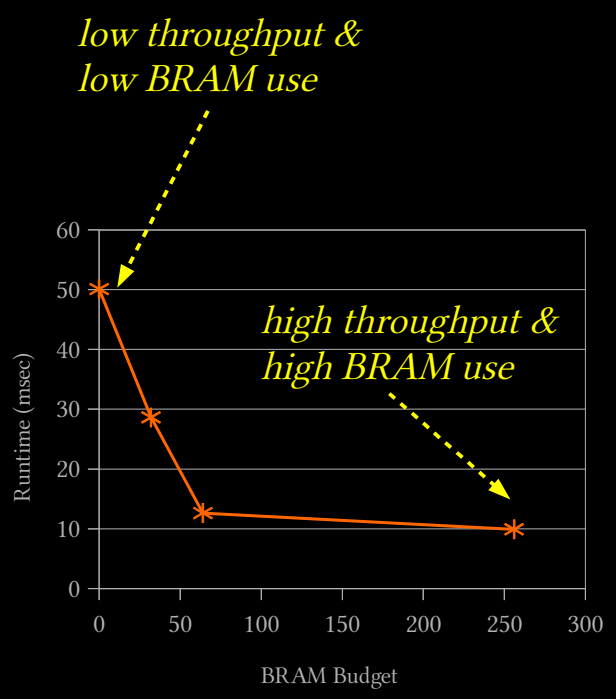
- single transpose SMC
- GLOBAL to GLOBAL
- 32-bit, 1024x1024 array

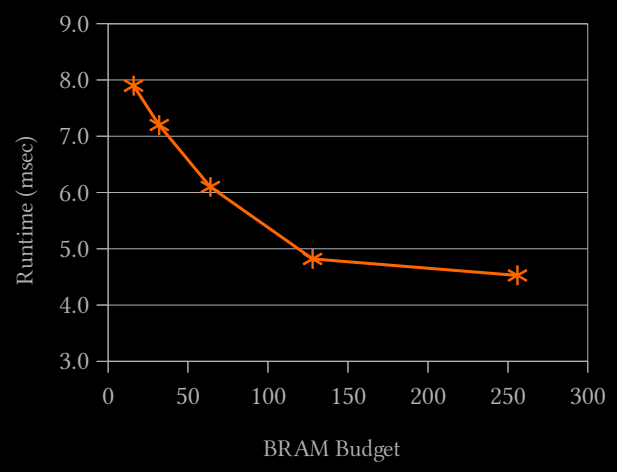
Video Watermarking

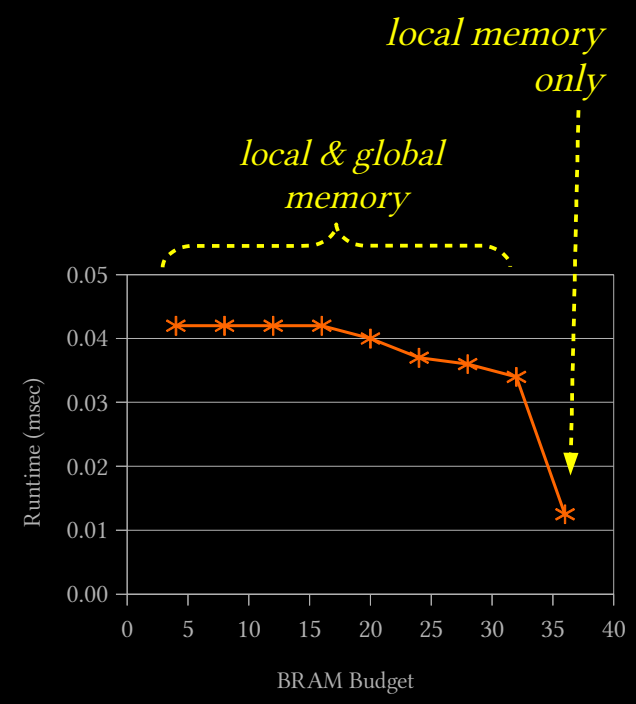
- four tile SMCs
- GLOBAL to PIPE
- PIPE to GLOBAL
- 32-bit, 1024x1024 array

FIR Filter

- single stencil SMCs
- PIPE to PIPE
- 9x9 FIR filter
- 32bit data







Experimental Evaluation

- Naive vs. SMC Optimized Performance -

Benchmark	Run-Time (msec)	BRAMs	DSPs	FF	LUTs	Run-Time (msec)	BRAMs	DSPs	FF	LUTs	Speedup
Video Watermarking	**15.4**	9	0	1,923	2,148	**4.82**	128	128	8,390	6,478	**3.2**
Matrix Multiply	**48.3**	0	4	2,380	2,834	**54**	96	4	13,477	29,799	**8.9**

Thank you