# MPack: Global Memory Optimization for Stream Applications in High-Level Synthesis

*Jasmina Vasiljevic and Paul Chow*

FPGA 2014

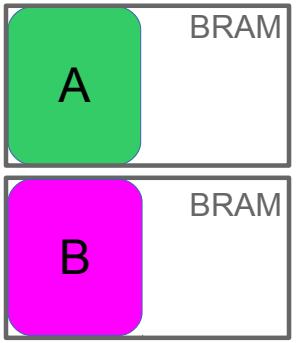
# Introduction

- Automated on-chip memory optimization
- Stream applications on FPGAs:
  - Use lots of memory
  - Steady and predictable data accesses

# Introduction

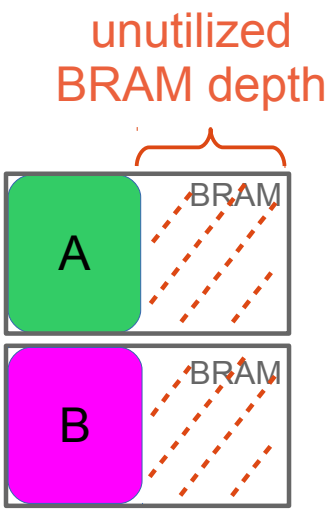
- Automated on-chip memory optimization
- Stream applications on FPGAs:
  - Use lots of memory
  - Steady and predictable data accesses
- *Buffer dimensions* do not always match *BRAM dimensions*
  - Result in low memory utilization
- *Buffer packing* can increase utilization
- Trade-off between *throughput* and *BRAM use*
  - Two *high-level MPack pragmas*

# Buffer Packing



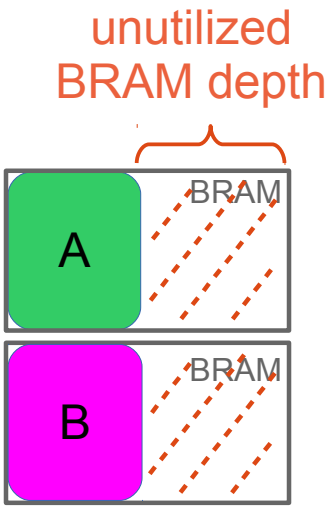

*a) unpacked buffers*

# Buffer Packing

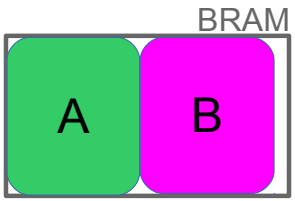



*a) unpacked buffers*

# Buffer Packing

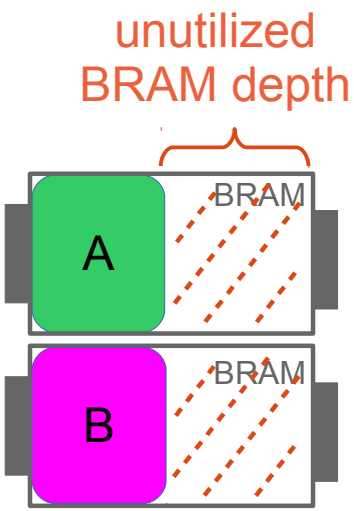



*a) unpacked buffers*

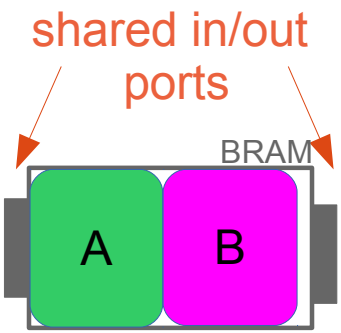



*b) horizontal packing*

# Buffer Packing

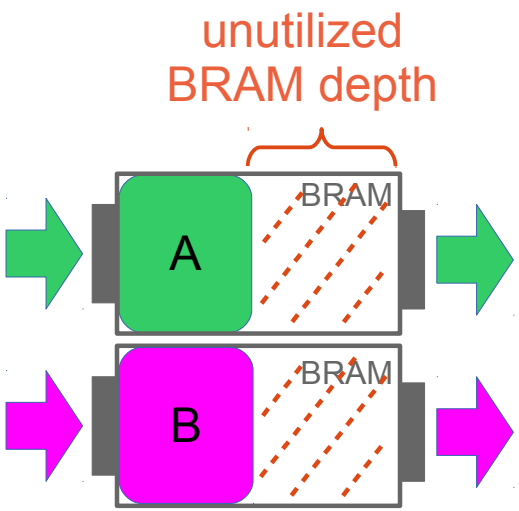



*a) unpacked buffers*

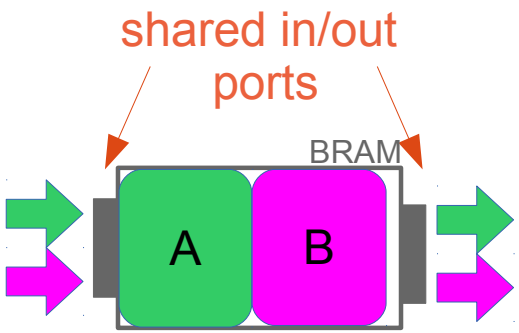



*b) horizontal packing*

# Buffer Packing

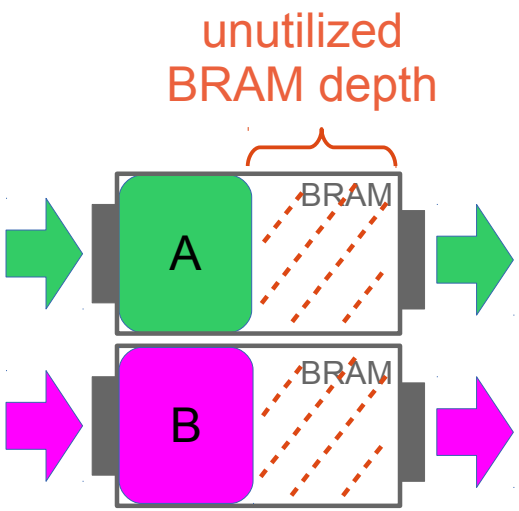



*a) unpacked buffers*

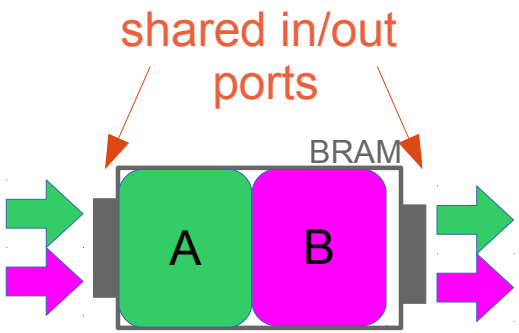



*b) horizontal packing*

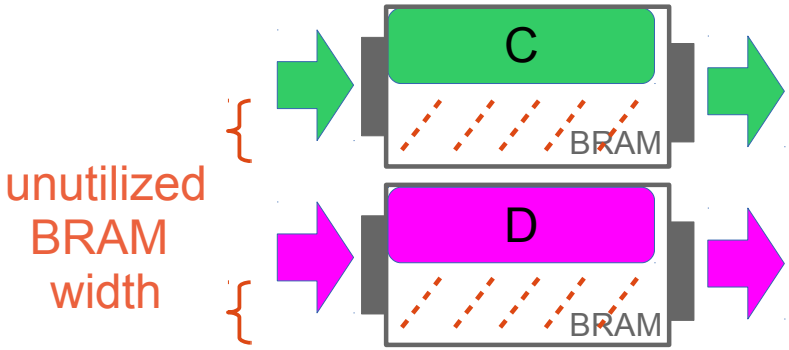
# Buffer Packing



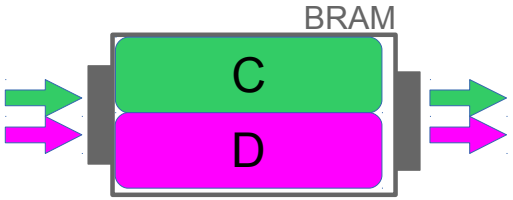

*a) unpacked buffers*




*b) horizontal packing*

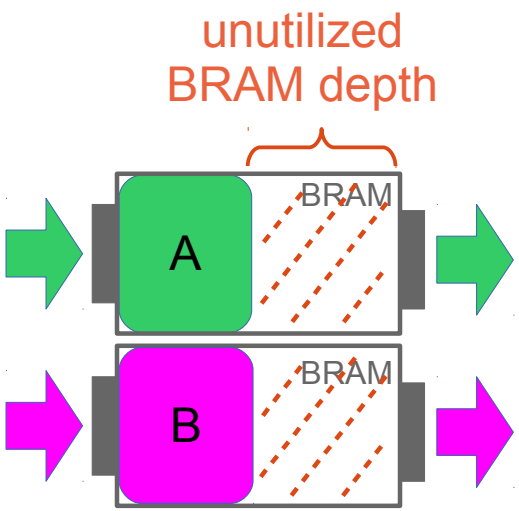



*c) unpacked buffers*

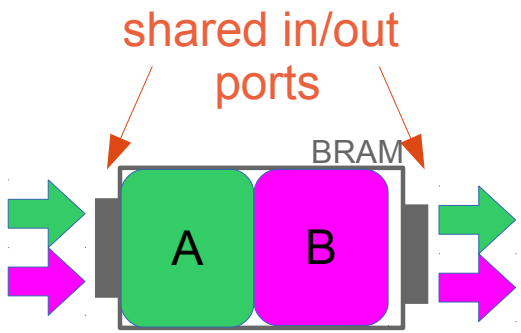



*d) vertical packing*

# Buffer Packing



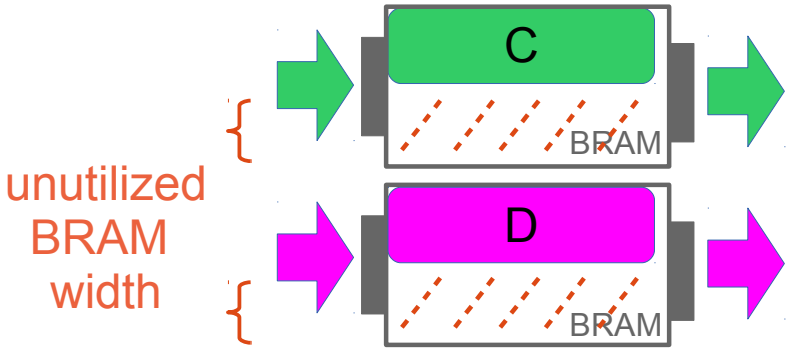

*a) unpacked buffers*



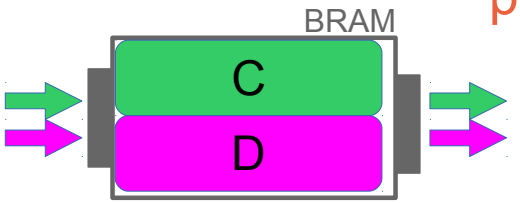

```
set_directive_array_map -instance AB -mode horizontal foo A
set_directive_array_map -instance AB -mode horizontal foo B
```

*b) horizontal packing*

XILINX
VivadoHLS
pragmas/directives




*c) unpacked buffers*



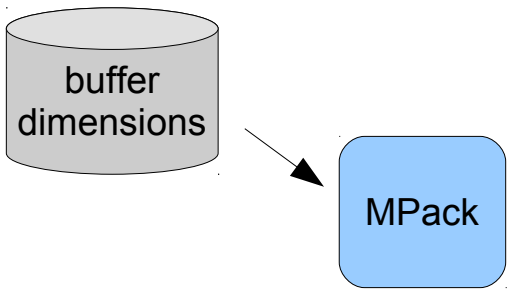

```
set_directive_array_map -instance AB -mode vertical foo A
set_directive_array_map -instance AB -mode vertical foo B
```

*d) vertical packing*

# Tool Flow

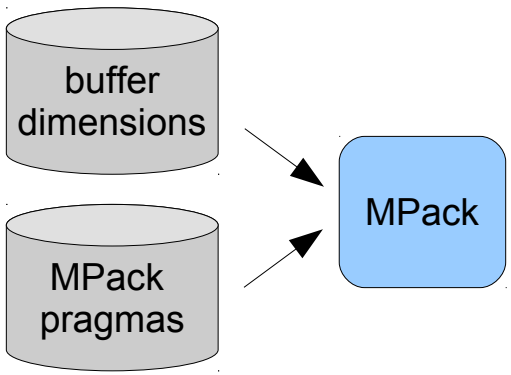
# Tool Flow

| Name | Words | Bit-width |
|---|---|---|
| buffer1 | 100 | 9 |
| buffer2 | 500 | 9 |
| buffer3 | 700 | 16 |
| buffer4 | 1800 | 16 |
| buffer5 | 1800 | 16 |
| buffer2 | 500 | 9 |
| buffer3 | 700 | 16 |
| buffer4 | 1800 | 16 |
| buffer4 | 1800 | 16 |



# Tool Flow

| Name | Words | Bit-width |
|---|---|---|
| buffer1 | 100 | 9 |
| buffer2 | 500 | 9 |
| buffer3 | 700 | 16 |
| buffer4 | 1800 | 16 |
| buffer5 | 1800 | 16 |
| buffer2 | 500 | 9 |
| buffer3 | 700 | 16 |
| buffer4 | 1800 | 16 |
| buffer4 | 1800 | 16 |




*High-level MPack Pragmas:*

`mpack_system_thr`

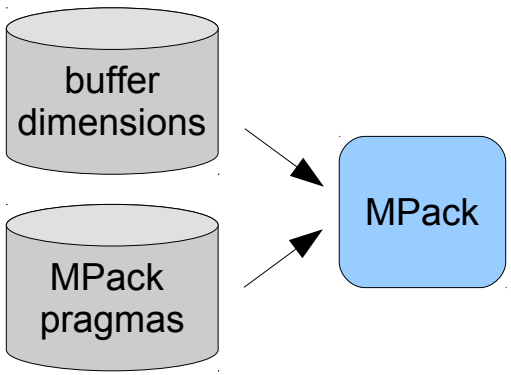
OR

`mpack_brams`

# Tool Flow

| Name | Words | Bit-width |
|---|---|---|
| buffer1 | 100 | 9 |
| buffer2 | 500 | 9 |
| buffer3 | 700 | 16 |
| buffer4 | 1800 | 16 |
| buffer5 | 1800 | 16 |
| buffer2 | 500 | 9 |
| buffer3 | 700 | 16 |
| buffer4 | 1800 | 16 |
| buffer4 | 1800 | 16 |

- Brute force optimization
- All buffer packing combinations



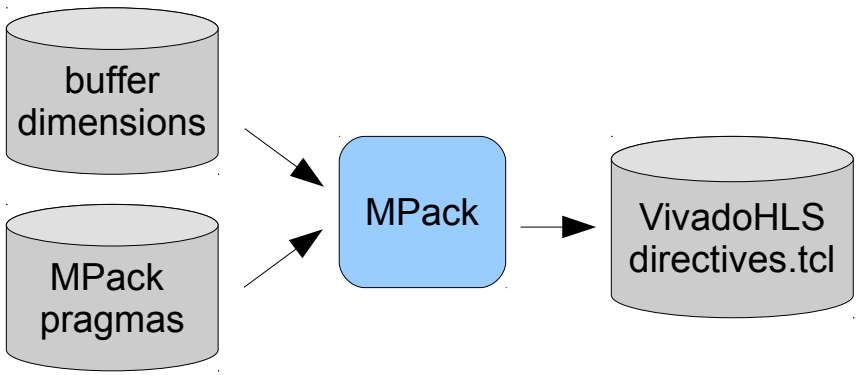

*High-level MPack Pragmas:*

`mpack_system_thr`
OR
`mpack_brams`

# Tool Flow

| Name | Words | Bit-width |
|---|---|---|
| buffer1 | 100 | 9 |
| buffer2 | 500 | 9 |
| buffer3 | 700 | 16 |
| buffer4 | 1800 | 16 |
| buffer5 | 1800 | 16 |
| buffer2 | 500 | 9 |
| buffer3 | 700 | 16 |
| buffer4 | 1800 | 16 |
| buffer4 | 1800 | 16 |

- Brute force optimization
- All buffer packing combinations




*High-level MPack Pragmas:*

*mpack_system_thr*
OR
*mpack_brams*

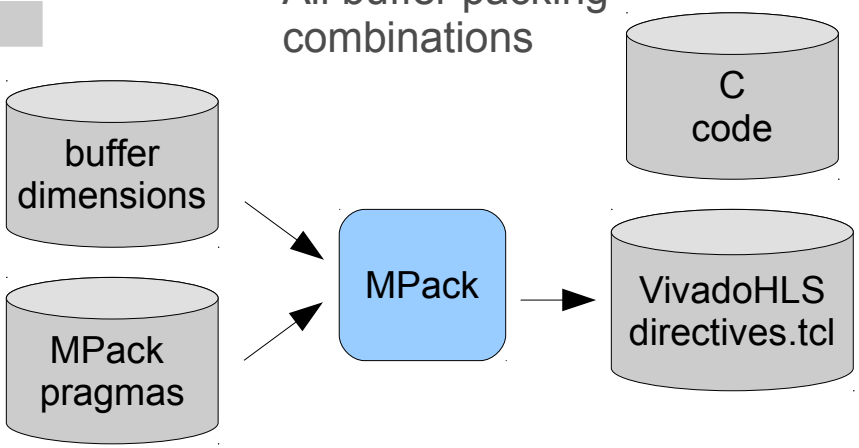
*set_directive_array_map -instance AB -mode horizontal foo buffer1*
*set_directive_array_map -instance AB -mode horizontal foo buffer2*
*set_directive_array_map -instance C -mode vertical foo buffer3*
*set_directive_array_map -instance C -mode vertical foo buffer4*
*set_directive_array_map -instance D -mode horizontal foo buffer5*
*set_directive_array_map -instance D -mode horizontal foo buffer6*
*set_directive_array_map -instance E -mode vertical foo buffer7*
*set_directive_array_map -instance E -mode vertical foo buffer8*

# Tool Flow

| Name | Words | Bit-width |
|---|---|---|
| buffer1 | 100 | 9 |
| buffer2 | 500 | 9 |
| buffer3 | 700 | 16 |
| buffer4 | 1800 | 16 |
| buffer5 | 1800 | 16 |
| buffer2 | 500 | 9 |
| buffer3 | 700 | 16 |
| buffer4 | 1800 | 16 |
| buffer4 | 1800 | 16 |

- Brute force optimization
- All buffer packing combinations

- Stream application




*High-level MPack Pragmas:*

`mpack_system_thr`
OR
`mpack_brams`

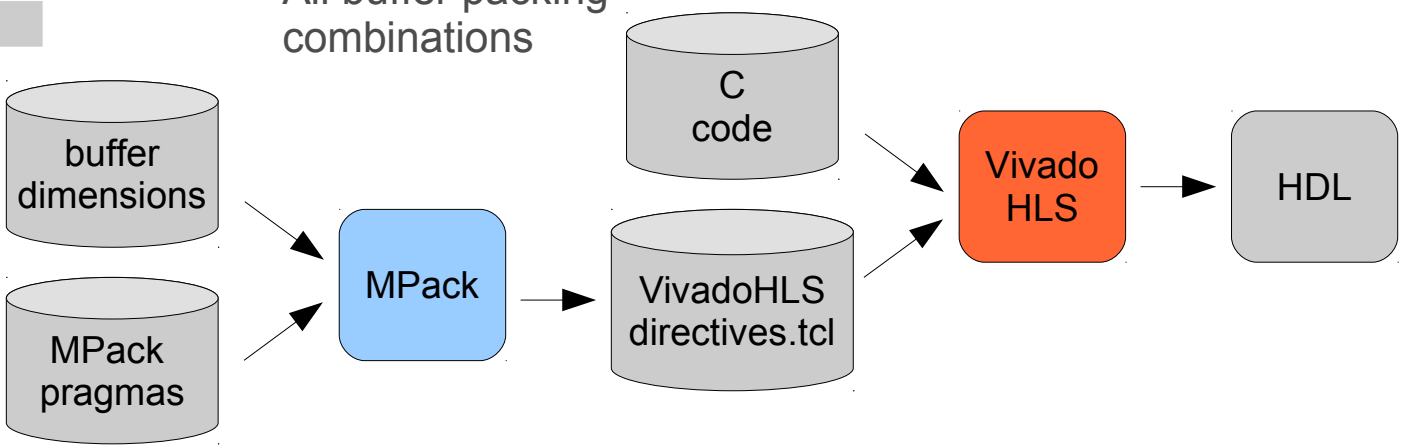
```
set_directive_array_map -instance AB -mode horizontal foo buffer1
set_directive_array_map -instance AB -mode horizontal foo buffer2
set_directive_array_map -instance C -mode vertical foo buffer3
set_directive_array_map -instance C -mode vertical foo buffer4
set_directive_array_map -instance D -mode horizontal foo buffer5
set_directive_array_map -instance D -mode horizontal foo buffer6
set_directive_array_map -instance E -mode vertical foo buffer7
set_directive_array_map -instance E -mode vertical foo buffer8
```

# Tool Flow

| Name | Words | Bit-width |
|---|---|---|
| buffer1 | 100 | 9 |
| buffer2 | 500 | 9 |
| buffer3 | 700 | 16 |
| buffer4 | 1800 | 16 |
| buffer5 | 1800 | 16 |
| buffer2 | 500 | 9 |
| buffer3 | 700 | 16 |
| buffer4 | 1800 | 16 |
| buffer4 | 1800 | 16 |

- Brute force optimization
- All buffer packing combinations

- Stream application




*High-level MPack Pragmas:*

`mpack_system_thr`
OR
`mpack_brams`

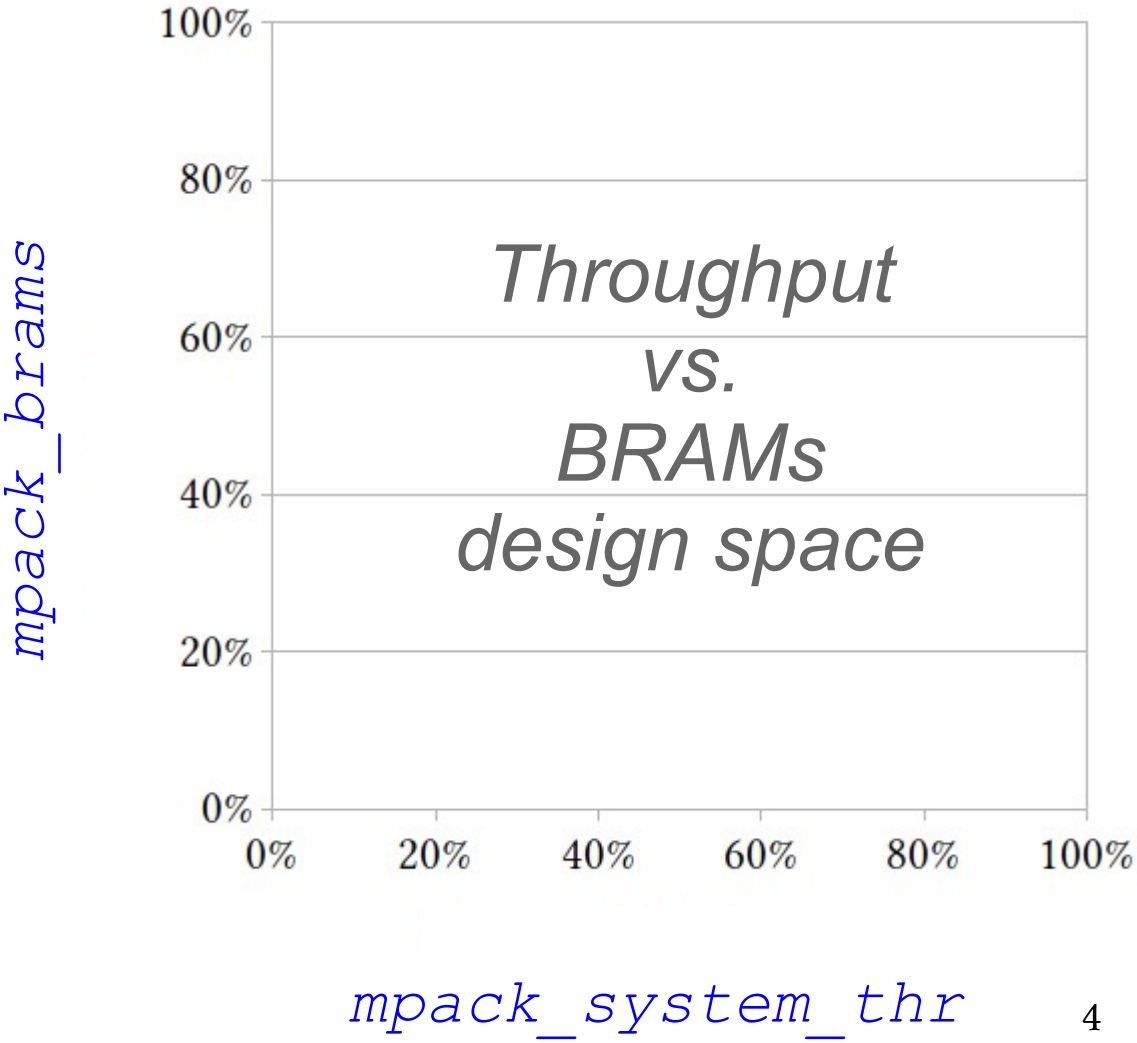
```
set_directive_array_map -instance AB -mode horizontal foo buffer1
set_directive_array_map -instance AB -mode horizontal foo buffer2
set_directive_array_map -instance C -mode vertical foo buffer3
set_directive_array_map -instance C -mode vertical foo buffer4
set_directive_array_map -instance D -mode horizontal foo buffer5
set_directive_array_map -instance D -mode horizontal foo buffer6
set_directive_array_map -instance E -mode vertical foo buffer7
set_directive_array_map -instance E -mode vertical foo buffer8
```

# Results

- MPack high-level pragmas:

  1. `mpack_system_thr`
     - Data throughput of the entire design

  2. `mpack_brams`
     - The upper bound on BRAMs



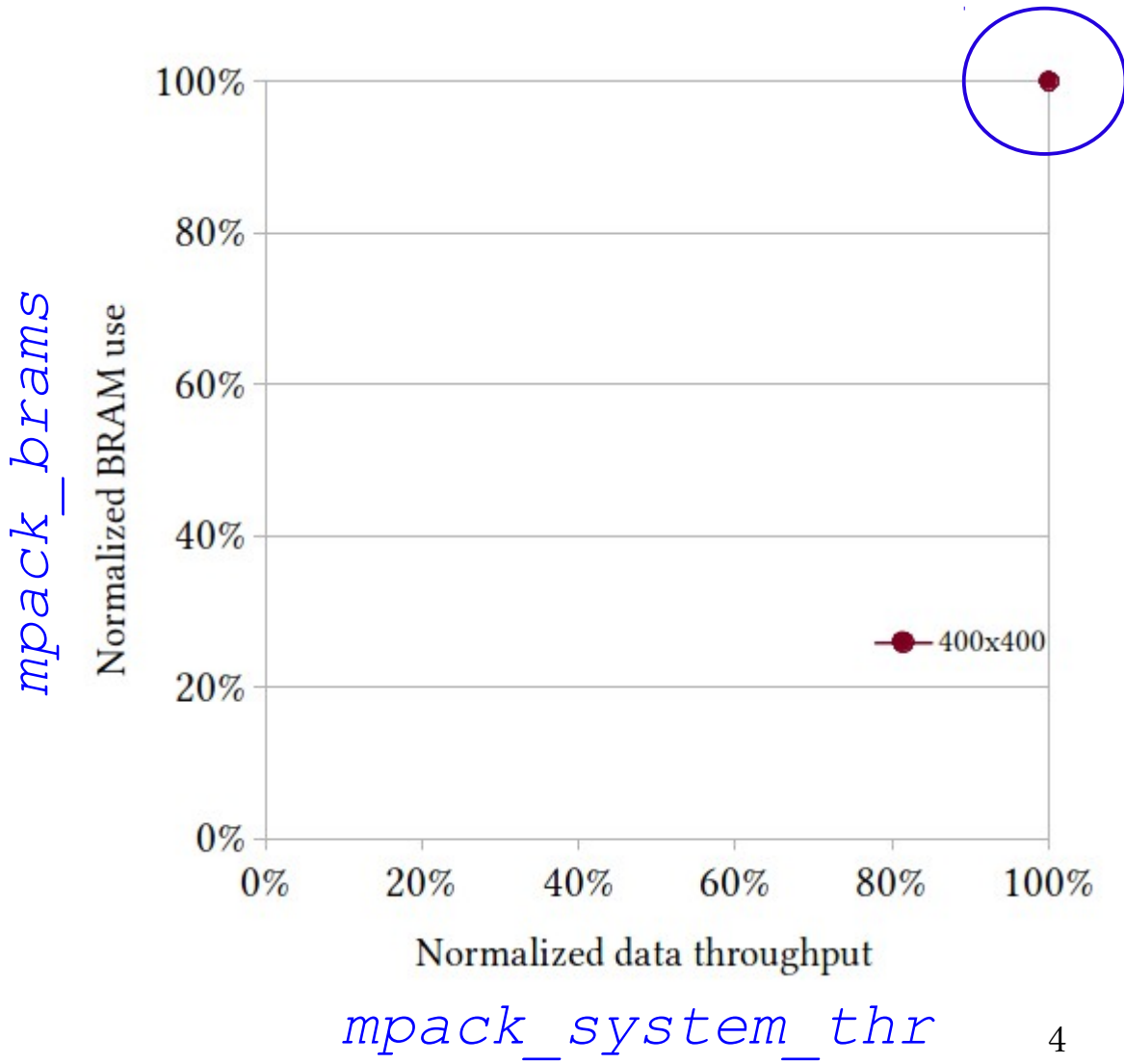
# Results

- MPack high-level pragmas:

  1. `mpack_system_thr`
     - Data throughput of the entire design

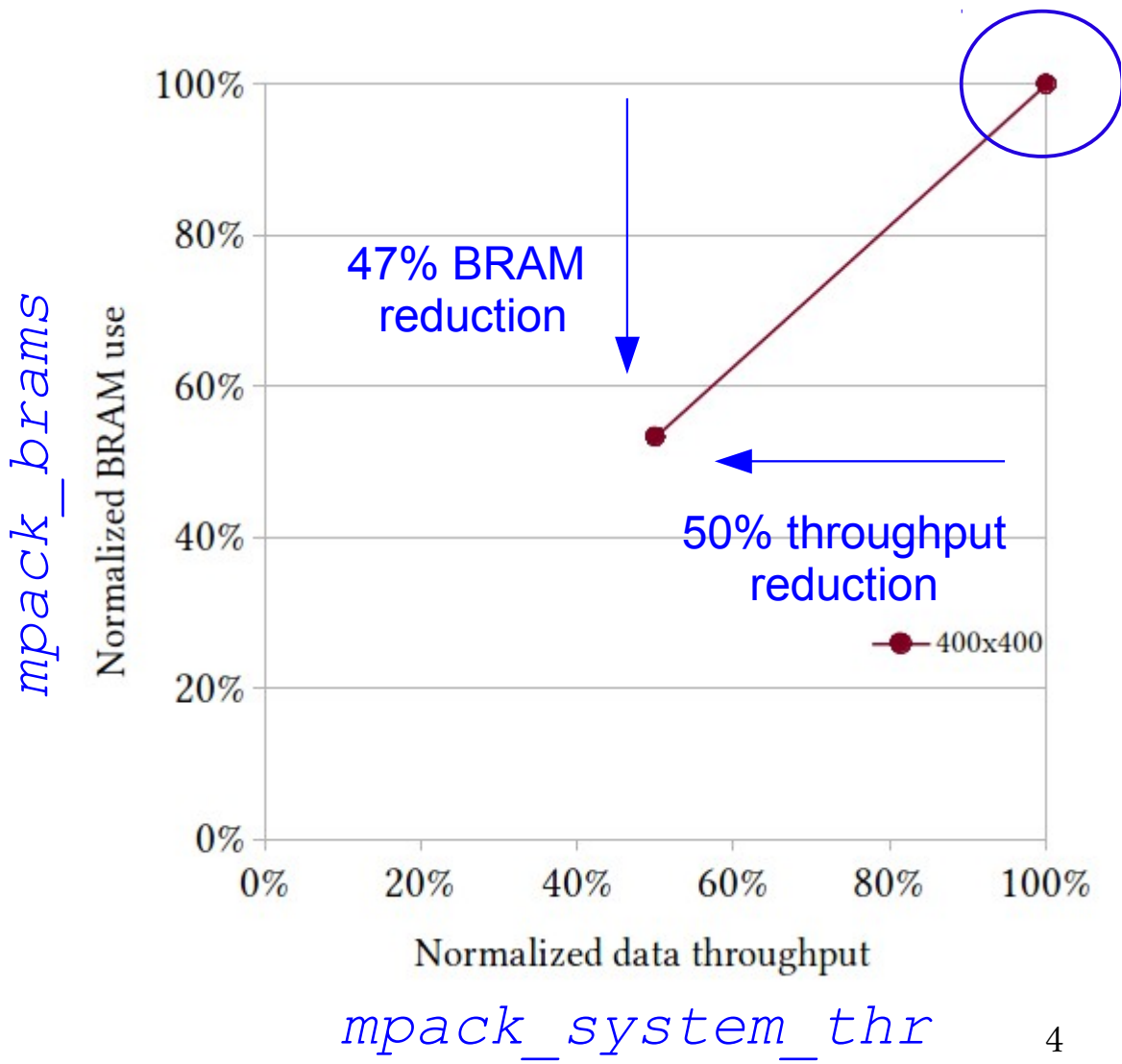
  2. `mpack_brams`
     - The upper bound on BRAMs

- Gaussian pyramid benchmark



# Results

- MPack high-level pragmas:
  1. `mpack_system_thr`
     - Data throughput of the entire design
  2. `mpack_brams`
     - The upper bound on BRAMs
- Gaussian pyramid benchmark



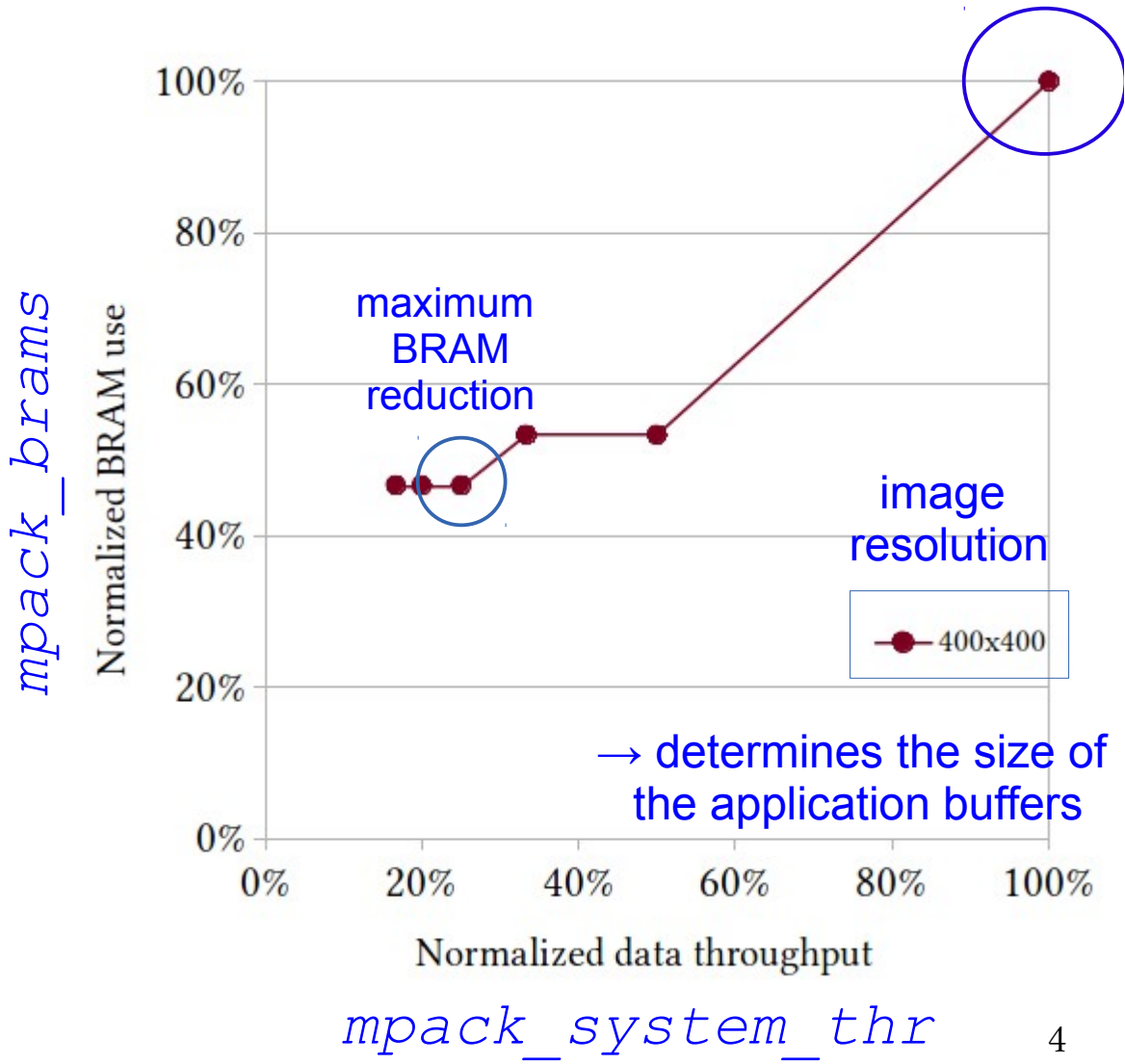
# Results

- MPack high-level pragmas:

  1. `mpack_system_thr`
     - Data throughput of the entire design

  2. `mpack_brams`
     - The upper bound on BRAMs

- Gaussian pyramid benchmark



# Results

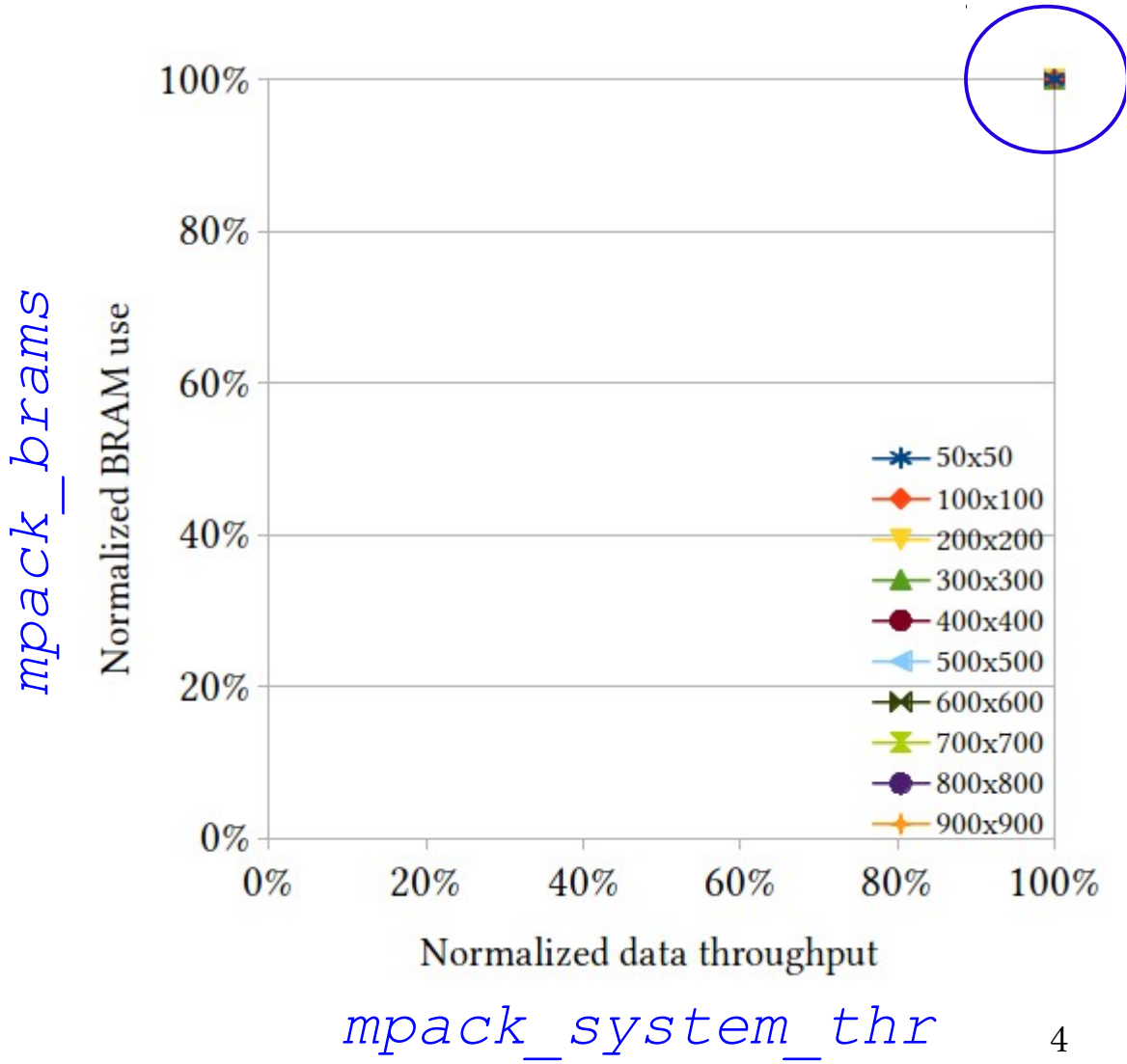
- Default design
- No buffer packing

- MPack high-level pragmas:

  1. `mpack_system_thr`
     - Data throughput of the entire design

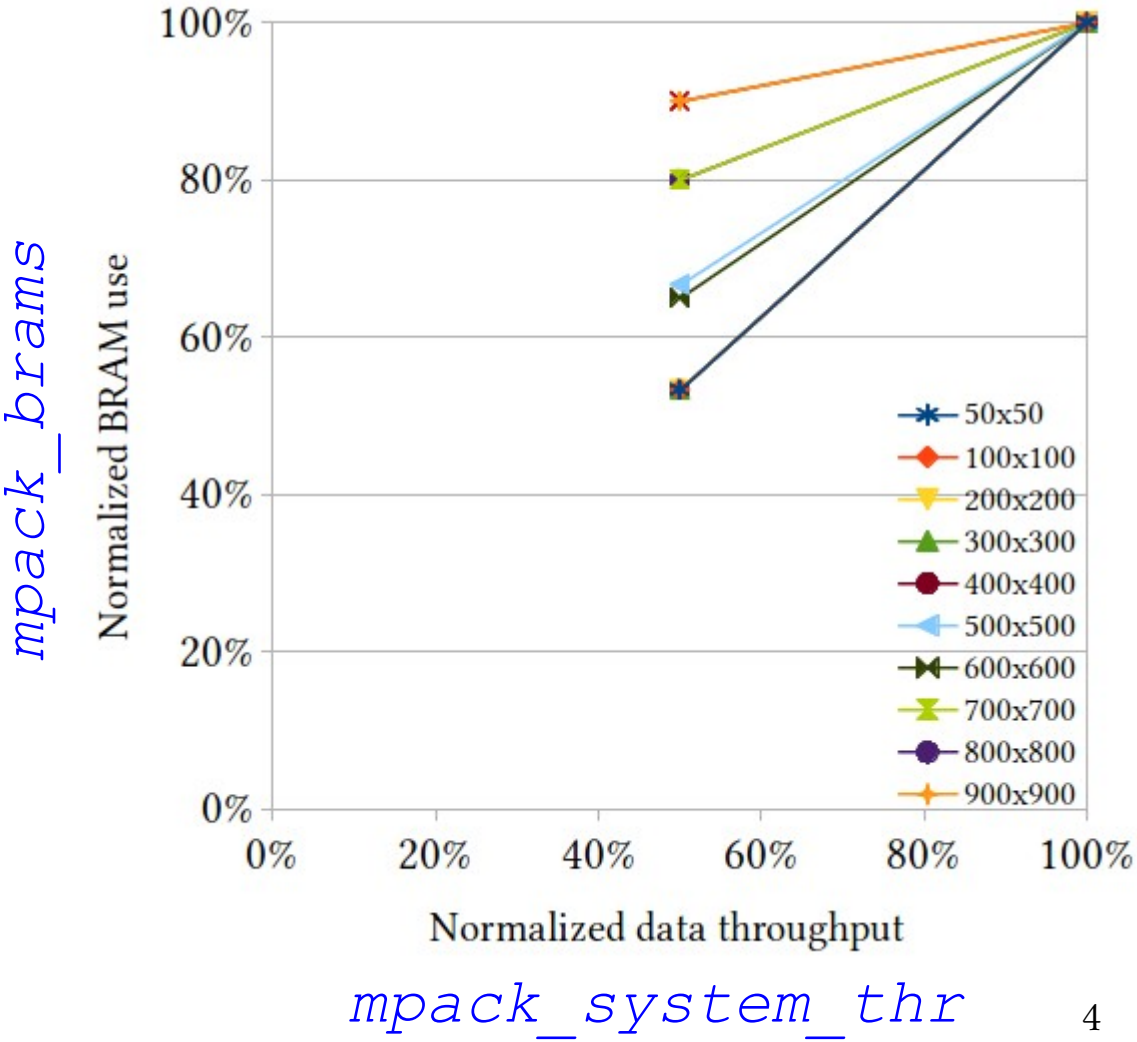
  2. `mpack_brams`
     - The upper bound on BRAMs

- Gaussian pyramid benchmark



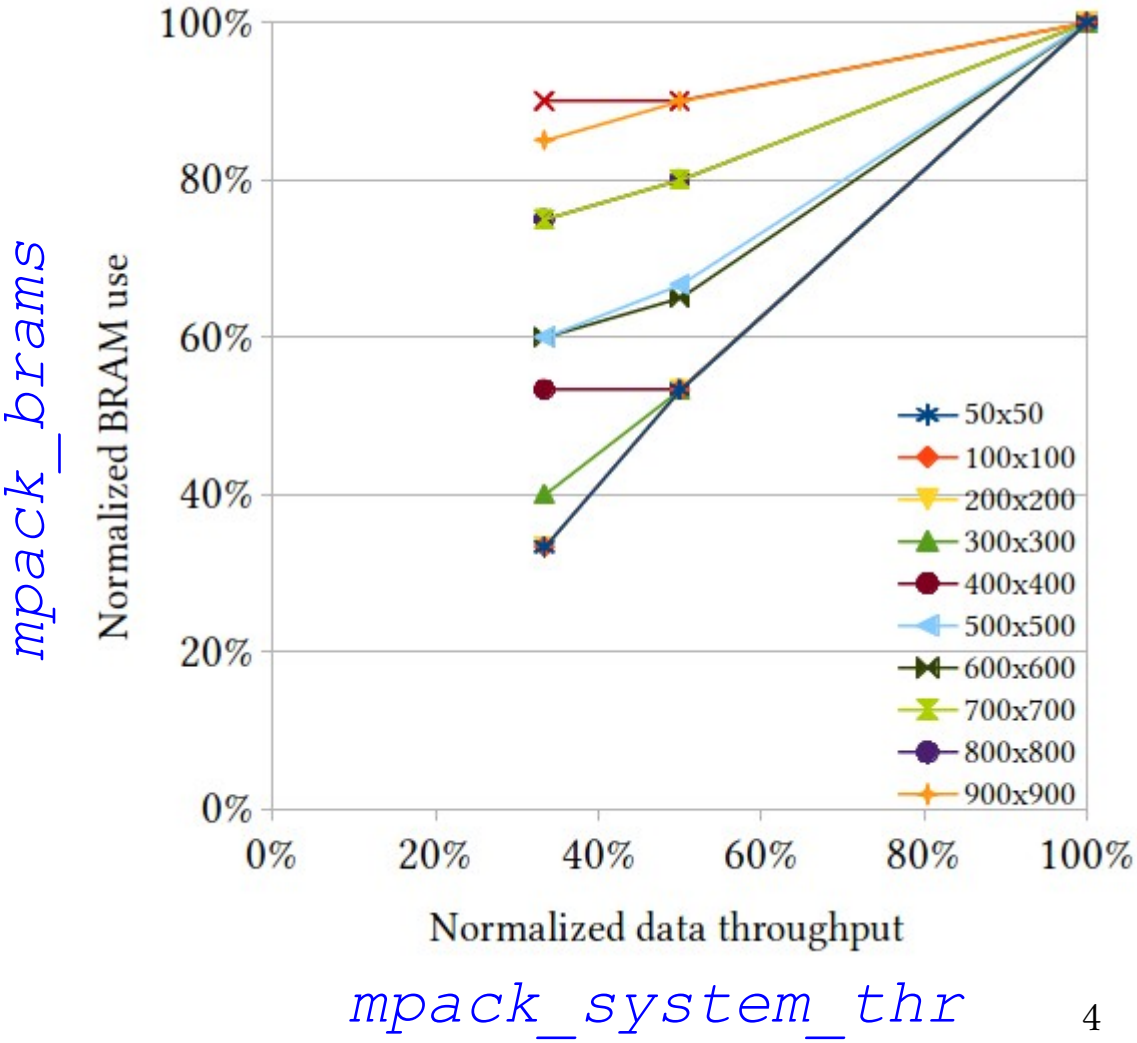
# Results

- MPack high-level pragmas:
  1. `mpack_system_thr`
     - Data throughput of the entire design
  2. `mpack_brams`
     - The upper bound on BRAMs
- Gaussian pyramid benchmark



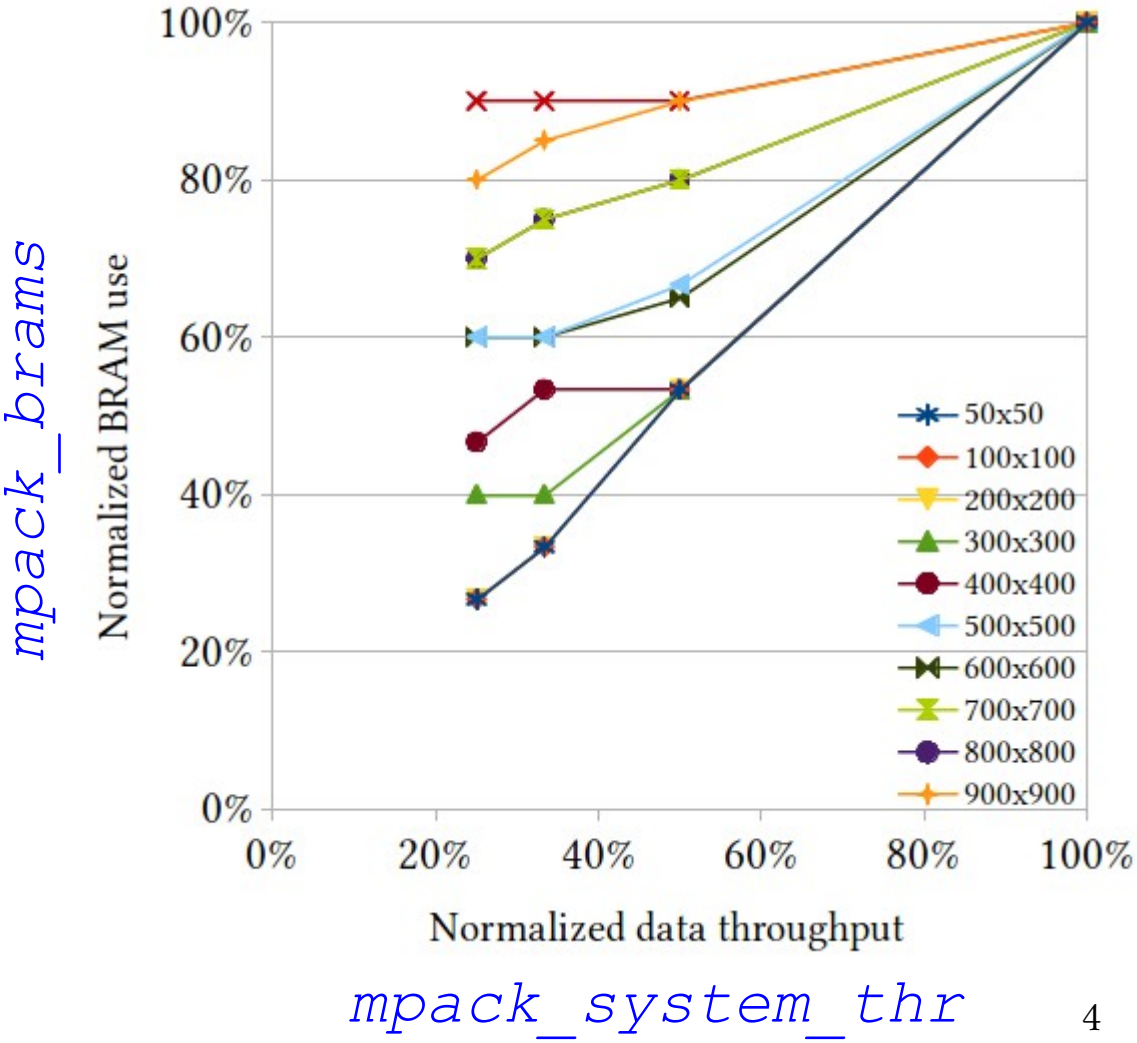
# Results

- MPack high-level pragmas:
  1. `mpack_system_thr`
     - Data throughput of the entire design
  2. `mpack_brams`
     - The upper bound on BRAMs
- Gaussian pyramid benchmark



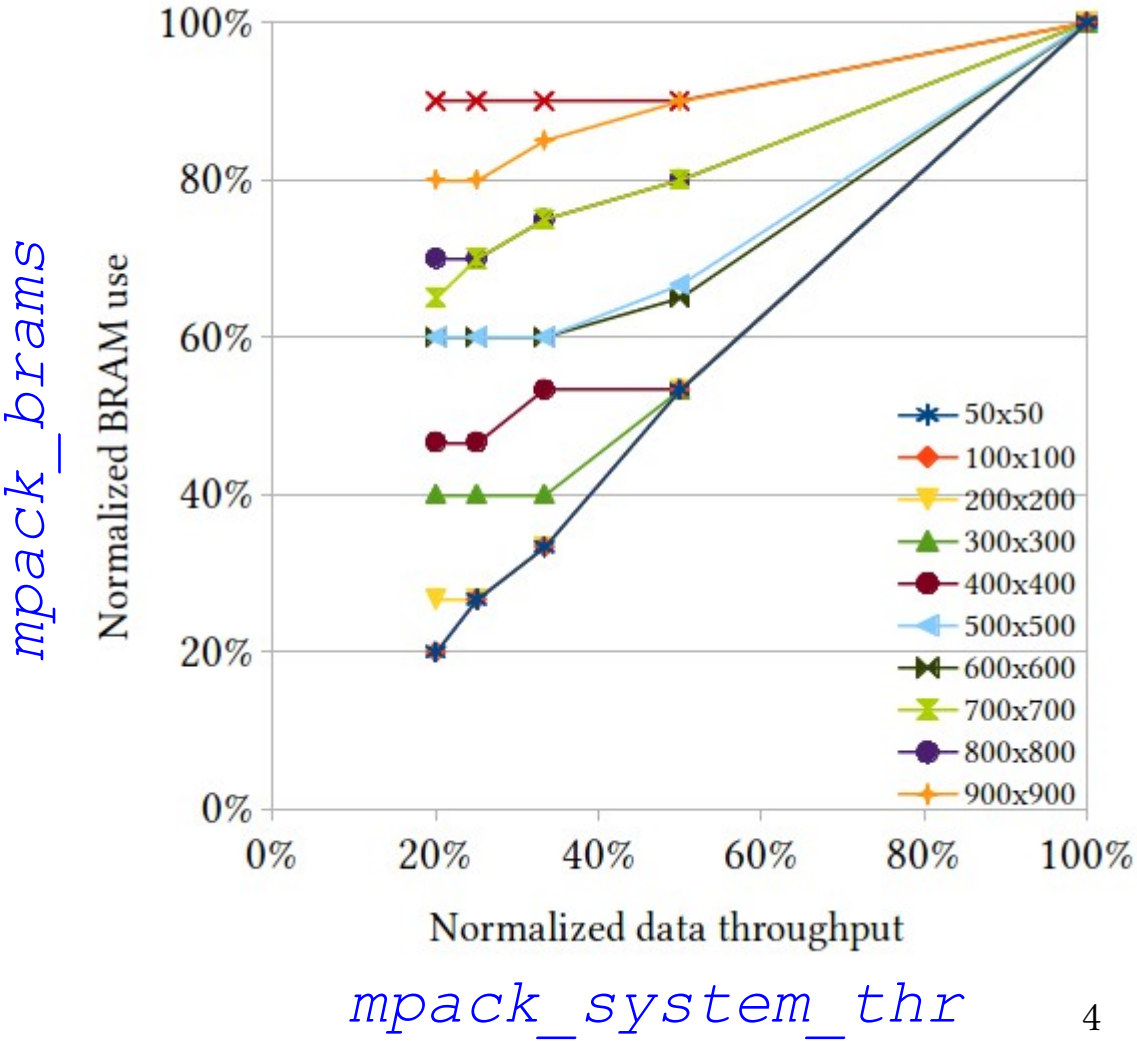
# Results

- MPack high-level pragmas:
  1. `mpack_system_thr`
     - Data throughput of the entire design
  2. `mpack_brams`
     - The upper bound on BRAMs
- Gaussian pyramid benchmark



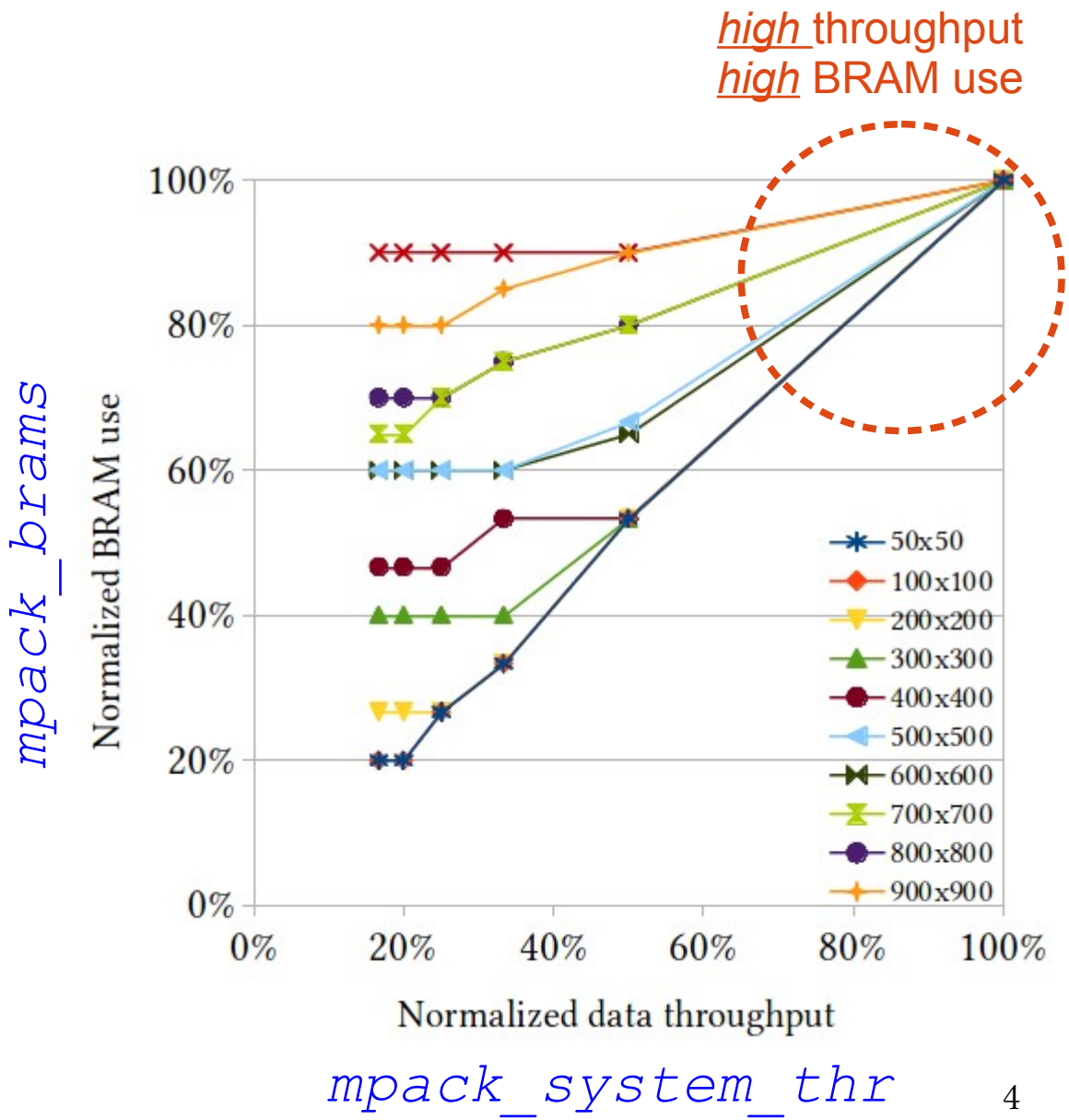
# Results

- MPack high-level pragmas:
  1. `mpack_system_thr`
     - Data throughput of the entire design
  2. `mpack_brams`
     - The upper bound on BRAMs
- Gaussian pyramid benchmark



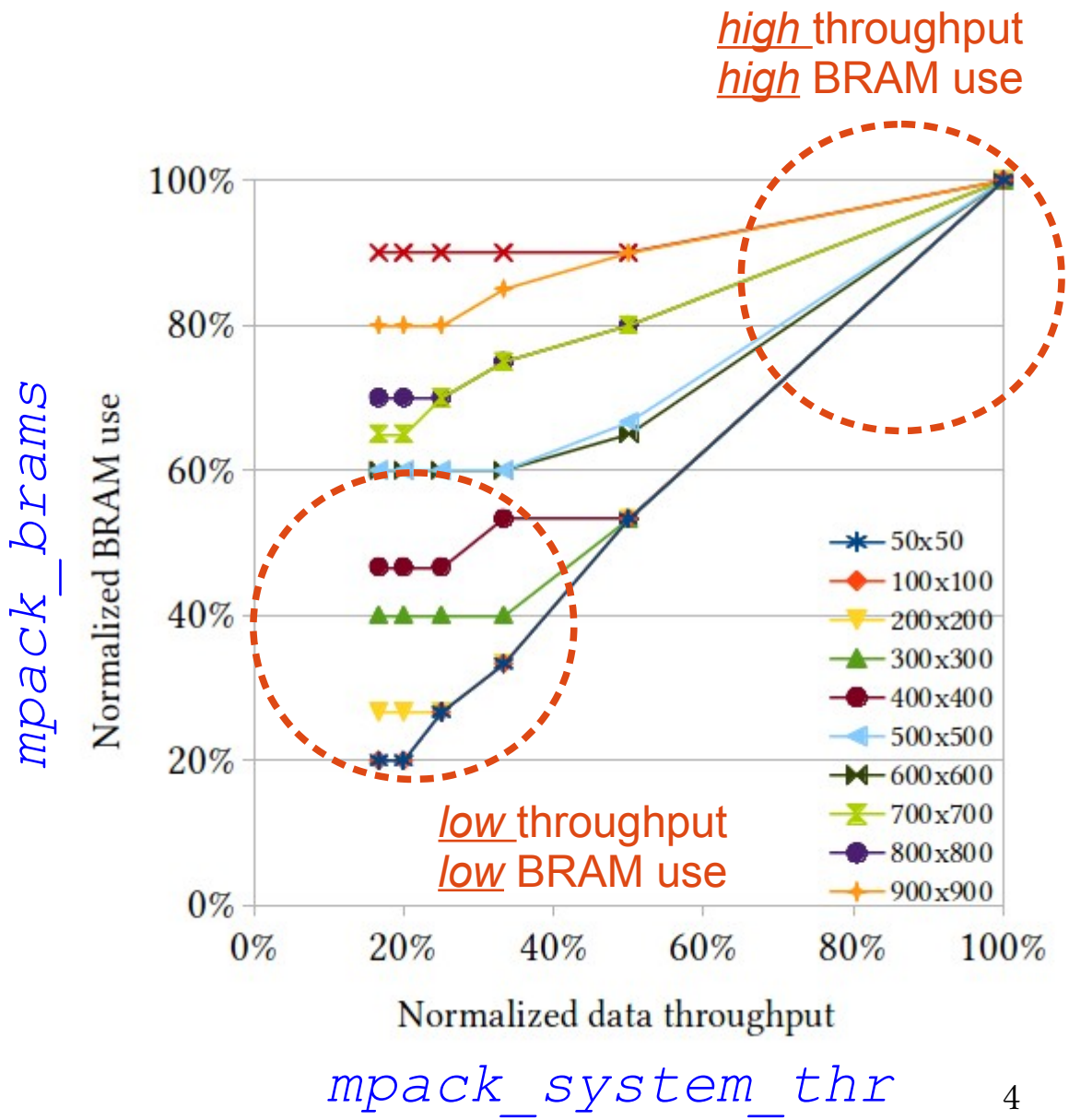
# Results

- MPack high-level pragmas:
  1. `mpack_system_thr`
     - Data throughput of the entire design
  2. `mpack_brams`
     - The upper bound on BRAMs
- Gaussian pyramid benchmark



# Results

- MPack high-level pragmas:
  1. `mpack_system_thr`
     - Data throughput of the entire design
  2. `mpack_brams`
     - The upper bound on BRAMs
- Gaussian pyramid benchmark



# Conclusion

- Fast memory design space exploration for stream applications
- Two high-level pragmas
  - Buffer packing approach
  - Throughput vs. BRAM use trade-off

# Conclusion

- Fast memory design space exploration for stream applications
- Two high-level pragmas
  - Buffer packing approach
  - Throughput vs. BRAM use trade-off
- Future work
  - Using buffer partitioning to scale the throughput up
  - Improve packing algorithm
  - More buffer packing approaches
  - Low-level optimizations

*Thank you*