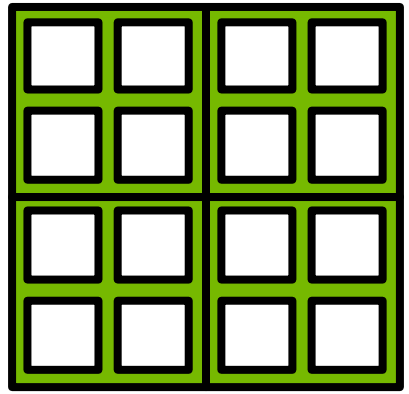
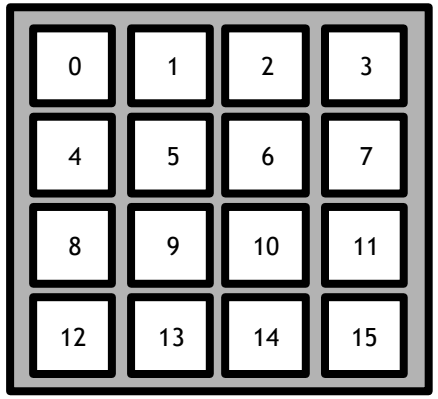
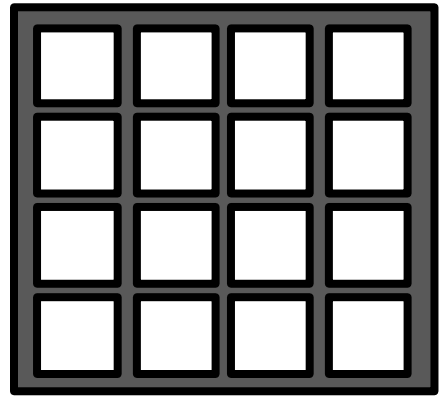
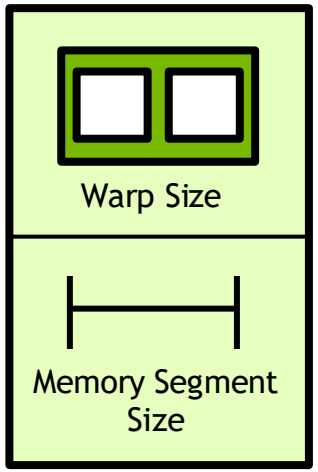
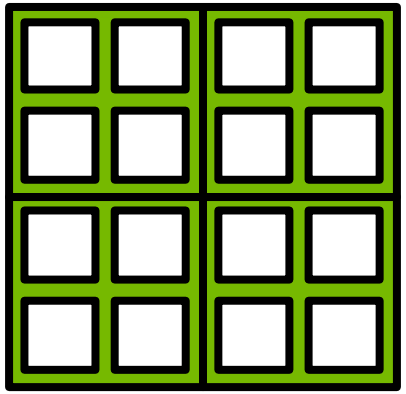
(2, 2)
(2, 2) (4,4)





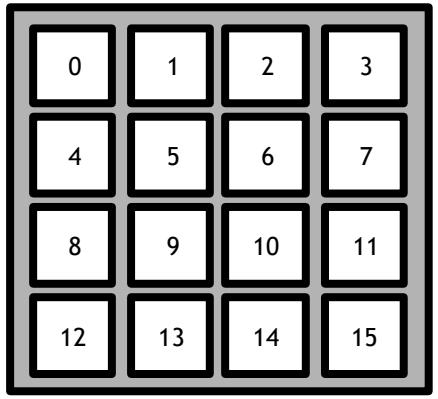


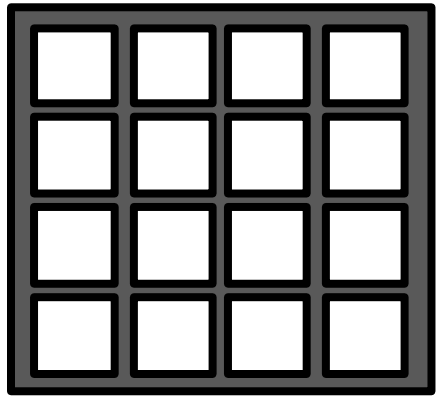


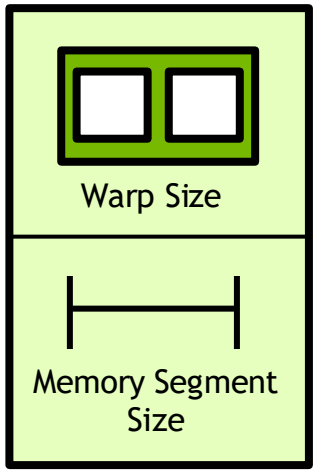


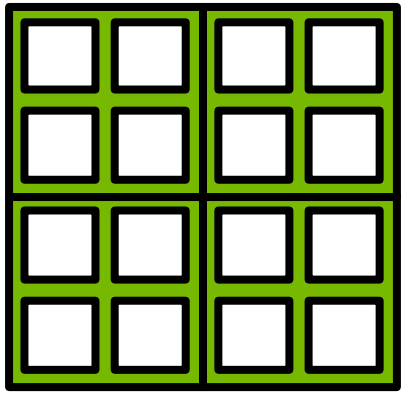
Warp

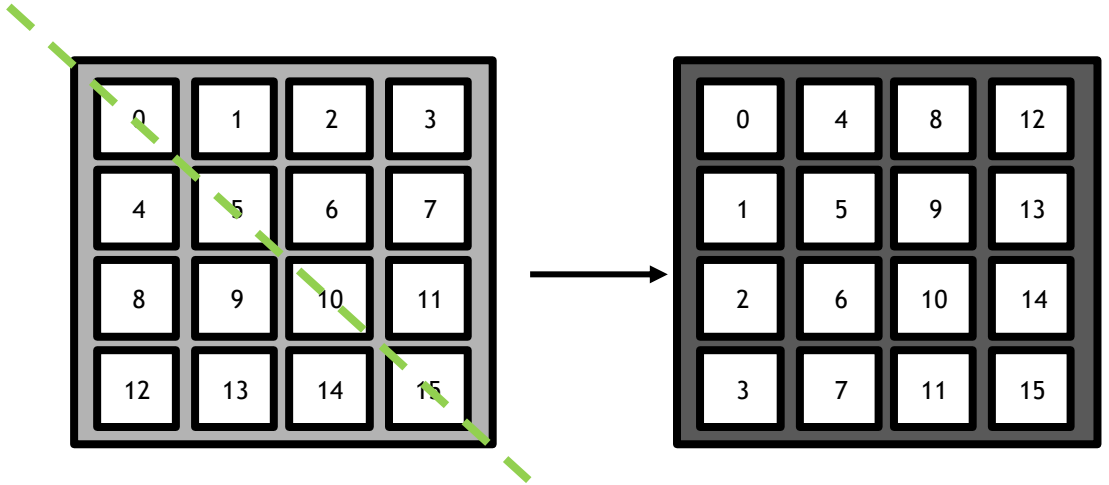
2 2

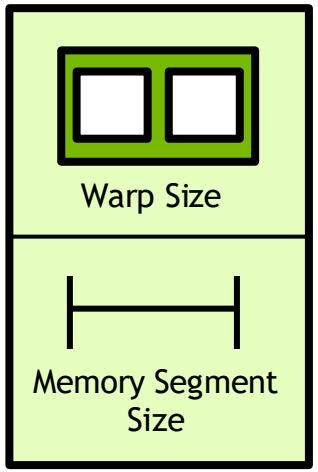


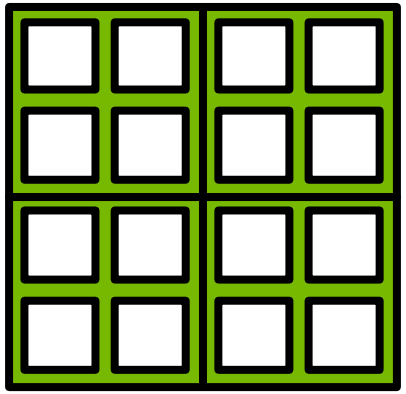








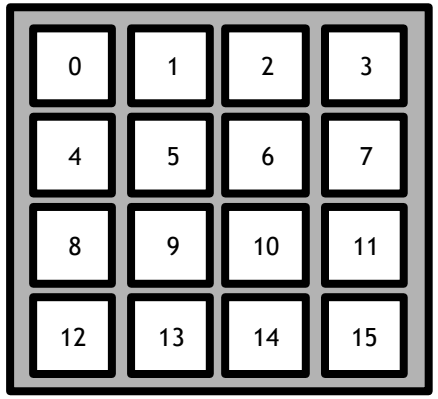


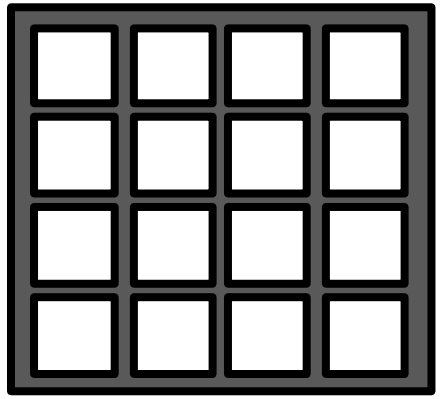


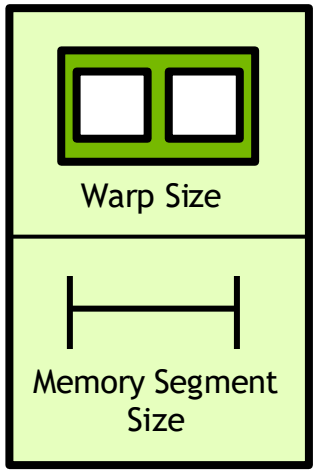
助

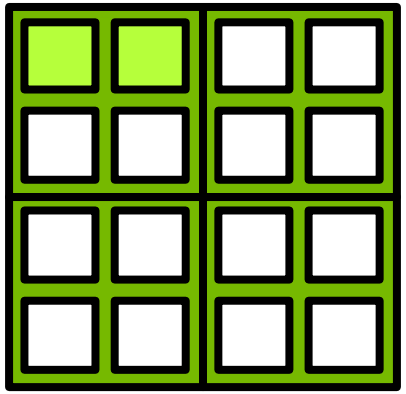
1

```
x, y = cuda.grid(2)

out[x][y] = in[y][x]
```



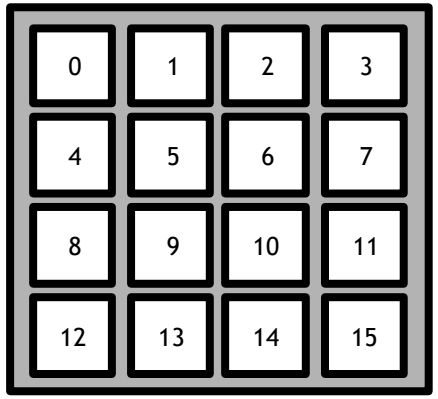


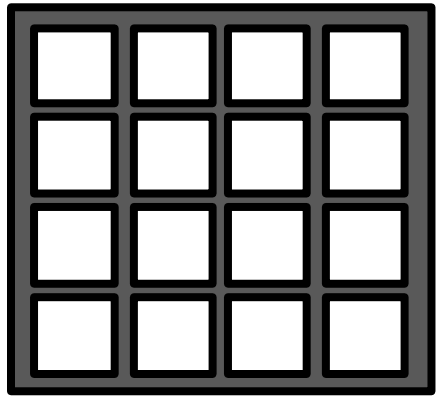


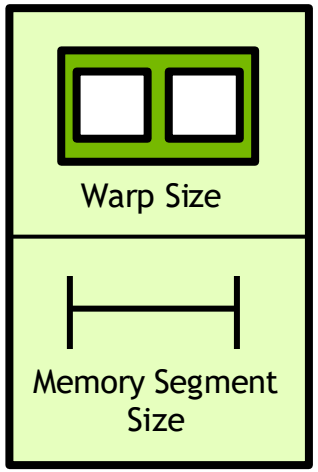


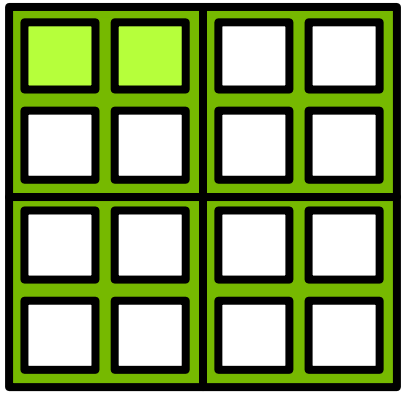
Warp

```
x, y = cuda.grid(2)

out[x][y] = in[y][x]
```





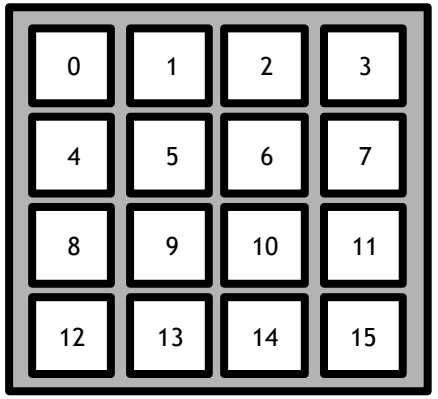


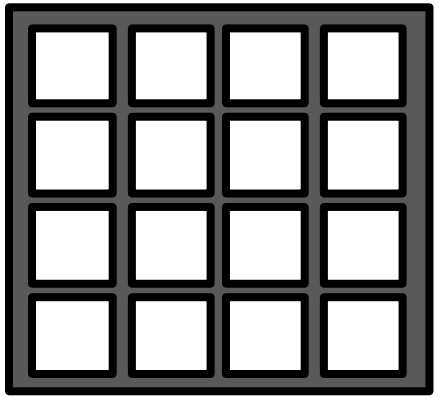


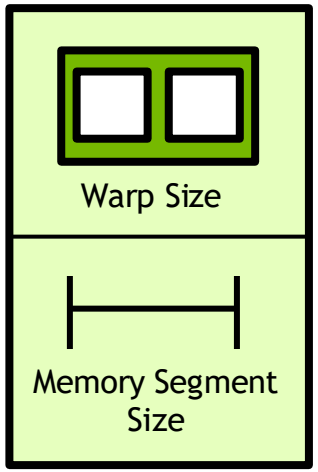
warp

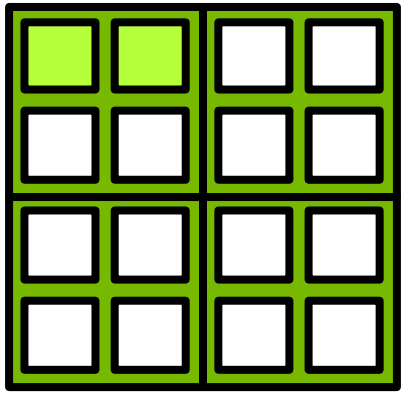
x

```
x = blockIdx.x * blockDim.x + threadIdx.x

y = blockIdx.y * blockDim.y + threadIdx.y

out[x][y] = in[y][x]
```

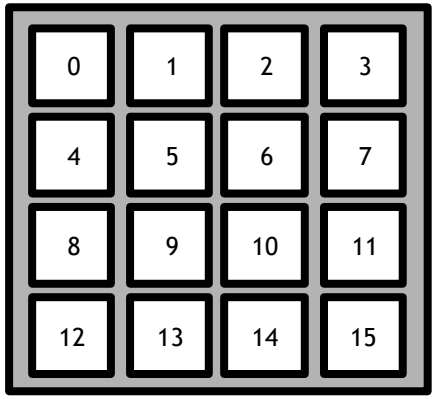


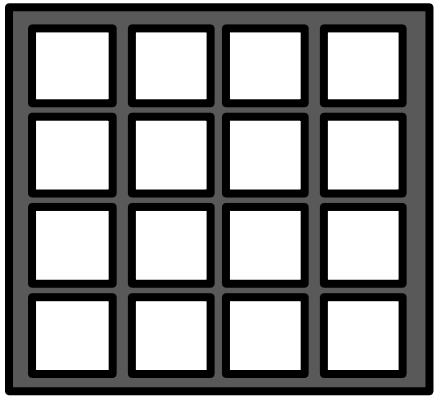


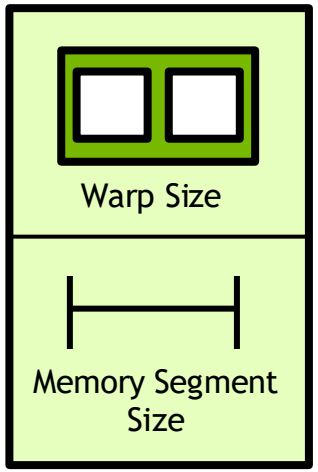


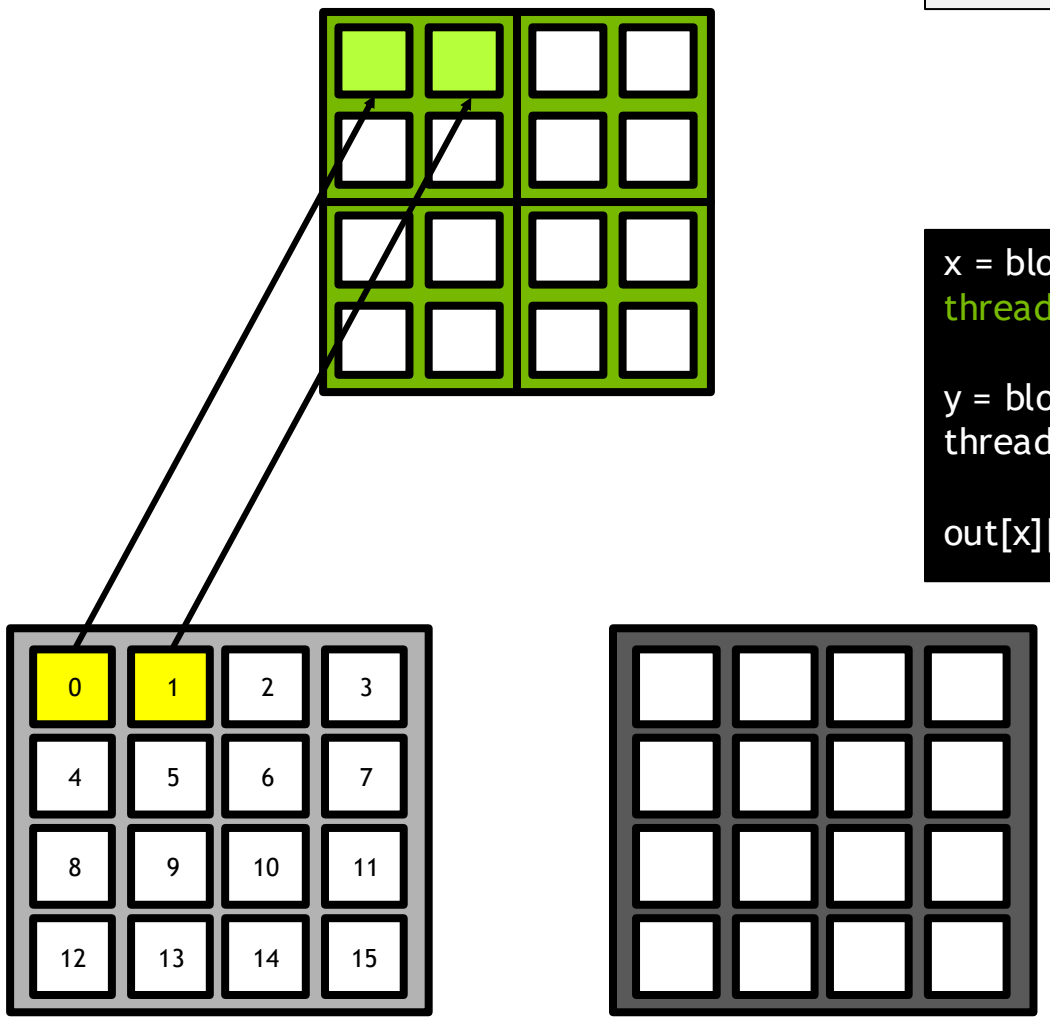


```
x = blockIdx.x * blockDim.x + threadIdx.x

y = blockIdx.y * blockDim.y + threadIdx.y

out[x][y] = in[y][x]
```

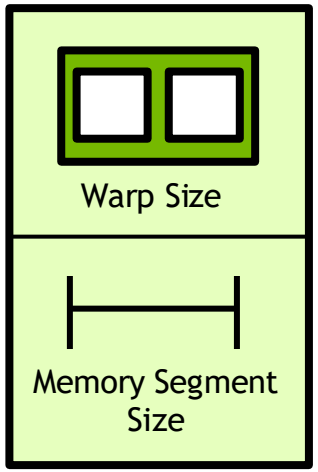


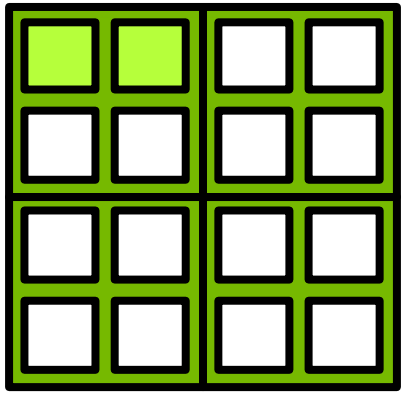






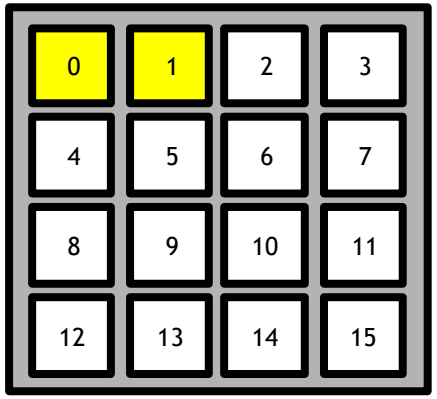

```
x = blockIdx.x * blockDim.x + threadIdx.x

y = blockIdx.y * blockDim.y + threadIdx.y

out[x][y] = in[y][x]
```

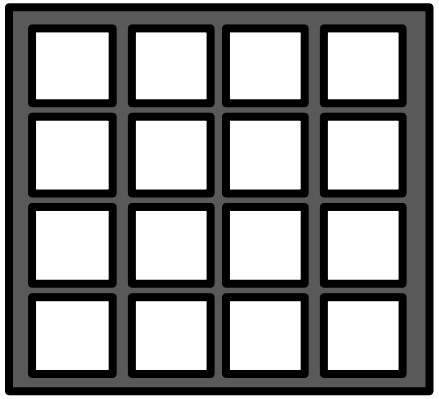


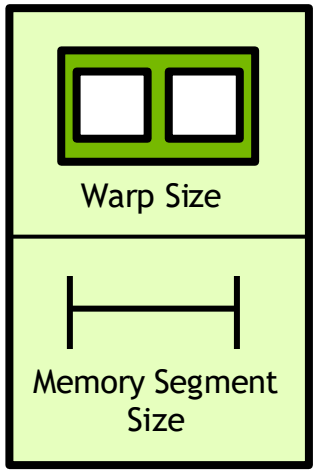


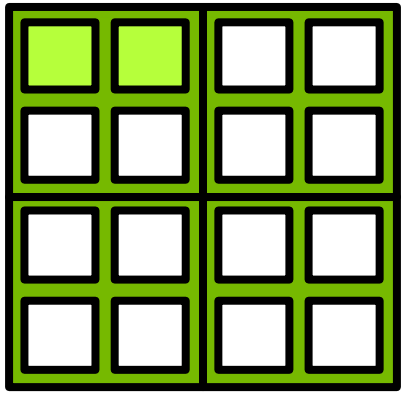
Warp

```
x = blockIdx.x * blockDim.x          +
threadIdx.x

y = blockIdx.y * blockDim.y          +
threadIdx.y

out[x][y] = in[y][x]
```



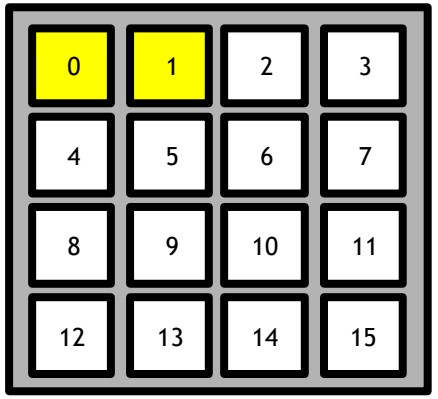


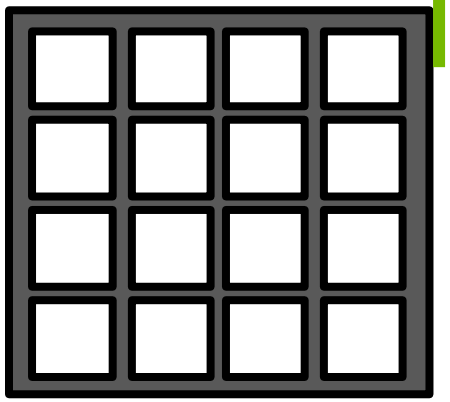


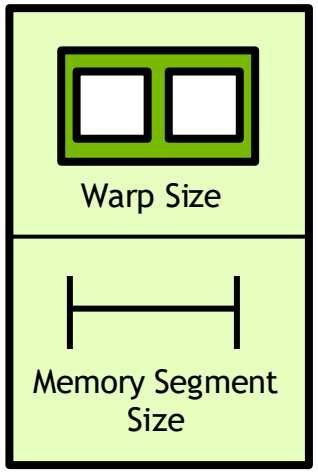


Warp

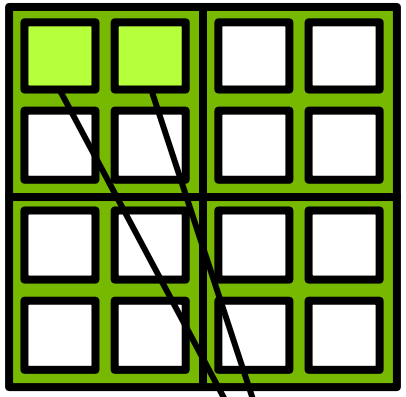
```
x = blockIdx.x * blockDim.x + threadIdx.x

y = blockIdx.y * blockDim.y + threadIdx.y

out[x][y] = in[y][x]
```



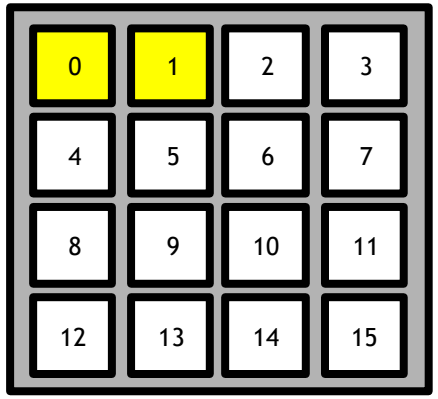


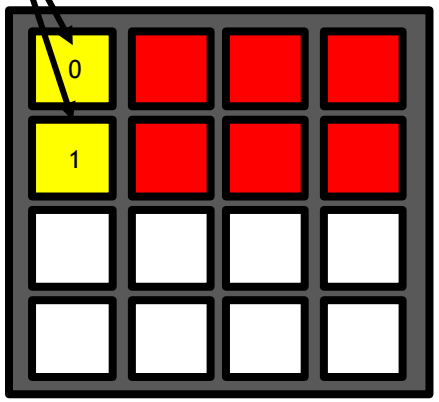


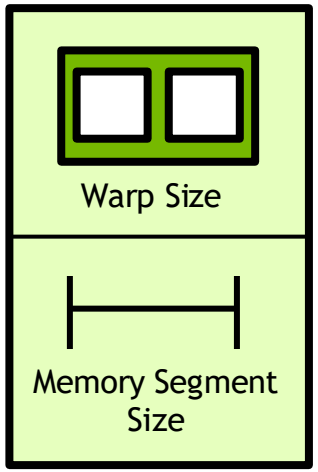
与

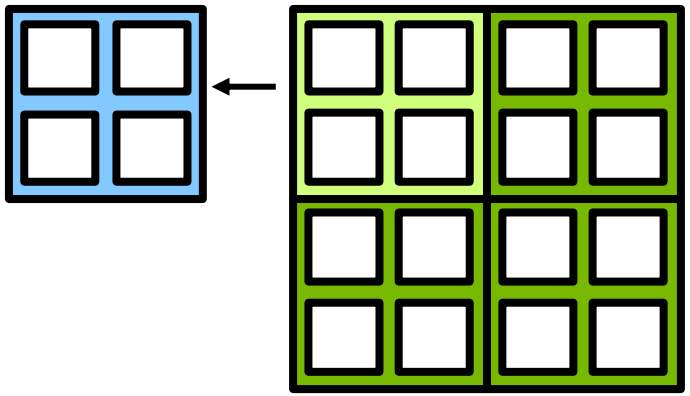


```
x = blockIdx.x * blockDim.x + threadIdx.x

y = blockIdx.y * blockDim.y + threadIdx.y

out[x][y] = in[y][x]
```



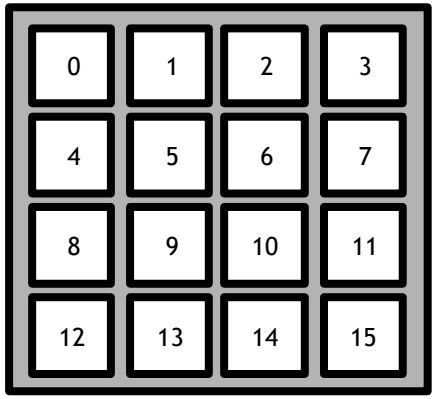


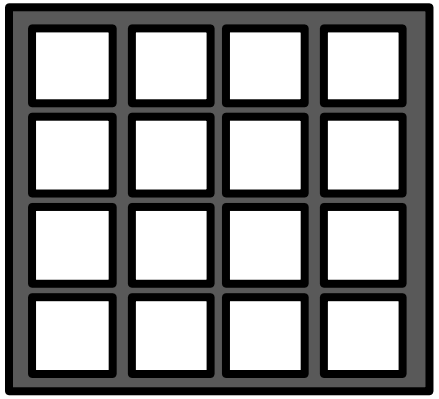


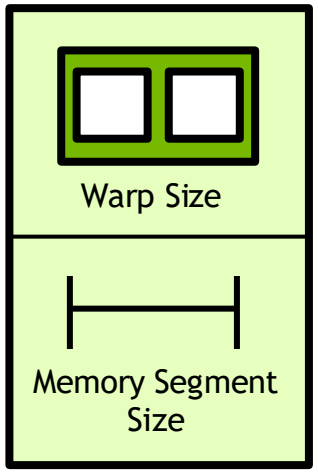


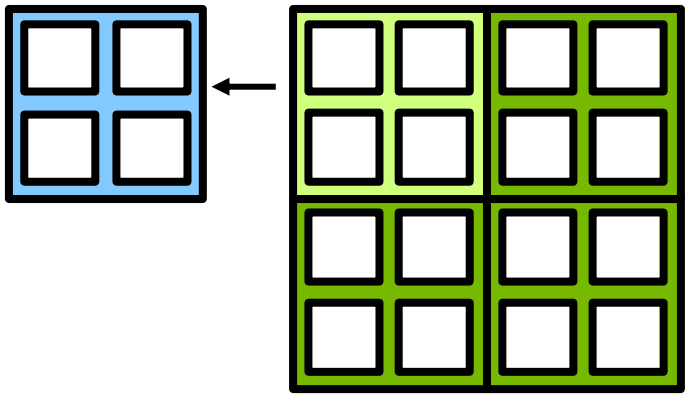
2,2

```
tile = cuda.shared.array(2,2)
```





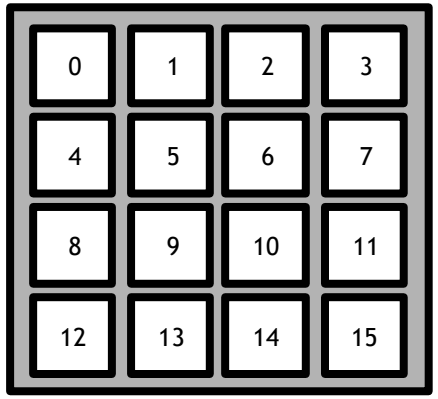


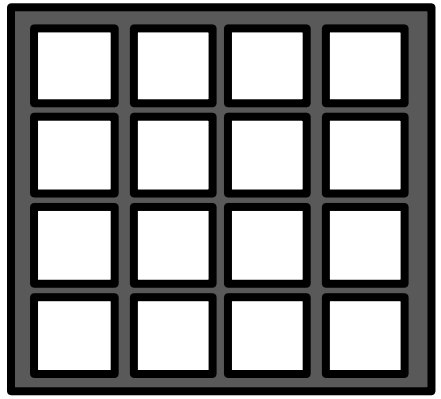


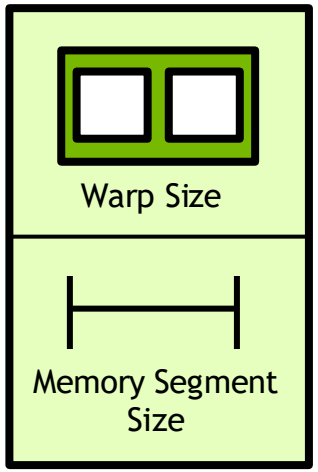
Warp 2

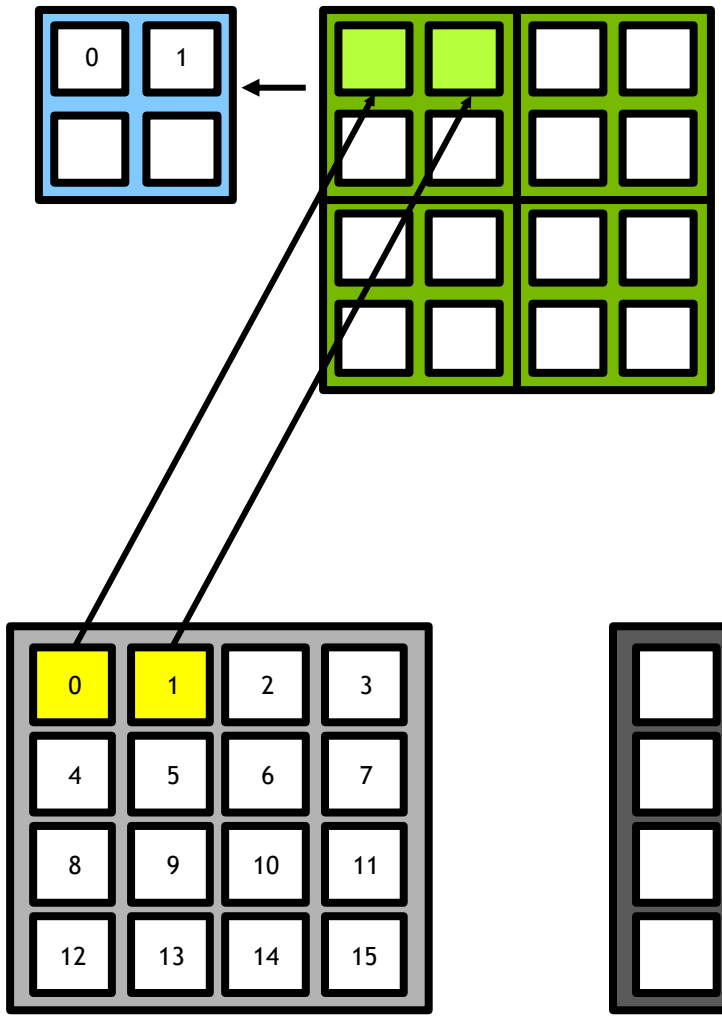
Warp 32

```
tile = cuda.shared.array(2,2)
```

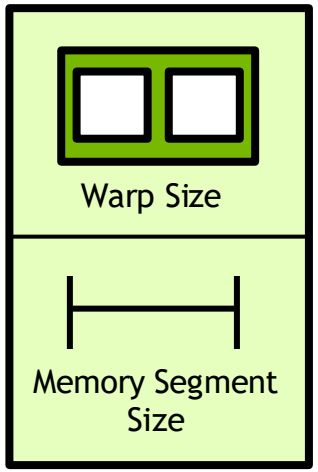


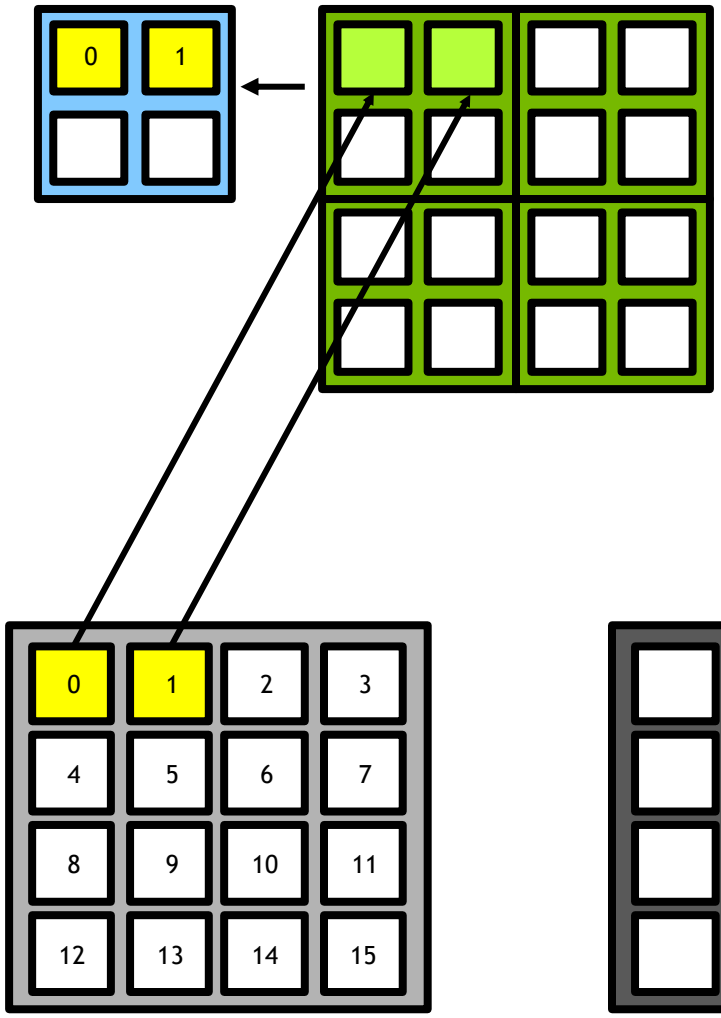







```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
```

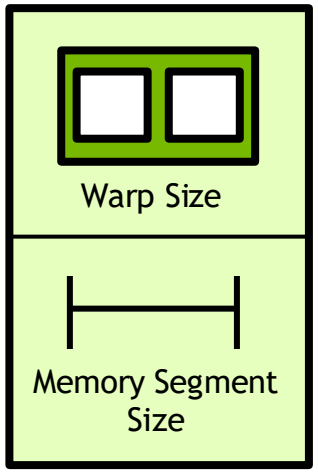


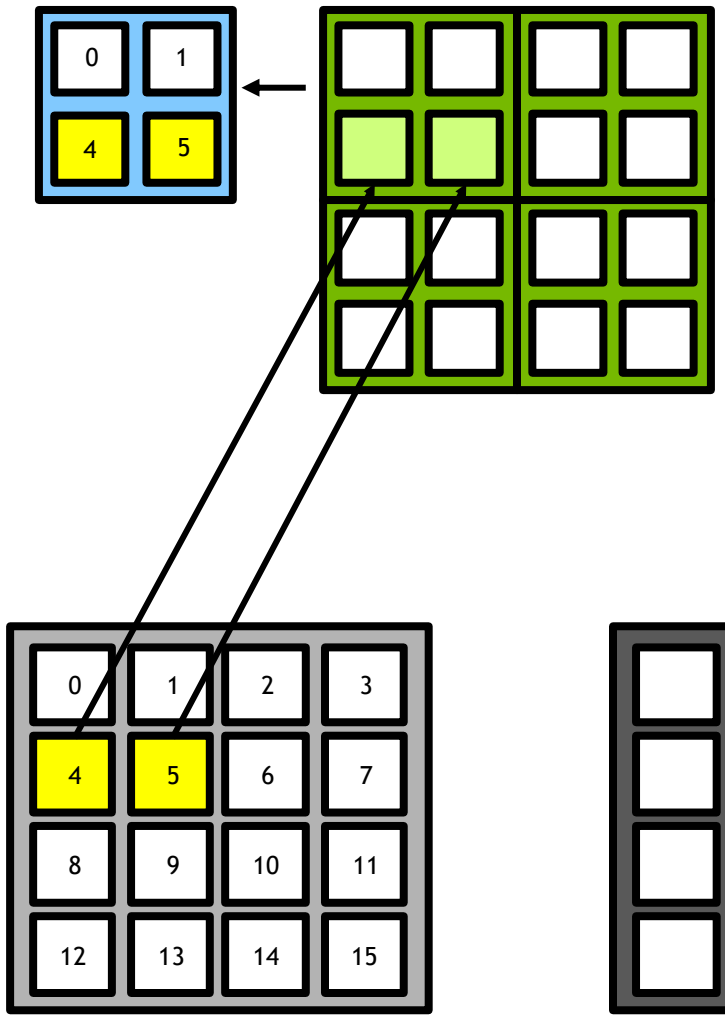


)与

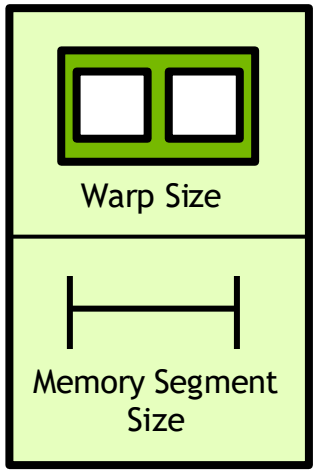
与

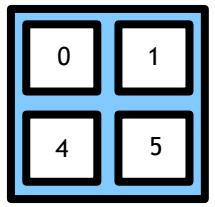
```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
```

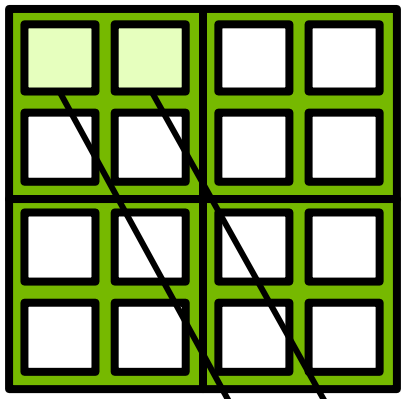





```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()
```

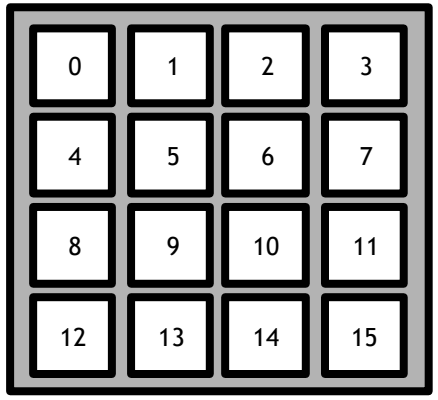


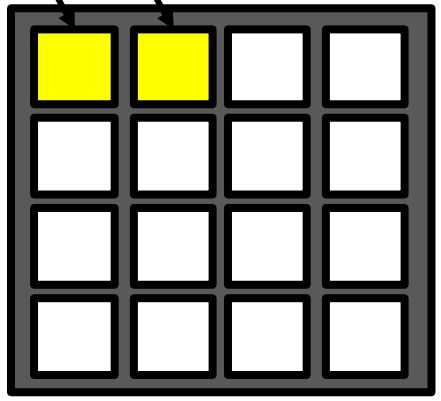


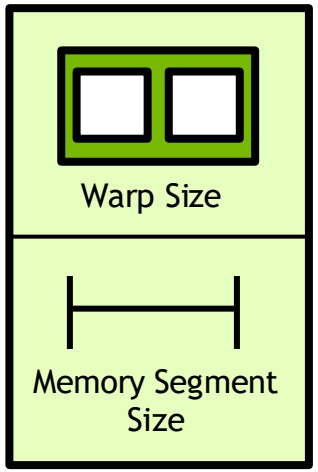


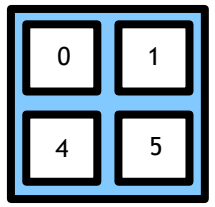
Warp

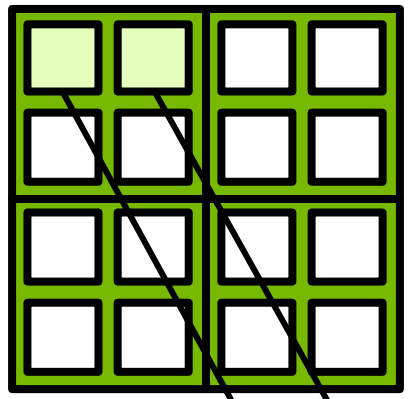
```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
```





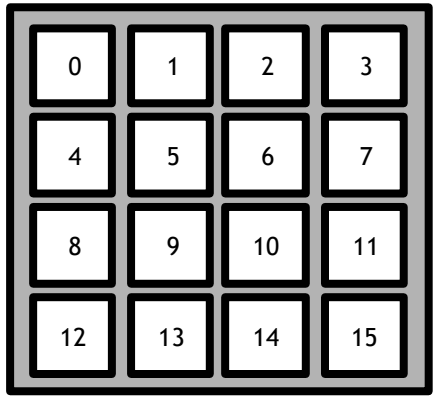


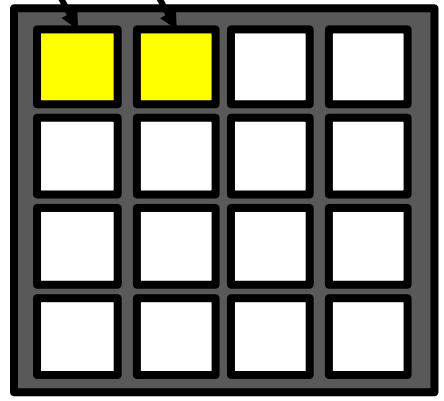


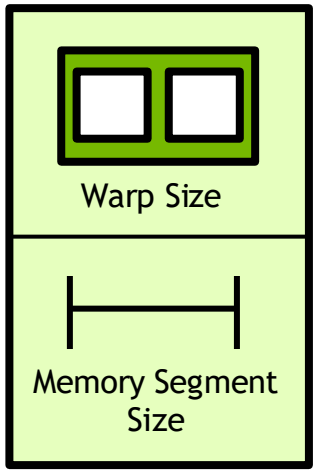


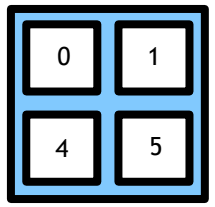
blockIdx.y blockDim.y
x .....

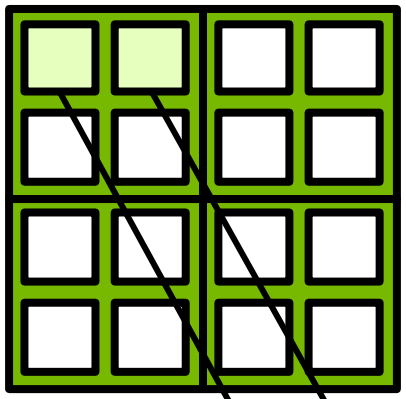
```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
```

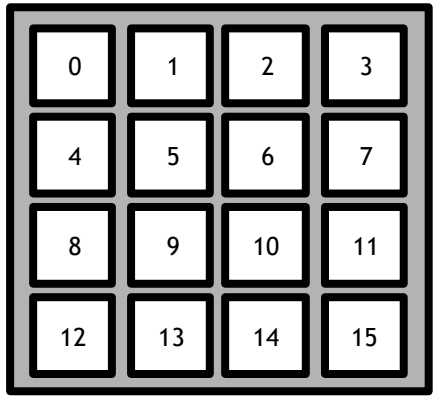


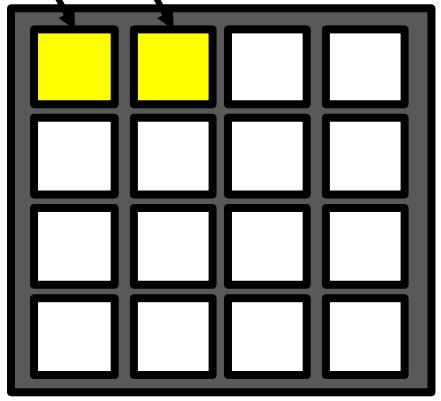






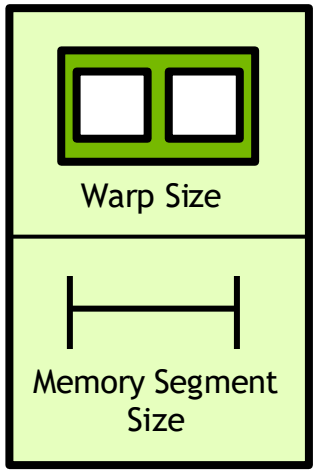


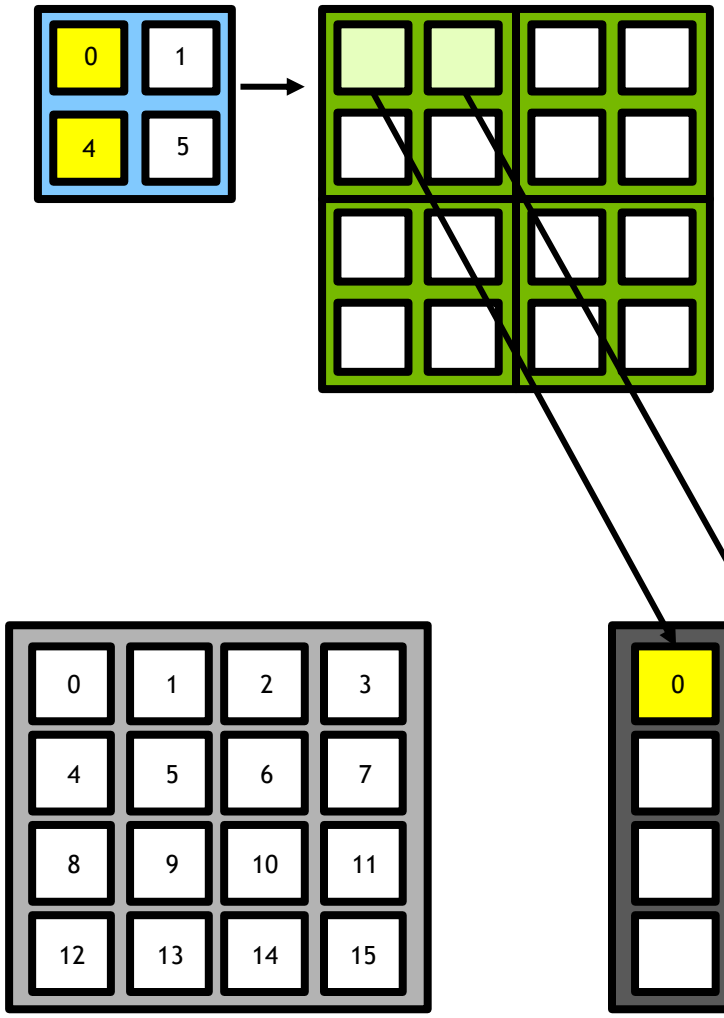




... 我 从
x
threadIdx.x

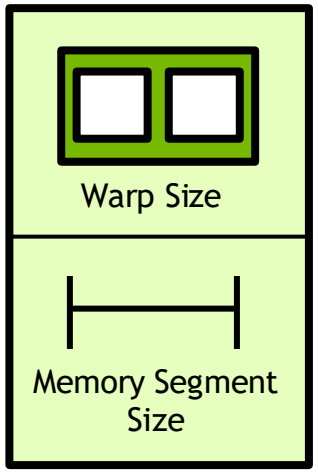
```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
```

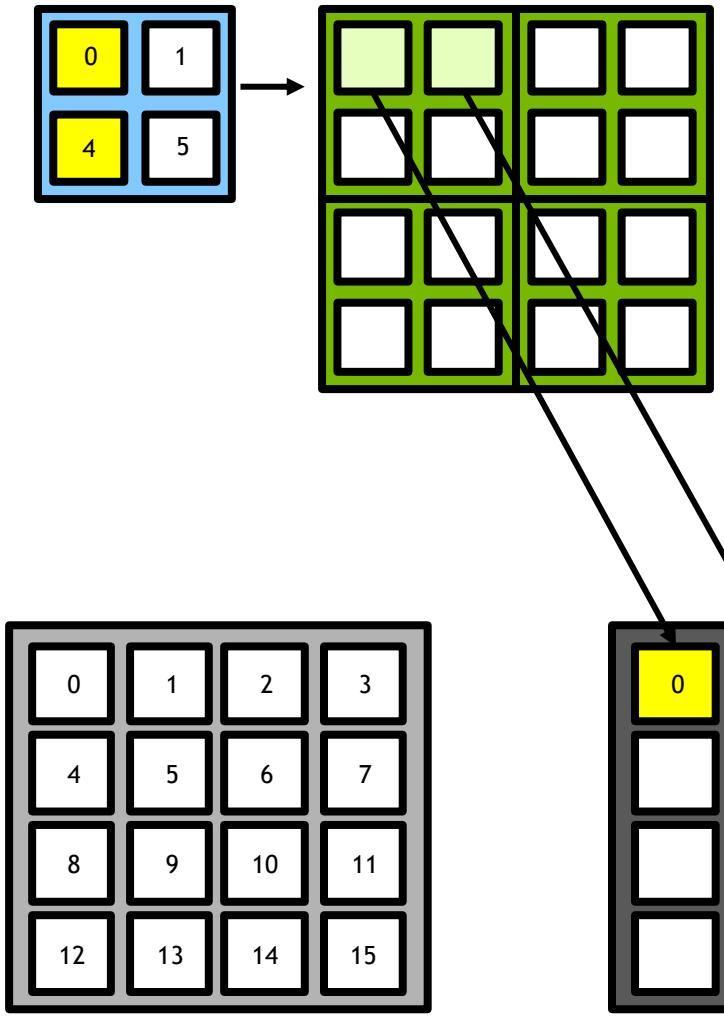




Warp

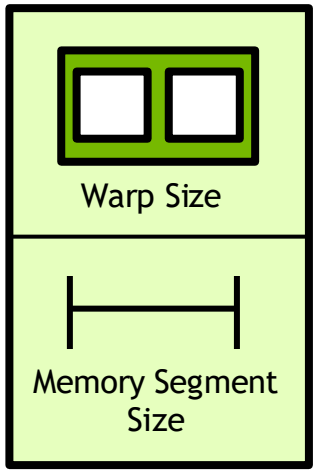
```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
o[o_y][o_x] = tile[tIdx.x][tIdx.y]
```

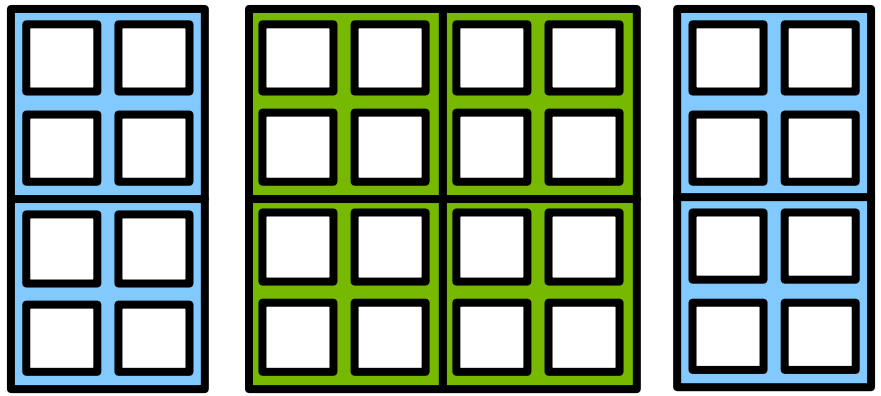




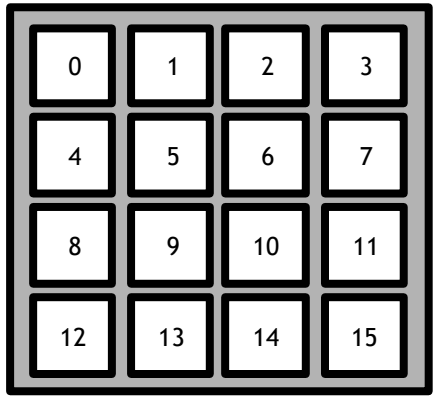
/

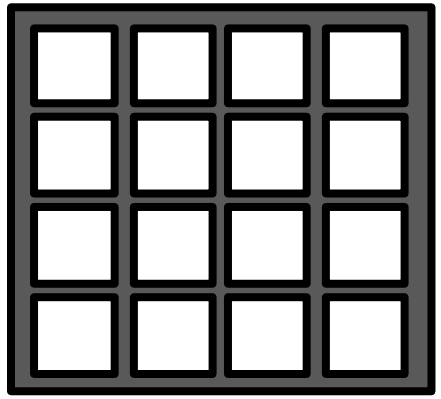
```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
o[o_y][o_x] = tile[tIdx.x][tIdx.y]
```

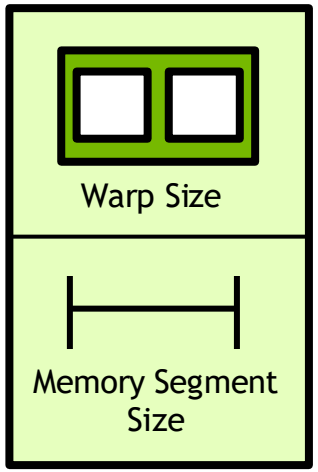


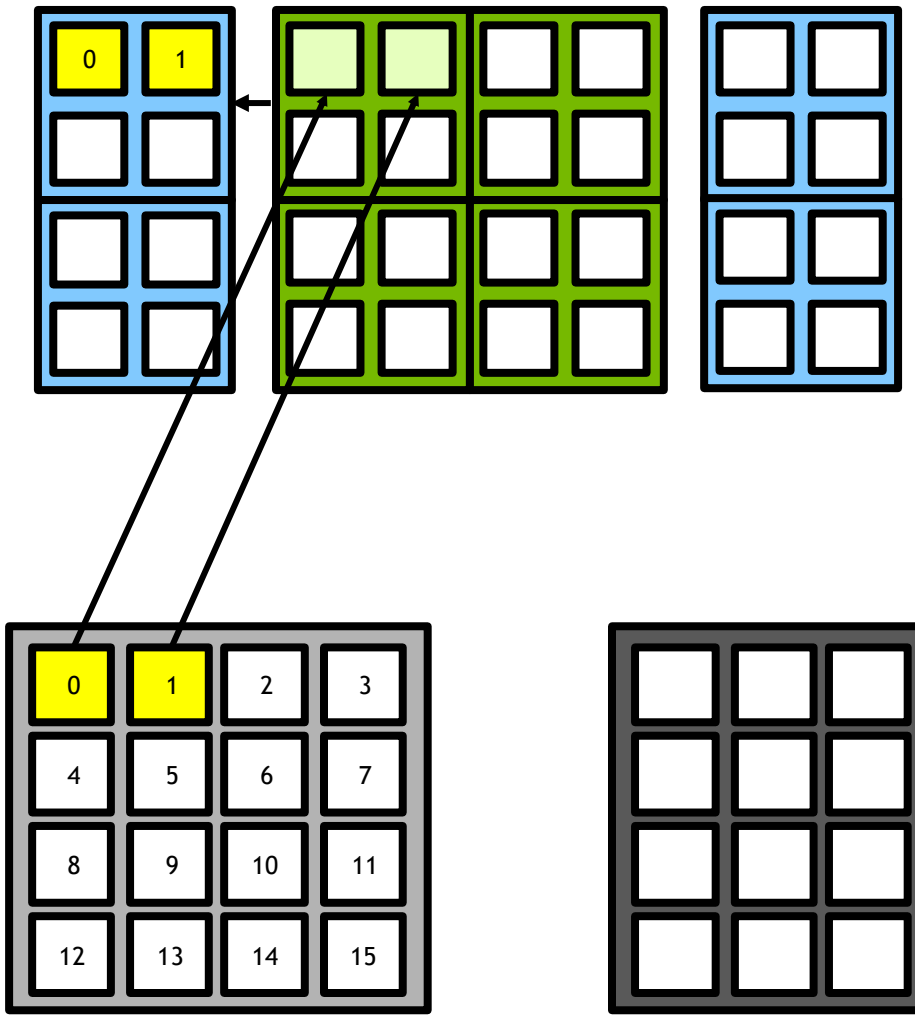


```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
o[o_y][o_x] = tile[tIdx.x][tIdx.y]
```

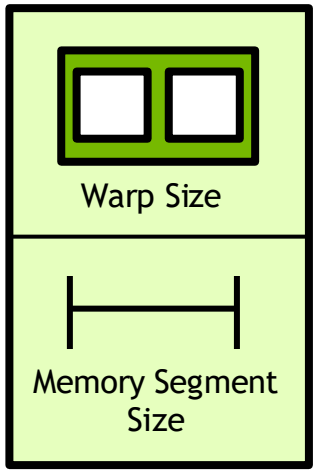


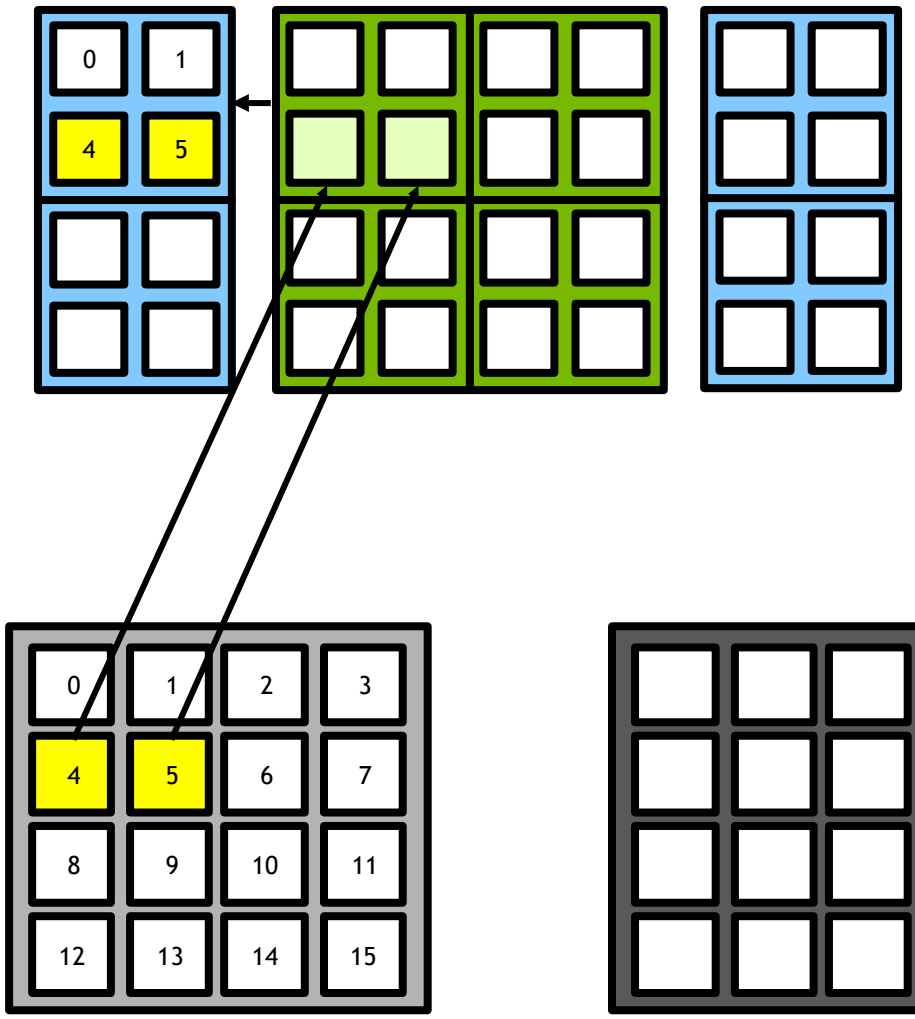




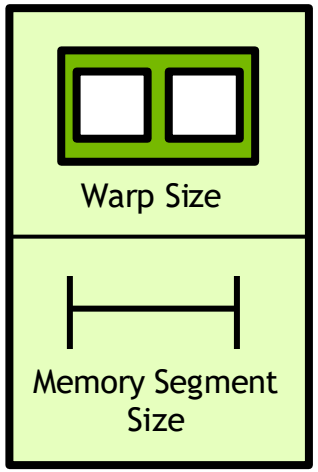


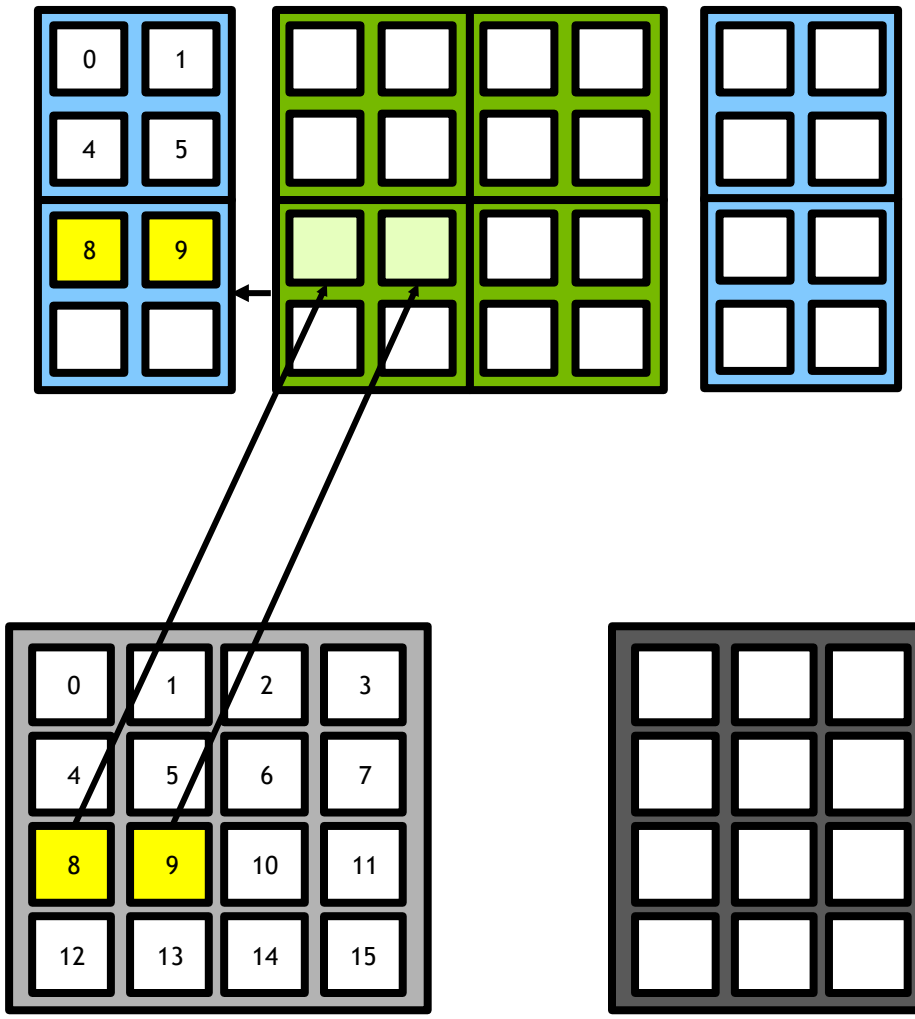

```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
o[o_y][o_x] = tile[tIdx.x][tIdx.y]
```



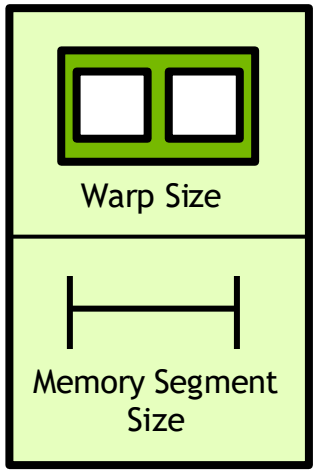


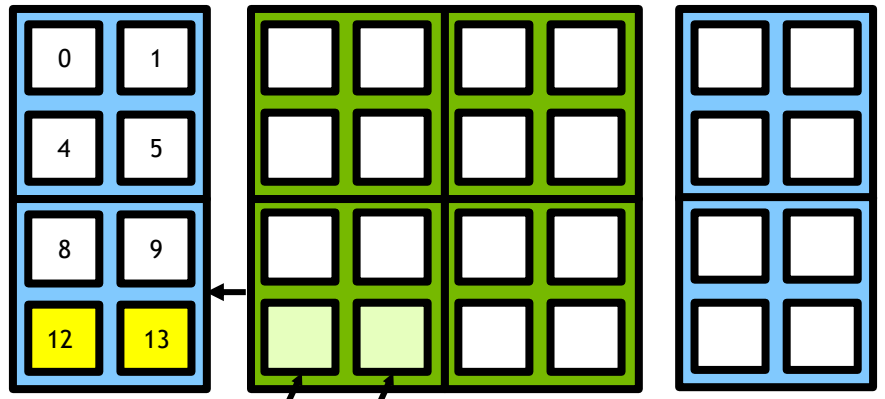

```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
o[o_y][o_x] = tile[tIdx.x][tIdx.y]
```

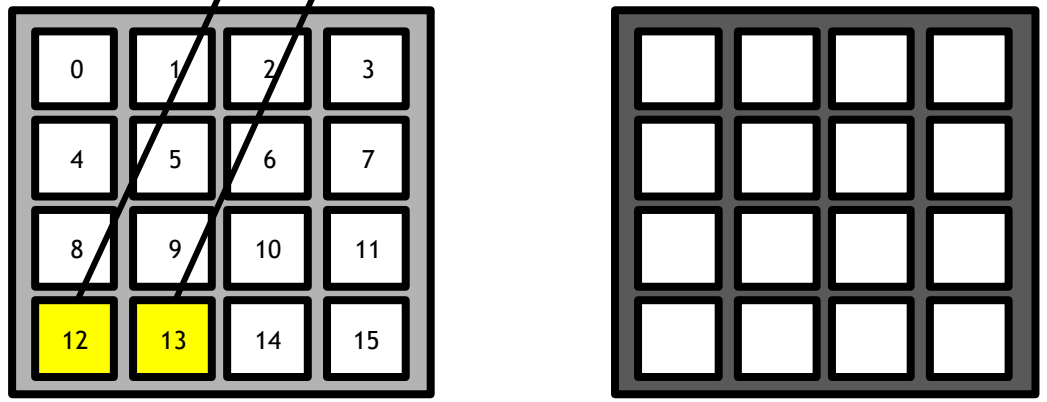




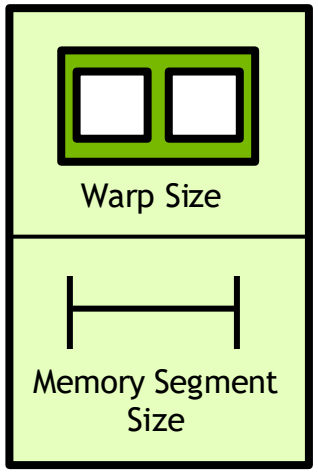

```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
o[o_y][o_x] = tile[tIdx.x][tIdx.y]
```

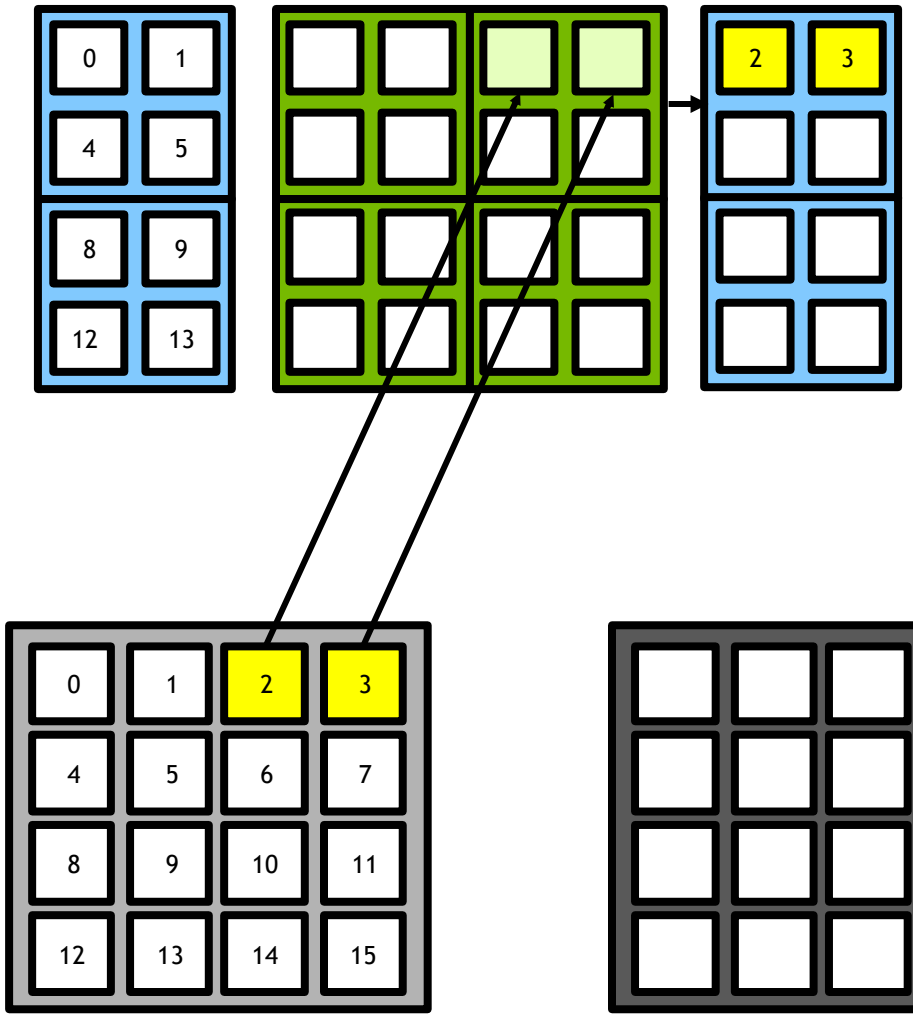




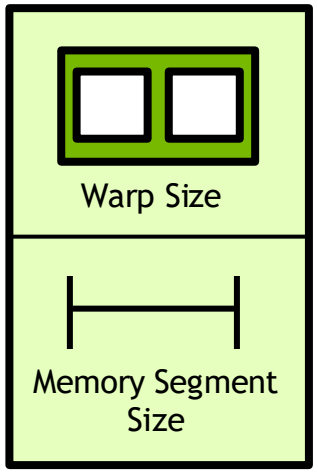


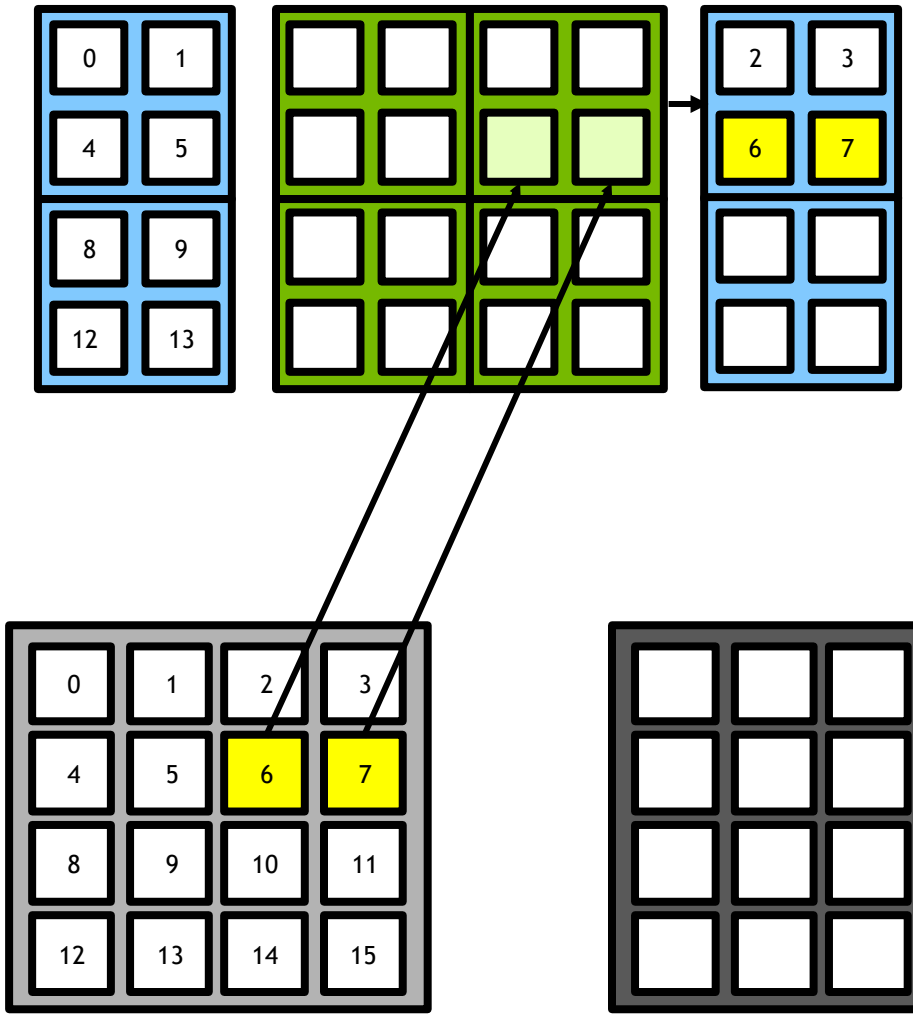

```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
o[o_y][o_x] = tile[tIdx.x][tIdx.y]
```



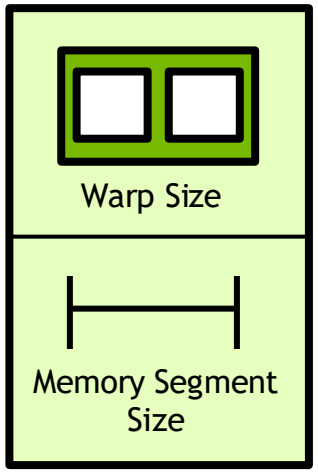


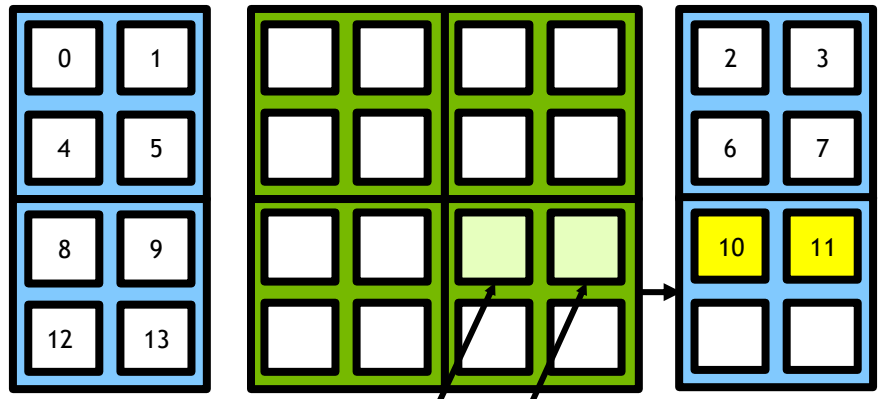

```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
o[o_y][o_x] = tile[tIdx.x][tIdx.y]
```



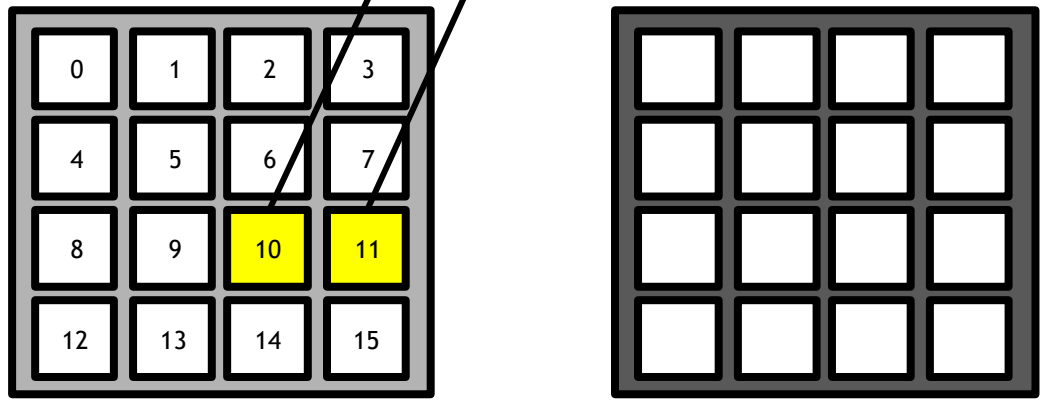


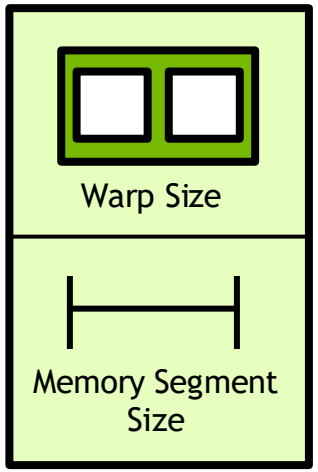

```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
o[o_y][o_x] = tile[tIdx.x][tIdx.y]
```

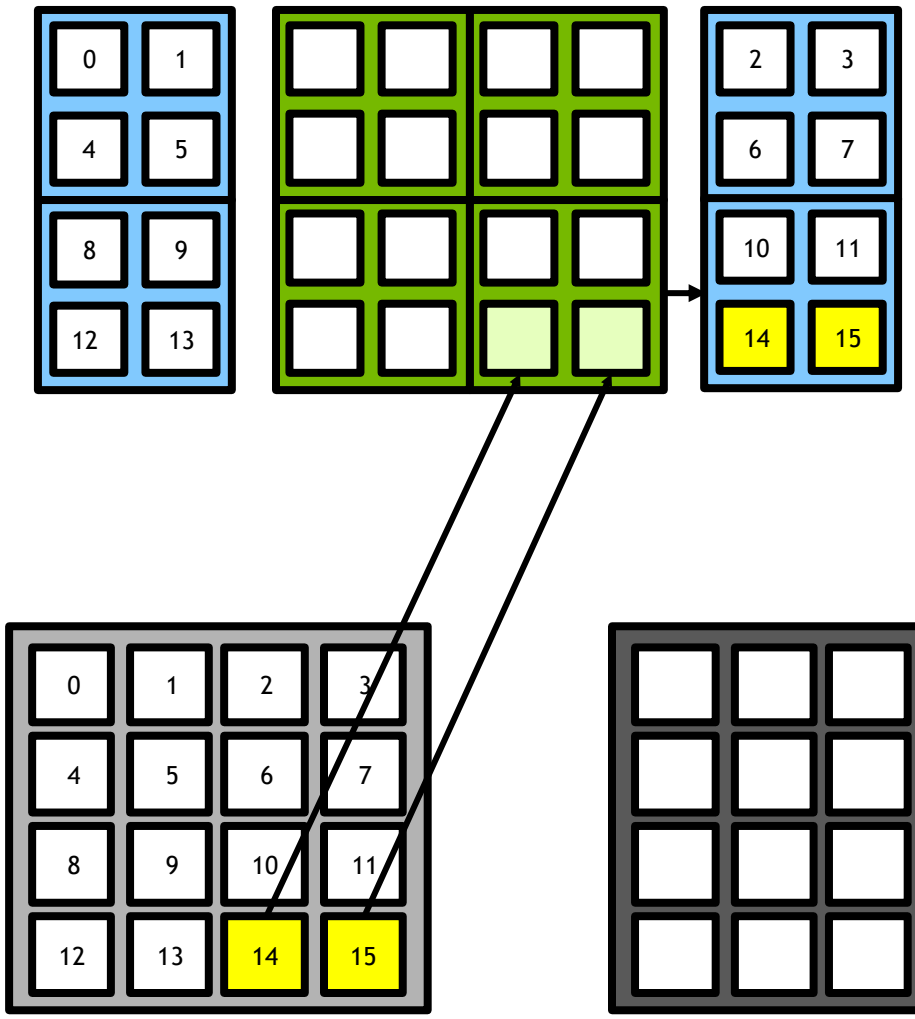




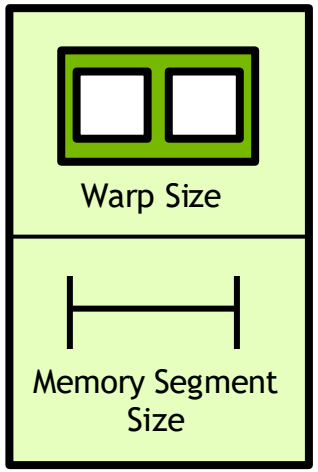

```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
o[o_y][o_x] = tile[tIdx.x][tIdx.y]
```

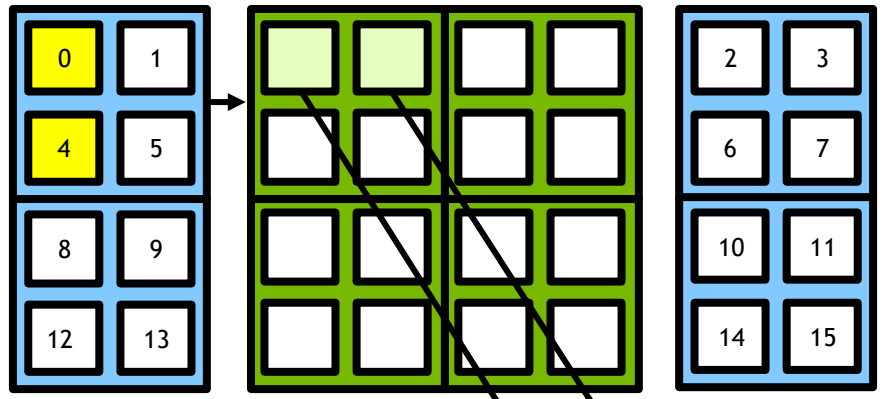




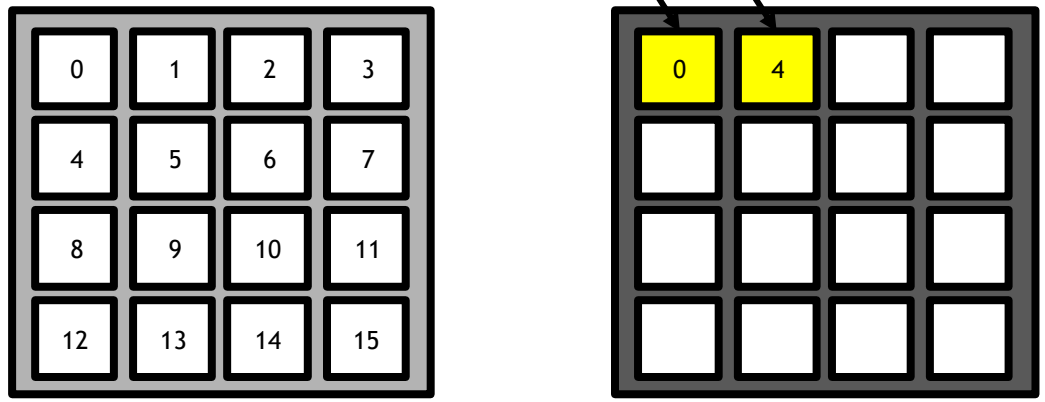


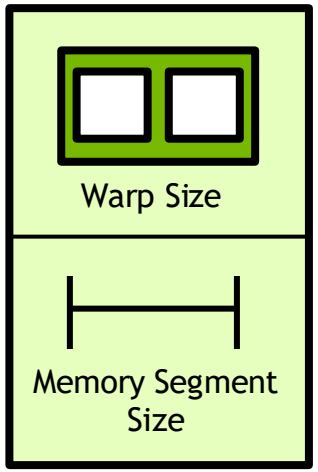

```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
o[o_y][o_x] = tile[tIdx.x][tIdx.y]
```

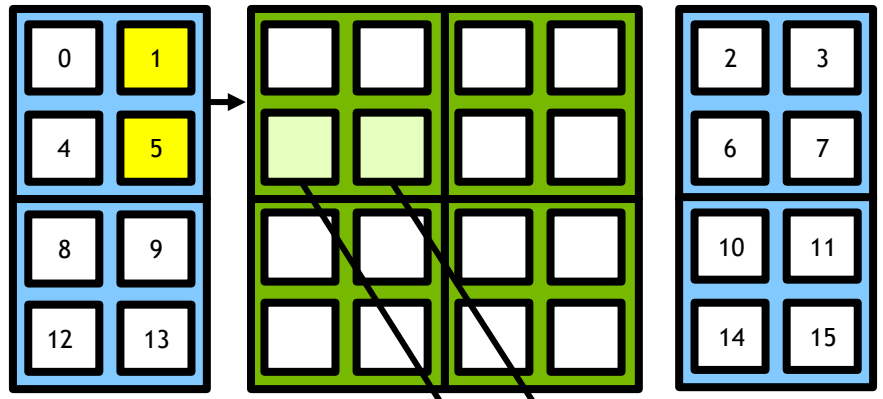




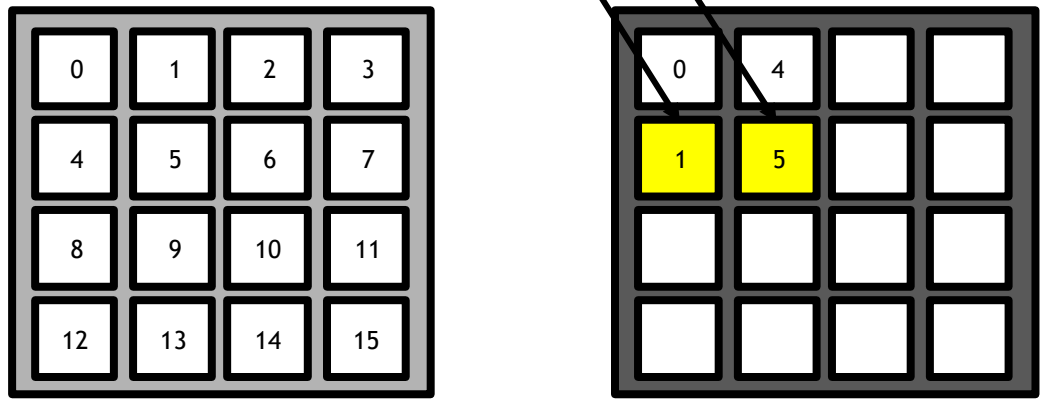

```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
o[o_y][o_x] = tile[tIdx.x][tIdx.y]
```

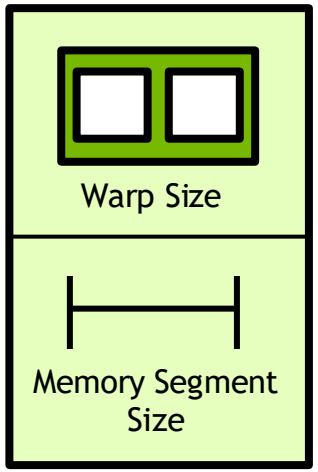


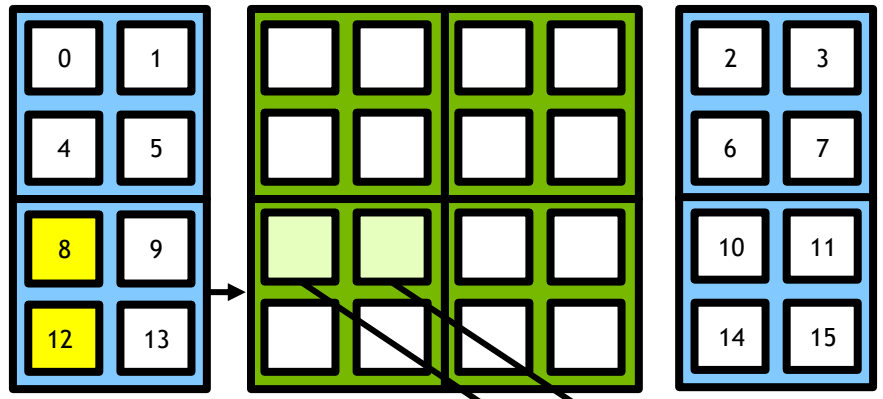




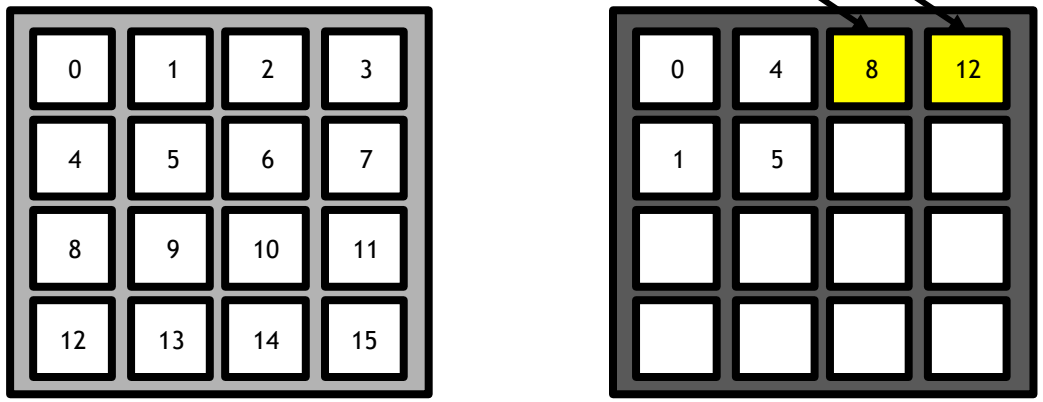

```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
o[o_y][o_x] = tile[tIdx.x][tIdx.y]
```

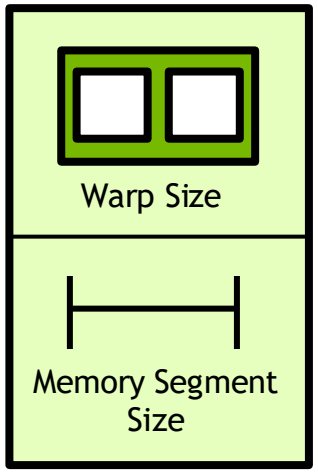


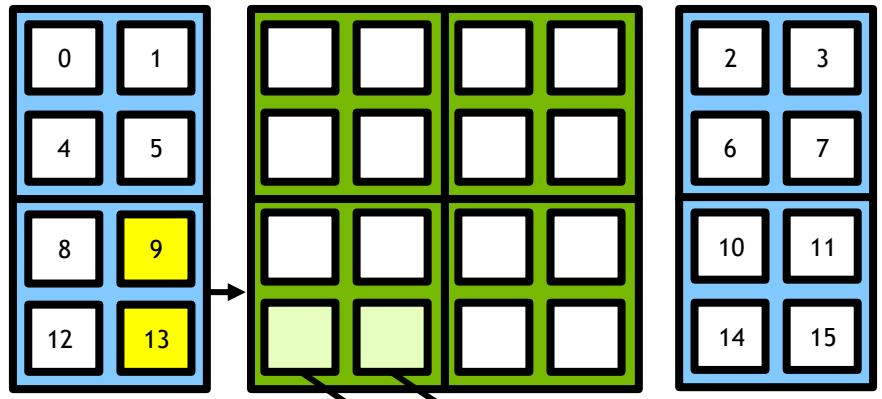




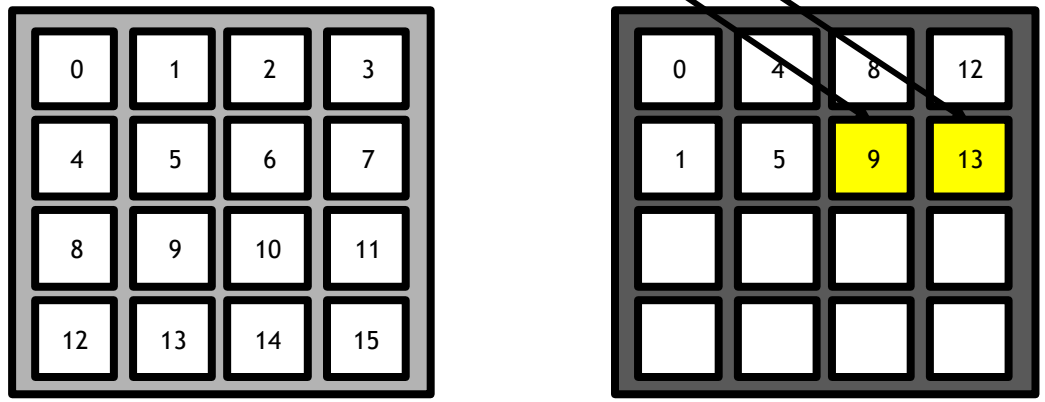

```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
o[o_y][o_x] = tile[tIdx.x][tIdx.y]
```

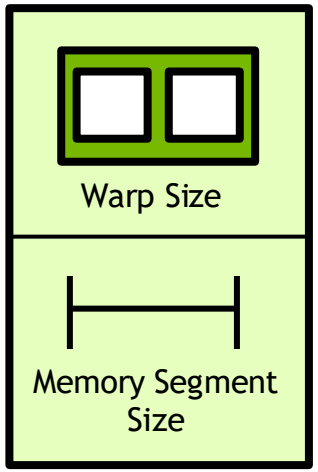


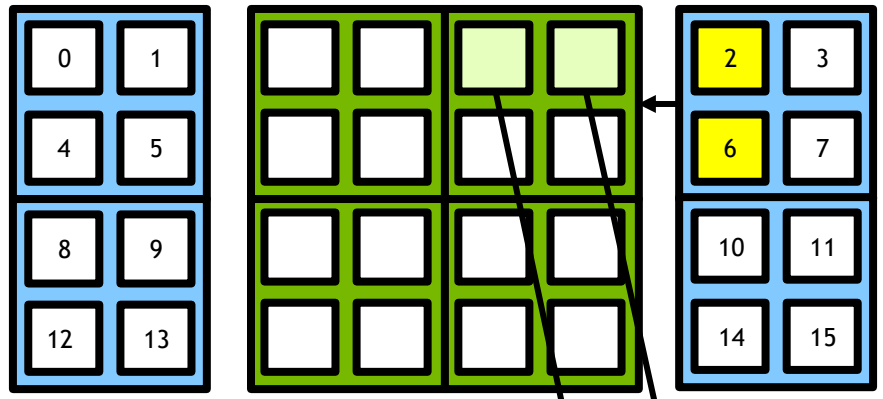




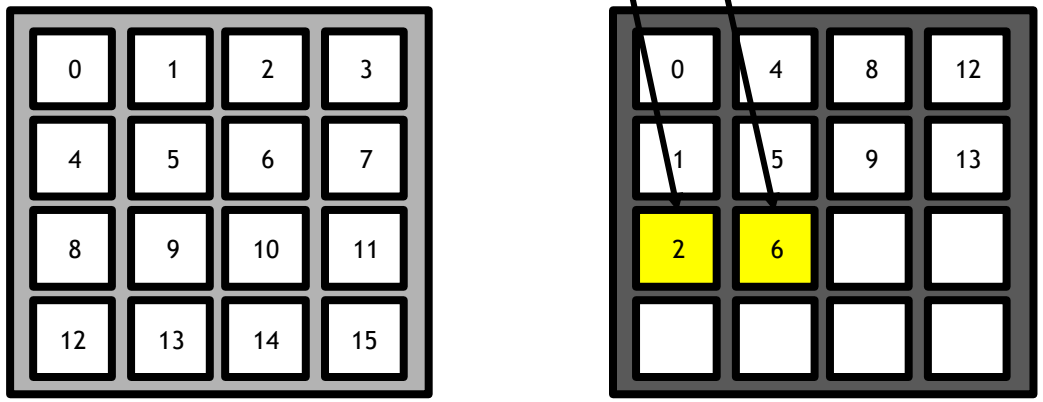

```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
o[o_y][o_x] = tile[tIdx.x][tIdx.y]
```

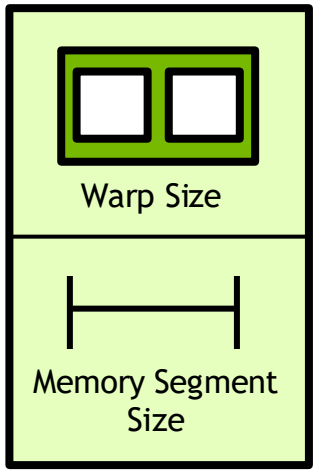


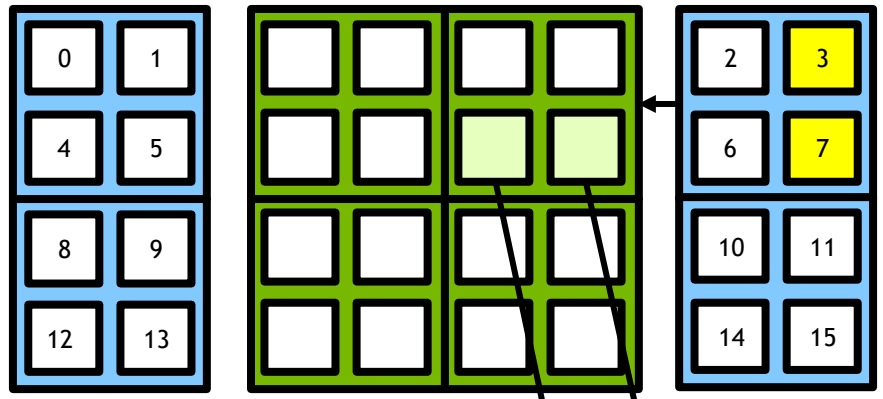




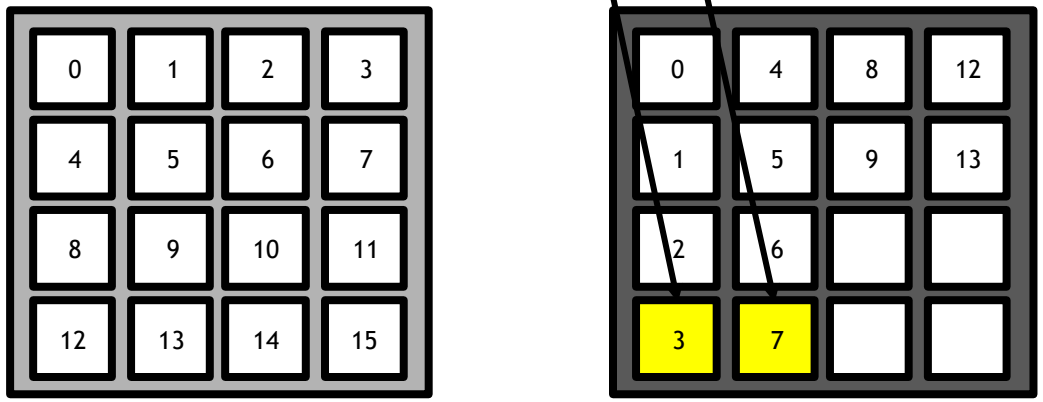

```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
o[o_y][o_x] = tile[tIdx.x][tIdx.y]
```

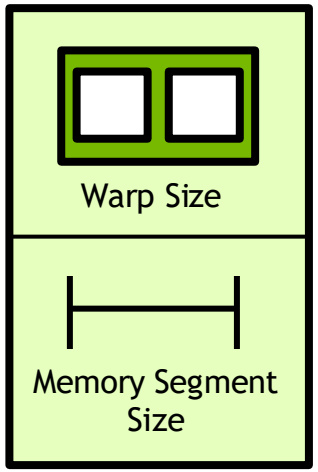


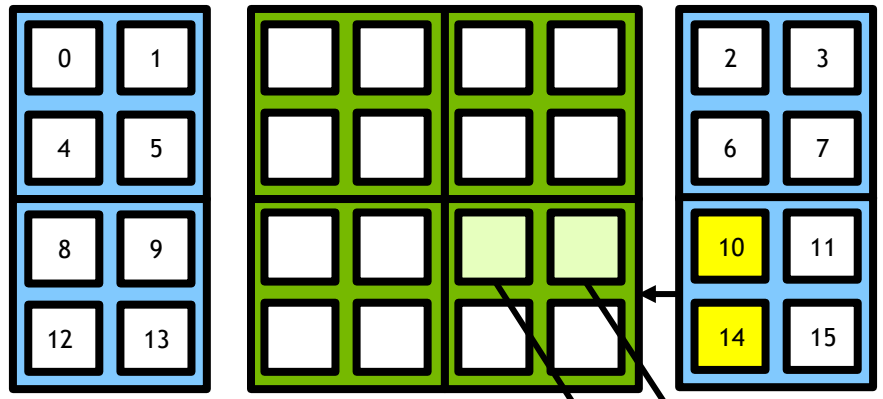




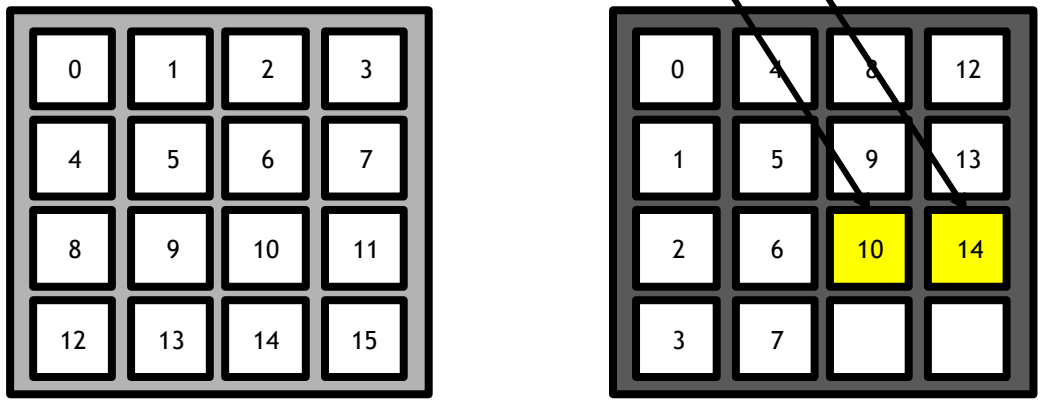

```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
o[o_y][o_x] = tile[tIdx.x][tIdx.y]
```

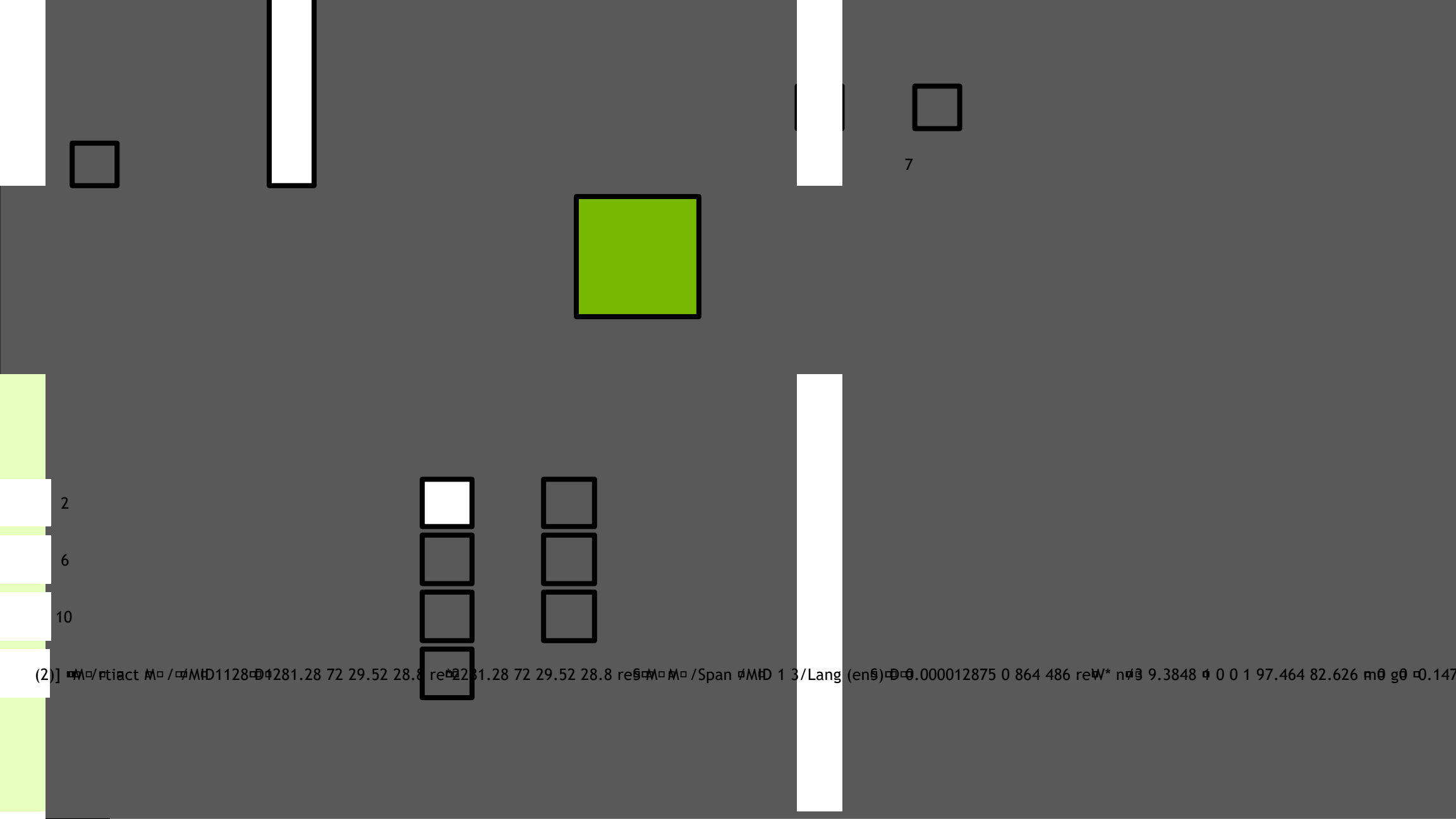




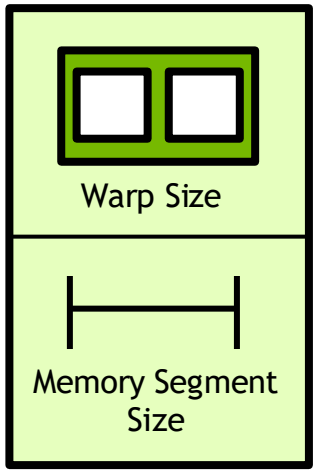


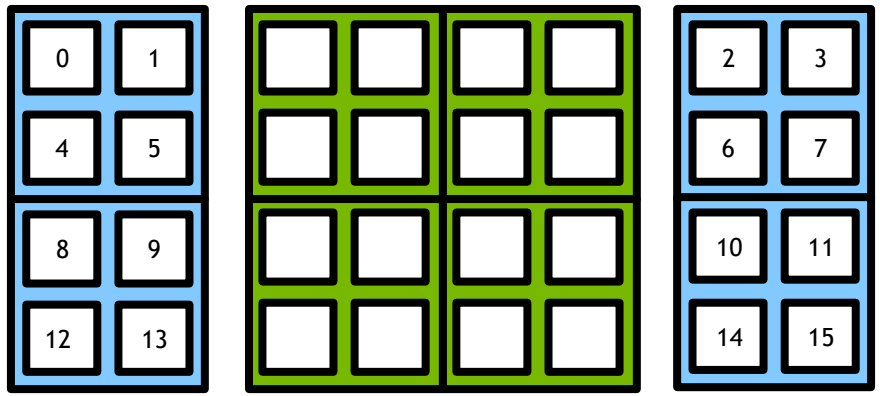

```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
o[o_y][o_x] = tile[tIdx.x][tIdx.y]
```



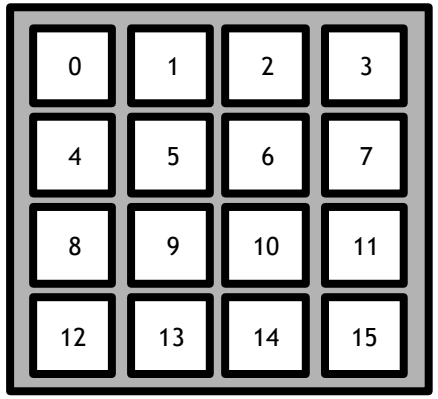


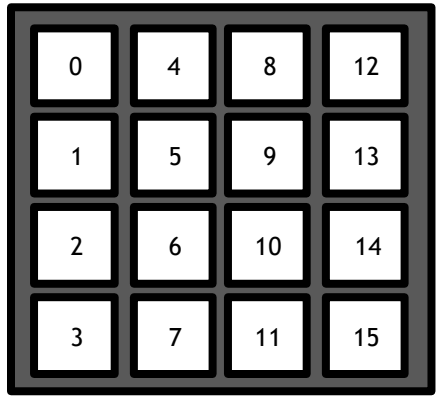
(2)] ▫M▫/▫tiact M▫ /▫/MHD1128▫D▫1281.28 72 29.52 28.8 re▫2281.28 72 29.52 28.8 re▫S▫M▫ M▫ /Span /MHD 1 3/Lang (en▫)▫D▫0.000012875 0 864 486 re▫W* n▫3 9.3848 ▫ 0 0 1 97.464 82.626 ▫m 0 g0 ▫0.147






```
tile = cuda.shared.array(2,2)
x, y = cuda.grid(2)

tile[tIdx.y][tIdx.x] = in[y][x]
cuda.syncthreads()

o_x = bId.y*bDim.y + tId.x
o_y = bId.x*bDim.x + tId.y
o[o_y][o_x] = tile[tIdx.x][tIdx.y]
```





NVIDIA | DEEP LEARNING INSTITUTE

www.nvidia.cn/DLI