Harold Molina-Bulla DTSC-UC3M

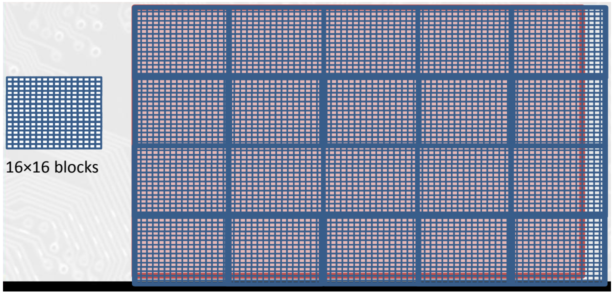
HETEROGENEOUS COMPUTATION BASED ON GPU

INTRODUCTION

Harold Molina-Bulla DTSC-UC3M

PROCESSING 2D MATRICES

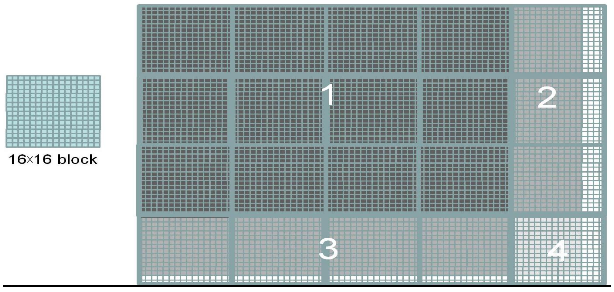
- Example: Define blocks of 16x16 blocks (256 parallel threads)
- Define N blocks, where N x Block size must fit in number of cores available in device.
- Each position will be addressed by r/o index provided by the execution context.



Harold Molina-Bulla DTSC-UC3M

PROCESSING 2D MATRICES

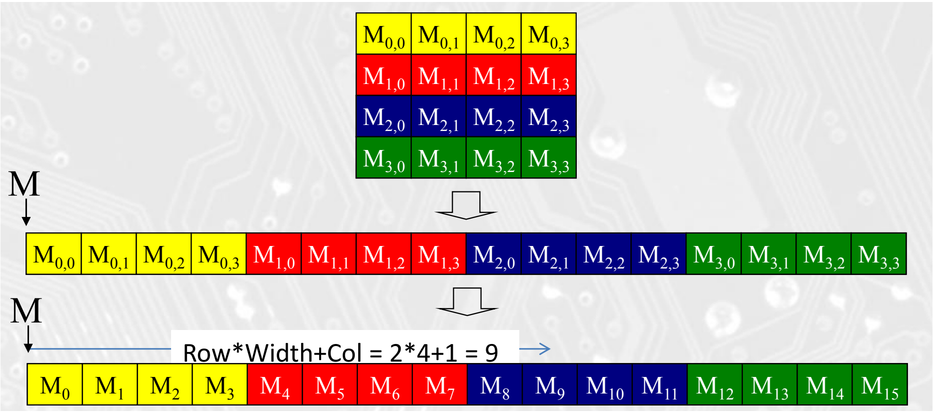
- If block not fits in data:
  - Check if indexes index a defined memory position
    - If yes, execute program
    - If not, return null
  - BIG PROBLEM HERE
    - The GPU does not have branch prediction, then double the execution time.



Harold Molina-Bulla DTSC-UC3M

MEMORY INDEXING

All the data is a vector, then you have to construct the memory position using the indexes



$M_{0,0}$	$M_{0,1}$	$M_{0,2}$	$M_{0,3}$
$M_{1,0}$	$M_{1,1}$	$M_{1,2}$	$M_{1,3}$
$M_{2,0}$	$M_{2,1}$	$M_{2,2}$	$M_{2,3}$
$M_{3,0}$	$M_{3,1}$	$M_{3,2}$	$M_{3,3}$

M

$M_{0,0}$	$M_{0,1}$	$M_{0,2}$	$M_{0,3}$	$M_{1,0}$	$M_{1,1}$	$M_{1,2}$	$M_{1,3}$	$M_{2,0}$	$M_{2,1}$	$M_{2,2}$	$M_{2,3}$	$M_{3,0}$	$M_{3,1}$	$M_{3,2}$	$M_{3,3}$

M

Row*Width+Col = 2*4+1 = 9

M_0	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}	M_{11}	M_{12}	M_{13}	M_{14}	M_{15}

Harold Molina-Bulla DTSC-UC3M

CALCULATE VECTOR POSITION

- GOAL: Calculate the position where the data will be stored in a row-wise representation of a 2 dimensional matrix
- How?
  - Based on the system variables blockDim, threadIdx and blockIdx, calculate the position in the matrix
  - Check if the position is a valid value (to be coded by the students)
  - If it is valid, write in the result vector, the value of his position
- Check in the notebook

Harold Molina-Bulla DTSC-UC3M

SQUARE VECTOR

- GOAL: Calculate the square value of input vector
- How

1. Calculate the position assigned using the blockDim, blockIdx and threadIdx,
2. Check if the memory position is valid
3. Fetch the value from the original vector a, calculate it square and save in the result vector b

MATRIX MULTIPLICATION

- GOAL: Multiply 2 matrices of fixed size and recover the result
- How:
- First: each thread will calculate each cell in the result matrix c

1. Calculate the assigned cell coordinates
2. Check if the coordinates are valid
3. If yes, use those coordinates to walk through the vector representation of the matrix a and b, and calculate the product and sum of each term
4. Assign the result to the vector position in result vector c.