GPU Computing with CUDA
Lab 1 - CUDA threads

Christopher Cooper
Boston University

August, 2011
UTFSM, Valparaíso, Chile

Objectives

- Familiarize yourself to CUDA
- Understand the different steps in a CUDA program
- Understand how threads work

Vector add

- Implement a vector add
- Steps
  - Allocate and initialize array on CPU
  - Allocate on the GPU with cudaMalloc
  - Copy from CPU to GPU with cudaMemcpy
  - Define number of blocks and threads per block
  - Launch kernel
  - Copy results to CPU with cudaMemcpy

Vector add

- Save file as *filename*.cu
- Compile with nvcc

```
nvcc filename.cu -o file
```

Matrix-Matrix multiplication

- This time, you will only write the kernel
- We will be using the following struct

```
typedef struct
{
  int width;
  int height;
  float *elements;
} Matrix;
```

- Elements is a 1D array with the elements of the matrix flattened out
- Considering row major

```
global_index = y_index*width + x_index
```

Matrix-Matrix multiplication

- Idea
  - Each thread will solve one element of the result matrix. Each thread will loop through a row of one matrix and a column of the other

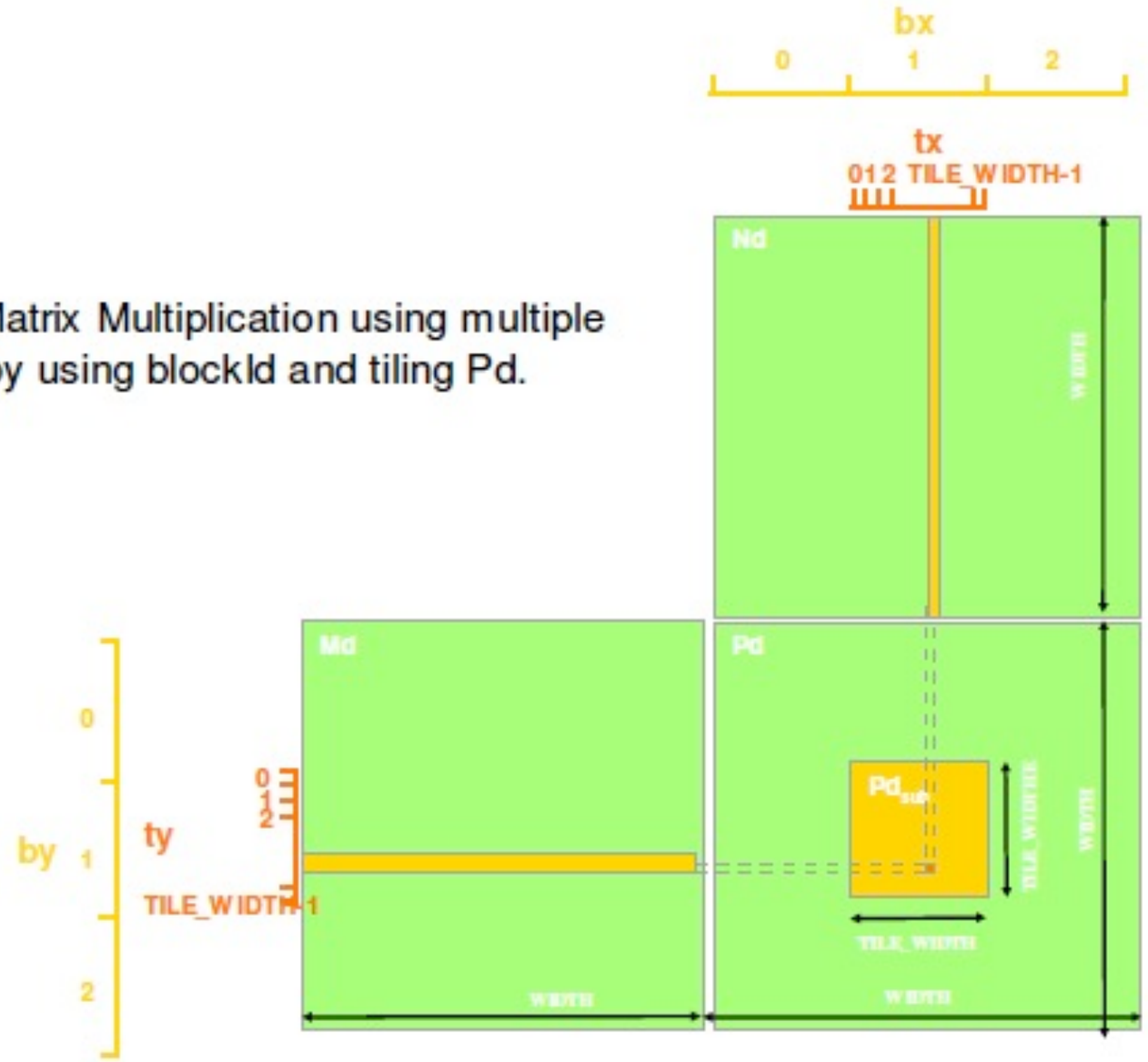

Figure 3.2 Matrix Multiplication using multiple blocks by using blockId and tiling Pd.