Intensive Computation

Prof. A. Massini

June 9, 2023

PART A

Student's Name

_____

Matricola number

_____

Exercise 1 (6 points)	
Exercise 2 (3 points)	
Exercise 3 (5 points)	
Exercise 4 (4 points)	
Exercise 5 (6 points)	
Exercise 6 (4 points)	
Question (4 points)	
Total (32 points)	

Exercise 1 (6 points) - GPU & CUDA

Consider a matrix of size 1800x1800. You would like to assign one thread to each matrix element.

a) How would you select the **2D grid** dimensions and **2D square block** dimensions of your kernel to **minimize the number of idle threads** on a device having **compute capability 1.3**? and on a device having **compute capability 3.5**?

b) How would you select the 2D block dimensions of your kernel if you want tridimensional blocks of maximum size and you do not need blocks are square, on a device having **compute capability 1.3**?

Technical specifications	Compute capability (version)									
	1.0	**1.1**	**1.2**	**1.3**	**2.x**	**3.0**	**3.5**	**3.7**	**5.0**	**5.2**
Maximum dimensionality of grid of thread blocks	2					3				
Maximum x-dimension of a grid of thread blocks	65535					2^{31}-1				
Maximum y-, or z-dimension of a grid of thread blocks	65535									
Maximum dimensionality of thread block	3									
Maximum x- or y-dimension of a block	512				1024					
Maximum z-dimension of a block	64									
Maximum number of threads per block	512				1024					
Warp size	32									
Maximum number of resident blocks per multiprocessor	8						16		32	
Maximum number of resident warps per multiprocessor	24		32		48		64			
Maximum number of resident threads per multiprocessor	768		1024		1536		2048			
Technical specifications	**1.0**	**1.1**	**1.2**	**1.3**	**2.x**	**3.0**	**3.5**	**3.7**	**5.0**	**5.2**
	Compute capability (version)									

Exercise 2 (3 points) – Pipeline

Consider an architecture where each instruction (unpipelined) takes 84 ns. Consider the pipeline implementation of instructions taking 102 ns, using 6 pipe stages.

i) Compute the time required to execute 120 instructions without and with pipeline.

ii) Compute the speedup of the pipelined solution with respect to the unpipelined one (for 120 instructions).

Exercise 3 (5 points) - Number representation

- Represent the natural number range [0; 1199] using the residue number system, considering:
  - * the moduli set S1 consisting of the three power-of-2 based moduli S1=$\{2^n; 2^n+1; 2^{n+1}+1\}$
  - * a moduli set S2 consisting of 4 moduli at your choice.
- Give an estimation of the representational efficiency in both cases.
- Represent A= 39 in **both residue systems** S1 and S2 defined and in the **mixed radix representation** associated.

Exercise 4 (4 points) – Number representation

a) Given the values A= 00 10 01 11 11 and B = 00 01 10 10 11 in the signed RB (Redundant Binary) representation, convert A and B in decimal.

b) Show the execution of operation A-B using the look-up table for addition. Verify the correctness of the result.

Exercise 5 (6 points)

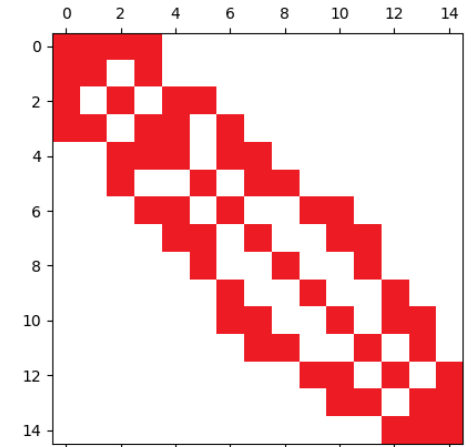a) Consider the sparse matrix whose pattern is shown in the figure on the right.

Specify which arrays you need for the following compressed representations and how many bytes they occupy in memory.
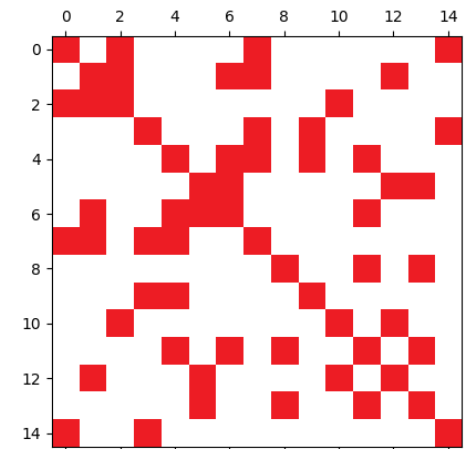
CSR

_____

_____

_____

_____

_____

_____

_____

_____

Ellpack-Itpack

_____

_____

_____

_____

_____

_____

_____

_____

b) Consider the matrix reordering giving the pattern shown in the figure on the right. Explain how arrays change after the reordering and quantify the gain in terms of memory occupation.

CSR

_____
_____
_____
_____
_____
_____
_____
_____

Ellpack-Itpack

_____
_____
_____
_____
_____
_____
_____
_____

c) Considering the reordered matrix, what scheme would be more advantageous among CSR, Ellpack-Itpack, Skyline and Diagonal? Explain why giving the memory occupation for each of them and a comparison.

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

Exercise 6 (4 points) – Instruction pipeline

a) Consider the following loop expressed in a high level language:

for (i =0; i < N; i ++)
{
vectA[i] = vectA[i] + k;
vectC[i] = vectB[i] + vectC[i];
}

The program has been written in MIPS assembly code, assuming that registers $t6 and $t7 have been initialized with values 0 and 4N respectively. The symbols VECTA, VECTB and VECTC is a 16-bit constant.

Let us consider the loop executed by 5-stage pipelined MIPS processor WITHOUT any optimization in the pipeline.

1. Identify the Hazard Type (Data Hazard or Control Hazard) in the last column
2. In the first column identify the number of stalls to be inserted before each instruction (or between stages IF and ID of each instruction) necessary to solve the hazards
3. For each hazard, add an ARROW to indicate the pipeline stages involved in the hazard

Num. Stalls	INSTRUCTION	C1	C2	C3	C4	C5	C7	C6	C8	C9	C10	C11	C12	C13	C14	Hazard Type
	FOR: beq $t6,$t7,END	IF	ID	EX	ME	WB										
	lw $t2,VECTA($t6)		IF	ID	EX	ME	WB									
	lw $t3,VECTB($t6)			IF	ID	EX	ME	WB								
	lw $t4,VECTC($t6)				IF	ID	EX	ME	WB							
	addi $t2,$t2,k					IF	ID	EX	ME	WB						
	sw $t2,VECTA($t6)						IF	ID	EX	ME	WB					
	add $t4,$t3,$t4							IF	ID	EX	ME	WB				
	sw $t4,VECTC($t6)								IF	ID	EX	ME	WB			
	addi $t6,$t6,4									IF	ID	EX	ME	WB		
	j FOR										IF	ID	EX	ME	WB	

b) Specify if there are cases in which the number of stalls actually inserted is lower, taking into account that solving some hazards can help resolve the following ones.

Question (4 points)

Briefly describe the main characteristics of the vector architectures.