HW1: Summation and Prefix-Sums Assignment

Course:	ENEE159V/H
Title:	Summation and Prefix-Sums Assignment
Date Assigned:	February 12, 2009
Date Due:	February 26, 2009

1 Problem statements

Summation Problem
Input: An array $A = A(0),...,A(n-1)$ and a two-dimensional array $B = B(0)(0),...,B(log(n))(n-1)$ of elements (integers).
Task: Compute A(0) + . . . + A(n-1). For examples, see sections 2.1.1 and 2.2 of the class notes, observe the examples dealing with the summation problem.

Prefix-Sums Problem
Input: An array $A = A(0),...,A(n-1)$ and 2 two-dimensional arrays $B = B(0)(0),...,B(log(n))(n-1)$ and $C = C(0)(0),...,C(log(n))(n-1)$ of elements (integers).
Task: For each element in the array, compute the sum of the preceding elements. For examples, see pages 19-21 of the class notes.

2 Assignment

- Write XMTC programs realizing the serial and parallel algorithms for the summation problem.
- Write an XMTC program realizing the parallel algorithm for the prefix-sums problem.

3 The program

You should name your XMTC files as follows: summation.s.c (serial), summation.p.c (parallel), prefixsums.p.c (parallel).

3.1 Setting up the environment

The header files and the binary files can be downloaded from the course website. To get the data files, enter the following commands from your Linux environment:

```
$ wget http://terpconnect.umd.edu/~jspeiser/homework1.tgz
$ tar xzvf homework1.tgz
```

This will create the directory homework1 containing the directories *summation* and *prefixsums*, each with the following folders: *data*, *src*, and *doc*. Data files are available in data directory.

3.2 Input format

You are given: two arrays *A* and *B* that contain *n* integers. Additionally, you are given the integer logn.

Summation Problem	Inputs
int n	The size of the arrays
int A[n]	The array A
int B[log n+1][n]	The 2D array B
int C[log n+1][n]	The 2D array C (**Prefix sums only**)
int logn	Log base 2 of n

3.3 Data sets

Run all your programs (serial and parallel) using the following data files. You can directly include the header file into your XMTC code with *#include* or you can include the header file with the compile option -*include*. To run the compiled program you will need to specify the binary data with –*binload* option. When you are ready to collect clock cycle information, be sure to add the –*cycle* option to the xmtsim command. Note that both parts of the assignment will utilize the same data set files.

Data Set	**Header File**	**Binary file**
n = 16	data/16/summation.h	data/16/summation.xbo
n = 4096	data/4096/summation.h	data/4096/summation.xbo
n = 16384	data/16384/summation.h	data/16384/summation.xbo
n = 65536	data/65536/summation.h	data/65536/summation.xbo

3.4 Output

Prepare and fill the following table for the Summation problem: Create a text file named table.txt in doc. Remove any printf statements from your code while taking these measurements. Printf statements increase the clock and instruction count. Therefore the measurements with printf statements and may not reflect the actual time and work done. After having filled the table, discuss and explain any significant differences between the parallel and serial implementations in terms of clock cycles.

Input size	**n = 16**	**n = 4096**	**n = 16384**	**n = 65536**
Serial Clock Cycles				
Parallel Clock Cycles				

3.5 Command Examples

When developing, generally use:

```
$ xmtcc summation.s.c -include ../data/65536/summation.h ../data/summation.xbo -o myprogram
$ xmtsim myprogram.sim -binload myprogram.b
```

When ready to collect cycle counts use simply add the –cycle option as mentioned above.

3.6 Submission

For this project, all work should be contained within the homework1 directory. Upon completing the project use the following command: *tar czvf homework1.tgz homework1*.