
Manipal Institute of Technology, Manipal

(A Constituent Institute of Manipal University)

II SEMESTER M.TECH END SEMESTER EXAMINATIONS, MAY/JUNE 2016

SUBJECT: MULTI CORE PROGRAM OPTIMIZATION (OPEN ELECTIVE) [CSE 536]

REVISED CREDIT SYSTEM
17/05/2016

Time: 3 Hours MAX. MARKS: 50

Instructions to Candidates:

- Answer **ANY FIVE FULL** questions.
- Missing data, if any, may be suitably assumed.

1A. Which has the lower miss rate: a 16 KB instruction cache with a 16 KB data cache or a 32 KB unified cache? Given the miss per 1000 instructions as follows:
16 KB instruction cache: 3.82
16 KB data cache: 40.9
32 KB unified cache: 43.3
The percentage of instruction references is about 74%
Assume 36% of the instructions are data transfer instructions. Assume a hit takes 1 clock cycles and miss penalty is 100 clock cycles. A load or store hit takes 1 extra clock cycle on a unified cache. What is the average memory access time in each case? Assume write through cache. 5M

1B. Explain the MESI write back invalidation protocol with a neat state transition diagram. Analyze how coherence and sequential consistency are satisfied here. Using this protocol show the state transitions and bus transactions for the following example:
P0 reads x, P2 reads x, P2 writes x, P0 reads x, P1 reads x, P1 writes x. (Show the state of the memory block in all processors at the end of each processor action, the bus transaction generated if any and the entity supplying the data). 5M

2A. Explain Instruction Level Parallelism and Thread Level Parallelism. 3M

2B. Explain Hyper Threading Technology architecture. 5M

2C. If 50% of the time is spent in executing the serial portion of the code, what speedup would be achieved in an eight core processor? 2M

3A. Explain Fence and Barrier mechanisms of synchronization. 4M

3B. Explain thread synchronization primitive available in Pthreads library with an example. 3M

3C. Write a simple program for thread creation in .NET framework. 3M

4A. Is it possible to optimize the following code to avoid performance loss due to mispredicted branches? If yes write the optimized code.

```
void MyFunc(int size, int limit, float dst[ ], float src1[ ], float src2[ ] ){
    int j;
    for ( j=0; j<size; j++){
            if (limit == 255)
```

```
            dest[j] = src1[j];
        else if ( limit == 0 )
            dest[j] = src2[j];
        else
            dest[j] = ( src1[j] * limit + src2[j] * (255 - limit) ) / 256;}}
```
2M

4B. Consider a matrix of size 1024 x 1024. An algorithm needs to be developed such that a function F() is to be executed on all the elements. The function F takes the current value of data element, and its position. After the evaluation of the function, the resulting value is added to the neighboring data elements as shown in Fig Q.4B. Identify the decomposition type and Parallel pattern applicable for the problem. Explain a parallel algorithm such that the application can be run faster. Determine the latency at each parallel task if any. 4M

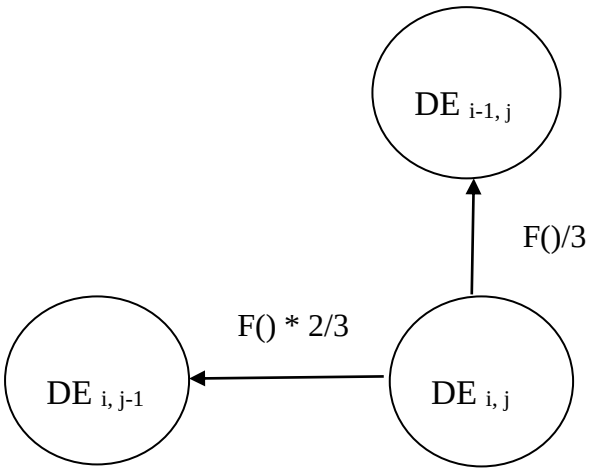

Fig. Q.4B.

4C. Analyze the following function and optimize as much as possible for single core.

```
void RGBtoBW (DWORD *pBitmap, DWORD width, DWORD height, long stride){
    DWORD row, col;
    DWORD pixel, red, green, blue, alpha, bw;
    for (row = 0; row < height; row++){
        for (col = 0; col < width; col++){
            pixel = pBitmap[col + row*stride/4];
            alpha = (pixel >> 24) & 0xff;
            red = (pixel >> 16) & 0xff;
            green = (pixel >> 8) & 0xff;
            blue = pixel & 0xff;
            bw = (DWORD)(red * 0.299 + green * 0.587 + blue * 0.114);
            pBitmap[col + row*stride/4] =
            (alpha<<24) + (bw<<16) + (bw<<8) + (bw);}}}
```
4M

5A. Explain the reduction keyword. Discuss the properties of the operator applicable for reduction. Explain with pseudo-code how reduction can be implemented efficiently for the '|' (Bitwise OR) operator. 3M

5B. Demonstrate how one algorithm design is better than other for computing greatest common factor of two numbers. 4M

5C. Explain Loop distribution and loop interchanging with an example. 3M

6A. Explain all the work sharing constructs of OpenMP with an example for each. 5M

6B. Write a sequential code for multiplying a matrix with a vector. Optimize the code for multicore execution using OpenMP. 5M