```
__global__ void mul( float *Ad, float *Bd, float *Cd, int msize, int tile, int task){

        extern __shared__ float shared[];// first half is for Shared A, second half is for Shared B
        int tx, ty;
        int r, c;
        float Cv;
        float Av, Bv;
        int m;

        for ( tx = 0; tx < task; tx++){
                c = blockIdx.x * tile + threadIdx.x * task + tx;
                for ( ty = 0; ty < task; ty++){
                        r = blockIdx.y * tile + threadIdx.y * task + ty;
                        Cv = (float)0;
                        Av = Ad[ r * msize]; // initialize
                        Bv = Bd[ c ];
                        for ( m = 0; m < msize; m++){
                                shared[ threadIdx.y * task + ty] = Av; // put cur tile to shared mem
                                shared[ tile * msize + threadIdx.x * task + tx] = Bv;
                                __syncthreads();
                                if( (m + 1) < msize){
                                        Av = Ad[ r * msize + m + 1]; //load next tile to reg
                                        Bv = Bd[ m * msize + c];
                                }
                                Cv += shared[ threadIdx.y * task + ty ] * shared[ tile * msize +
                                    threadIdx.x * task + tx];
                                __syncthreads();
                        }
                        Cd[ r * msize + c] = Cv;
                }
        }
}
```