

Computing $y = ax + y$ with a serial loop:

```
void saxpy_serial(int n, float alpha, float *x, float *y)
{
    for(int i = 0; i<n; ++i)
        y[i] = alpha*x[i] + y[i];
}
// Invoke serial SAXPY kernel
saxpy_serial(n, 2.0, x, y);
```

Computing $y = ax + y$ in parallel using CUDA:

```
__global__
void saxpy_parallel(int n, float alpha, float *x, float *y)
{
    int i = blockIdx.x*blockDim.x + threadIdx.x;

    if( i<n ) y[i] = alpha*x[i] + y[i];
}

// Invoke parallel SAXPY kernel (256 threads per block)
int nblocks = (n + 255) / 256;
saxpy_parallel<<<nblocks, 256>>>(n, 2.0, x, y);
```

FIGURE C.3.4 Sequential code (top) in C versus parallel code (bottom) in CUDA for SAXPY (see Chapter 6). CUDA parallel threads replace the C serial loop—each thread computes the same result as one loop iteration. The parallel code computes n results with n threads organized in blocks of 256 threads.