**CS 4400 - Problem Set 7**

**Rob Johansen**

**u0531837**



1. Problem 5.21.

2-way loop unrolling and reassociation:

**void** **psum2way**(**float** a[], **float** p[], **long** **int** n)

{

**long** **int** i;

**int** limit = n - 1;

**float** last\_val, val;

last\_val = p[0] = a[0];

**for** (i = 1; i < limit; i += 2) {

**int** a\_i = a[i];

**int** a\_i\_1 = a[i+1];

p[i] = last\_val + a\_i;

val = last\_val + (a\_i + a\_i\_1);

p[i+1] = val;

last\_val = val;

}

**for** (; i < n; i++) {

val = last\_val + a[i];

p[i] = val;

last\_val = val;

}

}

3-way loop unrolling and reassociation:

**void** **psum3way**(**float** a[], **float** p[], **long** **int** n)

{

**long** **int** i;

**int** limit = n - 2;

**float** last\_val, val;

last\_val = p[0] = a[0];

**for** (i = 1; i < limit; i += 3) {

**int** a\_i = a[i];

**int** a\_i\_1 = a[i+1];

**int** a\_i\_2 = a[i+2];

p[i] = last\_val + a\_i;

p[i+1] = last\_val + (a\_i + a\_i\_1);

val = last\_val + (a\_i + (a\_i\_1 + a\_i\_2));

p[i+2] = val;

last\_val = val;

}

**for** (; i < n; i++) {

val = last\_val + a[i];

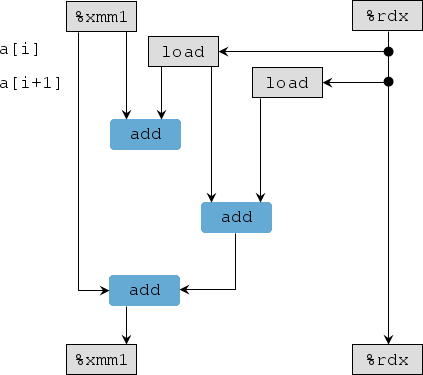
p[i] = val;

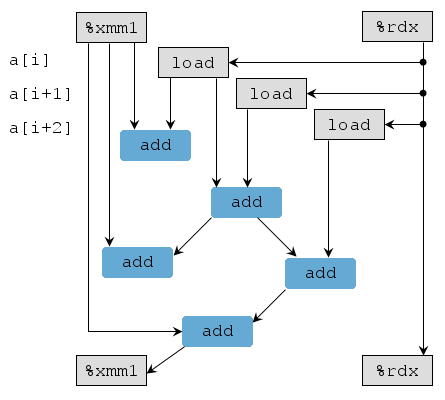
last\_val = val;

}

}

1. Data flow diagrams for my solutions to 5.21:





1. Problem 5.22: Improving part B would provide a speedup of 1.25, while improving part C would only provide a speedup of 1.20. Thus, speeding up part B would be a better choice for maximizing performance. However, my going rate is $10,000 for this kind of speedup. ☺
2. Problem 6.27:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Cache | m | C | B | E | S | t | s | b |
| 1. | 32 | 2048 | 8 | 1 | 256 | 21 | 8 | 3 |
| 2. | 32 | 2048 | 4 | 4 | 128 | 23 | 7 | 2 |
| 3. | 32 | 1024 | 2 | 8 | 64 | 25 | 6 | 1 |
| 4. | 32 | 1024 | 32 | 2 | 16 | 23 | 4 | 5 |