HW3 – U1072596(vikas Kumar rao)

1) scalar processes:-

i) optimized scheduling without unrolling

Loop: L.D F1, 0(R1)

L.D F2, 0(R2)

DADDUI R1, R1, #-8

MUL F3, F1, F2

DADDUI R2, R2, #-8

DADDUI R3, R3, #-8

STALL

BNE R1, R4, Loop

S.D F3, 8(R3)

1 - STALL

4 – loop body instructions

4 – loop counter instructions

ii)optimized scheduling with unrolling the loop twice

Loop: L.D F1, 0(R1)

L.D F2, 0(R2)

L.D F4, -8(R1)

L.D F5, -8(R2)

MUL F3, F1, F2

MUL F6, F4, F5

DADDUI R1, R1, #-16

DADDUI R2, R2, #-16

DADDUI R3, R3, #-16

S.D F3, 16(R3)

BNE R1, R4, Loop

S.D F7, 8(R3)

0-STALLS

8- loop body instructions

4 – loop counter instructions

iii) software pipelined version of the code –

process flow – (value in brackets indicate iteration number/PROCESS CYCLES)

S.D(1) – read as store from 1st process iteration

MUL(2) – read as mult from 2nd process iteration

DADDUIR3(2)

BNE(2)

LDF1(3) – read as load from 3rd process iteration

LDF2(3)

DADDUIR1(3)

DADDUIR2(3)

Pipelined code version –

S.D. F3,-16(R3)

F.MUL F3,F2,F1

DADDUI R3,R3,#-24

BNE R1,R4,Loop

Loop: L.D. F1,0(R1)

L.D. F2,0(R2)

DADDUI R1,R1,#-8

DADDUI R2,R2,#-8

2) software optimization:-

Optimized scheduling without unrolling (assuming same setup as q1 with no stalls after mult for any other instruction other than branch)

Loop: L.D F2,0(R2)

L.D F3, 0(R3)

L.D F4,0(R4)

MULT.D F1,F2,F3

ADD.D F5,F4,F1

DADDUI R2,R2,#-8

DADDUI R3,R3,#-8

DADDUI R4,R4,#-8

DADDUI R5,R5,#-8

BNE R2,R1,Loop

S.D F5,8(R5)

0-STALLS

5 – LOOP COUNTER INSTRUCTIONS

6- LOOP BODY INSTRUCTIONS

3)software optimization:-

i)optimized scheduling without enrolling

|  |  |
| --- | --- |
| INTEGER PIPELINE OPERATIONS | FLOATING POINT PROCESSOR OPERATIONS |
| Loop:L.D. F1,0(R1) | STALL |
| DADDUI R1,R1,#-8 | STALL |
| DADDUI R2,R2,#-8 | F.MUL F3,F1,F2 |
| STALL | STALL |
| STALL | F.ADD F5,F3,F4 |
| STALL | STALL |
| STALL | STALL |
| STALL | STALL |
| BNE R1,R3,Loop | STALL |
| S.D F5,8(R2) | STALL |

4- STALL CYCLES (IN BOTH PROCESSES)

4 LOOP BODY INSTRUCTIONS

3 LOOP COUNTER INSTRUCTIONS

ii)optimized with unrolling

code: *for(i=1000;i>0;i--)*

*{*

*x[i]=s\*y[i]+z;*

*x[i-1]=s\*y[i-1]+z;*

*x[i-2]=s\*y[i-2]+z;*

*}*

to avoid stall cycles, the loop needs to be unrolled thrice, the schedule is given as –

|  |  |
| --- | --- |
| INTEGER PIPELINE OPERATIONS | FLOATING POINT PROCESSOR OPERATIONS |
| Loop:L.D. F1,0(R1) | STALL |
| L.D. F6,-8(R1) | STALL |
| L.D. F7,-16(R1) | F.MUL F3,F1,F2 |
| STALL | F.MUL F8,F6,F2 |
| STALL | F.ADD F5,F3,F4 |
| STALL | F.MUL F9,F7,F2 |
| STALL | F.ADD F10,F8,F4 |
| STALL | F.ADD F11,F9,F4 |
| DADDUI R1,R1,#-24 | STALL |
| S.D F5,0(R2) | STALL |
| DADDUI R2,R2,#-24 | STALL |
| S.D F10,16(R2) | STALL |
| BNE R1,R3,Loop | STALL |
| S.D F11,8(R2) | STALL |

Constants representing s and x are assumed to be in the same registers F2 and F4 respectively.

0 stall cycles

3 loop counter instructions

12 loop body instructions