Code Example: SAXPY

- **SAXPY** (Single-precision A X Plus Y)
  - Linear algebra routine (used in solving systems of equations)
  - Part of early “Livermore Loops” benchmark suite

```
for (i=0;i<N;i++)
  Z[i]=A*X[i]+Y[i];

0: ldf X(r1),f1       // loop
1: mulf f0,f1,f2      // A in f0
2: ldf Y(r1),f3       // X,Y,Z are constant addresses
3: addf f2,f3,f4
4: stf f4,Z(r1)
5: addi r1,4,r1       // i in r1
6: blt r1,r2,0        // N*4 in r2
```

(destination register on right)

SAXPY Performance and Utilization

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
`ldf X(r1),f1`	F	D	X	M	W															
`mulf f0,f1,f2`	F	D	D*	D*	E*	E*	E*	E*	E*	W										
`ldf Y(r1),f3`		F	D	D*	X	M	W													
`addf f2,f3,f4`		F	F*	F*	D	D*	D*	D*	D*	E+	E+	W								
`stf f4,Z(r1)`			F	F*	D	D*	D*	D*	D*	D*	X	M	W							
`addi r1,4,r1`					F	F*	F*	F*	F*	D*	D	X	M	W						
`blt r1,r2,0`					F	F*	F*	F*	F*	F*	D	D*	X	M	W					
`ldf X(r1),f1`											F	D	X	M	W					

- Dual issue pipeline (fluid)
  - Same + any two insns per cycle + embedded taken branches
  - \+ **Performance**: 7 insns / 10 cycles = 0.70 IPC
  - – **Utilization**: 0.70 actual IPC / 2 peak IPC = 35%
  - – More hazards → more stalls (why?)
  - – Each stall is more expensive (why?)

SAXPY Performance and Utilization Cycle 1

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
`ldf X(r1),f1`	F																			
`mulf f0,f1,f2`	F																			
`ldf Y(r1),f3`																				
`addf f2,f3,f4`																				
`stf f4,Z(r1)`																				
`addi r1,4,r1`																				
`blt r1,r2,0`																				
`ldf X(r1),f1`																				

- Fetch first 2 instructions (“first group”)

SAXPY Performance and Utilization Cycle 2

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
`ldf X(r1),f1`	F	D																		
`mulf f0,f1,f2`	F	D																		
`ldf Y(r1),f3`		F																		
`addf f2,f3,f4`		F																		
`stf f4,Z(r1)`																				
`addi r1,4,r1`																				
`blt r1,r2,0`																				
`ldf X(r1),f1`																				

- First two instructions advanced to D
- Fetch next two instructions (“second group”)

SAXPY Performance and Utilization Cycle 3

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
`ldf X(r1),f1`	F	D	X																	
`mulf f0,f1,f2`	F	D	D*																	
`ldf Y(r1),f3`		F	D																	
`addf f2,f3,f4`		F	F*																	
`stf f4,Z(r1)`			F																	
`addi r1,4,r1`																				
`blt r1,r2,0`																				
`ldf X(r1),f1`																				

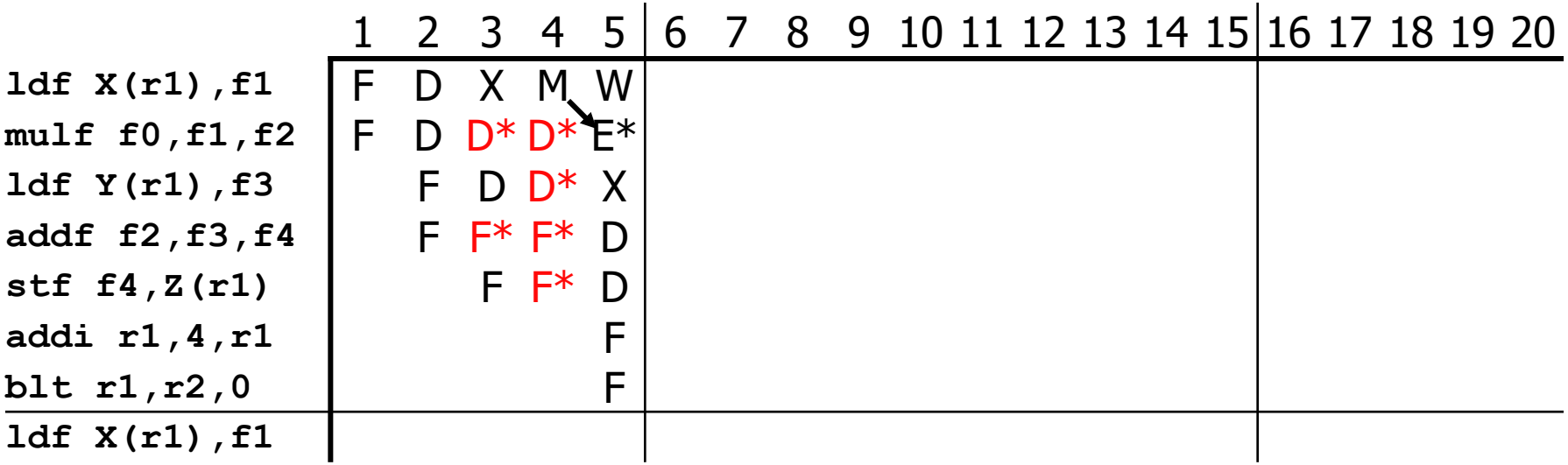
- ldf X(r1), f1 advances to X
- mulf f0, f1, f2 **does not** advance to X
  - Load-to-use stall on ldf X(r1), f1 → stay in D
- ldf Y(r1), f3 advances to D
  - **Fluid** pipeline design, so older instruction in next pair can advance
  - Reform groups (new groups: ldf Y(r1), f3 + mulf ; addf + stf)
- addf: pipeline hazard (no space in D), so stays in F
- Fetch stf (now space in F with our new groups)

SAXPY Performance and Utilization Cycle 4

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
`ldf X(r1),f1`	F	D	X	M																
`mulf f0,f1,f2`	F	D	D*	D*																
`ldf Y(r1),f3`		F	D	D*																
`addf f2,f3,f4`		F	F*	F*																
`stf f4,Z(r1)`			F	F*																
`addi r1,4,r1`																				
`blt r1,r2,0`																				
`ldf X(r1),f1`																				

- Group 1: ldf X(r1) advances to M
- Group 2:
  - mulf still has load-to-use stall on ldf X(r1) → stay in D
  - ldf Y(r1) is younger insn in pair, so it is also stalled in D
- Group 3:
  - addf & stf stuck stalling in F because prior pair stalled in D (structural hazard)

SAXPY Performance and Utilization Cycle 5



- Group 1: ldf X(r1) advances to W (now can forward f1)
- Group 2
  - Forward f1 to mulf, it advances to E*
  - Ldf Y(r1) advances to X
- Group 3: addf & stf finally can advance to D
- Group 4: fetch addi and blt

SAXPY Performance and Utilization Cycle 6

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
`ldf X(r1),f1`	F	D	X	M	W															
`mulf f0,f1,f2`	F	D	d*	d*	E*	E*														
`ldf Y(r1),f3`		F	D	p*	X	M														
`addf f2,f3,f4`		F	p*	p*	D	d*														
`stf f4,Z(r1)`			F	p*	D	p*														
`addi r1,4,r1`					F	p*														
`blt r1,r2,0`					F	p*														
`ldf X(r1),f1`																				

- Group 2
  - mulf still in E*
  - ldf Y(r1) advances to M (**potentially bad for exceptions?**)
- Group 3:
  - addf must stall in D until mulf can forward f2 (RAW hazard)
  - stf is younger insn in pair, must also stall in D
- Group 4:
  - addi and blt stall in F because prior pair stalled in D (structural hazard)

SAXPY Performance and Utilization Cycle 7

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
`ldf X(r1),f1`	F	D	X	M	W															
`mulf f0,f1,f2`	F	D	D*	D*	E*	E*	E*													
`ldf Y(r1),f3`		F	D	D*	X	M	W													
`addf f2,f3,f4`		F	F*	F*	D	D*	D*													
`stf f4,Z(r1)`			F	F*	D	D*	D*													
`addi r1,4,r1`					F	F*	F*													
`blt r1,r2,0`					F	F*	F*													
`ldf X(r1),f1`																				

- Group 2
  - mulf still in E*
  - ldf Y(r1) to W (could forward f3 to addf, but addf waiting on f2)
- Group 3:
  - addf must stall in D until mulf can forward f2 (RAW hazard)
  - stf is younger insn in pair, must also stall in D
- Group 4:
  - addi and blt stall in F because prior pair stalled in D (structural hazard)

SAXPY Performance and Utilization Cycle 8

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
`ldf X(r1),f1`	F	D	X	M	W															
`mulf f0,f1,f2`	F	D	D*	D*	E*	E*	E*	E*												
`ldf Y(r1),f3`		F	D	D*	X	M	W													
`addf f2,f3,f4`		F	F*	F*	D	D*	D*	D*												
`stf f4,Z(r1)`			F	F*	D	D*	D*	D*												
`addi r1,4,r1`					F	F*	F*	F*												
`blt r1,r2,0`					F	F*	F*	F*												
`ldf X(r1),f1`																				

- Group 2
  - mulf still in E*
- Group 3:
  - addf must stall in D until mulf can forward f2 (RAW hazard)
  - stf is younger insn in pair, must also stall in D
- Group 4:
  - addi and blt stall in F because prior pair stalled in D (structural hazard)

SAXPY Performance and Utilization Cycle 9

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
`ldf X(r1),f1`	F	D	X	M	W															
`mulf f0,f1,f2`	F	D	D*	D*	E*	E*	E*	E*	E*											
`ldf Y(r1),f3`		F	D	D*	X	M	W													
`addf f2,f3,f4`		F	F*	F*	D	D*	D*	D*	D*											
`stf f4,Z(r1)`			F	F*	D	D*	D*	D*	D*											
`addi r1,4,r1`					F	F*	F*	F*	F*											
`blt r1,r2,0`					F	F*	F*	F*	F*											
`ldf X(r1),f1`																				

- Group 2
  - mulf still in E*
- Group 3:
  - addf must stall in D until mulf can forward f2 (RAW hazard)
  - stf is younger insn in pair, must also stall in D
- Group 4:
  - addi and blt stall in F because prior pair stalled in D (structural hazard)

SAXPY Performance and Utilization Cycle 10

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
`ldf X(r1),f1`	F	D	X	M	W															
`mulf f0,f1,f2`	F	D	D*	D*	E*	E*	E*	E*	E*	W										
`ldf Y(r1),f3`		F	D	D*	X	M	W													
`addf f2,f3,f4`		F	F*	F*	D	D*	D*	D*	D*	E+										
`stf f4,Z(r1)`			F	F*	D	D*	D*	D*	D*	D*										
`addi r1,4,r1`					F	F*	F*	F*	F*	D*										
`blt r1,r2,0`					F	F*	F*	F*	F*	F*										
`ldf X(r1),f1`																				

- Group 2
  - mulf advances to WB, forwards f2 to addf
- "Group 3":
  - addf enters E+
  - stf has RAW hazard on addf f4, stalls in D
- Group 4:
  - Advance addi to D and form new pair with stf
  - blt stall in F because already 2 instructions in D (structural hazard)

SAXPY Performance and Utilization Cycle 11

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
`ldf X(r1),f1`	F	D	X	M	W															
`mulf f0,f1,f2`	F	D	D*	D*	E*	E*	E*	E*	E*	W										
`ldf Y(r1),f3`		F	D	D*	X	M	W													
`addf f2,f3,f4`		F	F*	F*	D	D*	D*	D*	D*	E+	E+									
`stf f4,Z(r1)`			F	F*	D	D*	D*	D*	D*	D*	X									
`addi r1,4,r1`					F	F*	F*	F*	F*	D*	D									
`blt r1,r2,0`					F	F*	F*	F*	F*	F*	D									
`ldf X(r1),f1`											F									

- "Group" 3:
  - addf still in E+
- "Group" 4
  - stf can get f4 forwarded in next cycle (E+ → M), so stf goes to X (don't need f4 in X since it's the data we're storing) – reforms Group 3 with addf
  - addi stalls in D -- only 2 instrs per stage at a time (structural hazard)
- Group 5
  - blt can now advance to D because stf in X – reforms Group 4 with addi
  - Fetch ldf X(r1) from next loop

SAXPY Performance and Utilization Cycle 12

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
`ldf X(r1),f1`	F	D	X	M	W															
`mulf f0,f1,f2`	F	D	D*	D*	E*	E*	E*	E*	E*	W										
`ldf Y(r1),f3`		F	D	D*	X	M	W													
`addf f2,f3,f4`		F	F*	F*	D	D*	D*	D*	D*	E+	E+	W								
`stf f4,Z(r1)`			F	F*	D	D*	D*	D*	D*	D*	X	M								
`addi r1,4,r1`					F	F*	F*	F*	F*	D*	D	X								
`blt r1,r2,0`					F	F*	F*	F*	F*	F*	D	D*								
`ldf X(r1),f1`											F	D								

- "Group" 3:
  - addf advances to WB, forwards f4 to stf
  - stf advances to M
- Group 4:
  - addi advances to X
  - blt has RAW hazard on r1, stalls in D (assume branch resolution in X)
- Group 5
  - ldf X(r1) advances to D, forms new pair with blt

SAXPY Performance and Utilization Cycle 13

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
`ldf X(r1),f1`	F	D	X	M	W															
`mulf f0,f1,f2`	F	D	D*	D*	E*	E*	E*	E*	E*	W										
`ldf Y(r1),f3`		F	D	D*	X	M	W													
`addf f2,f3,f4`		F	F*	F*	D	D*	D*	D*	D*	E+	E+	W								
`stf f4,Z(r1)`			F	F*	D	D*	D*	D*	D*	D*	X	M	W							
`addi r1,4,r1`					F	F*	F*	F*	F*	D*	D	X	M							
`blt r1,r2,0`					F	F*	F*	F*	F*	F*	D	D*	X							
`ldf X(r1),f1`											F	D	X							

- "Group" 3:
  - stf advances to WB
- Group 4:
  - addi advances to M, forwards r1 to blt
- Group 5
  - blt advances to X (branch resolved, but taken as predicted)
  - ldf X(r1) advances to X

SAXPY Performance and Utilization Cycle 14

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
`ldf X(r1),f1`	F	D	X	M	W															
`mulf f0,f1,f2`	F	D	D*	D*	E*	E*	E*	E*	E*	W										
`ldf Y(r1),f3`		F	D	D*	X	M	W													
`addf f2,f3,f4`		F	F*	F*	D	D*	D*	D*	D*	E+	E+	W								
`stf f4,Z(r1)`			F	F*	D	D*	D*	D*	D*	D*	X	M	W							
`addi r1,4,r1`					F	F*	F*	F*	F*	D*	D	X	M	W						
`blt r1,r2,0`					F	F*	F*	F*	F*	F*	D	D*	X	M						
`ldf X(r1),f1`											F	D	X	M						

- Group 4:
  - addi advances to WB
- Group 5
  - blt advances to M
  - ldf X(r1) advances to M

SAXPY Performance and Utilization Cycle 15

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
`ldf X(r1),f1`	F	D	X	M	W															
`mulf f0,f1,f2`	F	D	D*	D*	E*	E*	E*	E*	E*	W										
`ldf Y(r1),f3`		F	D	D*	X	M	W													
`addf f2,f3,f4`		F	F*	F*	D	D*	D*	D*	D*	E+	E+	W								
`stf f4,Z(r1)`			F	F*	D	D*	D*	D*	D*	D*	X	M	W							
`addi r1,4,r1`					F	F*	F*	F*	F*	D*	D	X	M	W						
`blt r1,r2,0`					F	F*	F*	F*	F*	F*	D	D*	X	M	W					
`ldf X(r1),f1`											F	D	X	M	W					

- Group 5
  - blt advances to WB
  - ldf X(r1) advances to WB