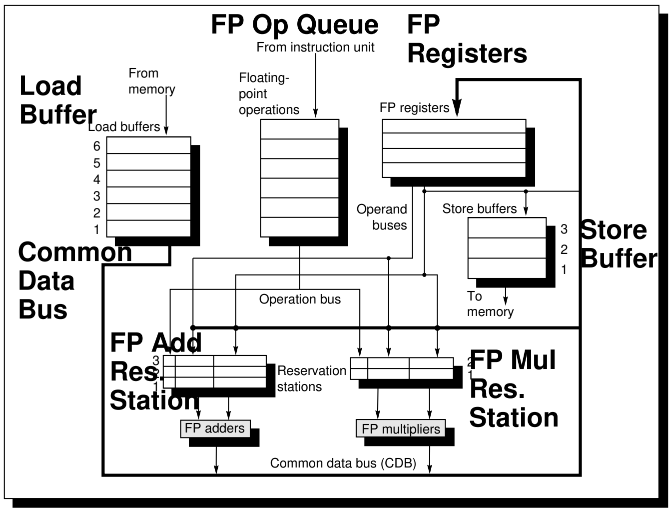
Review: Scoreboard Summary

- Speedup 1.7 from compiler; 2.5 by hand BUT slow memory (no cache)
- Limitations of 6600 scoreboard
  - No forwarding
  - Limited to instructions in basic block (small *window*)
  - Number of functional units(structural hazards)
  - wait for WAR hazards
  - prevent WAW hazards

Another Dynamic Algorithm: Tomasulo Algorithm

- For IBM 360/91 about 3 years after CDC 6600
- Goal was to have High Performance without special compilers
- Differences between IBM 360 & CDC 6600 ISA
  - IBM has only 2 register specifiers/instr vs. 3 in CDC 6600
  - IBM has 4 FP registers vs. 8 in CDC 6600
- Differences between Tomasulo Algorithm & Scoreboard
  - Control & buffers distributed with Function Units vs. centralized in scoreboard; called "reservation stations"
  - Registers in instructions replaced by pointers to reservation station buffer
  - HW renaming of registers to avoid WAR, WAW hazards
  - Common Data Bus broadcasts results to all FUs
  - Load and Stores treated as FUs as well

Tomasulo Organization



Reservation Station Components

Op—Operation to perform in the unit (e.g., + or –)
Qj, Qk—Reservation stations producing source registers
Vj, Vk—Value of Source operands
Rj, Rk—Flags indicating when Fj, Fk are ready
Busy—Indicates reservation station and FU is busy

Register result status—Indicates which functional unit will write each register, if one exists. Blank when no pending instructions that will write that register.

3 Stages of Tomasolu Algorithm

1. **Issue**—get instruction from FP Op Queue

    If reservation station free, the scoreboard issues instr & sends operands (renames registers).

2. **Execution**—operate on operands (EX)

    When both operands ready then execute; if not ready, watch CDB for result

3. **Write result**—finish execution (WB)

    Write on Common Data Bus to all awaiting units

Tomasulo Loop Example

Loop:	LD	F0	0	R1
	MULTD	F4	F0	F2
	SD	F4	0	R1
	SUBI	R1	R1	#8
	BNEZ	R1	Loop	

- Multiply takes 4 clocks
- Load have cache misses

Tomasulo Summary

- Prevents Register as bottleneck
- Avoids WAR, WAW hazards of Scoreboard
- Allows loop unrolling in HW
- Not limited to basic blocks (provided branch prediction)
- Lasting Contributions
  - Dynamic scheduling
  - Register renaming
  - Load/store disambiguation
- Next stop: More branch prediction