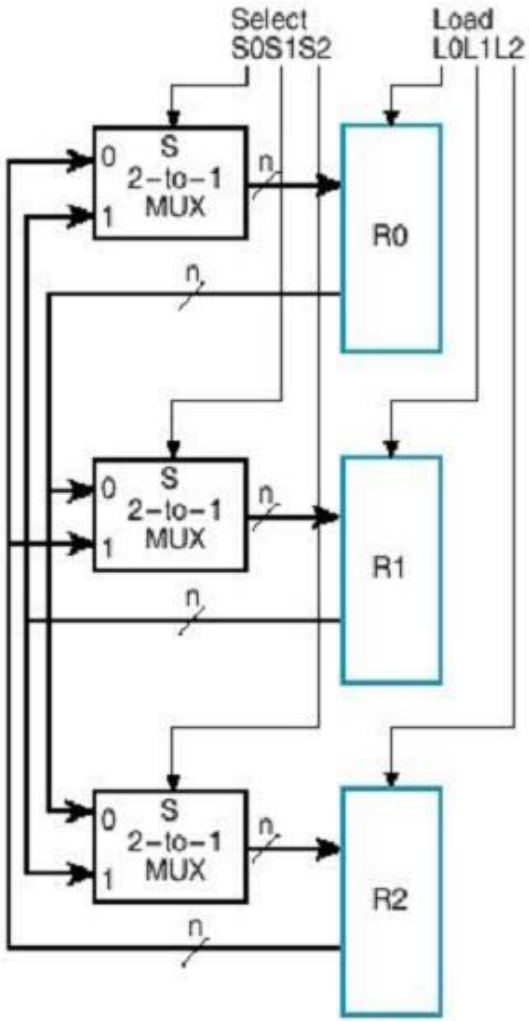
CS2022 Bus-based Transfers

- Digital systems typically have a considerable number of registers N.
  - Typically $8 \leq N \leq 256$
  - Programmer need to be able to make transfers between any pair of them.
  - Let us consider the N=3 and use 2:1 MUXs to interconnect R0, R1, R2.
    - The result is on the next slide.

CS2022 Register Transfer via MUX Diagram



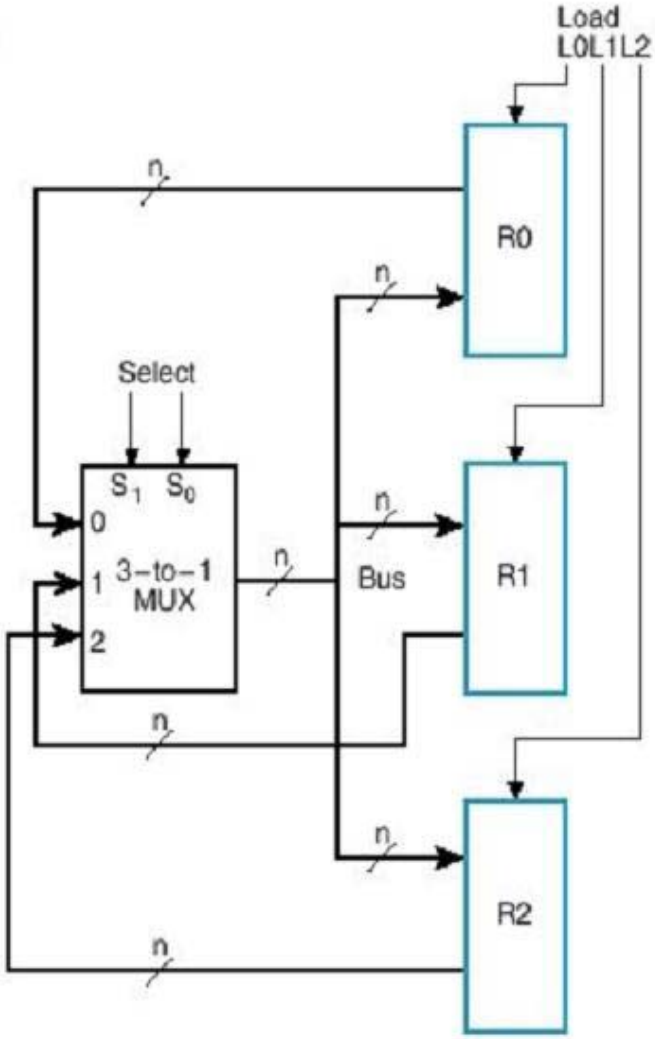
CS2022 Register Transfer via MUXs

- This is a very flexible system for it can make up to three independent transfers in one clock period.

RT	S2 S1 S0	L2 L1 L0	Description
R2 ← R1	1 X X	1 0 0	Point-to-Point
R2 ← R1, R1 ← R2	1 1 X	1 1 0	Reg. Exchange
R2 ← R1, R1 ← R0 R0 ← R2	1 0 0	1 1 1	Reg. Rotate
R2 ← R0, , R1 ← R0	0 0 X	1 1 0	Reg. broadcast

- But this is very costly in terms of interconnect, requiring 6*n MUX input connections.
- To connect N*n-bit registers will require (N-1)*N*n wires.

CS2022 Register Transfer via MUX & Bus



- To reduce the amount of interconnects we can use 3-1 MUXs with a single MUX-to-Register bus connection.

CS2022 Register Transfer via 3-1 MUXs and MUX–to-Register Bus

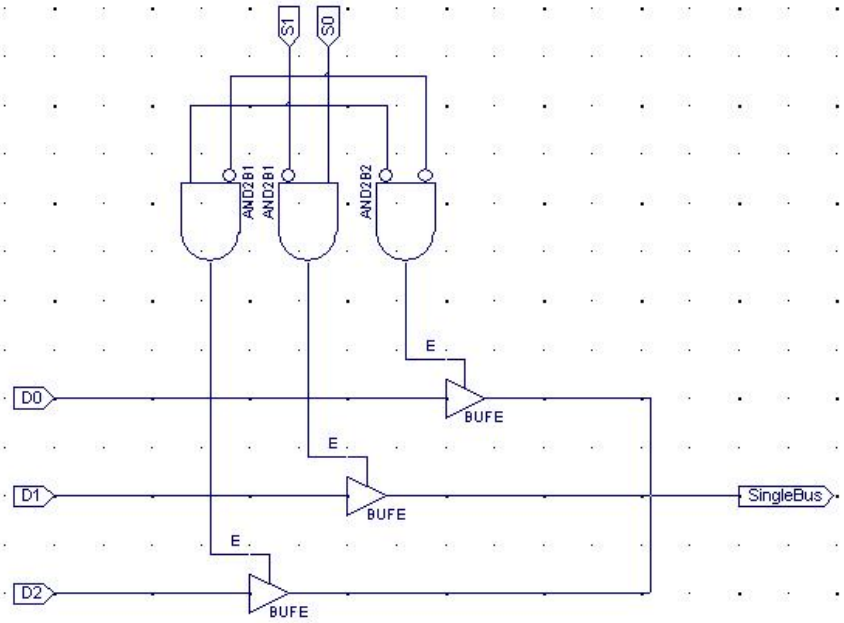
- This results in the loss of some flexibility:

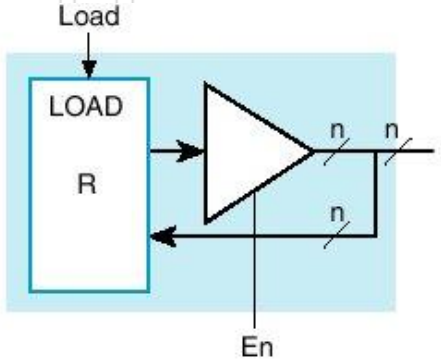
RT	S1 S0	L2 L1 L0	Description
R0 ← R2	1 0	0 0 1	Point-to-Point
R0 ← R1, R2 ← R1	0 1	1 0 1	Reg. Broadcast
R0 ← R1, R1 ← R0	IMPOSSIBLE		Single source only

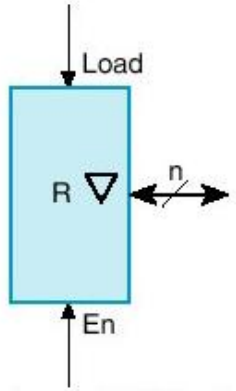
- But the MUX input connections have reduced from 6*n to 3*n.
- For N Register we need only N*n wires.

CS2022 Tri-state Bus

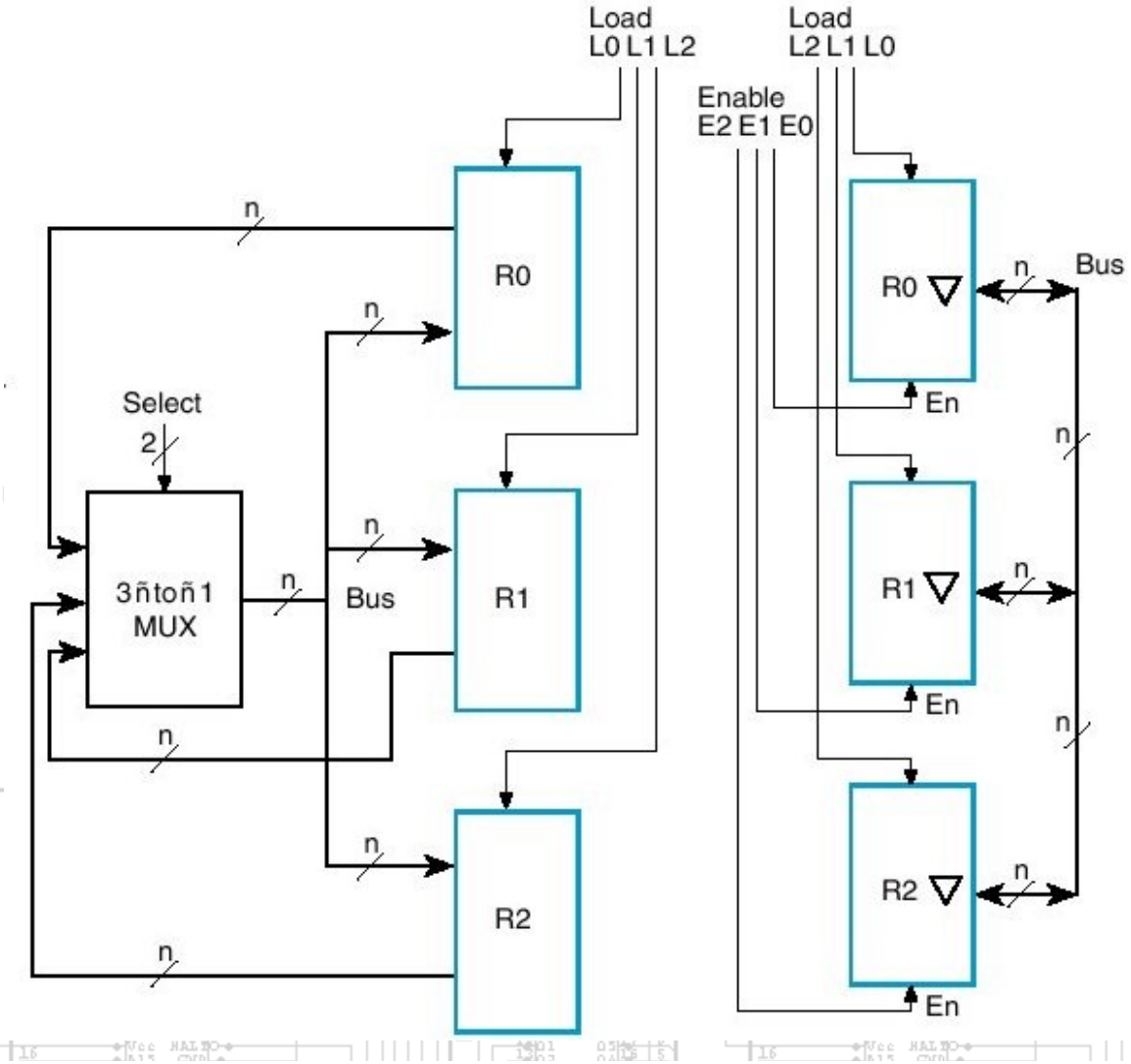
- Tri-state buffers provide the means to construct a wired-or of arbitrary fan-in, with which we can effectively disperse the MUX on the previous slide right back to the register latches. That is we build our 3:1 MUX as:







CS2022 Tri-state-bus vs. MUX-bus



- This results in a solution seen on the right (tri-state), which has the same functionality as the MUX based solution on the left, using just a single bi-directional bus.

CS2022 Bi-directional Tri-state Bus

RT	E2 E1 E0	L2 L1 L0	Description
R0 ← R2	1 0 0	0 0 1	Point-to-Point
R0 ← R1, R2 ← R1	0 1 0	1 0 1	Reg. Broadcast
R0 ← R1, R1 ← R0	IMPOSSIBLE		Single source only

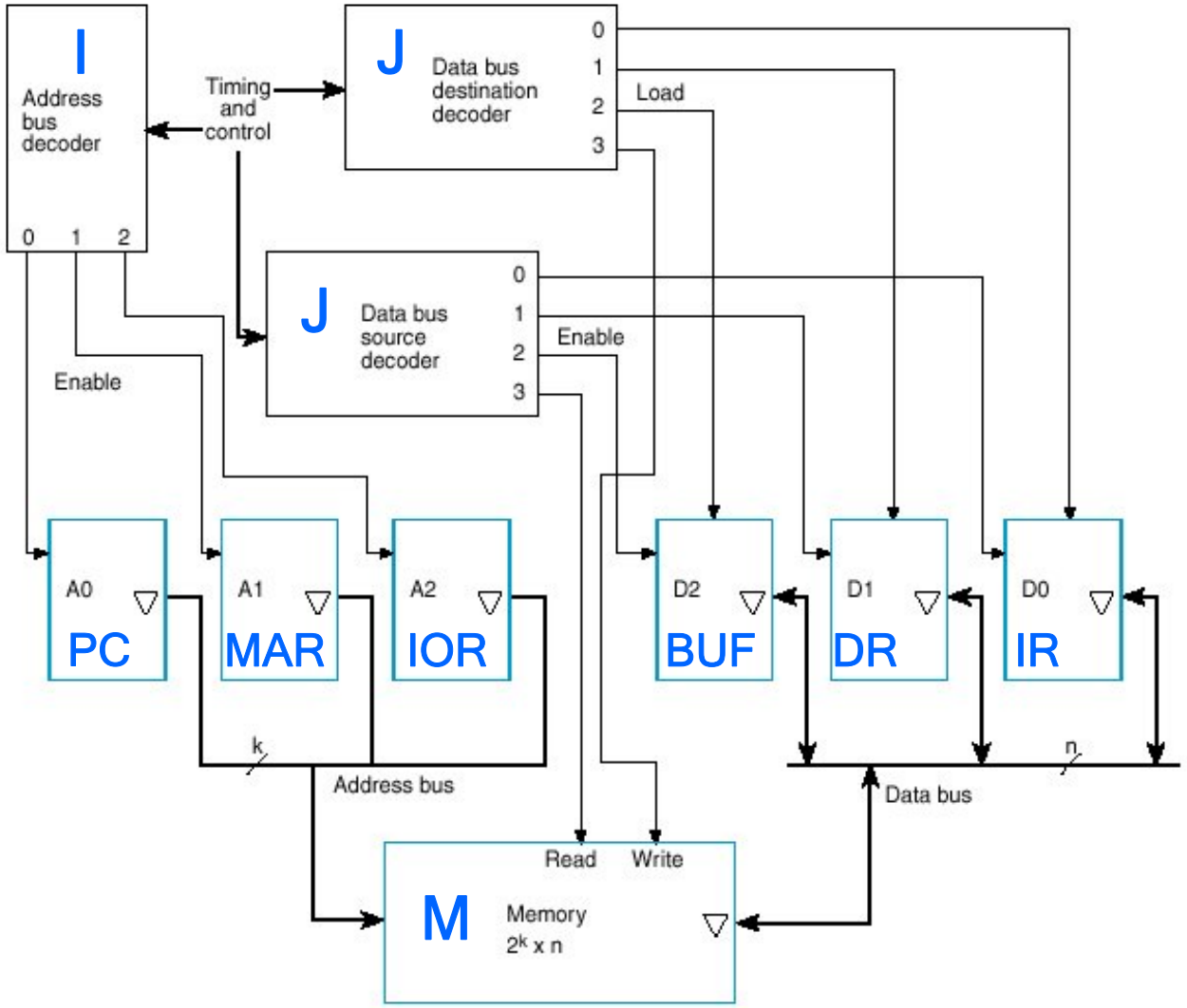
- With this arrangement it is possible to connect N*n-bit register with N-1 paths of width n.

CS2022 Memory Transfers

- Typically processor memories are addressed by a number of address registers:
  - **PC**, **MAR**, **IOR**
- Address register information is transmitted over an address bus to memory **M**.
- Similarly data is transferred to/from a number of data registers over a data bus.
  - **Data register: IR**, **DR**, **BUF**
- The schematic on the next slide illustrates how we may organise three address register **A**, selected by I, and data register D, selected by **J**, to implement read/write on memory **M**.

CS2022

READ: D[J] ← M[A[I]]
WRITE: M[A[I]] ← D[J]

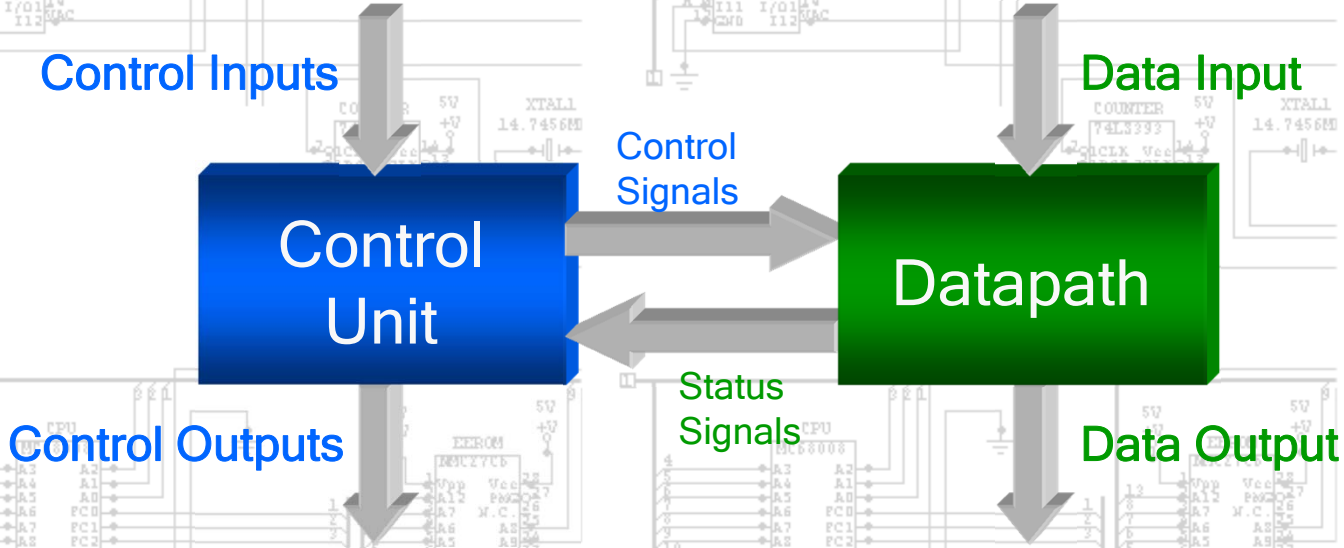


CS2022 Datapath

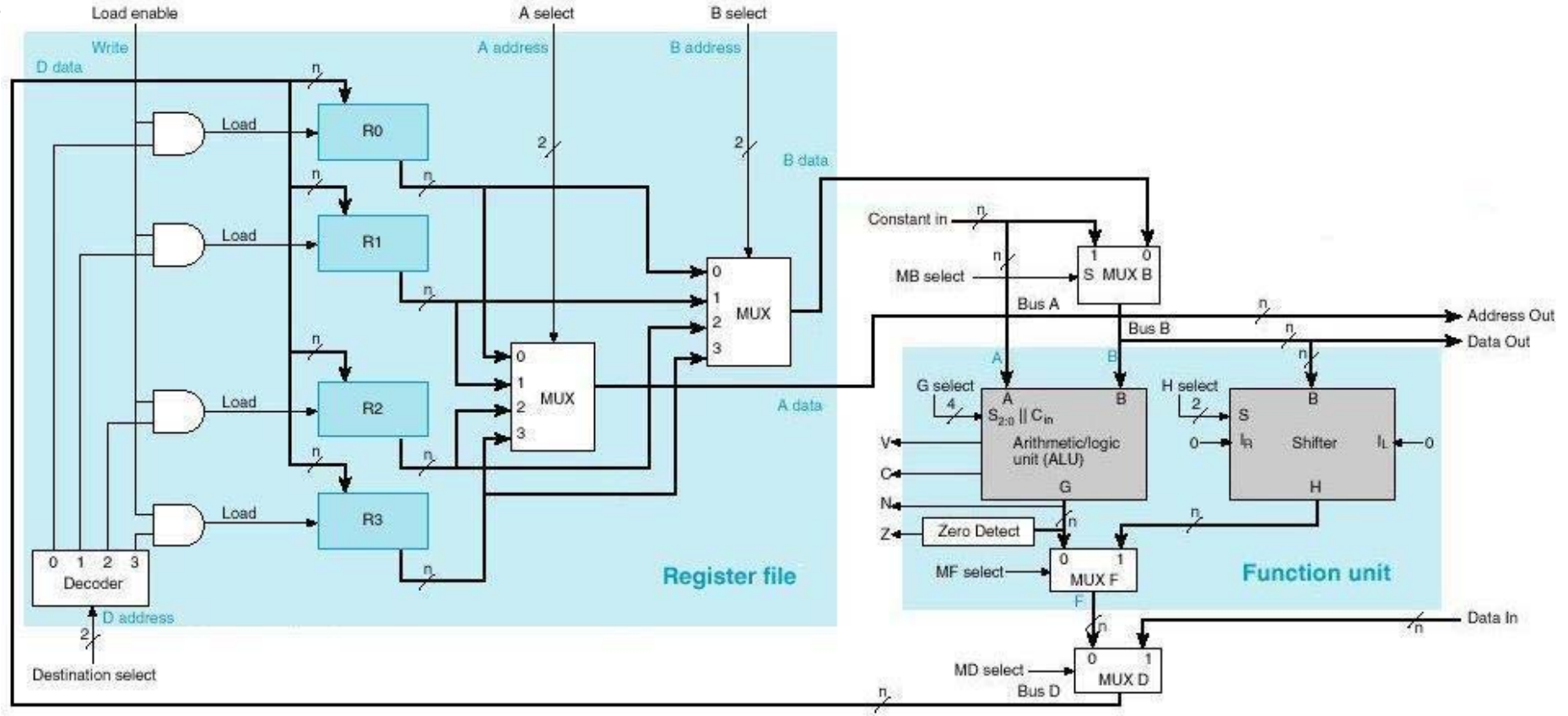
- In all digital systems we can partition the structure into two sections:

a) The data path which performs data-processing operations on the data stream.

b) The control unit which determines the schedule for these data processing operations.



CS2022 Datapath Schematic



CS2022 Register File & Functional Unit

- In practice the three functional micro-ops are implemented in one compact circuit, the Function Unit, composed of the ALU and Shifter.
- Data for this Functional Unit comes from a physical adjacent Register file, with dual MUX-busses able to supply two operands per clock cycle.
- Since the register file is itself modest in size typically 8-32 register, there must be provision to send and receive data to/from the main memory system via DATA IN and DATA OUT.