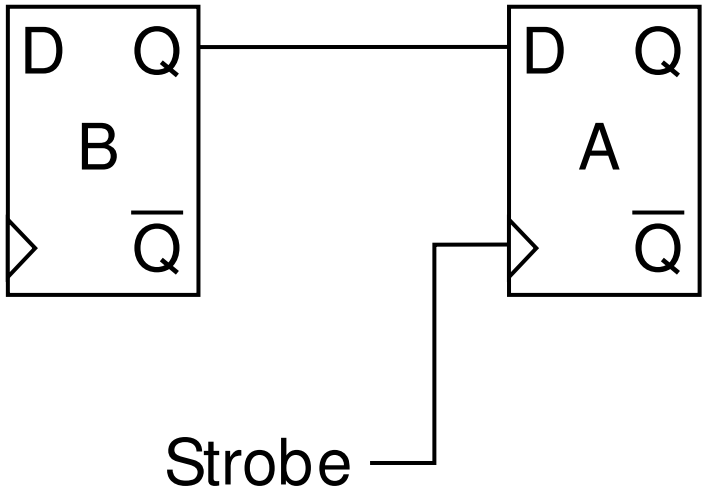
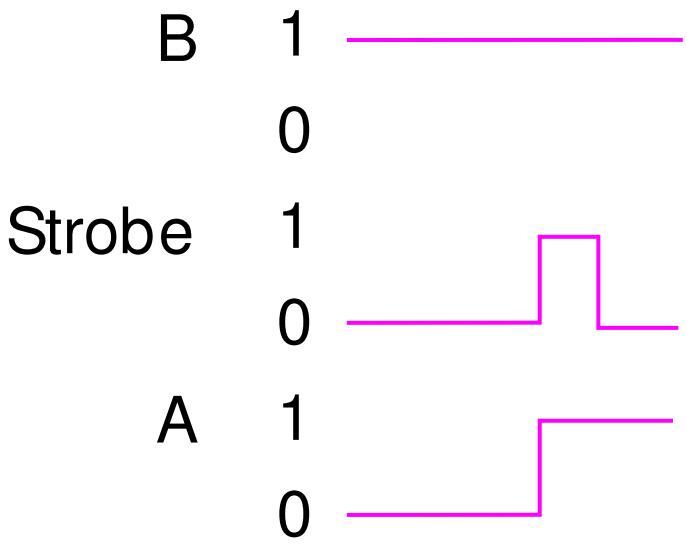
Fig 2.11 Register Transfers Hardware and Timing for a Single-Bit Register Transfer: A ← B

- **Implementing the RTN statement A ← B**

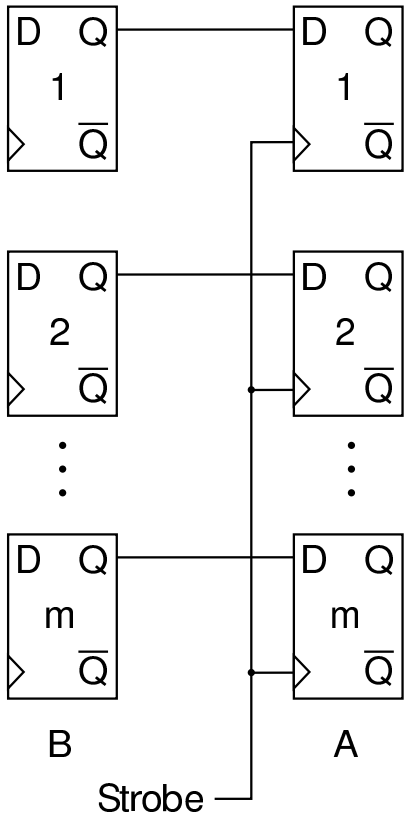


(a) Hardware

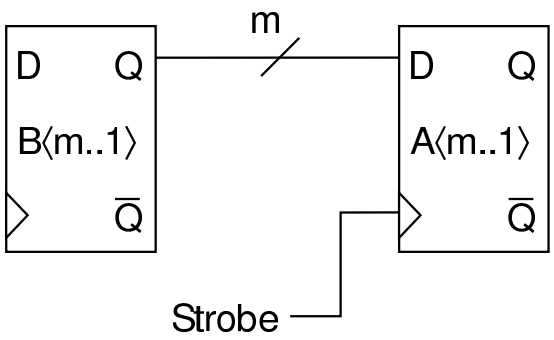


(b) Timing

Fig 2.12 Multiple Bit Register Transfer: A⟨m..1⟩ ← B⟨m..1⟩



(a) Individual flip-flops



(b) Abbreviated notation

Fig 2.13 Data Transmission View of Logic Gates

- **Logic gates can be used to control the transmission of data:**

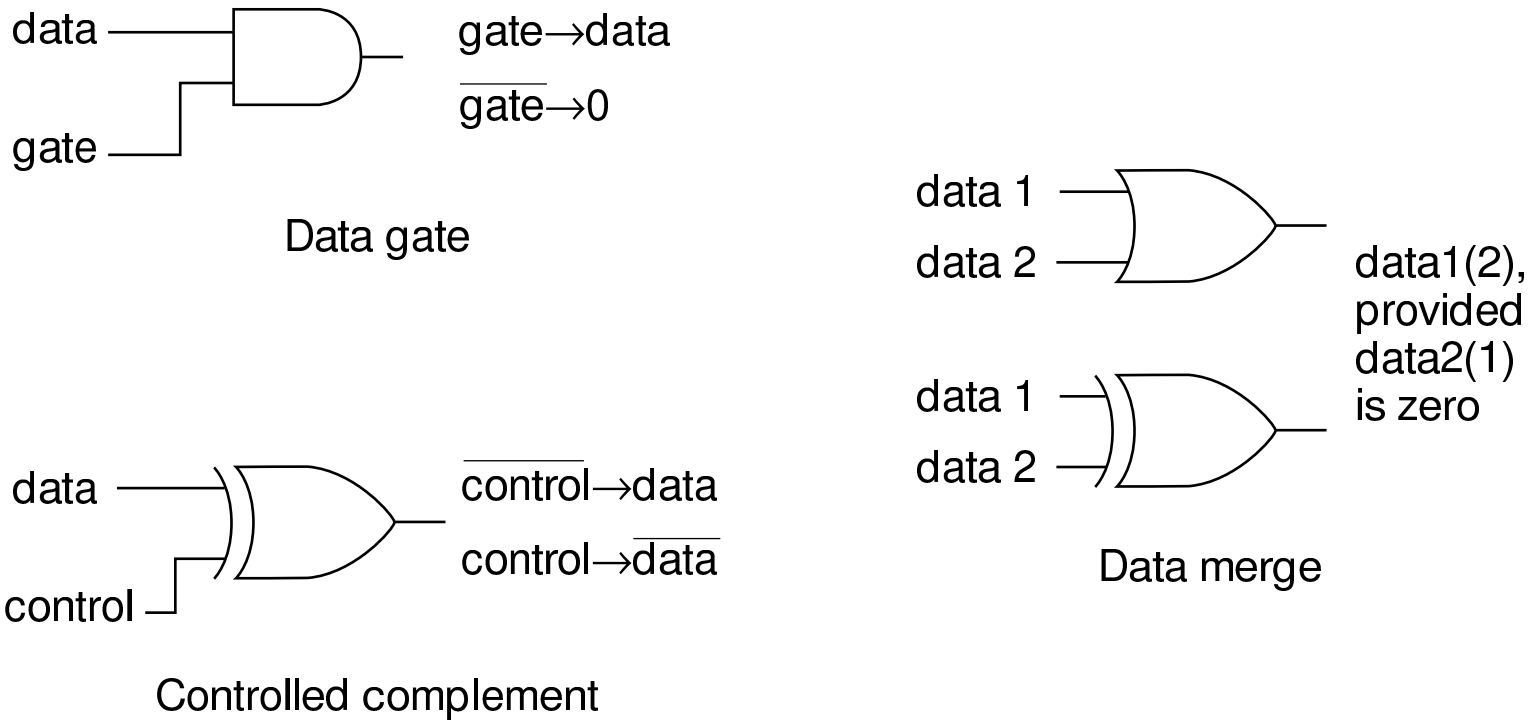


Fig 2.14 Two-Way Gated Merge, or Multiplexer

- **Data from multiple sources can be selected for transmission**

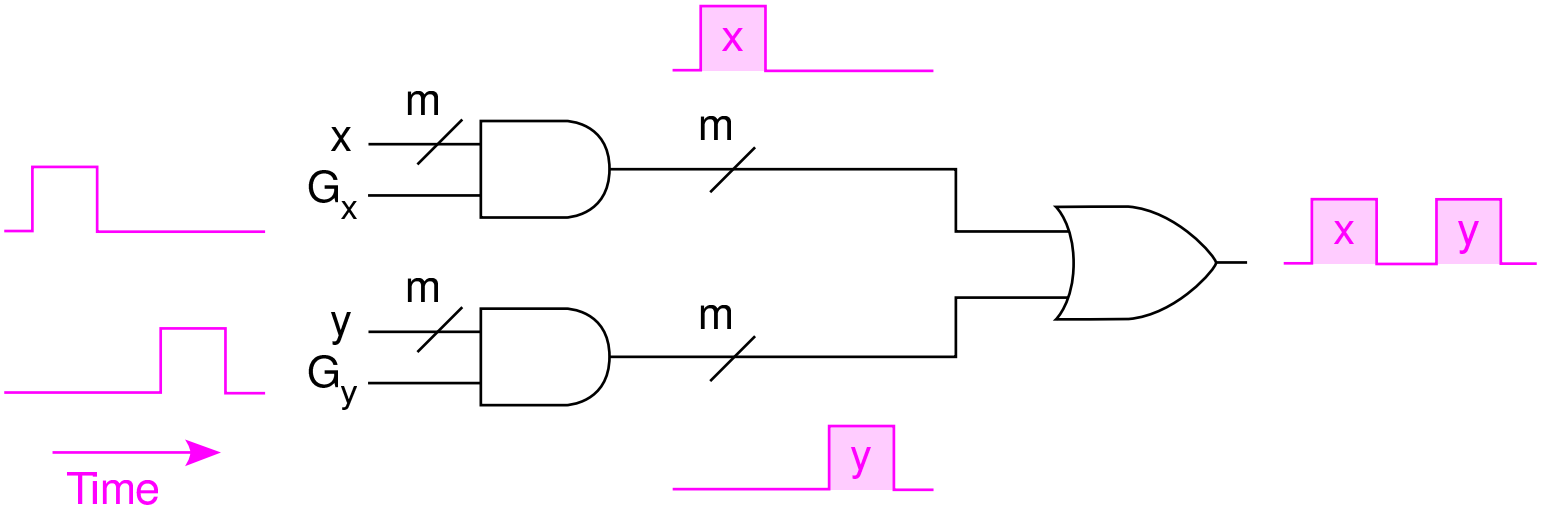
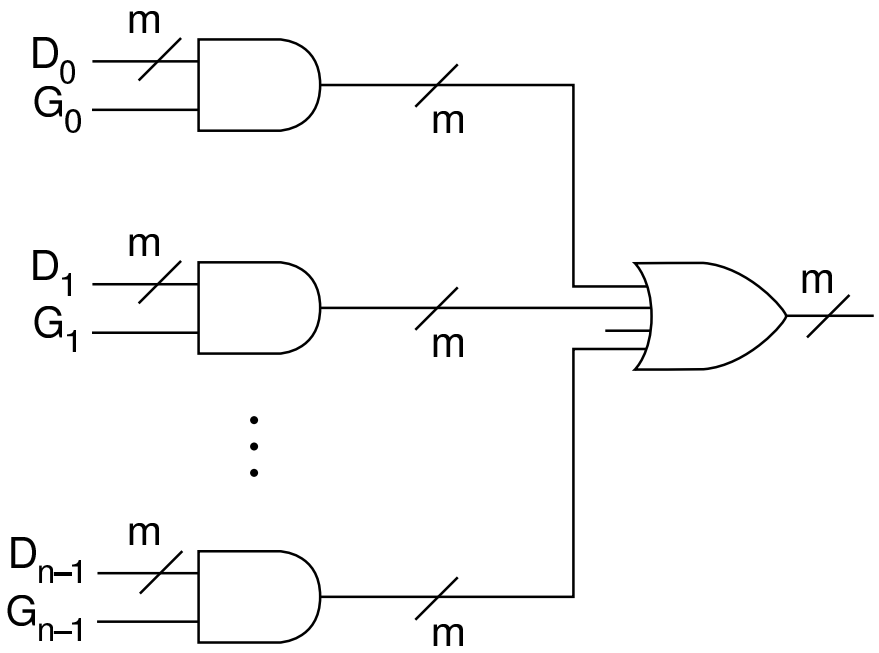


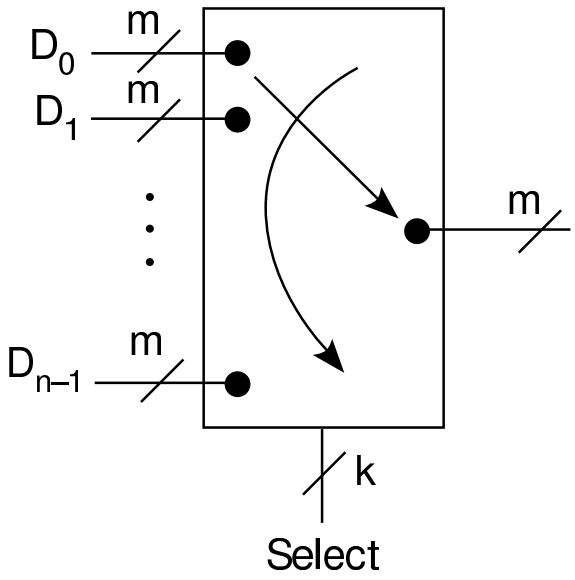
Fig 2.15 Basic Multiplexer and Symbol Abbreviation

An n-way gated merge



(a) Multiplexer in terms of gates

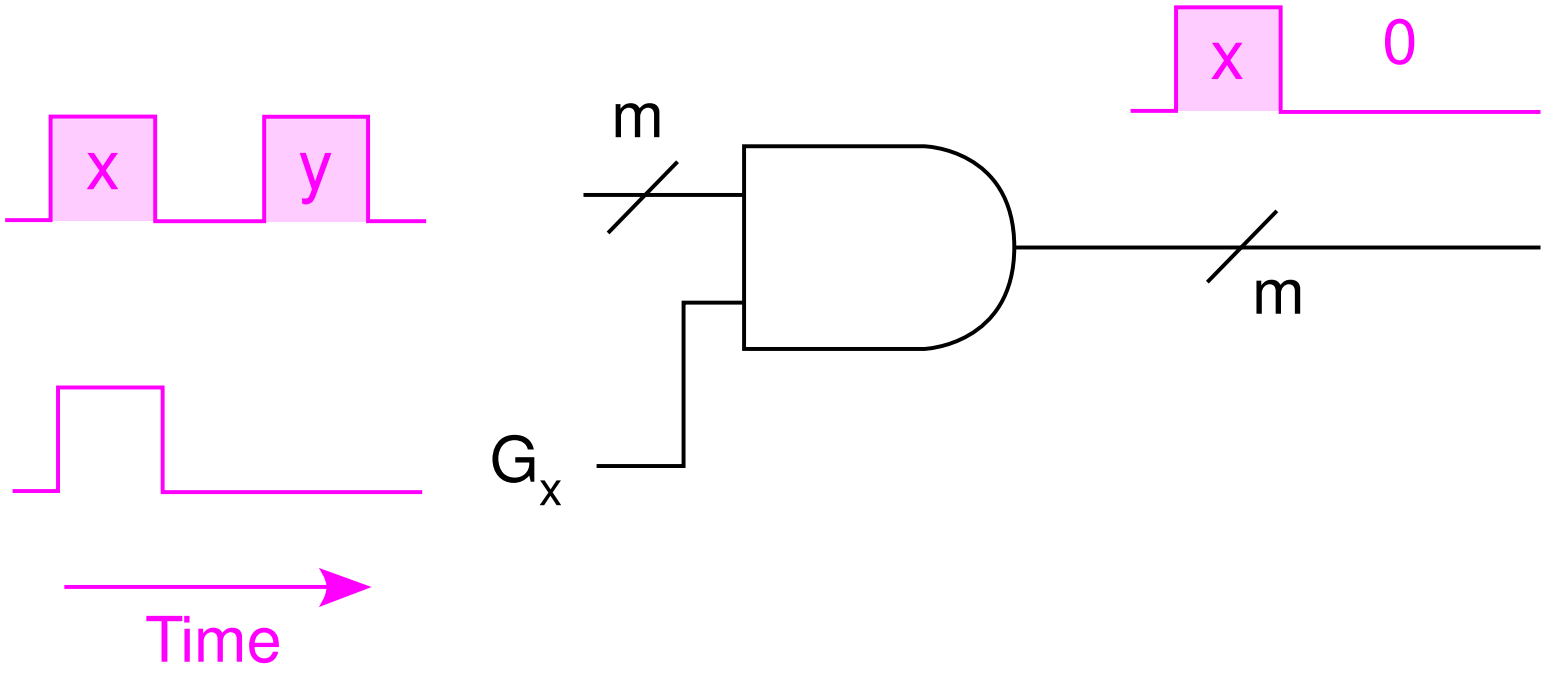
An n-way multiplexer with decoder



(b) Symbol abbreviation

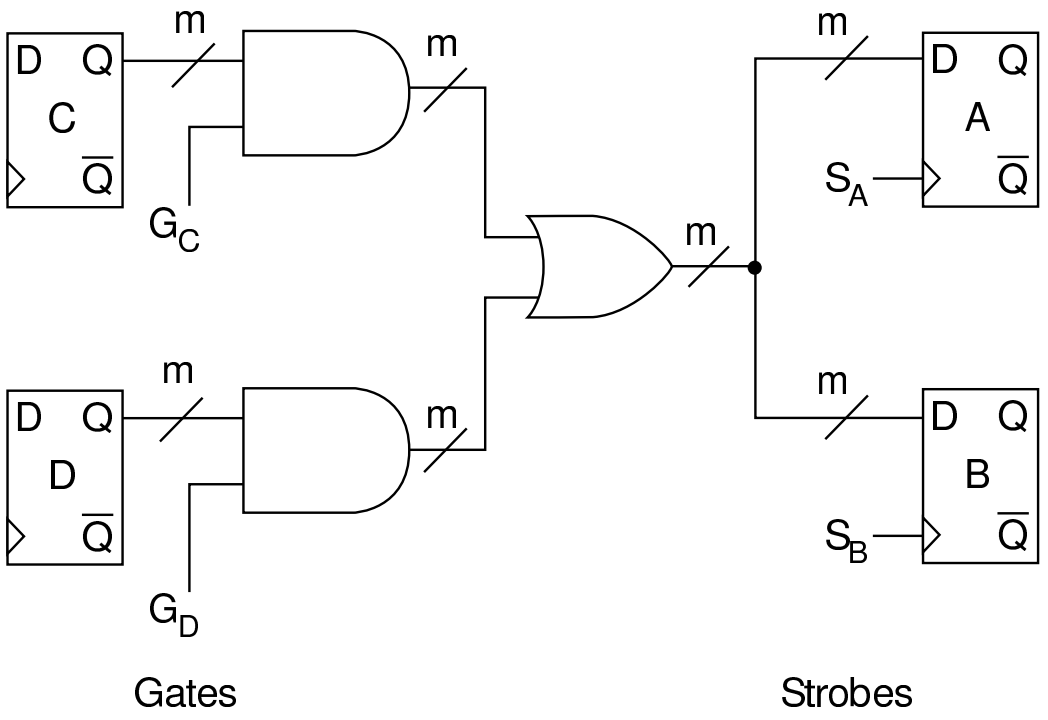
- **Multiplexer gate signals G_i may be produced by a binary to one-out-of-n decoder**

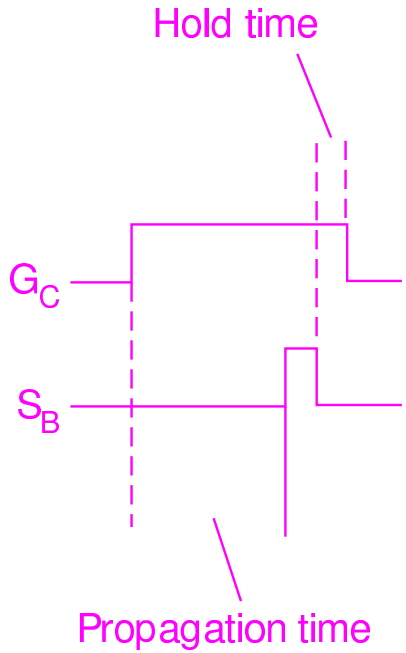
Fig 2.16 Separating Merged Data



- **Merged data can be separated by gating at the right time**
- **It can also be strobed into a flip-flop when valid**

Fig 2.17 Multiplexed Register Transfers Using Gates and Strobes



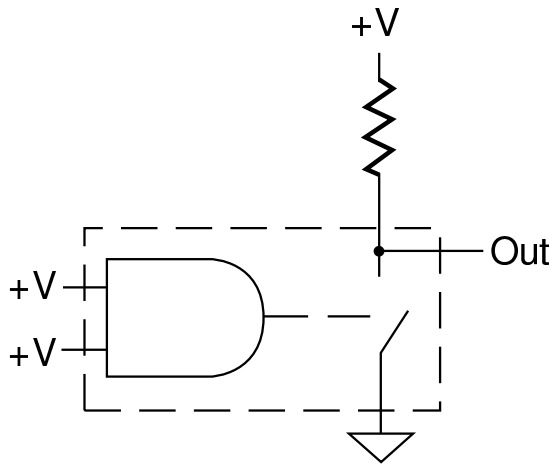


- **Selected gate and strobe determine which RT**
- **A←C and B←C can occur together, but not A←C and B←D**

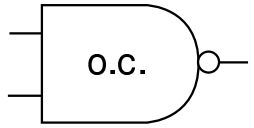
Fig 2.18 Open-Collector NAND Gate Output Circuit

Inputs		Output	
0v	0v	Open	(Out = +V)
0v	+V	Open	(Out = +V)
+V	0v	Open	(Out = +V)
+V	+V	Closed	(Out = 0v)

(a) Open-collector NAND truth table

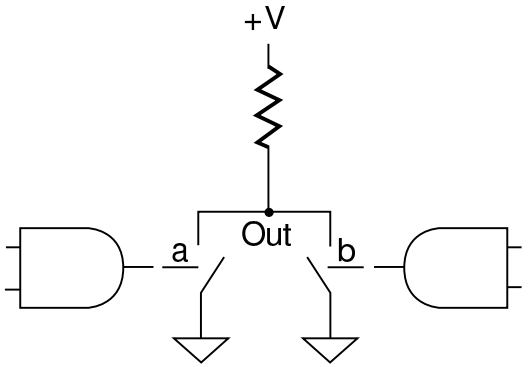


(b) Open-collector NAND

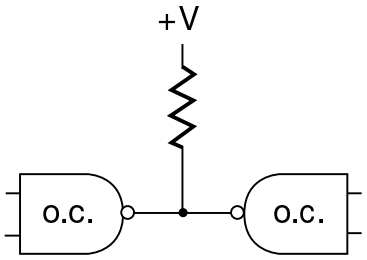


(c) Symbol

Fig 2.19 Wired AND Connection of Open-Collector Gates



(a) Wired AND connection



(b) With symbols

Switch		Wired AND output
a	b	
Closed(0)	Closed(0)	0v (0)
Closed(0)	Open (1)	0v (0)
Open (1)	Closed(0)	0v (0)
Open (1)	Open (1)	+V (1)

(c) Truth table

Fig 2.20 Open-Collector Wired OR Bus

- DeMorgan's OR by not of AND of NOTS
- Pull-up resistor removed from each gate - open collector
- One pull-up resistor for whole bus
- Forms an OR distributed over the connection

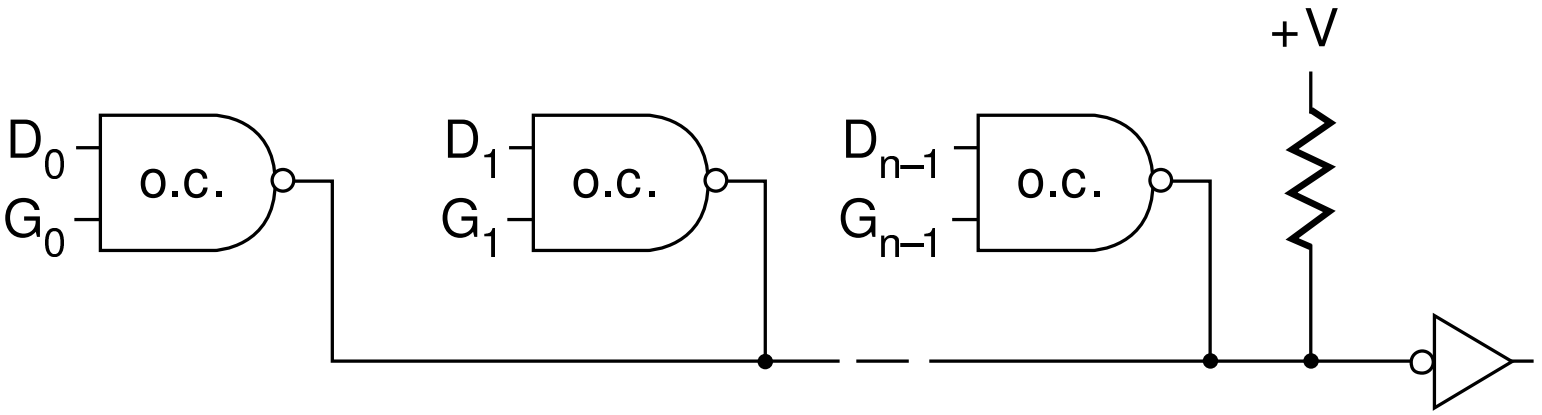
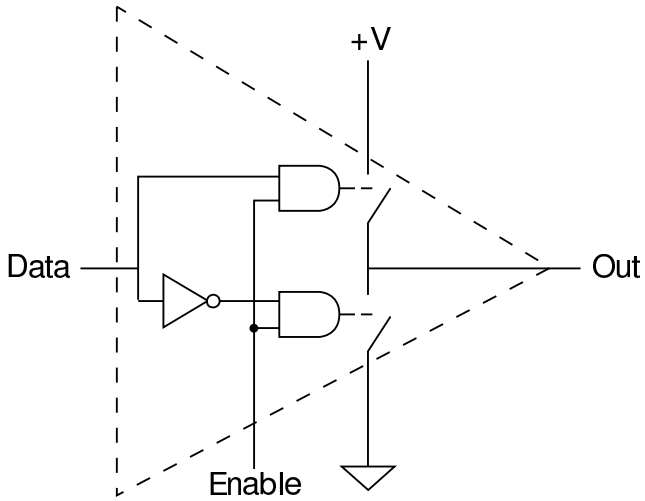
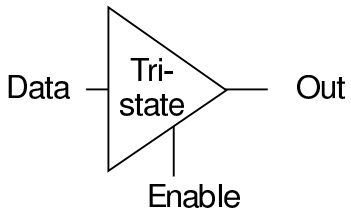


Fig 2.21 Tri-State Gate Internal Structure and Symbol



(a) Tri-state gate structure

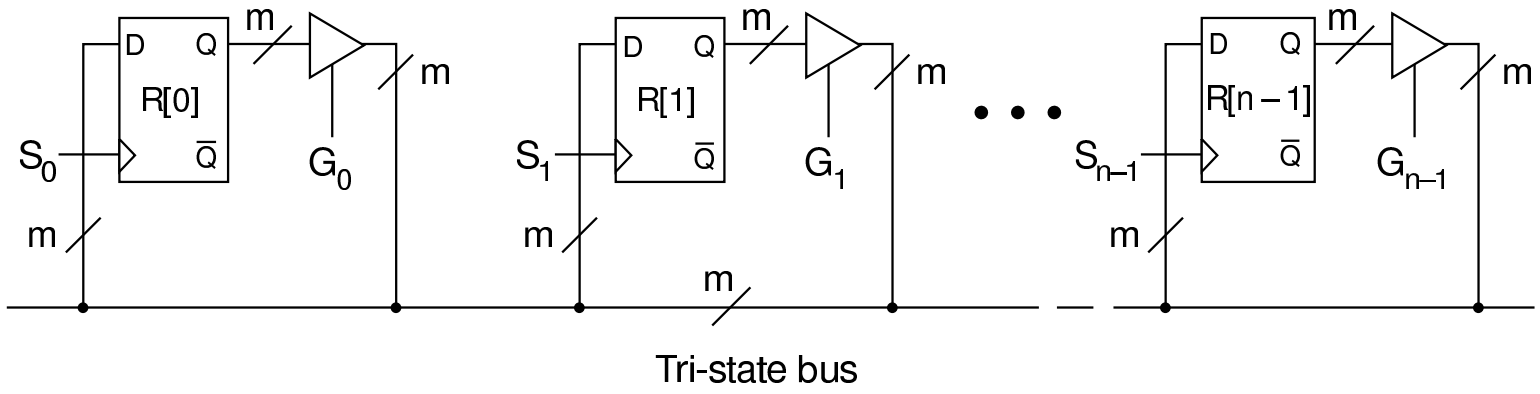


(b) Tri-state gate symbol

Enable	Data	Output
0	0	Hi-Z
0	1	Hi-Z
1	0	0
1	1	1

(c) Tri-state gate truth table

Fig 2.22 Registers Connected by a Tri-State Bus



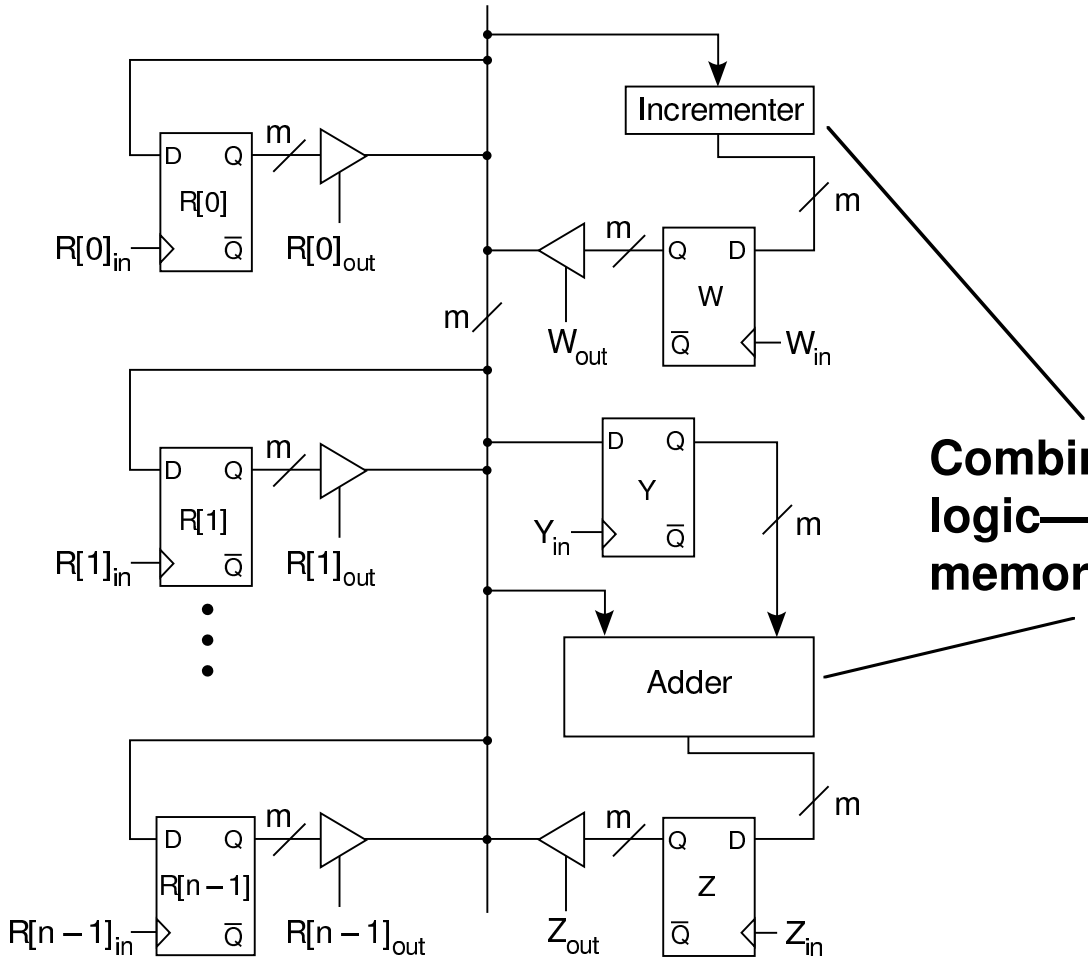
- **Can make any register transfer R[i]←R[j]**
- **Can't have $G_i = G_j = 1$ for $i \neq j$**
- **Violating this constraint gives low resistance path from power supply to ground—with predictable results!**

Fig 2.23 Registers and Arithmetic Units Connected by One Bus

Example:
Abstract RTN
R[3] ← R[1]+R[2];

Concrete RTN
Y ← R[2];
Z ← R[1]+Y;
R[3] ← Z;

Control Sequence
$R[2]_{out}$, Y_{in};
$R[1]_{out}$, Z_{in};
Z_{out}, $R[3]_{in}$;



Notice that what could be described in one step in the abstract RTN took three steps on this particular hardware

RTs Possible with the One-Bus Structure

- R[i] or Y can get the contents of anything but Y
- Since result different from operand, it cannot go on the bus that is carrying the operand
- Arithmetic units thus have result registers
- Only one of two operands can be on the bus at a time, so adder has register for one operand
- R[i] ← R[j] + R[k] is performed in 3 steps: Y←R[k]; Z←R[j] + Y; R[i]←Z;
- R[i] ← R[j] + R[k] is high level RTN description
- Y←R[k]; Z←R[j] + Y; R[i]←Z; is concrete RTN
- Map to control sequence is: $R[2]_{out}$, Y_{in}; $R[1]_{out}$, Z_{in}; Z_{out}, $R[3]_{in}$;

From Abstract RTN to Concrete RTN to Control Sequences

- The ability to begin with an abstract description, then describe a hardware design and resulting concrete RTN and control sequence is powerful.
- We shall use this method in Chapter 4 to develop various hardware designs for SRC.

Chapter 2 Summary

- Classes of computer ISAs
- Memory addressing modes
- SRC: a complete example ISA
- RTN as a description method for ISAs
- RTN description of addressing modes
- Implementation of RTN operations with digital logic circuits
- Gates, strobes, and multiplexers