DEPARTMENT OF
INFORMATION SYSTEMS
AND COMPUTER SCIENCE

More Combinational Logic

Programmable Logic

Lecture Time!

- Logic: Same Rules Apply
- Muxes and Demuxes: Controlling Output
- ROMs and PLAs: Programmable Logic

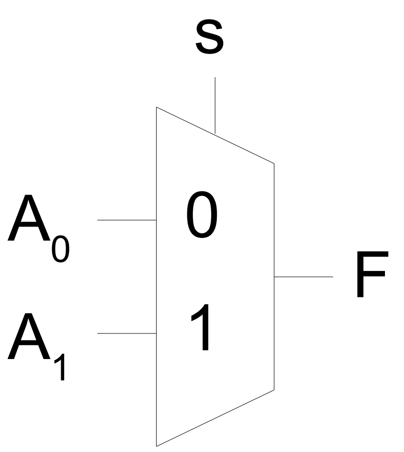
Combinational Logic Tricks

- Remember, rules in Boolean algebra apply to digital hardware as well!
- XOR can be used for encryption/decryption!
  - But easily broken since the key can be obtained if you have both the actual and encrypted text.
- Plaintext: 01110101010101110100010010100001011100
- Key: 11101010010111000101010000001010111010
- Ciphertext: 100111110000101100010000010110100100110

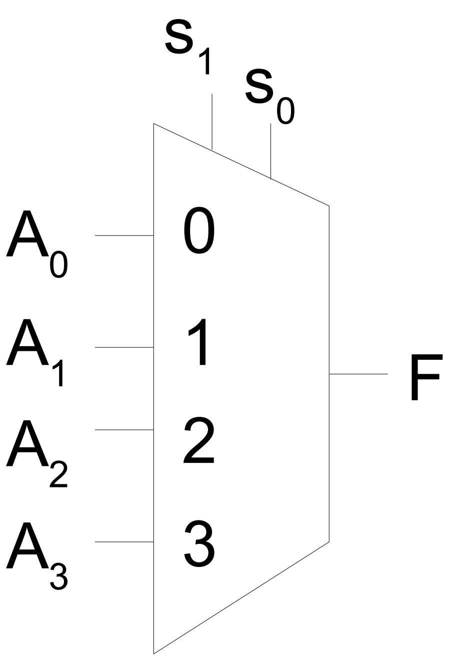
Multiplexer (Mux, Selector)

- Select 1 out of many inputs.
  - "Many to 1"
- $F = A_s$
  - Implementation of the 2-way mux?
    - Generate truth table first?
    - What about for the 4-way?

2-way mux

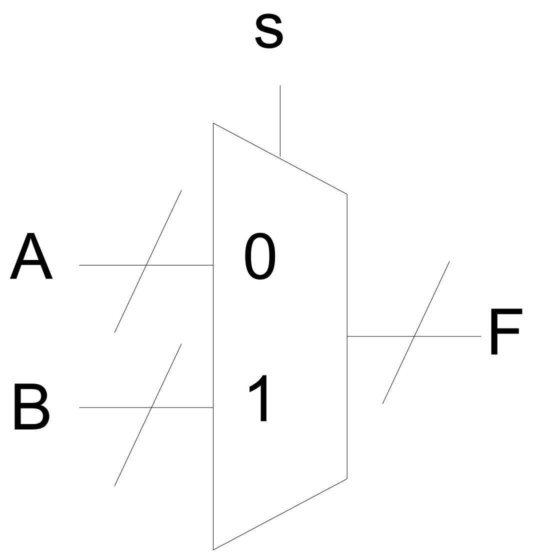


4-way mux

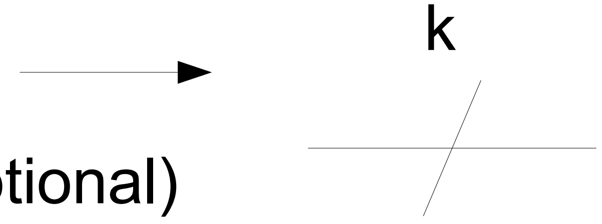


Multi-bit Wires / Busses

- If you need to choose from one of two different X-bit inputs, it might take a while to draw X 2-way multiplexers...
  - You should use the multi-bit notation indicated below.
  - Actual implementation of the multi-bit multiplexer below is still X 2-way muxes, though.



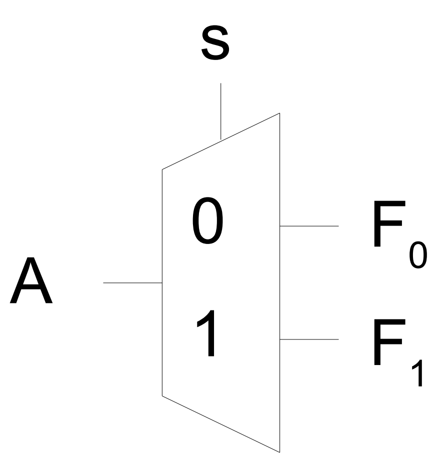
k-bit wire (writing k is sometimes optional)



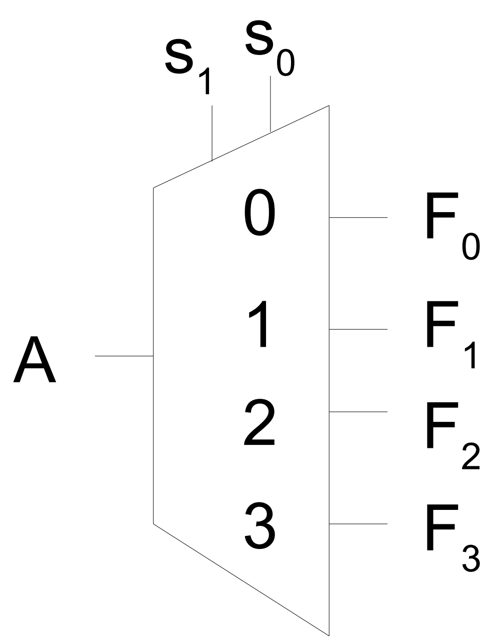
Demultiplexer (Demux)

- Pass input to one of many outputs.
  - "1 to Many"
- $F_s = A$, $F_i = 0$ where i != s
  - Implementation of the 2-way demux?
    - Generate truth table first?
    - What about for the 4-way?

2-way demux



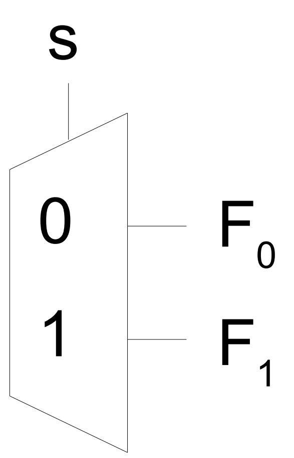
4-way demux



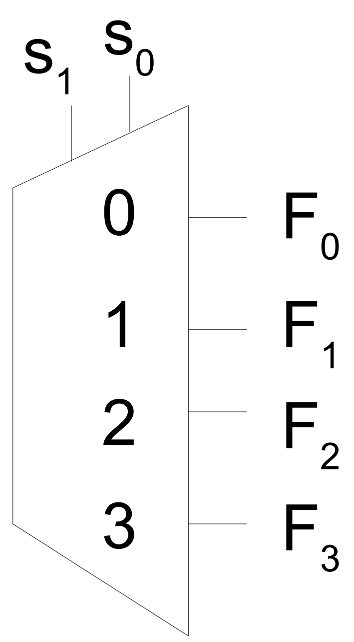
Decoder

- Assert exactly one of many outputs.
- $F_s = 1$, $F_i = 0$ where i != s
  - Implementation is similar to a demux, but input A is tied to 1.
  - Technically has no multi-bit version.

2-way decoder

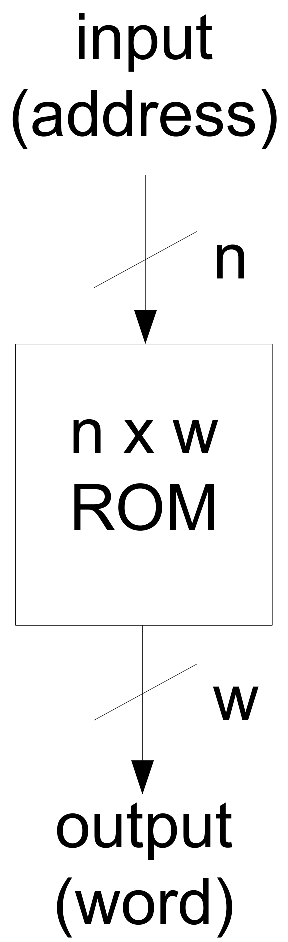


4-way decoder



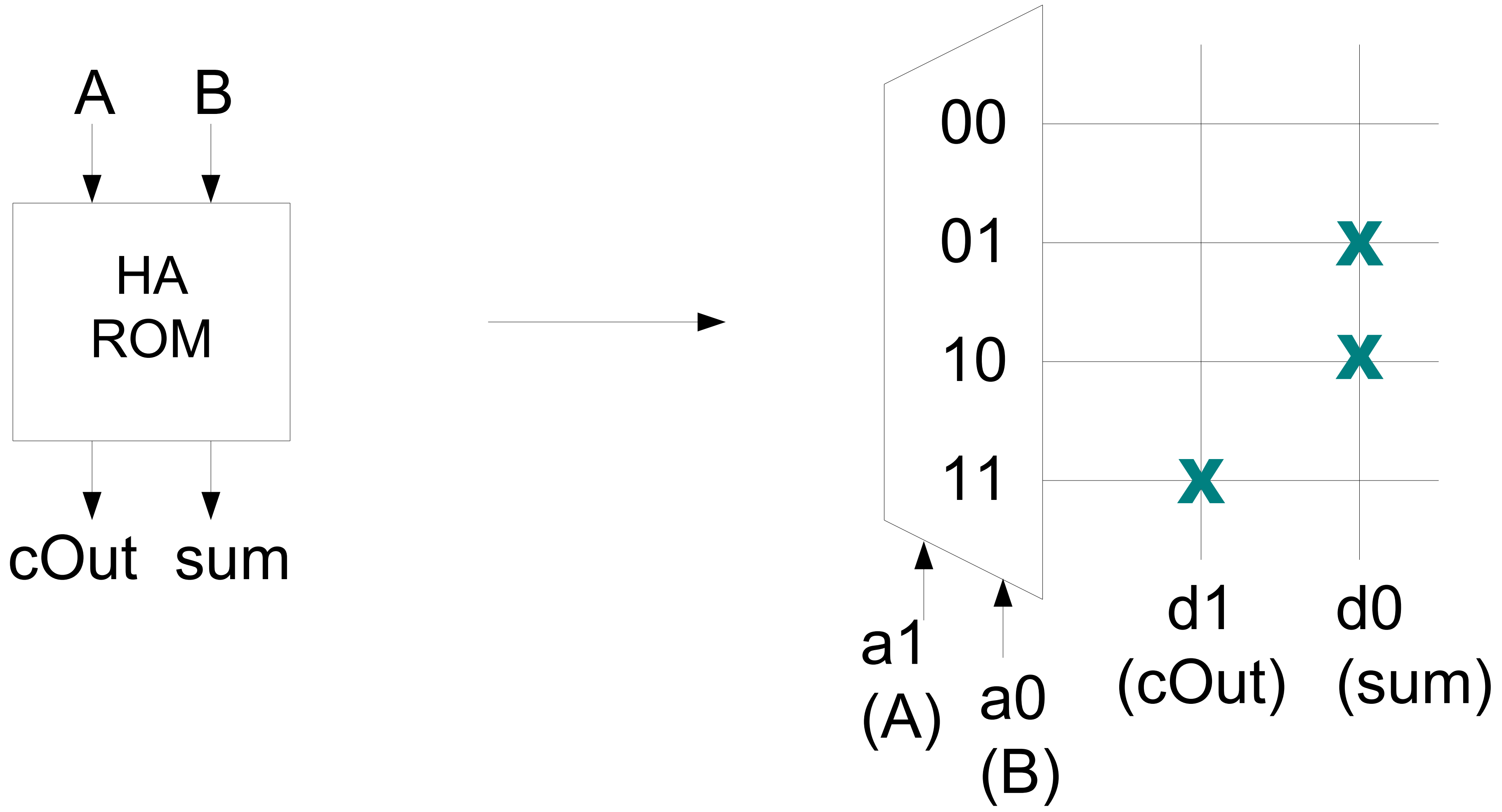
Read-Only Memory (ROM)

- Sometimes, we need PROGRAMMABLE logic.
  - Simplest form: ROM
- ROM is like a big table:
  - Input is composed of n bits, called an “address”.
  - Outputs is composed of w bits, called a “word”.
    - w is called bit-width or wordsize.
  - Can be used to remember words of data or to do functions.



ROM Implementation

- ROM is just a two-dimensional array of 0's and 1's.
- Example: Half adder



ROM Logic

- ROM is just a truth table in hardware!
- Any n-input function can be built using ROM w/ n-bit address (since you can choose whatever output you want given specific inputs).
- ROM with w-bit wordsize computes w funcs at the same time.
- No need to minimize, since doing so won't affect actual implementation anyway.

Programming ROMs

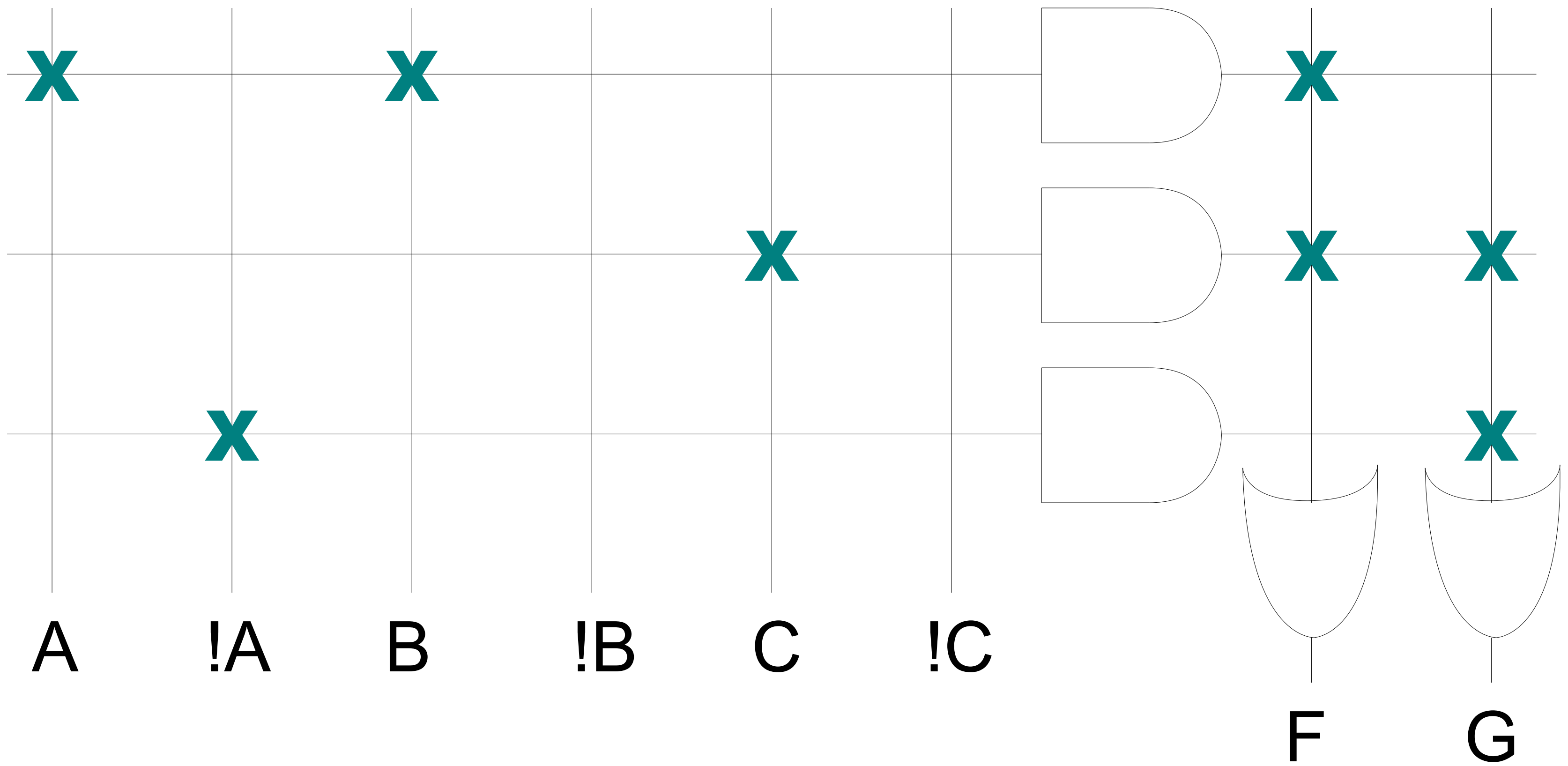
- ROM is programmed by “burning-in” connections.
  - ROM: burned-in by manufacturer (rare today, but remember cartridge-based consoles?).
  - PROM: can burn each bit only once (programmable).
  - EPROM: can “unburn” by exposing to UV light (erasable).
  - EEPROM: one type is the flash memory, found in USB flash drives (electrically erasable).
- Total size/cost is proportional to total # of bits on right side of truth table (rows*columns).

Some ROM Questions

- How many different ways can you program a 2 x 3 ROM?
  - How about a n x w ROM?
- Design a ROM for a full-adder:
  - What size ROM do you need?
    - What's n?
    - What's w?
  - What's the truth table?
  - Draw ROM circuit?
- What size ROM do you need for a 4-bit adder/subtractor?

Programmable Logic Array (PLA)

- PLA exploits structure of expression.
  - In PLA, # of rows == number of distinct Product Terms.
  - In ROM, # of rows == all 2^k possibilities for k inputs.
- PLAs use less space for more functions than ROMs!



F = AB + C

G = !A + C