IO Interface Options

Memory Mapped IO vs Instruction Based IO

- Memory Mapped:
  - IO device registers are mapped to memory addresses
  - Same CPU instructions to access memory or IO
    - Reuse all memory accessing modes/instructions
  - Reduces available address space for memory
    - Only an issue if address space is small
  - Extra logic required outside of the CPU to access memory or IO based on address supplied by CPU

- Instruction Based/Port Mapped IO
  - Special instructions that are only for IO
  - Can have separate bus for IO instructions
  - Extra logic required in the CPU to handle IO
  - Limited IO instructions

- LC3
  - Memory mapped
  - x86 can do both

Asynchronous vs Synchronous Data Transfer

- Asynchronous
  - Useful when IO or data source produces/consumes data at varying rates
  - Requires additional data/overhead for synchronization
    - checking the status register

- Synchronous
  - CPU and IO operate in lockstep at some multiple of the CPU clock
    - Read keyboard data every 100 million cycles
    - Producer and Consumers use clock/timer for synchronization

- LC3
  - Asynchronous
    - Synchronized by keyboard or display status register

Polling vs Interrupts

- Polling
  - CPU checks status at regular intervals
  - Inefficient if IO is infrequent
  - No extra interrupt logic required
  - CPU checks device

- Interrupts
  - Extra logic that changes the CPUs normal operation when IO occurs
  - IO can occur at any time
  - Inefficient if IO is frequent and predictable
  - Device alerts CPU

- LC3
  - Can do both
  - Only polling is currently implemented because interrupts require extra logic
https://en.wikipedia.org/wiki/Memory-mapped_I/O
https://en.wikipedia.org/wiki/Asynchronous_I/O
https://www.geeksforgeeks.org/asynchronous-serial-data-transfer/