

## 6.3 Sequential Circuits

### Combinational circuits.

- Basic abstraction = switch.
- In principle, can build TOY computer with a combinational circuit.
  - $255 \times 16 = 4,080$  inputs  $\Rightarrow 2^{4080}$  rows in truth table!
  - no simple pattern
  - each circuit element used at most once

Bottom line from last lecture. ALU.

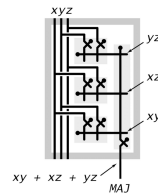
Sequential circuits. Reuse circuit elements by storing bits in "memory."

Machine architecture. Wire components together to make computer.

### Sequential vs. Combinational Circuits

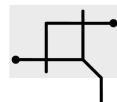
#### Combinational circuits.

- Output determined solely by inputs.
- Can draw with no loops.
- Ex: majority, adder, ALU.



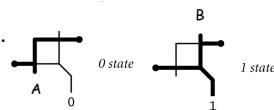
#### Sequential circuits.

- Output determined by inputs **and** previous outputs.
- Ex: memory, program counter, CPU.



Ex. Simplest feedback loop.

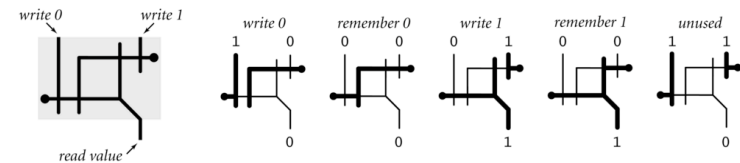
- Two relays A and B, both connected to power, each blocked by the other.
- State determined by whichever switches first.
- Stable.



### Flip-Flop

#### Flip-flop.

- A way to control the feedback loop.
- Abstraction that "remembers" one bit.
- Basic building block for memory and registers.



Caveat. Need to deal with switching delay.

## Memory Overview

Computers and TOY have several memory components.

- Program counter.
- Registers.
- Main memory.

**Implementation.** Use one flip-flop for each bit of memory.

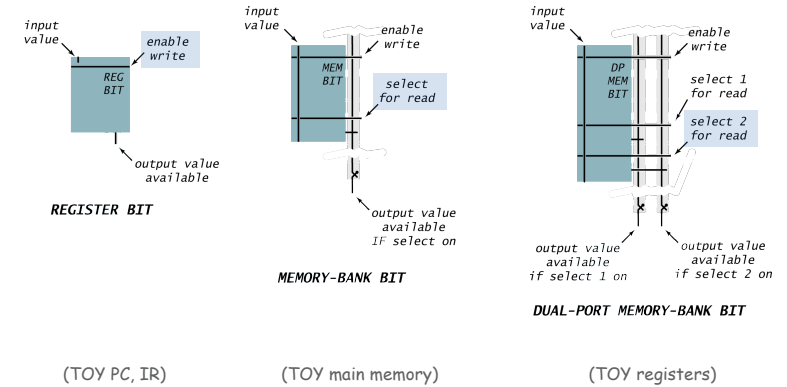
**Access.** Memory components have different access mechanisms.

**Organization.** Need mechanism to manipulate **groups** of related bits.

TOY has 16 bit words,  
8 bit memory addresses, and  
4 bit register names.

## Memory Bit: Interface

**Memory bit.** Extend a flip-flop to allow easy access to values.

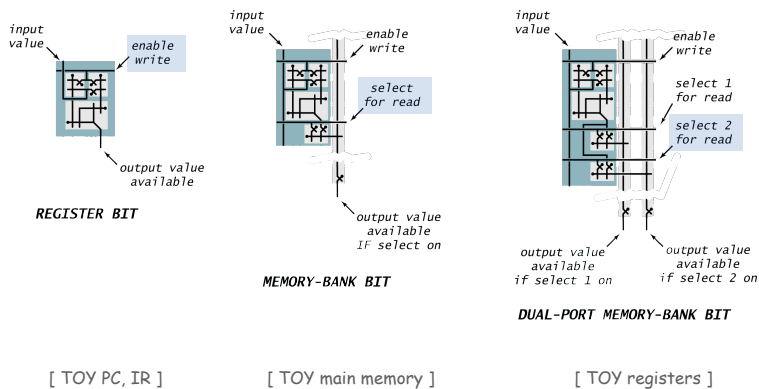


5

6

## Memory Bit: Switch Level Implementation

**Memory bit.** Extend a flip-flop to allow easy access to values.



7

## Processor Register

**Processor register.** ← don't confuse with TOY register

- Stores k bits.
- Register contents always available on output bus.
- If enable write is asserted, k input bits get copied into register.

Ex 1. TOY program counter (PC) holds 8-bit address.

Ex 2. TOY instruction register (IR) holds 16-bit current instruction.



8

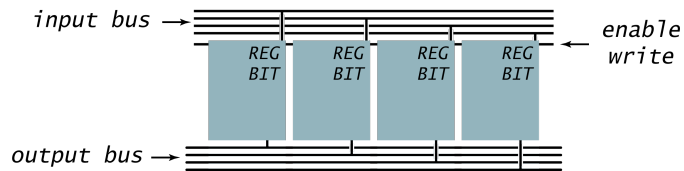
## Processor Register

Processor register. ← don't confuse with TOY register

- Stores  $k$  bits.
- Register contents always available on output bus.
- If enable write is asserted,  $k$  input bits get copied into register.

Ex 1. TOY program counter (PC) holds 8-bit address.

Ex 2. TOY instruction register (IR) holds 16-bit current instruction.



9

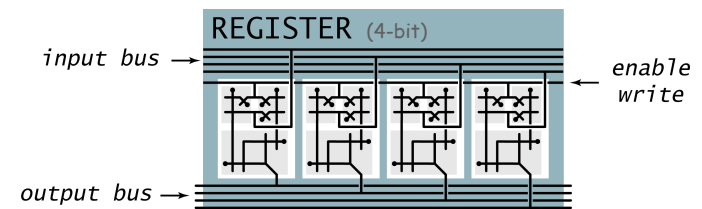
## Processor Register

Processor register. ← don't confuse with TOY register

- Stores  $k$  bits.
- Register contents always available on output bus.
- If enable write is asserted,  $k$  input bits get copied into register.

Ex 1. TOY program counter (PC) holds 8-bit address.

Ex 2. TOY instruction register (IR) holds 16-bit current instruction.



10

## Memory Bank

Memory bank.

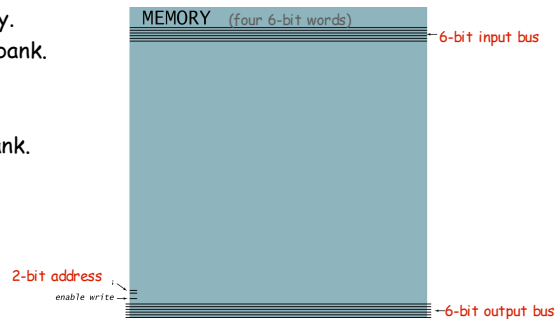
- Bank of  $n$  registers; each stores  $k$  bits.
- Read and write information to *one* of  $n$  registers.
- Address inputs specify which one. ←  $\log_2 n$  address bits needed
- Addressed bits always appear on output.
- If write enabled,  $k$  input bits are copied into addressed register.

Ex 1. TOY main memory.

- 256-by-16 memory bank.

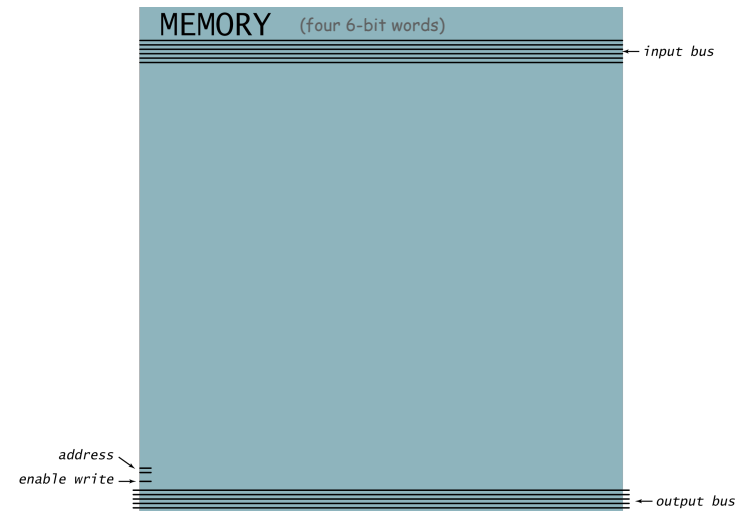
Ex 2. TOY registers.

- 16-by-16 memory bank.
- Two output buses.



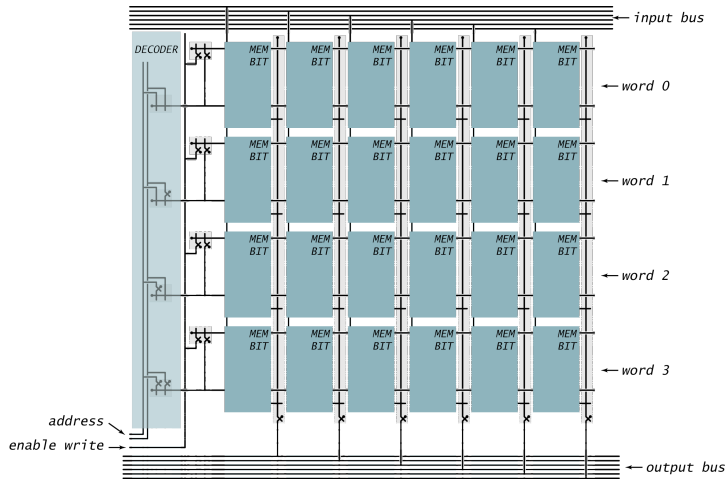
11

## Memory: Interface



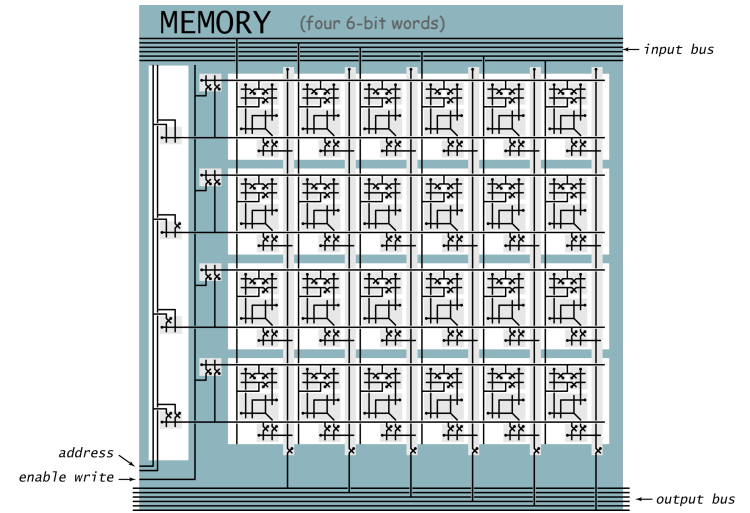
12

## Memory: Component Level Implementation



13

## Memory: Switch Level Implementation



14

## Summary

Sequential circuits add "state" to digital hardware.

- Flip-flop. represents 1 bit
- TOY word. 16 flip-flops
- TOY registers. 16 words
- TOY main memory. 256 words

Modern technologies for registers and main memory are different.

- Few registers, easily accessible, high cost per bit.
- Huge main memories, less accessible, low cost per bit.
- Drastic evolution of technology over time.

Next time. Build a complete TOY computer.

15