Finite State Machines

A formal model used to design sequential circuits to perform tasks that only require a finite number of distinct states of the flip flops.

A classic example:
Stop light at an intersection

- Outputs:
  - NSlite = 1 (green), NSlite = 0 (red)
  - EWlite = 1 (green), EWlite = 0 (red)

- Inputs:
  - NScar = 1 (NS bound car waiting)
  - EWcar = 1 (EW bound car waiting)

Stop light at an intersection

- States: NSgreen and EWgreen
- Next-state function:
  \[ f(\text{current-state}, \text{input}) = \text{next-state} \]
  \[ f(\text{NSgreen}, \neg \text{NScar} \text{ and } \neg \text{EWcar}) = \text{NSgreen} \]

- Output function:
  \[ g(\text{current-state}) = \text{output} \]
  \[ g(\text{NSgreen}) = \text{NSLite and } \neg \text{EWlite} \]

Finite State Machine Circuit

Finite State Machine state transition diagram
Other common finite state machines in the processor

- Counters
- Shift registers

MIPS Datapath
(Single Cycle and Multi-Cycle)

Basic MIPS Implementation

- For a limited subset of the MIPS instructions
  - Memory reference: LW and SW
  - Arithmetic-logical: add, sub, and, or, slt
  - Branch: beq
- Hardware components: PC, registers, memory units, ALU, multiplexors, decoders

Single Cycle v. Multi-Cycle

- Single cycle is not realistic - just for understanding of the implementation.
- Single cycle: one (long) clock cycle to process each instruction
- Multi-cycle: divide the processing of each instruction into 5 stages and allocate one clock cycle per stage

Major Functional Units (Fig. 5.1)

Functional Units and Control Lines (Fig. 5.2)
HW to fetch instruction from memory and increment the PC

HW for memory transfer or R-type instructions

HW to evaluate branch condition and compute branch target address

Single Cycle Datapath Details

• Chapter 5.4 gives the bit-level details of how this HW works to fetch and execute these basic instructions.
• This section is optional reading for this class.

Multicycle Datapath

• Break the operations on an instruction into a series of 5 steps.
• One clock cycle per step

• Instruction Fetch (IF)
• Instruction Decode (ID)
• Execute (EX)
• Memory Access (MEM)
• Write Back (WB)

Multicycle Datapath

• HW changes:
  – Single memory unit for both instructions and data
  – Single ALU for all arithmetic operations
  – Extra registers needed to hold values between each steps
  – Instruction Register (IR) holds the instruction
  – Memory Data Register (MDR) holds the data coming from memory
  – A, B hold operand data coming from the registers
  – ALUOut holds output coming out of the ALU
Extra hardware for multicycle datapath

Multicycle datapath (Figure 5.26)

Complete multicycle datapath (Fig. 5.28)

Multicycle Datapath: the 5 steps

Figure 5.30 -- see handout