ECE 153A HW 3

Oviya Seeniraj

Question 1

Two states are equivalent if they have all the same inputs, which, in combination with each state as a current state, transitions to the same next state and output pairings. So, we can check for identical next state and output pairings to find equivalent states.

States c and e both have the same output values and next states given the same inputs:

0, c → a, 0 0, e → a, 0
1, c → c, 0 1, e → c, 0

Thus, states c and e are equivalent. Thus, we can collapse them into the same state (c) and reroute all states resulting in e to state c to minimize the FSM.

Similarly, b and f share transitions as well and can be collapsed to a single state "b". While their transitions are not identical, they share the same transition for an input of 1, and for an input of 0, transition to each other. Thus, states b and f are equivalent.

Minimized FSM:

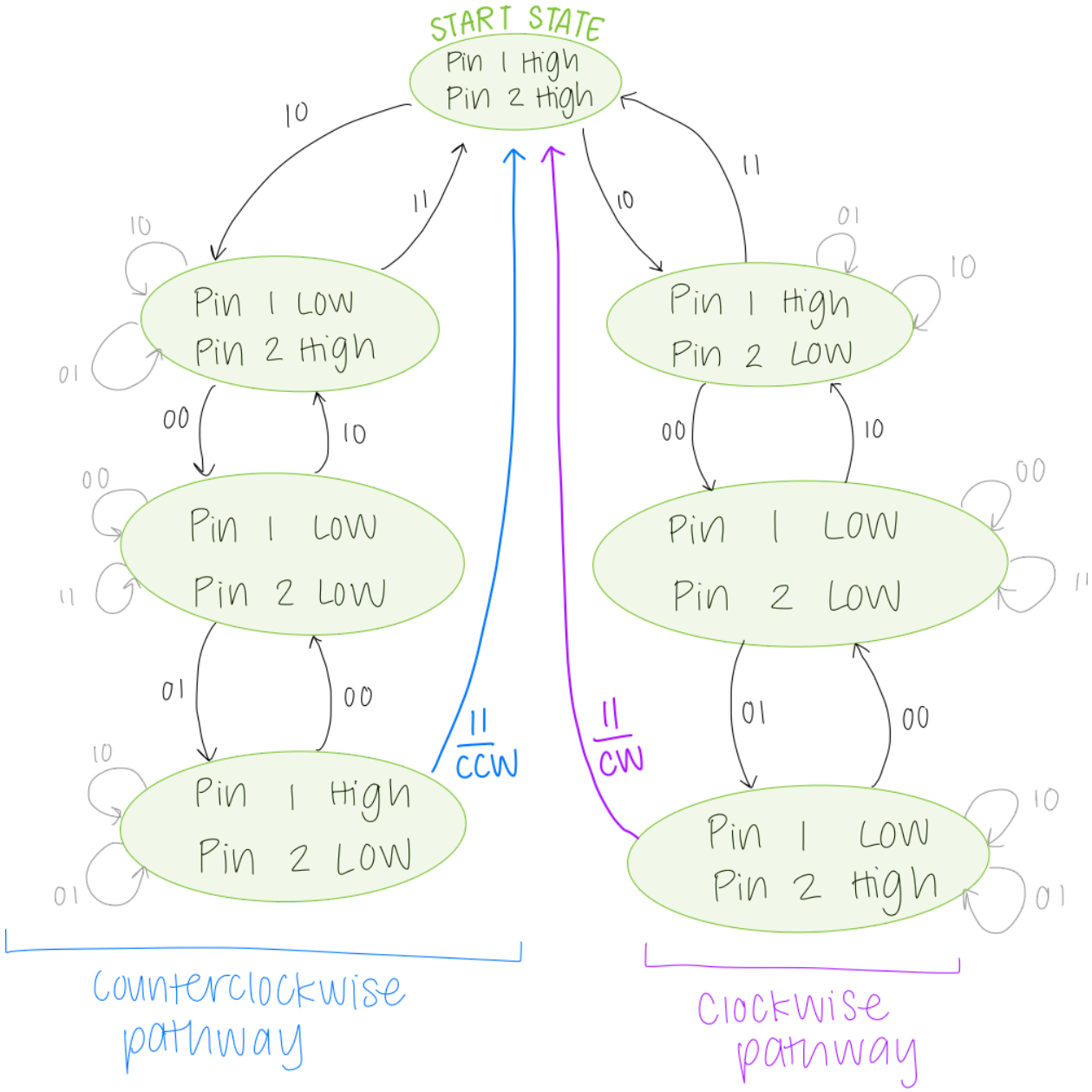
- States: a, b, c, d
- Transitions:
  - 0, a → b, 1
  - 1, a → c, 0
  - 0, b → b, 0
  - 1, b → d, 0
  - 0, c → a, 0
  - 1, c → c, 0
  - 0, d → c, 1
  - 1, d → c, 0

Question 2

Part A & B

The solution for part B (below) is also a solution for Part A, whose only condition is that a clockwise or counterclockwise signal is correctly produced, which B's machine does (along with accounting for debouncing).

State/"bubble" diagram:



This FSM checks for clockwise and counterclockwise turns by checking for each pin input change that is required to compose a turn (for CW, 11 → 10 → 00 → 01 and for CCW 11 → 01 → 00 → 10) and once that path is complete, returns to the start state and outputs whether the recently completed turn was CW or CCW.

It accounts for debouncing by returning to the previous state when the pin from the last input that was held constant continues to be held constant, and the pin whose input changed changes again (aka, one pin “bounces” between its two states, hence why we need to create de-bouncing logic). When the pins continue to change state along the earlier specified CW or CCW state, it resumes its regular cycle on the way to produce a CW/CCW signal.

The transitions labeled in gray are not needed for this machine, as it is not possible to change two bits at once, so if this input was received, nothing should change as it is invalid (aka, loop back to itself). No change in the inputs would not trigger a change if this is an asynchronous circuit, and if this is a clock-based circuit, the gray would account for this by looping back to itself.