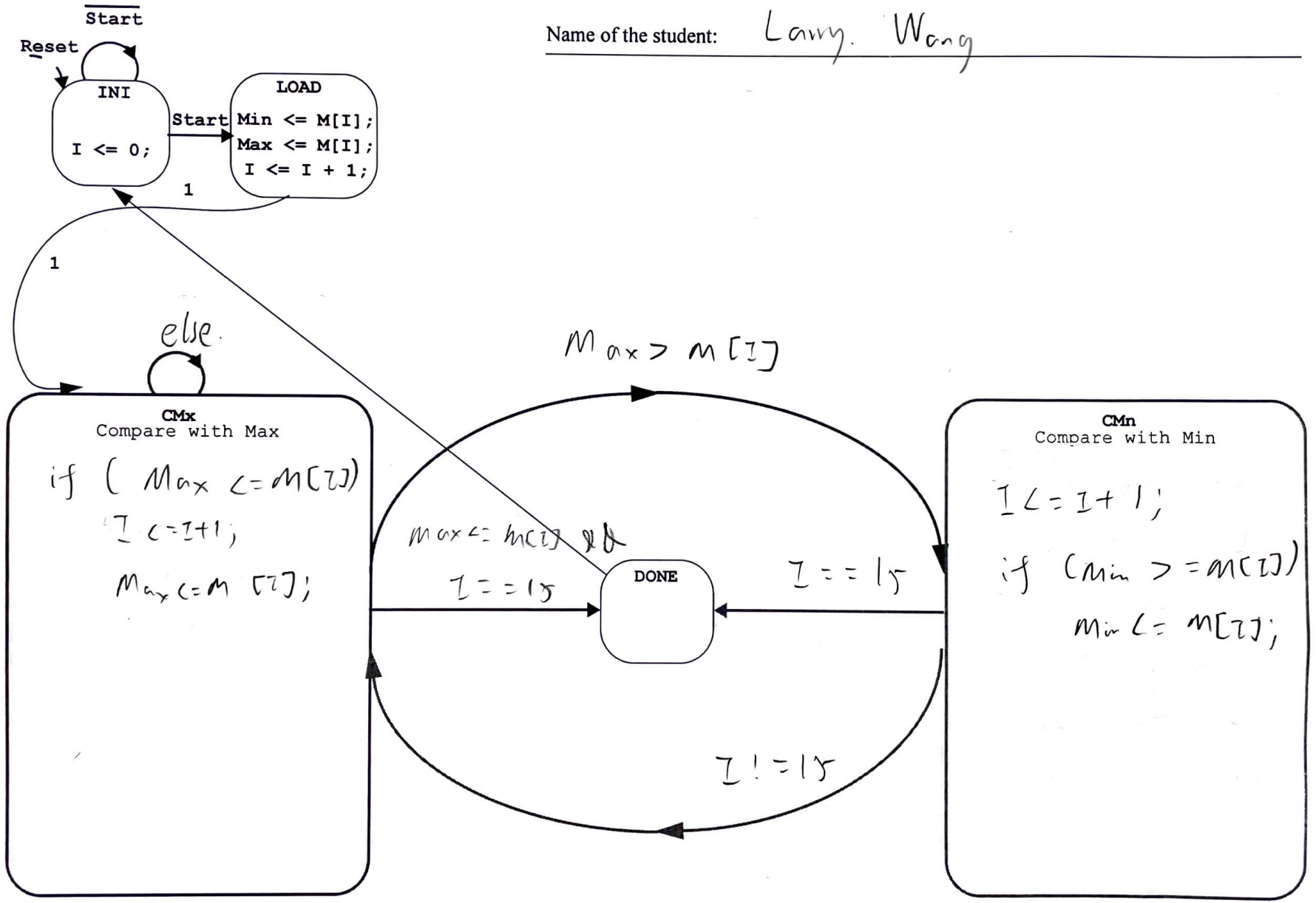
State Diagram for Part 2 for 1 comparator

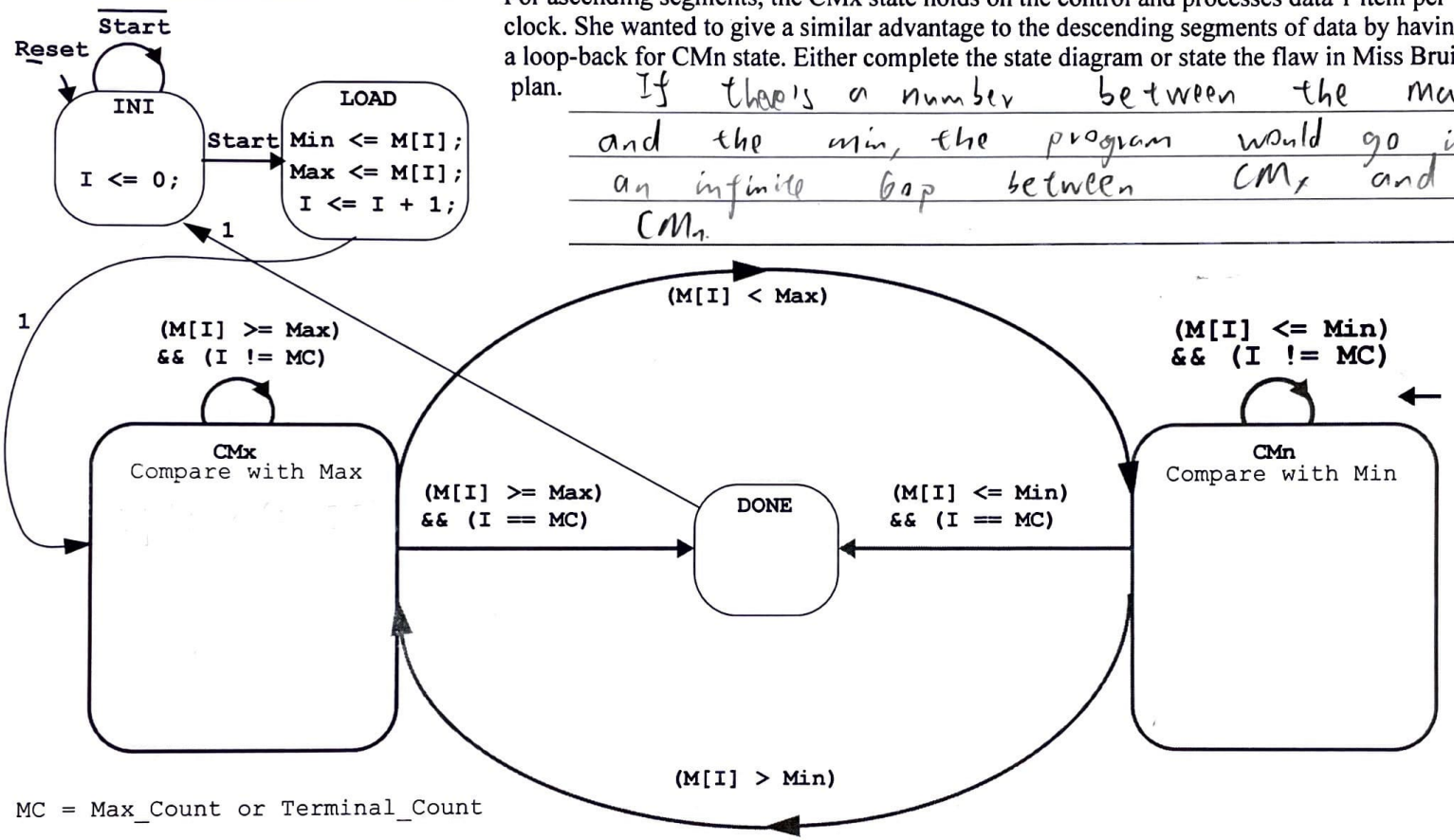
Here a new M[I] is first compared with Max in CMx and then only if, needed, it is compared with Min in CMn state.

Name of the student: Larry. Wang



Miss Bruin's improvement for Part 2

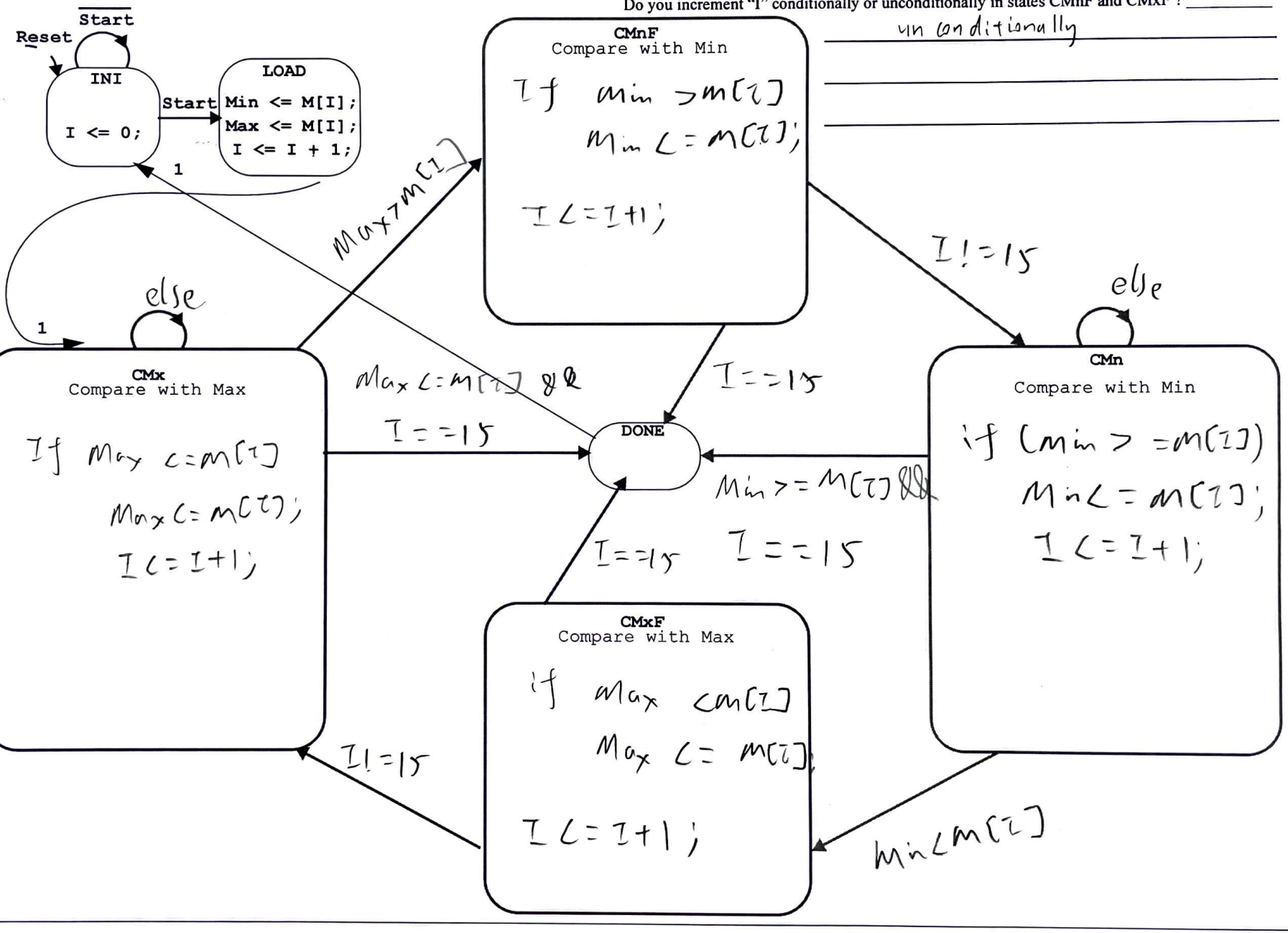
Miss Bruin was told that the data has long sequences of ascending and descending segments. For ascending segments, the CMx state holds on the control and processes data 1 item per clock. She wanted to give a similar advantage to the descending segments of data by having a loop-back for CMn state. Either complete the state diagram or state the flaw in Miss Bruin's plan. If there's a number between the max and the min, the program would go into an infinite loop between CMx and CMn.



MC = Max_Count or Terminal_Count

State Diagram for Part 3 Method 1

CMnF: Compare with Min. for the first time after a series of comparisons in CMx state
CMxF: Compare with Max. for the first time after a series of comparisons in CMn state
Do you increment "I" conditionally or unconditionally in states CMnF and CMxF ? unconditionally



State Diagram for Part 3 Method 3 (similar to Method 1)

Earlier CMnF is now merged with CMn state. When control is received from CMx to CMn, the Flag is received in SET condition for the CMn state to recognize that the current M[I] is already compared with the Max and hence the "I" can be incremented unconditionally at the end of the current clock irrespective of whether M[I] is less than (or equal) to Min.
Similarly the earlier CMxF is now merged with CMx.