*Preliminary Work:*

**Step 1: State the inputs and outputs of the state registers.**

Inputs: N2, N1, N0

Outputs: S2, S1, S0

**Step 2: State the inputs and outputs of the combinational block.**

Inputs: S2, S1, S0, X

Outputs: N2, N1, N0, Y1, Y0

**Step 3: Write each output (including next state bits) as a function of the inputs.**

N2 = X((S2(S1+S0)’)+S2’S1S0)

N1 = (S2’X)(S1^S0)

N0 = ((S2’X’)+((S1+S0)’(S2X’)))+(S1X(S2+S0)’)

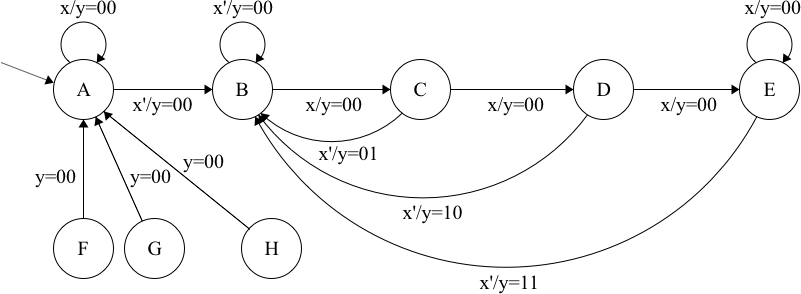
Y1 = X’((S2(S1+S0)’)+(S2’S1S0))

Y0 = (S2^S1)(S0+X)’

**Step 4: Draw the truth table for the combinational circuit.**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **S2** | **S1** | **S0** | **X** | **N2** | **N1** | **N0** | **Y1** | **Y0** |
| **0** | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| **1** | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| **2** | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| **3** | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |
| **4** | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| **5** | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 |
| **6** | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| **7** | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| **8** | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| **9** | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| **10** | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| **11** | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| **12** | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **13** | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| **14** | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| **15** | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |

**Step 5: Draw the finite state machine by using the truth table.**



(Mealy FSM with state encoding A: 000 – B: 001 – C: 010 - … - H: 111)

**Step 6: How many unreachable states does the finite state machine contain? (No explanation, only short answer)**

# of Unreachable States: 3

**Step 7: Briefly explain the relation between the input and the output.**

Explanation: It’s a sequence detector that:

* Ignores the initial 1’s.
* Gives output y=01 on input sequence 010.
* Gives output y=10 on input sequence 0110.
* Gives output y=11 on input sequence 011…10 (3 or more 1’s).
* Detects overlapping input sequences such as 01010 (gives output y=01 twice for this case).
* Gives output y=00 otherwise.