# COMP I22/L Lecture 27 Kyle Dewey 

## Outline

- Finite state machines


## Finite State Machines

## Finite State Machines

Basic idea: computation is done via traversal of states, where the states are known ahead of time.

## Finite State Machines

Basic idea: computation is done via traversal of states, where the states are known ahead of time.


## Example: Lock and Key



## Example: Counting Change



## Significance

- Can encode many problems using finite state machines (FSMs)
- FSMs can be implemented with sequential circuits
- Internals of processors can be encoded with FSMs


## Significance

- Can encode many problems using finite state machines (FSMs)
- FSMs can be implemented with sequential circuits
- Internals of processors can be encoded with FSMs



## FSMs to Circuits

Step I: Encode each state in binary

## FSMs to Circuits

## Step I: Encode each state in binary



## FSMs to Circuits

## Step I: Encode each state in binary



## FSMs to Circuits

Step 2: Make truth table mapping current state to next state

## FSMs to Circuits

Step 2: Make truth table mapping current state to next state


## FSMs to Circuits

Step 2: Make truth table mapping current state to next state


| S1 | S0 | N1 | N0 |
| :---: | :---: | :---: | :---: |
| 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 1 | $X$ | $X$ |

## FSMs to Circuits

Step 3: Simplify truth table (with Boolean algebra / K-maps)

## FSMs to Circuits

Step 3: Simplify truth table (with Boolean algebra / K-maps)

|  |  |  |  | S1 |  | For N1 |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| S1 | SO | N1 | NO | SO | 0 | 1 |  |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 |  |
| 0 | 1 | 1 | 0 | 1 | 1 | X |  |
| 1 | 0 | 0 | 0 |  |  |  |  |
| 1 | 1 | X | X |  |  |  |  |

## FSMs to Circuits

Step 3: Simplify truth table (with Boolean algebra / K-maps)

|  |  |  |  | S1 |  | For N1 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| S1 | S0 | N1 | N0 | S0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 | $X$ |
| 1 | 0 | 0 | 0 |  |  |  |
| 1 | 1 | $X$ | $X$ |  |  |  |

## FSMs to Circuits

Step 3: Simplify truth table (with Boolean algebra / K-maps)

|  |  |  |  | S1 |  | For N1 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| S1 | S0 | N1 | N0 | S0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 | $X$ |
| 1 | 0 | 0 | 0 |  |  |  |
| 1 | 1 | $X$ | $X$ |  |  |  |

## FSMs to Circuits

Step 4: Build sequential circuit

## FSMs to Circuits

Step 4: Build sequential circuit

| S1 | S0 | N1 | N0 |
| :---: | :---: | :---: | :---: |
| 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 1 | $X$ | $X$ |
| $N 0=!S 1!S 0$ |  |  |  |
| NO $=$ S 0 |  |  |  |
| N |  |  |  |

## FSMs to Circuits

Step 4: Build sequential circuit


## FSMs with External Inputs

Same process, but with more inputs in the truth table

## FSMs with External Inputs

Same process, but with more inputs in the truth table


## FSMs with External Inputs

## Same process, but with more inputs in the truth table



| $K I$ | $K R$ | $K T$ | SI | S0 | N1 | N0 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $\ldots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |

# FSMs with Outputs 

Additional outputs in truth table.
Output on the corresponding state.

## FSMs with Outputs

Additional outputs in truth table.
Output on the corresponding state.


## FSMs with Outputs

Additional outputs in truth table.
Output on the corresponding state.


## FSMs with Outputs

Additional outputs in truth table.
Output on the corresponding state.

| Green 00 | S1 | SO | N1 | NO | G | Y | R |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| !GY!R | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| Red 10 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| G. MR | 1 | 1 | X | X | X | X | X |

