



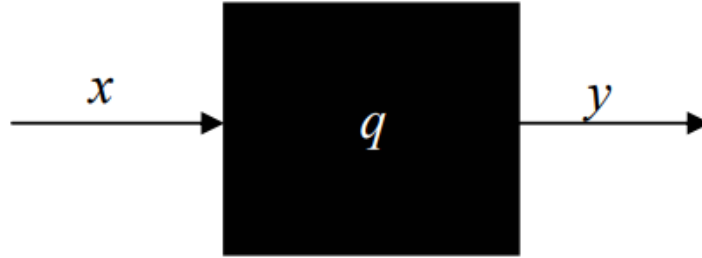
Introduction to Embedded Systems, Lecture 2

Modeling Embedded Systems by Finite Automata (State Machines)

Dmitry Zaitsev

<http://daze.ho.ua>

Verbal definition



An automaton is a system whose function can be described using the internal states $Q = \{q\}$ in such a way that for each internal state it is indicated which state the system will reach next upon obtaining input symbol x and what symbol y is generated at the output. The initial state of the system is indicated. In the case of finite sets X , Y , and Q , the automaton is called finite.

Example of automaton

- **3 story's building lift control:**
- lift can either dwell on a floor or move between floors
- there is a set of buttons, corresponding to floors, inside the lift and on each floor

Scheme of lift

Storys of building

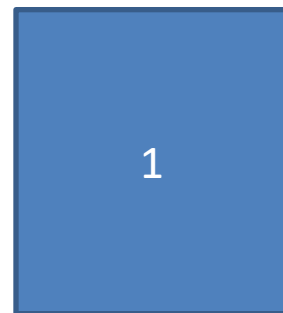


3

buttons



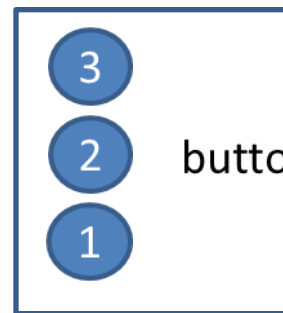
2



1



Lift



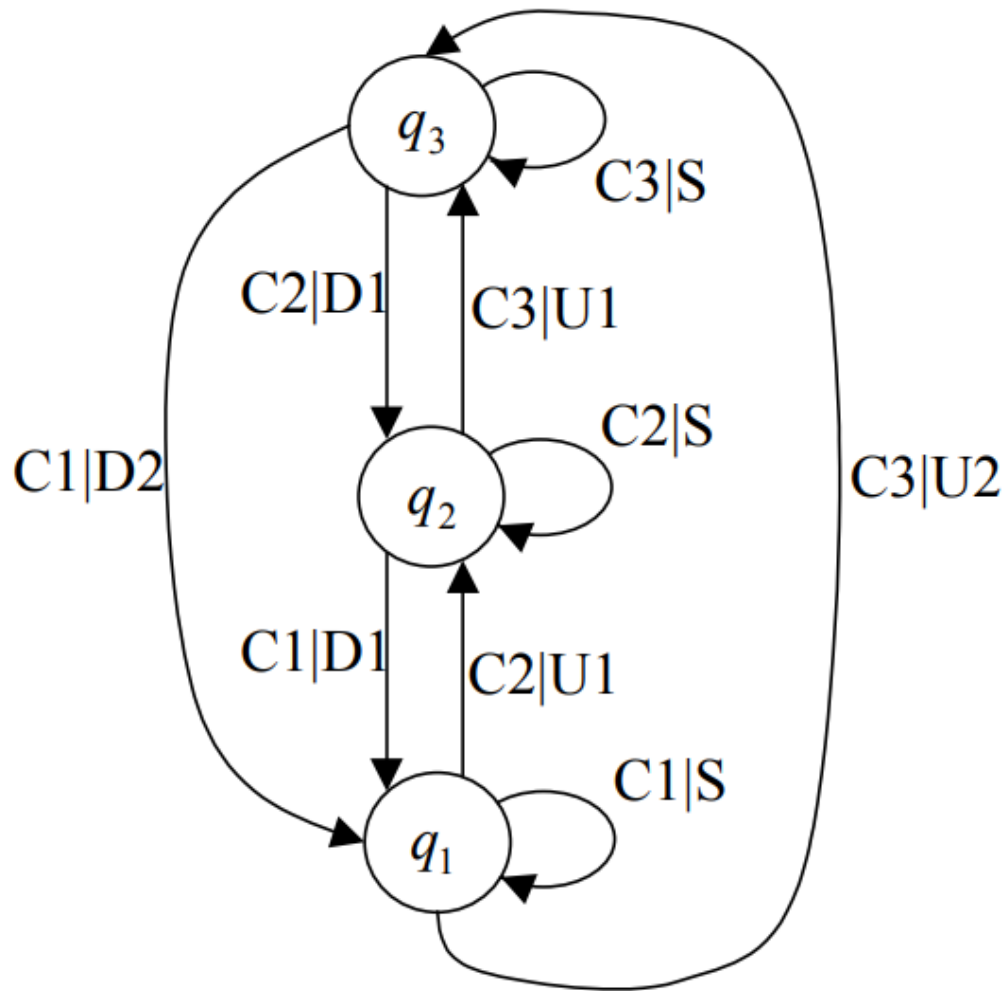
buttons

Abstract synthesis of automata

Specifying basic sets and initial state:

- the input alphabet consists of pressing the call button of the corresponding floor: $X = \{C1, C2, C3\}$
- the output alphabet consists of moves of one or two floors up or down, or stop: $Y = \{U1, U2, D1, D2, S\}$
- the state corresponds to the floor on which the automaton is: $Q = \{q1, q2, q3\}$
- the initial state $q1$

Lift behavior: State diagram



Lift behavior:

State transition table

Q/X	C1	C2	C3
q_1	$q_1 S$	$q_2 U1$	$q_3 U2$
q_2	$q_1 D1$	$q_2 S$	$q_3 U1$
q_3	$q_1 D2$	$q_2 D1$	$q_3 S$

Mealy and Moore Machines

- **Mealy machine** – the output symbol depends on combination of the input symbol and the current state
- **Moore machine** – the output symbol depends on the current state only

Definition of Mealy Machine

- $A=(X,Q,Y,q_0,F)$
- $X=\{x\}$ – finite set of input symbols
- $Q=\{q\}$ – finite set of internal states
- $Y=\{y\}$ – finite set of output symbols
- q_0 – initial state
- $F: X \times Q \rightarrow Y \times Q$ – transition function

Definition of Moore Machine

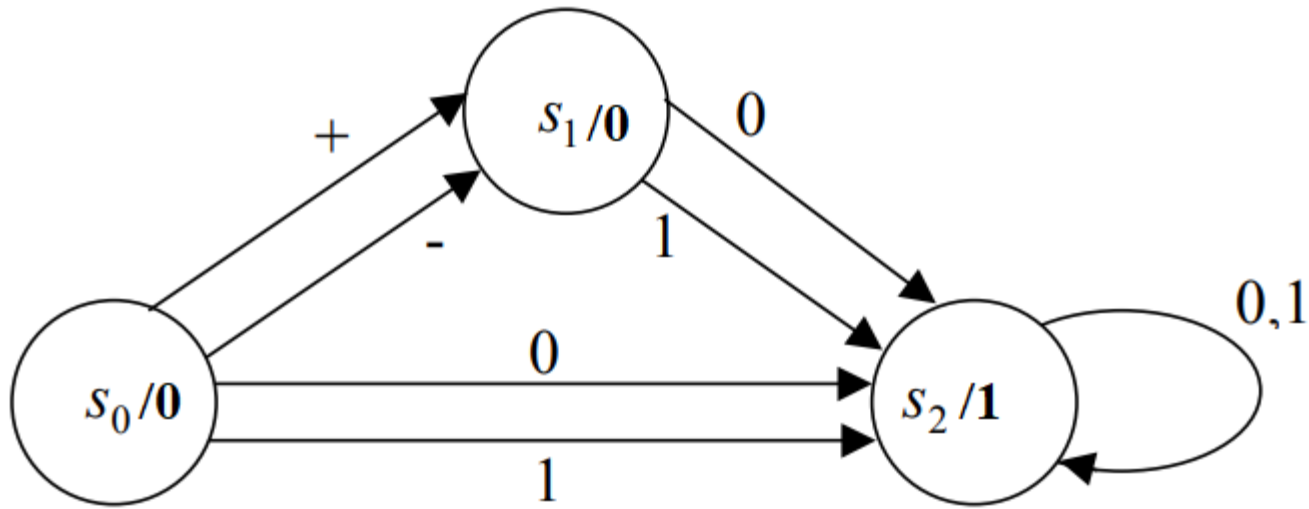
- $B=(X,S,Y,q_0,P,R)$
- $X=\{x\}$ – finite set of input symbols
- $S=\{s\}$ – finite set of internal states
- $Y=\{y\}$ – finite set of output symbols
- q_0 – initial state
- $P: X \times S \rightarrow S$ – transition function
- $R: S \rightarrow Y$ – output function

Example of Moore machine

Recognize binary integer numbers with a sign:

- a sequence of numbers from the set $\{0,1\}$ possibly starting with a sign $\{-,+\}$
- output symbols: 0 – indefinite, 1 – correct, 2 – error
- *Examples of correct numbers : 0101, -110, +10110*

State diagram of Moore machine

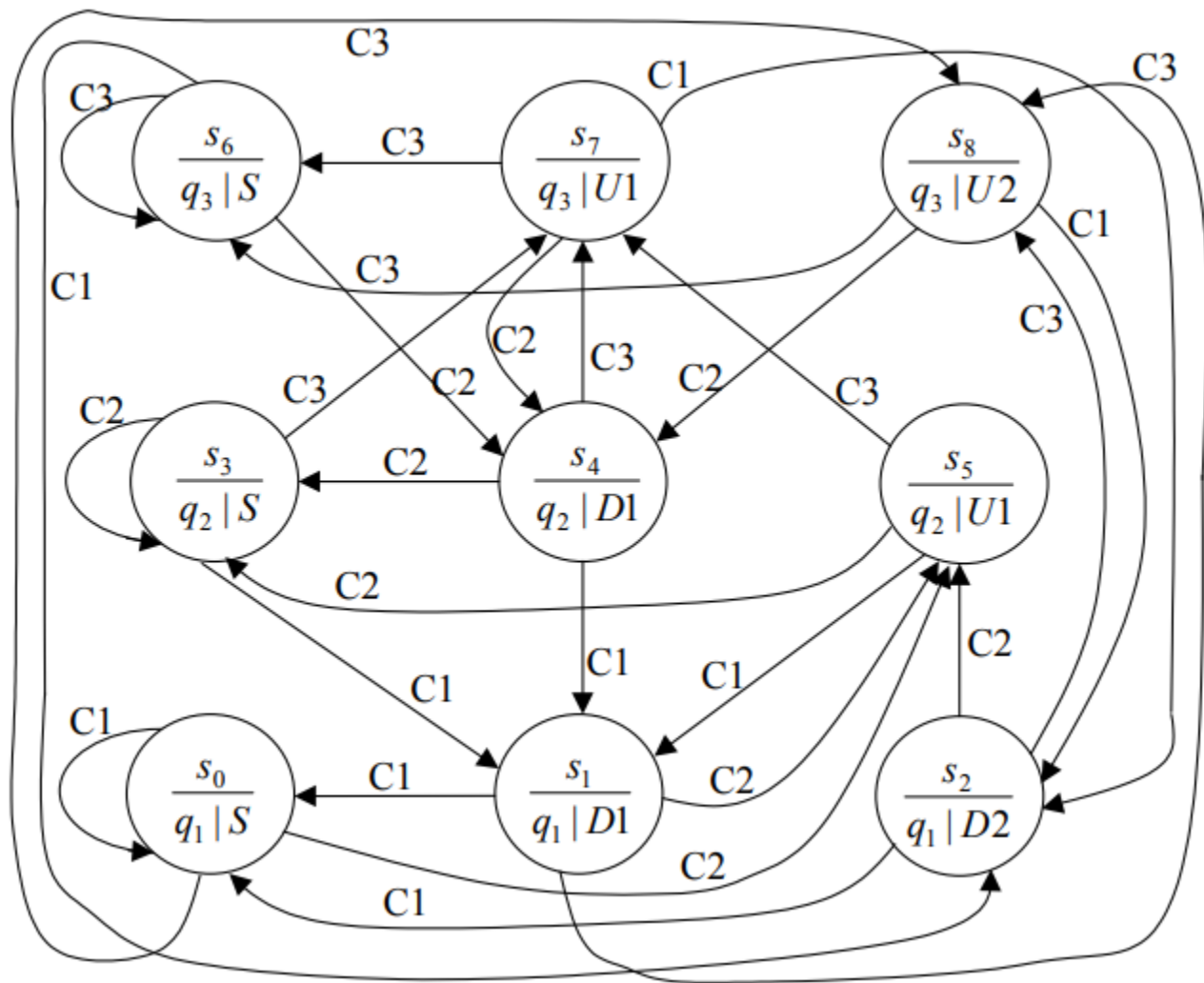


Concepts of Mealy and Moore machines are equivalent

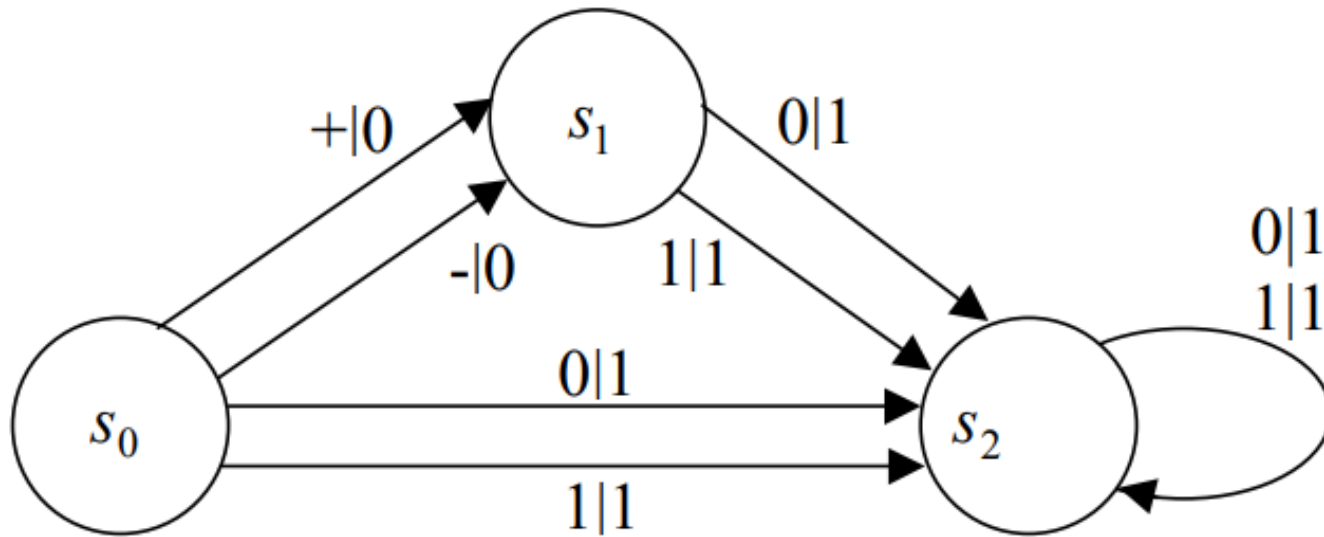
Mutual transformations

- From Moore to Mealy: one-to-one correspondence of state; output symbols move to the incoming arcs
- From Mealy to Moore: states of Moore automaton corresponds to the combinations of state and output symbol of Mealy automaton

Moore machine for lift control



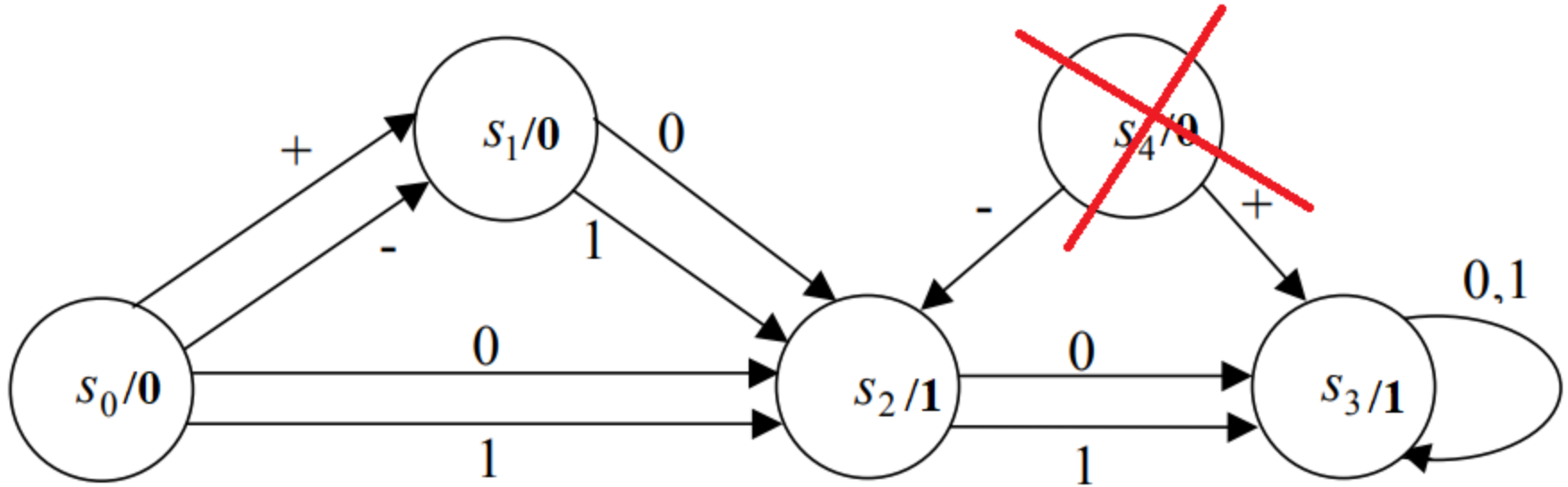
Mealy machine for recognition of binary integer numbers with sign



Minimization of finite automata

- I. Remove states unreachable from the initial state
- II. Replace a set of equivalent states by a single state

Removing unreachable states



State s_4 is unreachable

Equivalence of states

- Two states of the automaton s_1 and s_2 are called **n -equivalent** if, for an arbitrary input chain σ of length n , the output chains of symbols coincide.
- Two states of the automaton s_1 and s_2 are called **equivalent** if they are n equivalent for any integer n .

Central minimization theorem

- **Theorem.** In a finite automaton with m states, two arbitrary states are equivalent if and only if they are $(m-2)$ -equivalent.
- Idea of sufficiency proof: sequential composition of 0-equivalence, 1-equivalence, ... classes of states. Each next equivalence splitting contains at least one more class (if it does not coincide with the previous one). There are m states.

Minimization algorithm

- Build sequentially classes of i -equivalent states starting from 0 -equivalence
- The process stops for a definite $i \leq m-2$ when there is no splitting of the i -equivalence classes
- States of the minimal automaton corresponds to the equivalence classes
- Condense the transition function according to new states

Minimization example

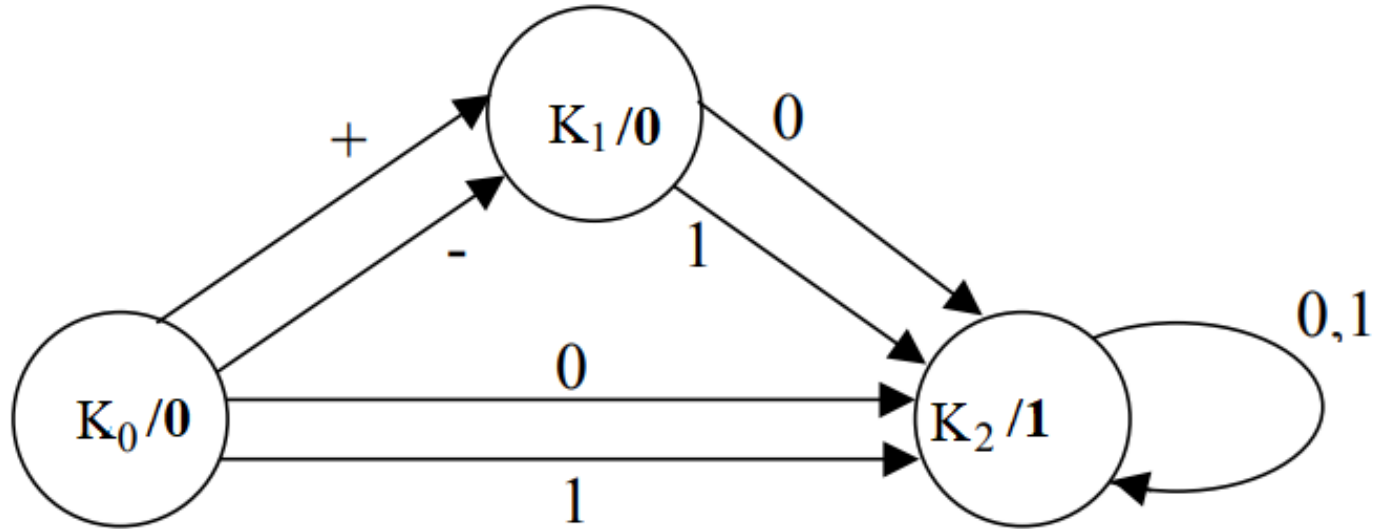
0-equivalence table

Class	State	+	-	0	1
K0	s_0	K0	K0	K1	K1
	s_1			K1	K1
K1	s_2			K1	K1
	s_3			K1	K1

1-equivalence table

Class	State	+	-	0	1
K0	s_0	K1	K1	K2	K2
K1	s_1			K2	K2
K2	s_2			K2	K2
	s_3			K2	K2

Minimal automaton



Equivalence of automata

- If it is necessary to find out whether the automata B_1 and B_2 are equivalent, it is enough to perform the minimization of the automaton $B_1 \cup B_2$ and determine whether their initial states fall into the same equivalence class

Specific synthesis of finite automata

- Specific synthesis uses the result of the abstract synthesis and minimization
- It takes into consideration specific requirements of the available hardware (logic gates and flip-flops)
- It can be implemented either in hard-wired form or in the form to upload into a generic device such as PLM or FPGA