

# Rete circuitale di automi cellulari

## Presentazione per il corso di Elettronica Digitale

Rodolfo Rocco

[rocco.1494012@studenti.uniroma1.it](mailto:rocco.1494012@studenti.uniroma1.it)

18 gennaio 2014



SAPIENZA  
UNIVERSITÀ DI ROMA

# Indice

---

1 Introduzione agli automi cellulari

2 Il circuito del singolo automa

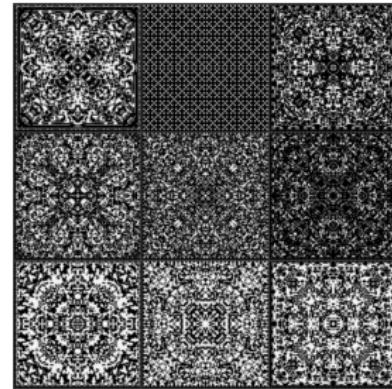
3 Una rete di automi cellulari

4 Conclusioni

Gli automi cellulari sono particolari macchine a stati, introdotte da J. von Neumann, il cui funzionamento dipende dal loro stato al clock precedente e da quello degli automi circostanti (primi vicini). Tutti gli automi sono disposti su un reticolo.



J. von Neumann

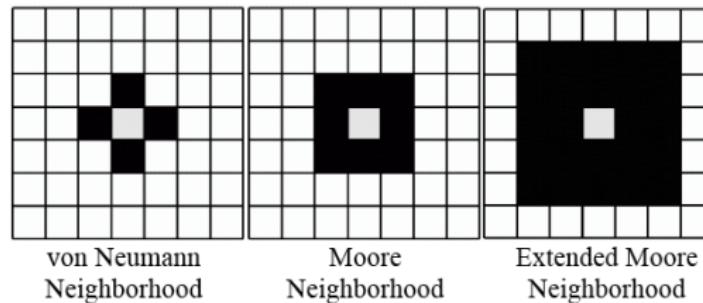


9 istantanee dell'evoluzione di altrettante reti di automi

# Le caratteristiche di un automa cellulare

E' possibile classificare gli automi cellulari in base a:

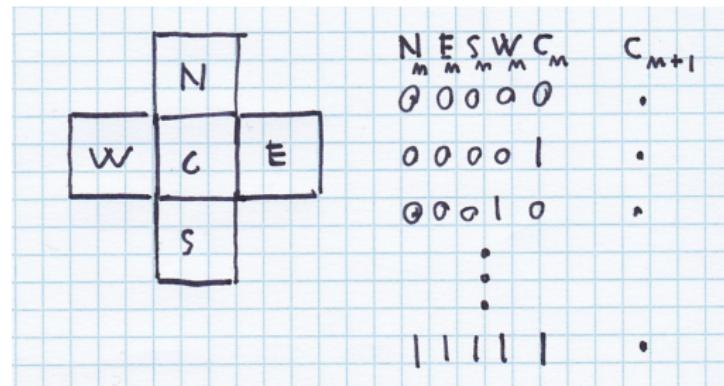
- **dimensioni:** 1D, 2D, 3D etc.
- **# stati:** 2 stati (vivo/morto), 3 stati (RGB), 4 stati etc.
- **primi vicini:** alla Von Neumann o alla Moore



Nel nostro caso abbiamo un automa cellulare **bidimensionale** a **due stati**. I **primi vicini** sono quelli di **VN** e l'automa è **programmabile**, ovverosia e' possibile specificare la regola che ne determina il funzionamento.

# Le regole

Abbiamo **4 PV**. Ogni automa ha **due stati** (acceso/spento), pertanto  $2^4$  combinazioni di primi vicini. Inoltre a seconda che l'automa che consideriamo sia acceso o spento le regole possono cambiare, pertanto in totale  $2 * 2^4$  configurazioni, esprimibili mediante **una parola a 32 bit** o **due parole a 16 bit**.



Possibile concezione per la regola

# L'automa come macchina a stati

L'automa cellulare è una particolare macchina a stati. Quello da noi considerato ha solo due configurazioni più un terzo stato di idle nel quale la macchina entra fintanto che è alzato il reset.

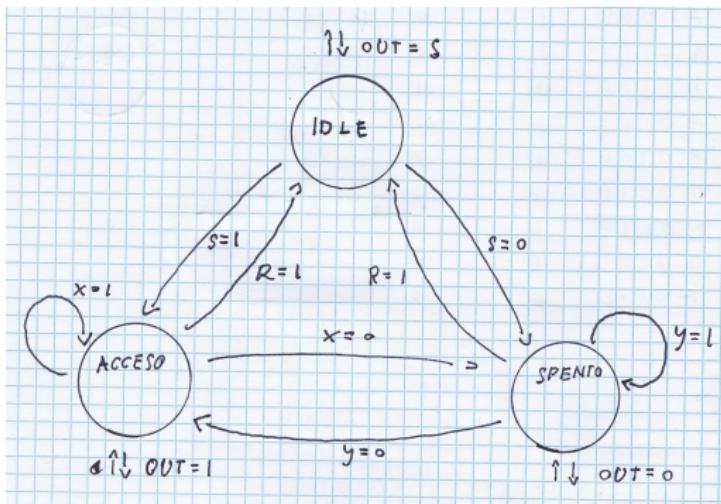


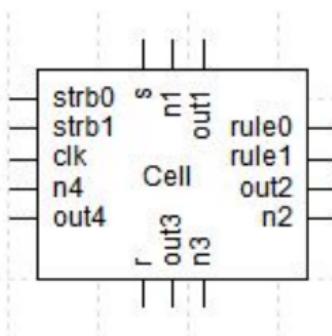
Diagramma della macchina a stati alla base dell'automa

**Di cosa abbiamo bisogno:**

- 4 ingressi per i PV
- 1 ingresso per la condizione iniziale S
- 1 ingresso per il reset R
- 1 uscita per lo stato dell'automa OUT
- 1 ingresso seriale per immettere la regola
- clock

# Il circuito del singolo automa

Come appare esternamente...



- **n1,n2,n3,n4:** PV
- **s:** set dello stato iniziale
- **r:** reset della macchina
- **rule0, rule1:** ingressi seriali per le due parole a 16 bit che costituiscono la regola
- **clk:** clock per i flip-flop
- **strb0, strb1:** clock per gli SR che ricevono rule0, rule1
- **out1,out2,out3,out4:** uscita dello stato della macchina

## Alcune considerazioni

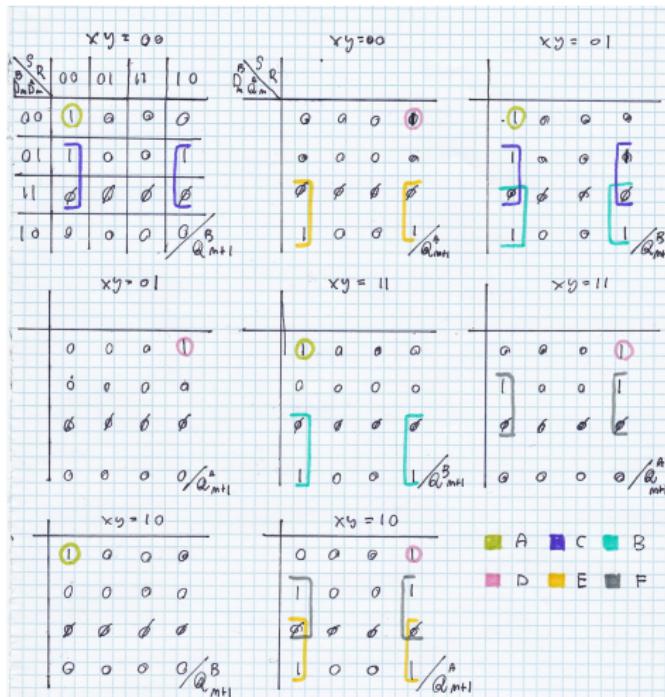
Abbiamo 4 pin **out** ma sono tutti lo stesso segnale; abbiamo **strb0** e **strb1** (oltre a **clk**) in quanto non vogliamo che i bit della regola shiftino di posizione (gli SR sono dunque clockati dalle tastiere usate per immettere le due parole)

A partire dalle precedenti specifiche, usando flip-flop D, abbiamo la seguente tavola delle transizioni.

$D_m^B$	$D_m^A$	S	R	X	Y	$Q_{m+1}^B$	$Q_{m+1}^A$	
IDLE 0	0	0	0	∅	∅	1	0	SPENTO
	1	0	0	∅	∅	0	1	ACCESO
	∅	1	0	∅	∅	0	0	IDLE
ACCESO 1	0	∅	0	0	∅	1	0	SPENTO
	∅	0	1	∅	∅	0	1	ACCESO
	∅	1	∅	∅	∅	0	0	IDLE
SPENTO 1	0	∅	0	∅	0	0	1	ACCESO
	∅	0	∅	1	1	1	0	SPENTO
	∅	1	∅	∅	∅	1	1	IDLE

Tavola delle transizioni avendo deciso di usare flip-flop di tipo D

# Dalla tavola delle transizioni passiamo alle mappe di Karnaugh...

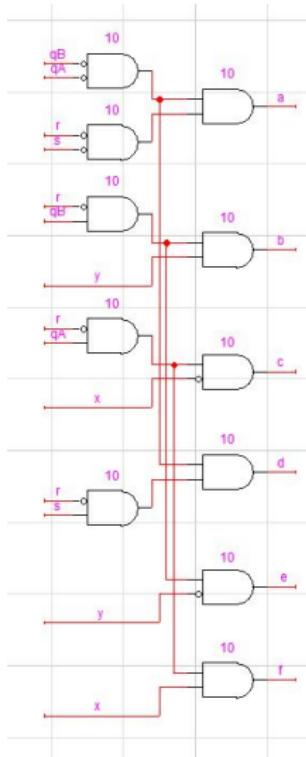


Mappe di Karnaugh

...e infine alla rete logica combinatoria

$$Q_{m+1}^B = \overline{D_m^B} \overline{D_m^A} \overline{S} \overline{R} + D_m^A \overline{R} \overline{x} + D_m^B \overline{R} y$$

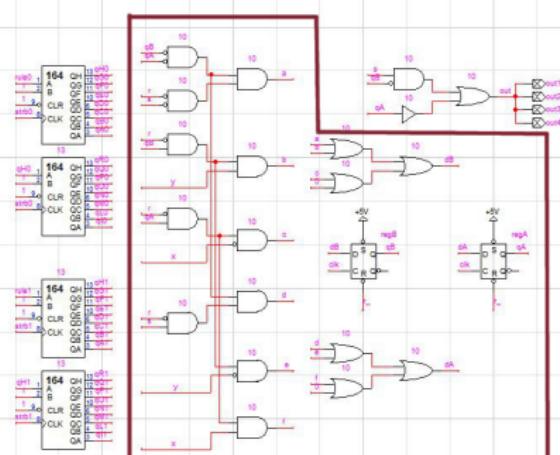
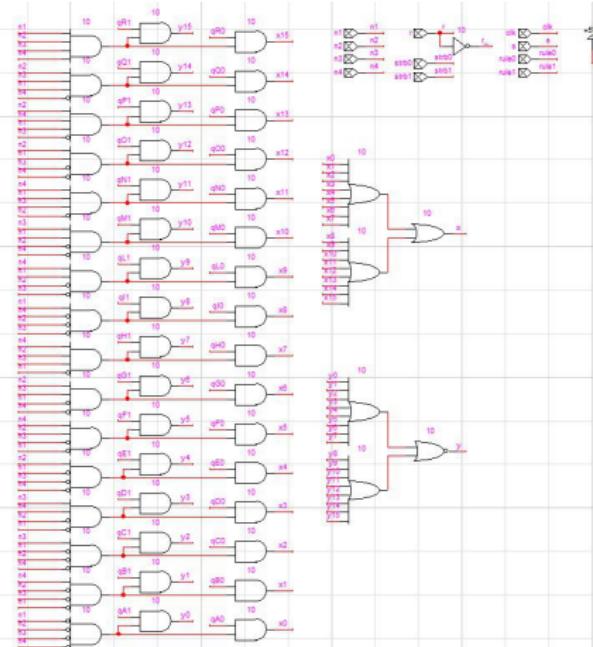
$$Q_{m+1}^A = \overline{D_m^B} \overline{D_m^A} S \overline{R} + D_m^B \overline{R} \overline{y} + D_m^A \overline{R} X$$



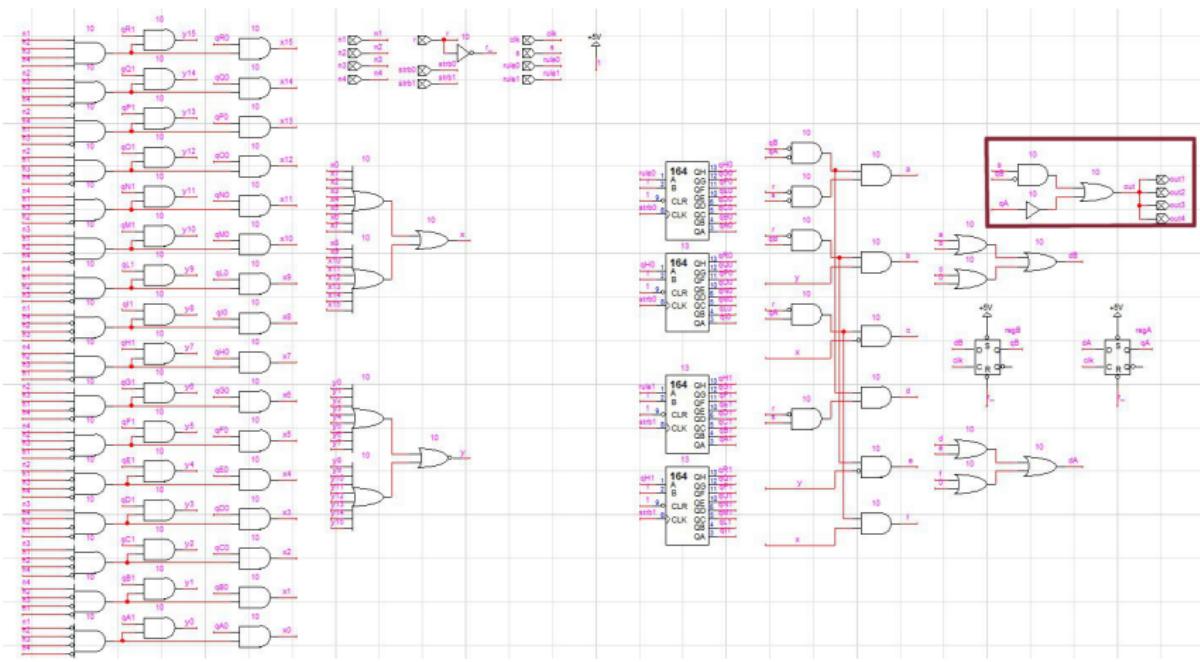
La RLC come appare  
in LogicWorks (meno  
gli OR)

# Il circuito del singolo automa

...come appare internamente

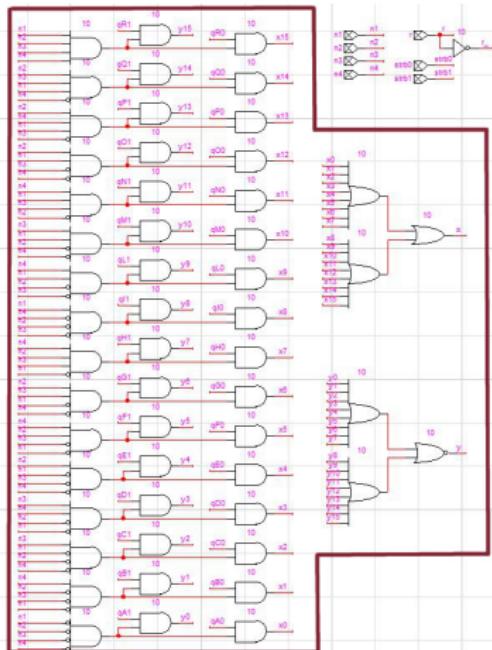


Flip-flop e RLC in ingresso



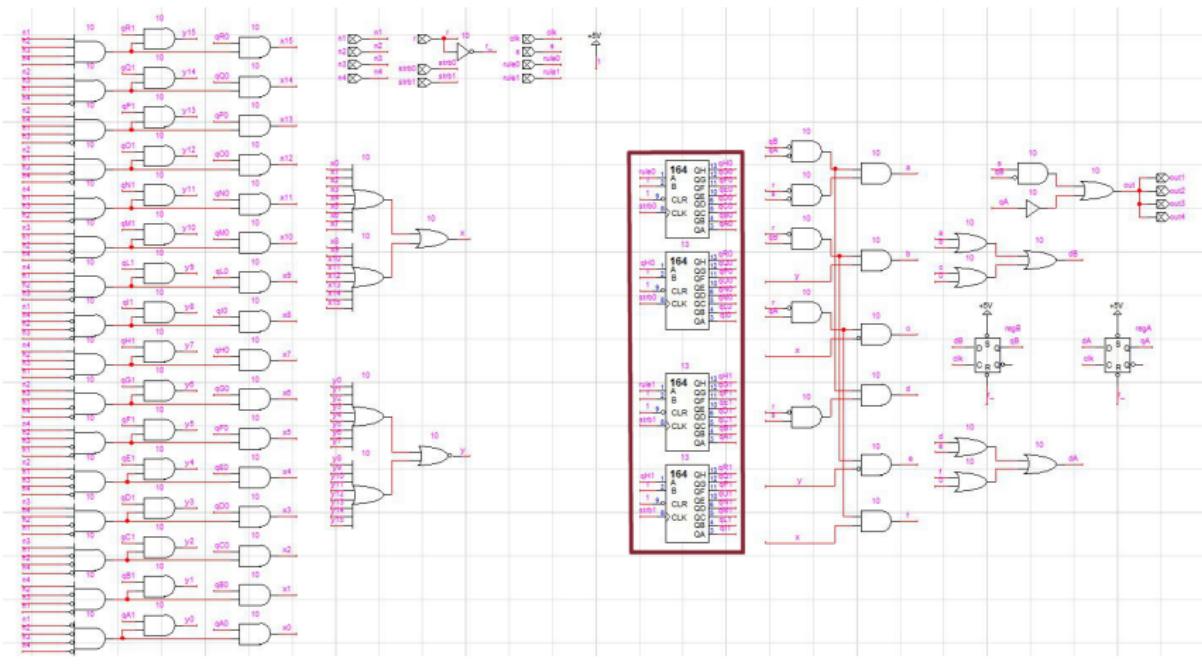
RLC in uscita ai flip-flop

La RLC in uscita serve a generare il segnale OUT. In particolare nello stato di idle esso è pari al segnale di set, S.



Generazione delle variabili interne, X e Y

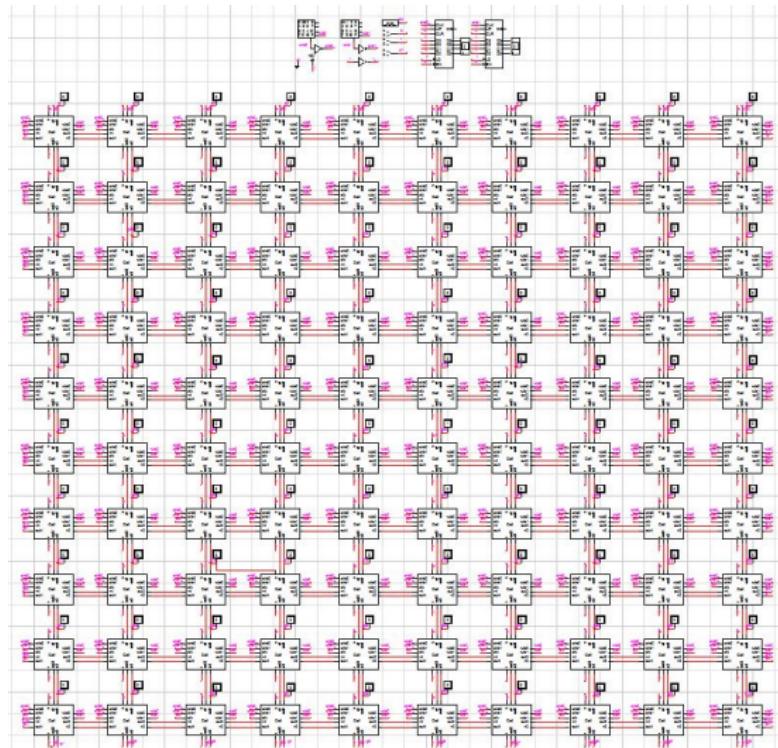
Per generare le variabili interne eseguiamo i prodotti fondamentali tra le possibili configurazioni di primi vicini e i corrispettivi bit della regola. Infine facciamo la somma dei prodotti.



SR per inserimento seriale della regola

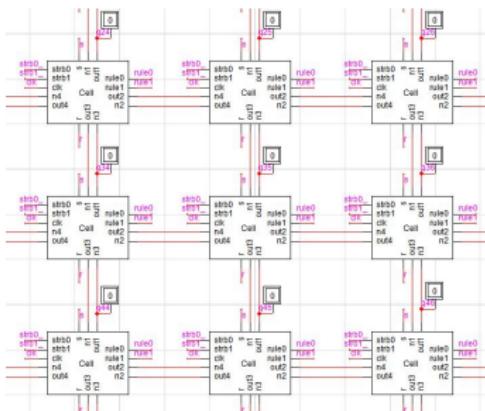
4 SR da 8 bit per due parole a 16 bit, ciascuna delle quali rappresenta la regola per uno dei due stati della macchina.

# Vista globale...

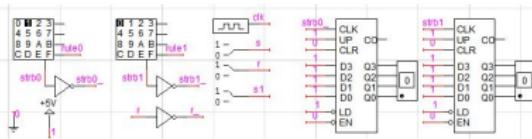


# Le caratteristiche di questa rete

- Griglia **quadrata** di **lato 10**, pertanto 100 automi in totale
- **Circuteria ancillare** per inserimento della regola
- Condizioni al contorno **toroidali**
- Condizioni iniziali tali per cui tutti gli automi sono inizialmente spenti, ad eccezione di uno (**seed**)



Zoom sulla rete



Circuteria ancillare

Circuteria consistente di tastiere, contatori (per tener conto del bit che si sta inserendo) e switch per il set e reset.

# Verifica del funzionamento della rete

---

Per verificare il funzionamento si sceglie una regola, la si immette, sia nel circuito che nel simulatore software di terze parti, e infine si verifica che le evoluzioni coincidano. Più precisamente...

- ① Si **esportano i timing** dei segnali OUT di ciascuno dei 100 automi in un file di testo
- ② Si **formatta il file** di testo tramite apposito programma in C, in modo che gnuplot lo possa leggere
- ③ A partire dal file (dunque dai timing) **si genera un'animazione** dell'evoluzione degli stati dei 100 automi (essenzialmente una finestra di 100 pixel che si accendono e si spengono...)
- ④ **Si confronta questa animazione** con quanto ci mostra il software...

# Conclusioni

In questo lavoro abbiamo:

- Disegnato un **automa cellulare** a 2 stati con PV alla von Neumann
- Implementato una **rete di 100 automi programmabili**
- Scritto un programma in C per la **interpretazione dei timing** e la **generazione delle animazioni** della rete
- Verificato il **funzionamento della rete** tramite confronto delle animazioni con simulazione software

## Possibili sviluppi...

Implementazione del **costruttore universale di von Neumann**, un automa cellulare con 29 stati in grado di **evolversi e replicarsi**