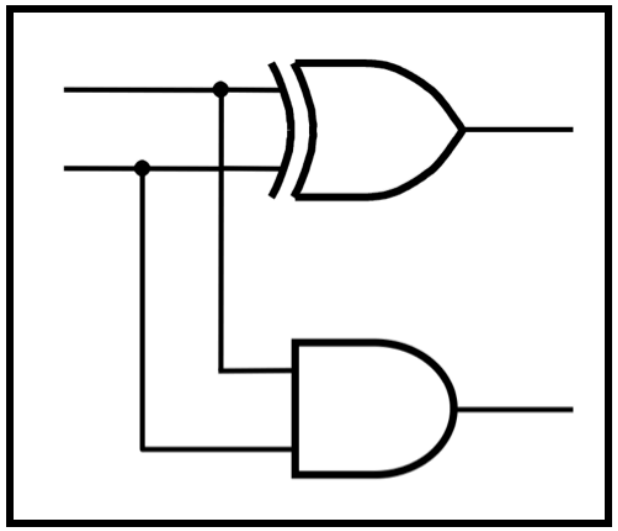


CprE 281: Digital Logic

Instructor: Alexander Stoytchev

http://www.ece.iastate.edu/~alexs/classes/

FSM as an Arbiter Circuit

CprE 281: Digital Logic
Iowa State University, Ames, IA

Administrative Stuff

- **Homework 11 is out**
- **It is due on Monday Nov 26 @ 4pm**

Administrative Stuff

- **Homework 12 is out**
- **It is due on Monday Dec 3 @ 4pm**

Administrative Stuff

- **Final Project (7% of your grade).**
- **By now you should have selected a project.**
- **Read the instructions for the project carefully.**
- **Also, posted on the class web page (Labs section).**
- **This is your lab for the last two weeks.**
- **This is due during your last lab (dead week).**

Arbiter Circuit

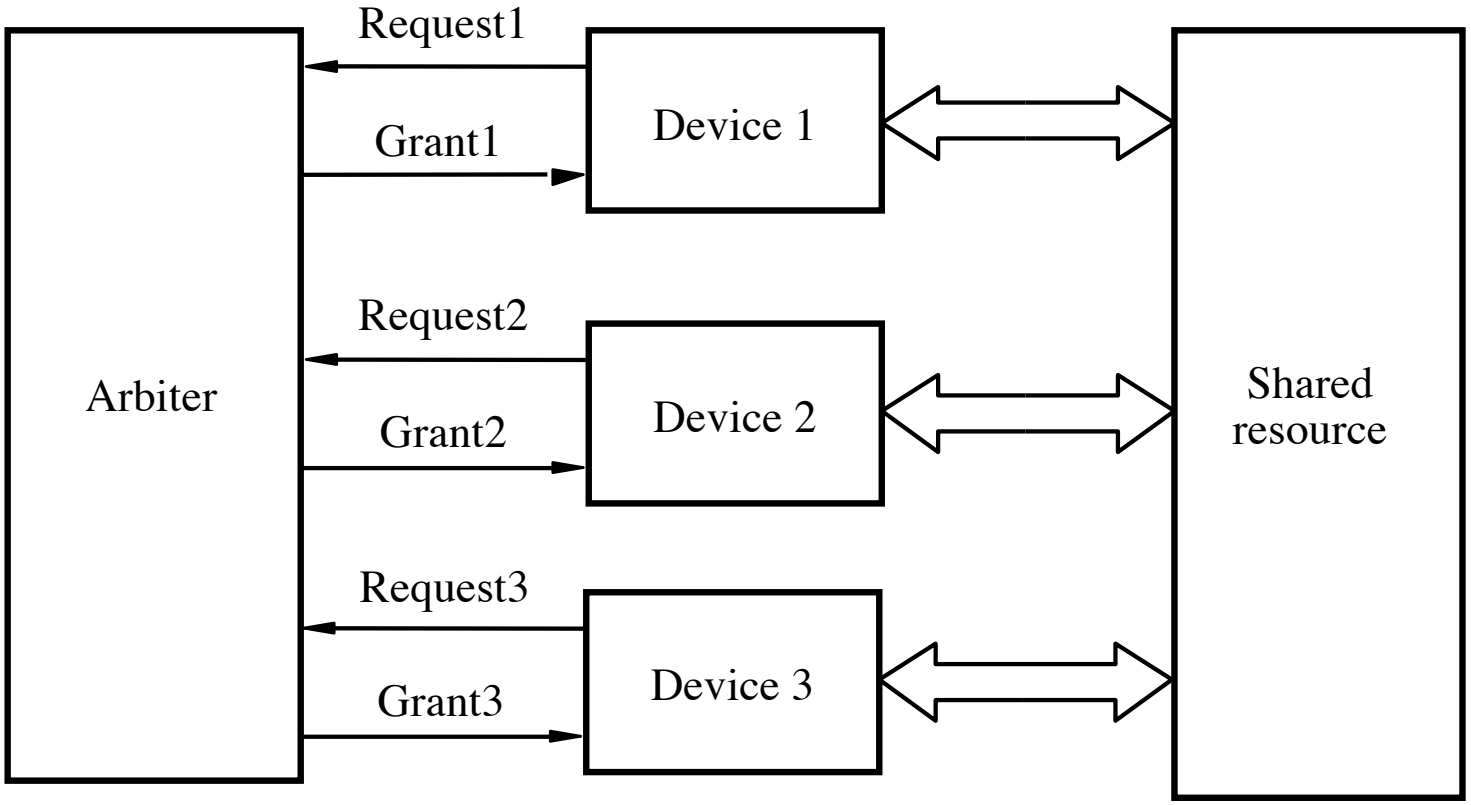
Goal

- **Design a machine that controls access by several devices to a shared resource.**
- **The resource can be used by only one device at a time.**
- **Any changes can occur only on the positive edge of the clock signal.**
- **Each device provides one input to the FSM, which is called a request.**
- **The FSM produces one output for each device, which is called a grant.**

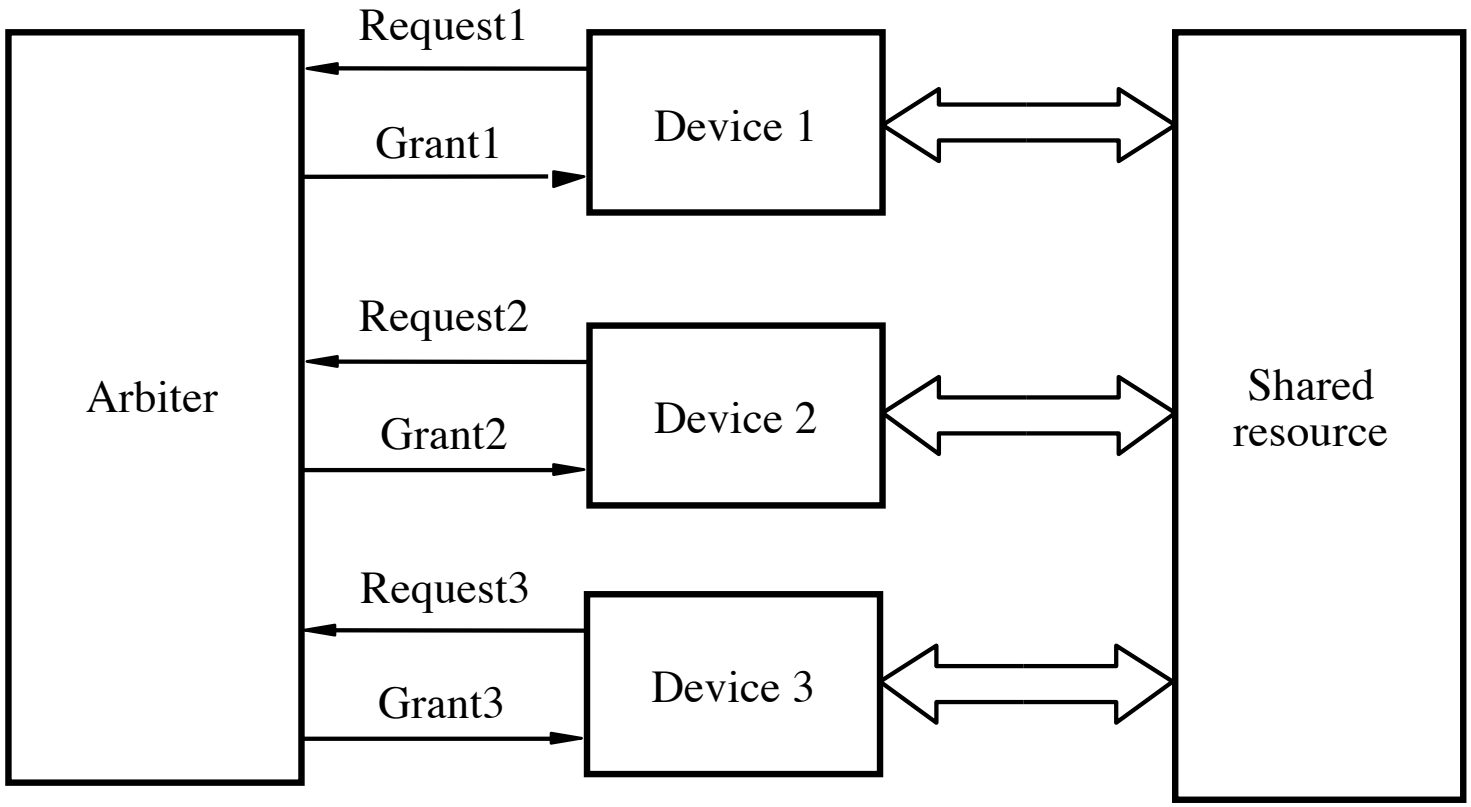
Goal

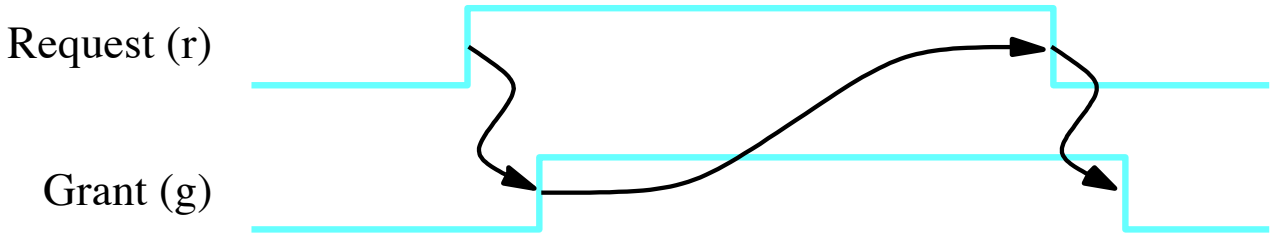
- **The requests from the devices are prioritized.**
- **If two requests are active at the same time, then only the device with the highest priority will be given access to the shared resource.**
- **After a device is done with the shared resource, it must make its request signal equal to 0.**
- **If there are no outstanding requests, then the FSM stays in an Idle state.**

Conceptual Diagram



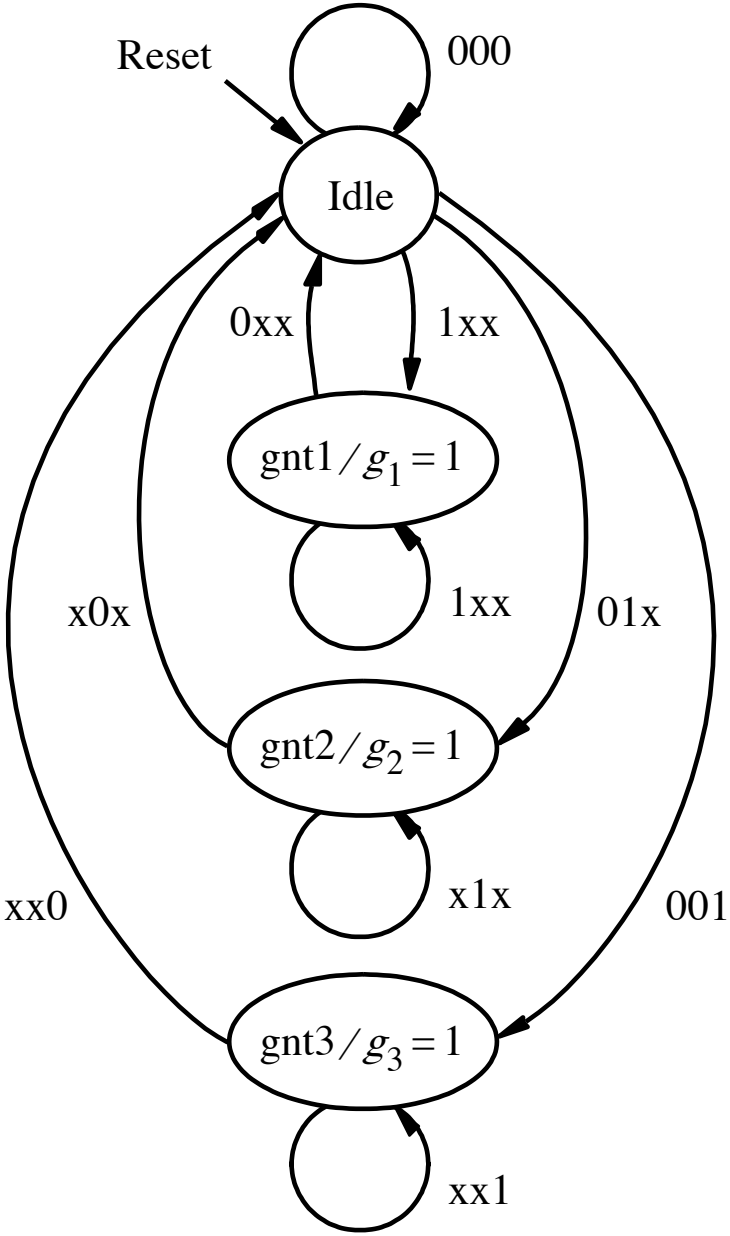
Conceptual Diagram





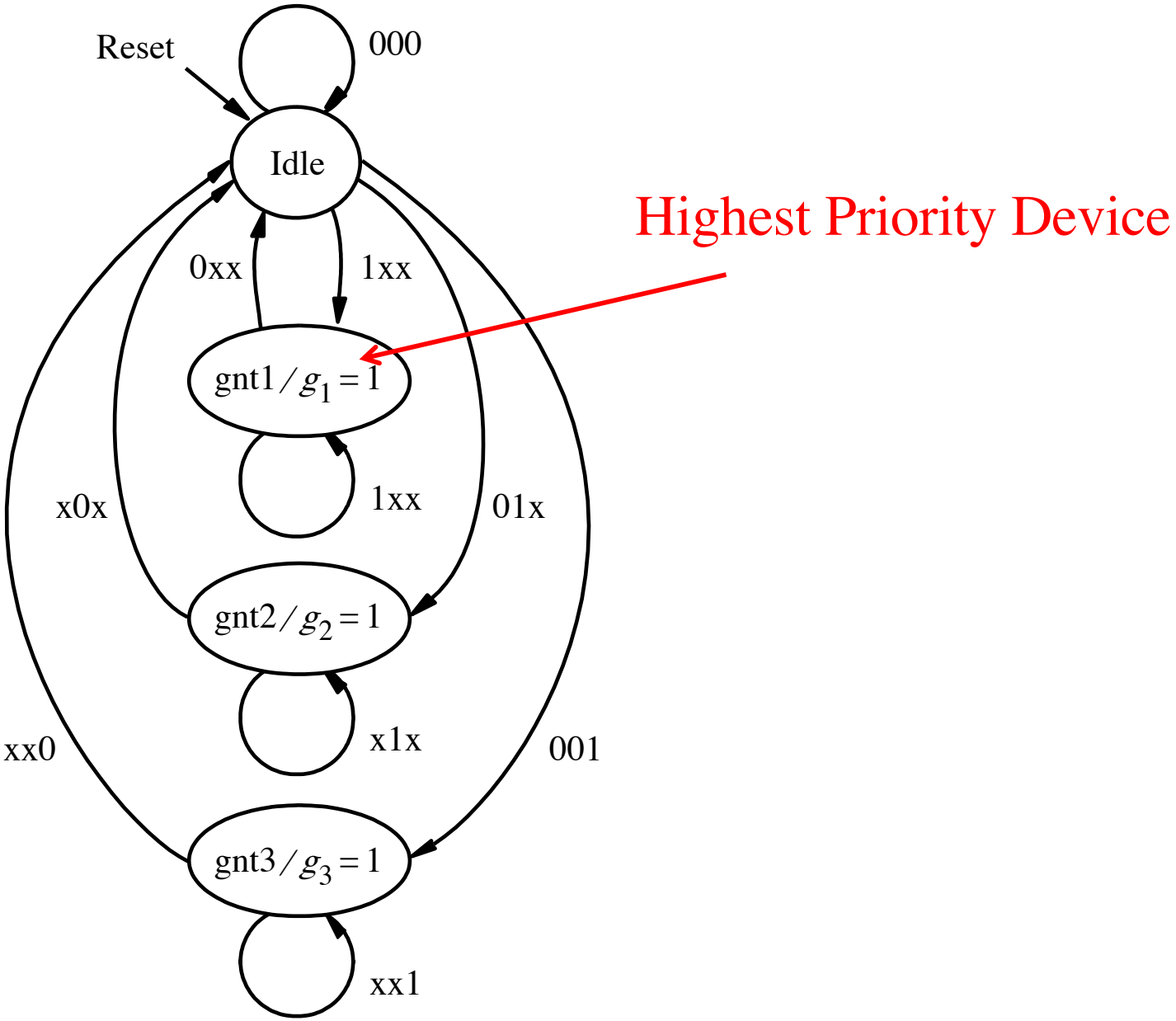
[Figure 9.20 from the textbook]

State diagram for the arbiter

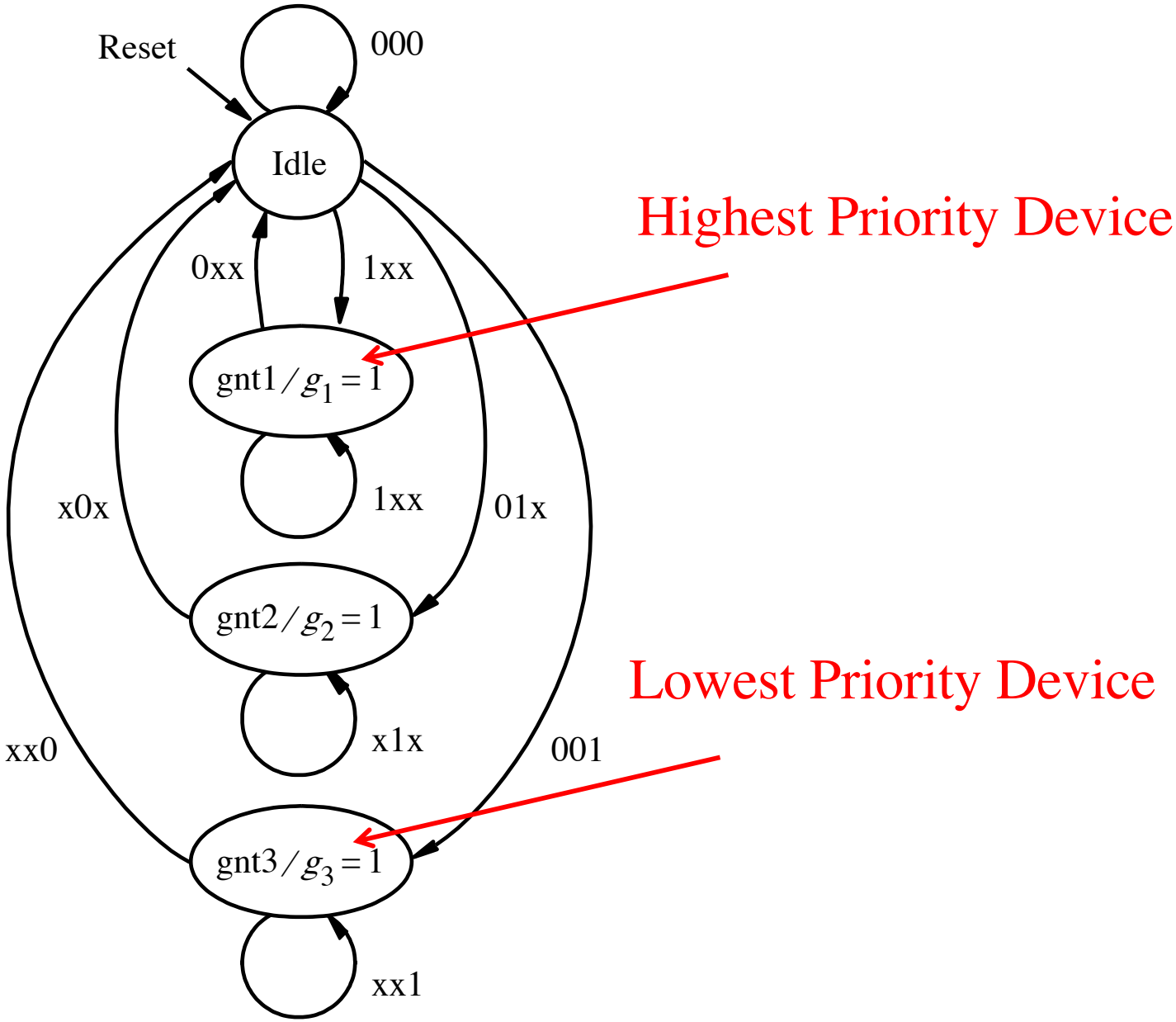


[Figure 6.72 from the textbook]

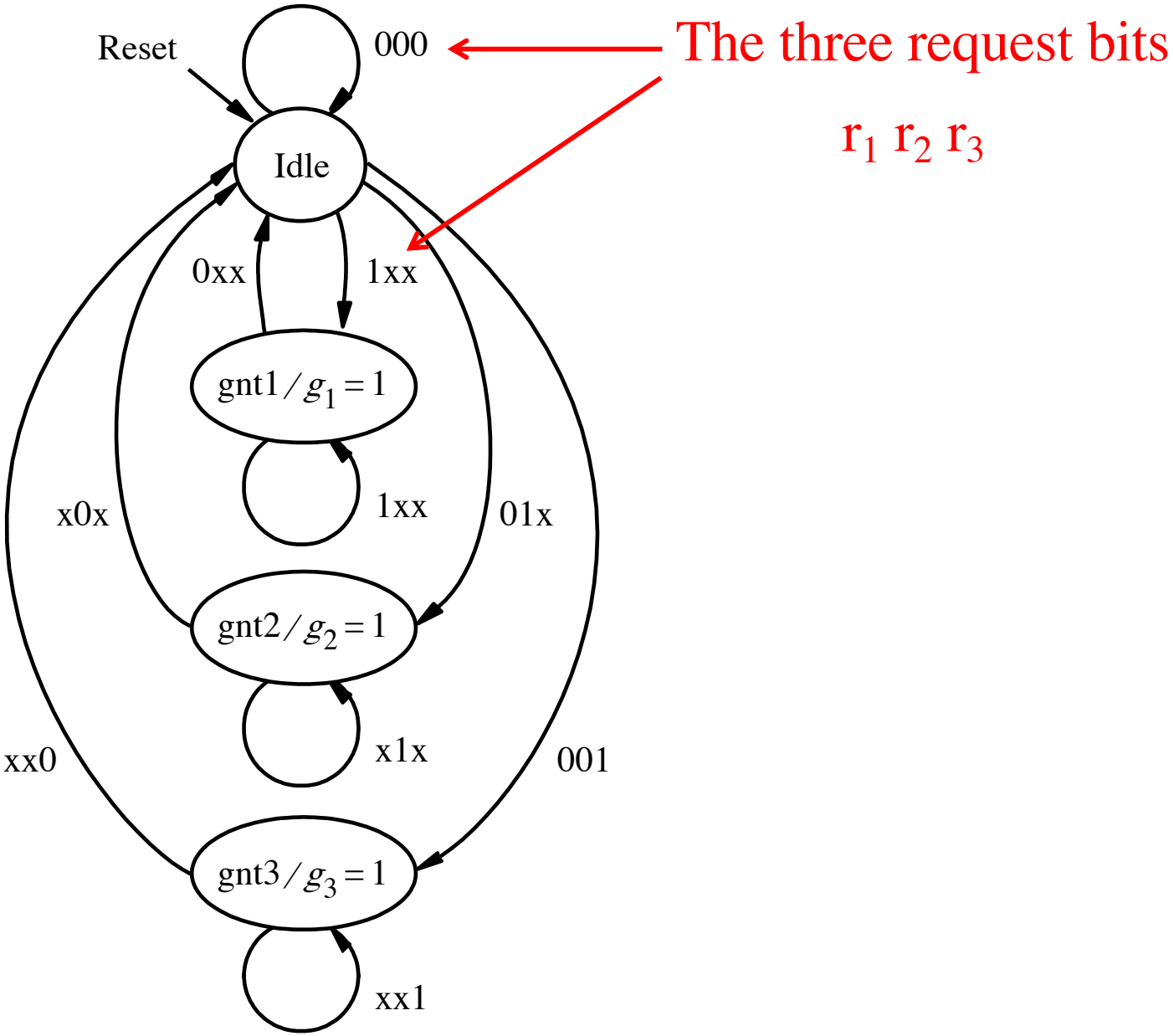
State diagram for the arbiter



State diagram for the arbiter

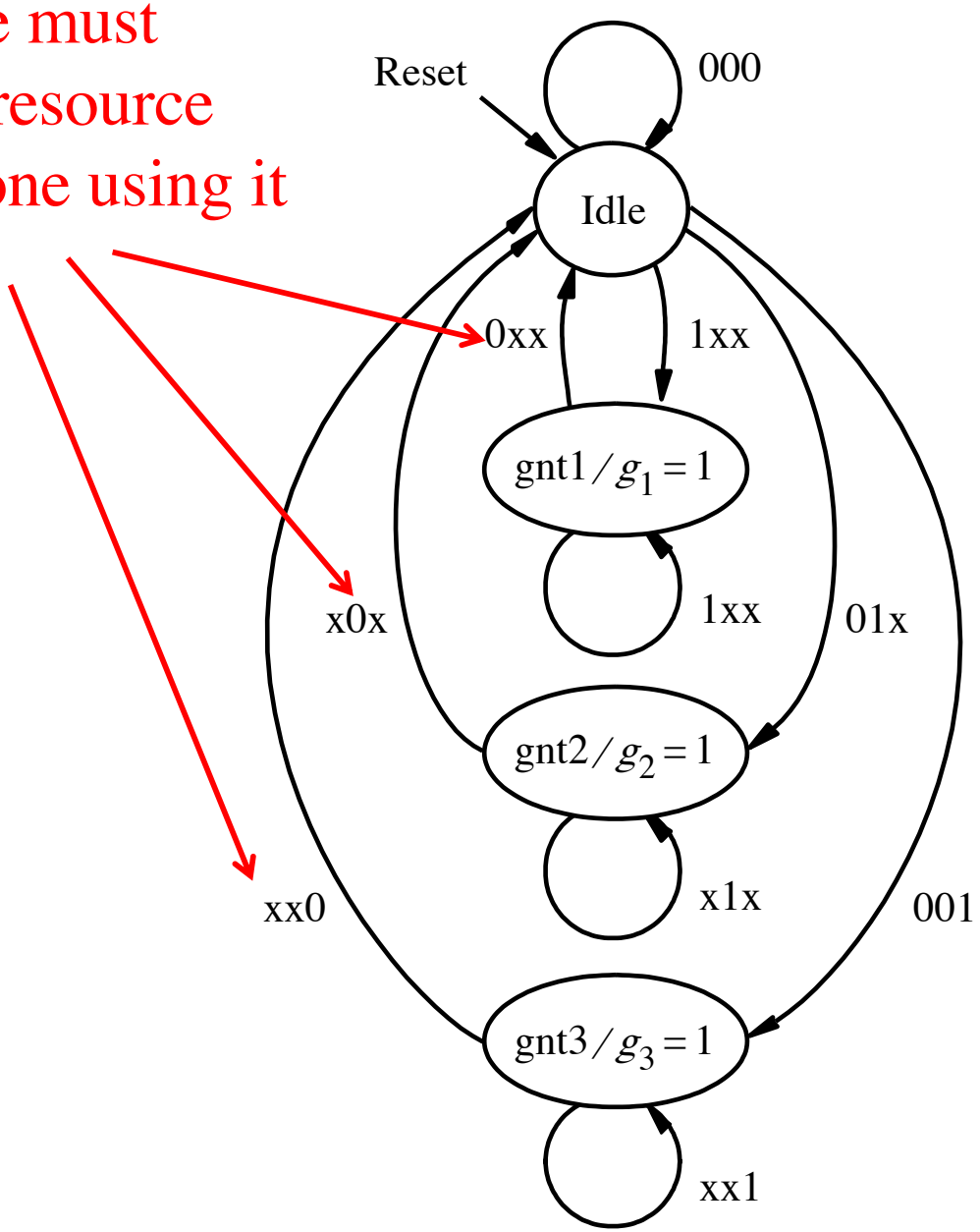


State diagram for the arbiter

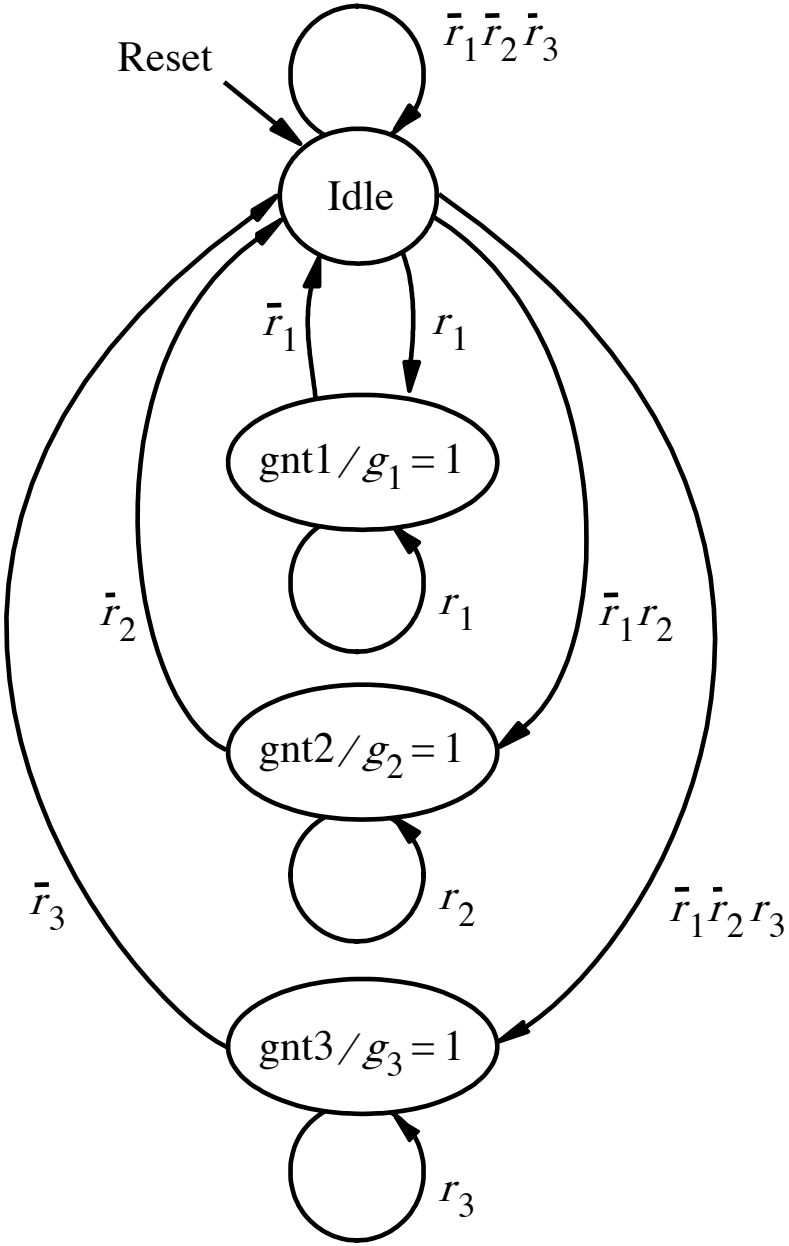


State diagram for the arbiter

Each device must release the resource after it is done using it



Alternative style of state diagram for the arbiter

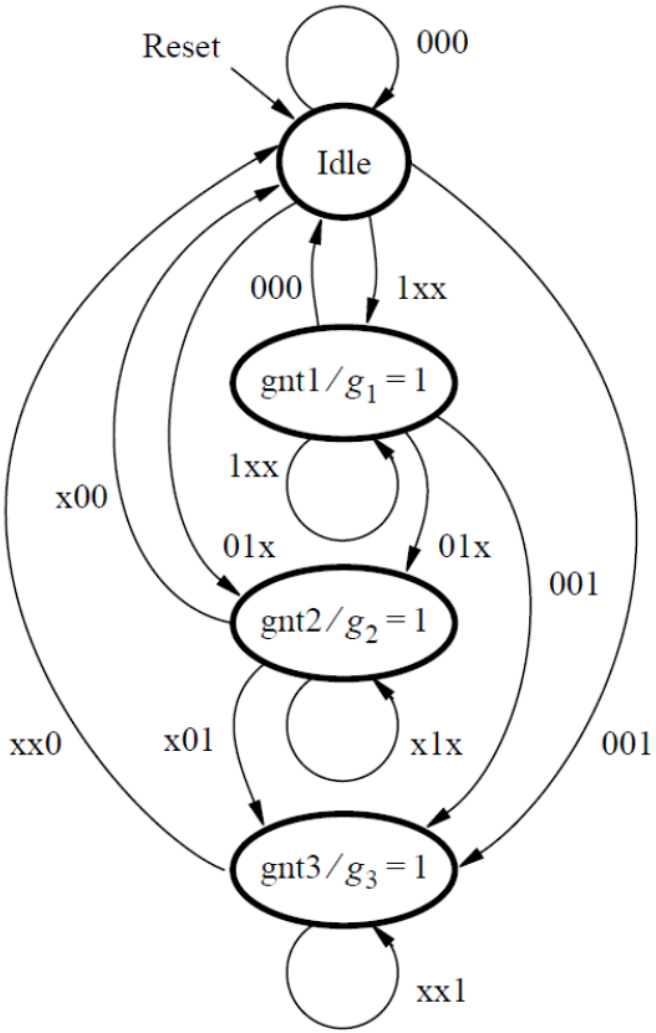


[Figure 6.73 from the textbook]

This design has one flaw:

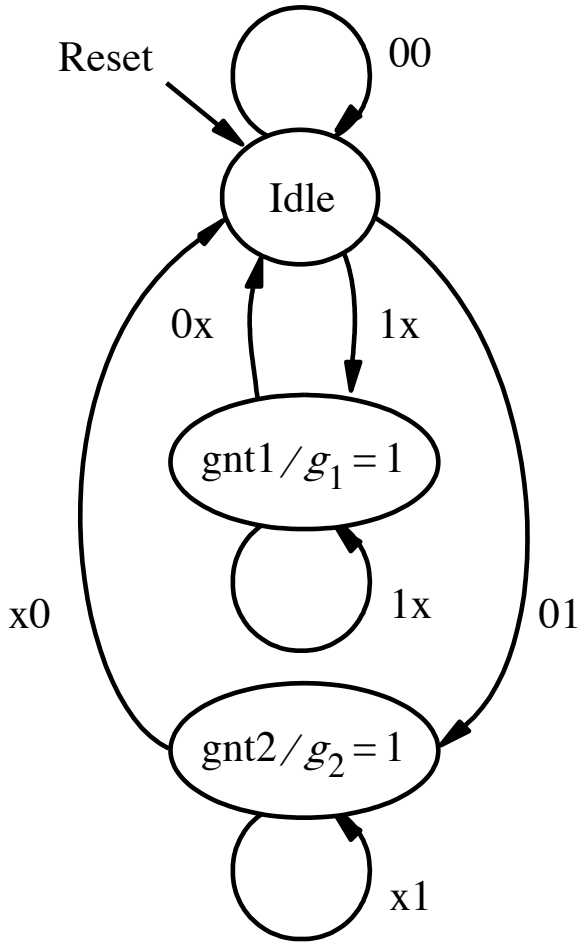
If device1 and device2 raise requests all the time, then device3 will never get serviced.

This state diagram solves this problem

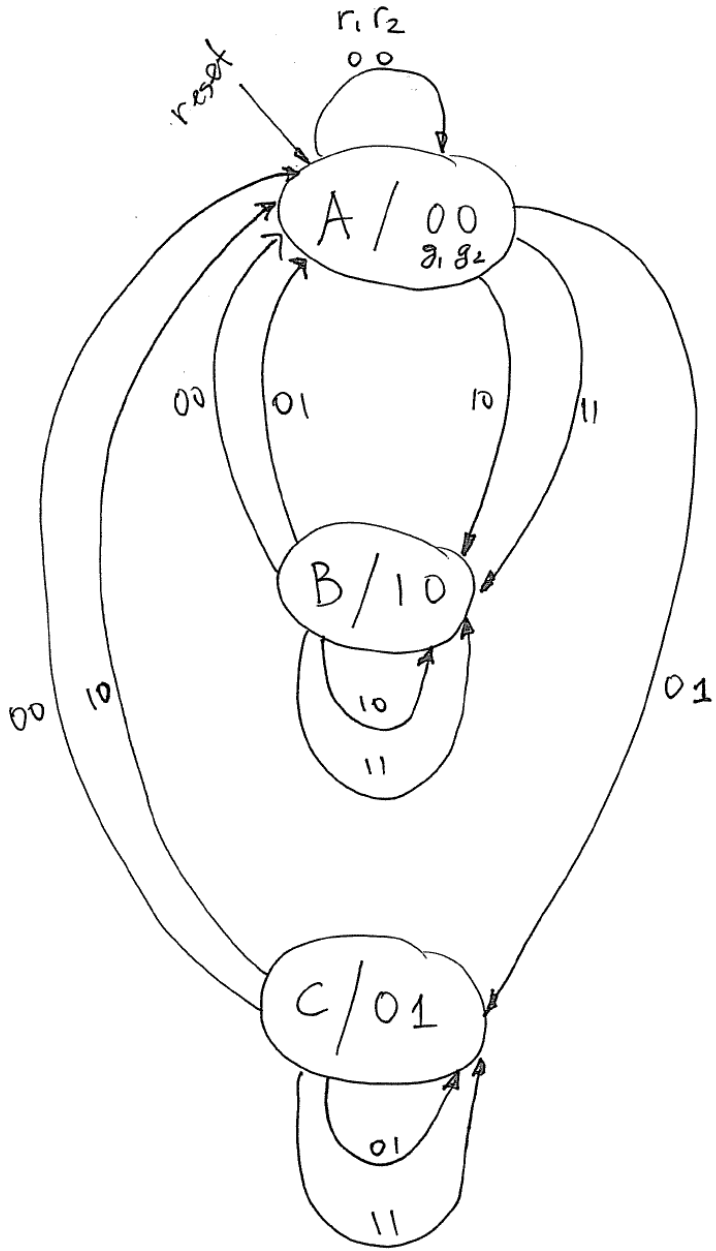


Let's look at a simpler example with only two devices that need to use the shared resource

State diagram for the simpler arbiter



State diagram for the arbiter circuit



State Table

	$r_1 r_2 =$ 00	01	10	11	Output
A	A	C	B	B	00
B	A	A	B	B	10
C	A	C	A	C	01

State-Assigned Table

		$r_1 r_2 = 00$	01	10	11	
	$Y_2 Y_1$	$Y_2 Y_1$	$Y_2 Y_1$	$Y_2 Y_1$	$Y_2 Y_1$	$g_1 g_2$
A	00	00	10	01	01	00
B	01	00	00	01	01	10
C	10	00	10	00	10	01
	11	dd	dd	dd	dd	dd

Output Expressions

Output expressions

$g_1 = Y_1$

$g_2 = Y_2$

Next State Expressions

Y_2

$r_1 r_2$ \ $Y_2 Y_1$	00	01	11	10
00	0	0	d	0
01	1	0	d	1
11	0	0	d	1
10	0	0	d	0

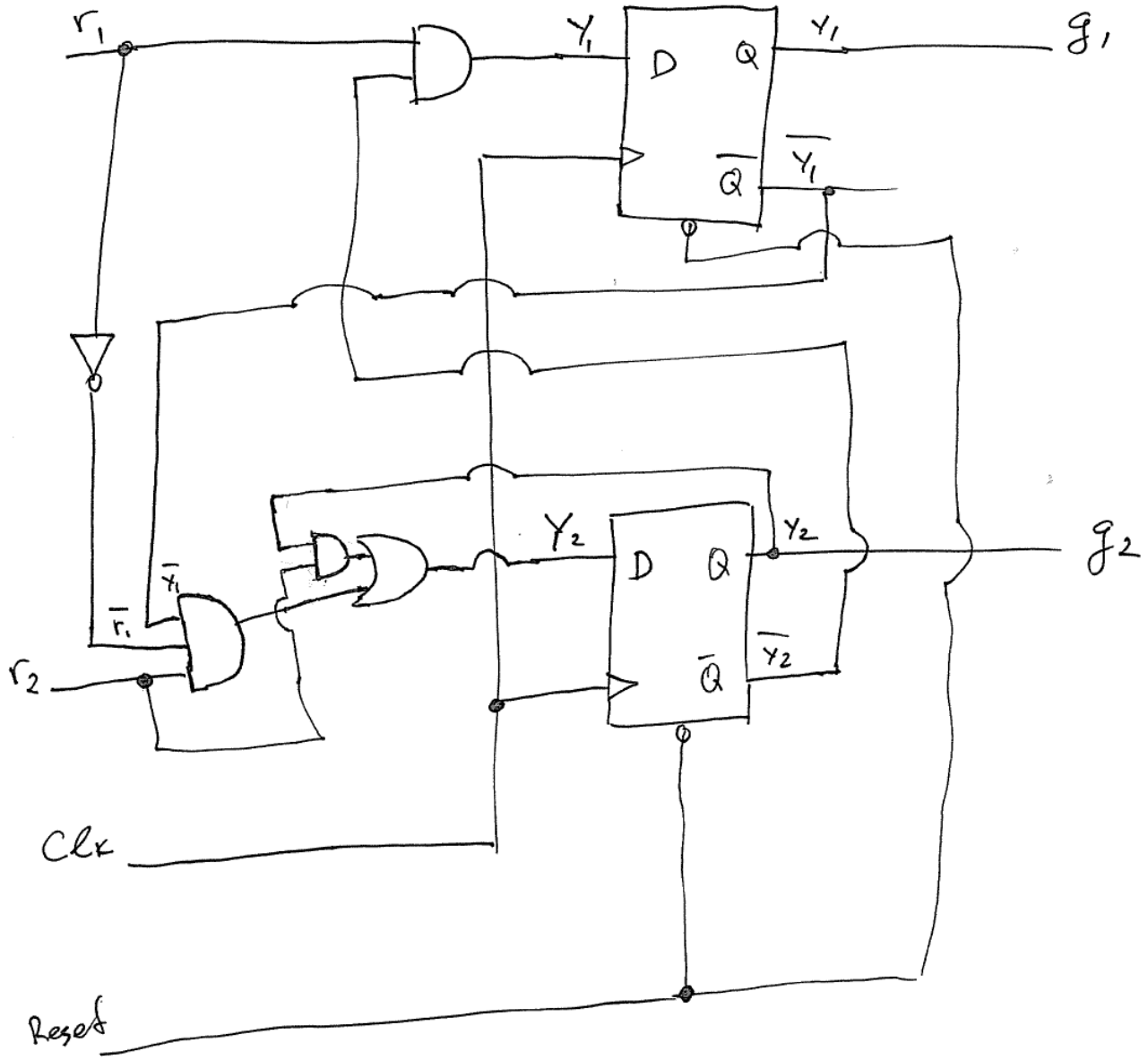
$$Y_2 = r_2 Y_2 + \overline{r_1} r_2 \overline{Y_1}$$

Y_1

$r_1 r_2$ \ $Y_2 Y_1$	00	01	11	10
00	0	0	d	0
01	0	0	d	0
11	1	1	d	0
10	1	1	d	0

$$Y_1 = r_1 \overline{Y_2}$$

Circuit Diagram



Questions?

THE END