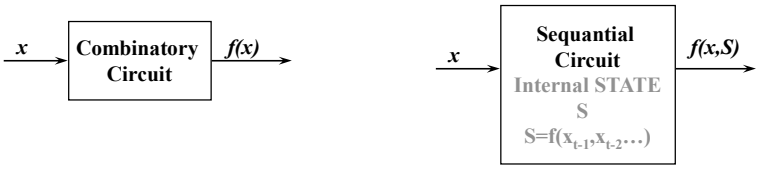
Sequential Circuits

NovaIMS, V.1.2 V.Lobo 2021

Sequential Circuits

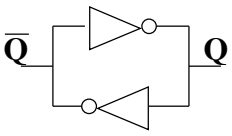
- **Combinatory Circuits**
  - The output (in each point in time) depend ONLY on the inputs present at that moment.
  - They have no memory
- **Sequential Circuits**
  - The outputs depend on the STATE of the circuit
  - The STATE of the circuit depends on the previous inputs
  - They have MEMORY

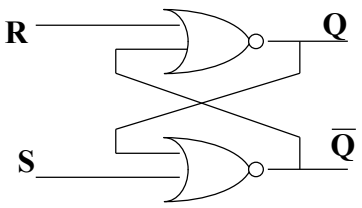


1

Basic memory element

- **How can we memorize a bit using only gates ?**
  - A pair of negations will maintain whatever state they have
  - It is not easy to change the state only with negations
- **S-R LATCH (Set-Reset)**
  - S (Set) input forces the output to 1
  - R (Reset) input forces the output to 0
  - 0





Can we describe a SR LATCH using truth tables ?

2

Description of sequential circuits

Sistemas Lógicos (3)

- **Truth tables**
  - **The PREVIOUS STATE is used as an input**
  - **We may simplify the truth table, specifying the output as a function of the previous state**

S	R	Q	$Q_{(t+1)}$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	x
1	1	1	x

S	R	$Q_{(t+1)}$
0	0	Q
0	1	0
1	0	1
1	1	x

Q \ S,R	00	01	11	10
0	0	0	x	1
1	1	0	x	1

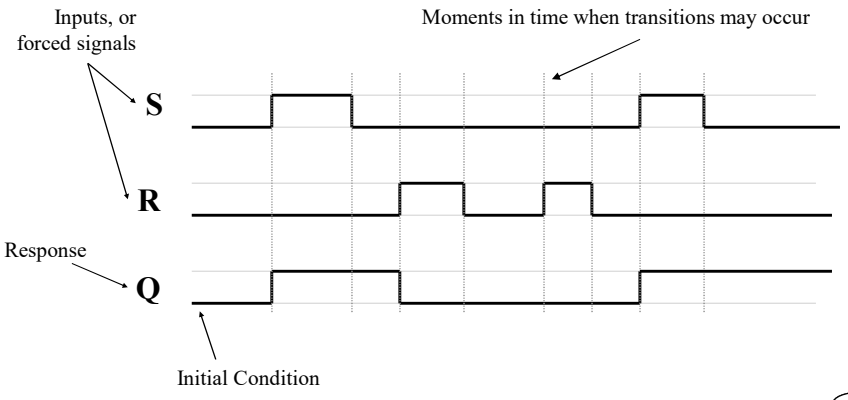
3

3

Description of sequential circuits

Sistemas Lógicos (3)

- **Temporal diagrams**
  - **Show the** *behaviour in time* **of the circuits when they receive a given input**
  - **They do not define completely the behaviour of a circuit ! (just the behaviour to a PARTICULAR input)**



4

4

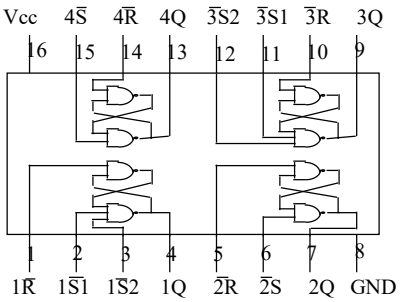
Descriptions of Sequential circuits

- **Asynchronous circuits**
  - **As soon as the inputs change, the outputs may change immediately**
  - **They can be very fast, but are not (yet) used much, because they are difficult to design and have problems with "critical runs"**
    → Promising future, as discussed in Proceedings of the IEEE, February 1999
- **Synchronous circuits**
  - **There is synchronizing signal (called CLOCK) that regulates when transitions may occur**
  - **There ONLY are transitions on the CLOCK EDGES**
  - **In temporal diagrams we only need to analyze what happens in clock edges.**
  - **The NEXT STATE is a function of what is present BEFORE the clock edge!**
  - **After a CLK edge, where changes may occur, there is a "relaxation time" where changes will take place internally, and after they all stabilize, a new clock edge may occur. This limits the maximum speed.**
  - **These are by far the most common circuits !**

5

SR Latches

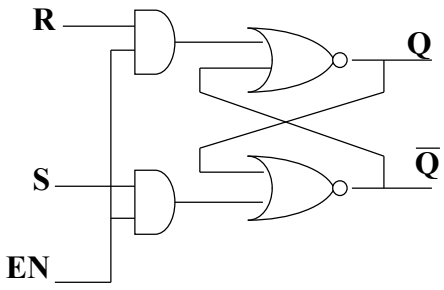
- **In the TTL family**
  - **74279**



74279

Nota: Todas as entradas são activas a 0

- **GATED LATCH**
  - **Have an ENABLE input**
  - **When ENABLE = 0, they keep the previous state. When ENABLE=1 it reacts to the R and S inputs.**

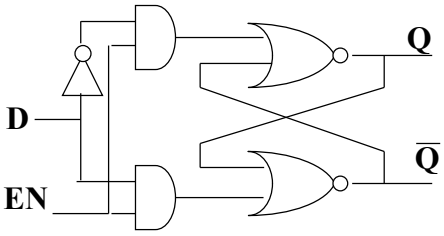


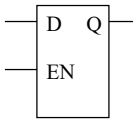
6

D type LATCH

- They have only one signal input D (from DELAY), and one enable signal.
- It is the basic MEMORY ELEMENT
  - The output is always available.
  - EN=1 ⇒ Store internally what is in the input
  - EN=0 ⇒ Keep what was stored previously

Is this synchronous?



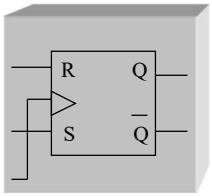


74373 has 8 Latches

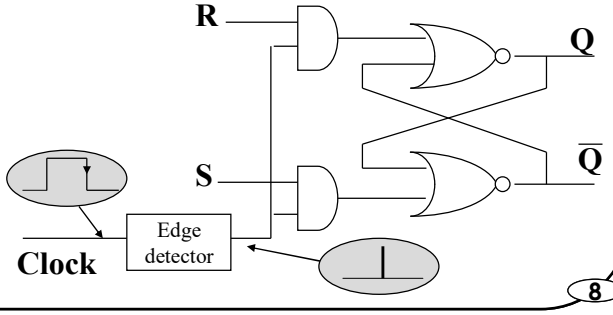
7

FLIP-FLOPs

- FLIP-FLOP
  - Only changes its outputs when a CLOCK edge occurs
  - Only one of the edges (RISING edge of the clock, or FALLING edge) leads to possible changes of the output.
- EDGE TRIGGERED FLIP-FLOP
  - Are sensitive to inputs in the moment just before the CLK edge
  - The "sensitivity window" is very narrow



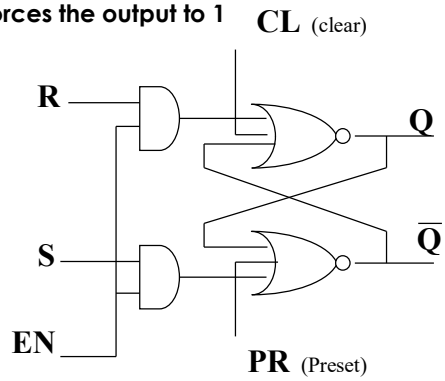
▷ means Edge-Triggered



8

Edge-Triggered Flip-Flop

- **A flip-flop may have** *Asynchronous inputs*, **that act independently from the CLK**
  - *CLEAR* **- forces the output to 0**
  - *PRESET* **- forces the output to 1**

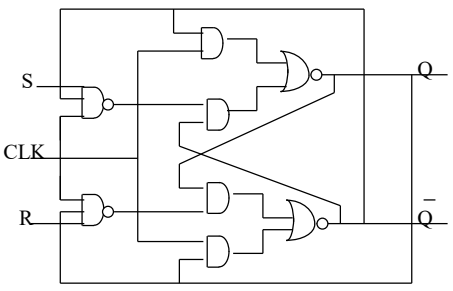


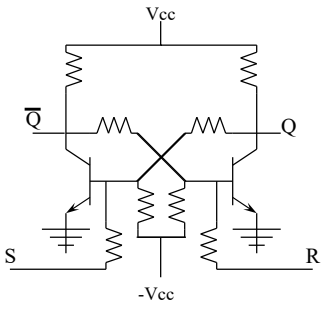
- **They are normally used to force RESETs or impose special conditions** (initial states, for example)

9

Other implementations

An Edge-Triggered FLIP-FLOP





RS LATCH
(using only two transistors)

- **The way a circuit is physically implemented can vary considerably…**

10

Flip-flops according to inputs: D,RS,JK,T

Sistemas Lógicos (3)

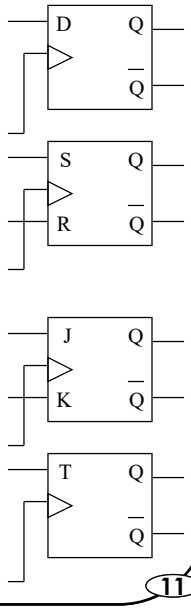
- **D Flip-flops (delay)**
  - Memorize and Delay the input
- **RS Flip-flops (Set-Reset)**
  - Turned "ON" (SET) and "OFF" (RESET) by S and R inputs
- **JK Flip-Flops**
  - Almost identical to SR, but solve the ambiguity of R=S=1, performing a *TOGGLE* (inverting the previous state)
  - O *J* performs a *Set*, and *K* a *Reset*
  - Most widely used flip-flops
- **T Flip-Flops (Toggle)**
  - Invert the outputs when the input is 1

D	Q
0	0
1	1

S	R	Q
0	0	Q
0	1	0
1	0	1
1	1	?

J	K	Q
0	0	Q
0	1	0
1	0	1
1	1	$\overline{Q}$

T	Q
0	Q
1	$\overline{Q}$



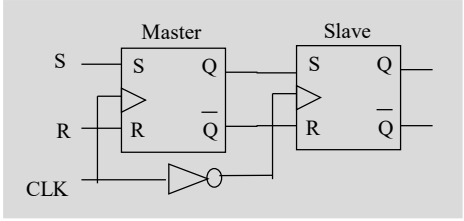
11

11

Master-Slave FLIP-FLOP

Sistemas Lógicos (3)

- **Memorize what happens when CLK is active, but only produce effects when the edge comes**
- **Simpler than the Edge-Triggered**
- **Composed of two latches in a row**
  - When one is "transparent", the other is "closed"
  - There is never a direct path from input to output
  - The first (Master) is connected to the input, and feeds its output to the second (Slave)
- **It has "1's catching "**
  - "Catches" peaks



12

12

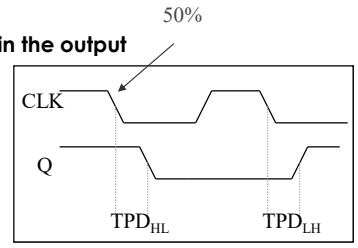
Simple Problems

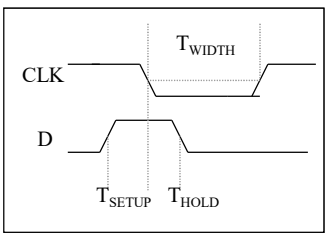
- **In an automobile one of the blinkers has a frequency of 2Hz. We wish to reduce that frequency to 0,5Hz. Project the circuits that reduces the frequency of the blinker.**
- **We wish to design an alarm system that has two sensors (for example infra-red laser beams) that send a "1" signal when there is nobody in front of it, and "0" when someone interrupts it. Assume there are 4 such beams, and when one of them is interrupted the alarm shoud go off, and ony stop when someone presses the "reset" button of the system. Project this system.**
- **Design a system to control traffic lights that sets the light red when it receives a 1, and green when it receives a 0. During the transitions, the system should turn on the yellow light on during 2 second before changing (both from 1 to 0 and 0 to 1).**

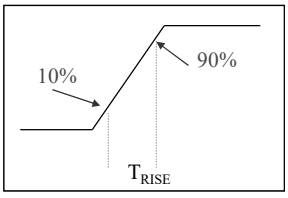
13

Flip-Flops – Time issues

- **Flip-flops can react to ascending or descending edges** (in this latter case, there is a negation in the clock input)
- **Propagation Delay**
  - **Time between the clock edge and the change in the output**
  - **May be different fot H-L and L-H**
- **Set-Up Time**
- **Hold Time**
- **Rise Time**
- **Pulse Width**



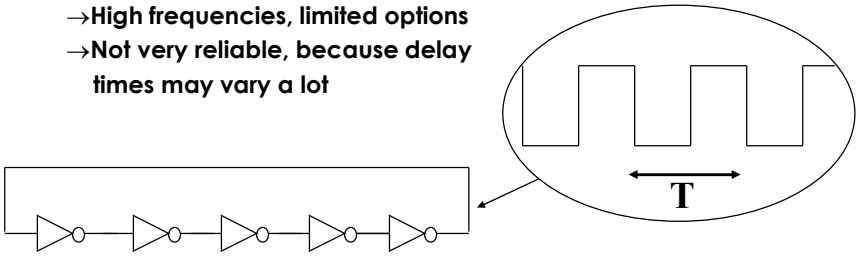




14

Timers and clock generators

- **Clock generators**
  - Generate square waves with a given frequency
- **Implementations with logical gates**
  - →High frequencies, limited options
  - →Not very reliable, because delay times may vary a lot



$$T \cong 2 \times n \times t_{pd}$$

$$f \cong 1/(2 \times n \times t_{pd})$$

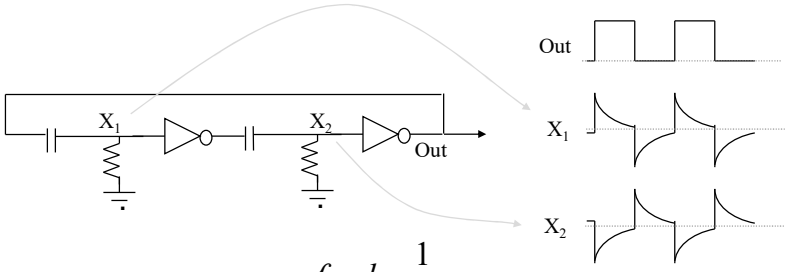
$$\text{Frequência} = \frac{1}{\text{Período}}$$

5 gates com t_{pd}= 9ns ⇒ f = 11MHz

15

Timers

- **With RC circuits and logical gates**
  - Easily adjusted
  - Not very precise (they depend on the resistor and capacitor tolerances, and temperature)



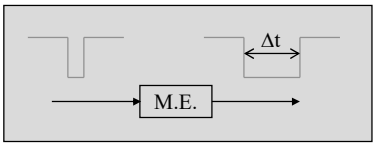
$$f = k\frac{1}{2RC}$$

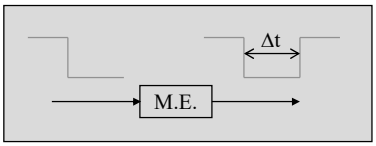
- **With Crystals**
  - Very precise
  - Only for reasonably high frequencies
  - Normally specialized integrated circuits are used as clock generators, using a quartz crystal as reference

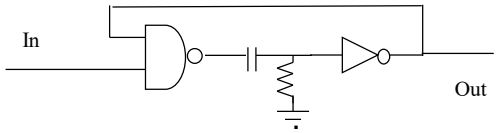
16

Monostable Timers

- **Monostable**
  - AKA "one-shot" timers
  - Generate a fixed width pulse
  - Can used to:
    - →Generate fixed delays
    - →Rectify short pulses



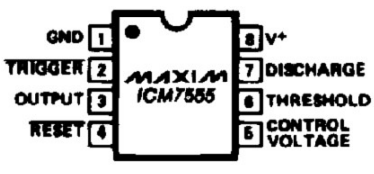


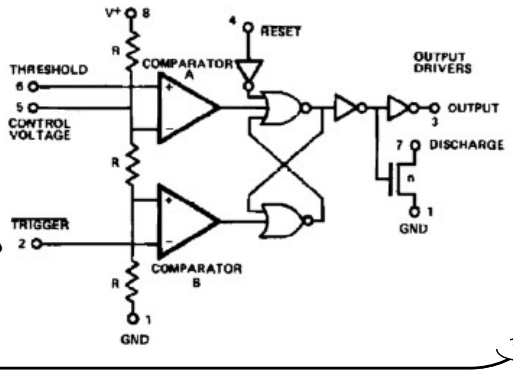


17

Integrated circuits for timers

- ***555***
  - Very flexible and widely used timer
  - May be used as monostable or clock generator
  - Adjustable "Duty Cycle"
  - Various clones, and integrated into other circuits
  - Circuit:
  - Pinout:

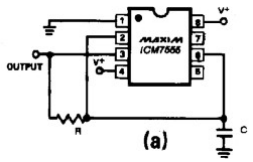




18

Integrated Circuits for clock generation

- **Typical mountings for the 555**



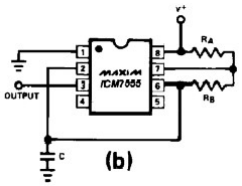
A) 50% duty-cycle Clock

$f = 1/(1,4\ RC)$

B) Variable duty-cycle Clock

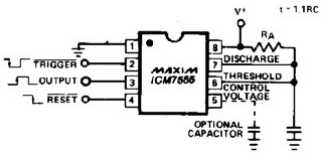
$f = 1,46 / ((Ra+2Rb)\ C)$

$d = Rb / (Ra+2Rb)$



C) Monostable

$T_{pulso} = 1,1\ RC$

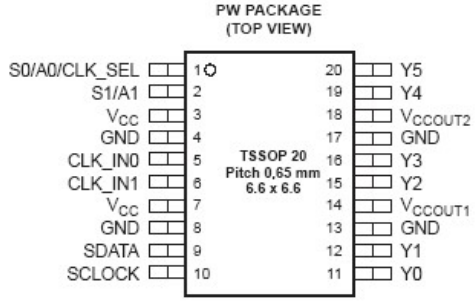


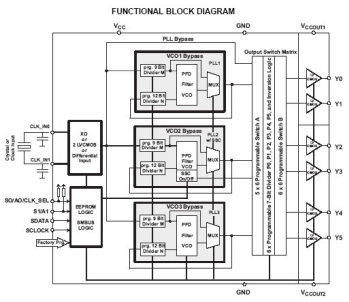
19

19

Other Clock Generators

- **Texas Instruments CDCE706 and TICE906**
  - Use only 1 crystal (or another clock signal)
  - Can generate 6 clock signals with different frequencies and duty-cycles
  - Programmable by software
  - Keeps programming in an EEPROM
  - (more information on http://focus.ti.com/docs/prod/folders/print/cdce706.html)



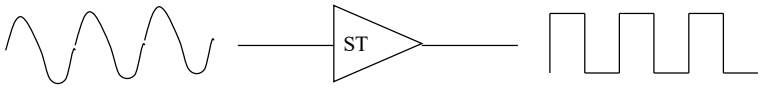


20

20

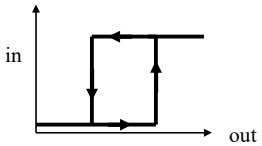
Schmitt-Triggers

- **May receive analog signals, but output digital**
  - **Output has only 2 levels**
  - **They are used to *regenerate* digital signals**



- **They have a hysteresis cycle:**
  - **To go from 0 to 1 the signal has to be significantly high, but to go from 1 to 0 the signal has to be significantly low**
  - **They eliminate "Cross-over noise"**

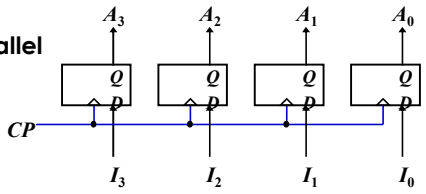
What types of problems can be solved by these circuits ?



21

Registers

- **How can we store information that needs more than 1 bit?**
  - **Use a set of Flip-Flops**
  - **Some sort of ordering is necessary to organize the flip-flops (FF)**
- **Registers**
  - **Basically, a collection of Flip-flops**
  - **They differ in how data is LOADED into the flips-flops; how that data is available to be READ from the outside; and possibly how data is passed from one flip-flop to another.**
- **Parallel Load Registers**
  - **Data is input and output in parallel**
  - **Operations:**
    - **→Parallel-in/Parallel-out**

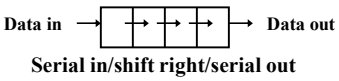


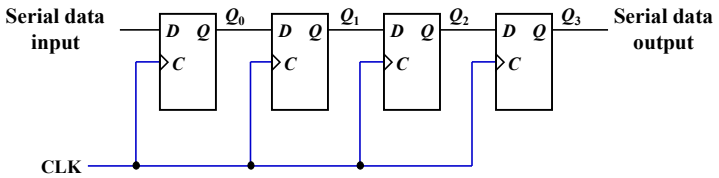
22

Shift Register

Sistemas Lógicos (3)

- **Shift Register** *(Registo de deslocamento)*
  - They are *tapped delays*
  - The have a *Serial-in* input and a *Serial-out* output
  - Data (stored internally) is usually also available to be read in parallel
    - →May be used for series to parallel conversions
  - Some registers can do *shift-right* or *shift-left* operations
  - Application example
    - →Ring counter
    - →Detect a sequence of bits (for example Morse code 00011000)





23

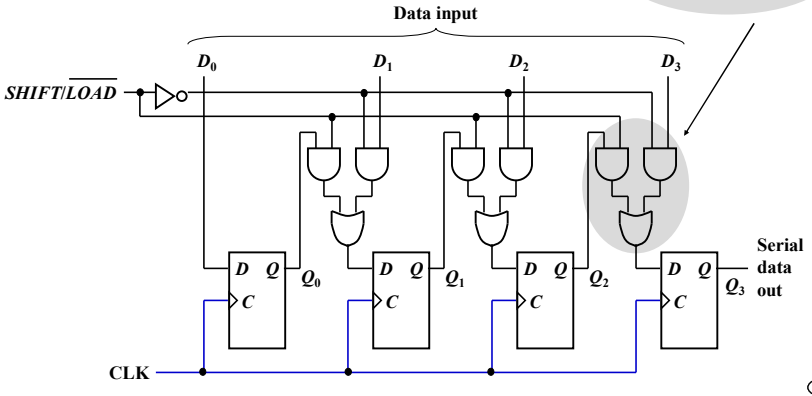
23

Multifunction Registers

Sistemas Lógicos (3)

- **Parallel-in/serial-out registers**
  - May be used for Parallel/Serial conversions
  - They have a control input that selects which operation will be done:
    - →Shift data (in this case from left to right)
    - →Load new data in parallel

This is basically A MUX that selects the source of data



24

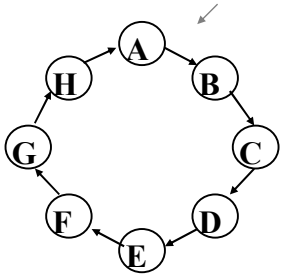
24

Counters

Sistemas Lógicos (3)

- **What is a counter ?**
  - A system that CYCLES through a series of STATES
  - Examples of sequences:
    - →0,1,2,3,,0,1,2,3,0,1,2,3…
    - →9,7,5,3,1,9,7,5,3…
  - Natural Binary Counter
    - →Corresponds to the traditional idea of counter
    - →Counts 0,1,2,3…(2^n-1)
  - Ring counter
    - →Counts 100,010,001...100,010….

State diagram



- **Modulus of a counter**
  - Number of states before returning to the initial state

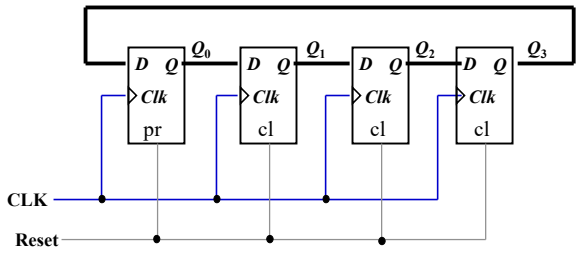
25

25

Ring Counter

Sistemas Lógicos (3)

- ***N* flip-flops, with only one set at 1 while others are 0**
- **The active flip-flop passes it's "1" onto the next when the clock edge arrives**
  - It's a shift register with feedback from the last bit
- ***N* flip-flops ⇒ *N* different states**
- **Example of a ring counter with 4 bits**
  - **Counting sequence:**



reset

1000
0100
0010
0001
1000
0100
.
.
.

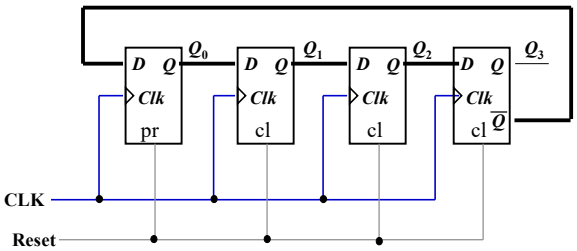
26

26

Johnson Counter

Sistemas Lógicos (3)

- Similar to the ring counter, but the negation of the output is fed back to the input (instead of the output itself)
- *N* flip-flops ⇒ 2*N* different states
- Example of a 4 bit Johnson counter
  - Counting Sequence



reset

Q
1000
1100
1110
1111
0111
0011
0001
0000
1000
1100
.
.

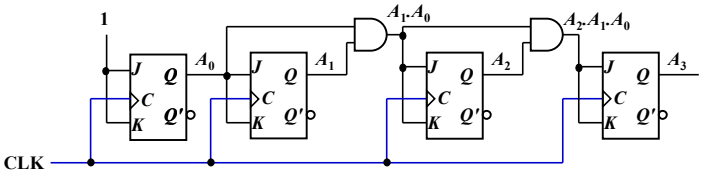
27

27

Synchronous Binary Counter (mod 2^n)

Sistemas Lógicos (3)

- Counts in natural binary
  - The LSB (Least Significant Bit) always toggles
  - The more significant bits only toggle when all the previous bits are 1
- *N* flip-flops ⇒ 2^N different states
- Example of a 4 bit synchronous binary counter
  - Counting Sequence



$A_3A_2A_1A_0$
0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111
0000
0001
.

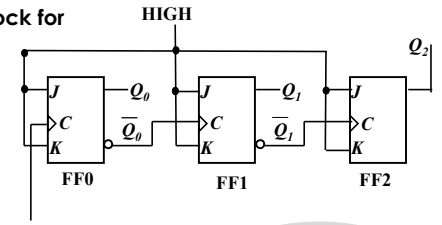
28

28

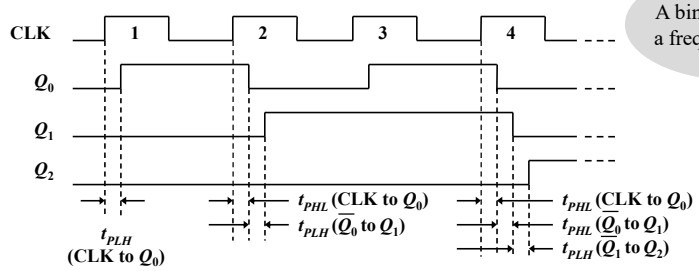
Asynchronous Binary Counter (ripple-mod 2^n)

Sistemas Lógicos (3)

- The clock is not common to all FF
  - The output of one flip-flip acts as clock for the next one
  - There are slight delays between the Fli-flops
- They are easier to build
  - They do not need external gates



Note:
A binary counter is a frequency divider



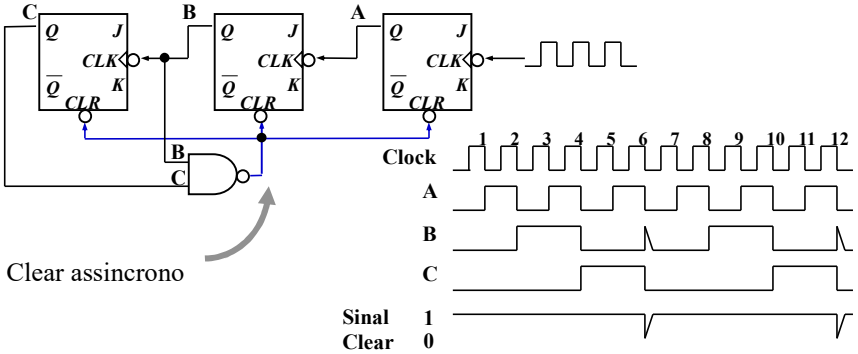
29

29

Arbitrary modulus Binary Counter

Sistemas Lógicos (3)

- Key idea → Reset the counter before it finishes the natural cycle
  - Detect the first "unwanted" state, and force an asynchronous clear (reset)
    - → Use AND gates to detect the unwanted state
    - → The unwanted state will exist during a small peak



30

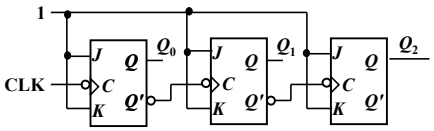
30

UP/DOWN Counters

Sistemas Lógicos (3)

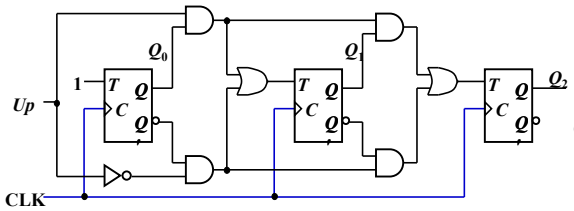
- **Down counters**
  - **Use NOT(Q) instead of Q**

Asynchronous 3 bit DOWN counter



111
110
101
100
011
010
001
000
111
110
101
100
011
.
.

- **UP/DOWN counters**
  - **Can count UP (0,1,2…) or DOWN (9,8,7…)**
  - **The function (UP or DOWN) is normally selected by an extra control input**



Contador UP/DOWN de 3 bits, síncrono

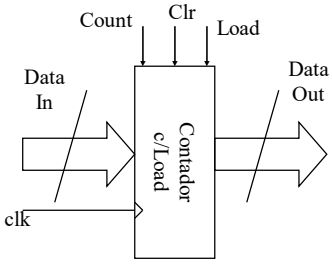
31

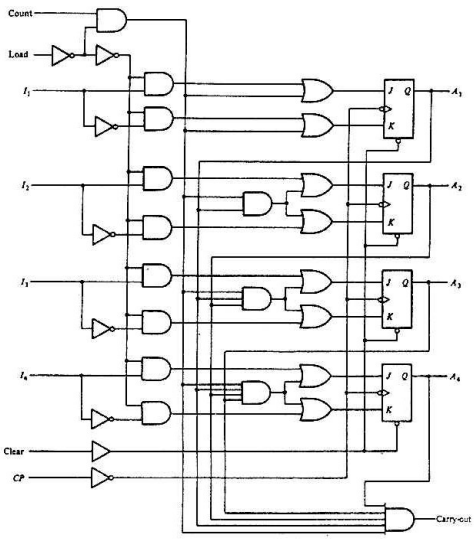
31

Counters with parallel load

Sistemas Lógicos (3)

- **Counters with parallel load**
  - **Instead of just a RESET input to force an initial 0 state, they may be reset to any given state (i.e., loaded with any data)**





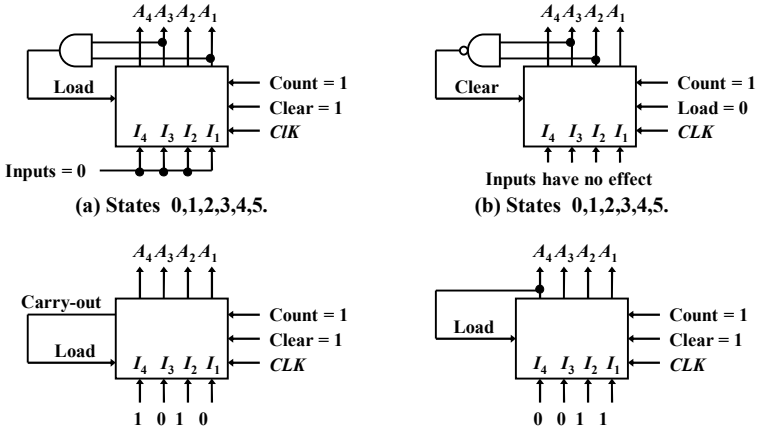
32

32

Examples modulus 6 counters

Sistemas Lógicos (3)

- 4 different ways of obtaining a modulus 6 counter with a modulus 16 binary counter with "parallel load" (synchronous) and clear (asynchronous)



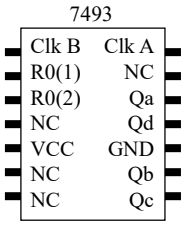
33

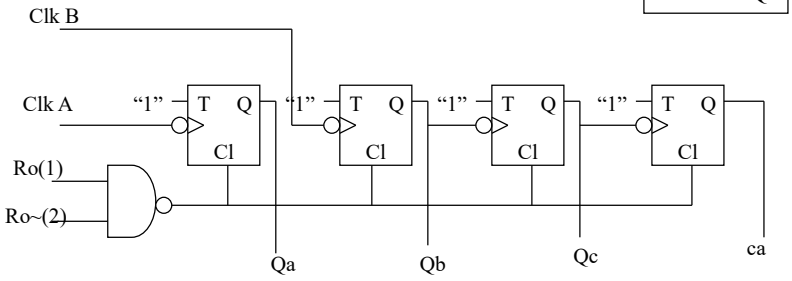
33

Counters – Available digital circuits

Sistemas Lógicos (3)

- 7493
  - Asynchronous Binary Counter (ripple-counter) with 4 bits
  - 1st bit is independent from the others
    →It must be connected externally



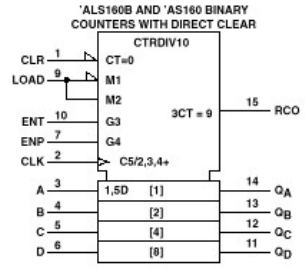


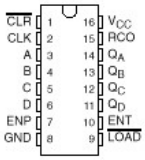
34

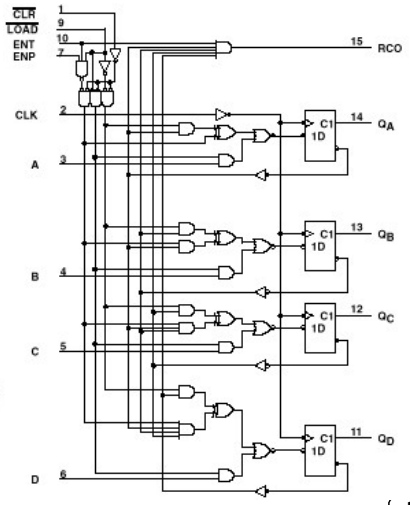
34

Contadores – Available devices

- **74163**
  - Synchronous 4 bit binary counter, with pre-load and enable
- **Others**
  - 74160 – Synchronous BCD counter
  - 74190 - Synchronous BCD up/down counter with pre-Load,





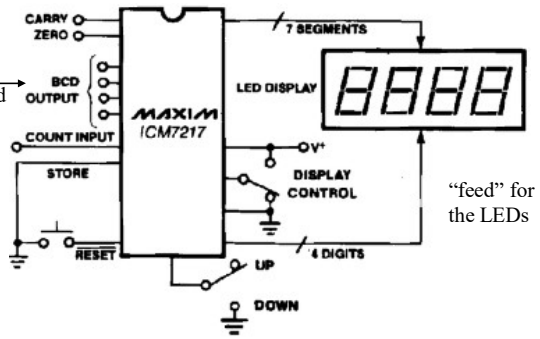


35

Counters – More complex devices

- **Maxim 7217**
  - Up/down 4 digit counter
  - Direct outputs for 7 segment displays (common cathode or anode and variable frequency)
  - Initial and final value are programable
  - Output carry for cascading
  - Allows for very compact systems

BCD output of the data being displayed (doubles as input for programming initial and final values)

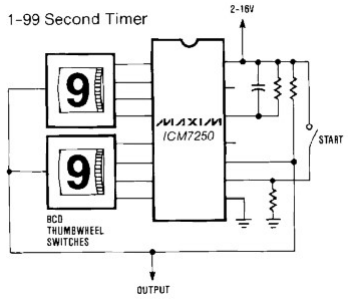


"feed" for the LEDs

36

Counters – More complex devices

- **7250**
  - **Digitally programable timer (0-99)**
  - **Unit delay defined by a RC circuit (thus, "tuneable")**
  - **May generate delays from micro-seconds to days**



Practical problem

For security reasons, we whish to prevent that more than 230 people get aboard one of boats that cross the Tagus river. In the entrance to the boats, there some rotating barriers that count how many people get in (sending a "1" pulse on the COUNT output each time someone passes).These barriers may be blocked by sending a "1" to the BLOCK input. The barrier should be blocked when 230 people have passed, and must only be unlocked when the boat leaves the pier.When it does, there is a sensor in the pier that sends a "1" pulse to it's LEFT output.
Project this system