Logic Circuits

Combinational or Memoryless Logic Circuits
- Function of Current Input Only

Sequential or Memory Logic Circuits
- Function of Current Input plus Past Inputs

State Table (Outputs & Next State)
- Next State = Present State + Current Input

Moore & Mealy Machines

Moore Machine
- Outputs Function of State Only

Mealy Machine
- Outputs Function of State and Input

Sequential Machines

Synchronous Sequential Machines
- Defined only at discrete times
  - Controlled by external clock
    - Uses Flip-Flops to hold
      - state variables between clock pulses

Asynchronous Sequential Machines
- Defined for all times
  - No need for explicit memory
    - Simpler - Two Implementation Restrictions

Asynchronous Machines

No more than one input variable may change at any on time. State variable must be assigned in such a way that no more than one state variable changes for any possible state changes.

"Simultaneous" Signal Changes
- Finite Propagation Times
- If 00 > 11 may happen in several ways
  - 00 > 01 > 11 or
  - 00 > 10 > 11
- Depends on "who wins the race"
- May have "Don't Care" States

Mealy & Moore Machines

Moore Machine is a finite-state machine whose output values are determined solely by its current state and can be defined as six elements (S, S_0, Σ, Λ, T, G), consisting of the following:

a finite set of states (S)
a start state (also called initial state) S_0 which is an element of (S)
a finite set called the input alphabet (Σ)
a finite set called the output alphabet (Λ)
a transition function ($T : S \times \Sigma \rightarrow S$) mapping a state and the input alphabet to the next state
an output function ($G : S \rightarrow \Lambda$) mapping each state to the output alphabet.

Mealy Machine output values are determined both by its current state and by the values of its inputs and can be defined as six elements (S, S_0, Σ, Λ, T, G), consisting of the following:

a finite set of states (S)
a start state (also called initial state) S_0 which is an element of (S)
a finite set called the input alphabet (Σ)
a finite set called the output alphabet (Λ)
a transition function ($T : S \times \Sigma \rightarrow S$) mapping a state and the input alphabet to the next state

an output function ($G : S \times \Sigma \rightarrow \Lambda$) mapping pairs of a state and an input symbol to the corresponding output symbol.

http://en.wikipedia.org/wiki/Theory_of_Computation

Digital Logic Signal Levels and State Variables

Simple Positive Logic

Define "Lo" = State "0" = 0; i.e., "near 0 volts, or maybe +0.7V, or less than +2.1V, etc."
Define "Hi" = State "1" = 1; i.e., "near Vcc,
say +5V, or greater than +3.9V, etc. for TTL;
or +15V, or greater than +13.1V, etc. for CMOS."

Remember these are arbitrary definitions.
Notice however, that State "1" is more positive than State "0". With this in mind, we can even define
"Hi" = State "1" = 1 = 0 volts, and
"Lo" = State "0" = 0 = -5 volts.
We still have State "1 more positive than State "0".
And Boolean Algebra doesn't care!

Simple Negative Logic

Try reversing things, such that State "1" is more negative than State "0"; i.e.,
State "1" = 0 volts, and
State "0" = +5 volts, or even

State "1" = -5 volts, and
State "0" = 0 volts.
In both cases, State "1" is more negative than State "0".

Positive Logic Truth Tables

A	B	AND	OR
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

Negative Logic Truth Tables

$\overline{A}$	$\overline{B}$	OR	AND
1	1	1	1
1	0	1	0
0	1	1	0
0	0	0	0

Note: **Positive AND Logic = Negative $\overline{\text{OR}}$ Logic**
Positive OR Logic = Negative $\overline{\text{AND}}$ Logic

DeMorgan's Law

$$\overline{A \bullet B} = \overline{A} + \overline{B} \qquad \overline{A + B} = \overline{A} \bullet \overline{B}$$

$$\overline{\overline{A \bullet B}} = \overline{\overline{A} + \overline{B}} = \overline{\overline{A}} \bullet \overline{\overline{B}} = A \bullet B \qquad \overline{\overline{A + B}} = \overline{\overline{A} \bullet \overline{B}} = \overline{\overline{A}} + \overline{\overline{B}} = A + B$$

Positive Logic $A \bullet B$ = Negative Logic $\overline{\overline{A} + \overline{B}}$

Positive Logic $A + B$ = Negative Logic $\overline{\overline{A} \bullet \overline{B}}$

Making Sense of SR Flip Flop Seemingly Contradictory Explanations

SR Flip Flop

Unfortunately, there is no consistency in describing the operation of SR Flip Flops (Set Reset); in fact, many of us even refer to them as RS Flip Flops.
However, one property description is pretty much universal:

Set **S** implies Q = 1
Reset **R** implies Q = 0

Adding even more to the confusion, is an error in the Scherz textbook, *Practical Electronics for Inventors*, 3ed, page 770, Figure 12-70 / 4ed, page 759, Figure 12-56 Cross NAND SR Flip Flop; the outputs **Q** and $\overline{Q}$ are reversed. **Q** should be associated with the S input NAND gate and $\overline{Q}$ should be associated with the R input NAND gate.

Be careful, don't confuse yourself when using other resources; some authors associate Q and $\overline{Q}$ with S & R respectively, other authors reverse the association. And then there is the confusion with respect to NOR SR Flip Flops, NAND SR Flip Flops, and inverted inputs to both NOR and NAND Flip Flops. For our purposes, the following concepts apply:

Set **S** implies	Q = 1	
Reset **R** implies	Q = 0	
Not Allowed	NOR Gates	S = 1 and R = 1
	NAND Gates	S = 0 and R = 0

If provisions for a clock pulse are not available, the circuit is known as an asynchronous flip flop.

Triggered or T Flip Flops

If the S and R inputs are gated with a clock pulse, the circuit is known as a synchronous flip flop.
If gated by a NAND, the S and R inputs are only enabled when the clock pulse is high.
When the clock is low, the inputs are disabled and the flip flop is placed in the Hold mode.

Latched Data or D Flip Flops (Single Input Device)

Invert the S input and apply to the R input:
if S = 0 then R = 1
if S = 0 then R = 0
but never S = R
Rename S as D:

D	S	R	Q	
0	0	1	0	(Reset)
1	1	0	1	(Set)

Each change in the input data toggles a change in the output.

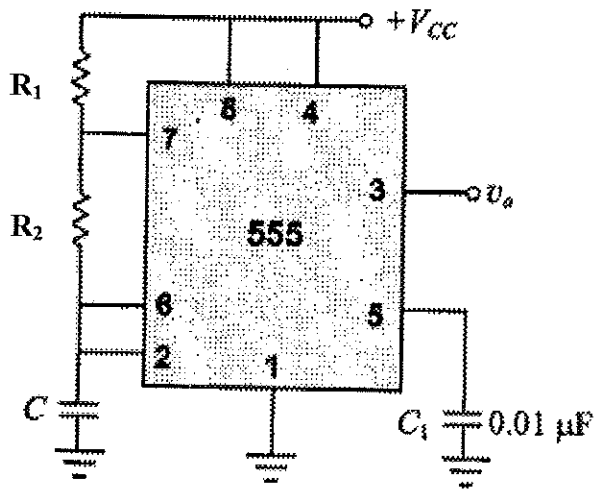
J K Master-Slave Flip Flop

Inputs: J, K, Set, Clear, Clock
Outputs: Q $\overline{Q}$
Trailing Edge Triggered Flip Flop
Master triggers on the clock up-tick (slave inactive)
Slave follows master on the clock down-tick

Control	Q	
Set	1	
Clear	0	

Input		
J K	Q	
0 0	Q	Hold
0 1	0	Reset
1 0	1	Set
1 1	$\overline{Q}$	Toggle

555 Astable Multivibrator Characteristics



The following computational formulas apply to the 555 configuration shown above.

On-Time = $t_h = 0.69\ (R_1 + R_2)\ C$

Off-Time = $t_l = 0.69\ (R_2)\ C$

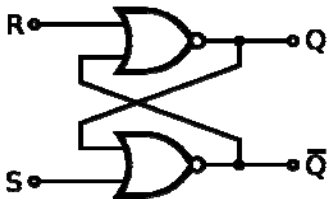
Period = $t_l + t_h = 0.69\ (R_1 + 2R_2)\ C$

Frequency = 1 / Period = $1.44 / (R_1 + 2R_2)\ C$

Duty Cycle = $t_h / (t_l + t_h) = (R_1 + R_2) / (R_1 + 2R_2)$

RS Flip Flop Truth Tables

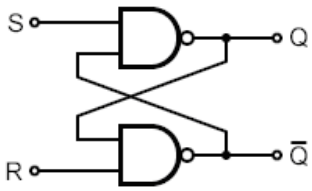
In order to eliminate ambiguity and to achieve some sense of continuity, we will follow the convention:
Set implies **Q** = **1**.
Reset implies **Q** = **0**.



Set Reset
NOR Gates

S	**R**	**Q**
0	0	Q Hold
0	1	0 Reset
1	0	1 Set
1	1	X

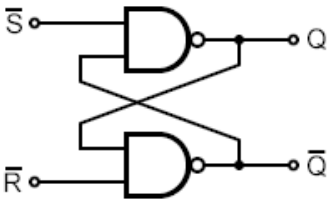
X = Not Allowed
S=1 => Set (Q=1)
R=1 => Reset (Q=0)



Set Reset
NAND Gates

S	**R**	**Q**
0	0	X
0	1	1 Set
1	0	0 Reset
1	1	Q Hold

X = Not Allowed
S=0 => Set (Q=1)
R=0 => Reset (Q=0)

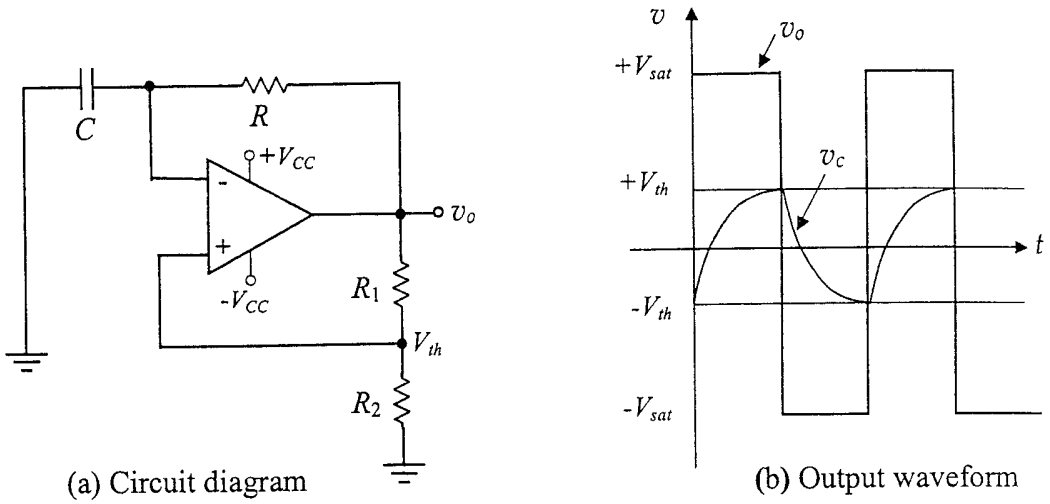


Set Reset
NAND Gates with Inverted S & R inputs

S	**R**	$\overline{\mathbf{S}}$	$\overline{\mathbf{R}}$	**Q**
0	0	1	1	Q Hold
0	1	1	0	0 Reset
1	0	0	1	1 Set
1	1	0	0	X

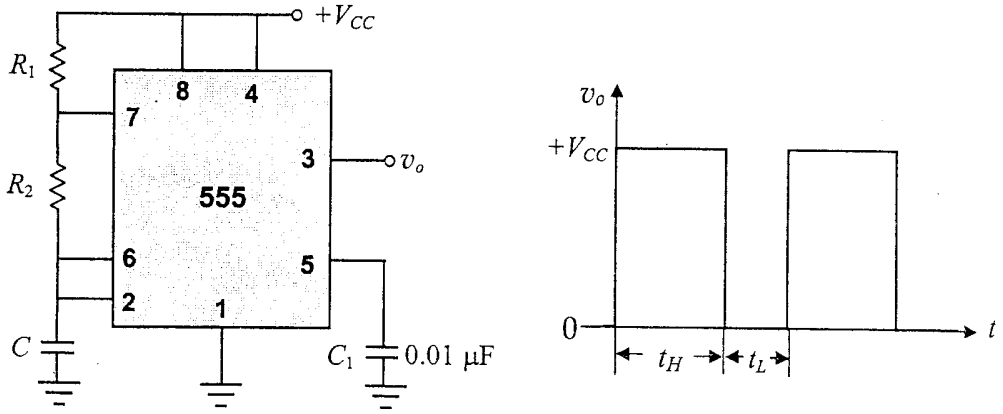
X = Not Allowed
S=1 => $\overline{S}$=0 => Set (Q=1)
R=1 => $\overline{R}$=0 => Reset (Q=0)

As you can see, there is consistency for **Set** means **Q=1** and **Reset** means **Q=0**; but there can be confusion trying to decide whether-or-not S & R are 0 or 1 depending on the type of gates (NOR or NAND).
If inverted S & R inputs are used with the NAND gates, then S=1 is the Set input and R=1 is the Reset input; which is the same as the NOR gates implementation.



(a) Circuit diagram (b) Output waveform

Figure 16-15: Square-wave generator



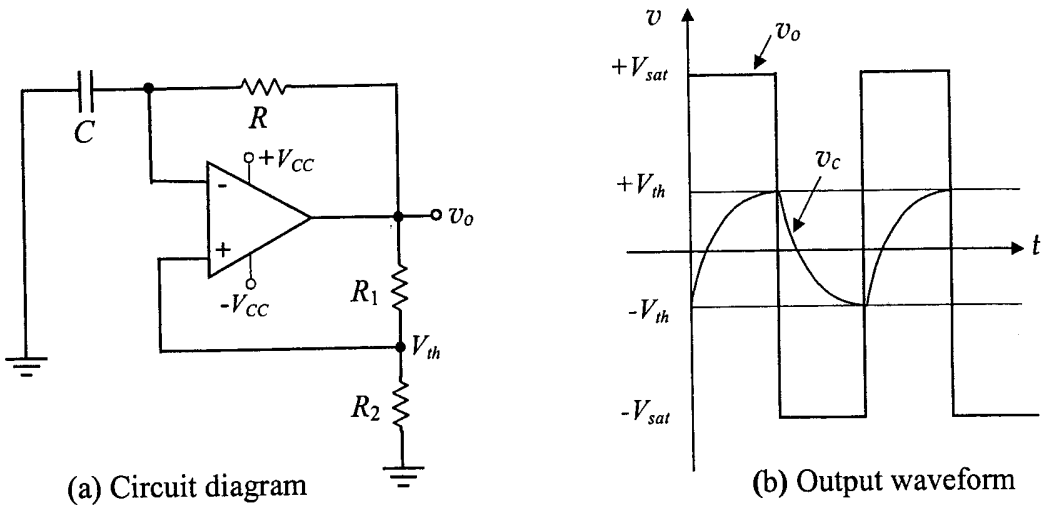
(a) Circuit diagram (b) Output waveform

Figure 16-17: The 555 timer connected as a rectangular waveform generator

Electronic Devices: a design approach, Ali Aminian & Marian Kazimierczuk, 2004

SQUARE-WAVE GENERATOR

Recall that the output of the Schmitt trigger, which was introduced in Chapter 11 as a bi-reference level comparator, is a square wave with $\pm v_{o(p)} = \pm V_{sat}$ of the op-amp. With the addition of a capacitor C and a feedback resistor R, as shown in Figure 16-15(a), the need for an input signal is eliminated and the output frequency can also be controlled by proper selection of the R and C.



(a) Circuit diagram (b) Output waveform

Figure 16-15: Square-wave generator

Referring to Equations 11-7 and 11-8, the upper and lower threshold voltages (V_{UT} & V_{LT}) or ($\pm V_{th}$) can be written in one equation as follows:

$$\pm V_{th} = \pm V_{sat} \frac{R_2}{R_1 + R_2} \tag{16-64}$$

It can be shown, with some considerable algebraic effort, that the period of the output waveform is as follows:

$$T = 2RC \ln\left(\frac{2R_2}{R_1} + 1\right) \tag{16-65}$$

$$f_0 = \frac{1}{T} = \frac{1}{2RC \ln(2R_2 / R_1 + 1)} \tag{16-66}$$

However, if we select R_1 and R_2 such that $(1 + 2R_2/R_1) = 2.178$ (the natural log base), then $\ln(1 + 2R_2/R_1)$ will equal unity.

$$\frac{2R_2}{R_1} + 1 = 2.718 \tag{16-67}$$

$$2R_2 = 1.718R_1 \tag{16-68}$$

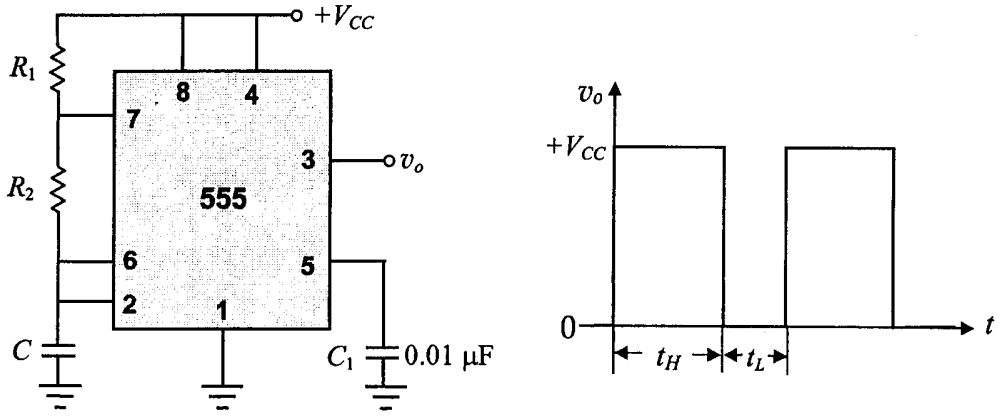
$$R_2 = 0.859R_1 \tag{16-69}$$

Hence, the output frequency is a function of R and C only, and its equation simplifies as follows:

$$f_o = \frac{1}{2RC} \tag{16-70}$$

16.8 THE 555 TIMER

The 555 timer is a popular 8-pin integrated circuit (*IC*), which may be used in many applications including rectangular waveform generation. Figure 16-17 shows the common configuration of the 555 timer as it is connected to produce a rectangular waveform.



(a) Circuit diagram (b) Output waveform

Figure 16-17: The 555 timer connected as a rectangular waveform generator

The time duration for which the output is high (t_H) is given by the following equation:

$$t_H = 0.69(R_1 + R_2)C \tag{16-71}$$

The time duration for which the output is low (t_L) is given by the following equation:

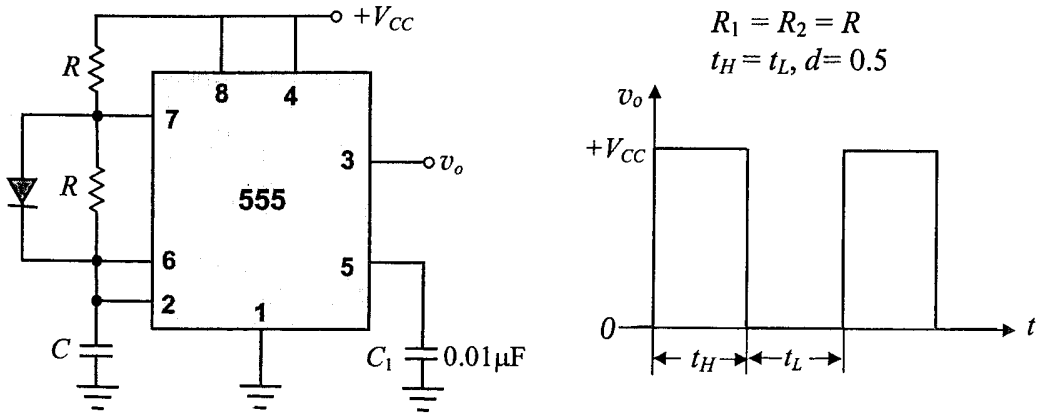
$$t_L = 0.69(R_2)C \tag{16-72}$$

Therefore, the period and frequency of the waveform are as follows:

$$T = t_H + t_L = 0.69(R_1 + 2R_2)C \tag{16-73}$$

$$f_o = \frac{1}{T} = \frac{1}{0.69(R_1 + 2R_2)C} \tag{16-74}$$

For a rectangular waveform, the ratio of the pulse duration (t_H) to the period T is referred to as the *duty cycle* (d) of the waveform. A square wave is a rectangular waveform with $d = 0.5$ or 50% duty cycle. Examining the equations for t_H and t_L, we notice that it would not be possible to produce a square wave with the circuit of Figure 16-16. However, there is a simple solution for this problem, and that is to connect a diode across the R_2 and let $R_1 = R_2 = R$, as illustrated in Figure 16-18(a).



(a) Circuit diagram (b) Output waveform

Figure 16-18: The 555 timer connected as a square-wave generator

When the output is high, the diode is forward-biased, shorting out R_2; hence,

$$t_H = 0.69(R_1)C = 0.69RC \tag{16-75}$$

When the output is low, the diode is unbiased, behaving like an open-circuit; hence,

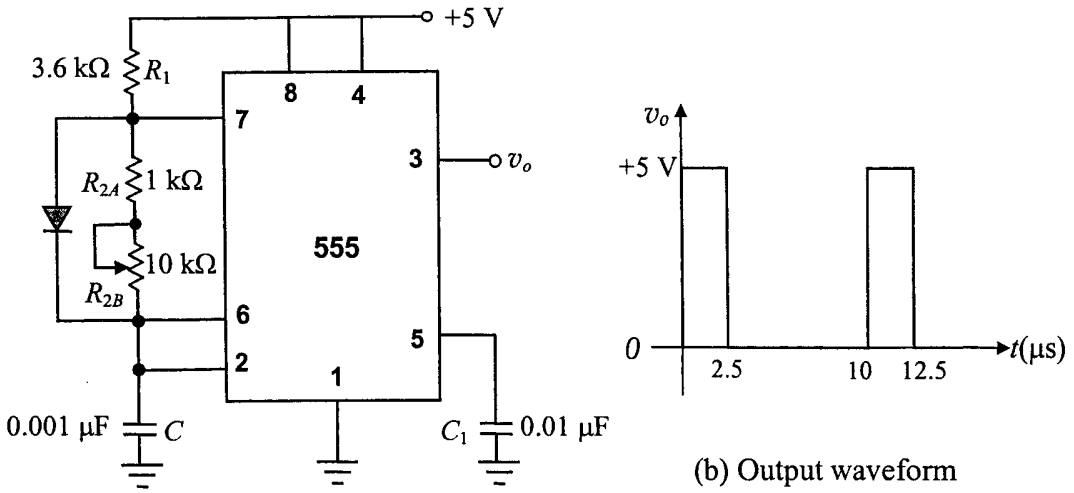
$$t_L = 0.69(R_2)C = 0.69RC \tag{16-76}$$

$$T = t_H + t_L = 0.69RC + 0.69RC = 1.38RC \tag{16-77}$$

$$f_o = \frac{1}{T} = \frac{1}{1.38RC} \tag{16-78}$$

$$d = \frac{t_H}{T} = \frac{0.69RC}{1.38RC} = 0.5 \tag{16-79}$$

In order to produce a rectangular waveform with a duty cycle less than 50% ($t_H < t_L$), we can pick R_2 larger than R_1, as required. However, the practical solution is to split R_2 into a series combination of a fixed resistor and a potentiometer, so that R_2 can be adjusted for a desired duty cycle.



(a) Circuit diagram (b) Output waveform

Figure 16-19: Rectangular waveform generator of Design Example 16-6