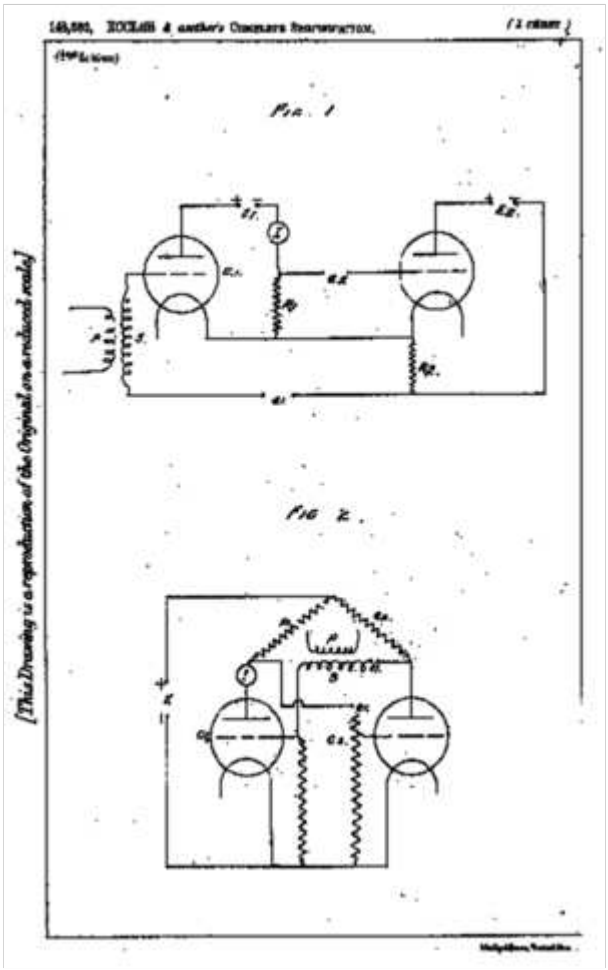
Flip-flop (electronics)

From Wikipedia, the free encyclopedia

In digital circuits, a **flip-flop** is a term referring to an electronic circuit (a bistable multivibrator) that has two stable states and thereby is capable of serving as one bit of memory.

A flip-flop is usually controlled by one or two control signals and/or a gate or clock signal. The output often includes the complement as well as the normal output.



Flip-flop schematics from the Eccles and Jordan patent filed 1918, one drawn as a cascade of amplifiers with a positive feedback path, and the other as a symmetric cross-coupled pair

Contents

History

The first electronic flip-flop was invented in 1918 by William Eccles and F. W. Jordan.[1][2] It was initially called the *Eccles–Jordan trigger circuit* and consisted of two active elements (vacuum tubes).[3]

The flip-flop types discussed below (RS, D, T, JK) were first discussed in a 1954 UCLA course on computer design by Montgomery Phister,[*citation needed*] and in his book *Logical Design of Digital Computers*.[4] The author was at the time working at Hughes Aircraft under Dr. Eldred Nelson, who had coined the term JK for a flip-flop which changed states when both inputs were on.[*citation needed*] The other names were coined by Phister. They differ slightly from some of the definitions given below.

The origin of the name for the JK flip-flop is detailed by P. L. Lindley, a JPL engineer, in a letter to *EDN*, an electronics design magazine. The letter is dated June 13, 1968, and was published in the August edition of the newsletter. In the letter, Mr. Lindley explains that he heard the story of the JK flip-flop from Dr. Eldred Nelson, who is responsible for coining the term while working at Hughes Aircraft. Flip-flops in use at Hughes at the time were all of the type that came to be known as J-K. In designing a logical system, Dr. Nelson assigned letters to flip-flop inputs as follows: #1: A & B, #2: C & D, #3: E & F, #4: G & H, #5: J & K.

Implementation

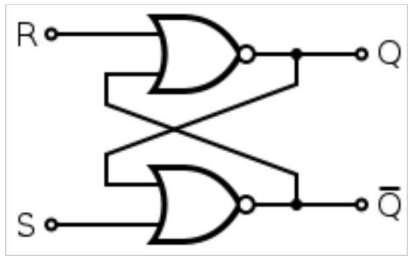
Flip-flops can be either simple (transparent) or clocked. Simple flip-flops can be built around a pair of cross-coupled *inverting* elements: vacuum tubes, bipolar transistors, field effect transistors, inverters, and inverting logic gates have all been used in practical circuits—perhaps augmented by some gating mechanism (an enable/disable input). The more advanced clocked (or non-transparent) devices are specially designed for synchronous (time-discrete) systems; such devices therefore ignore its inputs except **at** the transition of a dedicated clock signal (known as clocking, pulsing, or strobing). This causes the flip-flop to either change or retain its output signal based upon the values of the input signals at the transition. Some flip-flops change output on the rising edge of the clock, others on the falling edge.

Clocked flip-flops are typically implemented as master–slave devices[5] where two basic flip-flops (plus some additional logic) collaborate to make it insensitive to spikes and noise between the short clock transitions; they nevertheless also often include asynchronous *clear* or *set* inputs which may be used to change the current output independent of the clock.

Flip-flops can be further divided into types that have found common applicability in both asynchronous and clocked sequential systems: the **RS** ("set-reset"), **D** ("data" or "delay"[6]), **T** ("toggle"), and **JK** types are the common ones; all of which may be synthesized from (most) other types by a few logic gates. The behavior of a particular type can be described by what is termed the characteristic equation, which derives the "next" (i.e., after the next clock pulse) output, Q_{next}, in terms of the input signal(s) and/or the current output, Q.

RS (Reset-Set) flip-flop

The fundamental latch is the simple *RS flip-flop* (also commonly known as *SR flip-flop*), where R and S stand for *reset* and *set*, respectively. It can be constructed from a pair of cross-coupled NAND or NOR logic gates. The stored bit is present on the output marked Q.



An RS latch, constructed from a pair of cross-coupled NOR gates

Normally, in storage mode, the R and S inputs are both low, and feedback maintains the Q and $\overline{Q}$ outputs in a constant state, with $\overline{Q}$ the complement of Q. If S is pulsed high while R is held low, then the Q output is forced high, and stays high even after S returns low; similarly, if R is pulsed high while S is held low, then the Q output is forced low, and stays low even after R returns low.

The next-state equation of the RS flip-flop is

$$Q_{next} = S + \overline{R}Q$$

where Q is the current state. Q_{next} becomes Q (the stored value) at clock edge. This equation originates from Claude Shannon's 1937 master's thesis, *A Symbolic Analysis of Relay and Switching Circuits*.[7]

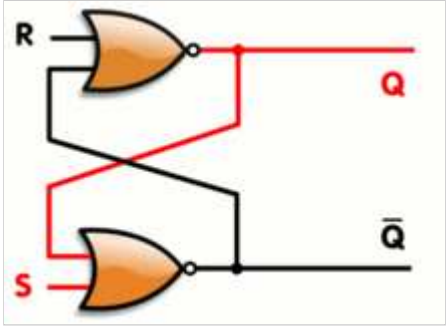


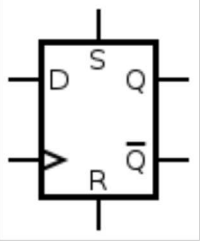
Illustration of RS latch operation. Red and black mean logical '1' and '0', respectively.

RS Flip-Flop operation (BUILT WITH NOR GATES) [8]							
Characteristic table			**Excitation table**				
S	**R**	**Action**	**Q(t)**	**Q(t+1)**	**S**	**R**	**Action**
0	0	Keep state	0	0	0	X	No change
0	1	Q = 0	1	0	0	1	reset
1	0	Q = 1	0	1	1	0	set
1	1	Race Condition	1	1	X	0	No Change

('X' denotes a Don't care condition; meaning the signal is irrelevant)

D (Delay) flip-flop

The D flip-flop is the most common flip-flop in use today. It is better known as *delay* flip-flop (as its output Q looks like a delay of input D) or *data latch*.



D flip-flop symbol

The Q output takes on the state of the D input at the moment of a positive edge at the clock pin (or negative edge if the clock input is active low).[9] It is called the **D** flip-flop for this reason, since the output takes the value of the **D** input or *Data* input, and *Delays* it by one clock cycle. The D flip-flop can be interpreted as a primitive memory cell, zero-order hold, or delay line. Whenever the clock pulses, the value of Q_{next} is D and Q_{prev} otherwise.

Truth table:

Clock	**D**	**Q**	**Q_{prev}**

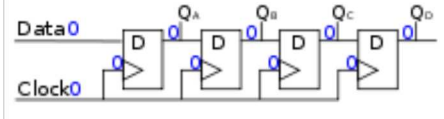
Rising edge	0	0	X
Rising edge	1	1	X
Non-Rising	X	Q_{prev}	

('X' denotes a *Don't care* condition, meaning the signal is irrelevant)

Most D-type flip-flops in ICs have the capability to be forced to the set or reset state (which ignores the D and clock inputs), much like an SR flip-flop. Usually, the illegal S = R = 1 condition is resolved in D-type flip-flops. By setting S = R = 0, the flip-flop can be used as described above.

Inputs				Outputs	
S	**R**	**D**	**>**	**Q**	**Q'**
0	1	X	X	0	1
1	0	X	X	1	0
1	1	X	X	1	1

These flip-flops are very useful, as they form the basis for shift registers, which are an essential part of many electronic devices. The advantage of the D flip-flop over the D-type "transparent latch" is that the signal on the D input pin is captured the moment the flip-flop is clocked, and subsequent changes on the D input will be ignored until the next clock event. An exception is that some flip-flops have a "reset" signal input, which will reset Q (to zero), and may be either asynchronous or synchronous with the clock.

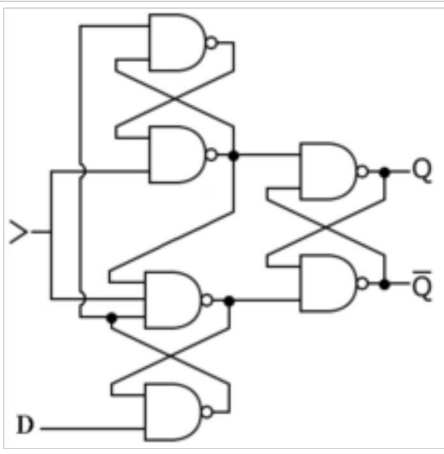


4-bit serial-in, serial-out (SISO) shift register

The above circuit shifts the contents of the register to the right, one bit position on each active transition of the clock. The input X is shifted into the leftmost bit position.

Classical positive-edge-triggered D flip-flop

This clever circuit[10] consists of two stages implemented by $\overline{SR}$ NAND latches. The input stage (the two latches on the left) processes the clock and data signals to ensure correct input signals for the output stage (the single latch on the right). If the clock is low, both the output signals of the input stage are high regardless of the data input; the output latch is unaffected and it stores the previous state. When the clock signal changes from low to high, only one of the output voltages (depending on the data signal) goes low and sets/resets the output latch: if D = 0, the lower output becomes low; if D = 1, the upper output becomes low. If the clock signal continues staying high, the outputs keep their states regardless of the data input and force the output latch to stay in the corresponding state as the input logical zero remains active while the clock is high. Hence the role of the output latch is to store the data only while the clock is low.



A positive-edge-triggered D flip-flop

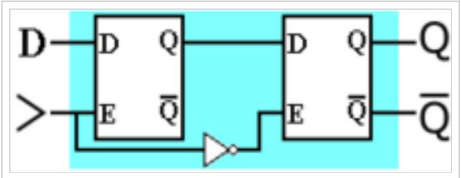
The circuit is closely related to the gated D latch as both the circuits convert the two D input states (0 and 1) to two input combinations (01 and 10) for the output $\overline{SR}$ latch by inverting the data input signal (both the

circuits split the single D signal in two complementary $\overline{S}$ and $\overline{R}$ signals). The difference is that in the gated D latch simple NAND logical gates are used while in the positive-edge-triggered D flip-flop $\overline{SR}$ NAND latches are used for this purpose. The role of these latches is to "lock" the active output producing low voltage (a logical zero); thus the positive-edge-triggered D flip-flop can be thought as of a gated D latch with latched input gates.

Master–slave (pulse-triggered) D flip-flop

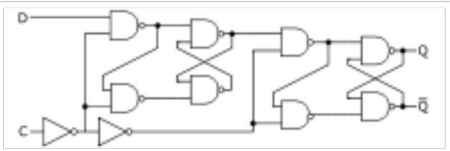
A master–slave D flip-flop is created by connecting two gated D latches in series, and inverting the *enable* input to one of them. It is called master–slave because the second latch in the series only changes in response to a change in the first (master) latch.

The term *pulse-triggered* means that data is entered on the rising edge of the clock pulse, but the output does not reflect the change until the falling edge of the clock pulse.



A master–slave D flip-flop. It responds on the negative edge of the *enable* input (usually a clock).

For a positive-edge triggered master–slave D flip-flop, when the clock signal is low (logical 0) the "enable" seen by the first or "master" D latch (the inverted clock signal) is high (logical 1). This allows the "master" latch to store the input value when the clock signal transitions from low to high. As the clock signal goes high (0 to 1) the inverted "enable" of the first latch goes low (1 to 0) and the value seen at the input to the master latch is "locked". Nearly simultaneously, the twice inverted "enable" of the second or "slave" D latch transitions from low to high (0 to 1) with the clock signal. This allows the signal captured at the rising edge of the clock by the now "locked" master latch to pass through the "slave" latch. When the clock signal returns to low (1 to 0), the output of the "slave" latch is "locked", and the value seen at the last rising edge of the clock is held while the "master" latch begins to accept new values in preparation for the next rising clock edge.



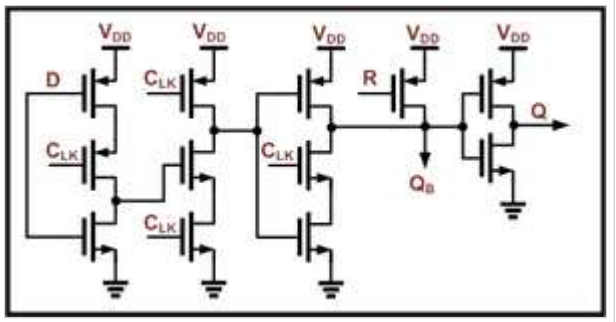
An implementation of a master–slave D flip-flop that is triggered on the positive edge of the clock

By removing the leftmost inverter in the circuit at side, a D-type flip flop that strobes on the *falling edge* of a clock signal can be obtained. This has a truth table like this:

D	Q	>	Q_{next}
0	X	Falling	0
1	X	Falling	1

Edge-triggered dynamic D flip-flop

A more efficient way to make a D flip-flop is not so easy to understand, but it works the same way. While the master–slave D flip-flop is also triggered on the edge of a clock, its components are each triggered by clock levels. The "edge-triggered D flip-flop" does not have the master–slave properties.



A CMOS IC Implementation of a True Single Phase Edge Triggered Flip Flop with Reset

Edge Triggered D flip flops are often implemented in integrated high speed operations using dynamic logic. This means that the digital output is stored on parasitic device capacitance while the device is not transitioning. This design of dynamic flip flops also enable simple resetting since the reset operation can be performed by simply discharging one or more internal nodes. A

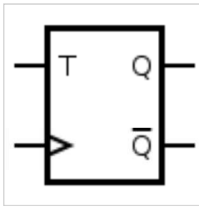
common dynamic flip flop variety is the True Single Phase Clock (TSPC) which performs the flip flop operation with little power and at high speeds. However these types of dynamic flip flops will not work at static or low clocked speeds, since given enough time the parasitic capacitance will discharge through leakage paths and will cause the logic levels to enter invalid states.

T (Toggle) flip-flop

If the T input is high, the T flip-flop changes state ("toggles") whenever the clock input is strobed. If the T input is low, the flip-flop holds the previous value. This behavior is described by the characteristic equation:

$$Q_{next} = T \oplus Q = T\overline{Q} + \overline{T}Q$$ expanding the XOR operator

and can be described in a truth table:



A circuit symbol for a T-type flip-flop, where > is the clock input, T is the toggle input and Q is the stored data output

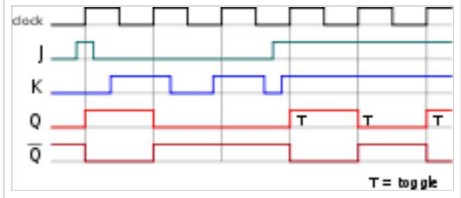
T Flip-Flop operation [8]							
Characteristic table				**Excitation table**			
T	***Q***	***Q*next***	**Comment**	***Q***	***Q*next***	***T***	**Comment**
0	0	0	hold state (no clk)	0	0	0	No change
0	1	1	hold state (no clk)	1	1	0	No change
1	0	1	toggle	0	1	1	Complement
1	1	0	toggle	1	0	1	Complement

When T is held high, the toggle flip-flop divides the clock frequency by two; that is, if clock frequency is 4 MHz, the output frequency obtained from the flip-flop will be 2 MHz. This "divide by" feature has application in various types of digital counters. A T flip-flop can also be built using a JK flip-flop (J & K pins are connected together and act as T) or D flip-flop (T input and $Q_{previous}$ is connected to the D input through an XOR gate).

Asynchronous T flip-flop

An asynchronous toggle flip-flop can also be built using an edge-triggered D flip-flop with its D input fed from its own inverted output.

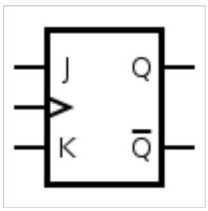
JK flip-flop



JK flip-flop timing diagram

The JK flip-flop augments the behavior of the SR flip-flop (J=Set, K=Reset) by interpreting the S = R = 1 condition as a "flip" or toggle command. Specifically, the combination J = 1, K = 0 is a command to set the flip-flop; the combination J = 0, K = 1 is a command to reset the flip-flop; and the combination J = K = 1 is a command to toggle the flip-flop, i.e., change its output to the logical complement of its current value. Setting J = K = 0 does NOT result in a D flip-flop, but rather, will hold the current state. To synthesize a D flip-flop, simply set K equal to the complement of J. The JK flip-flop is therefore a universal flip-flop, because it can be configured to work as an SR flip-flop, a D flip-flop, or a T flip-flop.

NOTE: The flip-flop is positive-edge triggered (rising clock pulse) as seen in the timing diagram.



A circuit symbol for a **Positive edge triggered** JK flip-flop, where > is the clock input, J and K are data inputs, Q is the stored data output, and Q' is the inverse of Q

The characteristic equation of the JK flip-flop is:

$$Q_{next} = J\overline{Q} + \overline{K}Q$$

and the corresponding truth table is:

JK Flip Flop operation [8]								
Characteristic table				**Excitation table**				
J	**K**	**Q_{next}**	**Comment**	**Q**	**Q_{next}**	**J**	**K**	**Comment**
0	0	Q_{prev}	hold state	0	0	0	X	No change
0	1	0	reset	0	1	1	X	Set
1	0	1	set	1	0	X	1	Reset
1	1	$\overline{Q}_{prev}$	toggle	1	1	X	0	No change

Uses

- A single flip-flop can be used to store one bit, or binary digit, of data. See preset.
- Any one of the flip-flop type can be used to build any of the others.
- Many logic synthesis tools will not use any other type than D flip-flop and D latch.
- Level sensitive latches cause problems with Static Timing Analysis (STA) tools and Design For Test (DFT). Therefore, their usage is often discouraged.
- Many FPGA devices contain only edge-triggered D flip-flops
- The data contained in several flip-flops may represent the state of a sequencer, the value of a counter, an ASCII character in a computer's memory or any other piece of information.
- One use is to build finite state machines from electronic logic. The flip-flops remember the machine's previous state, and digital logic uses that state to calculate the next state.
- The T flip-flop is useful for constructing various types of counters. Repeated signals to the clock input will cause the flip-flop to change state once per high-to-low transition of the clock input, if its T input is "1". The output from one flip-flop can be fed to the clock input of a second and so on. The final output of the circuit, considered as the array of outputs of all the individual flip-flops, is a count, in binary, of the number of cycles of the first clock input, up to a maximum of 2^n-1, where *n* is the number of flip-flops used. See: Counters
  - One of the problems with such a counter (called a *ripple counter*) is that the output is briefly invalid as the changes ripple through the logic. There are two solutions to this problem. The first is to sample the output only when it is known to be valid. The second, more widely used, is to use a different type of circuit called a *synchronous counter*. This uses more complex logic to ensure that the outputs of the counter all change at the same, predictable time. See: Counters
- Frequency division: a chain of T flip-flops as described above will also function to divide an input in frequency by 2^n, where *n* is the number of flip-flops used between the input and the output.

A flip-flop in combination with a Schmitt trigger can be used for the implementation of an arbiter in asynchronous circuits.

Clocked flip-flops are prone to a problem called metastability, which happens when a data or control input is changing at the instant of the clock pulse. The result is that the output may behave unpredictably, taking

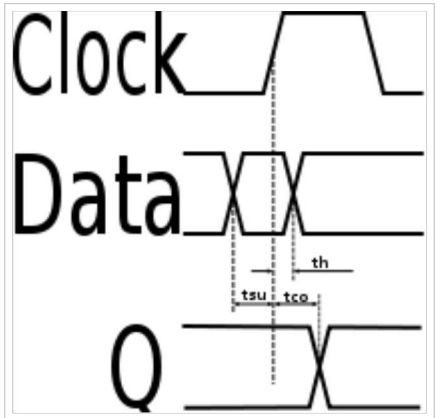
many times longer than normal to settle to its correct state, or even oscillating several times before settling. Theoretically it can take infinite time to settle down. In a computer system this can cause corruption of data or a program crash.

Setup and hold times

Setup time is the minimum amount of time the data signal should be held steady **before** the clock event so that the data are reliably sampled by the clock. This applies to synchronous circuits such as the **flip-flop**.

Hold time is the minimum amount of time the data signal should be held steady **after** the clock event so that the data are reliably sampled. This applies to synchronous circuits such as the flip-flop.

To summarize: Setup time -> Clock flank -> Hold time.



Flip-flop setup, hold and clock-to-output timing parameters

The metastability in flip-flops can be avoided by ensuring that the data and control inputs are held valid and constant for specified periods before and after the clock pulse, called the **setup time** (t_{su}) and the **hold time** (t_h) respectively. These times are specified in the data sheet for the device, and are typically between a few nanoseconds and a few hundred picoseconds for modern devices.

Unfortunately, it is not always possible to meet the setup and hold criteria, because the flip-flop may be connected to a real-time signal that could change at any time, outside the control of the designer. In this case, the best the designer can do is to reduce the probability of error to a certain level, depending on the required reliability of the circuit. One technique for suppressing metastability is to connect two or more flip-flops in a chain, so that the output of each one feeds the data input of the next, and all devices share a common clock. With this method, the probability of a metastable event can be reduced to a negligible value, but never to zero. The probability of metastability gets closer and closer to zero as the number of flip-flops connected in series is increased.

So-called metastable-hardened flip-flops are available, which work by reducing the setup and hold times as much as possible, but even these cannot eliminate the problem entirely. This is because metastability is more than simply a matter of circuit design. When the transitions in the clock and the data are close together in time, the flip-flop is forced to decide which event happened first. However fast we make the device, there is always the possibility that the input events will be so close together that it cannot detect which one happened first. It is therefore logically impossible to build a perfectly metastable-proof flip-flop.

Propagation delay

Another important timing value for a flip-flop (F/F) is the clock-to-output delay (common symbol in data sheets: t_{CO}) or propagation delay (t_P), which is the time the flip-flop takes to change its output after the clock edge. The time for a high-to-low transition (t_{PHL}) is sometimes different from the time for a low-to-high transition (t_{PLH}).

When cascading F/Fs which share the same clock (as in a shift register), it is important to ensure that the t_{CO} of a preceding F/F is longer than the hold time (t_h) of the following flip-flop, so data present at the input of the succeeding F/F is properly "shifted in" following the active edge of the clock. This relationship between t_{CO} and t_h is normally guaranteed if the F/Fs are physically identical. Furthermore, for correct operation, it is easy to verify that the clock period has to be greater than the sum $t_{su} + t_h$.

Chaos

Balthasar van der Pol was one of the first people to show electronic circuits may exhibit chaos in 1927, with the introduction of the Van der Pol oscillator. Then, Leon O. Chua showed circuits may exhibit chaos in 1983 through the introduction of Chua's circuit. Due to the qualitative nature of flip-flops, especially the Set/Reset Flip-Flop, one may intuitively feel it can exhibit chaos. This has been suggested in the works of Danca et al.[11] and of Hamill et al.,[12] which discusses the qualitative nature of circuits:

> Voltages or currents may increase exponentially with time until limited, perhaps by power supply clipping, when the circuit may latch up. This type of instability is put to good use in circuits such as Schmitt triggers and flip-flops.[9] and the waveforms may be noise like or chaotic, in which case they never repeat or latch up; as yet this type of behavior has few applications and is the least well understood.[9]

More recently, in Blackmore et al.[13] it is shown that discrete models of the Set/Reset Flip-Flop can exhibit chaos.

Generalizations

Flip-flops can be generalized in at least two ways: by making them 1-of-N instead of 1-of-2, and by adapting them to logic with more than two states. In the special cases of 1-of-3 encoding, or multi-valued ternary logic, these elements may be referred to as *flip-flap-flops*.[14]

In a conventional flip-flop, exactly one of the two complementary outputs is high. This can be generalized to a memory element with N outputs, exactly one of which is high (alternatively, where exactly one of N is low). The output is therefore always a one-hot (respectively *one-cold*) representation. The construction is similar to a conventional cross-coupled flip-flop; each output, when high, inhibits all the other outputs.[15] Alternatively, more or less conventional flip-flops can be used, one per output, with additional circuitry to make sure only one at a time can be true.[16]

Another generalization of the conventional flip-flop is a memory element for multi-valued logic. In this case the memory element retains exactly one of the logic states until the control inputs induce a change.[17] In addition, a multiple-valued clock can also be used, leading to new possible clock transitions.[18]

Flip-flop integrated circuits

Integrated circuits (ICs) exist that provide one or more flip-flops. For example, the 7473 dual JK master–slave flip-flop, or the 74374 octal D flip-flop, in the 7400 series.

See also

- Astable
- Deadlock
- Latch (electronics)
- Monostable
- Pulse transition detector
- Biochemical_switches_in_the_cell_cycle#Feedback_loops - biochemical, bistable, control circuits

Notes

1. ^ William Henry Eccles and Frank Wilfred Jordan, "Improvements in ionic relays (http://v3.espacenet.com/origdoc?DB=EPODOC&IDX=GB148582&F=0&QPN=GB148582) " British patent number: GB 148582 (filed: 21 June 1918; published: 5 August 1920).
2. ^ W. H. Eccles and F. W. Jordan (19 September 1919) "A trigger relay utilizing three-electrode thermionic vacuum tubes," *The Electrician*, vol. 83, page 298. Reprinted in: *Radio Review*, vol. 1, no. 3, pages 143–146 (December 1919).
3. ^ Emerson W. Pugh, Lyle R. Johnson, John H. Palmer (1991). *IBM's 360 and early 370 systems* (http://books.google.com/?id=MFGj_PT_clIC&pg=PA10&dq=eccles-jordan-trigger+vacuum&q=eccles-jordan-trigger%20vacuum) . MIT Press. p. 10. ISBN 9780262161237. http://books.google.com/?id=MFGj_PT_clIC&pg=PA10&dq=eccles-jordan-trigger+vacuum&q=eccles-jordan-trigger%20vacuum.
4. ^ Montgomery Phister (1958). *Logical Design of Digital Computers*. (http://books.google.com/?id=Ri1IAAAAIAAJ&q=inauthor:phister+j-k-flip-flop&dq=inauthor:phister+j-k-flip-flop) . Wiley. p. 128. http://books.google.com/?id=Ri1IAAAAIAAJ&q=inauthor:phister+j-k-flip-flop&dq=inauthor:phister+j-k-flip-flop.
5. ^ Early master–slave devices actually remained (half) open between the first and second edge of a clocking pulse; today most flip-flops are designed so they may be clocked by a **single** edge as this gives large benefits regarding noise immunity, without any significant downsides.
6. ^ PHY107 Delay Flip-Flop (http://www.shef.ac.uk/physics/teaching/phy107/dff.html)
7. ^ Claude Shannon, "A Symbolic Analysis of Relay and Switching Circuits," (http://dspace.mit.edu/bitstream/handle/1721.1/11173/34541425.pdf?sequence=1) , pg. 21, unpublished MS Thesis, Massachusetts Institute of Technology, Aug. 10, 1937.
8. ^ ***a b c*** Mano, M. Morris; Kime, Charles R. (2004). *Logic and Computer Design Fundamentals, 3rd Edition*. Upper Saddle River, NJ, USA: Pearson Education International. pp. pg283. ISBN 0-13-1911651.
9. ^ The D Flip-Flop (http://www.play-hookey.com/digital/d_nand_flip-flop.html)
10. ^ SN7474 TI datasheet (http://focus.ti.com/lit/ds/symlink/sn7474.pdf)
11. ^ Danca M-F. (2008). "Numerical approximation of a class of switch dynamical systems". *Chaos, Solitons and Fractals* **38**: 184–191. doi:10.1016/j.chaos.2006.11.003 (http://dx.doi.org/10.1016%2Fj.chaos.2006.11.003) .
12. ^ Hamill D, Deane J, Jeffries D (1992). "Modeling of chaotic DC/DC converters by iterated nonlinear maps". *IEEE Trans Power Electronics* **7**: 25–36. doi:10.1109/63.124574 (http://dx.doi.org/10.1109%2F63.124574) .
13. ^ Blackmore, D, Rahman, A, Shah, J (2009). "Discrete dynamical modeling and analysis of the R–S flip-flop circuit". *Chaos, Solitons and Fractals* **42**: 951. doi:10.1016/j.chaos.2009.02.032 (http://dx.doi.org/10.1016%2Fj.chaos.2009.02.032) .
14. ^ Often attributed to Don Knuth (1969) (see Midhat J. Gazalé (2000). *Number: from Ahmes to Cantor* (http://books.google.com/?id=hARkwMkeliUC&pg=PA57&dq=flip-flap-flop+knuth&q=flip-flap-flop%20knuth) . Princeton University Press. p. 57. ISBN 9780691005157. http://books.google.com/?id=hARkwMkeliUC&pg=PA57&dq=flip-flap-flop+knuth&q=flip-flap-flop%20knuth.), the term *flip-flap-flop* actually appeared much earlier in the computing literature, for example, Edward K. Bowdon (1960). *The design and application of a "flip-flap-flop" using tunnel diodes (Master's thesis)* (http://books.google.com/?id=0pA7AAAAMAAJ&q=flip-flap-flop+core&dq=flip-flap-flop+core) . University of North Dakota. http://books.google.com/?id=0pA7AAAAMAAJ&q=flip-flap-flop+core&dq=flip-flap-flop+core.
15. ^ "Ternary "flip-flap-flop"" (http://www.goldenmuseum.com/1411FlipFlap_engl.html) . http://www.goldenmuseum.com/1411FlipFlap_engl.html.
16. ^ US patent 6975152
17. ^ Irving, Thurman A. and Shiva, Sajjan G. and Nagle, H. Troy (March 1976). "Flip-Flops for Multiple-Valued Logic". *Computers, IEEE Transactions on* **C-25** (3): 237–246. doi:10.1109/TC.1976.5009250 (http://dx.doi.org/10.1109%2FTC.1976.5009250) .
18. ^ Wu Haomin1 and Zhuang Nan2 (1991). "Research into ternary edge-triggered JKL flip-flop". *Journal of Electronics (China)* **8** (Volume 8, Number 3 / July, 1991): 268–275. doi:10.1007/BF02778378 (http://dx.doi.org/10.1007%2FBF02778378) .

References

- Hwang, Enoch (2006). *Digital Logic and Microprocessor Design with VHDL* (http://faculty.lasierra.edu/%7Eehwang/digitaldesign) . Thomson. ISBN 0-534-46593-5. http://faculty.lasierra.edu/~ehwang/digitaldesign.
- Salman, E., Dasdan, A., Taraporevala, F., Kucukcakar, K., Friedman, E. (2006). "Pessimism Reduction in Static

Timing Analysis Using Interdependent Setup and Hold Times". *Proc. of Int. Symp. on Quality Electronic Design (ISQED)*. pp. 159–164. (This paper explains the interdependence of setup time, hold time, and clock-to-q delay and shows how to use it for pessimism reduction in static timing analysis.)
- Schulz, Klaus-E. (2007). Ideal pulse circuit without RC-combination and non-clocked JK flip-flops (look discussion) (http://www.hpc-berlin.de/dokumente/flipflop_en.pdf)
- Keating, M., Bricaud, P. (2002). *ReuseMethodology Manual*. KAP. ISBN 1-4020-7141-4.