Next: About this document ... **Up:** Iowa Logic Simulator User's **Previous:** Using the Logic Simulator

Subsections

74LS TTL Components

A standard set of subcircuits is provided with the logic simulator for designing logic circuits using standard TTL chips. The TTL family of integrated circuits was introduced about 20 years ago by Texas Instruments. TTL stands for *Transistor Transistor Logic*, which signifies that two transistors are used to drive each output of each chip, one for pulling the output down to a low level, and one for pulling the output up to a high level. Chips made using TTL technology are faster than the older RTL (*Resistor Transistor Logic*) and DTL (*Diode Transistor Logic*) families of integrated circuits, and they consume more power than the MOS (*Metal Oxide Semiconductor*) technology used in most VLSI (*Very Large Scale Integrated circuit*) chips.

The TTL family has at least 6 subfamilies which offer different speed/power tradeoffs. These are summarized in the following table, where they are listed in roughly the order in which they were introduced:

family		delay (ns)	power (mW)
basic		10	10
low-power	L	35	1
Schottky	S	3	18
low-power Schottky	LS	9	2
advanced Schottky	AS	1.5	10
advanced low-power Schottky	ALS	4	1

The term *Schottky* refers to a technology for making faster transistors. It is interesting to note that, in each generation of the TTL family, the low power representative of that generation is about 3 times slower than the other member, but consumes about 1/10 the power. Today, the low-power Schottky subfamily is the most widely used member of the TTL family; chips from this family are available from a wide variety of manufacturers

All manufacturers of TTL chips use a common naming system, as exemplified by the chip name ``SN74LS00''. The prefix SN indicates that the chip was made by Texas Instruments; other manufacturers have their own prefix codes, but if the remainder of the chip name matches, the chips should perform exactly the same function. Additional one letter codes may be added as prefixes or suffixes to this code, for example, RSN indicates radiation hardened chips made by Texas Instruments, and SNM indicates the use of quality control procedures specified by the military specification MIL-STD-883. The numeric code 74 indicates that the chip conforms to the requirements of the civilian computer industry, being able to operate over a temperature range of 0° to 70°C, while the code 54 indicates the ability to operate over the more extreme temperature range of -55° to 125°C required by many military and industrial applications. The letters LS indicate which subfamily the chip belongs to. Finally, the last two digits indicate the logical function performed by the chip.

Each Iowa Logic Specification Language TTL subcircuit description corresponds to one standard chip, and these chips are packaged one per file, in a form appropriate for inclusion using the `use` statement. The actual file names of these circuits will depend on your installation, but in general, the prefix indicating the manufacturer and military rating will not be included in the name, since it conveys nothing about the logical function of the chip. The following circuit descriptions are currently available (listed in numeric order):

LS00 Quad 2-input nand gates.
LS02 Quad 2-input nor gates.
LS04 Hex inverters.
LS08 Quad 2-input and gates.
LS10 Triple 3-input nand gates.
LS11 Triple 3-input and gates.
LS20 Dual 4-input nand gates.
LS21 Dual 4-input and gates.
LS27 Triple 3-input nor gates.
LS30 8-input nand gate.
LS32 Quad 2-input or gates.
LS42 BCD-to-decimal decoder (or 3-line to 8-line decoder with enable).
LS74A Dual positive-edge-triggered D flipflop.
LS85 4-bit binary magnitude comparator.
LS86 Quad 2-input exclusive-or gates.
LS109A Dual positive-edge-triggered J-$\overline{\text{K}}$ flipflop.
LS125A Quad bus-buffer gates with three-state outputs.
S133 13-input nand gate.
LS139 Dual 2-line to 4-line decoders/demultiplexers.
LS153 Dual 4-line to 1-line data selectors/multiplexers.
LS157 Quad 2-line to 1-line data selectors/multiplexers.
LS158 Quad 2-line to 1-line mux (as LS157), with inverted outputs.
LS161A Synchronous 4-bit binary counter.
LS164 8-bit serial to parallel shift register.
LS166 8-bit parallel to serial shift register.
LS174 Hex positive-edge-triggered D flipflops with common clock and clear.
LS175 Quad positive-edge-triggered D flipflops with common clock and clear.
S182 Look-ahead carry generators (see LS381A).
LS183 Dual full adders.
LS240 Octal inverting three-state driver.
LS244 Octal non-inverting three-state driver.
LS273 Octal positive-edge-triggered D flipflops with common clock and clear.
LS352 Dual 4-line to 1-line mux (as LS153), with inverted outputs.
LS374 Octal three-state positive-edge-triggered D flipflops with
LS381A Arithmetic logic unit with outputs for look-ahead carry.
LS382 Arithmetic logic unit with outputs for ripple carry.

The descriptions of these circuits are based on the details given in *The TTL Data Book* published by Texas Instruments. Equivalent data books are published by all of the major chip manufacturers such as Signetics and National Semiconductor. An effort has been made to make the simulated timings of these circuits match the documented timings of the corresponding chips. In general, the *Data Book* should be used as a source for information about these chips, while the information in the circuit descriptions should be considered secondary and subject to error.

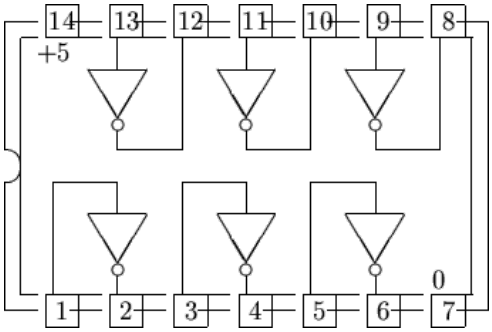
Each circuit description in the TTL collection follows a common scheme. The input and output pins of the chip have numeric names corresponding to the pin numbers used for a dual-in-line package; for example, for a 14 pin chip, these are `p1`, `p2`, `p3`, and so on up to `p14`. For any given chip, 2 of these pins are reserved for power (+5 volts) and ground (0 volts); these pins are not included in the chip descriptions, since they serve electronic purposes below the level of abstraction with which the logic simulator deals. The remaining pins are divided between input and output connections.

For example, consider the SN74LS00 chip; this has 4 nand gates. The inputs of the first nand gate are `p1` and `p2`, and its output is `p3`. The inputs of the second nand gate are `p4` and `p5`, and its output is `p6`. The remaining nand gates are connected to `p8` through `p13`, and pins 7 and 14 are power and ground, and so are not given names in the specification.

The input and output pins of the various chips are documented in the following sections, where the gates are listed in order by function. When a group of chips all have similar input and output connections, a diagram is given for only one of them, with the others listed below it. The diagrams on these pages are abbreviated and somewhat simplified compared with those in the data books; the data books should be used whenever an authoritative chip description is needed and these descriptions should only be used as a guide.

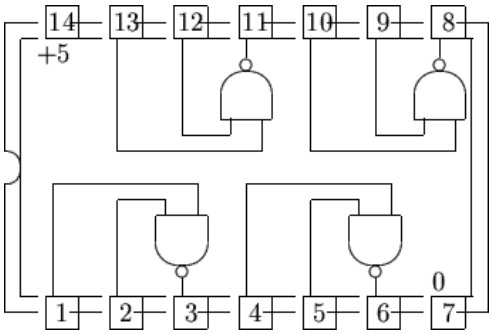
Primitive Gates

Inverters



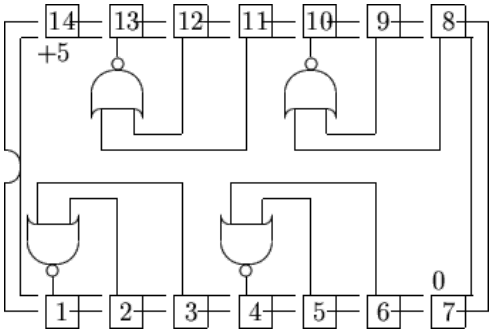
LS04 : Hex inverters. 9.5ns

Two Input Gates



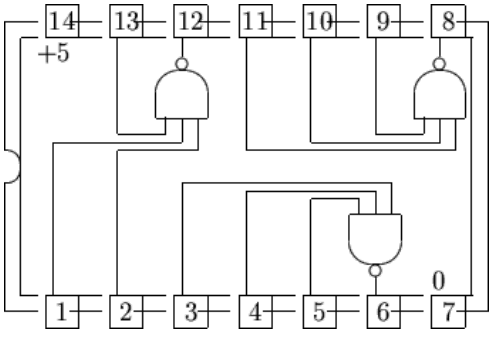
LS00 : Quad 2-input nand gates. 9.5ns
LS08 : Quad 2-input and gates. 9.0ns
LS32 : Quad 2-input or gates. 14ns
LS86 : Quad exclusive or gates. 10ns

Two Input Nor Gates



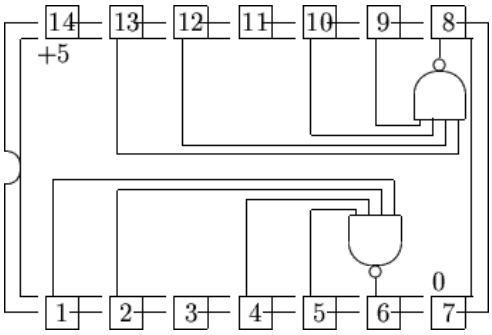
LS02 : Quad 2-input nor gates. 10ns

Three Input Gates



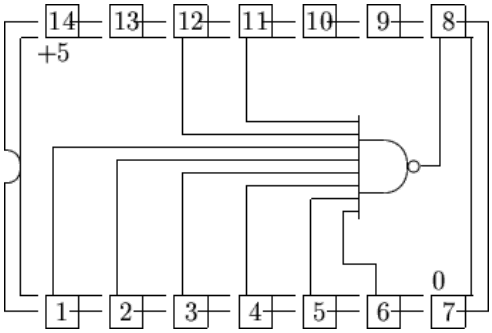
LS10 : Triple 3-input nand gates. 9.5ns
LS11 : Triple 3-input and gates. 9.0ns
LS27 : Triple 3-input nor gates. 10ns

Four Input Gates



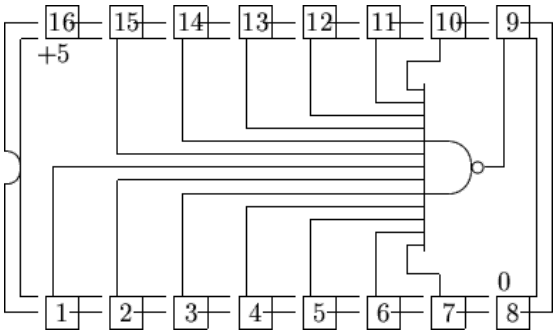
LS20 : Dual 4-input nand gates. 9.5ns
LS21 : Dual 4-input and gates. 9.0ns

Eight Input Nand Gates



LS30 : Single 8-input nand gates. 10.5ns

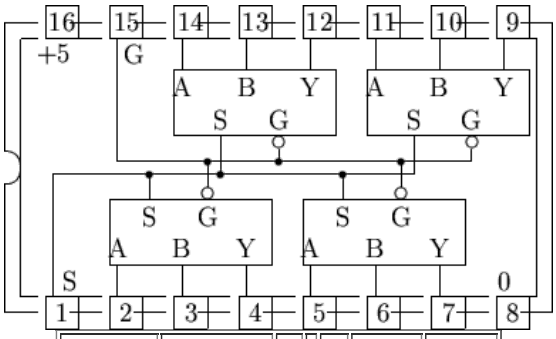
Thirteen Input Nand Gates



S133 : Single 13-input nand gates. 6ns

Multiplexers

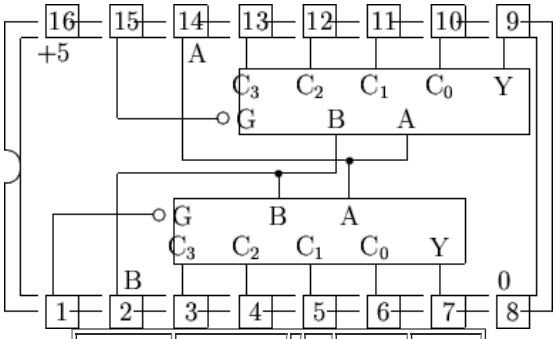
Two Input Multiplexers



4c||inputs	2coutput Y				
G	S	A	B	LS157	LS158
H	x	x	x	L	H
L	L	L	x	L	H
L	L	H	x	H	L
L	H	x	L	L	H
L	H	x	H	H	L

LS157 : Quad 2-line to 1-line multiplexers. 9ns
LS158 : Quad 2-line to 1-line inverting multiplexers. 7ns

Four Input Multiplexers

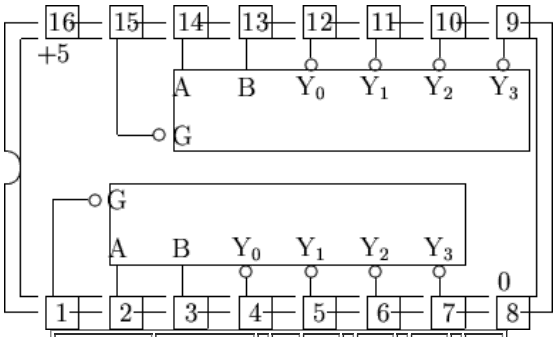


3c||inputs	2coutput Y			
G	B	A	LS153	LS352
H	x	x	L	H
L	L	L	C_0	$\overline{C_0}$
L	L	H	C_1	$\overline{C_1}$
L	H	L	C_2	$\overline{C_2}$
L	H	H	C_3	$\overline{C_3}$

LS153 : Dual 4-line to 1-line multiplexers. 22ns
LS352 : Dual 4-line to 1-line inverting multiplexers. 22ns

Decoders

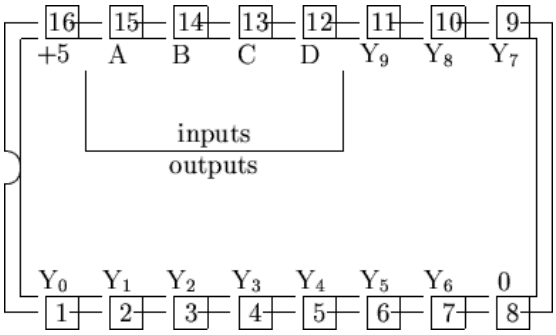
Four Output Decoders



3c||inputs	4coutputs						
G	B		A	Y_0	Y_1	Y_2	Y_3
H	x		x	H	H	H	H
L	L		L	L	H	H	H
L	L		H	H	L	H	H
L	H		L	H	H	L	H
L	H		H	H	H	H	L

LS139 : Dual 2-line to 4-line decoders. 22ns

Ten (or Eight) Output Decoders



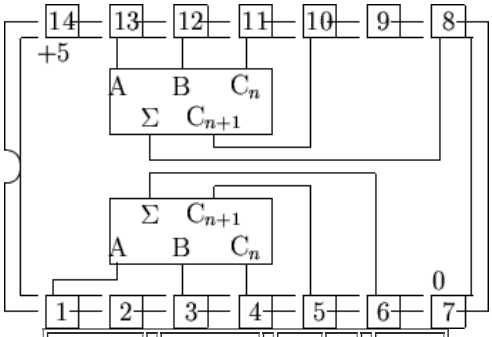
4c||inputs	c@ c@	10coutputs		c@ c@ c@ c@ c@ c@ c@ c@	
D	c@ c@	C	B	c@ c@ c@ c@ c@ c@ c@ c@	A
L	c@ c@	L	L	c@ c@ c@ c@ c@ c@ c@ c@	L
L	c@ c@	L	L	c@ c@ c@ c@ c@ c@ c@ c@	H
L	c@ c@	L	H	c@ c@ c@ c@ c@ c@ c@ c@	L
L	c@ c@	L	H	c@ c@ c@ c@ c@ c@ c@ c@	H
L	c@ c@	H	L	c@ c@ c@ c@ c@ c@ c@ c@	L
L	c@ c@	H	L	c@ c@ c@ c@ c@ c@ c@ c@	H
L	c@ c@	H	H	c@ c@ c@ c@ c@ c@ c@ c@	L
L	c@ c@	H	H	c@ c@ c@ c@ c@ c@ c@ c@	H
H	c@ c@	L	L	c@ c@ c@ c@ c@ c@ c@ c@	L
H	c@ c@	L	L	c@ c@ c@ c@ c@ c@ c@ c@	H
H	c@ c@	L	H	c@ c@ c@ c@ c@ c@ c@ c@	x
H	c@ c@	H	x	c@ c@ c@ c@ c@ c@ c@ c@	x

LS42 : BCD-to-decimal or binary-to-octal decoder. 17ns

To use the LS42 as a binary-to-octal decoder, ignore outputs Y_8 and Y_9, apply the binary input to C, B and A, and use input D as a negative-logic enable signal.

Arithmetic and Comparison

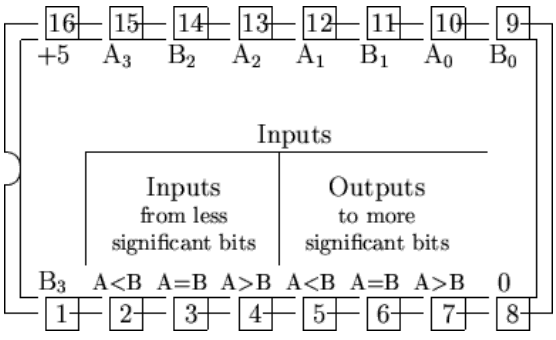
Dual Full Adders



3c‖inputs		2coutputs			
A		B	C_n	Σ	C_{n+1}
L		L	L	L	L
L		L	H	H	L
L		H	L	H	L
L		H	H	L	H
H		L	L	H	L
H		L	H	L	H
H		H	L	L	H
H		H	H	H	H

LS183 : Dual full adders. 15ns

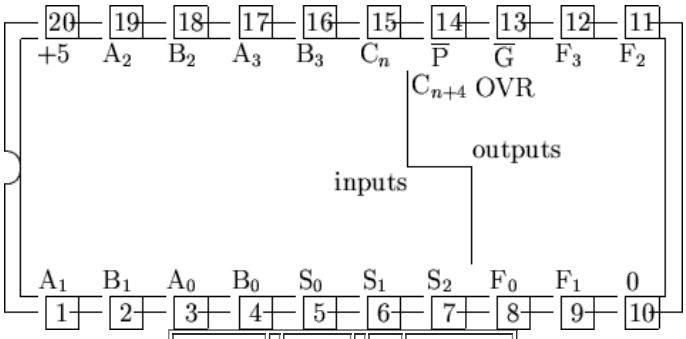
Binary Magnitude Comparators



LS85 : 4-bit binary magnitude comparators 24ns

In A and B, the two words to be compared, bit 0 is the least significant bit, and positive logic is used. The <, =, and > inputs to the least significant end of a chain of comparators should be L, H, and L respectively; the outputs from the most significant end are positive logic.

Arithmetic Logic Units



3c||select		output		
S_2		S_1	S_0	F
L		L	L	0000
L		L	H	B minus A
L		H	L	A minus B
L		H	H	A plus B
H		L	L	A xor B
H		L	H	A or B
H		H	L	A and B
H		H	H	1111

LS381A : 4-bit ALU slice with look-ahead carry outputs. 17ns
LS382 : 4-bit ALU slice with ripple carry outputs. 17ns

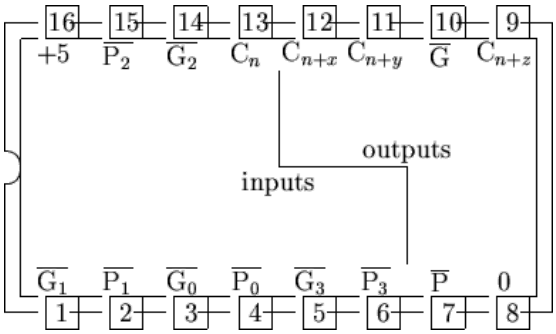
In A and B, the two operands, as well as in F, the result, bit 0 is the least significant bit, and positive logic is used. The delay given is from A and B to F; the function select input S must typically be set up at least 38ns prior to examining the outputs, while the carry input C_n typically takes only 16ns to reach the outputs.

The LS381A produces $\overline{P}$ and $\overline{G}$ outputs on pins 14 and 13 which can be used as inputs to S182 chips for look-ahead carry generation. When $\overline{P}$ is low, the 4-bit slice will propagate a carry, while when $\overline{G}$ is low, the 4-bit slice will generate a carry.

The LS382 produces C_{n+4} on pin 14, allowing ripple carry between 4-bit slices. The OVR output on pin 13 indicates signed two's complement overflow; effectively, it indicates that $C_{n+3} \neq C_{n+4}$.

Note that when using either ALU to perform subtraction, both the carry in and carry out signals indicate the absence of a borrow. Thus, C_0 (the carry in to the least significant bit) should be 1 during subtraction.

Look-Ahead Carry Generators



S182 : Look-ahead carry generators. 6ns

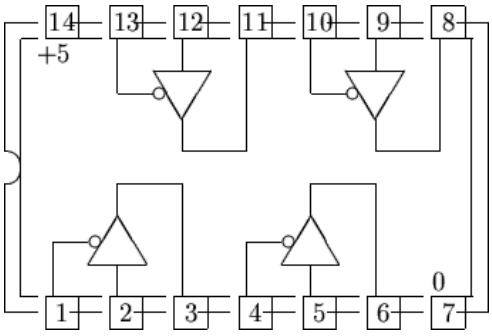
The S182 look-ahead carry generator can provide look-ahead carry propagation for as many as 4 ALUs such as the LS381A, and a tree of 5 S182 chips will serve to provide fast carry across 16 LS381A ALUs, allowing 64 bit arithmetic operations in typically 45ns (from data inputs to data outputs, assuming that the ALU function has been selected in advance).

For both inputs and outputs, $\overline{P}$ will be low if an ALU or group of ALUs will propagate a carry, and $\overline{G}$ will be low if an ALU or group of ALUs will generate a carry. $\overline{P_0}$ through $\overline{P_3}$ and $\overline{G_0}$ through $\overline{G_3}$ are inputs from ALUs or groups of ALUs, with increasing subscripts indicating more significant ALUs. $\overline{P}$ and $\overline{G}$ are outputs from this group of ALUs to higher levels in the tree.

The C_n input (positive logic) indicates carry in to this group of ALUs, and it should be the same as the carry in to the least significant ALU in the group. C_{n+x}, C_{n+y} and C_{n+z} are outputs to the 3 more significant ALU's in the group, in order of increasing significance. These do not depend on $\overline{P_3}$ or $\overline{G_3}$, so the most significant ALU used for a full word operation need not produce look-ahead signals.

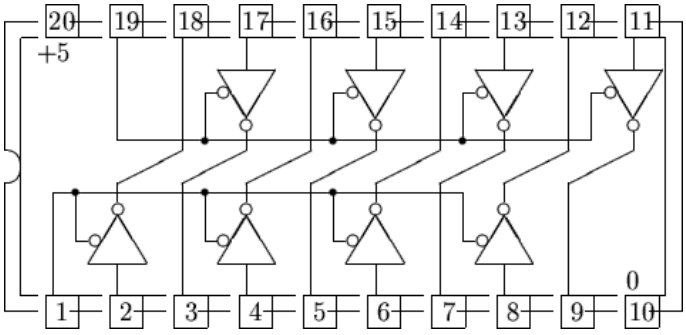
Three-state Bus Drivers

Quadruple Bus Drivers



LS125A : Quadruple non-inverting three-state buffers. 9ns

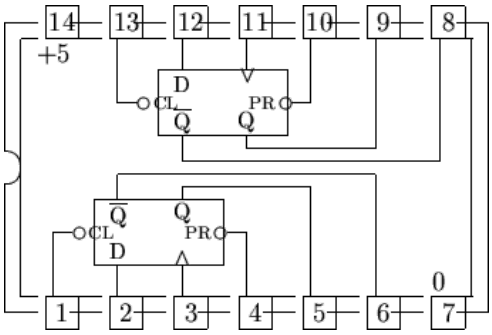
Octuple Bus Drivers



LS240 : Octal inverting three-state buffers. 10.5ns
LS244 : Octal non-inverting three-state buffers. 12ns

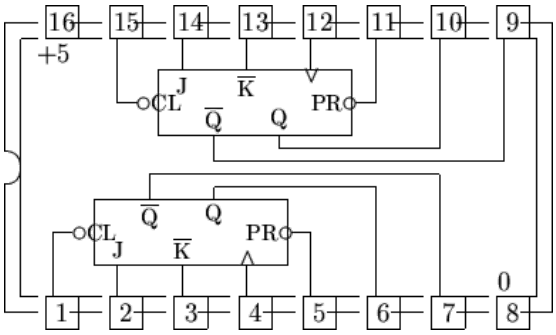
Flipflops

Dual D Flipflops



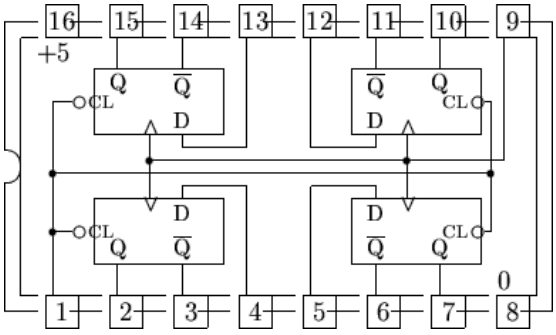
LS74A : Dual D positive-edge-triggered flipflops. 25ns

Dual J-K Flipflops



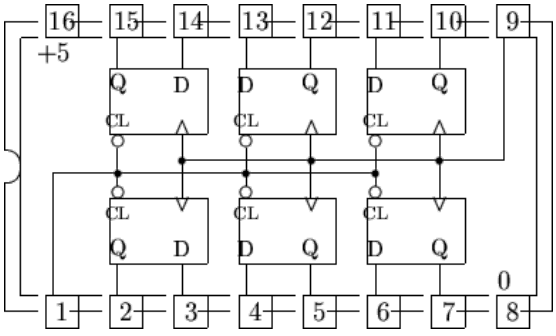
LS109A : Dual J-$\overline{K}$ positive-edge-triggered flipflops. 24ns

Quadruple D Flipflops



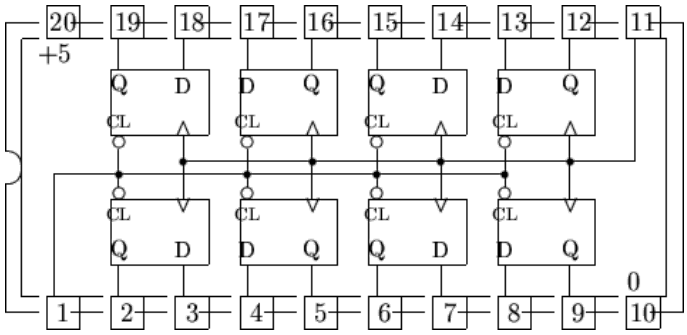
LS175 : Quad D positive-edge-triggered flipflops. 20ns

Hextuple D Flipflops



LS174 : Hex D positive-edge-triggered flipflops. 20ns

Octuple D Flipflops

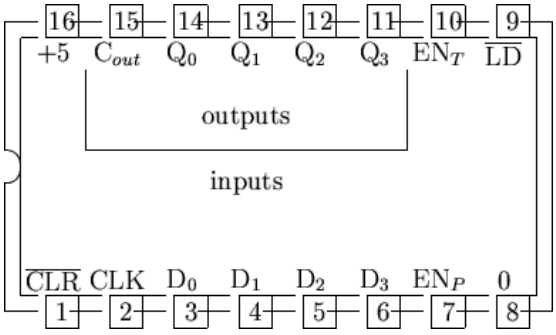


LS273 : Octal D flipflops with clear. 18ns
LS374 : Octal D flipflops with three-state outputs. 20ns

The clear input CL to each flipflop on the LS273 is replaced on the LS374 by an output-enable input called OE. Both CL and OE use negative logic, and all of these flipflops are positive edge triggered.

Counters

Synchronous 4-bit Binary Counters



LS161A : Synchronous 4-bit binary counter. 14ns

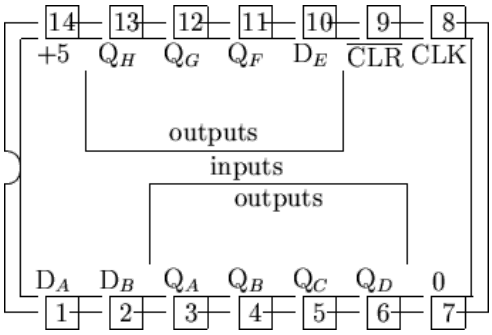
The LS161A binary counter has a positive edge triggered clock input CLK, and a negative logic asynchronous clear input $\overline{\mathrm{CLR}}$.The value stored in the counter is continuously presented on the Q outputs (Q_0 is least significant, positive logic is used).

When $\overline{\mathrm{LD}}$ is held low, data from the D inputs will be loaded into the counter when it is clocked. When EN_T, EN_P and $\overline{\mathrm{LD}}$ are all held high, the counter will increment when clocked. Holding either EN_T or EN_P inputs low will inhibit counting. The C_{out} output indicates that the counter holds the value 1111 and the EN_T input is high.

To build high precision synchronous counters, wire the C_{out} output of each stage to the EN_T input of the next more significant stage, and wire the CLK, $\overline{\mathrm{CLR}}$, $\overline{\mathrm{LD}}$ and EN_P inputs in parallel across all stages.

Shift Registers

8-bit serial to parallel shift registers



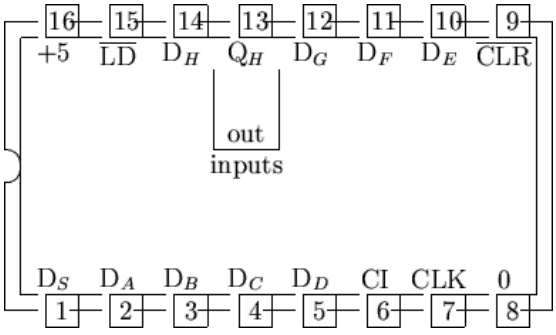
LS164 : 8-bit serial to parallel shift register. 28ns

The LS164 shift register has a positive edge triggered clock CLK and a negative logic asynchronous clear $\overline{\mathrm{CLR}}$.All 8 bits stored in the register are continuously displayed on the data outputs Q_A to Q_H, where Q_A was the most recent value shifted into the register.

The serial data inputs D_A and D_B are anded together; both must be high to shift a high into the register; either or both may be low to shift a low into the register.

To make a longer shift register, wire the Q_H output of one stage to the serial data inputs of the next stage, and wire all clock and clear inputs in parallel across all stages.

8-bit Parallel to Serial Shift Registers



LS166 : 8-bit parallel to serial shift register. 28ns

The LS166 shift register has a positive edge triggered clock CLK and a positive logic clock-inhibit input CI. CI should not change from low to high except when CLK is already high. $\overline{\mathrm{CLR}}$ is a negative logic asynchronous clear. The value stored in the 8th bit of the register is continuously presented on the output Q_H.

When $\overline{\mathrm{LD}}$ is held low, data from the inputs D_A to D_H will be loaded into the register when it is clocked. When $\overline{\mathrm{LD}}$ is held high, the register will shift when it is clocked, with new data entering stage A of the register from the serial data input D_S.

Longer shift registers can be made by chaining the Q_H output of one stage to the D_S input of the next, and connecting the CLK, CI, $\overline{\mathrm{CLR}}$, and $\overline{\mathrm{LD}}$ inputs in parallel across all stages.

Next: About this document ... **Up:** Iowa Logic Simulator User's **Previous:** Using the Logic Simulator
author-address