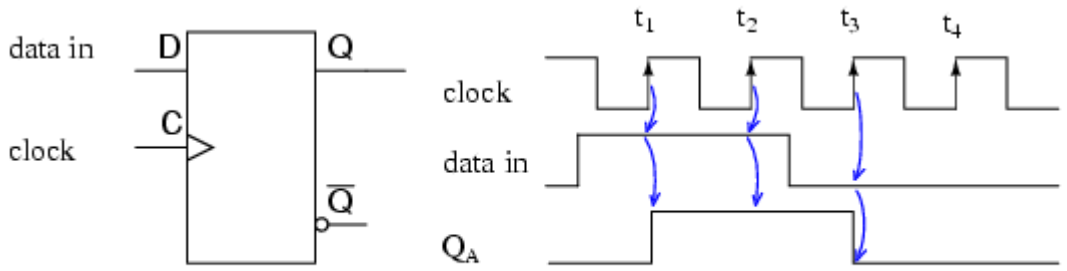
Serial in serial out shift register

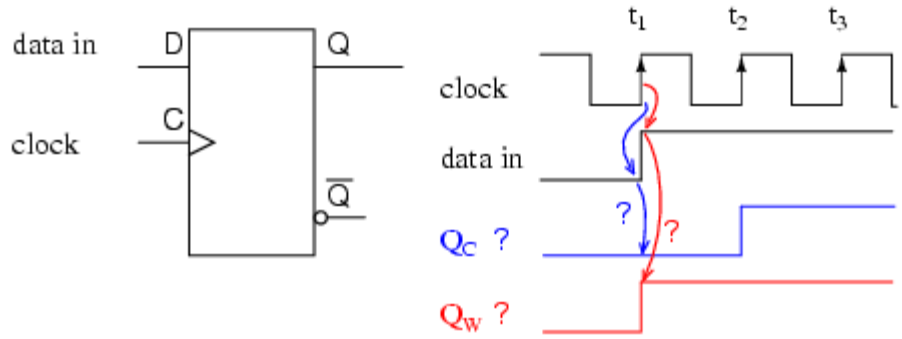
Serial-in, serial-out shift registers delay data by one clock time for each stage. They will store a bit of data for each register. A serial-in, serial-out shift register may be one to 64 bits in length, longer if registers or packages are cascaded.

Below is a single stage shift register receiving data which is not synchronized to the register clock. The "data in" at the **D** pin of the type **D FF** (Flip-Flop) does not change levels when the clock changes for low to high. We may want to synchronize the data to a system wide clock in a circuit board to improve the reliability of a digital logic circuit.



Data present at clock time is transfered from D to Q.

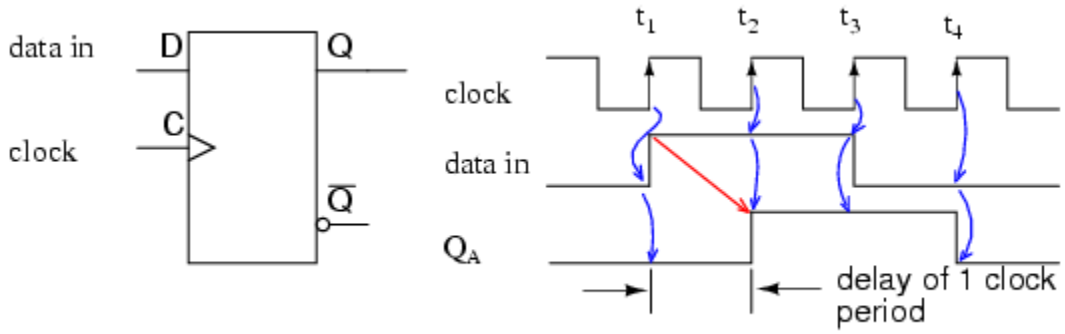
The obvious point (as compared to the figure below) illustrated above is that whatever "data in" is present at the **D** pin of a type **D** FF is transfered from D to output Q at clock time. Since our example shift register uses positive edge sensitive storage elements, the output **Q** follows the **D** input when the clock transitions from low to high as shown by the up arrows on the diagram above. There is no doubt what logic level is present at clock time because the data is stable well before and after the clock edge. This is seldom the case in multi-stage shift registers. But, this was an easy example to start with. We are only concerned with the positive, low to high, clock edge. The falling edge can be ignored. It is very easy to see **Q** follow **D** at clock time above. Compare this to the diagram below where the "data in" appears to change with the positive clock edge.



Does the clock t_1 see a 0 or a 1 at data in at D? Which output is correct, Q_C or Q_W ?

Since "data in" appears to changes at clock time t_1 above, what does the type **D** FF see at clock time? The short over simplified answer is that it sees the data that was present at **D** prior to the clock. That is what is transfered to **Q** at clock time t_1. The correct waveform is Q_C. At t_1 Q goes to a zero if it is not already zero.

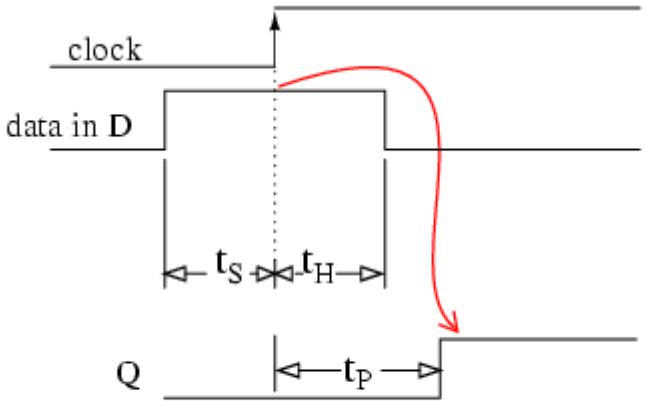
The **D** register does not see a one until time t_2, at which time Q goes high.



Data present t_H before clock time at D is transfered to Q.

Since data, above, present at **D** is clocked to **Q** at clock time, and **Q** cannot change until the next clock time, the **D** FF delays data by one clock period, provided that the data is already synchronized to the clock. The Q_A waveform is the same as "data in" with a one clock period delay.

A more detailed look at what the input of the type **D** Flip-Flop sees at clock time follows. Refer to the figure below. Since "data in" appears to changes at clock time (above), we need further information to determine what the **D** FF sees. If the "data in" is from another shift register stage, another same type **D** FF, we can draw some conclusions based on *data sheet* information. Manufacturers of digital logic make available information about their parts in data sheets, formerly only available in a collection called a *data book*. Data books are still available; though, the manufacturer's web site is the modern source.



Data must be present (t_S) before the clock and after(t_H) the clock. Data is delayed from D to Q by propagation delay (t_P)

The following data was extracted from the CD4006b data sheet for operation at $5V_{DC}$, which serves as an example to illustrate timing.
[*]

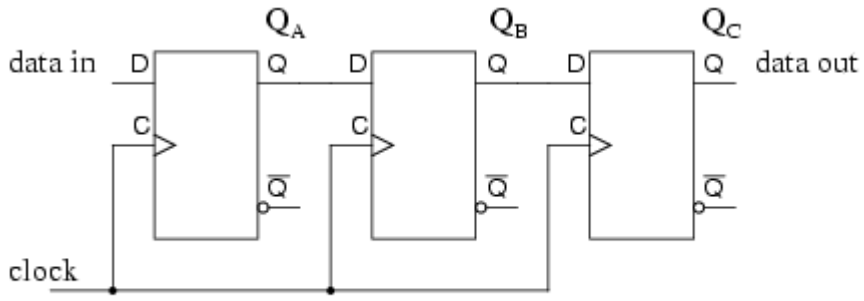
- t_S=100ns

- t_H=60ns
- t_P=200-400ns typ/max

t_S is the *setup time*, the time data must be present before clock time. In this case data must be present at **D** 100ns prior to the clock. Furthermore, the data must be held for *hold time* t_H=60ns after clock time. These two conditions must be met to reliably clock data from **D** to **Q** of the Flip-Flop.

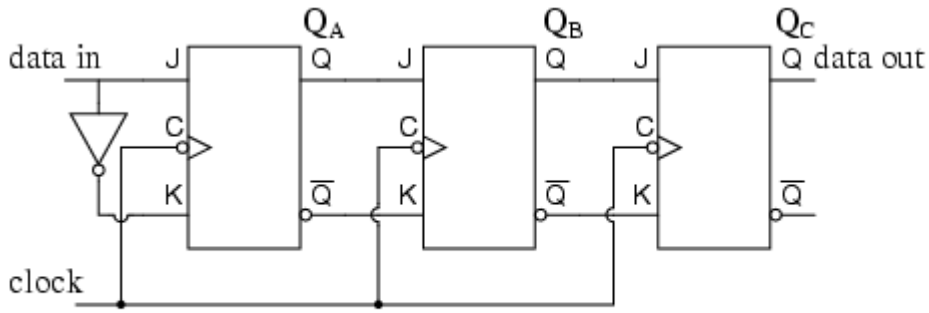
There is no problem meeting the setup time of 60ns as the data at **D** has been there for the whole previous clock period if it comes from another shift register stage. For example, at a clock frequency of 1 Mhz, the clock period is 1000 µs, plenty of time. Data will actually be present for 1000µs prior to the clock, which is much greater than the minimum required t_S of 60ns.

The hold time t_H=60ns is met because D connected to Q of another stage cannot change any faster than the propagation delay of the previous stage t_P=200ns. Hold time is met as long as the propagation delay of the previous **D** FF is greater than the hold time. Data at **D** driven by another stage **Q** will not change any faster than 200ns for the CD4006b.
To summarize, output **Q** follows input D at nearly clock time if Flip-Flops are cascaded into a multi-stage shift register.



Serial-in, serial-out shift register using type "D" storage elements

Three type **D** Flip-Flops are cascaded Q to D and the clocks paralleled to form a three stage shift register above.

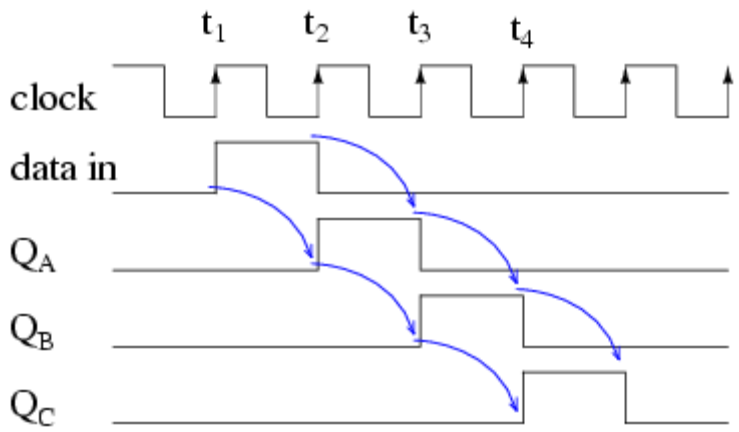


Serial-in, serial-out shift register using type "JK" storage elements

Type **JK** FFs cascaded Q to J, Q' to K with clocks in parallel to yield an alternate form of the shift register above.

A serial-in/serial-out shift register has a clock input, a data input, and a data output from the last stage. In general, the other stage outputs are not available Otherwise, it would be a serial-in, parallel-out shift register..

The waveforms below are applicable to either one of the preceding two versions of the serial-in, serial-out shift register. The three pairs of arrows show that a three stage shift register temporarily stores 3-bits of data and delays it by three clock periods from input to output.



At clock time t_1 a "data in" of **0** is clocked from **D** to **Q** of all three stages. In particular, **D** of stage **A** sees a logic **0**, which is clocked to Q_A where it remains until time t_2.

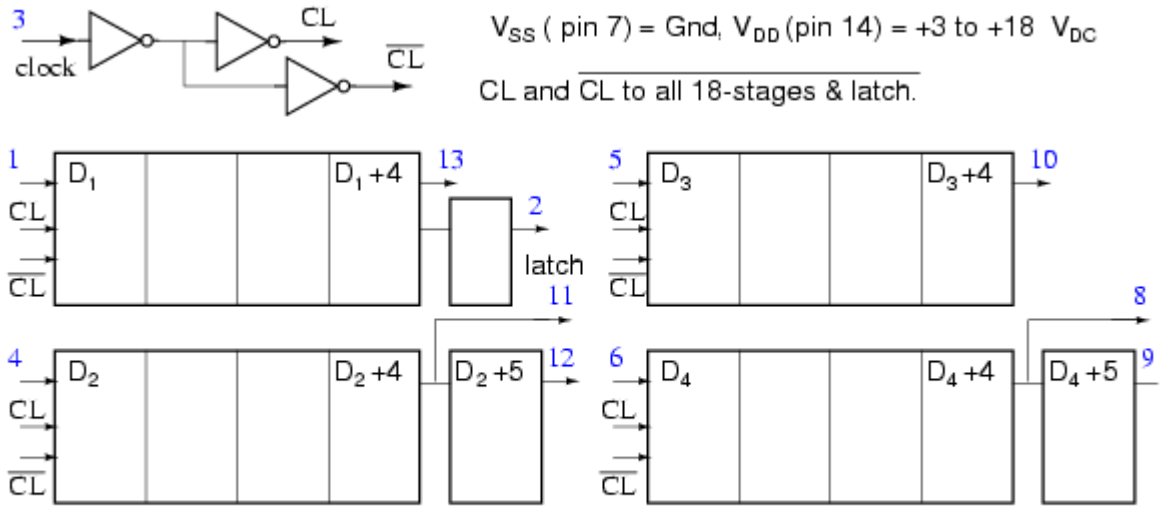
At clock time t_2 a "data in" of **1** is clocked from **D** to Q_A. At stages **B** and **C**, a **0**, fed from preceding stages is clocked to Q_B and Q_C.
At clock time t_3 a "data in" of **0** is clocked from **D** to Q_A. Q_A goes low and stays low for the remaining clocks due to "data in" being **0**. Q_B goes high at t_3 due to a **1** from the previous stage. Q_C is still low after t_3 due to a low from the previous stage.
Q_C finally goes high at clock t_4 due to the high fed to **D** from the previous stage Q_B.
All earlier stages have **0**s shifted into them. And, after the next clock pulse at t_5, all logic **1**s will have been shifted out, replaced by **0**s.

Serial-in/serial-out devices

We will take a closer look at the following parts available as integrated circuits, courtesy of Texas Instruments. For complete device data sheets follow the links.

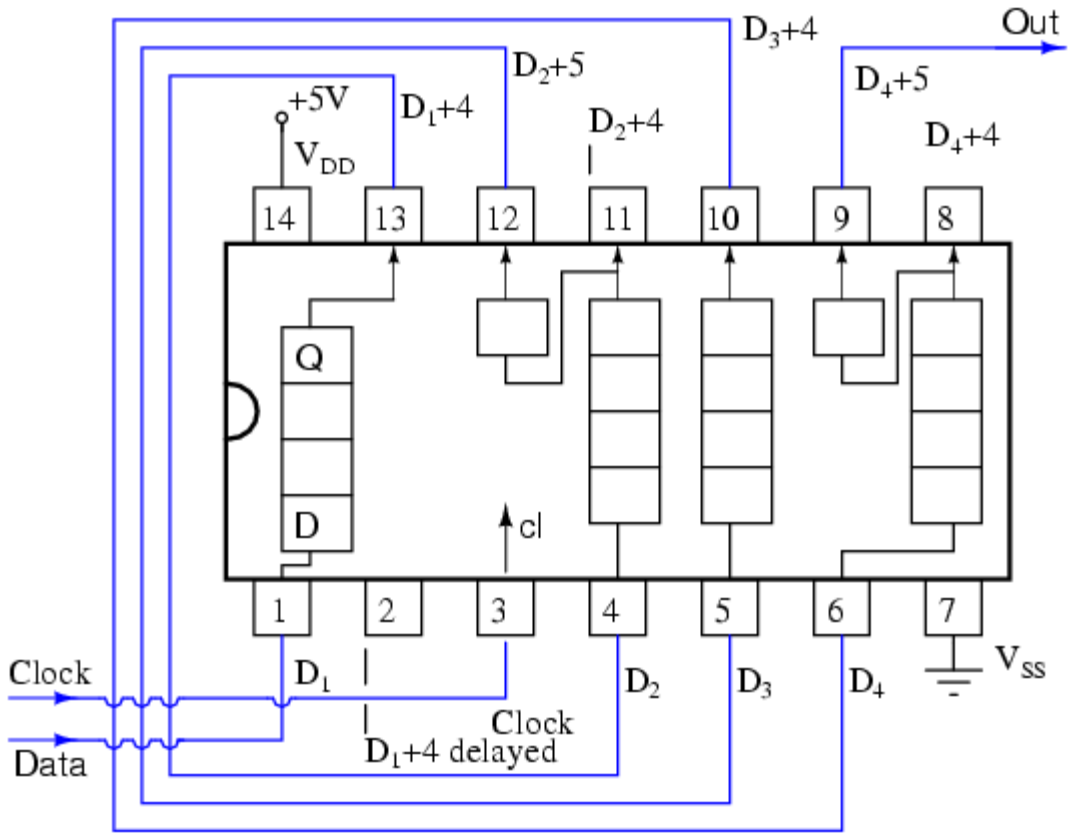
- CD4006b 18-bit serial-in/ serial-out shift register
  [*]
- CD4031b 64-bit serial-in/ serial-out shift register
  [*]
- CD4517b dual 64-bit serial-in/ serial-out shift register
  [*]

The following serial-in/ serial-out shift registers are 4000 series *CMOS* (Complementary Metal Oxide Semiconductor) family parts. As such, They will accept a V_{DD}, positive power supply of 3-Volts to 15-Volts. The V_{SS} pin is grounded. The maximum frequency of the shift clock, which varies with V_{DD}, is a few megahertz. See the full data sheet for details.



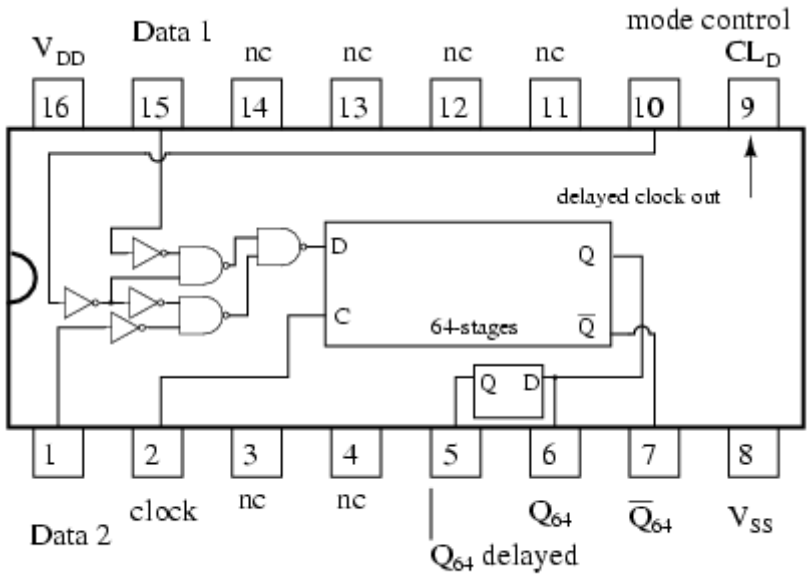
CD4006b Serial-in/ serial-out shift register

The 18-bit CD4006b consists of two stages of 4-bits and two more stages of 5-bits with a an output tap at 4-bits. Thus, the 5-bit stages could be used as 4-bit shift registers. To get a full 18-bit shift register the output of one shift register must be cascaded to the input of another and so on until all stages create a single shift register as shown below.



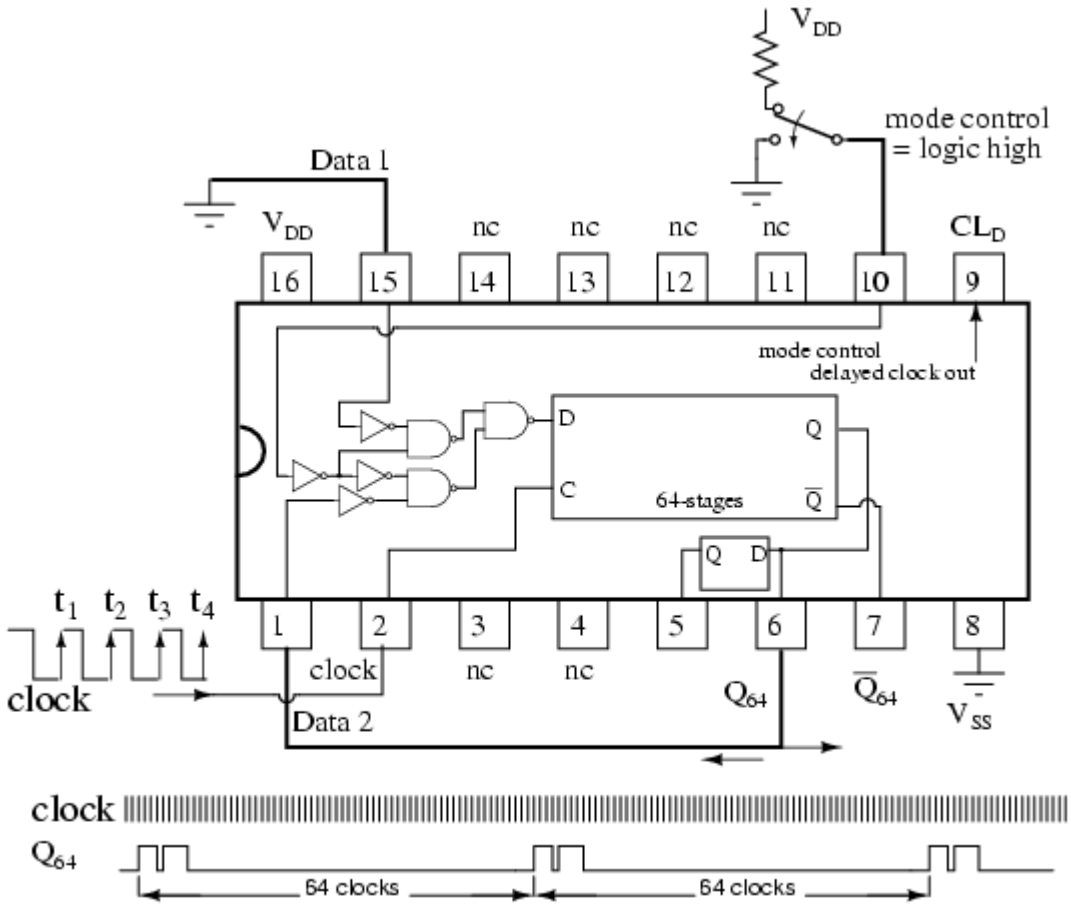
CD4006b 18-bit serial-in/ serial-out shift register

A CD4031 64-bit serial-in/ serial-out shift register is shown below. A number of pins are not connected (nc). Both Q and Q' are available from the 64th stage, actually $Q_{64}$and Q'_{64}. There is also a Q_{64} "delayed" from a half stage which is delayed by half a clock cycle. A major feature is a data selector which is at the data input to the shift register.



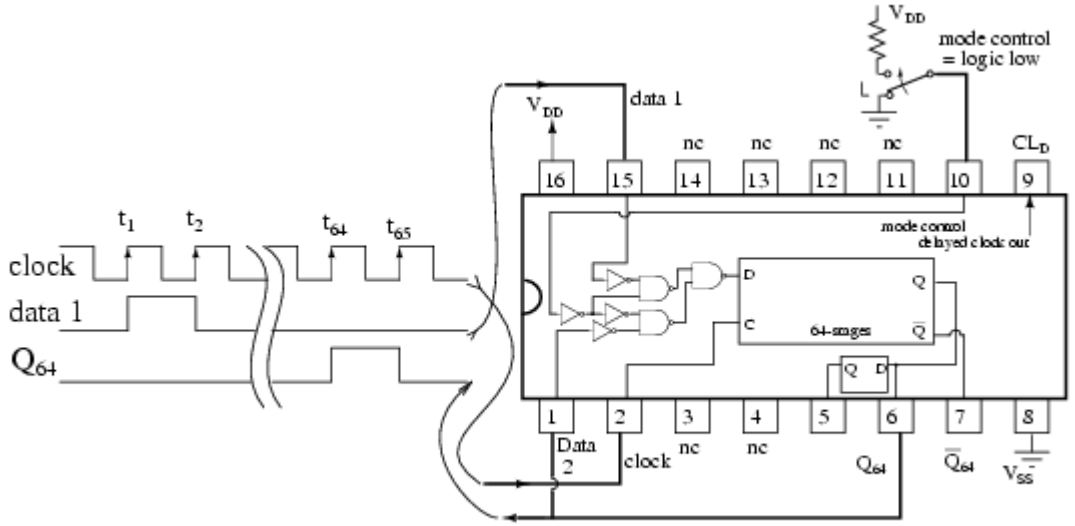
CD4031 64-bit serial-in/ serial-out shift register

The "mode control" selects between two inputs: data 1 and data 2. If "mode control" is high, data will be selected from "data 2" for input to the shift register. In the case of "mode control" being logic low, the "data 1" is selected. Examples of this are shown in the two figures below.



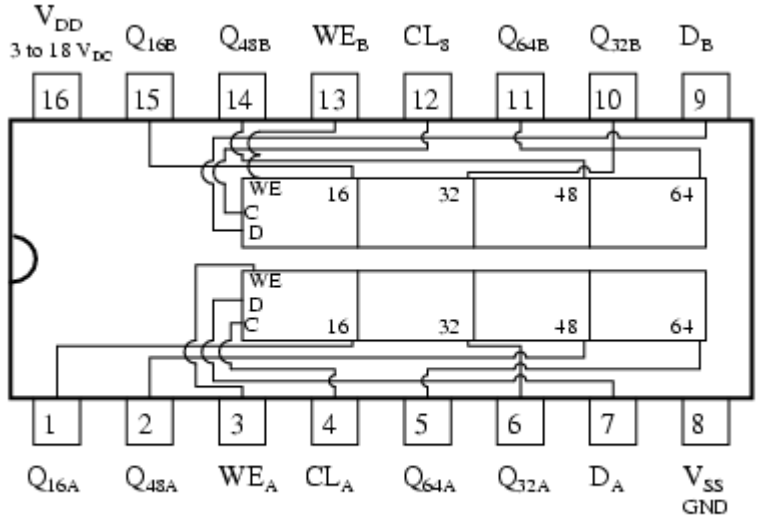
CD4031 64-bit serial-in/ serial-out shift register recirculating data.

The "data 2" above is wired to the Q_{64} output of the shift register. With "mode control" high, the Q_{64} output is routed back to the shifter data input D. Data will *recirculate*from output to input. The data will repeat every 64 clock pulses as shown above. The question that arises is how did this data pattern get into the shift register in the first place?



CD4031 64-bit serial-in/ serial-out shift register load new data at Data 1.

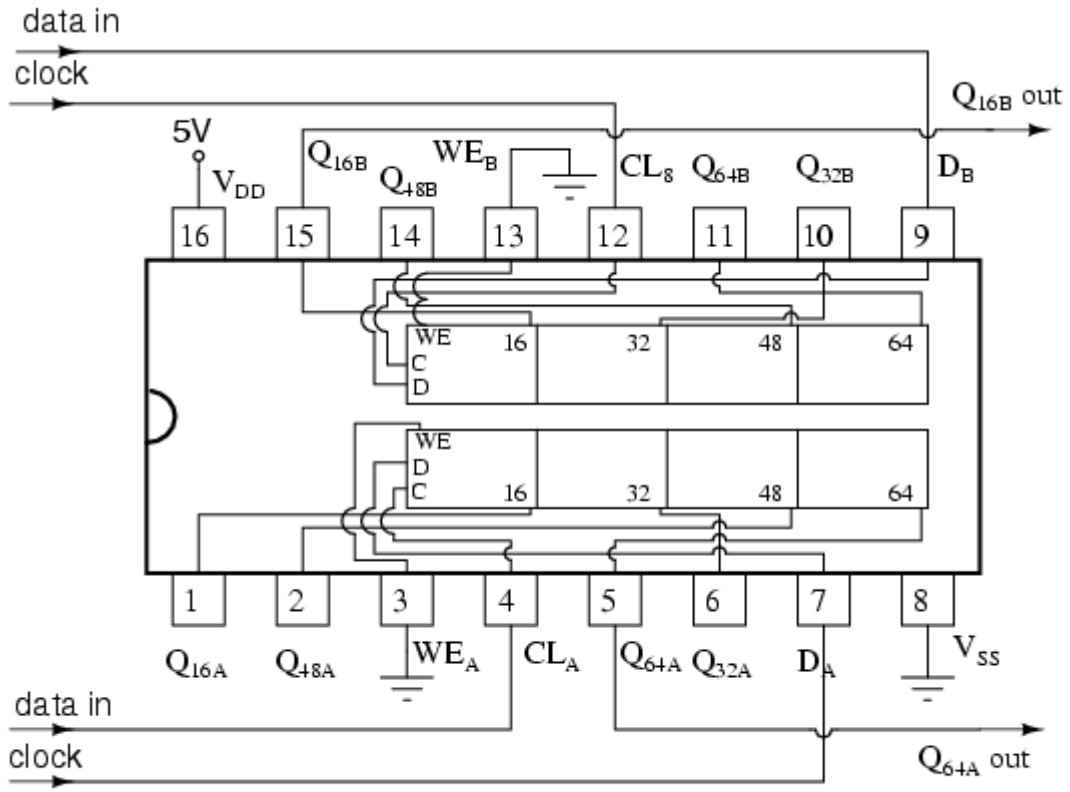
With "mode control" low, the CD4031 "data 1" is selected for input to the shifter. The output, Q_{64}, is not recirculated because the lower data selector gate is *disabled*. By disabled we mean that the logic low "mode select" inverted twice to a low at the lower NAND gate prevents it for passing any signal on the lower pin (data 2) to the gate output. Thus, it is disabled.



CD4517b dual 64-bit serial-in/ serial-out shift register

A CD4517b dual 64-bit shift register is shown above. Note the taps at the 16th, 32nd, and 48th stages. That means that shift registers of those lengths can be configured from one of the 64-bit shifters. Of course, the 64-bit shifters may be cascaded to yield an 80-bit, 96-bit, 112-bit, or 128-bit shift register. The clock CL_A and CL_Bneed to be paralleled when cascading the two shifters. WE_B and WE_B are grounded for normal shifting operations. The data inputs to the shift registers A and B are $D_A$and D_B respectively.

Suppose that we require a 16-bit shift register. Can this be configured with the CD4517b? How about a 64-shift register from the same part?



CD4517b dual 64-bit serial-in/ serial-out shift register, wired for 16-shift register, 64-bit shift register

Above we show A CD4517b wired as a 16-bit shift register for section B. The clock for section B is CL_B. The data is clocked in at CL_B. And the data delayed by 16-clocks is picked of off Q_{16B}. WE_B , the write enable, is grounded.
Above we also show the same CD4517b wired as a 64-bit shift register for the independent section A. The clock for section A is CL_A. The data enters at CL_A. The data delayed by 64-clock pulses is picked up from Q_{64A}. WE_A, the write enable for section A, is grounded.

Source: http://www.allaboutcircuits.com/vol_4/chpt_12/2.html