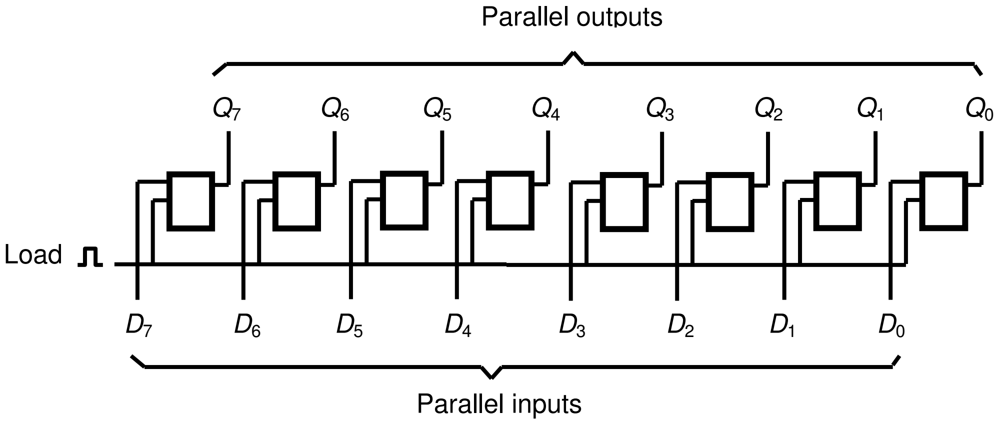
Tutorial Solution

24.11 A suitable circuit is below. The devices are D master/slave flip-flops. The various inputs go the D input in each case, the 'load' input goes to the clock of each device, and the outputs come from the Q outputs of each device.

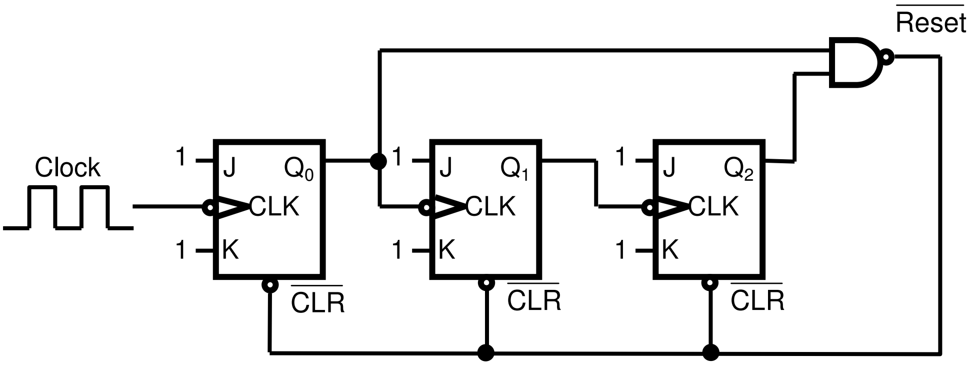


24.12 This is described in Section 24.7 of the text.

24.13 In a synchronous counter all the elements change state at the same time, this being determined by a clock signal. This ensures that a short time after the clock signal changes, all the stages will have responded and the counter may be read.

In an asynchronous counter the output of one stage forms the input of the next. Thus changes ripple through the counter with each stage producing a slight delay. This arrangement is simpler than a synchronous counter but has problems at high speeds.

24.14 A suitable circuit is shown below.

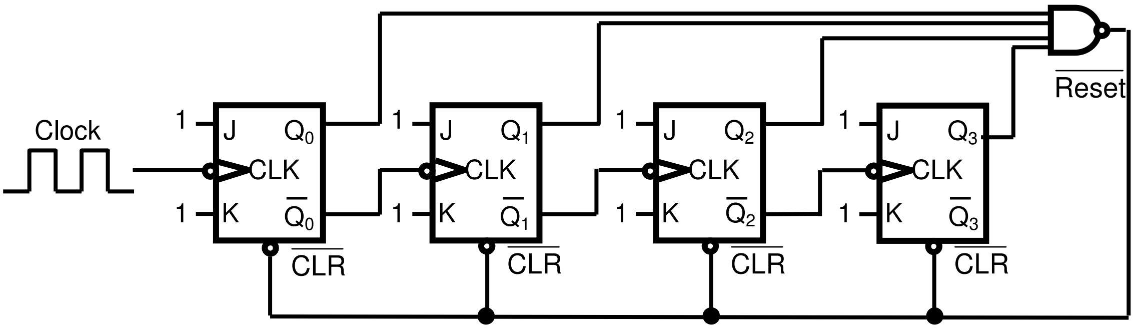


24.15 The circuit for this simulation can be easily produced by modifying the demonstration file for Computer Simulation Exercise 24.2.

24.16 A suitable circuit may be formed by adding a divide-by-two stage at the end of the modulo-5 counter described in the last exercise.

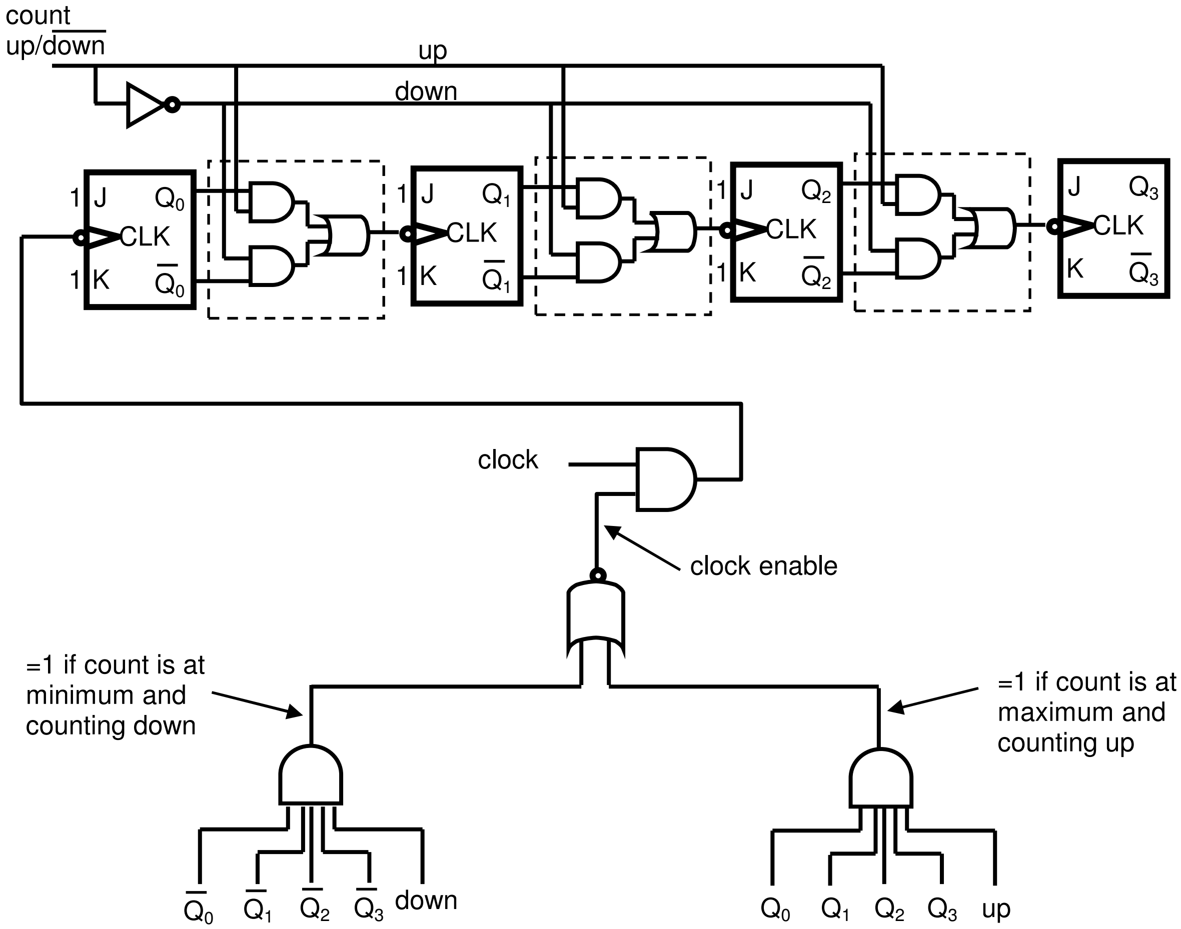
24.17 The above circuit can be simulated very easily. Enter the circuit using 7473 J-K flip-flops and a 7400 2-input NAND gate. Use a digital source for the clock input. Suitable parameters for this digital signal source are DELAY = 0.5us, ONTIME = 0.5us and OFFTIME = 0.5us. Hi logic nodes can be used to set all the J and K input to logic 1. The CLR input of the output flip-flop should also be set Hi. Set up the analysis options to perform a transient analysis. Suitable parameters are a Print Step of 200ns and a Final Time of 12 us. Put level markers on the outputs of each flip-flop and on the reset signal. Then run the analysis and observe the outputs. This should show a square wave on the final output with a period of 10us as expected.

24.18 The functionality required can be achieved by combining the techniques used to generate the decade up-counter, with those of the ripple down-counter. This takes a four bit down counter and adds circuitry to detect the number following the last required number, and uses the occurrence of this number to reset the system into the first number of the sequence required. In this example the last number of the sequence is zero and the following number is therefore 15 (all 1's). Therefore circuitry is added to detect the number 15 and to reset the counter to 9. The resultant circuit is shown below.



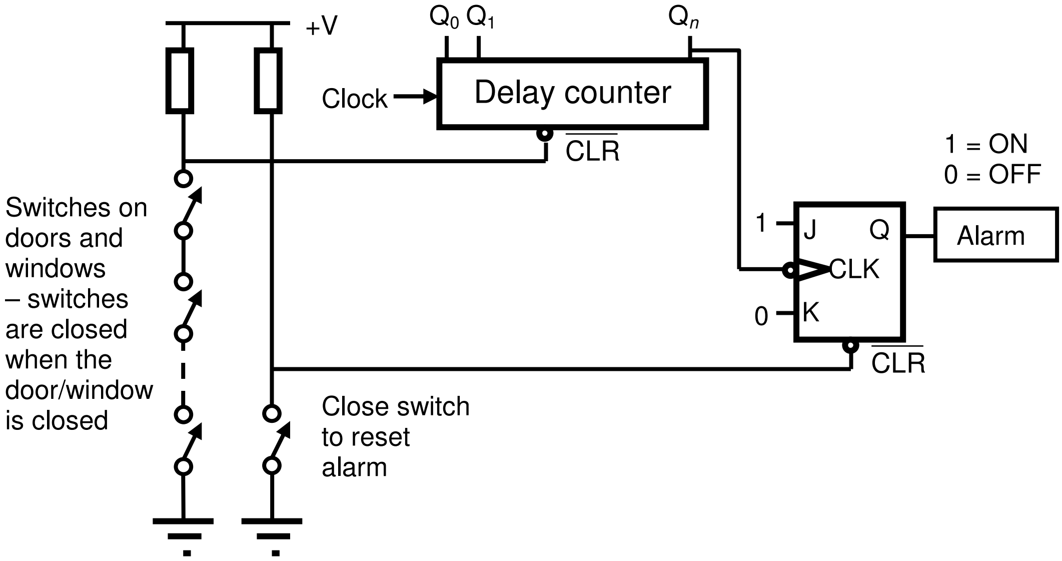
24.19 If the above circuit is simulated (using an approach similar to that outlined for exercise 24.17 above) it will be found that the circuit does *not* operate correctly. The problems encountered are related to *races* within the circuit caused by the small delays introduced by each device. Our simple theory would suggest that the reset output of the NAND gate should be low only after the counter has counted down to zero and then moved on to a count of all 1's. However, if the output of this gate is displayed it will be seen that the output goes low at other times during the counting process. This is caused by the finite time that it takes for each flip-flop to change state as discussed in Section 24.2.7. Although the fairly simply circuits given in the text give an insight into the design of various forms of counters, the circuits used within integrated circuit counters are invariably slightly more complicated to deal with issues such as race hazards. This example provides a useful illustration of some of the limitations of these simple circuits.

24.20 The circuit below shows a possible solution.


count
up/$\overline{\text{down}}$
up
down
1 J Q0
CLK
1 K $\overline{Q_0}$
1 J Q1
CLK
1 K $\overline{Q_1}$
1 J Q2
CLK
1 K $\overline{Q_2}$
J Q3
CLK
K $\overline{Q_3}$
clock
clock enable
=1 if count is at minimum and counting down
=1 if count is at maximum and counting up
$\overline{Q_0}$ $\overline{Q_1}$ $\overline{Q_2}$ $\overline{Q_3}$ down
Q0 Q1 Q2 Q3 up

Such a counter might be used in an application where a gain or level is being set. For example a digital volume control where one might push a button to increase or decrease the volume. One would not want the volume to go to minimum if one continued to push the up button when maximum had been reached.

24.21 The diagram below shows a possible solution.


+V
Q_0 Q_1
Q_n
Clock
Delay counter
$\overline{CLR}$
1 = ON
0 = OFF
Switches on doors and windows – switches are closed when the door/window is closed
1
J
Q
Alarm
CLK
0
K
$\overline{CLR}$
Close switch to reset alarm

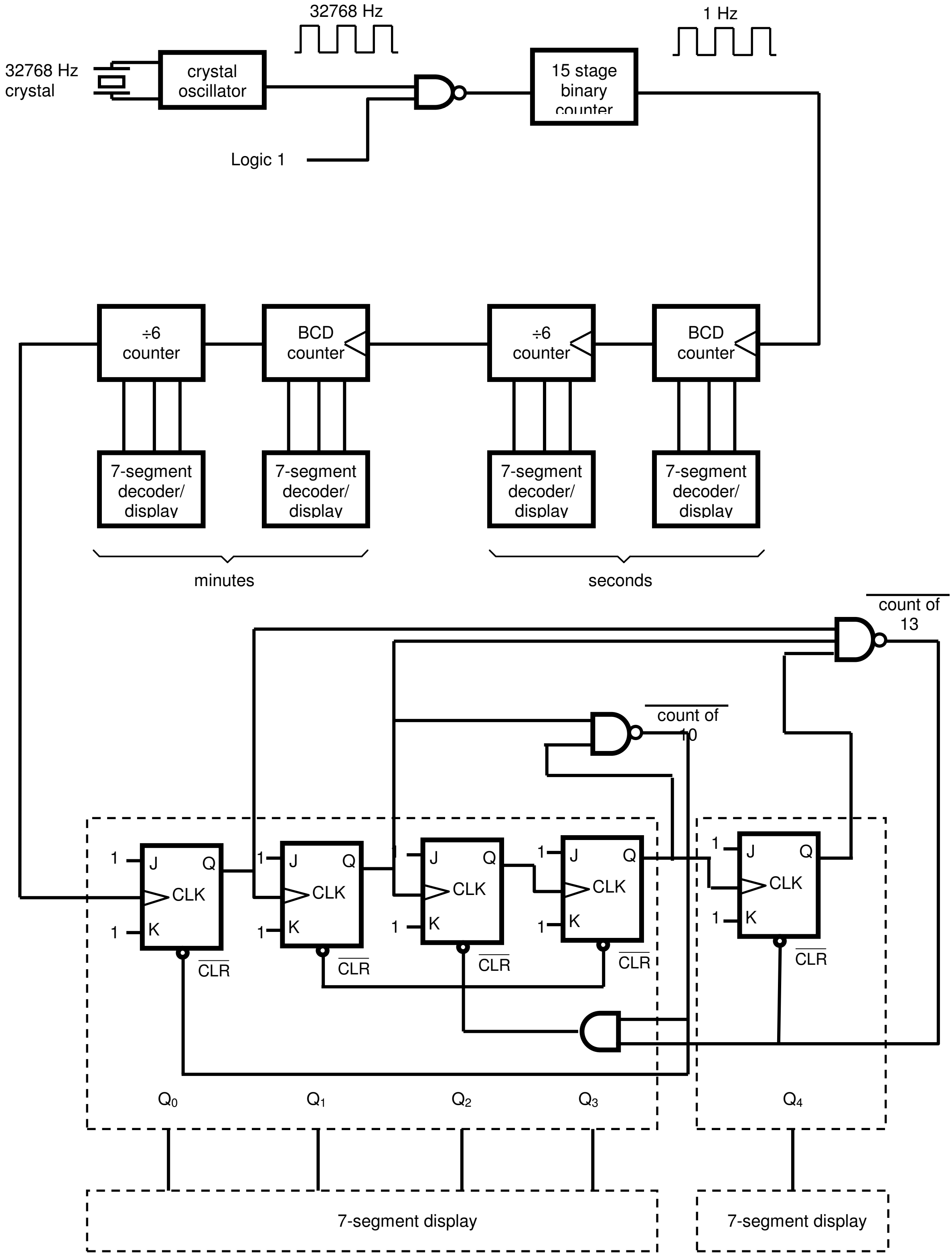
Here the S-R bistable has been replaced by a clocked J-K device and a counter has been added to provide a 30 second delay. The delay is in the form of a counter which counts pulses from an oscillator. The counter and oscillator frequency are chosen such that if the counter starts from zero the most significant bit of the counter, Q_n, is set after 30 seconds.

The Q_n output from the counter acts as a clock to a J-K bistable which is arranged such that a positive transition on the clock will set the Q output to 1. This output is connected to the alarm. The various door and window switches are connected in series with a pull-up resistor and the combination is connected to the $\overline{\text{CLR}}$ input of the counter. Therefore while the doors and windows are closed the counter is held with all its outputs at zero. If now one of the switches is opened the counter will no longer be held in its reset mode and will begin to count up.

After 30 seconds the Q_n output will go high and the positive going transition will act as a clock trigger to the bistable setting the Q output high and turning ON the alarm. Either before or after the alarm has been triggered the bistable may be cleared by closing the reset switch. This will turn OFF the alarm if it is sounding, or prevent it from sounding if it is left closed.

This simple arrangement fulfils the requirements given in the Exercise, but is perhaps not an ideal solution since a door may be opened and closed quickly without triggering the alarm. A more sophisticated design would therefore be adopted. It is worth noting that even at this relatively low level of complexity a real system would probably use a single-chip microcomputer rather than discrete counters and bistables. This would allow much more sophisticated facilities to be produced in software.

24.22 A suitable clock may be formed by extending the arrangement shown in Example 24.6, to include a counter and display for the hours function.

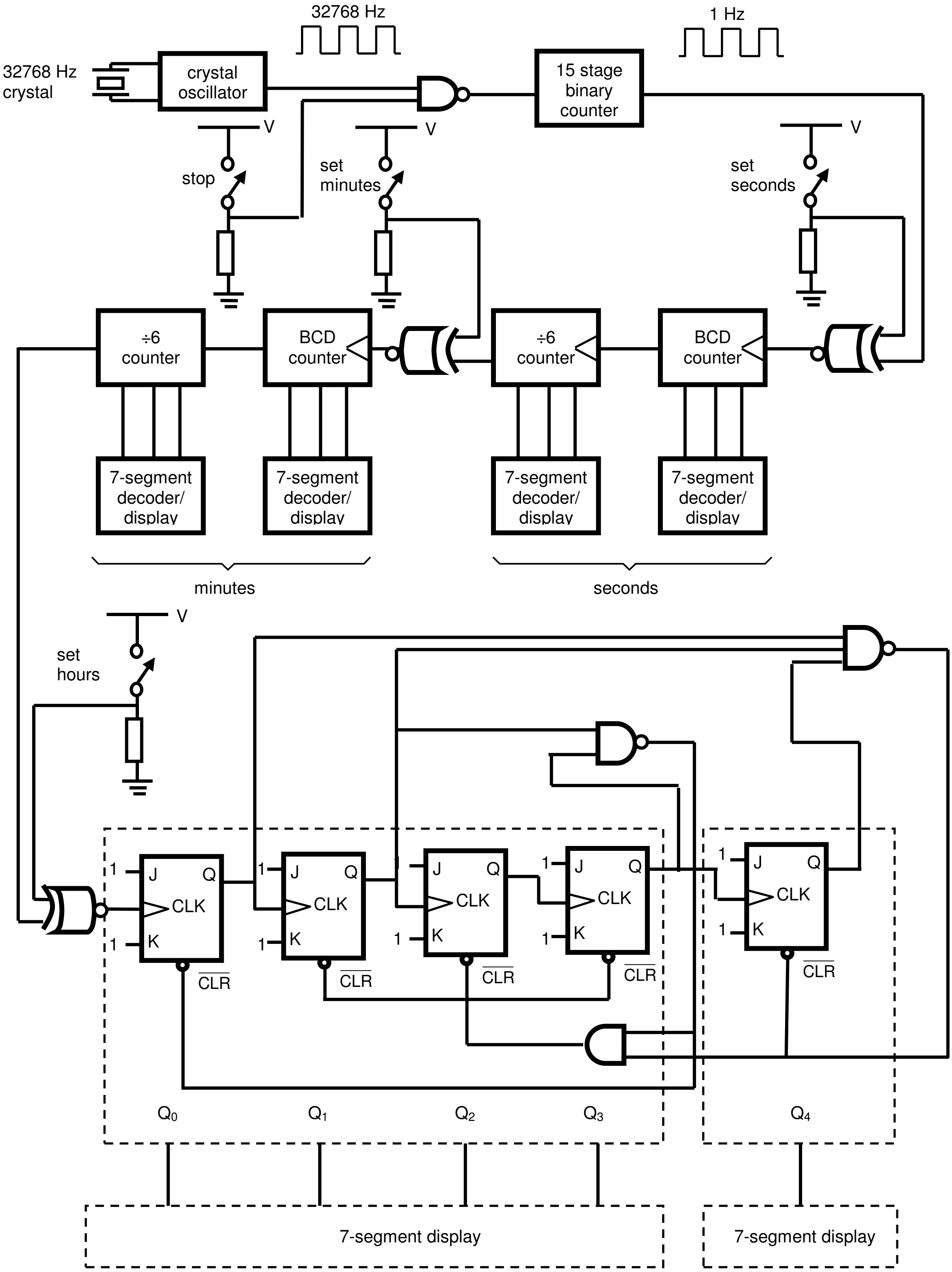


The circuit is driven from the output of the minutes stage in the same way that the minutes are driven from the seconds stage. Since we are now considering a clock rather than a stop-watch, the stop, start and reset functions are inappropriate. We will consider setting of the clock in the next exercise.

The hours section requires a two stage counter to measure units and tens of hours. The unusual feature of this stage is that the counter must count from 1 to 12 rather than from 0 to 11. This is achieved using a modified version of the divide-by-N counter where a count of 13 is detected and used to clear the bits of the count. In this case the clear is applied to all bits except the least significant bit. Since this bit is set to 1 for a count of 13 the result is that the counter is reset to 1 rather than to 0. This arrangement is shown overleaf.

The least significant bit of the 4-bit counter is cleared when a count of ten is detected but not when a count of 13 is reached. The other bits of the 4-bit counter a cleared both at a count of 10 and a count of 13. The divide by 2 stage is cleared when 13 is reached.

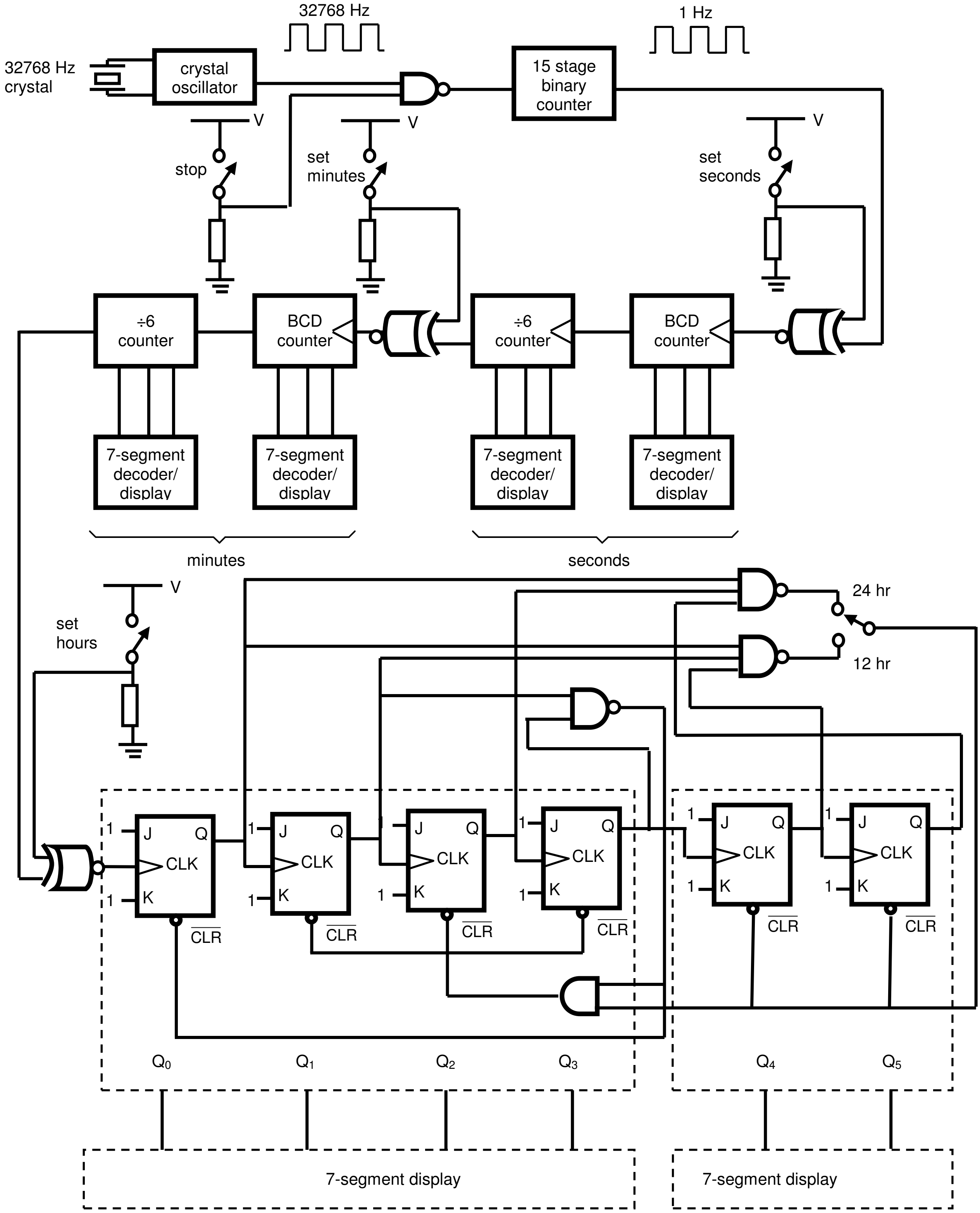
24.23 One method of incrementing the seconds, minutes and hours settings of the clock of the last exercise is to use pushbuttons as shown below.



It is useful to be able to stop the normal timing function of the clock while setting the time. This is achieved by gating the oscillator output through a NAND gate with one input to this gate coming from a switch. Incrementing the individual stages of the clock is achieved by adding an exclusive-OR gate at the input of each stage. One of the inputs to each of these gates comes from a normally-open push-button. When the switch is open the exclusive-OR gate has a 0 on one of its inputs and it therefore passes the other input to its output unchanged. If the push-button is now depressed the output signal will be inverted (irrespective of its state). Thus depressing and releasing the push-button will cause the input to the corresponding stage to see an input which changes state and then reverts to its original state. This will always produce a single positive going edge which will increment that stage by one. The process is repeated until the appropriate time is reached.

Although the above design is appropriate for students at this level it not an ideal solution. Firstly, incrementing one stage of the counter through its maximum value and back to zero will have the effect of incrementing the next stage. Also, the switches will certainly exhibit switch bounce which will result in multiple increments for one depression of the switch. It is worth pointing out these deficiencies but probably not worth attempting to overcome them within the design. In practise a design of this complexity would not be performed using discrete gates.

24.24 Some 24 hour clocks display hours in the range 1 to 24 and others in the range 0 to 23. For simplicity in this exercise we will assume the former operation. In order to count up to 24 hours an additional bistable must be added to the tens counter of the hours stage. The only other modification required to our clock is that additional circuitry must be added to detect a count of 25, and the reset signal to both stages must be switchable between this detector and that detecting a count of 13. This arrangement is shown overleaf.



24.25 These problems are discussed in Section 24.8.5 and synchronous counters are discussed in Section 24.8.6.