Digital electronics assignment

Digital electronics is classified into combinational logic and sequential logic. Combinational logic output depends on the inputs levels, whereas sequential logic output depends on stored levels and also the input levels. [pic] [pic] [pic] The memory elements are devices capable of storing binary info. The binary info stored in the memory elements at any given time defines the state of the sequential circuit. The input and the present state of the memory element determines the output.

Memory elements next state is also a function of external inputs and present state. A sequential circuit is specified by a time sequence of inputs, outputs, and internal states. [pic] There are two types of sequential circuits. Their classification depends on the timing of their signals: [pic] ??? Synchronous sequential circuits ??? Asynchronous sequential circuits [pic] [pic][pic][pic] [pic] [pic]Asynchronous sequential circuit This is a system whose outputs depend upon the order in which its input variables change and can be affected at any instant of time. pic] Gate-type asynchronous systems are basically combinational circuits with feedback paths. Because of the feedback among logic gates, the system may, at times, become unstable. Consequently they are not often used. [pic] [pic] [pic] [pic]Synchronous sequential circuits This type of system uses storage elements called flip-flops that are employed to change their binary value only at discrete instants of time. Synchronous sequential circuits use logic gates and flip-flop storage devices. Sequential circuits have a clock signal as one of their inputs.

All state transitions in such circuits occur only when the clock value is either 0 or 1 or happen at the rising or falling edges of the clock depending on the type of memory elements used in the circuit. Synchronization is achieved by a timing device called a clock pulse generator. Clock pulses are distributed throughout the system in such a way that the flip-flops are affected only with the arrival of the synchronization pulse. Synchronous sequential circuits that use clock pulses in the inputs are called clocked-sequential circuits.

They are stable and their timing can easily be broken down into independent discrete steps, each of which is considered separately. [pic] [pic] [pic] A clock signal is a periodic square wave that indefinitely switches from 0 to 1 and from 1 to 0 at fixed intervals. Clock cycle time or clock period: the time interval between two consecutive rising or falling edges of the clock. [pic] Clock Frequency = 1 / clock cycle time (measured in cycles per second or Hz) [pic] Example: Clock cycle time = 10ns clock frequency = 100Mhz Concept of Sequential Logic

A sequential circuit as seen in the last page, is combinational logic with some feedback to maintain its current value, like a memory cell. To understand the basics let’s consider the basic feedback logic circuit below, which is a simple NOT gate whose output is connected to its input. The effect is that output oscillates between HIGH and LOW (i. e. 1 and 0). Oscillation frequency depends on gate delay and wire delay. Assuming a wire delay of 0 and a gate delay of 10ns, then oscillation frequency would be (on time + off time = 20ns) 50Mhz. [pic] [pic] [pic]

The basic idea of having the feedback is to store the value or hold the value, but in the above circuit, output keeps toggling. We can overcome this problem with the circuit below, which is basically cascading two inverters, so that the feedback is in-phase, thus avoids toggling. The equivalent circuit is the same as having a buffer with its output connected to its input. [pic] [pic] [pic] But there is a problem here too: each gate output value is stable, but what will it be? Or in other words buffer output can not be known. There is no way to tell. If we could know or set the value we would have a simple 1-bit storage/memory element. pic] The circuit below is the same as the inverters connected back to back with provision to set the state of each gate (NOR gate with both inputs shorted is like a inverter). I am not going to explain the operation, as it is clear from the truth table. S is called set and R is called Reset. [pic] [pic] [pic] | S | R | Q | Q+ | | 0 | 0 | 0 | 0 | | 0 | 0 1 | 1 | | 0 | 1 | X | 0 | | 1 | 0 | X | 1 | | 1 | 1 | X | 0 | [pic]

There still seems to be some problem with the above configuration, we can not control when the input should be sampled, in other words there is no enable signal to control when the input is sampled. Normally input enable signals can be of two types. [pic] ??? Level Sensitive or (LATCH) ??? Edge Sensitive or (Flip-Flop) [pic] Level Sensitive: The circuit below is a modification of the above one to have level sensitive enable input. Enable, when LOW, masks the input S and R. When HIGH, presents S and R to the sequential logic input (the above circuit two NOR Gates).

Thus Enable, when HIGH, transfers input S and R to the sequential cell transparently, so this kind of sequential circuits are called transparent Latch. The memory element we get is an RS Latch with active high Enable. [pic] [pic] [pic] Edge Sensitive: The circuit below is a cascade of two level sensitive memory elements, with a phase shift in the enable input between first memory element and second memory element. The first RS latch (i. e. the first memory element) will be enabled when CLK input is HIGH and the second RS latch will e enabled when CLK is LOW. The net effect is input RS is moved to Q and Q’ when CLK changes state from HIGH to LOW, this HIGH to LOW transition is called falling edge. So the Edge Sensitive element we get is called negative edge RS flip-flop. [pic] [pic] [pic] Now that we know the sequential circuits basics, let’s look at each of them in detail in accordance to what is taught in colleges. You are always welcome to suggest if this can be written better in any way. Latches and Flip-Flops There are two types types of sequential circuits. pic] ??? Asynchronous Circuits. ??? Synchronous Circuits. [pic] As seen in last section, Latches and Flip-flops are one and the same with a slight variation: Latches have level sensitive control signal input and Flip-flops have edge sensitive control signal input. Flip-flops and latches which use this control signals are called synchronous circuits. So if they don’t use clock inputs, then they are called asynchronous circuits. [pic] [pic]RS Latch RS latch have two inputs, S and R. S is called set and R is called reset.

The S input is used to produce HIGH on Q (i. e. store binary 1 in flip-flop). The R input is used to produce LOW on Q (i. e. store binary 0 in flip-flop). Q’ is Q complementary output, so it always holds the opposite value of Q. The output of the S-R latch depends on current as well as previous inputs or state, and its state (value stored) can change as soon as its inputs change. The circuit and the truth table of RS latch is shown below. (This circuit is as we saw in the last page, but arranged to look beautiful ????). [pic] [pic] [pic] S | R | Q | Q+ | | 0 | 0 | 0 | 0 | | 0 | 0 | 1 | 1 | | 0 | 1 | X | 0 | | 1 | 0 | X | 1 | | 1 | 1 | X | 0 | [pic] The operation has to be analyzed with the 4 inputs combinations together with the 2 possible previous states. [pic] ??? When S = 0 and R = 0: If we assume Q = 1 and Q’ = 0 as initial condition, then output Q after input is applied would be Q = (R + Q’)’ = 1 and Q’ = (S + Q)’ = 0. Assuming Q = 0 and Q’ = 1 as initial condition, then output Q after the input applied would be Q = (R + Q’)’ = 0 and Q’ = (S + Q)’ = 1. So it is clear that when both S and R inputs are LOW, the output is retained as before the application of inputs. (i. e. there is no state change). When S = 1 and R = 0: If we assume Q = 1 and Q’ = 0 as initial condition, then output Q after input is applied would be Q = (R + Q’)’ = 1 and Q’ = (S + Q)’ = 0. Assuming Q = 0 and Q’ = 1 as initial condition, then output Q after the input applied would be Q = (R + Q’)’ = 1 and Q’ = (S + Q)’ = 0. So in simple words when S is HIGH and R is LOW, output Q is HIGH. ??? When S = 0 and R = 1: If we assume Q = 1 and Q’ = 0 as initial condition, then output Q after input is applied would be Q = (R + Q’)’ = 0 and Q’ = (S + Q)’ = 1. Assuming Q = 0 and Q’ = 1 as initial condition, then output Q after the input applied would be Q = (R + Q’)’ = 0 and Q’ = (S + Q)’ = 1. So in simple words when S is LOW and R is HIGH, output Q is LOW. When S = 1 and R = 1 : No matter what state Q and Q’ are in, application of 1 at input of NOR gate always results in 0 at output of NOR gate, which results in both Q and Q’ set to LOW (i. e. Q = Q’). LOW in both the outputs basically is wrong, so this case is invalid. [pic] The waveform below shows the operation of NOR gates based RS Latch. [pic] [pic] [pic] It is possible to construct the RS latch using NAND gates (of course as seen in Logic gates section). The only difference is that NAND is NOR gate dual form (Did I say that in Logic gates section?). So in this case the R = 0 and S = 0 case becomes the invalid case. The circuit and Truth table of RS latch using NAND is shown below. [pic] [pic] [pic] S | R | Q | Q+ | | 1 | 1 | 0 | 0 | | 1 | 1 | 1 | 1 | | 0 | 1 | X | 0 | | 1 | 0 | X | 1 | | 0 | 0 | X | 1 | [pic] If you look closely, there is no control signal (i. e. no clock and no enable), so this kind of latches or flip-flops are called asynchronous logic elements. Since all the sequential circuits are built around the RS latch, we will concentrate on synchronous circuits and not on asynchronous circuits.