Help us translate!And the multiplexer basically, basically picks the previous output from the ori, from the old value of the DFF.Take the previous output and fit it into another input of the multiplexer and then load, choose between them, this is exactly the correct functionality.And from this we'll need all the kinds of the sequential circuitry that we need, memories, counters, and so on. One thing that I'd like to mention at this point is the meaning of the little triangles that we see at the bottom, at the bottom of the D flip flop diagram.So this doesn't quite work, really the way to combine these two possible sources into the D flip-flop, one source which is an output from the previous stage, and another source which is a new input.As opposed to all previous chips are all combinatorial chips that we had so far, whose output only depended on their own inputs at any given point in time.The combinatorial logic which happens instantaneously and the sequential logic which we don't want to, to thinking about how it, how it is constructed from lower levels combinatorial stuff.In fact, there is a generic paradigm of how we're going to build all our logic in the computer, and it's going to be a combination of remembering information via this basic D flip flops.And see what our new implementation does when it's fed these two signals and inputs, and what it produces as output.0:03 In a previous unit, we discussed how we're dealing with time and sequential logic in computing systems.Now, if we look about that kind of thing in the way we're, we're in this abstract way that we're thinking about these discrete time units, at that exact time when we switch between time unit t minus 1 to time unit t. This new element must remember the bit.This flip flop has a single input and a single output and it basically remembers the input from last time unit and outputs it in the next time unit.So at time 1, we don't exactly know what our output will be because we haven't specified what happened in the previous time unit.Similarly at time unit 3, we know it has to be 0 because that was what was the input was at time 2, and so on. At every time point we are having simply the previous signal that was fed to the input, shifted one time unit to the right.You can actually construct the flip flop from Nand gates eh, when you actually take the Nand gate and put them into some kind of a loop eh, between each other.The first step is one, is what I've just described having this kind of cycle that allows it to somehow remember information.And then manipulating them using combinatorial logics that we built in the first two lectures.7:18 Their output is going to be fed into some combinatorial logic together with the new input that you get in this time unit.We will take the output, what we currently remember and let's plug it back into the input of the D flip flop.But we already know exactly how to combine two sources into one output and this is exactly the multiplexer.We know what load is, we know what in is, we can compute what the next situation is, and at every given point in time, we can write down the values and all the different wires just like we did, did in combinatorial logic.15:16 So at this point, we've looked, we've, we've looked at what is the basic unit that allows us to do sequential logic on a computer.What we'll do in this unit is actually talk about the actual elements, the chips that will allow us to do, provide this kind of functionality.We have lots of combinatorial log, logic so far that can do any kind of manipulations that we want within a single time unit.They flip to 0 and then can flop back to 1, and the point is that this flipping and flopping is something they remember.So let us view now the basic flip flops that we will you, be using in this course which is called the Clocked Data Flip Flop.3:02 Then when we look at what the D flip flop will do, at any time unit it will actually return the value that was in the input in the previous time unit.3:49 And this is going to be the only element that will provide all the sequential logic that we need in this course.This means that the implementation will also need some kind of access to this clock that we mentioned in the last unit, that actually breaks down the physical continuous time into discrete time units.But just, you are going to use it, just like you viewed the Nand gate and from D Flip Flop, and the Nand gate you will provide, you will actually build everything in the course.And with some additional logic you will also be able to manipulate this date from the outside.And the second type step is to actually have some kind of logic that actually provide isolation between subsequent time units.For example, if it's a counter we're going to remember a number in all these flip flops and the combinatorial logic will basically add one to the counter.So the D flip flop provided the basic functionality of remembering a bit for one time unit.When the load bit goes down to zero we want to the, the chip to keep remembering the last time that the, the last input that was loaded into it for infinity until a new load operation is performed.Before we look at that, let us see what this bit chip needs to do. So again, let's look at two possible signals for load and for in. So for example, we ask for loading in time units 1 and time unit 4, while input is, let's say it's 1 in the first time unit and then it goes down, down to 0.So we're going to keep on maintaining the same value for the next 2 time units until load goes up to 1 and then we'll have to see what we're going to do. 10:34 Now notice at for this time units, whatever input was, whether input went up or down, it does not matter.How do we put the real input in, if we want to also put the output of the D flip flop back into itself?We don't know what the previous state of the D flip flop was, what the previous out was because it's not specified in this example.To its eh, down to its lower input, we still can tell very well what the output is, what the output of the multiplexer is and that's going to be 1.13:42 Now eh, that we know what the input to the D flip flop at time unit 1 is, so even though we don't know the output at this point time unit we know everything about the input.In the next time step, the one that was the input of the D flip flop is going to be passed and is going to be the output of the D flip flop in the next time unit exactly when we switch between time units.We know what load is, we have the previous out so we can calculate exactly what the multiplexer does.15:00 And as we see, this implementation actually provides the required functionality, the functionality of whenever load is being pulled to high.At every time unit t, we want something that can depend with value to compute some kind of function and the output.Some kind of function and the value at the previous time unit, the time t minus 1.So what kind of element do we need in order to provide such a functionality?Well, the missing element, we need something as a very basic point, is we need something to actually remember one bit of information, move one bit of information from time t minus 1 to time t. This is what we have missing.1:35 Now, this means that at this transition point between two consecutive time units it must have state.1:53 These two physical states, it must be able to move between them according to the logic of the previous time unit.An element that can do that thing, that can flip change situation between two different kind of such states are called Flip-Flops.2:46 So, if we again look at the diagram of our time units and assume that our input looks like this.So for us, it's gray, we just don't know what the output is. But once we get to time unit 2, then we know exactly what the output needs to be. It needs to be exactly .what the input was at time 1