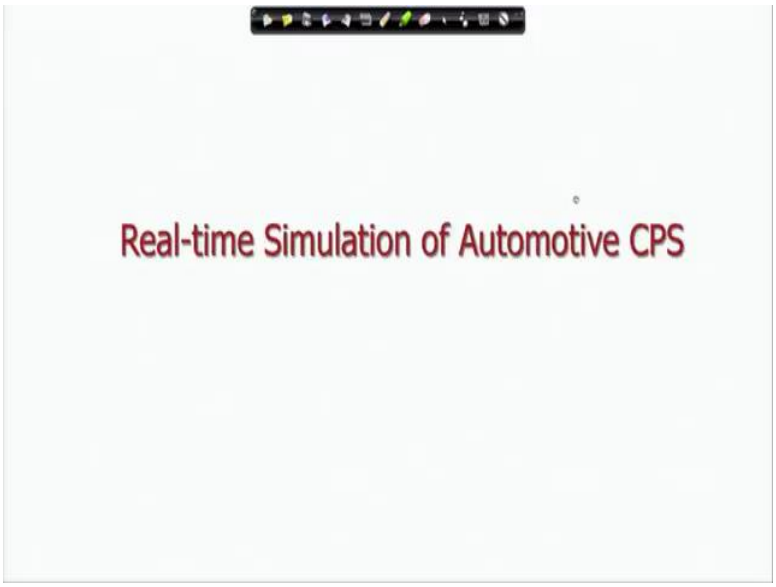
Foundations of Cyber Physical Systems
Prof. Soumyajit Dey
Department of Computer Science and Engineering
Indian Institute of Technology – Kharagpur

Lecture – 03
CPS : Motivational Examples and Compute Platforms (Continued)

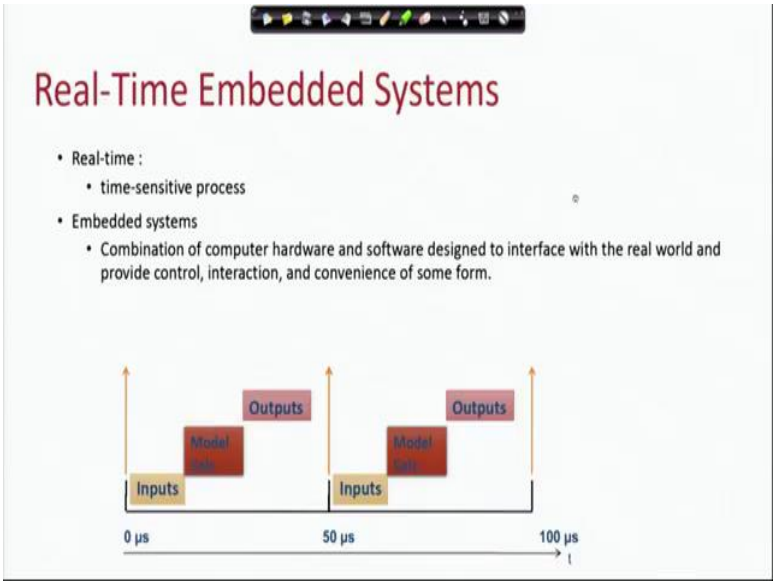
Hello and welcome back to our lectures on foundations of cyber physical system.

(Refer Slide Time: 00:32)



So, in this lecture we will be ah focusing on ah Real-Time Simulation of Automotive Cyber Physical Systems.

(Refer Slide Time: 00:40)



So, what does it really mean first of all? Well, there is simulation but what is real-time simulation? So, let let us understand these fundamental characteristics of embedded systems that they need to execute inside some given deadline. If you are executing a standard software in a desktop system ah there is no real-time deadline for most cases right but you are executing software in an embedded system.

In most cases, it is supposed to execute inside a deadline. So, typically, if I look at a control system, the controller is a software which will get some input from the plant. Then it will execute some control law and then the control actuation commands must be actuating the plant inside the sampling period of the controller. Why? Because in the next period the controller is going to again sample new values and it is going to observe well the control command I gave earlier.

Due to that how did the plant change it is state? ok So, ah whenever I am implementing such a cyber physical system or I I just given control theory example but in most standard CPS examples, we will have this thing that well. There Thus, the the the the software is, what we call as reactive. In nature, reactive means that well, it will do something ah based on the environmental situations it will react to the environment.

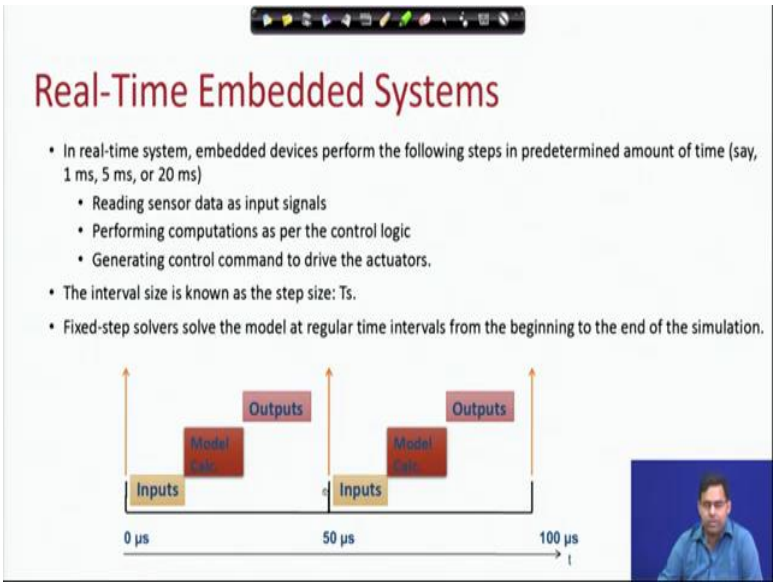
Because it has a target objective. Let us say driving the vehicle in such a way that I maintain a distance of 5 meters from the vehicle in front. So that is a control objective and I react to the scenario that right now, this distance is off from 5 meters but by 1 or 2 meter. Then, accordingly, there is a trigger for me and I will work ok so that is one way of implementation. Or the reactiveness can also come from the sampling period that every after every fixed period of let us say 10 milliseconds an interrupt is generated and that is dictating this control software to wake up, execute, take some decision and then again wait for the next event to be generated. And the next event will come on the expiry of the timer which is set for up to the sampling period and after that again the software will wake up and do something. right The other important thing is the control softwares are executed in feedback loop and as is common for most cyber physical softwares.

That means they will execute they will do something based on that the plant will evolve and by observing by how the plant is evolving they will again do something. So, whatever they do now is a function of what they did in many cases earlier in the many instances earlier. right So,

coming back to this, this is a example of periodic task execution and the task is doing something based on inputs.

And it is outputting something and this is going on and on in multiple different periods consecutively. right And all these periods have their given deadlines in the system, ok

(Refer Slide Time: 03:55)



So, suppose I am trying to simulate such a real-time system. The question is when I execute my software, how do I really know that the software is really executing inside the actual time bound inside which it is supposed to execute? Well, you see if I am doing this experiment on a desktop computer running, let us say windows or Linux the operating system is not making the processor of the system always available to me. Right

Because there are other processors and I am running my control software as one process. So then, ah I will be getting ah access to the CPU to execute this specific piece of code only in some those time slices when the operating system gives me access. And in those time slices where the operating system is giving the access to some other process I am not really having the CPU right, so that is the problem.

Now, the question is ah even if that is not the case, how do I know that the computer is executing my process exactly in real-time? Because it may have it is own logical notion of time attached to that process. ok So, what we really need for evaluating embedded control systems? Is that we will need a way to measure real-time and we will need the events produced by the software to have an to happen ah inside those real-time boundaries. Ok

Now, this problem is further complicated by the requirement that let us say there is some physical system which I am also modelling. And my control input goes to the physical system and now I am supposed to simulate that physical system up to some time and then observe the output and then accordingly again actuate the control input. right So that means the physical system has a model.

Let us say in case of vehicle, there is a vehicle dynamics model that model has to execute inside that real time. Right And it should, I should, I should then get the output ah which is let us say the current state of the model and I will use it in my some embedded processor to execute the control logic. And then inside my deadline I should give back this control command to the computer, where the model is simulating.

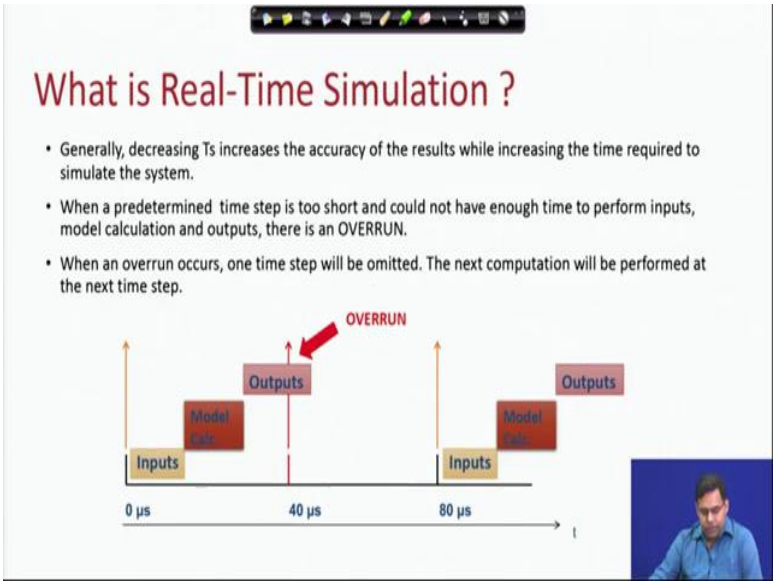
Now, like I said that the model simulation time should be faithful with the real clock. right It should not be like a process like a desktop computer, where this code is running along with other codes and I really do not know that well when the model is producing its output. What is the real-time value? Whether it is as per the real-time requirement here. right So, for this we use what we call as hardware in the loop real-time simulators.

The other important thing that we have in this slide is when I am solving that model so, fundamental is a set of dynamical equations and it is a set of difference, differential equations and they need to be solved. right So, there would be a solver in the back end and it is a continuous time thing. right So but I will have to choose a discrete time step internally, after which those solution values are available.

If that is happening at a very fine granularity then the time taken by the solver to give me the solution of the model at the end of the sampling period. That will be too high and it may be more than the more than the sampling period of the system itself. right So, the solver needs to choose suitable step size so that the number of times the solver has to work inside the sampling period is manageable inside the real-time deadline.

So, this is an involved thing, we will come back to this. We are just trying to enforce upon you that when I am modelling reals, embedded processors, I am running some embedded software and I am trying to see how it works with the plant model. The plant model has to run in real-time and that output has to come to the embedded processor and the processor should run its code. And again, give back the commands to the plant model in real-time.

(Refer Slide Time: 08:08)



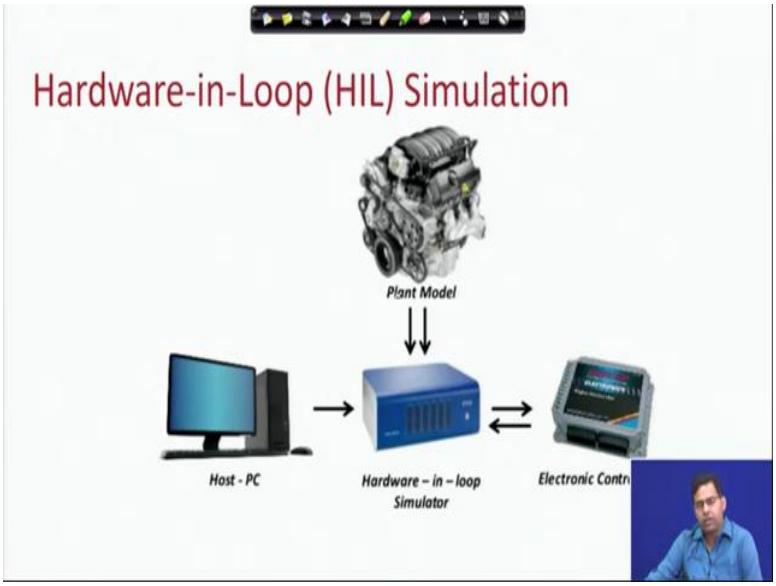
And if that does not happen, this is what it going going to occur. The output computation will be delayed and there will be some time over run.

(Refer Slide Time: 08:18)

And so, how is all this thing done? So, typically for that companies provide you with things called hardware in the loop real-time simulators. For example, here is one example system that is just shown and there are things coming from various other companies also. right So, you

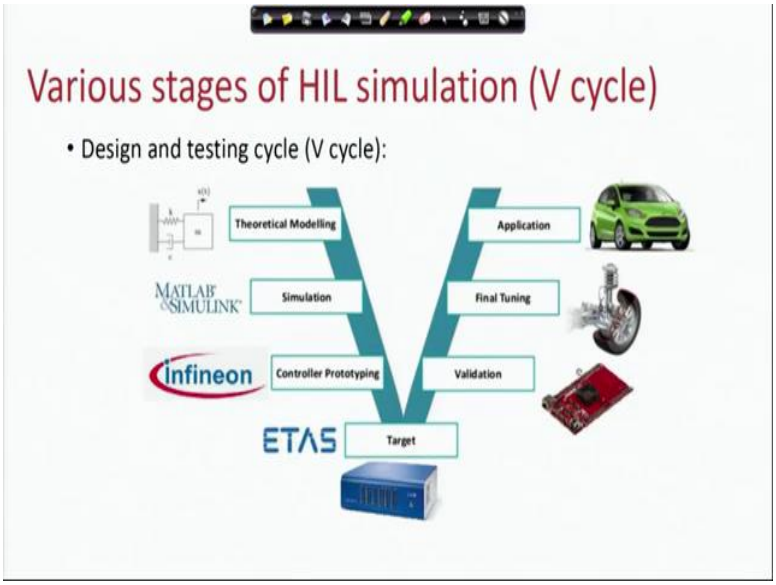
create a model and you can run in this kind of computer. What it will do is, unlike a normal computer, it will show it will show you the model updates happening as per real-time clock. Ok

(Refer Slide Time: 08:48)



So, any kind of physics or any kind of physics based model or that means which replicates a physical scenario can be run in this kind of a hardware in the loop simulator. And then you can see the output in the host PC of the system and it can be connected with the embedded processor, where you have the corresponding software logic running. Ok

(Refer Slide Time: 09:11)



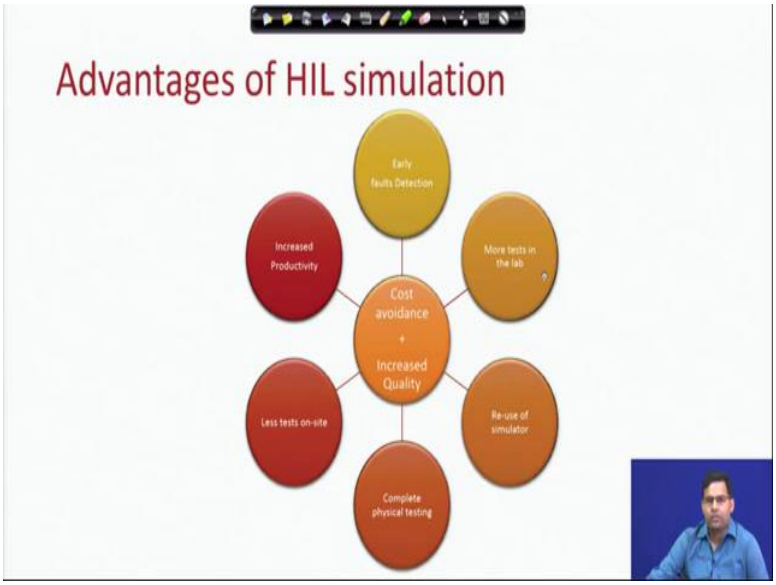
So, this um thing that is ah I mean testing a system can be done at various levels in a in an automotive space. right And this is typically the V cycle of design and testing. So, you can have theoretical modelling then you can have simulation in Matlab and then you can design

the controller and prototype it in a embedded software processor board. Then you can attach it with the target system ah which is basically a model of the actual physics physical system and that model is running in a real time computer.

And then you can do validation, then you can do final tuning or maybe you can attach ah this processors and it is output to the sub part of the physical plant. And then you can do some so, you can do validation at this level by attaching this software ah running on the processor, with the model running on the hardware in the loop simulator. And make them simulate together, then you can go to this level where you can attach the sub part of the big physical thing and see that how it is behaving.

And then you can go to the actual application that you have the entire thing may be a car and you bring in the software here you attach that board here and see that well how this is driving around and the required functionalities are working in some test track or test scenario. Ok

(Refer Slide Time: 10:39)

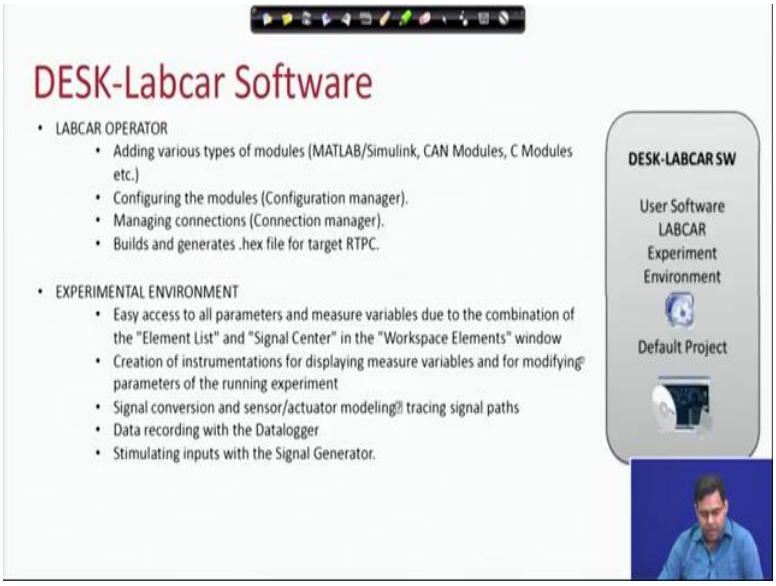


So that is how it works usually so, of course, ah you it may be difficult to get the real system. right So, anyway the preferred mode ah would be to use this kind of a simulator and make your software run in the loop with the simulated model of the plant instead of having the real plant. OK Because of this various advantages that you have with respect, to cost, quality, less number of tests to be done on site, etcetera, etcetera.

(Refer Slide Time: 11:11)

So, we we are just giving some names of some simulators that can be used and there are free like (11:18) I have been saying that there are many other offerings from many companies. So, this is a combination that used to work in our lab. So, you can have ah simulate that HIL simulator, like Labcar and along with that you can have any of your favourite embedded processors ah which can run your control software. Right

(Refer Slide Time: 11:37)

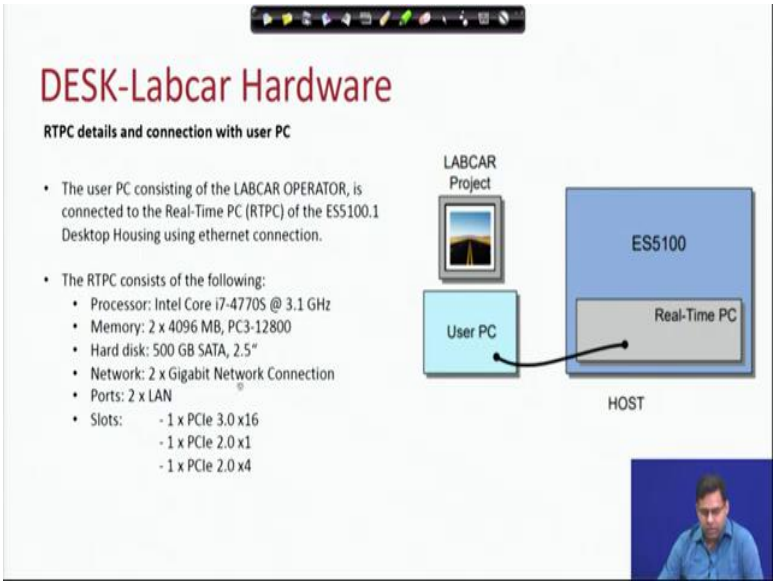


And you can just configure them suitably.

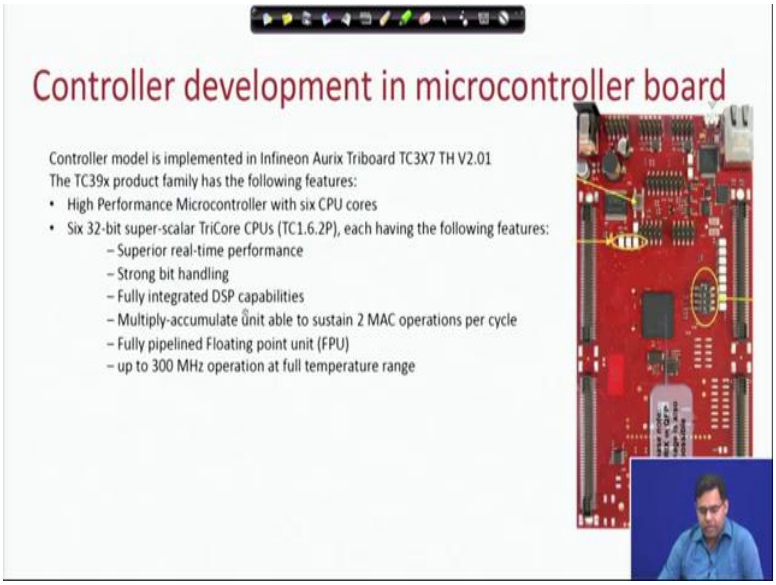
(Refer Slide Time: 11:41)

And you can connect them together. So, this is just like a broader screenshot of how an entire system looks. So, at the back there will be lot of ports just like a computer but there are many additional ports also to interface with your embedded processor.

(Refer Slide Time: 11:57)



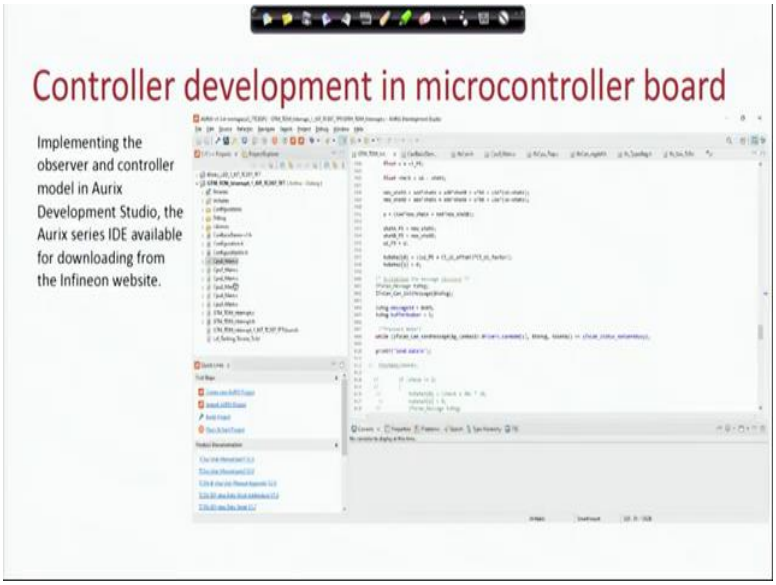
(Refer Slide Time: 12:00)



And then ah well you can you can just interface it with some embedded processor like this. So, this is an example microcontroller board we are showing is infineon boards are typically used in automotive systems along with many other kinds of boards. So, this a typical board will have some multiple one or multiple CPU cores. For example, this specific board has six CPU cores.

That would mean that you can really run programs compiled to these boards in each of this course simultaneously. Right

(Refer Slide Time: 12:28)

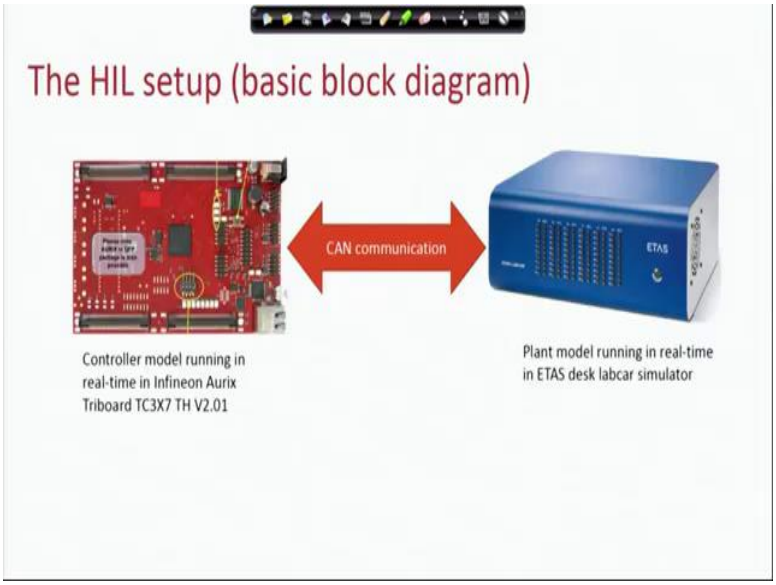


And this is like a screenshot of the IDE, the Infineon IDE, the Aurix IDE through which you can write your control software, the actual control software which will sends messages which will execute some control logic. And then which will which will send some control actions

over any of the interfaces that this boards provide. ok It would provide a typically it would provide a CAN interface through which you can connect these boards output to a CAN cable.

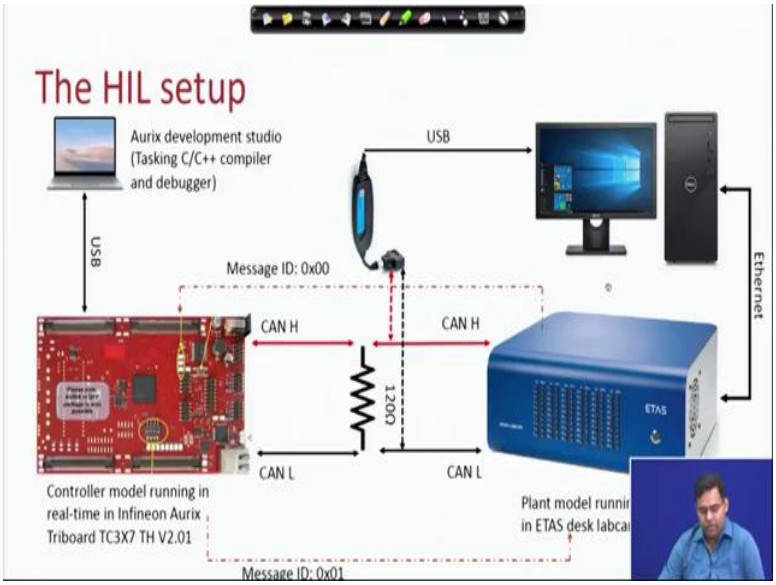
And that CAN cable will you can attach to the HIL system to see that well how the plant reacts to the control actions that are communicated here. Ok

(Refer Slide Time: 13:06)



So, yeah this is the example picture that we are talking about. You can have your control communication you can make the plant model ah evolve and you can watch it is trajectory.

(Refer Slide Time: 13:21)



Now so, this is a typical HIL setup, so, the CAN is basically a set of twisted pair wires with suitable resistance. Ok Now, you can also log the CAN data in another PC and you can see that well what kind of data packets are going because your vehicle sends values from the vehicle.

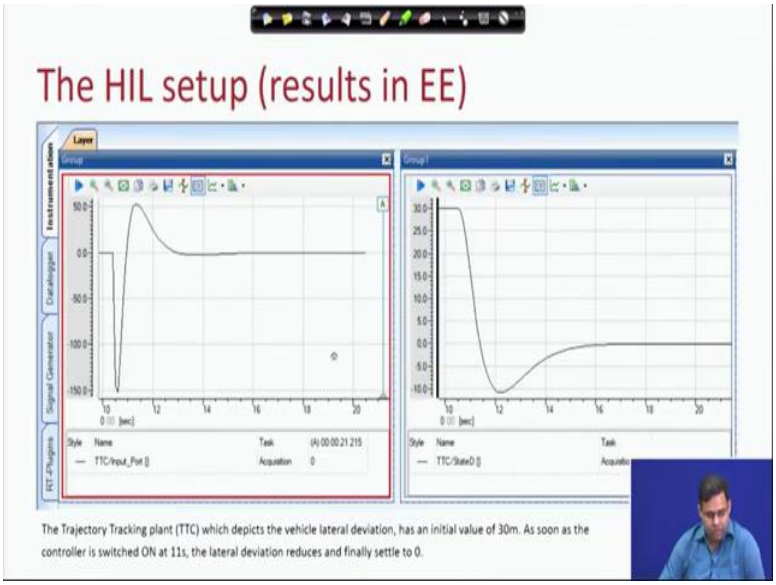
Let us say let us take an example I am modelling a vehicle dynamics. I am having wheel speed and linear acceleration. ok ah

And let us say some more things like the vehicle's steering angle, I am monitoring this signals and I am sensing those signals and sending through the CAN bus to the ECU. The issue is executing some controller. Let us say the adaptive cruise controller, let us say the ABS controller and those control commands go back to the con through the CAN bus through the plant model which is evolving here. right Ah

So, let us say this controller wakes up once every 10 milliseconds. That means I will be computing the dynamics of that plant. ah I want the output of the dynamics once every 10 millisecond and that should be communicated back here. Now, when I evolve those continuous dynamics inside 10 milliseconds this system would choose a suitable step, such of the solver that whether it will solve the dynamics once after the 10 millisecond or multiple rounds in between for increased accuracy. Ok

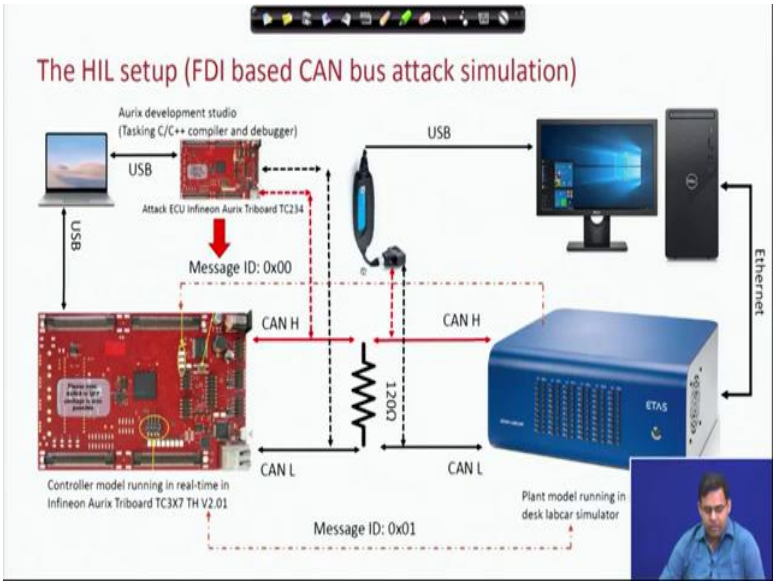
And then once the those equations are solved here right and you get back here. right So that simulation is happening here in real-time because this is a at a real time, computer here. And so that means once that is done, you will get those changed values and then you will be computing ah the control signal. And you will be sending that back through the CAN bus. And then with some modified steering angle or modified breaking value or modified acceleration value those are the discrete actions that your controller to here plant model will again evolve right.

(Refer Slide Time: 15:29)



So that is how a typical HIL setup will work and these are some examples. Let us say you take a trajectory tracking plant. So, essentially, you are following a previous vehicle. And so, there was some lane deviation initially by the previous vehicle and you are slowly collecting the deviation. So, finally, this deviation goes to 0. right So that is a typical example we are trying to show again not going to the details here.

(Refer Slide Time: 15:51)

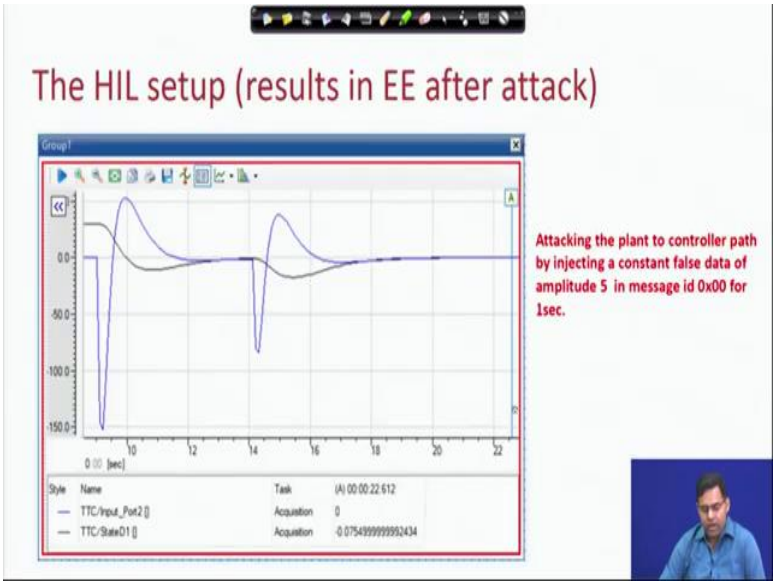


So, this kind of a testbed has more usage. Suppose I am trying to do some experiment on automotive security which is nowadays quite a hot topic. right Suppose I have an idea that the vehicle can be attacked by ah at attacked over the internet or through some other surface. The vehicle can be attacked and communication between the plant and the controller that is the vehicle and let us say one of his ECUs has gone corrupt.

So, the CAN data packets, they are getting transmitted, they may be spiked by the attacker and I want to see that well, how is that going to affect my system? So, let us say I have an attacker ECU whose whole job is to spike these messages with some. ah in some way that means it would, it would do something it would do something to make the vehicle model think that well this is not the ECU which is supposed to send the actual CAN messages.

So, it will send this ECU to an off mode and this attacker would start replicating this ECU's role. And it would start sending spiked messages. ok So that is an example of how you can create a test bed where you are able to replicate automotive security experiments.

(Refer Slide Time: 17:09)



Now, if you do that you may see how things change maybe you can see that well due to such attack deviations start happening. And that would mean the trajectory tracking you are doing that you are following a vehicle and the the set deviation you are supposed to have from the previous vehicle that keeps on altering. ok So, those things can happen and this is the way that you can have a testbed through which you can actually see such experiments carried out.

You can actually, see that let us say you are developing a security scheme and if you prevent that CPU scheme here in this victim ECU. You can actually, figure out that well, this victim ECU able to work or not. And what is the effect of that attack or what is the effect of that counter measure on the plant dynamics? So, this is. This is the important thing here. right We are talking about cyber physical systems.

So, even if you have a security measure, we will like to know what is the effect on the physical dynamics? So that is why you will need the software logic. You will need some safeguard and security but final validation would require the physical model of the system. And you will like to see that how the model evolves and well in case I have taken a security measure if that have some effect on the physical dynamics of the system. ok

So that is how all these concepts get link up and you can get to do some interesting thing on CPS security here. So, with this interesting example, we will be ending our lecture. Thank you for your attention.