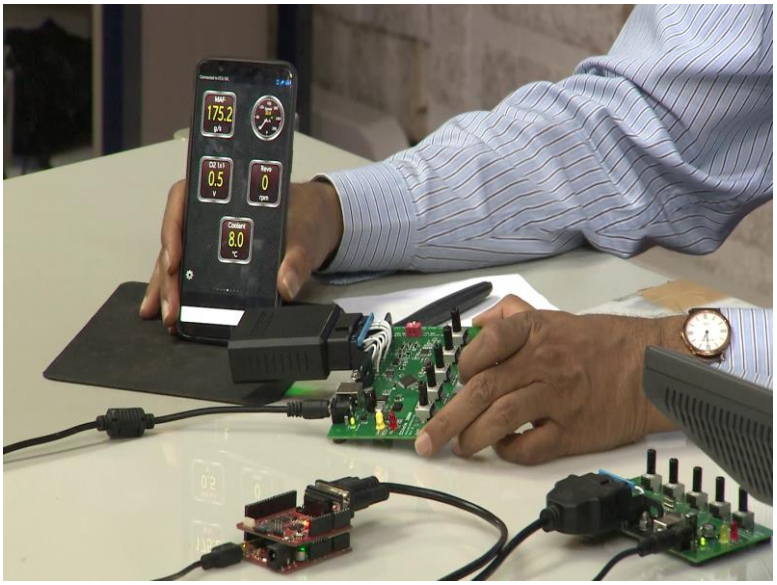
Advanced IOT Applications
Dr. T V Prabhakar
Department of Electronic Systems Engineering
Indian Institute of Science, Bangalore

Lecture – 23
OBD - II and stream processing demonstration

Alright, so, what we will do now is we will show you a simple demonstration of what a mechanic would want to check if a car or a vehicle comes for maintenance or if he wants to do a regular a preventive maintenance check of all the parameter IDs specific to the vehicle. Please note that I mentioned to you sometime back, that the number of ECUs in a vehicle can exceed even 70 to 90. While that is what will happen in reality, as far as this lab demonstration is concerned; let us put our attention on the engine ECU.

(Refer Slide Time: 01:10)



So, you can see there is an engine ECU here we will show you some of the parts of this engine ECU. You already seen this 5 knobs and these 5 knobs essentially correspond to coolant temperature, engine RPM, vehicle speed, oxygen and mass air flow. So, you can have anomalies of different types happening within the engine. Now, back to this picture here you will see that, OBD II is written here. This is a OBD II connector and this OBD II connector essentially has another controller called ELM 327. This ELM 327 essentially has the ability to connect on one side to OBD II connector based protocol, which essentially uses CAN protocol(ISO 15765 - 4).

And ultimately it has to be useful to a mechanic, the mechanic can connect a handheld device and pass commands directly. But, if you look at an IoT device you could essentially use a mobile phone to do the same. And therefore, this device has ability to take OBD II commands on one side and convert them into equivalent Bluetooth Low Energy packets, which means that it now becomes amenable for viewing on several Google Play Apps that are available.

Once such app is is called torque “t o r q u e”. So, you could download any of these apps and look up the engine ECU parameters.

Now, you can see that there is an LED blinking here and this LED blink simply indicates that everything is connected and particularly the BLE is connected; that means, the mobile phone is connected. This phone along with this OBD II setup is one possible setup that you can imagine. Now, imagine that this particular engine ECU is part of the car or a vehicle. And all that the mechanic has to do is locate the OBD II port and push this connector inside and then use this mobile phone to see the different parameters.

So, now what I will do is I will ask Vinod Hedge to you know move these knobs and then I will perhaps hold this phone for you to see what is the impact of these connectors. All right. So, maybe he should come and then turn the knobs one by one. What are you turning now? He is turning mass air flow: Can you now see? It does increase now.

So, that is a nice thing there. The second one is oxygen. So, you can see oxygen is also showing is also increasing. Please note oxygen is an output sensor, mass air flow is an input sensor essentially the mass air flow is connected between the throttle body, there is a throttle body. There is an air filter and essentially this throttle body input basically takes the mass air flow which is measured. Oxygen is what you are measuring as an output and that is expected to be as low as possible only then you realize that the combustion has happened successfully.

So, let us go back and see how the oxygen sensor manipulates. So, let us change the oxygen sensor value. Ok, there you are. There are some typical numbers which can happen. Essentially, if you look back the mechanic can do things like the following: he can connect this device to the car or vehicle and then drive it for about a kilometer or 2 and then look up these values as and when things happen and see it spots some anomaly by looking.

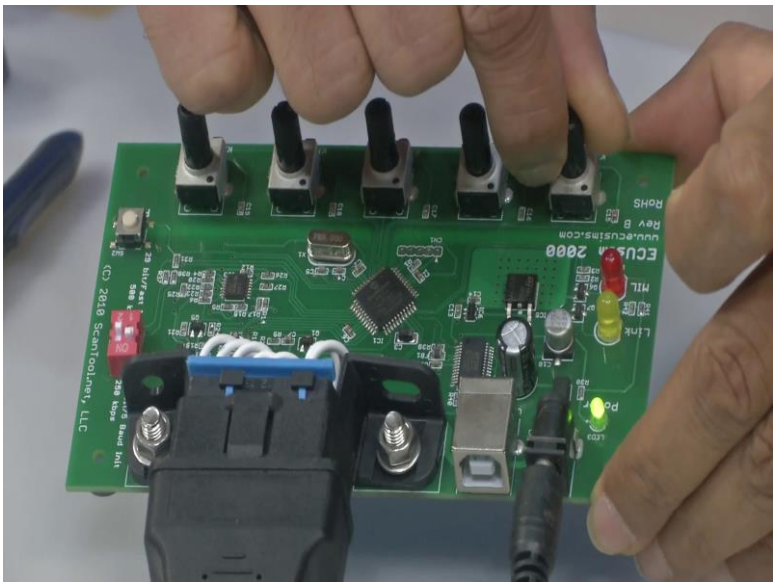
But, this is not the best way to do. You need a complex event processor, which essentially stores these values onboard and perhaps taking some real time decisions. And showing you some real time thresholds as and when the driver is actually moving along the vehicle right, when he is actually driving the vehicle. This is what I am showing you right now is for a mechanic to spot the anomaly is by connecting a device. This is not the way you want to do. You want to be alerted as and when you are driving at high speed or whatever on a highway if there is an anomaly, it should show up.

I also mentioned to you that the front panel of most of the cars that you see today, you will have a MIL indicator: it is called the malfunction indicator lamp. And that is the only indicator that sort of tells you something about the fact that the engine throttle body there is some issue there. So, that is the only light that will essentially go up. Battery is shown separately, the oil indication is shown separately, there is a temperature dial which is also there measuring the temperature of the engine and corresponding systems connected also to the coolant temperature.

So, therefore, this is what would be very useful to connect. So, one can now see that it is highly limited: this is something only that the mechanic can do. Let shift focus to see if we can essentially automate all this inside a vehicle and then see if there are automatic indications that can be generated.

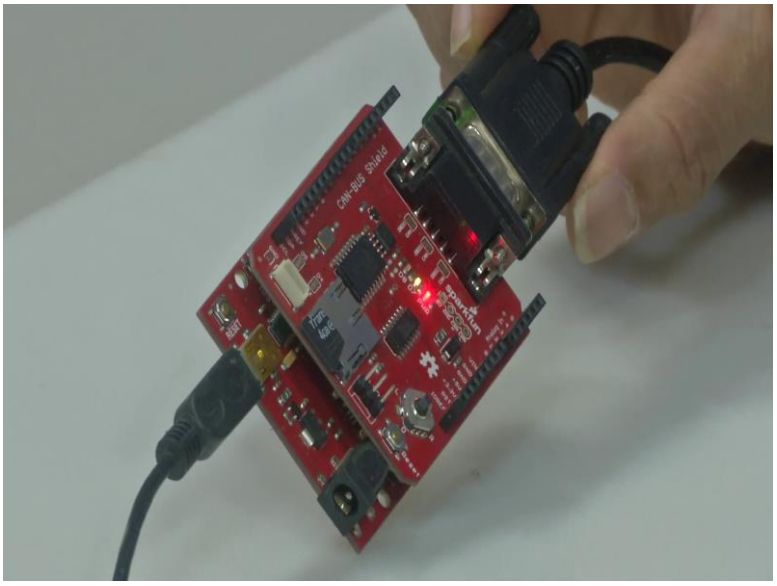
Let us move to the following scenario of automatic alerts that have to be generated or automatic action that has to be taken in real time inside a vehicle. What we did previously was only for a mechanic to identify some bugs in the system.

(Refer Slide Time: 09:22)

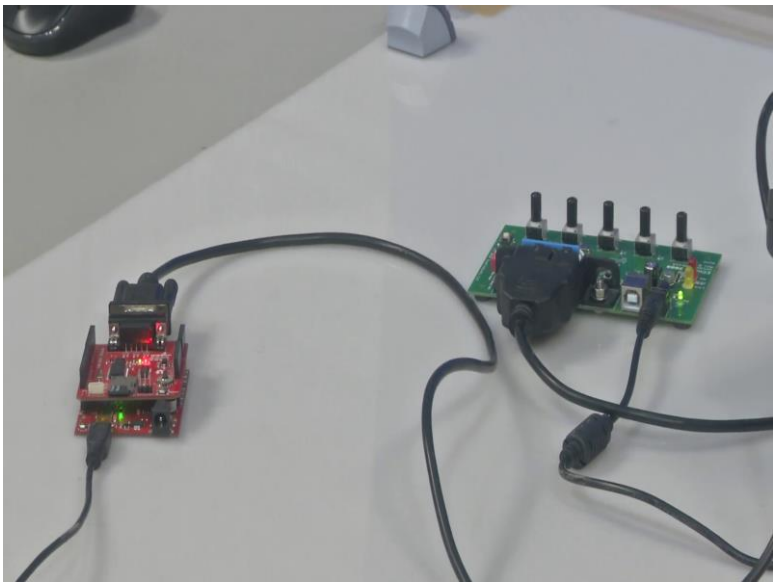


Now, imagine you are inside a car and are driving at some good speed, and this ECU is here. ECU as usual is monitoring several parameters for you. You can see that this CAN interface here in this demo the CAN interface does not have a Bluetooth OBD II output ELM 327 based system, but indeed it is going in this direction and then going and connecting to this little piece of hardware, which is CANBus shield obtained from SparkFun.

(Refer Slide Time: 09:53)



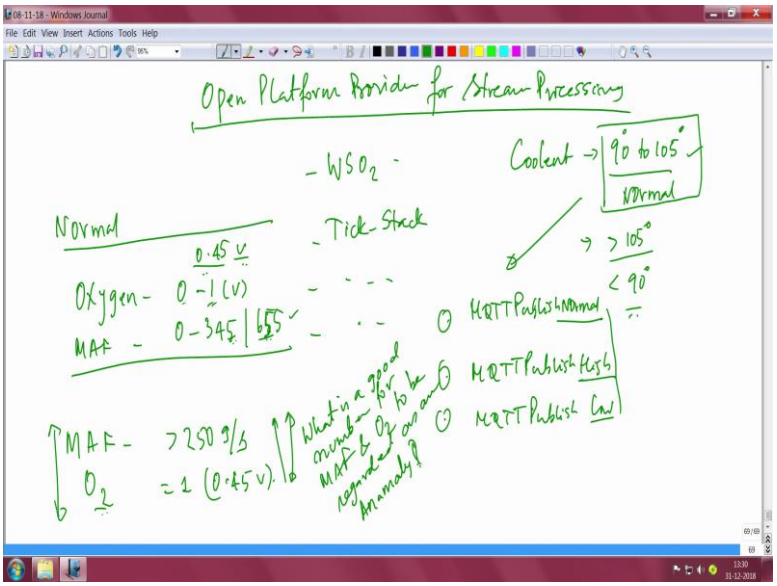
(Refer Slide Time: 10:12)



This board the one down below here connected to an onboard computer in the automobile. This setup has to detect anomalies in real time to ensure that it alerts you; recall what we discussed about mass air flow and anomalies with respect to throttle body, the butterfly valve and so on. Or engine coolant temperature for instant rise in coolant temperature is a big anomaly, it has to the vehicle perhaps has to halt almost immediately to take corrective action.

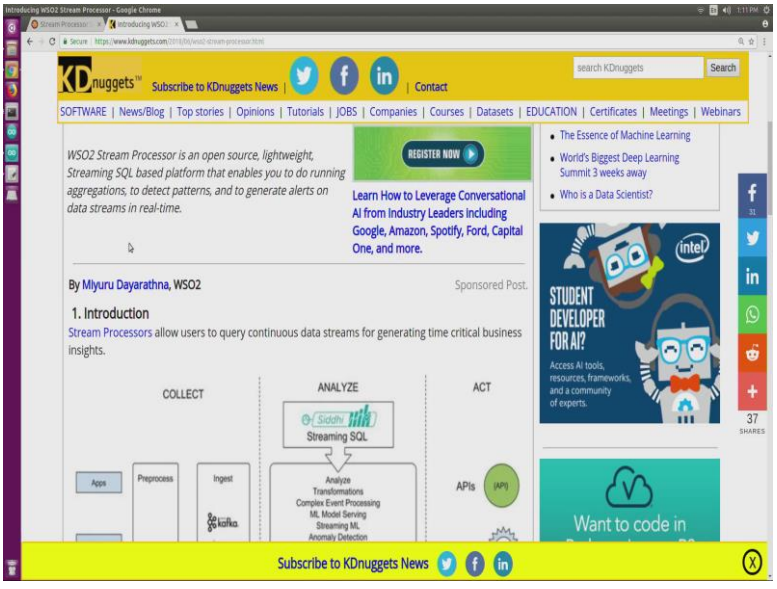
So, somehow you have to do that actuation, right. So, those issues have to be highlighted. Therefore, how will you do these things as a programmer or as an automotive engineer to put this system onboard of a vehicle.

(Refer Slide Time: 11:24)



For that what you would do is you would use a stream processing platform and such a stream processing platform is WSO2. There are many open source platforms; one such is called TICKStack. So, you can use any open source platform you are very comfortable with and to understand what exactly is the WSO2, because we will do a demonstration with WSO2. Let us turn to the computer which essentially shows a display of what WSO2 is.

(Refer Slide Time: 12:04)



Here you can see, the WSO2 is a stream processor, is an open source, light weighted streaming SQL based platform. And enables you to do running aggregations to detect patterns and generate alert on data streams in real time. Beautifully connects to the kind of anomalies that we are looking at. This picture says it all: left side are several app services and sensors which are there, like MAF sensor, oxygen sensor, then coolant temperature sensor. All these sensors are there which does some sort of preprocessing. And the ECU has all this data. This data which the ECU has will have to be somehow pushed onto the dashboard, for some sort of display the onboard computer has to take the data and then has to make it available to an onboard computer based system.

So, essentially all the different sensors which are giving away data will have to be collected by the ECU. One can think of protocols like MQTT, COAP. Here you can see that this WSO2 support MQTT as a possible IoT protocol, all that collection happens and then now comes to an onboard computer. The onboard computer is a place where actually the WSO2 is running and that is where all the analysis is happening. You can see it does analysis, transformations, complex, event processing, the machine learning model serving, streaming machine learning, anomaly detection.

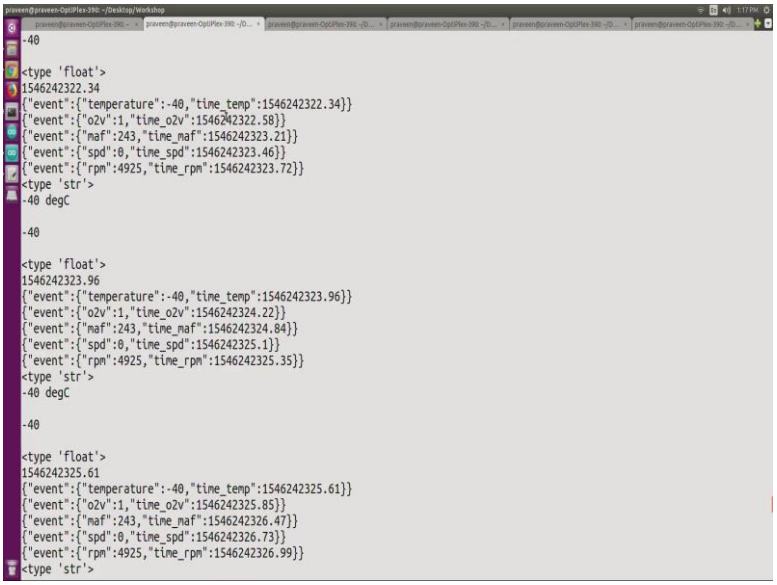
All that data that is coming from the sensor; some of them that are useful are kept, some of them are not so useful which are well within standard limits are thrown away and the analysis is happening in real time as and when the data is coming from the sensors. So, the left side is feeding to the middle one. Analyze. Now, if you see an anomaly you should act upon it.

So, the act could be in terms of an alert, it could be in terms of something appearing on the head-on display of an automobile or it could be in terms of actions, halting the vehicle or in terms of a standardized interface for other actuators to work with, which is essentially the APIs that you can see there. I suppose if you have third party actuators you need to release an API, which essentially we will feed to that API and from third party would essentially pick that system and then do its own actuation; perhaps it has its own generic actuation.

So, now what we will do is we will have to show you what the OBC is actually collecting all the data. Remember there is the collect data, analyze data and act upon the data. So, the

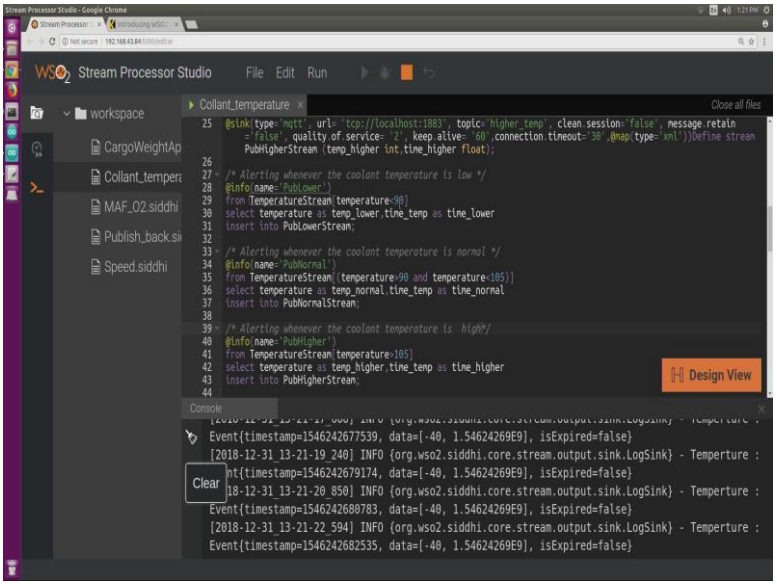
collection part is coming from several sensors. In this case we have limited to just the engine ECU related sensors.

(Refer Slide Time: 15:13)



So, turning our attention to the screen the first one that you see is temperature and then you have oxygen, you have mass air flow, you have speed and you have RPM. All of them are set to some typical numbers and then you will have to now you are essentially defining those typical numbers and then now catch anomaly based on those typical numbers that have been set up here all right. So, this OBC is continuously getting this data.

(Refer Slide Time: 15:42)



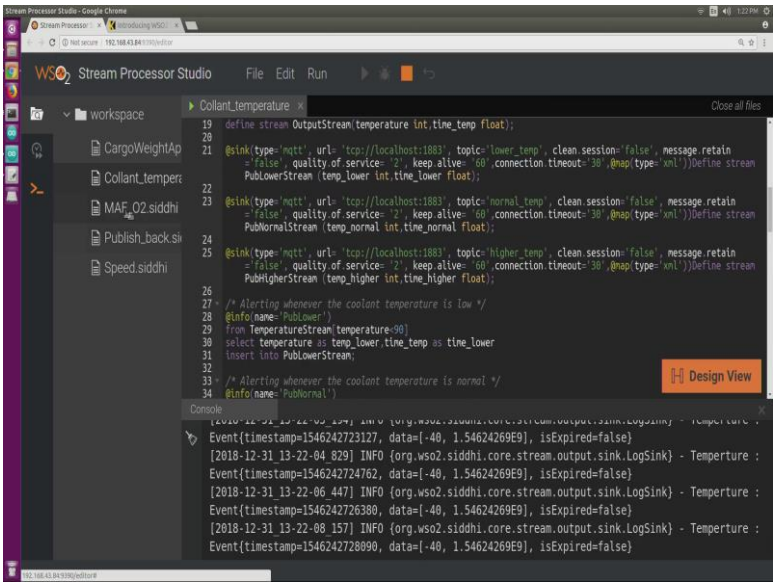
Now, let us turn our attention to what is actually happening. You can see that this part of the code essentially is collecting data by from all these sensors using a protocol like MQTT. And let us just concentrate on coolant at the moment, ok. You can see that the paper was saying 90 degrees to 105 degrees is considered to be normal, anything greater than 105 degree is bad, is dangerous and less than 90 is also not the right thing to happen.

So, if you are really looking at an MQTT system and OBC is collecting data and processing data and all that. If you are repeatedly getting 90 to 105, to save space you look at the data and discard that data. So, you can have one MQTT publish; MQTT publish which says normal. Like that you can say MQTT publish, you can also say high, you can also say MQTT publish low. We will show you one by one; perhaps for you to get a feel. So, what would happen? If, you see these 3 things and if I turn the knob of coolant temperature, the dashboard should indicate that there is a problem with the coolant temperature. Because it has intelligence inside.

The complex event processor has been assigned certain threshold and it is just looking at these thresholds and alerting once the value cross a limit. And to for this particular display purpose since we do not have an automobile here or a dashboard, we will switch to a normal mobile phone. And maybe we can show you some numbers on that mobile phones. Let us see if you can pull off this demo all right.

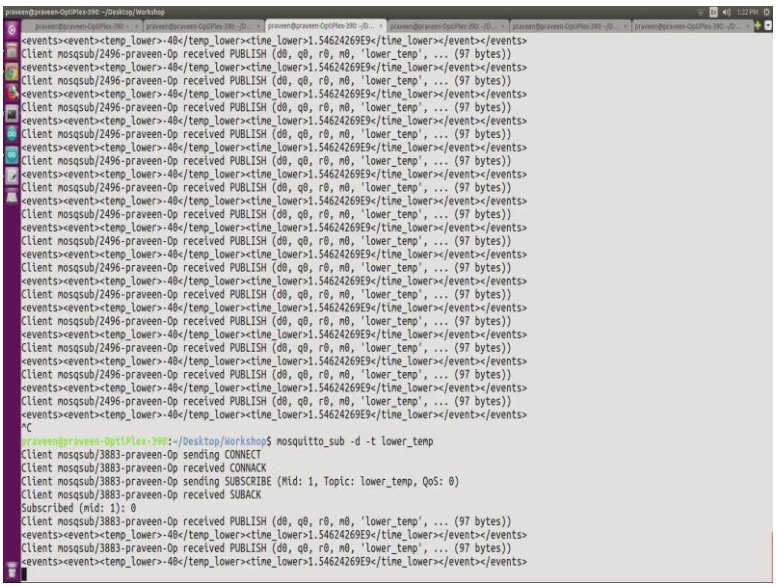
So, now is to look at the code and look at where is normal, where is high and where is low as far as MQTT publish is concerned. So, let us look at that part of the code. You see PUBlower is that lower stream temperature lower. It is been defined as less than 90. Now, let us go to publish normal. What is normal? Normal is 2 thresholds: between 90 and 105. Then you have publish higher. So, you have these 3 thresholds.

(Refer Slide Time: 18:39)



Now, the system is setup for checking whether coolant if it goes high, what is it that is going to happen?

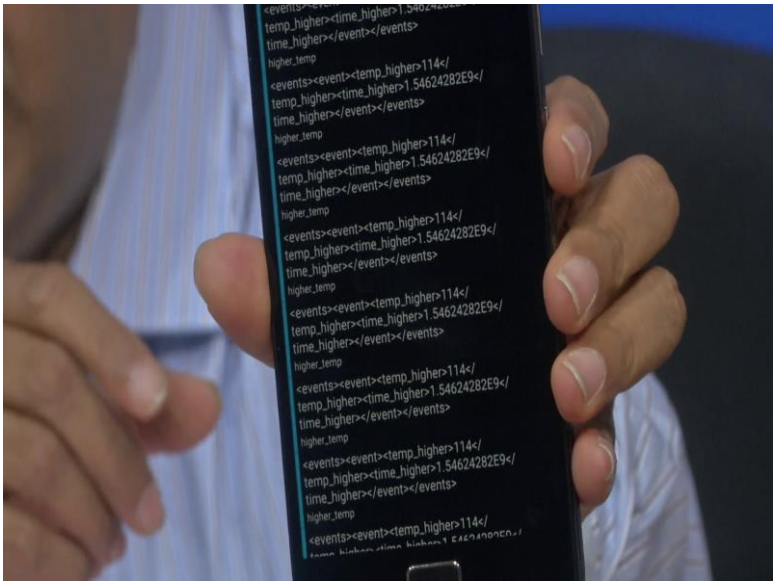
(Refer Slide Time: 18:47)



So, now let us see what is it is current status of coolant temperature: it is at very low value some -40. And now what we will do is we will turn the knob and we will ask our camera to catch that anomaly. So, the coolant temperature is being modified and moment it catches a value, it was indicated a higher value. You can see it is 114 now, which is a clear indicator

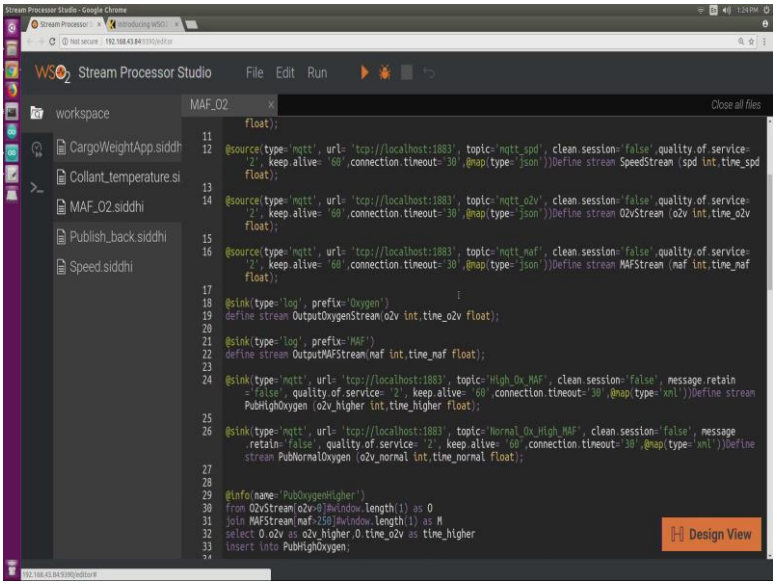
that the value has gone up. Now, it is ok to see it on the OBC, but the driver should get it right, which means that part of the screen should also show here.

(Refer Slide Time: 19:50)



So, let us see for our camera can catch this screen as close as possible. So, it shows a 114, this is what the dashboard of the driver will indicate. So, one can go on like this and essentially look at different anomalies what could be happening in the using this kind of a framework. The second anomaly that we may want to show is perhaps with respect to MAF.

(Refer Slide Time: 20:24)



So, let us see what the code of MAF is and let us see what the current value of MAF is. By the way this code will be available to you on the usual website. So, if you wish to open it and then have a look at it should never be a problem. So, let us look at MAF code. So, the MAF code is shown here. MAF value is another sensor again published by MQTT. You can see that the MAFs vary there you can go on like this and you can look at several anomalies.

One other anomaly that you can look at is the mass air flow in ECU system that is available to you. The oxygen is measured in voltage levels; it is between 0 and 1 volt. If you look at other sensors pertaining to the engine ECU, oxygen sensor is an output sensor. You would see a very small value of oxygen, which is corresponding to the fact that all the air that has come in, there is a good mixture of air and fuel and at the right time the ignition has happened. So, the fuel injection system is working efficiently well.

Typically it is not going to be 0. So, it is going to have some value from what we know that these values typically range from 0 to 1. 1 means high extremely high, 0 means there is no oxygen at all: very ideal situation. So, you typically look for a normal range which is around 0.45 or so. So, it should be as low as possible, but ok. So, 0.5 – 0.45 volts if you are able to measure on our ECU system, we would regard it as something which is which is not a MAF functioning.

Similarly, MAF is typically between anywhere up to a maximum value of 655, you cannot have 0 MAF, because some oxygen has to go in. So, you would typically say it should be higher as I have close to something as much as 655. Again in this ECU this range is around 345 it would go till there. And now what is the anomaly that you want to create, the anomaly is that if MAF is greater than 250: is not really an anomaly, but you want to do it along with oxygen sensor.

So, you are not absorbing single value single parameter, but you are looking at multiple parameters in one go. This can be regarded as an exercise that you have to look up, read literature and understand what is a good number?

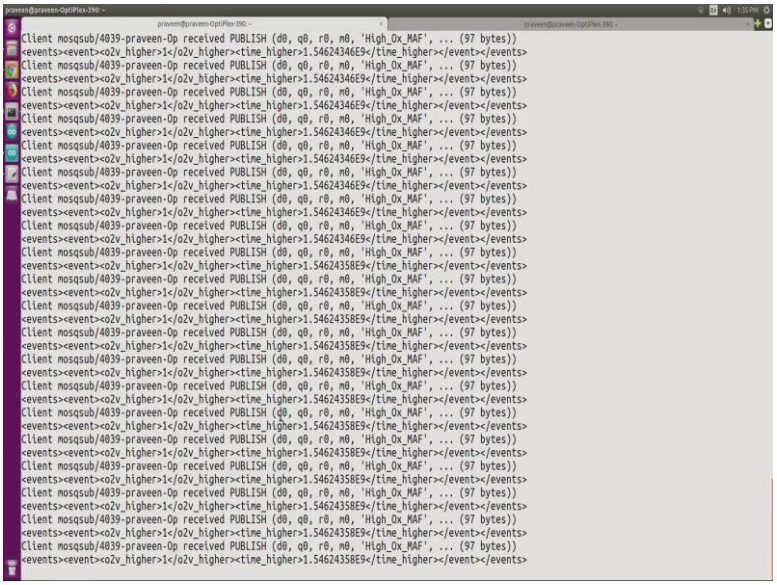
So, while put the question here, what is a good number, what is a good number for MAF and O2 to be regarded as an anomaly? This is a question. We are not going to answer this you will have to figure out: read literature. So, we are looking at 2 sensors together. The point I am trying to make is if you look at the setup which we have put together from our

lab. The WSO2 platform has also the ability to do stream processing on multiple parameters, very complex set of parameters can be tuned and you can catch anomalies with respect to those multiple parameters.

This is indeed the power of complex event processing, which is essentially doing stream processing, multiple thresholding is something that you can do. Therefore, it is a very powerful engine which is running not just about analyzing, but it can also do you know storage of this anomalies events, which can then be uploaded to a cloud for slow loop analysis. So, the combination of real timeness and slow loop analysis by uploading data is an interesting part of this stream processing application.

So, now let us look at these 2 MAF as well as oxygen sensor. You can see that MAF sensor and the oxygen sensor are tuned to create an anomalous event. And now let us go back and look at the screen here. The screen here is essentially indicating a value which is greater than 250, which is indicated as a sort of an anomaly. And oxygen at the moment is 0 Which is ok, but if it is changed back to a different value and then it should indicate one.

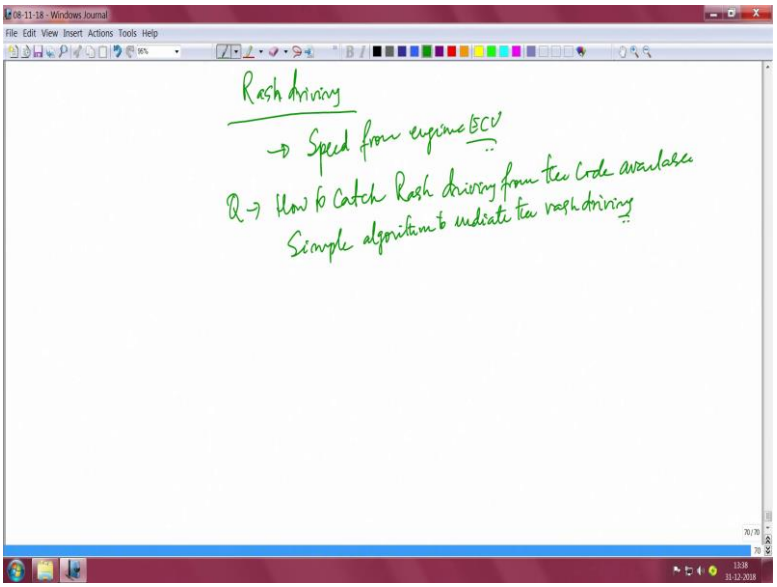
(Refer Slide Time: 25:31)



So, let us see if it is indicating one. Yes, you can see that there is a subscription to high oxygen and MAF; that means, these are 2 ECU parameters and oxygen is indeed high. So, let us see if it can indicate those anomalies.

So, you can see that it is indeed indicating one, which means it is an anomaly right. So, you can see oxygen voltage is higher and has a created an event. One more anomaly that you may want to catch with respect to rash driving.

(Refer Slide Time: 26:07)



Well, you should be able to get this from this speed. Essentially speed should be an indication and therefore, if the speed you know is going beyond a certain threshold, you may want to say that driver is under a rash condition and indicate that trash condition immediately.

I am sure you all experienced sometimes when you hire a cab or something or some other the public transport service and you find that the driver is rash you may want to report: that is one way. The other way is the values that are coming from ECU itself can log it, and store it and also upload it to cloud, or it can immediately indicate to the driver that the driver is actually driving this vehicle very rash.

So, the second question is how to catch rash driving from the code available to you. So, you will have to write simple algorithm to indicate the rash driving. If you wish you can look up that paper that we discussed.

Thank you very much.