



**MIDDLE EAST TECHNICAL UNIVERSITY**  
**ELECTRICAL-ELECTRONICS ENGINEERING**  
**EE300 Summer Practice Report**

**Student Name:** Mehmet ATAŞ

**Student ID:** 2304020

**Company Name:** ASELSAN ELEKTRONİK SANAYİ ve TİCARET A.Ş.

**Divison:** Ulaşım, Güvenlik, Enerji, Otomasyon ve Sağlık Sistemleri Sektör Bşk.

Test Mühendisliği Mdl.

**SP Date:** 16.08.2021 – 13.09.2021

**Supervisor Engineer:** Mehmet Efe Uluyurt

**Contact:** +90 530 642 32 56

## TABLE OF CONTENTS

1. Introduction	1
2. Description of the company	2
2.1. Company Name	2
2.2. Company Location	2
2.3. Vision	2
2.4. Mission	2
2.5. Shareholding Structure and Financial Status	2
2.6. General Description of the Company and Division	3
2.7. Organization	3
2.8. Capabilities	4
2.9. History of the Company	4
3. Test Engineering	5
4. Security Systems	5
5. First Project	6
5.1. Ranorex: Test Automation for GUI Testing	6
5.2. Overview for Fast Forward Tests	7
5.3. Preparation Process for Fast Forward Tests	7
5.3.1 Data Source and Data Binding	9
5.3.2 Characteristics of the Camera Streams	10
5.4. Operation Process for Fast Forward Tests	12
5.5. Fast Rewind Tests	21
5.6. Extra Work on First Project	22
5.7. Summary of First Project	22
6. Second Project	23
6.1. Extra Work on Second Project	25
7. Conclusion	28
8. References	28

## 1. Introduction

I have performed EE300 Summer Practice in ASELSAN which is an integrated electronics industry company that develops, manufactures, installs, markets, and carries out after-sales services of modern electronic devices and systems. My summer practice lasted fully 4 weeks, started at 16.08.2021, and ended in 13.09.2021. Mehmet Efe Uluyurt, who is a senior test design engineer was my supervisor and he arranged my internship program.

This internship program allowed me to experience engineering practices, working conditions, and professional life. Also, it helped me to observe the applications of the theoretical knowledge I gained, increased my professional experience. I believe that it will guide me to make informed career decisions after graduation.

This report describes in detail what I did and observed during the summer practice. It starts with the description of the company. I give detailed information about what I have done in projects such as codes, simulations, etc. Then, I summarize my report in the "Conclusion" part. I include the sources of the documents from which I took help in the "References" part.

The defense industry sector stands out as a very attractive sector throughout the world and in our country. Especially the use of high technologies and the size of the investments are among the factors that make the sector attractive. The coming together of these two phenomena is exciting for me. Turkey's large investments in the defense industry and the introduction of high technologies to the world market with its nationalization policy are raising appetite in terms of national service. ASELSAN shows how serious they are in the sector with their investments.

I preferred ASELSAN since occupational health and safety is at the forefront and the quality of the staff and the environment are meticulously planned. Currently, the CVs of the employees at ASELSAN are quite full, which has contributed to me at the maximum level both as a role model and as a mentor in my career planning.

The division where I have performed my work is the Test Engineering Department which is the part of Security Systems Technologies Division. During that internship, I learned many things about test engineering and observed the facilities of the company.

Briefly what I have done during the SP was to perform fast forward and reverse tests in synchronous recorded video playback in the CCTV monitoring and client user interface. Also, I did Linux application server management interface tests and I created a .exe file using the windows form application.

On the last day of my internship, I gave a PowerPoint presentation to the test engineering group about what I did during the summer internship.

***In this report, some points cannot be explained due to company confidentiality.***

## 2. Description of the company

### 2.1. Company Name

ASELSAN ELEKTRONİK SANAYİ ve TİCARET A.Ş.

### 2.2. Company Location

Overall, ASELSAN has 8 facilities. My summer practice was in Macunköy.

ASELSAN Macunköy Tesisleri (Genel Müdürlük, HBT, SST ve UGES Sektör Başkanlıkları)

Mehmet Akif Ersoy Mahallesi 296. Cadde No: 16, 06200 Yenimahalle /Ankara, Türkiye

Phone: +90 (312) 592 10 00

### 2.3. Vision

To be a reliable, competitively preferred, environment-friendly, and human conscious technology firm which preserves its sustainable growth in the global market via the values created for stakeholders, as well as serving its establishment purposes.

### 2.4. Mission

By focusing primarily on the needs of the Turkish Armed Forces; to provide high-value-added, innovative and reliable products and solutions to both local and foreign customers in the fields of electronic technologies and system integration; continuing activities in line with global targets as well as increasing brand awareness and contributing to the technological independence of Turkey.

### 2.5. Shareholding Structure and Financial Status

Turkish Armed Focus Foundation owns 74.20% of ASELSAN's shares, while 25.80% of the shares are publicly traded shares in Borsa İstanbul.

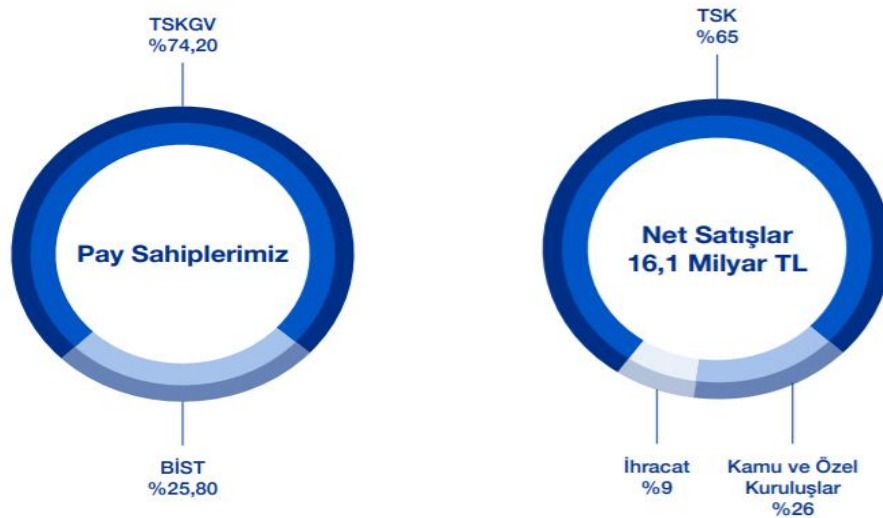


Figure 1. Shareholding Structure and Financial Status of ASELSAN

## 2.6. General Description of the Company and Division

ASELSAN is an establishment of the Turkish Armed Forces Foundation. ASELSAN provides high-value-added, innovative and reliable products and solutions to both local and foreign customers in the fields of electronic technologies and system integration by focusing primarily on the needs of the Turkish Armed Forces. It is aimed to continue its activities in line with global targets, to increase brand awareness and to contribute to Turkey's technological independence.

According to Defense News Top 100, ASELSAN is the 48th largest defense industry company in the world.

ASELSAN UGES Security Systems, which pioneered the sector by using the developing technology in its products, constantly develops new products within the scope of R&D activities financed by the company's own resources, expands the scope of the integrated security system, and offers end-to-end solutions to meet the security needs of many companies in different sectors.

## 2.7. Organization

In line with the company's vision of becoming a global company, the organization is structured to strengthen the critical technological capabilities of the company, to reach the new technological capabilities, to increase its research and development capabilities at the global level. Each sector presidency has its own design, production and testing departments.

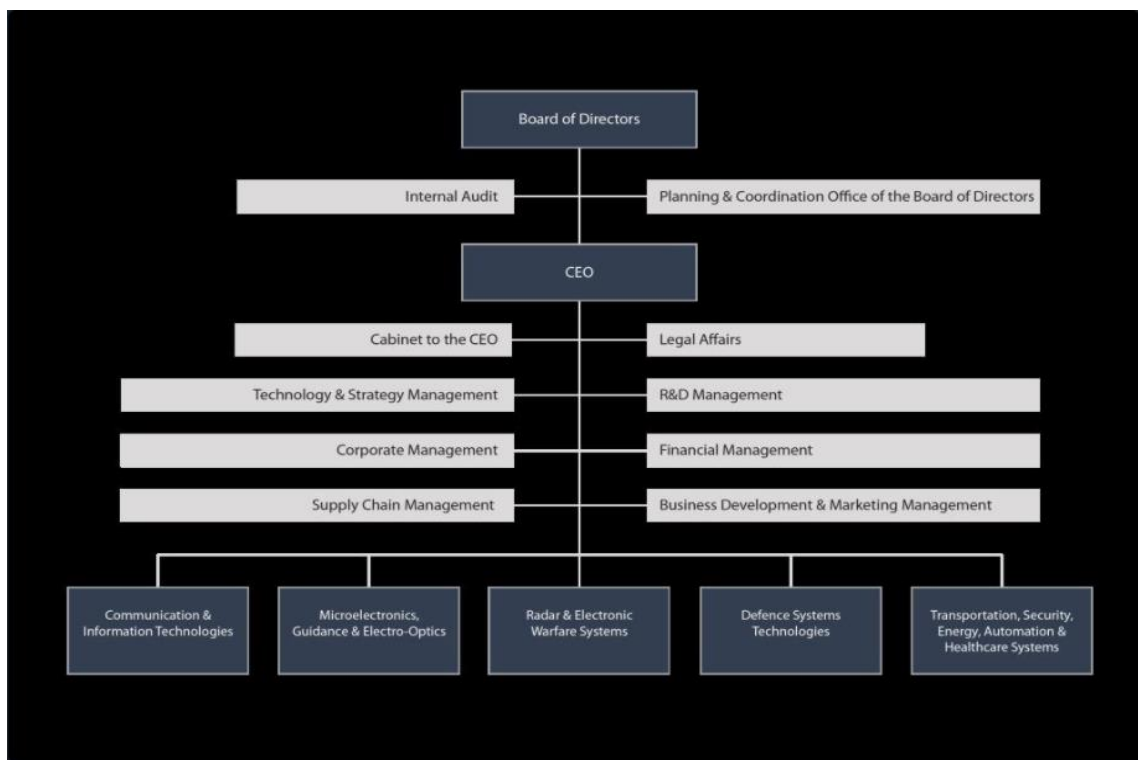


Figure 2. Organizational Structure of ASELSAN

## 2.8. Capabilities

Healthcare Systems	Microwave Products
Information Technology Systems	Energy Systems
Avionics and Navigation Systems	Radar Systems
Air and Missile Defense Systems	Security Systems
Traffic & Automation Systems	Electro Optic Systems
Command Control Communication Computer Systems	Space Technologies
Military Communication Systems	Unmanned Systems
Land and Weapon Systems	Electronic Warfare
Security Systems	Naval Systems
Public Safety Communication Systems	
Systems Transportation Systems Cryptography and Information	

## 2.9. History of the Company

In 1978, the first premises in Macunköy Facility were completed and the manufacturing operation started.

In 1980, the first manpack and tank wireless radios were delivered to the Turkish Armed Forces.

In 1981, the first hand-held radio and Bank Alarm Systems were designed.

In 1983, the first export was realized.

In 1986, ASELSAN contributed to the power of the Turkish Armed Forces with the Electronic Warfare and Data Terminal appliances it developed.

In 1988, ASELSAN produced the first avionic appliance for the F-16 program.

In 1997, ASELSAN 1919 Mobile Phone was launched to the market.

In 1998, thermal cameras, thermal weapon sight and thermal vision devices with target coordination addressing devices were submitted to the use of Turkish Army.

In 2003, agreements covering big projects such as the F-16 electronic warfare auto defense system, military police integrated communication and information system were executed.

In 2008, ATAK agreement and multi-band digital common wireless radio project were executed and ASELSAN delivered the first originally developed Air Defense Radar.

In 2013, ASELSAN has continued its climb for the aim of being one of the top 50 defense companies and ranked 74th according to annual sales.

In 2021, ASELSAN became 48th according to "Defense News Top 100" ranking.

### 3. Test Engineering

A test engineer design, build, and updates test cases to make them more efficient, reusable, and reliable. The test engineer's tasks include designing a test case to test the program and creating a test case, creating the expected results corresponding to each scenario according to the analysis. In other words, test engineers are the people who perform tests on the developed software. They work to ensure that the products created to meet the user requests and ensure the satisfaction of the users. By playing an active role in the analysis process, they control the product developed from the very beginning to the last moment before it reaches the customer. Test automation is the process of performing a test plan without the need for manual intervention. Test cases are written and executed using automation tools. The goal of test automation is to improve software value by increasing testing efficiency. When a system is operated manually, problems might occur due to the excessive amount of code and detail. Thanks to automation, the test team can prevent human errors and complete checks in a shorter amount of time. Continuously developing systems will require repeated execution of the same test plan. It is possible to save this test plan and replay it as needed by using a test automation tool. Based on requirements-related artifacts, test automation engineers develop an automated testing approach, automation test plans, and test scripts.

### 4. Security Systems

ASELSAN is in the security systems field of activity with over 40 years of experience. It offers unique and integrated solutions to public institutions and organizations, and private companies in areas such as facility security, city security, border security, coast security, police station security, pipeline security, mobile security systems. Security systems division carries out activities such as design, development, manufacturing, test, deployment, and integrated logistic support for security and surveillance projects. In first project, I worked on a management client interface which is developed by ASELSAN for surveillance. For privacy reasons, I couldn't share any names, photos, and details in this report. But I will explain in detail most of the parts related to my work. In this report, I will use the name "CCTV monitoring and client user interface" for this program. Figure 3 gives a general idea about this surveillance program.



- Access to all system components with a single UI
- User friendly interface.
- User interface for system management
- Basic alarm and event management UI
- Export UI

Figure 3. CCTV monitoring

## 5. First Project

Occupational health and safety and electrostatic discharge training was given in the first three days of the summer practice. Orientation training was given in the remaining 2 days of the first week. I had the opportunity to get to know test engineering closely. No specific task was given in the first week due to insufficient computers and space. During this time, my supervisor engineer asked an intern who was about to finish his internship to tell me what he had done during his internship. I was going to use Ranorex just like him. In short, I analyzed the work of the intern before me in the first week and got to know Ranorex.

At the beginning of the 2nd week, my supervisor engineer showed me Ranorex and the CCTV monitoring and client user interface, explained in detail and gave me my task. My task was to perform fast-forward tests in synchronous recorded playback mode in the CCTV monitoring and client user interface using Ranorex.

I have never worked on Ranorex and C# before. Here, I first examined thoroughly the codes that my supervisor engineer had written before. In other words, I started to duty by reverse engineering concept. With this, I became familiar with Ranorex and C#, and I understood better what to do and how to do it.

### 5.1. Ranorex: Test Automation for GUI Testing

Ranorex Studio test automation tools are flexible and give us permission to do all-in-one UI-testing where we can run automated tests smoothly and continuously across all devices. Ranorex automated testing tools allow us to automate our desktop, web and mobile UI testing by recording our UI testing actions without having to write any code. The core functionality of Ranorex is an Application Programming Interface based on the .NET framework.

The test automation tools generate C# code. Ranorex's object recognition gives us to identify the UI elements of our software application, regardless of whether it is a web, mobile, or Windows-based UI. Testers and Developers can easily write and run UI tests in their preferred environment. Simple automation modules can be easily shared between developers and testers.

There are 2 options to prepare test automation with Ranorex. These are the record module and the code module. By using the record module in Ranorex, we can find elements in Graphical User Interfaces (GUI ) easily and quickly. We may prefer to use a code module in places that require repetition. By the way, Ranorex supports C# as its programming language. It also allows us to use both modules together.



## 5.2. Overview for Fast Forward Tests

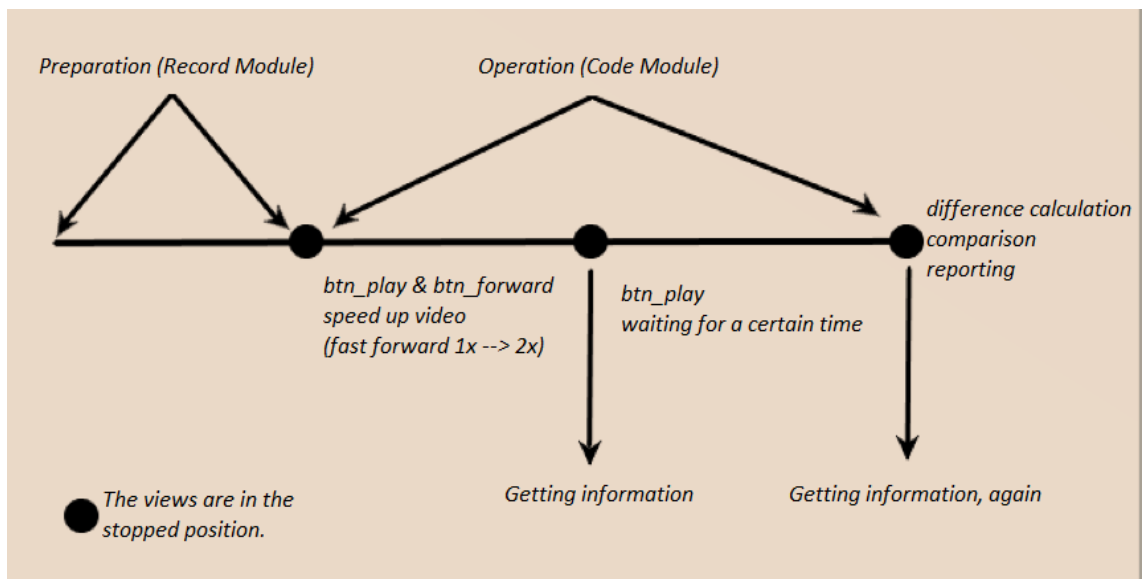
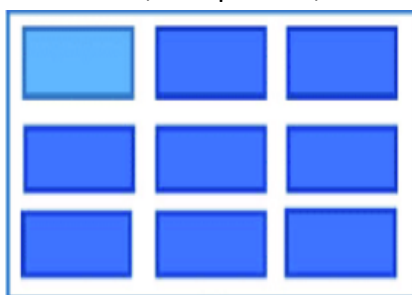


Figure 4. Overview for fast forward tests

In Figure 4, I have prepared a diagram showing the path I followed to give a general idea. The project will be explained in two parts as preparation and operation. While I use the record module in the preparation part, I use the code module in the operation part. The general process goes as follows. There is a preparation process at the beginning. At the end of the preparation process, the views are paused situation. Afterward play, accelerate and pause operations are performed respectively. After stopping, I get information such as time, current video speed, I-frame value, P-frame value. Then, I play the video and wait for a certain time. Again, I pause it and get the information. In the last part of the project, I do operations such as difference calculation, comparison, and reporting.



As shown in Figure 5, there are 9 video screens for 3\*3 workspace patterns. For all cameras, all these operations must be done synchronously, and all these operations must also be done for 2,4,8,16,32 speeds

Figure 5. Workspace pattern for 3x3

## 5.3. Preparation Process for Fast Forward Tests

In the preparation part, the operations that I made in CCTV monitoring and client user interface are opening the sync menu, entering and selecting camera names, entering the start date and time, entering the end date and time into the system, selecting the screen workspace pattern, performing operations such as play and stop, opening camera statistics etc.



### 5.3.1 Data Source and Data Binding

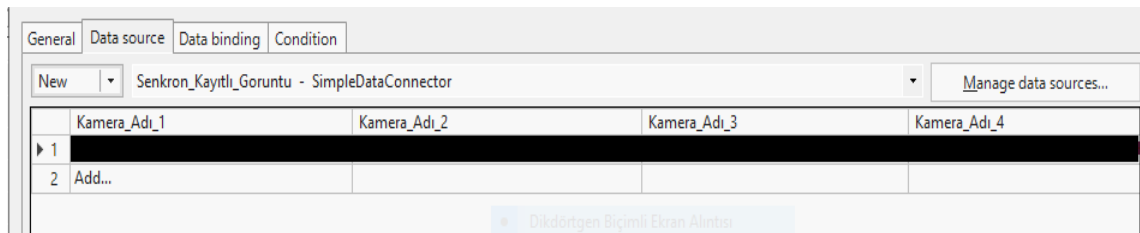


Figure 7. Data source tab in Ranorex

This is one of my favorite topics on Ranorex. Generally, the tests we run will be based on different input conditions. We test the application against different input conditions, then try to validate the output. In the data source part, we can assign a variable and tell it its value. For example, here there is a variable that I have assigned as Camera\_Name 1. In Figure 7, I draw a black bar on it because there is a camera IP and it is forbidden for me to share it.

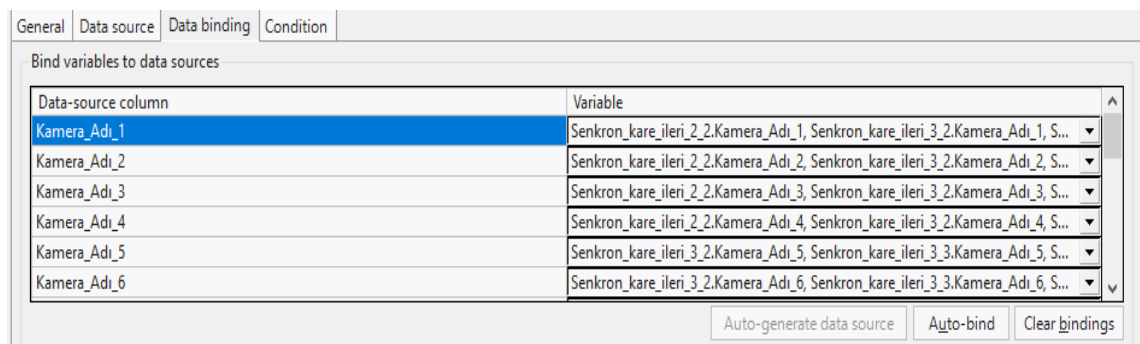


Figure 8. Data binding tab in Ranorex

In the data binding tab shown in Figure 8, I match the module variable to the data source column. Now the input values are taken from the data source when the camera names are called. The purpose of this is to make data-driven testing easier and more maintainable. These features show that Ranorex is an easy-to-use, efficient, and powerful test automation program.

In data-driven testing, a test case takes input values from a data source. The test is automatically repeated for each row of data in the data source. The essential components of a data-driven test are variables, data sources, and data binding. Data-driven testing allows us to further control test execution.

Test cases involve recording modules containing the actions performed during testing and the repository items from which these actions were performed. An external data source that provides the test data can be bound to variables that are made of these actions and repository items.

### 5.3.2 Characteristics of the Camera Streams

At the end of the preparation part, I open the camera statistics with Ranorex. Due to company confidentiality, any photo with camera statistics cannot be shared here. Camera statistics include information such as: Received I and P frames, Decoded I and P frames, Displayed I and P frames, Bitrate, FPS, Time, Playback Speed, etc. At this point, I've done some research on these concepts. All these concepts will be checked quantitatively during the test, except bitrate. Received represents all incoming frames. Decoded represents frames whose files can be opened. Displayed represents frames that can be projected onto the screen.

JPEG and MPEG are two different types of compression formats. JPEG is mainly used for image compression. On the other hand, MPEG has various standards for audio and video compression. MPEG output consists of three kinds of frames called I-frames, P-frames, B-frames. The concept of I-frames, P-frames, and B-frames is fundamental to the field of video compression.

I-frames (Intracoded) are just still pictures. They can be coded with JPEG or something similar. In I-frames, macroblocks are intra-coded. I-frames are much larger than P or B frames, but they are decodable independent of other frames. This interdependence between frames has a serious implication for the Fast Forward mechanism based on frame skipping. Many implementations of Fast Forward work on the principle of transmitting only the I-Frames.

P-frames (Predictive) refer to previously encoded I/P-frames. In [1], a useful example about P-frames is. In Figure 9, there are three consecutive frames that have the same background. The only difference is in the position of one person. The macroblocks containing the background scene will be the same, but the macroblocks containing the person will be offset in position by some unknown amount and will have to be tracked down.



Figure 9. Three consecutive frames.

B-frame (Bidirectional) can refer to frames that occur both before and after it. B frame has the highest compression ratio. It only reflects the change of the main motion body between reference frames. In B-frames, macroblock is one of the five possible types: forward predicted, backward predicted, bi-directional predicted, intra-coded, and skipped.

FPS: FPS is a key metric that indicates how many video frames per second a security camera can capture. It determines how smooth the video looks. For example, the camera I'm working with takes 25fps, 25 frames in one second. 24 P-frames change while 1 I-frame changes per second.

Bitrate: The video bitrate represents the amount of data transferred in a certain period. We measure bitrate in bits-per-second (bps). Bitrate is a very important concept for video quality. More bits per second means better quality on the screen.

GOP: AVC and HEVC, involve grouping a series of frames into sequences called GOP. GOP stands for a group of pictures and specifies the order in which intra-frames and inter-frames are arranged. File sizes can be reduced by processing and compressing GOPs.

As described in [2], A codec converts the light and sound recorded by the camera into a digital format. Computers then read the format and play it back to us as it was originally captured. I worked with two different codecs which are HEVC (High Efficiency Video Codec) also known as H.265 and AVC (Advanced Video Codec) also known as H.264. Then I made some research on their differences. The current encoding standard for video streaming is H.264. H.265 is a more advanced encoding method than H.264. H.265 reduces the file size of live video streams and thus helps reduce the required bandwidth. This is extremely beneficial for high resolutions and live streaming.

Application layer messages are carried between application endpoints by the transport layer of the internet. There are two transport protocols on the Internet, TCP and UDP, either of which can transport application-layer messages. TCP and UDP are both in use for IP cameras in ASELSAN. TCP provides a connection-oriented service. This service includes the guaranteed delivery and control of application layer messages to the destination. TCP also splits long messages into shorter segments and provides a congestion control mechanism. Recorded video tends to have fewer errors with TCP. On the other hand, the UDP protocol provides connectionless service. This is a simple service that does not provide reliability, flow control and congestion control. Live viewing tends to be smoother with UDP. More information about UDP and TCP can be found in [1] and [3].

There are three transmission modes of IP packets called unicast, multicast, and broadcast. I've worked on unicast and multicast. Unicast is a type of communication where data is sent from one point to another in the network. In this case, there is just one sender, and one receiver. Multicast is the term used to describe one-to-many communication. In this case, the sender transmits only one copy of data to many devices that are interested in that traffic.

## 5.4. Operation Process for Fast Forward Tests

RECORD

RUN RECORDING

+ Add new action

✂

📄

🗑

↶

↷

⬆

⬇

Turbo mode

Screenshot

#	Action	
47	Mouse	Click
48	Mouse	Click
49	User code	Statistics_Of_hızlı_ileri_2x2 (RepoItemInfo textInfo)

<

Figure 10. The code module after the recording modules

After the preparation part, I start the operation part. As you can see in the Figure 10, I use Statistics\_Of\_hızlı\_ileri\_2x2 from the user code after the recording modules. I will show the functionality of the code by explaining the important parts of the code, I will not dwell on the repetitive parts in detail.

```
[UserCodeMethod] //kare geri'de en son görüntü pause durumdadır.
public static void Statistics_Of_hızlı_ileri_2x2(RepoItemInfo textInfo)
{
    for (int i = 0; i <= 4; i++)
    {
        senkron_hızlı_ileri_2x2( textInfo,Convert.ToInt32(Math.Pow(2,i+1)));
    }
}
```

Figure 11. Beginning of the code

In Figure 11, there is a code written for 2x2 workspace pattern. The code starts here and calls another function, named senkron\_hızlı\_ileri\_2x2, for playback speeds of 2,4,8,16,32. This is desirable as it is intended to be tested at these speeds.

```
[UserCodeMethod] //hızlı ileri sonrası en son görüntü pause durumdadır.
public static void senkron_hızlı_ileri_2x2( RepoItemInfo textInfo, int forwardSpeed)
{
    int sleepDurationInSeconds = Convert.ToInt32(TestSuite.Current.CurrentTestContainer.DataContext.CurrentRow["sleepDurationInSeconds1"]);
    Ranorex.ReportLevel ReportPass = new Ranorex.ReportLevel("Pass",110,"color:green; font-weight:bold");
    Ranorex.ReportLevel ReportFail = new Ranorex.ReportLevel("Fail",120,"color:red; font-weight:bold");
    Ranorex.ReportLevel Title = new Ranorex.ReportLevel("Title",55,"color:black; font-weight:bold");
    Ranorex.ReportLevel ReportData = new Ranorex.ReportLevel("Data",20,"color:black; font-weight:bold");
    string[] statistics = new string[22];
    int hücre_sayisi = 4 ;
    Ranorex.Text[] hücreler = new Ranorex.Text[4];
    hücreler[0] = repo.FormTransparentText0.LblCameraInfo0 ;
    hücreler[1] = repo.FormTransparentText1.LblCameraInfo1 ;
    hücreler[2] = repo.FormTransparentText2.LblCameraInfo2 ;
    hücreler[3] = repo.FormTransparentText3.LblCameraInfo3 ;
    int timeFirst, timeSecond;
```

Figure 12. Variable assignments

If we look at *senkron\_hızlı\_ileri\_2x2* shown in Figure 12 in detail, we can see that the waiting time is taken from the data source. Then, I specify what will appear in the last report section, how it will look (Pass, Fail, Title, Data). I define a string array called Statistics.

The reason it is 22 is that there are 22 lines in the camera statistics. Ranorex creates a repository for each video screen where the camera statistics appear. I am separating the camera information from the repository into cells (hücreler). There are 4 cells for the video screens 2x2 workspace pattern. Later, I will use them to hold the information.

```
repo.DEFNEAselsanVideoİzlemeVeYonetim.Panel1.BtnPlay.Click();
System.Threading.Thread.Sleep(2000);
repo.DEFNEAselsanVideoİzlemeVeYonetim.BtnForward.Click();
System.Threading.Thread.Sleep(2000);
repo.DEFNEAselsanVideoİzlemeVeYonetim.Panel1.BtnPause.Click();
```

Figure 13. Play, speed-up and re-pause operations

Play, speed-up and re-pause operations are performed respectively as shown in Figure 13.

(btn\_play, btn\_forward, btn\_pause & fast forward 1x->2x ...)

```
Report.Log(Title, "Title", "KAYITLI GÖRÜNTÜ HIZLI İLERİ - I FRAME KONTROLÜ HIZ: " + forwardSpeed.ToString()+"x", textInfo);
for (int k=0; k < hücre_sayisi; k++)
{
    statistics = (hücreler[k].TextValue).Split('\n');
    int start = statistics[22].LastIndexOf(":")+1;
    int stop = statistics[22].Length;
    int currentSpeed = Convert.ToInt32(statistics[22].Substring(start, stop-start));
    if (currentSpeed != forwardSpeed)
    {
        Report.Log(ReportFail, "HIZ HATALI", "Olması Gereken Hız: "+forwardSpeed.ToString() + ". Şu anki hız: " +currentSpeed.ToString(), textInfo);
    }
    else

```

Figure 14. Starting the loop

At the beginning of the Figure 14, I log an article that shows which playback speed test will be performed. Since the operations must be performed for each camera, it is looping as many times as the number of cells. I split the information into the cells and throw them into the string array called statistics. Then, I read current speed with index and substring methods. If the current speed is different from the playback speed I tested, I report "Wrong Speed"(Hız Hatalı). Moreover, I report the speed which is tested (Olması Gereken Hız) and the current speed (Şu anki hız).

```

else
{
#region indexes
int RI = statistics[0].IndexOf('I')+"I:".Length; //Index of Received I
int RP1 = statistics[0].LastIndexOf('P')+"-".Length; //Index of Received P
int RP2 = statistics[0].IndexOf('P')+"P:".Length; //Index of Received P
int A = statistics[0].LastIndexOf('A')+"-".Length; //Index of Received A
int DCI = statistics[1].IndexOf('I')+"I:".Length; //Index of Decoded I
int DCP1 = statistics[1].LastIndexOf('P')+"-".Length; //Index of Decoded P
int DCP2 = statistics[1].Length-1; //Index of Decoded P
int DPI = statistics[2].IndexOf('I')+"I:".Length;
int DPP1 = statistics[2].LastIndexOf('P')+"-".Length;
int DPP2 = statistics[2].Length-1;
int T1 = statistics[21].LastIndexOf(':');
#endregion

```

Figure 15. IndexOf method

If the current speed is the same as the playback speed that I tested, the I and P frames for received decoded and displayed should now be read. I'm looking for a specific character in a string with the IndexOf method. IndexOf method is a string method in C#. This method is used to find the zero-based index of the first occurrence of a specified character or string. It returns the index number of the character it finds in int type. If the searched character is not in the string value, "-1" is returned. As depicted in Figure 15, String.IndexOf and String.LastIndexOf methods are used. The String.IndexOf method searches for the specified character in the string expression. The String.LastIndexOf method has the same logic, the only difference is it starts from the end.

```

#region variables
int recI_First=Convert.ToInt32(statistics[0].Substring(RI,RP1-RI));
int decI_First=Convert.ToInt32(statistics[1].Substring(DCI,DCP1-DCI));
int disI_First=Convert.ToInt32(statistics[2].Substring(DPI,DPP1-DPI));
int recP_First=Convert.ToInt32(statistics[0].Substring(RP2,A-RP2));
int decP_First=Convert.ToInt32(statistics[1].Substring(DCP1+3,DCP2-DCP1-2));
int disP_First=Convert.ToInt32(statistics[2].Substring(DPP1+3,DPP2-DPP1-2));

string longTime1 = statistics[21].ToString();
timeFirst = findTimeInMiliseconds(statistics[21],T1) / 1000;|
#endregion

```

Figure 16. Substring method

Three types of methods are used as illustrated in Figure 16, namely Convert.ToInt32, Substring, ToString.

The Convert.ToInt32 converts an object to an integer and returns 0 if the value was null.

The String.Substring method retrieves a substring from a string instance in C#. It retrieves a substring from this instance from the specified position for the specified length. The substring method does not change the original string.



In C#, the ToString method is used to get instance of String.

I read I and P frame values for received, decoded and displayed by using IndexOf and Substring methods. I also record the first time the information was received, both as an integer and a string. I'll use them later in the difference calculation and comparison sections.

As it can be seen from Figure 16, another function called *findTimeInMilisecond* is used.

```
[UserCodeMethod]
public static int findTimeInMiliseconds(string data,int T1)
{
    int T_sa=Convert.ToInt32(data.Substring(T1-8,2))*3600000;
    int T_dk=Convert.ToInt32(data.Substring(T1-5,2))*60000;
    int T_sn=Convert.ToInt32(data.Substring(T1-2,2))*1000;
    int T_ms=Convert.ToInt32(data.Substring(T1+1,3))*1;
    return (T_sa + T_dk + T_sn + T_ms);
}
```

Figure 17. Finding time in milliseconds

In Figure 17, the function simply takes the values of hours (sa), minutes (dk), seconds (sn), and milliseconds (ms) as strings. Then it returns them to an integer.

The function converts hours to milliseconds by multiplying by 3600000.

The function converts minutes to milliseconds by multiplying by 60000.

The function converts seconds to milliseconds by multiplying by 1000.

The function converts milliseconds to milliseconds by multiplying by 1.

Also, it returns their sum as an integer.

```
Report.Log(ReportData,"TIME", "TIME: " + longTime1, textInfo);
Report.Log(ReportData,"I FRAME", "REC-DEC-DIS I FRAME: " + recI_First + "-" +decI_First+"-"+disI_First, textInfo);
Report.Log(ReportData,"P FRAME", "REC-DEC-DIS P FRAME: " + recP_First + "-" +decP_First+"-"+disP_First, textInfo);

repo.DEFNEAselsanVideoİzlemeVeYonetim.Panel1.BtnPlay.Click();
System.Threading.Thread.Sleep(sleepDurationInSeconds*1000);
int butonbulma1 = findTimeInMiliseconds(System.DateTime.Now.ToString("yyyy.MM.dd HH:mm:ss:fff"),19);
repo.DEFNEAselsanVideoİzlemeVeYonetim.Panel1.BtnPause.Click();
int butonbulma2 = findTimeInMiliseconds(System.DateTime.Now.ToString("yyyy.MM.dd HH:mm:ss:fff"),19);
double butonsurefark1 = Math.Round((butonbulma2 - butonbulma1)/1000.0, 1) ;
Report.Log(ReportData,"butonbulma", "butonbulmasuresi: " + butonsurefark1.ToString() + " saniyedir.", textInfo);
```

Figure 18. Reporting statistics and button finding time

After the camera statistics are taken for the first time, they should be logged into the report. As provided in Figure 18, I and P frame values for received, decoded, and displayed are logged to report. The time is also logged here.

After the first stats are recorded, I move on to the waiting part. Here are my steps in order: play video streams, wait 10 seconds, stop video streams.

There is an incidence like button finding time here because there is some delay in Ranorex's finding and pressing the button. For example, it finds and clicks 11 seconds later, instead of exactly 10 seconds later. Therefore, it is important and necessary to account for button find and click time. I get a value immediately after 10 seconds, and a value right after pressing the button. As reflected in Figure 18, these values are also returned from the *findTimeInMilliseconds* function. Next, finding the difference of these values is done because this difference shows us the button finding time. Then I log this value as well.

In Figure 18, we see the *Math.Round* function which returns the value of a number rounded to the nearest integer.

```
statistics = (hücreler[k].TextValue).Split('\n');
indexes

variables

Report.Log(ReportData,"TIME", "TIME: " + longTime2, textInfo);
Report.Log(ReportData,"I FRAME", "REC-DEC-DIS I FRAME: " + recI_Second + "-" + decI_Second + "-" + disI_Second, textInfo);
Report.Log(ReportData,"P FRAME", "REC-DEC-DIS P FRAME: " + recP_Second + "-" + decP_Second + "-" + disP_Second, textInfo);

differenceCalculation

//P frame'de degisiklik olmaması lazım.

Karsilastirma
}
```

Figure 19. Reporting statistics for the second time

After receiving the statistics for the first time and waiting for a certain time, it is now necessary to get the statistics for the second time. All methods such as index and substring used in the first part are used here as well. In other words, the same logic is used here to get the statistics for the second time. According to Figure 19, I report the values of time, I-frame, and P-frame here as well. Then I'll move on to the difference calculation and comparison parts.

```
#region differenceCalculation
double sleepDuration = sleepDurationInSeconds;
double min = ((sleepDuration+butonsurefarki)*1.0) - 1; //aradaki tıklama delayinden dolayı fark eklenmiştir.
double max = ((sleepDuration+butonsurefarki)*1.0) + 1; //aradaki tıklama delayinden dolayı fark eklenmiştir.
double result_recI = (recI_Second-recI_First) / (forwardSpeed*1.0);
double result_decI = (decI_Second-decI_First) / (forwardSpeed*1.0);
double result_disI = (disI_Second-disI_First) / (forwardSpeed*1.0);
double timeDiff = (timeSecond - timeFirst) / (forwardSpeed*1.0);
#endregion
```

Figure 20. Difference calculation

In Figure 20, after adding the sleepDuration time and the button finding time (butonsurefarki), I assign plus 1 second to the max value and minus 1 second to the min value. The min and max range will be the pass range for the test. For received, decoded, and displayed, I take the difference of the initial and final I-frame values and divide by the playback speed. For example, in 10 seconds, 10 frames pass at 1x speed, while 20 frames pass at 2x speed. I need to get 10 instead of 20 to use in the comparison part. The same situation is true for the time as well. However, it is not valid for P-frames because P-frames don't change in fast forward i.e. 2x, 4x, 8x, 16x, 30x playback speeds. The research in [5] presents alternative fast forward algorithms and visualizations that are possible with digital video because it allows for random access to the media. Fast forward algorithms in video streaming formats (like H.264) use only I-frames to sample the video at a faster than normal speed. In addition to the I-frames displayed, when other frames than I-frames are transmitted, it offers extra network overhead and extra computational complexity in the video transcoder.

```
#region Karsilastirma
if ( (recP_Second == recP_First) && (decP_Second == decP_First) && (disP_Second == disP_First) )
{
    Report.Log(ReportPass, "P FRAME", "P Frame is not changed", textInfo);
    //zaman farkı, I frame farkı kontrolü
    if ( ((min <= timeDiff) && (timeDiff <= max)) )
    {
        Report.Log(ReportPass, "TIME", "START TIME: " + longTime1 + " ---- END TIME : " + longTime2, textInfo);
    }
    else
    {
        Report.Log(ReportFail, "TIME", "START TIME: " + longTime1 + " ---- END TIME : " + longTime2, textInfo);
    }

    if ( ((min <= result_recI) && (result_recI <= max)) )
    {
        Report.Log(ReportPass, "RECEIVED I FRAME DIFFERENCE", "First: " + recI_First.ToString() + "--- Second: " + recI_Second.ToString(), textInfo);
    }
    else
    {
        Report.Log(ReportFail, "RECEIVED I FRAME DIFFERENCE", "First: " + recI_First.ToString() + "--- Second: " + recI_Second.ToString(), textInfo);
    }

    if ( ((min <= result_decI) && (result_decI <= max)) )
    {
        Report.Log(ReportPass, "DECODED I FRAME DIFFERENCE", "First: " + decI_First.ToString() + "--- Second: " + decI_Second.ToString(), textInfo);
    }
    else
    {
        Report.Log(ReportFail, "DECODED I FRAME DIFFERENCE", "First: " + decI_First.ToString() + "--- Second: " + decI_Second.ToString(), textInfo);
    }

    if ( ((min <= result_disI) && (result_disI <= max)) )
    {
        Report.Log(ReportPass, "DISPLAYED I FRAME DIFFERENCE", "First: " + disI_First.ToString() + "--- Second: " + disI_Second.ToString(), textInfo);
    }
    else
    {
        Report.Log(ReportFail, "DISPLAYED I FRAME DIFFERENCE", "First: " + disI_First.ToString() + "--- Second: " + disI_Second.ToString(), textInfo);
    }
}
```

Figure 21. Comparison

As can be seen from the Figure 21, the comparison part starts here. It is first checked whether the P-frame value has changed. If the P-frame value has not changed, I report "Pass, P frame is not changed". If the time difference is between min and max values, I assign it as pass, otherwise, I assign it as fail. Also in both cases, the time in the first and second cases is reported. If the received I-frame value which is taken from the difference calculation part is in the min and max range, I assign it as pass. I report "Received I frame difference". I also report the received I values in the first and second cases. A procedure with the same logic is processed for decoded and displayed I-frames.

```
else
{
    Report.Log(ReportFail, "P FRAME", "P Frame change is not correct, check this.", textInfo);
    Report.Log(ReportFail, "Received_P", "Check This: " + "First: " + recP_First.ToString() + "--- Second: " + recP_Second.ToString(), textInfo);
    Report.Log(ReportFail, "Displayed_P", "Check This: " + "First: " + decP_First.ToString() + "--- Second: " + decP_Second.ToString(), textInfo);
    Report.Log(ReportFail, "Decoded_P", "Check This: " + "First: " + disP_First.ToString() + "--- Second: " + disP_Second.ToString(), textInfo);
}
```

Figure 22. Reporting if P-frame changes

As illustrated in Figure 22, if P-frame value changes, I report "Fail, P frame change is not correct, check this". I report the P-frame values in the first and second states for received, decoded, and displayed. The code and operation process for the fast-forward tests of the 2x2 workspace pattern is completed here.

01:26.837	Fail	HIZ HATALI	Olması Gereken Hız: 2. Şu anki hız: 1
01:26.887	Fail	HIZ HATALI	Olması Gereken Hız: 2. Şu anki hız: 1
01:26.937	Fail	HIZ HATALI	Olması Gereken Hız: 2. Şu anki hız: 1
01:34.725	Title	Title	KAYITLI GÖRÜNTÜ HIZLI İLERİ - I FRAME KONTROLÜ HIZ: 4x
01:34.782	Fail	HIZ HATALI	Olması Gereken Hız: 4. Şu anki hız: 2
01:34.853	Fail	HIZ HATALI	Olması Gereken Hız: 4. Şu anki hız: 2
01:34.918	Fail	HIZ HATALI	Olması Gereken Hız: 4. Şu anki hız: 1
01:34.970	Fail	HIZ HATALI	Olması Gereken Hız: 4. Şu anki hız: 1
01:42.694	Title	Title	KAYITLI GÖRÜNTÜ HIZLI İLERİ - I FRAME KONTROLÜ HIZ: 8x
01:42.762	Fail	HIZ HATALI	Olması Gereken Hız: 8. Şu anki hız: 2
01:42.843	Fail	HIZ HATALI	Olması Gereken Hız: 8. Şu anki hız: 2
01:42.882	Fail	HIZ HATALI	Olması Gereken Hız: 8. Şu anki hız: 1
01:42.937	Fail	HIZ HATALI	Olması Gereken Hız: 8. Şu anki hız: 1
01:50.759	Title	Title	KAYITLI GÖRÜNTÜ HIZLI İLERİ - I FRAME KONTROLÜ HIZ: 16x
01:50.826	Fail	HIZ HATALI	Olması Gereken Hız: 16. Şu anki hız: 2
01:50.875	Fail	HIZ HATALI	Olması Gereken Hız: 16. Şu anki hız: 2
01:50.927	Fail	HIZ HATALI	Olması Gereken Hız: 16. Şu anki hız: 1
01:50.974	Fail	HIZ HATALI	Olması Gereken Hız: 16. Şu anki hız: 1
01:58.828	Title	Title	KAYITLI GÖRÜNTÜ HIZLI İLERİ - I FRAME KONTROLÜ HIZ: 32x
01:58.888	Fail	HIZ HATALI	Olması Gereken Hız: 32. Şu anki hız: 2
01:58.952	Fail	HIZ HATALI	Olması Gereken Hız: 32. Şu anki hız: 2
			Olması Gereken Hız: 32. Şu anki hız: 1

Figure 23. Report showing fail

When I ran all the record and code modules I prepared in Ranorex, I got a report like in Figure 23. When I researched the reason why it fails, what I noticed is that when we stop after pressing the fast forward button in synchronous recorded video playback mode at CCTV monitoring and client user interface, the speed returns to 1x. In other words, testing is done for playback speeds of 1, 2, 4, 8, 16, 32 respectively. However, the current speed in CCTV monitoring and client user interface gets 1, 2, 1, 2, 1, 2.

When I talked about this issue with my supervisor engineer, he told me that this was a software error, and he would report this to the software developers. He also stated that my test was as it should be.

01:12.393	Title	Title	KAYITLI GÖRÜNTÜ HIZLI İLERİ - I FRAME KONTROLÜ HIZ: 2x
01:12.433	Data	TIME	TIME: Time - 24.08.2021 08:00:33:828
01:12.487	Data	I FRAME	REC-DEC-DIS I FRAME: 5-5-5
01:12.540	Data	P FRAME	REC-DEC-DIS P FRAME: 42-34-34
01:22.588	Data	butonbulma	butonbulmasuresi: 0 saniyedir.
01:22.653	Data	TIME	TIME: Time - 24.08.2021 08:00:53:783
01:22.704	Data	I FRAME	REC-DEC-DIS I FRAME: 25-25-25
01:22.755	Data	P FRAME	REC-DEC-DIS P FRAME: 42-34-34
01:22.807	Pass	P FRAME	P Frame is not changed
01:22.861	Pass	TIME	START TIME: Time - 24.08.2021 08:00:33:828 ----- END TIME : Time - 24.08.2021 08:00:53:783
01:22.910	Pass	RECEIVED I FRAME DIFFERENCE	First: 5--- Second: 25
01:22.955	Pass	DECODED I FRAME DIFFERENCE	First: 5--- Second: 25
01:23.001	Pass	DISPLAYED I FRAME DIFFERENCE	First: 5--- Second: 25
01:23.053	Data	TIME	TIME: Time - 24.08.2021 08:00:54:549
01:23.102	Data	I FRAME	REC-DEC-DIS I FRAME: 26-26-26
01:23.155	Data	P FRAME	REC-DEC-DIS P FRAME: 39-33-33
01:33.207	Data	butonbulma	butonbulmasuresi: 0 saniyedir.
01:33.254	Data	TIME	TIME: Time - 24.08.2021 08:01:15:512
01:33.305	Data	I FRAME	REC-DEC-DIS I FRAME: 47-47-47
01:33.352	Data	P FRAME	REC-DEC-DIS P FRAME: 39-33-33
01:33.410	Pass	P FRAME	P Frame is not changed
01:33.456	Pass	TIME	START TIME: Time - 24.08.2021 08:00:54:549 ----- END TIME : Time - 24.08.2021 08:01:15:512
01:33.508	Pass	RECEIVED I FRAME DIFFERENCE	First: 26--- Second: 47
01:33.552	Pass	DECODED I FRAME DIFFERENCE	First: 26--- Second: 47
01:33.605	Pass	DISPLAYED I FRAME DIFFERENCE	First: 26--- Second: 47

Figure 24. Report showing pass

Also, he asked me to do this fast forward test for 2x3, 2x3, 3x3, 3x4, 4x4 workspace patterns. Before moving on to these tasks, I researched to see how I could pass the test in this way. After several unsuccessful attempts, I realized that this fail report has happened because I stopped it in synchronous mode. When I performed the test without stopping in synchronous mode, I saw that there was a pass report like in Figure 24. However, there was a different situation here. First, the information of the first camera is taken and tests are performed. Then, respectively, the second, third, and fourth cameras were tested. In other words, the tests were run sequentially, rather than synchronously. This was not the desired situation, but it was still nice alternative work.

```
[UserCodeMethod] //kare geri'de en son görüntü pause durumdadır.
public static void Statistics_Of_hızlı_ileri_3x3(RepoItemInfo textInfo)
{
    for (int i = 0; i <= 4; i++)
    {
        senkron_hızlı_ileri_3x3( textInfo, Convert.ToInt32(Math.Pow(2,i+1)));
    }
}

[UserCodeMethod] //hızlı ileri sonrası en son görüntü pause durumundadır.
public static void senkron_hızlı_ileri_3x3( RepoItemInfo textInfo, int forwardSpeed)
{
    int sleepDurationInSeconds = Convert.ToInt32(TestSuite.Current.CurrentTestContainer.DataContext.CurrentRow["sleepDurationInSeconds1"]);
    Ranorex.ReportLevel ReportPass = new Ranorex.ReportLevel("Pass",110,"color:green; font-weight:bold");
    Ranorex.ReportLevel ReportFail = new Ranorex.ReportLevel("Fail",120,"color:red; font-weight:bold");
    Ranorex.ReportLevel Title = new Ranorex.ReportLevel("Title",55,"color:black; font-weight:bold");
    Ranorex.ReportLevel ReportData = new Ranorex.ReportLevel("Data",20,"color:black; font-weight:bold");

    string[] statistics = new string[22];

    int hücre_sayisi = 9 ;
    Ranorex.Text[] hücreler = new Ranorex.Text[9];
    hücreler[0] = repo.FormTransparentText0.LblCameraInfo0 ;
    hücreler[1] = repo.FormTransparentText1.LblCameraInfo1 ;
    hücreler[2] = repo.FormTransparentText2.LblCameraInfo2 ;
    hücreler[3] = repo.FormTransparentText3.LblCameraInfo3 ;
    hücreler[4] = repo.FormTransparentText4.LblCameraInfo4 ;
    hücreler[5] = repo.FormTransparentText5.LblCameraInfo5 ;
    hücreler[6] = repo.FormTransparentText6.LblCameraInfo6 ;
    hücreler[7] = repo.FormTransparentText7.LblCameraInfo7 ;
    hücreler[8] = repo.FormTransparentText8.LblCameraInfo8 ;
```

Figure 25. The difference of the code written for the 3x3 from the 2x2

Then I started working for the 2x3, 3x3, 3x4, and 4x4 workspace patterns. While doing these, I directly copied what I did for 2x2. I just made minor modifications. I use functions called *Statistics\_Of\_hızlı\_ileri\_3x3* and *senkron\_hızlı\_ileri\_3x3*. As indicated in the Figure 25, I have defined 9 cells for the 3x3 workspace pattern. Previously there were 4 cells for the 2x2. I completed the 2nd week of my internship by doing these.

## 5.5. Fast Rewind Tests

At the beginning of the 3rd week, my supervisor engineer and gave me another task. The task was to perform fast-rewind tests in synchronous recorded playback mode in the CCTV monitoring and client user interface using Ranorex. In fact, the fast reverse tests were not much different from the fast forward tests. In fast reverse tests, I press the rewind button instead of the forward button. The operations performed in the fast forward tests (preparation process and operation process) are also performed here with the same logic.

```
#region differenceCalculation
double sleepDuration = sleepDurationInSeconds;
double min = ((sleepDuration+butonsurefarkı)*1.0) - 2.0; // *.5x ve .25x olduğu için -1 +1 yerine -2 +2 yerine kullanıldı
double max = ((sleepDuration+butonsurefarkı)*1.0) + 2.0;
double result_recI = (recI_Second-recI_First) / (forwardSpeed);
double result_decI = (decI_Second-decI_First) / (forwardSpeed);
double result_disI = (disI_Second-disI_First) / (forwardSpeed);
double result_recP = (recP_Second-recP_First) / (forwardSpeed*(FPS-1));
double result_decP = (decP_Second-decP_First) / (forwardSpeed*(FPS-1));
double result_disP = (disP_Second-disP_First) / (forwardSpeed*FPS);
double timeDiff = Math.Abs((timeSecond - timeFirst)) / (forwardSpeed);
#endregion

#region karsilastirma
if (((min <= result_recP) && (result_recP <= max)) && ((min <= result_decP) && (result_decP <= max)) && ((min <= result_disP) && (result_disP <= max)))
{
    //zaman farkı, I-P frame farkı kontrolü
    if ( ((min <= timeDiff) && (timeDiff <= max)) )
    {
        Report.Log(ReportPass,"TIME", "START TIME: " + longTime1 + " ---- END TIME : " + longTime2,textInfo);
    }
    else
    {
        Report.Log(ReportFail,"TIME", "START TIME: " + longTime1 + " ---- END TIME : " + longTime2,textInfo);
    }

    //zaman farkı, I-P frame farkı kontrolü
    if ( ((min <= result_recI) && (result_recI <= max)) )
    {
        Report.Log(ReportPass,"RECEIVED I FRAME DIFFERENCE", "First: " + recI_First.ToString() + "--- Second: " + recI_Second.ToString(),textInfo);
    }
    else
    {
        Report.Log(ReportFail,"RECEIVED I FRAME DIFFERENCE", "First: " + recI_First.ToString() + "--- Second: " + recI_Second.ToString(),textInfo);
    }
}
```

Figure 26. Difference calculation and comparison parts of fast-rewind tests

The big change here is at 0.5x and 0.25x playback speeds. At these speeds, both I and P-frames change. As detailed in the Figure 26, we take into account both I and P frames in the difference calculation part. The reason for being FPS-1 is as I mentioned before, the cameras are 25 FPS. In every 1 second, 1 I-frame passes while 24 P-frames pass. In the comparison part, I check whether the P-frames are between min and max values. This part is also different from before. I was checking whether P-frames were equal in fast forward tests. I won't go into detail about the other parts of the fast reverse tests because the logic is the same as for the fast forward tests.



## 5.6. Extra Work on First Project

```
[UserCodeMethod] //kare geri'de en son görüntü pause durumdadır.
public static void goruntu_acilma_suresi(RepoItemInfo textInfo)
{
    Ranorex.ReportLevel ReportPass = new Ranorex.ReportLevel("Pass",110,"color:green; font-weight:bold");
    Ranorex.ReportLevel ReportFail = new Ranorex.ReportLevel("Fail",120,"color:red; font-weight:bold");
    Ranorex.ReportLevel Title = new Ranorex.ReportLevel("Title",55,"color:black; font-weight:bold");
    Ranorex.ReportLevel ReportData = new Ranorex.ReportLevel("Data",20,"color:black; font-weight:bold")

    int first_button=findTimeInMiliseconds(System.DateTime.Now.ToString("yyyy.MM.dd HH:mm:ss:fff"),19);
    while(repo.DEFNEAselsanVideoIzlemeVeYonetim.BtnForward.Enabled == false )
    {
        System.Threading.Thread.Sleep(100);
    }
    int second_button=findTimeInMiliseconds(System.DateTime.Now.ToString("yyyy.MM.dd HH:mm:ss:fff"),19);
    double timediffer = Math.Round((second_button - first_button)/1000.0, 1) ;
    Report.Log(ReportData,"butonbulma", "acilmasuresi: " + timediffer.ToString() + " saniyedir.", textInfo);
}
```

Figure 27. Image opening time function

My supervisor asked me to do extra work on the first project before starting the 2nd project. The purpose of this study was to find out how many seconds does it take the image to appear on the screen after pressing the play button. I add a new code module right after pressing the play button in the record module. Next, I run the code that appears in Figure 27. The buttons are activated when the image appears on the screen for the first time. Using this feature made my extra work easier. First, I save the time value right after pressing the play button. Then I wait until the buttons activate. Again, I save the time value right after the buttons are activated. These values are also returned from the *findTimeInMilisecond* function. Finally, the difference between these values gives us the image opening time and I report it as data. Thus, I have completed the extra work.

## 5.7. Summary of First Project

Senkron_hızlı_ileri_2_2	Bound variables: 8
senkron_hızlı_ileri_2_3	Bound variables: 10
senkron_hızlı_ileri_3_3	Bound variables: 13
senkron_hızlı_ileri_3_4	Bound variables: 16
senkron_hızlı_ileri_4_4	Bound variables: 20
senkron_hızlı_geri_2_2	Bound variables: 8
Senkron_hızlı_geri_2_3	Bound variables: 10
senkron_hızlı_geri_3_3	Bound variables: 13
senkron_hızlı_geri_3_4	Bound variables: 16
senkron_hızlı_geri_4_4	Bound variables: 20

Figure 28. The files prepared for the first project

At the end of the first project, all the works was looking like in Figure 28. Fast forward and reverse tests were performed for 5 different workspace patterns. Moreover, I found out how many seconds does it take the image to appear on the screen after pressing the play button. Inside each of these files are the records I have saved and the codes I have written. As can be seen, the number of bound variables increases as the number of cameras increases.



## 6. Second Project

In the last week of my internship, I was given a new task. The task was to test the Linux application server management interface with Ranorex. I could show my work more easily by sharing this interface, but I cannot share it due to company confidentiality.

Nonetheless, I can say that this is an interface designed to make things easier to manage and understandable to everyone. I worked on a port on a Linux server. My tasks were hostname identification, setting the system clock, SM open and close, NVR open and close, SNMP settings...

In this admin interface, I worked under 3 tabs named overview, NVR management and SNMP respectively.

In the overview tab, I edited the hostname and system time using the recording module feature of Ranorex. To synchronize the system time, I used NTP. In [4], NTP (Network Time Protocol) is a basic application layer protocol that allows computers on a network to synchronize their clocks. For Transport layer services, NTP relies on UDP. It is vital, despite its simplicity. When determining the most effective path for data via a network, time is essential. Time synchronization across a network is also necessary for time-stamped security mechanisms and ensuring accuracy and consistency across numerous storage systems. At the Transport layer, NTP takes advantage of UDP's rapid, connectionless nature. NTP is a time-sensitive protocol that cannot wait for TCP's error checking.

In another tab, NVR management, I perform tests such as opening and closing system manager, restart NVR, stop NVR, restart SM, stop SM with Ranorex. Here, it would be better to explain the terms IP cameras, SM and NVR in more detail.

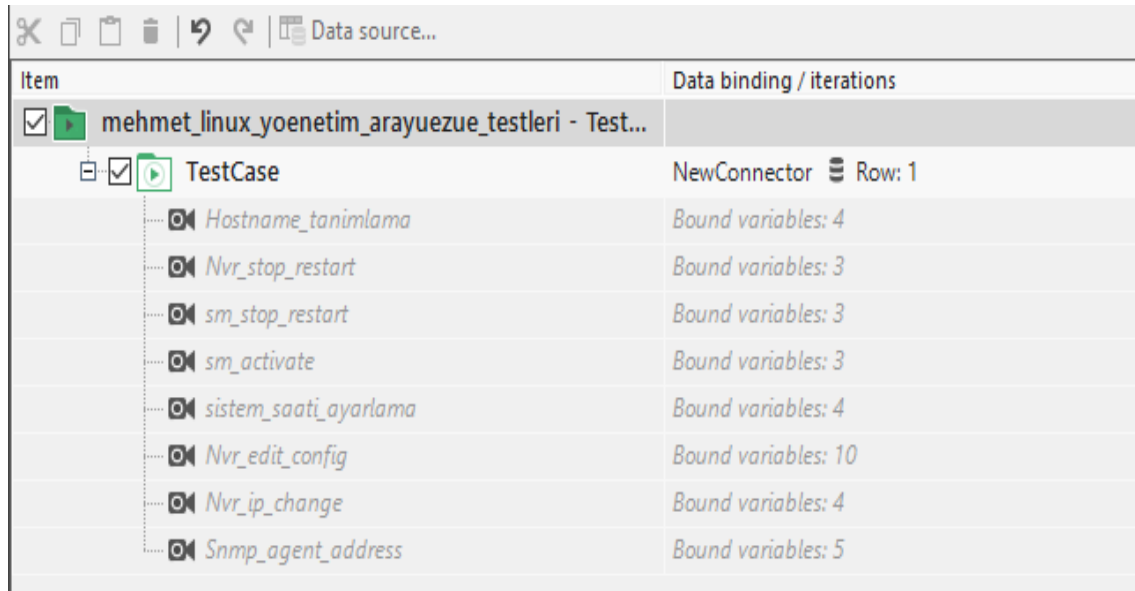
IP cameras perform the same functions as analog cameras, but with far more capabilities. IP cameras provide higher-resolution images as well as more versatile features such as remote zooming and repositioning. They also offer the ability to view images in a web browser option.

NVR stands for network video recording. It is a device designed to record video with internet protocol. Thanks to the software it contains, it makes camera controls easier. IP cameras are connected to NVR via LAN. It supports Windows or Linux environments. It makes it easy for us to register. Network video recorders are similar to digital video recorder, but they are compatible with IP cameras. A network switch or router connects the cameras and NVR. Through a web browser or a mobile app, we may simply retrieve footage stored on an NVR. NVRs do not convert IP camera footage because it is already digital; they merely store it and allow it to be viewed.

On the other hand, SM stands for the system manager. It allows us to manage NVRs. Both NVR and SM have their own databases and ports.

Here are the questions that came to my mind. Why am I doing these on Ranorex? Where is test automation in this work? Then I found out that what I've done is only for one NVR. Later, when there are so many NVRs here, what I do will make much more sense. I have assigned the necessary places as variables so that the maintainability of my work and the people after me can easily continue the work.

Another tab is SNMP. SNMP (Simple network management protocol), as the name suggests, is a simple application layer protocol that helps the network administrator while managing the network. It is mainly designed to facilitate the management and control of devices in large networks. SNMP was built and adopted swiftly at a time when the need for network administration was becoming painfully evident, and it immediately gained universal support. SNMP has become the most extensively used and implemented network management framework today. SNMP is a protocol for exchanging data and orders between a controlling entity and an agent acting on its behalf within a managed network device. More information about SNMP can be found in [3] and [4]. In this tab, I perform test automation by assigning the transport type and address as variables. In TCP, the data is divided into pieces and after each piece is sent, the return of the one it has reached is expected. The receiver can signal the transmitter to slow down. UDP, on the other hand, is a simpler protocol. It does not deal with information such as resending lost data, flow control. It is a faster protocol and is preferred with real-time applications. I have worked on both types in my summer practice.



Item	Data binding / iterations
<input checked="" type="checkbox"/> mehmet_linux_yonetim_arayuezue_testleri - Test...	
<input checked="" type="checkbox"/> TestCase	NewConnector Row: 1
Hostname_tanimlama	Bound variables: 4
Nvr_stop_restart	Bound variables: 3
sm_stop_restart	Bound variables: 3
sm_activate	Bound variables: 3
sistem_saati_ayarlama	Bound variables: 4
Nvr_edit_config	Bound variables: 10
Nvr_ip_change	Bound variables: 4
Snmp_agent_address	Bound variables: 5

Figure 29. The files prepared for the first project

As depicted in Figure 29, there are the files I have prepared and the works I have done. In these files are the recording modules that I made. But since I didn't write any code here, my work did not take much time. I mostly tried to understand network structures and protocols. Since I finished my task early, my supervisor gave me extra work on the second project.

## 6.1. Extra Work on Second Project

When I turn off the SM and NVR in the Linux application server management interface, I had to test whether it really shuts down in the background. I was able to observe the shutdown but he asked me to do it with Ranorex and get a report. I tried to do it using CCTV monitoring and client user interface and Telnet. But in both cases, I ran into difficulties. CCTV monitoring and client user interface did not work when SM is off. Ranorex was not detecting Telnet efficiently.

As described in [4], Telnet is a terminal emulation protocol that uses the TCP/IP protocol suite to connect to remote hosts. A TCP connection is established via Telnet. Telnet is frequently used to connect two systems that are not compatible. Telnet allows you to operate a remote host through LANs and WANs. Network administrators can use Telnet to connect to a router from a computer on their LAN and change the router's configuration. Telnet, on the other hand, is notoriously unsafe, so telnetting to a router over a public network isn't recommended. For this reason, other, more secure techniques of remotely connecting to a host have supplanted Telnet.

My supervisor engineer also asked me to create a “.exe” file using winform application to make this work more efficient and sustainable. This file would have the following function. It will get an IP address from the user and if Telnet is selected, it will also get the port number and display whether that port is open or not. Later, this “.exe” file would be used in Ranorex.

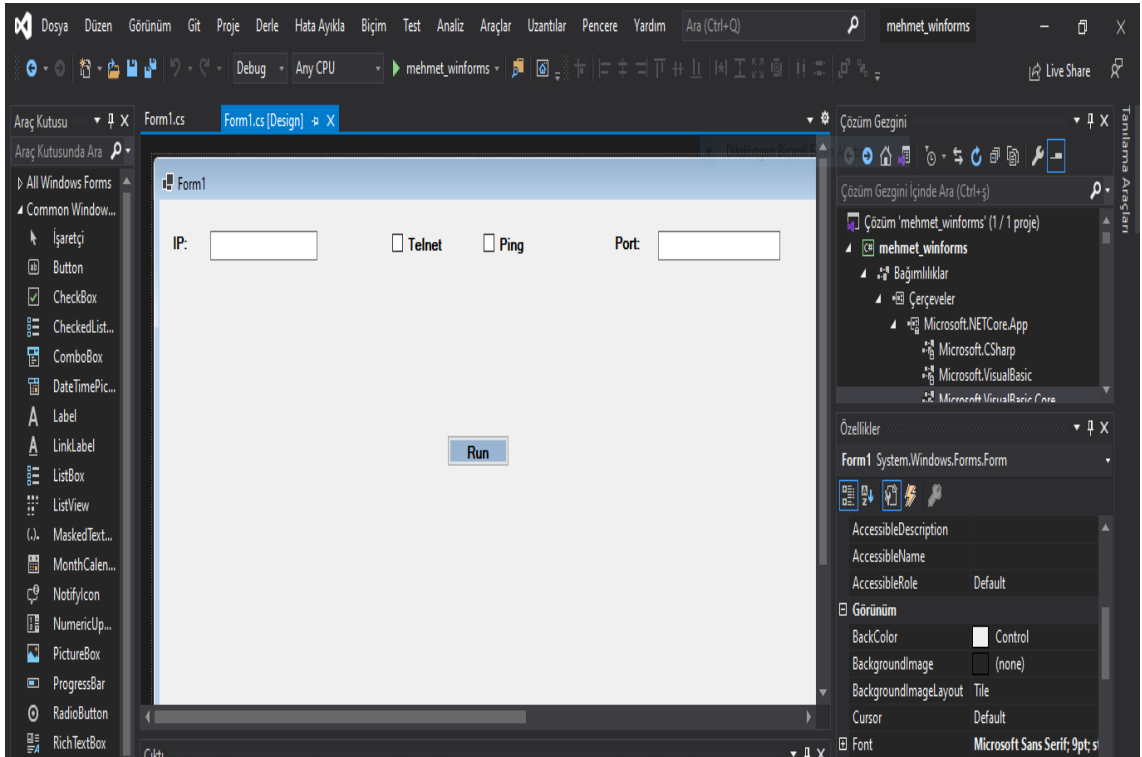


Figure 30. The form prepared using winform application

I created a form using winform application, as shown in the Figure 30. However, I must state that I had no experience in this matter. I did a lot of research before starting the work.

```
1 başvuru
private void checkBox1_CheckedChanged(object sender, EventArgs e)
{
    if(checkBox1.Checked == true)
    {
        textBox2.Enabled = true;
    }
    else
    {
        textBox2.Enabled = false;
    }
}

1 başvuru
private void button1_Click(object sender, EventArgs e)
{
    try
    {
        TcpClient baglanti = new TcpClient();
        baglanti.Connect(textBox1.Text, Convert.ToInt32(textBox2.Text));
        MessageBox.Show("Port Open");
    }
    catch(Exception)
    {
        MessageBox.Show("Port Close");
    }
}
```

Figure 31. The code written for extra work on second project

As seen in the code I wrote in Figure 31, if the Telnet checkbox is selected, the port textbox is enabled. I used System.Net.Sockets. For developers who need careful control over network access, it provides a managed implementation of the Windows Sockets interface. Finally, I display a message on the screen depending on whether that port is open or not.

If the port is open, I display a message like "Port Open".

If the port is close, I display a message like "Port Close".

NVR and SM have their own ports. This is how I check whether these ports are open, or not. Then I automate the test by using this ".exe" file in Ranorex.

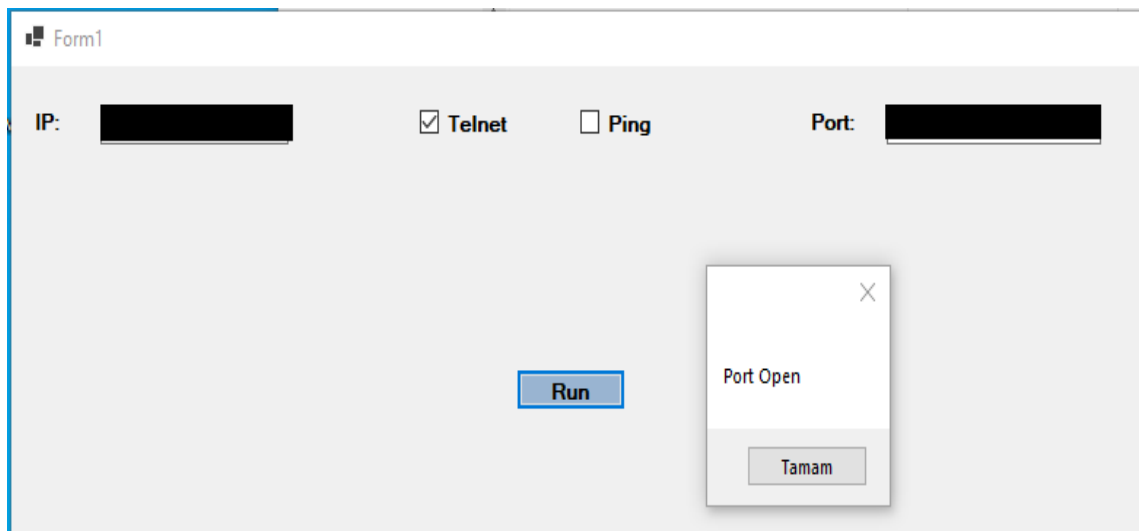


Figure 32. Port Open

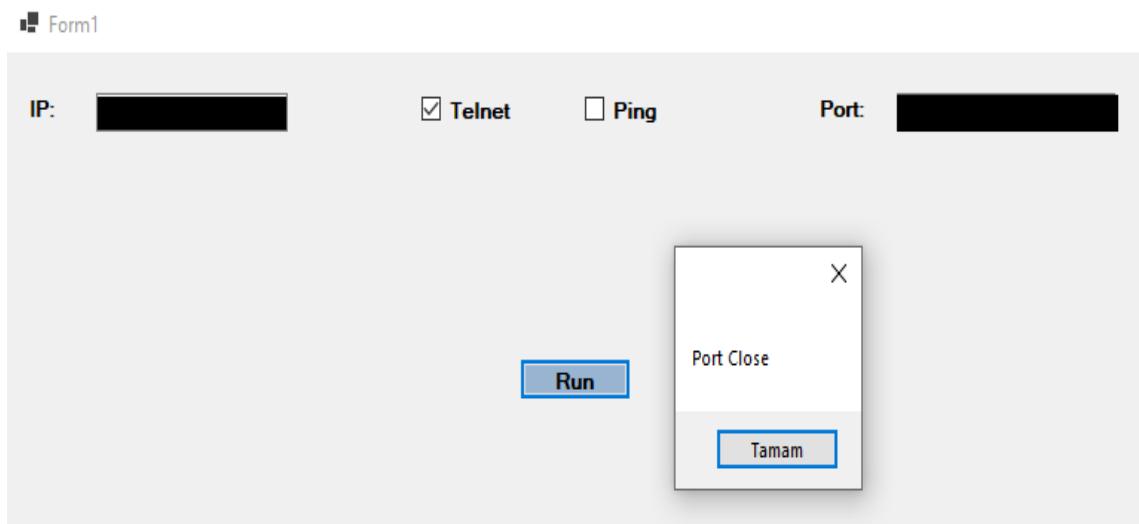


Figure 33. Port Close

When I closed the SM and NVR ports and tested the program, the screen in the Figure 32 was displayed.

When I opened the SM and NVR ports and tested the program, the screen in the Figure 33 was displayed.

As you can see in Figure 32 and Figure 33, the Telnet checkbox was enabled in both cases.

Due to company confidentiality, I'm not allowed to share IP addresses and ports of the NVR and SM that I've tested. Therefore, the black bar has been drawn so that those parts are not visible.

## 7. Conclusion

In conclusion, I experienced and learned many new things about security systems and test engineering during my summer practice. I had the opportunity to observe the projects carried out in the security department. I learned what the duties of a test engineer are, how to collaborate with other disciplines when a test fails, what is test automation, why this technique is used, how we can increase productivity and efficiency by automating test cases. I had the opportunity to get to know programs that I had not used before, such as Ranorex and Winform application. I developed myself in the C# programming language. I have researched and gained experience on concepts such as I-frame, P-frame, TCP, UDP, multicast, unicast, IP cameras, NVR, Bitrate, FPS, Playback Speed, NTP, SNMP, AVC, and HEVC.

What I have done during the summer practice was perform fast forward and reverse tests in synchronous recorded video playback mode in CCTV monitoring and client user interface. I also ran tests of the Linux application server admin interface and created a .exe file using the Windows forms application.

I believe that I made good use of the internship opportunity I had. I didn't have any free time during my internship. I was constantly asked to do new tasks and new research, which was good for me. I think that I got along well with my colleagues and managers and worked in harmony, and I felt valued at work. Every experience and knowledge I have gained during my summer practice will help me when I start my working life. I think that I make them feel that I respect the work of the engineers responsible for me and that I am there to learn from them. The working environment at ASELSAN, Turkey's largest defense electronics company, is very sincere, and everyone supports each other. I would recommend ASELSAN as an internship location to anyone who wants to improve their skills such as problem-solving, analytical thinking, and systematic working with the support and information they receive from senior engineers.

## 8. References

- [1] Andrew S. Tanenbaum, David J. Wetherall, *Computer Networks*, 5<sup>th</sup> ed., Prentice-Hall, 2011.
- [2] B. Furht, O. Marques, *Handbook of Video Databases: Design and Applications*, 1<sup>st</sup> ed., CRC Press, 2003.
- [3] James F. Kurose, Keith W. Ross, *Computer Networking*, 6<sup>th</sup> ed., Addison Wesley, 2014.
- [4] Tamara Dean, *Network+ Guide to Networks*, 5<sup>th</sup> Edition. Course Technology - Cengage Learning, 2010.
- [5] T.-K. Lee, C.-H. Fu, Y.-L. Chan, and W.-C. Siu, "A new motion vector composition algorithm for fast-forward video playback in H.264," *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, 2010.