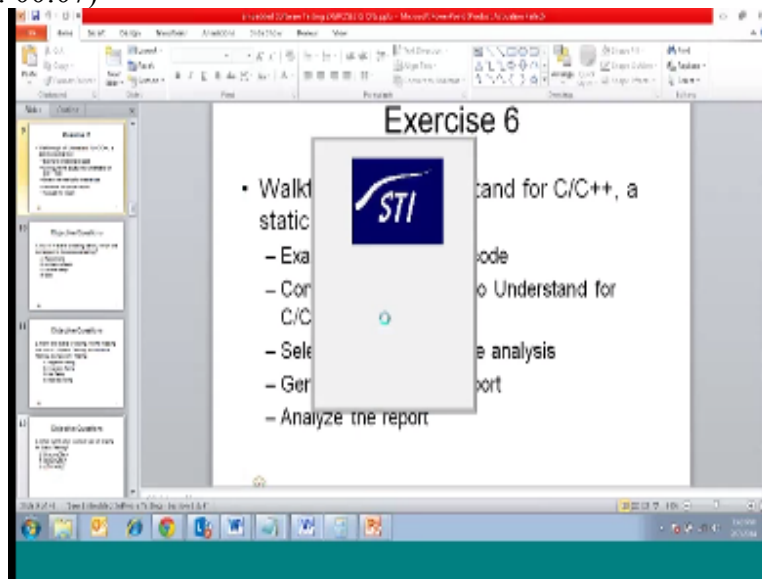
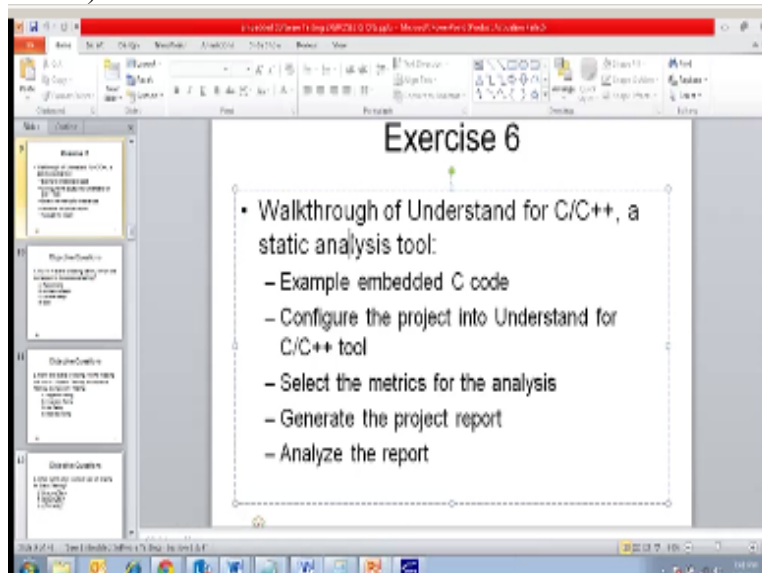


Hi all welcome to the next session of embedded software testing.  
(Refer Slide Time: 00:07)



In practical demonstration today we will study on the other tool called & for C/C++ this is used for statically analysis purpose so we will try to go through sample code how to configure the project and using the understand the process and we will try to select some of the metrics on the analysis the and finally we will read generate the report and how you can see the report, so we can statical analysis example with understand for C/C purpose.

Okay we will try to evoke the tool so the tool be mist rolled something like this understand and.  
(Refer Slide Time: 01:07)



This tool is statical analysis from tool verse starting so you can also get a evaluation version of this practicing and this can be downloadd okay, we know that what are the things that we do in

a static analysis like we know the difference between dynamic and the static analysis what we do is we will do a testing of a embedded software when the software is arounding of the target. Where as in the static testing static analysis testing we do not run the embers of the testing interest basically what we do is offline or statically we try to run the various source are the various source aspects of the embedded software so one of the important that we need to be drawing about the static analysis that.  
(Refer Slide Time: 02:28)

**Exercise 6**

- Walkthrough of Understand for C/C++, a static analysis tool:
  - Example embedded C code
  - Configure the project into Understand for C/C++ tool
  - Select the metrics for the analysis
  - Generate the project report
  - Analyze the report

static analysis need to target execution need inputs based on the test strategy which requires manual inputs - source code, test scenario requirements...

control coupling, data coupling

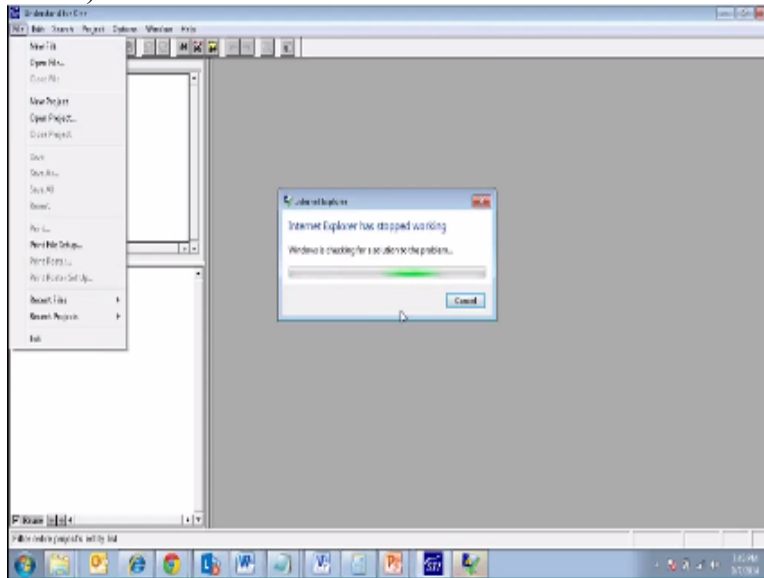
We know that, so no need of target exclusion need the inputs whereas on the test strategy which basically requires manual input in the sense basically inputs could be source code for such cases officeonar we need then we need a dollar design document or requirement document website and the stagergy what is that test scenario what we have told so wherever we need that stactical analysis the type of testing that we required for example.

This is have control coupling, data coupling type so this is one of the static analysis aspects of a verification methodology we will see the flow graph of the embedded software with the help of the control coupling were the focus try getting generated, whose is calling home and whether red code is a etc with that I have help of the control flow, we do the control coupling so that we verifies with that the other ways.

Interface between various variables between additional functions we need use the metric called data coupling okay, so we have understand foe C++ it is from STI tools we can get more information about the STI page is called SA tech or SA tools which are common it is try to look into the website so at more information and get SA tools .com

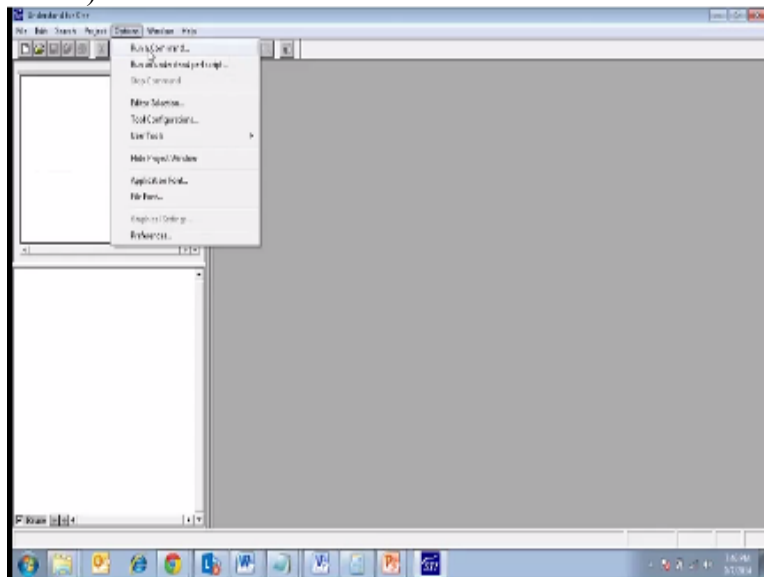
Okay this is connecting before we understand here we have a couple of window this is standard windows we know items.

(Refer Slide Time: 05:28)



Selecting the new file or creating a new file creating a project understands the discusses so understand C++ what is basically it is a project that of the source software source we configure and it will have such we that it will try to understand the source and generate the report.

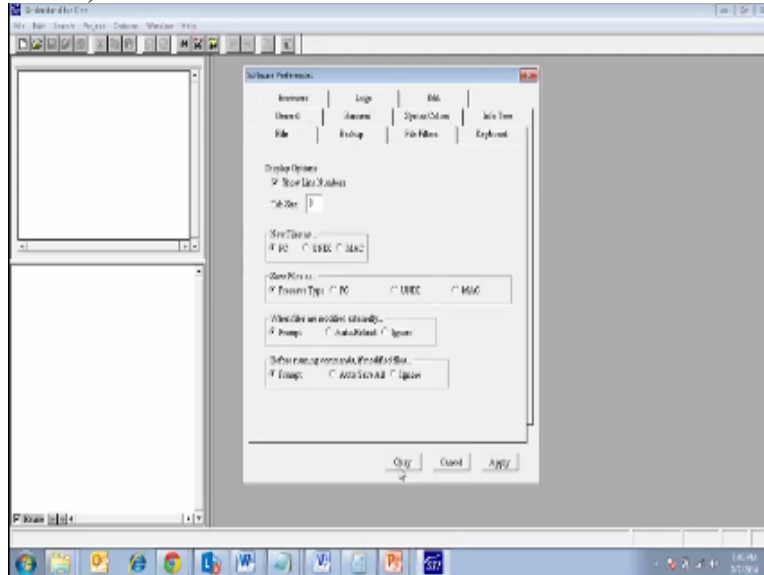
(Refer Slide Time: 06:01)



Similarly we have edit search project, project has conquering the various a type of matrix we want here and we had more files analyze and change the files on the and check for any updates of the files source files we can do this we can generate the reports with this and matrix summary the entire project can file this is expert can we export to a such as c files and HTML complete report is available in appropriate with this.

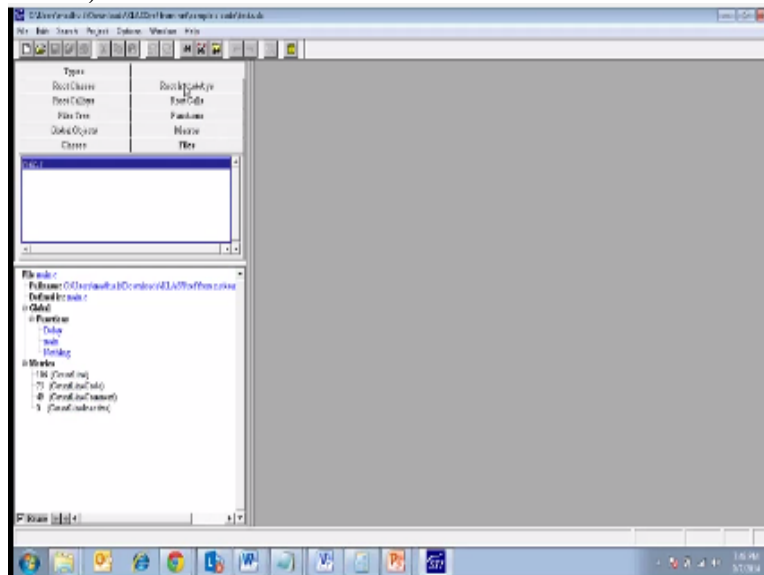
We can run through command also and we can add pulse script and the additional thing and any user tools that configuration and all that it is a part of the HTML it is a software configuration can use this option.

(Refer Slide Time: 07:02)



Of course this standard option are available and have what file what we had that we are going to use will go through this and we select the few sample okay.

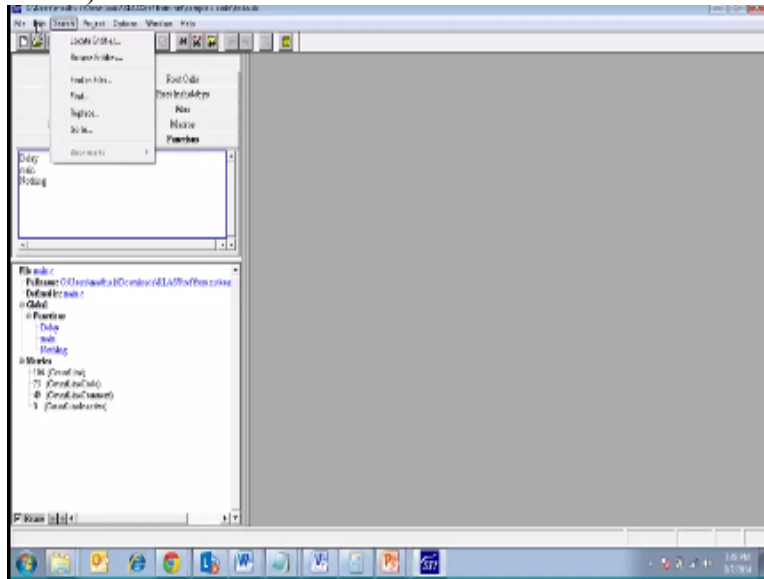
(Refer Slide Time: 07:15)



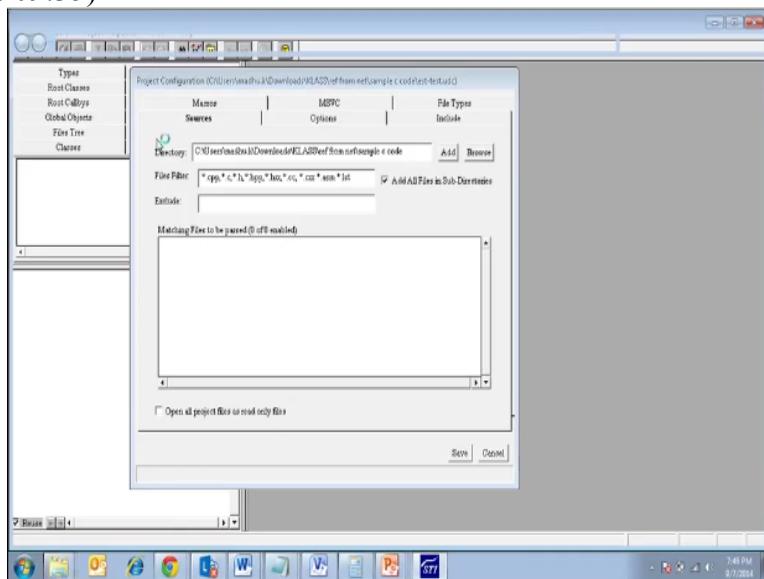
So try to open up, file for test name of the project uses do we see, and this is sample for create called main. C and as soon as the, it is a array to the project of this so on it stands it sequence plus and declaring all the details and I required in these thing fine that objects are so based on the then there we see several hosts in the content will generate the various items such as with define the path were is define the global.



That are we used the functions global functions there is a delay functions is a main function and nothing added different number of sequencing and total mater of this file shown we can see count line 1 count line 4 comment prompt in active on executable, executable line and a commenter line and etc all these lines based on the matrix you chosen this is going to show. (Refer Slide Time: 08:40)

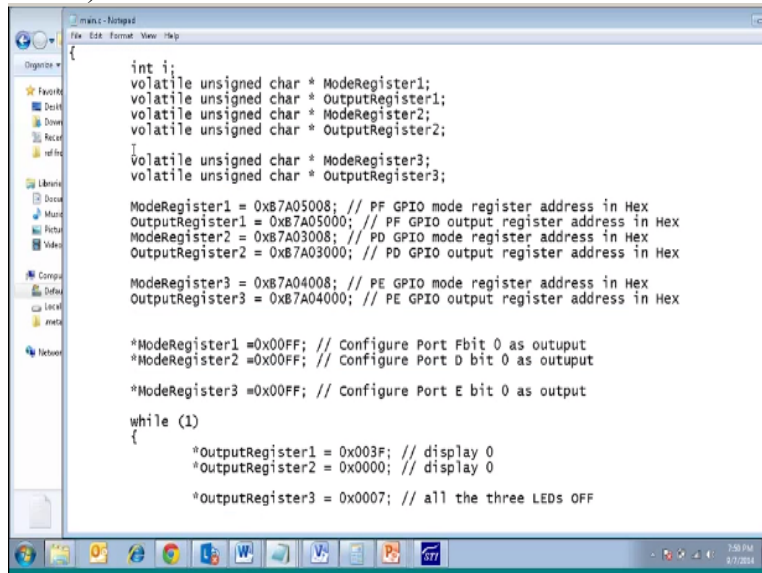


Similarly we have different type of option since here with the C++ process global objects whose is calling all is options as we saying so we on choosing this project they have selected this file probably we trying to reconfigured, what we do is I will, so we have this path we will to create it impossible, so it go to current project so current project it will close it creates a list of test we have a source, sample source for this class, same pass way was to generate. (Refer Slide Time: 09:35)



Okay now as soon as your created the project which is showing, the project configuration how this project is static analyzes understand for six phrases tool can be configured, in this source we have to tell that, so they should prick like the source sample C code we have, to see that , so in the C file created for example.

(Refer Slide Time: 10:21)



```
main.c - Notepad++
File Edit Format View Help
{
    int i;
    volatile unsigned char * ModeRegister1;
    volatile unsigned char * OutputRegister1;
    volatile unsigned char * ModeRegister2;
    volatile unsigned char * OutputRegister2;
    I
    volatile unsigned char * ModeRegister3;
    volatile unsigned char * OutputRegister3;

    ModeRegister1 = 0xB7A05008; // PF GPIO mode register address in Hex
    OutputRegister1 = 0xB7A05000; // PF GPIO output register address in Hex
    ModeRegister2 = 0xB7A03008; // PD GPIO mode register address in Hex
    OutputRegister2 = 0xB7A03000; // PD GPIO output register address in Hex

    ModeRegister3 = 0xB7A04008; // PE GPIO mode register address in Hex
    OutputRegister3 = 0xB7A04000; // PE GPIO output register address in Hex

    *ModeRegister1 = 0x00FF; // Configure Port F bit 0 as output
    *ModeRegister2 = 0x00FF; // Configure Port D bit 0 as output

    *ModeRegister3 = 0x00FF; // Configure Port E bit 0 as output

    while (1)
    {
        *OutputRegister1 = 0x003F; // display 0
        *OutputRegister2 = 0x0000; // display 0

        *OutputRegister3 = 0x0007; // all the three LEDs OFF
    }
}
```

If some software are simple example, which has a main, which has some declaration such as into I that I volatile some registers basically some tangle GPIO tangle project which uses symptoms of outputting value, in a segment seven segment attached of the display you should see the three of display zero, initially the basically to start with you know any embed software will while one blow, this is any live of the software.

And you can see there are different functionalities of this, so let us not bother about what it does, so different sort of a live, so structure we have a extra function call.

(Refer Slide Time: 11:14)

```

main()
{
    *OutputRegister1 = 0x003F; // display 8
    *OutputRegister2 = 0x0008; // display 8
    *OutputRegister3 = 0x0000; //green, Yellow, & Red LEDs ON

    for (i=0; i<1000000; i++); //Delay

    *OutputRegister1 = 0x0027; // dis9
    *OutputRegister2 = 0x0008; // dis9
    *OutputRegister3 = 0x0000; //green, Yellow, & Red LEDs ON

    for (i=0; i<1000000; i++); //Delay

}
return 0;
}
delay()
{
    for (i=0; i<1000000; i++)
    {
        Nothing(); //delay
    }
    return 0;
}
Nothing()
{
    return 0;
}

```

So other mention set etcetera, so there are derive function what it does is, it will display zero, one, two, three continuously, so that is the function within the while one go, it is continuously going, it is going to call different function which is delay you have to say for I is equal to for zero to some one lakh is going to run the processor clock is honors that much delay it goes to happen typically they use this, this is calculated often this function is called nothing, nothing just as a nothing it can return.

Or the term will have zero instead it is not doing anything okay, so we will save this having a (Refer Slide Time: 12:22)

```

main()
{
    *OutputRegister1 = 0x000F; // display 3
    *OutputRegister2 = 0x0008; // display 3
    *OutputRegister3 = 0x0003; //green LED ON

    for (i=0; i<1000000; i++); //Delay

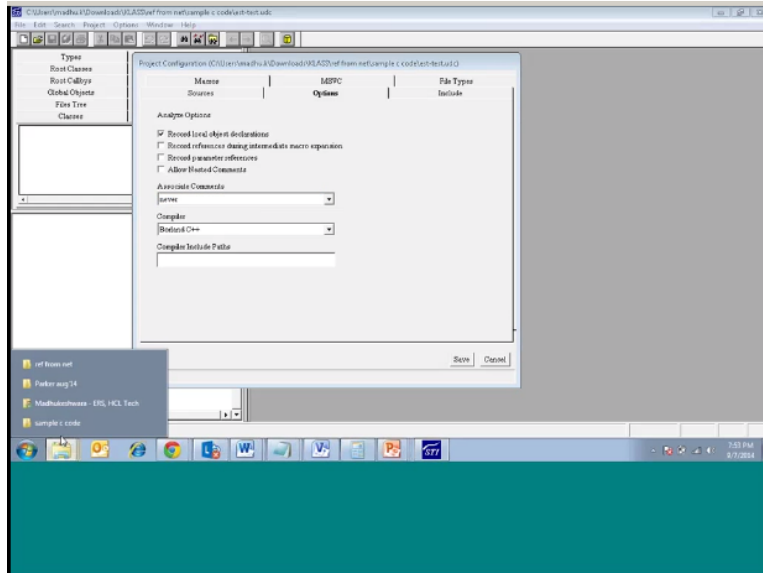
    *OutputRegister1 = 0x0026; // display 4
    *OutputRegister2 = 0x0008; // display 4

}
return 0;
}

```

Two sum function one main function main one, simplest main void C, we will try to configure in our static.

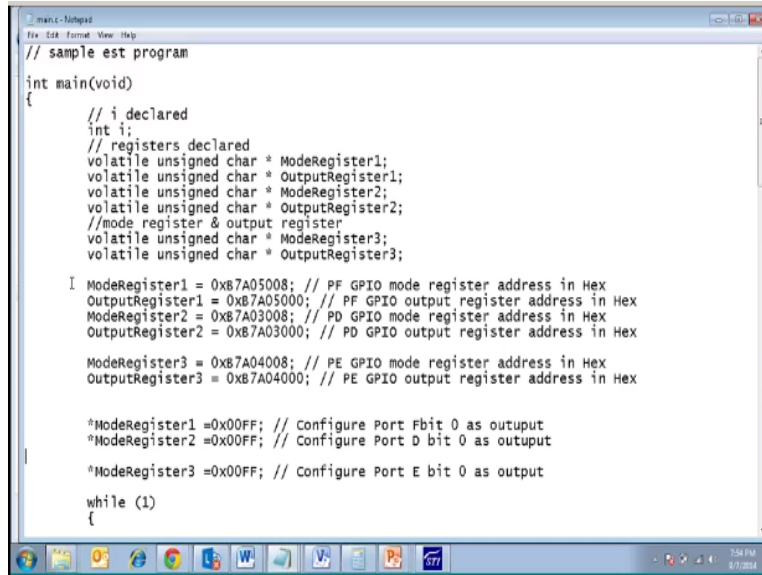
(Refer Slide Time: 12:31)



So file filter, what are the types of five filter select it star dot CPP have but nothing wrong with that select assembly five, some compiler will provide star dot list file C++ based extension CC files another files H files, C files, add with this, so regular we do with the sources so next we have options, so based on versa analyses premade, we can select this options, very local object recursion we want, parameter reference any parameter you have in this case you do not need, any nested command of pair we can use it.

Giving a information special command are not standard with C++, usually it is better to avoid this, and every nation command support can produce ms office, he is asking not to enabalate, so better not use the associate command, associate command so command increases the rate source code right, so let us see whether we have a writable command, the straight of some commands, testing program.

(Refer Slide Time: 14:27)

A screenshot of a Notepad++ window titled "main - Iddipad". The window contains C code for a sample program. The code starts with a comment "// sample est program" and a main function. Inside the main function, it declares an integer 'i' and three volatile unsigned char pointers: ModeRegister1, OutputRegister1, ModeRegister2, OutputRegister2, ModeRegister3, and OutputRegister3. It then assigns hexadecimal addresses to these registers: ModeRegister1 = 0x87A05008, OutputRegister1 = 0x87A05000, ModeRegister2 = 0x87A03008, OutputRegister2 = 0x87A03000, ModeRegister3 = 0x87A04008, and OutputRegister3 = 0x87A04000. Finally, it configures the output bits for each register with the value 0x00FF and enters a while loop with a condition of 1.

```
// sample est program
int main(void)
{
    // i declared
    int i;
    // registers declared
    volatile unsigned char * ModeRegister1;
    volatile unsigned char * OutputRegister1;
    volatile unsigned char * ModeRegister2;
    volatile unsigned char * OutputRegister2;
    //mode register & output register
    volatile unsigned char * ModeRegister3;
    volatile unsigned char * OutputRegister3;

    I ModeRegister1 = 0x87A05008; // PF GPIO mode register address in Hex
    OutputRegister1 = 0x87A05000; // PF GPIO output register address in Hex
    ModeRegister2 = 0x87A03008; // PD GPIO mode register address in Hex
    OutputRegister2 = 0x87A03000; // PD GPIO output register address in Hex

    ModeRegister3 = 0x87A04008; // PE GPIO mode register address in Hex
    OutputRegister3 = 0x87A04000; // PE GPIO output register address in Hex

    *ModeRegister1 =0x00FF; // Configure Port Fbit 0 as output
    *ModeRegister2 =0x00FF; // Configure Port D bit 0 as output

    *ModeRegister3 =0x00FF; // Configure Port E bit 0 as output

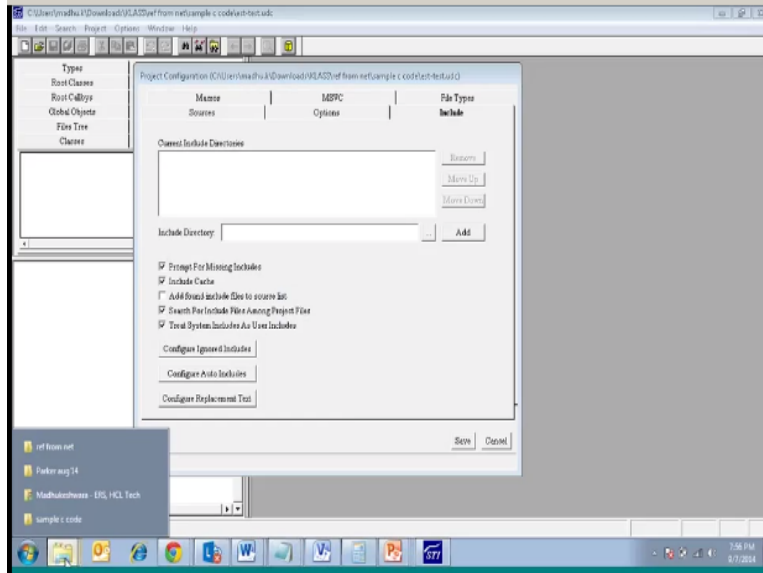
    while (1)
    {
```

I declaration similarly registers declare, mode register and output register, of course we can see this command is side to this but it is advice not to have this here ready to have a worry, first line the timing will keep it here any way the project will understand here, the commands which are above or below likewise, we have command as well for each of the line, so that associating the commands so we are saying that command before definition, after definition longest command before or after definition.

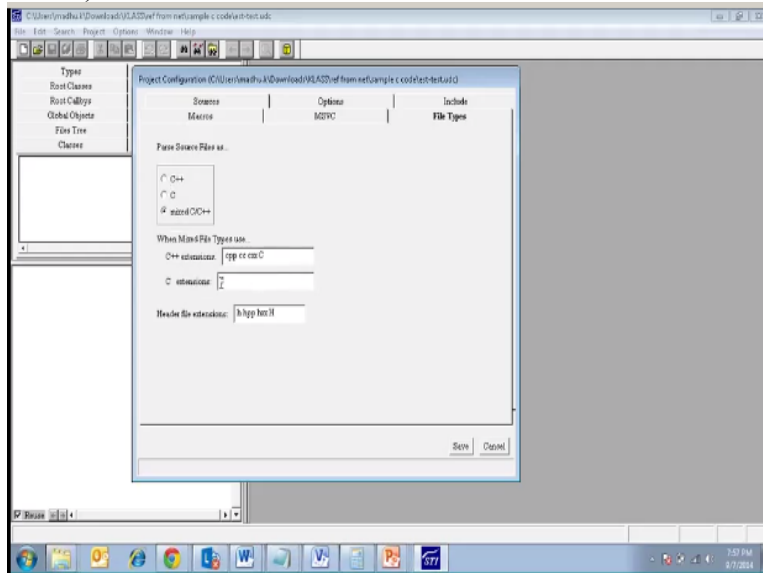
So you just say command before definition of the variables of the usage of the below is a compiler, basically it does not do the compile but what should the compiler we are using, why because it is going to generate the report for a type of C language and what sort of a compiler that we use for a base of error, this needed times of understanding what we are trying to generate it, so we can use anything for example.

R for alpha, HBX, IAR, KLC likewise we have let us choose whatever we need, when the compiler separately user first call the specific you want to negative we can add the path, that is the option here okay.

(Refer Slide Time: 16:26)



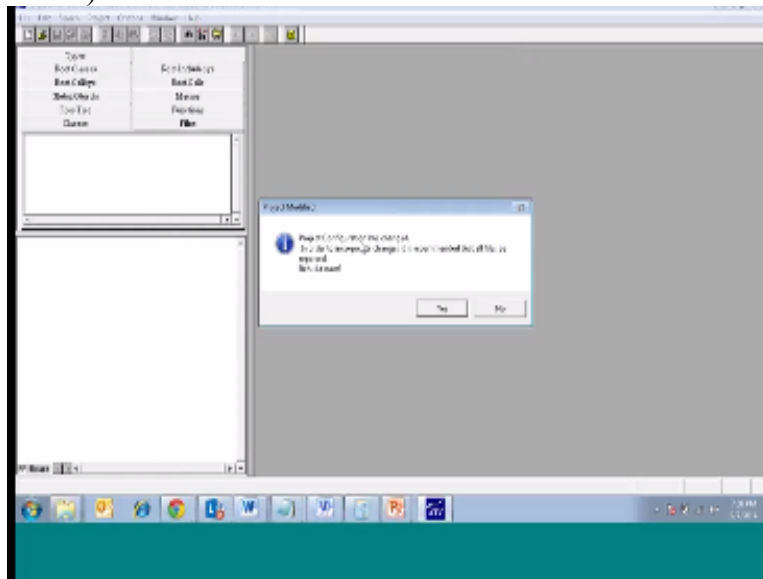
Next one is a include you know that include is required for profile any process specific kind of file you want to add this, you can add with this, in this case we do not have any other file, spot time we will live it off, so sometime what will happen is there are sparer ID like STD IO .H which are the part of the compiler in build, in library we are not able to show the path, in that case what we can do is we can ignore that particular profiles It does not take care of it. (Refer Slide Time: 17:11)



The next one is the macros any macros that you want to use it for the project you can name that and define where it is, that you can add it, next one is a MSVC so it is for the VC process read only that file types, what are the types of file that are used including another file in the project is what we have to tell, we have C++ type, C type, mix C, C++ type ,so first C is used next C++ I will choose C because we have on the dot C file.

And once we used the C++ or C you can select CPP extensions, so C++ extension C extension it uses C,  
Otherwise default uses C header file, what are the header file sometime h will be use in general we can mentioned. So all the things we are done so we have going to add now the particular c code and you can see part of the and it shows with that cross mark. So the main is a show is here and after all this option is been done. So I am going to show this so the project contribution has changed.

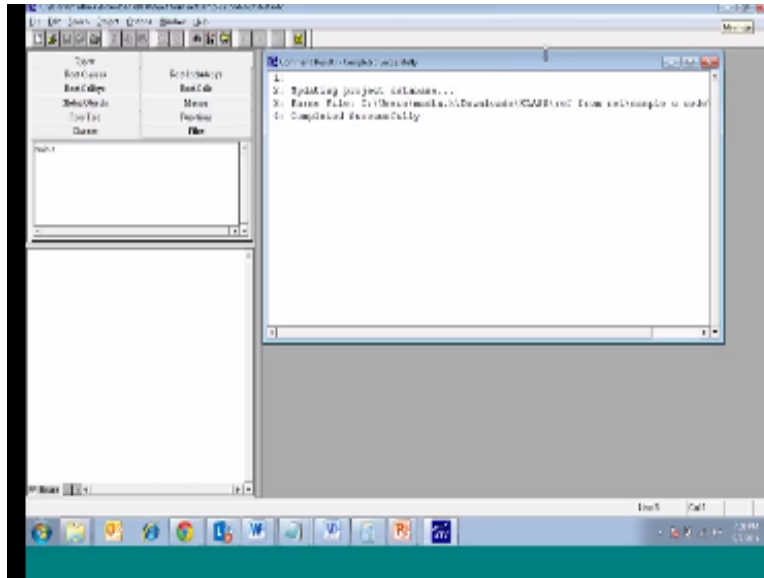
(Refer Slide Time: 18:29)



In not incorporate change it recommend all file reposed so rebuild round that means what are the options we have now the project will rebuild with this options I will say yes. You can see the project is added and the new project with the database is created here, completed successfully we can see there is something related of created all the test EST test UDC and similarly generate the process it will show of all the specific files.

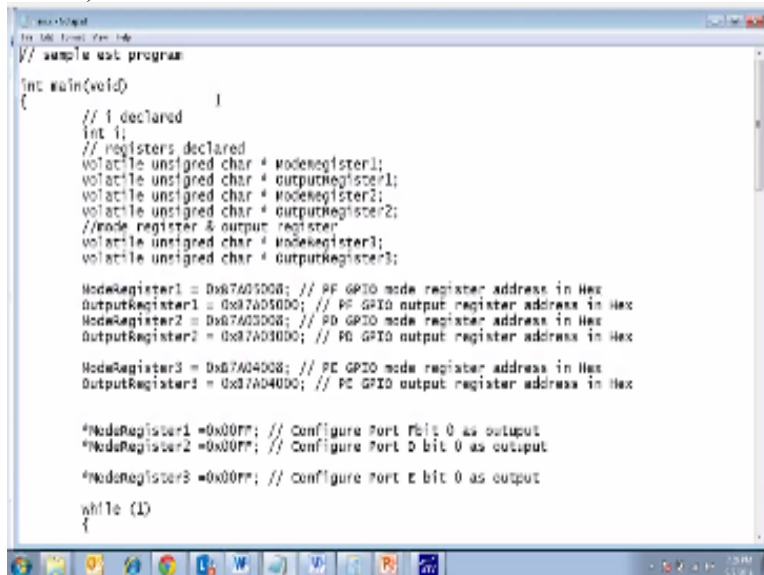
Under this folder not this folder because is for created TSTP test holder under that all the specific report files on the stole okay.

(Refer Slide Time: 19:22)



Now the first job we have done why doing the project creation and adding all the files. Now let us try to concavity process it which we have done. These try to change some file change analyzed changed files. So it will do if any changes are winner it will be analyze and if any changes for require to analyses with the congregation. It can do it analyst all files so it check for object files. All one source file unchanged we can see if this file.

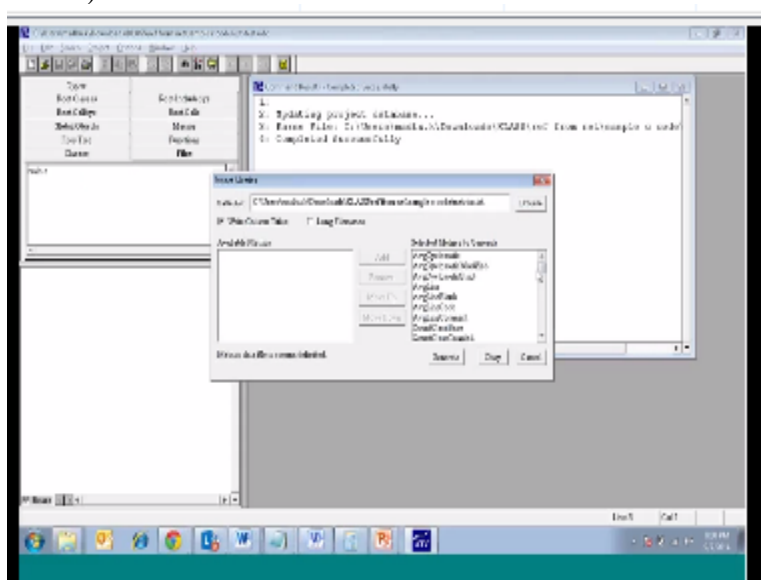
(Refer Slide Time: 20:20)



Just put a space and closed it will be analyst because we have change check for update the files so that showing one of unfiled for the file has got changed that again we need to analyst all the files. Multiple file to be analyzing now you can create the report the source operation H a test report we can do. Now it is not enable Y because we array to generate the report once you



generate the report we can leave it text. Other thing is matrix export is all important option what we do is what should of matrix that we need is all event have okay.  
(Refer Slide Time: 21:23)



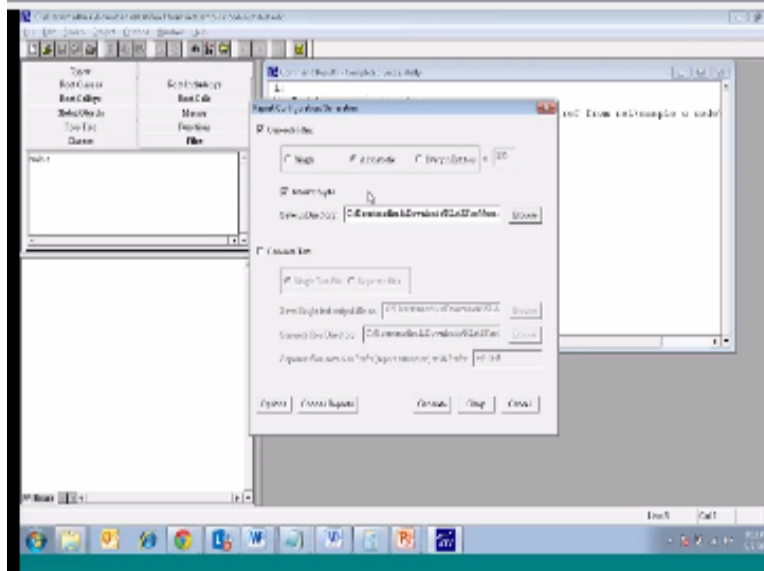
So cyclamate matrix one of the you know that adjust the complicity of the programme it is so those part which array interest will add it. So there is cyclamates are basically selected matrix to generate already added many things. So what we do is we select all and remove it and we select again whatever is. We need a cyclamatic, we need lines of code, and underlines of comments are there declaration calls function classes.

It C+ has specific like that methods are matrix font line may be you can take out this you can add the ever will take these are that will count line, count line how many blank lines are there code declared how much is there count line how much is there count line code is already they are related statement cyclamen. Statement executable cyclamen for average cyclamen take it out we just have cyclamate function.

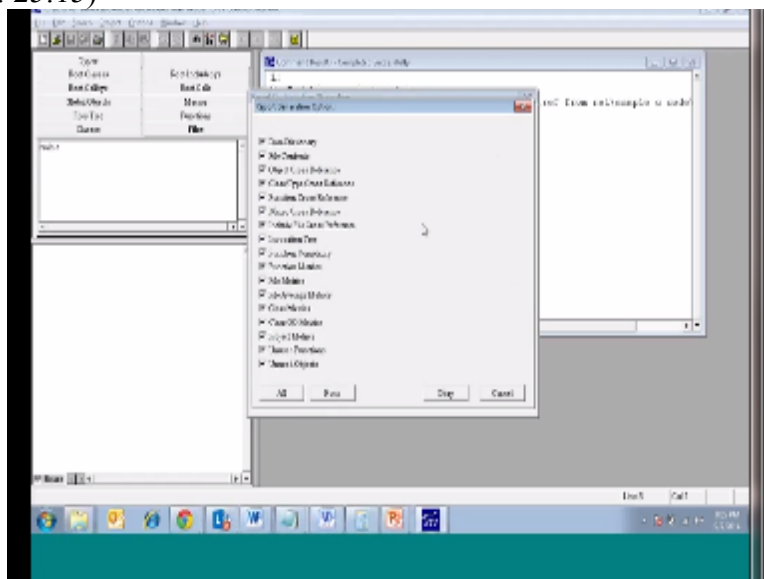
We can also select exam cyclamatic on the established function this file which are selected we can have the different types of cyclamatical results puts modified excreta. That we want we can do rest command to code hoe much present usually you respected to the 100% let us see how much in is we add that. Now once we have all this we can select other options like where I want to generate. The cyclamen matrix brought text.

So we will insert something like matrix EST text dot text then we will say okay it will be say it will not generate that we generate it and okay for generating the project.

(Refer Slide Time: 24:05)

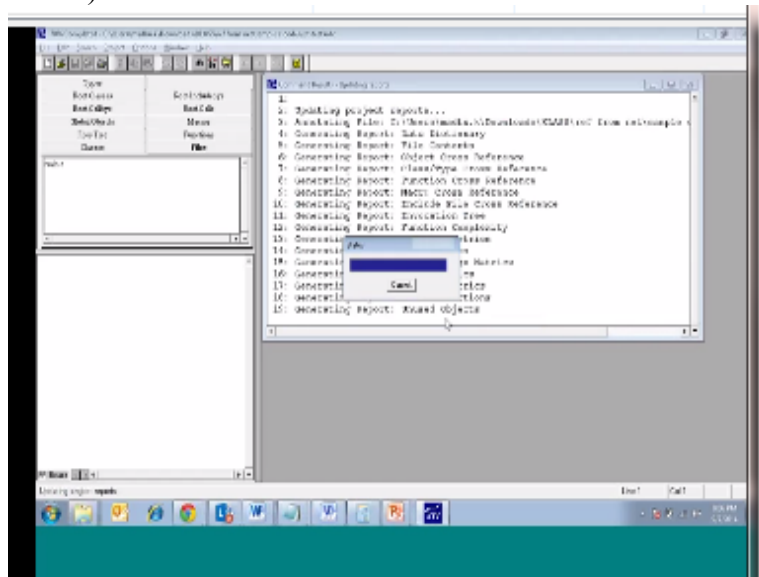


Report so there are several options, so though options are something like want to generate that text type of an output or you want to general STML or both are up to how you want to generate. Let us taking generate board and see and next time also you can have a single fundamental HTML it can be automatic in every end at this evaluated general like number of dish MLs that puts for alphabetic so that H repeatable and that will be good. Allow scripts, you want to allow scripts you can and the links and all be repeated similarly we take to generate text also. In the part is equals to generate ESTT dot text. And there Randal options like general time in report like times stamp becoming have it what time this project complicity of the project analysis report have been generated with the custom, custom will request all are be done.  
 (Refer Slide Time: 25:15)



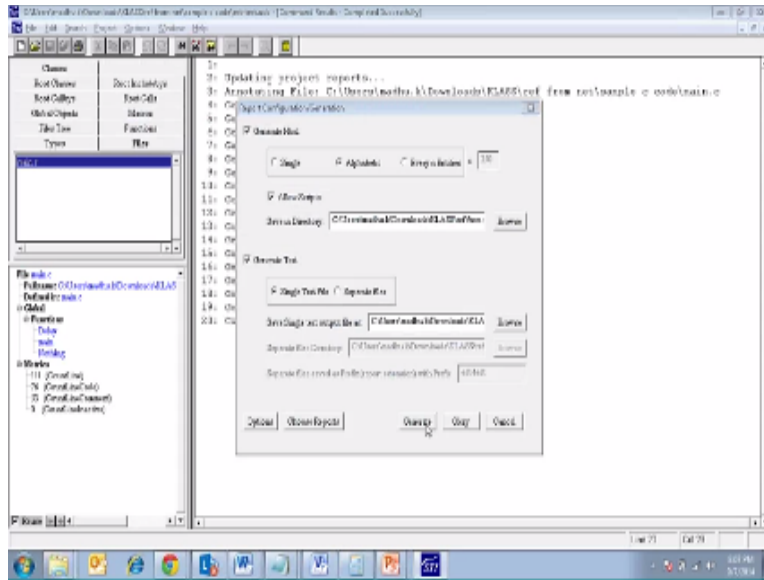
So the reports the kind of report has at we want we can have it we can fixes the multiple file projects put to have file matrix clause O matrix which has do not need UNKNOWN: it is objects project matrix unused function red code we can select all this and we can say okay like there is a file contain data dictionary base on the objects and data will available in the Toole this hash. Marco reference is anything is there function process plain includes file entry that is very important.

For the control floor, and control supply function complicity all this will be generated will say okay. So now selecting all that like part and everything we can have generated pressed. So I will take generate now, so there is directory called this one, this one is created plus add generate the file. Generate has a simple one file.  
(Refer Slide Time: 26:24)



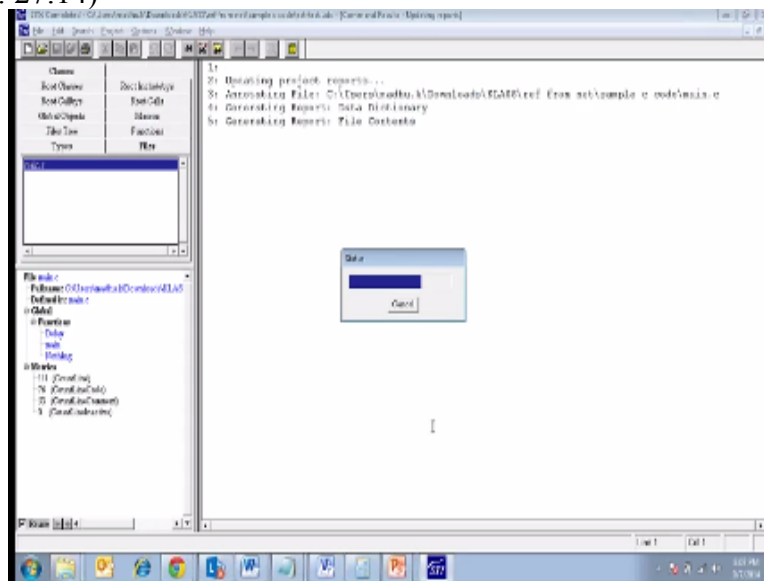
We can see here in this particular generated, generating report and all the locate entry and left okay. Let will say after some other thing like we can see we can select what are files are available here. And this file you can see the main delay function are there main thing like this 3 are there you can see count line is 11 excreta. So clash states C the report now we can see this able to the compiler and generate the report .Now we can see because we are configuring on generated the requirement.

(Refer Slide Time: 27:06)



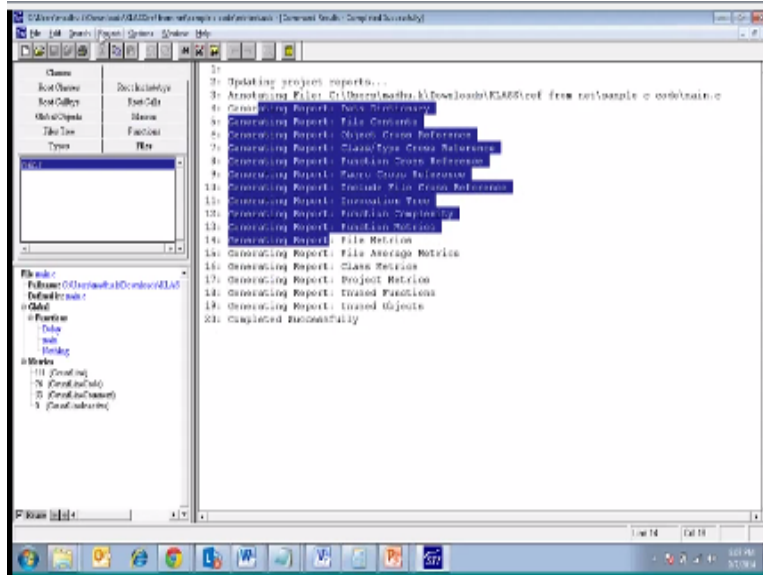
We can generate again and again how many times you want you can do it is going to generate the report.

(Refer Slide Time: 27:14)

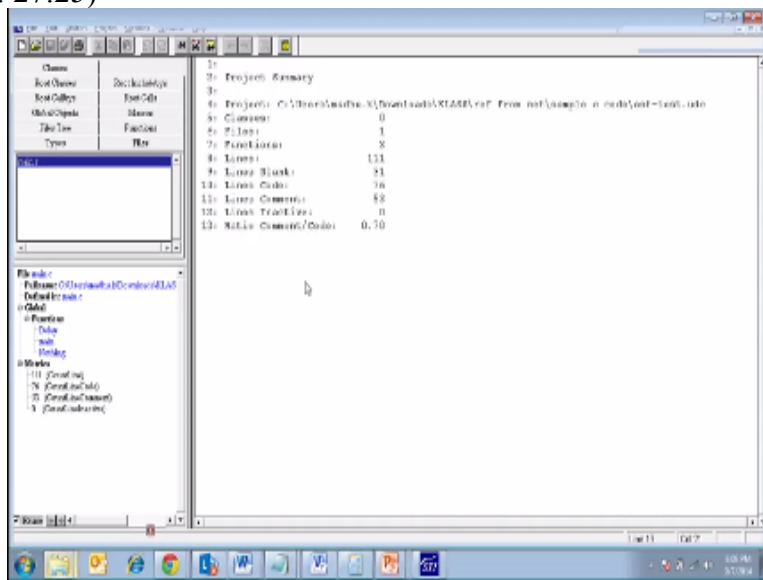


And we can expect the matrix that we have done.

(Refer Slide Time: 27:18)

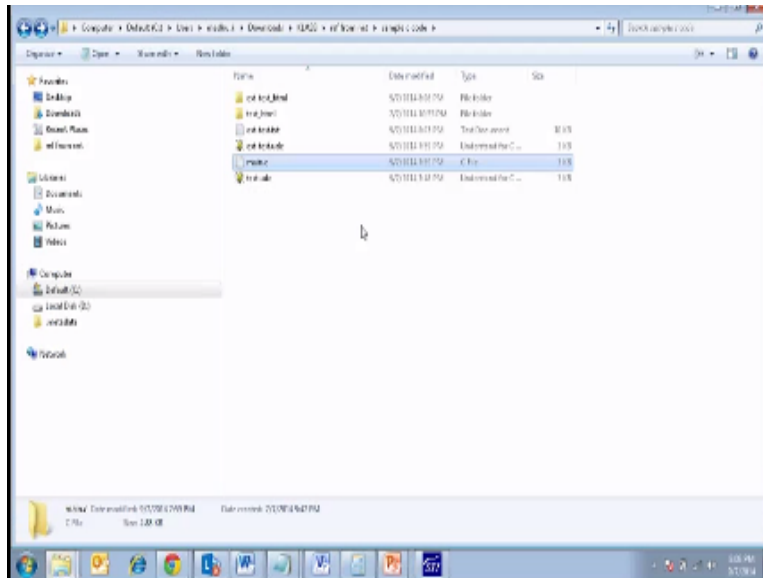


We can see the summary of this project can see.  
(Refer Slide Time: 27:25)

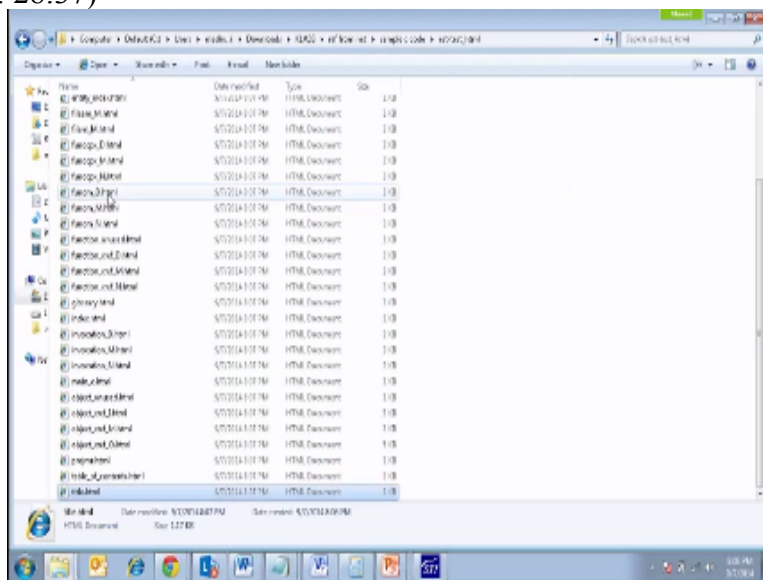


Were the project is report is available how many classes there is no class 0 of how many files one there are three functions total number of lines is 111 there are 31 gland lines there are 76 line numbers a code is the 53 lines in the comment is the in active means it not used right this is not here if you ratio comment was code is .70 is 70% so it is expected that there every line of codes that it is 1.0 or 100% okay.

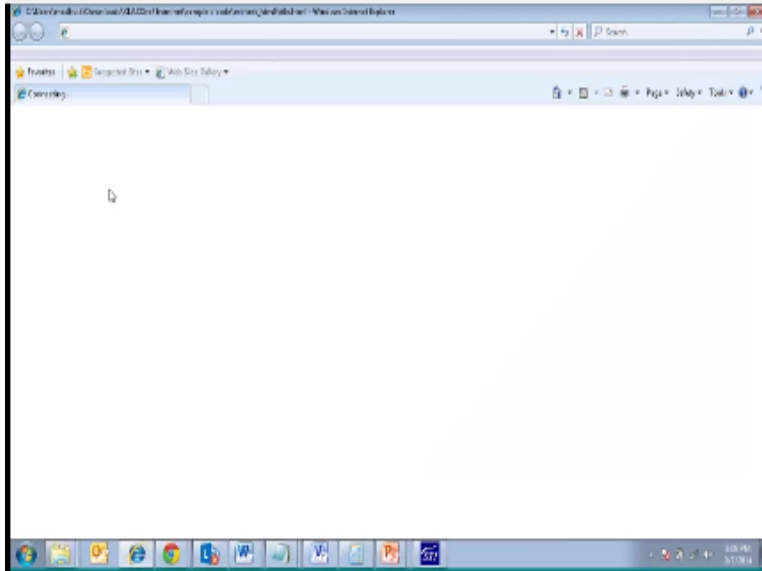
Now let us see the 0 either we can see from directly from this folder let the folder, that is created this folder the reports as we see this, different HTML files.  
(Refer Slide Time: 28:27)



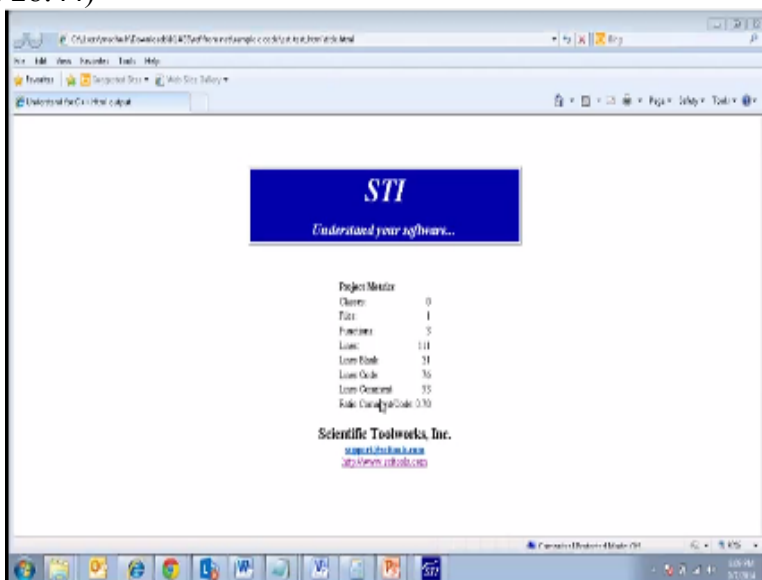
(Refer Slide Time: 28:37)



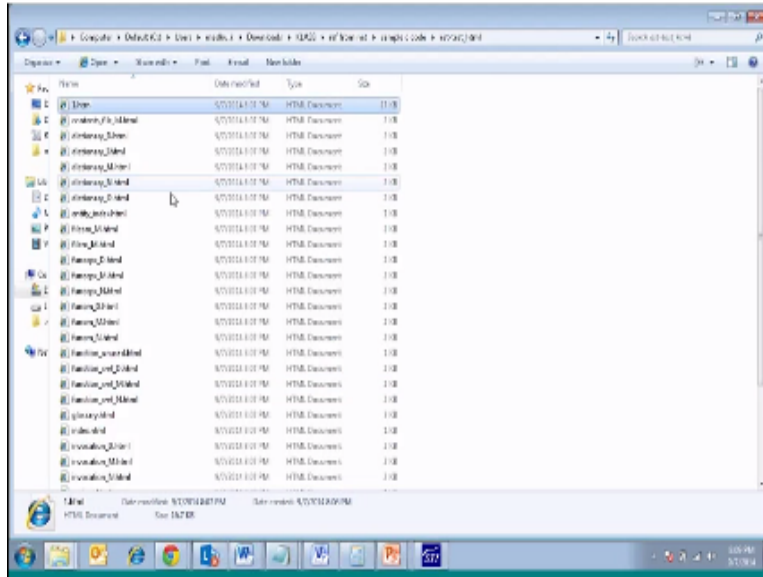
Let us see index or title has try to open this, okay.  
 (Refer Slide Time: 28:41)



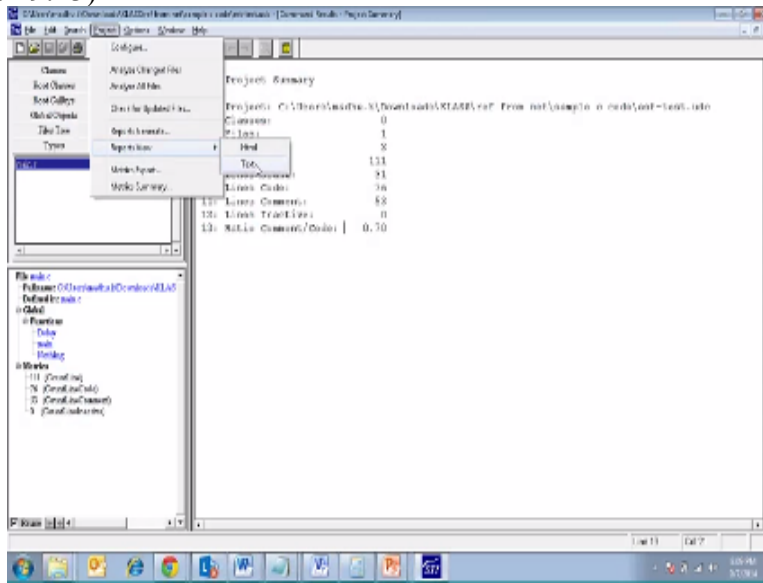
So we can see.  
(Refer Slide Time: 28:44)



Which very good that we reported a nice way in HTML whatever are has to this can be reported and center the customer or, as justification saying that is use what the coding id there like wise, same thing.  
(Refer Slide Time: 29:06)

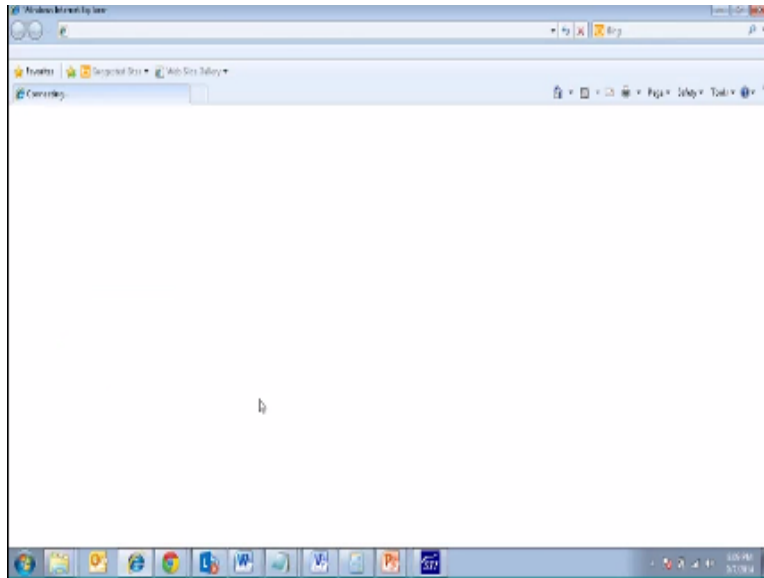


Can we see all this report whatever has generated that to also?  
(Refer Slide Time: 29:13)

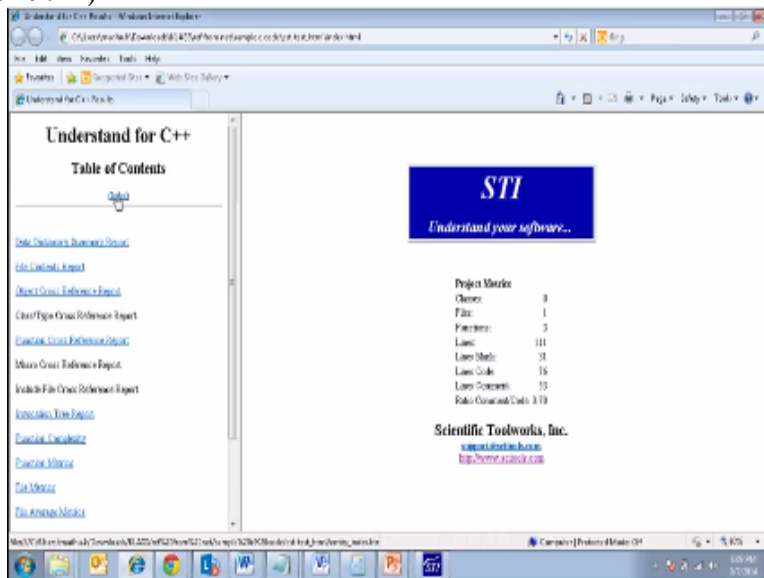


With this option report here HTML.  
(Refer Slide Time: 29:22)

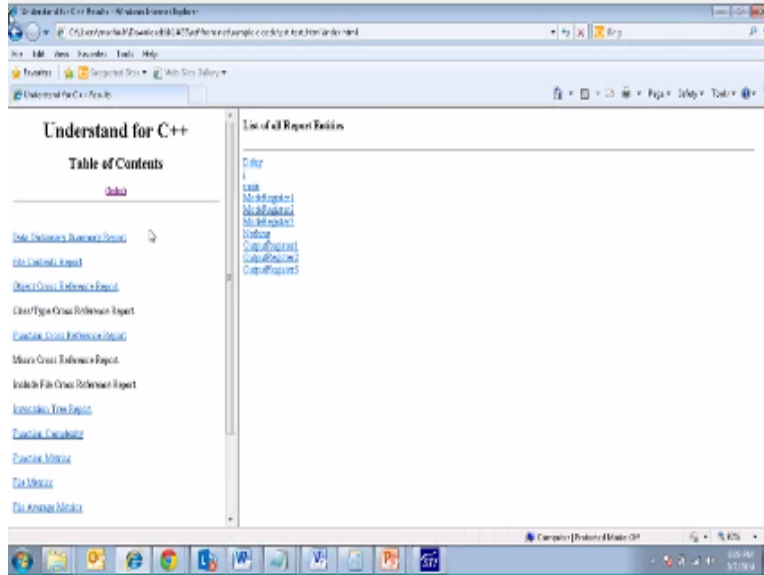




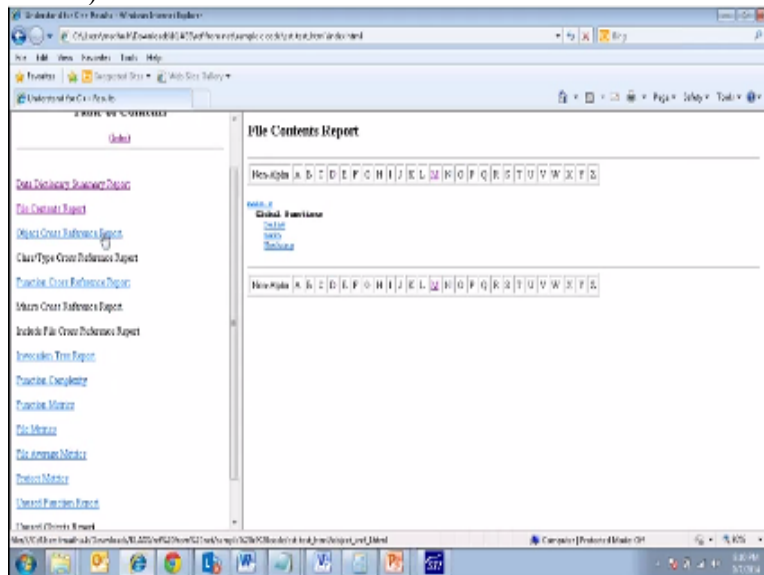
If you they complete the project report it is option.  
 (Refer Slide Time: 29:24)



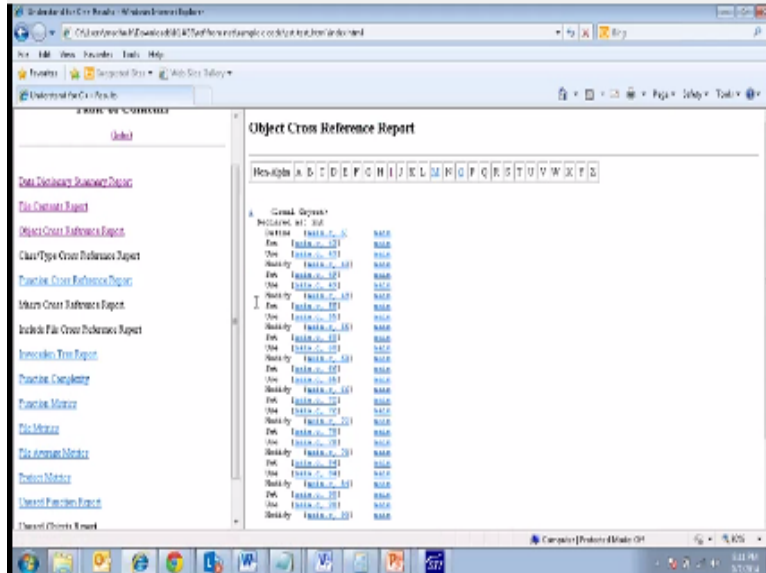
We can see the right hand side the option is available, the final report that ready we can see we was table of content a documentation it does we can collect the index what we want to see we can plug the report,  
 (Refer Slide Time: 29:46)



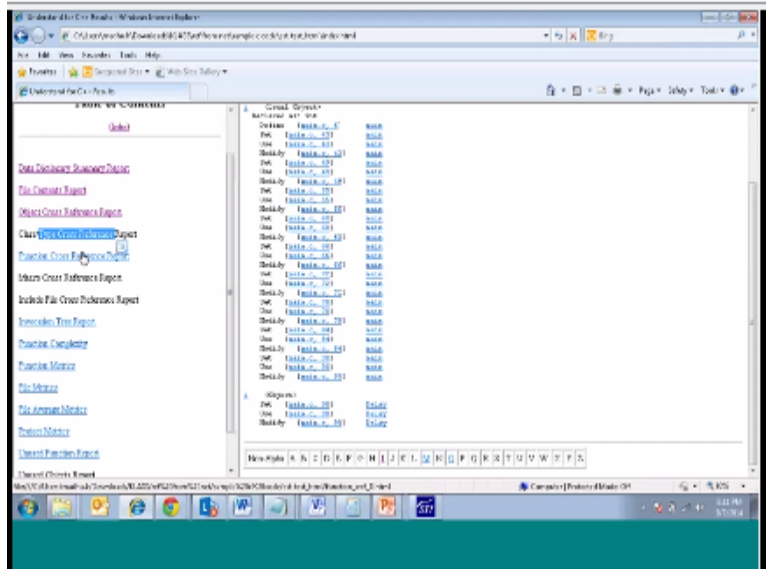
Similar to whatever we have browsing same way we can see their dictionary summary report show what are the directory dictionary that is available.  
 (Refer Slide Time: 29:58)



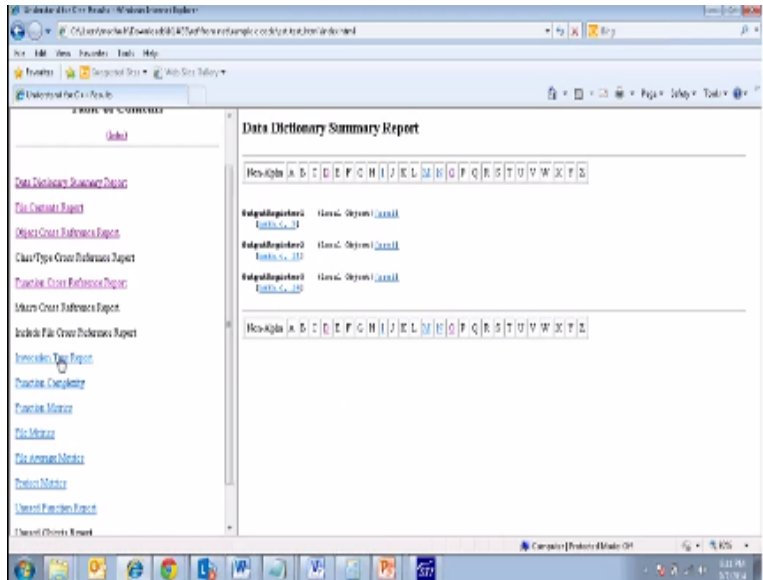
We can see the table data available that data later let us try to icon d is available what it is, yeah this is a function called in and that data it is which claim the function and front of the main . C 96 line is order so something like a good linking report it generates that is a good data dictionary summary and can see the file contents report what are the files that in an additional it is an available and main. C similarly it will list completely for the entire project. Since here only one file is here showing the below global function it will be functions you can see delay main, nothing.  
 (Refer Slide Time: 30:51)



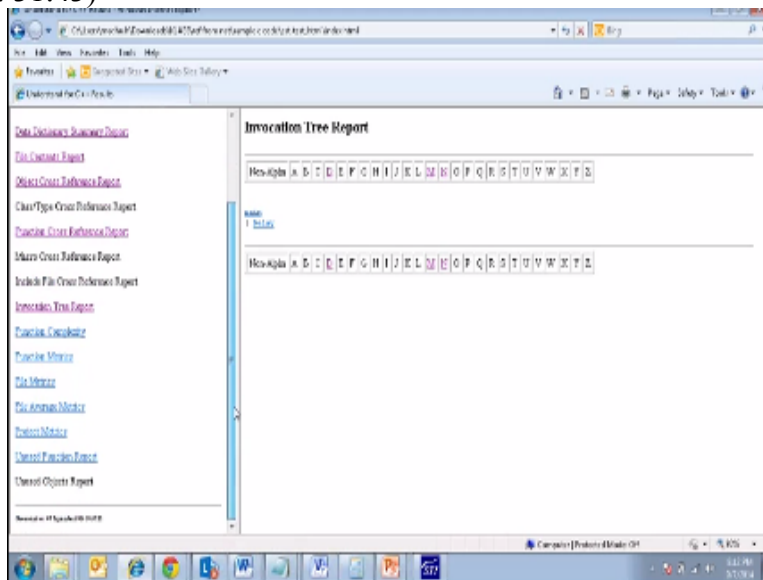
Object cross reference which will see there are, this many objects local objects that used modify Affects modify how many are there each line of the file.  
(Refer Slide Time: 31:07)



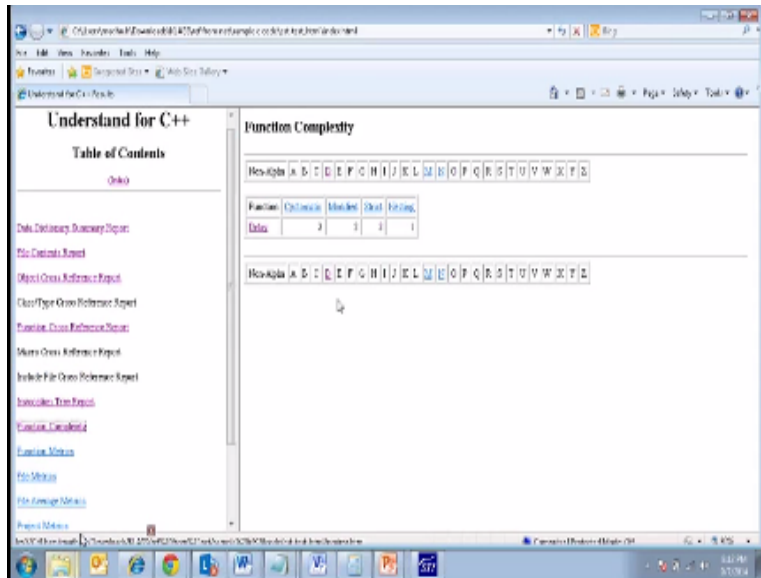
Delay also have about the set used modify file so these are class or cross functional cross reference reports restrictions so as these functions main delay, and data dictionary along with this additional thing data.  
(Refer Slide Time: 31:31)



This is output registers can see, okay next one we can see the in location report. (Refer Slide Time: 31:43)



So delay then nothing this delay is calling nothing location main should be calling related also we see, we can see at from main delay is getting call early nothing is independent not calling it so what are the calls are there main is calling is higher level one level has delay these the report (Refer Slide Time: 32:09)



So below one is the, function complexity in test we will try to see what it does the function complexity should be is a that what complexity level is there so cyclamate which to we know that delay has to just, we know where it is.  
 (Refer Slide Time: 32:49)

```

// sample est program
int main(void)
{
    // i declared
    int i;
    // registers declared
    volatile unsigned char * ModeRegister1;
    volatile unsigned char * outputRegister1;
    volatile unsigned char * ModeRegister2;
    volatile unsigned char * outputRegister2;
    //mode register & output register
    volatile unsigned char * ModeRegister3;
    volatile unsigned char * outputRegister3;

    ModeRegister1 = 0x87A05008; // PF GPIO mode register address in Hex
    outputRegister1 = 0x87A05000; // PF GPIO output register address in Hex
    ModeRegister2 = 0x87A03008; // PD GPIO mode register address in Hex
    outputRegister2 = 0x87A03000; // PD GPIO output register address in Hex

    ModeRegister3 = 0x87A04008; // PE GPIO mode register address in Hex
    outputRegister3 = 0x87A04000; // PE GPIO output register address in Hex

    *ModeRegister1 =0x00FF; // Configure Port F bit 0 as output
    *ModeRegister2 =0x00FF; // Configure Port b bit 0 as output
    *ModeRegister3 =0x00FF; // Configure Port E bit 0 as output

    while (1)
    {
  
```

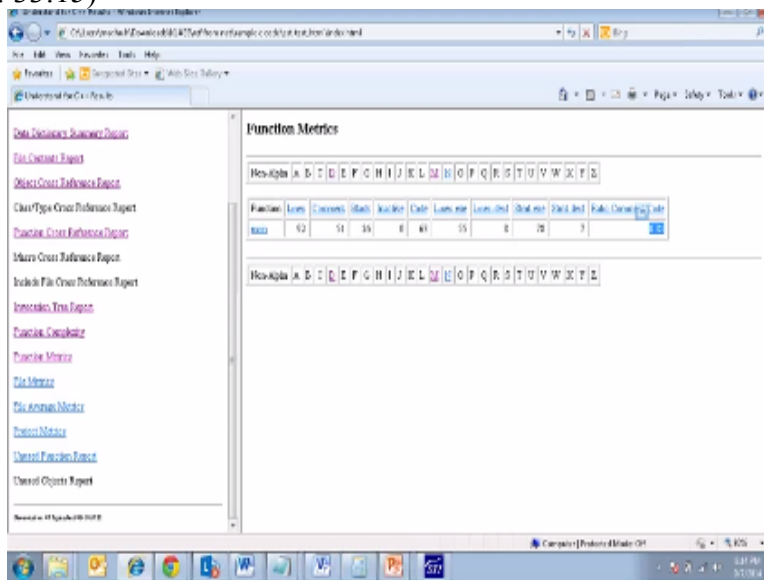
(Refer Slide Time: 32:51)

```

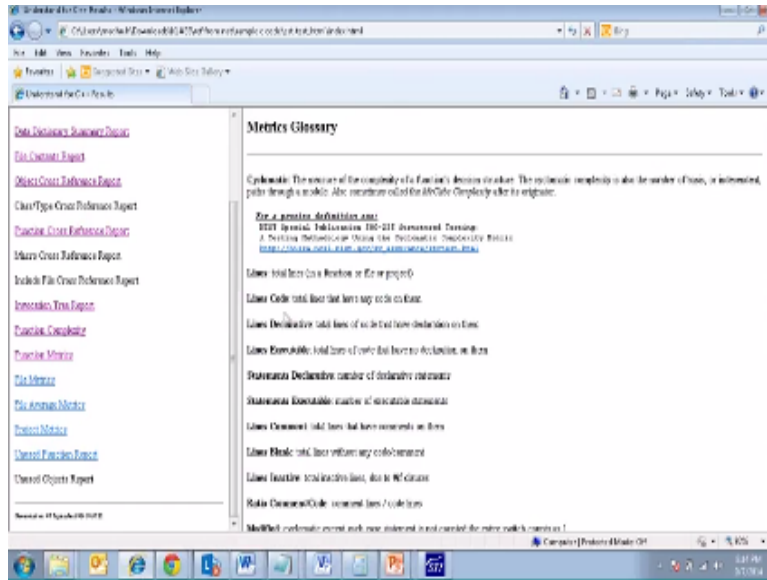
*outputRegister3 = 0x0000; //green, Yellow, & Red LEDs ON
for (i=0; i<1000000; i++); //delay
*outputRegister1 = 0x0027; // disB
*outputRegister2 = 0x0008; // disB
*outputRegister3 = 0x0000; //green, Yellow, & Red LEDs ON
for (i=0; i<1000000; i++); //delay
}
return 0;
}
delay()
{
for (i=0; i<1000000; i++)
nothing(); //delay
return 0;
}
Nothing()
{
return 0;
}
}

```

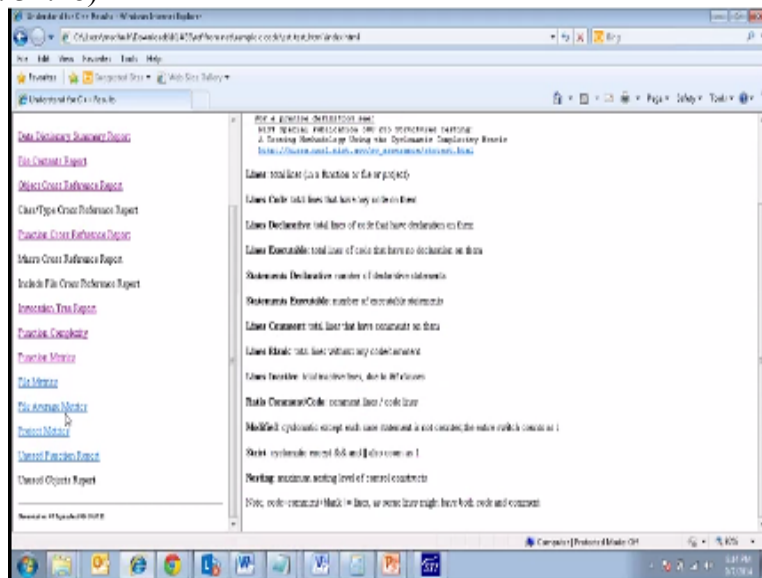
So the remind this one and this one it can break two path right one is with this one is with this, for loop is 1 are this is the normal that is why showing as cyclomatic form this is one nesting we are seeing that right.  
(Refer Slide Time: 33:15)



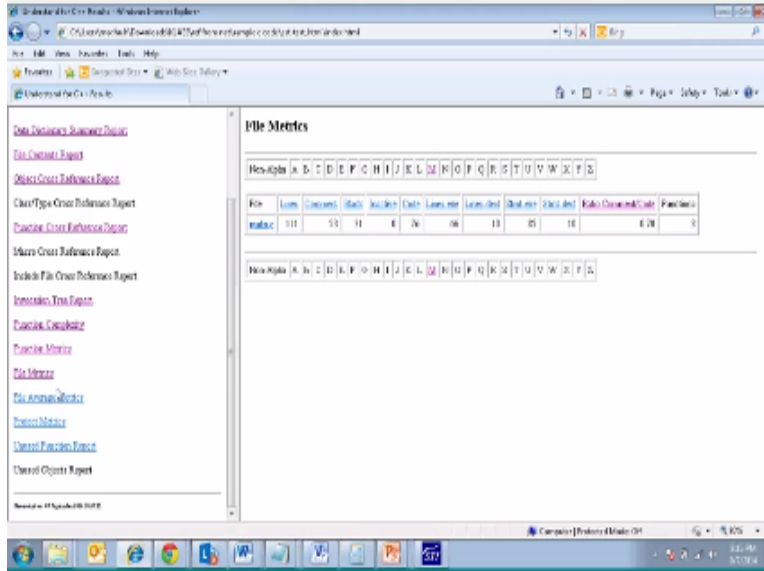
Each function matrices is going to show out for delay total number of line is stands comment is 1, it will not comment it is delay okay, total number of 2 etc so here is very bad ratio comment code that has to be make it 100% when we here again, similarly we can select other function main it is total number of line is line to comment is 51 blank line is 26 is no in active 63 line sofa code is there line executable it should be 5.  
They declared it 75 and 8 statement is executable is 78, statement declaration is 7 81% is the ratio of the comment was has.  
(Refer Slide Time: 34:20)



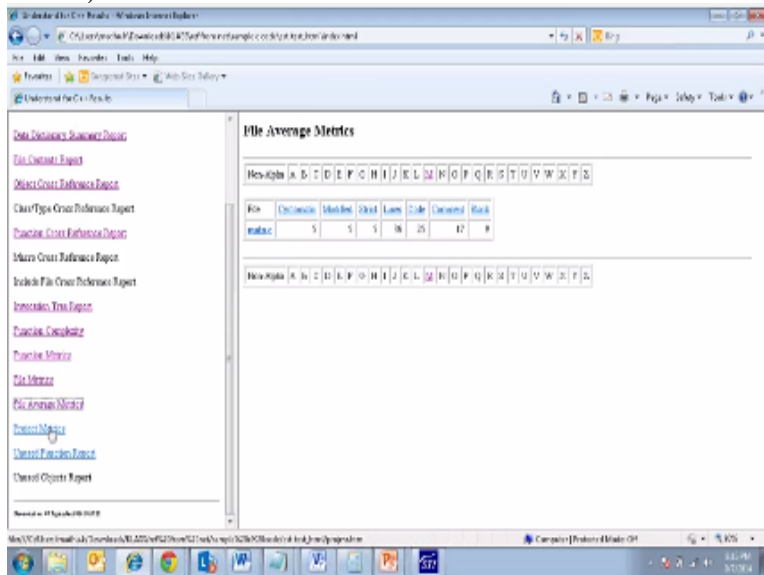
We can see definition of each of this the measure of complexity or a function the cyclamate is also a number of basics or independent part 2 order we know this here the about diplomatic for active complexity embedded software testing class.  
(Refer Slide Time: 34:48)



So all the definitions of comment matrix that we have generated  
(Refer Slide Time: 34:53)

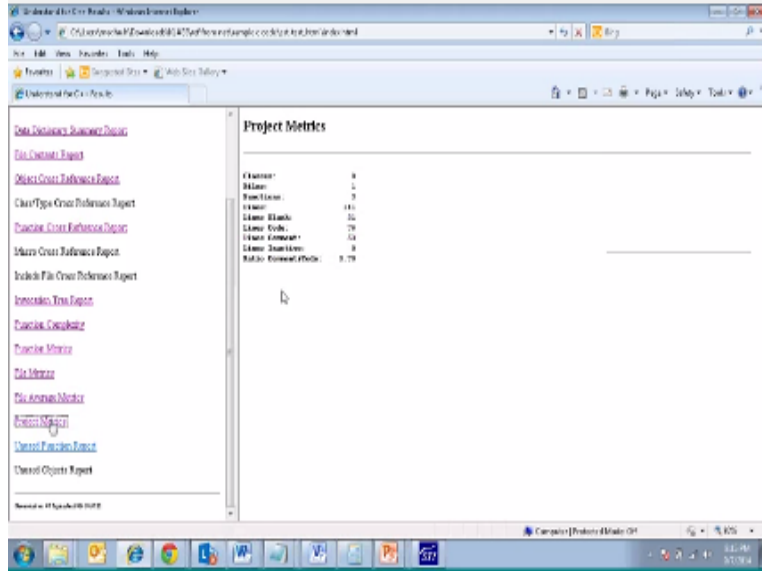


File matrix the entire files like this we have both this will sure as many files has we have total number of files in similar way, at the functional level shown as be find out.  
(Refer Slide Time: 35:14)

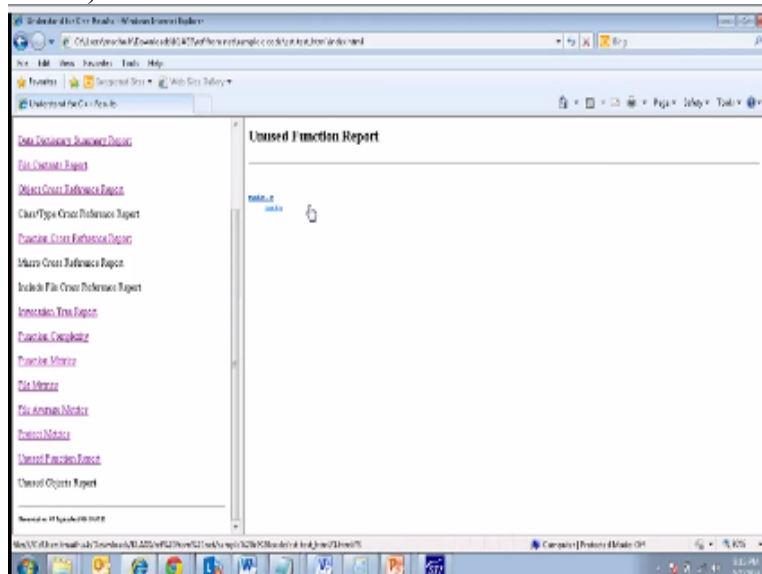


File average matrix  
(Refer Slide Time: 35:17)

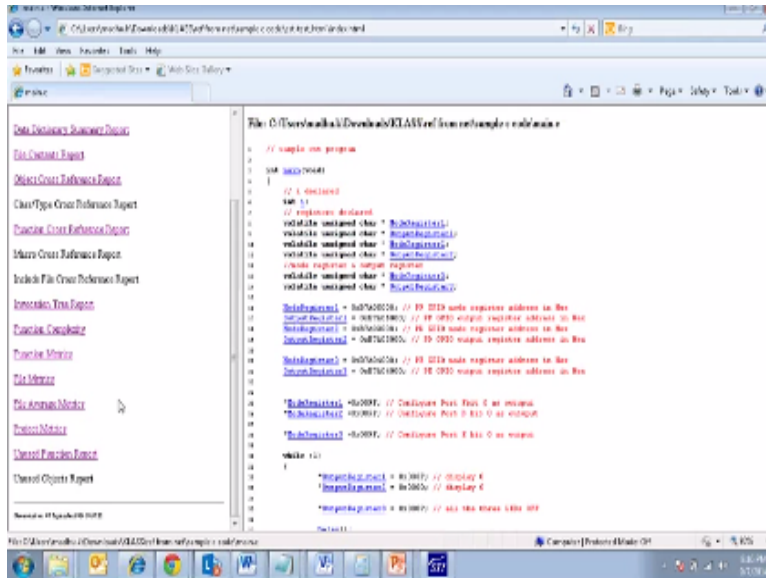




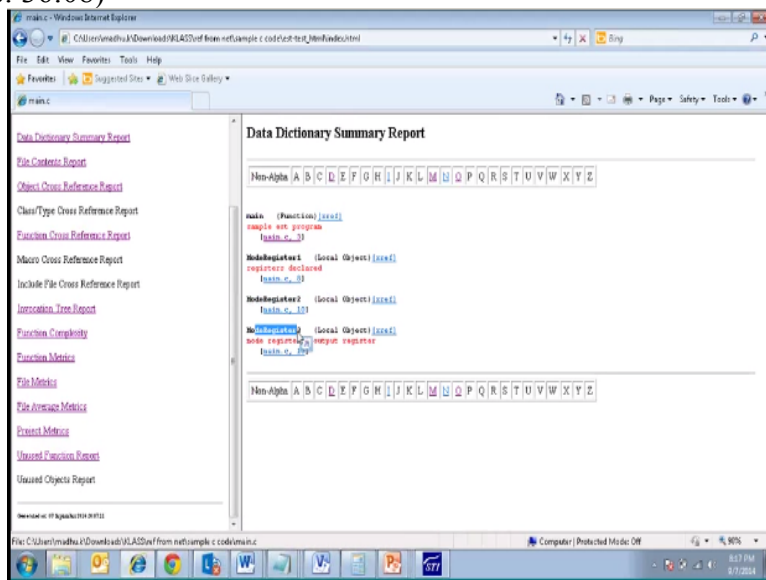
Entire project matrix now sees this report here and display here.  
 (Refer Slide Time: 35:25)



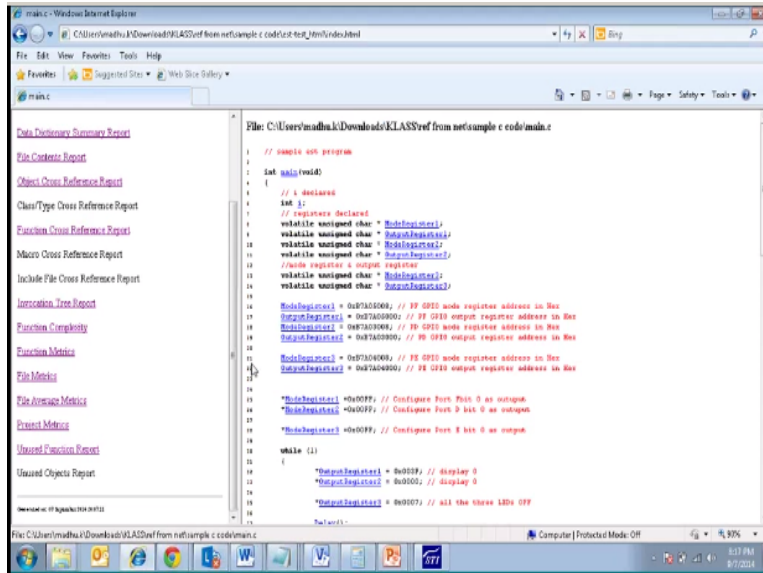
Unused functions anything is used that sure it is interesting what is unused.  
 (Refer Slide Time: 35:31)



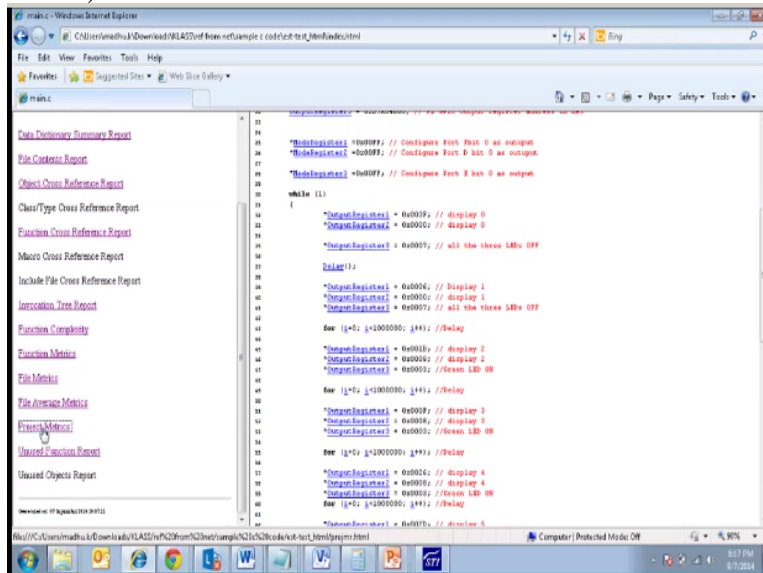
Okay so there are three unused functions are there, unused.  
 (Refer Slide Time: 36:08)



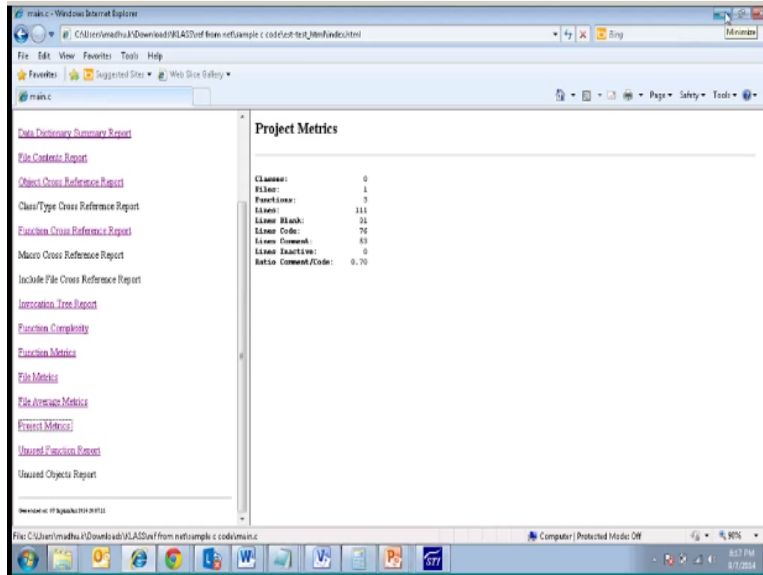
Unused functions it says as a function, basically the variable or anything could be mod register 1, mod register 2, mod register 3, these are used I am saying.  
 (Refer Slide Time: 36:19)



Mod register 1 is assign based, which are not used anywhere right; it is showing yes, it is unused, defined and declared here.  
(Refer Slide Time: 36:37)



Is never used, so likewise you can, get for the complex, just imagine a program is having some 10% of the code, so we just use the project matrix, to analyze.  
(Refer Slide Time: 36:47)



How good shape it is, and what is vector, it is available in terms of matrix, the complexity in all the term, okay.

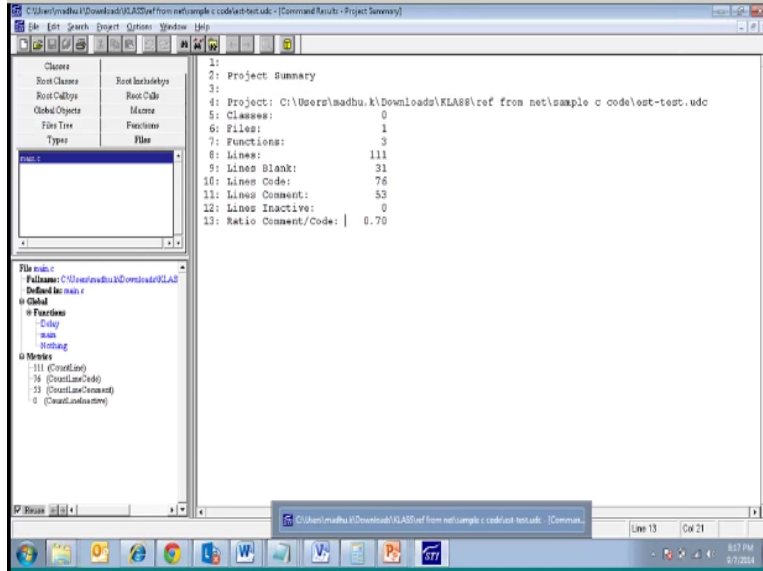
(Refer Slide Time: 37:03)

## Exercise 6

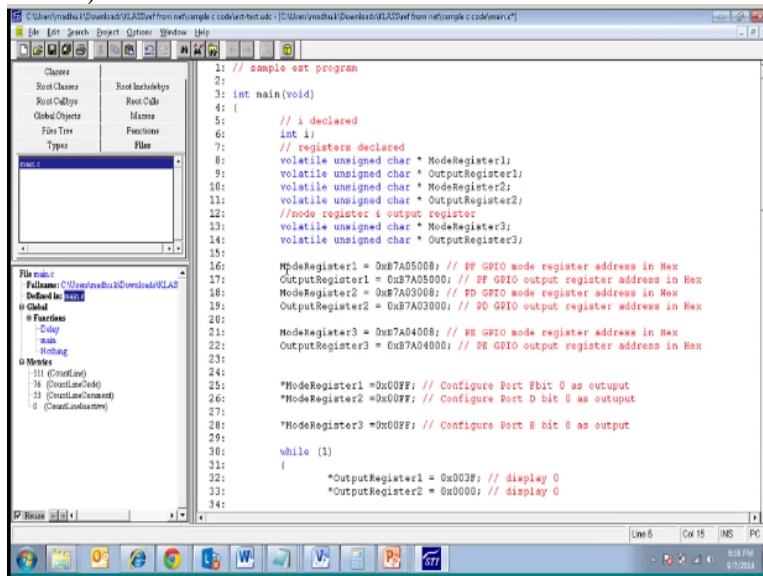
- Walkthrough of Understand for C/C++, a static analysis tool:
  - Example embedded C code
  - Configure the project into Understand for C/C++ tool
  - Select the metrics for the analysis
  - Generate the project report
  - Analyze the report

So the other thing, that we can see, the next one is calling a type.

(Refer Slide Time: 37:06)



Of course, we can source file here also.  
 (Refer Slide Time: 37:13)



You can edit also, you can do this like editor the project, also as well.  
 (Refer Slide Time: 37:26)

```

78:         for (i=0; i<1000000; i++); //Delay
79:
80:         *OutputRegister1 = 0x003F; // display 8
81:         *OutputRegister2 = 0x0008; // display 8
82:         *OutputRegister3 = 0x0000; //green, Yellow, & Red LEDs ON
83:
84:         for (i=0; i<1000000; i++); //Delay
85:
86:         *OutputRegister1 = 0x0027; // dis9
87:         *OutputRegister2 = 0x0008; // dis9
88:         *OutputRegister3 = 0x0000; //green, Yellow, & Red LEDs ON
89:
90:         for (i=0; i<1000000; i++); //Delay
91:
92:     }
93:     return 0;
94: }
95:
96: Delay()
97: {
98:     for (i=0; i<1000000; i++)
99:     {
100:         Nothing(); //Delay
101:     }
102:     return 0;
103: }
104:
105: }
106:
107: Nothing()
108: {
109:     return 0;
110: }
111: }

```

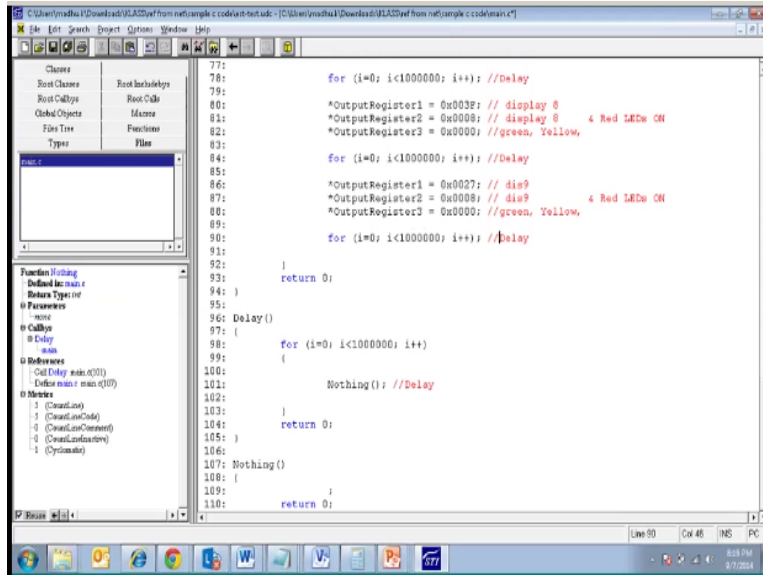
Similarly it has the linked delay, for each option, for each function, you can find where and all it is used, what is the line, what is the property, likewise you can see, the details. (Refer Slide Time: 37:40)

```

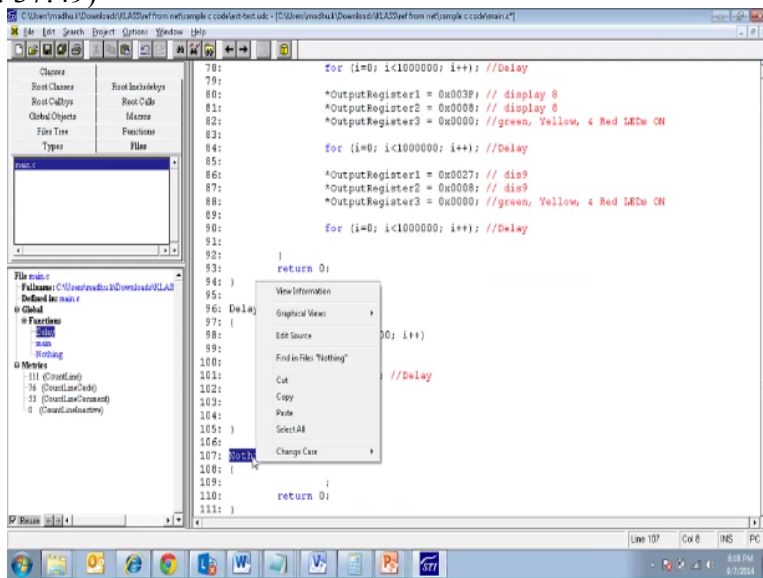
2: Properties for main.c
3:
4: File: C:\Users\madhu.k\Downloads\VLAS8\ref from net/sample code/main.c
5:
6: Type: PC
7: Code Type: C/C++
8: Size: 2957
9: Created: Mon Jul 07 21:47:14 2014
10: Last Modified: Sun Sep 07 19:59:52 2014
11: File Parse State: File unchanged
12:
13: Classes: 0
14: Functions: 3
15: Lines: 111
16: Blank Lines: 31
17: Code Lines: 76
18: Comment Lines: 52
19: Statements: 0
20: Ratio Comment/Code: 0.70

```

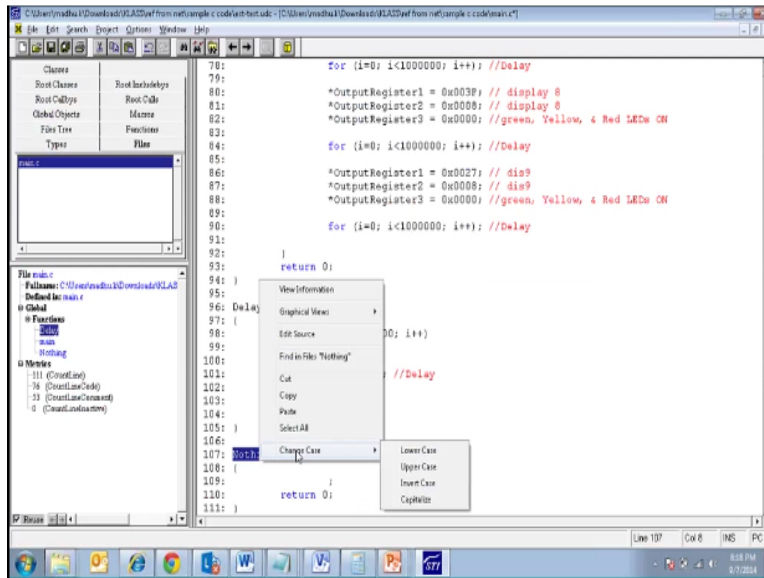
We can go forth and back for these different tools. (Refer Slide Time: 37:44)



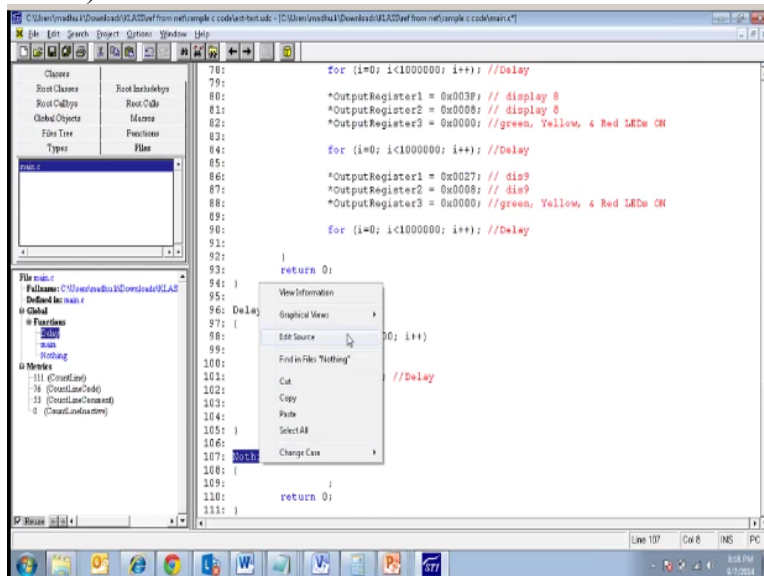
Let us try to understand other options, like each function,  
(Refer Slide Time: 37:49)



We can change case,  
(Refer Slide Time: 37:53)

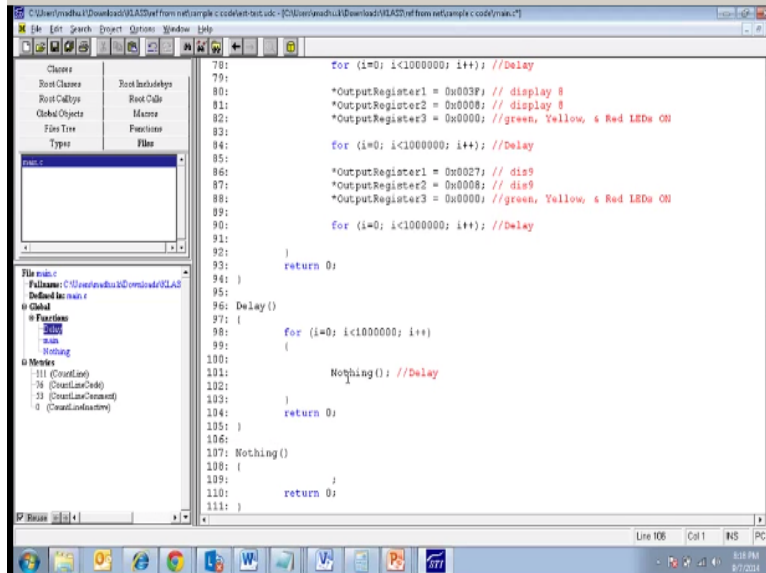


Like an example, edit source  
(Refer Slide Time: 37:54)

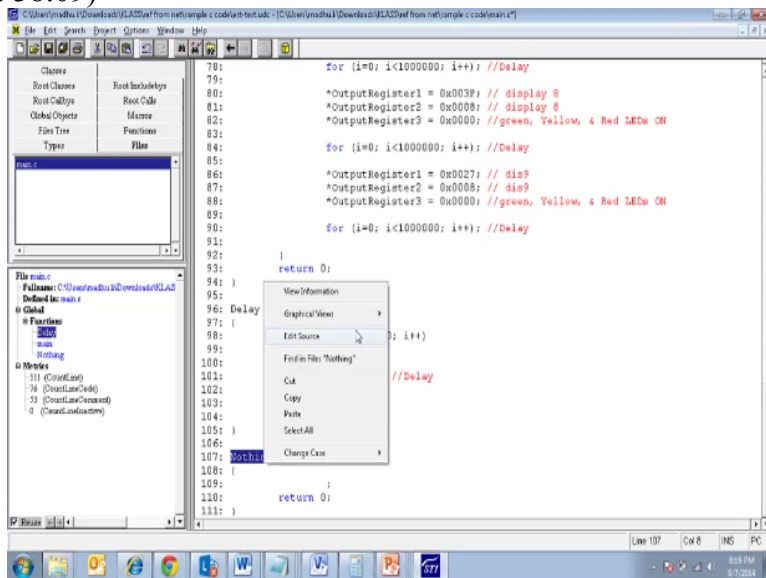


It will open the source, and since there is no separate linked function.  
(Refer Slide Time: 38:01)





Now let us see view information,  
(Refer Slide Time: 38:09)



The left hand side we can see,  
(Refer Slide Time: 38:15)

```

78:                                     for (i=0; i<1000000; i++) //Delay
79:
80:                                     *OutputRegister1 = 0x003F; // display 8
81:                                     *OutputRegister2 = 0x0008; // display 8
82:                                     *OutputRegister3 = 0x0000; //green, Yellow, & Red LEDs ON
83:
84:                                     for (i=0; i<1000000; i++) //Delay
85:
86:                                     *OutputRegister1 = 0x0027; // dis9
87:                                     *OutputRegister2 = 0x0008; // dis9
88:                                     *OutputRegister3 = 0x0000; //green, Yellow, & Red LEDs ON
89:
90:                                     for (i=0; i<1000000; i++) //Delay
91:
92:                                     }
93:                                     return 0;
94: }
95:
96: Delay ()
97: {
98:     for (i=0; i<1000000; i++)
99:     {
100:
101:         Nothing(); //Delay
102:
103:     }
104:     return 0;
105: }
106:
107: Nothing ()
108: {
109: }
110:     return 0;
111: }

```

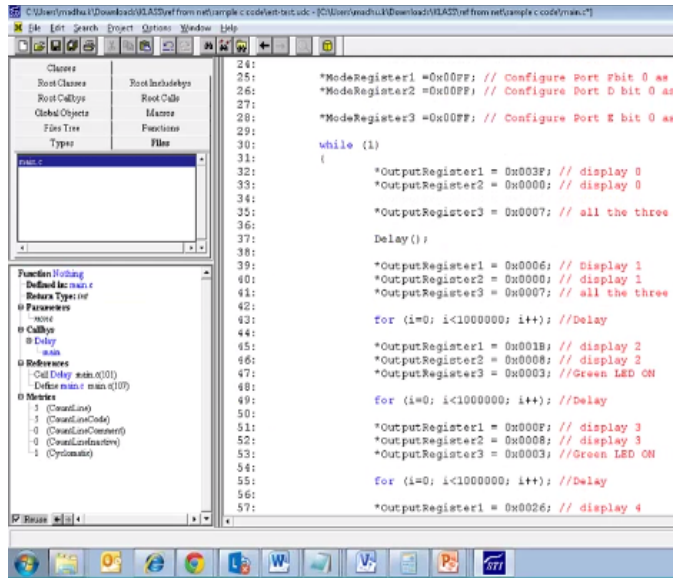
The parameters is nothing, and the return type is an integer, by default it is a c compiler, or whatever they are saying, it is showing that the other, one is a call yes, (Refer Slide Time: 38:27)

```

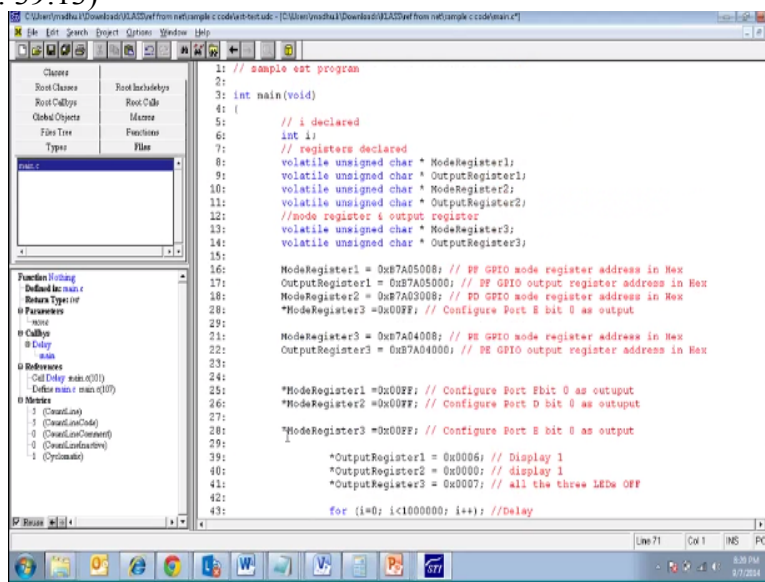
78:                                     for (i=0; i<1000000; i++) //Delay
79:
80:                                     *OutputRegister1 = 0x003F; // display 8
81:                                     *OutputRegister2 = 0x0008; // display 8
82:                                     *OutputRegister3 = 0x0000; //green, Yellow, & Red LEDs ON
83:
84:                                     for (i=0; i<1000000; i++) //Delay
85:
86:                                     *OutputRegister1 = 0x0027; // dis9
87:                                     *OutputRegister2 = 0x0008; // dis9
88:                                     *OutputRegister3 = 0x0000; //green, Yellow, & Red LEDs ON
89:
90:                                     for (i=0; i<1000000; i++) //Delay
91:
92:                                     }
93:                                     return 0;
94: }
95:
96: Delay ()
97: {
98:     for (i=0; i<1000000; i++)
99:     {
100:
101:         Nothing(); //Delay
102:
103:     }
104:     return 0;
105: }
106:
107: Nothing ()
108: {
109: }
110:     return 0;
111: }

```

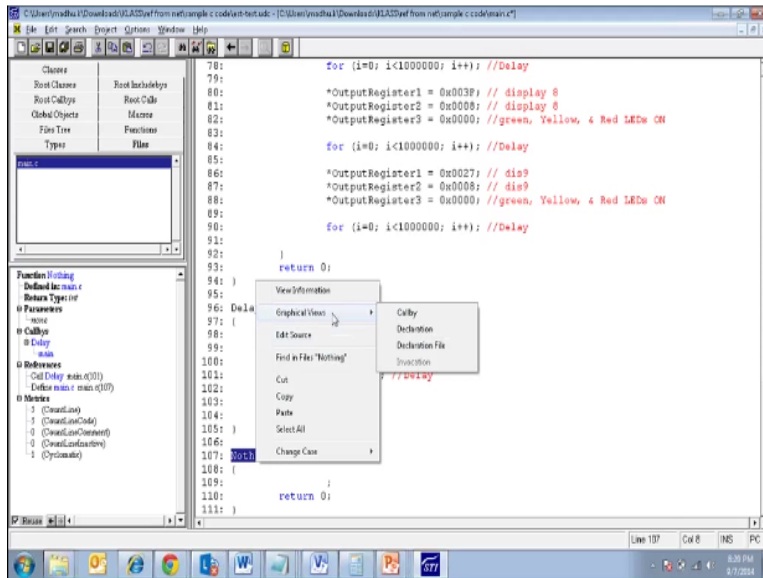
Who are all calling this guy, delay is calling ,and delay is getting interrupt called by main, similarly call references, call delay main call under one line. (Refer Slide Time: 38:50)



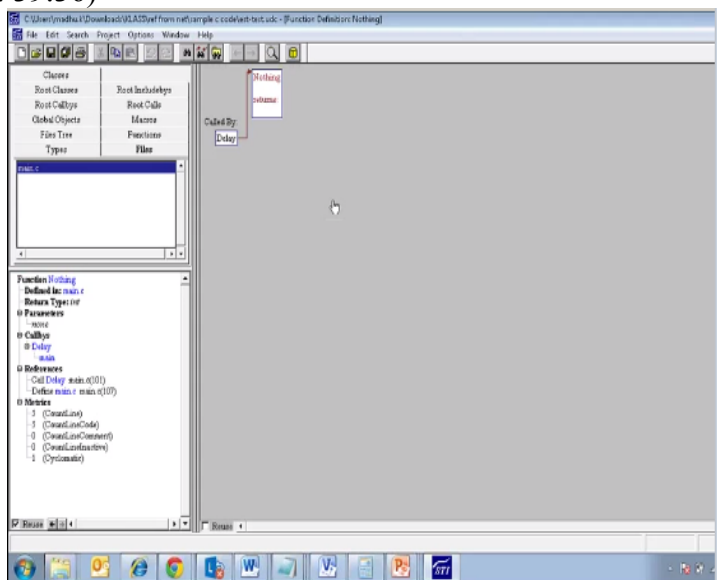
We define main.c, it is 107 , it is showing the line number, and matrix, how many number of lines are there ,there are 5,the comment is 0,which has no comment in c ,inactive is also nothing, cyclometric is just one path, that is how simple it is.  
(Refer Slide Time: 39:13)



Now for static analysis, where you have a complex program, as I was showing in one of the embedded class, testing class, there is a call tree, with this option you can see,  
(Refer Slide Time: 39:26)

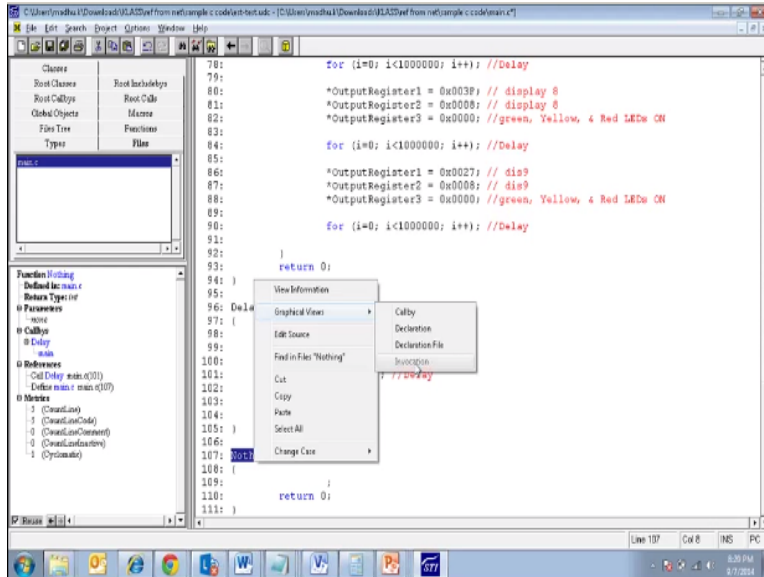


There you can see call by, declaration, declaration profile, like this you can see called by. (Refer Slide Time: 39:36)

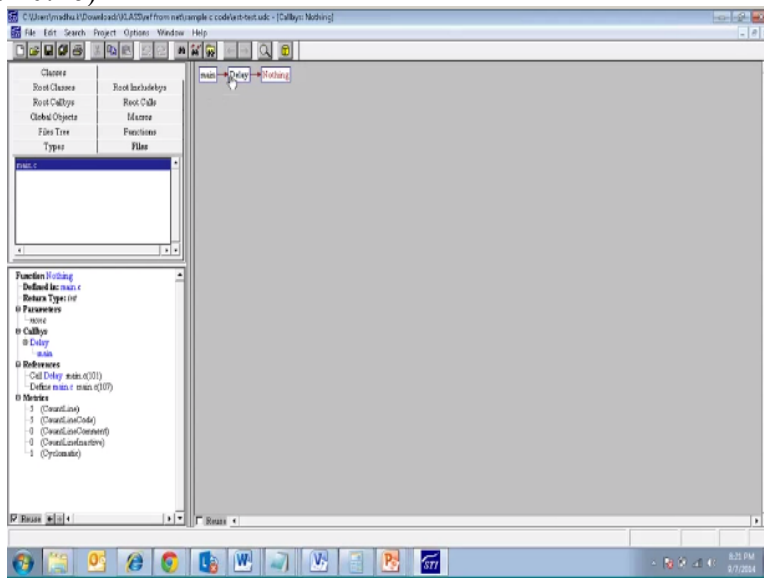


And what it returns, you can see it, similarly, the declaration file, (Refer Slide Time: 39:47)

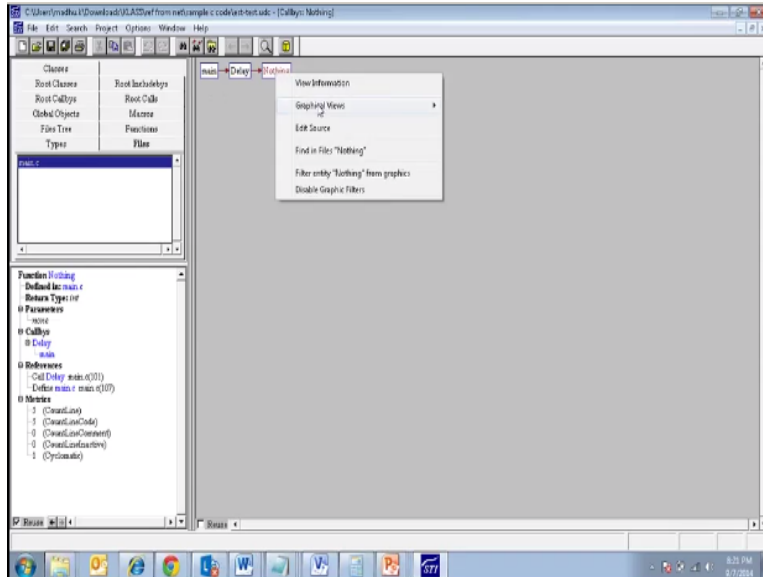




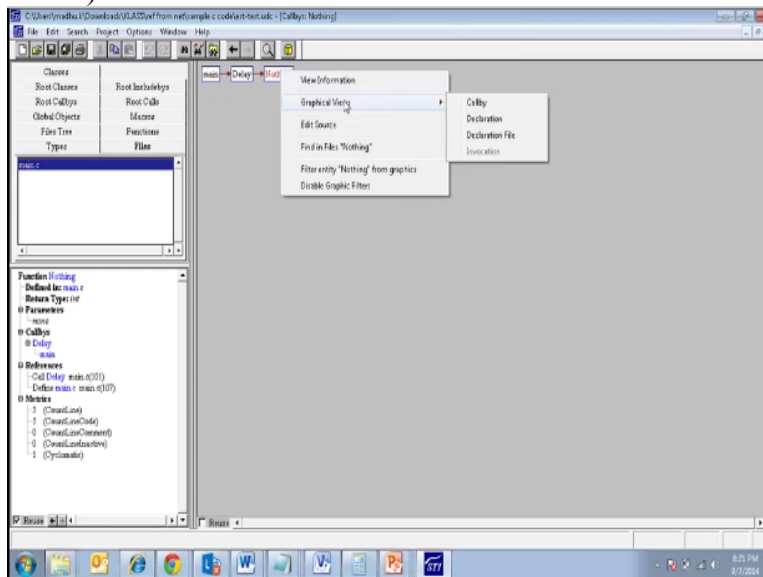
We did not invoke any other function, it will just return it, whereas the higher functions, we will see that ,call by we can see overall calling, we should delay and delay should be called by main, we can see nicely (Refer Slide Time: 40:16)



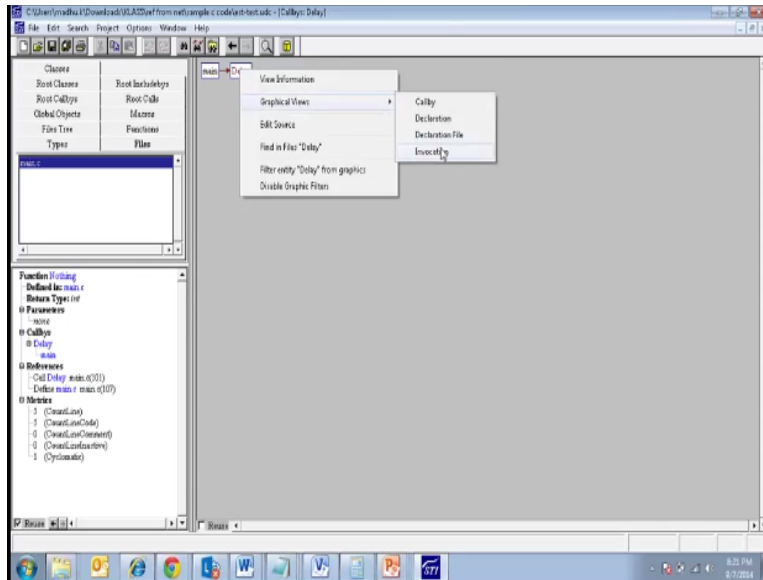
When the main is calling delay, delay is calling the attribute, we can see individual also. (Refer Slide Time: 40:19)



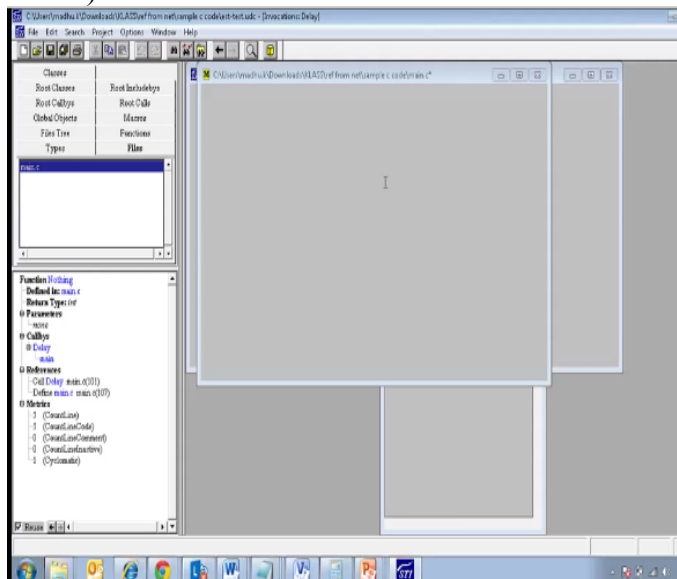
The call by is showing the same thing ,and you can select, how many levels, whatever level you want,  
 (Refer Slide Time: 40:31)



Then next, let us try to see for delay, call by, system main, and similarly, it is calling, invocation it is enabled right,  
 (Refer Slide Time: 40:44)

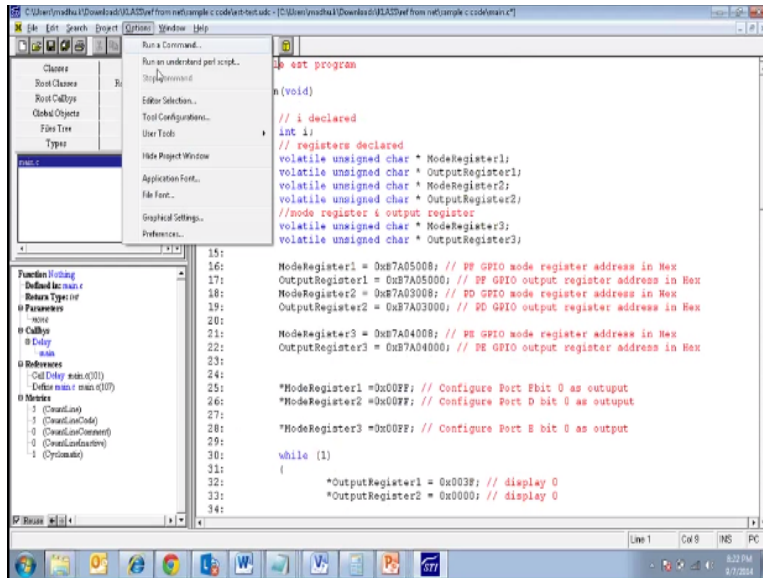


The delay is getting invoked, nothing, so like this it is very useful, to have a control flow, or control graph, it is better to have, these kinds of reports for static analysis, we use it for review, (Refer Slide Time: 40:58)

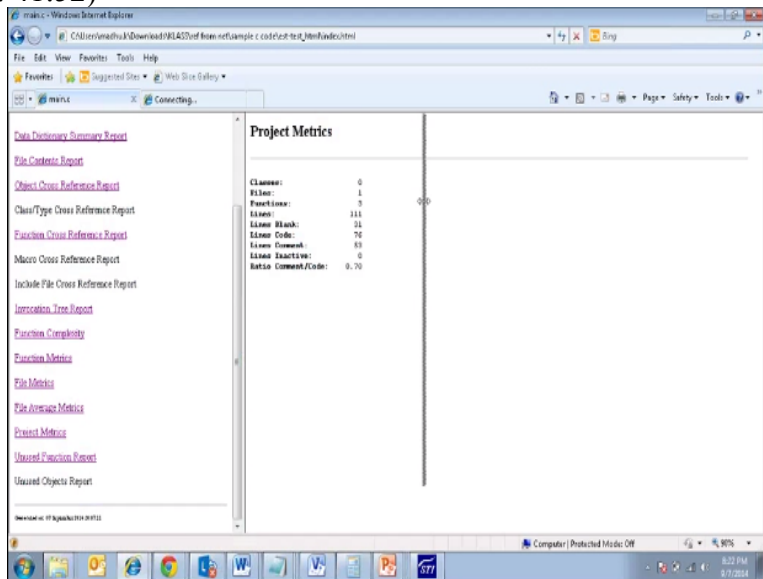


And finding the bugs, anything is there statically, that is how you can use these, understand for C++, in general, okay so preferences, anything we have missed, we can have times and all that, there are lot of options, but basic options. (Refer Slide Time: 41:33)





What we have studied about, creating project, configuring it, and selecting the kind of report, okay now let us see some , this one independently, you can see the first report, (Refer Slide Time: 41:52)



It is showing the first file, I will just enable it, these are sub lines, you can see it automatically, when you click it will automatically, show the database, (Refer Slide Time: 42:02)

```

// simple led program
int main(void)
{
    // I declared
    int i;
    // registers declared
    volatile unsigned char * OutputRegister1;
    volatile unsigned char * OutputRegister2;
    volatile unsigned char * OutputRegister3;
    //make register 4 output register
    volatile unsigned char * OutputRegister4;
    // registers
    *OutputRegister1 = 0x7A0000; // 7A 7A00 mode register address in Max
    *OutputRegister2 = 0x7A0001; // 7F 0F20 output register address in Max
    *OutputRegister3 = 0x7A0000; // 7F 0F20 mode register address in Max
    *OutputRegister4 = 0x7A0001; // 7F 0F20 output register address in Max
    *OutputRegister1 = 0x7A0000; // 7F 0F20 mode register address in Max
    *OutputRegister2 = 0x7A0001; // 7F 0F20 output register address in Max
    *OutputRegister1 = 0x00FF; // Configure Port With 8 as output
    *OutputRegister2 = 0x00FF; // Configure Port 0 bit 0 as output
    *OutputRegister1 = 0x00FF; // Configure Port 8 bit 0 as output
    while (1)
    {
        *OutputRegister1 = 0x003F; // display 0
        *OutputRegister2 = 0x0000; // display 0
        *OutputRegister3 = 0x0000; //display 0
        *OutputRegister4 = 0x0007; // all the these LEDs OFF
    }
}

```

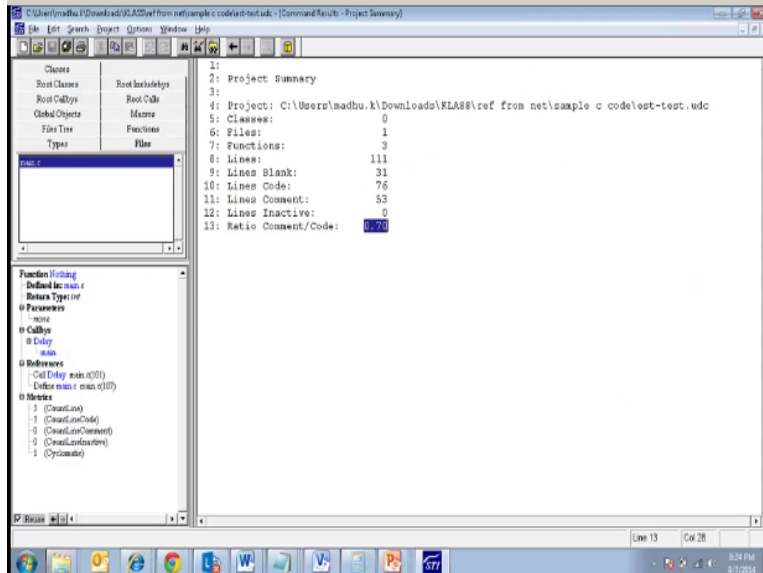
What are the objects, when it is showing, etc, similarly if you call anything else, delay is the function, referred by main, similarly and how I is used, it is defined here, modified here, okay now we have seen this, anything else you want to try, probably, we will try to put the ratio, as 100%, wherever the executable lines, comment (Refer Slide Time: 43:21)

```

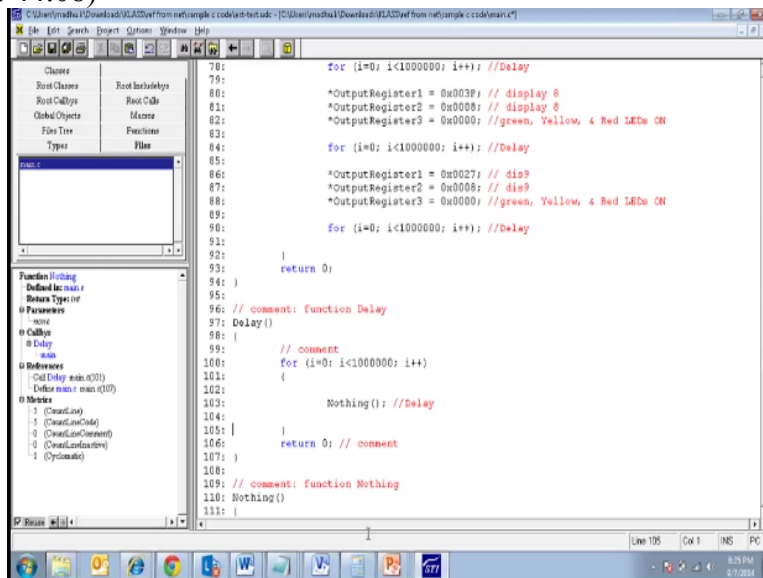
78:         for (i=0; i<1000000; i++); //Delay
79:
80:         *OutputRegister1 = 0x003F; // display 8
81:         *OutputRegister2 = 0x0008; // display 8
82:         *OutputRegister3 = 0x0000; //green, Yellow, 4 Red LEDs ON
83:
84:         for (i=0; i<1000000; i++); //Delay
85:
86:         *OutputRegister1 = 0x0027; // dis9
87:         *OutputRegister2 = 0x0008; // dis9
88:         *OutputRegister3 = 0x0000; //green, Yellow, 4 Red LEDs ON
89:
90:         for (i=0; i<1000000; i++); //Delay
91:
92:     }
93:     return 0;
94: }
95:
96: Delay()
97: {
98:     for (i=0; i<1000000; i++)
99:     {
100:         Nothing(); //Delay
101:     }
102: }
103:
104: return 0; // comment
105: }
106:
107: Nothing()
108: {
109: }
110: return 0;
111: }

```

And we will see, total projects, 70% let us try to make it 100%, (Refer Slide Time: 43:34)



By adding the missing command, this already the comment is there,  
 (Refer Slide Time: 44:08)



This should show as 100%, return 0,  
 (Refer Slide Time: 44:15)

```

81:         *OutputRegister2 = 0x0000; // display 0
82:         *OutputRegister3 = 0x0000; //green, Yellow, & Red LED ON
83:
84:         for (i=0; i<1000000; i++); //Delay
85:
86:         *OutputRegister1 = 0x0027; // dis9
87:         *OutputRegister2 = 0x0000; // dis9
88:         *OutputRegister3 = 0x0000; //green, Yellow, & Red LED ON
89:
90:         for (i=0; i<1000000; i++); //Delay
91:
92:     }
93:     return 0; // c
94: }
95:
96: // comment: function Delay
97: Delay()
98: {
99:     // comment
100:    for (i=0; i<1000000; i++)
101:    {
102:
103:        Nothing(); //Delay
104:    }
105:    return 0; // comment
106: }
107:
108:
109: // comment: function Nothing
110: Nothing()
111: {
112:    // comment
113:    return 0; // comment
114: }

```

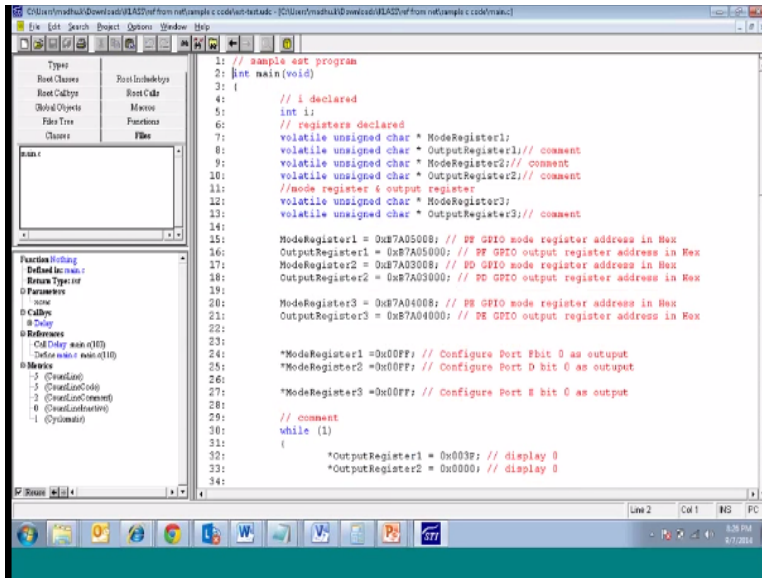
So you can add something like, so write a line of code, we have a comment, let us make sure that, anything we have missed, there is a comment, suppose 20% of violation is used, as if we are developer we trying to put the comments, we no need to put for the first one, so once it is done, better we need to save it, and analyze all files check for operate files this is file change at to now analyze the change files.  
(Refer Slide Time: 45:20)

```

1: #PROJECT_PATH#
2:
3:
4: Project: C:\Bore\midea\divisions\KLAR\test\test\sample\c\code\test-test.cpp
5: Classes: 0
6: Files: 1
7: Comments: 0
8: Lines: 114
9: Lines Blank: 25
10: Lines Code: 76
11: Lines Comments: 89
12: Lines Functions: 0
13: Non-Comment/Code: 12

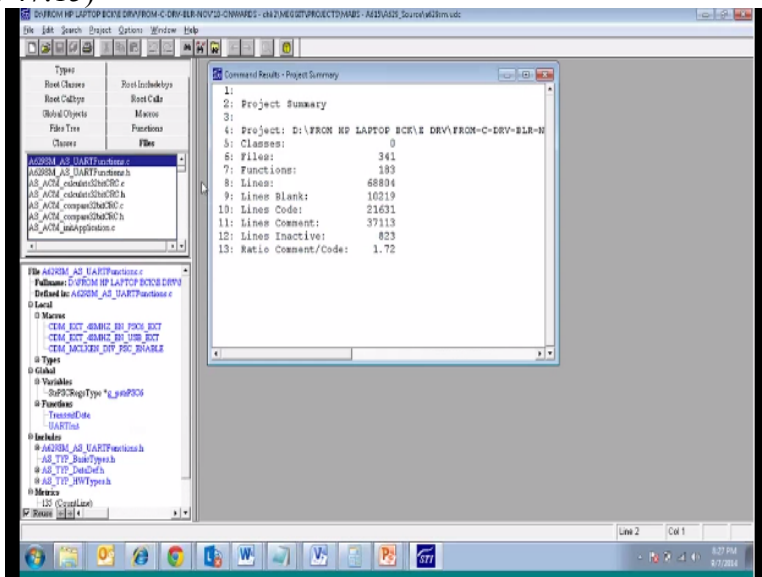
```

So let us straight analyzed it is completed successful if there error or something thinking of it will be a warning of it shows some errors. Okay now produce to see at this summary, the Varsity improvement is the F from 70 to 88% it has be posted. Still that means well it has missed it is about 12 lines you abuse to for the command.  
(Refer Slide Time: 45:51)



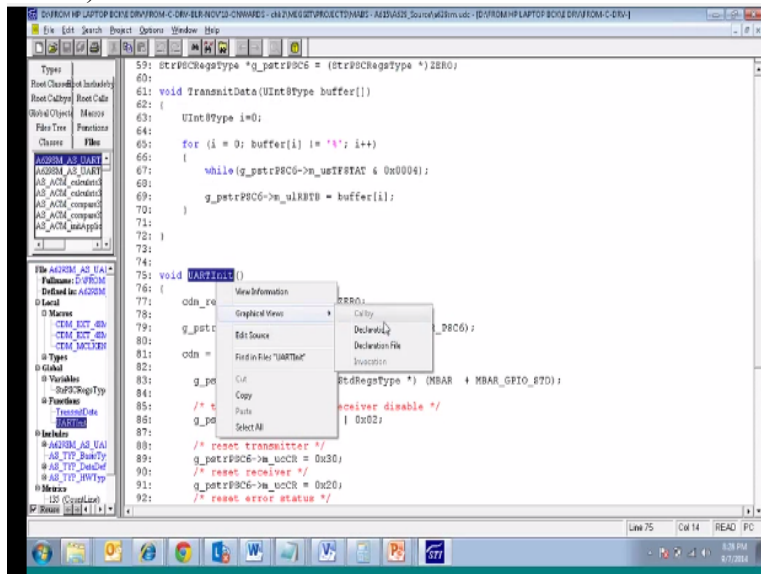
That you can take it array saves and fixed it and we can real on this okay next session are we want to try yourself with understand for this intrudes. Okay, so likewise we can comp lest targets have few other sample projects let us see it is dispose the reference. Okay close the current project and all it is okay. So nothing else okay, say other project is error then okay one more example it has see which will able to more complex project here. Shown of the sample of programme we can see there are files are there, there that is the command.

(Refer Slide Time: 47:15)

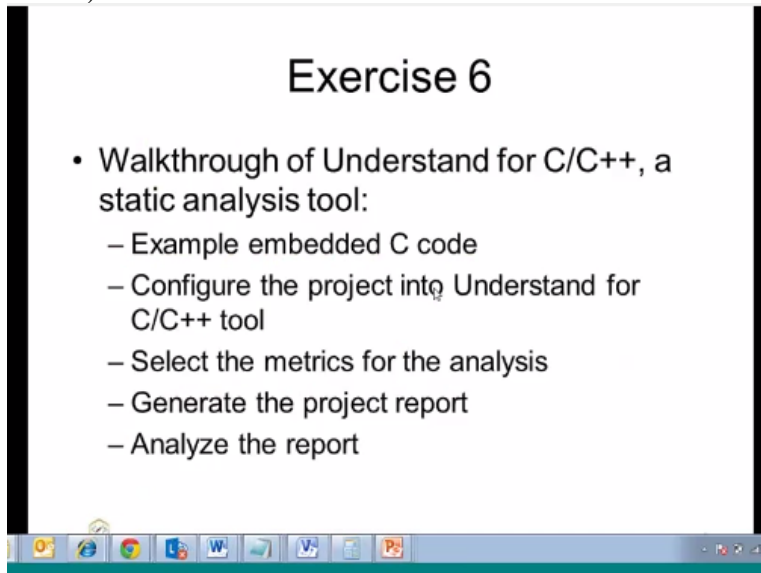


That the ratio was less it was less it was ratio more commands are there so it is more than one also is there so chance for that, also can happen why because you had of put more commands for

each are the line that is the reason. That is will showing more than 1.0 so we can see one example is one let state at C not showing call by see can sure this not showing for more okay. (Refer Slide Time: 47:43)



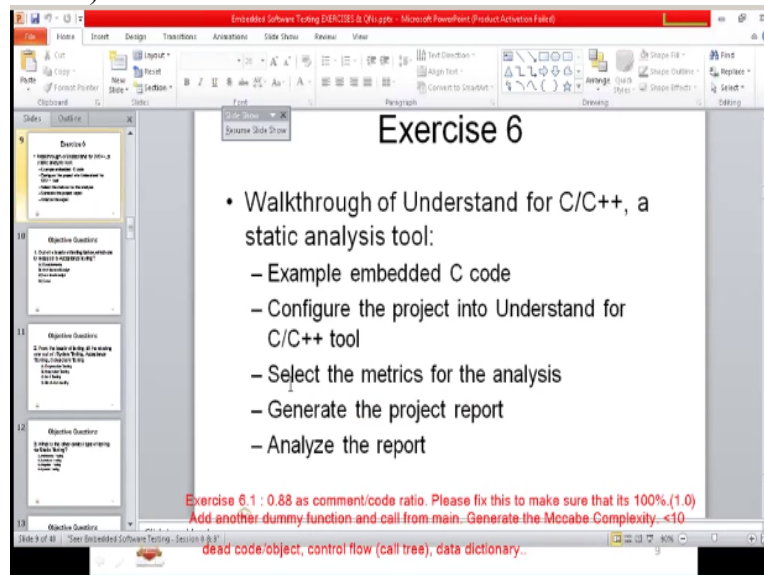
We basically added for yes so likewise you can develop a complex for the truanacy biggest thing. So that is we can prove the static analysis generating the report and configuring and analyzing the embedded software programme. (Refer Slide Time: 48:30)



So basically what we need is we will defined the matrix understand this standards what we need and try to do it. Okay so now let us try to put an exercise for today is base on this files, exercise are 6.1 else say so currently we have 0.88 has a comment was as ratio code. Please fix this to make sure that it is 100 % it should be 1.0 that is one example one exercise sorry other exercise could be, after doing this add another dummy function.

And calls from main generate the McCabe complexity and see that it is well within than have a simpler program that, must be complexity of again project should not be more than ten, search what is the meaning, so we need to do the exercise follow even code, I will share the code, and the same code you can use example configure the project and understand for C++, select the metrics, like McCabe lines of code, excibile lines, commands, anshusting, function references of course.

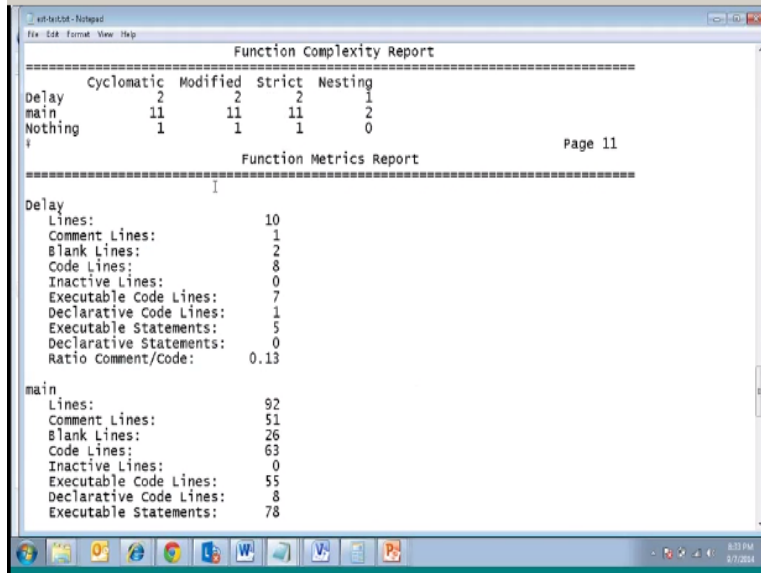
Important thing is dead code or objects, control flow or call tree we can use this, data flow or delta coupling you can do it or data dictionary, all the reports you can generate with the help of this understand for C++, generate the project report unless the report provided summary, so the summary you can using what we are saying using the HTML or using the tool itself.  
(Refer Slide Time: 51:35)



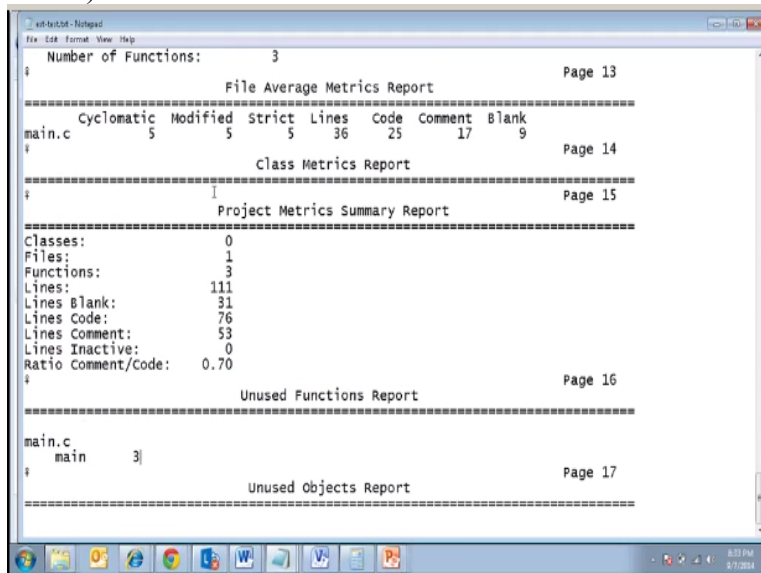
Sometime the tool may not be available, what we will do is? If the customer does not have tool or something we will genute the HTML and C. We can also see in next file, but text file is little tricky to understand.

(Refer Slide Time: 51:52)





HTML is a better way, same generated on HTML, you can see object cross references, while condone, and each file will how it is set each of the registers of the output local or variables, the object means referred variables, you can see for this cyclomatic modified strict nesting and all it shows, the various type of cyclomatic complex city, so that any way we have understood about basic cyclomatic complex city.  
(Refer Slide Time: 52:28)



And each function you can see the reports will lying commands etcetera, per delay main and nothing this functions that is the file metrics report, average metric report also used, and used function report, there is a three and used a object that are used, so that is what in the report how we can generate and use it okay.




So that will end understand for this principle tool we will go through, we will record we will touch base on this tool and understand more about more complex call which, that you can take it as an exercise and test case right, in the next exercise we have to study about other aspects like a test case management.

(Refer Slide Time: 53:29)

## Exercise 5

- Test Case & Defect Management with Testlink / Bugzilla:
  - Using the below link, establish the Testlink requirements, test cases, execution and results and the traceability
  - <http://demo.testlink.org/latest/login.php?note=logout>
  - Generate the Testlink report
  - Go through the Bugzilla possibly with an example.


8

Defect management using test link and bugzilla, and also one of the exercises, exercise 4 (Refer Slide Time: 53:36)

## Exercise 4 using CodeWarrior example

*222 lab #1 .mp4*

*222 lab #2 .mp4*

7

Target based execution, how it can be used? (Refer Slide Time: 53:43)

## Exercise 3 – unit testing

- Stub function creation and unit testing using LDRA unit testing tool

*Automate Unit Test Generation & Management with LDRAunit (HD).mp4*



We have seen LDRA cover sessions, so well to see about SKS generation, traceability, and check list filling in the next session, will that I will end the today's class thank you, bye.