



# **SIMIOBOARD SC-PASCAL**

## **QUICK START**

Doc. Ver.: SIMIOB-QUICKSTART V.1.0 En

Date: 28/07/2013

URL: [www.simioboard.com](http://www.simioboard.com)



# TABLE OF CONTENT

---

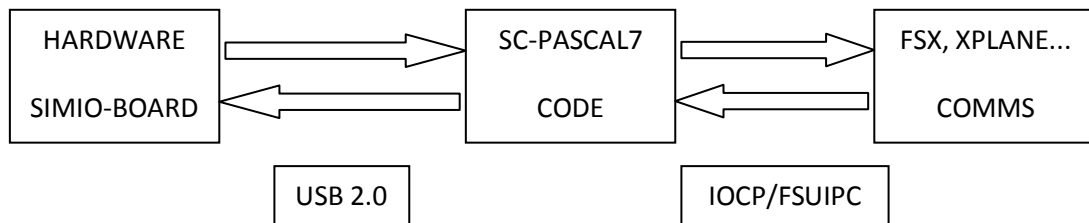
1. Intro.....	3
2. Requirements.....	3
3. Initial example.....	4
4. Connection example.....	11
5. Connections and hardware example.....	20

---



## 1. Intro

To start working with SC-PASCAL7 there is no need to have a high programming level but we have to have a clear idea of how the whole system (Simulator-SC-PASCAL-Hardware) is structured. There are three main parts in SC-PASCAL7:



SC-PASCAL lays between hardware and simulation software.

There's no need to manage the communications between SC-PASCAL7 and simulator, as it's managed internally by the compiler.

## 2. Requirements.

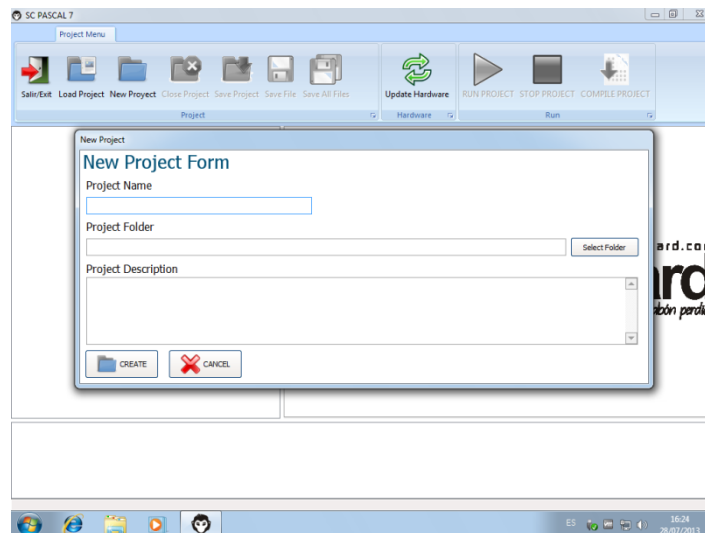
In order to follow this quick start you need the following:

- a. FSX
- b. IOCP Server for FSX
- c. FSUIPC for FSX
- d. SC-PASCAL7
- e. SIMIO BOARD USB MAIN (for chapter 4).
- f. SIMIO BOARD EXP 32 DISPLAYS (for chapter 4).
- g. One encoder connected to USB MAIN (for chapter 4).
- h. 6 pcs 7-segments displays connected to one displays card (for chapter 4).
- i. SC-PASCAL7 programmer user manual.

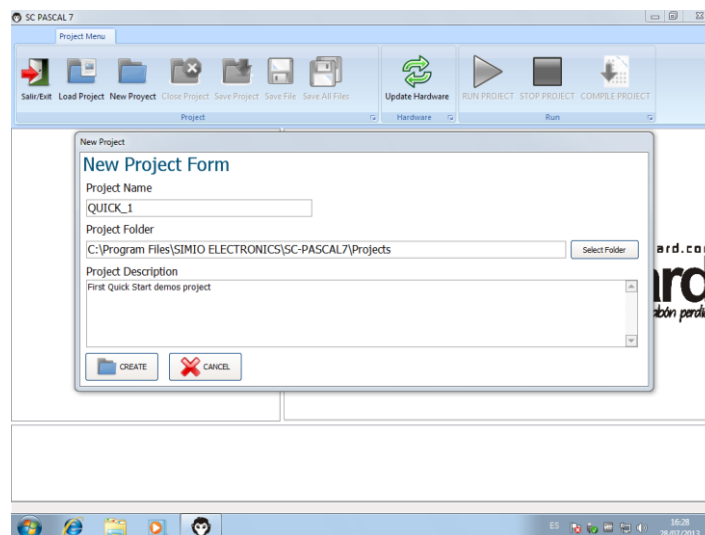


### 3. Initial example

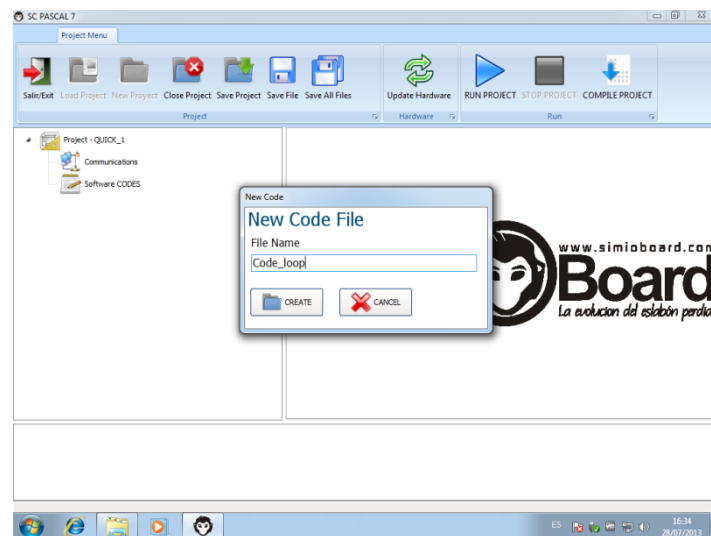
As initial example, we are going to write a small code without connecting with any external hardware or software. This example is a loop that when get a certain value stops and shows a message.



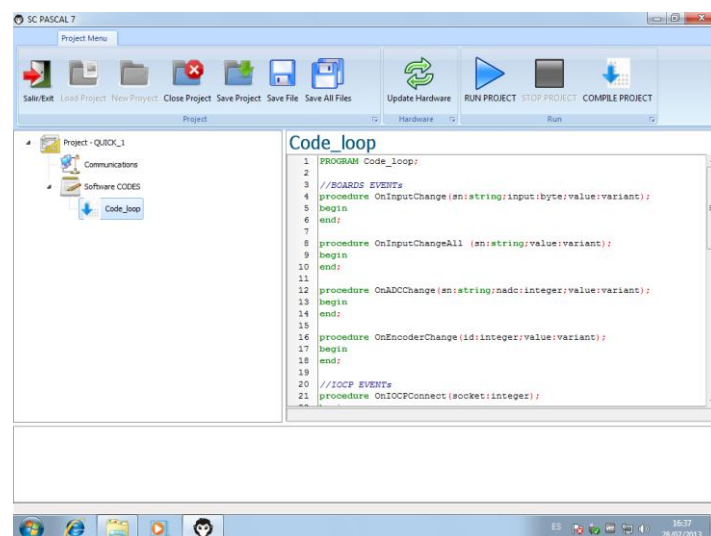
1. We create a new Project naming it “Quick 1” for example. Then we choose the folder “Projects” where SC-PASCAL7 is installed to save it.



2. Right-click on “Software CODES” on the objects window and click on “New CODE”.



3. Write the name of the code that we are going to create ("Code\_loop") and click on "CREATE".
4. Once the file is created, click on it and the working area will be shown on the right side.



5. In the code, look for the structure:

***//MAIN***

***Begin***

***//WRITE HERE YOUR CODE***

***End.***

6. This structure is the MAIN procedure, where the program starts when launched.
7. Inside this structure we are going to write the code for the repetitive structure FOR (see chapter 6.3 of programmer manual).



*Begin*

*for x:=1 to 10 do begin*

*end;*

*//WRITE HERE YOUR CODE*

*End.*

8. We have created the FOR structure that will be executed from x=1 until x=10; will be executed 10 times.
9. Inside FOR structure, we are going to create an IF Ahora dentro de la estructura FOR, vamos a crear un condicional IF (see chapter 5.1 of programmer manual). This IF structure will write a message when x=5.

*Begin*

*for x:=1 to 10 do begin*

*if x=5 then begin*

*debug('X is 5');*

*end;*

*end;*

*End.*

10. Now we will include a message that will be shown when the program finishes.

*Begin*

*for x:=1 to 10 do begin*

*if x=5 then begin*

*debug('X is 5');*

*end;*

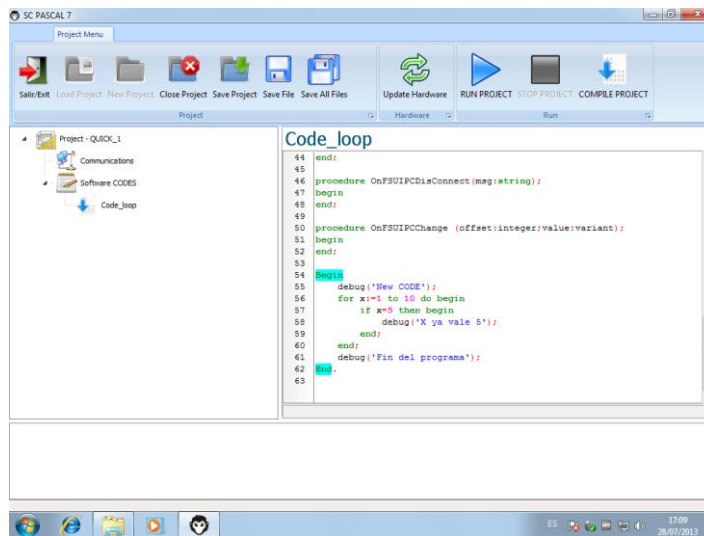
*end;*

*debug('End of program');*

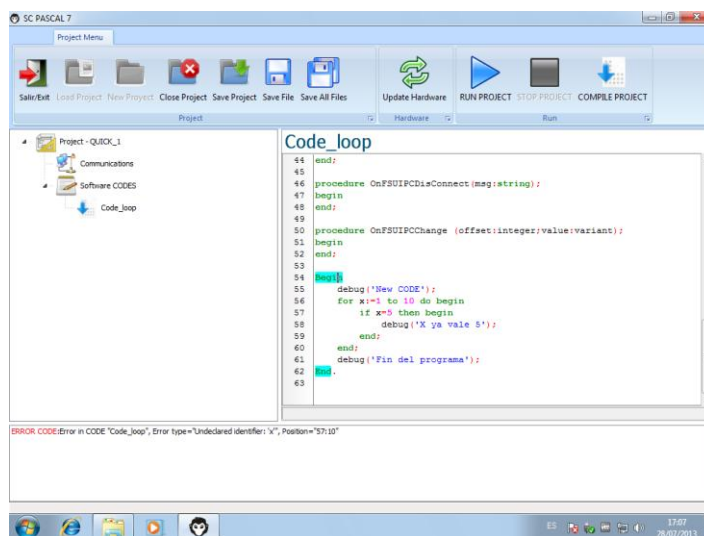
*End.*



11. Before compiling our program, we have to save it by clicking on “Save File” button on the menu area. Then we right-click on the code name and save it.



12. As you can see, there is a compilation error.

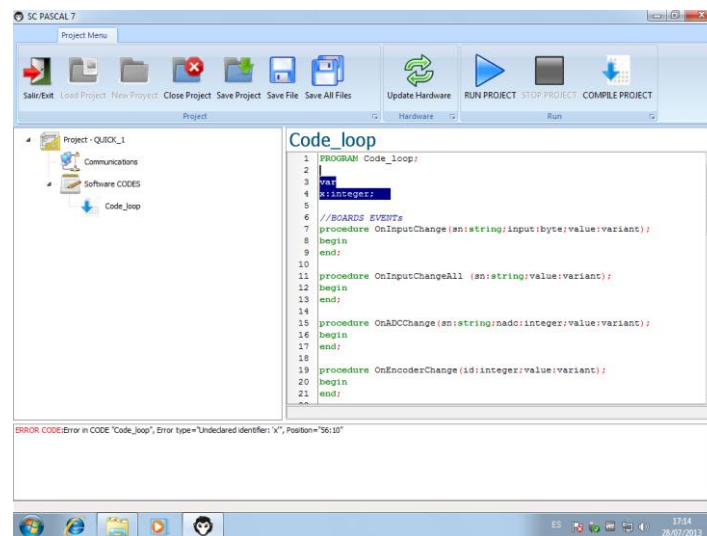


13. The error is shown in the debug window, and it's located in the line 57 (line 10 of code). We haven't declared what X is. We have to declare X just below the clause “PROGRAM”.

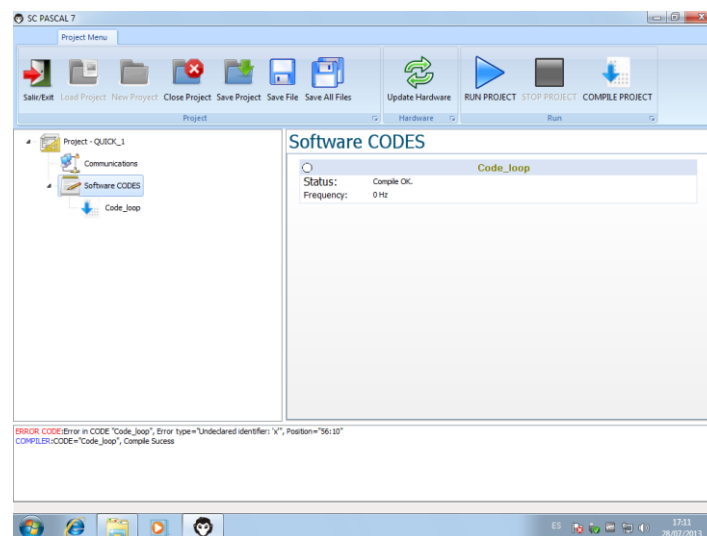
**var**

**x:integer;**

14. Once the variable is declared, we save again the file before compiling it again. At this point, is useful saving the project too. The button “Save project” is on the menu bar.

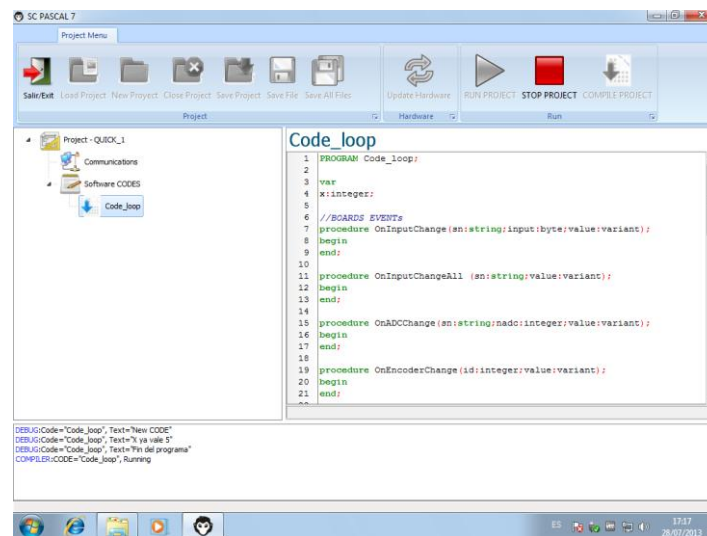


15. We compile again by right-clicking on the name of the code in the objects window, getting a “Compile OK” message now.



16. We can run our project by clicking on “RUN PROJECT” button.





17. As we can see the program has been executed successfully. Now we are going to add one more debug to the existing code.

Begin

debug('New CODE');

for x:=1 to 10 do begin

if x=5 then begin

debug('X is 5');

end else begin

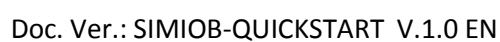
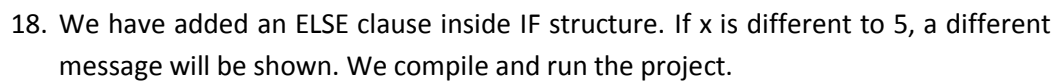
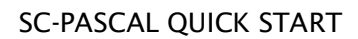
debug('X is not 5 but '+inttostr(x));

end;

end;

debug('End of program');

End.

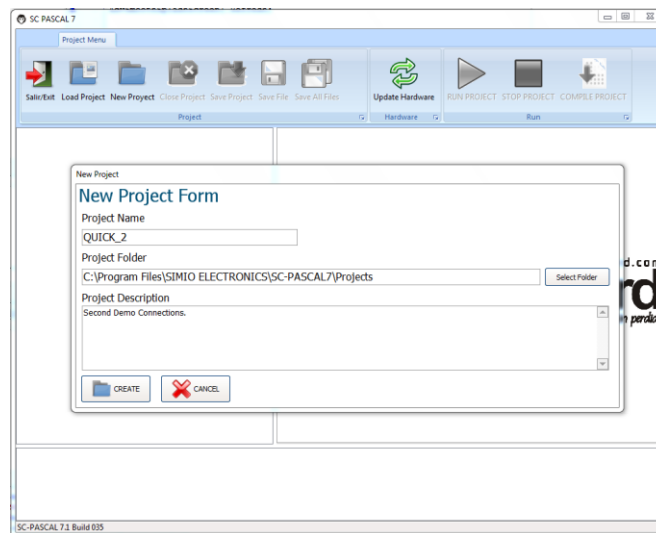




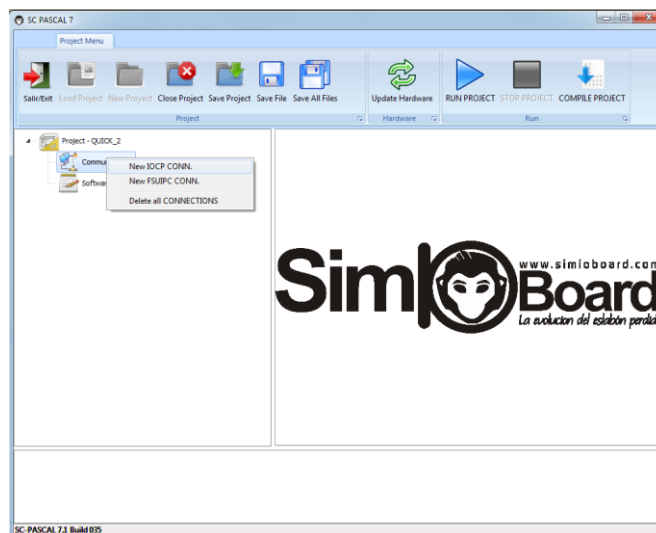
## 4. Connection example

Now we are going to make a connection with FSX using FSUIPC and IOCP to get an offset value. Firstly we will make two separate connections, and then the same example with different connections.

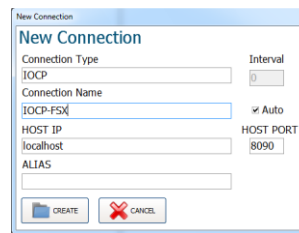
1. Open SC-PASCAL7 and start a new Project.



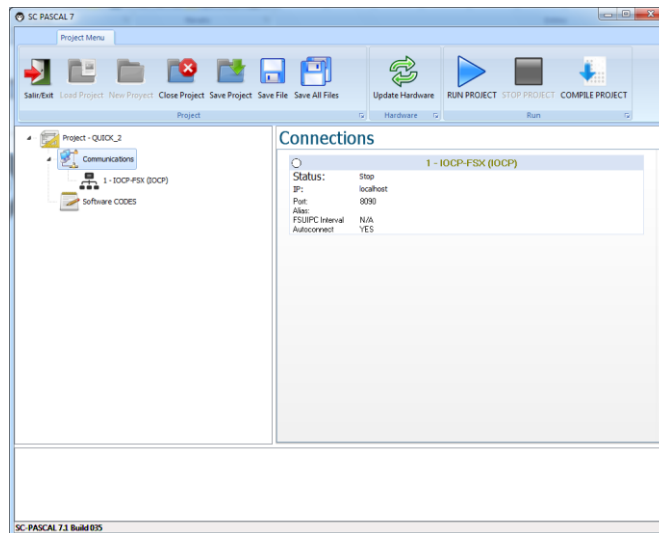
2. Right-click on Communications label on Objects window and select "New IOCP Conn".



3. Write the connection parameters. In this case the name of the connection is "IOCP-FSX" and the "HOST IP" is localhost (because FSX is running in the same machine as SC-PASCAL7).



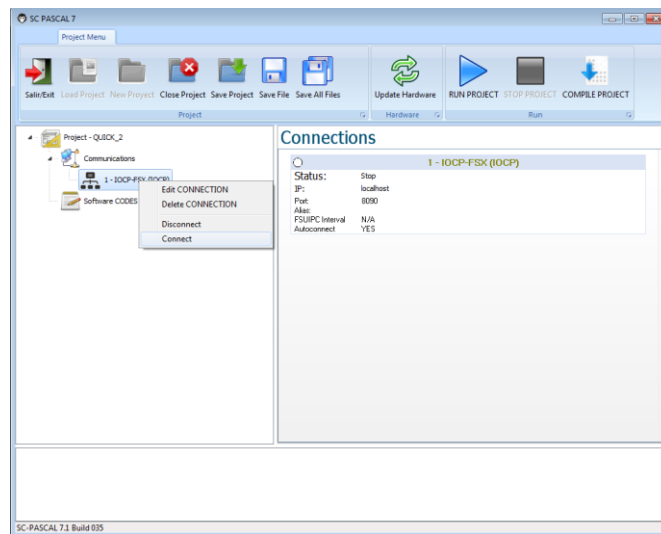
4. Click on “CREATE” and the connection is created.



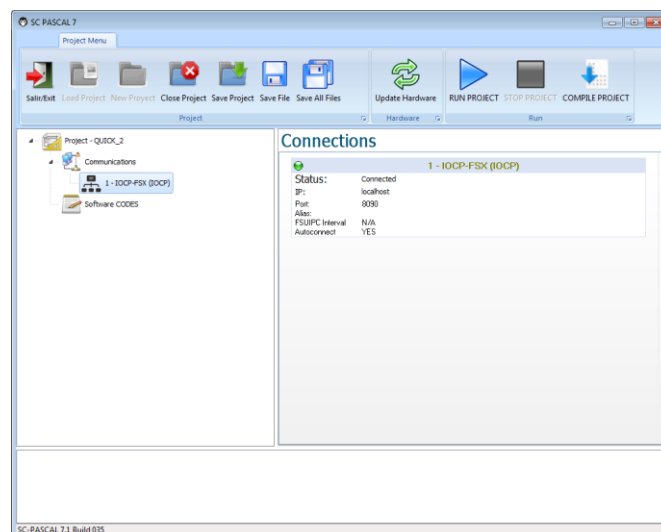
5. Now we start FSX.



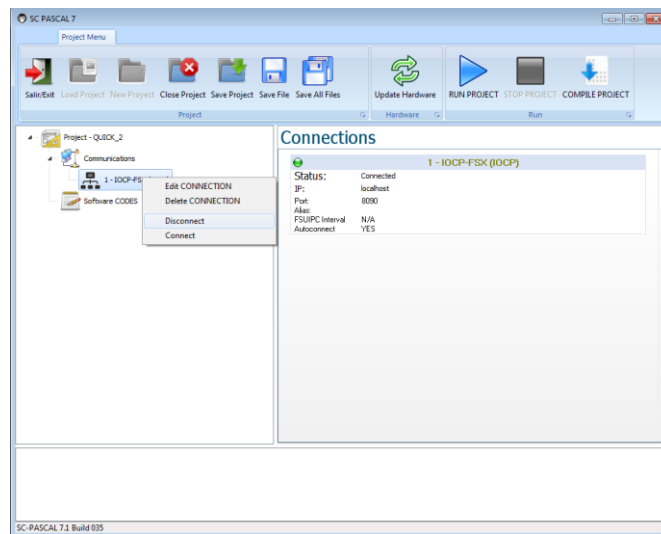
6. Now right-click on the connection name label on the Objects window and choose “Connect”.



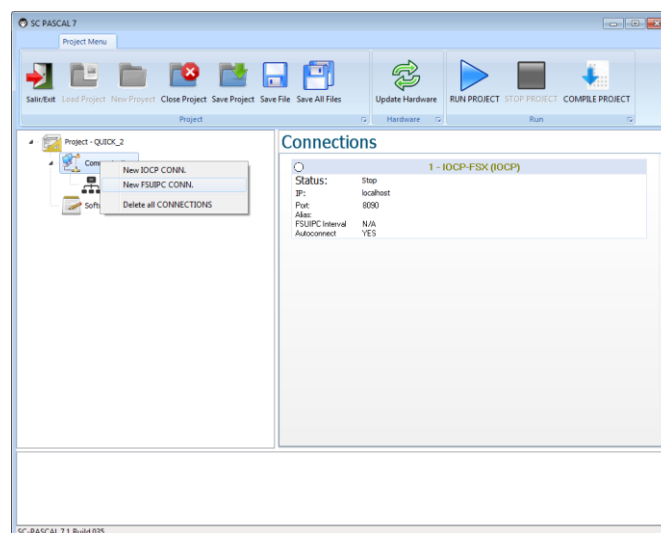
7. We can check that the connection is established on the right window.



8. The connection icon is green and the "STATUS" is "Connected", so everything is running fine.
9. Now we disconnect from IOCP by right-clicking on the connection name and clicking on "Disconnect".



10. Now we are going to make a connection with FSUIPC. Again right-clicking on “Communication” we choose “New FSUIPC Conn.”

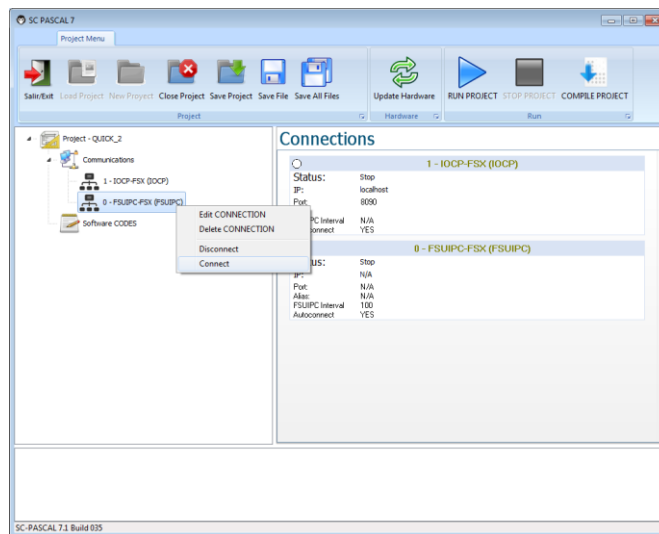


11. We fill in the connection parameters: the name and the interval (we will try with 100 milliseconds).

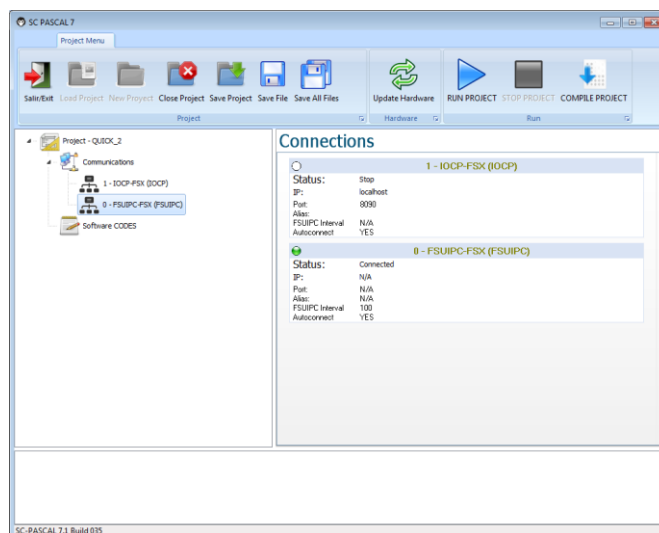
12. Click on “CREATE” and on “Communications” to check that both connection are created now. If FSUIPC is installed in our FSX, we can connect with it.



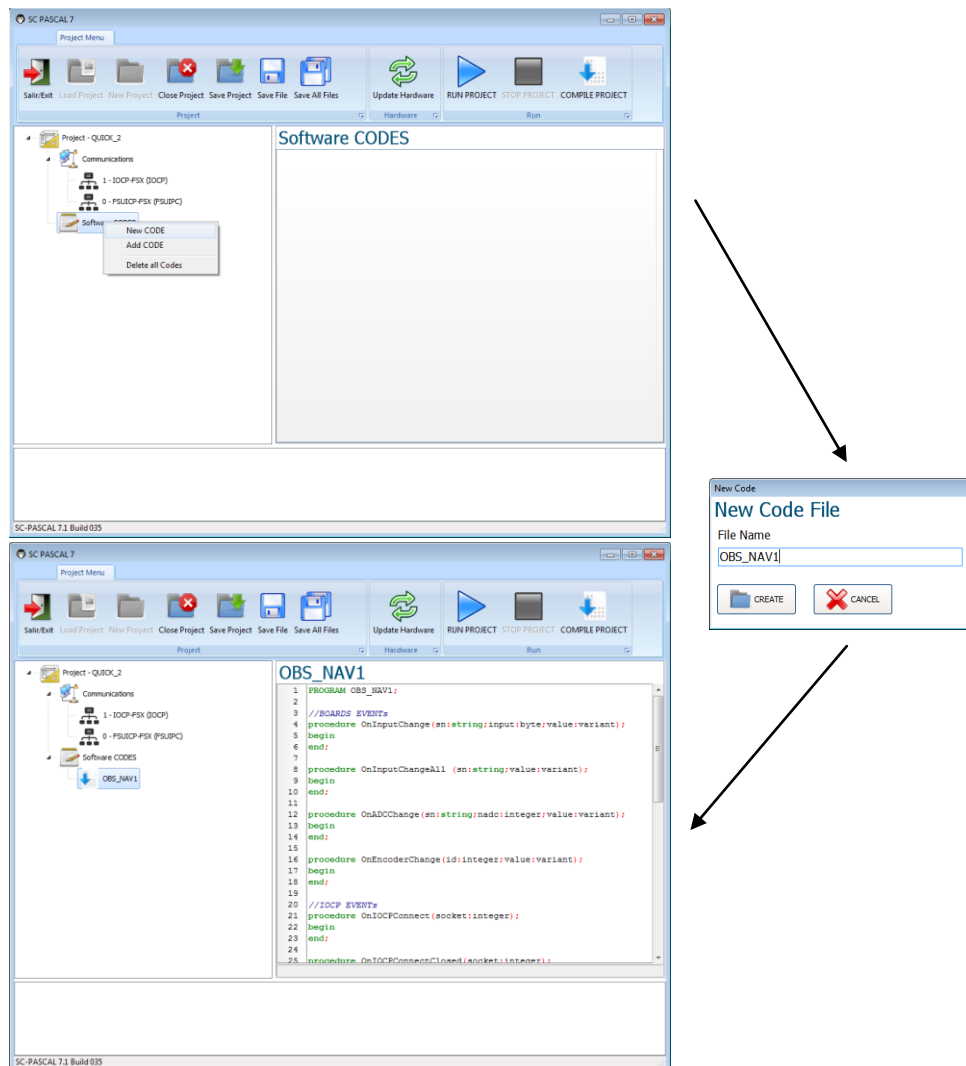
13. Right-click on the FSUIPC label and choose “Connect”.



14. The icon changes to green and “STATUS” changes to “Connected”.



15. Now disconnect by right-clicking on FSUIPC label and choosing “Disconnect”. We have already checked both connections and it’s time to read an offset using both protocols. For this example we will read OBS\_NAV1. For IOCP is offset 43, and for FSUIPC is offset \$0C4E. We know that the values for this offset is from 0 to 359, or from 1 to 360.
16. Create a new code, and name it “OBS\_NAV1”.



17. We have to register offsets. Firstly the one for IOCP (see point 7.3 from SC-PASCAL7 programmers manual).

```
RegIOCPOffset(1,43);
```

18. We have to write the previous line in the event OnIOCPConnect because, once connected, we have to inform which offsets must to be sent when they change. We have changed “1” by “socket” because it already has the connection identifier.

```
//IOCP EVENTS
```

```
procedure OnIOCPConnect(socket:integer);
```

```
begin
```

```
    RegIOCPOffset(socket,43);
```

```
end;
```

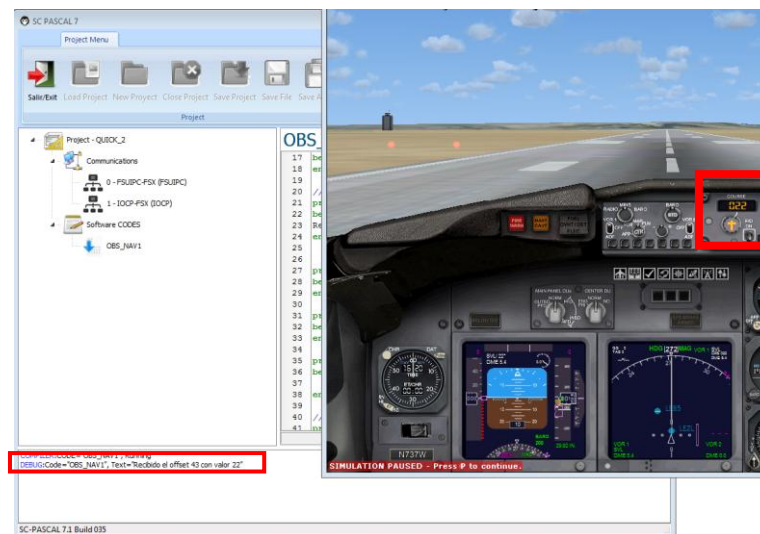




19. Once the offset is registered, we expect that if that offset changes in FSX, SC-PASCAL7 must receive this new value. This happens in the event “OnIOCPChange”, so we will include a debug in that event to check which value SC-PASCAL is receiving.

```
procedure OnIOCPChange(socket,offset,value:integer);  
  
begin  
  
    debug('Offset '+inttostr(offset)+' received with value '+inttostr(value));  
  
end;
```

20. Summary before compiling: We want that FSX sends us the value of OBS\_NAV1. So we have registered offset 43 in the event OnIOCPConnect. When this value changes on FSX, the event OnIOCPChange will be executed by SC-PASCAL7. With only two steps we get the desired value. Now we compile and run the project:



21. As we can see on the debug window, we have received 22 degrees from FSX. If we change the course on FSX, the shown value on the debug window will change too.



22. We can use this value to be sent to a displays card for example.
23. Now we are going to do the same but with FSUIPC. In this case is \$0C4E (2 bytes lenght). We have to register this offset when the connection is done, and use InitOffsetFSUIPC (see point 7.4 from SC-PASCAL7 programmer manual). The event is OnFSUIPCConnect.

```
procedure OnFSUIPCConnect (msg:string);
```

```
begin
```

```
    InitOffsetFSUIPC($0C4E,2)
```

```
end;
```

24. Now we show the value on the debug window.

```
procedure OnFSUIPCChange (offset:integer;value:variant);
```

```
begin
```

```
    debug('Received offset FSUIPC '+inttostr(offset)+' with value '+inttostr(value));
```

```
end;
```

25. To make clear which is each value, we have written the name of the connection in the debug message. Now we compile and run the project.

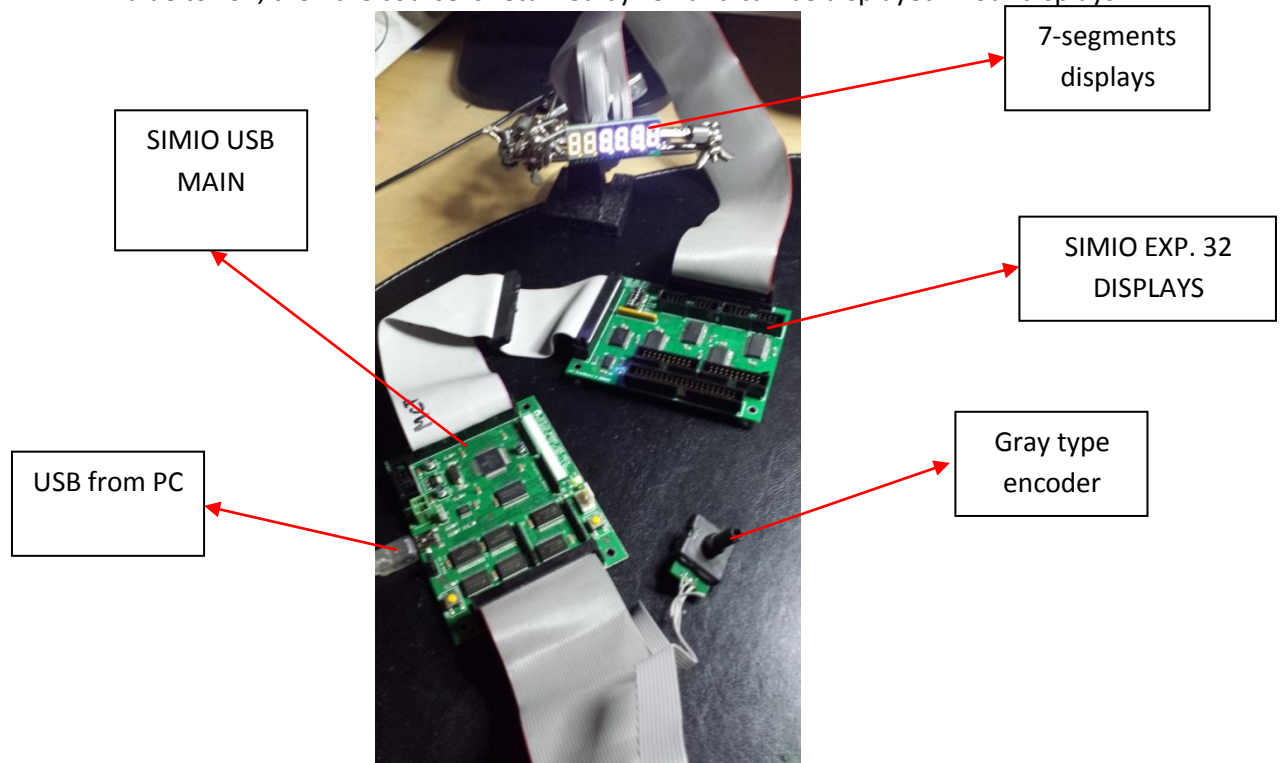


26. When we change the course in FSX, we get the new value from two different connections.
27. We are ready to go to example number three, where we will use hardware.



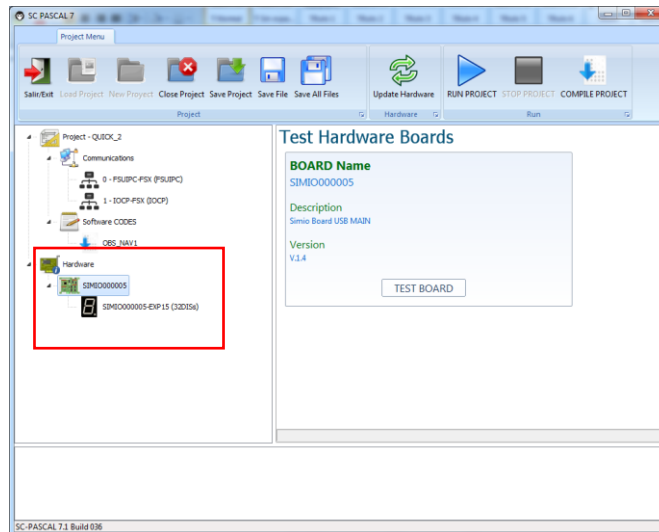
## 5. Connections and hardware example.

For this example we are going to use the same as before but adding new code for hardware. We are going to use one SIMIO USB MAIN card (SIMIO000005 in this example) and one SIMIO BOARD 32 DISPLAYS card (SIMIO000005-EXP15 in this case). We will connect one encoder to MAIN card, and 6 7-segments displays to Displays card (we only will use the first three figures). When we move the encoder we send the new value to FSX; then the course is returned by FSX and can be displayed in our displays.

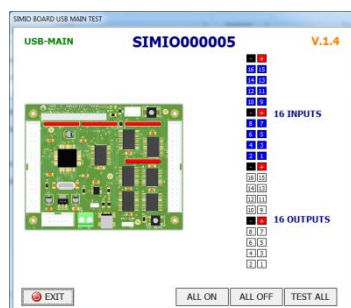




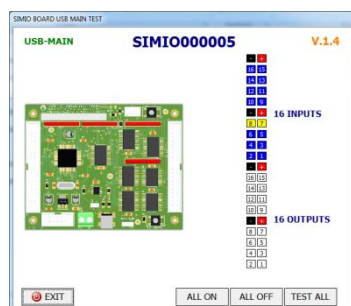
1. With the previous elements already connected to our computer, we can open the previous example. The connected hardware will be shown in the Objects area.



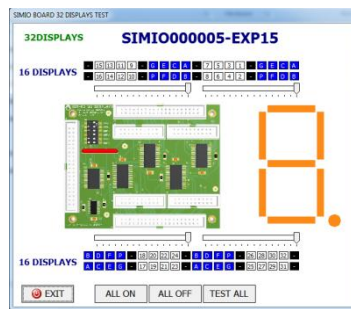
2. Click on Main SIMIO000005 and in the working area click on “TEST BOARD”.



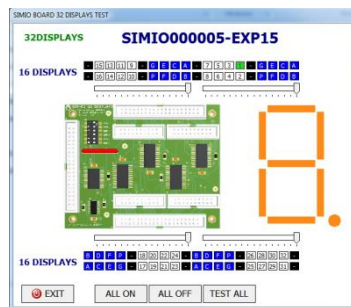
3. Now move the encoder to check to which inputs it is connected.



4. Inputs number 7 and 8 have changed.
5. We can close the test window.
6. Now we can click on the displays card “SIMIO000005-EXP15”. Click on “TEST BOARD”.



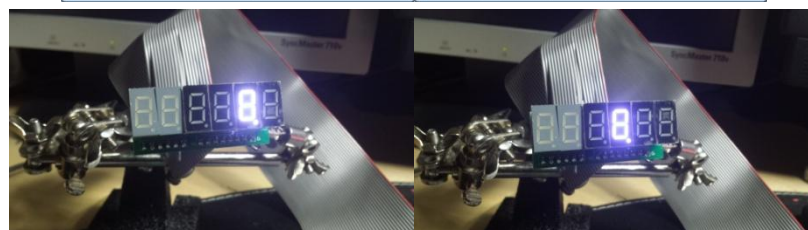
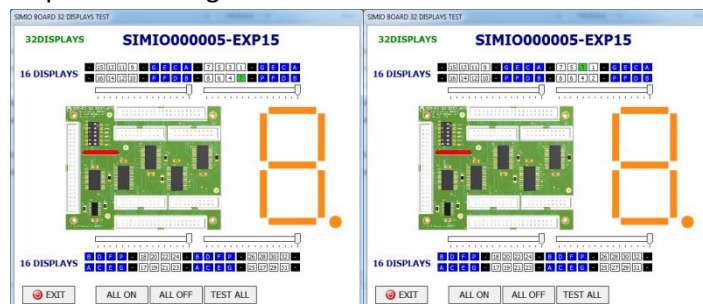
7. Now we can test every display by clicking on Digit 1.



8. It lights up.



9. Now the same process for digit 2 and 3.

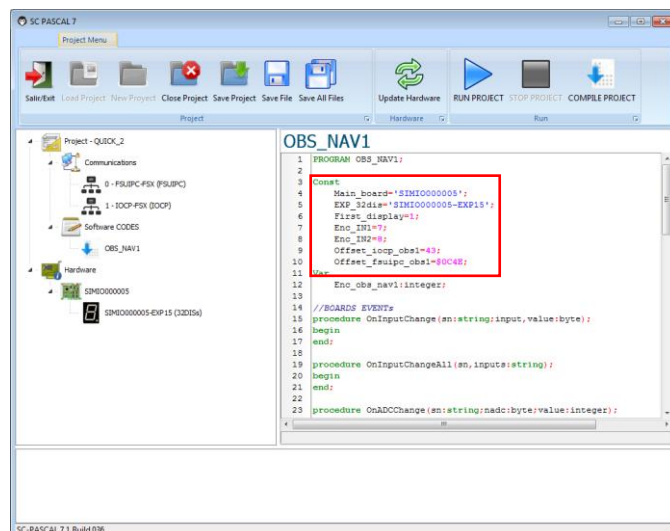




10. Everything is checked now. We can start writing the new part of the code.
11. Summary of the hardware we have to control:
  - a. 1 SIMIO USB MAIN card: SIMIO000005.
  - b. 1 SIMIO EXP. 32 DISPLAYS card: SIMIO000005-EXP15.
  - c. 3 Displays connected to displays card (starting from display no. 1).
  - d. 1 Encoder connected to inputs number 7 and 8 at MAIN Card.
12. We start by defining the hardware with the clause CONST.

### Const

```
Main_board='SIMIO000005';  
  
EXP_32dis='SIMIO000005-EXP15';  
  
First_display=1;  
  
Enc_IN1=7;  
  
Enc_IN2=8;  
  
Offset_iocp_obs1=43;  
  
Offset_fsuipt_obs1=$0C4E;
```

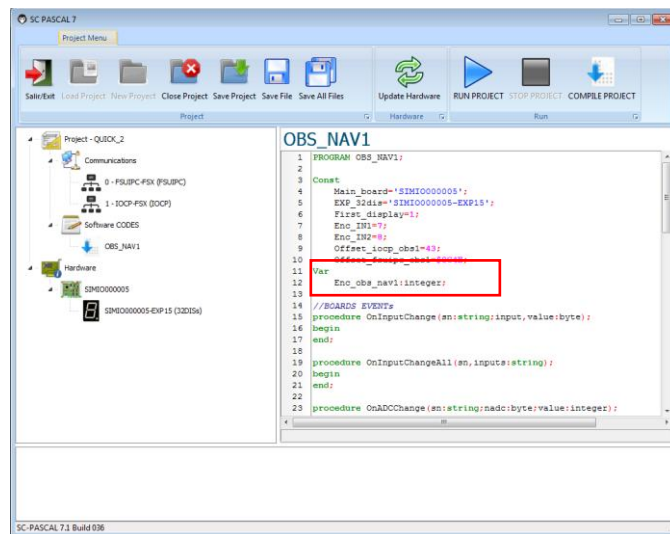


13. We have to define a global variable in order to store there the value for the encoder.

### Var

```
Enc_obs_nav1:integer;
```





14. In the MAIN part of the code, we define the encoder (see point 7.7 from SC-PASCAL7 programmer manual).

Begin

```
Enc_obs_nav1:=RegEncoder(Main_board, Enc_IN1, Enc_IN2,0,360,1,0,false,100);
```

End.

15. Now the encoder is registered. When we move it, the event OnEncoderChange will notify this change. The value given by this event will be sent to FSX. We are using IOCP; then we will use the event WriteIOCP to do this (see point 7.7 from SC-PASCAL7 programmer manual).

```
procedure OnEncoderChange(id_encoder:integer;value:integer);
```

```
begin
```

```
if id_encoder=Enc_obs_nav1 then begin
```

```
WriteIOCP(1,Offset_iocp_obs1,value);
```

```
end;
```

```
end;
```

16. As you can see, we have used an IF clause. This makes that only the encoder for OBS\_NAV1 is the one considered in the code. This trick will be very useful when we have lots of encoders in the same program.
17. We can now compile and run the updated project. And check that when we move the encoder, the course changes on FSX.
18. The last step is make the updated course is displayed in our displays. OnIOCPChange event will receive the value OBS\_NAV1 or COURSE, and this value





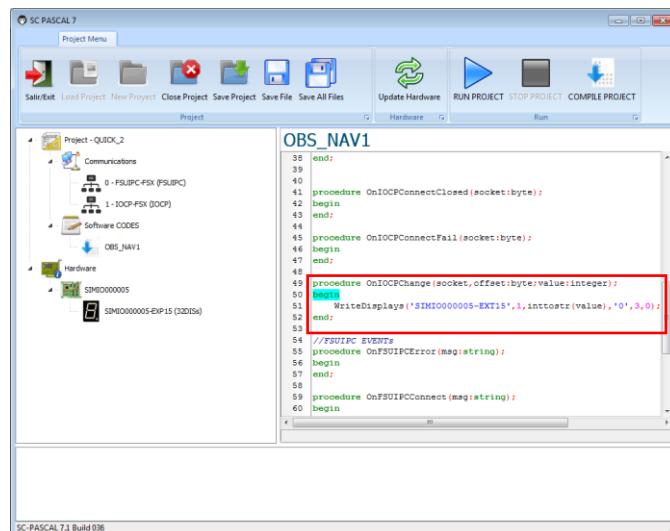
will be sent to our displays with the event WriteDisplays (see point 7.8 from SC-PASCAL7 programmer manual).

```
procedure OnIOCPChange(socket,offset:byte;value:integer);
```

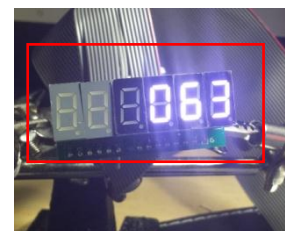
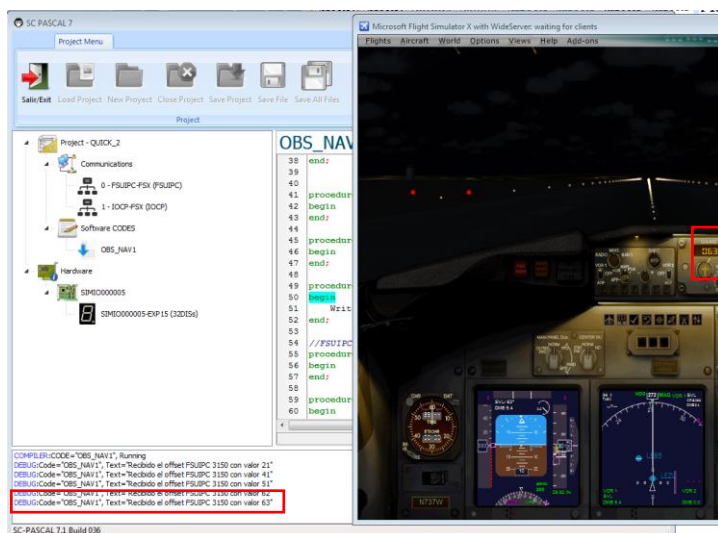
```
begin
```

```
WriteDisplays(EXP_32dis, First_display, inttostr(value),'0',3,0);
```

```
end;
```



19. We compile and run the project, checking that the course is displayed properly.



20. The whole code will remain as follows:



```
PROGRAM OBS_NAV1;
Const
  Main_board='SIMIO000005';
  EXP_32dis='SIMIO000005-EXP15';
  First_display=1;
  Enc_IN1=7;
  Enc_IN2=8;
  Offset_iocp_obs1=43;
  Offset_fsuipc_obs1=$0C4E;
Var
  Enc_obs_nav1:integer;

//BOARDS EVENTS
procedure OnEncoderChange(id_encoder,value:integer);
begin
  if id_encoder=Enc_obs_nav1 then begin
    WriteIOCP(1, Offset_iocp_obs1,value);
  end;
end;

//IOCP EVENTS
procedure OnIOCPConnect(socket:byte);
begin
  RegIOCPOffset(socket, Offset_iocp_obs1);
end;

procedure OnIOCPChange(socket,offset:byte;value:integer);
begin
  WriteDisplays(EXP_32dis, First_display,inttostr(value),'0',3,0);
end;

//MAIN
Begin
  Enc_obs_nav1:=RegEncoder(Main_board, Enc_IN1, Enc_IN2,0,360,1,0,false,100);
End.
```

21. This example has been made using IOCP offsets. The same can be done using FSUIPC ones with minor changes on the code. The whole code in this case would be:



```
PROGRAM OBS_NAV1;
Const
  Main_board='SIMIO000005';
  EXP_32dis='SIMIO000005-EXP15';
  First_display=1;
  Enc_IN1=7;
  Enc_IN2=8;
  Offset_iocp_obs1=43;
  Offset_fsuipc_obs1=$0C4E;
Var
  Enc_obs_nav1:integer;

//BOARDS EVENTS
procedure OnEncoderChange(id_encoder,value:integer);
begin
  if id_encoder=Enc_obs_nav1 then begin
    WriteFSUIPC(Offset_fsuipc_obs1,2,value);
  end;
end;

//FSUIPC EVENTS
procedure OnFSUIPCConnect(msg:string);
begin
  InitOffsetFSUIPC(Offset_fsuipc_obs1,2)
end;

procedure OnFSUIPCChange(offset:integer;value:variant);
begin
  WriteDisplays(EXP_32dis, First_display,inttostr(value),'0',3,0);
end;

//MAIN
Begin
  Enc_obs_nav1:=RegEncoder(Main_board, Enc_IN1, Enc_IN2,0,360,1,0,false,100);
End.
```

22. We can improve the code by introducing a new Pascal structure inside events OnIOCPChange and OnFSUIPCChange. We can identify what to do depending on the offset.
23. OnIOCPChange will receive any change on the registered offsets, and the previous code wouldn't work if we had registered more than one offset.



```
procedure OnIOCPChange(socket,offset:byte;value:integer);
begin
    WriteDisplays(EXP_32dis, First_display,inttostr(value),'0',3,0);
end;
```

24. In this case we will use a CASE structure, (see point 5.2 from SC-PASCAL7 programmer manual).

```
procedure OnIOCPChange(socket,offset:byte;value:integer);
begin
    case offset of
        Offset_iocp_obs1:WriteDisplays(EXP_32dis, First_display,inttostr(value),'0',3,0);
    end;
end;
```

25. The same example but for FSUIPC offsets would be:

```
procedure OnFSUIPCChange(offset:integer;value:variant);
begin
    case offset of
        Offset_fsuipc_obs1:WriteDisplays(EXP_32dis, First_display,inttostr(value),'0',3,0);
    end;
end;
```

26. For the last example we will add another offset. The value from indicated airspeed IAS will be displayed on displays 4 to 6.

Below the added code is blue colored.

```
PROGRAM OBS_NAV1;

Const
    Main_board='SIMIO000005';
    EXP_32dis='SIMIO000005-EXP15';
    First_display=1;
    Enc_IN1=7;
    Enc_IN2=8;
    Offset_iocp_obs1=43;
    Offset_fsuipc_obs1=$0C4E;
    Offset_iocp_IAS=114; //MCP HDG from IOCP
    Offset_fsuipc_IAS=$07E2; //MCP_HDG from FSUIPC
```



```
Second_display=4;
Var
  Enc_obs_nav1:integer;

//BOARDS EVENTS
procedure OnEncoderChange(id_encoder,value:integer);
begin
  if id_encoder=Enc_obs_nav1 then begin
    WriteIOCP(1,Offset_iocp_obs1,value);
    WriteIOCP(1,Offset_iocp_IAS,value);
    //WriteFSUIPC(Offset_fsuipc_obs1,2,value);
    //WriteFSUIPC(Offset_fsuipc_IAS,2,value);
  end;
end;

//IOCP EVENTS
procedure OnIOCPConnect(socket:byte);
begin
  RegIOCPOffset(socket,Offset_iocp_obs1);
  RegIOCPOffset(socket,Offset_iocp_IAS);
end;

procedure OnIOCPChange(socket,offset:byte;value:integer);
begin
  case offset of
    Offset_iocp_obs1:WriteDisplays(EXP_32dis, First_display,inttostr(value),'0',3,0);
    Offset_iocp_IAS:WriteDisplays(EXP_32dis, Second_display,inttostr(value),'0',3,0);
  end;
end;

//FSUIPC EVENTS
procedure OnFSUIPCConnect(msg:string);
begin
  InitOffsetFSUIPC(Offset_fsuipc_obs1,2);
  InitOffsetFSUIPC(Offset_fsuipc_IAS,2);
end;

procedure OnFSUIPCChange(offset:integer;value:variant);
begin
  //case offset of

  //Offset_fsuipc_obs1:WriteDisplays(EXP_32dis, First_display,inttostr(value),'0',3,0);
```



```
//Offset_fsuipc_IAS:WriteDisplays(EXP_32dis, Second_display,inttostr(value),'0',3,0);  
  
//end;  
end;  
  
//MAIN  
Begin  
  Enc_obs_nav1:=RegEncoder(Main_board, Enc_IN1, Enc_IN2,0,360,1,0,false,100);  
End.
```