# 

Eyetracker Classification of subjects with Parkinson’s using Machine Learning based on FPGA

PROYECTO DE DISEÑO DE SISTEMAS DIGITALES

**Profesor:** Ing. Victor Asanza

**Grupo # 2**

**Elaborado por:**

* Viviana Vásquez Briones
* Jonathan Yagual Peredo

# Arquitectura de Hardware y Diagrama de flujo del procesador.

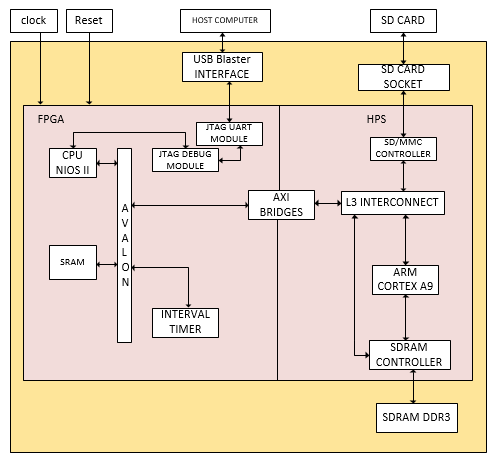


Figura 1. Arquitectura de Hardware del sistema.

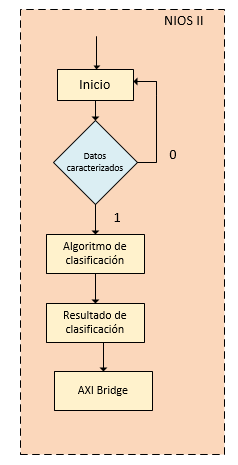
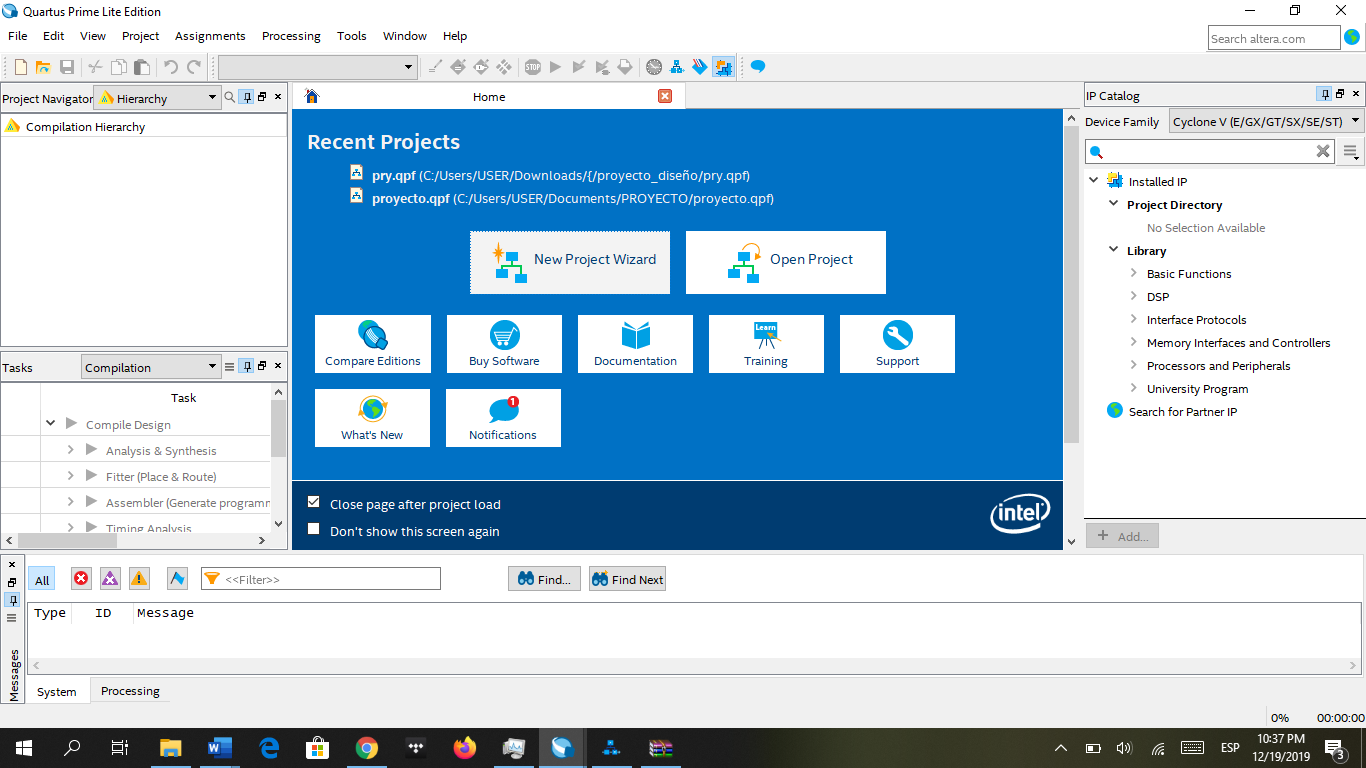


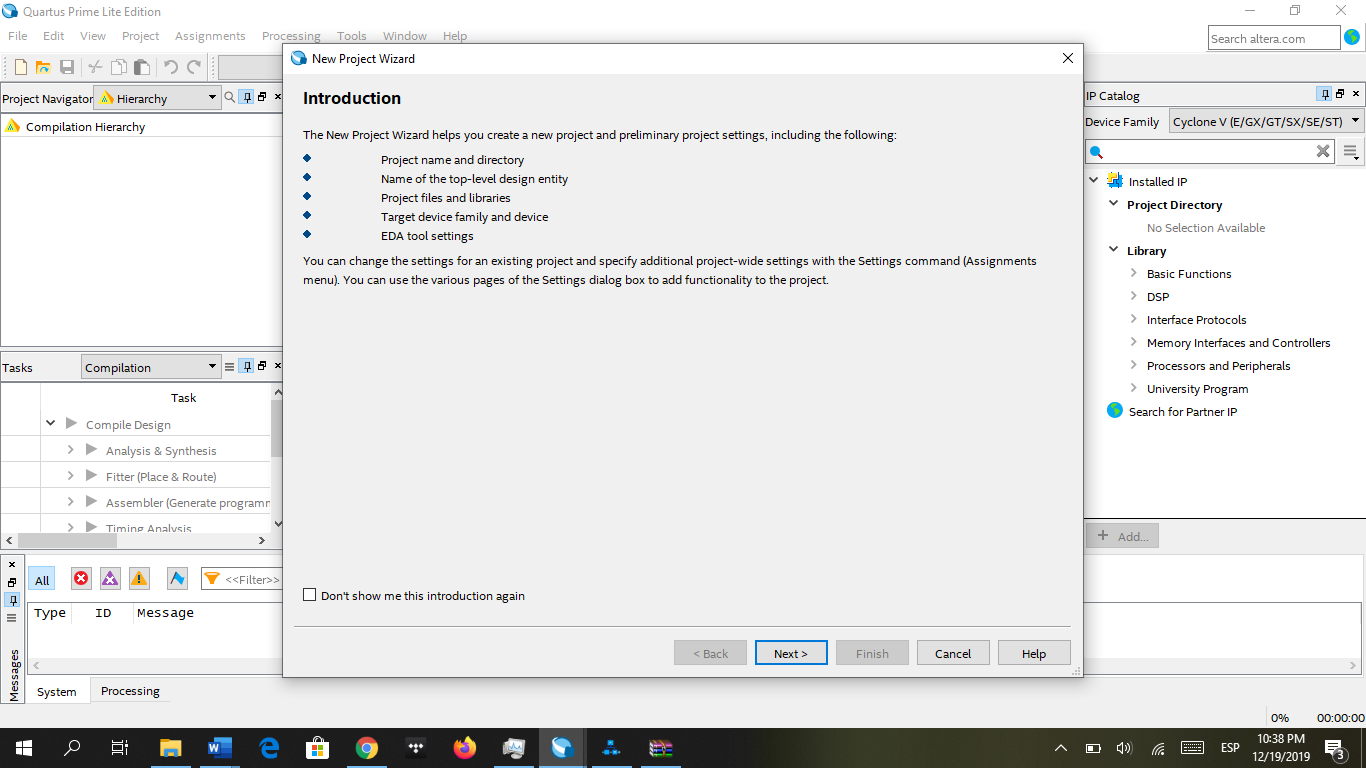
Figura 2. Diagrama de flujo del procesador.

# Diseño de la arquitectura de hardware en Quartus – Platform Designer

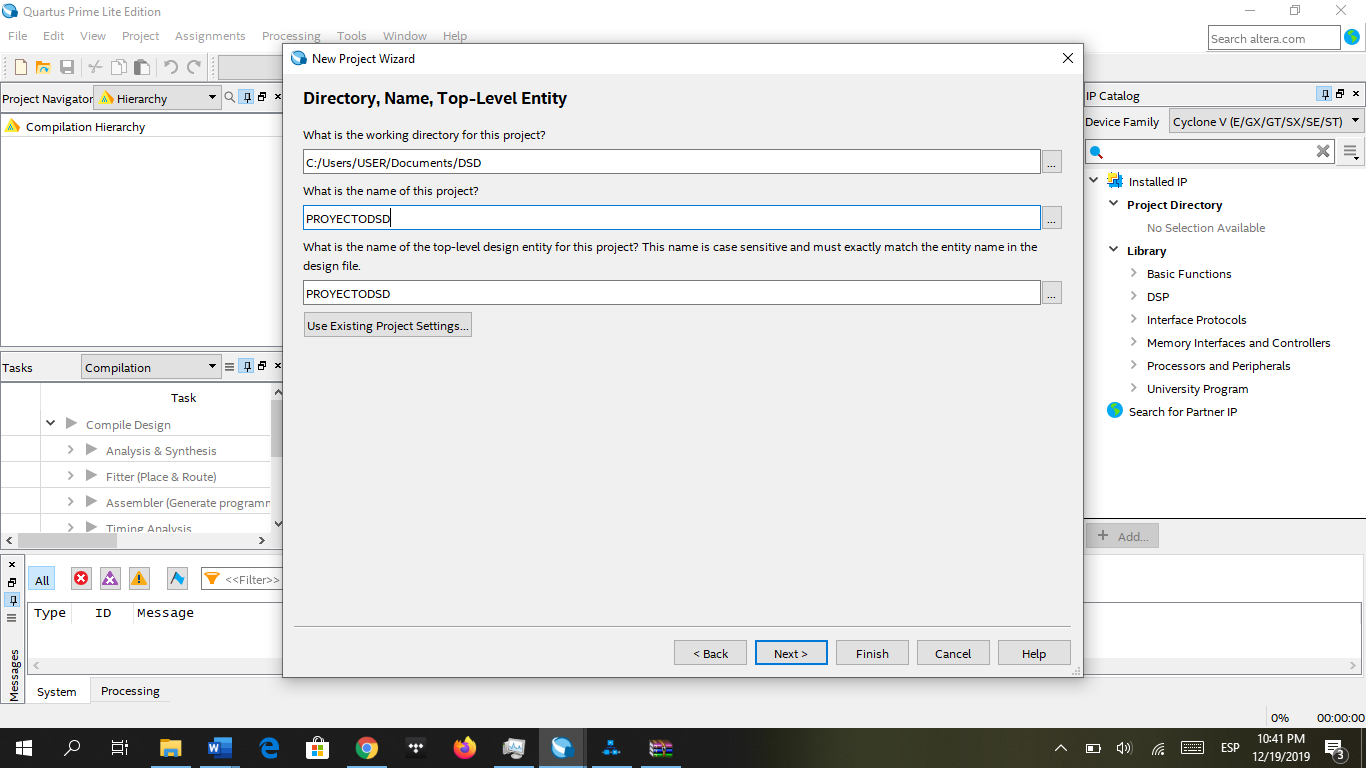
1. Abrir Quartus y crear un nuevo proyecto (New Project Wizard).



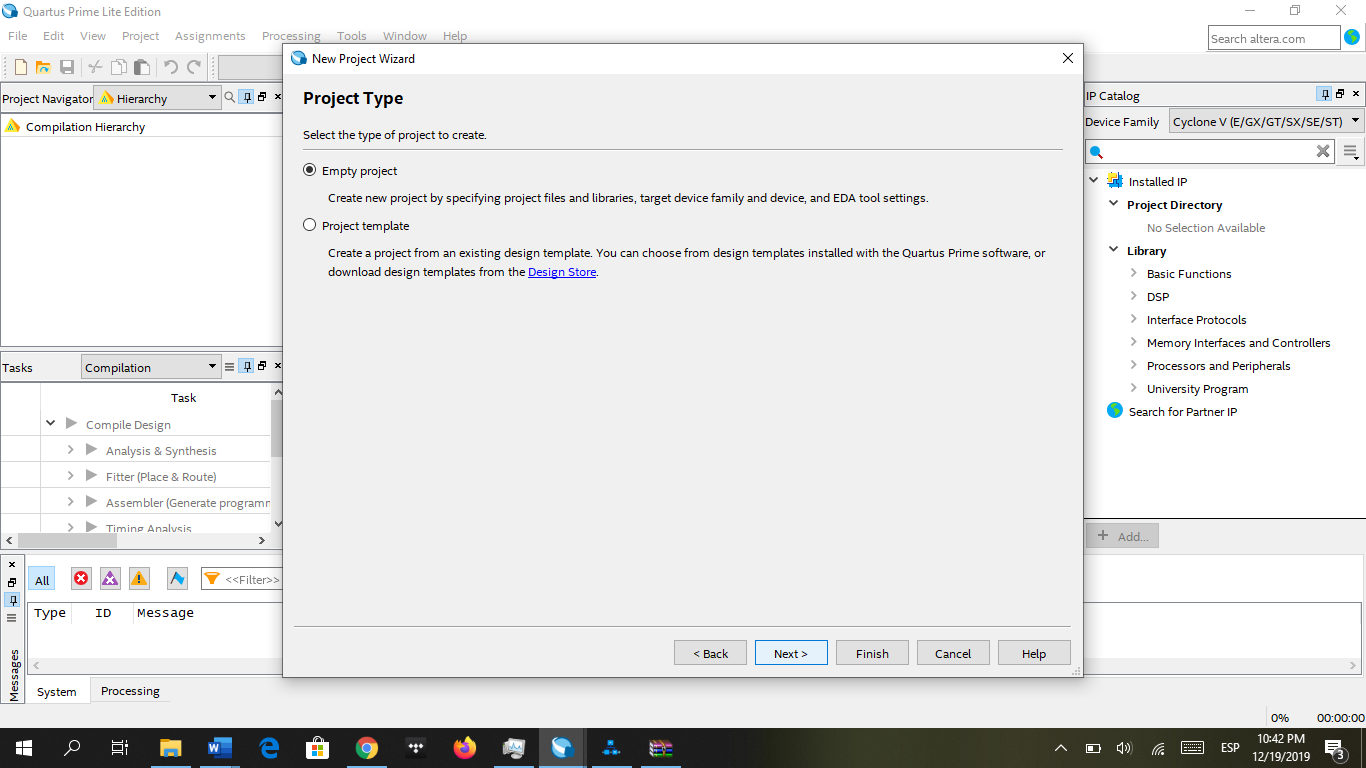
1. Clic en **NEXT.**



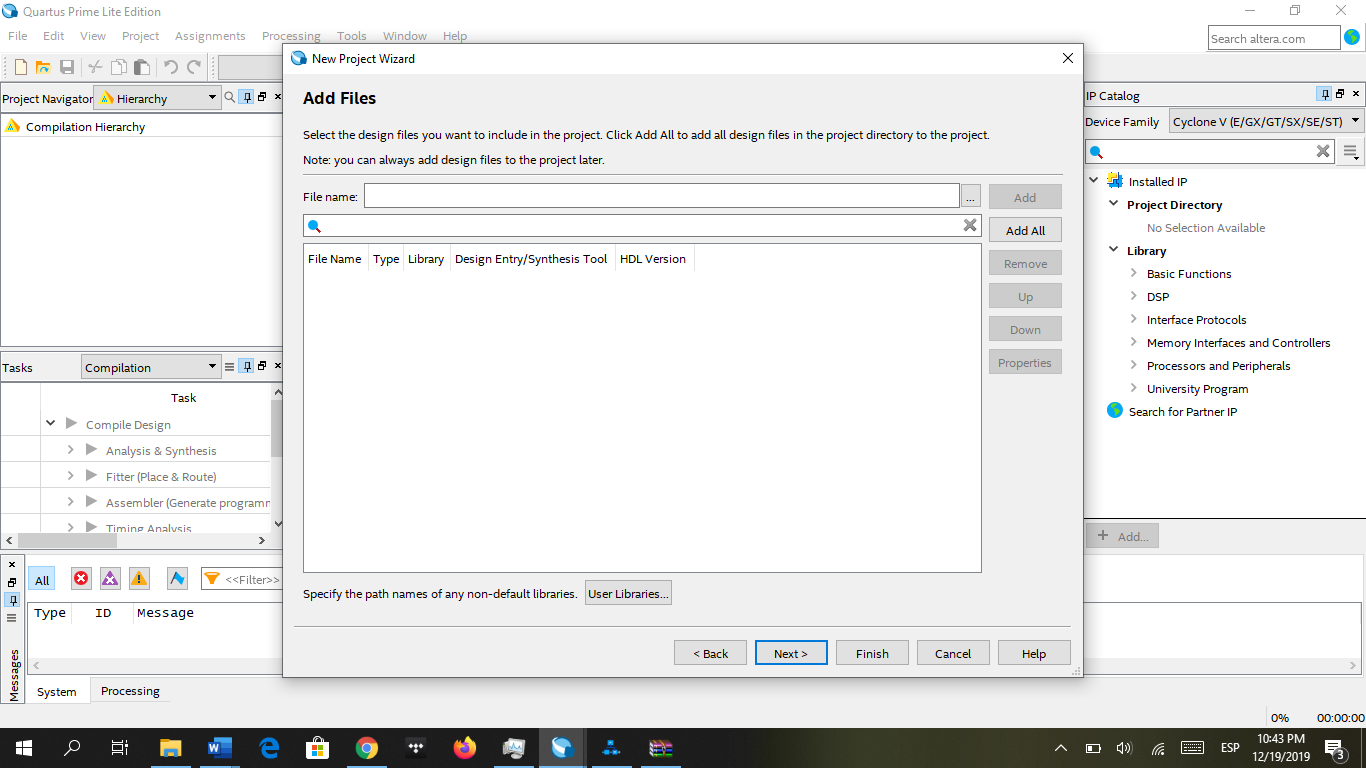
1. Seleccione como directorio una carpeta vacía, en un directorio accesible y nombre el proyecto como desee.



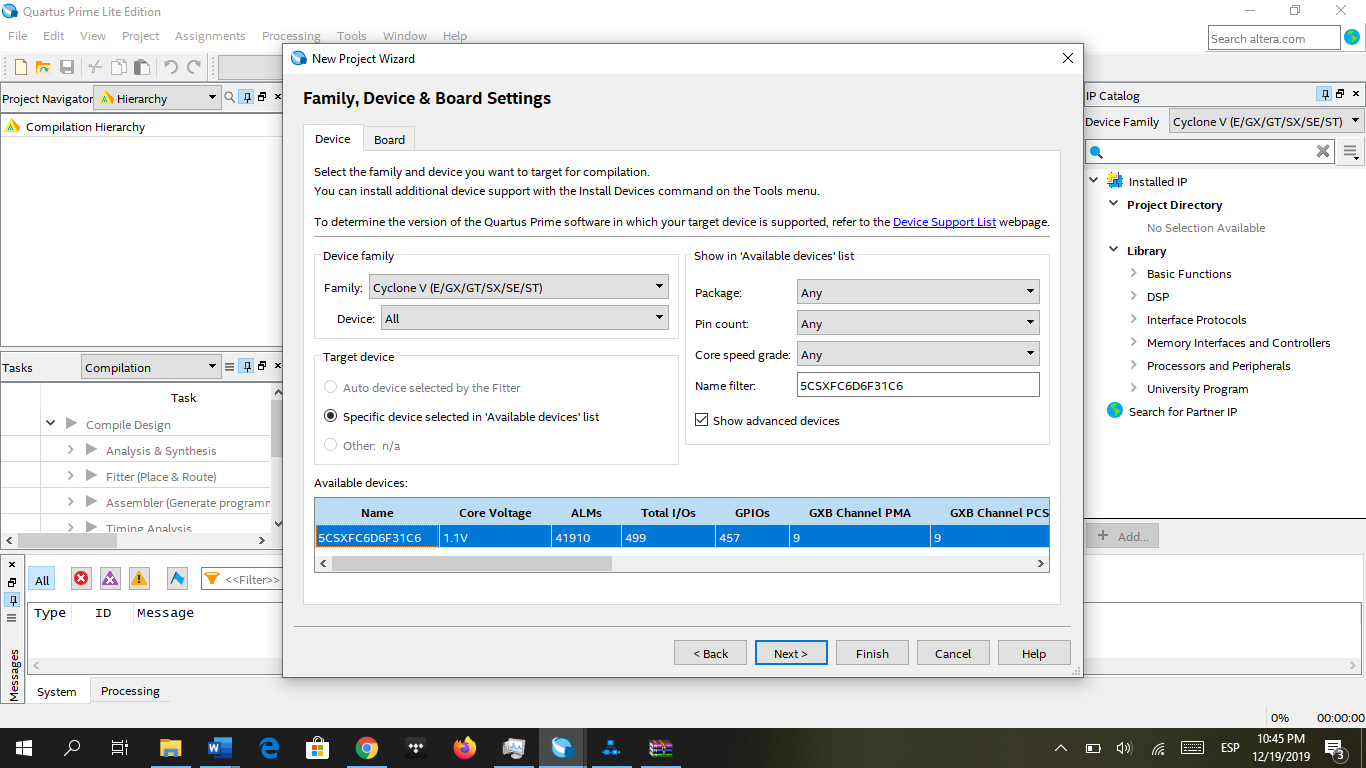
1. Seleccione **EMPTY PROJECT.**



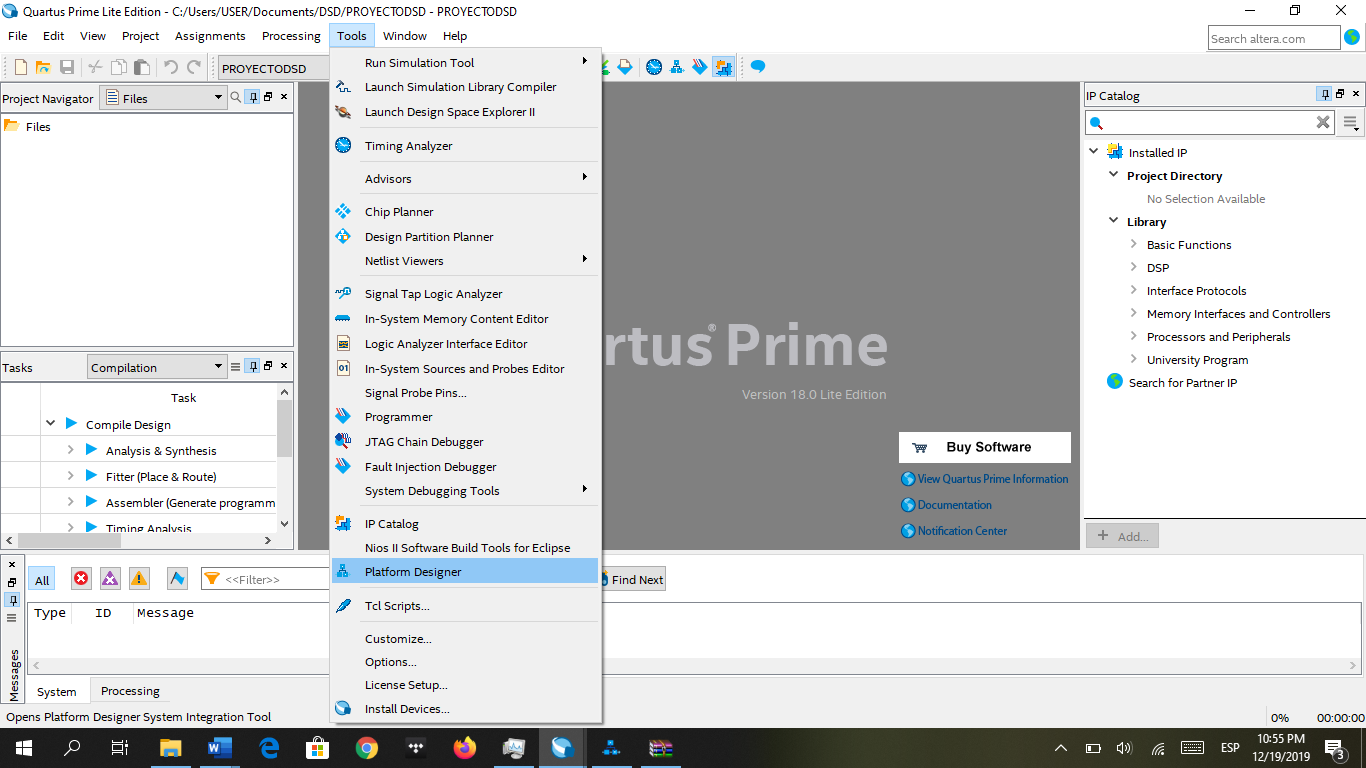
1. No se cargará ningún archivo, seleccione **NEXT**.



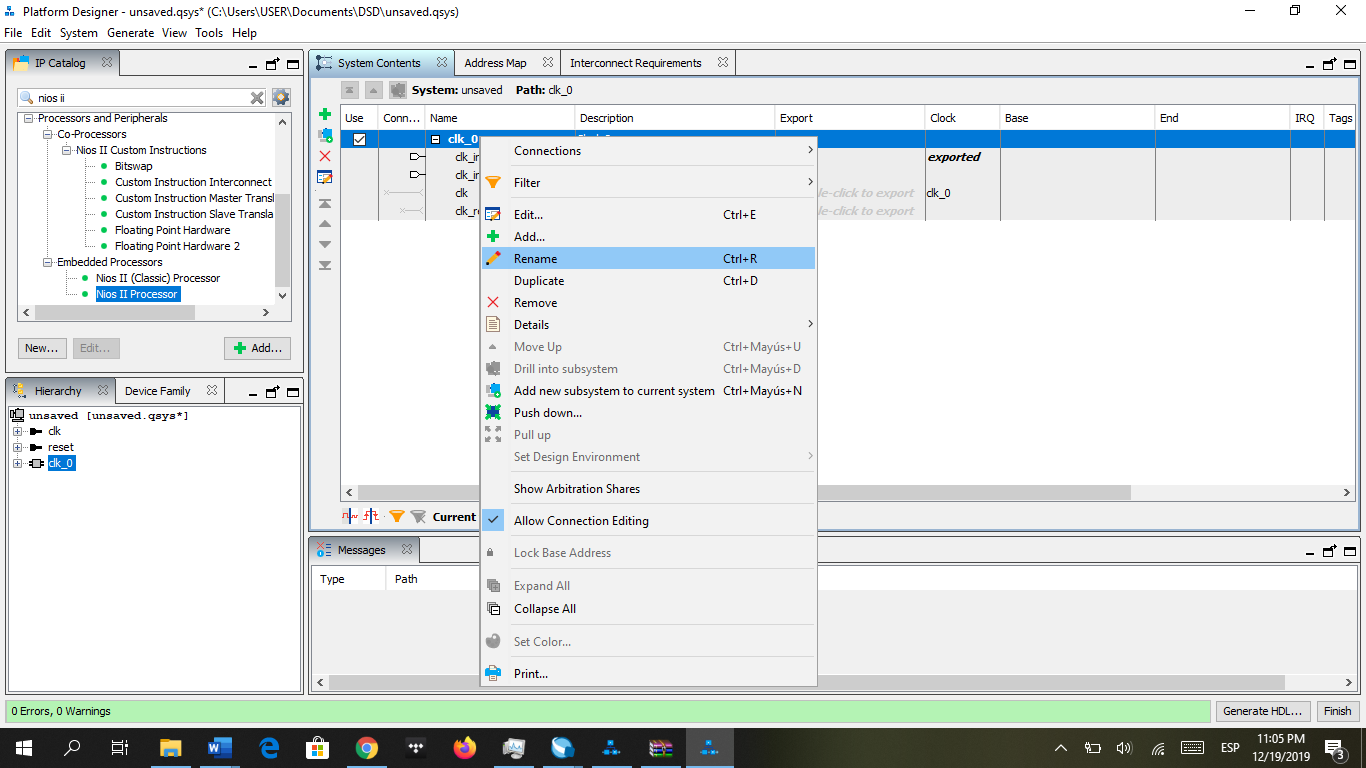
1. Seleccione la FPGA SoC 5CSXFC6D6F31C6 de la familia Cyclone V.



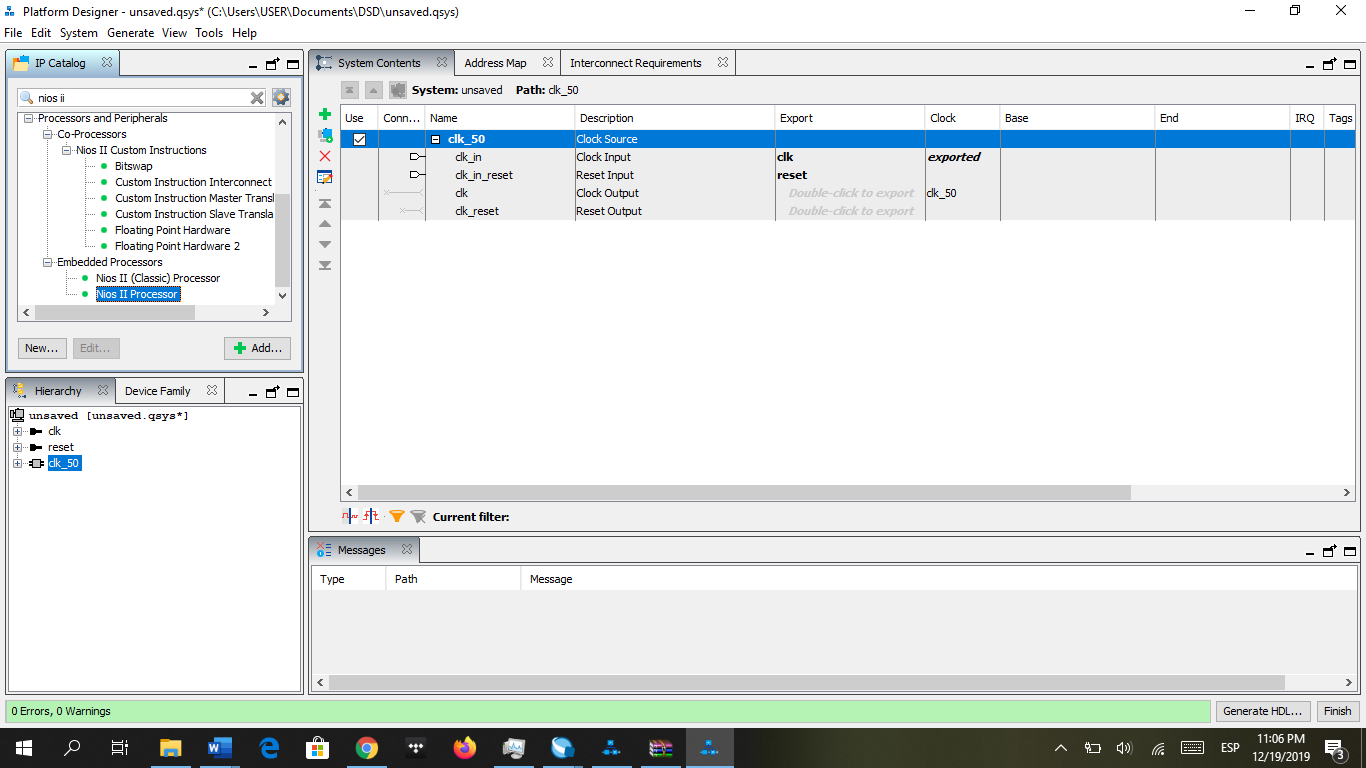
1. Por último, seleccione **NEXT** y **FINISH,** para terminar de crear el proyecto.
2. Abrir el Platform Designer (Qsys) accediendo al menú **Tools**.



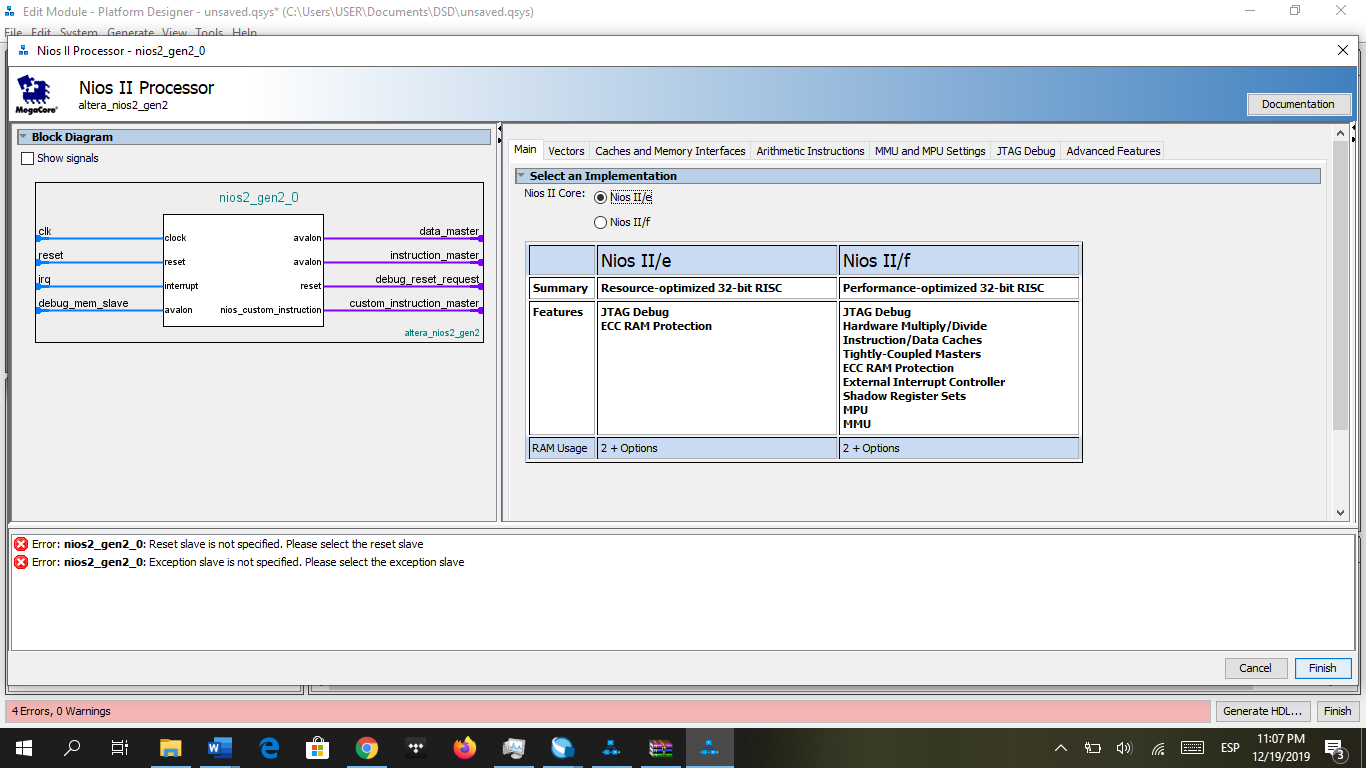
1. Aparecerá un componente que permitirá conectar señales externas de reloj hacia los demás elementos del Qsys. Renombrar CLK\_0 por CLK\_50.



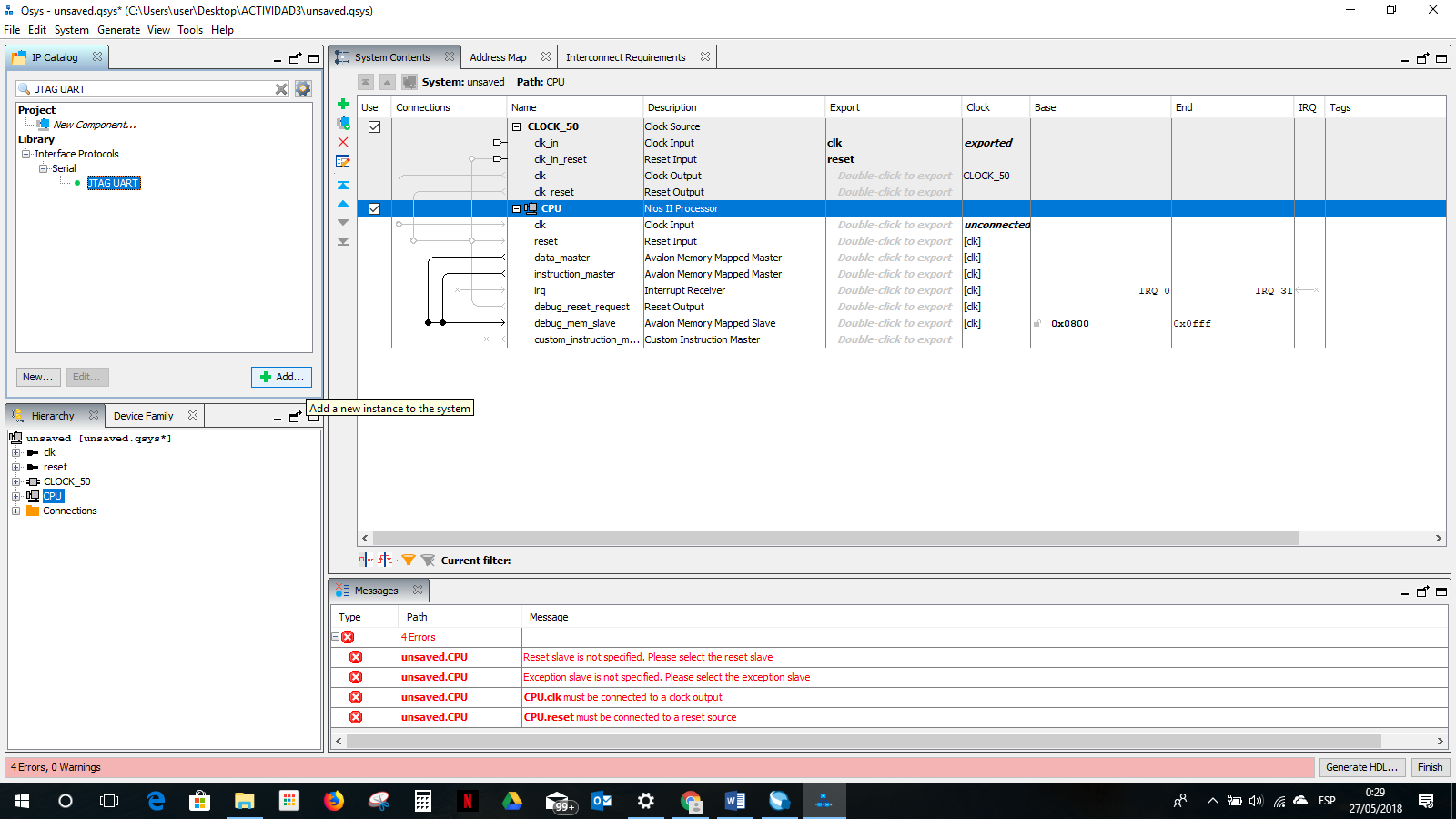
1. En la ventana izquierda **IP Catalog**, buscar **NIOS II Processor**, selecciónelo dando doble clic.



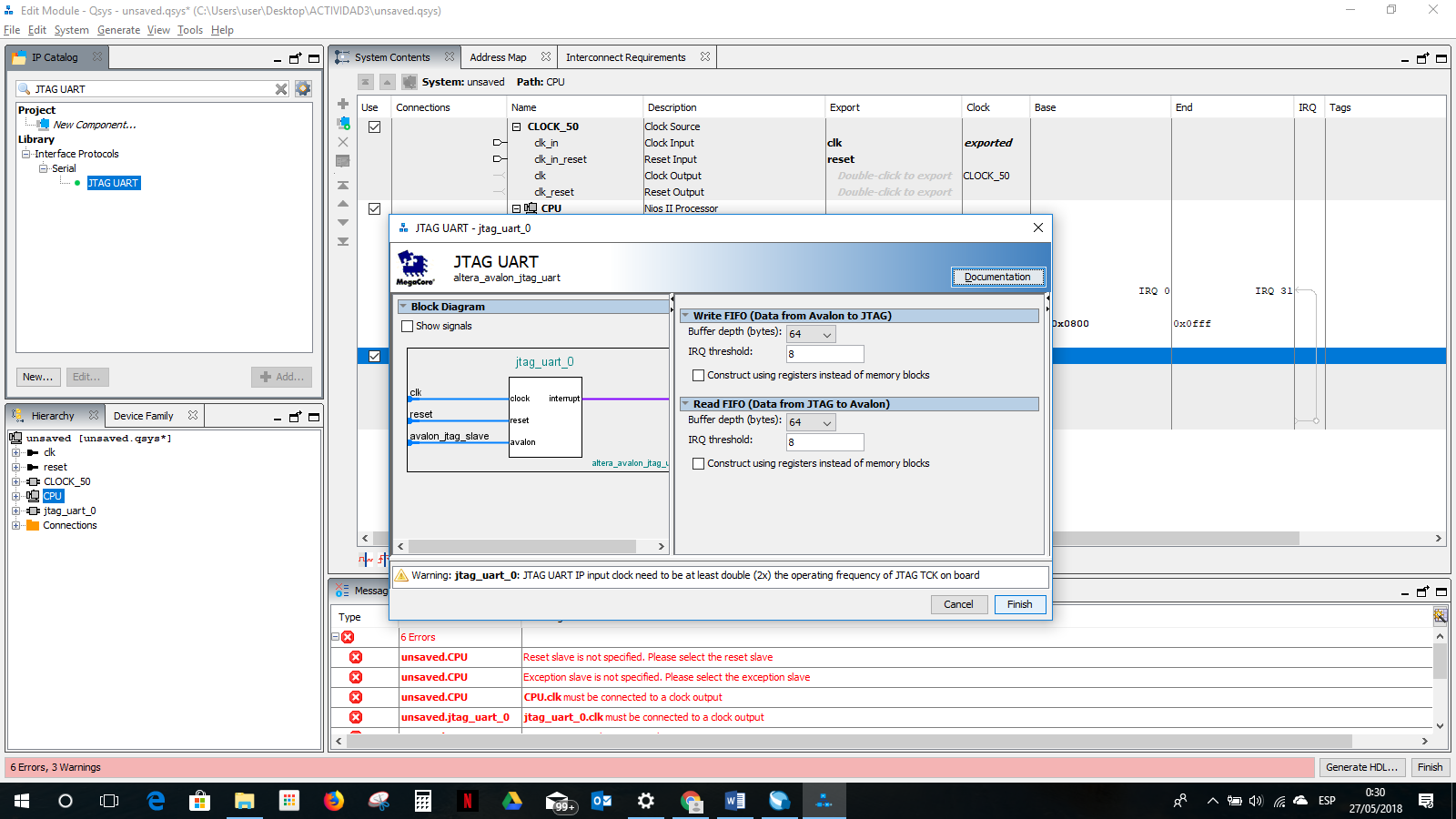
1. Seleccione **NIOS II/e** y de clic en **Finish**. A continuación, renombre el componente como **CPU**.



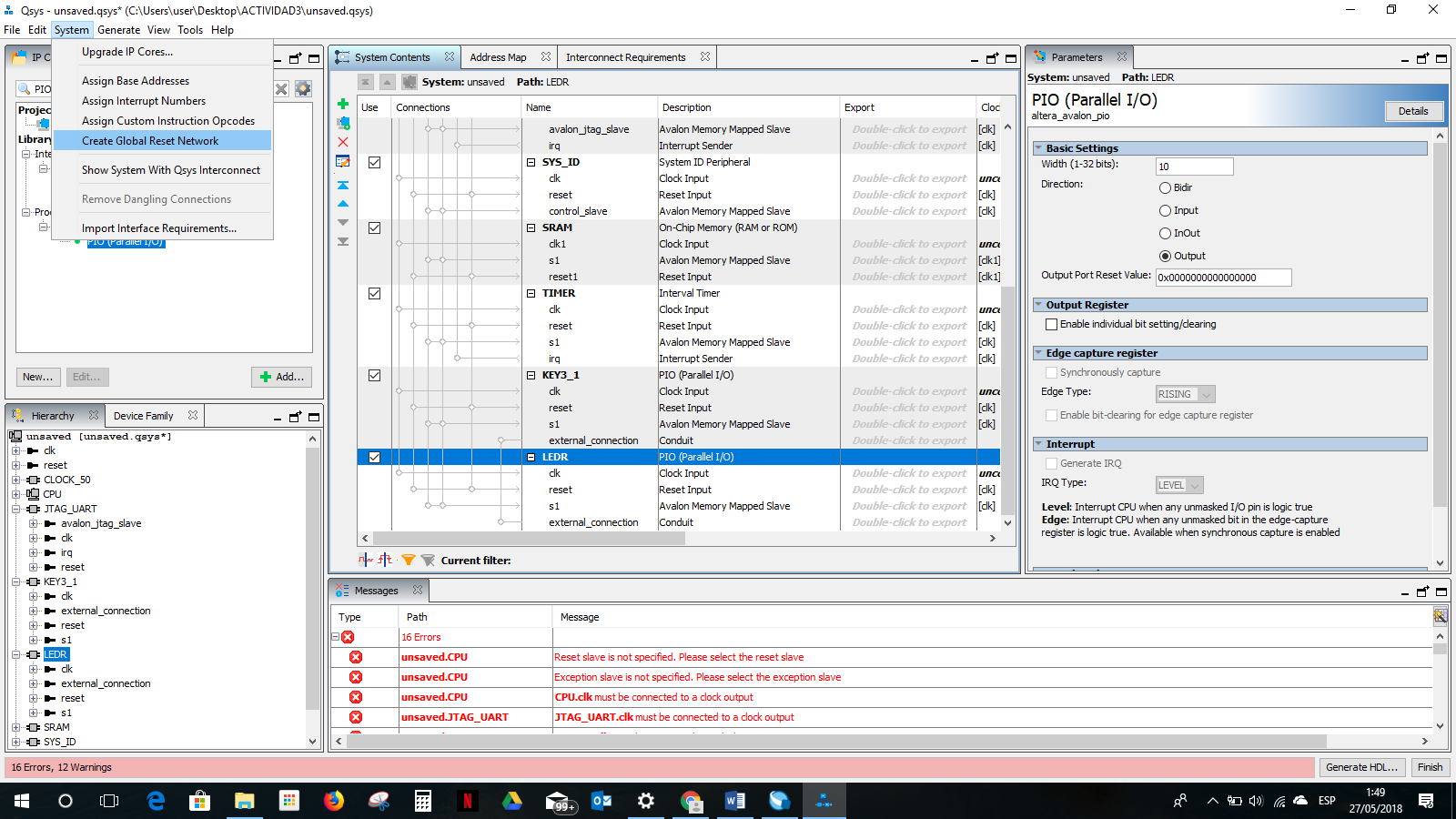
1. En el buscador de la subventana **IP Catalog**, coloque el nombre **JTAG UART** y añádalo.



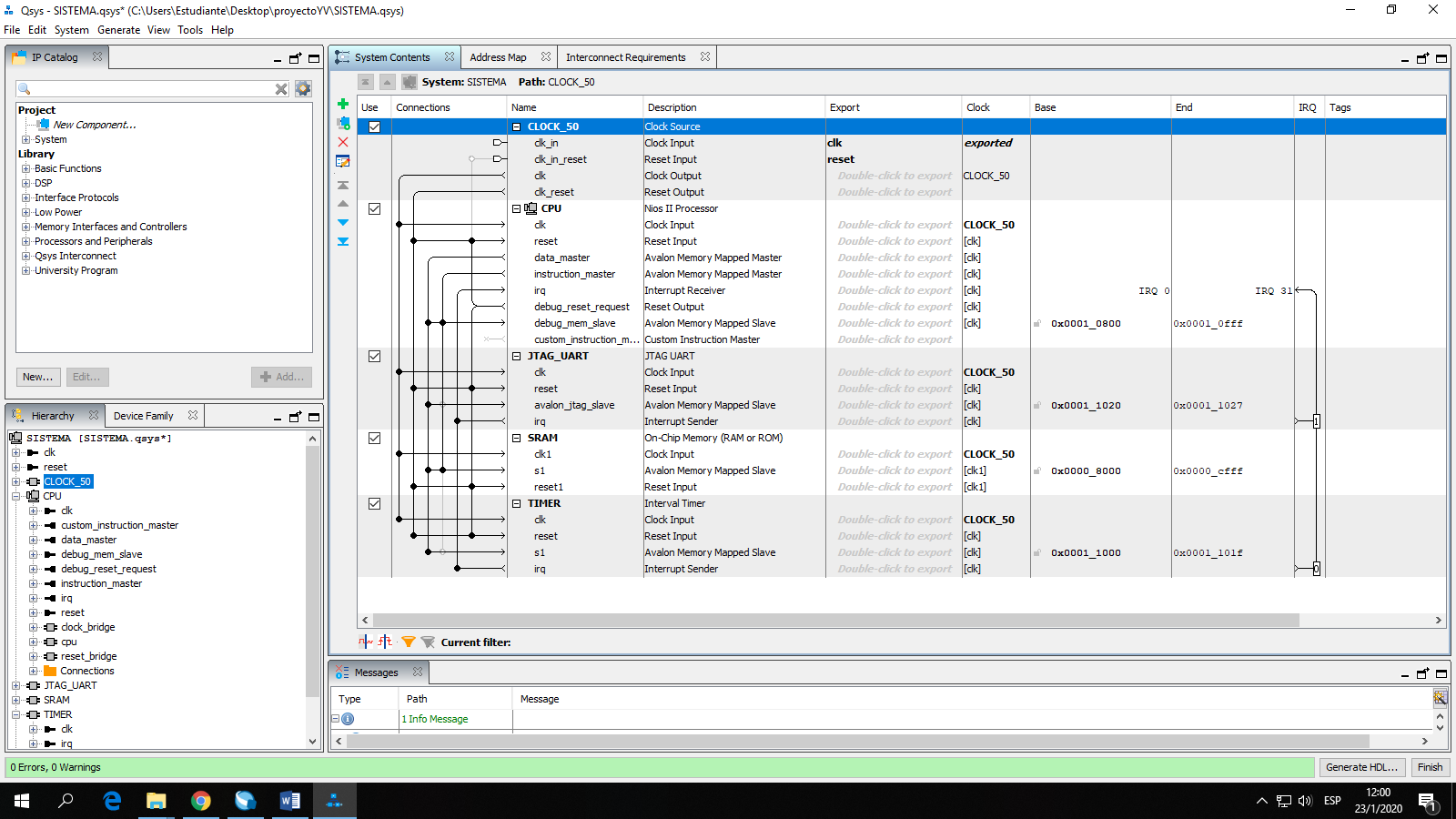
1. Desde su ventana de configuración de clic en **Finish**. A continuación, renombre el elemento como **JTAG\_UART**.



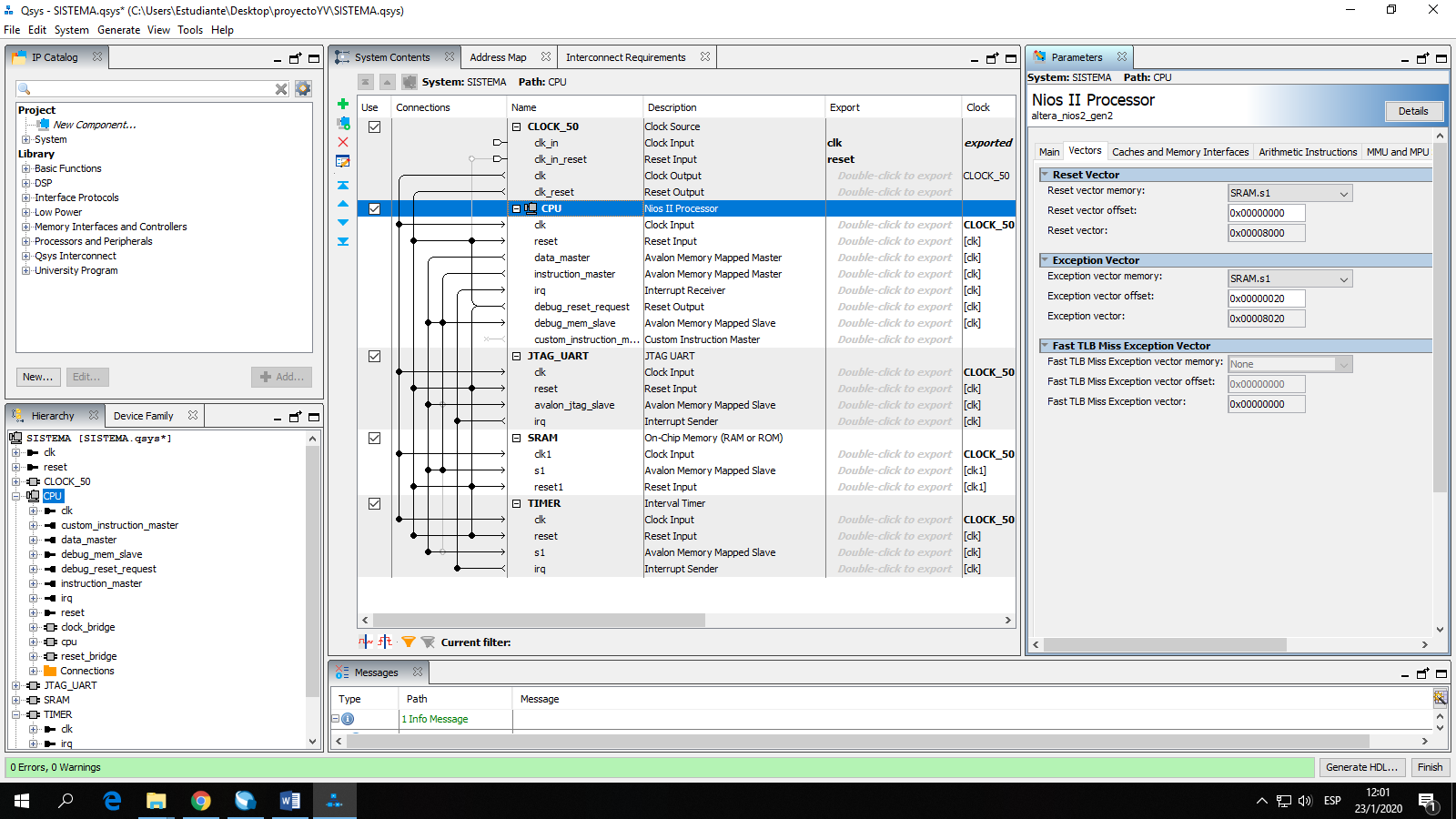
1. En el buscador de la subventana **IP Catalog**, coloque **“On-Chip RAM”** y seleccione el componente **On-Chip Memory (Ram or Rom)** y agréguelo al proyecto.
2. En la ventana de configuración, diríjase a **Size** y modifique el campo **Total memory size** a 20480 bytes. Dé clic en **Finish** y renómbrelo como **SRAM**.
3. En el buscador de la subventana **IP Catalog**, coloque **“Interval Timer”** y añádalo al proyecto. En la ventana de configuración del componente dé clic en **Finish** y finalmente, renombre el componente como **Timer**.
4. Una vez agregados todos los bloques, proceda a realizar las conexiones. Inicie con la señal de reset, para lo que deberá dirigirse al menú **System** y seleccione **Create Global Reset Network**, como se muestra en la figura.



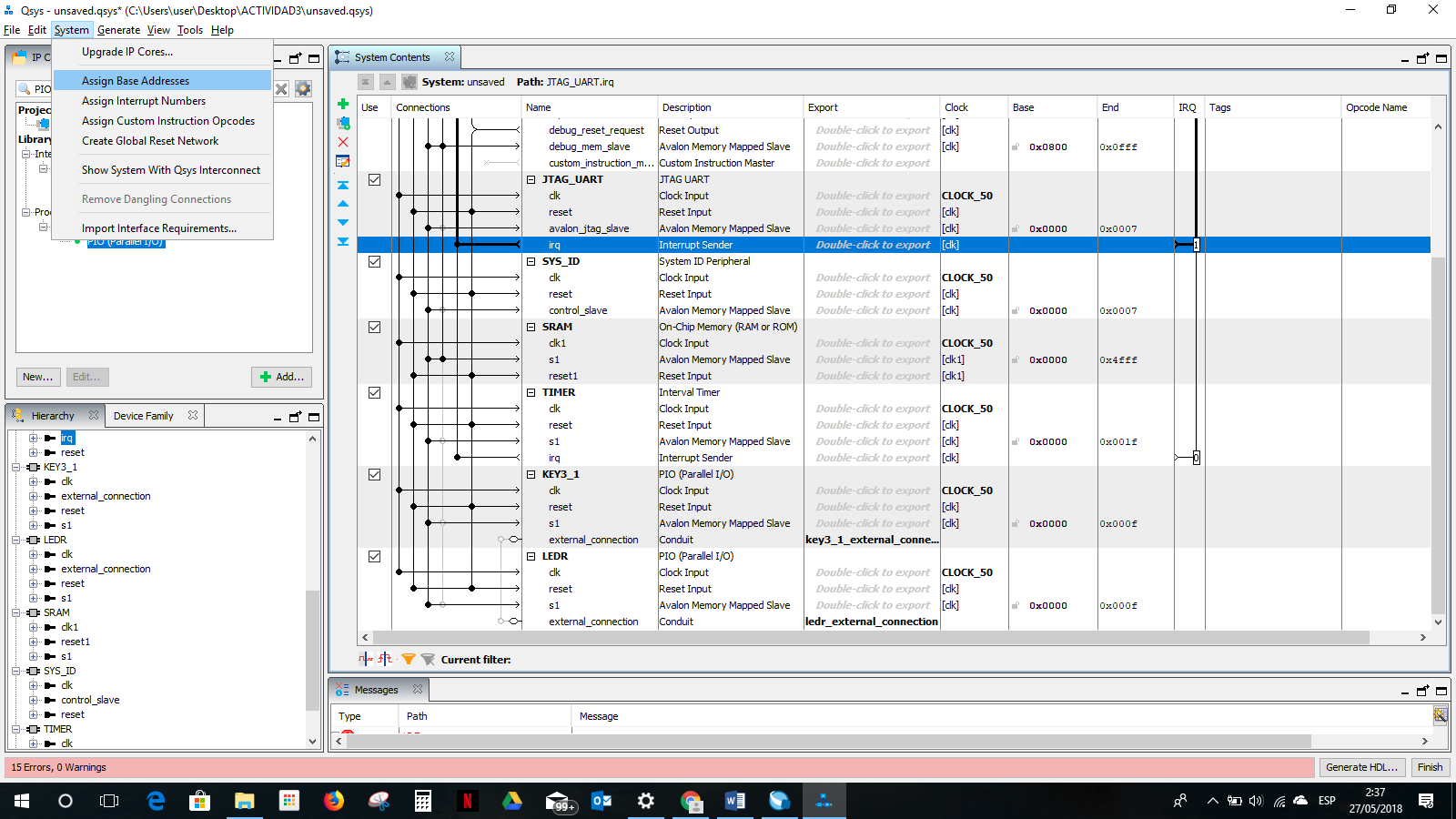
1. A continuación, proceda a conectar la señal de reloj, para esto seleccione el punto donde la señal clk de CLK\_50 se conecta con el resto de bloques.
2. Proceda a conectar todos los componentes como se observa en la siguiente figura, incluyendo los IRQ.



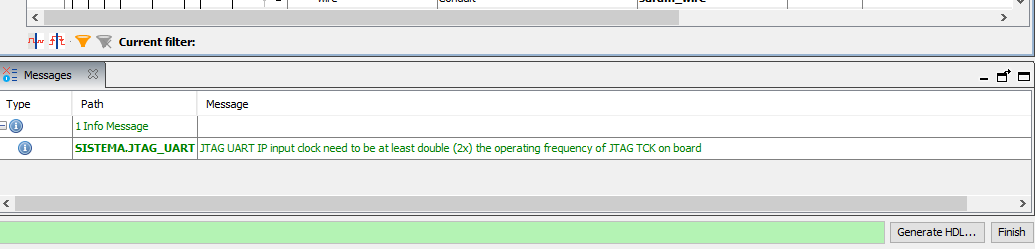
1. Dé doble clic sobre **CPU** y en la ventana **Parameters** diríjase a la pestaña **Vector**. En las secciones **Reset Vector** y **Exception Vector** coloque **SRAM.s1** respectivamente.

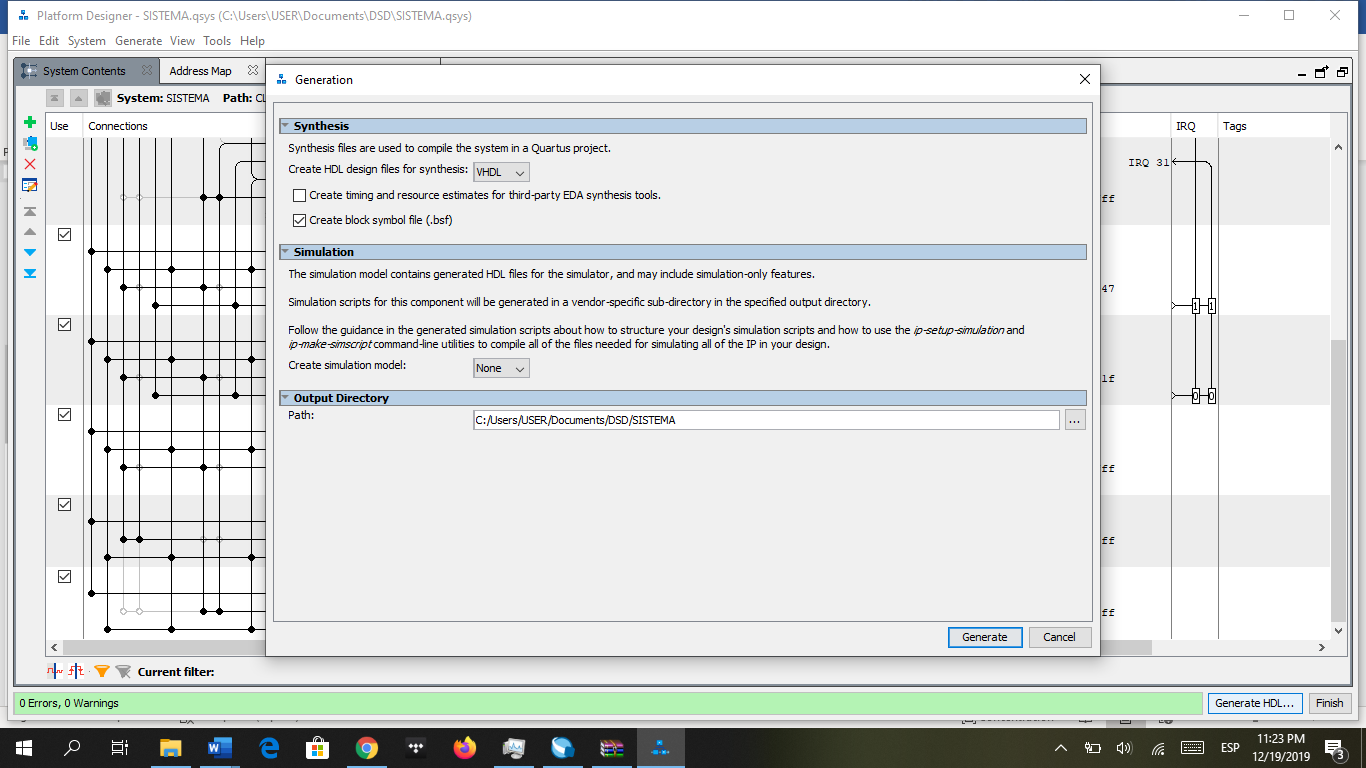


1. Diríjase al menú **System** y seleccione la opción **Assign Base Addresses**, como se muestra en la siguiente figura.

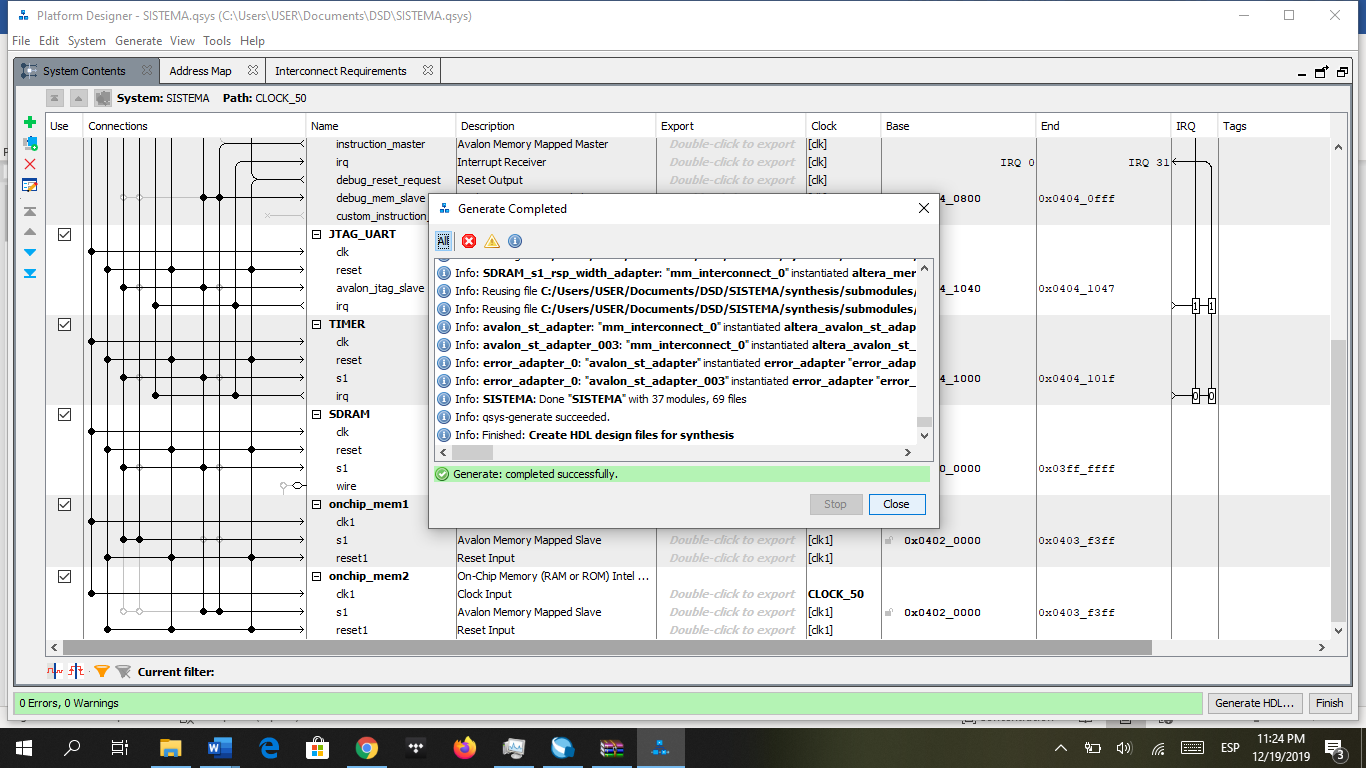


1. Diríjase al menú **File** y seleccione **Save As**. Guarde el archivo como **SISTEMA.qsys** en la carpeta del proyecto que creó.
2. Finalmente, seleccione **Generate HDL**, seleccione **VHDL** y dé clic en **Generate**.

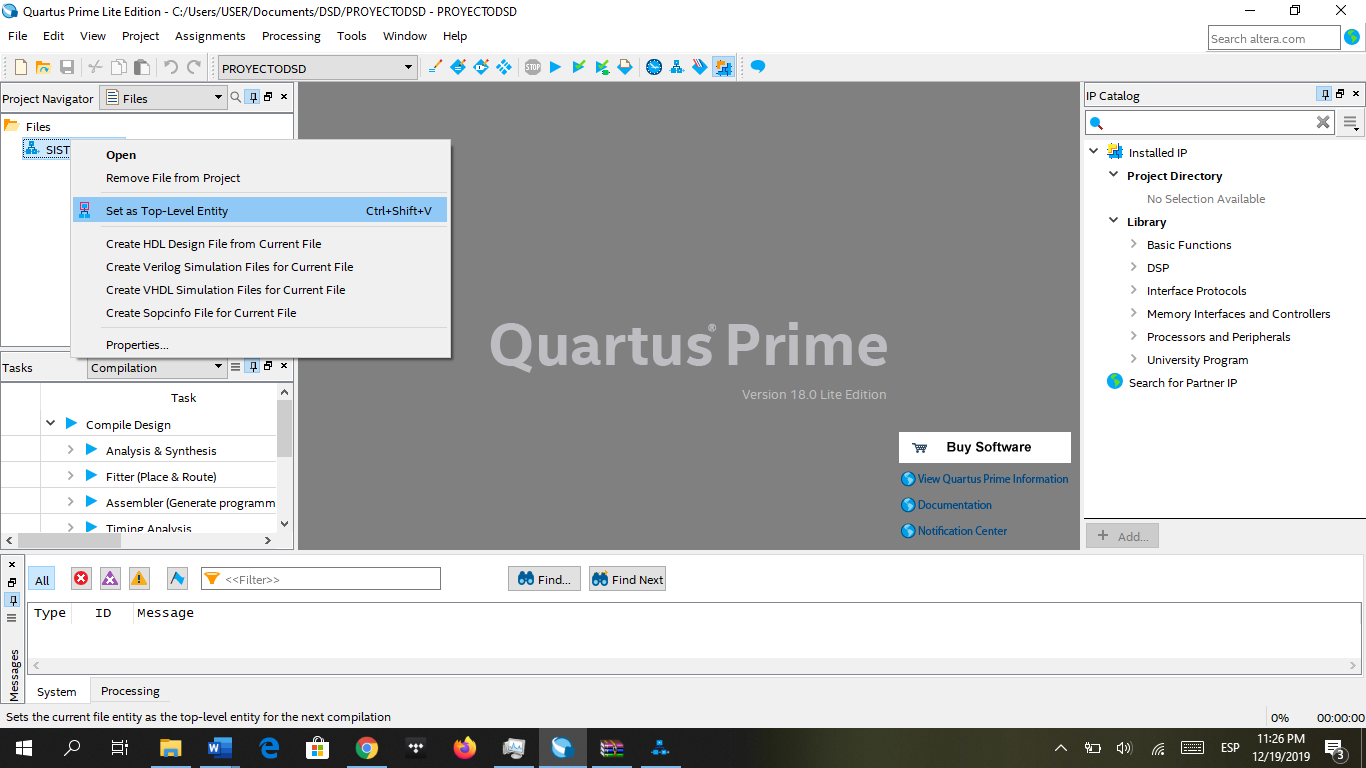




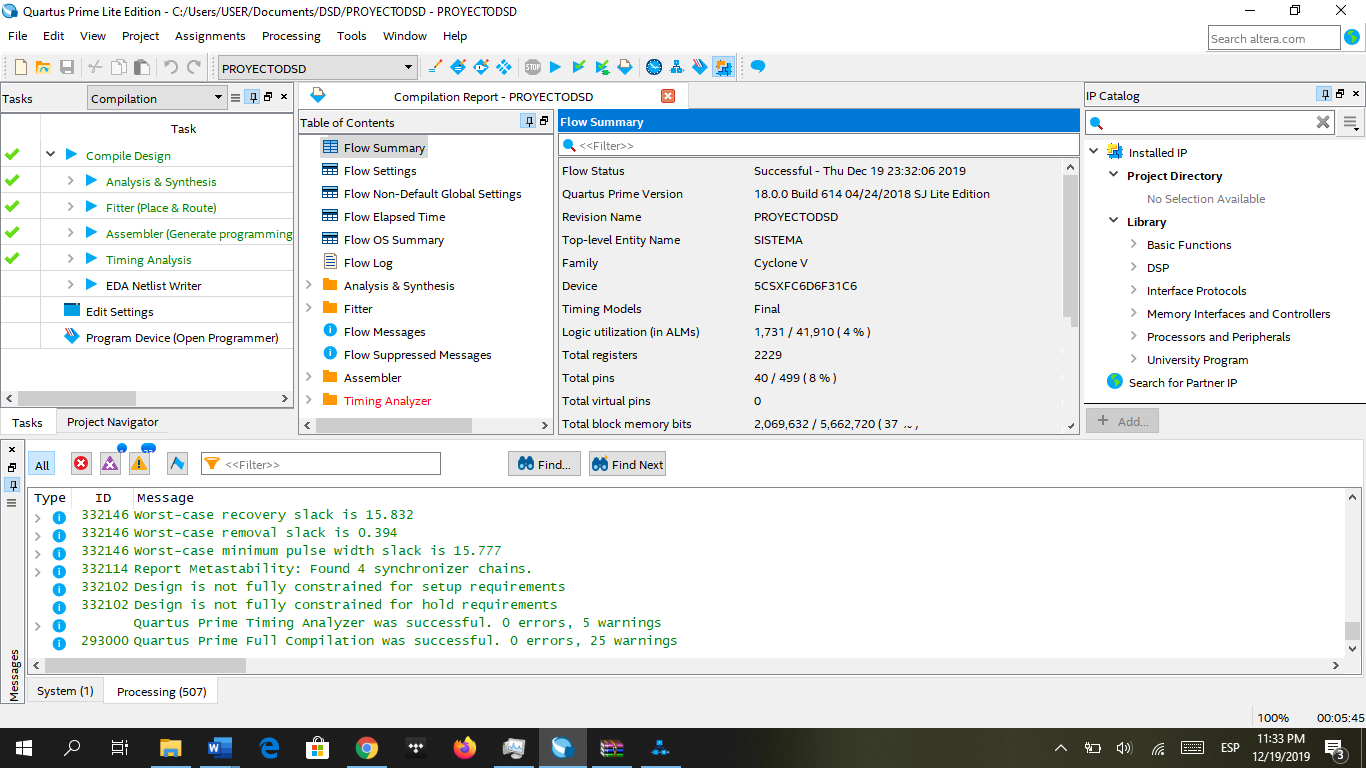
1. Esperar a que termine de Generar y seleccione **close**.



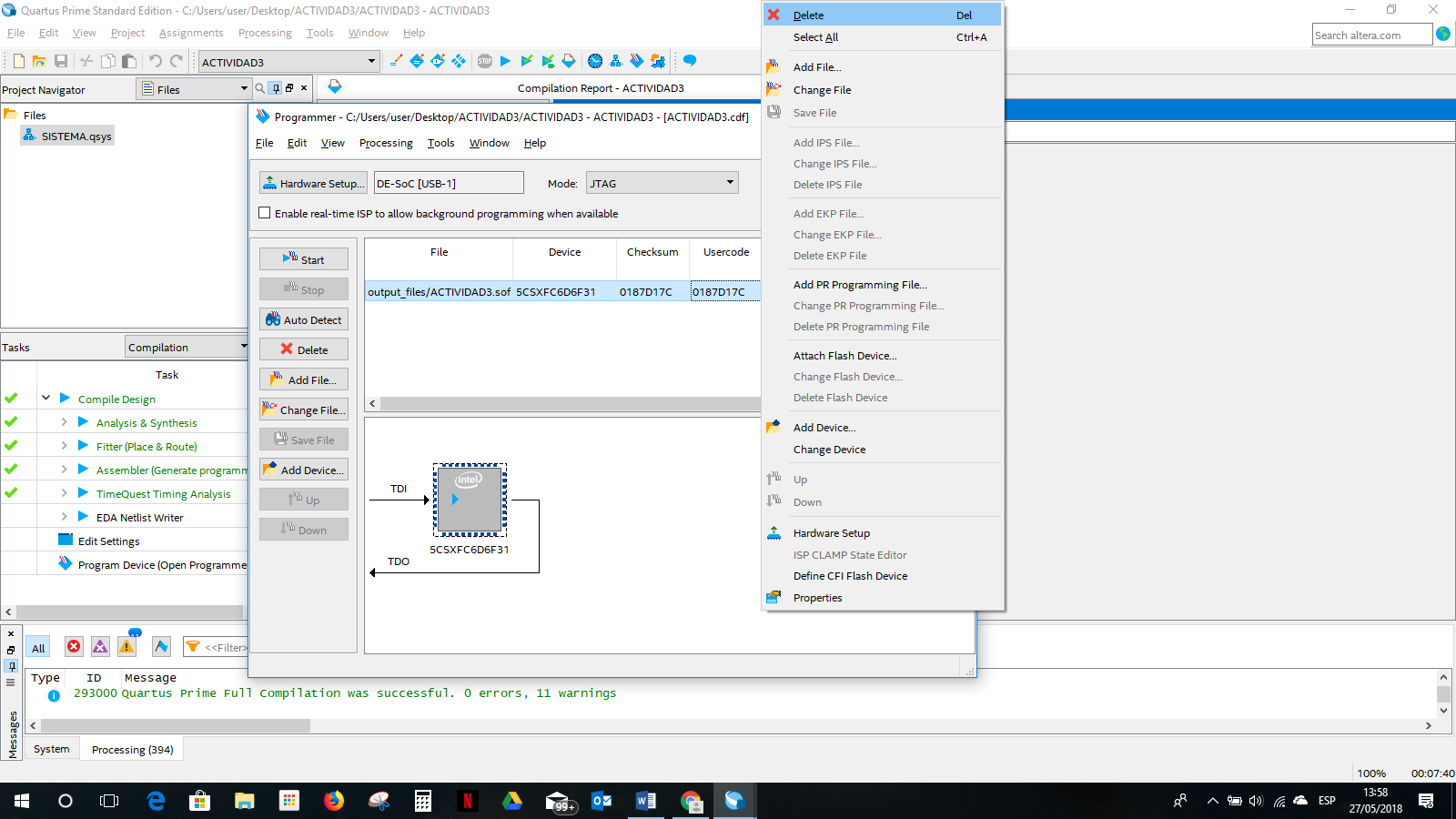
1. En Quartus, seleccione **SISTEMA.qsys**, establézcalo como **TOP LEVEL ENTITY** y compile el proyecto. En caso de no aparecer el archivo .qsys, haga clic derecho en la carpeta Files, seleccione la opción **Add/Remove Files in Project**, y en la ventana que aparecerá dé clic en **Add All**.



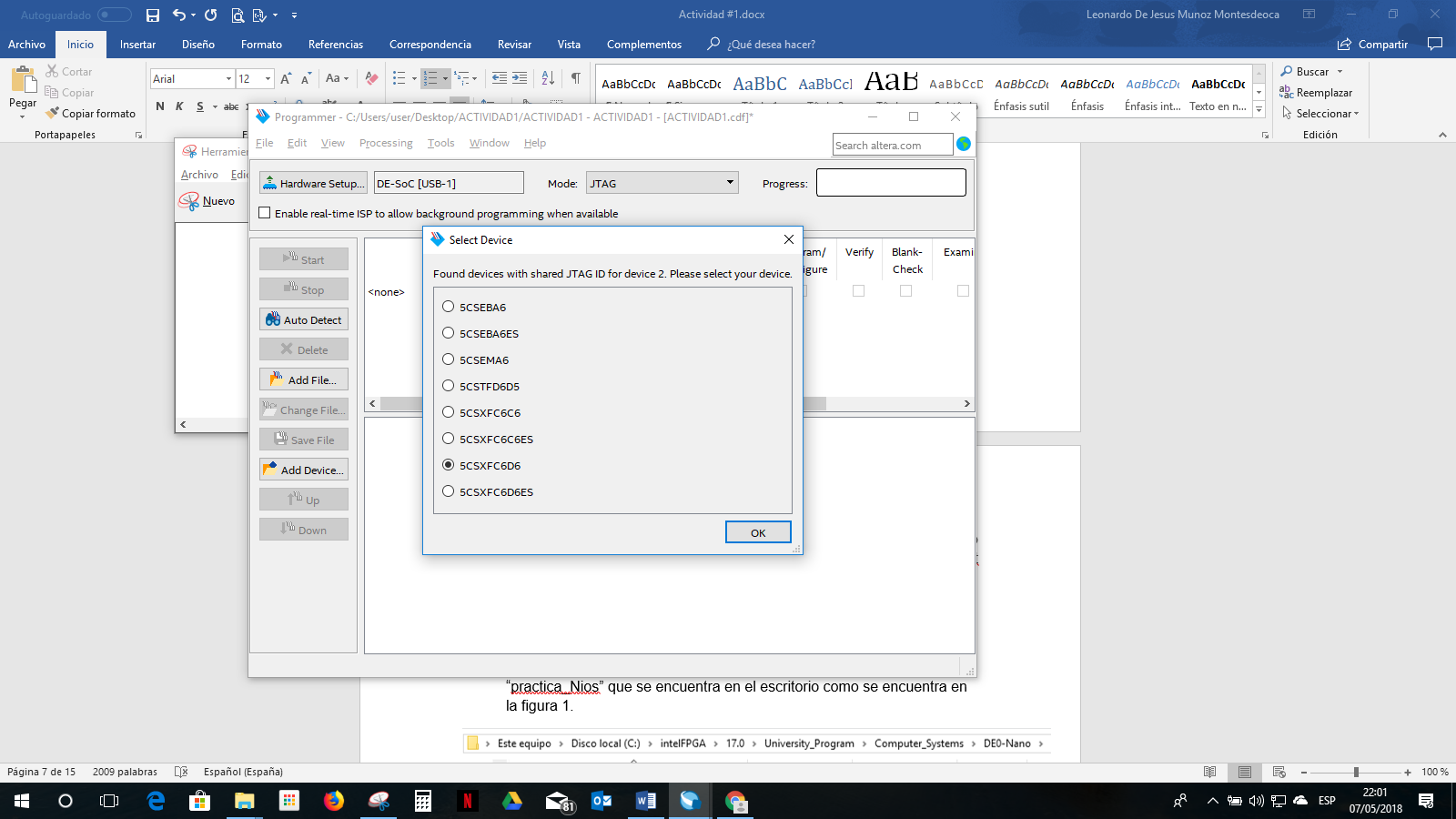
1. Verificar que no haya errores en la compilación.



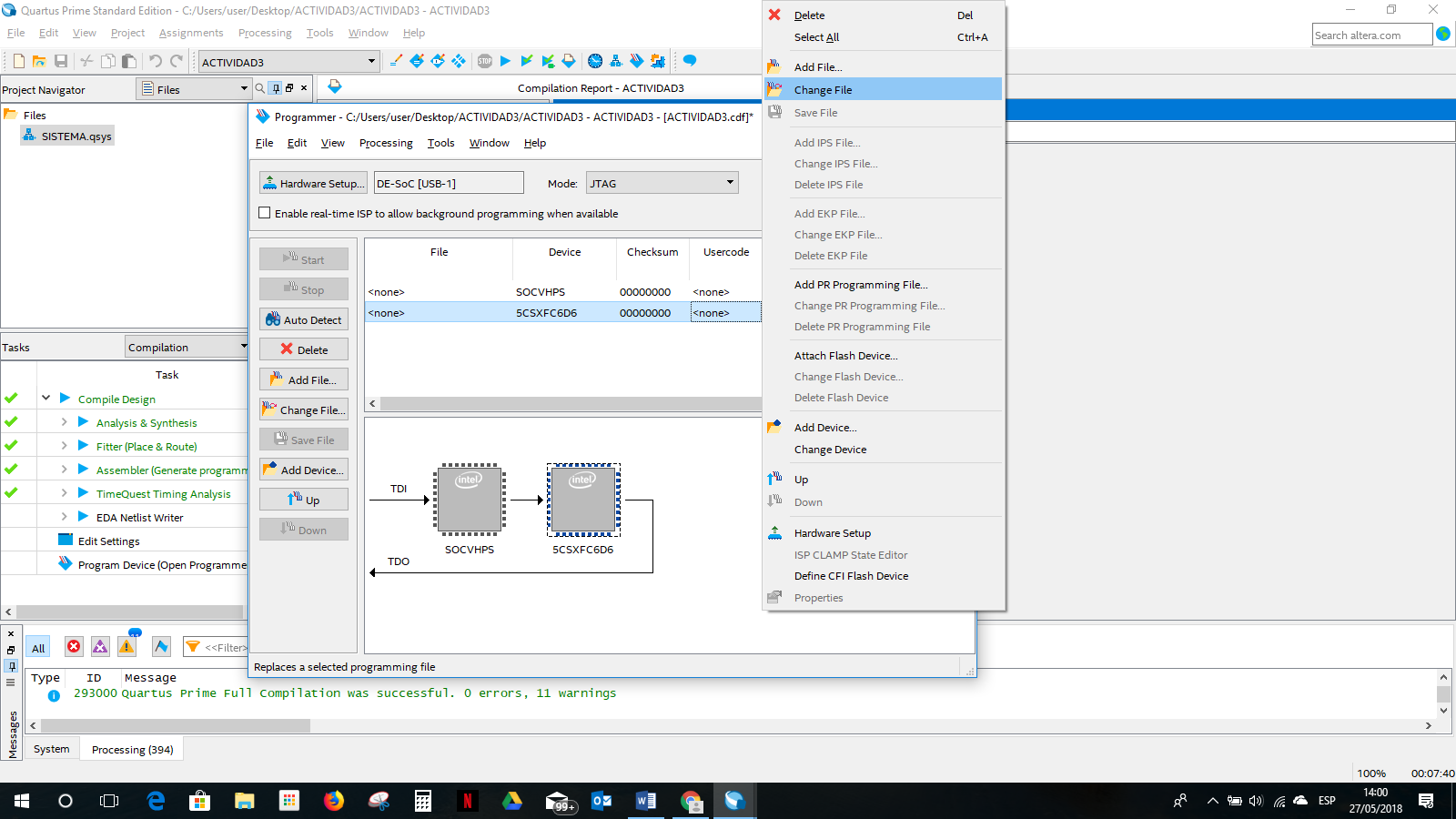
1. Proceda a conectar la tarjeta de desarrollo **DE10\_Standard** tanto a la PC como a la fuente de alimentación.
2. Una vez completada la compilación, diríjase a **Programmer** que se encuentra en la barra de tareas.
3. Proceda a eliminar cualquier archivo que se encuentre en la tarjeta dando clic derecho en el mismo y escogiendo la opción **Delete**.



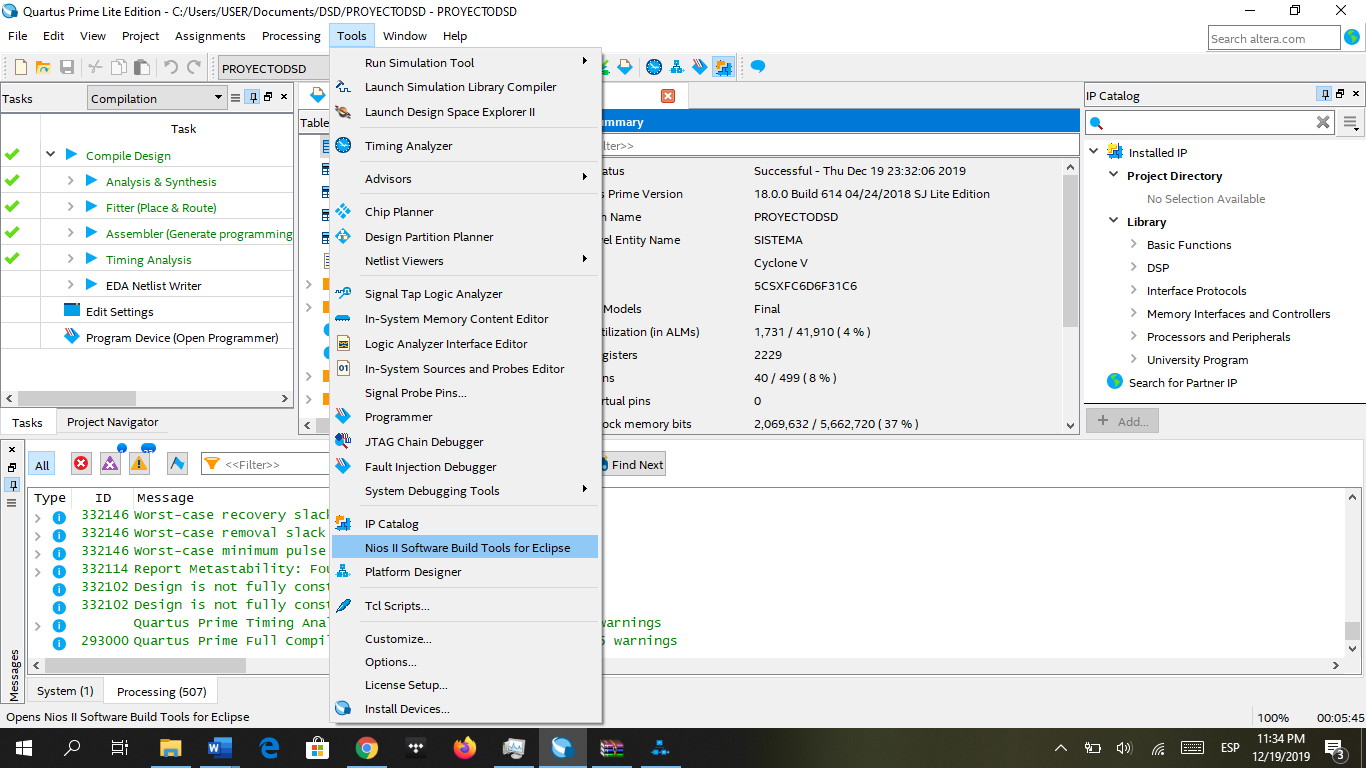
1. Una vez eliminado el archivo y que no se encuentre ningún otro en la ventana Programmer, escoja la opción **Auto Detect**, y seleccione la opción **5CSXFC6D6** y de clic en **OK**.



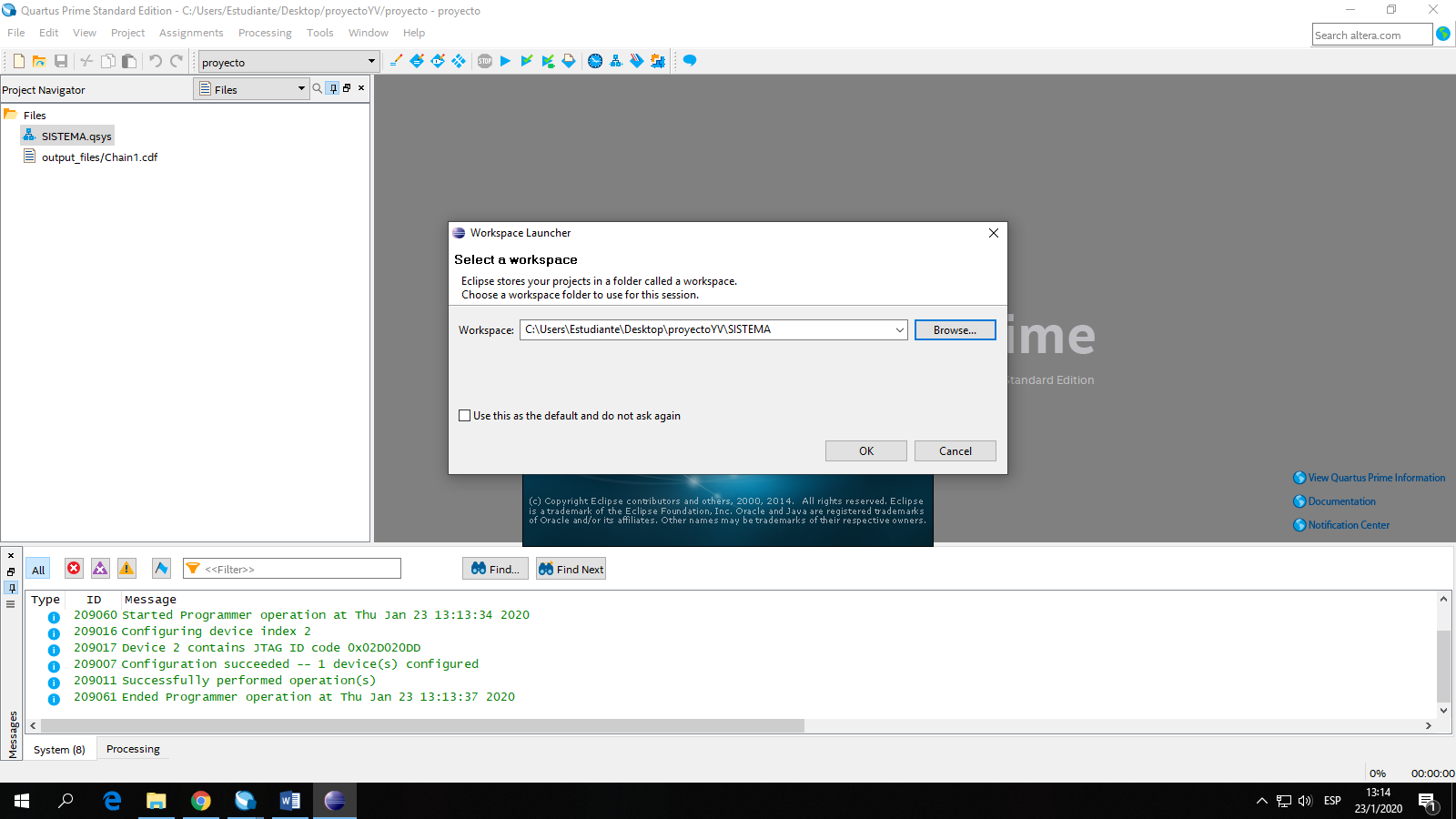
1. De los dos archivos que aparecen: **SOCBHPS** y **5CSXFC6D6** se da clic derecho en el segundo y se escoge **Change File**, y se lo reemplaza por el archivo .sof generado en la carpeta **Output\_Files** que se encuentra en la carpeta del proyecto.



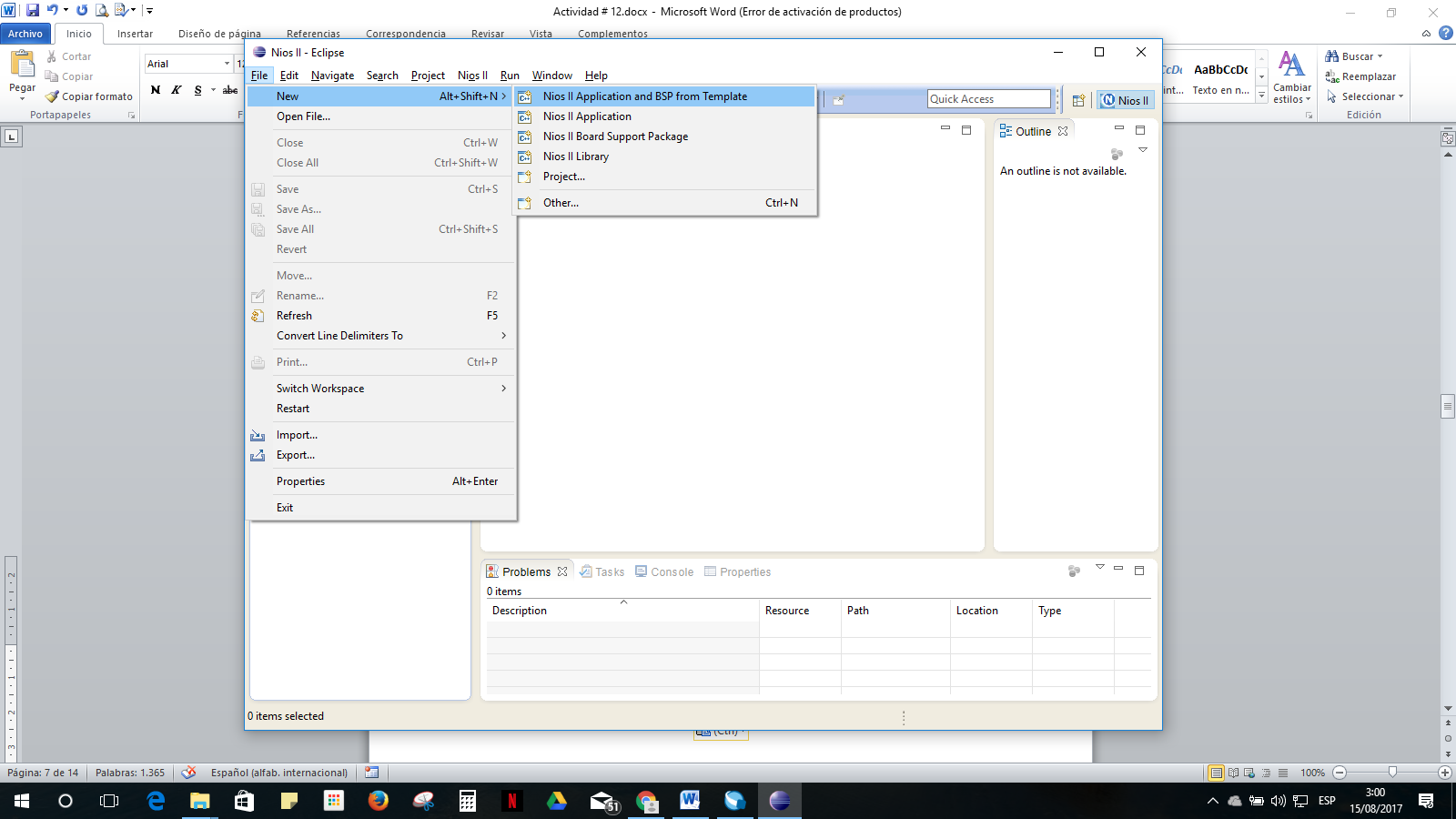
1. Se da clic en la casilla correspondiente a la columna **Program/Configure** del archivo .sof y luego se da clic en **Start** para que se programe la FPGA. Se habrá realizado la programación de manera exitosa una vez que aparezca el 100% en el recuadro de Progress.
2. Abrir **NIOS II Software Build Tools for Eclipse** desde el menú **Tools**.



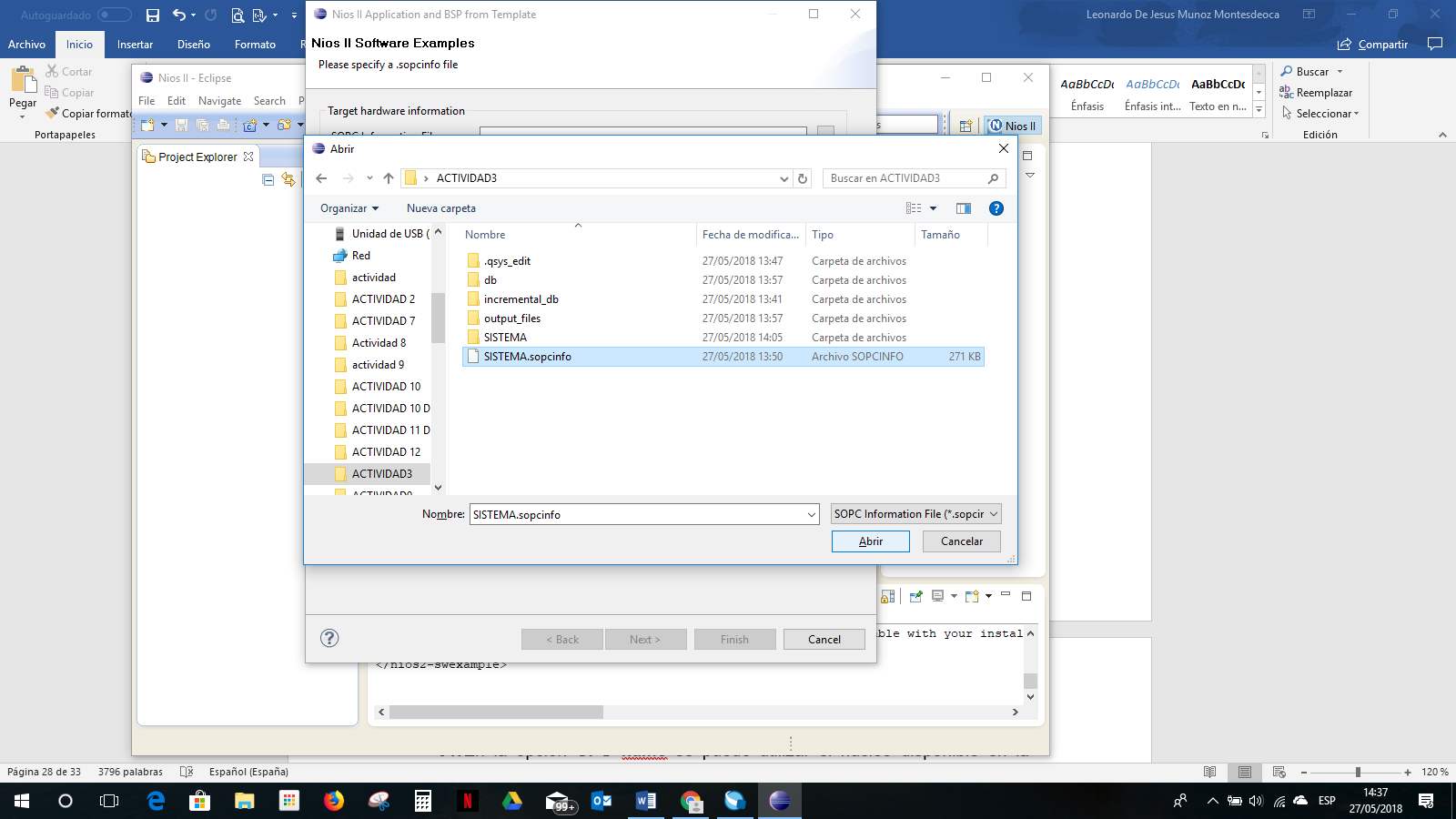
1. Seleccionar el Workspace (Carpeta SISTEMA dentro de la carpeta del Proyecto).



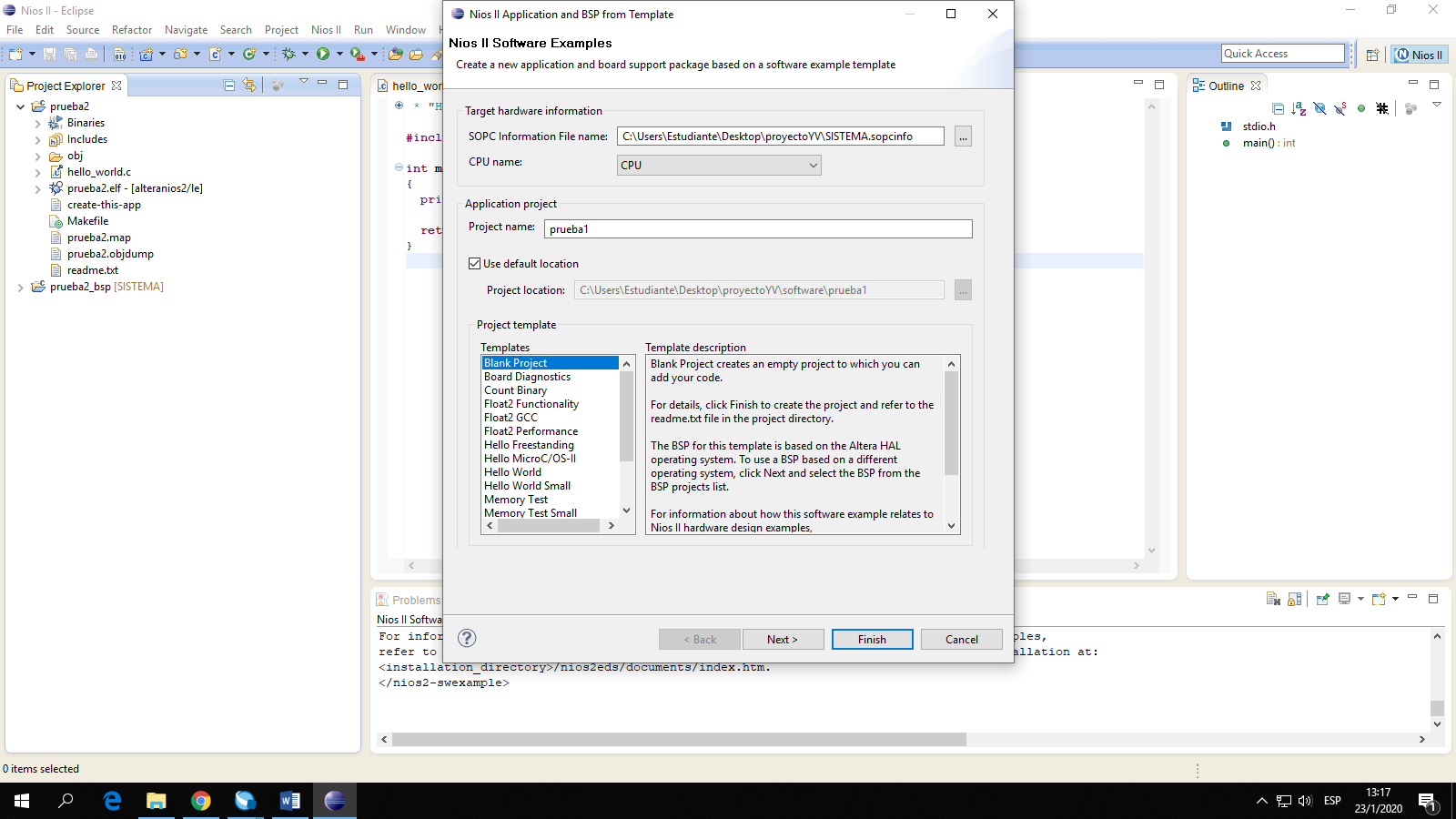
1. Dentro del entorno de Eclipse cree un nuevo proyecto, para ello seleccione **File→New→Nios II Application and BSP from Template**.



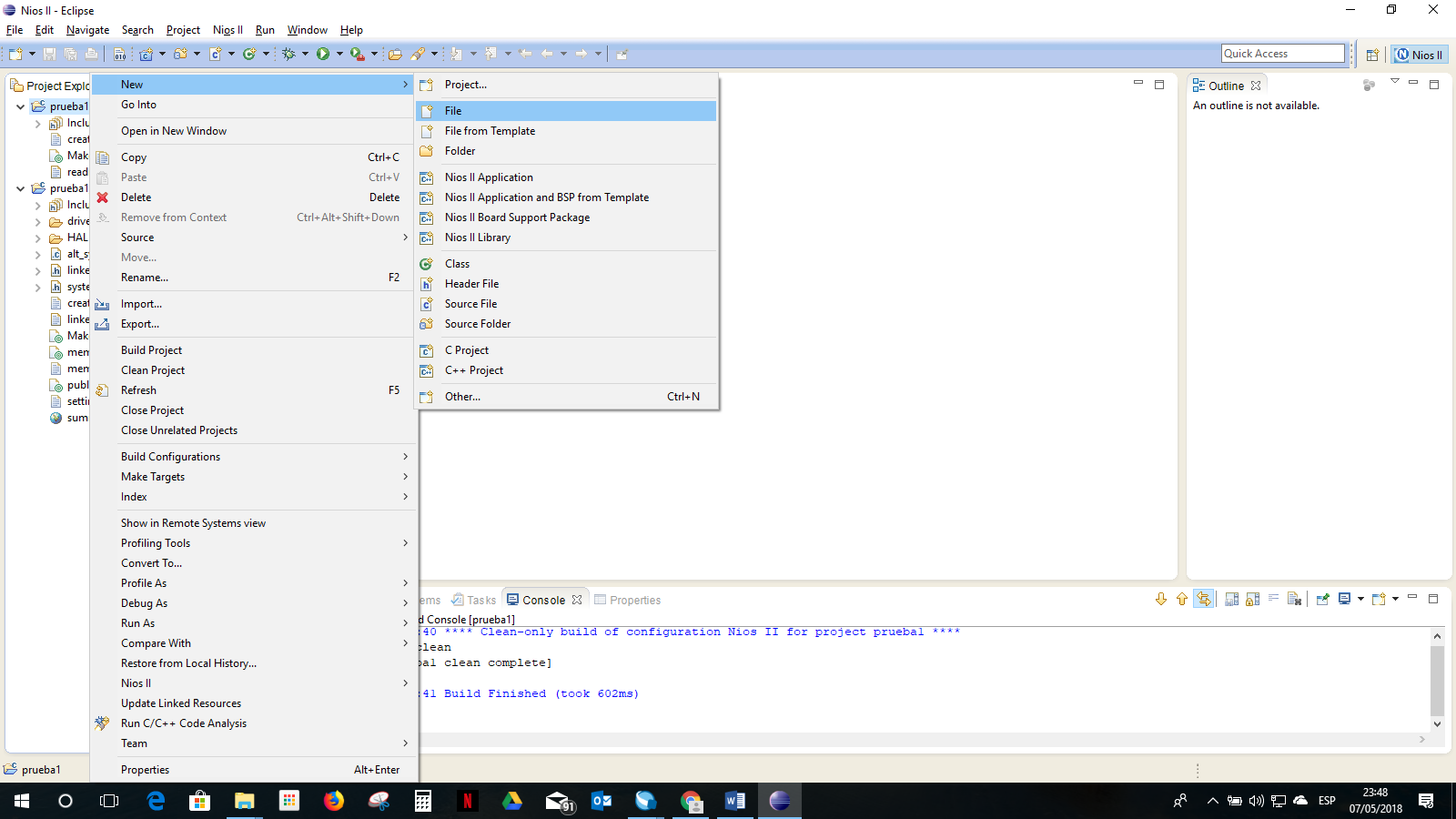
1. En la opción **SOPC Information File Name**, selecciona el archivo SISTEMA.sopcinfo que tendrá la información de la computadora embebida.



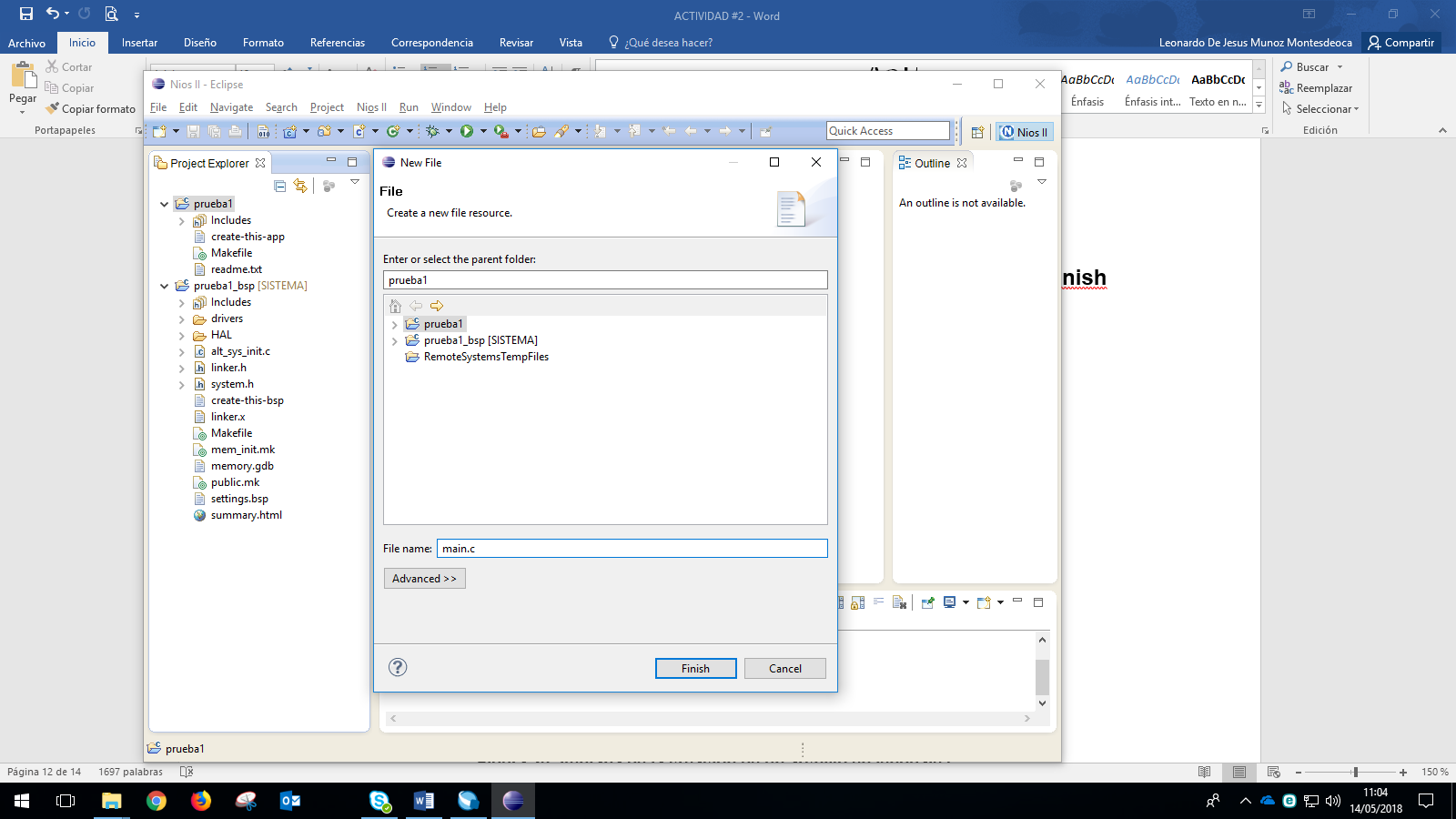
1. En la opción **CPU name** se seleccionará el **CPU**.
2. En **Project name** colocará el nombre del proyecto, en este caso, **prueba**.
3. En **Project Template**, seleccione la opción **Blank Project** y dé clic en **Finish**.



1. En la subventana de **Project Explorer** podrá visualizar dos carpetas llamadas prueba y prueba\_bsp. Para crear un archivo en lenguaje C haga clic derecho en la carpeta prueba y seleccione New **→ File.**



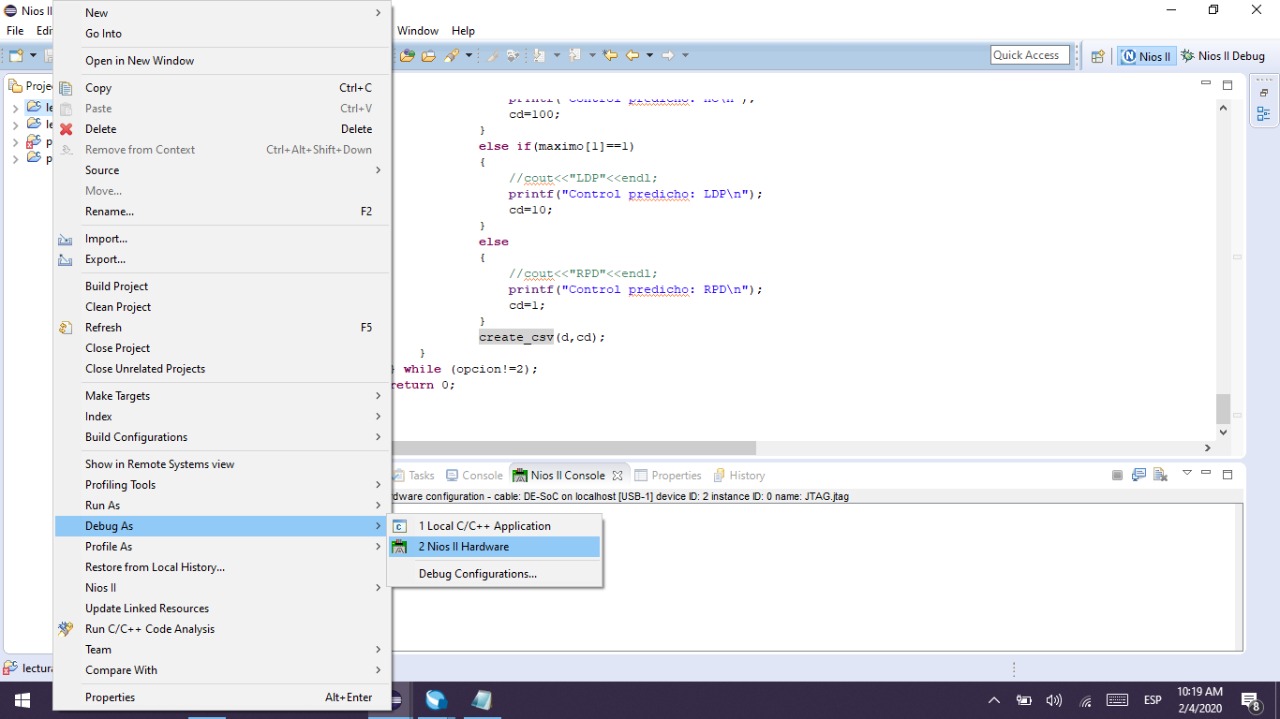
1. En la ventana **New File**, coloque el nombre **main.c** y luego haga clic en **Finish**.



1. Copie la siguiente programación en el archivo main.c creado.

|  |
| --- |
| #include <stdio.h> |
|  | //#include <io.h> |
|  | #include <unistd.h> |
|  | //#include "system.h" |
|  | #include <math.h> |
|  | float \*mapminmax(float x[]) |
|  | { |
|  | //int n=0,j=0; |
|  | int j=0; |
|  | float ymax=1,ymin=-1; |
|  | float xmin[]={-1696.79999999999995452526491135,-345.760975609756087578716687858,0}; |
|  | float xmax[]={513.5,505.326780821917679986654547974,60}; |
|  | static float norm[3]; |
|  | for (j=0; j!=3; ++j) |
|  | { |
|  | norm[j]=((ymax-ymin)/(xmax[j]-xmin[j]))\*(x[j]-xmin[j])+ymin; |
|  | } |
|  | return norm; |
|  | } |
|  |  |
|  | float dot(float A[], float B[]) |
|  | { |
|  | int i=0; |
|  | float dotproduct=0; |
|  | for (i=0;i!=3;++i) |
|  | { |
|  | dotproduct=A[i]\*B[i]+dotproduct; |
|  | } |
|  | return dotproduct; |
|  | } |
|  | float dot2(float A[], float B[]) |
|  | { |
|  | int i=0; |
|  | float dotproduct2=0; |
|  | for(i=0;i!=10;++i) |
|  | { |
|  | dotproduct2=A[i]\*B[i]+dotproduct2; |
|  | } |
|  | return dotproduct2; |
|  | } |
|  | float \*layer1(float p[]) |
|  | { |
|  | //int i=0; |
|  | static float iz[10]; |
|  | float w1[]={0.89968280104466424874,1.6564714048995743401,0.12815531236621699462}; |
|  | float w2[]={-0.377241511468634938,-2.2518470763549163216,-3.8261554752454505213}; |
|  | float w3[]={5.1558877863038121703,13.308246878694783177,-1.1414099471159522281}; |
|  | float w4[]={4.74352232856336542,2.9228731944090937667,5.2763101882772449613}; |
|  | float w5[]={-14.381330388647150187,-4.2178709056964915902,-0.30483328401431458721}; |
|  | float w6[]={-2.9250524588283925986,-0.41199028040865603995,-13.258363138701009021}; |
|  | float w7[]={1.0058667546690249761,15.465903205838323231,-4.1039896212119924002}; |
|  | float w8[]={8.0707947086012250537,1.2832288879346758215,3.0761627106087123806}; |
|  | float w9[]={2.380827744116928546,-1.3843225498493145764,-3.3511625643480780212}; |
|  | float w10[]={-1.0238747474247833491,0.52756053306743666553,-2.0054709368566068228}; |
|  | iz[0]=dot(p,w1); |
|  | iz[1]=dot(p,w2); |
|  | iz[2]=dot(p,w3); |
|  | iz[3]=dot(p,w4); |
|  | iz[4]=dot(p,w5); |
|  | iz[5]=dot(p,w6); |
|  | iz[6]=dot(p,w7); |
|  | iz[7]=dot(p,w8); |
|  | iz[8]=dot(p,w9); |
|  | iz[9]=dot(p,w10); |
|  | return iz; |
|  | } |
|  | float \*tansig(float u[]) |
|  | { |
|  | int i=0; |
|  | static float t[10]; |
|  | for(i=0;i!=10;++i) |
|  | { |
|  | t[i]=2/(1+exp(-2\*u[i]))-1; |
|  | } |
|  | return t; |
|  | } |
|  | float \*softmax(float entrada[]) |
|  | { |
|  | int i=0,j=0; |
|  | static float salida[3]; |
|  | float variable1=0; |
|  | for(i=0;i!=3;++i) |
|  | { |
|  | variable1=(1/(exp(entrada[i])+variable1)); |
|  | } |
|  | for(j=0;j!=3;++j) |
|  | { |
|  | salida[j]=exp(entrada[j])\*variable1; |
|  | } |
|  | return salida; |
|  | } |
|  | float \*layer2(float p2[]) |
|  | { |
|  | static float der[3]; |
|  | float w1[]={-0.14915101799897698132,-1.2755991304137603848,-2.8844768204750614871,-2.5161017432584542597,-6.1312441080655863956,-2.5223250155606344869,2.4247589619245566439,-4.9274869449973648372,-0.26063773735266954068,0.555259005684032525}; |
|  | float w2[]={2.1418545456717743569,1.8769413943650685717,5.0913702656480825937,1.1272969504397540863,1.4559314971317005583,7.4038023774013490552,-4.4365715270141468096,0.96729739219398713779,-2.3252303987057310053,1.3474353246642727377}; |
|  | float w3[]={-2.1534215473279991748,1.7235205612693003108,-2.5921664246055997616,2.1041388438141788164,5.0978507614848069807,-5.3379176212354764175,2.7061122032974274099,3.6011652460881622417,3.3488214929753881144,-1.9937513850388983716}; |
|  | der[0]=dot2(p2,w1); |
|  | der[1]=dot2(p2,w2); |
|  | der[2]=dot2(p2,w3); |
|  | return der; |
|  | } |
|  | void create\_csv(int cd){ |
|  | FILE \*fp; |
|  |  |
|  | fp=fopen("./predicho.csv","a+"); |
|  | fprintf(fp,"%d\n",cd); |
|  | fclose(fp); |
|  | } |
|  |  |
|  | const char\* getfield(char\* line, int num) |
|  | { |
|  | const char\* tok; |
|  | for (tok = strtok(line, ","); |
|  | tok && \*tok; |
|  | tok = strtok(NULL, ",\n")) |
|  | { |
|  | if (!--num) |
|  | return tok; |
|  | } |
|  | return NULL; |
|  | } |
|  | int main () |
|  | { |
|  | FILE\* stream = fopen("C:\\Users\\USER\\Downloads\\Alldata.csv", "r"); |
|  | char line[1024]; |
|  | while (fgets(line, 1024, stream)){ |
|  | int i=0,j=0,k=0,q=0,r=0; |
|  | //float a=0,b=0,c=0; |
|  | //static int d=0; |
|  | char\* tmp = strdup(line); |
|  | char\* tmp1 = strdup(line); |
|  | char\* tmp2 = strdup(line); |
|  | //printf("%s,%s,%s\n",x,y,z); |
|  | float A[3]={atof(getfield(tmp,1)),atof(getfield(tmp1,2)),atof(getfield(tmp2,3))}; |
|  | //99,89.1409476661953,368 // 100 HC |
|  | //138.800000000000,119.976675786594,34 // 010 LDP |
|  | //99.8000000000000,111.940947666195,37 // 001 RPD |
|  | float p[3]; |
|  | float outl1[10]; |
|  | float \*puntero1=mapminmax(A); |
|  | float netsum[10]; |
|  | float netsum2[3]; |
|  | float outl2[10]; |
|  | float outl3[3]; |
|  | float nnout[3]; |
|  | float maximo[]={0,0}; |
|  | for(i=0;i!=3;++i) |
|  | { |
|  | p[i]=\*(puntero1+i); |
|  | } |
|  | float \*puntero2=layer1(p); |
|  | for(j=0;j!=10;++j) |
|  | { |
|  | outl1[j]=\*(puntero2+j); |
|  | } |
|  | netsum[0]=-4.3443923455882753615+outl1[0]; |
|  | netsum[1]=-1.9345122521365221147+outl1[1]; |
|  | netsum[2]=-3.9021835054654658848+outl1[2]; |
|  | netsum[3]=-3.8344123411362138754+outl1[3]; |
|  | netsum[4]=6.6384971583323313382+outl1[4]; |
|  | netsum[5]=5.5743667202569282537+outl1[5]; |
|  | netsum[6]=-1.7301759667010074839+outl1[6]; |
|  | netsum[7]=-1.4470829006873584532+outl1[7]; |
|  | netsum[8]=-0.48131314172894840153+outl1[8]; |
|  | netsum[9]=-3.9080618488954046441+outl1[9]; |
|  | float \*puntero3=tansig(netsum); |
|  | for(k=0;k!=10;++k) |
|  | { |
|  | outl2[k]=\*(puntero3+k); |
|  | } |
|  | float \*puntero4=layer2(outl2); |
|  | for(q=0;q!=3;++q) |
|  | { |
|  | outl3[q]=\*(puntero4+q); |
|  | } |
|  | netsum2[0]=0.78802390874671768284+outl3[0]; |
|  | netsum2[1]=-2.5825335193302128012+outl3[1]; |
|  | netsum2[2]=1.7262136109376675641+outl3[2]; |
|  | float \*puntero5=softmax(netsum2); |
|  | for(r=0;r!=3;++r) |
|  | { |
|  | nnout[r]=\*(puntero5+r); |
|  | if(nnout[r]>=maximo[0]) |
|  | { |
|  | maximo[0]=nnout[r]; |
|  | maximo[1]=r; |
|  | } |
|  | //printf("%d\n",maximo[1]); |
|  | } |
|  |  |
|  | static int cd=0; |
|  | if(maximo[1]==0) |
|  | { |
|  | //cout<<"HC"<<endl; |
|  | //printf("Control predicho: HC\n"); |
|  | cd=100; |
|  | } |
|  | else if(maximo[1]==1) |
|  | { |
|  | //cout<<"LDP"<<endl; |
|  | //printf("Control predicho: LDP\n"); |
|  | cd=10; |
|  | } |
|  | else |
|  | { |
|  | //cout<<"RPD"<<endl; |
|  | //printf("Control predicho: RPD\n"); |
|  | cd=1; |
|  | } |
|  | free(tmp); |
|  | free(tmp1); |
|  | free(tmp2); |
|  | //free(cd); |
|  | create\_csv(cd); |
|  | } |
|  | } |

1. Haga click derecho en la carpeta prueba\_bsp -> Properties -> Nios II BSP Properties. A continuación, dé click en BSP Editor.
2. Diríjase a la pestaña Software Packages dentro del BSP Editor y marque la casilla de altera\_hostfs y a continuación dé click en File -> Save.
3. Haga click en Generate y proceda a cerrar el BSP Editor.
4. Asegúrese de guardar todos los cambios. A continuación, de clic en **Build All**.
5. Haga click derecho sobre el proyecto y seleccione Debug As -> NIOS II Hardware.



1. Ahora el programa está listo para leer el archivo que contiene todos los datos a clasificar.