Pràctica 4: Estació Meteorològica

Durada: 3 sessions

Introducció

Un cop dissenyat i verificat el mestre I^2C, l'utilitzarem com a interfície de comunicacions I^2C d'una estació meteorològica simple basada en el sensor BME280. El sensor BME280 és un sensor digital combinat d'humitat, pressió i temperatura, que permet monitoritzar les condicions ambientals en temps real.

Objectius

- **Analitzar i combinar mòduls donats per generar un sistema digital complex** amb interfície d'usuari, adquisició, post-processat i visualització de dades.
- **Definir l'arquitectura del l'estació meteorològica**, en base a mòduls donats. Seguint els criteris de disseny adients, tal com l'ús de sincronitzadors, restabliments per encesa (de l'anglès *power-on-reset*), implementació d'IPs (de l'anglès *Intellectual Property*), entre d'altres.
- Definició de restriccions temporals necessàries per el correcte funcionament del sistema (SDC).
- **Implementació en FPGA Cyclone V 5CEBA4F23C7N de la placa de desenvolupament DE0-CV**.

Material

El material de suport el podeu descarregar del campus virtual:

- **Fitxer Zip amb el directori i fitxers pel projecte**
  - **misc:** models funcionals per verificació del sistema complet.
    - **i2c_slave_model.v** : model funcional de esclau I2C, emulant el sensor BME280.
    - **bme280_regs.mem** : contingut de la memòria /registres del esclau i2c
    - **pll_fm.v** : model funcional PLL.
    - **sys_model.v :** model del sistema, genera el rellotge base i el reset actiu per nivell baix.

    - **timescale.v :** defineix l'escala temporal pel fitxer on està inclòs.
  - **rtl**: codi font pels diferents controladors necessaris
    - **top_meteo_de0cv.v** : top del sistema a **completar**.
    - **bme280_compensation.v** :
    - **bme280_reader.v** : Control del procés de configuració i lectura del sensor BME280.
    - **bme280_i2c_ctrl.v** : Controlador del mestre i2c que implementa les seqüències per poder escriure i llegir registres del sensor.
    - **digital_por.v** : power on reset digital.
    - **gray_timer.v** : Temporitzador en format gray.
    - **hex2seg.v** : converso Hexadeciman a 7 segments.
    - **bcd2seg.v** : converso BCD (binary code decimal) a 7 segments.
    - **bin2bcd.v** : converso de binari a BCD (Binay Code Decimal)
    - **shifreg.v** : registre de desplaçament.
    - **sync_reg.v** : registre de sincronisme.
  - **sdc** : fitxers de restriccions de disseny.
  - **tb** : fitxers de testbench a **completar.**

- Plantilla informe de la pràctica.

Requisits tècnics de l'estació meteorològica

1. Ha de fer una adquisició de temperatura, pressió i humitat per segon.
2. Les dades es mostraran per les pantalles de set segments, en format BCD (de l'anglès *Binary-Coded Decimal*)
3. L'usuari ha de poder seleccionar quina magnitud (temperatura, pressió o humitat) es mostra per les pantalles, utilitzant els interruptors (**Taula 4**).

Taula 4 Connexions I^2C sensor BME280.

SW2	SW1	SW0	Magnitud
x	x	1	Temperatura
x	1	0	Pressió
1	0	0	Humitat

4. Ha de tenir una bandera d'error en el LED0.
5. Ús d'un PLL per generar dos dominis de rellotge diferents: un de 100 MHz que serà el rellotge del sistema, i un de 1 MHz per controlar quan es realitza una nova mesura.
6. Reset extern asíncron actiu per nivell baix (al polsador Key0).
7. Recordeu d'implementar el *power-on-reset* per cada domini de rellotge.
8. El mòdul del sensor BME280 es connectarà al GPIO 1, tal com indica la **Taula 5**.

Taula 5 Connexions I^2C sensor BME280.

Senyal	BME PIN	DE0-CV	Observacions
SCL_io	SCK	GPIO_1_D27	
SDA_io	SDI	GPIO_1_D31	
SlaveAddr_LSbit_o	SDO	GPIO_1_D29	Fixe a "1" o "0"
SlaveI2C_en_o	CS	GPIO_1_D33	Fixe a "1"
VDD	VIN	VCC3P3	
GND	GND	GND	

9. Recordeu que el bus I^2C les línies de rellotge i dades són bidireccionals que requereixen de resistències *pull-up*. Per tant, cal afegir la descripció Verilog dels buffers de tres estats al top del sistema.

```
assign SCL_io = sclPadEn ? 1'bz : sclPadOut;
assign SDA_io = sdaPadEn ? 1'bz : sdaPadOut;
assign sclPadIn = SCL_io;
assign sdaPadIn = SDA_io;
```

Tasques a realitzar

1. Descarregueu-vos i descomprimiu el directori lab4 del campus virtual.
2. Examineu el contingut de les diferents carpetes i fitxers.
3. Enumereu i descriviu els estats del mòdul ***bme280_reader***, i dibuixeu-ne el diagrama d'estats.
4. Feu el diagrama de blocs (no RTL) de l'arquitectura de l'estació meteorològica que implementareu, en base al requisits anteriors.
5. Feu una taula enumerant els mòduls que utilitzeu i la seva funció principal.

6. Genereu l'entitat de primer nivell (top de l'estació meteorològica) en base a l'arquitectura definida. Tant el mòdul com el fitxer s'han d'anomenar ***top_meteo_de0cv.***

   Recordeu d'utilitzar sincronitzadors per les senyals que creuen dominis de rellotge.

   Recordeu afegir la descripció Verilog dels buffers de tres estats.

   Verifiqueu l'adreça de l'esclau del vostre dispositiu.
7. Penseu i llisteu quins tests i simulacions RTL heu de fer per verificar que el sistema funciona. Especifiqueu-los per cada mòdul.
8. Verifiqueu el correcte funcionament de la vostra estació meteorològica simulant a nivell RTL els tests que heu definit abans. Podeu utilitzar el fitxer de test així com els diferents models funcionals que trobareu a les carpetes *tb* i *misc*.
9. Implementeu el disseny a la DE0-CV (Cyclone V 5CEBA4F23C7N):
   A. Afegiu el fitxer de restriccions temporals (SDC) al projecte i examineu-ne el contingut. Podeu consultar l'ajuda del mateix Quartus per saber que fa cada una de les comandes.
   B. Actualitzeu el nom de les entrades i sortides del fitxer SDC així com el nom de la instancia del PLL.
   C. Per generar la IP del PLL, heu de buscar a la catàleg d'IPs del Quartus (*Tools* > *IP Catalog* > *PLL Intel FPGA IP*).
   D. Genereu la IP a la carpeta *ips*, amb el nom de *pll_cv.v*, i configurar-lo com es mostra en la **Figura 11**.
   E. Aneu a la carpeta *ips* i comproveu els fitxers generats. Per instanciar el PLL al vostre disseny utilitzeu el fitxer *pll_cv.v*. Quan l'instancieu poseu la senyal de reset a zero.
   F. Per fer l'assignació de pins, consulteu el manual de la DE0-CV que trobareu al campus virtual.

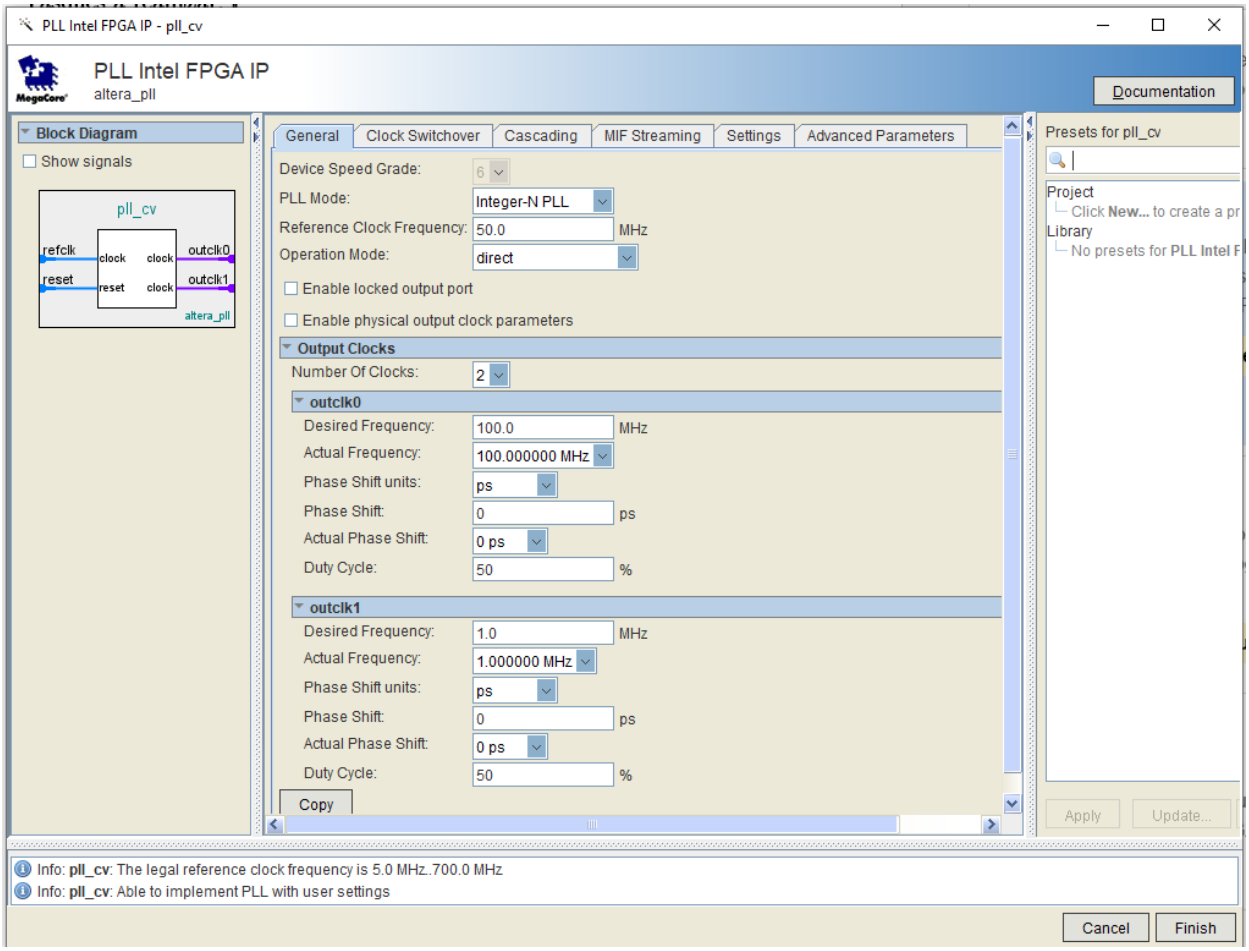

Figura 11 Paràmetres de configuració de la IP del PLL.

10. Per evitar utilitzar resistències de pull-up externes, aprofitarem els recursos dels que disposa la Cyclone V. Els *pads* de propòsit general de la Cyclone-V disposen de resistències de *pull-up* dèbils (25 kΩ) que es poden activar. Seguiu els passo descrits en l'Annex de la Pràctica 4.

11. Comproveu els missatges d'alerta i error així com l'anàlisi temporal, es compleixen els requisits temporals? Quin és el problema? Com es podria solucionar?
12. Connecteu el sensor BME280 a la placa de desenvolupament DE0-CV i comproveu que podeu comunicar-vos amb ell.

Entrega

Fer demostració al professor un cop s'ha implementat a la DE0-CV. En cas de no poder fer la demostració a l'aula, heu de gravar un vídeo que adjuntareu dins la carpeta doc.

Un fitxer zip amb el directori de treball:

1. A la carpeta **rtl**: el codi RTL, amb el *top_meteo_de0cv* i *bme280_reader* comentats.
2. A la carpeta **tb**: el codi del testbench comentat.
3. A la carpeta **sdc**: el fitxer de restriccions temporals (.sdc) comentat**.**
4. Dins la carpeta **doc** hi heu de posar l'informe complimentat que trobareu al campus virtual, que constarà de:
    - Diagrama de blocs de l'arquitectura de l'estació meteorològica implementada i descripció breu de l'arquitectura, enumerant cada mòdul i descrivint la seva funcionalitat, així com les seves entrades i sortides.
    - Diagrama d'estats del mòdul *bme280_reader*, així com una descripció breu del seu funcionament enumerant els diferents passos que fa fins a realitzar la segona tanda d'adquisició de dades.
    - La llista de tests i les captures de les simulacions RTL corresponents, amb una explicació breu i ressaltant les zones d'interès.
    - Captura (o captures) del esquema RTL generat pel Quartus (expandiu les caixetes) i taula de recursos utilitzats.
    - Preguntes que trobareu a l'informe.

Annex Pràctica 4

Sincronitzadors

Introducció

En qualsevol circuit seqüencial, quan es produeixen violacions de temps de *setup* o de *hold* en un *flip-flop*, aquest entra en un estat on la seva sortida és impredictible. D'aquest estat en diem metastable i es pot propagar per tot el disseny causant pèrdua de dades o un mal funcionament del xip. Els circuits on hi ha creuaments de rellotge, és a dir, el *Clock Domain Crossing* (CDC), són especialment susceptibles a patir metastabilitats, ja que no hi ha ningú que asseguri que es compleixen els temps de *setup* o *hold*.

Per solucionar aquest problema utilitzem sincronitzadors. Els sincronitzadors són circuits utilitzats en disseny VLSI per transferir dades de manera segura i fiable entre dominis de rellotge diferents. El sentit en què es produeix la transferència de dades (del rellotge lent al ràpid o del ràpid al lent) pot tenir implicacions diferents en la metastabilitat i les necessitats del disseny.

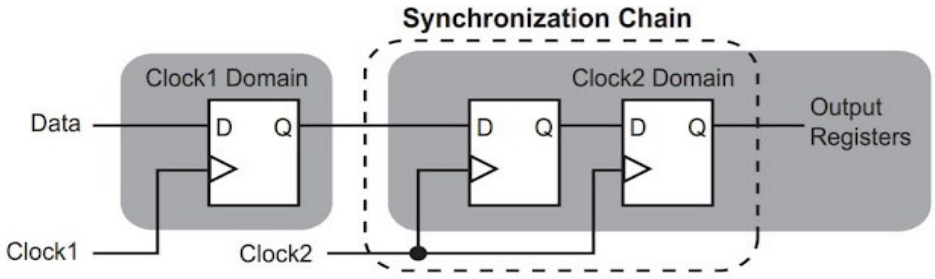

Figura 12 Sincronitzador basat en 2 *flip-flops*.

Com s'implementen en Verilog?

En *Verilog*, es poden utilitzar diferents tipus de sincronitzadors, com ara els basats en flanc de rellotge o els basats en senyal de control. Un exemple bàsic d'un sincronitzador *Verilog* que ens sincronitza una dada d'un domini de rellotge lent (*clk_1*) a un de més ràpid (*clk_2*) és el de la **Figura 12**. La seva implementació en *Verilog* podria ser:

```
module sincronitzador (
    input wire Clk_1,      // Rellotge del domini lent
    input wire Clk_2,      // Rellotge del domini ràpid
    input wire Reset,      // Senyal de reset
```

```
    input wire Data_in,       // Dades d'entrada del domini lent
    output wire Data_out      // Dades de sortida sincronitzades al domini ràpid
);
    reg data_sync1;          // Primer registre de sincronització
    reg data_sync2;          // Segon registre de sincronització

    always @(posedge Clk_1 or posedge Reset) begin
        if (Reset) begin
            data_sync1 <= 1'b0;
        end else begin
            data_sync1 <= Data_in;
        end
    end

    always @(posedge Clk_2) begin
        data_sync2 <= data_sync1;
    end

    assign Data_out = data_sync2;
endmodule
```

En aquest exemple, s'utilitzen dos registres de sincronització (*data_sync1* i *data_sync2*). Les dades d'entrada s'agafen en el primer registre mitjançant el rellotge lent (*clk_1*). Després, es traslladen al segon registre amb el rellotge ràpid (*clk_2*). La sortida sincronitzada (*data_out*) s'obté del segon registre. Aquesta estructura de doble registre garanteix que les dades es transmetin de manera segura i redueixen el risc de metastabilitat quan es passa de domini de rellotge lent a ràpid.

Habilitació de *Pull-Ups* interns

Hi ha dues maneres d'habilitar les resistències internes de pull-up al Quartus. No obstant això primer cal habilitar l'opció de *Weak Pull-Up Resistor* al nostre projecte de Quartus. Per fer-ho cal anar al menú *Assignments-> Settings* des de la finestra principal (o premeu Ctrl+Shift+E).

A la finestra emergent, seleccioneu del menu de l'esquerra l'opció *Compiler Settings* i seguidament premeu al botó *Advanced Settings (Fitter)* que trobareu a la part dreta. A la finestra emergent trobareu un llistat d'opcions. Busqueu l'opció *Weak Pull-Up Resistor* i activeu-la seleccionant *On*.

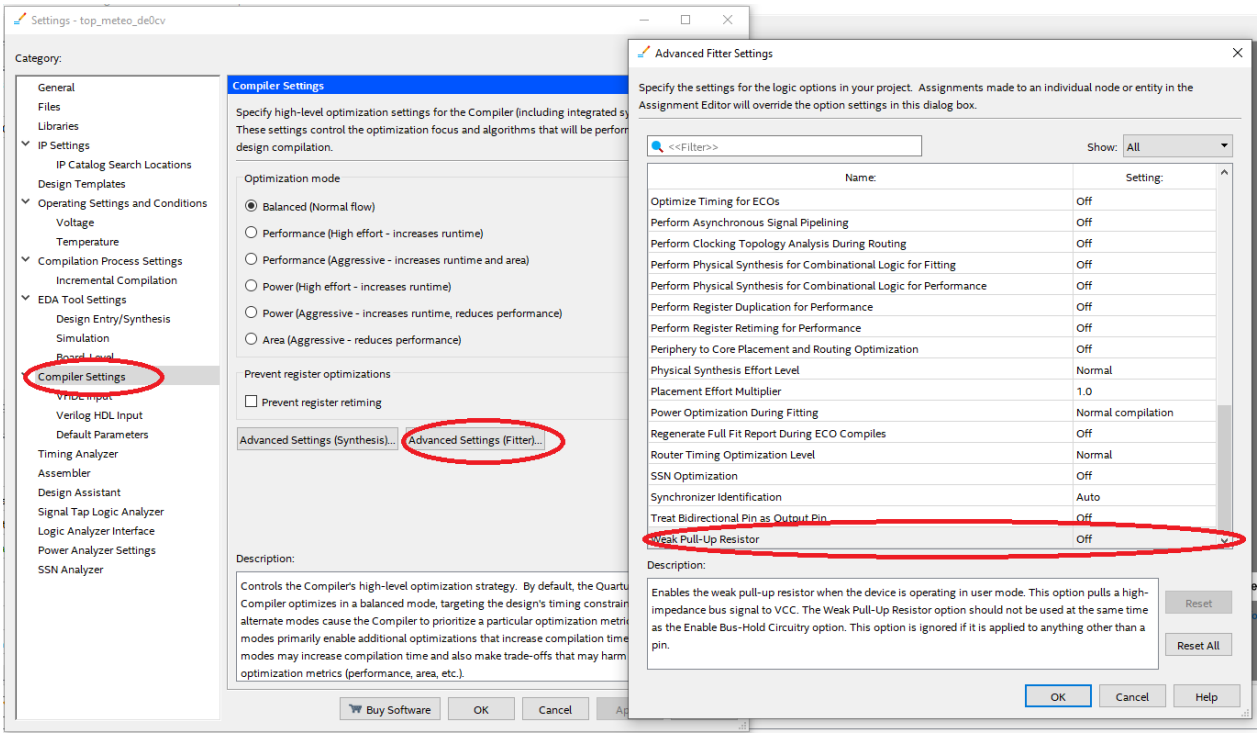

Figura 13 Habilitació de l'opció *Weak Pull-Up Resistor* al projecte de Quartus.

Un cop activada aquesta opció i durant l'assignació de pins podrem habilitar els *pull-up* pels senyals del bus I^2C *SCL_io* i *SDA_io* mitjançant el *Pin Planner* o l'*Assignment Editor* del Quartus.

Pin Planner

Des de la finestra principal del vostre projecte, aneu al menú *Assignment* > *Pin Planner* (o premeu Ctrl+Shift+N). A la finestra de planificació de pins, a la vista de *All Pins* a la part inferior, feu clic amb el botó dret del ratolí a qualsevol capçalera de columna i seleccioneu *Customise Columns*. A la finestra emergent, desplaceu-vos cap avall al costat esquerre i trobeu l'opció *Weak Pull-Up Resistor*. Seleccioneu-la i afegiu-la a les columnes visibles pitjant la bot de la fletxa cap a l adreta. Guardeu els canvis clicant al boto *Ok*.

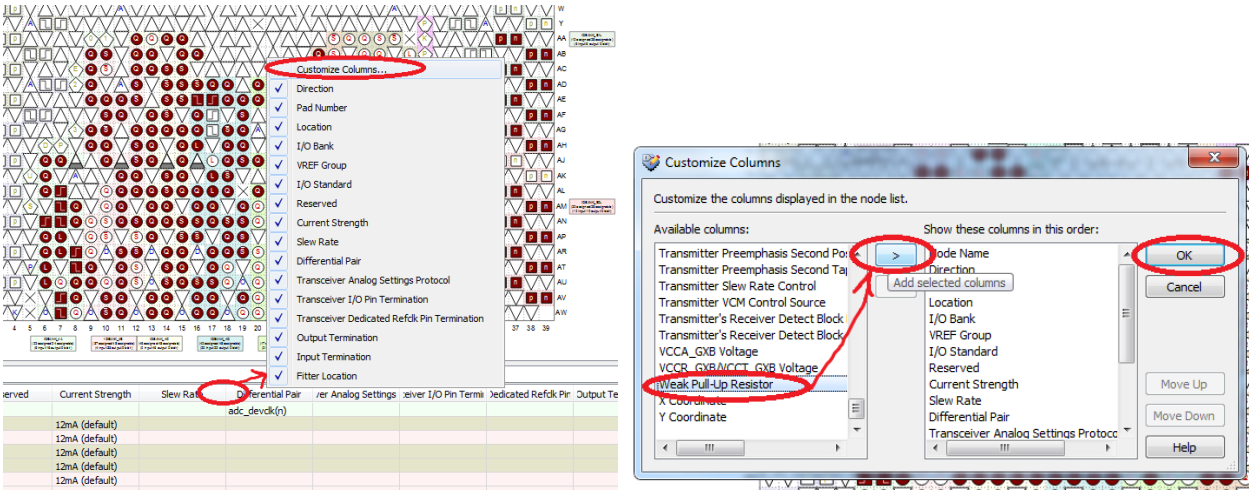

Figura 14 Personalització de les columnes del *Pin Planner*.

A la finestra de planificació de pins, hauríeu de veure una nova columna anomenada *Weak Pull-Up Resistor*. Per a cada pin que vulgueu activar el *pull-up*, simplement feu clic a la casella de la columna i seleccioneu *On*. Per defecte, està desactivada (en blanc que és equivalent a l'opció *Off*.

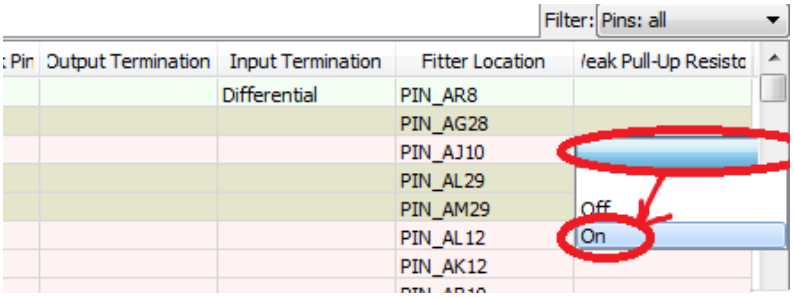

Figura 15 Columna de per activar el *pull-up* intern.

Assignment Editor

Obriu l'eina de l'editor d'assignacions a la finestra principal anant al menú *Assignment* > *Assignment Editor* (o premeu Ctrl+Shift+A).

A la part inferior de la llista d'assignacions, hi ha una fila on totes les entrades són <<*New*>>. Feu clic a la columna *Assignment Name* i seleccioneu *Weak Pull-Up Resistor*. Després, a la columna*Value*, seleccioneu *On*. Finalment, a la columna *To*, introduïu el nom

del pin (que pot incloure el caràcter comodí * per seleccionar busos sencers o fins i tot totes les senyals d'entrada i sortida del nostre projecte).

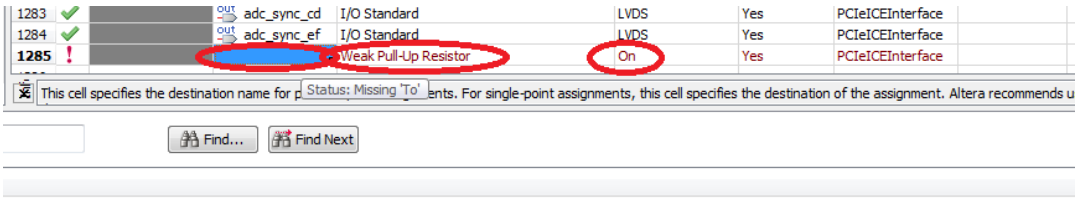

Figura 16 Activació del *Weak Pull-Up Resistor* per mitja del *Assignment Editor*.