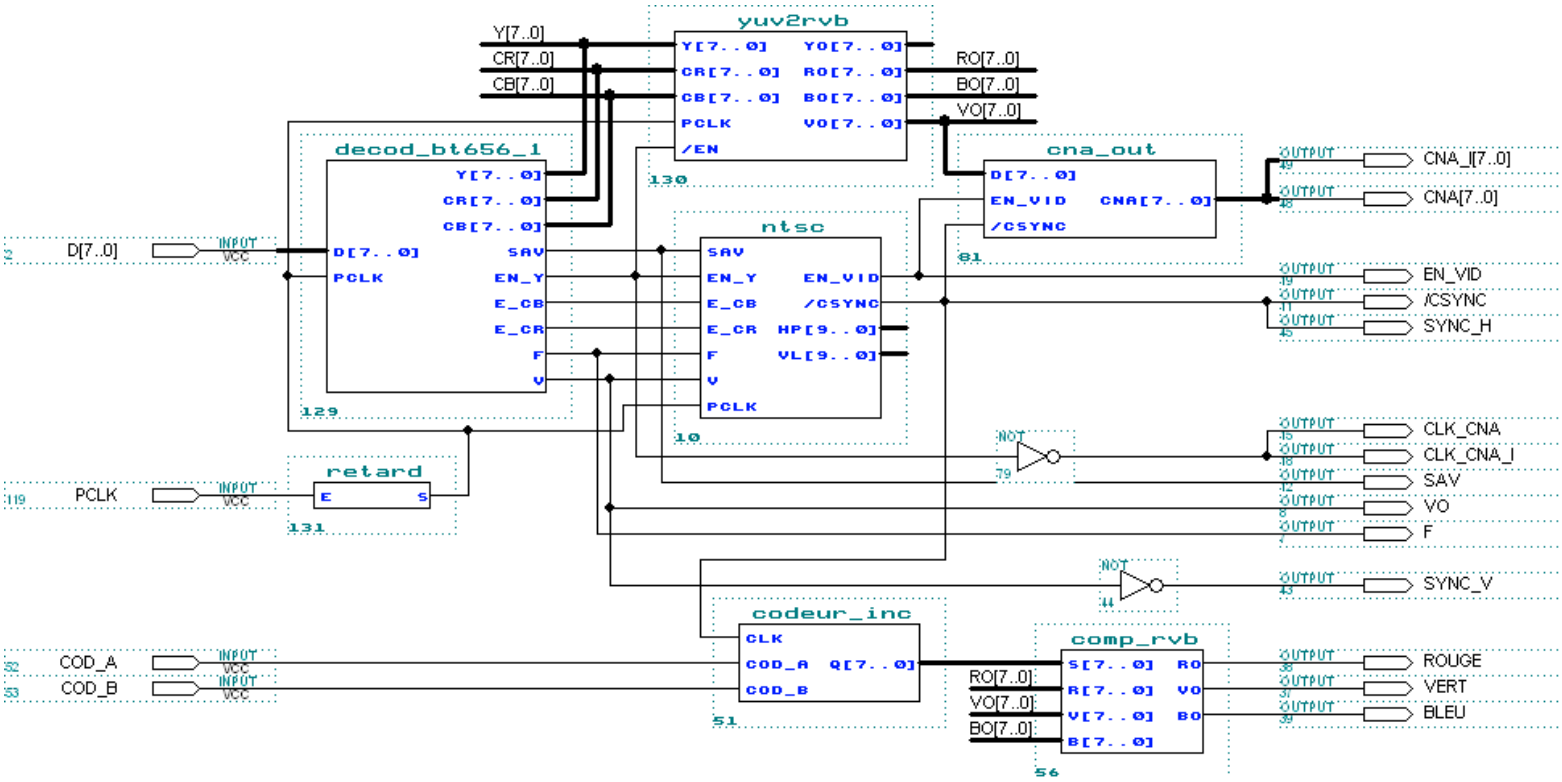
Applications maquette d'étude EP10K20

DÉMULTIPLEXEUR BT.656

SUR MODULE CAMÉRA C38A (OV7620)

SCHÉMAS ET DESCRIPTIONS AHDL

1. Schéma principal



Le démultiplexeur proprement dit est la fonction "Decod_BT656_1". Les différentes composantes du signal vidéo BT.656 sont isolées par cette fonction :

- Y[7..0] : luminance codée en binaire naturel. Échantillonnée dans un registre avec PCLK aux moments adéquats (quand EN_Y est vrai). Etendue : 16 à ~~235~~252
- CR[7..0] : différence couleur R – Y codée en C2. Échantillonnée dans un registre avec PCLK aux moments adéquats (quand E_CR est vrai). Etendue : –~~112~~105 à +~~112~~120
- CB[7..0] : différence couleur B – Y codée en C2. Échantillonnée dans un registre avec PCLK aux moments adéquats (quand E_CB est vrai). Etendue : –112 à +112
- SAV : Start Activ Video : passe à "1" pdt une période PCLK à l'apparition de ce code dans le flux BT.656 (voir norme).
- F : indicateur de trame (mémorisation du bit F de la trame) :
  - F = "0" → Field1
  - F = "1" → Field2
- V : vertical blanking (mémorisation du bit V de la trame) : à "1" pendant les lignes inactives.

A noter l'insertion de **x cellules "LCELL"** (fonction "Retard") permettant de retarder le signal PCLK et ainsi obtenir un échantillonnage correct des composantes numériques du signal BT.656 dans la fonction "Decod_BT656_1". En effet, l'EPLD utilisée sur la maquette d'étude est la plus lente de sa catégorie : EPF10K20RC240-4; les temps de propagation de base sont de l'ordre de 15nS, à comparer avec la période de PCLK : 37nS (27MHz).

Les autres fonctions permettent :

- **YUV2RVB :** de produire les signaux YO[] (luminance), RO[], VO[] et BO[] (composantes rouge, verte et bleue) codés en BN sur 8 bits à partir des composantes YUV du flux BT.656 :
  - cette fonction produit d'abord les signaux internes YI, R, V et B suivants :
    - $YI[14..0]_{C2} = 37.(Y[7..0]_{BIN} - 16)$. Note $YI[14..0]_{C2}$ est toujours positif (YI14="0")
    - $R[14..0]_{C2} = YI[14..0]_{C2} + 50.CR[7..0]_{C2}$
    - $B[14..0]_{C2} = YI[14..0]_{C2} + 64.CB[7..0]_{C2} + 1$. Note : "+1" évite les dépassements
    - $V[14..0]_{C2} = YI[14..0]_{C2} - 25.CR[7..0]_{C2} - 12.CB[7..0]_{C2}$

    Note : les opérateurs "multiplication" sont de type "câblés" et sont très gourmants en ressources matérielles. On peut utiliser les structures EAB des EPF10K20.
  - Ces signaux sont ensuite divisés par 32 et converti en BN sur 8 bits pour être enfin échantillonnés dans des registres synchrones de PCLK et validés par l'état bas de /EN. Les sorties de ces registres sont les signaux YO[], RO[], VO[] et BO[]. Ils sont codés en BN et utilisent toute l'étendue possible (0 à 255) pour représenter l'intensité de la composante.
- un affichage de l'image N&B sur un moniteur via la sortie VQ_OUT (75 Ω) de la maquette :
  - NTSC : produit (entre autres) les signaux EN_VID (à "1" pendant la fenêtre active) et /CSYNC (synchro composite simplifiée)
  - Video_out : la fonction combine l'information D[] codée en BN sur 8 bits (étendue : 0 à 255) et les signaux binaires EN_VID et /CSYNC pour produire un signal vidéo N&B composite conforme. L'information binaire peut être l'un des signaux YO[], RO[], VO[] ou BO[] produits par la fonction "YUV2RVB". Il suffit de modifier le câblage et de relancer une compilation.
  - Le signal vidéo est dupliqué sur la sortie VI_OUT (50 Ω) pour une observation à l'oscilloscope.
- un affichage en couleurs sur un écran multi-synchro (il doit accepter 525 lignes / 50Hz) connecté sur la prise DB15 de la maquette :
  - L'interface VGA de la maquette ne traite que des signaux binaires. Il faut donc convertir les signaux numériques RO[7..0], VO[7..0] et BO[7..0] en binaire. Ceci est réalisé par la fonction "Comp_RVB" qui compare chacune de ces signaux avec un seuil S[7..0] réglable pour produire les signaux binaires RO, BO et VO.
  - Le seuil S[7..0] est réglé via le codeur incrémental de la maquette et la fonction "Codeur_inc"
  - Les signaux /CYNC et V sont utilisés comme signaux de synchronisation SYNC_H et SYNC_V du moniteur multi-synchro

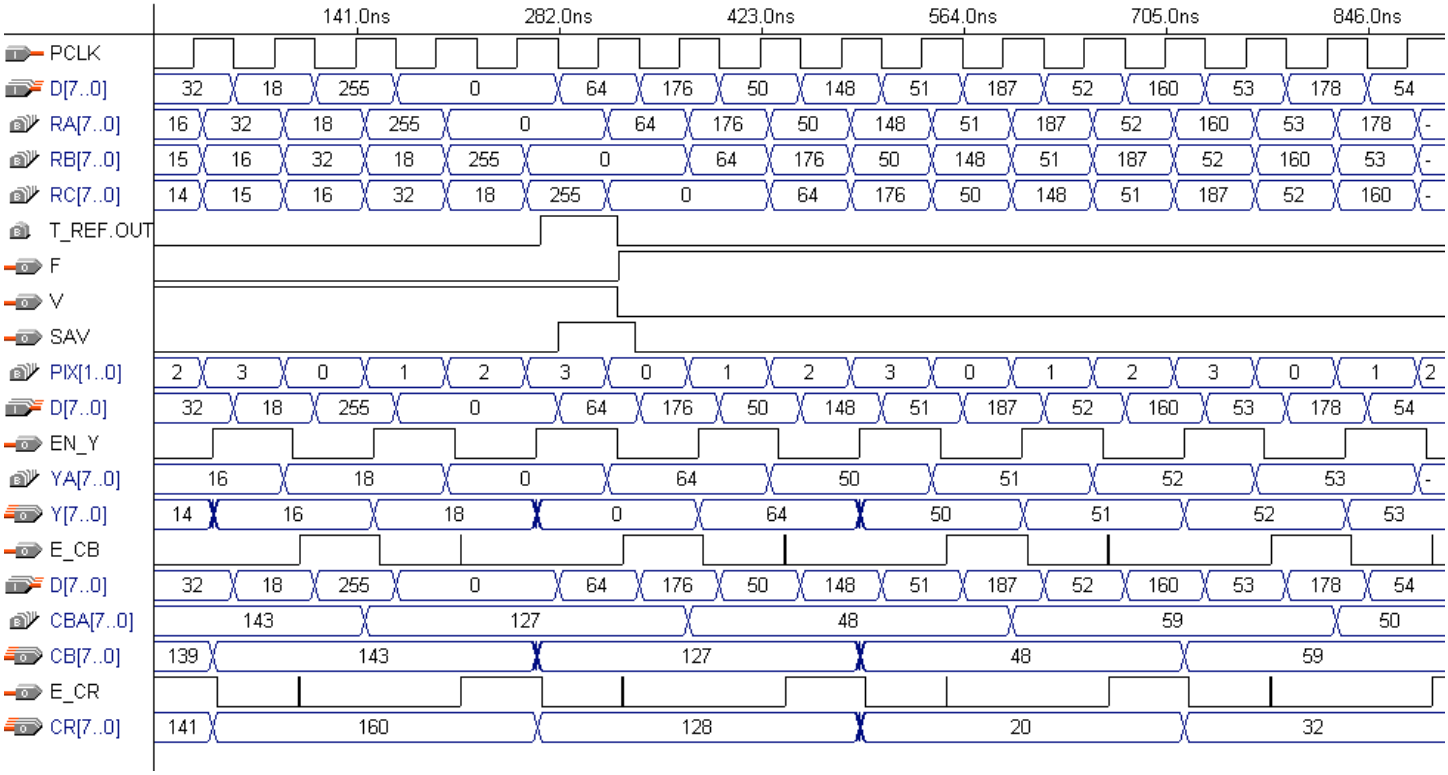
2. Decod_BT656_1

```
subdesign DECOD_BT656_1
(
 D[7..0]  : input;  -- Données BT.656
 PCLK     : input;  -- Horloge "pixels" BT.656 (27MHz)
 Y[7..0]  : output; -- Composante luminance : 13,5M échantillons/s
 CR[7..0] : output; -- Composante CR : 6,75M échantillons/s
 CB[7..0] : output; -- Composante CB : 6,75M échantillons/s
 SAV      : output; -- Start Activ Video
                    -- Passe à "1" pdt une période PCLK à l'apparition
                    -- de ce code dans le flux BT.656 (voir norme).
 EN_Y     : output; -- A "1" quand D[] transmet Y[]
 E_CB     : output; -- A "1" quand D[] transmet CB[]
 E_CR     : output; -- A "1" quand D[] transmet CR[]
 F        : output; -- Indicateur de trame (mémorisation du bit F de la trame)
                    -- "0" -> Field1, "1" -> Field2
 V        : output; -- Vertical blanking (mémorisation du bit V de la trame)
                    -- à "1" pendant les lignes inactives.
)
variable
 RA[7..0], RB[7..0], RC[7..0] : DFF; -- Constituent un registre à décalage
                                     -- des 3 derniers échantillons de D[]
 Y[7..0], CR[7..0], CB[7..0] : DFFE; -- Composantes extraites du signal BT.656
 YA[7..0], CBA[7..0]         : DFFE; -- Pour mettre en phase Y[] et CB[]
                                     -- avec CR[]
```

```
 F, V      : DFFE; -- Voir ci-dessus
 PIX[1..0] : DFF ; -- Compteur pixel modulo 2. Utilisé pour identifier les
                   -- composantes dans le flux BT.656
 T_REF     : node; -- Passe à "1" pdt 1 période PCLK juste après la séquence
                   -- D[] = H"FF" , H"00" et H"00"
                   -- Identifie donc dans le flux BT.656 l'octet regroupant
                   -- les bits SAV ou EAV, F , V (voir norme)
begin
 RA[].clk=PCLK; RB[].clk=PCLK; RC[].clk=PCLK; RC[].clk=PCLK;
 RA[].d=D[]; RB[].d=RA[].q; RC[].d=RB[].q;
 T_REF=(RA[].q==0) and (RB[].q==0) and (RC[].q==255);
 F.clk=PCLK; F.ena=T_REF; F.d=D6; -- F est le bit D6 de l'octet identifié par T_REF
 V.clk=PCLK; V.ena=T_REF; V.d=D5; -- V est le bit D5 de l'octet identifié par T_REF
 SAV=!D4 & T_REF;                 -- H est le bit D4 de l'octet identifié par T_REF
                                  -- Son état défini SAV ou EAV
 PIX[].clk=PCLK;
 if T_REF then PIX[].d=0;         -- Raz synchrone du compteur PIX[] à l'apparition
          else PIX[].d=PIX[].q+1; -- de T_REF
 end if;
 EN_Y=PIX0.q; E_CB=PIX[].q==0; E_CR=PIX[].q==2; -- Après T_REF, la séquence du flux
                                                -- BT.656 est CB, Y, CR, Y
 Y[].clk=PCLK; YA[].clk=PCLK;
 CR[].clk=PCLK; CB[].clk=PCLK; CBA[].clk=PCLK;
 YA[].ena=EN_Y; YA[].d=D[];                      -- Echantillonnage de Y en binaire
 Y[].ena=B"1"; Y[].D=YA[].q;                     -- Retard d'une période PCLK
 CBA[].ena=E_CB; CBA[6..0].d=D[6..0]; CBA7.d=!D7; -- Echantillonnage de CB en C2
 CB[].ena=E_CR; CB[].d=CBA[].q;                  -- Retard de 2 périodes PCLK
 CR[].ena=E_CR; CR[6..0].d=D[6..0]; CR7.d=!D7;   -- Echantillonnage de CR en C2
end;
```

Les explications se trouvent dans les commentaires de la description.

Résultats de simulation (attention : débit incorrect) :

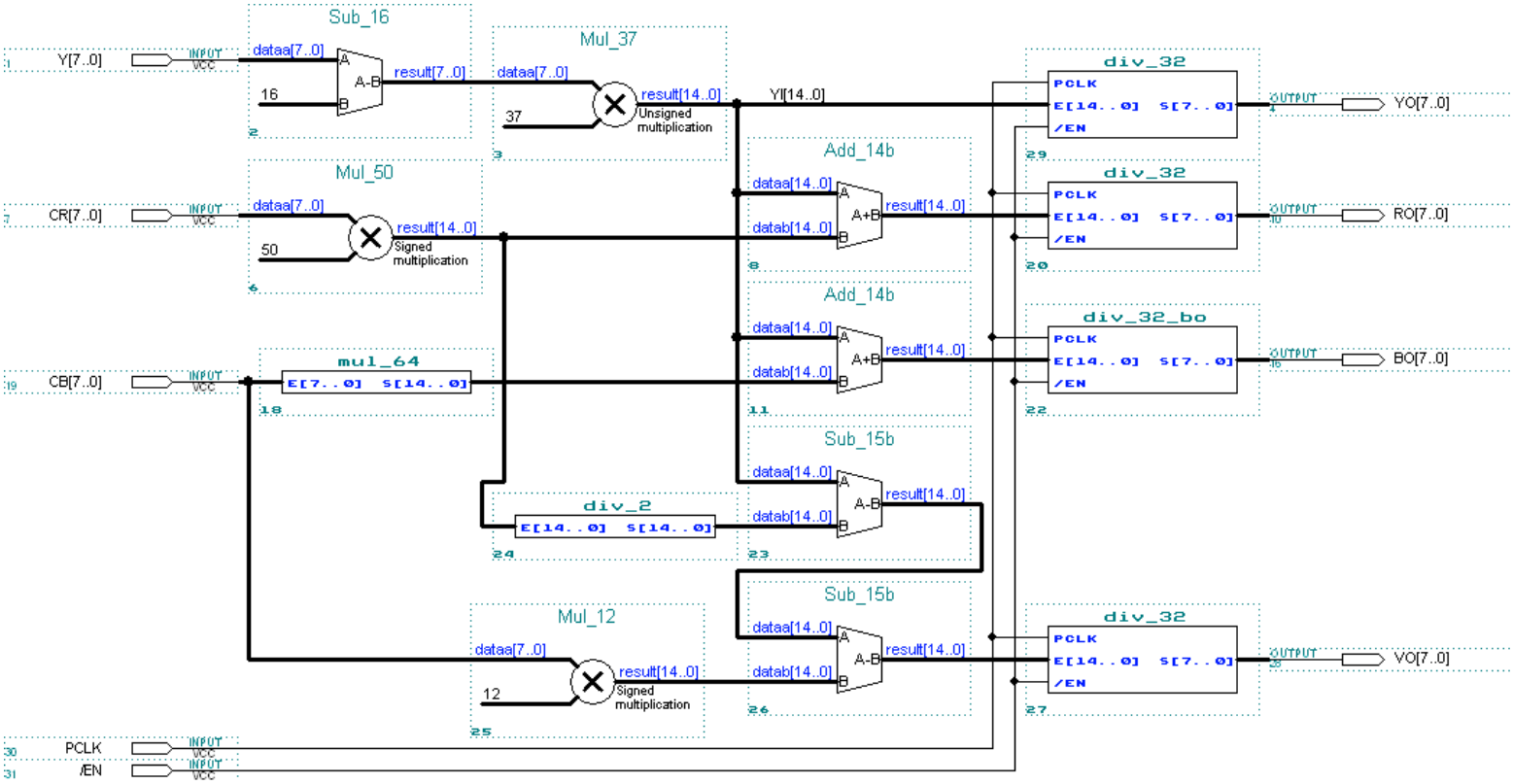


- RA[], RB[] et RC[] copient D[] avec un retard d'une période PCLK
- T_REF passe à "1" à la réception de la séquence "FF"-"00"-"00" : c'est la référence temporelle du flux BT.656
- L'échantillon qui suit cette séquence vaut 64 dans cet exemple, soit en binaire : 0100 0000. Le bit F est donc à "1" et les bits V et H sont à "0". Cela signifie que la référence temporelle T_REF correspond au début (H="0" → SAV) d'une ligne active (V="0") de la trame 2 (F="1").

- Les signaux F, V et SAV produits par la fonction sont donc conformes.
- Le compteur PIX est incrémenté modulo 4 à chaque ↑ de PCLK. Il est mis à 0 de façon synchrone qd T_REF="1", mais il est déjà synchronisé dans cet exemple (état précédent = 3).
- EN_Y passe à "1" pour les valeurs impaires de PIX et identifie bien les échantillons Y dans la trame BT.656. Dans cet exemple ils prennent les valeurs 50, 51, 52, 53, …
- Ces échantillons Y sont mémorisés successivement dans les registres synchrones 8 bits YA et Y (le 1° validé par EN_Y) pour introduire un retard volontaire de 1 période PCLK.
- E_CB passe à "1" pour les valeurs 0 de PIX et identifie ainsi les échantillons CB du flux. Dans cet exemple, ils prennent les valeurs :+48, +59, +50, … codés en Bd (soit 176, 187, 178, … en binaire).
- Ces échantillons CB sont mémorisés **en code C2** successivement dans les registres synchrones 8 bits CBA et CB (le 1° validé par E_CB, le 2° par E_CR) pour introduire un retard volontaire de 2 périodes PCLK.
- E_CR passe à "1" pour les valeurs 2 de PIX et identifie ainsi les échantillons CR du flux. Dans cet exemple, ils prennent les valeurs :+20, +32, … codés en Bd (soit 148, 160, … en binaire).
- Ces échantillons CR sont mémorisés **en code C2** dans le registre synchrone 8 bits CR validé par E_CR).
- Les retards insérés dans les flux Y (1 période PCLK) et CB (2 périodes PCLK) permettent de synchroniser les 3 flux Y, CB et CR. On constate bien que l'information "différence couleur" est la même pour les 2 premiers pixels et que l'information luminance est distincte.
- Le format "4:2:2" est respecté.

3. YUV2RVB

3.1 Schéma



Les informations appliquées aux fonctions DIV_32 sont, de haut en bas :

- $YI[14..0]_{C2} = 37.(Y[7..0]_{BIN} - 16)$
- $R[14..0]_{C2} = YI[14..0]_{C2} + 50.CR[7..0]_{C2}$
- $B[14..0]_{C2} = YI[14..0]_{C2} + 64.CB[7..0]_{C2} + 1$. Note : "+1" évite les dépassements
- $V[14..0]_{C2} = YI[14..0]_{C2} - 25.CR[7..0]_{C2} - 12.CB[7..0]_{C2}$

Ces relations respectent la recommandation BT.601 de quantification des échantillons Y, CR et CB du flux BT.656 (sauf le "+1" de B qui évite les dépassements).

~~1.2~~3.2 Fonctions DIV_32 et DIV_32_BO

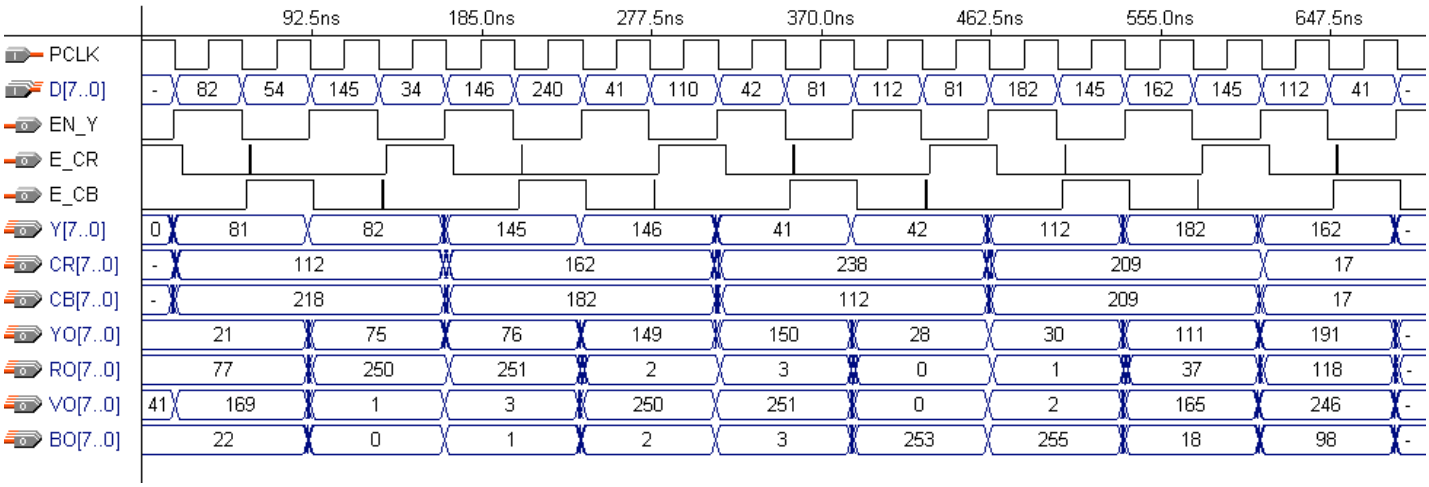
```
subdesign Div_32
(
 PCLK     : input;
 E[14..0] : input;
 /EN      : input;
 S[7..0]  : output;
)
variable S[7..0] : dffe;
begin
 S[].clk=PCLK; S[].ena=!/EN; -- Echantillonnage le + tard possible
 S[7..0].d=E[12..5];         -- Division par 32 (résultat tjrs > 0)
end;
```

```
subdesign Div_32_BO
(
 PCLK     : input;
 E[14..0] : input;
 /EN      : input;
 S[7..0]  : output;
)
variable S[7..0] : dffe;
begin
 S[].clk=PCLK; S[].ena=!/EN; -- Echantillonnage le + tard possible
 S[7..0].d=E[12..5]+1;       -- Division par 32 et "+1"
end;
```

Les 2 fonctions sont identiques en dehors de l'opération "+1".

~~1.3~~3.3 Résultats de simulation

La séquence d'échantillons est réalisée **cette fois avec le débit réel de 27M échantillons/s**.



On identifie les informations des 4 premiers pixels de la ligne représentée :

- Premier pixel : Y = 81, CR = +112, CB = –38 (218 en BN),
- Deuxième pixel : Y = 82, CR = +112, CB = –38
- Troisième pixel : Y = 145, CR = –94 (162), CB = –74 (182),
- Quatrième pixel : Y = 146, CR = –94 (162), CB = –74 (182),

Les multiplicateurs et additionneurs de la fonction YUV2RVB sont suffisamment rapides vis à vis de la période des pixels (2/27MHz = 74nS). L'entrée Y est affectée à chaque ↑ de PCLK et CR et CB le sont aux ↑ de PCLK quand E_CR="1"; les résultats des calculs sur ces entrées sont alors échantillonnées dans les registres YO, RO, VO et BO au ↑ suivant de PCLK.

On vérifie :

- Premier pixel : YO = [37.(Y-16)]/32 = [37.(81–16)]/32 = **75**

RO = [37.(Y-16)+50.CR]/32 = [37.(81-16)+50.112]/32 = **250**
VO = [37.(Y-16)–25.CR–12.CB]/32 = [37.(81-16)–25.112+12.38]/32 = **1**
BO = [37.(Y-16)+64.CB]/32+1=[37.(81-16)–64.38]/32+1=**0**

- Troisième pixel : YO = [37.(145–16)]/32 = **149** ; RO = [37.(145-16)–50.94]/32 = **2**
  VO = [37.(145-16)+25.94+12.74]/32 = **250**; BO = [37.(145-16)–64.74]/32+1=**2**

Note importante : Les signaux RO, BO et VO changent d'état pratiquement simultanément pour chaque pixel, ce qui peut être utile à certains usages de ces signaux (affichage par exemple).

4. NTSC

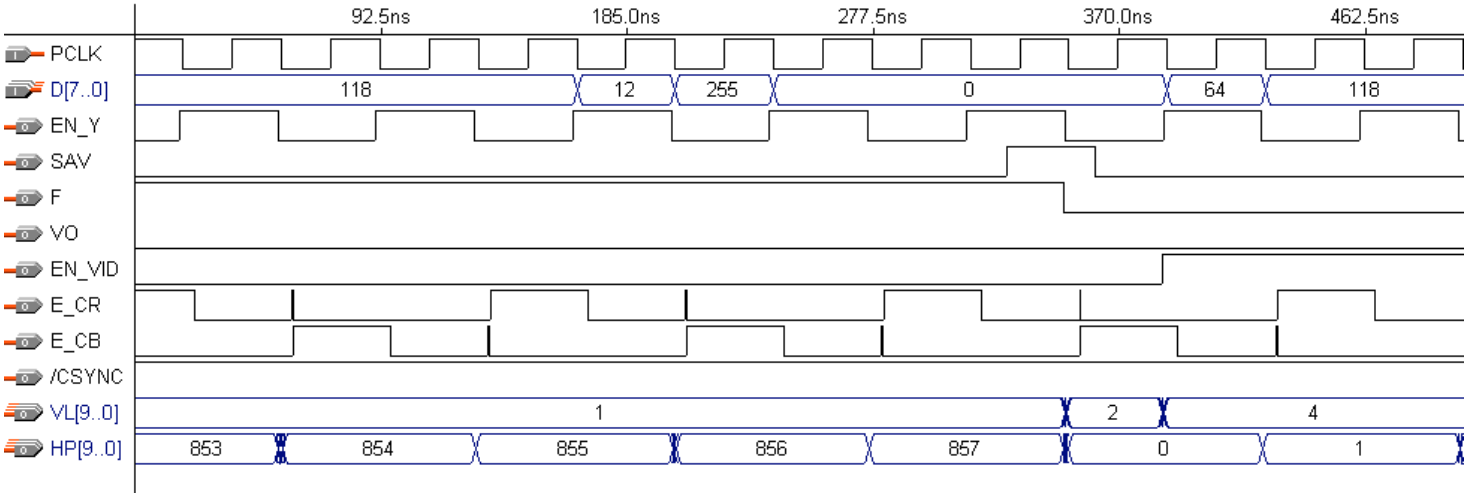
Cette fonction produit le signal de synchronisation composite /CSYNC, le signal de validation de la fenêtre vidéo EN_VID (en horizontal et vertical) et les compteurs "pixels" (HP) et lignes (HV)
Ces compteurs permettent, le cas échéant, d'adresser une mémoire d'image.

```
subdesign NTSC
(
 SAV : input;  -- Start Activ Video
               -- Passe à "1" à chaque ligne pdt une période PCLK à l'apparition
               -- de la trame de réf. dans le flux BT.656 (voir norme).
 EN_Y : input; -- Indique les intervalles de temps Y dans le flux BT.656
 E_CB : input; -- Indique les interv de tmps CB dans BT.656 (non utilisé)
 E_CR : input; -- Indique les interv de tmps CR dans BT.656 (non utilisé)
 F : input;    -- Indicateur de trame (Field1/2 de BT.656)
 V : input;    -- Vertical Blanking. A "1" pdt les lignes inactives
 PCLK : input; -- Horloge "pixel" du bus BT.656
 EN_VID  : output; -- Vidéo active
 /CSYNC  : output; -- Synchronisation composite "simplifiée"
 HP[9..0]: output; -- Compteur pixels H modulo 858
 VL[9..0]: output; -- Compteur lignes modulo 525
)
variable
  HP[9..0] : DFFE; -- Compteur pixel H modulo 858
  VL[9..0] : DFFE; -- Compteur ligne modulo 525
  F_START : DFF;   -- Pour détecter un changement d'état
  EN_VID, /CSYNC : DFF; -- Voir ci-dessus
begin
 HP[].clk=PCLK; HP[].ena=EN_Y or SAV;
 if SAV then HP[].d=0; -- Raz synchrone a l'apparition de SAV
                       -- HP[] est ainsi conforme aux chronogrammes typiques
                       -- de la norme BT.656 (premier pixel : H[]=0)
        else if HP[].q==857 then HP[].d=0; -- Comptage modulo 858
                            else HP[].d=HP[].q+1;
             end if;
 end if;
 VL[].clk=PCLK; VL[].ena=(HP[]==857) or (!F and F_START.q);
 -- Compteur VL[] incrémenté à chaque fin de ligne et affectation qd F passe à "0"
 F_START.clk=PCLK; F_START.d=F;     -- Pour détecter un changement d'état de F
 if !F and F_START.q then VL[].d=4; -- Pour être conforme à la norme
                     else if VL[].q==524 then VL[].d=0; -- Comptage modulo 525
                                         else VL[].d=VL[].q+1;
                          end if;
 end if;
 EN_VID.clk=PCLK; EN_VID.d=(HP[]<720) and !V;
 -- Fenêtre active : V="0" et HP[] entre 0 et 719
 /CSYNC.clk=PCLK;
 /CSYNC.d=!(V or (HP[]>=720+19 and HP[]<=720+19+52));
 -- Synchro composite simplifiée : à "0" si
 --    - V="1" ou
 --    - HP[] entre 739 et 791 soit 1,4µS après le dernier pixel
 --                                 et pendant 3,85µS
end;
```

Les explications se trouvent dans les commentaires de la description.

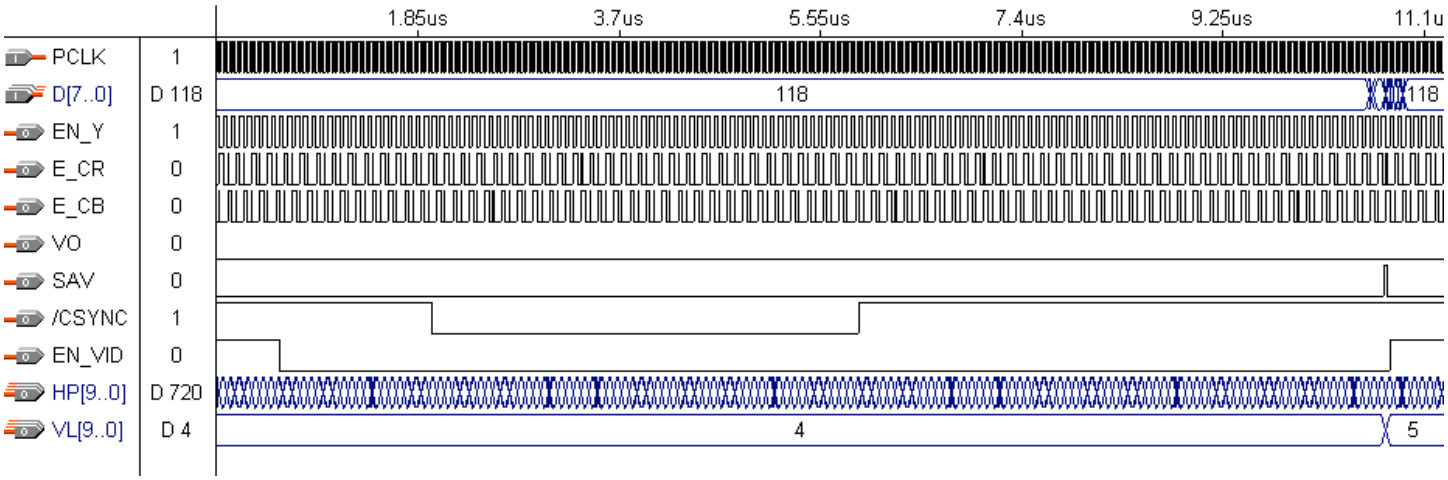
Résultats de simulation :

1. Incrémentation et initialisation des compteurs HP et VL :



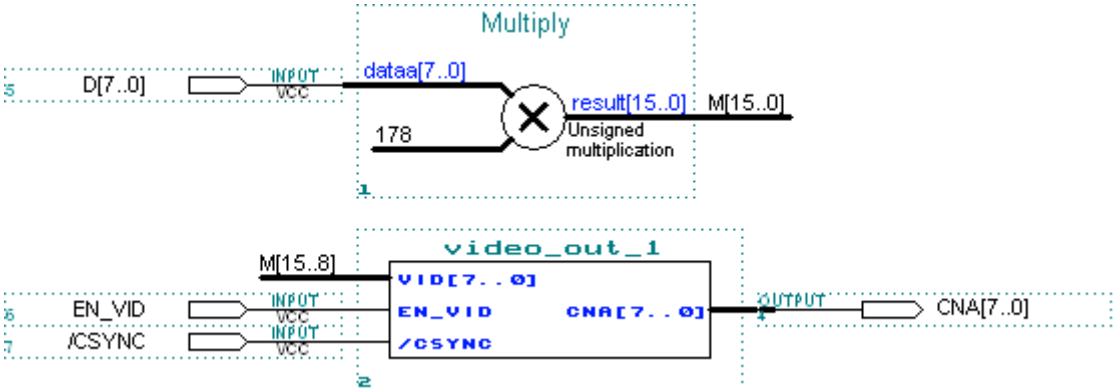
- SAV passe à "1" à la réception de la séquence "FF"-"00"-"00"-"00". Le dernier octet est caractérisé par D4=0 (soit SAV=1, voir §2) et D6=0, qui correspond au bit F. Le signal F produit par la fonction "Decod_BT656" mémorise l'état de ce bit et passe donc à "0".
- Le compteur HP est incrémenté à chaque ↑ de PCLK quand EN_Y est à "1". L'information HP représente donc la position horizontale du "pixel". C'est un compteur modulo 858, mais il est de toute façon remis à 0 à l'apparition du signal SAV pour le synchroniser sur le flux BT.656.
- Le compteur VL est incrémenté qd HP passe de 857 à 0. Mais il est aussi initialisé à 4 de façon synchrone quand F passe à "0". C'est le cas ci-dessus. On respecte ainsi la numérotation des lignes de la norme BT.656.

2. Signaux /CSYNC et EN_VID



- EN_VID est à "1" qd HP est compris entre 0 et 719
- /CSYNC est à "0" qd HP est compris entre 739 et 791, soit pendant 3,85µS à chaque ligne active. /CSYNC est aussi à "0" qd VO="1" (non représenté), ce qui permet une synchro trame simplifiée.

5. CNA_out



~~1.1~~5.1 Multiply

L'équivalent décimal des 8 bits MSB du résultat (M[15..8]) varie de 0 à 177 quand celui de D[] varie de 0 à 255. Cette caractéristique est nécessaire à la fonction "Video_out_1".

~~1.2~~5.2 Video_out_1

```
subdesign VIDEO_OUT_1
 (
  VID[7..0] : input; -- Luminance codée sur 8 bits : de 77 à 255
  EN_VID    : input; -- Fenêtres de validation
  /CSYNC    : input; -- Synchro composite
  CNA[7..0] : output;
 )
begin
 if EN_VID then CNA[]=VID[]+77;
           else if /CSYNC then CNA[]=77;
                          else CNA[]=0;
                end if;
 end if;
end;
```

La structure de la sortie VQ_OUT de la maquette produit V_{Q_OUT} = 0V qd CNA[] = 0 et V_{Q_OUT} = 1V quand CNA[] = 255.
Les valeurs affectées à CNA[] permettent de produire un signal vidéo analogique conforme : hauteur de la composante "synchro" = 0,3V (CNA[]=77) et hauteur de la composante utile = 0,7V (255-77=178).

RELEVÉS EXPÉRIMENTAUX

Conditions :

- Carte d'étude n° 1
- Horloge : HP8647A réglé sur 35,46895MHz et +8dBm. Liaison capactive sur "Global Clock"
- Sortie oscillo sur VI_OUT chargée avec 50Ω

1. Fonction sync_ccir_pal