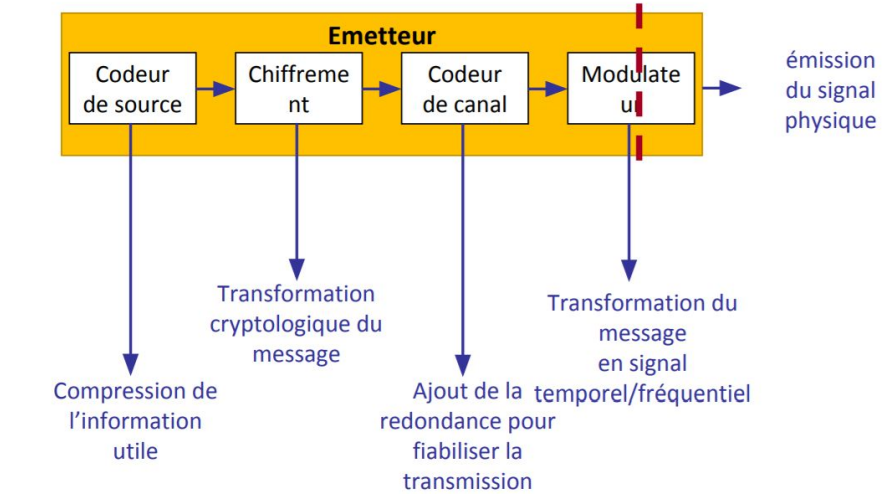
Co-simulation VHDL/MATLAB d'une chaîne de communication numérique OFDM

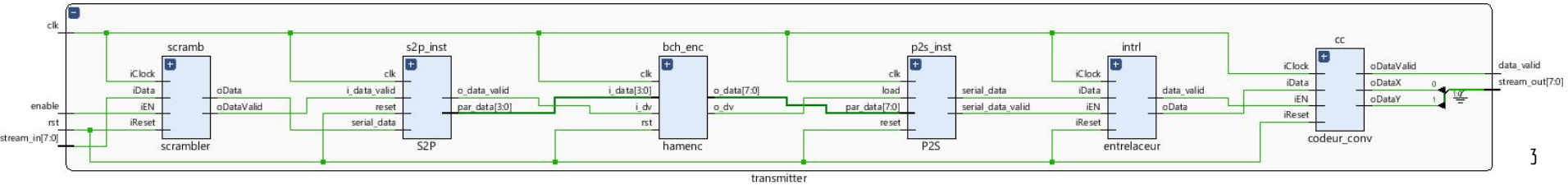
CHOLLET Benjamin -- CLAIN Adrien

Table des matières

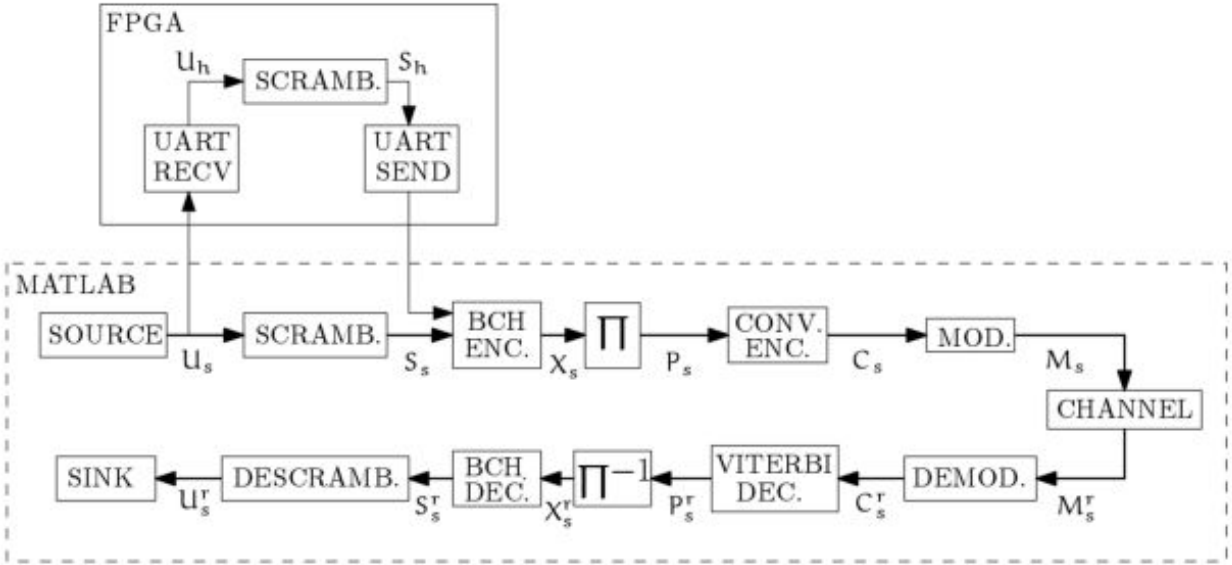
Objectifs du projet

La chaîne d'émission: mise en forme de l'information





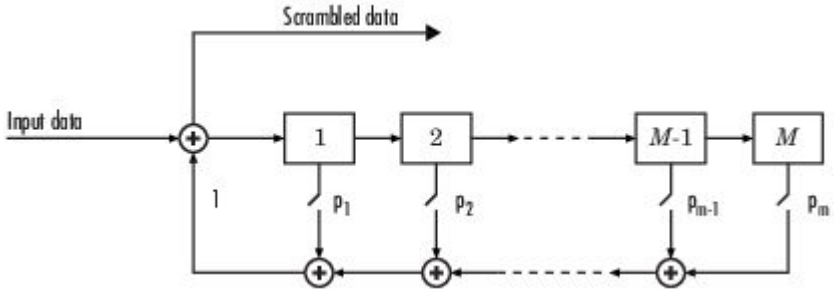
Partie 1 : Co-Simulation Matlab/VHDL



Scrambler

1) Etude du code Matlab

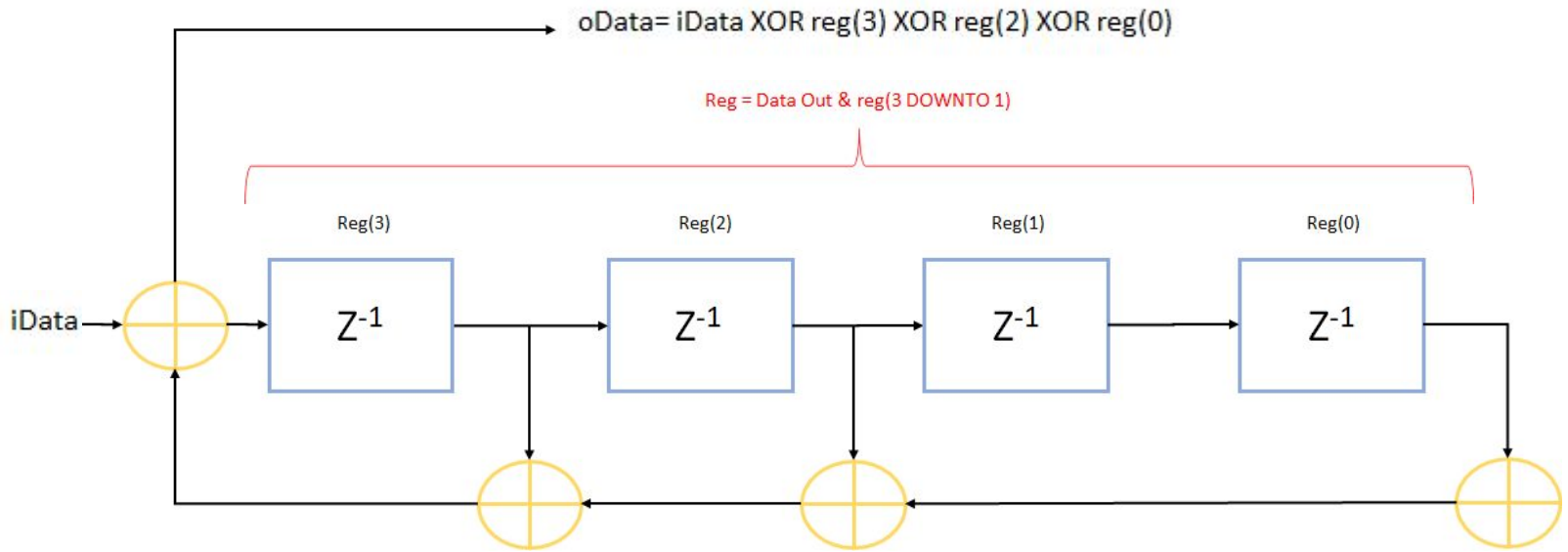
```
scrambler = comm.Scrambler(base,poly,cond)
```



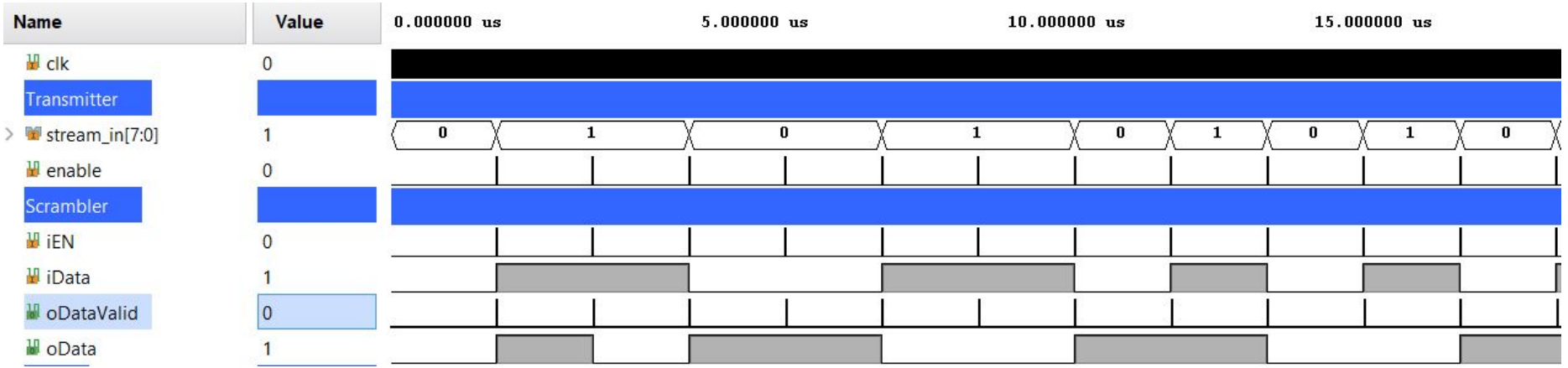

```
%% Scrambler parameters
scramb_polynomial=[1 1 1 0 1]; % 1 + z^-1 + z^-2 + 0 + z^-4
scramb_init_state=[0 0 0 0]; %condition initiale mise à zéro
Scrambler_U_obj=comm.Scrambler(2,scramb_polynomial,scramb_init_state);
```

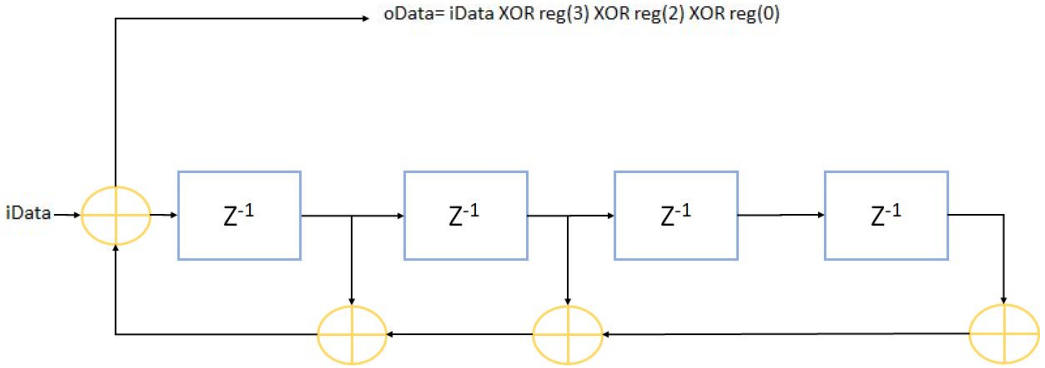
Scrambler

2) Description VHDL



Scrambler


Name
Value
0.000000 us
5.000000 us
10.000000 us
15.000000 us
clk
0
Transmitter
stream_in[7:0]
1
0
1
0
1
0
1
0
1
0
enable
0
Scrambler
iEN
0
iData
1
oDataValid
0
oData
1


oData= iData XOR reg(3) XOR reg(2) XOR reg(0)
iData
Z-1
Z-1
Z-1
Z-1

Codeur BCH

1) Etude du code Matlab

```
X_gf_soft = bchenc(gf(reshape(S_soft, bch_k, bch_cwd_nb).',1), bch_n, bch_k);
```

```
G1 = bchenc(gf([1 0 0 0]), bch_n, bch_k);
G2 = bchenc(gf([0 1 0 0]), bch_n, bch_k);
G3 = bchenc(gf([0 0 1 0]), bch_n, bch_k);
G4 = bchenc(gf([0 0 0 1]), bch_n, bch_k);
```

1	0	0	0	1	0	1
0	1	0	0	1	1	1
0	0	1	0	1	1	0
0	0	0	1	0	1	1

Codeur BCH

(1) MSB
(7) LSB

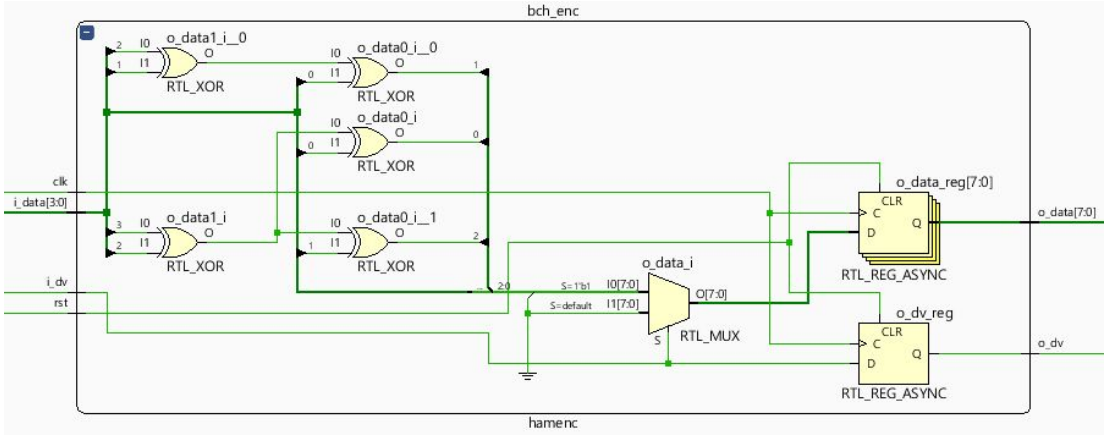
2) Etude du codeur

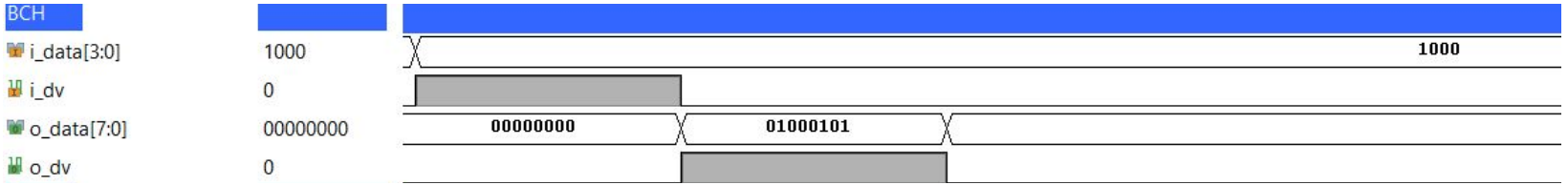
$$\begin{matrix} A & B & C & D \end{matrix} \begin{pmatrix} 1 & 0 & 0 & 0 \end{pmatrix} \times \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{matrix} \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

(1) = A
(2) = B
(3) = C
(4) = D
(5) = A+B+C
(6) = B+C+D
(7) = A+B+D

Codeur BCH

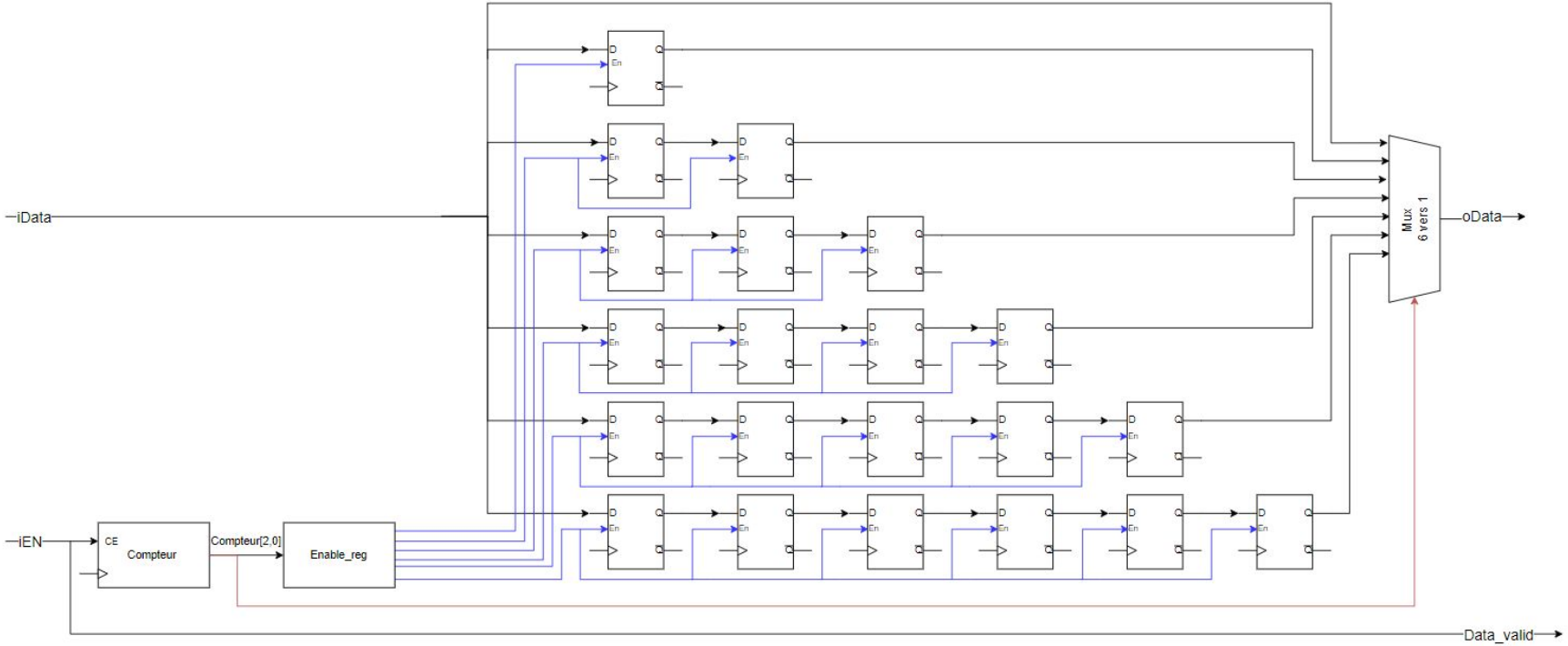
3) Implémentation VHDL & Chronogramme



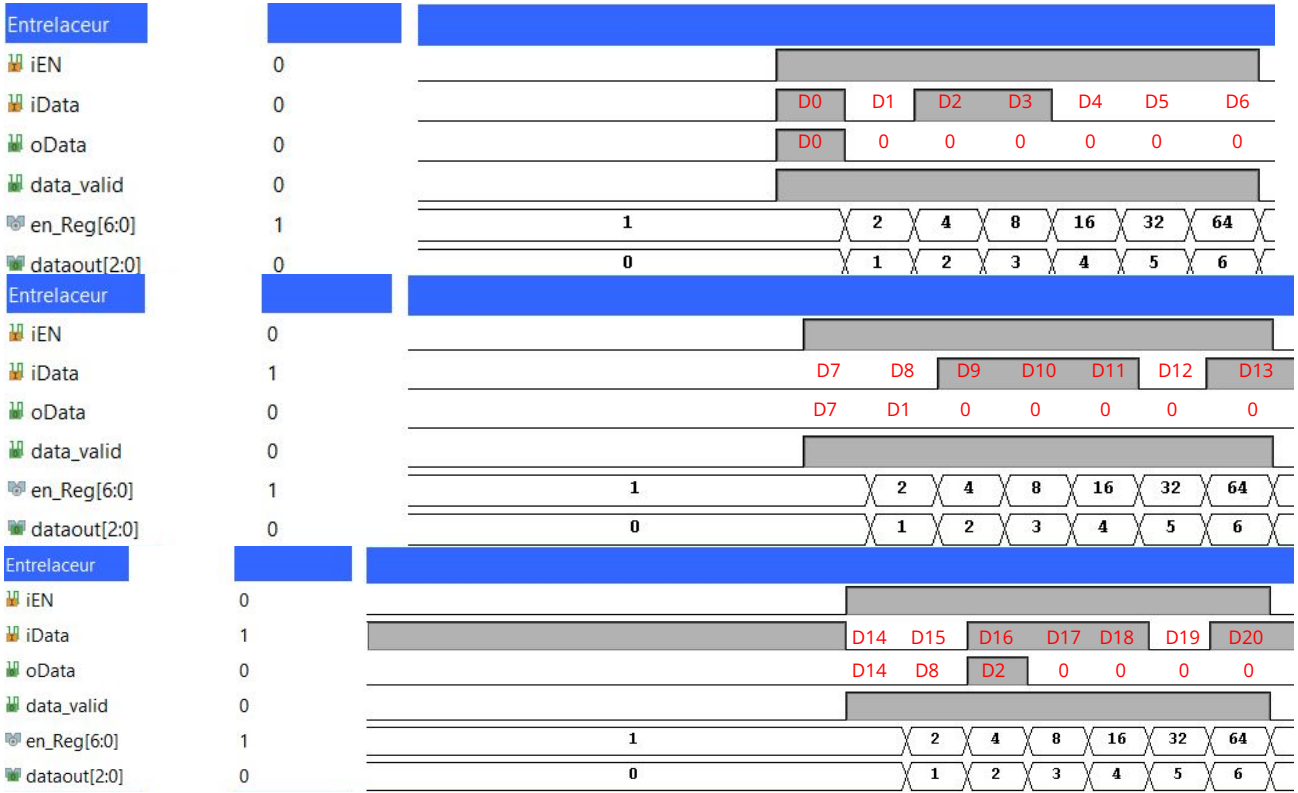


Entrelaceur

1) Implémentation VHDL & Chronogramme



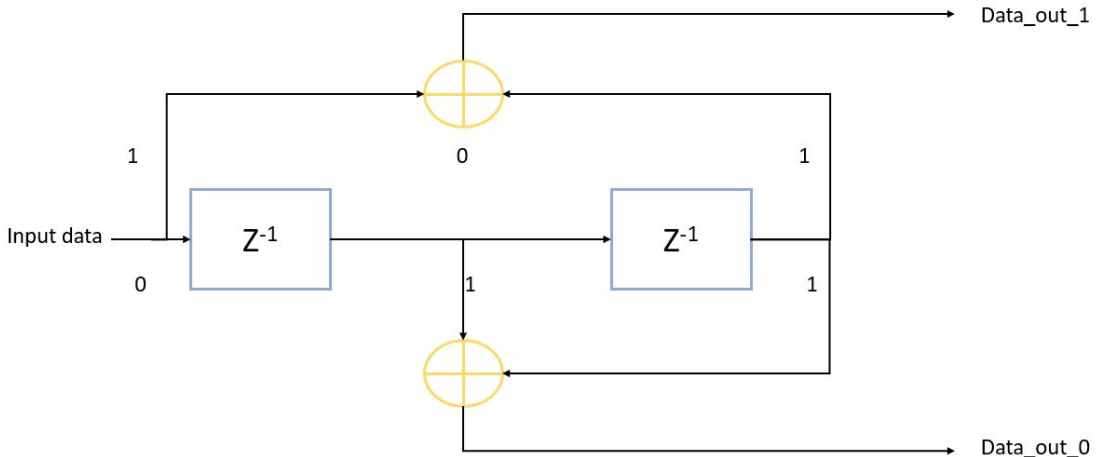
Entrelaceur



Codeur Convolutif

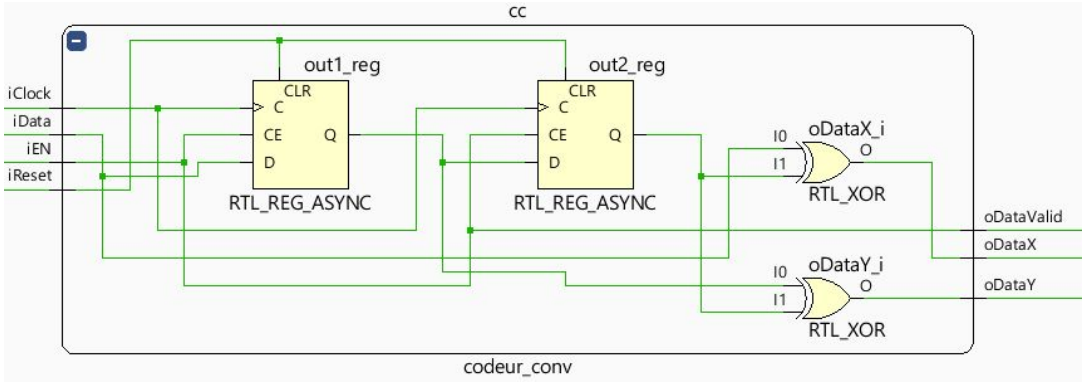
1) Etude du code matlab

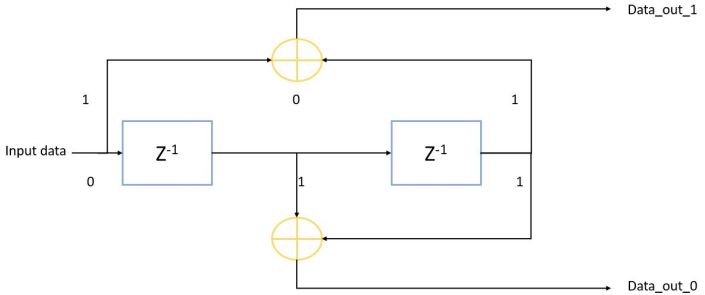
```
trellis = poly2trellis(3,[5 3]); % generator polynomial : (15,13)

C_soft = convenc(P_soft,trellis);
```

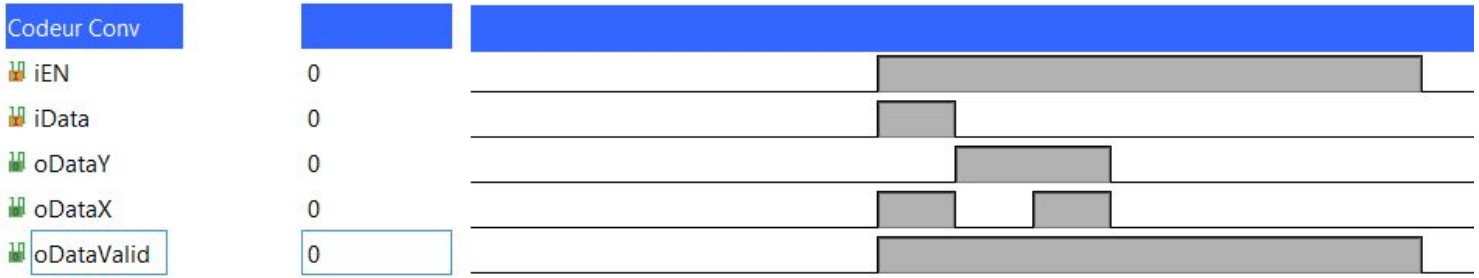


Codeur Convolutif

2) Implémentation VHDL & Chronogramme







Partie 2 : Génération chaîne de communication OFDM sur Matlab

1) Mise en place du modulateur/démodulateur sur canal AWGN

Données à disposition :

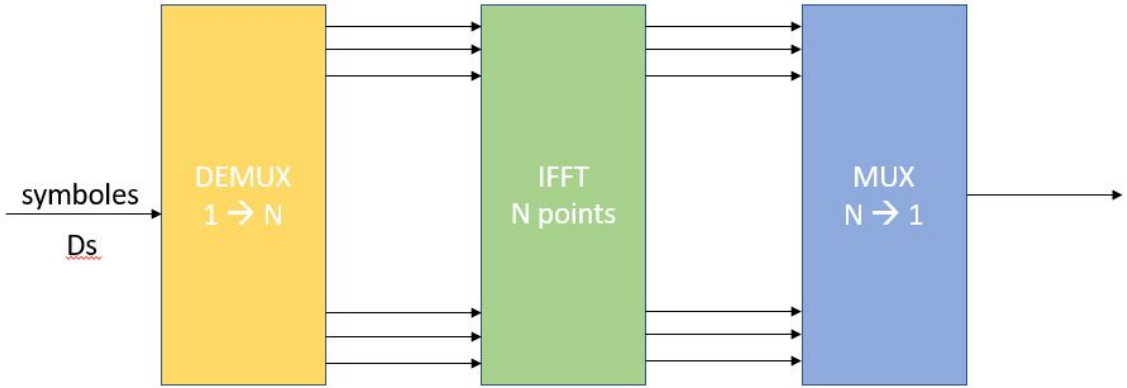
- 2 bits / symboles (modulation QPSK)
- 448 bits à transmettre
- 64 sous-porteuses
- ⇒ 3,5 symboles / sous-porteuses

Modulateur OFDM

Au niveau modulateur :

- Mélangeurs
- Filtres adaptés
- Oscillateurs locaux

Peuvent être remplacés par une IFFT



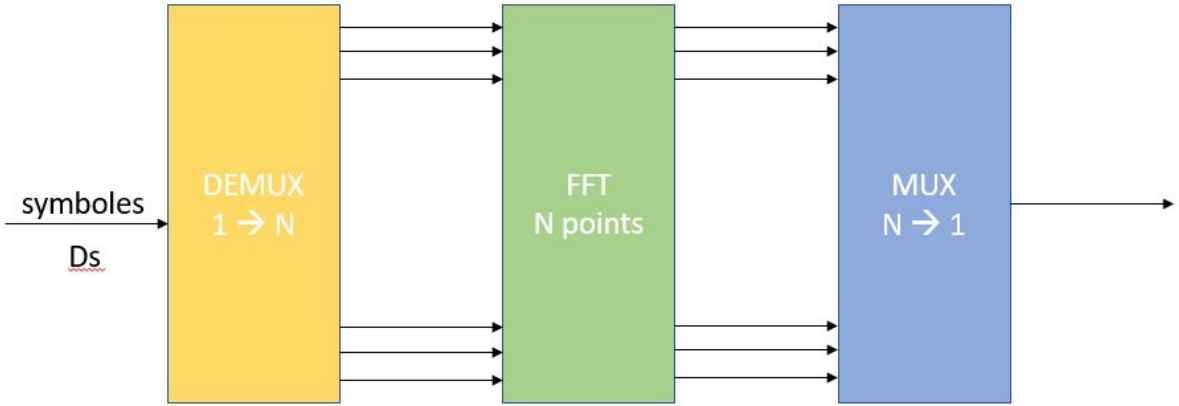
Modulateur OFDM

Code Matlab à générer :

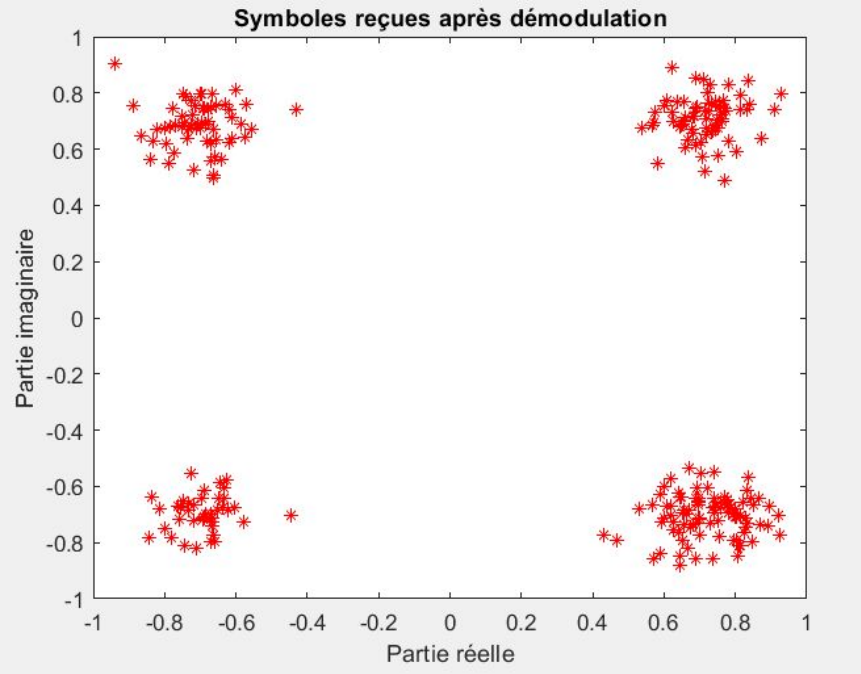
```
symb_utiles_padding = reshape(symb_utiles, [64,4]); % Demux 1 vers 64

OFMD_mod = ifft(symb_utiles_padding,64);
OFMD_mod = [ OFMD_mod(:,1),OFMD_mod(:,2),OFMD_mod(:,3),OFMD_mod(:,4)]; % Mux 64 vers 1
```

Démodulateur OFDM

Démodulateur opposé au modulateur ⇒ FFT au lieu de IFFT



Démodulateur OFDM



Modulateur OFDM (préfixe cyclique)

Mise en place du modulateur/démodulateur sur canal Multi Trajets avec bruit

```
h=sqrt(1/(2*L))*(randn(1,L)+1i*randn(1,L));
```

L : nombre de trajets discrets équivalents dans la réponse impulsionnelle du canal

CP contient au moins D échantillons >= L

```
D = L; % Taille du préxife cyclique
CP(:,1)= OFMD_mod([NFFT-D+1 : NFFT],1);
CP(:,2)= OFMD_mod([NFFT-D+1 : NFFT],2);
CP(:,3)= OFMD_mod([NFFT-D+1 : NFFT],3);
CP(:,4)= OFMD_mod([NFFT-D+1 : NFFT],4);
```

Modulateur OFDM (préfixe cyclique)

Pour éviter l'IES (interférences entre symboles) :

- Récupération des D derniers bits
- Rajouter devant les symboles utiles

```
symb_utiles_padding = reshape(symb_utiles, [64,4]);

OFMD_mod = ifft(symb_utiles_padding,64);

D = L; % Taille du préxife cyclique
CP(:,1)= OFMD_mod([NFFT-D+1 : NFFT],1);
CP(:,2)= OFMD_mod([NFFT-D+1 : NFFT],2);
CP(:,3)= OFMD_mod([NFFT-D+1 : NFFT],3);
CP(:,4)= OFMD_mod([NFFT-D+1 : NFFT],4);

CP_OFDM = [ CP(:,1); OFMD_mod(:,1); CP(:,2); OFMD_mod(:,2); CP(:,3); OFMD_mod(:,3); CP(:,4); OFMD_mod(:,4)];
```

Démodulateur OFDM (préfixe cyclique)

Suppression du préfixe cyclique avant la FFT

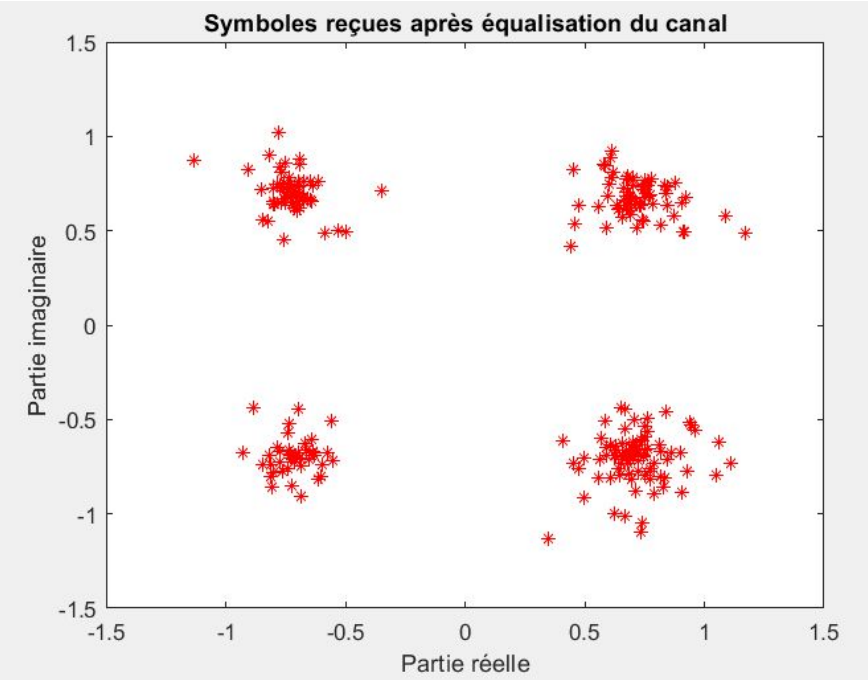
```
z = reshape(z, [NFFT+D,4]);                % Démux 1 vers N
CP_OFDM_RX(:,1) = z([D+1:NFFT+D],1);
CP_OFDM_RX(:,2) = z([D+1:NFFT+D],2);
CP_OFDM_RX(:,3) = z([D+1:NFFT+D],3);
CP_OFDM_RX(:,4) = z([D+1:NFFT+D],4);

OFMD_mod_RX = fft(CP_OFDM_RX);
```

Channel equalizer

On vient chercher à supprimer la contribution du canal

```
H = fft(h,NFFT);
CP_OFDM_E(:,1) = OFMD_mod_RX(:,1)./H.';
CP_OFDM_E(:,2) = OFMD_mod_RX(:,2)./H.';
CP_OFDM_E(:,3) = OFMD_mod_RX(:,3)./H.';
CP_OFDM_E(:,4) = OFMD_mod_RX(:,4)./H.';
```

Channel equalizer



Démonstration sur carte en co-simulation