

INSTITUT NATIONAL DES SCIENCES APPLIQUÉES

CENTRE VAL DE LOIRE

Prog. FPGA & SoC

STI5A

TP 2

Intégration d'un circuit de LFSR dans un SoC

■■■ Création d'un circuit LFSR

1 – On va définir un circuit calculant un LFSR, en sélectionnant son équation dans la page 5 du document suivant: https://docs.xilinx.com/v/u/en-US/xapp052

On sélectionnera un LFSR sur 32bits avec les coefficients 32, 22, 2, 1.

L'implémentation du LFSR est dans le contenu du fichier « circuit.v » :

```
module circuit
  input clk,
  input resetn,
  input [3:0] reg_seed_we,
  input [31:0] reg_seed_di,
  output [31:0] reg_seed_do,
  input
              reg_dat_we,
                 reg_dat_re,
 input [31:0] reg_dat_di,
  output [31:0] reg_dat_do,
  output
            reg_dat_wait
);
reg fini;
reg [31:0] seed;
reg [1:0] state;
reg [31:0] compteur; reg [31:0] lfsr;
reg seed_modifie = 0;
assign reg_dat_wait = !fini;
assign reg_dat_do = lfsr;
assign reg_seed_do = seed;
wire bit_lfsr = lfsr[31] ^ lfsr[21] ^ lfsr[1] ^ lfsr[0];
always @(posedge clk) begin
 if (!resetn) begin
    seed <= 32'h dead_beef;</pre>
   seed_modifie <= 0;</pre>
  end else begin
   if (reg_seed_we[0]) seed[ 7: 0] <= reg_seed_di[ 7: 0];</pre>
       (reg_seed_we[1]) seed[15: 8] <= reg_seed_di[15: 8];</pre>
   if (reg_seed_we[2]) seed[23:16] <= reg_seed_di[23:16];</pre>
    if (reg_seed_we[3]) seed[31:24] <= reg_seed_di[31:24];</pre>
    if (reg_seed_we != 4'b 0000) seed_modifie <= 1;</pre>
    else seed_modifie <= 0;</pre>
  end
end
always @(posedge clk) begin
 if (!resetn) begin
    fini <= 0;
    state <= 0;
    compteur <= 0;
    lfsr <= seed;</pre>
  end
  case(state)
    0: begin
      if (seed_modifie) begin
        lfsr <= seed:
      end
      if (reg_dat_re) begin
        state <= 1;
        compteur <= 0;
        fini <= 0;
      end
    end
    1: begin
    compteur <= compteur +1;</pre>
```

```
lfsr <= {lfsr[30:0],bit_lfsr};
if (compteur == 31) begin
    state <= 2;
    end
end
2: begin
    fini <= 1;
    state <= 0;
end
default: begin
    state <= 0;
    fini <= 1;
end
end
endmodule</pre>
```

Pour pouvoir tester que notre LFSR fonctionne, il faut tester une implémentation : on va utiliser Python.

Le contenu du fichier « $test_lfsr.py$ »

```
#!/usr/bin/python
lfsr = int('babecafe',16)
rep = format(lfsr, "032b")
print(rep)
compteur = 0
while 1:
    compteur += 1;
    bit = int(rep[31-31], 2) ^int(rep[31-21], 2) ^int(rep[31-1], 2) ^int(rep[31-0], 2)
    print(str(bit))
    rep = rep[1:]+str(bit)
    print (rep)
    if compteur == 32:
         lfsr = int(rep, 2)
         print (hex(lfsr))
         break
compteur = 0
while 1:
    compteur += 1;
    bit = int(rep[31-31], 2)^int(rep[31-21], 2)^int(rep[31-1], 2)^int(rep[31-0], 2)
    print(str(bit))
    rep = rep[1:]+str(bit)
    print(rep)
    if compteur == 32:
         lfsr = int(rep, 2)
         print(hex(lfsr))
         break
```

Ce script Python vous permettra d'obtenir les deux premières séquences du LFSR.

Ouestions:

a. Pouvez vous écrire une simulation pour vérifier que le circuit fonctionne bien en récupérant les deux premières valeurs du LFSR calculées par le circuit ?

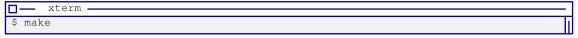
2 – Pour récupérer le PicoSoC :

```
$ git clone https://git.p-fb.net/PeFClic/fpga_picosoc.git
```

Questions:

- a. Qu'est-ce que fait le fichier «load_firmware.v»?
- b. Qu'est-ce que font les fichiers «afficheur_hexa.v» et «hex_converter.v»?
- c. À quoi sert le fichier « pll. v » ? Qu'est-ce qui le crée ?

3 – Vous synthétiserez le circuit complet :



Puis pour le mettre sur le FPGA:



- a. Vérifiez dans le firmware que les adresse d'accès sont bien correctes par rapport aux périphériques utilisés et par rapport à la définition du SoC dans le fichier «picosoc.v »
- b. Comment fonctionne les fonctions d'affichage?
- c. Est-ce que les valeurs fournies par le LFSR sont correctes?

Vous pourrez envoyer le firmware au SoC directement :

```
term

color firmware
color firmware
color firmware.txt > /dev/ttyUSB0

color firm
```

Si vous voulez modifer le firmware, il vous faut le compilateur capable de produire du code « riscv32imc » :

```
xterm ______$ sudo apt install binutils-riscv64-unknown-elf gcc-riscv64-unknown-elf
```