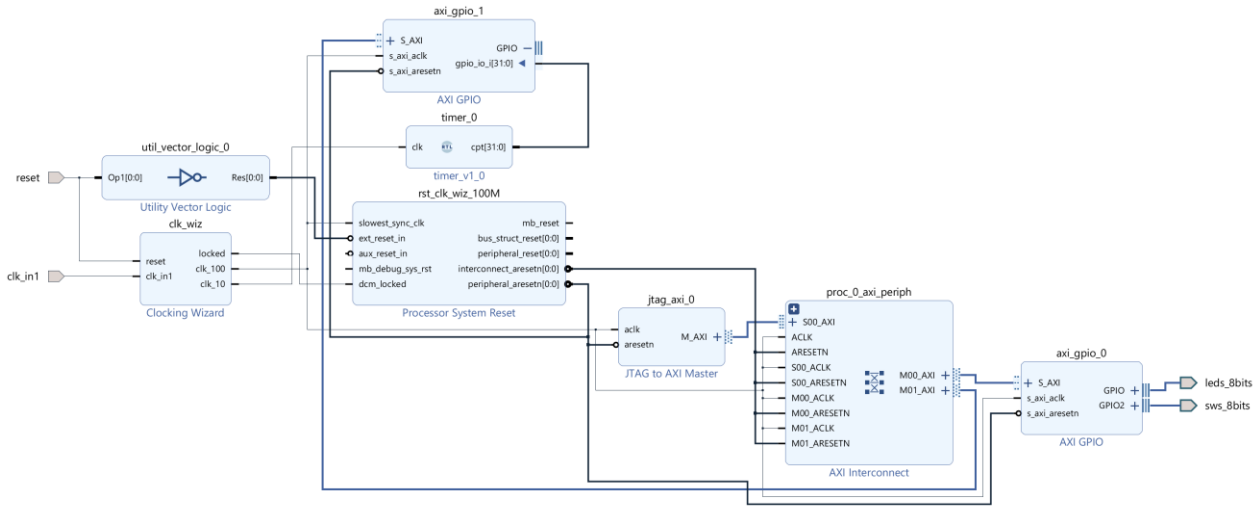
Q1 : Créer le design suivant :



Le code VHDL pour le module timer_0 est le suivant :

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;
use IEEE.std_logic_unsigned.ALL;

entity timer is
    Port ( clk : in STD_LOGIC;
           cpt : out STD_LOGIC_VECTOR (31 downto 0));
end timer;

architecture Behavioral of timer is
signal val:std_logic_vector(31 downto 0) :=X"00000000";
begin
process(clk)
begin
if rising_edge(clk) then
 val <= val+1;
end if;
end process;

cpt <= val;

end Behavioral;
```

On utilisera le fichier de contrainte zedboard_hls4.xdc donné
L'adresse de base de GPIO0 doit être 0x40000000
L'adresse de base de GPIO1 doit être 0x40010000

Q2 : Tester le design sur la carte. Pour accéder au jtag axi, on doit taper les commandes suivantes sous xsct :

- **connect**
- **target 6**

On peut ensuite lire et écrire dans les registres des périphériques à l'aide des commandes mrd et mwr

On veut maintenant créer un programme qui va enchainer les commandes à l'aide d'une IP HLS programmée en Cpp.

Q3 : sous HLS, créer une IP avec le code suivant :

```
#include <string.h>

#define GPIO0 0x40000000
#define GPIO1 0x40010000

void proc(volatile unsigned int *port,unsigned int cpt)
{
unsigned volatile int tempo;
#pragma HLS INTERFACE m_axi port=port depth=100

while(1)
	{
	tempo=cpt;
	*(port+(GPIO0/4))=0xff;
	while((cpt-tempo) < 5000000);
	tempo=cpt;
	*(port+(GPIO0/4))=0x0;
	while((cpt-tempo) < 5000000);
	}
}
```

Exporter l'IP et l'intégrer dans le design précédent. Tester le fonctionnement.

Q4 : Proposer une modification qui n'utilise pas le port `unsigned int cpt`

Q5 : Modifier le programme de l'IP pour modifier la fréquence de clignotement en fonction de l'état des switchs de la carte

Q6 : Supprimer un l'IP proc, ajouter un IP Zync et développer un programme C qui pilotera les composant AXI sous Vitis.