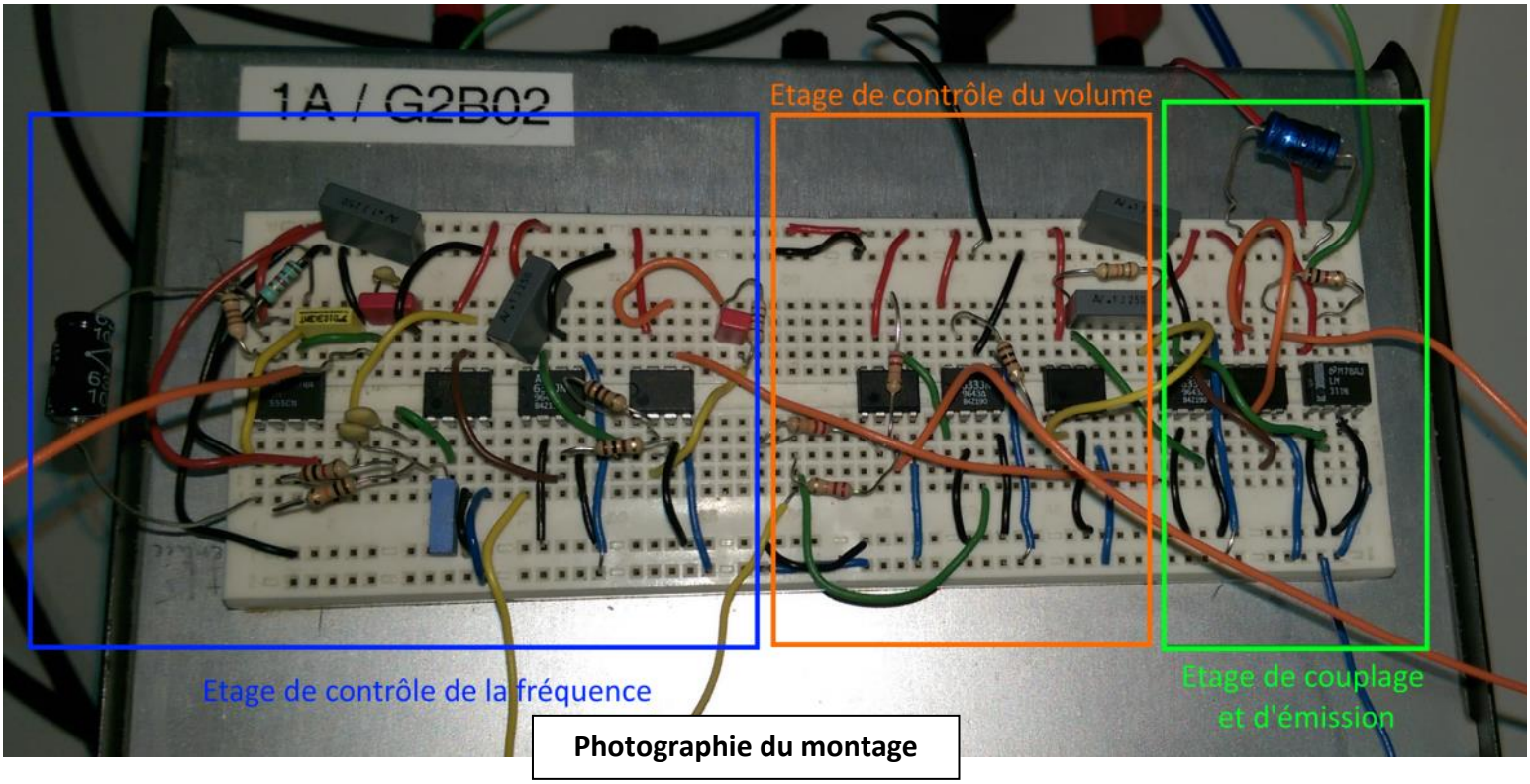
# *Le Thérémine :*
# *Documentation technique*

## ***Le montage*** :




**Photographie du montage**

Le thérémine utilise habituellement des antennes filaires alimentées par des tensions très élevées (80V environ), dans notre cas les limites imposées par les composants nous ont conduit à utiliser des antennes formées de feuilles d'aluminium, ce qui augmente la surface et permet d'utiliser une tension d'alimentation de 15V.

L'utilisation d'une plaquette plutôt qu'un circuit imprimé adapté induit de nombreuses capacités parasites. Le LM555 cause une certaine instabilité des alimentations, ce qui nous a obligé à utiliser des condensateurs supplémentaires pour découpler les alimentations (ces condensateurs ne seront pas représentés sur les schémas suivants).

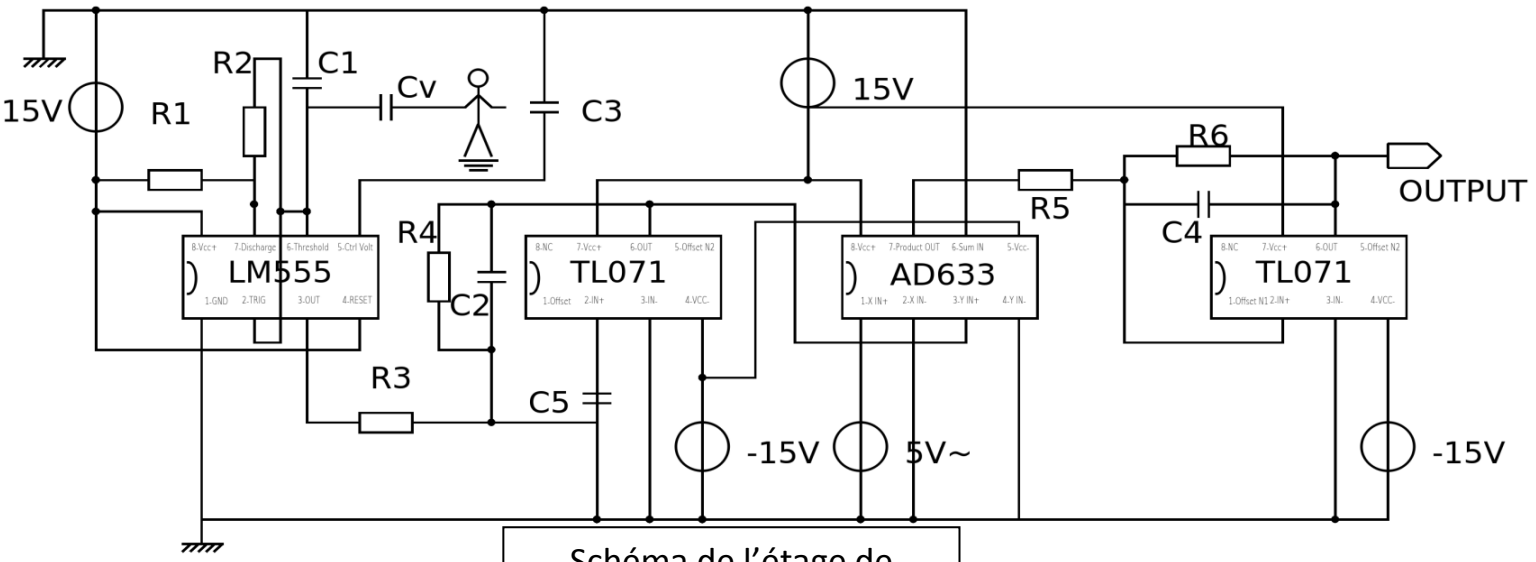



Schéma de l'étage de contrôle de la fréquence

## Valeurs des composants :

R1=120Ω  R2=1.8kΩ  R3=10kΩ  R4=100Ω  R5=50Ω  R6=100Ω
C1=4.6pF  C2=79pF  C3=0.01µF  C4=40nF  C5=16nF

La fréquence du signal sinusoïdal utilisée pour la multiplication est de 172kHz (valeur historique). Pour une antenne d'environ $10^{-2}m^2$ la variation maximale de fréquence vaut environ 6kHz, cette plage de fréquence ne recouvre pas l'ensemble du spectre audible par l'oreille humaine. La modification de la fréquence du signal utilisé pour le multiplieur permet de modifier la plage de fréquences balayées par l'instrument.

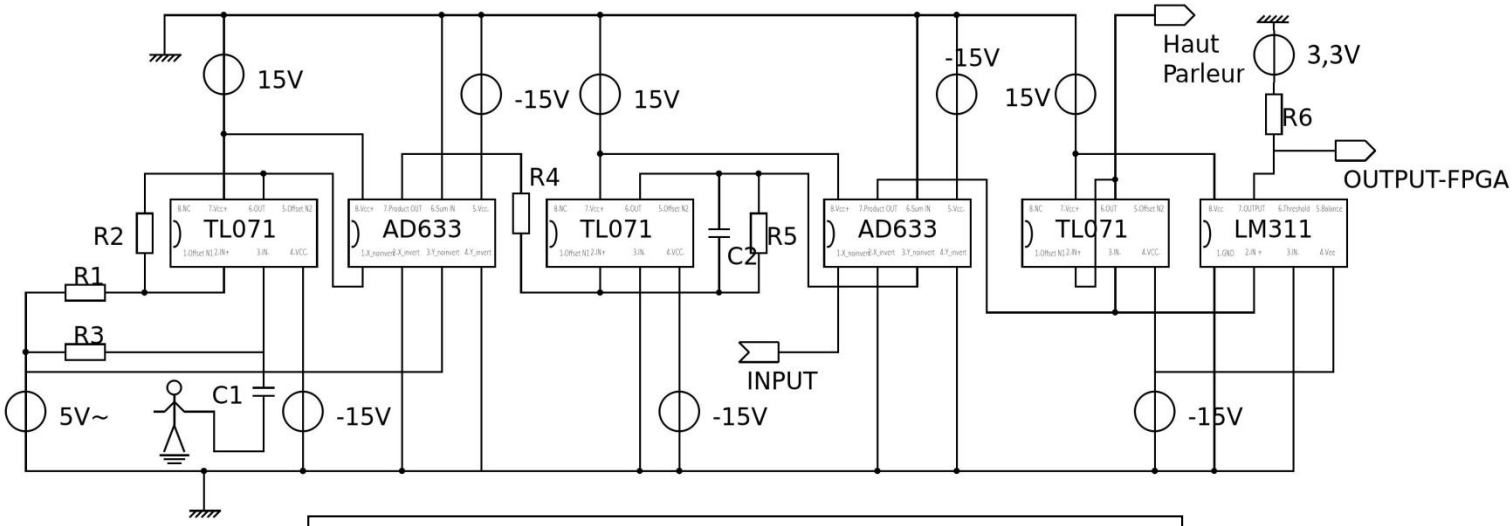



**Schéma des étages de génération de la tension de contrôle du volume ainsi que du couplage des signaux et d'émission**

## Valeurs des composants :

R1=R2=R3=22kΩ  R4=10Ω R5=18kΩ  R6=1kΩ
C2=4nF

Cette fois la capacité variable n'est pas utilisée en complément d'une autre, ainsi le déphasage φ est directement relié à la proximité de l'utilisateur vis-à-vis de l'antenne.

La fréquence du signal sinusoïdal utilisé pour générer la tension variable n'as pas d'importance, elle doit simplement être suffisamment élevée pour permettre un filtrage permettant de facilement récupérer uniquement la composante de fréquence nulle du signal. Cette fréquence doit également être suffisamment faible pour que le signal ne soit pas détérioré par le slew-rate de l'amplificateur opérationnel utilisé, dans notre cas une fréquence de 21kHz a permis d'obtenir la tension variable recherchée.

Le traitement numérique que nous avons réalisé devait permettre d'afficher la fréquence du son émis, cependant la mise au format créneau du signal à l'aide du LM311 n'est pas parfaite, le signal obtenu présentait des oscillations de plus haute fréquence (voir schéma ci-dessous). L'affichage de la fréquence était donc faussé de façon répétée et nous n'avons pas eu le temps de modifier notre code VHDL pour corriger le problème.

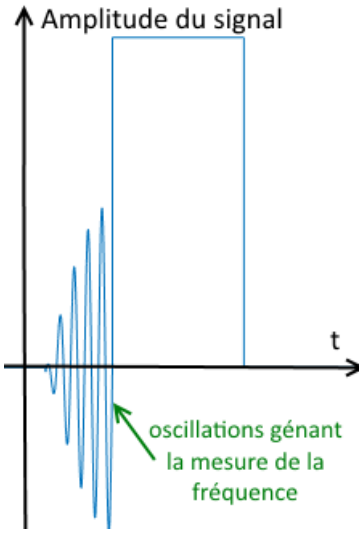



**Schéma d'un créneau obtenu après utilisation du LM311**

## ***Le code VHDL utilisé :***

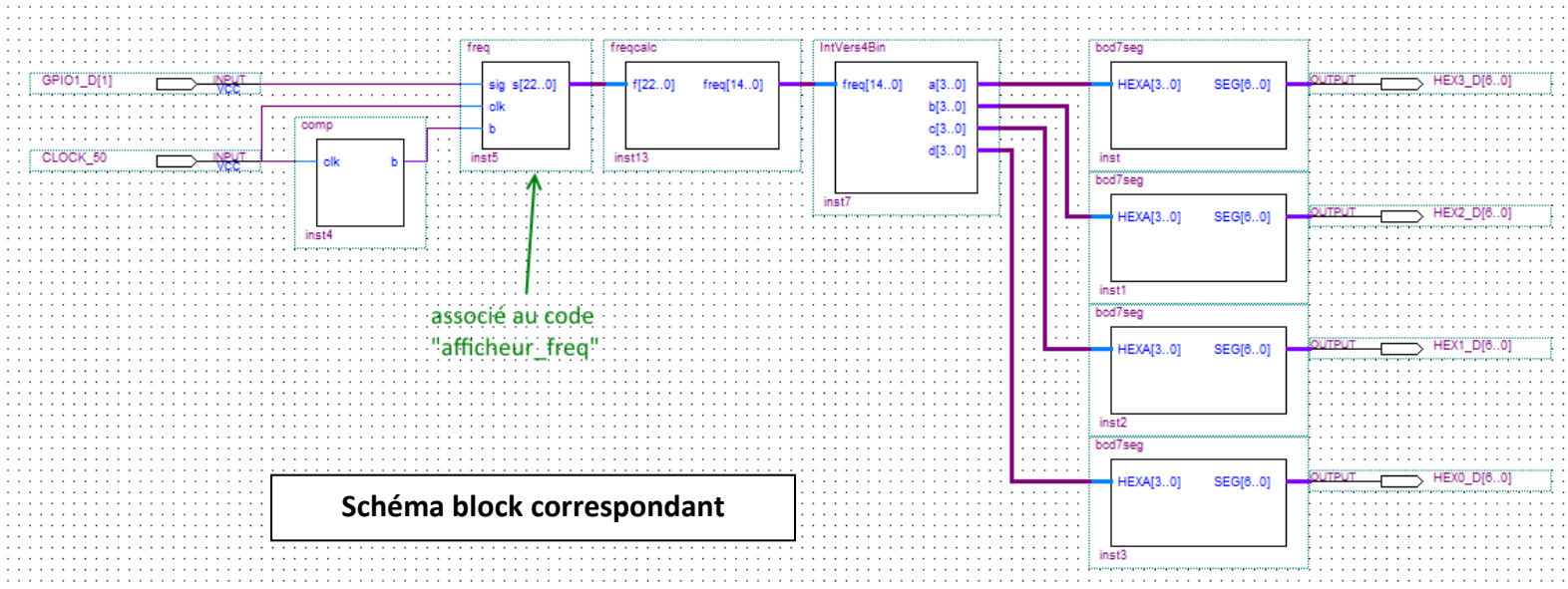



**Schéma block correspondant**

## *Listing des différents codes :*

bcd7seg

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
-- ENTITE --
entity bcd7seg is
	port(
		HEXA : in BIT_VECTOR(3 DOWNTO 0);
		SEG : out BIT_VECTOR(6 DOWNTO 0)
	);
end bcd7seg;
-- ARCHITECTURE --
architecture simple of bcd7seg is
begin
		--	gfedcba
		SEG <= "1000000" when HEXA = "0000" else
			"1111001" when HEXA = "0001" else
```

```
                              "0100100" when HEXA = "0010" else
                              "0110000" when HEXA = "0011" else
                              "0011001" when HEXA = "0100" else
                              "0010010" when HEXA = "0101" else
                              "0000010" when HEXA = "0110" else
                              "1111000" when HEXA = "0111" else
                              "0000000" when HEXA = "1000" else
                              "0010000" when HEXA = "1001" else
                              "0001000" when HEXA = "1010" else
                              "0000011" when HEXA = "1011" else
                              "1000110" when HEXA = "1100" else
                              "0100001" when HEXA = "1101" else
                              "0000110" when HEXA = "1110" else
                              "0001110";
end simple;
```

## Comp

(compteur)

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity comp is
       port(
       clk : in STD_LOGIC;
       b : out bit
       );
end comp;
architecture Behavorial of comp is
signal s : integer range 0 to 50000000;
begin
       cpt : process(clk)
              begin
              if(clk 'event and clk = '1') then
                     s<=s+1;
              end if;
              if(s<=25000000) then
              b<='1';
              elsif(s>25000000) then
              b<='0';
              end if;
       end process;
end Behavorial;
```

## freqcalc

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity freqcalc is
       port(
       f : in integer range 0 to 6000000;
       freq : out integer range 0 to 20000
       );
end freqcalc;
architecture Behavorial of freqcalc is
begin
              freq<=50000000/f;
end Behavorial;
```

## IntVers4Bin

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

```
entity IntVers4Bin is
	port(
	freq : in integer range 0 to 20000;
	a,b,c,d : out integer range 0 to 9
	);
end IntVers4Bin;
architecture Behavorial of IntVers4Bin is
begin
			a<=freq/10000;
			b<=(freq mod 10000)/1000;
			c<=(freq mod 1000 )/100;
			d<=(freq mod 100  )/10;
end Behavorial;
```

Afficheur_freq

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity freq is
	port(
		sig,clk : in std_logic;
		b : in bit;
		s : out integer range 0 to 6000000
		);
end freq;
architecture Behavorial of freq is
signal nc : integer range 0 to 6000000;
signal nc2 : integer range 0 to 6000000;
signal bin : bit;
begin
	cpt: process(sig,clk)
	begin
	if(b= '0')then
		if(sig = '1') then

			if(clk 'event and clk='1') then
				nc <= nc+1;
			end if;
		end if;
		nc2<=nc;
		if (sig ='0') then
		nc<= 0;
		end if;
	end if;
	end process;
	s<=2*nc2;
end Behavorial;
```