22/10/2021

PROJET ALARME

Noumane JAFAIS
IUT VILLETANEUSE

PREMIERE PARTIE :

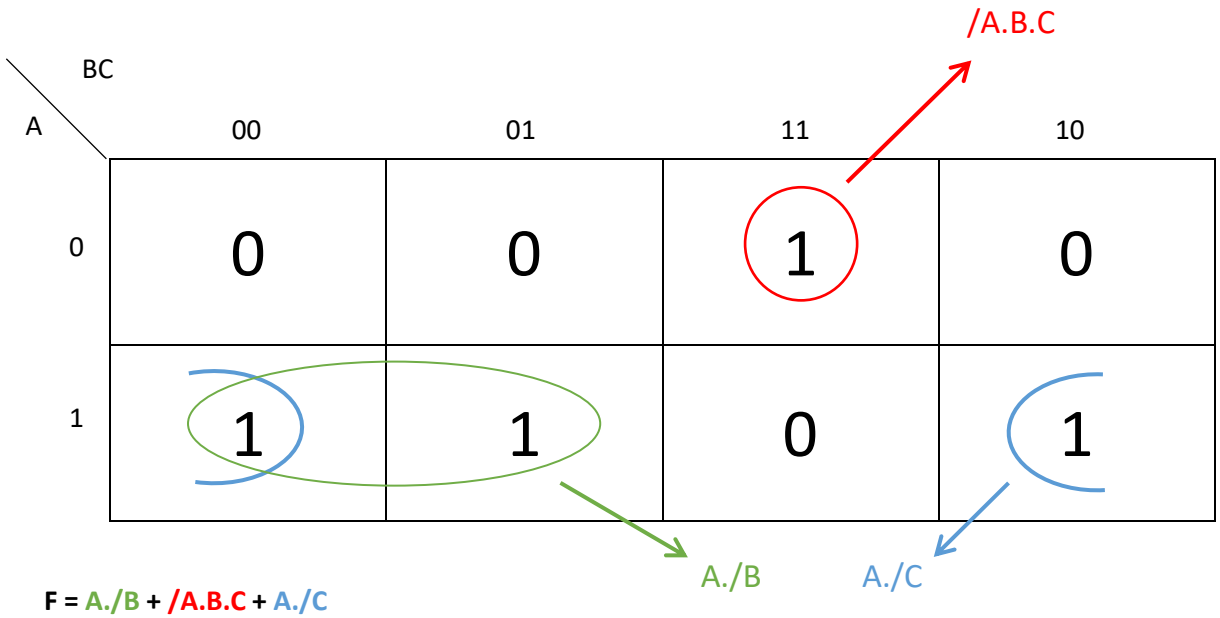
Rappels sur la logique combinatoire :

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

D'après le tableau de vérité, nous avons la fonction logique suivante :

F = /A.B.C + A./B./C + A./B.C + A.B./C

A partir de cette fonction logique, nous pouvons en déduire le tableau de Karnaugh :



F = A./B + /A.B.C + A./C

En simplifiant, nous avons :

F = /A.B.C + A.(/C + /B)

Nous réalisons le logigramme de cette fonction sur le logiciel Quartus II. La *figure 1* représente le logigramme obtenu.

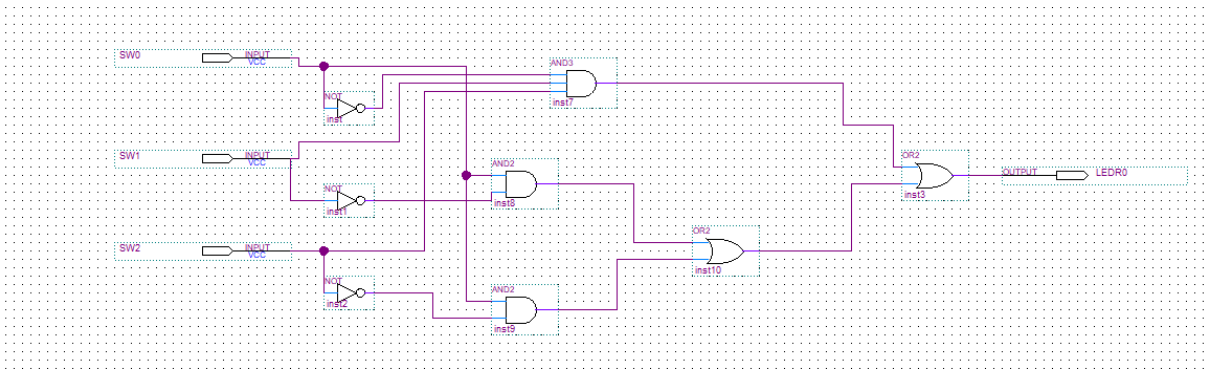
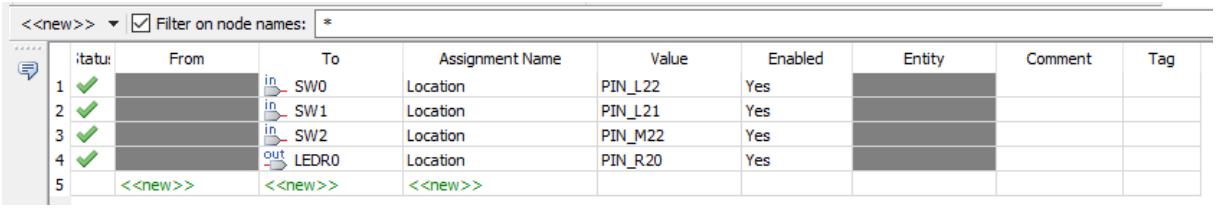


Figure 1 : Logigramme de la fonction

Après avoir fait le schéma, on fait la compilation. Une fois la compilation terminée, on peut passer à l'étape de l'assignation des entrées et des sorties. Pour cela, on choisit quelle patte de la patte correspond à l'entrée et à la sortie.



	Status	From	To	Assignment Name	Value	Enabled	Entity	Comment	Tag
1	✓		SW0	Location	PIN_L22	Yes			
2	✓		SW1	Location	PIN_L21	Yes			
3	✓		SW2	Location	PIN_M22	Yes			
4	✓		LEDR0	Location	PIN_R20	Yes			
5		<<new>>	<<new>>	<<new>>					

Figure 2 : Assignation des entrées/sorties

Voici le chronogramme de la fonction logique :

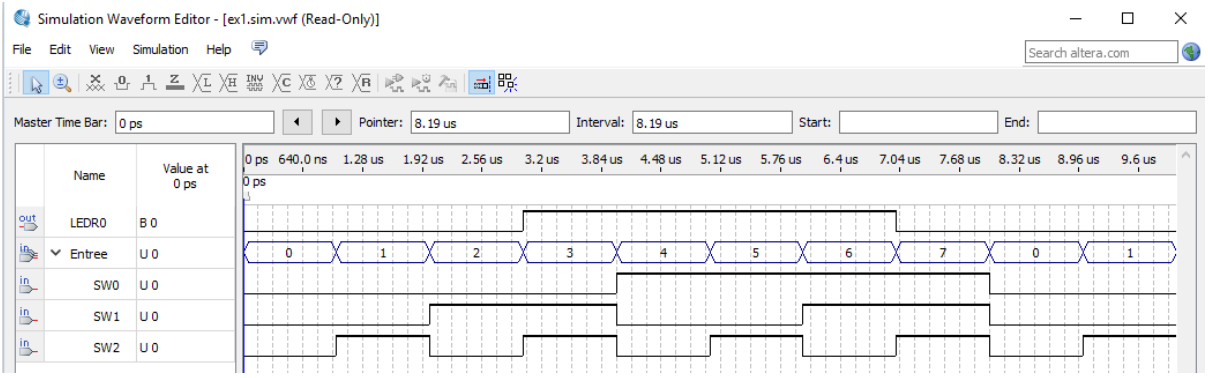


Figure 3 : Chronogramme

Ce chronogramme vérifie bien le tableau de vérité.

Dernière étape, c'est la programmation. On relie la carte à l'ordinateur et on débute la programmation.

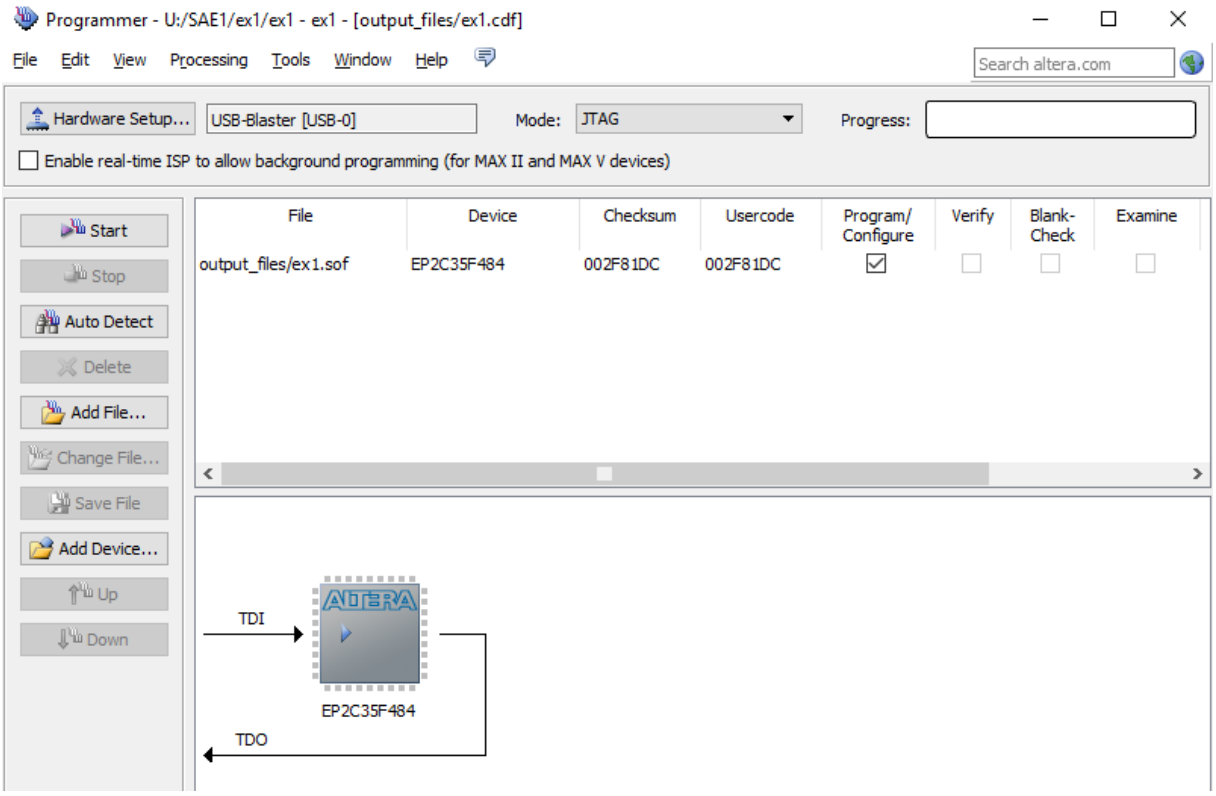


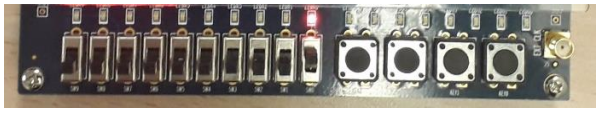
Figure 4 : Fenêtre de programmation

On obtient donc les résultats :

F = A./B.C

F = /A.B.C

F = A.B./C



F = A./B./C

TROISIEME PARTIE :

Introduction

Le but de ce projet est d'élaboration d'un circuit permettant de contrôler l'ouverture d'une porte protégée par un code d'accès.

Pour implémenter ce projet nous utiliserons la carte didactique ALTERA DE1 et le logiciel Quartus II. Sur la carte, nous utiliserons les interrupteurs, les boutons poussoirs, l'afficheur 7 segments et les LED rouge et verte.

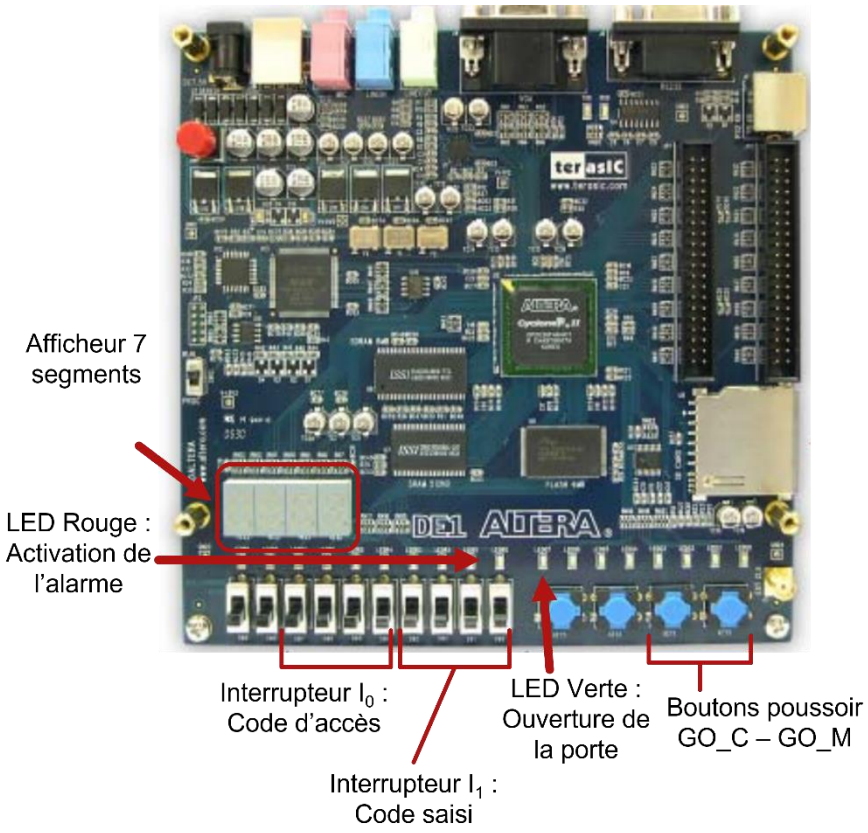


Figure 5 : Carte didactique ALTERA DEL

Étude d'un circuit de contrôle d'accès :

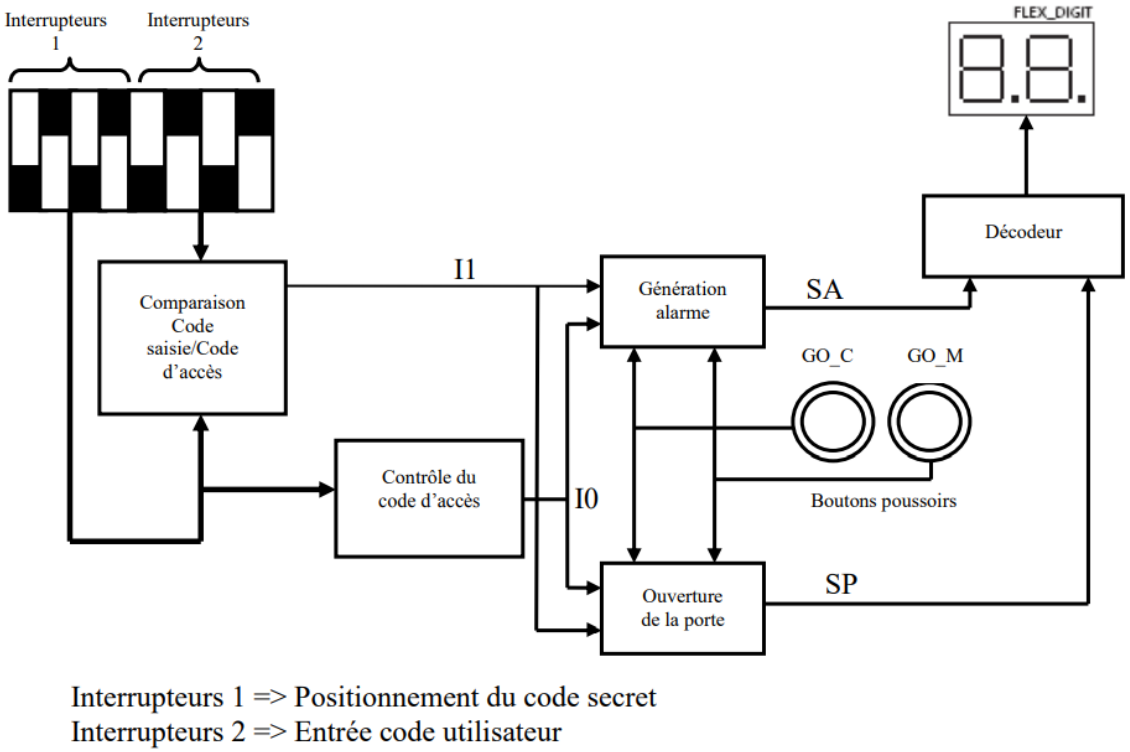


Figure 6 : Schéma de principe du circuit de contrôle d'accès

Principe de fonctionnent du circuit :

Au démarrage du système, un code d'accès sur 4 bits doit être positionné par le responsable de la sécurité sur les interrupteurs I_0 (il est bien entendu que ces interrupteurs I_0 ne sont accessible qu'au responsable de sécurité).

Pour permettre de changer le code sans déclencher l'alarme, il faut que GO_M passe à 0 (Bouton poussoir appuyé). Lorsque le code d'accès a été positionné, il faut que GO_M repasse à 1 (Bouton poussoir relâché) afin que le nouveau code soit pris en compte.

Pour des raisons de sécurité, le code d'accès doit comporter au moins deux 1 consécutifs. Si ce n'est pas le cas une alarme est déclenchée et la porte ne peut être ouverte.

Pour ouvrir la porte, l'utilisateur doit entrer le code d'accès qui lui a été communiqué auparavant par le responsable de sécurité. L'utilisateur entre le code par l'intermédiaire des interrupteurs I_1 et le valide en faisant passer GO_C à 0

(Bouton poussoir appuyé). GO_C doit être remis ensuite à 1 (Bouton poussoir relâché) pour permettre de rentrer un nouveau code. Le code entré par l'utilisateur sera comparé avec celui enregistré. Si le code est correct, la porte sera ouverte, sinon l'alarme sera activée.

1) Conception du bloc « génération de l'alarme »

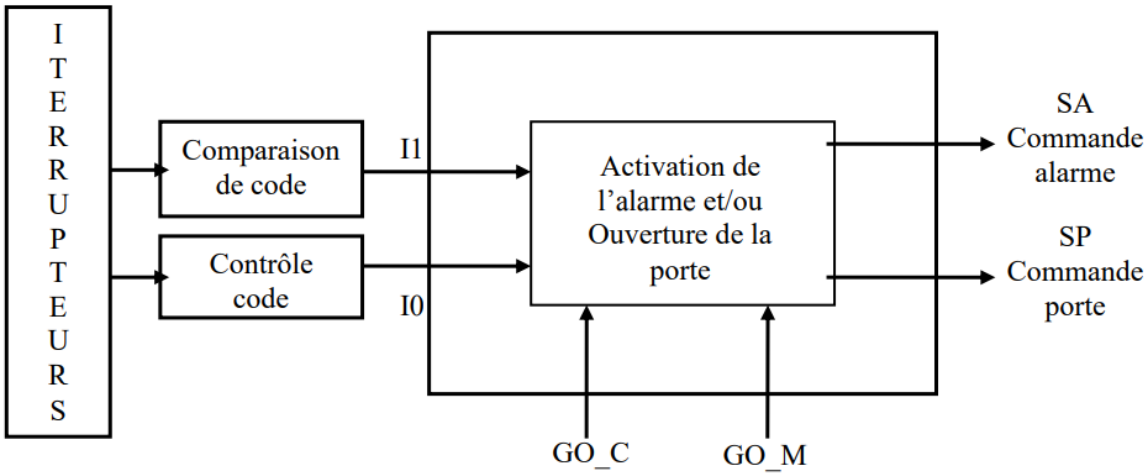


Figure 7 : Bloc de génération de l'alarme

Le bloc de génération de l'alarme comporte 4 entrées (I_0, I_1, GO_C, GO_M) et deux sorties (SA et SP).
Le bloc de génération de l'alarme doit permettre d'afficher la lettre « **FE** » sur les afficheurs 7 segments de la carte ALTERA si un code est mal entré. **Deux événements seulement doivent déclencher l'alarme :**

1. Le positionnement d'un code d'accès non valide par le responsable de la sécurité : on obtient alors lorsque le bouton poussoir est relâché **(GO_M =1)** la variable $\mathbf{I_0 = 0}$ et donc l'alarme se déclenche et la porte reste fermée.

2. Le positionnement d'un code erroné par l'utilisateur : on obtient alors lorsque le bouton poussoir est relâché **(GO_C = 1)** la variable $\mathbf{I_1 = 1}$ et de la même manière que précédemment, l'alarme se déclenche et la porte reste fermée.

L'affichage des lettres « **FE** » symbolise que l'alarme se déclenche, à l'inverse l'affichage « **OU** » symbolise l'ouverture de la porte.

GO_C	GO_M	I_1	I_0		SA
0	0	0	0		0
0	0	0	1		0
0	0	1	0		0
0	0	1	1		0
0	1	0	0		1
0	1	0	1		0
0	1	1	0		1
0	1	1	1		0
1	0	0	0		0
1	0	0	1		0
1	0	1	0		1
1	0	1	1		1
1	1	0	0		1
1	1	0	1		0
1	1	1	0		1
1	1	1	1		1

On réalise le tableau de Karnaugh :

SA	I_1 I_0	00	01	11	10
GO_C GO_M					
	00	**0**	**0**	**0**	**0**
	01	**1**	**0**	**0**	**1**
	11	**1**	**0**	**1**	**1**
	10	**0**	**0**	**1**	**1**

On a la fonction suivante :

SA = GO_M./I_0 + GO_C.I_1

Grâce à cette fonction, on obtient le logogramme suivant :

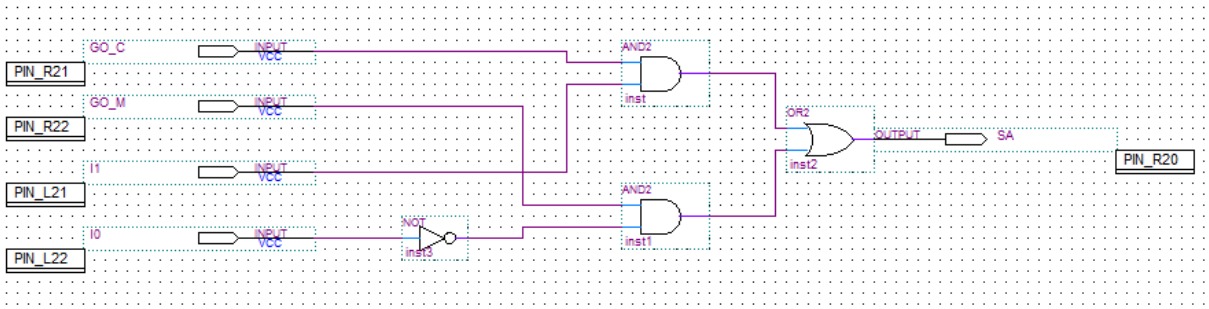


Figure 8 : Logigramme de l'alarme

En simulant, on obtient le chronogramme suivant :

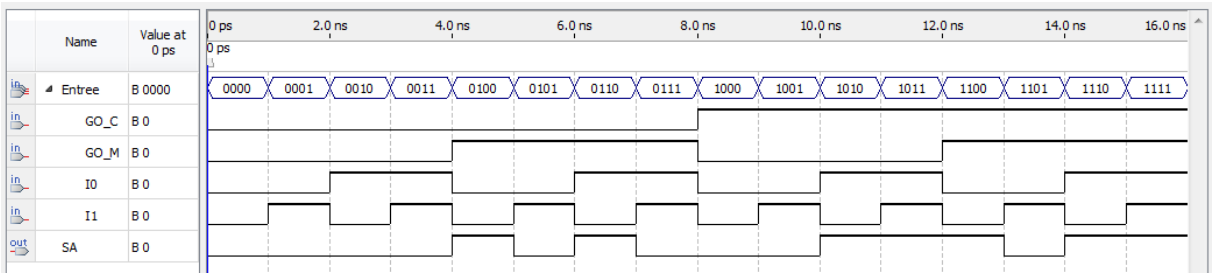


Figure 9 : Chronogramme de l'alarme

2) Conception du bloc « ouverture de la porte »

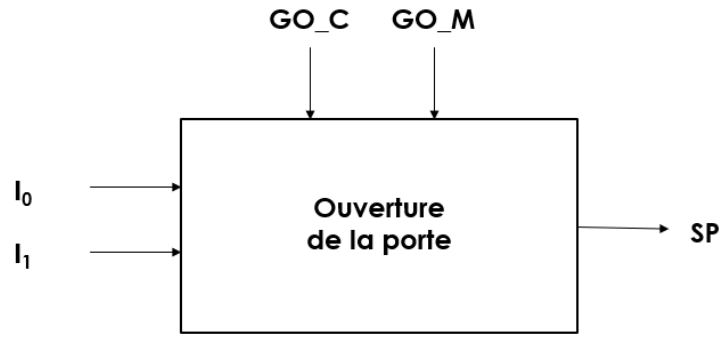


Figure 10 : Bloc d'ouverture de la porte

Ce bloc comporte 4 entrées et une sortie. Le bloc de génération de l'alarme doit permettre d'afficher la lettre « **OU** » sur les afficheurs 7 segments de la carte ALTERA si un code est bien entré.

La porte doit rester fermée tant que :

1. Le responsable de la sécurité est entrain de positionner le code d'accès sur les interrupteurs n°1 **(GO_M = 0)**,
2. Un utilisateur n'a pas terminé de taper un code **(GO_C = 0)**,
3. Si un code non valide a été positionné **(GO_M = 1 et I_0 = 0)**,
4. Si un mauvais code a été entré par l'utilisateur **(GO_C= 1 et I_1 = 1)**.

GO_C	GO_M	I1	I0		SP
0	0	0	0		0
0	0	0	1		0
0	0	1	0		0
0	0	1	1		0
0	1	0	0		0
0	1	0	1		0
0	1	1	0		0
0	1	1	1		0
1	0	0	0		0
1	0	0	1		0
1	0	1	0		0
1	0	1	1		0
1	1	0	0		0
1	1	0	1		1
1	1	1	0		0
1	1	1	1		0

On a donc la fonction suivante :

$$SP = GO\_C.GO\_M./I_1.I_0$$

Grâce à cette fonction, on trouve le logigramme suivant :

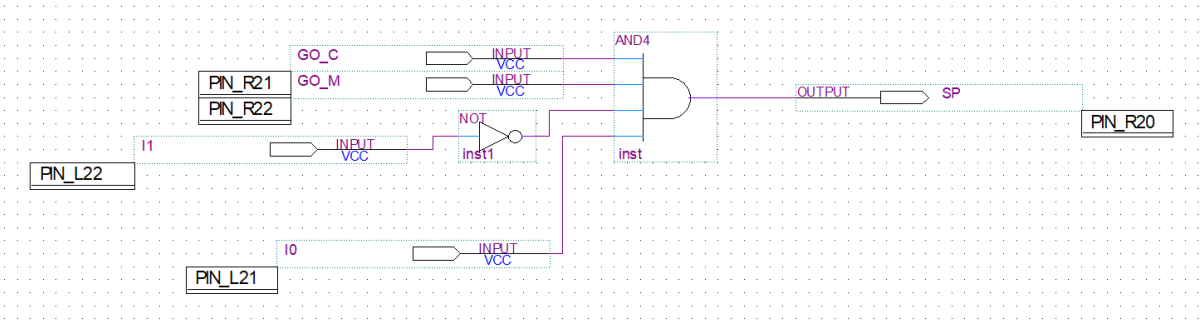


Figure 11 : Logigramme de la porte

Puis, en simulant on obtient le chronogramme suivant :

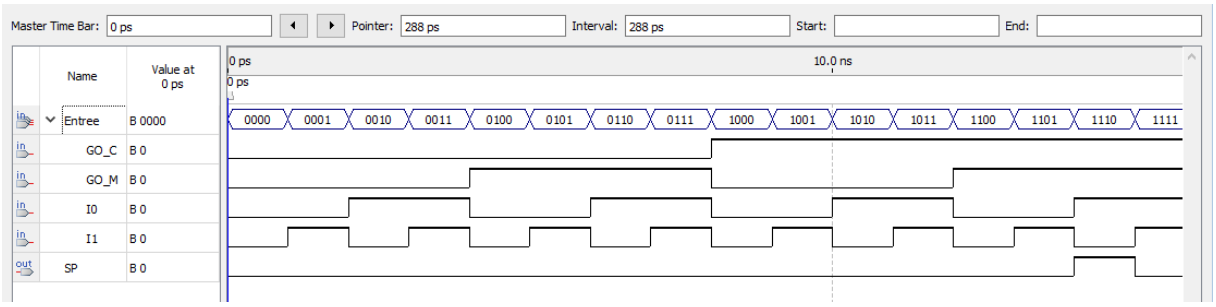


Figure 12 : Chronogramme de la porte

Ce chronogramme vérifie bien la table de vérité.

Maintenant, on combine la sortie SP avec la sortie SA. Voici cette table de vérité.

GO_C	GO_M	I1	I0		SP	SA
0	0	0	0		0	0
0	0	0	1		0	0
0	0	1	0		0	0
0	0	1	1		0	0
0	1	0	0		0	1
0	1	0	1		0	0
0	1	1	0		0	1
0	1	1	1		0	0
1	0	0	0		0	0
1	0	0	1		0	0
1	0	1	0		0	1
1	0	1	1		0	1
1	1	0	0		0	1
1	1	0	1		1	0
1	1	1	0		0	1
1	1	1	1		0	1

3) Conception du bloc « comparaison code saisi/code d'accès »

Le bloc de comparaison permet de comparer le code saisi et le code d'accès.

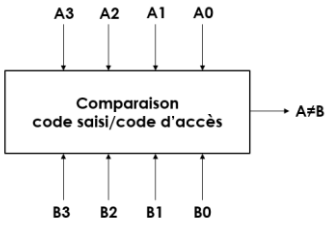


Figure 13 : Bloc de comparaison

L'entrée A correspond au code saisi par l'utilisateur et l'entrée B correspond au code d'accès positionné par le responsable de la sécurité. Si les deux entrées (codées en binaire sur 4 bits) sont différentes alors la sortie de ce bloc doit être à 1 sinon elle doit être à 0.

Pour réaliser le bloc comparateur, nous utiliserons le circuit intégré 7485.

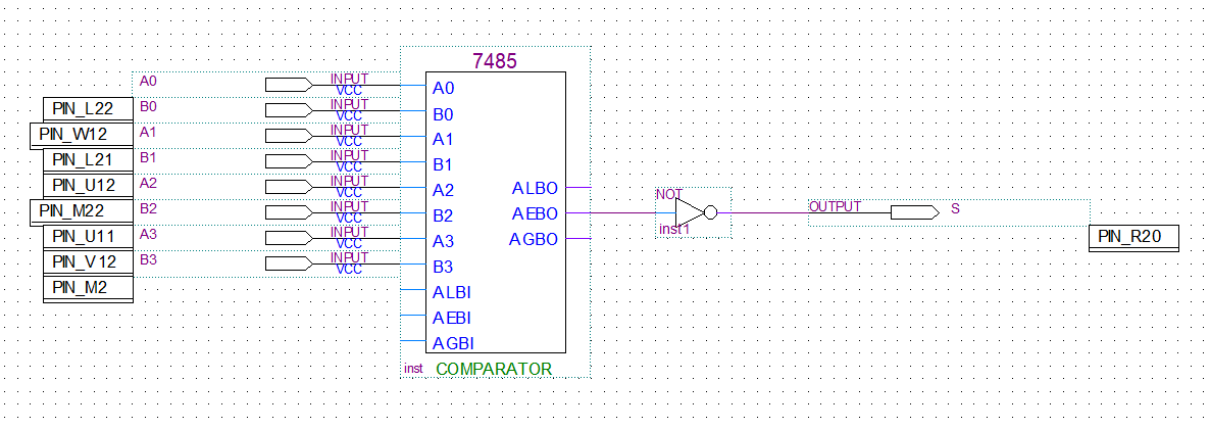


Figure 14 : Logigramme pour la comparaison

Nous avons réalisé le bloc comparateur avec 8 entrées et une sortie.

Voici ci-dessous le chronogramme du bloc comparateur.

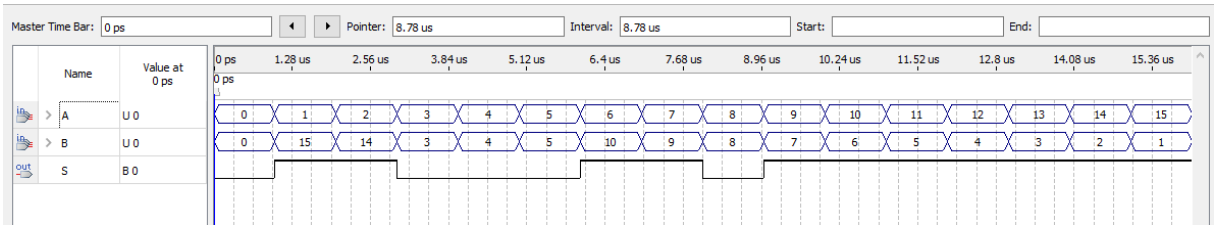


Figure 15 : Chronogramme du comparateur

D'après ce chronogramme, on peut voir que si A et B sont égales alors la sortie est à 0 sinon elle est à 1.

4) Conception du bloc « contrôle du code d'accès » :

Figure 16 : Bloc de contrôle d'accès

Le bloc du multiplexeur comporte une entrée C qui correspond au code d'accès et une sortie. Si le code mis en mémoire comporte au moins deux 1 consécutifs alors la sortie de ce bloc doit être à 1.

Nous avons ici le bloc contrôle du code d'accès qui va être traité à l'aide d'un multiplexeur 8 :1, 74151.

On réalise la table de vérité de ce bloc.

C3	C2	C1	C0		OK	
0	0	0	0		0	0
0	0	0	1		0	
0	0	1	0		0	C0
0	0	1	1		1	
0	1	0	0		0	0
0	1	0	1		0	
0	1	1	0		1	1
0	1	1	1		1	
1	0	0	0		0	0
1	0	0	1		0	
1	0	1	0		0	C0
1	0	1	1		1	
1	1	0	0		1	1
1	1	0	1		1	
1	1	1	0		1	1
1	1	1	1		1	

OK

C1 C0

C3 C2

	00	01	11	10
00	0	0	1	0
01	0	0	1	1
11	1	1	1	1
10	0	0	1	0

OK = C1.C0 + C2.C1 + C3.C2

Pour réaliser le bloc du contrôle d'accès, nous utiliserons le circuit intégré 74151 (multiplexeur 8 :1)

Voici ci-dessous la fonction du logigramme multiplexeur.

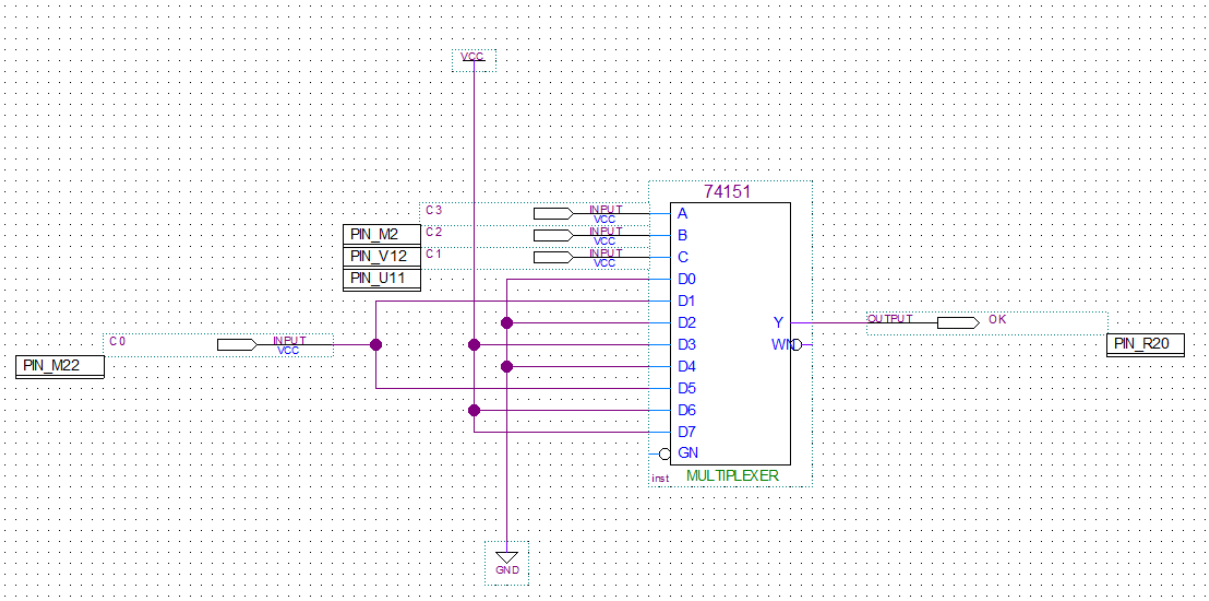


Figure 17 : Logigramme pour le contrôle d'accès

En simulant, voici le chronogramme qu'on obtient par la fonction multiplexeur.

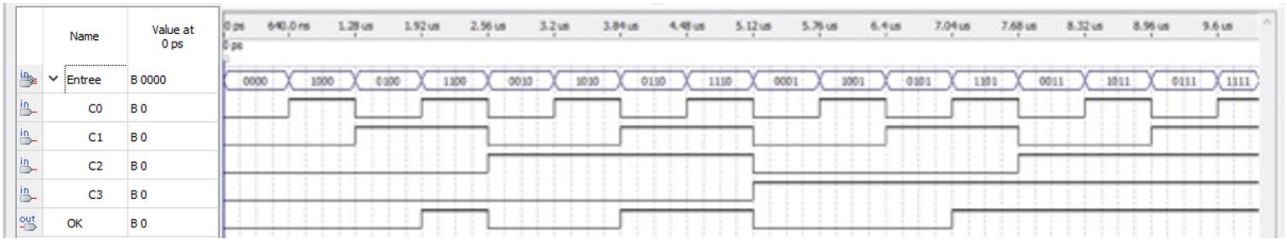


Figure 18 : Chronogramme du multiplexeur

On voit bien que ce chronogramme vérifie la table de vérité. Quand deux 1 consécutifs se suivent alors la sortie est à 1.

5) Conception du circuit complet de contrôle de code d'accès :

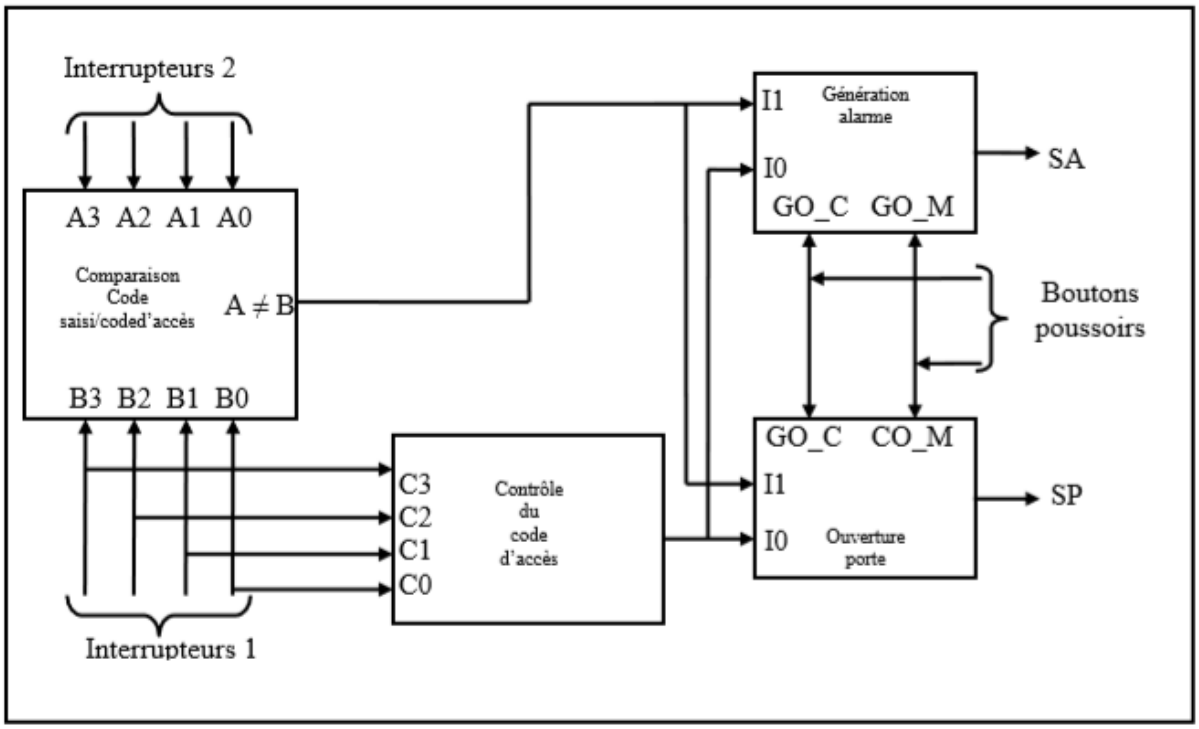


Figure 19 : Schéma fonctionnel complet du système de contrôle d'accès codé

Maintenant, on relie tous les schémas de chaque partie, et on obtient ce logigramme.

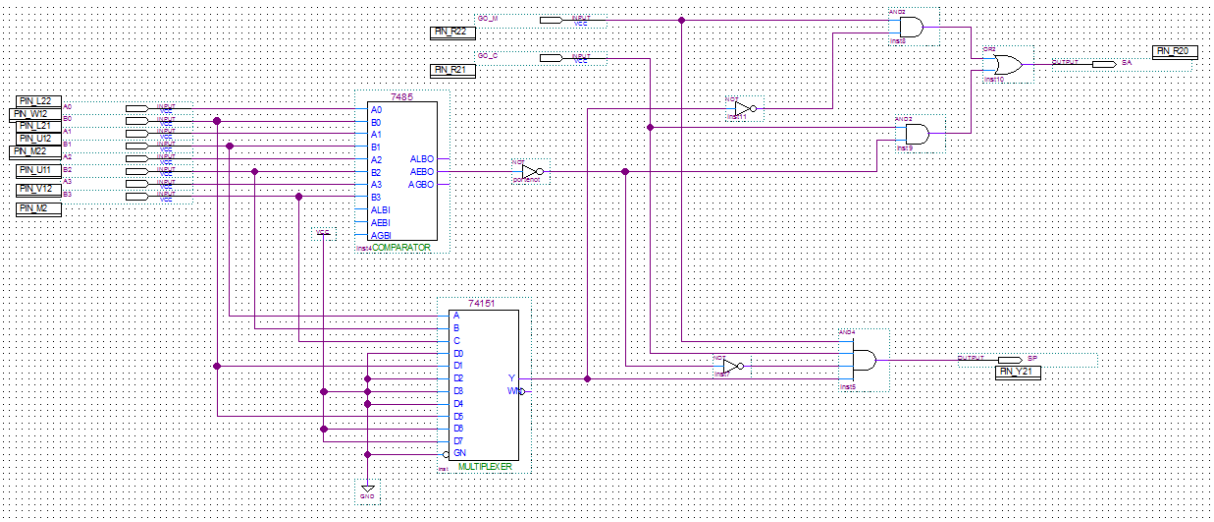


Figure 20 : Logigramme complet du système de contrôle d'accès codé

En version plus simplifiée avec les blocs.

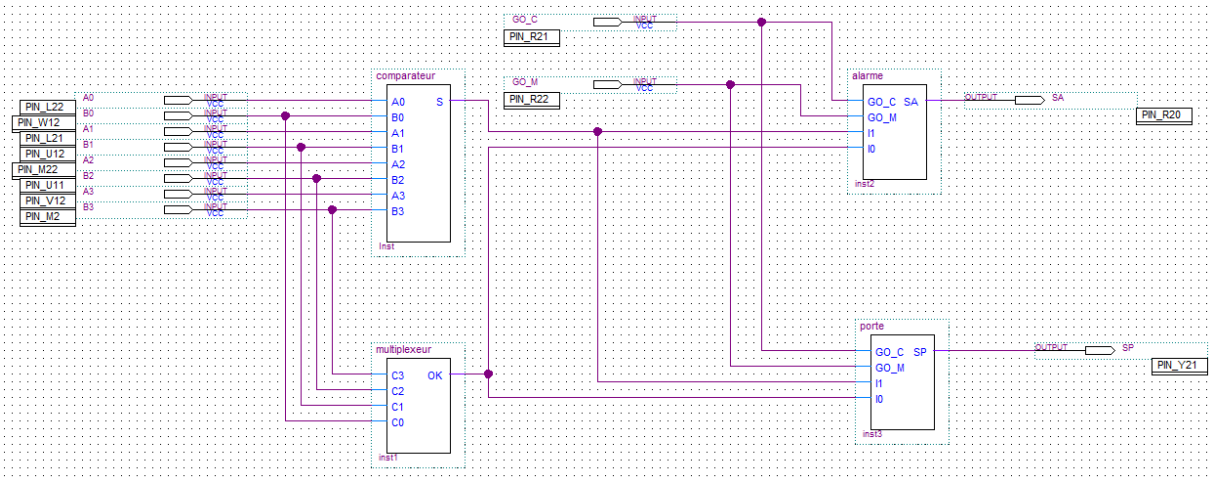


Figure 21 : Logigramme complet du système de contrôle d'accès codé (avec blocs)

On simule et on obtient ce chronogramme.

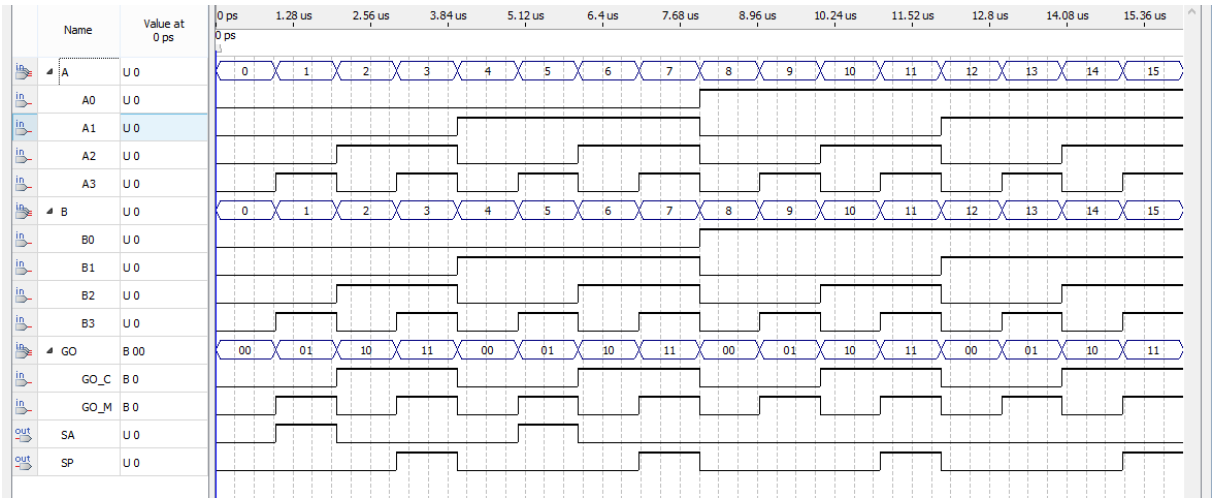


Figure 22 : Chronogramme complet du système de contrôle d'accès

Maintenant, on ajoute les sorties pour les afficheurs 7 segments.

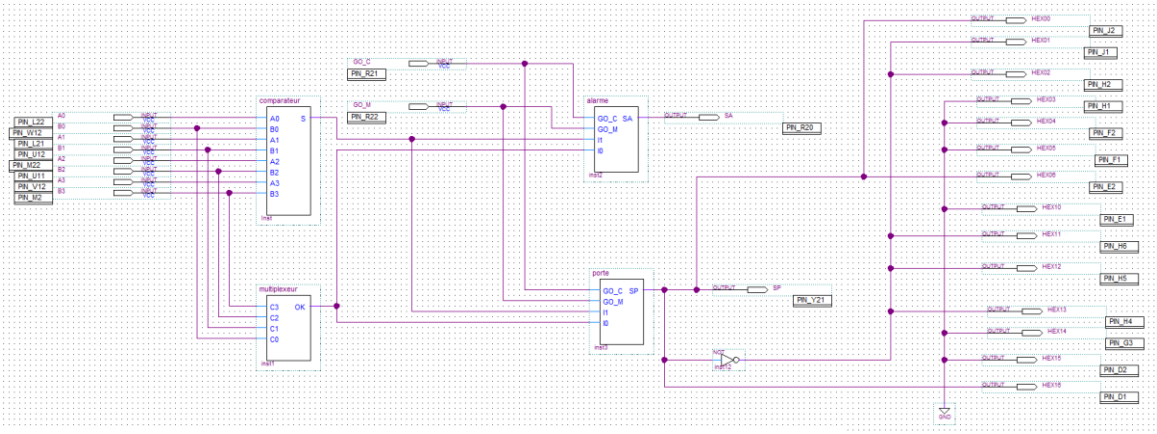
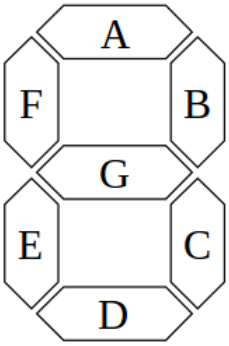
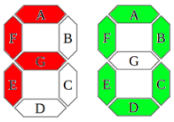


Figure 23 : Chronogramme complet du système de contrôle d'accès avec les sorties pour les afficheurs 7 segments

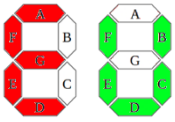
L'afficheur 7 segments s'allume quand la valeur d'un segment est à 0.



Pour le premier afficheur, on doit afficher **F** ou **O**. 

Pour **F** : A,F,G,E,D = 0 et B,C,D = 1 ; Pour **O** : A,B,C,D,E,F = 0 et G = 1

Donc A,E,F toujours = 0

Pour le deuxième afficheur, on doit afficher **E** ou **U**. 

Pour **E** : A,F,G,E,D = 0 et B,C = 1 ; Pour **U** : B,C,D,E,F = 0 et A,G = 1

Donc D,E,F toujours = 0

On obtient ces résultats et on voit que le circuit permettant de contrôler l'ouverture d'une porte protégée par un code d'accès est fonctionnel.

A3.A2./A1.A0 + B3.B2./B1.B0 = SP

A3.A2./A1.A0 + B3.B2.B1.B0 = SA

A3.A2./A1./A0 + B3.B2./B1./B0 = SP

Conclusion

Ce projet s'est révélé très productif. En effet, le respect de délais et le travail seront primordial pour le métier qu'on voudrait faire. De plus, ce projet nous a permis d'appliquer nos connaissances sur les fonctions logiques, les tables de vérité et les tableaux de Karnaugh, nous avons pu comprendre comment fonctionne le logiciel Quartus. J'ai vraiment aimé réaliser ce projet et j'ai vraiment passer du temps dessus. Ce type de projet nous permet de nous rapprocher à notre futur métier.

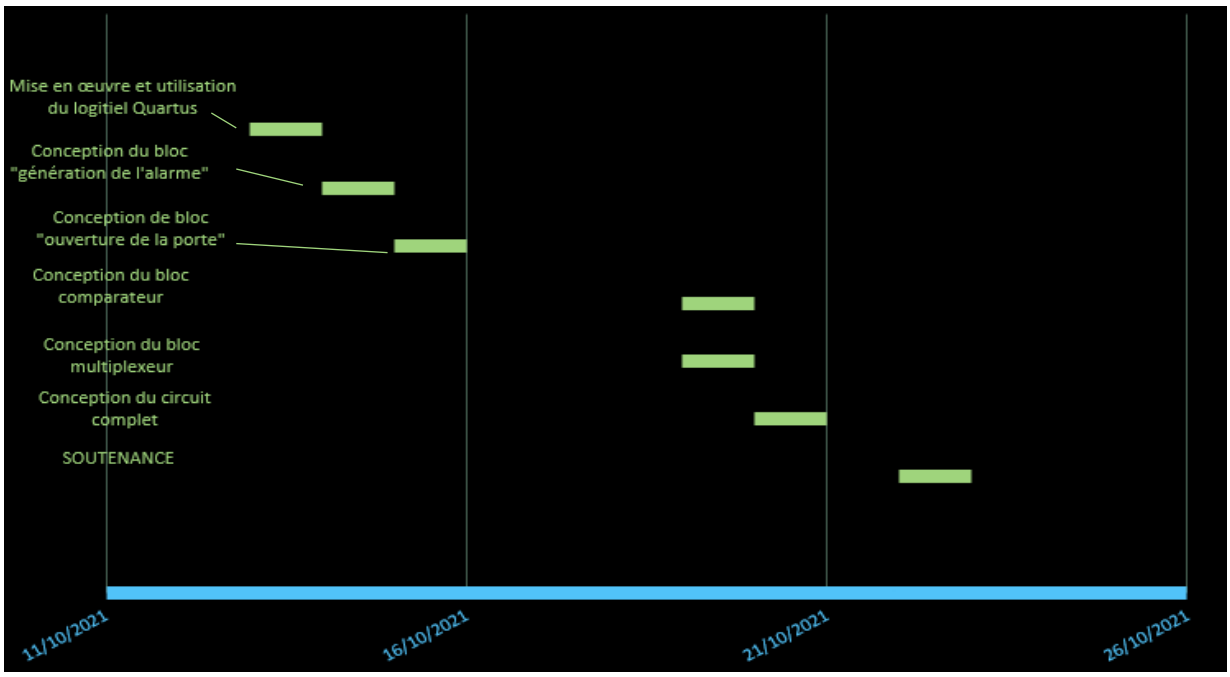


Figure 24 : Diagramme de Gantt du projet