Nom:PENG Prenom:Bo

Rapport
Projet Programmation synchrone
Déploiement du mât de communication

Couteau thermique

Dans ce parti, on a 2 version , la version sans automate et la version avec automate.Tous ce deux versions a la même fonction. Donc pour chaque version, on a 2 entrée qui et power1 et power2. 2 sorti qui et cut et error.

Sans automate

Dan la version sans automate, on fait une or entre power1 et power2 pour obtenir il y a au moins une power est active. Et on fait une et entre power1 et power2 pour obtenir tous les deux power est active. Puis, on utilise le compteur pour compter le power est active pendant combien de cycle.pour le cut, c'est 2, pour le error , c'est 4. Et s'il y a un cycle que le power n'a pas active. Le compter va rester.

Pour le cut, si on a cut dans un cycle, les autre cycle va cut aussi. Donc on a mis un or de FBY de cut avec le résultat de compteur.

Pour le error. S'il error a cause de la alimentation de power pendant 4 cycle. Il va toujours reste error. Donc on a ajouter une truc comme on a ajouter dans le cut. Pour les autre situation, il ne va pas toujours reste error.

Pour tous les cut et error. On change le valeur apres une cycle. C'est a dire si le power a activer pendent 2 cycle, le cut va égaler true depuis le 3eme cycle.Pour ça , on a ajouter un FBY avant chaque situation.

Avec uniquement des automates

Dans le version avec automate,on a séparer le cut et power dans 2 stage machine.

Pour le cut. On a 3 état. Dans le 1ere état, le cut = false. Quand on a power1 or power2, on transmettre a le état cut1.si le power1 or power2 !=true, on va retour a la état avant. Si non, on va aller a le etat cut2.A ce moment la, le cut va toujours =true.

Pour le error,on a 3 choses qui va défini le résultat de error. Le power1, le power2 et le cut. Ci-dessous une tableau.

Power1	Power2	Cut	Error	+1?	
0	1	1	1	+1	p1 and p2 or (cut and(p1 xor p2))
1	0	1	1	+1	
1	1	1	1	+1	
1	1	0	1	+1	
0	1	0	0	+1	not cut and(p1 xor p2)
1	0	0	0	+1	
0	0	0	0		not(p1 or p2)
0	0	1	0		

Avec ce tableau , on a fait 8 état.

Le état no_error qui est le état initial. Et on a deux parti pour compter le numéro de la activation du power. En haut, on a error1, error2 et error3,qui correspond le **not cut and(p1 xor p2).**En bas, on a state8, state9, state10 qui correspond le **p1 and p2 or(cut and(p1 xor p2))**
A cause de ca, chaque fois si on a **not(p1 or p2)**, on va retour a no_error qui est le etat initial. Si on a **not cut and (p1 xor p2**), on transmettre a les état en haut, si on a **p1 and p2 or(cut and(p1 xor p2))**, on transmettre les état en bas. Et pour les état en haut et en bas, les 1ere états en haut et en bas vas transmettre les 2eme état s en haut et en bas. Par exemple , quand je suis dans error1 , si on a **not cut and(p1 xor p2)**, on transmettre a error2, si on a **p1 and p2 or(cut and(p1 xor p2))**, on transmettre a state9. Quand on a le state 11. c'est a dire il a feu. Le error va toujour = true.

Parce que on change le valeur dans le cycle prochain, donc tous les transaction sont weak.

De puis on a une opérateur Couteau_themique_Ver pour vérifier les deux versions sont équivalents.On obtenir le résultat qui est valide.

Batterie

D'abord ,on a besoin de verfier si le requested_power >=0 s'il <0 on le mettre a 0.

Pour le batterie, chaque fois si le power_load <6 , on +1, sinon, on fait rien. Puis on comparer ça avec le requested_power.si il >= requested_power, on fait power_load=power_load-request_power,et le delivered_power va = requested_power, Si non. Le delivered_power va = 0.

A la fin on a fait une operateur de Obs_batterie pour verfier 0<=power_loade<=6

Contrôle de puissance

Pour le contrôle de puissance, on utilise un opérateur de Bool_Int pour transférer les bool a int, et chaque fois on l'ajoute avec an acc. Donc dans le power, on a une entrée de bool^n. On fait une fold de diamantions de n pour Bool_Int. Puis on obtenir le numéro de true dans le bool^n qui va être le requested_power de batterie. Si le delivered_power de batterie =0 , on mettre le on_delivered de power a un false^n, sinon, on fait on_delivered=on_requested.
Et on a fait un opérateur de p4 pour valider par simulation une instance n=4 de l'opérateur « power ».

Pour le batterie imparfaite, on a crée a opérateur de unsafe_battery, il utilise le opérateur de batterie et ajouter une entrée de faiture. Si le faiture = true, le entrée de batterie va =0 , sinon, il va =requested_power.

Pour le contrôle de puissance imparfaite, on fait comme le power le seulement modification est on remplace le batterie a unsafe_battery.

Système

Pour le mystes. Il nous reste le contrôleur. Pour le contrôleur, on a 4 entrée de les 4 cut de TK.on inverser ces 4 entre d'abord. On ajouter une compteur dans ce contrôleur, il compte de 1 a 4. si le sorti de ce compteur =1, on mettre le 5eme chiffre de sorti1 a false, c'est a dire a ce moment la, on utilise le 1eme power. Quand le sorti de ce compteur =3, on mettre le 5 eme chiffre de sorti2 a false, c'est a dire a ce moment la, on utilise le 2eme power. Pour préciser, on utilise power1 a un cycle, puis on attente, puis on utilise le power2 puis on attente, puis....on fait comment avant.

Quand on fait la simulation ,on peut avoir aucun de error ou cut.