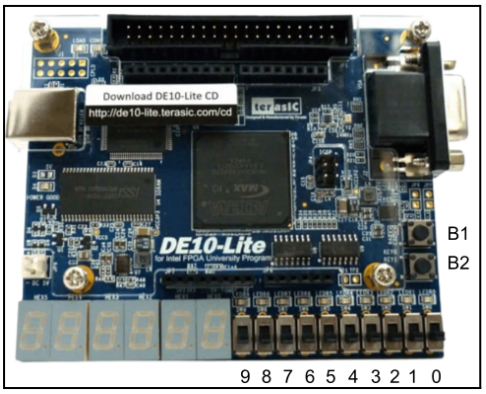
Christian Matala-tala et Gregory Malyk

RAPPORT PROJET SYSTEMES LOGIQUES



Pour notre projet, nous avons implémenté les fonctionnalités suivantes à notre carte: les Fonctionnalités de base telles que l'affichage et Set de l'heure, et le Réveil et alarme sonore. Les Fonctionnalités additionnelles telles qu'un Chronomètre et un Countdown. Finalement deux Fonctions complexes: les Messages défilants et les Animations de LEDs.

Nomenclature du rapport:

Ci-dessus les **DIP-Switch numérotés de 0 à 9**, puis les deux **boutons poussoir**, **B1 et B2.**

1) Mode d'emploi:

Démarrage: Au démarrage du programme, un message de bienvenue *"HELLO"* défile.

Mode 01 Time:

- ➔ Switch 0 : Passer en mode réglage de l'heure avec utilisation de l'encodeur pour changer les valeurs (voir plus loin) / Confirmer le réglage de l'heure et revenir à son affichage.

Mode 02 Chronomètre:

- ➔ *Switch 0* : Réinitialiser le chronomètre.
- ➔ *B1 :* Enclencher/Mettre en pause le Chronomètre

Mode 03 Timer:

- ➔ Switch 0 : Passer en mode réglage du Timer avec utilisation de l'encodeur pour changer les valeurs (voir plus loin) / Confirmer le réglage du Timer et revenir à son affichage.
- ➔ B1 : Enclencher/Mettre en pause le Timer
- ➔ Switch 08 : Éteindre la sonnerie du buzzer.

Mode 04 Alarme:

- ➔ Switch 0 : Passer en mode réglage du Timer avec utilisation de l'encodeur pour changer les valeurs (voir plus loin) / Confirmer le réglage de l'alarme et revenir à son affichage. **Attention à bien mettre le Switch 2 à zéro avant de régler un Alarme.**
- ➔ Switch 2 : Activer/Désactiver l'alarme.
- ➔

Encodeur:

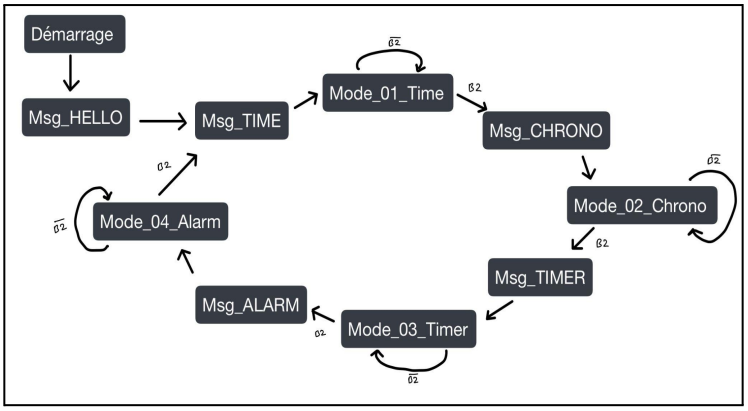
Le réglage de l'heure, du timer ou de l'alarme se fait avec l'encodeur.

- ➔ Bouton de l'encodeur : alterner entre le réglage des secondes/minutes/heures
- ➔ Pivoter l'encodeur dans le sens trigonométrique : décrémenter la valeur choisie
- ➔ Pivoter l'encodeur dans le sens inverse : incrémenter la valeur choisie.
- ➔

Switch 08 → Éteindre la sonnerie

Switch 09 → Program Reset

2) Specifications Techniques



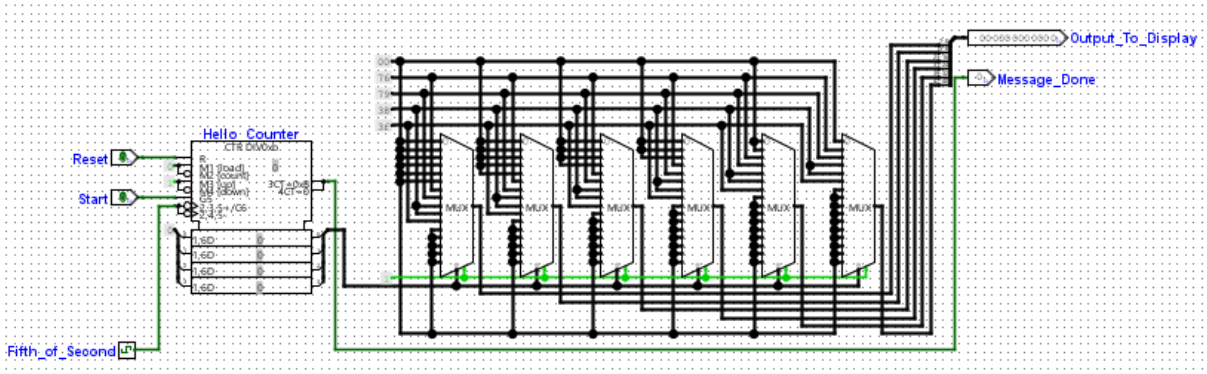
Machine d'état fini générale:

Le fichier main constitue la machine d'état finie générale et se charge de contrôler le chemin les valeurs d'entrée (DIP-Switch et bouton poussoir) vers le bloc du mode courant. Il contrôle aussi l'animation des leds à afficher en fonction du mode courant. Tous ces contrôles sont fait à l'aide de Multiplexers qui prennent la valeur du mode actuel (nombre entre 0 et 8) en guise de source. Le circuit *Mode_FSM* se charge lui de changer la valeur du mode a chaque pression du bouton 2 et de faire défiler sur les 6 afficheurs 7-Segments le nom du mode suivant selon la séquence illustrée par le schéma ci-dessus.

Le Switch 09 envoie un signal de reset a chaque sous blocs, y compris le circuit *Mode_FSM* afin de tout réinitialiser. Cet état n'a pas été représenté sur le schéma pour y garder un certaine lisibilité et simplicité.

Affichage de caractères:

5 Circuits indépendants ont été développés pour faire défiler le nom des cinq modes de droite à gauche (*TIME, CHRONO, TIMER, ALARM*). Les circuits chargés d'afficher un message défilant ont tous la même structure: composé de 6 Multiplexers et 1 compteur (voir Schéma ci-dessous), et son chacun appelé dans la Machine d'état fini générale à chaque changement de mode afin d'informer l'utilisateur vers quel mode il se dirige. Chaque Multiplexer va en fait afficher chaque lettre du message une par une, mais de manière décalée afin de faire apparaître un mouvement horizontal de défilement. Pour ce faire, sachant que le select des 6 Multiplexers est la même, il faut décaler la position du message sur les entrées des multiplexers, Comme illustré sur la capture d'écran de logisim ci dessous. Le compteur lui se chargera de faire cycler les bit select des multiplexers entre 0 et la longueur du message. La clock quand a elle détermine la vitesse de défilement du message.

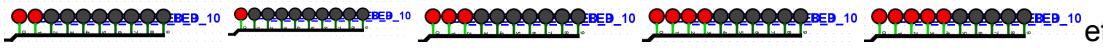


Fonctionnalitées en "arrière plan":

Les modes Timer et Alarme permettent de définir respectivement des alarmes et des timers, mais les sonneries causées par ceux-ci peuvent retentir indépendamment du mode actuel du programme. Les compteurs des trois autres modes, Time, Chrono et Timer vont continuer de s'exécuter eux aussi indépendamment du mode actuel. On peut donc dire que notre

programme effectue des tâches en arrière plan. Par exemple, le chronomètre tourne toujours même si l'utilisateur est dans le mode Alarm.

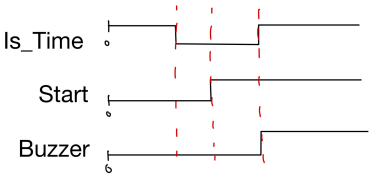
Animations LEDS:
Lorsque l'utilisateur est dans les modes Time, Chrono et Timer, les LEDs sont activées une par une (de gauche à droite pour les modes Time et Chrono et de droite à gauche pour le mode Timer) chaque seconde pour créer une animation progressive qui reboucle chaque seconde, illustrée sur les captures d'écran chronologiques ci-dessous.


etc…

3) Solutions techniques originales apportées à la résolution de problèmes rencontrés dans le développement

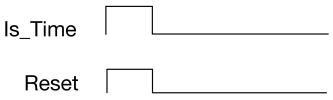
Gated Clocks:
La création du projet sans utilisation d'une seule gated clock n'a pas été facile mais nous avons à chaque fois trouvé des moyens de contourner leur utilisation. L'utilisation d'une variable de contrôle supplémentaire, présente soit dans les compteurs pré-fait de logisim, soit ajoutée comme entrée à la table de vérité de la logique de transition suffisait en général à résoudre le problème.

Sonnerie du Timer:



Pour empêcher le timer de sonner au démarrage du programme (car le timer est à 00h00 et le bit Is_Time passe donc a 1) on utilise l'entré Start (relié au Switch 0) comme bit de contrôle pour bloquer une sonnerie avant d'avoir appuyé au moins une fois sur start.

Sonnerie de l'Alarme:



Fais sonner le buzzer lorsque le temps que l'utilisateur a réglé pour l'alarme est égal au temps de l'horloge. Comme ces deux valeurs sont a 00:00:00 au démarrage, l'Alarme s'est mise a retentir. Nous avons résolu ce problème en créant une seule et unique impulsion au démarrage qui va qui forcer la sortie du buzzer à zéro le temps que l'heure de soit plus 00:00:00 après la mise en marche de la carte.

Synchronisation de l'animations des LEDs:
Les circuits LtR_FSM et RtL_FSM (Left To Right et Right To Left), chargés de l'animation progressive des leds chaque seconde étant piloté par une clock différente de celle qui pilote le module à animer correspondant (TIME, CHRONO ou TIMER), l'animation n'était pas synchronisée avec les secondes de ceux-ci. Pour résoudre ce problème, le module (TIME, CHRONO ou TIMER) envoie à chaque changement de seconde un signal au module d'animation correspondant pour réinitialiser l'animation de LEDS, ce qui rends donc ces deux modes synchrones.