Éléments de réflexion pour le projet

Objectifs du projet

- Exploiter des notions vues en cours et analyser leur intégration sous forme d'un circuit intégré
- Implémenter des modules analogiques, numériques et de conversion analogique/numérique
- Durant mon absence, implémenter la phase 1

Cahier des charges de base (dia 2)

Architecture de base (dia 3), imposée pour des raisons de temps et de simplicité. Elle est à l'image du découpage proposé dans le cahier des charges

Phase 1: partie gestion des dates (dia 5)

- Pas de contraintes d'interface
- La gestion des dates suppose deux catégories de fonctions:
  - Le réglage des dates
  - L'évolution autonome des dates
- Les fonctions à régler (selon un ordre logique)

Les autres phases seront développées au fur et à mesure (ordre lié au cours)

- Acquisition des informations (choix des capteurs - analogique)
- Conditionnement des mesures (amplification, filtrage: analogique)
- échantillonnage et conversion
- Multiplexage des signaux (analogique)
- gestion des mesures (stockage: numérique)
- Traitement numérique des mesures stockées (numérique)

Cahier des charges du projet

Présentation succincte du projet qui s'affinera dans les semaines à venir.

L'objectif est de créer un chip de type ASIC, créé pour aider les automobilistes.
Il dispose de deux principaux groupes de fonctions:

- La gestion du temps qui fait l'objet de la phase 1
- La gestion des paramètres du véhicule. Elle inclut la consommation de la voiture à partir d'une série de capteurs (tachymètre pour la vitesse du véhicule et la distance parcourue, jauge volumétrique pour la quantité d'essence disponible, débit-mètre pour la quantité d'essence injectée, accéléromètre,….

Le choix n'est pas exhaustif et pourra être discuté.

La gestion du temps est explicitement décrite dans la dia 4.

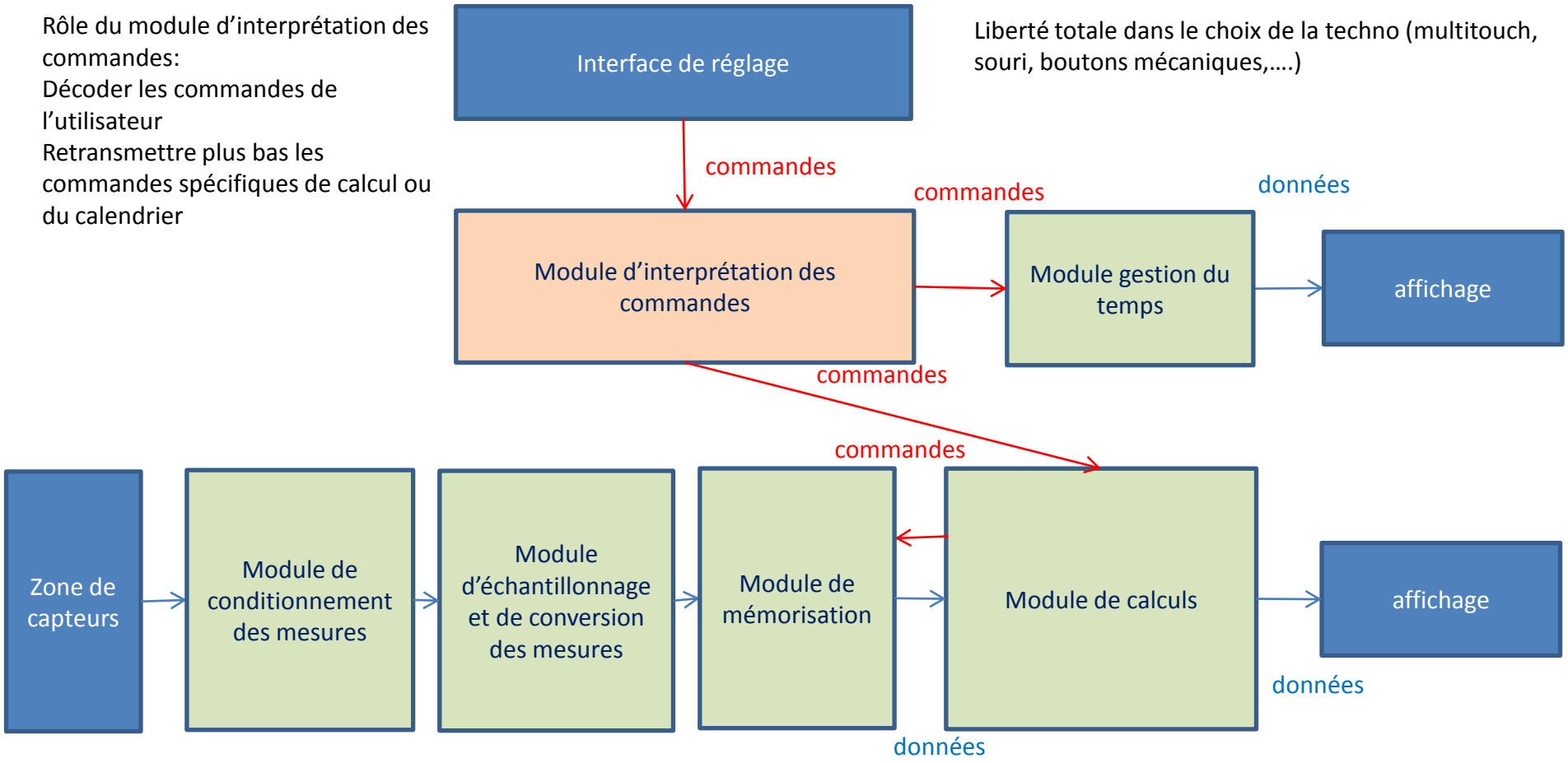
Gestion des paramètres du véhicule:

- Les différents capteurs génèrent des mesures qu'il faudra conditionner, convertir en numérique et stocker en mémoire. On stockage en mémoire doit permettre d'identifier sans ambiguïté la localisation de chaque type de mesures sans pour autant solliciter une structure trop complexe. Pour simplifier la partie structurelle on impose les contraintes suivantes:
  - Le taux d'échantillonnage est le même pour l'ensemble des types de mesures.
  - Un seul convertisseur pour l'ensemble des mesures
  - Une seule zone mémoire pour l'ensemble des mesures
- Les différentes mesures stockées sont ensuite analysées sur commande de l'utilisateur pour générer des comptes rendus de la vitesse instantanée, vitesse moyenne depuis le km 0, vitesse moyenne sur un trajet, consommation instantanée, consommation moyenne depuis le km 0, consommation moyenne sur un trajet,…

Architecture du projet

Synoptique du projet: version simplifiée

Rôle du module d'interprétation des commandes:
Décoder les commandes de l'utilisateur
Retransmettre plus bas les commandes spécifiques de calcul ou du calendrier



Liberté totale dans le choix de la techno (multitouch, souri, boutons mécaniques,….)

On part du principe que la version simple suppose deux unités d'affichage indépendantes.
Possibilité d'avoir une seule unité d'affichage prise en charge par le module de réglage. Ce qui suppose des échanges plus complexes entre les sous modules de calcul et de gestion du temps, et d'interprétation des commandes.

Phase 1: partie gestion des dates

Remarques:

- La partie interface graphique ne présente pas de contraintes ici. On suppose qu'un code est généré correspondant à la fonction que l'on souhaite exécuter
- Les commandes issues de la même interface, correspondent soit à un réglage des dates, soit au traitement des mesures (voir lors des semaines suivantes).
- On part du principe que les commandes n'ont pas besoin d'être stockées. L'utilisateur attend que la commande courante soit exécutée avant de sélectionner une nouvelle commande

Fonctions de base de gestion des dates

- Le module des dates est soit en mode réglage, soit en mode exécution.
- l'utilisateur peut modifier l'année, le mois, le jour, l'heure, les minutes éventuellement les secondes (l'ordre et le parallélisme sont laissés au choix de chaque groupe
- L'utilisateur peut sélectionner une fonction chronomètre incluant le reset, le démarrage, le stop de l'affichage (mais sans provoquer l'arrêt du chronomètre qui continue son évolution), le redémarrage de l'affichage du chronomètre (qui poursuivait son exécution), le stop définitif du chronomètre, la sortie du mode chronomètre
- L'évolution continuelle du temps à partir de son dernier réglage. L'affichage de l'heure et du jour peut être obtenu globalement.
- ATTENTION: lorsque la fonction chronomètre est sélectionnée, le temps courant continue d'évoluer et doit pouvoir être affiché correctement dès l'arrêt définitif du chronomètre.

On demande:

- de créer les algorithmes de gestion du temps et de les implémenter à l'aide d'un langage procédural ou orienté objet au choix de chaque groupe,
- puis de simuler le comportement des algorithmes.
- L'interface sera émulée par des touches sur le clavier.

Remarque: Le programme écrit en entier, nous permettra ultérieurement de déterminer la complexité de son intégration sous forme de circuit intégré. Sa simulation suggère plutôt l'exploitation du multithreading