Projet ETML-ES – Modification POBJ

PROJET:	2202_Interface_ StarterKit-ARM (1701)		
Entreprise/Client:	ETML	**Département:**	SLO
Demandé par (Prénom, Nom):	M.Bovey	**Date:**	19.03.2024
Objet (No ou réf, pièce, PCB...)	StarterKit-ARM (1701)		
Version à modifier:	V1.0		

Auteur (ETML-ES):	Samuel Pitton	**Filière:**	SLO
Nouvelle version:	V1.1	**Date:**	19.03.2024

1 Projet

1.1 Description sommaire du projet à modifier

En quelques phrases, que fait le projet ? Fonctions principales ?

Le projet permet de communiquer avec le Starter kit-ARM de l'ES via son port série avec un programme en C sur Visual studio. Le but et de pouvoir concevoir un programme sur Visual studio qui permettra de piloter le starter Kit-ARM.

1.2 Référence conception

K:\ES\PROJETS\SLO\1740_KitArmM0\17400_KitArmM0_Mainboard\soft\2202_Interface_ StarterKit-ARM (1701)

2 Objectifs principaux des modifications ou ajouts

- Il faut modifier la structure du projet Visual studio et Keil uvison en mettant au propre et modifier certaine fonction pour une meilleurs compréhension du code.

- Puis ajouter des librairies Visual studio de fonction propre à chaque fonctionnalité du kit ARM (bouton, valeur du DAC, capteur de température).

- Création de fonction et librairie sur Keil uvison pour permettre l'envoie de donnée. (État des bouton valeur du DAC, valeur du capteur de température).

3 Résumé de l'état actuel

Actuellement, le principe de la communication série est opérationnel seulement du PC vers le microcontrôleur. Cependant, le programme est encore à l'état de démonstration. Il est nécessaire que j'implémente de nouvelles fonctionnalités et améliore certaines fonctions existantes.
Car pour le moment il est seulement possible d'écrire une ligne de texte sur le LCD et allumer les LED.

4 Tâches à réaliser

4.1 Livrable

Projet Visulal Studio et Keil uvison contenant les librairies de fonction firmware et software (.c et .h) permettant le pilotage des périphériques du stater kit ARM, contenant le numéro de la nouvelle version et la documentation avec une architecture globale du projet et un structogramme de chaque fonction de pilotage. Et fournir un programme exemple.

4.2 Archivage sur Github

https://github.com/Toxik24/2202_Interface_-StarterKit-ARM-1701-

4.3 Table des tâches à réaliser

#	Priorité	Description	Fait	Vu
1	**1**	Lire le rapport effectué par Subahskanth Mohanasarma	SPN	OK
2	**1**	Tester le fonctionnement du projet	SPN	OK
4	**1**	Modification des bugs	SPN	OK
5	**1**	Création de défininition pour rendre le code plus lisible et compréhensible.	SPN	OK
6	**1**	Création de librairie VS (.c et .h) pour les fonctions des LED (Structo + code + tester).		
7	**1**	Création de librairie VS (.c et .h) pour les fonctions du LCD (Structo + code + tester		
8	**1**	Création de librairie VS (.c et .h) pour les fonctions la réception des information série (Structo + code + tester)		
9	**1**	Création de librairie Keil uvision (.c et .h) pour les fonctions la transmission des information série (Structo + code + tester)		
10	**1**	Création de librairie VS (.c et .h) pour les fonctions la lecture des boutons (Structo + code + tester)		
11	**1**	Création de librairie Keil uvision (.c et .h) pour les fonctions la lecture et la transmission des boutons (Structo + code + tester)		
12	**2**	Création de librairie VS (.c et .h) pour les fonctions la lecture du capteur de température. (Structo + code + tester)		
13	**2**	Création de librairie Keil uvision (.c et .h) pour les fonctions la lecture et la transmission du capteur de température (Structo + code + tester)		
14	**3**	Création de librairie VS (.c et .h) pour les fonctions la lecture du capteur de l'ADC. (Structo + code + tester)		
15	**3**	Création de librairie Keil uvision (.c et .h) pour les fonctions la lecture et la transmission de l'ADC (Structo + code + tester)		
16	**1**	Création d'un programme de démonstration pour tester les nouvelles fonctionnalités (.c et .h) dans VS Structo + code + tester)		

17	**1**	Documenter tout le travail effectué.		

1 = Elevé 2 = Moyen 3 = Faible

5 Détail des modifications ou ajouts

#	Description	Fait	Approuvé
1	Réorganiser le projet en créant des librairies VS propres à chaque fonction du microcontrôleur. Exemple : LCD.C, LED.C, Comm_serie.c	SPN	
2	Modification de la fonction d'écriture dans le LCD. Pour permettre de limiter le nombre de caractère.		
3	Ajouter une fonction à la librairie VS et Keil du LCD pour permettre le choix de la ligne d'écriture du LCD.		
4	Mettre en place une librairie VS et keil pour permettre communication série du microcontrôleur vers le PC. (uC -> PC)		
5	Crée une librairie de fonction VS et Keil qui permettre la lecture de l'état des boutons poussoir (S2 à S5). Exemple :Bouton.c		
6	Crée une librairie de fonction VS et Keil qui permettre la lecture des capteurs de température (LM70CIMMX-5). Exemple : Sens_temp.c		
7	Crée une librairie de fonction VS et Keil qui permettre la lecture de L ADC Exemple : ADC.c		

6 Matériel nécessaire

- Visual studio version 2022 et version compilateur C++ 2022 00482-90000-00000-AA702
- Keil uvision 5 et version compilateur V5.06 update 7 (build 960)
- Starter Kit-ARM ES
- Débuggeur STLINK

6.1 Stockage du fichier

Ce fichier sera stocké à la racine du dossier **/doc** du projet 2202_Interface_ StarterKit-ARM (1701)

Ainsi, tous les fichiers de modifications des pièces ou PCBs faisant partie du projet sont centralisés dans le même répertoire. La numérotation devient implicite.

6.2 Stockage des logiciels

Les codes sources sont stockés ici :

racine du dossier **/soft**