INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

Compte rendu de Bureau d'Etude d'Informatique Embarqué - Microcontrôleur

MISE EN PLACE D'UN VOILIER DE MODELISME COMMANDÉ PAR UN STM32

Laschon Nathan

Gadanho Paul

Cunnac Florian

Challoub Yorgo

Guillermin Antoine

4 IMACS-AE/SE Groupe 3

SOMMAIRE

INTRODUCTION

Lors de ce Bureau d'Etude, nous avons réalisé une maquette de voilier piloté par une télécommande avec transmission bidirectionnelle d'informations (batterie, angles…), gestion automatique des voiles en fonction du vent et un système anti-chavirement.

Explications du logiciel

Description de l'application (cf figure 1)

L'application se décompose en 3 parties : une première partie permet de configurer les différents services, une seconde définit les différents callbacks liés aux services (essentiellement ceux liés aux Systick) et enfin le démarrage desdits services.

Le programme commence par initialiser les différents services ainsi que les fonctions de callback nécessaire (cf rectangle Setup()).

Puis il lance les différents services afin qu'il soit actifs (cf rectangle Start()).

Ensuite le programme effectue une boucle infinie (cf rectangle Loop()) qui permet de maintenir le microcontrôleur en activité. Rien n'est exécuté dans cette boucle puisque l'ensemble des services sont appelés par des interruptions ou en appellent.

Au même niveau que la boucle loop, le systick est maintenant configuré et actif. Il va exécuter trois fonctions principales régulièrement.

-Batterie, qui va lire l'état de la batterie et envoyer un message si celui-ci est trop faible toutes les 2s.

-Bordage, qui vérifie si le voilier chavire et en fonction oriente les voiles comme demandé toutes les 20ms.

-General_infos qui envoie les informations souhaitées toutes les 3s.

D'autre part le programme reçoit des informations (cf rectangle Communication) et fait tourner le plateau en conséquence.

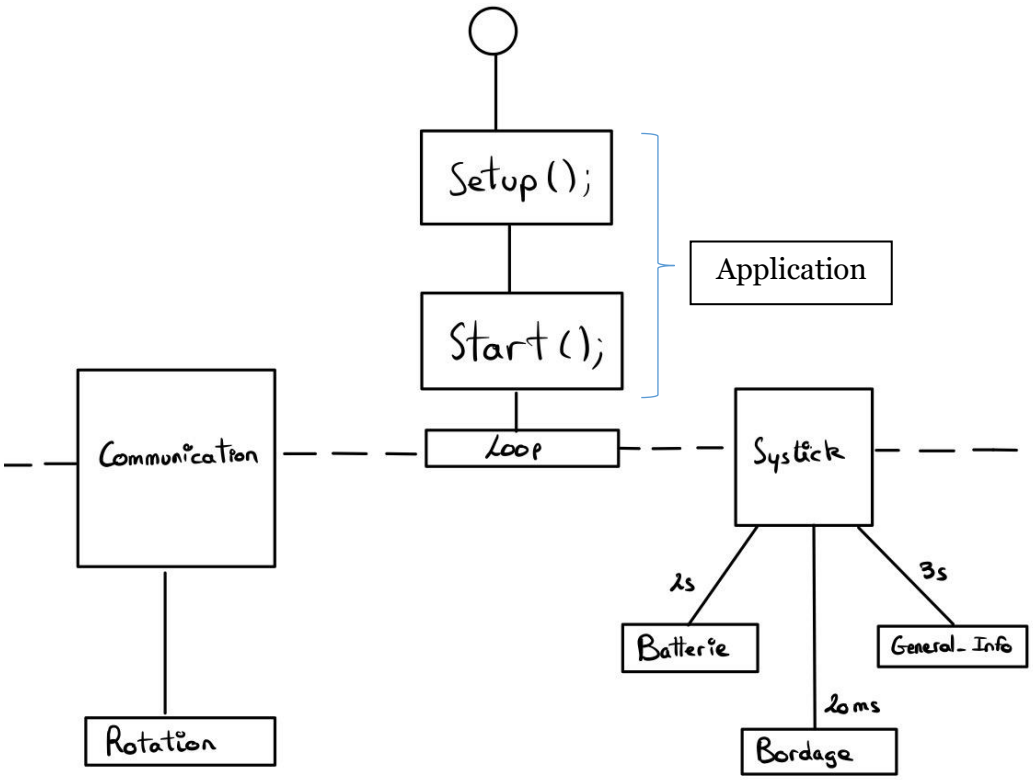

Figure 1 : Schéma simplifié du fonctionnement du microcontrôleur.

Description brève des différents services

Le service de la Communication permet à la fois d'envoyer et recevoir des données de la télécommande via l'UART, il est possible d'envoyer et recevoir plusieurs types de données, notamment l'heure récupérée par le service RTC.

Le service de la Rotation permet la rotation du plateau, en pratique, il pilote un GPIO pour le sens de rotation et applique une PWM sur un second GPIO pour la vitesse.

Le service de la Batterie renvoie un pourcentage de batterie par lecture via un ADC.

Le service du Systick, permet dans un premier temps l'enregistrement de fonctions appelées à un temps T donné et dans un second temps, appelle ces fonctions périodiquement.

Le service de la Girouette, récupère l'angle du vent afin d'adapter celui des voiles.

Le service du Bordage permet la gestion des voiles en fonction de l'angle de la girouette ou bien de lâcher les voiles complétement en cas de chavirement.

Le service de l'Accéléromètre, récupère l'angle de roulis du voilier afin de détecter un potentiel chavirement.

Le service de la Real Time Clock, permet de définir l'heure dans un premier temps et de la lire périodiquement.

Les drivers quant à eux permettent de communiquer avec les périphériques éponymes.

Remarques :

- Configurations :
  - pour que le programme fonctionne en « simulé », il faut définir le MODE à 1 dans le fichier conf.h, 2 pour le mode « réel ». En effet, lorsque la RTC n'est pas connecté au microcontrôleur, le driver lié à celle-ci bloque le fonctionnement du programme. Le mode 1 désactive la RTC.
  - Il y a 2 modes configurables dans le conf.h pour le bordage en fonction de la position du servo-moteur sur la girouette.
  - De même il y a 2 modes configurables dans le conf.h pour la rotation en fonction du câblage du moteur de rotation.
- Programmation « orienté objet »

Informations sur la "Programmation Orientée Objet" dans certains services ainsi que dans le driver UART :

Le langage C n'étant pas un langage orienté objet, il a fallu établir des règles de programmation afin d'obtenir une POO proche de la POO traditionnelle. Pour ce faire j'ai suivi la norme proposée par le site developpez.com : https://chgi.developpez.com/c/objet/

Bien sûr cette norme ne fait qu'approcher la "vraie" POO mais ce choix a été fait pour faciliter le développement, la lisibilité et la maintenance(théorique) de ce code.

Plan du dossier

\Application\conf.h : fichier contenant toutes les configurations des services

\Application\Src

\main.c : à la manière d'Arduino, contient un setup et une loop. Dans le setup, on initialise et démarre l'application

\app.h

\app.c : programme de l'application en imitant l'orienté objet du C++ (lien de la méthode dans les commentaires du main.C)

\Services

\Include : fichiers head des services

\Src : fichiers sources des services

\Drivers

\Include : fichiers head des drivers

\Src : fichiers sources des drivers

Github

https://github.com/aguiller31/Voilier

Fonctionnalités du cahier des charges mises en place

Bordage automatique	✓
Orientation du voilier, cap	✓
Système anti-chavirement	✓
Transmission d'informations	✓
Transmission de l'heure	✓

Liste des périphériques

Services\Périphériques	TIMER				UART			ADC		GPIO				I2C		SPI	
	1	2	3	4	1	2	3	1	2	A	B	C	D	1	2	1	2
Communication	-	-	-	-	-	-	✓	-	-	-	✓	-	-	-	-	-	-
Rotation	-	-	-	✓	-	-	-	-	-	-	✓	✓	-	-	-	-	-
Batterie	-	-	-	-	-	-	-	✓	-	-	-	✓	-	-	-	-	-
Systick	✓	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Bordage	-	-	✓	-	-	-	-	-	-	✓	-	-	-	-	-	-	-
Girouette	-	✓	-	-	-	-	-	-	-	✓	-	✓	-	-	-	-	-
Horloge	-	-	-	-	-	-	-	-	-	-	-	-	-	✓	-	-	-
Accéléromètre	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	✓	-

Liste des IO

Services\IO	Détail	Alternate Function	Pin Name	Configuration IO
Communication	Transmission/Réception USART	USART3_TX	PB10	AltOut_Ppull
		USART3_RX	PB11	In_Floating
Rotation	Sens de la rotation	-	PC6	Out_Ppull
	Vitesse de la rotation (PWM)	TIM4_CH3	PB8	AltOut_Ppull
Batterie	Lecture analogique (ADC)	ADC12_IN14	PC4	In_Analog
Systick	-	-	-	-
Bordage	PWM	TIM3_CH3	PB0	AltOut_Ppull
Girouette	Signal A	TIM2_CH1	PA0	In_PullDown

	Signal B	TIM2_CH2	PA1	In_PullDown
	RST	-	PC9	In_Floating
RTC	TX	I2C1_SCL	PB6	Out_OD
	RX	I2C1_SDA	PB7	Out_OD
Accéléromètre	CS	SPI1_NSS	PA4	AltOut_Ppull
	CLK	SPI1_SCK	PA5	AltOut_Ppull
	RX	SPI1_MISO	PA6	In_Floating
	TX	SPI1_MOSI	PA7	AltOut_Ppull

Répartition des tâches par étudiant

Tâches	Paul Gadanho	Florian Cunnac	Nathan Laschon	Yorgo Challoub	Antoine Guillermin
Application	20 %	20 %	20 %	20 %	20 %
Communication	-	-	-	50 %	50 %
Rotation	-	-	-	50 %	50 %
Batterie	-	-	-	0 %	100 %
Systick	-	-	-	0 %	100 %
Bordage	-	50 %	50 %	-	-
Girouette	-	50 %	50 %	-	-
RTC	100 %	-	-	-	-
Accéléromètre	100 %	-	-	-	-
GPIO	95 %	100 %	100 %	95 %	95 %
UART	-	-	-	50 %	50 %
TIMER	90 %	100 %	100 %	90 %	90 %
ADC	100 %	100 %	100 %	100 %	100 %

Application	Services	Drivers

Diagramme des classes du voilier réalisé par
Laschon Nathan
Gadanho Paul
Cunnac Florian
Challoub Yorgo
Guillermin Antoine
4 IMACS-AE/SE Groupe 3

Application

Services

Drivers