

# Nauteff Conception du matériel et du logiciel

Emmanuel Gautier

21 novembre 2025

## **Résumé**

Nauteff P-1 est une maquette de pilote automatique pour navires. Il est destiné à développer et tester son matériel et ses algorithmes. Ce document contient les informations relatives à la conception de cette maquette.

# Historique du document

| Version | Date                             | Description                                   |
|---------|----------------------------------|---|
|         | 15 avril 2020<br>16 février 2022 | Début de rédaction<br>Début rédaction pour P2 |

# Table des matières

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>Buts du document</b>                                 | <b>5</b>  |
| 1.1      | But . . . . .   | 5         |
| 1.2      | Guide de lecture . . . . .                              | 5         |
| <b>2</b> | <b>Documents applicables et de référence</b>            | <b>6</b>  |
| 2.1      | Documents applicables . . . . .                         | 6         |
| 2.2      | Documents de référence . . . . .                        | 6         |
| <b>3</b> | <b>Matériel</b>   | <b>7</b>  |
| 3.1      | STM32F446RE . . . . .                                   | 7         |
| 3.1.1    | Cur . . . . .   | 7         |
| 3.1.2    | Mémoire . . . . .                                       | 7         |
| 3.1.3    | Horlogerie . . . . .                                    | 7         |
| 3.1.4    | Interruptions . . . . .                                 | 8         |
| 3.1.5    | Affectation des broches du STM32 . . . . .              | 8         |
| 3.1.6    | Organisation de la mémoire STM32 . . . . .              | 10        |
| 3.2      | Capteurs MEMS . . . . .                                 | 10        |
| 3.3      | Commande de l'actionneur . . . . .                      | 10        |
| <b>4</b> | <b>Architecture générale de l'application</b>           | <b>11</b> |
| 4.1      | Architecture statique . . . . .                         | 11        |
| 4.1.1    | Système . . . . .                                       | 11        |
| 4.1.2    | Pilotes de périphériques . . . . .                      | 11        |
| 4.1.3    | Tâches . . . . .  | 11        |
| 4.1.4    | Calculs . . . . .                                       | 11        |
| 4.1.5    | Traitements des trames NMEA0183 . . . . .               | 11        |
| 4.1.6    | IMU . . . . .   | 11        |
| 4.2      | Relations dynamiques . . . . .                          | 11        |
| 4.3      | Justification des choix d'architecture . . . . .        | 11        |
| <b>5</b> | <b>Contexte de la conception</b>                        | <b>12</b> |
| 5.1      | présentation de l'application . . . . .                 | 12        |
| 5.2      | Principales exigences applicables . . . . .             | 12        |
| 5.2.1    | Exigences liées à la sûreté de fonctionnement . . . . . | 12        |

|          |   |           |
|----------|---|-----------|
| 5.2.2    | Architecture matérielle opérationnelle . . . . .    | 12        |
| 5.2.3    | Logiciels imposés . . . . .                         | 13        |
| 5.2.4    | Interface avec d'autres applications . . . . .      | 13        |
| 5.3      | Contraintes de développement . . . . .              | 13        |
| 5.3.1    | Méthode et formalisme . . . . .                     | 13        |
| 5.3.2    | Langage de programmation . . . . .                  | 13        |
| 5.3.3    | Logiciels extérieurs . . . . .                      | 13        |
| 5.3.4    | Outils . . . . .                                    | 13        |
| 5.3.5    | Environnement matériel de développement . . . . .   | 13        |
| <b>6</b> | <b>Architecture du logiciel</b>                     | <b>15</b> |
| 6.1      | architecture statique . . . . .                     | 15        |
| 6.1.1    | Noyau temps réel : FreeRTOS . . . . .               | 15        |
| 6.1.2    | fonctions système sys . . . . .                     | 15        |
| 6.1.3    | fonctions auxiliaires d'accès au matériel . . . . . | 15        |
| 6.1.4    | Gestionnaires de capteurs . . . . .                 | 15        |
| 6.1.5    | nauteff . . . . .                                   | 15        |
| 6.1.6    | Tâches . . . . .                                    | 16        |
| 6.1.7    | Tâche principale . . . . .                          | 16        |
| 6.2      | Canaux de communications . . . . .                  | 17        |
| 6.2.1    | Message gestionnaire principal . . . . .            | 17        |
| 6.2.2    | Commandes envoyées à la tâche de gestion du moteur  | 17        |

# Table des figures

# **Chapitre 1**

## **Buts du document**

### **1.1 But**

Ce document contient les informations de la conception du Nauteff P-1 et vise à en permettre un développement et une maintenance efficace. Il est principalement destiné au développeurs.

### **1.2 Guide de lecture**

La documentation nécessaire à la maîtrise du STM32 est volumineuse et répartie dans plusieurs documents mentionnés dans le chapitre suivant.

## Chapitre 2

# Documents applicables et de référence

### 2.1 Documents applicables

Liste 1 des documents de référence

La documentation des STM32 est répartie dans plusieurs documents. Le principal document intitulé **reference manuel STM32F446RM** contient la description détaillée des mémoires, périphériques, horloges. Le document Datasheet production data **STM32F446xCE** contient, en particulier, les adresses et disponibilité des périphériques. Le document programming manual **STM32M4PM** contient, en particulier la description des registres NVIC. Les documentations de ST Microelectronics comportent peu de renvois, le lecteur ne devra pas hésiter à naviguer dans ces sources. La documentation d'ARM, par exemple **CORTEXM4TRM**, est source complémentaire d'informations.

### 2.2 Documents de référence

# Chapitre 3

## Matériel

### 3.1 STM32F446RE

Nauteff-P2 est réalisé avec une carte Nucleo 446 RE, cette carte comporte un ST-Link V2 et fournit un signal d'horloge à 8 Mhz au microcontrôleur. La plupart des broches de signaux sont disponibles.

#### 3.1.1 Cur

le tableau suivant donne les principales caractéristiques du microcontrôleur :

|              |                  |
|--------------|------------------|
| Cur          | CORTEX M4F       |
| Architecture | armv7e-m         |
| FPU          | Simple précision |
| Fréquence    | jusqu'à 180 Mhz  |

#### 3.1.2 Mémoire

| Mémoire | Adresse     | Taille | Usage |                              |
|---------|-------------|--------|-------|------------------------------|
| FLASH   | 0x0800 0000 | 512K   |       | Permanente                   |
| SRAM1   | 0x2000 0000 | 112K   |       | principale                   |
| SRAM2   | 0x1000 0000 | 16K    |       | supplémentaire, non utilisée |

#### 3.1.3 Horlogerie

Nauteff-P2 utilise l'horloge externe fournie par le ST-Link à 8 Mhz

### 3.1.4 Interruptions

#### Liste

| Interruption          | n° | Usage                                  |
|-----------------------|----|--|
| Exceptions du système |    |  |
| Reset                 |    | appel de la fct main()                 |
| NMI                   |    |  |
| HardFault             |    |  |
| MemManage             |    |  |
| BusFault              |    |  |
| UsageFault            |    |  |
| SVCall                |    | vPortSVCHandler (fct de FreeRTOS)      |
| PendSV                |    | xPortPendSVHandler (fct de FreeRTOS)   |
| SysTick               |    | xPortSysTickHandler (fct de FreeRTOS ) |
| Événements extérieurs |    |  |
| EXTI0                 |    | Bouton Marche                          |
| EXTI1                 |    | Bouton Veille                          |
| EXTI2                 |    | Touche +1                              |
| EXTI3                 |    | Touche -1                              |
| EXTI4                 |    | Touche +10                             |
| EXTI10_15             |    | Touche -10                             |
| I2C1_Event            |    | Évènement I2C1                         |
| I2C1_Error            |    | Erreur I2C1                            |
| USART1_Event          |    | Événement USART 1                      |

#### USART

#### I2C1

Le bus I2C1 sert à la communication avec les capteurs MEMS.

### 3.1.5 Affectation des broches du STM32

Le STM32F303K8 comporte 32 broches.

| STM  | Nucleo   | Fonction          | Périphérique | Usage                          |
|------|----------|-------------------|--------------|--------------------------------|
| PA0  | CN7, 28  | Entrée analogique | ADC1, IN0    | Tension alimentation           |
| PA1  | CN7, 30  | Entrée analogique | ADC2, IN1    | Courant moteur                 |
| PA2  | CN10, 35 | Alt. fct. n° 7    | USART2 TX    | Sortie NMEA 0183               |
| PA3  | CN10, 37 | Alt. fct. n° 7    | USART2 RX    | Entrée NMEA 0183               |
| PA4  | CN7, 32  | GPIO Output       |              | Motor                          |
| PA5  | CN10, 11 | GPIO Output       |              | Clutch and LED                 |
| PA6  | CN10, 13 | GPIO Output       |              | Motor INA                      |
| PA7  | CN10, 15 | GPIO Output       |              | Motor INB                      |
| PA8  | CN10, 23 | Alt. fct. n° 4    | I2C3 SCL     | I2C3 serial clock              |
| PA9  | CN10, 21 | Alt. fct. n° 7    | USART1 TX    | Sortie Série                   |
| PA10 | CN10, 33 | Alt. fct. n° 7    | USART1 RX    | Entrée série                   |
| PA11 | CN10, 12 | Alt. fct. n° 9    | CAN1 RX      | Sortie NMEA2000 / CAN          |
| PA12 | CN10, 10 | Alt. fct. n° 9    | CAN1 TX      | Entrée NMEA2000 / CAN          |
| PA13 | NC       | -                 |              | SYS_JTMS-SWDIO                 |
| PA14 | NC       | -                 |              | SYS_JTCK-SWCLK                 |
| PA15 | Réservé  | Anti-noise        |              | Réservé                        |
| PB0  | CN7, 34  | GPIO Output PU    |              | Buzzer alarme                  |
| PB1  | CN10, 24 | GPIO input        |              | Gyro/Acc data ready            |
| PB2  | CN10, 22 | GPIO input        |              | Mag. data ready                |
| PB3  | NC       | -                 |              | SYS_JTDO-SWO                   |
| PB4  | CN10, 27 | Alt. fct. n° 4    | I2C3 SDA     | I2C3 serial data               |
| PB5  | CN10, 29 | Alt. fct. n° 9    | CAN2 RX      | Entrée NMEA2000/CAN            |
| PB6  | CN10, 17 | Alt. fct. n° 9    | CAN2 TX      | sortie NMEA2000/CAN            |
| PB7  | CN7, 21  | GPIO Output       |              | LSM9DS1 CS_M                   |
| PB8  | CN10, 3  | GPIO Output       |              | LSM9DS1 CS_A/G                 |
| PB9  | CN10, 5  | Anti noise        |              | Réservé                        |
| PB10 | CN10, 25 | Alt. fct. n° 7    | USART3 TX    | bus seatalk TX&RX              |
| PB11 | -        | ??                |              | Relié à une capa               |
| PB12 | CN10, 16 |                   |              | Réservé                        |
| PB13 | CN10, 30 | Alt. fct. n° 5    | SPI2 SCK     | SPI serial clock               |
| PB14 | CN10, 28 | Alt. fct. n° 5    | SPI2 MISO    | LSM9DS1 SDO_M & SDO_A/G        |
| PB15 | CN10, 26 | Alt. fct. n° 5    | SPI2 MOSI    | LSM9DS1 DS1                    |
| PC0  | CN7, 38  | Entrée GPIO, PU   | EXTI0        | Bouton marche                  |
| PC1  | CN7, 36  | Entrée GPIO, PU   | EXTI1        | Bouton veille                  |
| PC2  | CN7, 35  | Entrée GPIO, PU   | EXTI2        | Touche +1                      |
| PC3  | CN7, 37  | Entrée GPIO, PU   | EXTI3        | Touche -1                      |
| PC4  | CN10, 34 | Entrée GPIO, PU   | EXTI4        | Touche +10                     |
| PC5  | CN10, 6  | Entrée GPIO, PU   | EXTI5        | Touche -10                     |
| PC6  | CN10, 4  | Entrée GPIO, PU   | EXTI?        | Touche auxiliaire 1            |
| PC7  | CN10, 4  | Entrée GPIO, PU   | EXTI?        | Touche auxiliaire 2            |
| PC8  | CN10, 2  | Inutilisée        |              | Libre                          |
| PC9  | CN10, 1  | Sortie GPIO       | MCO          | Sortie horloge (mise au point) |
| PC10 | CN7, 1   | Alt. fct. n° 8    | UART4 TX     | Sortie série                   |
| PC11 | CN7, 2   | Alt. fct. n° 8    | UART4 RX     | Entrée série                   |
| PC12 | CN7, 3   | Alt. fct. n° 89   | USART5 TX    | Sortie série                   |
| PC13 | CN7, 23  |                   | Bouton bleu  | fonction spéciale              |
| PC14 |          |                   |              | Quartz 32,768 khz              |
| PC15 |          |                   |              | Quartz 32,768 khz              |
| PD2  | CN7, 4   | Alt. fct. n° 8    | USART5 RX    | Sortie Série                   |

La broche P9 sert à observer l'horloge système à l'oscilloscope. Elle pourra

servir plus tard à une autre fonction.

### 3.1.6 Organisation de la mémoire STM32

| Mémoire | Adresse     | taille |                        |
|---------|-------------|--------|------------------------|
| FLASH   | 0x0800 0000 | 128K   | Permanente, plus lente |
| RAM1    | 0x2000 0000 | 112K   |                        |
| RAM2    | 0x2000 0000 | 16K    | Core coupled memory    |

## 3.2 Capteurs MEMS

Les capteurs MEMS sont montés sur une carte d'essai de Pololu AltIMU-10 v5. Cette carte contient les circuits de ST Microelectronics suivants :

- un compas LIS3MDL ;
- un gyromètre et accéléromètre LSMDS33 ;
- un baromètre LPS25H, non utilisé.

## 3.3 Commande de l'actionneur

La commande de l'actionneur est réalisée par un circuit VNH 5019 de ST Microelectronics monté sur une plaque d'essai de Pololu. Ce circuit comprend un pont en H pouvant délivrer un courant de 12A en permanence avec des protections contre les surcharges et court-circuits, des signaux de diagnostic et une tension proportionnelle au courant. Ce circuit a une partie logique et contrôle alimentée avec la carte Nucleo et une partie de puissance alimentée par une source de 12V.

Les signaux suivants sont utilisés :

- INA ,sens pont A) relié à B4 ;
- INB (sens pont B) relié à B5 ;
- CS relié PA7 (ADC2\_IN4) ;
- PWM relié à PB0.

L'actuel STM32 F303K8 n'a pas suffisamment de broches pour y raccorder ENA/DIAG et ENB/DIAG.

## **Chapitre 4**

# **Architecture générale de l'application**

### **4.1 Architecture statique**

4.1.1 Système

4.1.2 Pilotes de périphériques

4.1.3 Tâches

4.1.4 Calculs

4.1.5 Traitements des trames NMEA0183

4.1.6 IMU

0

### **4.2 Relations dynamiques**

### **4.3 Justification des choix d'architecture**

# Chapitre 5

## Contexte de la conception

### 5.1 présentation de l'application

Nauteff est un pilote automatique pour navires.

### 5.2 Principales exigences applicables

#### 5.2.1 Exigences liées à la sûreté de fonctionnement

Le navigateur, confiant la timonerie de son navire à Nauteff, attend de ce dernier un fonctionnement fiable et sûr. Il doit être conçu pour éviter toute panne ou comportement mettant en danger le navire et son équipage.

Les principes suivants sont adoptés :

- commentaires précis et détaillés dans le code ;
- Utilisation réduite de librairies extérieures et programmation "bare metal" ;
- Absence d'allocation dynamique après la phase d'initialisation ;
- Usage réduit de variables locales dans les fonctions et utilisation de variables statiques ;
- application des règles MISRA-C:2004 **MISRAC** ;
- Adoption du principe KISS ("Keep It Simple Stupid") ;
- Adoption de programmation défensive ;
- Usage du langage C sans.

L'absence d'allocation dynamique permet d'éviter un arrêt brutal par manque de mémoire disponible.

#### 5.2.2 Architecture matérielle opérationnelle

Nauteff utilise une carte Nucleo dotée d'un STM32 F446RE et d'une sonde ST-Link accessible par prise USB. Le STM32 F446RE comporte que 32 broches, le prochain Nauteff

### 5.2.3 Logiciels imposés

Néant.

### 5.2.4 Interface avec d'autres applications

Interfaces NMEA 0183 Cette interface est aussi utilisée pour la sortie de messages pendant le développement.

## 5.3 Contraintes de développement

### 5.3.1 Méthode et formalisme

L'organisation du développement est principalement celle de cycle en V. La programmation est essentiellement fonctionnelle et les concepts objets ne sont pas utilisés.

### 5.3.2 Langage de programmation

La programmation est réalisée entièrement avec le langage C11. Le langage C permet d'accéder très facilement à la mémoire et à tous les registres. Le recours à l'assembleur n'est pas envisagé.

### 5.3.3 Logiciels extérieurs

Nauteff-P2 utilise FreeRTOS V 10.0.1 qui fournit le noyau temps réel, les horloges, les sémaphores et les canaux de communication Nauteff P-2 utilise la librairie standard libc et la librairie mathématiques lm. Nauteff P-2 utilise les fonctions les plus simples de la libc, il utilise sprintf uniquement dans les versions expérimentales.

### 5.3.4 Outils

|                |                   |                                    |
|----------------|-------------------|------------------------------------|
| EDI, au bureau | Eclipse C/C++     | eclipse.org                        |
| Compilation    | arm-none-eabi-gcc | chaîne du GNU, compilation croisée |
| sonde          | ST-Link           | intégrée sur carte Carte Nucleo    |
| Accès sonde    | openocd 0.10.0    |                                    |
| Documentation  | Latex             | avec TeXstudio                     |

### 5.3.5 Environnement matériel de développement

Le développement au bureau est réalisé avec les matériels suivant :

- un PC de bureau permettant de faire fonctionner eclipse, la chaîne de compilation... ;
- un oscilloscope à deux voies minimun ;
- occasionnellement un voltmètre ;

— petit outillage : pinces, tournevis, dénudeuse, . . . ;

Lors des essais en mer du prototype un petit PC consommant peu mais ne permettant pas d'utiliser eclipse permet de faire des modifications du logiciel et, en particulier, des ajustements de paramètres. Il embarque la chaîne de compilation et openocd. L'édition est faite avec vi et la compilation avec la commande make.

# Chapitre 6

# Architecture du logiciel

## 6.1 architecture statique

6.1.1 Noyau temps réel : FreeRTOS

6.1.2 fonctions système sys

I2C

UART

SPI

CAN Control Area Network, pour NMEA 2000

Initialisations

6.1.3 fonctions auxiliaires d'accès au matériel

Calculs et géométrie

6.1.4 Gestionnaires de capteurs

6.1.5 nauteff

moteur

Cette tâche reçoit les ordres de la tâche de gestion, elle commande l'embrayage du moteur et son fonctionnement : sens et durée. Elle corrige la durée de fonctionnement en fonction de l'effort du moteur. Lorsque le moteur tourne, elle mesure le courant du moteur, vérifie que le moteur n'est pas bloqué ou débranché et renvoie une estimation de l'effort du moteur. Elle informe la tâche principale lorsque le moteur a fini un mouvement demandé. Les entrées sont les suivantes :

- Ordres de commande du moteur et de l'embrayage en provenance de la tâche principale ;
- valeurs du courant et de la tension en provenance de l'ADC ;

- horloge (à définir).

Les sorties sont les suivantes :

- ordres vers le moteur mise en marche et sens ;
- ordres vers l'embrayage : enclenchement et désenclenchement ;
- informations vers la tâche principale sur le fonctionnement du moteur estimation de l'effort, de la position, blocage ou court-circuit.

### **mems : Gestion des capteurs MEMS**

Cette tâche lit les informations des capteurs MEMs, les filtre, calcule le cap et le lacet et envoie les informations traitées à la tâche principale. Les documents de ré

Le signal d'interruption n'étant pas disponible sur la carte de Pololu la tâche fonctionne par scrutation.

La tâche envoie les données vers la tâche principale.

### **keyboard**

La tâche utilise la fonction nommée TaskKeyboard qui n'utilise aucun argument. Elle fonctionne par scrutation périodique des entrées sur lesquelles sont branchées les touches. La scrutation périodique est utilisée car les touches provoquaient de très nombreux rebonds parasites. Elle envoie les ordres à la tâche de gestion de l'état par l'intermédiaire d'un `MessageBuffer_t` nommé `buffclav`.

### **NMEA 0183/2000**

### **Étalonnage**

### **Seatalk**

### **AutoPilot**

#### **6.1.6 Tâches**

| Tâche                     | Priorité | Nom          |
|---------------------------|----------|--------------|
| Gestion du clavier        | 6        | TaskKeyboard |
| Contrôle du moteur        | 5        | TaskMotor    |
| Gestion de l'état         | 3        | TaskCore     |
| Gestion des capteurs MEMS | 3        | TaskMEMS     |

#### **6.1.7 Tâche principale**

### **Entrées**

- Commandes en provenance du clavier ;
- Informations du moteur ;
- Informations des capteurs MEMS ;

- Informations en provenance des interfaces NMEA
- horloge (à définir).

### Sorties

- Commandes moteurs ;
- alarmes ;

## 6.2 Canaux de communications

| Canal                                   | Nom         | Type            | Taille |
|---|-------------|-----------------|--------|
| Messages vers le gestionnaire principal | bufferCore  | MessageBuffer_t | 10     |
| Ordres vers le moteur                   | bufferMotor | MessageBuffer_t | 5      |

### 6.2.1 Message gestionnaire principal

Ces messages sont envoyés par les tâches TaskKeyboard, TaskMotor et TaskMEMs. Ils sont stockés dans une structure dont les deux premiers octets comportent un code indiquant la nature du message et, en option, d'autres informations.

| Ordre  | Code et valeurs optionnelles              |
|--|---|
| Ordres en provenance de la tâche du clavier vers la tâche principale |   |
| Mise en mode automatique   | MSG_KBD_AUTO                              |
| Mise en veille   | MSG_KBD_STDBY                             |
| +1   | MSG_KBD_STARBOARD_ONE                     |
| -1   | MSG_KBD_PORT_ONE                          |
| +10  | MSG_KBD_STARBOARD_TEN                     |
| -10  | MSG_KBD_PORT_TEN                          |
| +1 relachée  | MSG_KBD_STARBOARD_ONE_END                 |
| -1 relachée  | MSG_KBD_PORT_ONE_END                      |
| Informations en provenance de la tâche de gestion du moteur          |   |
| Position estimée du moteur   | MSG_MOT_POSITION_ESTIMEE                  |
| Effort moteur  | MSG_MOT EFFORT + float effort             |
| Moteur bloqué ou en butée  | MSG_MOT_BLOCKED                           |
| Court-circuit sortie moteur  | MSG_MOT_SHORT_CIRCUIT                     |
| Fin de déplacement actuateur   | MSG_MOT_END_MOVE + float : effort         |
| Informations des capteurs MEMs                                       |   |
| Info Capteurs MEMs   | MSG_MEMS_VALUES + 2 floats : cap et lacet |
| Info MEMs indisponible   | MSG_MEMS_FAILURE                          |

### 6.2.2 Commandes envoyées à la tâche de gestion du moteur

Ces messages sont envoyés par la tâche principale ou un timer à la tâche de gestion du moteur.

| Ordre                           | Code et valeurs optionnelles                 |
|---------------------------------|--|
| Commande de la tâche principale |  |
| Enclenchement de l'embrayage    | MSG_MOT_EMBRAYE                              |
| Désenclenchement de l'embrayage | MSG_MOT_DEBRAYE                              |
| Mise en marche vers tribord     | MSG_MOT_STARBOARD                            |
| Mise en marche vers babord      | MSG_MOT_PORT                                 |
| Mouvement vers tribord          | MSG_MOT_MOVE_STARBOARD + float : valeur      |
| Mouvement vers babord           | MSG_MOT_MOVE_PORT + float : valeur           |
| Arrêt du moteur                 | MSG_MOT_STOP                                 |
| Demande de mesure d'effort      | MSG_MOT_DMD EFFORT                           |
| commande du timer               |  |
| Commande du TIMER               | MSG_MOT_MOVING_CONTROL<br><br>(key)<br>(key) |

La tâche de gestion du moteur effectue les déplacements dont l'amplitude doit être proportionnelle à la valeur en paramètre, elle adapte la durée au courant du moteur.