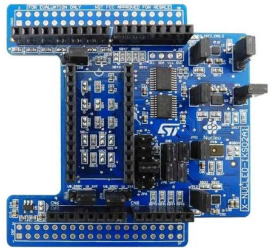
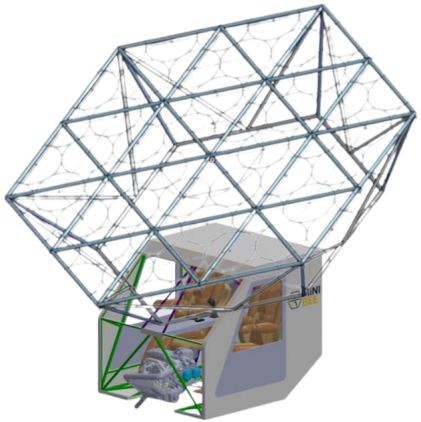
Projet Mini-Bee

Prototypage du Flight Control Unit à partir des composants STM

life.augmented

Soutenance du 25 Avril 2023





OLIVIER Thomas, PROQUOT Mattéo, GRAYON Damien
Ecole d'ingénieur informatique 3ème année option informatique embarquée
année 2022-2023
Référent : DUTERTRE Xavier

Mini-Bee

Avion hybride à atterrissage vertical et à décollage (VTOL) à des fins médicales

Peut être transporté dans la soute d'un avion civil

Peut facilement être assemblé sur un tarmac

Projet collaboratif avec des écoles et des entreprises

Licence Lesser Open Source

Ecoles partenaires

ESIGELEC
ÉCOLE D'INGÉNIEURS-ES GÉNÉRALISTES

Tests moteurs BLDC

CentraleSupélec

Modélisation de la chaîne de production

centralelille
ÉCOLE CENTRALE DE LILLE

Chaîne de puissance

CY TECH

Prototype du FCU et des capteurs

ESTACA
ECOLE D'INGENIEURS

Structure et chaîne de propulsion

LYCÉE LOUIS ARMAND - NOGENT SUR MARNE

Structure du Mini-Bee

isae supméca

Revue de conception clôture et TRL3

Sommaire

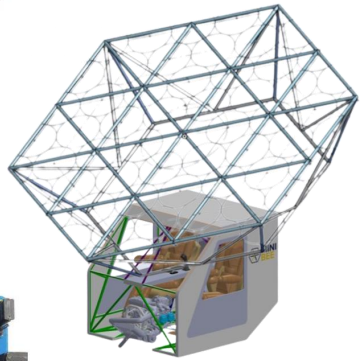
- Rappel des objectifs
- WBS
- Planning du projet
- Définition du FCU
- Prototype
- Package pédagogique
- Conclusion et orientations

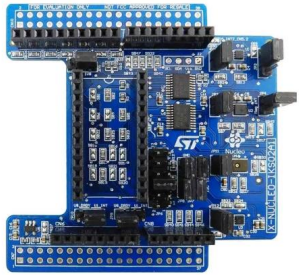
Rappel des objectifs

1. **Définition**
- Définir le FCU du Mini-Bee
- Définir les composants de contrôle de la chaîne de puissance

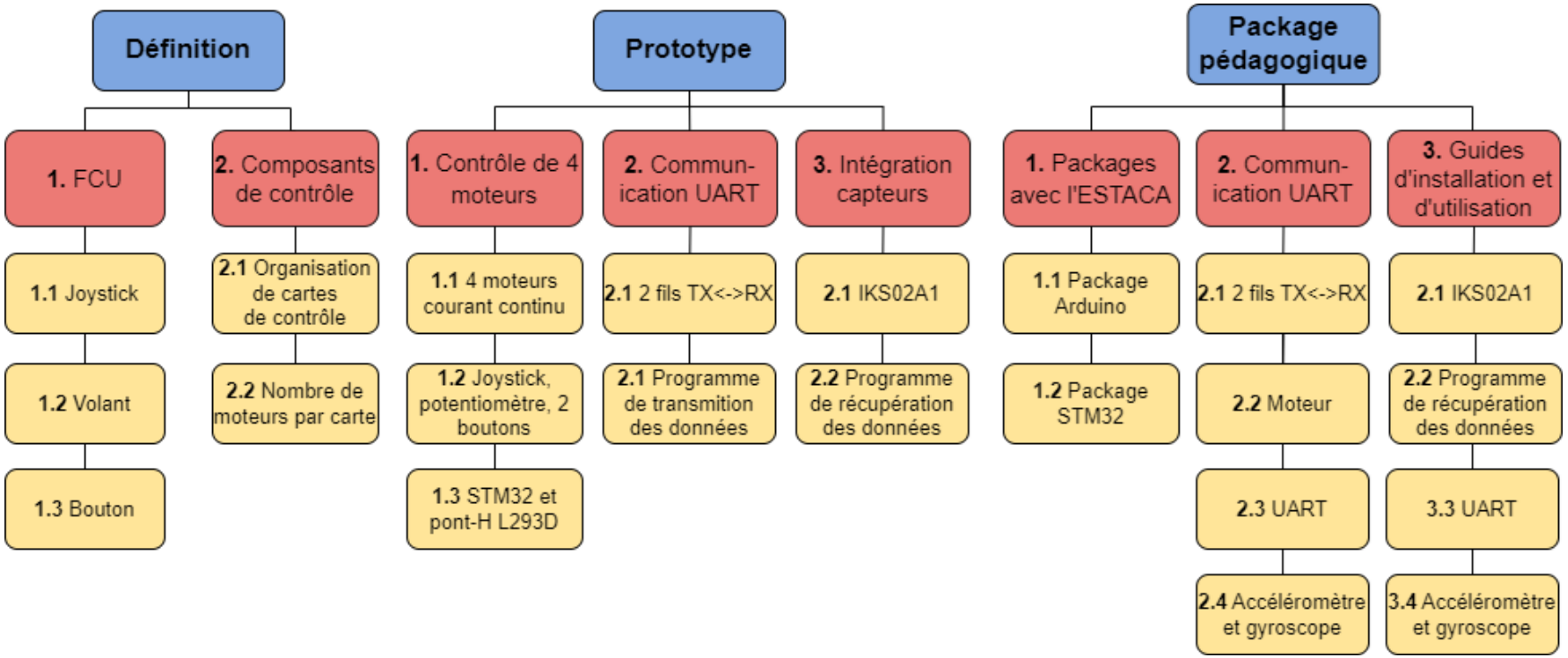
1. **Prototype**
- Contrôler 4 moteurs avec la carte STM32
- Faire passer des informations entre deux cartes STM32 avec UART
- Intégrer la carte NUCLEO-iks contenant les capteurs

1. **Package pédagogique**
- Participer à la création du package pédagogique avec l'Estaca
- Créer des librairies/codes d'exemples
- Rédiger des guides d'installation/utilisation





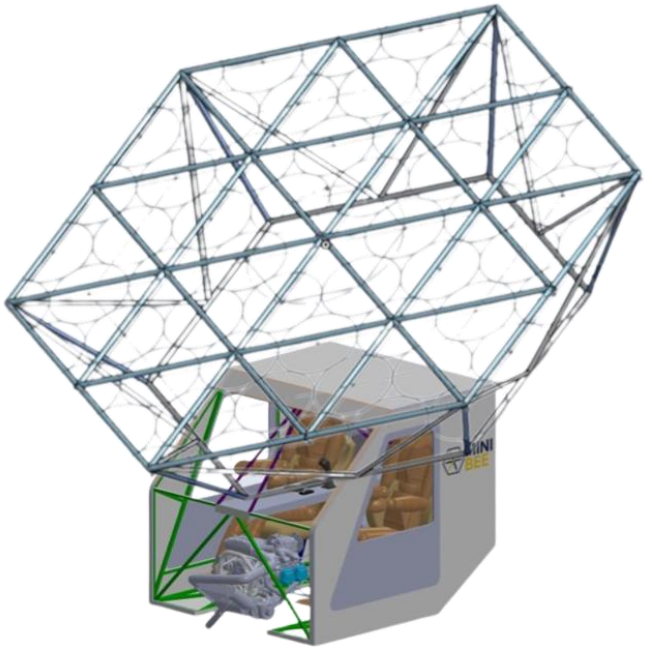
Découpage des tâches (WBS)



Planning du projet

	03/01/2023	16/01/2023	30/01/2023	13/02/2023	06/03/2023	20/03/2023	03/04/2023	24/04/2023
Composante de contrôle								
Contrôle de 4 moteurs								
Communication UART								
Intégration capteurs								
Package avec l'ESTACA								
Librairie et code d'exemple								
Guide d'installation et d'utilisation								

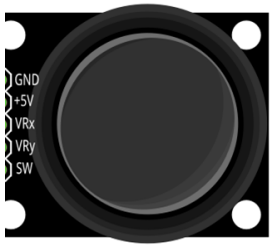
1.1 Définition du FCU : Contrôles du Mini-Bee



Définition du FCU : Prototype des commandes

- Joystick
- Contrôle altitude
- Potentiomètre : simulation d'un volant

Joystick



Tangage avant

Tangage arrière

Roulis Gauche

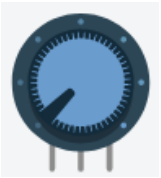
Roulis Droite

Altitude

Up

Down

Volant

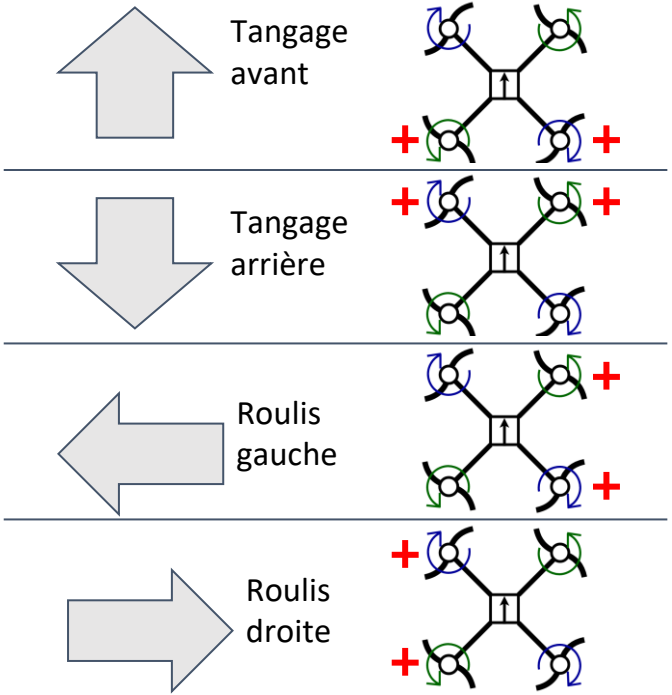


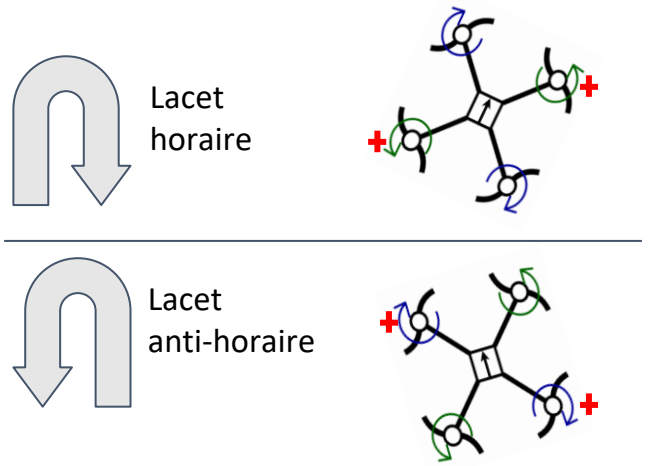
Potentiomètre

Lacet horaire

Lacet anti-horaire

Définition du FCU : Prototype des commandes





Conclusion / Orientation

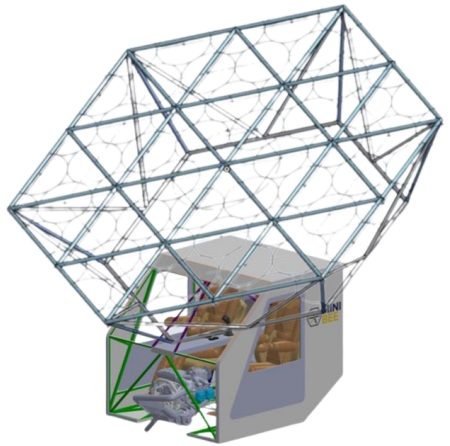
- Les commandes sont, pour l'instant, plus proches de celles d'un drone que d'un avion/hélicoptère

Définition du FCU : contrôle à l'aide de plusieurs cartes STM32

Objectifs
Au maximum, 4 moteurs indépendants par carte STM32

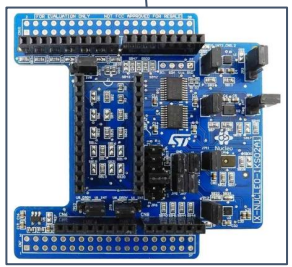
Problème : Comment contrôler les 60 rotors ?

Conclusion / orientation
Solution : Utiliser plusieurs cartes STM32 et communiquer les informations entre elles à l'aide de l'UART (Protocole).



1.2 Définition des composants (Version annulée)

Carte calculateur



UART

Cartes commandes

4 sorties

12 rotors

4 sorties → 12 rotors

2 sorties → 6 rotors

Conclusion / Orientation

- Au total, 6 cartes commandes sont nécessaires pour contrôler les 2x30 rotors
- Les données envoyée aux cartes commandes par la carte calculateur sont à définir

Définition des composants (Version définitive)

Carte calculateur

5 sorties

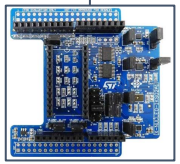
30 rotors

UART

5 sorties

30 rotors

Carte calculateur



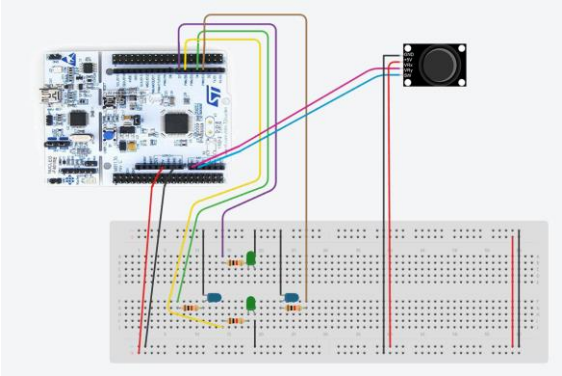
Conclusion / Orientation

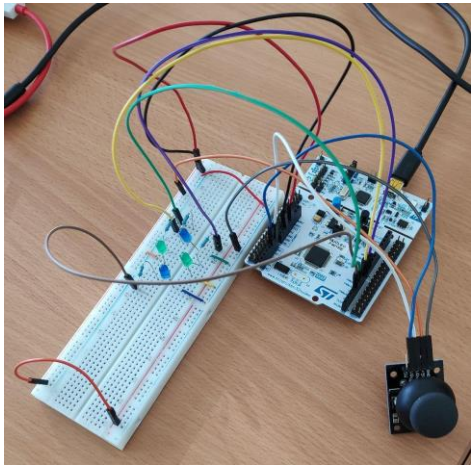
- 2 STM32 avec 5 sorties est nécessaire pour contrôler les 2 * 5 * 6 rotors
- Les données communiquées entre les deux cartes calculateur sont à définir

2.1 Prototypage du MiniBee : Joystick + LEDS

Objectifs

- Récupérer la position du Joystick avec une carte STM32F401RE
- Allumer la LED correspondant à l'orientation du Joystick





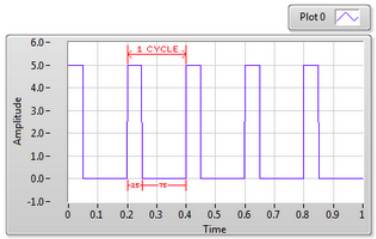
Conclusion / Orientation

La carte récupère les valeurs en abscisse et ordonnée du Joystick allant de 0 à 4095

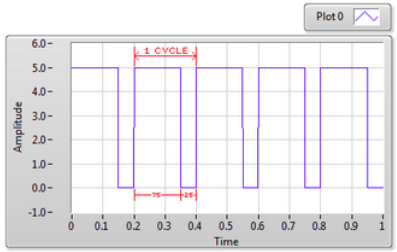
Prototypage du MiniBee : contrôler un moteur

Utilisation d'un moteur brushed à notre disposition (à balais, courant continu)

-> Pas de contrôle de vitesse : ON si alimenté, OFF sinon



Duty cycle of 25%



Duty cycle of 75%

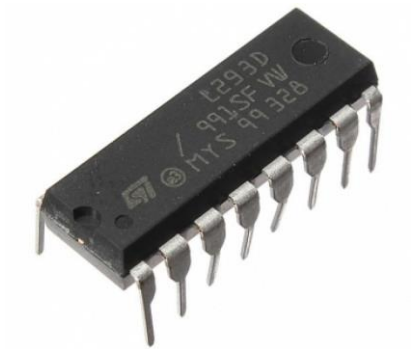
Solution : Utiliser un signal PWM (Pulse Width Modulation) pour faire varier la vitesse du moteur

Prototypage du MiniBee : contrôler un moteur

1ère étape : alimentation du moteur via la carte STM32F401RE

2ème étape : contrôle de la vitesse du moteur

- Utilisation d'un pont-H L293D (puce qui contient 4 transistors)
    - Permet le contrôle de la vitesse et du sens de 2 moteurs



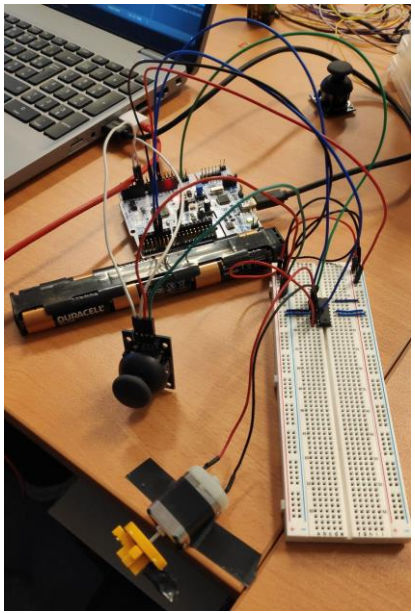
Prototypage du MiniBee : Joystick + moteur

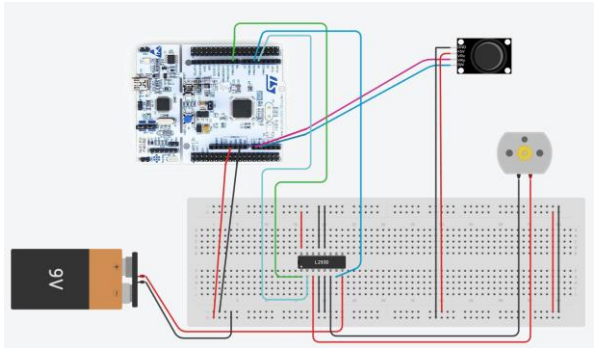
Objectif
- Ajout d'un joystick au montage précédent pour gérer la vitesse du moteur

Résultats
- Montage fonctionnel
- Permet d'ajouter facilement + de moteurs et + de joysticks

Mais
- Manque de précision dû au joystick qui diffère d'un joystick de cockpit

Objectifs suivants
- Ajouter d'autres éléments : moteurs, volant, boutons, capteurs

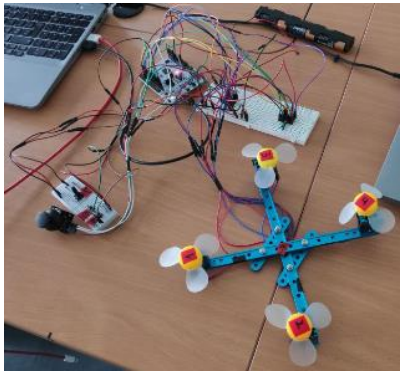




Prototypage du MiniBee : Contrôle 4 moteurs

Objectifs

- Tester la carte STM32
- Ajouter 3 moteurs afin de simuler un drone (Fait)
- Ajouter un “volant”, “accélérateur” et “frein” (Fait)



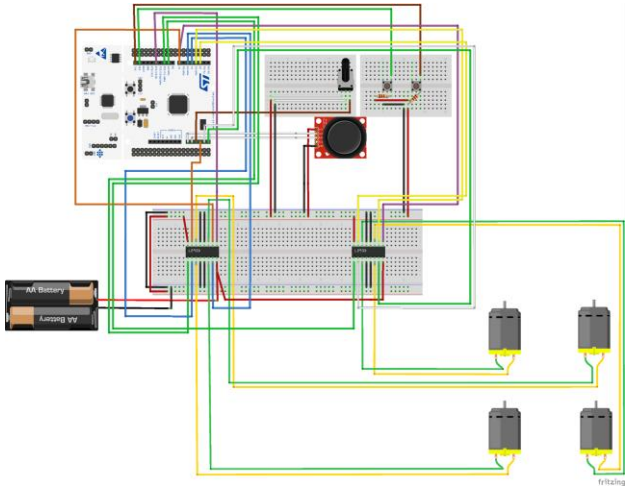
Données techniques

- Joystick, potentiomètre, 2 buttons
- Carte STM32F401RE
- 2 contrôleurs moteurs courant continu L293D
- 4 moteurs

Conclusion / Orientation

Une carte STM32F401RE contrôle 4 moteurs indépendamment

Prototypage du MiniBee : Contrôle 4 moteurs



Objectifs

- Faire un package pédagogique simple à comprendre
- Clarifier le montage électrique du prototype de drone

Données techniques

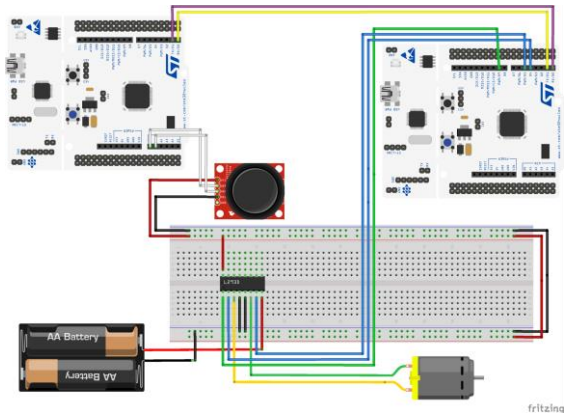
- Utilisation du logiciel Fritzing pour clarifier les schémas et être capable de les reproduire

Conclusion / Orientation

- Envoyer les schémas à l'Estaca pour qu'ils les intègrent à leur wiki

2.2 Prototypage du MiniBee : contrôle à l'aide de plusieurs cartes STM32

carte calculateur **carte contrôle**



Objectifs

- Faire un package pédagogique simple à comprendre
- Clarifier le montage électrique de la communication entre deux cartes

Données techniques

- Utilisation du logiciel Fritzing pour clarifier les schémas et être capable de les reproduire

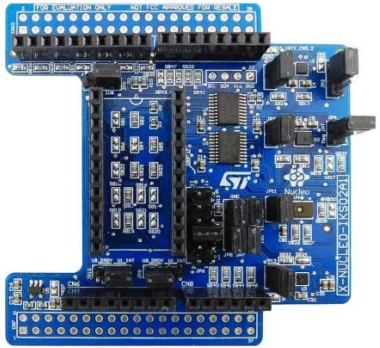
Conclusion / Orientation

- Envoyer les schémas à l'Estaca pour qu'ils les intègrent à leur wiki

2.3 Prototypage du MiniBee : Capteurs

Objectifs

- Prise en main de la carte X-NUCLEO-IKS02A1
- Récupération des informations des capteurs : accéléromètre, gyroscope, magnétomètres



Données techniques

- ISM330DHCX MEMS 3D accelerometer (±2/±4/±8/±16 g) plus 3D gyroscope (±125/±250/±500/±1000/±2000 dps)
- IIS2DLPC MEMS 3D accelerometer low power (±2/±4/±8/±16 g)

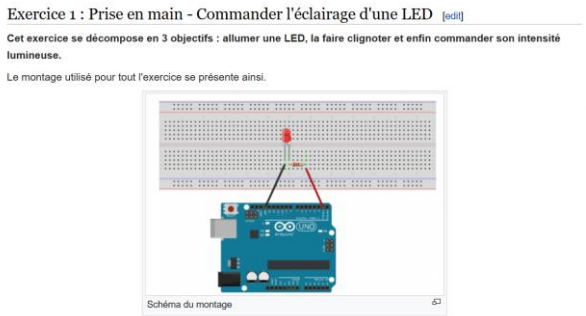
Conclusion / Orientation

- Données des capteurs affichées dans le terminal

3.1 Package pédagogique :Collaboration avec L'ESTACA

Objectifs

- Revue de code fait par l'ESTACA
- Test sur nos propres machines




```
void setup()
{
 pinMode(1, OUTPUT);    //initialise la borne numérique numéro 1 de la carte Arduino en mode sortie
}
void loop()
{
 digitalWrite(1, HIGH);         //le courant est envoyé sur la borne 1, la LED s'allume
}
```

Données techniques

- Utilisation de l'IDE ARDUINO pour toutes les cartes même celle STM

Conclusion / Orientation

- Les consignes du wiki sont claires et permettent une utilisation facile des composants ARDUINO

Package pédagogique : Bibliothèque

Objectifs

- Création d'un package pédagogique (stand-by)
- Création d'une notice (stand-by)

Conclusion

- La bibliothèque recensera toutes les fonctions dont le package pédagogique aura besoin pour débuter

```c
//Welcome to the library for new developers

void TurnOnLED(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin){
    GPIOx->BSRR = GPIO_Pin;
}

void TurnOffLED(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin){
    GPIOx->BSRR = (uint32_t)GPIO_Pin << 16U;
}

int IsButtonPressed(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin){
      /* Check the parameters */
      assert_param(IS_GPIO_PIN(GPIO_Pin));

      if((GPIOx->IDR & GPIO_Pin) != 0){
        return GPIO_PIN_SET;
      }
      else{
        return GPIO_PIN_RESET;
      }
}
```

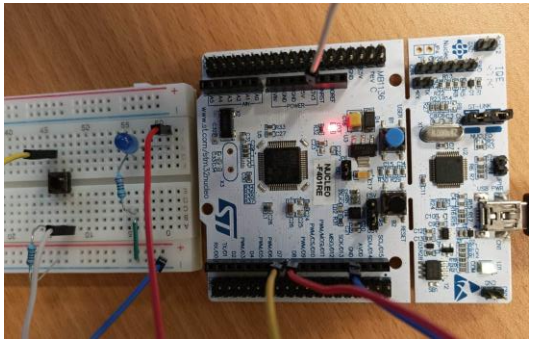
Mise en pratique pour un bouton et une LED

Objectifs

- simplification des fonctions pour le bouton (fait)
- simplification des fonctions pour la LED (fait)

Conclusion

- Les fonctions créées permettront une meilleure compréhension du code avec un langage familier



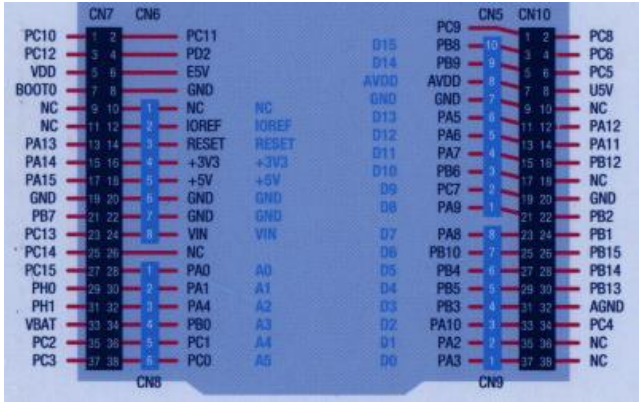
```
while (1)
{
    //if pinSet = 0 -> button release, if pinSet = 1, button press
    pinSet = IsButtonPressed(GPIOA, GPIO_PIN_8);
    if(pinSet)
    {
        TurnOnLED(GPIOA, GPIO_PIN_9);
     }
     else
     {
         TurnOffLED(GPIOA, GPIO_PIN_9);
     }
  /* USER CODE END WHILE */

  /* USER CODE BEGIN 3 */
}
```

Documentation en début de package

Objectifs

- Documentation pour lire les différentes pins (stand-by)



Données techniques

- correspondance de la notation de la datasheet et des notations des fonctions

Exemple

datasheet : PC8 -> fonction :GPIOC , GPIO_PIN_8

Conclusion

- L'utilisateur pourra comprendre rapidement comment débuter les branchement et le code

3.3 Guide d'installation et d'utilisation

Objectifs

- prise en main facile du projet par des futurs groupes

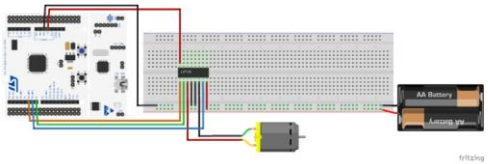
DC_MOTOR_Set_Speed() sert à choisir la vitesse d'un moteur (ici 1000).

```
  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
      DC_MOTOR_Set_Speed(DC_MOTOR1, 1000);
  }
```

DC_MOTOR_Stop() sert à arrêter les moteurs.

Partie Hardware :



Données techniques

- mise en place d'un fichier README pour l'exploitation de nos ressources
- le code à été commenté pour permettre une lecture claire

Conclusion

- Tout nouvel utilisateur de l'IDE pourra simplement récupérer les différents codes et les exploiter.

Continuité du package pédagogique

- Discuter avec ESTACA pour continuité du package pédagogique
- Rédaction d'une introduction pour la compréhension des pins et GPIO
- Photo des montages plus simples ou plus durs (LED seule, plusieurs LED, plusieurs boutons)
- Intégration de la bibliothèque au projet joystick

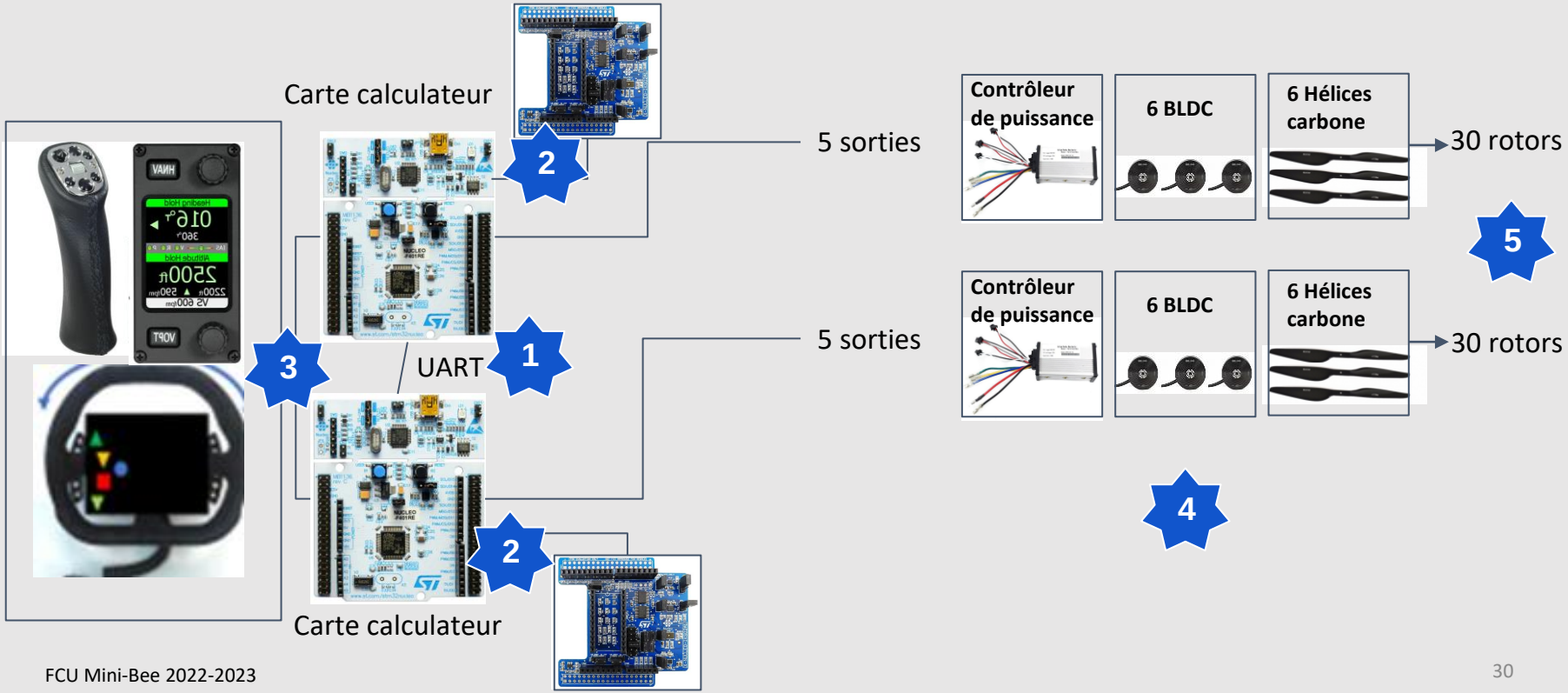
Conclusion nos apports

- Validation de l'architecture des composants
- Création et tests des packages pédagogiques avec l'Estaca
- Création du prototype avec le contrôle de 4 moteurs sur 1 carte STM32
- Dialogue entre 2 cartes STM32
- Début de la prise en main de la carte Nucléo reçue de STM

Orientations futures sur le FCU du Mini-BEE

1. Valider l’architecture détaillée du code entre les cartes
2. Réaliser l’intégration des capteurs
3. Définir les lois de commandes
4. Tester l’architecture multirotors à échelle 1
5. Valider la capacité de contrôle réelle à partir des vitesses de rotation des rotors

Définition des composants (Version définitive)



Questions ?