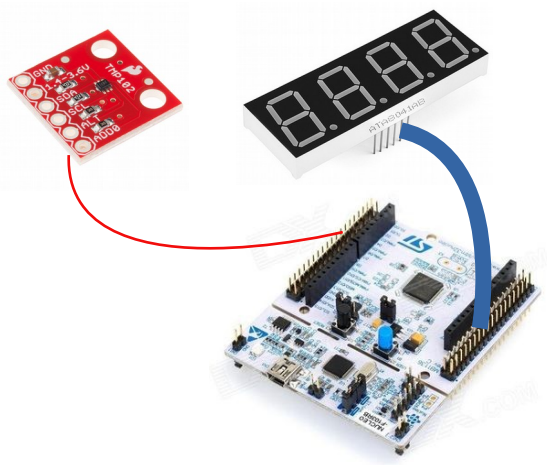
**STM32 mini projet : horloge/thermomètre**

On se propose de réaliser une horloge heures/minutes avec thermomètre ambiant, un bouton poussoir permet de choisir l'affichage (heure ou temperature).
Vous utiliserez la carte NUCLEO mise à votre disposition. Les documentations utiles seront recherchées sur Internet.



Le choix des périphériques est libre (GPIO, I2C UART)
L'étiquette d'afficheurs 7 segments ATA8041AB sera connectée sur les GPIO de la carte Nucleo.
Le capteur de température TMP102 communique sur bus I2C
Le bouton poussoir bleu de la carte Nucleo permettra d'afficher l'heure et les minutes par défaut et la température au 1/100 degré lorsqu'il sera enfoncé.
La mise à l'heure et la mise au point (debug) se feront pat l'UART sur USB.

## 1 Réalisation du matériel

Compléter le tableau d'interconnexion des éléments du projet faisant apparaître les numéros des broches des composants TMP102 et ATA8041AB ainsi que les numéros de broches des connecteursde la carte NUCLEO. (le bouton poussoir sera le bouton bleu de la carte NUCLEO)

| **Fonction Aff 7seg** | Broche Aff 7seg | Broche Nucleo (Arduino) | **Fonction Capteur t°** | Broche Capteur t° | Broche Nucleo (Arduino) |
|---|---|---|---|---|---|
| **a** | | | **VDD** | | |
| **b** | | | **GND** | | |
| **c** | | | **SCL** | | |
| **d** | | | **SDA** | | |
| **e** | | | **ALT** | | |
| **f** | | | **ADD0** | | |
| **g** | | | | | |
| **dp** | | | | | |
| **Aff1** | | | | | |
| **Aff2** | | | | | |
| **Aff3** | | | | | |
| **Aff4** | | | | | |

1. Réaliser ensuite le câblage du matériel sur platine à insertion.

## 2 Réalisation de l'horloge temps réel

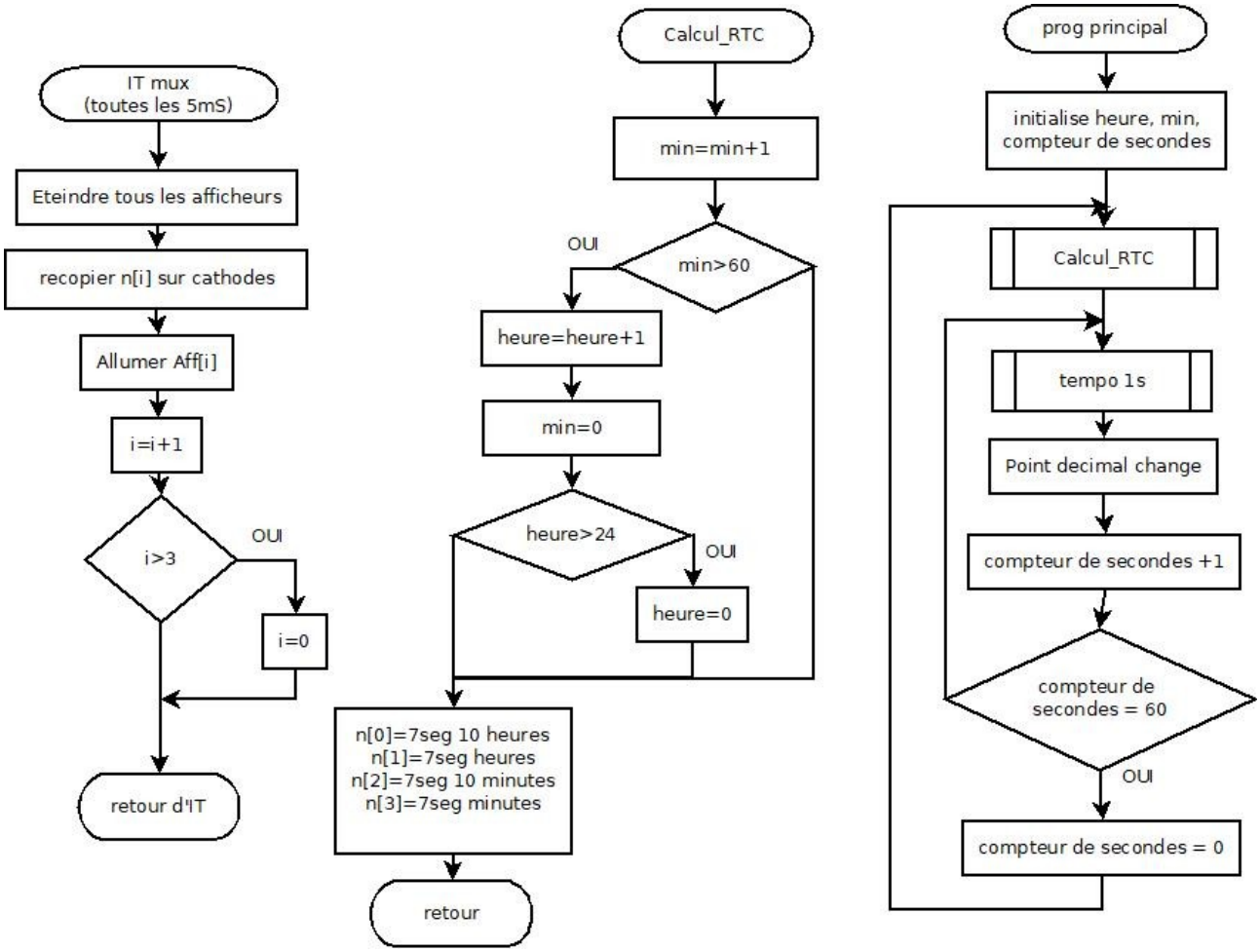
On propose de réaliser la partie horloge du logiciel à partir des algorigrammes ci dessous :

Le programme principal initialise les variables et lance une boucle sans fin dans laquelle l'heure est calculée toutes les secondes, ( le *point décimal entres les étiquettes heures et minutes clignotera avec les secondes* ).

La fonction Calcul_RTC , est appelée toutes les minutes et met à jour les variables heure et min puis met à jour le tableau à afficher (aff) qui contient quatre caractères.
Ex : pour 10h45m aff[ ]=

| *indice* | *0* | *1* | *2* | *3* |
|---|---|---|---|---|
| **valeur** | **7seg de 1** | **7seg de 0** | **7seg de 4** | **7seg de 5** |

La fonction IT_mux est appelée automatiquement par un objet TICKER toutes les 5mS.
Elle met à jour les afficheurs et assure le multiplexage.




2. Dans un premier temps réaliser un programme de comptage des secondes qui mettra à jour le tableau aff[ ] . La fonction IT_mux assurera le multiplexage (la fonction Calc_RTC n'est donc pas à réaliser pour l'instant)

3. Réaliser l'horloge temps réel demandée.

### 4. Ajout de le mesure de température

Le programme précédent nécessitera peut de modifications. MBED propose dans son cookbook une classe de gestion du capteur TMP102
A partir de l'exemple d'application proposé sur MBED ajouter la gestion du bouton et la mise à jour de l'affichage par la température ou l'horloge.

### 5. Ajout de le mesure de température

Modifier le programme de manière à proposer de mettre l'horloge à l'heure lors la mise sous tension manuellement à l'aide d'un terminal sur la liaison UART sur USB.

## Annexe : squelette du programme horloge

```
#include <mbed.h>

unsigned char heure, minute,cptsec;
unsigned char 7seg[]={0,1,2,3,4,5,6,7,8,9};  // a modifier/completer
unsigned char aff[4];  // tableau des codes 7 seg a afficher (n[0] represente les dizaines d'heures)

// exemple a adapter
BusOut cathport(PA_10,PB_3,PB_5,PB_4,PB_10,PA_8,PA_9,PC_7); // cathodes (aff cathodes communes)
BusOut anoport(PA_10,PB_3,PB_5,PB_4);                       // anodes (aff anodes communes)

void Calcul_RTC()
{
    minute++;
    if (minute>60)
    {
        // a completer
    }
    if (heure>24)
    {
        // a completer
    }
    aff[0]=7seg[heure/10];
    aff[1]=7seg[(heure-(heure/10)*10)];
       // a completer
}

void IT_mux(void)
{
static int i=1; // i est l'afficheur allume
static int j=0; // j est l'indice du tableau n a afficher
    anoport.write(0); // extinction de tous les afficheurs
    i<<=1;      // afficheur suivant
    if (i>8) i=1; // 4 afficheurs ( 1,2,4,8)
    j++;
    if (j>3) j=0;
    cathport.write=aff[j]; // code 7seg a afficher
    anoport.write(i);    // allumage afficheur
}

void change_point_decimal(void)
{
    // a completer
    // changer le bit correspondnat au point decimal
    // utiliser un masque en ou-exclusif ( aff[x]=aff[x]^0x80 ) fait basculer le bit 7 de aff[x]
}

int main()
{
    heure=0;
    minute=0;
    cptsec=0;
    // a faire
    // mise en place du ticker 5mS qui appelle IT_mux()
    while(1)
    {
        Calcul_RTC();
        wait(1);
        change_point_decimal();
        cptsec++;
        if (cptsec>60)  cptsec=0;
    }
}
```