```
//Déclaration variable pour programmation FONCTION
int autorise_touche_fonction;
int compteur_touche_fonction;
long toucheAfonction;//Utilisée pour enregistrer la première touche appuyée
byte toucheBfonction;//Utilisée pour enregistrer la deuxième touche appuyée
byte toucheCfonction;//Utilisée pour enregistrer la troisième touche appuyée
byte toucheDfonction;//Utilisée pour enregistrer la quatrième touche appuyée
byte toucheEfonction;//Utilisée pour enregistrer la cinquième touche appuyée

int nbre_fonction;
int traitement_appui_touches_fonction;
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
//Déclaration variable pour programmation CV
int compteur_touche_cv;
byte toucheAcv;//Utilisée pour enregistrer la première touche appuyée
byte toucheBcv;//Utilisée pour enregistrer la deuxième touche appuyée
byte toucheCcv;//Utilisée pour enregistrer la troisième touche appuyée
byte toucheDcv;//Utilisée pour enregistrer la quatrième touche appuyée

int nbre_cv;
int adresse_cv;
int autorise_touche_Cv;
int cont7cv;int cont6cv;int cont5cv;int cont4cv;
int cont3cv;int cont2cv;int cont1cv;int cont0cv;

int comm7cv = 1;int comm6cv = 1;int comm5cv = 1;int comm4cv = 0;
int comm3cv = 1;int comm2cv = 1;int comm1cv = 0;int comm0cv = 0;

unsigned char adr7cv;unsigned char adr6cv;unsigned char adr5cv;unsigned char adr4cv;
unsigned char adr3cv;unsigned char adr2cv;unsigned char adr1cv;unsigned char adr0cv;
```

```
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
//Déclaration variable pour programmation CV2
int compteur_touche_cv2;
byte toucheAcv2;//Utilisée pour enregistrer la première touche appuyée
byte toucheBcv2;//Utilisée pour enregistrer la deuxième touche appuyée
byte toucheCcv2;//Utilisée pour enregistrer la troisième touche appuyée
byte toucheDcv2;//Utilisée pour enregistrer la quatrième touche appuyée

int nbre_cv2;
int adresse_cv2;
int autorise_touche_Cv2;

unsigned char adr7cv2;unsigned char adr6cv2;unsigned char adr5cv2;unsigned char adr4cv2;
unsigned char adr3cv2;unsigned char adr2cv2;unsigned char adr1cv2;unsigned char adr0cv2;
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
//Déclaration variable pour programmation CV3
int compteur_touche_cv3;
byte toucheAcv3;//Utilisée pour enregistrer la première touche appuyée
byte toucheBcv3;//Utilisée pour enregistrer la deuxième touche appuyée
byte toucheCcv3;//Utilisée pour enregistrer la troisième touche appuyée
byte toucheDcv3;//Utilisée pour enregistrer la quatrième touche appuyée

int nbre_cv3;
int adresse_cv3;
int autorise_touche_Cv3;

unsigned char adr7cv3;unsigned char adr6cv3;unsigned char adr5cv3;unsigned char adr4cv3;
unsigned char adr3cv3;unsigned char adr2cv3;unsigned char adr1cv3;unsigned char adr0cv3;
////////////////////////////////////////////////////////////////////////////////
```

```
////////////////////////////////////////////////////////////////////////////////
//Déclaration variable pour programmation CV4
int compteur_touche_cv4;
byte toucheAcv4;//Utilisée pour enregistrer la première touche appuyée
byte toucheBcv4;//Utilisée pour enregistrer la deuxième touche appuyée
byte toucheCcv4;//Utilisée pour enregistrer la troisième touche appuyée
byte toucheDcv4;//Utilisée pour enregistrer la quatrième touche appuyée

int nbre_cv4;
int adresse_cv4;
int autorise_touche_Cv4;

unsigned char adr7cv4;unsigned char adr6cv4;unsigned char adr5cv4;unsigned char adr4cv4;
unsigned char adr3cv4;unsigned char adr2cv4;unsigned char adr1cv4;unsigned char adr0cv4;
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
int dureetempo=30;
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
byte compteur_touche_locomotive = 0;
byte traitement_appui_touches_locomotive = 0;
byte toucheA_locomotive;//Utilisée pour enregistrer la première touche appuyée
byte toucheB_locomotive;//Utilisée pour enregistrer la deuxième touche appuyée
byte toucheC_locomotive;//Utilisée pour enregistrer la troisième touche appuyée
byte toucheD_locomotive;//Utilisée pour enregistrer la quatrième touche appuyée

//Déclaration des variables de sélection de l'adresse de la locomotive octet 1
unsigned char adr0;unsigned char adr1;unsigned char adr2;unsigned char adr3;
unsigned char adr4;unsigned char adr5;unsigned char adr6;unsigned char adr7;

//Déclaration des variables de commande de la locomotive
```

```
unsigned char comm0;unsigned char comm1;unsigned char comm2;unsigned char comm3;
unsigned char comm4;unsigned char comm5;unsigned char comm6;unsigned char comm7;

//Déclaration des variables pour le calcul du OU exclusif
unsigned char cont0;unsigned char cont1;unsigned char cont2;unsigned char cont3;
unsigned char cont4;unsigned char cont5;unsigned char cont6;unsigned char cont7;

//Déclaration variable valeur Lue pour potentiomètre de vitesse
unsigned int ValeurLue; //Détermination de la vitesse par le potentiomètre

//Déclaration envoi trame
int trame_locomotiv;//Interdit ou permet la rentrée dans la fonction trame_locomotive

//Déclaration sortie DCC
unsigned char sdcc = 2; //Pin 2 en sortie pour signal DCC
unsigned char sdcc2 = 3; //Pin 3 en sortie pour signal DCC 2

//Déclaration des variables des LED
char ledbleue=23;//Pin 23
char ledverte=25;//Pin 25
char ledjaune=A12;//Pin A12
char ledrouge=A13;//Pin A13
char ledSensArriere=A14;//Pin A14
char ledSensAvant=A15;//Pin A15

//Déclaration des boutons en entrée
unsigned char ArretUrg = 7;//Déclaration bouton Arrêt Urgence Pin 7

unsigned char SensdeMarche = 8;//Déclaration bouton Valid Trame Pin 8
unsigned char VarComm5;//Variable état sens de marche
```

```
int sensmarcheloc1;

int pas_de_vitesse;

int adresse_locomotive;//copie du nombre_locomotive
int nbre_locomotive;//Utilisée pour enregistrer l'intégralité du chiffre saisie

int i;
int compteur;//compteur pour la trame fonction

////////////////////////////////////////////////////////////////////////////
int retourdetrame;//Equivaut à une pression sur la touche *
////////////////////////////////////////////////////////////////////////////
//Déclaration des variable de fonction
int FonctionFL;int FonctionF1;int FonctionF2;int FonctionF3;int FonctionF4;

int boucleTrameFL0;//Permet la permutation de la fonction FL Haut ou Bas
int boucleTrameFL1;//Permet la permutation de la fonction FL Haut ou Bas
int boucleTrameF11;//Permet la permutation de la fonction F1 Haut ou Bas
int boucleTrameF10;//Permet la permutation de la fonction F1 Haut ou Bas
int boucleTrameF21;//Permet la permutation de la fonction F2 Haut ou Bas
int boucleTrameF20;//Permet la permutation de la fonction F2 Haut ou Bas
int boucleTrameF31;//Permet la permutation de la fonction F3 Haut ou Bas
int boucleTrameF30;//Permet la permutation de la fonction F3 Haut ou Bas
int boucleTrameF41;//Permet la permutation de la fonction F4 Haut ou Bas
int boucleTrameF40;//Permet la permutation de la fonction F4 Haut ou Bas

int octetFonction = 0b10000000;//Prépositionne l'octet de commande de fonction
int octetFonction1011 = 0b10110000;//Prépositionne l'octet de commande de fonction
int octetFonction1010 = 0b10100000;//Prépositionne l'octet de commande de fonction
int octetFonction11011110=0b00000000;//Prépositionne l'octet de commande de fonction
```

```
int octetFonction11011111=0b00000000;//Prépositionne l'octet de commande de fonction
////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////
byte fonc17;byte fonc16;byte fonc15;byte fonc14;
byte fonc13;byte fonc12;byte fonc11;byte fonc10;
////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////
byte fonc27;byte fonc26;byte fonc25;byte fonc24;
byte fonc23;byte fonc22;byte fonc21;byte fonc20;
////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////
//Déclaration des pins de l'arduino pour les boutons de fonction
unsigned char Fonction0 = 13;//Déclaration bouton FonctionFL Pin 13
unsigned char Fonction1 = 12;//Déclaration bouton FonctionFL Pin 12
unsigned char Fonction2 = 11;//Déclaration bouton FonctionFL Pin 11
unsigned char Fonction3 = 10;//Déclaration bouton FonctionFL Pin 10
unsigned char Fonction4 = 9;//Déclaration bouton FonctionFL Pin 9

// --- Déclaration des variables du clavier ---
byte autorise_touche_locomotive;

// --- Fonctionnalités utilisées ---
// Utilise un afficheur LCD alphanumérique 4x20 en mode 4 bits
// Utilise un Clavier matriciel 4x4 (16 touches)

// --- Circuit à réaliser ---
// Connecter sur la broche 37 la Colonne 1 du Clavier
// Connecter sur la broche 35 la Colonne 2 du Clavier
// Connecter sur la broche 33 la Colonne 2 du Clavier
// Connecter sur la broche 31 la Colonne 4 du Clavier
```

```
// Connecter sur la broche 45 la Ligne 1 du Clavier
// Connecter sur la broche 43 la Ligne 2 du Clavier
// Connecter sur la broche 41 la Ligne 3 du Clavier
// Connecter sur la broche 39 la Ligne 4 du Clavier

// Connecter sur la broche 32 la broche RS du LCD
// Connecter sur la broche 30 la broche E du LCD
// Connecter sur la broche 22 la broche D4 du LCD
// Connecter sur la broche 24 la broche D5 du LCD
// Connecter sur la broche 26 la broche D6 du LCD
// Connecter sur la broche 28 la broche D7 du LCD

// --- Inclusion des librairies utilisées ---

#include <LiquidCrystalFast.h> // Inclusion de la librairie pour afficheur LCD
#include <Keypad.h>

// --- Déclaration des constantes ---

//--- Constantes utilisées avec le clavier 4x4
const byte LIGNES = 4; // 4 lignes
const byte COLONNES = 4; //4 colonnes

// --- constantes des broches ---
//Clavier
//Colonnes
const byte C1=45; //déclaration constante de broche 45
const byte C2=43; //déclaration constante de broche 43
const byte C3=41; //déclaration constante de broche 41
const byte C4=39; //déclaration constante de broche 39
//Lignes
```

```
const byte L1=37; //déclaration constante de broche 37
const byte L2=35; //déclaration constante de broche 35
const byte L3=33; //déclaration constante de broche 33
const byte L4=31; //déclaration constante de broche 31

//LCD
const byte RS=32; //déclaration constante de broche 32
const byte E=30;  //déclaration constante de broche 30
const byte D4=22; //déclaration constante de broche 22
const byte D5=24; //déclaration constante de broche 24
const byte D6=26; //déclaration constante de broche 26
const byte D7=28; //déclaration constante de broche 28

// --- Déclaration des variables de boucle du clavier ---
// --- Déclaration des variables globales ---
//--- Définition des touches tableau 1
char touches[LIGNES][COLONNES] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'E','0','=','D'}
};
// tableaux de lignes et colonnes
byte BrochesLignes[LIGNES] = {L1, L2, L3, L4};
byte BrochesColonnes[COLONNES] = {C1, C2, C3, C4};

char touche; // variable de stockage valeur touche appuyée

// --- Déclaration des objets utiles pour les fonctionnalités utilisées ---
LiquidCrystalFast lcd(RS, E, D4, D5, D6, D7);
```

```
// création d'un objet keypad = initialisation clavier

Keypad keypad = Keypad( makeKeymap(touches), BrochesLignes, BrochesColonnes, LIGNES,
COLONNES );


//**************** FONCTION SETUP = Code d'initialisation *****

// La fonction setup() est exécutée en premier et 1 seule fois


void setup()   { // début de la fonction setup()

pinMode(ledbleue, OUTPUT);

pinMode(ledverte, OUTPUT);

pinMode(ledjaune, OUTPUT);

pinMode(ledrouge, OUTPUT);

pinMode(ledSensArriere, OUTPUT);

pinMode(ledSensAvant, OUTPUT);


//Déclaration des boutons en entrées

pinMode(Fonction0, INPUT_PULLUP);//On met le bouton en entrée 13

pinMode(Fonction1, INPUT_PULLUP);//On met le bouton en entrée 12

pinMode(Fonction2, INPUT_PULLUP);//On met le bouton en entrée 11

pinMode(Fonction3, INPUT_PULLUP);//On met le bouton en entrée 10

pinMode(Fonction4, INPUT_PULLUP);//On met le bouton en entrée 9


//Déclaration des boutons en entrées

pinMode(ArretUrg, INPUT_PULLUP);//On met le bouton en entrée 7

pinMode(SensdeMarche, INPUT_PULLUP);//On met le bouton en entrée 8

////////////////////////////////////

////////////////////////////////////

//Déclaration sortie DCC

pinMode(sdcc, OUTPUT); //Pin 2 en sortie pour signal DCC

pinMode(sdcc2,OUTPUT); //Pin 3 en sortie pour signal DCC 2

////////////////////////////////////
```

```
////////////////////////////////////
// --- ici instructions à exécuter au démarrage ---

lcd.begin(20,4); // Initialise le LCD avec 20 colonnes x 4 lignes

delay(10); // Pause rapide pour laisser le temps d'initialisation

// Test du LCD
lcd.setCursor(5,0);//Place le curseur colonne 5, ligne 1
lcd.print("CENTRALE DCC") ; // Affiche la chaîne texte

lcd.setCursor(0,2);//Place le curseur colonne 0, ligne 3
lcd.print("Programme by AERONEF") ; // Affiche la chaîne texte
delay(1500); // pause de 2 secondes

lcd.noDisplay();//Efface le texte, mais le garde en mémoire
delay(1000); // pause de 1 seconde

lcd.display();//Affiche le texte gardé en mémoire
delay(1500); // pause de 2 secondes

lcd.clear(); // // Efface l'écran
delay(10); // Pour laisser temps effacer écran

lcd.setCursor(0,1);//Place le curseur colonne 0, ligne 2
lcd.print("Le Site Ferroviaire") ; // Affiche la chaîne texte
delay(1500); // pause de 1.5 secondes

lcd.noDisplay();//Efface le texte, mais le garde en mémoire
delay(1000); // pause de 1 seconde
```

```
lcd.display();//Affiche le texte gardé en mémoire
delay(1500); // pause de 1.5 secondes

lcd.clear(); // // Efface l'écran
delay(10); // Pour laisser temps effacer écran
lcd.setCursor(0,0);//Place le curseur en haut à gauche
lcd.blink();

lcd.print("     En ATTENTE") ; // Affiche la chaîne texte
lcd.setCursor(0,2);//Place le curseur colonne 0, ligne 3
lcd.print("    APPUI TOUCHE") ; // Affiche la chaîne texte

// les broches de lignes et d'entrée sont configurées automatiquement
// lors de l'initialisation du clavier

// ------- Broches en sortie -------
// ------- Broches en entrée -------
// ------- Activation du rappel au + interne des broches en entrée si nécessaire -------
//sensmarcheloc1=1;
//vitesseloc1=1;
trame_locomotiv = 0;//au lancement de la centrale des trames idle sont générées

} // fin de la fonction setup()
// ******************************************************************************
// ******************************************************************************
// ******************************************************************************
//*************** FONCTION LOOP = Boucle sans fin = coeur du programme *************
// la fonction loop() s'exécute sans fin en boucle aussi longtemps que l'Arduino est sous tension

void loop(){ // Début de la fonction loop()
//Détermination de la combinaison des boutons activés
```

```
//Trame Arrêt Urgence
while (digitalRead(ArretUrg) == HIGH )
{
digitalWrite(ledrouge,HIGH);//Lampe URG Allumée, centrale bloquée
digitalWrite(ledbleue,LOW);
digitalWrite(ledverte,LOW);
digitalWrite(ledjaune,LOW);

TrameArretUrg();//Envoie Trame Arrêt Urgence
}
digitalWrite(ledrouge,LOW);

// --- ici instructions à exécuter par le programme principal ---

touche = keypad.getKey(); // Lecture de la touche appuyée

if (touche != NO_KEY)// Si une touche a été frappée -- gestion de la touche appuyée
   delay(15); // Pause entre 2 appuis: anti-rebond

   if (touche == '=' || retourdetrame == 1)
{
   retourdetrame = 0;//empêche la rentrée dans la boucle en permanence

   trame_locomotiv = 0;

   //Gestion des claviers
   autorise_touche_locomotive = 0;//Interdit l'entrée dans la boucle locomotive
   autorise_touche_fonction = 0;//Interdit l'entrée dans la boucle fonction
   autorise_touche_Cv = 0;//Interdit la saisie de chiffre pour programmation cv
   autorise_touche_Cv2 = 0;//Interdit la saisie de chiffre pour programmation_Cv2
   autorise_touche_Cv3 = 0;//Interdit la saisie de chiffre pour programmation_Cv3
```

```
autorise_touche_Cv4 = 0;//Interdit la saisie de chiffre pour programmation_Cv3

//Remise à zéros des compteurs pour touche
compteur_touche_locomotive = 0;
compteur_touche_fonction = 0;
compteur_touche_cv2 = 0;
compteur_touche_cv4 = 0;

//Remise à zéro des copies de touche saisies
toucheA_locomotive = 0;//Réinitialise toucheA
toucheB_locomotive = 0;//Réinitialise toucheB
toucheC_locomotive = 0;//Réinitialise toucheC
toucheD_locomotive = 0;//Réinitialise toucheD

toucheAcv2 = 0;//Réinitialise toucheA
toucheBcv2 = 0;//Réinitialise toucheB
toucheCcv2 = 0;//Réinitialise toucheC

toucheAcv4 = 0;//Réinitialise toucheA
toucheBcv4 = 0;//Réinitialise toucheB
toucheCcv4 = 0;//Réinitialise toucheC

toucheAfonction = 0;//Réinitialise toucheA fonction
toucheBfonction = 0;//Réinitialise toucheB fonction
toucheCfonction = 0;//Réinitialise toucheC fonction
toucheDfonction = 0;//Réinitialise toucheD fonction
toucheEfonction = 0;//Réinitialise toucheE fonction

//RAZ du nbre des touches appuyées
nbre_locomotive = 0;
nbre_fonction = 0;
```

```
    nbre_cv = 0;
    nbre_cv2 = 0;
    nbre_cv4 = 0;

    lcd.clear(); // Efface écran si appui
    lcd.setCursor(0,0);//Place le curseur colonne 1, ligne 1

    lcd.print("     En ATTENTE") ; // Affiche la chaîne texte
    lcd.setCursor(0,2);//Place le curseur colonne 0, ligne 3
    lcd.print("    APPUI TOUCHE") ; // Affiche la chaîne texte

    lcd.blink();
  }//Fin du if (touche == '=' || retourdetrame == 1)

    if (touche == 'A') //Locomotive
{
    retourdetrame = 0;
    trame_locomotiv = 0; //envoi trame idle

    compteur_touche_locomotive = 0;
    compteur_touche_fonction = 0;
    compteur_touche_cv2 = 0;
    compteur_touche_cv4 = 0;

    //Gestion des claviers
    autorise_touche_locomotive = 1;
    autorise_touche_fonction = 0;
    autorise_touche_Cv = 0;//Interdit l'entrée et l'affichage de nombres dans la boucle
programmationCv
    autorise_touche_Cv2 = 0;//Interdit la saisie de chiffre pour programmation_cv2
    autorise_touche_Cv3 = 0;//Interdit la saisie de chiffre pour programmation_Cv3
```

```
autorise_touche_Cv4 = 0;//Interdit la saisie de chiffre pour programmation_Cv4

traitement_appui_touches_locomotive = 1;

//Remise à zéro des copies de touche saisies
toucheA_locomotive = 0;//Réinitialise toucheA
toucheB_locomotive = 0;//Réinitialise toucheB
toucheC_locomotive = 0;//Réinitialise toucheC
toucheD_locomotive = 0;//Réinitialise toucheD

toucheAcv2 = 0;//Réinitialise toucheA
toucheBcv2 = 0;//Réinitialise toucheB
toucheCcv2 = 0;//Réinitialise toucheC

toucheAcv4 = 0;//Réinitialise toucheA
toucheBcv4 = 0;//Réinitialise toucheB
toucheCcv4 = 0;//Réinitialise toucheC

toucheAfonction = 0;//Réinitialise toucheA fonction
toucheBfonction = 0;//Réinitialise toucheB fonction
toucheCfonction = 0;//Réinitialise toucheC fonction
toucheDfonction = 0;//Réinitialise toucheD fonction
toucheEfonction = 0;//Réinitialise toucheE fonction

lcd.clear(); // Efface écran si appui = sinon  affiche touche
lcd.setCursor(0,0);//Place le curseur colonne 1, ligne 1

lcd.print("N Locomotive : ?"); // Affiche la chaîne texte

lcd.setCursor(0,2);//Place le curseur colonne 0, ligne 3
lcd.print ("Choix MPJ");
```

```
    lcd.setCursor(0,3);//Place le curseur colonne 0, ligne 4
    lcd.print ("Choix VITESSE");

    lcd.setCursor(8,1);//Place le curseur colonne 8, ligne 2

    lcd.blink();

} // fin du if (touche == 'A')

if (touche == 'B') //CV
{
   retourdetrame = 0;
   trame_locomotiv = 0;//Interdit la rentrée dans la fonction trame_locomotive

   compteur_touche_locomotive = 0;//remet compteur_touche à 0 pour prise en compte nouvel
appui
   compteur_touche_fonction = 0;
   compteur_touche_cv = 0;//remet compteur_touche à 0 pour prise en compte nouvel appui
   compteur_touche_cv2 = 0;
   compteur_touche_cv4 = 0;

   //Gestion des claviers
   autorise_touche_locomotive = 0;
   autorise_touche_fonction = 0;
   autorise_touche_Cv = 1;//Autorise l'entrée et l'affichage de nombres dans la boucle
programmation_Cv
   autorise_touche_Cv2 = 0;//Interdit la saisie de chiffre pour programmation_cv2
   autorise_touche_Cv3 = 0;//Interdit la saisie de chiffre pour programmation_Cv3
   autorise_touche_Cv4 = 0;//Interdit la saisie de chiffre pour programmation_Cv4

    //Remise à zéro des copies de touche saisies
```

```
 toucheA_locomotive = 0;//Réinitialise toucheA
 toucheB_locomotive = 0;//Réinitialise toucheB
 toucheC_locomotive = 0;//Réinitialise toucheC
 toucheD_locomotive = 0;//Réinitialise toucheD

 toucheAcv = 0;//Réinitialise toucheA
 toucheBcv = 0;//Réinitialise toucheB
 toucheCcv = 0;//Réinitialise toucheC
 toucheDcv = 0;//Réinitialise toucheD

 toucheAcv2 = 0;//Réinitialise toucheA
 toucheBcv2 = 0;//Réinitialise toucheB
 toucheCcv2 = 0;//Réinitialise toucheC
 toucheDcv2 = 0;//Réinitialise toucheD

 toucheAcv4 = 0;//Réinitialise toucheA
 toucheBcv4 = 0;//Réinitialise toucheB
 toucheCcv4 = 0;//Réinitialise toucheC
 toucheDcv4 = 0;//Réinitialise toucheD

 toucheAfonction = 0;//Réinitialise toucheA fonction
 toucheBfonction = 0;//Réinitialise toucheB fonction
 toucheCfonction = 0;//Réinitialise toucheC fonction
 toucheDfonction = 0;//Réinitialise toucheD fonction
 toucheEfonction = 0;//Réinitialise toucheE fonction

nbre_cv = 0;//Remise à zéro du calcul de la première adresse

lcd.clear(); // Efface écran si appui = sinon  affiche touche
lcd.setCursor(2,0);//Place le curseur colonne 2, ligne 1
lcd.print("Programmation Cv"); // Affiche la chaîne texte
```

```
  lcd.setCursor(0,1);//Place le curseur colonne 0, ligne 2
  lcd.print("Atte Adresse depart"); // Affiche la chaîne texte
  lcd.setCursor(9,2);//Place le curseur colonne 9, ligne 3

  lcd.clear(); // Efface écran si appui = sinon  affiche touche
  lcd.setCursor(2,0);//Place le curseur colonne 2, ligne 1
  lcd.print("Programmation Cv"); // Affiche la chaîne texte

  lcd.setCursor(0,1);//Place le curseur colonne 0, ligne 2
  lcd.print("Atte Adresse depart"); // Affiche la chaîne texte
  lcd.setCursor(9,2);//Place le curseur colonne 9, ligne 3

  lcd.blink();

} //Fin if (touche == 'B')


if (touche == 'D') //FONCTION
{
   retourdetrame = 0;
   trame_locomotiv = 0; //envoi trame idle

   //Gestion des claviers
   autorise_touche_locomotive = 0;
   autorise_touche_fonction = 1;
   autorise_touche_Cv = 0;
   autorise_touche_Cv2 = 0;//Interdit la saisie de chiffre pour programmation_cv2
   autorise_touche_Cv3 = 0;//Interdit la saisie de chiffre pour programmation_Cv3
   autorise_touche_Cv4 = 0;//Interdit la saisie de chiffre pour programmation_Cv4
```

```
traitement_appui_touches_fonction = 1;

//Remise à zéros des compteurs pour touche
compteur_touche_locomotive = 0;
compteur_touche_fonction = 0;
compteur_touche_cv = 0;
compteur_touche_cv2 = 0;
compteur_touche_cv4 = 0;

//Remise à zéro des copies de touche saisies
toucheA_locomotive = 0;//Réinitialise toucheA
toucheB_locomotive = 0;//Réinitialise toucheB
toucheC_locomotive = 0;//Réinitialise toucheC
toucheD_locomotive = 0;//Réinitialise toucheD

toucheAcv = 0;//Réinitialise toucheA
toucheBcv = 0;//Réinitialise toucheB
toucheCcv = 0;//Réinitialise toucheC
toucheDcv = 0;//Réinitialise toucheD

toucheAcv2 = 0;//Réinitialise toucheA
toucheBcv2 = 0;//Réinitialise toucheB
toucheCcv2 = 0;//Réinitialise toucheC
toucheDcv2 = 0;//Réinitialise toucheD

toucheAcv4 = 0;//Réinitialise toucheA
toucheBcv4 = 0;//Réinitialise toucheB
toucheCcv4 = 0;//Réinitialise toucheC
toucheDcv4 = 0;//Réinitialise toucheD

toucheAfonction = 0;//Réinitialise toucheA fonction
```

```
   toucheBfonction = 0;//Réinitialise toucheB fonction
   toucheCfonction = 0;//Réinitialise toucheC fonction
   toucheDfonction = 0;//Réinitialise toucheD fonction
   toucheEfonction = 0;//Réinitialise toucheE fonction

   lcd.clear(); // Efface écran si appui = sinon  affiche touche
   lcd.setCursor(7,0);//Place le curseur colonne 7, ligne 1
   lcd.print ("ENTRER");

   lcd.setCursor(4,1);//Place le curseur colonne 4, ligne 2
   lcd.print ("Le NUMERO de");

   lcd.setCursor(6,2);//Place le curseur colonne 6, ligne 3
   lcd.print ("FONCTION");

   lcd.setCursor(8,3);//Place le curseur colonne 6, ligne 4

   lcd.blink();
}//Fin if (touche == 'D')
////////////////////////////////////////////////////////////////////////////////
///Permet la saisie d'un chiffre si une autorise_touche_locomotive est à 1//////////////
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////

if (touche == '0' && autorise_touche_locomotive == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_locomotive();
} // fin gestion de la touche appuyée

 if (touche == '1' && autorise_touche_locomotive == 1)
```

```
{
   lcd.print(touche);
   touche_appuyee_numerique_locomotive();
} // fin gestion de la touche appuyée

if (touche == '2' && autorise_touche_locomotive == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_locomotive();
} // fin gestion de la touche appuyée

if (touche == '3' && autorise_touche_locomotive == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_locomotive();
} // fin gestion de la touche appuyée

if (touche == '4' && autorise_touche_locomotive == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_locomotive();
} // fin gestion de la touche appuyée

if (touche == '5' && autorise_touche_locomotive == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_locomotive();
} // fin gestion de la touche appuyée

if (touche == '6' && autorise_touche_locomotive == 1)
{
```

```
   lcd.print(touche);
   touche_appuyee_numerique_locomotive();
} // fin gestion de la touche appuyée

if (touche == '7' && autorise_touche_locomotive == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_locomotive();
} // fin gestion de la touche appuyée

if (touche == '8' && autorise_touche_locomotive == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_locomotive();
} // fin gestion de la touche appuyée

if (touche == '9' && autorise_touche_locomotive == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_locomotive();
} // fin gestion de la touche appuyée
////////////////////////////////////////////////
////////////////////////////////////////////////
 if ((touche == 'E' && traitement_appui_touches_locomotive == 1
))traitement_appui_touche_locomotive();}

 //Les chiffres saisis individuellement sont stockés sous la forme d'un nombre dans la variable "nbre"
 // fin gestion de la touche appuyée
////////////////////////////////////////////////
////////////////////////////////////////////////
//Appel fonctions
```

```
FonctionFL = digitalRead(Fonction0);
FonctionF1 = digitalRead(Fonction1);
FonctionF2 = digitalRead(Fonction2);
FonctionF3 = digitalRead(Fonction3);
FonctionF4 = digitalRead(Fonction4);

//Appel fonctionFL

if (FonctionFL == HIGH && (boucleTrameFL1) == 0)
{
boucleTrameFL1 = 1;//Empêche la rentrée dans la boucle en permanence. Croisement des boucles
boucleTrameFL0 = 0;//Permet la rentrée dans la boucle. Croisement des boucles

octetFonction = bitSet (octetFonction, 4);//Active la fonction

envoi_trame_fonction();// Appel trame_fonction
}//Fin FonctionFL

if (FonctionFL == LOW && (boucleTrameFL0) == 0)
{
boucleTrameFL1 = 0;//Permet la rentrée dans la boucle
boucleTrameFL0 = 1;//Empêche la rentrée dans la boucle en permanence.

octetFonction = bitClear (octetFonction, 4);//Active la fonction//Désactive la fonction

envoi_trame_fonction();// Appel trame_fonction

}//Fin FonctionFL
//////////////////////////////////////////
//////////////////////////////////////////
//Appel fonctionF1
```

```
if (FonctionF1 == HIGH && (boucleTrameF11) == 0)
{
boucleTrameF11 = 1;//Empêche la rentrée dans la boucle en permanence.
boucleTrameF10 = 0;//Permet la rentrée dans la boucle. Croisement des boucles

octetFonction = bitSet (octetFonction, 0);//Active la fonction

envoi_trame_fonction();// Appel trame_fonction
}//Fin FonctionF1

if (FonctionF1 == LOW && (boucleTrameF10) == 0)
{
boucleTrameF11 = 0;//Permet la rentrée dans la boucle. Croisement des boucles
boucleTrameF10 = 1;//Empêche la rentrée dans la boucle en permanence.

octetFonction = bitClear (octetFonction, 0);//Active la fonction//Désactive la fonction

envoi_trame_fonction();// Appel trame_fonction

}//Fin FonctionF1
//////////////////////////////////////////
//////////////////////////////////////////
//Appel fonctionF2

if (FonctionF2 == HIGH && (boucleTrameF21) == 0)
{
boucleTrameF21 = 1;//Empêche la rentrée dans la boucle en permanence.
boucleTrameF20 = 0;//Permet la rentrée dans la boucle. Croisement des boucles

octetFonction = bitSet (octetFonction, 1);//Active la fonction
```

```
envoi_trame_fonction();// Appel trame_fonction
}//Fin FonctionF1

if (FonctionF2 == LOW && (boucleTrameF20) == 0)
{
boucleTrameF21 = 0;//Permet la rentrée dans la boucle. Croisement des boucles
boucleTrameF20 = 1;//Empêche la rentrée dans la boucle en permanence.

octetFonction = bitClear (octetFonction, 1);//Active la fonction//Désactive la fonction

envoi_trame_fonction();// Appel trame_fonction

}//Fin FonctionF2
//////////////////////////////////////////
//////////////////////////////////////////
//Appel fonctionF3

if (FonctionF3 == HIGH && (boucleTrameF31) == 0)
{
boucleTrameF31 = 1;//Empêche la rentrée dans la boucle en permanence.
boucleTrameF30 = 0;//Permet la rentrée dans la boucle. Croisement des boucles

octetFonction = bitSet (octetFonction, 2);//Active la fonction

envoi_trame_fonction();// Appel trame_fonction
}//Fin FonctionF1

if (FonctionF3 == LOW && (boucleTrameF30) == 0)
{
boucleTrameF31 = 0;//Permet la rentrée dans la boucle. Croisement des boucles
```

```
boucleTrameF30 = 1;//Empêche la rentrée dans la boucle en permanence.

octetFonction = bitClear (octetFonction, 2);//Active la fonction//Désactive la fonction

envoi_trame_fonction();// Appel trame_fonction

}//Fin FonctionF3
//////////////////////////////////////////
//////////////////////////////////////////
//Appel fonctionF4

if (FonctionF4 == HIGH && (boucleTrameF41) == 0)
{
boucleTrameF41 = 1;//Empêche la rentrée dans la boucle en permanence.
boucleTrameF40 = 0;//Permet la rentrée dans la boucle. Croisement des boucles

octetFonction = bitSet (octetFonction, 3);//Active la fonction

envoi_trame_fonction();// Appel trame_fonction
}//Fin FonctionF1

if (FonctionF4 == LOW && (boucleTrameF40) == 0)
{
boucleTrameF41 = 0;//Permet la rentrée dans la boucle. Croisement des boucles
boucleTrameF40 = 1;//Empêche la rentrée dans la boucle en permanence.

octetFonction = bitClear (octetFonction, 3);//Active la fonction//Désactive la fonction

envoi_trame_fonction();// Appel trame_fonction

}//Fin FonctionF4
```

```
////////////////////////////////////
////////////////////////////////////
//Détermine le sens de marche
//valeur bit commande 5 sens de marche
VarComm5 = digitalRead(SensdeMarche);

//Choisir le sens en fonction du mouvement de la loc avant d'envoyer la salve
if (VarComm5 == HIGH)
{
comm5=1;
PORTK |= (1<<7);//Equivalent à digitalWrite(LedAvant,HIGH)
PORTK &=~ (1<<6);//Equivalent à digitalWrite(LedArriere,LOW)
}
if (VarComm5 == LOW)
{
comm5=0;//Sens marche arrière
PORTK &=~ (1<<7);//Equivalent à digitalWrite(LedAvant,LOW)
PORTK |= (1<<6);//Equivalent à digitalWrite(LedArriere,HIGH)
}
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
////////////Permet la saisie d'un chiffre si Autorise_touche_Cv est à 1/////////////
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
if (touche == '0' && autorise_touche_Cv == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv();
 } // fin gestion de la touche appuyée

 if (touche == '1' && autorise_touche_Cv == 1)
```

```
{
   lcd.print(touche);
   touche_appuyee_numerique_cv();
} // fin gestion de la touche appuyée

if (touche == '2' && autorise_touche_Cv == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv();
 } // fin gestion de la touche appuyée

if (touche == '3' && autorise_touche_Cv == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv();
 } // fin gestion de la touche appuyée

if (touche == '4' && autorise_touche_Cv == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv();
 } // fin gestion de la touche appuyée

if (touche == '5' && autorise_touche_Cv == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv();
} // fin gestion de la touche appuyée

if (touche == '6' && autorise_touche_Cv == 1)
{
```

```
   lcd.print(touche);
   touche_appuyee_numerique_cv();
} // fin gestion de la touche appuyée

if (touche == '7' && autorise_touche_Cv == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv();
} // fin gestion de la touche appuyée

if (touche == '8' && autorise_touche_Cv == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv();
 } // fin gestion de la touche appuyée

if (touche == '9' && autorise_touche_Cv == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv();
 } // fin gestion de la touche appuyée
////////////////////////////////////////////////////////////////////////////////
////////Permet la saisie d'un chiffre si Autorise_touche_Cv2 est à 1//////////////
////////////////////////////////////////////////////////////////////////////////
if (touche == '0' && autorise_touche_Cv2 == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv2();
 } // fin gestion de la touche appuyée

 if (touche == '1' && autorise_touche_Cv2 == 1)
```

```
{
   lcd.print(touche);
   touche_appuyee_numerique_cv2();
 } // fin gestion de la touche appuyée

if (touche == '2' && autorise_touche_Cv2 == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv2();
} // fin gestion de la touche appuyée

if (touche == '3' && autorise_touche_Cv2 == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv2();
} // fin gestion de la touche appuyée

if (touche == '4' && autorise_touche_Cv2 == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv2();
} // fin gestion de la touche appuyée

if (touche == '5' && autorise_touche_Cv2 == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv2();

} // fin gestion de la touche appuyée

if (touche == '6' && autorise_touche_Cv2 == 1)
```

```
{
   lcd.print(touche);
   touche_appuyee_numerique_cv2();
} // fin gestion de la touche appuyée

if (touche == '7' && autorise_touche_Cv2 == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv2();
} // fin gestion de la touche appuyée

if (touche == '8' && autorise_touche_Cv2 == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv2();
} // fin gestion de la touche appuyée

if (touche == '9' && autorise_touche_Cv2 == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv2();
} // fin gestion de la touche appuyée
////////////////////////////////////////////////////////////////////////////////
/////////Permet la saisie d'un chiffre si Autorise_touche_Cv4 est à 1//////////////
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
if (touche == '0' && autorise_touche_Cv4 == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv4();
 } // fin gestion de la touche appuyée
```

```
 if (touche == '1' && autorise_touche_Cv4 == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv4();
 } // fin gestion de la touche appuyée

if (touche == '2' && autorise_touche_Cv4 == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv4();
 } // fin gestion de la touche appuyée

if (touche == '3' && autorise_touche_Cv4 == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv4();
 } // fin gestion de la touche appuyée

if (touche == '4' && autorise_touche_Cv4 == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv4();

} // fin gestion de la touche appuyée

if (touche == '5' && autorise_touche_Cv4 == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv4();
} // fin gestion de la touche appuyée
```

```
if (touche == '6' && autorise_touche_Cv4 == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv4();
 } // fin gestion de la touche appuyée

if (touche == '7' && autorise_touche_Cv4 == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv4();
  } // fin gestion de la touche appuyée

if (touche == '8' && autorise_touche_Cv4 == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv4();
  } // fin gestion de la touche appuyée

if (touche == '9' && autorise_touche_Cv4 == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_cv4();
 } // fin gestion de la touche appuyée
//////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////
////////////////////////Clavier RAZ///////////////////////////////
//////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////
if (touche == '0' && autorise_touche_fonction == 1)
{
```

```
   lcd.print(touche);
   touche_appuyee_numerique_fonction();
} // fin gestion de la touche appuyée

 if (touche == '1' && autorise_touche_fonction == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_fonction();
} // fin gestion de la touche appuyée

if (touche == '2' && autorise_touche_fonction == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_fonction();
} // fin gestion de la touche appuyée

if (touche == '3' && autorise_touche_fonction == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_fonction();
} // fin gestion de la touche appuyée

if (touche == '4' && autorise_touche_fonction == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_fonction();
} // fin gestion de la touche appuyée

if (touche == '5' && autorise_touche_fonction == 1)
{
   lcd.print(touche);
```

```
   touche_appuyee_numerique_fonction();
} // fin gestion de la touche appuyée

if (touche == '6' && autorise_touche_fonction == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_fonction();
} // fin gestion de la touche appuyée

if (touche == '7' && autorise_touche_fonction == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_fonction();
} // fin gestion de la touche appuyée

if (touche == '8' && autorise_touche_fonction == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_fonction();
} // fin gestion de la touche appuyée

if (touche == '9' && autorise_touche_fonction == 1)
{
   lcd.print(touche);
   touche_appuyee_numerique_fonction();
} // fin gestion de la touche appuyée
////////////////////////////////////////////////
////////////////////////////////////////////////
////////////////////////////////////////////////
////////////////////////////////////////////////
if (trame_locomotiv == 1)
```

```
{
digitalWrite(ledbleue,LOW);
digitalWrite(ledverte,HIGH);
digitalWrite(ledjaune,LOW);
trame_locomotive();}//Envoi trame_locomotive
else
{
digitalWrite(ledbleue,LOW);
digitalWrite(ledverte,LOW);
digitalWrite(ledjaune,HIGH);
trame_idle();}//Envoi trame_idle

//Fin du if
////////////////////////////////////////////////////
////////////////////////////////////////////////
////////////////////////////////////////////////
if ((touche == 'E' && autorise_touche_Cv == 1)){traitement_appui_touche_cv();}

 //Les chiffres saisis individuellement sont stockés dans la variable "nbre"
 // fin gestion de la touche appuyée
////////////////////////////////////////////////
////////////////////////////////////////////////
if ((touche == 'E' && autorise_touche_Cv2 == 1))
{
 traitement_appui_touche_cv2();
 }
  //Les chiffres saisis individuellement sont stockés dans la variable "nbre"
 // fin gestion de la touche appuyée
////////////////////////////////////////////////
////////////////////////////////////////////////
if ((touche == 'E' && autorise_touche_Cv4 == 1))
```

```
{
traitement_appui_touche_cv4();
}
//Les chiffres saisis individuellement sont stockés dans la variable "nbre"
 // fin gestion de la touche appuyée
////////////////////////////////////////////////
////////////////////////////////////////////////
 if ((touche == 'E' && autorise_touche_fonction == 1)){traitement_appui_touche_fonction();}

 //Les chiffres saisis individuellement sont stockés dans la variable "nbre"
 // fin gestion de la touche appuyée
////////////////////////////////////////////////
////////////////////////////////////////////////
}//fin du loop
////////////////////////////// ////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////////////////
//////////////////////////////AUTRES FONCTIONS DU PROGRAMME
////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////// ////////////////////////////////////////////////////////////////////////
////////////////////////// MISE EN MEMOIRE DES TOUCHES APPUYEES
LOCOMOTIVE//////////////////////
//////////////////////////////TOUCHE APPUYEE NUMERIQUE
LOCOMOTIVE////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////////////////
void touche_appuyee_numerique_locomotive() // lecture de la touche appuyée
{
  compteur_touche_locomotive = compteur_touche_locomotive + 1;

if (compteur_touche_locomotive == 1) //Teste si compteur_touche = 1
{ toucheA_locomotive = touche;}//Garde trace de la touche 1ère touche appuyée
```

```
if (compteur_touche_locomotive == 2) //Teste si compteur_touche = 2
{toucheB_locomotive = touche;}//Garde trace de la touche 2ième touche appuyée

if (compteur_touche_locomotive == 3) //Teste si compteur_touche = 3
{toucheC_locomotive = touche;}//Garde trace de la touche 3ième touche appuyée

if (compteur_touche_locomotive == 4) //Teste si compteur_touche = 4
{toucheD_locomotive = touche;}//Garde trace de la touche 4ième touche appuyée

}//Fin du VOID MISE EN MEMOIRE DES TOUCHES APPUYEES LOCOMOTIVE

////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
// ////////////////////////// TRAITEMENT DES TOUCHES APPUYEES LOCOMOTIVE//////////
//////////////////////////////////TRAITEMENT APPUI TOUCHE
LOCOMOTIVE//////////////////////////////////////////////////////////////////////////////////////////////
//
////////////////////////////////////////////////////////////////////////////
void traitement_appui_touche_locomotive()
{
if (compteur_touche_locomotive == 1)//Prise en compte du chiffre saisi pour une entrée
{
  nbre_locomotive = toucheA_locomotive-48;//toucheA contient l'intégralité de la saisie

  trame_locomotiv = 1; //Permet l'envoi de la trame_locomotive si à 1
  compteur_touche_locomotive = 0;//Réinitialise compteur_touche

  autorise_touche_locomotive = 0;
  autorise_touche_fonction = 0;
  autorise_touche_Cv = 0;//Interdit la prise en compte de l'appui touche E et programmation cv à 0
  autorise_touche_Cv2 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv2 à 0
  autorise_touche_Cv4 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv4 à 0
```

```
    toucheA_locomotive = 0;//Réinitialise toucheA locomotive
    toucheB_locomotive = 0;//Réinitialise toucheB locomotive
    toucheC_locomotive = 0;//Réinitialise toucheC locomotive

    lcd.clear(); // Efface écran

    lcd.setCursor(0,0);//Place le curseur colonne 0, ligne 1
    lcd.print("N Locomotive :"); // Affiche la chaîne texte

    lcd.print(nbre_locomotive); // Affiche la variable nbre

   adresse_locomotive = nbre_locomotive;//recopie dans adresse la valeur de nbre en binaire

    //Transforme la variable adresse en binaire et stocke dans les variables adr
    //Octet 1 d'adresse
    adr7 = 0;//Adresse courte
    adr6 = bitRead (adresse_locomotive, 6);//Met dans adr6, le bit 6 de la variable adresse
    adr5 = bitRead (adresse_locomotive, 5);
    adr4 = bitRead (adresse_locomotive, 4);
    adr3 = bitRead (adresse_locomotive, 3);
    adr2 = bitRead (adresse_locomotive, 2);
    adr1 = bitRead (adresse_locomotive, 1);
    adr0 = bitRead (adresse_locomotive, 0);

}// fin du if (compteur_touche_locomotive == 1)

if (compteur_touche_locomotive == 2)//Prise en compte du chiffre saisi pour deux entrées
{
    nbre_locomotive = ((toucheA_locomotive-48) * 10) + (toucheB_locomotive-48);
```

```
  compteur_touche_locomotive = 0;//Réinitialise compteur_touche
  trame_locomotiv = 1; //Permet l'envoi de la trame_locomotive si à 1

  autorise_touche_locomotive = 0;
  autorise_touche_fonction = 0;
  autorise_touche_Cv = 0;//Interdit la prise en compte de l'appui touche E et programmation cv à 0
  autorise_touche_Cv2 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv2 à
0
  autorise_touche_Cv4 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv4 à
0

  toucheA_locomotive = 0;//Réinitialise toucheA
  toucheB_locomotive = 0;//Réinitialise toucheB
  toucheC_locomotive = 0;//Réinitialise toucheC

  lcd.clear(); // Efface écran

  lcd.setCursor(0,0);//Place le curseur colonne 0, ligne 1
  lcd.print("N Locomotive :"); // Affiche la chaîne texte

  lcd.print(nbre_locomotive); // Affiche la variable nbre

  adresse_locomotive = nbre_locomotive;//recopie dans adresse la valeur de nbre en binaire

  //Transforme la variable adresse en binaire et stocke dans les variables adr
  //Octet 1 d'adresse
  adr7 = 0;//Adresse courte
  adr6 = bitRead (adresse_locomotive, 6);//Met dans adr6, le bit 6 de la variable adresse
  adr5 = bitRead (adresse_locomotive, 5);
  adr4 = bitRead (adresse_locomotive, 4);
  adr3 = bitRead (adresse_locomotive, 3);
  adr2 = bitRead (adresse_locomotive, 2);
```

```
   adr1 = bitRead (adresse_locomotive, 1);
   adr0 = bitRead (adresse_locomotive, 0);

}// fin du if (compteur_touche_locomotive == 2)

if (compteur_touche_locomotive == 3)//Prise en compte du chiffre saisi pour trois entrées
{
   nbre_locomotive = ((toucheA_locomotive-48) * 100) + ((toucheB_locomotive-
48)*10)+(toucheC_locomotive-48);

   if (nbre_locomotive <128)
   {
   compteur_touche_locomotive = 0;//Réinitialise compteur_touche
   trame_locomotiv = 1; //Permet l'envoi de la trame_locomotive si à 1

   autorise_touche_locomotive = 0;
   autorise_touche_fonction = 0;
   autorise_touche_Cv = 0;//Interdit la prise en compte de l'appui touche E et programmation cv à 0
   autorise_touche_Cv2 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv2 à
0
   autorise_touche_Cv4 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv4 à
0

   toucheA_locomotive = 0;//Réinitialise toucheA
   toucheB_locomotive = 0;//Réinitialise toucheB
   toucheC_locomotive = 0;//Réinitialise toucheC

   lcd.clear(); // Efface écran

   lcd.setCursor(0,0);//Place le curseur colonne 0, ligne 1
   lcd.print("N Locomotive :"); // Affiche la chaîne texte
```

```
   lcd.print(nbre_locomotive); // Affiche la variable nbre

   adresse_locomotive = nbre_locomotive;//recopie dans adresse la valeur de nbre en binaire

   //Transforme la variable adresse en binaire et stocke dans les variables adr
   //Octet 1 d'adresse
   adr7 = 0;//Adresse courte
   adr6 = bitRead (adresse_locomotive, 6);//Met dans adr6, le bit 6 de la variable adresse
   adr5 = bitRead (adresse_locomotive, 5);
   adr4 = bitRead (adresse_locomotive, 4);
   adr3 = bitRead (adresse_locomotive, 3);
   adr2 = bitRead (adresse_locomotive, 2);
   adr1 = bitRead (adresse_locomotive, 1);
   adr0 = bitRead (adresse_locomotive, 0);

   }//fin if nbre_locomotive <128
}// fin du if (compteur_touche_locomotive == 3)


 if (compteur_touche_locomotive > 3 || nbre_locomotive > 127)
{
   compteur_touche_locomotive = 0;//Réinitialise compteur_touche

   autorise_touche_locomotive = 0;
   autorise_touche_fonction = 0;
   autorise_touche_Cv = 0;//Interdit la prise en compte de l'appui touche E et programmation cv à 0
   autorise_touche_Cv2 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv2 à
0
   autorise_touche_Cv4 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv4 à
0

   toucheA_locomotive = 0;//Réinitialise toucheA
```

```
   toucheB_locomotive = 0;//Réinitialise toucheB
   toucheC_locomotive = 0;//Réinitialise toucheC

   lcd.clear(); // Efface écran si appui
   lcd.setCursor(0,0);//Place le curseur colonne 0, ligne 1

   lcd.print("   Nbre Trop Grand") ; // Affiche la chaîne texte

   for(int compteurtempo = 0; compteurtempo < dureetempo; compteurtempo++)
   {
    trame_idle();//Envoi trame_idle
   }//fermeture du for

   retourdetrame = 1;//retour à, comme une pression sur la touche *

}// fin if (compteur_touche_locomotive > 3 || nbre_locomotive > 127)

}//Fin du VOID TRAITEMENT DES APPUI TOUCHE LOCOMOTIVE
////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////
////////////////////// Génère un bit ZERO//////////////////////////
////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////
void bitzero()
{
PORTE &=~ (1<<4);//Equivalent à digitalWrite(sdcc,LOW), Sortie sdcc, état bas, broche 2
PORTE |= (1<<5);//Equivalent à digitalWrite(sdcc2,HIGH), Sortie sdcc2, état haut, broche 3
delayMicroseconds(105);       // Pause de 105 microsecondes
PORTE |= (1<<4);//Equivalent à digitalWrite(sdcc,HIGH), Sortie sdcc, état haut, broche 2
PORTE &=~ (1<<5);//Equivalent à digitalWrite(sdcc2,LOW), Sortie sdcc2, état bas, broche 3
delayMicroseconds(105);       // Pause de 105 microsecondes
```

```
}
////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////
////////////////////// Génère un bit UN//////////////////////
////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////
void bitun()   //Génère un bit à 1
{
PORTE &=~ (1<<4);//Equivalent à digitalWrite(sdcc,LOW), Sortie sdcc, état bas, broche 2
PORTE |= (1<<5);////Equivalent à digitalWrite(sdcc2,HIGH), Sortie sdcc2, état haut, broche 3
delayMicroseconds(58);       // Pause de 58 microsecondes
PORTE |= (1<<4);//Equivalent à digitalWrite(sdcc,HIGH), Sortie sdcc, état haut, broche 2
PORTE &=~ (1<<5);//Equivalent à digitalWrite(sdcc2,LOW), Sortie sdcc2, état bas, broche 3
delayMicroseconds(58);       // Pause de 58 microsecondes
}
////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////
////////////////////// TRAME LOCOMOTIVE//////////////////////
////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////
void trame_locomotive()
{
//Valeur Lue par le convertisseur analogique/numérique
ValeurLue = analogRead(A0); //la valeur lue sera comprise entre 0 et 1023 (potentiomètre) sur la broche A2

//La tension est égale à : ((ValeurLue*5)/1024)
lcd.setCursor (3,2);//Place le curseur colonne 9, ligne 3

//En fonction du résultat du convertisseur analogique/numérique.
if (ValeurLue < 10)(comm4=0,comm3=0,comm2=0,comm1=0,comm0=0,
```

```
 lcd.print("Vitesse : 00"), pas_de_vitesse = 0);
else if (ValeurLue > 10 && ValeurLue <= 30)(comm4=0,comm3=0,comm2=0,comm1=1,comm0=0,
lcd.print("Vitesse : 01"), pas_de_vitesse = 1); //1
else if (ValeurLue > 30 && ValeurLue <= 65)(comm4=1,comm3=0,comm2=0,comm1=1,comm0=0,
lcd.print("Vitesse : 02"), pas_de_vitesse = 2); //2
else if (ValeurLue > 65 && ValeurLue <= 100)(comm4=0,comm3=0,comm2=0,comm1=1,comm0=1,
lcd.print("Vitesse : 03"), pas_de_vitesse = 3); //3
else if (ValeurLue > 100 && ValeurLue <= 138)(comm4=1,comm3=0,comm2=0,comm1=1,comm0=1,
lcd.print("Vitesse : 04"), pas_de_vitesse = 4); //4
else if (ValeurLue > 138 && ValeurLue <= 175)(comm4=0,comm3=0,comm2=1,comm1=0,comm0=0,
lcd.print("Vitesse : 05"), pas_de_vitesse = 5); //5
else if (ValeurLue > 175 && ValeurLue <= 212)(comm4=1,comm3=0,comm2=1,comm1=0,comm0=0,
lcd.print("Vitesse : 06"), pas_de_vitesse = 6); //6
else if (ValeurLue > 212 && ValeurLue <= 249)(comm4=0,comm3=0,comm2=1,comm1=0,comm0=1,
lcd.print("Vitesse : 07"), pas_de_vitesse = 7); //7
else if (ValeurLue > 249 && ValeurLue <= 285)(comm4=1,comm3=0,comm2=1,comm1=0,comm0=1,
lcd.print("Vitesse : 08"), pas_de_vitesse = 8); //8
else if (ValeurLue > 285 && ValeurLue <= 320)(comm4=0,comm3=0,comm2=1,comm1=1,comm0=0,
lcd.print("Vitesse : 09"), pas_de_vitesse = 9); //9
else if (ValeurLue > 320 && ValeurLue <= 360)(comm4=1,comm3=0,comm2=1,comm1=1,comm0=0,
lcd.print("Vitesse : 10"), pas_de_vitesse = 10); //10
else if (ValeurLue > 360 && ValeurLue <= 395)(comm4=0,comm3=0,comm2=1,comm1=1,comm0=1,
lcd.print("Vitesse : 11"), pas_de_vitesse = 11); //11
else if (ValeurLue > 395 && ValeurLue <= 432)(comm4=1,comm3=0,comm2=1,comm1=1,comm0=1,
lcd.print("Vitesse : 12"), pas_de_vitesse = 12); //12
else if (ValeurLue > 432 && ValeurLue <= 467)(comm4=0,comm3=1,comm2=0,comm1=0,comm0=0,
lcd.print("Vitesse : 13"), pas_de_vitesse = 13); //13
else if (ValeurLue > 467 && ValeurLue <= 504)(comm4=1,comm3=1,comm2=0,comm1=0,comm0=0,
lcd.print("Vitesse : 14"), pas_de_vitesse = 14); //14
else if (ValeurLue > 504 && ValeurLue <= 540)(comm4=0,comm3=1,comm2=0,comm1=0,comm0=1,
lcd.print("Vitesse : 15"), pas_de_vitesse = 15); //15
```

```
else if (ValeurLue > 540 && ValeurLue <= 577)(comm4=1,comm3=1,comm2=0,comm1=0,comm0=1,
lcd.print("Vitesse : 16"), pas_de_vitesse = 16); //16
else if (ValeurLue > 577 && ValeurLue <= 603)(comm4=0,comm3=1,comm2=0,comm1=1,comm0=0,
lcd.print("Vitesse : 17"), pas_de_vitesse = 17); //17
else if (ValeurLue > 603 && ValeurLue <= 649)(comm4=1,comm3=1,comm2=0,comm1=1,comm0=0,
lcd.print("Vitesse : 18"), pas_de_vitesse = 18); //18
else if (ValeurLue > 649 && ValeurLue <= 687)(comm4=0,comm3=1,comm2=0,comm1=1,comm0=1,
lcd.print("Vitesse : 19"), pas_de_vitesse = 19); //19
else if (ValeurLue > 687 && ValeurLue <= 723)(comm4=1,comm3=1,comm2=0,comm1=1,comm0=1,
lcd.print("Vitesse : 20"), pas_de_vitesse = 20); //20
else if (ValeurLue > 723 && ValeurLue <= 760)(comm4=0,comm3=1,comm2=1,comm1=0,comm0=0,
lcd.print("Vitesse : 21"), pas_de_vitesse = 21); //21
else if (ValeurLue > 760 && ValeurLue <= 796)(comm4=1,comm3=1,comm2=1,comm1=0,comm0=0,
lcd.print("Vitesse : 22"), pas_de_vitesse = 22); //22
else if (ValeurLue > 796 && ValeurLue <= 833)(comm4=0,comm3=1,comm2=1,comm1=0,comm0=1,
lcd.print("Vitesse : 23"), pas_de_vitesse = 23); //23
else if (ValeurLue > 833 && ValeurLue <= 869)(comm4=1,comm3=1,comm2=1,comm1=0,comm0=1,
lcd.print("Vitesse : 24"), pas_de_vitesse = 24); //24
else if (ValeurLue > 869 && ValeurLue <= 906)(comm4=0,comm3=1,comm2=1,comm1=1,comm0=0,
lcd.print("Vitesse : 25"), pas_de_vitesse = 25); //25
else if (ValeurLue > 906 && ValeurLue <= 941)(comm4=1,comm3=1,comm2=1,comm1=1,comm0=0,
lcd.print("Vitesse : 26"), pas_de_vitesse = 26); //26
else if (ValeurLue > 941 && ValeurLue <= 977)(comm4=0,comm3=1,comm2=1,comm1=1,comm0=1,
lcd.print("Vitesse : 27"), pas_de_vitesse = 27); //27
else if (ValeurLue > 977 && ValeurLue <= 1024)(comm4=1,comm3=1,comm2=1,comm1=1,comm0=1,
lcd.print("Vitesse : 28"), pas_de_vitesse = 28); //28
//////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////
////////////////////// GENERE LA TRAME dcc LOCOMOTIVE///////////
//////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////
```

```
// Génération des paquets DCC
// Octet de synchronisation
 for ( i=0; i <= 16; i++)
 {bitun();}    // La centrale transmet 16 bits à 1
////////////////////////////////////
////////////////////////////////////
bitzero(); // Bit à 0 de séparation
////////////////////////////////////
////////////////////////////////////
// Octet d'adresse 1
//La centrale transmet l'octet d'adresse
bitzero();//adr7
if (adr6==1) bitun();  // Si bit à 1
if (adr6==0) bitzero(); // Si bit à 0
if (adr5==1) bitun();  // Si bit à 1
if (adr5==0) bitzero(); // Si bit à 0
if (adr4==1) bitun();  // Si bit à 1
if (adr4==0) bitzero(); // Si bit à 0
if (adr3==1) bitun();  // Si bit à 1
if (adr3==0) bitzero(); // Si bit à 0
if (adr2==1) bitun();  // Si bit à 1
if (adr2==0) bitzero(); // Si bit à 0
if (adr1==1) bitun();  // Si bit à 1
if (adr1==0) bitzero(); // Si bit à 0
if (adr0==1) bitun();  // Si bit à 1
if (adr0==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
// Bit à zéro de séparation
bitzero();
////////////////////////////////////
```

```
////////////////////////////////////
// Octet de commande
//La centrale transmet l'octet de commande
comm7 = 0;//28 pas de vitesse
comm6 = 1;//28 pas de vitesse

bitzero();//comm 7
bitun();//comm 6

if (comm5==1) bitun();  // Si bit à 1
if (comm5==0) bitzero(); // Si bit à 0
if (comm4==1) bitun();  // Si bit à 1
if (comm4==0) bitzero(); // Si bit à 0
if (comm3==1) bitun();  // Si bit à 1
if (comm3==0) bitzero(); // Si bit à 0
if (comm2==1) bitun();  // Si bit à 1
if (comm2==0) bitzero(); // Si bit à 0
if (comm1==1) bitun();  // Si bit à 1
if (comm1==0) bitzero(); // Si bit à 0
if (comm0==1) bitun();  // Si bit à 1
if (comm0==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
// Bit à zéro de séparation
bitzero();
////////////////////////////////////
////////////////////////////////////
// Envoi de l'octet de contrôle
//La centrale transmet l'octet de contrôle
//Calcul du OU Exclusif
cont0 = adr0 ^ comm0;  // ^ Calcul du OU EXCLUSIF bit 0 Octet de contrôle
```

```
cont1 = adr1 ^ comm1;  // ^ Calcul du OU EXCLUSIF bit 1 Octet de contrôle
cont2 = adr2 ^ comm2;  // ^ Calcul du OU EXCLUSIF bit 2 Octet de contrôle
cont3 = adr3 ^ comm3;  // ^ Calcul du OU EXCLUSIF bit 3 Octet de contrôle
cont4 = adr4 ^ comm4;  // ^ Calcul du OU EXCLUSIF bit 4 Octet de contrôle
cont5 = adr5 ^ comm5;  // ^ Calcul du OU EXCLUSIF bit 5 Octet de contrôle
cont6 = adr6 ^ comm6;  // ^ Calcul du OU EXCLUSIF bit 6 Octet de contrôle
cont7 = adr7 ^ comm7;  // ^ Calcul du OU EXCLUSIF bit 7 Octet de contrôle

if (cont7==1) bitun();  // Si bit à 1
if (cont7==0) bitzero(); // Si bit à 0
if (cont6==1) bitun();  // Si bit à 1
if (cont6==0) bitzero(); // Si bit à 0
if (cont5==1) bitun();  // Si bit à 1
if (cont5==0) bitzero(); // Si bit à 0
if (cont4==1) bitun();  // Si bit à 1
if (cont4==0) bitzero(); // Si bit à 0
if (cont3==1) bitun();  // Si bit à 1
if (cont3==0) bitzero(); // Si bit à 0
if (cont2==1) bitun();  // Si bit à 1
if (cont2==0) bitzero(); // Si bit à 0
if (cont1==1) bitun();  // Si bit à 1
if (cont1==0) bitzero(); // Si bit à 0
if (cont0==1) bitun();  // Si bit à 1
if (cont0==0) bitzero(); // Si bit à
////////////////////////////////////
////////////////////////////////////
// Bit à un de fin de transmission
bitun();
////////////////////////////////////
////////////////////////////////////
// Pause émission de 26 zéros (5ms) avant nouvelle transmission
```

```
for ( i=0; i<=25; i++)
bitzero();
////////////////////////////////////
////////////////////////////////////
}//Fin trame dcc locomotive
//////////////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////////////
////////////////////////////////////Génère la trame IDLE//////////////////////
//////////////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////////////
void trame_idle()//
{
  digitalWrite(ledrouge,LOW);//Lampe URG éteinte
  digitalWrite(ledbleue,LOW);
  digitalWrite(ledverte,LOW);
  digitalWrite(ledjaune,HIGH);

  //Bit de Synchronisation 16 bit à 1
// Octet de synchronisation
 for ( i=0; i <= 16; i++)
 {bitun(); }    // La centrale transmet 16 bits à 1
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
//bit zero
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de d'adresse 1
bitun();bitun();bitun();bitun();bitun();bitun();bitun();bitun();
////////////////////////////////////
```

```
////////////////////////////////////
//Bit de séparation
//bit zero
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de données 2
bitzero();bitzero();bitzero();bitzero();bitzero();bitzero();bitzero();bitzero();
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
//bit zero
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de données 3
bitun();bitun();bitun();bitun();bitun();bitun();bitun();bitun();
////////////////////////////////////
////////////////////////////////////
//Fin de transmission
//bit un
bitun();
////////////////////////////////////
////////////////////////////////////
// Pause émission de 25 zéros (5 ms)avant nouvelle transmission
for ( i=0; i<=25; i++)
bitzero();
}// Fin trame_idle()
////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
////////////////////////////////////GENERE La TRAME FONCTION//////////////////
```

```
////////////////////////////////////INTERRUPTEUR//////////////////////////
////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////
void envoi_trame_fonction()//envoi_trame_fonction()
{
digitalWrite(ledrouge,LOW);//Lampe URG éteinte
digitalWrite(ledbleue,HIGH);
digitalWrite(ledverte,LOW);
digitalWrite(ledjaune,LOW);

//Octet 2 de commande
//Lecture des bits et positionnement
   comm7 = 1;//Les 3 bits commandent les fonctions FL, F1, F2, F3, F4
   comm6 = 0;//Les 3 bits commandent les fonctions FL, F1, F2, F3, F4
   comm5 = 0;//Les 3 bits commandent les fonctions FL, F1, F2, F3, F4
   comm4 = bitRead (octetFonction, 4);
   comm3 = bitRead (octetFonction, 3);
   comm2 = bitRead (octetFonction, 2);
   comm1 = bitRead (octetFonction, 1);
   comm0 = bitRead (octetFonction, 0);
 ////////////////////////////////////////////
 ////////////////////////////////////////////
for(int compteur = 0; compteur < 5; compteur++)
{
  //Bit de Synchronisation 16 bit à 1
// Octet de synchronisation
 for ( i=0; i <= 20; i++)
 {bitun();}    // La centrale transmet 20 bits à 1
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
```

```
//bit zero
////////////////////////////////////////
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de données 1

if (adr7==1) bitun();  // Si bit à 1
if (adr7==0) bitzero(); // Si bit à 0

if (adr6==1) bitun();  // Si bit à 1
if (adr6==0) bitzero(); // Si bit à 0
if (adr5==1) bitun();  // Si bit à 1
if (adr5==0) bitzero(); // Si bit à 0
if (adr4==1) bitun();  // Si bit à 1
if (adr4==0) bitzero(); // Si bit à 0
if (adr3==1) bitun();  // Si bit à 1
if (adr3==0) bitzero(); // Si bit à 0
if (adr2==1) bitun();  // Si bit à 1
if (adr2==0) bitzero(); // Si bit à 0
if (adr1==1) bitun();  // Si bit à 1
if (adr1==0) bitzero(); // Si bit à 0
if (adr0==1) bitun();  // Si bit à 1
if (adr0==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
//bit zero
////////////////////////////////////
bitzero();
////////////////////////////////////
```

```
////////////////////////////////////
////////////////////////////////////
//Octet de données 2
if (comm7==1) bitun();  // Si bit à 1
if (comm7==0) bitzero(); // Si bit à 0
if (comm6==1) bitun();  // Si bit à 1
if (comm6==0) bitzero(); // Si bit à 0
if (comm5==1) bitun();  // Si bit à 1
if (comm5==0) bitzero(); // Si bit à 0
if (comm4==1) bitun();  // Si bit à 1
if (comm4==0) bitzero(); // Si bit à 0
if (comm3==1) bitun();  // Si bit à 1
if (comm3==0) bitzero(); // Si bit à 0
if (comm2==1) bitun();  // Si bit à 1
if (comm2==0) bitzero(); // Si bit à 0
if (comm1==1) bitun();  // Si bit à 1
if (comm1==0) bitzero(); // Si bit à 0
if (comm0==1) bitun();  // Si bit à 1
if (comm0==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
//bit zero
////////////////////////////////////
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de données 3
//Calcul du OU Exclusif
cont0 = adr0 ^ comm0;  // ^ Calcul du OU EXCLUSIF bit 0 Octet de contrôle
cont1 = adr1 ^ comm1;  // ^ Calcul du OU EXCLUSIF bit 1 Octet de contrôle
```

```
cont2 = adr2 ^ comm2;  // ^ Calcul du OU EXCLUSIF bit 2 Octet de contrôle
cont3 = adr3 ^ comm3;  // ^ Calcul du OU EXCLUSIF bit 3 Octet de contrôle
cont4 = adr4 ^ comm4;  // ^ Calcul du OU EXCLUSIF bit 4 Octet de contrôle
cont5 = adr5 ^ comm5;  // ^ Calcul du OU EXCLUSIF bit 5 Octet de contrôle
cont6 = adr6 ^ comm6;  // ^ Calcul du OU EXCLUSIF bit 6 Octet de contrôle
cont7 = adr7 ^ comm7;  // ^ Calcul du OU EXCLUSIF bit 7 Octet de contrôle
// Envoi de l'octet de contrôle
//La centrale transmet l'octet de contrôle
if (cont7==1) bitun();  // Si bit à 1
if (cont7==0) bitzero(); // Si bit à 0
if (cont6==1) bitun();  // Si bit à 1
if (cont6==0) bitzero(); // Si bit à 0
if (cont5==1) bitun();  // Si bit à 1
if (cont5==0) bitzero(); // Si bit à 0
if (cont4==1) bitun();  // Si bit à 1
if (cont4==0) bitzero(); // Si bit à 0
if (cont3==1) bitun();  // Si bit à 1
if (cont3==0) bitzero(); // Si bit à 0
if (cont2==1) bitun();  // Si bit à 1
if (cont2==0) bitzero(); // Si bit à 0
if (cont1==1) bitun();  // Si bit à 1
if (cont1==0) bitzero(); // Si bit à 0
if (cont0==1) bitun();  // Si bit à 1
if (cont0==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
//Fin de transmission
//bit un
//////////////////////////////////////////
bitun();
////////////////////////////////////
```

```
////////////////////////////////////
// Pause émission de 26 zéros avant nouvelle transmission
////////////////////////////////////////////
for ( i=0; i<=25; i++)
bitzero();
}//Fin for
}//Fin Fonction
//////////////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////////////
////////////////////////////////////Génère la trame URGENCE//////////////////
//////////////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////////////
// Génère la trame d'urgence (Emergency Stop, bit 5 ignoré)
void TrameArretUrg()
{
//Bit de Synchronisation 16 bit à 1
bitun();bitun();bitun();bitun();bitun();
bitun();bitun();bitun();bitun();bitun();
bitun();bitun();bitun();bitun();bitun();
bitun();bitun();bitun();bitun();bitun();
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
//bit zero
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de données 1
bitzero();bitzero();bitzero();bitzero();
bitzero();bitzero();bitzero();bitzero();
////////////////////////////////////
```

```
////////////////////////////////////
//Bit de séparation
//bit zero
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de données 2
bitzero();bitun();bitzero();bitun();
bitzero();bitzero();bitzero();bitzero();
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
//bit zero
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de données 3
bitzero();bitun();bitzero();bitun();
bitzero();bitzero();bitzero();bitzero();
////////////////////////////////////
////////////////////////////////////
//Fin de transmission
//bit un
bitun();
////////////////////////////////////
////////////////////////////////////
// Pause émission de 26 zéros (5ms) avant nouvelle transmission
 for ( i=0; i<=25; i++)
bitzero();
}// Fin Void TrameArretUrg()
////////////////////////////////////////////////////////////////////////////////
```

```
//////////////////////////// ////////////////////////////////////////////////////
/////////////////////// MISE EN MEMOIRE DES TOUCHES APPUYEES LOCOMOTIVE//////////
//////////////////////////////TOUCHE APPUYEE NUMERIQUE////////////////////////////
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
void touche_appuyee_numerique_cv() // lecture de la touche appuyée
{
compteur_touche_cv = compteur_touche_cv + 1;//Incrémente la variable compteur_touche

if (compteur_touche_cv == 1) //Teste si compteur_touche = 1
{toucheAcv = touche;}//Garde trace de la touche 1ere touche appuyée

if (compteur_touche_cv == 2) //Teste si compteur_touche = 2
{toucheBcv = touche;}//Garde trace de la touche 2ième touche appuyée

if (compteur_touche_cv == 3) //Teste si compteur_touche = 3
{toucheCcv = touche;}//Garde trace de la touche 3ième touche appuyée

}//Fin du VOID MISE EN MEMOIRE DES TOUCHES APPUYEES
////////////////////////////////////////////////////////////////////////////////
// ////////////////////////// TRAITEMENT DES TOUCHES APPUYEES PROGRAMMATION
CV//////////
////////////////////////////////TRAITEMENT APPUI TOUCHE////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
void traitement_appui_touche_cv()//Correspond à Appui touche 'E' et autorise_touche_cv
{

 if (compteur_touche_cv == 1)//Prise en compte  si 1 chiffre saisi
 {
    nbre_cv = (toucheAcv-48);//nbre contient l'intégralité de la saisie
```

```
   compteur_touche_cv = 0;//Remet le compteur à 0 pour prochaine saisie

   //Gestion des claviers
   autorise_touche_locomotive = 0;
   autorise_touche_fonction = 0;
   autorise_touche_Cv = 0;//Interdit la prise en compte de l'appui touche E et programmation cv à 0
   autorise_touche_Cv2 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv2 à
0
   autorise_touche_Cv4 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv4 à
0

   toucheAcv = 0;//Réinitialise toucheA
   toucheBcv = 0;//Réinitialise toucheB
   toucheCcv = 0;//Réinitialise toucheC

   adresse_cv = nbre_cv;//recopie dans adresse la valeur de nbre en binaire

   lcd.setCursor(9,2);
   lcd.print (adresse_cv);

   //Transforme la variable adresse en binaire et stocke dans les variables adr
   //Octet 1 d'adresse
   adr7cv = 0;//Adresse courte
   adr6cv = bitRead (adresse_cv, 6);
   adr5cv = bitRead (adresse_cv, 5);
   adr4cv = bitRead (adresse_cv, 4);
   adr3cv = bitRead (adresse_cv, 3);
   adr2cv = bitRead (adresse_cv, 2);
   adr1cv = bitRead (adresse_cv, 1);
   adr0cv = bitRead (adresse_cv, 0);

//Affiche le nombre adresse de départ en binaire
```

```
   lcd.setCursor(6,3);//Place le curseur colonne 6, ligne 4
   lcd.print (adr7cv);
   lcd.setCursor(7,3);//Place le curseur colonne 7, ligne 4
   lcd.print (adr6cv);
   lcd.setCursor(8,3);//Place le curseur colonne 8, ligne 4
   lcd.print (adr5cv);
   lcd.setCursor(9,3);//Place le curseur colonne 9, ligne 4
   lcd.print (adr4cv);
   lcd.setCursor(10,3);//Place le curseur colonne 10, ligne 4
   lcd.print (adr3cv);
   lcd.setCursor(11,3);//Place le curseur colonne 11, ligne 4
   lcd.print (adr2cv);
   lcd.setCursor(12,3);//Place le curseur colonne 12, ligne 4
   lcd.print (adr1cv);
   lcd.setCursor(13,3);//Place le curseur colonne 13, ligne 4
   lcd.print (adr0cv);

//delay (1000);
 lcd.clear();
lcd.setCursor (0,0);
lcd.print ("Adresse depart : ");
lcd.print(nbre_cv);

 ////Affiche le nombre adresse de départ en binaire 2ième affichage
   lcd.setCursor(6,1);//Place le curseur colonne 6, ligne 2
   lcd.print (adr7cv);
   lcd.setCursor(7,1);//Place le curseur colonne 7, ligne 2
   lcd.print (adr6cv);
   lcd.setCursor(8,1);//Place le curseur colonne 8, ligne 2
   lcd.print (adr5cv);
   lcd.setCursor(9,1);//Place le curseur colonne 9, ligne 2
```

```
   lcd.print (adr4cv);
   lcd.setCursor(10,1);//Place le curseur colonne 10, ligne 2
   lcd.print (adr3cv);
   lcd.setCursor(11,1);//Place le curseur colonne 11, ligne 2
   lcd.print (adr2cv);
   lcd.setCursor(12,1);//Place le curseur colonne 12, ligne 2
   lcd.print (adr1cv);
   lcd.setCursor(13,1);//Place le curseur colonne 13, ligne 2
   lcd.print (adr0cv);

  lcd.setCursor (0,2);
lcd.print ("Saisir donnee :");

lcd.setCursor (8,3);

//Gestion des claviers
autorise_touche_Cv2 = 1;//Autorise la prise en compte de l'appui touche + et autorise_touche_Cv2 à
1

}// fin du if = 1

if (compteur_touche_cv == 2)
{
  nbre_cv = ((toucheAcv-48) * 10) + (toucheBcv-48);//nbre contient l'intégralité de la saisie

   //Gestion des claviers
   autorise_touche_locomotive = 0;
   autorise_touche_fonction = 0;
   autorise_touche_Cv = 0;//Interdit la prise en compte de l'appui touche E et programmation cv à 0
   autorise_touche_Cv2 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv2 à
0
```

```
   autorise_touche_Cv4 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv4 à
0


   compteur_touche_cv = 0;//Remet le compteur à 0 pour prochaine saisie


   toucheAcv = 0;//Réinitialise toucheA
   toucheBcv = 0;//Réinitialise toucheB
   toucheCcv = 0;//Réinitialise toucheC


 adresse_cv = nbre_cv;//recopie dans adresse la valeur de nbre en binaire


   lcd.setCursor(9,2);
   lcd.print (adresse_cv);


   //Transforme la variable adresse en binaire et stocke dans les variables adr
   //Octet 1 d'adresse
   adr7cv = 0;//Adresse courte
   adr6cv = bitRead (adresse_cv, 6);
   adr5cv = bitRead (adresse_cv, 5);
   adr4cv = bitRead (adresse_cv, 4);
   adr3cv = bitRead (adresse_cv, 3);
   adr2cv = bitRead (adresse_cv, 2);
   adr1cv = bitRead (adresse_cv, 1);
   adr0cv = bitRead (adresse_cv, 0);


//Affiche le nombre adresse de départ en binaire
   lcd.setCursor(6,3);//Place le curseur colonne 6, ligne 4
   lcd.print (adr7cv);
   lcd.setCursor(7,3);//Place le curseur colonne 7, ligne 4
   lcd.print (adr6cv);
   lcd.setCursor(8,3);//Place le curseur colonne 8, ligne 4
```

```
    lcd.print (adr5cv);
    lcd.setCursor(9,3);//Place le curseur colonne 9, ligne 4
    lcd.print (adr4cv);
    lcd.setCursor(10,3);//Place le curseur colonne 10, ligne 4
    lcd.print (adr3cv);
    lcd.setCursor(11,3);//Place le curseur colonne 11, ligne 4
    lcd.print (adr2cv);
    lcd.setCursor(12,3);//Place le curseur colonne 12, ligne 4
    lcd.print (adr1cv);
    lcd.setCursor(13,3);//Place le curseur colonne 13, ligne 4
    lcd.print (adr0cv);

delay (1000);
 lcd.clear();
lcd.setCursor (0,0);
lcd.print ("Adresse depart : ");
lcd.print(nbre_cv);

    lcd.setCursor(6,1);//Place le curseur colonne 6, ligne 2
    lcd.print (adr7cv);
    lcd.setCursor(7,1);//Place le curseur colonne 7, ligne 2
    lcd.print (adr6cv);
    lcd.setCursor(8,1);//Place le curseur colonne 8, ligne 2
    lcd.print (adr5cv);
    lcd.setCursor(9,1);//Place le curseur colonne 9, ligne 2
    lcd.print (adr4cv);
    lcd.setCursor(10,1);//Place le curseur colonne 10, ligne 2
    lcd.print (adr3cv);
    lcd.setCursor(11,1);//Place le curseur colonne 11, ligne 2
    lcd.print (adr2cv);
    lcd.setCursor(12,1);//Place le curseur colonne 12, ligne 2
```

```
   lcd.print (adr1cv);
   lcd.setCursor(13,1);//Place le curseur colonne 13, ligne 2
   lcd.print (adr0cv);

  lcd.setCursor (0,2);
lcd.print ("Saisir la donnee :");

lcd.setCursor (8,3);

//Gestion des claviers
autorise_touche_Cv2 = 1;//Autorise la prise en compte de l'appui touche + et autorise_touche_Cv2 à
1

}//fin du if compteur touche = 2

if (compteur_touche_cv == 3)
{
  nbre_cv = ((toucheAcv-48) * 100) + ((toucheBcv-48)*10) + (toucheCcv-48);
  }

if (nbre_cv < 128)
   {
   //Gestion des claviers
   autorise_touche_locomotive = 0;
   autorise_touche_fonction = 0;
   autorise_touche_Cv = 0;//Interdit la prise en compte de l'appui touche E et programmation cv à 0
   autorise_touche_Cv2 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv2 à
0
   autorise_touche_Cv4 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv4 à
0

   compteur_touche_cv = 0;//Remet le compteur à 0 pour prochaine saisie
```

```
    toucheAcv = 0;//Réinitialise toucheA
    toucheBcv = 0;//Réinitialise toucheB
    toucheCcv = 0;//Réinitialise toucheC

 adresse_cv = nbre_cv;//recopie dans adresse la valeur de nbre en binaire

    lcd.setCursor(9,2);
    lcd.print (adresse_cv);

    //Transforme la variable adresse en binaire et stocke dans les variables adr
    //Octet 1 d'adresse
    adr7cv = 0;//Adresse courte
    adr6cv = bitRead (adresse_cv, 6);
    adr5cv = bitRead (adresse_cv, 5);
    adr4cv = bitRead (adresse_cv, 4);
    adr3cv = bitRead (adresse_cv, 3);
    adr2cv = bitRead (adresse_cv, 2);
    adr1cv = bitRead (adresse_cv, 1);
    adr0cv = bitRead (adresse_cv, 0);

//Affiche le nombre adresse de départ en binaire
    lcd.setCursor(6,3);//Place le curseur colonne 6, ligne 4
    lcd.print (adr7cv);
    lcd.setCursor(7,3);//Place le curseur colonne 7, ligne 4
    lcd.print (adr6cv);
    lcd.setCursor(8,3);//Place le curseur colonne 8, ligne 4
    lcd.print (adr5cv);
    lcd.setCursor(9,3);//Place le curseur colonne 9, ligne 4
    lcd.print (adr4cv);
    lcd.setCursor(10,3);//Place le curseur colonne 10, ligne 4
```

```
   lcd.print (adr3cv);
   lcd.setCursor(11,3);//Place le curseur colonne 11, ligne 4
   lcd.print (adr2cv);
   lcd.setCursor(12,3);//Place le curseur colonne 12, ligne 4
   lcd.print (adr1cv);
   lcd.setCursor(13,3);//Place le curseur colonne 13, ligne 4
   lcd.print (adr0cv);

delay (1000);
 lcd.clear();
lcd.setCursor (0,0);
lcd.print ("Adresse depart : ");
lcd.print(nbre_cv);

   lcd.setCursor(6,1);//Place le curseur colonne 6, ligne 2
   lcd.print (adr7cv);
   lcd.setCursor(7,1);//Place le curseur colonne 7, ligne 2
   lcd.print (adr6cv);
   lcd.setCursor(8,1);//Place le curseur colonne 8, ligne 2
   lcd.print (adr5cv);
   lcd.setCursor(9,1);//Place le curseur colonne 9, ligne 2
   lcd.print (adr4cv);
   lcd.setCursor(10,1);//Place le curseur colonne 10, ligne 2
   lcd.print (adr3cv);
   lcd.setCursor(11,1);//Place le curseur colonne 11, ligne 2
   lcd.print (adr2cv);
   lcd.setCursor(12,1);//Place le curseur colonne 12, ligne 2
   lcd.print (adr1cv);
   lcd.setCursor(13,1);//Place le curseur colonne 13, ligne 2
   lcd.print (adr0cv);
```

```
  lcd.setCursor (0,2);
lcd.print ("Saisir donnee");
lcd.setCursor (8,3);

//Gestion des claviers
autorise_touche_Cv2 = 1;//Autorise la prise en compte de l'appui touche + et autorise_touche_Cv2 à 1

   }//if (nbre_cv < 128)

}//fin du if compteur touche = 3

if (compteur_touche_cv > 3 || nbre_cv > 127)
{
   autorise_touche_locomotive = 0;
   autorise_touche_fonction = 0;
   autorise_touche_Cv = 0;//Interdit la prise en compte de l'appui touche E et programmation cv à 0
   autorise_touche_Cv2 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv2 à 0
   autorise_touche_Cv4 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv4 à 0

   compteur_touche_cv = 0;//Remet le compteur à 0 pour prochaine saisie

   toucheAcv = 0;//Réinitialise toucheA
   toucheBcv = 0;//Réinitialise toucheB
   toucheCcv = 0;//Réinitialise toucheC

   lcd.clear(); // Efface écran si appui
   lcd.setCursor(0,0);//Place le curseur colonne 0, ligne 1

   lcd.print("Taille Nbre Interdit") ; // Affiche la chaîne texte
```

```
   for(int compteurtempo = 0; compteurtempo < dureetempo; compteurtempo++)

   {
    trame_idle();//Envoi trame_idle
   }//fermeture du for

   retourdetrame = 1;//retour à, comme une pression sur la touche *

}//fin du if (compteur_touche_cv > 3 || nbre_cv > 127)

}//Fin du VOID TRAITEMENT DES TOUCHES APPUYEES
////////////////////////////////////////////////////////////////////////////////
//////////////////////////// ////////////////////////////////////////////////////
/////////////////////// MISE EN MEMOIRE DES TOUCHES APPUYEES LOCOMOTIVE//////////
////////////////////////////TOUCHE APPUYEE NUMERIQUE 2//////////////////////////
////////////////////////////////////////////////////////////////////////////////
void touche_appuyee_numerique_cv2() // // lecture de la touche appuyée
{
 compteur_touche_cv2 = compteur_touche_cv2 + 1;//Incrémente la variable compteur_touche

if (compteur_touche_cv2 == 1) //Teste si compteur_touche = 1
{toucheAcv2 = touche;}//Garde trace de la touche 1ere touche appuyée

if (compteur_touche_cv2 == 2) //Teste si compteur_touche = 2
{toucheBcv2 = touche;}//Garde trace de la touche 2ième touche appuyée

if (compteur_touche_cv2 == 3) //Teste si compteur_touche = 3
{toucheCcv2 = touche;}//Garde trace de la touche 3ième touche appuyée

}//Fin du VOID MISE EN MEMOIRE DES TOUCHES APPUYEES
////////////////////////////////////////////////////////////////////////////////
```

```
// ////////////////////////// TRAITEMENT DES TOUCHES APPUYEES PROGRAMMATION CV2//////////
////////////////////////////////TRAITEMENT APPUI TOUCHE////////////////////////
//////////////////////////////////////////////////////////////////////////////
void traitement_appui_touche_cv2()//Correspond à Appui touche 'E' et programmationcv = 1
{
 if (compteur_touche_cv2 == 1)//Prise en compte  si 1 chiffre saisi

{
   nbre_cv2 = (toucheAcv2-48);//nbre contient l'intégralité de la saisie

   compteur_touche_cv2 = 0;//Remet le compteur à 0 pour prochaine saisie

   //Gestion des claviers
   autorise_touche_locomotive = 0;
   autorise_touche_fonction = 0;
   autorise_touche_Cv = 0;//Interdit la prise en compte de l'appui touche E et programmation cv à 0
   autorise_touche_Cv2 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv2 à
0
   autorise_touche_Cv4 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv4 à
0

   toucheAcv2 = 0;//Réinitialise toucheA
   toucheBcv2 = 0;//Réinitialise toucheB
   toucheCcv2 = 0;//Réinitialise toucheC

   adresse_cv2 = nbre_cv2;//recopie dans adresse la valeur de nbre en binaire

  lcd.clear();
lcd.setCursor (0,0);
lcd.print ("Adresse depart : ");
lcd.print(nbre_cv);//affiche la variable nbre_cv. Chiffre saisi pour l'adresse de départ
```

```
//Affiche le nombre adresse de départ en binaire
   lcd.setCursor(6,1);//Place le curseur colonne 6, ligne 2
   lcd.print (adr7cv);
   lcd.setCursor(7,1);//Place le curseur colonne 7, ligne 2
   lcd.print (adr6cv);
   lcd.setCursor(8,1);//Place le curseur colonne 8, ligne 2
   lcd.print (adr5cv);
   lcd.setCursor(9,1);//Place le curseur colonne 9, ligne 2
   lcd.print (adr4cv);
   lcd.setCursor(10,1);//Place le curseur colonne 10, ligne 2
   lcd.print (adr3cv);
   lcd.setCursor(11,1);//Place le curseur colonne 11, ligne 2
   lcd.print (adr2cv);
   lcd.setCursor(12,1);//Place le curseur colonne 12, ligne 2
   lcd.print (adr1cv);
   lcd.setCursor(13,1);//Place le curseur colonne 13, ligne 2
   lcd.print (adr0cv);

lcd.setCursor (0,2);
lcd.print ("Donnee saisie :");
lcd.print(nbre_cv2);

//Transforme la variable adresse en binaire et stocke dans les variables adr
   //Octet 1 d'adresse
   adr7cv2 = 0;//Adresse courte
   adr6cv2 = bitRead (adresse_cv2, 6);
   adr5cv2 = bitRead (adresse_cv2, 5);
   adr4cv2 = bitRead (adresse_cv2, 4);
   adr3cv2 = bitRead (adresse_cv2, 3);
   adr2cv2 = bitRead (adresse_cv2, 2);
   adr1cv2 = bitRead (adresse_cv2, 1);
```

```
   adr0cv2 = bitRead (adresse_cv2, 0);

   lcd.setCursor(6,3);//Place le curseur colonne 6, ligne 4
   lcd.print (adr7cv2);
   lcd.setCursor(7,3);//Place le curseur colonne 7, ligne 4
   lcd.print (adr6cv2);
   lcd.setCursor(8,3);//Place le curseur colonne 8, ligne 4
   lcd.print (adr5cv2);
   lcd.setCursor(9,3);//Place le curseur colonne 9, ligne 4
   lcd.print (adr4cv2);
   lcd.setCursor(10,3);//Place le curseur colonne 10, ligne 4
   lcd.print (adr3cv2);
   lcd.setCursor(11,3);//Place le curseur colonne 11, ligne 4
   lcd.print (adr2cv2);
   lcd.setCursor(12,3);//Place le curseur colonne 12, ligne 4
   lcd.print (adr1cv2);
   lcd.setCursor(13,3);//Place le curseur colonne 13, ligne 4
   lcd.print (adr0cv2);

delay(1500);
lcd.clear();
lcd.setCursor (2,0);
lcd.print ("Numero de la Cv : ");
lcd.setCursor (9,1);

//Gestion des claviers
autorise_touche_Cv4 = 1;

}// fin du if (compteur_touche_cv2 == 1)

if (compteur_touche_cv2 == 2)
```

```
{
  nbre_cv2 = ((toucheAcv2-48) * 10) + (toucheBcv2-48);//nbre contient l'intégralité de la saisie

   //Gestion des claviers
   autorise_touche_locomotive = 0;
   autorise_touche_fonction = 0;
   autorise_touche_Cv = 0;//Interdit la prise en compte de l'appui touche E et programmation cv à 0
   autorise_touche_Cv2 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv2 à
0
   autorise_touche_Cv4 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv4 à
0

   compteur_touche_cv2 = 0;//Remet le compteur à 0 pour prochaine saisie

   toucheAcv2 = 0;//Réinitialise toucheA
   toucheBcv2 = 0;//Réinitialise toucheB
   toucheCcv2 = 0;//Réinitialise toucheC

 adresse_cv2 = nbre_cv2;//recopie dans adresse la valeur de nbre en binaire

  lcd.clear();
lcd.setCursor (0,0);
lcd.print ("Adresse depart : ");
lcd.print(nbre_cv);//affiche la variable nbre_cv. Chiffre saisi pour l'adresse de départ

//Affiche le nombre adresse de départ en binaire
   lcd.setCursor(6,1);//Place le curseur colonne 6, ligne 2
   lcd.print (adr7cv);
   lcd.setCursor(7,1);//Place le curseur colonne 7, ligne 2
   lcd.print (adr6cv);
   lcd.setCursor(8,1);//Place le curseur colonne 8, ligne 2
   lcd.print (adr5cv);
```

```
   lcd.setCursor(9,1);//Place le curseur colonne 9, ligne 2
   lcd.print (adr4cv);
   lcd.setCursor(10,1);//Place le curseur colonne 10, ligne 2
   lcd.print (adr3cv);
   lcd.setCursor(11,1);//Place le curseur colonne 11, ligne 2
   lcd.print (adr2cv);
   lcd.setCursor(12,1);//Place le curseur colonne 12, ligne 2
   lcd.print (adr1cv);
   lcd.setCursor(13,1);//Place le curseur colonne 13, ligne 2
   lcd.print (adr0cv);

lcd.setCursor (0,2);
lcd.print ("Donnee saisie :");
lcd.print(nbre_cv2);

//Transforme la variable adresse en binaire et stocke dans les variables adr
   //Octet 1 d'adresse
   adr7cv2 = 0;//Adresse courte
   adr6cv2 = bitRead (adresse_cv2, 6);
   adr5cv2 = bitRead (adresse_cv2, 5);
   adr4cv2 = bitRead (adresse_cv2, 4);
   adr3cv2 = bitRead (adresse_cv2, 3);
   adr2cv2 = bitRead (adresse_cv2, 2);
   adr1cv2 = bitRead (adresse_cv2, 1);
   adr0cv2 = bitRead (adresse_cv2, 0);

   lcd.setCursor(6,3);//Place le curseur colonne 6, ligne 4
   lcd.print (adr7cv2);
   lcd.setCursor(7,3);//Place le curseur colonne 7, ligne 4
   lcd.print (adr6cv2);
   lcd.setCursor(8,3);//Place le curseur colonne 8, ligne 4
```

```
   lcd.print (adr5cv2);
   lcd.setCursor(9,3);//Place le curseur colonne 9, ligne 4
   lcd.print (adr4cv2);
   lcd.setCursor(10,3);//Place le curseur colonne 10, ligne 4
   lcd.print (adr3cv2);
   lcd.setCursor(11,3);//Place le curseur colonne 11, ligne 4
   lcd.print (adr2cv2);
   lcd.setCursor(12,3);//Place le curseur colonne 12, ligne 4
   lcd.print (adr1cv2);
   lcd.setCursor(13,3);//Place le curseur colonne 13, ligne 4
   lcd.print (adr0cv2);

delay(1500);
lcd.clear();
lcd.setCursor (2,0);
lcd.print ("Numero de la Cv : ");
lcd.setCursor (9,1);

//Gestion des claviers
autorise_touche_Cv4 = 1;

}//fin du if (compteur_touche_cv2 == 2)

if (compteur_touche_cv2 == 3)
{
  nbre_cv2 = ((toucheAcv2-48) * 100) + ((toucheBcv2-48)*10) + (toucheCcv2-48);

if (nbre_cv2 < 256)
   {
   //Gestion des claviers
   autorise_touche_locomotive = 0;
```

```
   autorise_touche_fonction = 0;
   autorise_touche_Cv = 0;//Interdit la prise en compte de l'appui touche E et programmation cv à 0
   autorise_touche_Cv2 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv2 à
0
   autorise_touche_Cv4 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv4 à
0

   compteur_touche_cv2 = 0;//Remet le compteur à 0 pour prochaine saisie

   toucheAcv2 = 0;//Réinitialise toucheA
   toucheBcv2 = 0;//Réinitialise toucheB
   toucheCcv2 = 0;//Réinitialise toucheC

 adresse_cv2 = nbre_cv2;//recopie dans adresse la valeur de nbre en binaire

  lcd.clear();
lcd.setCursor (0,0);
lcd.print ("Adresse depart : ");
lcd.print(nbre_cv);//affiche la variable nbre_cv.

//Affiche le nombre adresse de départ en binaire
   lcd.setCursor(6,1);//Place le curseur colonne 6, ligne 2
   lcd.print (adr7cv);
   lcd.setCursor(7,1);//Place le curseur colonne 7, ligne 2
   lcd.print (adr6cv);
   lcd.setCursor(8,1);//Place le curseur colonne 8, ligne 2
   lcd.print (adr5cv);
   lcd.setCursor(9,1);//Place le curseur colonne 9, ligne 2
   lcd.print (adr4cv);
   lcd.setCursor(10,1);//Place le curseur colonne 10, ligne 2
   lcd.print (adr3cv);
   lcd.setCursor(11,1);//Place le curseur colonne 11, ligne 2
```

```
   lcd.print (adr2cv);
   lcd.setCursor(12,1);//Place le curseur colonne 12, ligne 2
   lcd.print (adr1cv);
   lcd.setCursor(13,1);//Place le curseur colonne 13, ligne 2
   lcd.print (adr0cv);

lcd.setCursor (0,2);
lcd.print ("Donnee saisie :");
lcd.print(nbre_cv2);

//Transforme la variable adresse en binaire et stocke dans les variables adr
   //Octet 1 d'adresse
   adr7cv2 = 0;//Adresse courte
   adr6cv2 = bitRead (adresse_cv2, 6);
   adr5cv2 = bitRead (adresse_cv2, 5);
   adr4cv2 = bitRead (adresse_cv2, 4);
   adr3cv2 = bitRead (adresse_cv2, 3);
   adr2cv2 = bitRead (adresse_cv2, 2);
   adr1cv2 = bitRead (adresse_cv2, 1);
   adr0cv2 = bitRead (adresse_cv2, 0);

   lcd.setCursor(6,3);//Place le curseur colonne 6, ligne 4
   lcd.print (adr7cv2);
   lcd.setCursor(7,3);//Place le curseur colonne 7, ligne 4
   lcd.print (adr6cv2);
   lcd.setCursor(8,3);//Place le curseur colonne 8, ligne 4
   lcd.print (adr5cv2);
   lcd.setCursor(9,3);//Place le curseur colonne 9, ligne 4
   lcd.print (adr4cv2);
   lcd.setCursor(10,3);//Place le curseur colonne 10, ligne 4
   lcd.print (adr3cv2);
```

```
   lcd.setCursor(11,3);//Place le curseur colonne 11, ligne 4
   lcd.print (adr2cv2);
   lcd.setCursor(12,3);//Place le curseur colonne 12, ligne 4
   lcd.print (adr1cv2);
   lcd.setCursor(13,3);//Place le curseur colonne 13, ligne 4
   lcd.print (adr0cv2);

delay(1500);
lcd.clear();
lcd.setCursor (2,0);
lcd.print ("Numero de la Cv : ");
lcd.setCursor (9,1);

//Gestion des claviers
autorise_touche_Cv4 = 1;

}//fin du if (compteur_touche_cv2 == 3)

if (compteur_touche_cv2 > 3 || nbre_cv2 > 255)
{
   compteur_touche_cv2 = 0; //Remet le compteur à 0 pour prochaine saisie

   //Gestion des claviers
   autorise_touche_locomotive = 0;
   autorise_touche_fonction = 0;
   autorise_touche_Cv = 0;
   autorise_touche_Cv2 = 0;
   autorise_touche_Cv4 = 0;

   toucheAcv2 = 0;//Réinitialise toucheA
   toucheBcv2 = 0;//Réinitialise toucheB
```

```
   toucheCcv2 = 0;//Réinitialise toucheC

   lcd.clear(); // Efface écran si appui
   lcd.setCursor(0,0);//Place le curseur colonne 0, ligne 1

   lcd.print("Taille Nbre Interdit") ; // Affiche la chaîne texte
   for(int compteurtempo = 0; compteurtempo < dureetempo; compteurtempo++)
   {
    trame_idle();//Envoi trame_idle
   }//fermeture du for

   retourdetrame = 1;//retour à, comme une pression sur la touche *

   }//(compteur_touche_cv2 > 3 || nbre_cv2 > 255

}//fin if (compteur_touche_cv2 == 3)

}//Fin du VOID TRAITEMENT DES TOUCHES APPUYEES CV2
////////////////////////////////////////////////////////////////////////////////
//////////////////////////// ////////////////////////////////////////////////////
////////////////////////// MISE EN MEMOIRE DES TOUCHES APPUYEES LOCOMOTIVE//////////
//////////////////////////////TOUCHE APPUYEE NUMERIQUE CV4////////////////////////////
////////////////////////////////////////////////////////////////////////////////
void touche_appuyee_numerique_cv4() // // lecture de la touche appuyée
{
  compteur_touche_cv4 = compteur_touche_cv4 + 1;

if (compteur_touche_cv4 == 1) //Teste si compteur_touche = 1
{toucheAcv4 = touche;}//Garde trace de la touche 1ere touche appuyée

if (compteur_touche_cv4 == 2) //Teste si compteur_touche = 2
```

```
{toucheBcv4 = touche;}//Garde trace de la touche 2ième touche appuyée

if (compteur_touche_cv4 == 3) //Teste si compteur_touche = 3
{toucheCcv4 = touche;}//Garde trace de la touche 3ième touche appuyée

}//Fin du VOID TOUCHE APPUYEE NUMERIQUE CV4
////////////////////////////////////////////////////////////////////////////
// ////////////////////////// TRAITEMENT DES TOUCHES APPUYEES PROGRAMMATION CV2///
//////////////////////////////////TRAITEMENT APPUI TOUCHE//////////////////////
////////////////////////////////////////////////////////////////////////////
void traitement_appui_touche_cv4()//Correspond à Appui touche 'E' et autorise_touche_cv4 = 1
{
 if (compteur_touche_cv4 == 1)//Prise en compte  si 1 chiffre saisi

{
   nbre_cv4 = (toucheAcv4-48);//nbre contient l'intégralité de la saisie

   compteur_touche_cv3 = 0;//Remet le compteur à 0 pour prochaine saisie

   //Gestion des claviers
   autorise_touche_locomotive = 0;
   autorise_touche_fonction = 0;
   autorise_touche_Cv = 0;//Interdit la prise en compte de l'appui touche E et programmation cv à 0
   autorise_touche_Cv2 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv2 à
0
   autorise_touche_Cv4 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv4 à
0

   toucheAcv4 = 0;//Réinitialise toucheA
   toucheBcv4 = 0;//Réinitialise toucheB
   toucheCcv4 = 0;//Réinitialise toucheC
```

```
   adresse_cv4 = nbre_cv4-1;//recopie dans adresse la valeur de nbre en binaire


 //Transforme la variable adresse en binaire et stocke dans les variables adr
   //Octet 1 d'adresse
   adr7cv4 = bitRead (adresse_cv4, 7);
   adr6cv4 = bitRead (adresse_cv4, 6);
   adr5cv4 = bitRead (adresse_cv4, 5);
   adr4cv4 = bitRead (adresse_cv4, 4);
   adr3cv4 = bitRead (adresse_cv4, 3);
   adr2cv4 = bitRead (adresse_cv4, 2);
   adr1cv4 = bitRead (adresse_cv4, 1);
   adr0cv4 = bitRead (adresse_cv4, 0);


//Affiche le nombre adresse de départ en binaire
   lcd.setCursor(6,2);//Place le curseur colonne 6, ligne 3
   lcd.print (adr7cv4);
   lcd.setCursor(7,2);//Place le curseur colonne 7, ligne 3
   lcd.print (adr6cv4);
   lcd.setCursor(8,2);//Place le curseur colonne 8, ligne 3
   lcd.print (adr5cv4);
   lcd.setCursor(9,2);//Place le curseur colonne 9, ligne 3
   lcd.print (adr4cv4);
   lcd.setCursor(10,2);//Place le curseur colonne 10, ligne 3
   lcd.print (adr3cv4);
   lcd.setCursor(11,2);//Place le curseur colonne 11, ligne 3
   lcd.print (adr2cv4);
   lcd.setCursor(12,2);//Place le curseur colonne 12, ligne 3
   lcd.print (adr1cv4);
   lcd.setCursor(13,2);//Place le curseur colonne 13, ligne 3
   lcd.print (adr0cv4);
```

```
delay(1500);

lcd.clear();
lcd.setCursor (4,1);
lcd.print ("Programmation");
lcd.setCursor (6,2);
lcd.print ("en cours");
delay(1000);

envoi_trame_programmation_cv();

}// fin du if (compteur_touche_cv4 == 1)

if (compteur_touche_cv4 == 2)
{
  nbre_cv4 = ((toucheAcv4-48) * 10) + (toucheBcv4-48);

   //Gestion des claviers
   autorise_touche_locomotive = 0;
   autorise_touche_fonction = 0;
   autorise_touche_Cv = 0;//Interdit la prise en compte de l'appui touche E et programmation cv à 0
   autorise_touche_Cv2 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv2 à
0
   autorise_touche_Cv4 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv4 à
0

   compteur_touche_cv3 = 0;//Remet le compteur à 0 pour prochaine saisie

   toucheAcv4 = 0;//Réinitialise toucheA
   toucheBcv4 = 0;//Réinitialise toucheB
   toucheCcv4 = 0;//Réinitialise toucheC
```

```
 adresse_cv4 = nbre_cv4-1;//recopie dans adresse la valeur de nbre en binaire


 //Transforme la variable adresse en binaire et stocke dans les variables adr
   //Octet 1 d'adresse
   adr7cv4 = bitRead (adresse_cv4, 7);
   adr6cv4 = bitRead (adresse_cv4, 6);
   adr5cv4 = bitRead (adresse_cv4, 5);
   adr4cv4 = bitRead (adresse_cv4, 4);
   adr3cv4 = bitRead (adresse_cv4, 3);
   adr2cv4 = bitRead (adresse_cv4, 2);
   adr1cv4 = bitRead (adresse_cv4, 1);
   adr0cv4 = bitRead (adresse_cv4, 0);


//Affiche le nombre adresse de départ en binaire
   lcd.setCursor(6,2);//Place le curseur colonne 6, ligne 3
   lcd.print (adr7cv4);
   lcd.setCursor(7,2);//Place le curseur colonne 7, ligne 3
   lcd.print (adr6cv4);
   lcd.setCursor(8,2);//Place le curseur colonne 8, ligne 3
   lcd.print (adr5cv4);
   lcd.setCursor(9,2);//Place le curseur colonne 9, ligne 3
   lcd.print (adr4cv4);
   lcd.setCursor(10,2);//Place le curseur colonne 10, ligne 3
   lcd.print (adr3cv4);
   lcd.setCursor(11,2);//Place le curseur colonne 11, ligne 3
   lcd.print (adr2cv4);
   lcd.setCursor(12,2);//Place le curseur colonne 12, ligne 3
   lcd.print (adr1cv4);
   lcd.setCursor(13,2);//Place le curseur colonne 13, ligne 3
   lcd.print (adr0cv4);
```

```
delay(1500);
lcd.clear();
lcd.setCursor (4,1);
lcd.print ("Programmation");
lcd.setCursor (6,2);
lcd.print ("en cours");
delay(1000);

envoi_trame_programmation_cv();

}//fin du if (compteur_touche_cv4 == 2)

if (compteur_touche_cv4 == 3)
{
  nbre_cv4 = ((toucheAcv4-48) * 100) + ((toucheBcv4-48)*10) + (toucheCcv4-48);

if (nbre_cv4 < 257)
   {
   //Gestion des claviers
   autorise_touche_locomotive = 0;
   autorise_touche_fonction = 0;
   autorise_touche_Cv = 0;//Interdit la prise en compte de l'appui touche E et programmation cv à 0
   autorise_touche_Cv2 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv2 à
0
   autorise_touche_Cv4 = 0;//Interdit la prise en compte de l'appui touche E et programmation cv4 à
0

   compteur_touche_cv3 = 0;//Remet le compteur à 0 pour prochaine saisie

   toucheAcv4 = 0;//Réinitialise toucheA
   toucheBcv4 = 0;//Réinitialise toucheB
   toucheCcv4 = 0;//Réinitialise toucheC
```

```
  adresse_cv4 = nbre_cv4-1;//recopie dans adresse la valeur de nbre en binaire

 //Transforme la variable adresse en binaire et stocke dans les variables adr
  //Octet 1 d'adresse
  adr7cv4 = bitRead (adresse_cv4, 7);
  adr6cv4 = bitRead (adresse_cv4, 6);
  adr5cv4 = bitRead (adresse_cv4, 5);
  adr4cv4 = bitRead (adresse_cv4, 4);
  adr3cv4 = bitRead (adresse_cv4, 3);
  adr2cv4 = bitRead (adresse_cv4, 2);
  adr1cv4 = bitRead (adresse_cv4, 1);
  adr0cv4 = bitRead (adresse_cv4, 0);

//Affiche le nombre adresse de départ en binaire
  lcd.setCursor(6,2);//Place le curseur colonne 6, ligne 3
  lcd.print (adr7cv4);
  lcd.setCursor(7,2);//Place le curseur colonne 7, ligne 3
  lcd.print (adr6cv4);
  lcd.setCursor(8,2);//Place le curseur colonne 8, ligne 3
  lcd.print (adr5cv4);
  lcd.setCursor(9,2);//Place le curseur colonne 9, ligne 3
  lcd.print (adr4cv4);
  lcd.setCursor(10,2);//Place le curseur colonne 10, ligne 3
  lcd.print (adr3cv4);
  lcd.setCursor(11,2);//Place le curseur colonne 11, ligne 3
  lcd.print (adr2cv4);
  lcd.setCursor(12,2);//Place le curseur colonne 12, ligne 3
  lcd.print (adr1cv4);
  lcd.setCursor(13,2);//Place le curseur colonne 13, ligne 3
  lcd.print (adr0cv4);
```

```
delay(1500);
lcd.clear();
lcd.setCursor (4,1);
lcd.print ("Programmation");
lcd.setCursor (6,2);
lcd.print ("en cours");
delay(1000);

envoi_trame_programmation_cv();

}//fin du if (compteur_touche_cv4 == 3)
}
if (compteur_touche_cv4 > 3 || nbre_cv4 > 256)
{
   compteur_touche_cv2 = 0; //Remet le compteur à 0 pour prochaine saisie

   //Gestion des claviers
   autorise_touche_locomotive =0;
   autorise_touche_Cv = 0;
   autorise_touche_Cv2 = 0;
   autorise_touche_Cv3 = 0;
   autorise_touche_Cv4 = 0;

   toucheAcv4 = 0;//Réinitialise toucheA
   toucheBcv4 = 0;//Réinitialise toucheB
   toucheCcv4 = 0;//Réinitialise toucheC

   lcd.clear(); // Efface écran si appui
   lcd.setCursor(0,0);//Place le curseur colonne 0, ligne 1
```

```
   lcd.print("Taille Nbre Interdit") ; // Affiche la chaîne texte
   for(int compteurtempo = 0; compteurtempo < dureetempo; compteurtempo++)
   {
    trame_idle();//Envoi trame_idle
   }//fermeture du for
   retourdetrame = 1;//retour à, comme une pression sur la touche *

}//fin if (compteur_touche_cv4 > 3 || nbre_cv4 > 256)

}//Fin du VOID TRAITEMENT DES TOUCHES APPUYEES
////////////////////////////////////////////////////////////////////////////////
//////////////////////////// ////////////////////////////////////////////////////
////////////////////////// MISE EN MEMOIRE DES TOUCHES APPUYEES LOCOMOTIVE//////////////
//////////////////////////////TOUCHE APPUYEE NUMERIQUE FONCTION//////////////////////
////////////////////////////////////////////////////////////////////////////////
void touche_appuyee_numerique_fonction() // // lecture de la touche appuyée
{
 compteur_touche_fonction = compteur_touche_fonction + 1;

if (compteur_touche_fonction == 1) //Teste si compteur_touche = 1
{toucheAfonction = touche;}//Garde trace de la touche 1ere touche appuyée

if (compteur_touche_fonction == 2) //Teste si compteur_touche = 2
{toucheBfonction = touche;}//Garde trace de la touche 2ième touche appuyée

if (compteur_touche_fonction == 3) //Teste si compteur_touche = 3
{toucheCfonction = touche;}//Garde trace de la touche 3ième touche appuyée

if (compteur_touche_fonction == 4) //Teste si compteur_touche = 4
{toucheDfonction = touche;}//Garde trace de la touche 4ième touche appuyée
```

```
if (compteur_touche_fonction == 5) //Teste si compteur_touche = 5
{toucheEfonction = touche;}//Garde trace de la touche 5ième touche appuyée

}//Fin du VOID MISE EN MEMOIRE DES TOUCHES APPUYEES
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
//////////////void traitement_appui_touche_fonction()/////////////////
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
void traitement_appui_touche_fonction()
{
if (compteur_touche_fonction == 1)//Prise en compte du chiffre saisi pour une entrée
{
nbre_fonction = (toucheAfonction-48);//nbre contient l'intégralité de la saisie



if (nbre_fonction == 0)
{
  octetFonction = bitClear (octetFonction, 4);//Désactive la fonction
  trame_fonction_clavier_100();
}
//////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 1)
{
  octetFonction = bitSet (octetFonction, 4);//Désactive la fonction
   trame_fonction_clavier_100();
}
}//compteur_touche_fonction == 1
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = FL/////////////////////////////////////
```

```
//////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////
////////////////FONCTION REMISE à ZEROS TTES FONCTIONS/////////////////
//////////////////////sf FL, F1, F2, F3, F4//////////////////////////
//////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////

if (compteur_touche_fonction == 1)//Prise en compte du chiffre saisi pour une entrée
{
nbre_fonction = (toucheAfonction-48);//nbre contient l'intégralité de la saisie



if (nbre_fonction == 9)
{

  trame_fonction_raz();
}
} //compteur_touche_fonction == 1 et nbre_fonction == 9

//////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////
////////////////FONCTION REMISE à ZEROS TTES FONCTIONS/////////////////
//////////////////////sf FL, F1, F2, F3, F4//////////////////////////
//////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////
////////////////////// FONCTION = F1/////////////////////////////////
//////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////
```

```
if (compteur_touche_fonction == 2)//Prise en compte du chiffre saisi pour une entrée
{
nbre_fonction = ((toucheAfonction-48) * 10) + (toucheBfonction-48);

if (nbre_fonction == 10)
{
  octetFonction = bitClear (octetFonction, 0);//Désactive la fonction
  trame_fonction_clavier_100();
}
////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 11)
{
  octetFonction = bitSet (octetFonction, 0);//Désactive la fonction
   trame_fonction_clavier_100();
 }
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F1///////////////////////////////////
//////////////////////////////////////////////////////////////////////
////////////////////FONCTION = F2///////////////////////////////////
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
if (nbre_fonction == 20)
{
  octetFonction = bitClear (octetFonction, 1);//Désactive la fonction
  trame_fonction_clavier_100();
}
////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 21)
{
  octetFonction = bitSet (octetFonction, 1);//Désactive la fonction
```

```
   trame_fonction_clavier_100();
 }
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F2//////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////FONCTION = F3//////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
if (nbre_fonction == 30)
{
  octetFonction = bitClear (octetFonction, 2);//Désactive la fonction
  trame_fonction_clavier_100();
}
////////////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 31)
{
  octetFonction = bitSet (octetFonction, 2);//Désactive la fonction
   trame_fonction_clavier_100();
}
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F3//////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////FONCTION = F4//////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
if (nbre_fonction == 40)
```

```
{
  octetFonction = bitClear (octetFonction, 3);//Désactive la fonction
  trame_fonction_clavier_100();
}
 ////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 41)
{
  octetFonction = bitSet (octetFonction, 3);//Désactive la fonction
   trame_fonction_clavier_100();
 }
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F4/////////////////////////////////////
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
////////////////////FONCTION = F5/////////////////////////////////////
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
if (nbre_fonction == 50)
{
  octetFonction = bitClear (octetFonction1011, 0);//Désactive la fonction
  trame_fonction_clavier_1011();
}
////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 51)
{
  octetFonction = bitSet (octetFonction1011, 0);//Désactive la fonction
   trame_fonction_clavier_1011();
}
//////////////////////////////////////////////////////////////////////
```

```
////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F5/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////FONCTION = F6/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
if (nbre_fonction == 60)
{
  octetFonction = bitClear (octetFonction1011, 1);//Désactive la fonction
  trame_fonction_clavier_1011();
}
////////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 61)
{
  octetFonction = bitSet (octetFonction1011, 1);//Désactive la fonction
   trame_fonction_clavier_1011();
 }
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F6/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////FONCTION = F7/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
if (nbre_fonction == 70)
{
  octetFonction = bitClear (octetFonction1011, 2);//Désactive la fonction
  trame_fonction_clavier_1011();
}
```

```
////////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 71)
{
  octetFonction = bitSet (octetFonction1011, 2);//Désactive la fonction
   trame_fonction_clavier_1011();
}
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F7/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////FONCTION = F8/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
if (nbre_fonction == 80)
{
  octetFonction = bitClear (octetFonction1011, 3);//Désactive la fonction
  trame_fonction_clavier_1011();
}
////////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 81)
{
  octetFonction = bitSet (octetFonction1011, 3);//Désactive la fonction
   trame_fonction_clavier_1011();
 }
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F8/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////FONCTION = F9/////////////////////////////////
```

```
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
if (nbre_fonction == 90)
{
  octetFonction = bitClear (octetFonction1010, 0);//Désactive la fonction
  trame_fonction_clavier_1010();
}
////////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 91)
{
  octetFonction = bitSet (octetFonction1010, 0);//Désactive la fonction
   trame_fonction_clavier_1010();
 }
}//fin du if 2
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F9//////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////////FONCTION = F10//////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
if (compteur_touche_fonction == 3)//Prise en compte du chiffre saisi pour une entrée
{
nbre_fonction = ((toucheAfonction-48) * 100) + ((toucheBfonction-48)*10) + (toucheCfonction-48);
if (nbre_fonction == 100)
{
  octetFonction = bitClear (octetFonction1010, 1);//Désactive la fonction
  trame_fonction_clavier_1010();
}
```

```
////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 101)
{
  octetFonction = bitSet (octetFonction1010, 1);//Désactive la fonction
   trame_fonction_clavier_1010();
 }
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F10/////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////////FONCTION = F11/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
if (nbre_fonction == 110)
{
  octetFonction = bitClear (octetFonction1010, 2);//Désactive la fonction
  trame_fonction_clavier_1010();
}
////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 111)
{
  octetFonction = bitSet (octetFonction1010, 2);//Désactive la fonction
   trame_fonction_clavier_1010();
 }
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F11/////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////////FONCTION = F12/////////////////////////////////
////////////////////////////////////////////////////////////////////
```

```
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
if (nbre_fonction == 120)
{
  octetFonction = bitClear (octetFonction1010, 3);//Désactive la fonction
  trame_fonction_clavier_1010();
}
////////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 121)
{
  octetFonction = bitSet (octetFonction1010, 3);//Désactive la fonction
   trame_fonction_clavier_1010();
 }
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F12/////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////////FONCTION = F13/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
if (nbre_fonction == 130)
{
  octetFonction = bitClear (octetFonction11011110, 0);//Désactive la fonction
  trame_fonction_clavier_11011110();
}
////////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 131)
{
  octetFonction = bitSet (octetFonction11011110, 0);//Active la fonction
   trame_fonction_clavier_11011110();
```

```
  }
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F13/////////////////////////////////
////////////////////////////////////////////////////////////////////
//////////////////////FONCTION = F14/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
if (nbre_fonction == 140)
{
  octetFonction = bitClear (octetFonction11011110, 1);//Désactive la fonction
  trame_fonction_clavier_11011110();
}
////////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 141)
{
  octetFonction = bitSet (octetFonction11011110, 1);//Active la fonction
   trame_fonction_clavier_11011110();
  }
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F14/////////////////////////////////
////////////////////////////////////////////////////////////////////
//////////////////////FONCTION = F15/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
if (nbre_fonction == 150)
{
  octetFonction = bitClear (octetFonction11011110, 2);//Désactive la fonction
```

```
  trame_fonction_clavier_11011110();
}
////////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 151)
{
  octetFonction = bitSet (octetFonction11011110, 2);//Active la fonction
   trame_fonction_clavier_11011110();
 }
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F15/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////FONCTION = F16/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
if (nbre_fonction == 160)
{
  octetFonction = bitClear (octetFonction11011110, 3);//Désactive la fonction
  trame_fonction_clavier_11011110();
}
////////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 161)
{
  octetFonction = bitSet (octetFonction11011110, 3);//Active la fonction
   trame_fonction_clavier_11011110();
 }
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F16/////////////////////////////////
////////////////////////////////////////////////////////////////////
```

```
//////////////////////FONCTION = F17////////////////////////////////////
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
if (nbre_fonction == 170)
{
  octetFonction = bitClear (octetFonction11011110, 4);//Désactive la fonction
  trame_fonction_clavier_11011110();
}
////////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 171)
{
  octetFonction = bitSet (octetFonction11011110, 4);//Active la fonction
   trame_fonction_clavier_11011110();
 }
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
//////////////////Fin FONCTION = F17////////////////////////////////////
//////////////////////////////////////////////////////////////////////
//////////////////////FONCTION = F18////////////////////////////////////
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
if (nbre_fonction == 180)
{
  octetFonction = bitClear (octetFonction11011110, 5);//Désactive la fonction
  trame_fonction_clavier_11011110();
}
////////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 181)
{
```

```
  octetFonction = bitSet (octetFonction11011110, 5);//Active la fonction
   trame_fonction_clavier_11011110();
 }
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F18/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////FONCTION = F19/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
if (nbre_fonction == 190)
{
  octetFonction = bitClear (octetFonction11011110, 6);//Désactive la fonction
  trame_fonction_clavier_11011110();
}
 ////////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 191)
{
  octetFonction = bitSet (octetFonction11011110, 6);//Active la fonction
   trame_fonction_clavier_11011110();
 }
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F19/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////FONCTION = F20/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
if (nbre_fonction == 200)
```

```
{
  octetFonction = bitClear (octetFonction11011110, 7);//Désactive la fonction
  trame_fonction_clavier_11011110();
}
////////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 201)
{
  octetFonction = bitSet (octetFonction11011110, 7);//Active la fonction
   trame_fonction_clavier_11011110();
 }
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F20/////////////////////////////////
//////////////////////////////////////////////////////////////////////
////////////////////FONCTION = F21/////////////////////////////////
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
if (nbre_fonction == 210)
{
  octetFonction = bitClear (octetFonction11011111, 0);//Désactive la fonction
  trame_fonction_clavier_11011111();
}
////////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 211)
{
  octetFonction = bitSet (octetFonction11011111, 0);//Active la fonction
   trame_fonction_clavier_11011111();
 }
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
```

```
//////////////////Fin FONCTION = F21////////////////////////////////////
//////////////////////////////////////////////////////////////////////
//////////////////////FONCTION = F22////////////////////////////////////
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
if (nbre_fonction == 220)
{
  octetFonction = bitClear (octetFonction11011111, 1);//Désactive la fonction
  trame_fonction_clavier_11011111();
}
////////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 221)
{
  octetFonction = bitSet (octetFonction11011111, 1);//Active la fonction
   trame_fonction_clavier_11011111();
 }
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
//////////////////Fin FONCTION = F22////////////////////////////////////
//////////////////////////////////////////////////////////////////////
//////////////////////FONCTION = F23////////////////////////////////////
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////
if (nbre_fonction == 230)
{
  octetFonction = bitClear (octetFonction11011111, 2);//Désactive la fonction
  trame_fonction_clavier_11011111();
}
////////////////////////////////////////////////////////////////////////////////////////////////
```

```
if (nbre_fonction == 231)
{
  octetFonction = bitSet (octetFonction11011111, 2);//Active la fonction
   trame_fonction_clavier_11011111();
 }
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F23/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////FONCTION = F24/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
if (nbre_fonction == 240)
{
  octetFonction = bitClear (octetFonction11011111, 3);//Désactive la fonction
  trame_fonction_clavier_11011111();
}
////////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 241)
{
  octetFonction = bitSet (octetFonction11011111, 3);//Active la fonction
   trame_fonction_clavier_11011111();
 }
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F24/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////FONCTION = F25/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
```

```
//////////////////////////////////////////////////////////////////
if (nbre_fonction == 250)
{
  octetFonction = bitClear (octetFonction11011111, 4);//Désactive la fonction
  trame_fonction_clavier_11011111();
}
//////////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 251)
{
  octetFonction = bitSet (octetFonction11011111, 4);//Active la fonction
   trame_fonction_clavier_11011111();
 }
//////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F25/////////////////////////////////
//////////////////////////////////////////////////////////////////
/////////////////////FONCTION = F26/////////////////////////////////
//////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////
if (nbre_fonction == 260)
{
  octetFonction = bitClear (octetFonction11011111, 5);//Désactive la fonction
  trame_fonction_clavier_11011111();
}
//////////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 261)
{
  octetFonction = bitSet (octetFonction11011111, 5);//Active la fonction
   trame_fonction_clavier_11011111();
 }
```

```
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F26/////////////////////////////////
////////////////////////////////////////////////////////////////////
//////////////////////FONCTION = F27/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
if (nbre_fonction == 270)
{
  octetFonction = bitClear (octetFonction11011111, 6);//Désactive la fonction
  trame_fonction_clavier_11011111();
}
////////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 271)
{
  octetFonction = bitSet (octetFonction11011111, 6);//Active la fonction
   trame_fonction_clavier_11011111();
 }
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F27/////////////////////////////////
////////////////////////////////////////////////////////////////////
//////////////////////FONCTION = F28/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
if (nbre_fonction == 280)
{
  octetFonction = bitClear (octetFonction11011111, 7);//Désactive la fonction
  trame_fonction_clavier_11011111();
```

```
}
////////////////////////////////////////////////////////////////////////////////////////////////
if (nbre_fonction == 281)
{
  octetFonction = bitSet (octetFonction11011111, 7);//Active la fonction
   trame_fonction_clavier_11011111();
 }
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
/////////////////Fin FONCTION = F28/////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
}//compteur_touche_fonction == 3

if ((nbre_fonction == 0)||(nbre_fonction == 1)
||(nbre_fonction == 10)||(nbre_fonction == 11)
||(nbre_fonction == 20)||(nbre_fonction == 21)
||(nbre_fonction == 30)||(nbre_fonction == 31)
||(nbre_fonction == 40)||(nbre_fonction == 41)
||(nbre_fonction == 50)||(nbre_fonction == 51)
||(nbre_fonction == 60)||(nbre_fonction == 61)
||(nbre_fonction == 70)||(nbre_fonction == 71)
||(nbre_fonction == 80)||(nbre_fonction == 81)
||(nbre_fonction == 90)||(nbre_fonction == 91)
||(nbre_fonction == 100)||(nbre_fonction == 101)
||(nbre_fonction == 110)||(nbre_fonction == 111)
||(nbre_fonction == 120)||(nbre_fonction == 121)
||(nbre_fonction == 130)||(nbre_fonction == 131)
||(nbre_fonction == 140)||(nbre_fonction == 141)
||(nbre_fonction == 150)||(nbre_fonction == 151)
||(nbre_fonction == 160)||(nbre_fonction == 161)
```

```
||(nbre_fonction == 170)||(nbre_fonction == 171)
||(nbre_fonction == 180)||(nbre_fonction == 181)
||(nbre_fonction == 190)||(nbre_fonction == 191)
||(nbre_fonction == 200)||(nbre_fonction == 201)
||(nbre_fonction == 210)||(nbre_fonction == 211)
||(nbre_fonction == 220)||(nbre_fonction == 221)
||(nbre_fonction == 230)||(nbre_fonction == 231)
||(nbre_fonction == 240)||(nbre_fonction == 241)
||(nbre_fonction == 250)||(nbre_fonction == 251)
||(nbre_fonction == 260)||(nbre_fonction == 261)
||(nbre_fonction == 270)||(nbre_fonction == 271)
||(nbre_fonction == 280)||(nbre_fonction == 281)
||(nbre_fonction == 9))

//Prise en compte du chiffre saisi pour deux entrées
{}
else
{
   compteur_touche_fonction = 0; //Remet le compteur à 0 pour prochaine saisie

   //Gestion des claviers
   autorise_touche_locomotive = 0;
   autorise_touche_fonction = 0;
   autorise_touche_Cv = 0;
   autorise_touche_Cv2 = 0;
   autorise_touche_Cv4 = 0;

   toucheAfonction = 0;//Réinitialise toucheA fonction
   toucheBfonction = 0;//Réinitialise toucheB fonction
   toucheCfonction = 0;//Réinitialise toucheC fonction
   toucheDfonction = 0;//Réinitialise toucheD fonction
```

```
   toucheEfonction = 0;//Réinitialise toucheE fonction

   lcd.clear(); // Efface écran si appui

   lcd.setCursor(7,0);//Place le curseur colonne 0, ligne 1
   lcd.print("Taille") ; // Affiche la chaîne texte

   lcd.setCursor(9,1);//Place le curseur colonne 0, ligne 2
   lcd.print("ou") ; // Affiche la chaîne texte

   lcd.setCursor(3,2);//Place le curseur colonne 0, ligne 3
   lcd.print("Nbre Interdit") ; // Affiche la chaîne texte

   for(int compteurtempo = 0; compteurtempo < 120; compteurtempo++)

   {
    trame_idle();//Envoi trame_idle
   }//fermeture du for
   retourdetrame = 1;//retour à, comme une pression sur la touche *

}//fin if nbre_fonction == 1 ||nbre_fonction == 2 ||

}//Fin du void trame_fonction_clavier
////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////
void envoi_trame_programmation_cv()
{
  digitalWrite(ledrouge,HIGH);
  digitalWrite(ledbleue,HIGH);
  digitalWrite(ledverte,HIGH);
```

```
  digitalWrite(ledjaune,HIGH);


  for(int compteur = 0; compteur < 5; compteur++)
{
  //Bit de Synchronisation 16 bit à 1
// Octet de synchronisation
for ( i=0; i <= 16; i++)
 {bitun();}    // La centrale transmet 16 bits à 1
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
//bit zero
/////////////////////////////////////////
bitzero(); // Bit à 0 de séparation
////////////////////////////////////
////////////////////////////////////
//Octet de d'adresse de départ
if (adr7cv==1) bitun();  // Si bit à 1
if (adr7cv==0) bitzero(); // Si bit à 0

if (adr6cv==1) bitun();  // Si bit à 1
if (adr6cv==0) bitzero(); // Si bit à 0
if (adr5cv==1) bitun();  // Si bit à 1
if (adr5cv==0) bitzero(); // Si bit à 0
if (adr4cv==1) bitun();  // Si bit à 1
if (adr4cv==0) bitzero(); // Si bit à 0
if (adr3cv==1) bitun();  // Si bit à 1
if (adr3cv==0) bitzero(); // Si bit à 0
if (adr2cv==1) bitun();  // Si bit à 1
if (adr2cv==0) bitzero(); // Si bit à 0
if (adr1cv==1) bitun();  // Si bit à 1
```

```
if (adr1cv==0) bitzero(); // Si bit à 0
if (adr0cv==1) bitun();  // Si bit à 1
if (adr0cv==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
//bit zero
//////////////////////////////////////////////////////
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de commande
comm7cv = 1;
comm6cv = 1;
comm5cv = 1;
comm4cv = 0;
comm3cv = 1;
comm2cv = 1;
comm1cv = 0;
comm0cv = 0;

bitun();  // bit à 1: les 3 premiers bits à 1 pour programmation décodeur
bitun(); // bit à 1 : les 3 premiers bits à 1 pour programmation décodeur
bitun();  // bit à 1: les 3 premiers bits à 1 pour programmation décodeur
bitzero(); // bit à 0 : pour pouvoir adresser toutes les cv
bitun();  // bit à 1: les deux bits à 1 permettent d'écrire dans la cv
bitun();  // bit à 1: les deux bits à 1 permettent d'écrire dans la cv

//choix de la cv
//les deux bits à zéro de cet octet font que 256 cv sont adressables par l'octet suivant
```

```
bitzero(); // Si bit à 0 : les deux bits à zéro de cet octet font
bitzero(); // que 256 cv sont adressables par l'octet suivant
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
//bit zero
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet du numéro de la cv
if (adr7cv4==1) bitun();  // Si bit à 1
if (adr7cv4==0) bitzero(); // Si bit à 0

if (adr6cv4==1) bitun();  // Si bit à 1
if (adr6cv4==0) bitzero(); // Si bit à 0
if (adr5cv4==1) bitun();  // Si bit à 1
if (adr5cv4==0) bitzero(); // Si bit à 0
if (adr4cv4==1) bitun();  // Si bit à 1
if (adr4cv4==0) bitzero(); // Si bit à 0
if (adr3cv4==1) bitun();  // Si bit à 1
if (adr3cv4==0) bitzero(); // Si bit à 0
if (adr2cv4==1) bitun();  // Si bit à 1
if (adr2cv4==0) bitzero(); // Si bit à 0
if (adr1cv4==1) bitun();  // Si bit à 1
if (adr1cv4==0) bitzero(); // Si bit à 0
if (adr0cv4==1) bitun();  // Si bit à 1
if (adr0cv4==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
```

```
//bit zero
bitzero();
////////////////////////////////////
//Octet de données à entrer dans la cv
if (adr7cv2==1) bitun();  // Si bit à 1
if (adr7cv2==0) bitzero(); // Si bit à 0

if (adr6cv2==1) bitun();  // Si bit à 1
if (adr6cv2==0) bitzero(); // Si bit à 0
if (adr5cv2==1) bitun();  // Si bit à 1
if (adr5cv2==0) bitzero(); // Si bit à 0
if (adr4cv2==1) bitun();  // Si bit à 1
if (adr4cv2==0) bitzero(); // Si bit à 0
if (adr3cv2==1) bitun();  // Si bit à 1
if (adr3cv2==0) bitzero(); // Si bit à 0
if (adr2cv2==1) bitun();  // Si bit à 1
if (adr2cv2==0) bitzero(); // Si bit à 0
if (adr1cv2==1) bitun();  // Si bit à 1
if (adr1cv2==0) bitzero(); // Si bit à 0
if (adr0cv2==1) bitun();  // Si bit à 1
if (adr0cv2==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
//bit zero
////////////////////////////////////
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de données 3
//Calcul du OU Exclusif
```

```
cont7cv = adr7cv ^ comm7cv ^ adr7cv4 ^ adr7cv2 ;  // ^ Calcul du OU EXCLUSIF bit 7 Octet de contrôle

cont6cv = adr6cv ^ comm6cv ^ adr6cv4 ^ adr6cv2 ; // ^ Calcul du OU EXCLUSIF bit 6 Octet de contrôle

cont5cv = adr5cv ^ comm5cv ^ adr5cv4 ^ adr5cv2 ;  // ^ Calcul du OU EXCLUSIF bit 5 Octet de contrôle

cont4cv = adr4cv ^ comm4cv ^ adr4cv4 ^ adr4cv2 ;  // ^ Calcul du OU EXCLUSIF bit 4 Octet de contrôle

cont3cv = adr3cv ^ comm3cv ^ adr3cv4 ^ adr3cv2 ;  // ^ Calcul du OU EXCLUSIF bit 3 Octet de contrôle

cont2cv = adr2cv ^ comm2cv ^ adr2cv4 ^ adr2cv2 ;  // ^ Calcul du OU EXCLUSIF bit 2 Octet de contrôle

cont1cv = adr1cv ^ comm1cv ^ adr1cv4 ^ adr1cv2 ;  // ^ Calcul du OU EXCLUSIF bit 1 Octet de contrôle

cont0cv = adr0cv ^ comm0cv ^ adr0cv4 ^ adr0cv2 ; // ^ Calcul du OU EXCLUSIF bit 0 Octet de contrôle


// Envoi de l'octet de contrôle

//La centrale transmet l'octet de contrôle

if (cont7cv==1) bitun();  // Si bit à 1

if (cont7cv==0) bitzero(); // Si bit à 0

if (cont6cv==1) bitun();  // Si bit à 1

if (cont6cv==0) bitzero(); // Si bit à 0

if (cont5cv==1) bitun();  // Si bit à 1

if (cont5cv==0) bitzero(); // Si bit à 0

if (cont4cv==1) bitun();  // Si bit à 1

if (cont4cv==0) bitzero(); // Si bit à 0

if (cont3cv==1) bitun();  // Si bit à 1

if (cont3cv==0) bitzero(); // Si bit à 0

if (cont2cv==1) bitun();  // Si bit à 1

if (cont2cv==0) bitzero(); // Si bit à 0

if (cont1cv==1) bitun();  // Si bit à 1

if (cont1cv==0) bitzero(); // Si bit à 0

if (cont0cv==1) bitun();  // Si bit à 1

if (cont0cv==0) bitzero(); // Si bit à 0
```

```
////////////////////////////////////
////////////////////////////////////
//Fin de transmission
//bit un
//////////////////////////////////////////
bitun();
////////////////////////////////////
////////////////////////////////////
// Pause émission de 26 zéros avant nouvelle transmission
//////////////////////////////////////////
for ( i=0; i<=25; i++)
{bitzero();}

}//Fermeture du for

lcd.clear(); // Efface écran si appui
   lcd.setCursor(0,0);//Place le curseur colonne 0, ligne 1
   lcd.print("   PROGRAMMATION") ; // Affiche la chaîne texte
   lcd.setCursor(0,2);//Place le curseur colonne 0, ligne 3
   lcd.print("    EFFECTUEE") ; // Affiche la chaîne texte
   delay(2000); // pause de 1 seconde

retourdetrame=1;//retour à, comme une pression sur la touche *

}//Fin void envoi_trame_programmation_cv
//////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////
////////////////// Génère la trame FONCTION 100//////////////////
////////////////////////////CLAVIER/////////////////////////////
//////////////////////////////////////////////////////////////////
void trame_fonction_clavier_100()
```

```
{
   digitalWrite(ledrouge,LOW);//Lampe URG éteinte
   digitalWrite(ledbleue,HIGH);
   digitalWrite(ledverte,LOW);
   digitalWrite(ledjaune,LOW);

   comm7 = 1;//Les 3 bits commandent les fonctions FL, F1, F2, F3, F4
   comm6 = 0;//Les 3 bits commandent les fonctions FL, F1, F2, F3, F4
   comm5 = 0;//Les 3 bits commandent les fonctions FL, F1, F2, F3, F4
   comm4 = bitRead (octetFonction, 4);
   comm3 = bitRead (octetFonction, 3);
   comm2 = bitRead (octetFonction, 2);
   comm1 = bitRead (octetFonction, 1);
   comm0 = bitRead (octetFonction, 0);

for(int compteur = 0; compteur < 5; compteur++)
{
//Bit de Synchronisation 16 bit à 1
// Octet de synchronisation
 for ( i=0; i <= 20; i++)
 {bitun();}    // La centrale transmet 16 bits à 1
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de d'adresse 1
if (adr7==1) bitun();  // Si bit à 1
if (adr7==0) bitzero(); // Si bit à 0
if (adr6==1) bitun();  // Si bit à 1
```

```
if (adr6==0) bitzero(); // Si bit à 0
if (adr5==1) bitun();  // Si bit à 1
if (adr5==0) bitzero(); // Si bit à 0
if (adr4==1) bitun();  // Si bit à 1
if (adr4==0) bitzero(); // Si bit à 0
if (adr3==1) bitun();  // Si bit à 1
if (adr3==0) bitzero(); // Si bit à 0
if (adr2==1) bitun();  // Si bit à 1
if (adr2==0) bitzero(); // Si bit à 0
if (adr1==1) bitun();  // Si bit à 1
if (adr1==0) bitzero(); // Si bit à 0
if (adr0==1) bitun();  // Si bit à 1
if (adr0==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de données 2
if (comm7==1) bitun();  // Si bit à 1
if (comm7==0) bitzero(); // Si bit à 0
if (comm6==1) bitun();  // Si bit à 1
if (comm6==0) bitzero(); // Si bit à 0
if (comm5==1) bitun();  // Si bit à 1
if (comm5==0) bitzero(); // Si bit à 0
if (comm4==1) bitun();  // Si bit à 1
if (comm4==0) bitzero(); // Si bit à 0
if (comm3==1) bitun();  // Si bit à 1
if (comm3==0) bitzero(); // Si bit à 0
if (comm2==1) bitun();  // Si bit à 1
```

```
if (comm2==0) bitzero(); // Si bit à 0
if (comm1==1) bitun();  // Si bit à 1
if (comm1==0) bitzero(); // Si bit à 0
if (comm0==1) bitun();  // Si bit à 1
if (comm0==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de données 3
//Calcul du OU Exclusif
cont0 = adr0 ^ comm0;  // ^ Calcul du OU EXCLUSIF bit 0 Octet de contrôle
cont1 = adr1 ^ comm1;  // ^ Calcul du OU EXCLUSIF bit 1 Octet de contrôle
cont2 = adr2 ^ comm2;  // ^ Calcul du OU EXCLUSIF bit 2 Octet de contrôle
cont3 = adr3 ^ comm3;  // ^ Calcul du OU EXCLUSIF bit 3 Octet de contrôle
cont4 = adr4 ^ comm4;  // ^ Calcul du OU EXCLUSIF bit 4 Octet de contrôle
cont5 = adr5 ^ comm5;  // ^ Calcul du OU EXCLUSIF bit 5 Octet de contrôle
cont6 = adr6 ^ comm6;  // ^ Calcul du OU EXCLUSIF bit 6 Octet de contrôle
cont7 = adr7 ^ comm7;  // ^ Calcul du OU EXCLUSIF bit 7 Octet de contrôle

if (cont7==1) bitun();  // Si bit à 1
if (cont7==0) bitzero(); // Si bit à 0
if (cont6==1) bitun();  // Si bit à 1
if (cont6==0) bitzero(); // Si bit à 0
if (cont5==1) bitun();  // Si bit à 1
if (cont5==0) bitzero(); // Si bit à 0
if (cont4==1) bitun();  // Si bit à 1
if (cont4==0) bitzero(); // Si bit à 0
if (cont3==1) bitun();  // Si bit à 1
```

```
if (cont3==0) bitzero(); // Si bit à 0
if (cont2==1) bitun();  // Si bit à 1
if (cont2==0) bitzero(); // Si bit à 0
if (cont1==1) bitun();  // Si bit à 1
if (cont1==0) bitzero(); // Si bit à 0
if (cont0==1) bitun();  // Si bit à 1
if (cont0==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
//Fin de transmission
bitun();
////////////////////////////////////
////////////////////////////////////
// Pause émission de 25 zéros (5 ms) avant nouvelle transmission
for ( i=0; i<=25; i++)
{bitzero();
}
retourdetrame=1;//retour à, comme une pression sur la touche *

}//Fin du for 5 fois fonction FL désactivée
}//Fin du void trame 100
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
////////////////////// Génère la trame FONCTION 1011//////////////////////
//////////////////////////////CLAVIER////////////////////////////////
////////////////////////////////////////////////////////////////////
void trame_fonction_clavier_1011()
{
   digitalWrite(ledrouge,LOW);//Lampe URG éteinte
   digitalWrite(ledbleue,HIGH);
   digitalWrite(ledverte,LOW);
```

```
   digitalWrite(ledjaune,LOW);

   comm7 = 1;//Les 3 bits commandent les fonctions F8, F7, F6, F5
   comm6 = 0;//Les 3 bits commandent les fonctions F8, F7, F6, F5
   comm5 = 1;//Les 3 bits commandent les fonctions F8, F7, F6, F5
   comm4 = 1;//Les 3 bits commandent les fonctions F8, F7, F6, F5
   comm3 = bitRead (octetFonction1011, 3);
   comm2 = bitRead (octetFonction1011, 2);
   comm1 = bitRead (octetFonction1011, 1);
   comm0 = bitRead (octetFonction1011, 0);

for(int compteur = 0; compteur < 5; compteur++)
{
//Bit de Synchronisation 16 bit à 1
// Octet de synchronisation
 for ( i=0; i <= 20; i++)
 {bitun();}    // La centrale transmet 16 bits à 1
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de d'adresse 1
if (adr7==1) bitun();  // Si bit à 1
if (adr7==0) bitzero(); // Si bit à 0
if (adr6==1) bitun();  // Si bit à 1
if (adr6==0) bitzero(); // Si bit à 0
if (adr5==1) bitun();  // Si bit à 1
if (adr5==0) bitzero(); // Si bit à 0
if (adr4==1) bitun();  // Si bit à 1
```

```
if (adr4==0) bitzero(); // Si bit à 0
if (adr3==1) bitun();  // Si bit à 1
if (adr3==0) bitzero(); // Si bit à 0
if (adr2==1) bitun();  // Si bit à 1
if (adr2==0) bitzero(); // Si bit à 0
if (adr1==1) bitun();  // Si bit à 1
if (adr1==0) bitzero(); // Si bit à 0
if (adr0==1) bitun();  // Si bit à 1
if (adr0==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de données 2
if (comm7==1) bitun();  // Si bit à 1
if (comm7==0) bitzero(); // Si bit à 0
if (comm6==1) bitun();  // Si bit à 1
if (comm6==0) bitzero(); // Si bit à 0
if (comm5==1) bitun();  // Si bit à 1
if (comm5==0) bitzero(); // Si bit à 0
if (comm4==1) bitun();  // Si bit à 1
if (comm4==0) bitzero(); // Si bit à 0
if (comm3==1) bitun();  // Si bit à 1
if (comm3==0) bitzero(); // Si bit à 0
if (comm2==1) bitun();  // Si bit à 1
if (comm2==0) bitzero(); // Si bit à 0
if (comm1==1) bitun();  // Si bit à 1
if (comm1==0) bitzero(); // Si bit à 0
if (comm0==1) bitun();  // Si bit à 1
```

```
if (comm0==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de données 3
//Calcul du OU Exclusif
cont0 = adr0 ^ comm0;  // ^ Calcul du OU EXCLUSIF bit 0 Octet de contrôle
cont1 = adr1 ^ comm1;  // ^ Calcul du OU EXCLUSIF bit 1 Octet de contrôle
cont2 = adr2 ^ comm2;  // ^ Calcul du OU EXCLUSIF bit 2 Octet de contrôle
cont3 = adr3 ^ comm3;  // ^ Calcul du OU EXCLUSIF bit 3 Octet de contrôle
cont4 = adr4 ^ comm4;  // ^ Calcul du OU EXCLUSIF bit 4 Octet de contrôle
cont5 = adr5 ^ comm5;  // ^ Calcul du OU EXCLUSIF bit 5 Octet de contrôle
cont6 = adr6 ^ comm6;  // ^ Calcul du OU EXCLUSIF bit 6 Octet de contrôle
cont7 = adr7 ^ comm7;  // ^ Calcul du OU EXCLUSIF bit 7 Octet de contrôle

if (cont7==1) bitun();  // Si bit à 1
if (cont7==0) bitzero(); // Si bit à 0
if (cont6==1) bitun();  // Si bit à 1
if (cont6==0) bitzero(); // Si bit à 0
if (cont5==1) bitun();  // Si bit à 1
if (cont5==0) bitzero(); // Si bit à 0
if (cont4==1) bitun();  // Si bit à 1
if (cont4==0) bitzero(); // Si bit à 0
if (cont3==1) bitun();  // Si bit à 1
if (cont3==0) bitzero(); // Si bit à 0
if (cont2==1) bitun();  // Si bit à 1
if (cont2==0) bitzero(); // Si bit à 0
if (cont1==1) bitun();  // Si bit à 1
```

```
if (cont1==0) bitzero(); // Si bit à 0
if (cont0==1) bitun();  // Si bit à 1
if (cont0==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
//Fin de transmission
bitun();
////////////////////////////////////
////////////////////////////////////
// Pause émission de 25 zéros (5 ms) avant nouvelle transmission
for ( i=0; i<=25; i++)
{bitzero();
}
retourdetrame=1;//retour à, comme une pression sur la touche *

}//Fin du for 5 fois fonction FL désactivée
}//Fin du void trame 1011
//////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////
////////////////// Génère la trame FONCTION 1010//////////////////
////////////////////////////CLAVIER///////////////////////////////
//////////////////////////////////////////////////////////////////
// Génère la trame FONCTION 1010
void trame_fonction_clavier_1010()
{
   digitalWrite(ledrouge,LOW);//Lampe URG éteinte
   digitalWrite(ledbleue,HIGH);
   digitalWrite(ledverte,LOW);
   digitalWrite(ledjaune,LOW);

   comm7 = 1;//Les 3 bits commandent les fonctions F12, F11, F10, F9
```

```
   comm6 = 0;//Les 3 bits commandent les fonctions F12, F11, F10, F9
   comm5 = 1;//Les 3 bits commandent les fonctions F12, F11, F10, F9
   comm4 = 0;//Les 3 bits commandent les fonctions F12, F11, F10, F9
   comm3 = bitRead (octetFonction1010, 3);
   comm2 = bitRead (octetFonction1010, 2);
   comm1 = bitRead (octetFonction1010, 1);
   comm0 = bitRead (octetFonction1010, 0);

for(int compteur = 0; compteur < 5; compteur++)
{
//Bit de Synchronisation 16 bit à 1
// Octet de synchronisation
 for ( i=0; i <= 20; i++)
 {bitun();}    // La centrale transmet 16 bits à 1
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de d'adresse 1
if (adr7==1) bitun();  // Si bit à 1
if (adr7==0) bitzero(); // Si bit à 0
if (adr6==1) bitun();  // Si bit à 1
if (adr6==0) bitzero(); // Si bit à 0
if (adr5==1) bitun();  // Si bit à 1
if (adr5==0) bitzero(); // Si bit à 0
if (adr4==1) bitun();  // Si bit à 1
if (adr4==0) bitzero(); // Si bit à 0
if (adr3==1) bitun();  // Si bit à 1
if (adr3==0) bitzero(); // Si bit à 0
```

```
if (adr2==1) bitun();  // Si bit à 1
if (adr2==0) bitzero(); // Si bit à 0
if (adr1==1) bitun();  // Si bit à 1
if (adr1==0) bitzero(); // Si bit à 0
if (adr0==1) bitun();  // Si bit à 1
if (adr0==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de données 2
if (comm7==1) bitun();  // Si bit à 1
if (comm7==0) bitzero(); // Si bit à 0
if (comm6==1) bitun();  // Si bit à 1
if (comm6==0) bitzero(); // Si bit à 0
if (comm5==1) bitun();  // Si bit à 1
if (comm5==0) bitzero(); // Si bit à 0
if (comm4==1) bitun();  // Si bit à 1
if (comm4==0) bitzero(); // Si bit à 0
if (comm3==1) bitun();  // Si bit à 1
if (comm3==0) bitzero(); // Si bit à 0
if (comm2==1) bitun();  // Si bit à 1
if (comm2==0) bitzero(); // Si bit à 0
if (comm1==1) bitun();  // Si bit à 1
if (comm1==0) bitzero(); // Si bit à 0
if (comm0==1) bitun();  // Si bit à 1
if (comm0==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
```

```
//Bit de séparation
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de données 3
//Calcul du OU Exclusif
cont0 = adr0 ^ comm0;  // ^ Calcul du OU EXCLUSIF bit 0 Octet de contrôle
cont1 = adr1 ^ comm1;  // ^ Calcul du OU EXCLUSIF bit 1 Octet de contrôle
cont2 = adr2 ^ comm2;  // ^ Calcul du OU EXCLUSIF bit 2 Octet de contrôle
cont3 = adr3 ^ comm3;  // ^ Calcul du OU EXCLUSIF bit 3 Octet de contrôle
cont4 = adr4 ^ comm4;  // ^ Calcul du OU EXCLUSIF bit 4 Octet de contrôle
cont5 = adr5 ^ comm5;  // ^ Calcul du OU EXCLUSIF bit 5 Octet de contrôle
cont6 = adr6 ^ comm6;  // ^ Calcul du OU EXCLUSIF bit 6 Octet de contrôle
cont7 = adr7 ^ comm7;  // ^ Calcul du OU EXCLUSIF bit 7 Octet de contrôle

if (cont7==1) bitun();  // Si bit à 1
if (cont7==0) bitzero(); // Si bit à 0
if (cont6==1) bitun();  // Si bit à 1
if (cont6==0) bitzero(); // Si bit à 0
if (cont5==1) bitun();  // Si bit à 1
if (cont5==0) bitzero(); // Si bit à 0
if (cont4==1) bitun();  // Si bit à 1
if (cont4==0) bitzero(); // Si bit à 0
if (cont3==1) bitun();  // Si bit à 1
if (cont3==0) bitzero(); // Si bit à 0
if (cont2==1) bitun();  // Si bit à 1
if (cont2==0) bitzero(); // Si bit à 0
if (cont1==1) bitun();  // Si bit à 1
if (cont1==0) bitzero(); // Si bit à 0
if (cont0==1) bitun();  // Si bit à 1
if (cont0==0) bitzero(); // Si bit à 0
```

```
////////////////////////////////////
////////////////////////////////////
//Fin de transmission
bitun();
////////////////////////////////////
////////////////////////////////////
// Pause émission de 25 zéros (5 ms) avant nouvelle transmission
for ( i=0; i<=25; i++)
{bitzero();
}
retourdetrame=1;//retour à, comme une pression sur la touche *

}//Fin du for 5 fois fonction FL désactivée
}//Fin du void trame 1010
////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////
/////////////// Génère la trame FONCTION 11011110/////////////////
////////////////////////////CLAVIER/////////////////////////////
////////////////////////////////////////////////////////////////
void trame_fonction_clavier_11011110()
{
   digitalWrite(ledrouge,LOW);//Lampe URG éteinte
digitalWrite(ledbleue,HIGH);
digitalWrite(ledverte,LOW);
digitalWrite(ledjaune,LOW);

   comm7 = 1;//L'octet commande les fonctions F20, F19, F18, F17, F16, F15, F14, F13
   comm6 = 1;//L'octet commande les fonctions F20, F19, F18, F17, F16, F15, F14, F13
   comm5 = 0;//L'octet commande les fonctions F20, F19, F18, F17, F16, F15, F14, F13
   comm4 = 1;//L'octet commande les fonctions F20, F19, F18, F17, F16, F15, F14, F13
   comm3 = 1;//L'octet commande les fonctions F20, F19, F18, F17, F16, F15, F14, F13
```

```
   comm2 = 1;//L'octet commande les fonctions F20, F19, F18, F17, F16, F15, F14, F13
   comm1 = 1;//L'octet commande les fonctions F20, F19, F18, F17, F16, F15, F14, F13
   comm0 = 0;//L'octet commande les fonctions F20, F19, F18, F17, F16, F15, F14, F13

for(int compteur = 0; compteur < 5; compteur++)
{
//Bit de Synchronisation 16 bit à 1
// Octet de synchronisation
for ( i=0; i <= 20; i++)
 {bitun();}    // La centrale transmet 16 bits à 1
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de d'adresse 1
if (adr7==1) bitun();  // Si bit à 1
if (adr7==0) bitzero(); // Si bit à 0
if (adr6==1) bitun();  // Si bit à 1
if (adr6==0) bitzero(); // Si bit à 0
if (adr5==1) bitun();  // Si bit à 1
if (adr5==0) bitzero(); // Si bit à 0
if (adr4==1) bitun();  // Si bit à 1
if (adr4==0) bitzero(); // Si bit à 0
if (adr3==1) bitun();  // Si bit à 1
if (adr3==0) bitzero(); // Si bit à 0
if (adr2==1) bitun();  // Si bit à 1
if (adr2==0) bitzero(); // Si bit à 0
if (adr1==1) bitun();  // Si bit à 1
if (adr1==0) bitzero(); // Si bit à 0
```

```
if (adr0==1) bitun();  // Si bit à 1
if (adr0==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de données 2
if (comm7==1) bitun();  // Si bit à 1
if (comm7==0) bitzero(); // Si bit à 0
if (comm6==1) bitun();  // Si bit à 1
if (comm6==0) bitzero(); // Si bit à 0
if (comm5==1) bitun();  // Si bit à 1
if (comm5==0) bitzero(); // Si bit à 0
if (comm4==1) bitun();  // Si bit à 1
if (comm4==0) bitzero(); // Si bit à 0
if (comm3==1) bitun();  // Si bit à 1
if (comm3==0) bitzero(); // Si bit à 0
if (comm2==1) bitun();  // Si bit à 1
if (comm2==0) bitzero(); // Si bit à 0
if (comm1==1) bitun();  // Si bit à 1
if (comm1==0) bitzero(); // Si bit à 0
if (comm0==1) bitun();  // Si bit à 1
if (comm0==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
bitzero();
////////////////////////////////////
////////////////////////////////////
```

```
//Octet de fonction
fonc17 = bitRead (octetFonction11011110, 7);
fonc16 = bitRead (octetFonction11011110, 6);
fonc15 = bitRead (octetFonction11011110, 5);
fonc14 = bitRead (octetFonction11011110, 4);
fonc13 = bitRead (octetFonction11011110, 3);
fonc12 = bitRead (octetFonction11011110, 2);
fonc11 = bitRead (octetFonction11011110, 1);
fonc10 = bitRead (octetFonction11011110, 0);

if (fonc17==1) bitun();  // Si bit à 1
if (fonc17==0) bitzero(); // Si bit à 0
if (fonc16==1) bitun();  // Si bit à 1
if (fonc16==0) bitzero(); // Si bit à 0
if (fonc15==1) bitun();  // Si bit à 1
if (fonc15==0) bitzero(); // Si bit à 0
if (fonc14==1) bitun();  // Si bit à 1
if (fonc14==0) bitzero(); // Si bit à 0
if (fonc13==1) bitun();  // Si bit à 1
if (fonc13==0) bitzero(); // Si bit à 0
if (fonc12==1) bitun();  // Si bit à 1
if (fonc12==0) bitzero(); // Si bit à 0
if (fonc11==1) bitun();  // Si bit à 1
if (fonc11==0) bitzero(); // Si bit à 0
if (fonc10==1) bitun();  // Si bit à 1
if (fonc10==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
bitzero();
////////////////////////////////////
```

```
////////////////////////////////////
//Octet de données 3
//Calcul du OU Exclusif
cont0 = adr0 ^ comm0 ^ fonc10;  // ^ Calcul du OU EXCLUSIF bit 0 Octet de contrôle
cont1 = adr1 ^ comm1 ^ fonc11;  // ^ Calcul du OU EXCLUSIF bit 1 Octet de contrôle
cont2 = adr2 ^ comm2 ^ fonc12;  // ^ Calcul du OU EXCLUSIF bit 2 Octet de contrôle
cont3 = adr3 ^ comm3 ^ fonc13;  // ^ Calcul du OU EXCLUSIF bit 3 Octet de contrôle
cont4 = adr4 ^ comm4 ^ fonc14;  // ^ Calcul du OU EXCLUSIF bit 4 Octet de contrôle
cont5 = adr5 ^ comm5 ^ fonc15;  // ^ Calcul du OU EXCLUSIF bit 5 Octet de contrôle
cont6 = adr6 ^ comm6 ^ fonc16;  // ^ Calcul du OU EXCLUSIF bit 6 Octet de contrôle
cont7 = adr7 ^ comm7 ^ fonc17;  // ^ Calcul du OU EXCLUSIF bit 7 Octet de contrôle

if (cont7==1) bitun();  // Si bit à 1
if (cont7==0) bitzero(); // Si bit à 0
if (cont6==1) bitun();  // Si bit à 1
if (cont6==0) bitzero(); // Si bit à 0
if (cont5==1) bitun();  // Si bit à 1
if (cont5==0) bitzero(); // Si bit à 0
if (cont4==1) bitun();  // Si bit à 1
if (cont4==0) bitzero(); // Si bit à 0
if (cont3==1) bitun();  // Si bit à 1
if (cont3==0) bitzero(); // Si bit à 0
if (cont2==1) bitun();  // Si bit à 1
if (cont2==0) bitzero(); // Si bit à 0
if (cont1==1) bitun();  // Si bit à 1
if (cont1==0) bitzero(); // Si bit à 0
if (cont0==1) bitun();  // Si bit à 1
if (cont0==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
//Fin de transmission
```

```
bitun();
////////////////////////////////////
////////////////////////////////////
// Pause émission de 25 zéros (5 ms) avant nouvelle transmission
for ( i=0; i<=25; i++)
{bitzero();}

retourdetrame=1;//retour à, comme une pression sur la touche *

}//Fin du for 5 fois fonction FL désactivée
}//Fin du void trame 11011110
//////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////
//////////////Génère la trame FONCTION 11011111//////////////////
////////////////////////////CLAVIER///////////////////////////////
//////////////////////////////////////////////////////////////////
void trame_fonction_clavier_11011111()
{
   digitalWrite(ledrouge,LOW);//Lampe URG éteinte
digitalWrite(ledbleue,HIGH);
digitalWrite(ledverte,LOW);
digitalWrite(ledjaune,LOW);

   comm7 = 1;//L'octet commande les fonctions F28, F27, F26, F25, F24, F23, F22, F21
   comm6 = 1;//L'octet commande les fonctions F28, F27, F26, F25, F24, F23, F22, F21
   comm5 = 0;//L'octet commande les fonctions F28, F27, F26, F25, F24, F23, F22, F21
   comm4 = 1;//L'octet commande les fonctions F28, F27, F26, F25, F24, F23, F22, F21
   comm3 = 1;//L'octet commande les fonctions F28, F27, F26, F25, F24, F23, F22, F21
   comm2 = 1;//L'octet commande les fonctions F28, F27, F26, F25, F24, F23, F22, F21
   comm1 = 1;//L'octet commande les fonctions F28, F27, F26, F25, F24, F23, F22, F21
   comm0 = 1;//L'octet commande les fonctions F28, F27, F26, F25, F24, F23, F22, F21
```

```
for(int compteur = 0; compteur < 5; compteur++)
{
//Bit de Synchronisation 16 bit à 1
// Octet de synchronisation
for ( i=0; i <= 20; i++)
 {bitun();}    // La centrale transmet 16 bits à 1

////////////////////////////////////
////////////////////////////////////
//Bit de séparation
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de d'adresse 1
if (adr7==1) bitun();  // Si bit à 1
if (adr7==0) bitzero(); // Si bit à 0
if (adr6==1) bitun();  // Si bit à 1
if (adr6==0) bitzero(); // Si bit à 0
if (adr5==1) bitun();  // Si bit à 1
if (adr5==0) bitzero(); // Si bit à 0
if (adr4==1) bitun();  // Si bit à 1
if (adr4==0) bitzero(); // Si bit à 0
if (adr3==1) bitun();  // Si bit à 1
if (adr3==0) bitzero(); // Si bit à 0
if (adr2==1) bitun();  // Si bit à 1
if (adr2==0) bitzero(); // Si bit à 0
if (adr1==1) bitun();  // Si bit à 1
if (adr1==0) bitzero(); // Si bit à 0
if (adr0==1) bitun();  // Si bit à 1
if (adr0==0) bitzero(); // Si bit à 0
```

```
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de données 2
if (comm7==1) bitun();  // Si bit à 1
if (comm7==0) bitzero(); // Si bit à 0
if (comm6==1) bitun();  // Si bit à 1
if (comm6==0) bitzero(); // Si bit à 0
if (comm5==1) bitun();  // Si bit à 1
if (comm5==0) bitzero(); // Si bit à 0
if (comm4==1) bitun();  // Si bit à 1
if (comm4==0) bitzero(); // Si bit à 0
if (comm3==1) bitun();  // Si bit à 1
if (comm3==0) bitzero(); // Si bit à 0
if (comm2==1) bitun();  // Si bit à 1
if (comm2==0) bitzero(); // Si bit à 0
if (comm1==1) bitun();  // Si bit à 1
if (comm1==0) bitzero(); // Si bit à 0
if (comm0==1) bitun();  // Si bit à 1
if (comm0==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de fonction
fonc27 = bitRead (octetFonction11011111, 7);
```

```
fonc26 = bitRead (octetFonction11011111, 6);
fonc25 = bitRead (octetFonction11011111, 5);
fonc24 = bitRead (octetFonction11011111, 4);
fonc23 = bitRead (octetFonction11011111, 3);
fonc22 = bitRead (octetFonction11011111, 2);
fonc21 = bitRead (octetFonction11011111, 1);
fonc20 = bitRead (octetFonction11011111, 0);

if (fonc27==1) bitun();  // Si bit à 1
if (fonc27==0) bitzero(); // Si bit à 0
if (fonc26==1) bitun();  // Si bit à 1
if (fonc26==0) bitzero(); // Si bit à 0
if (fonc25==1) bitun();  // Si bit à 1
if (fonc25==0) bitzero(); // Si bit à 0
if (fonc24==1) bitun();  // Si bit à 1
if (fonc24==0) bitzero(); // Si bit à 0
if (fonc23==1) bitun();  // Si bit à 1
if (fonc23==0) bitzero(); // Si bit à 0
if (fonc22==1) bitun();  // Si bit à 1
if (fonc22==0) bitzero(); // Si bit à 0
if (fonc21==1) bitun();  // Si bit à 1
if (fonc21==0) bitzero(); // Si bit à 0
if (fonc20==1) bitun();  // Si bit à 1
if (fonc20==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
//Bit de séparation
bitzero();
////////////////////////////////////
////////////////////////////////////
//Octet de données 3
```

```
//Calcul du OU Exclusif
cont0 = adr0 ^ comm0 ^ fonc20;  // ^ Calcul du OU EXCLUSIF bit 0 Octet de contrôle
cont1 = adr1 ^ comm1 ^ fonc21;  // ^ Calcul du OU EXCLUSIF bit 1 Octet de contrôle
cont2 = adr2 ^ comm2 ^ fonc22;  // ^ Calcul du OU EXCLUSIF bit 2 Octet de contrôle
cont3 = adr3 ^ comm3 ^ fonc23;  // ^ Calcul du OU EXCLUSIF bit 3 Octet de contrôle
cont4 = adr4 ^ comm4 ^ fonc24;  // ^ Calcul du OU EXCLUSIF bit 4 Octet de contrôle
cont5 = adr5 ^ comm5 ^ fonc25;  // ^ Calcul du OU EXCLUSIF bit 5 Octet de contrôle
cont6 = adr6 ^ comm6 ^ fonc26;  // ^ Calcul du OU EXCLUSIF bit 6 Octet de contrôle
cont7 = adr7 ^ comm7 ^ fonc27;  // ^ Calcul du OU EXCLUSIF bit 7 Octet de contrôle

if (cont7==1) bitun();  // Si bit à 1
if (cont7==0) bitzero(); // Si bit à 0
if (cont6==1) bitun();  // Si bit à 1
if (cont6==0) bitzero(); // Si bit à 0
if (cont5==1) bitun();  // Si bit à 1
if (cont5==0) bitzero(); // Si bit à 0
if (cont4==1) bitun();  // Si bit à 1
if (cont4==0) bitzero(); // Si bit à 0
if (cont3==1) bitun();  // Si bit à 1
if (cont3==0) bitzero(); // Si bit à 0
if (cont2==1) bitun();  // Si bit à 1
if (cont2==0) bitzero(); // Si bit à 0
if (cont1==1) bitun();  // Si bit à 1
if (cont1==0) bitzero(); // Si bit à 0
if (cont0==1) bitun();  // Si bit à 1
if (cont0==0) bitzero(); // Si bit à 0
////////////////////////////////////
////////////////////////////////////
//Fin de transmission
bitun();
////////////////////////////////////
```

```
////////////////////////////////////
// Pause émission de 25 zéros (5 ms) avant nouvelle transmission
for ( i=0; i<=25; i++)
{bitzero();}

retourdetrame=1;//retour à, comme une pression sur la touche *

}//Fin du for 5 fois fonction FL désactivée
}//Fin du void trame 11011111

////////////////////////////////////
////////////////////////////////////
//Remet toutes les fonctions à 0 sf Fl, F2, F3, F4 (Accès par interrupteurs)
void trame_fonction_raz()
{
bitClear (octetFonction11011111, 7);//F28
bitClear (octetFonction11011111, 6);//F27
bitClear (octetFonction11011111, 5);//F26
bitClear (octetFonction11011111, 4);//F25
bitClear (octetFonction11011111, 3);//F24
bitClear (octetFonction11011111, 2);//F23
bitClear (octetFonction11011111, 1);//F22
bitClear (octetFonction11011111, 0);//F21

bitClear (octetFonction11011110, 7);//F20
bitClear (octetFonction11011110, 6);//F19
bitClear (octetFonction11011110, 5);//F18
bitClear (octetFonction11011110, 4);//F17
bitClear (octetFonction11011110, 3);//F16
bitClear (octetFonction11011110, 2);//F15
bitClear (octetFonction11011110, 1);//F14
```

```
bitClear (octetFonction11011110, 0);//F13

bitClear (octetFonction1010, 3);//F12
bitClear (octetFonction1010, 2);//F11
bitClear (octetFonction1010, 1);//F10
bitClear (octetFonction1010, 0);//F9

bitClear (octetFonction1011, 3);//F8
bitClear (octetFonction1011, 2);//F7
bitClear (octetFonction1011, 1);//F6
bitClear (octetFonction1011, 0);//F5

/*bitClear (octetFonction, 4);//FL ou f0
bitClear (octetFonction, 3);//F4
bitClear (octetFonction, 2);//F3
bitClear (octetFonction, 1);//F2
bitClear (octetFonction, 0);//F1
*/
trame_fonction_clavier_1010();
trame_fonction_clavier_1011();
trame_fonction_clavier_11011110();
trame_fonction_clavier_11011111();

retourdetrame=1;//retour à, comme une pression sur la touche *


}//Fin du void trame fonction raz
```

```
// --- Fin programme ---
```