

```

/*
Fichier de configuration de l'application Centrale Domotique "DomoMaison 2017" sur carte Arduino Mega 2560 avec Ethernet
shield
utilisant le protocole Websockets.
Version 1.70 du 06/09/2017
*/

// -----A modifier suivant les matériels utilisés sur la carte Arduino ou les logiciels utilisés -----
// Zone de configuration du programme pour COMPILATION CONDITIONNELLE. Lignes à décommenter ou commenter suivant utilisation
// Partie concernant le débogage du programme. A commenter ou à décommenter selon la partie à contrôler
#define DEBUG // Mode DEBUG pour affichage de informations de test sur le serial port ( mettre
systématiquement)
#define DEBUG_MEM // pour affichage dans console la mémoire restante
// pour tout débogage
// #define DEBUG_TELECOM // Mode DEBUG pour la gestion des télécommandes (volets, portail, portes de
garage, prises électriques)
// #define DEBUG_TEMP // Mode DEBUG pour la gestion des températures
// #define DEBUG_ELEC // Mode DEBUG pour l'électricité et la téléinformation
// #define DEBUG_ALAR // Mode DEBUG pour les alarmes (Effraction et Incendie)
// #define DEBUG_CHAUF // Mode DEBUG pour les fils pilote et le chauffage
// #define DEBUG_MAIL // Mode DEBUG pour l'envoi de mail

// ***** Partie concernant les modules à installer ou non. Commenter ou décommenter selon les modules à installer
*****
#define CHAUFCONF // Mode CHAUFCONF pour permettre l'utilisation de commande "fil pilote" sur les
matériels électriques.
#define MAILCONF // Mode MAILCONF pour permettre l'envoi de e-mails;
// -----

//***** SECTION DE PARAMETRAGE SPECIFIQUE A CHAQUE CARTE ( Zone à modifier - Zone change)
*****
//adresse mac = identifiant unique du shield
byte MAC[] = {0x90, 0xA2, 0xDA, 0x00, 0x1A, 0x71 };
//adresse IP fixe à utiliser pour le shield Ethernet
IPAddress IP_LOCAL(192,168,1,160); // l'adresse IP locale du shield Ethernet
// ATTENTION : il faut utiliser une adresse hors de la plage d'adresses du routeur DHCP
// IPAddress IP_LOCAL(169,254,99,63); // Adresse Ip pour test avec cable RJ45 croisé

#define PORT_ARDUINO 1050 // Port d'écoute du serveur Internet TCP de la carte Arduino ,
exemple 4832 ou par défaut 80

// variable pour signaler si l'on veut une mise à jour du RTC par un serveur NTP
#define MISE_JOUR_NTP 0 // 0 pour pas de mise à jour, 1 pour mise à jour

// Identifiant de la carte
#define NUM_BOARD "ME01" // Valeurs ME00, ME01, ..., ME99

// Code d'autorisation d'accès à la carte.
#define CODE_ACCES "4523"

// Compteur de boucle de base pour la période de l'affichage sur LCD.
#define CYCLE_LCD 500 // soit 5 secondes pour un cycle effectif de 10 ms.

// Compteur de boucle de base avant déconnexion si le websocket est déjà déconnecté
#define NB_PERIODE_DECONNEXION 1000 // soit 1000 = 10 secondes pour un cycle effectif moyen de 10 ms

// Vitesse de transmission de la liaison avec le port série
#define SPEED_TRANSMISSION 115200 // en bit/s (utile uniquement en mode DEBUG).

// Compteur de boucle de base de l'horloge RTC pour la mise à jour de l'horloge interne.
#define CYCLE_RTC 6000 //pour un cycle effectif de 10ms: soit 10*6000 = 60 secondes environ.

// Compteur de boucle de base avant une demande de date/heure auprès du serveur NTP.
#define CYCLE_NTP 2 // 2 = pour un cycle de mise à jour tous les 2 jours.

// Adresse IP du serveur NTP à sélectionner (à commenter ou décommenter)
// IPAddress timeServeur(138,96,64,10); // adresses serveur NTP Jussieu, Oleanne, etc...
// IPAddress timeServeur(134,157,254,19);
// IPAddress timeServeur(130,149,17,21);
// IPAddress timeServeur(138,96,64,10); // adresses serveur NTP Jussieu, Oleanne, etc...
// IPAddress timeServer(192, 43, 244, 18); // time.nist.gov NTP server - PB -
// IPAddress timeServer(130, 149, 17, 21); // ntps1-0.cs.tu-berlin.de
// IPAddress TIME_SERVER(194, 2, 0, 28); // ntp0.oleanne.net
// IPAddress timeServer(195, 220, 194, 193); // ntp.sophia.cnrs.fr

#define PORT_TIME_SERVER 8888 // Port Ethernet pour l'Ecoute des trames UDP sur la carte Arduino

// Période pour l'envoi des entrées numérique ou analogiques (afin d'éviter des envois permanents)
// Si 3000 alors cycle de 3 secondes entre 2 envois successifs
#define PERIODE_CYCLE_ANA 3000

// ----- Définition des broches utilisées -----
// Définition de la broche de sélection de la carte SD sur le shield Arduino
#define PIN_SELECT_SD 4

// Définition de la broche attribuée à la sortie digitale du watchdog
#define PIN_WATCHDOG 8

// Définition de la broche attribuée à la sortie visuelle du watchdog
#define PIN_LED_WATCHDOG 9

// Définition des broches utilisées pour la commande de l'afficheur LCD
#define LCD_RS A5 // déclaration constante broche LCD RS
#define LCD_ENABLE A4 // déclaration constante broche LCD EN
#define LCD_D4 A3 // déclaration constante broche LCD D4
#define LCD_D5 A2 // déclaration constante broche LCD D5
#define LCD_D6 A1 // déclaration constante broche LCD D6
#define LCD_D7 A0 // déclaration constante broche LCD D7
#define PIN_LDR A6 // déclaration constante broche analogique LDR
#define PIN_BLLCD A7 // déclaration broche Back Light LCD

```

```

// ***** FIN SECTION SYSTEME
*****

//***** DEBUT SECTION D'AJOUT DE LA GESTION DES VOLETS ROULANTS, DU PORTAIL ET DES PORTES DE GARAGES
*****
// Nombre de volets à commander. Valeurs à modifier si le nombre de volets change.
byte NB_VOLETS = 8; // ATTENTION, le nombre de volets maximum est de 19, sinon risque de bugs

// Définition des broches de commandes des volets
#define PIN_VOL_GAUCHE 22 // déclaration constante broche Sélection gauche volets
#define PIN_VOL_MILIEU 23 // déclaration constante broche Sélection milieu volets
#define PIN_VOL_DROIT 24 // déclaration constante broche Sélection droit volets
#define PIN_VOL_HAUT 25 // déclaration constante broche haut volets
#define PIN_VOL_STOP 26 // déclaration constante broche stop volets
#define PIN_VOL_BAS 27 // déclaration constante broche bas volets

// Définition des broches de commandes des portails
#define PIN_PORTAIL 31 // déclaration constante broche bouton portail
#define PIN_LIBRE_1 30 // déclaration constante broche bouton portail (libre)

// Définition des broches de commandes des portes de garage
#define PIN_GARAGE_1 29 // déclaration constante broche bouton porte garage gauche
#define PIN_GARAGE_2 28 // déclaration constante broche bouton porte garage droite
//***** FIN SECTION D'AJOUT DE LA GESTION DES VOLETS ROULANTS, DU PORTAIL ET DES PORTES DE GARAGES
*****

//***** DEBUT SECTION D'AJOUT DE LA GESTION DES TELECOMMANDES DE PRISE ELECTRIQUES
*****
// Identifiants des commandes de prises électriques
char* ID_INTER[] = {"046a9f", "xxxxxx", "xxxxxx", "xxxxxx", "xxxxxx", "xxxxxx", "xxxxxx", "xxxxxx", "xxxxxx", "xxxxxx"};
//***** FIN SECTION D'AJOUT DE LA GESTION DES TELECOMMANDES DE PRISE ELECTRIQUES
*****

//***** DEBUT SECTION D'AJOUT DE LA GESTION DES TEMPERATURES (sondes filaires, sondes sans fil)
*****
// Nombre de sondes de température sans fil installées. Valeurs à modifier si le nombre de sondes change.
byte NB_SONDEST_S = 4; // ATTENTION, le nombre de sondes maximum est de 10, sinon risque de bugs
// Nombre de sondes de température filaires installées. Valeurs à modifier si le nombre de sondes change.
byte NB_SONDEST_F = 1; // ATTENTION, le nombre de sondes maximum est de 5, sinon risque de bugs
// Nombre de pièces de l'habitation (nécessaire pour l'appairage des sondes)
byte NB_PIECES = 15;

// Compteur de boucle pour la lecture des sondes de température filaires
#define CYCLE_FIL 1500 // pour un cycle effectif de 10ms: soit 10*1500 = 15 secondes environ.

// Compteur de boucle pour le calcul des minima te maxima des températures extérieure et intérieures
#define CYCLE_CALCUL_TE 6000 // pour un cycle effectif de 10ms: soit 10*6000 = 60 secondes environ.

// Définition du type de sondes filaires utilisées
#define DHT_TYPE DHT22 // DHT 22 (AM2302), AM2321

// Définition des broches pour les sondes de température DHT22 (5 sondes maxi)
#define PIN_DHT_1 34 // Sonde 1
#define PIN_DHT_2 35 // Sonde 2
#define PIN_DHT_3 36 // Sonde 3
#define PIN_DHT_4 37 // Sonde 4
#define PIN_DHT_5 38 // Sonde 5
#define PIN_DHT_6 39 // Sonde 6, température prise derrière le boîtier de la centrale domotique
//***** FIN SECTION D'AJOUT DE LA GESTION DES TEMPERATURES (sondes filaires, sondes sans fil)
*****

//***** DEBUT SECTION D'AJOUT DE LA GESTION DES DONNEES ELECTRIQUES ET TELEINFORMATION
*****
// Période pour l'envoi des informations du compteur électrique (afin d'éviter des envois permanents)
// Si 100 alors cycle de 1 seconde entre 2 envois successifs
#define PERIODE_CYCLE_ELECTRIQUE 3000

// Définition du temps maximum en cycle de base avant de déclarer la liaison téléinformation hors service
#define DETECTION_TELEINFO 6000 // pour un cycle effectif de 10ms soit 10*6000 = 60 secondes environ

// Type d'abonnement EDF afin d'éviter l'envoi de données téléinfo inutiles
#define ABONNEMENT "EJP"
// #define ABONNEMENT "BASE"
// #define ABONNEMENT "TEMPO"

// Type de compteur car les anciens compteurs ne disposent pas de certaines données
// #define TYPE_COMPTEUR "NOUV"
#define TYPE_COMPTEUR "ANCI"

// Définition de la broche pour le signa EJP
#define PIN_EJP 7 // broche d'entrée du signal EZJP (mise à la masse en cas d'EJP)
//***** FIN SECTION D'AJOUT DE LA GESTION DES DONNEES ELECTRIQUES ET TELEINFORMATION
*****

//***** DEBUT SECTION D'AJOUT DE LA GESTION DE L'ALARME et DES CAMERAS
*****
// Identifiants des contacteurs d'ouverture de l'alarme
char* ID_CONTACTEUR[] = {"98cf48", "4aca4a", "xxxxxx", "xxxxxx", "xxxxxx", "xxxxxx", "xxxxxx", "xxxxxx", "xxxxxx", "xxxxxx"};
// Zone à laquelle le contacteur est rattaché (1 à 3 : Intérieur Maison, Extérieur Maison, Dépendances)
byte ZONE_CONTACT[] = {3,3,1,1,1,1,1,1,1,1};
// Identifiants des détecteurs IR de l'alarme
char* ID_DETECTEUR[] = {"22220f", "aaaaaa", "0a2233", "xxxxxx", "xxxxxx", "xxxxxx", "xxxxxx", "xxxxxx", "xxxxxx", "xxxxxx"};
// Zone à laquelle le détecteur IR est rattaché (1 à 3 : Intérieur Maison, Extérieur Maison, Dépendances)
byte ZONE_DETECT[] = {1,1,2,1,1,1,1,1,1,1};
// Temporisation de mise en marche de l'alarme
#define TEMPO_MARCHE_ALARME 2 // en minutes
// Temporisation pour validation effraction en zone 1
#define TEMPO_VALID_ALARME 2 // en minutes

```

```

// Durée de fonctionnement de la sirène
#define DUREE_EFFRACTION_ALARME 3 // en minutes
// Définition de la broche pour la commande de la sirène de l'alarme
#define PIN_SIRENE 46
// Définition de la broche pour la commande du gyrophare de l'alarme
#define PIN_GYRO 47
// Définition du mode de fonctionnement de l'alarme lors d'un reset de la centrale (ON = activé, OFF = désactivée)
#define MODE_FONCT_ALARME "OFF"
// Définition du mode de fonctionnement du Gyro en mode silence (OFF = pas de gyro en mode silence)
#define MODE_GYRO "ON"
//***** FIN SECTION D'AJOUT DE LA GESTION DE L'ALARME et DES CAMERAS
*****

//***** DEBUT SECTION D'AJOUT DE LA GESTION D'ENVOI D'EMAIL
*****
#ifdef MAILCONF
//Zone de paramétrage de l'envoi de mail via un serveur SMTP
// Serveurs SMTP pour envoi mail avec addresses IP pour ORANGE/WANADOO par ordre de priorité.
byte SERVER_MAIL_00[] = { 193, 252, 22, 64};
byte SERVER_MAIL_01[] = { 193, 252, 22, 86};
byte SERVER_MAIL_02[] = { 193, 252, 22, 84};
byte SERVER_MAIL_03[] = { 80, 12, 242, 10};
byte SERVER_MAIL_04[] = { 80, 12, 242, 16};
byte SERVER_MAIL_05[] = { 80, 12, 242, 86};
//
#define PORT_COMPUTER_MAIL 25 //25, 587,...port du serveur internet SMTP distant.
#define LIB_EXPEDITEUR_MAIL "<xxxxxx@orange.fr>" // libellé expéditeur.
#define LIB_DESTINATAIRE_MAIL "<xxxxxxxx@gmail.com>" // libellé du destinataire.
#define LIB_COPIE_SMTP "<xxxxxx@orange.fr>" //libellé du destinataire en copie <xxxxxxxx@gmail.com>
ou "" si pas de destinataire du mail en copie.
#define LIB_CODE_AUTH_PLAIN "AGplCHJpbgB5eTQzZ3Vm" //code d'authentification de l'expéditeur au format
PLAIN.
#define LIB_SUBJECT_MAIL "Alerte_Centrale_Domotique" //Texte générique du message mail de la carte Arduino
Mega.
#define TEXTE_MAIL_00 "Reset carte Arduino" //30 caractères max.
#define TEXTE_MAIL_01 "Fonct. default de la centrale" //30 caractères max.
#define TEXTE_MAIL_02 "Fonct. normal de la centrale" //30 caractères max.
#define TEXTE_MAIL_03 "Detection presence par Ouvert" //30 caractères max.
#define TEXTE_MAIL_04 "Detection presence par PIR" //30 caractères max.
#define TEXTE_MAIL_05 "Depassement puissance elec." //30 caractères max.
//
#define HEURE_MAIL 19 //heure du mail journalier de contrôle.
#define MINUTE_MAIL 30 //minute du mail journalier de contrôle.
#define DUREE_FILTRE_EVENEMENT 300 //durée en seconde (ex: 5 minutes) permettant le filtrage
d'événements successifs après un envoi de mail (afin d'éviter une avalanche de messages).
//
#define ATTENTE_MAIL 1000 // nombre de boucle de base pour les délais d'attente en
cours de procédure d'envoi de email.
#define ATTENTE_SERVEUR_SMTP 1000 //nombre de boucle de base pour attente afin de faire
appel au serveur smtp suivant (ex: 10 secondes) sur cycle effectif de 10ms.
//
//Définition du seuil de puissance (information PAPP fourni par la Teleinfo du compteur électrique) pour transmettre une
alarme par mail si le commutateur présence/absence est sur absent.
#define SEUIL_MAIL_PUISSANCE_ABSCENT 40 //valeur max au delà duquel une alarme est transmise par
mail (Intensité instantanée).
//
#endif
//***** FIN SECTION D'AJOUT DE LA GESTION D'ENVOI D'EMAIL
*****

```