

Interface sans fil 868mhz pour Crouzet Millenium 3 : panMillenium3

Attention à
l'électricité
statique

Introduction**Fonctionnalités :**

panMillenium3, pour « personal area network for Millenium 3 » permet de transformer un automate Crouzet Millenium pour le faire communiquer sur un réseau sans fil dans la gamme de fréquence autorisée en Europe de 868mhz. Il utilise le module intégré « panSTamp » compatible Arduino.

L'automate pourra transmettre jusqu'à 24 mots (16 bits) vers d'autres points (nodes) ou d'autres automates, et recevoir et traiter jusqu'à 24 mots transmis par d'autres points/nodes ou d'autres automates. Le programme « FBD » de l'automate peut gérer ces informations à travers les blocs de fonction d'entrée sortie SLin et SLout.

Les kits peuvent être utilisés ensemble et/ou avec d'autres capteurs panSTamp sans aucune programmation, juste en paramétrant certains registres à distance. Il n'y a donc pas besoin d'un ordinateur central dans une installation panMillenium/panSTamp, sauf pour la programmation initiale.

L'électronique :

Le kit est construit autour d'un module panSTamp et de 2 connecteurs, montés sur un circuit imprimé de petite taille qui se loge dans l'emplacement normalement réservé au câble de programmation.

Le module panSTamp comprend un processeur (Atmel atmega328p à 8mhz) et un circuit d'interface RF (radio fréquence) CC1101, assurant le transport des paquets d'information sur la bande 868mhz.

Le connecteur 6 broches s'insère au fond du logement pour câble série. Le circuit imprimé recouvre une partie des 6 broches (environ 3mm) et sert ainsi de détrompeur et à maintenir légèrement le circuit en position, par la pression exercée sur le côté du connecteur femelle. Prêter une très grande attention lors de l'introduction du module et ne pas le solliciter lorsque l'automate est sous tension.

Le connecteur avant de 16 broches reprend les signaux 5V, OV, TXD et RXD de l'automate, et fournit quelques signaux complémentaires du circuit panSTamps (I2C, interruption, entrée analogiques...) qui peuvent être utilisés par une application spécifique et un circuit séparé fonctionnant sous 3.3 volts.

Ne jamais court-circuiter les broches sous peine d'endommager définitivement les entrées sorties correspondantes du module, ou de l'automate. Prévoir éventuellement un connecteur femelle borgne pour protéger les broches de tout contact.

Une LED verte permet de donner une indication sommaire sur le bon fonctionnement de la communication avec l'automate.

Brochage du connecteur, vue de face :

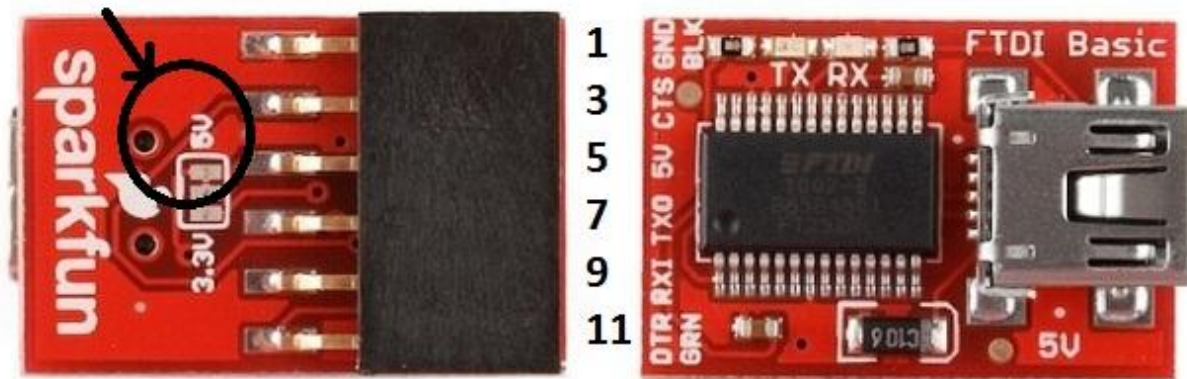
T1 – ADC7 (pin 12 et 20 panstamp)	16	15	PC4 – SDA (pin 9 panstamp)
T0 – ADC6 (pin 11 et 19 panstamp)	14	13	PC5 – SCL (pin10 panstamp)
INT1 (pin 18 panstamp)	12	11	Entrée reset2 (DTR sur FTDI)
Pin 17 panstamp (TXD)	10	9	RXD automate (RXI sur FTDI)
Pin 16 panstamp (RXD)	8	7	TXD automate (TXO sur FTDI)
0Volt	6	5	+5V automate (ou FTDI)
+3V3 régulé	4	3	0volts (CTS sur FTDI)
RST (pin 13 panstamp)	2	1	0Volts GND sur FTDI)

Les broches en rouge doivent être reliées à des circuits fonctionnant en 3V3 exclusivement.

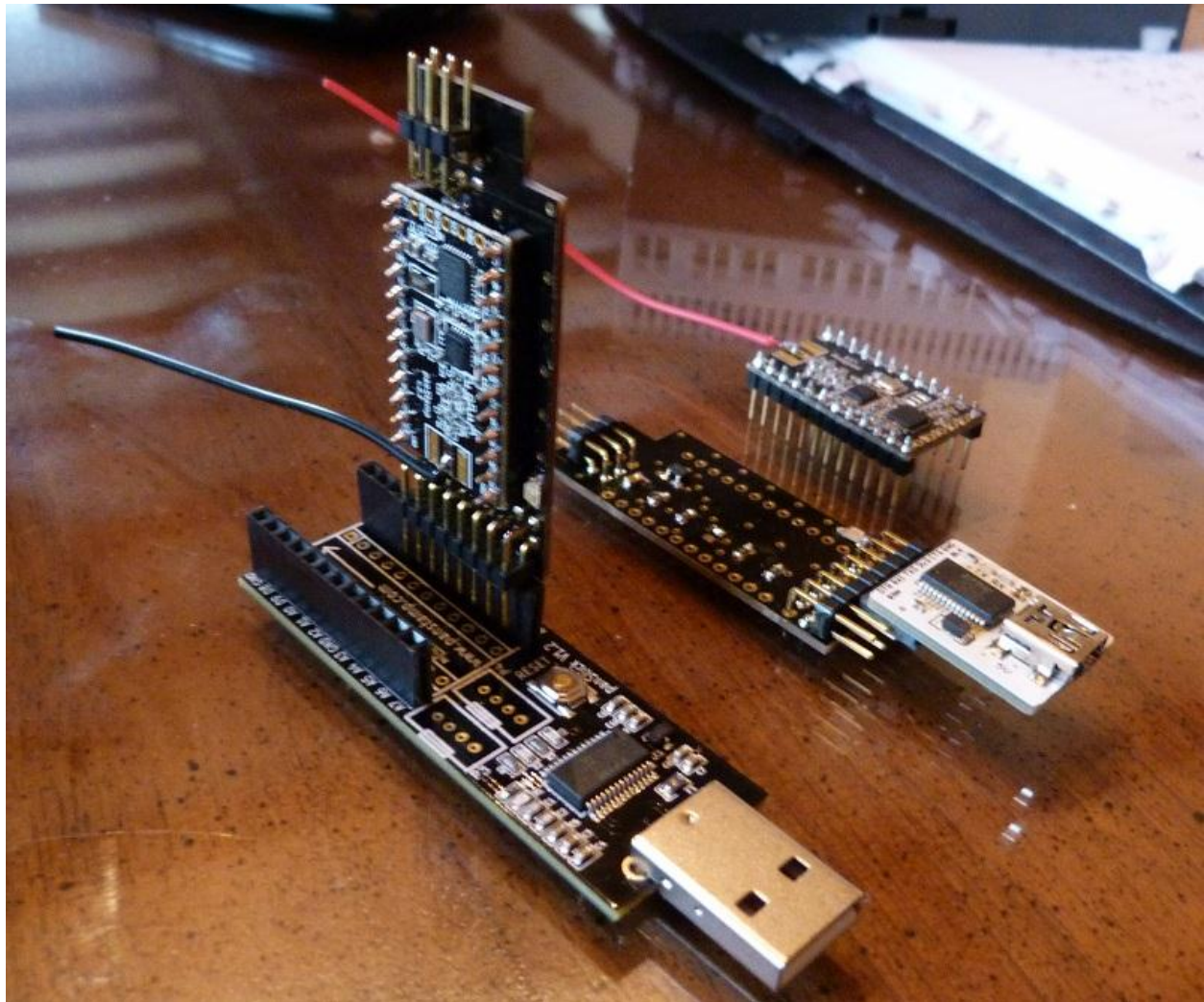
Programmation / raccordement PC:

Pour charger une nouvelle application dans le panMillenium, le kit doit impérativement être enlevé de l'automate, et relié à un convertisseur USB/série relié à un PC/MAC.

Cas d'un convertisseur type FTDI232 à 6 broches : il doit être raccordé sur les broches 1-3-5-7-9-11, dans le sens qui permet d'aligner les 2 broches généralement nommées GND et CTS avec les broches 1 et 3. Les broches TXO et RXI seront respectivement en face des broches 7 et 9.



Utilisation avec un panStick : le kit doit être monté verticalement sur l'une des rangées du panstick, pour faire coïncider les broches « impaires » 1 à 15 respectivement sur les broches RST à D5 du panstick.



Programation/telechargement :

Installation des programmes nécessaires :

Le kit panMillenium se programme avec l'environnement de développement Arduino IDE, disponible sur Windows, mac osx ou linux.

Télécharger et installer la version 1.03 (voir site web arduino.cc)

Télécharger la librairie panMillenium.zip (site google code <http://tinyurl.com/ct5pld6>) et la décompresser dans le dossier «arduino1.03\libraries », pour créer un sous répertoire panMillenium à cet endroit.

La librairie panMillenium est fournie avec 3 autres librairies nommées Scoop, IOFilter et modbusBasic. trois dossiers correspondant seront aussi créés dans le répertoire « arduino1.03\libraries »

Remarque : Si vous utilisez un système panstamp pour la première fois, il faut ajouter les lignes de texte contenues dans le fichier libraries\panMillenium\board.txt au début du fichier principal board.txt

contenu dans « arduino1.03\hardware\Arduino » cela permettra de choisir « panstamp 8mhz » dans le menu outils/tools de l'environnement Arduino.

Si les fichiers board.txt n'apparaissent pas correctement formaté (sur PC), il faudra utiliser un éditeur de texte différent (exemple Notepad++) qui supporte les fichiers textes au format « linux ».

Installation de l'interface USB/serie :

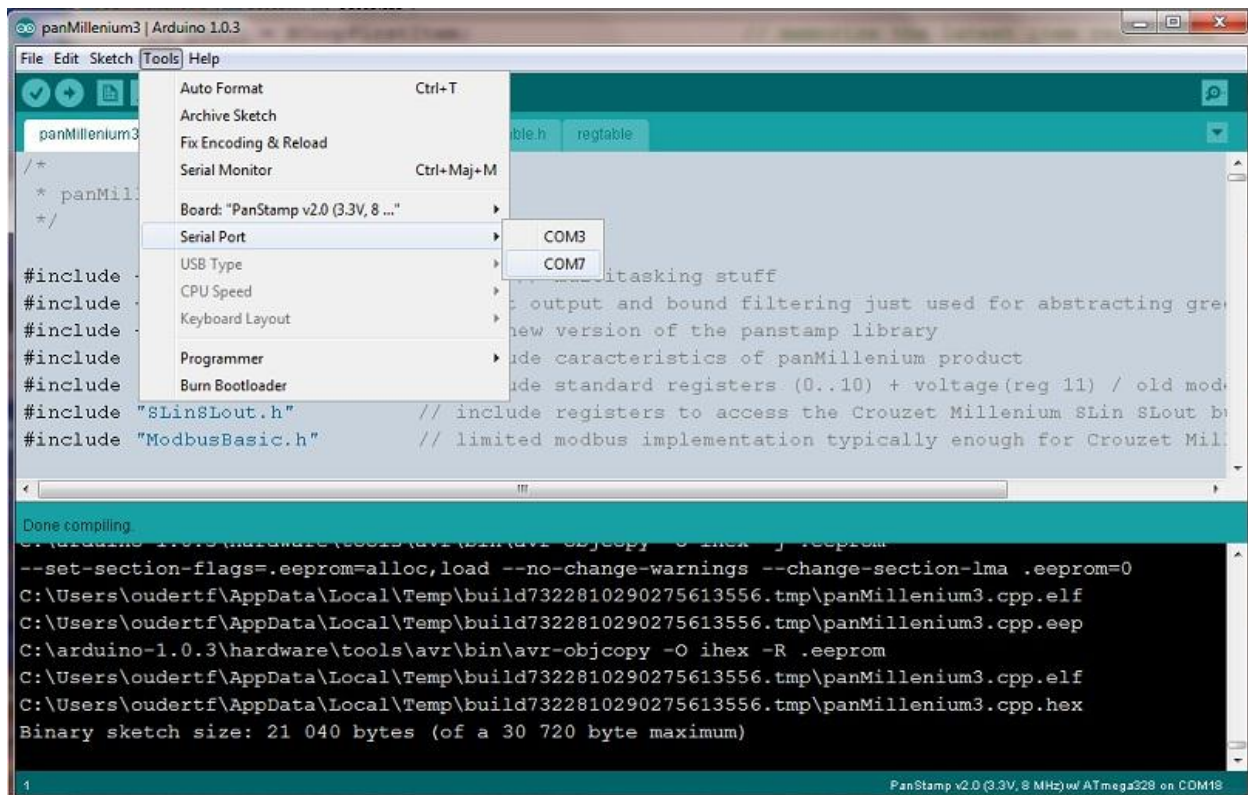
les circuits panstick et FTDI232 contiennent un composant d'interface USB/Serie qui est normalement reconnu automatiquement par les systèmes d'exploitation lorsque le connecteur USB est inséré. A défaut, installer le driver correspondant au circuit, pour que le convertisseur soit reconnu comme un port série COM. Vérifier dans le gestionnaire de périphérique qu'un port COM apparait bien.

Lancer le programme arduino v1.03. (conseils : brancher le convertisseur USB/Serie avant de lancer Arduino. Relancer aussi Arduino après toutes modifications dans le répertoire « arduino1.03\libraries ».)

Dans le menu outils, choisir la carte « panstamp 8mhz » et le port série correspondant au convertisseur.

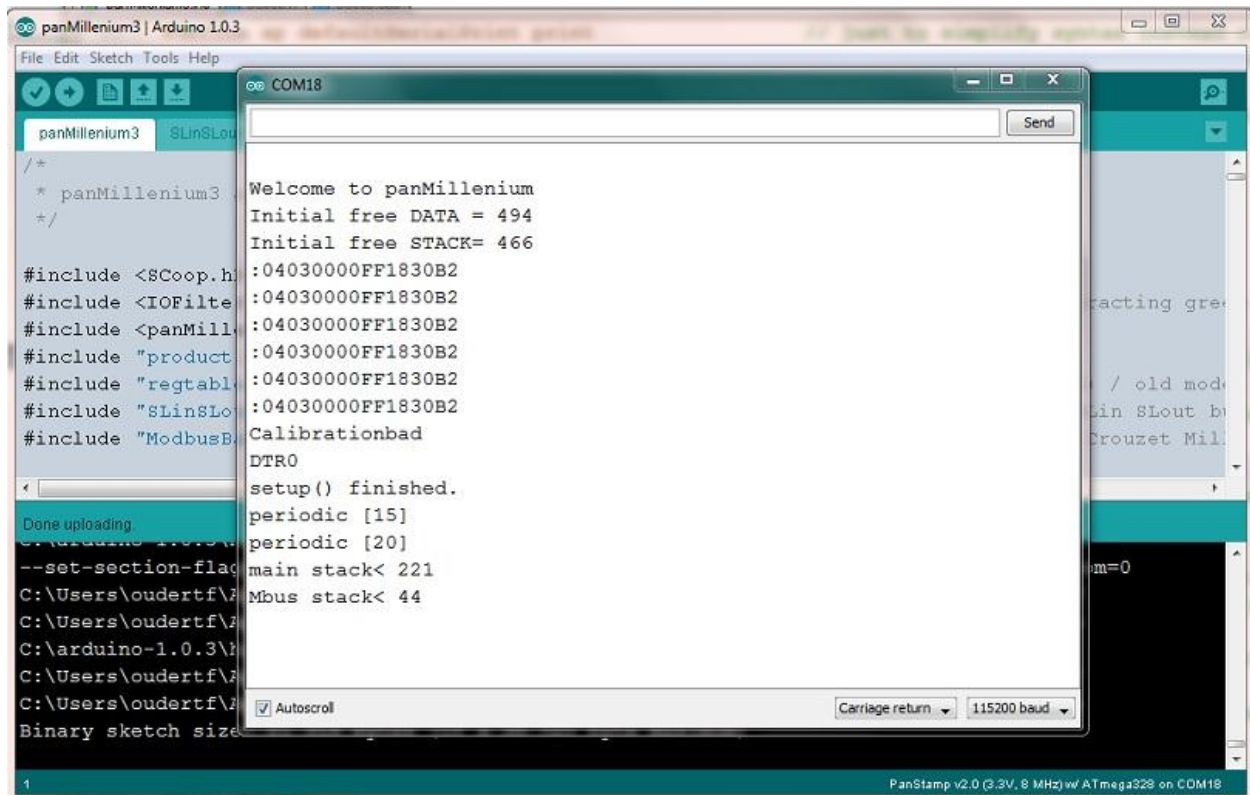
Charger l'application panMillenium3 par le menu « fichier\exemple\panMillenium »

Compiler, et télécharger.



Si l'opération s'est bien passée, la led verte du circuit panSTamp doit clignoter au début et ensuite rester allumé puisqu'il n'y a pas de communication modbus avec l'automate.

Pour vérifier le déroulement du programme, ouvrir le « moniteur série » configuré en 115200 bauds et vérifier les textes affichés (peuvent être différent de l'image ci-dessous) :



Le panMillenium est prêt à être débranché puis inséré dans l'automate (de préférence éteint au préalable pour minimiser les risques de faux contacts lors de l'insertion...).

Après quelques secondes d'initialisation et de recalibrage, la communication modbus est établie entre le panMillenium et l'automate. La led verte du circuit clignote rapidement, 5 fois par secondes.

Si elle reste allumée en permanence, il y a un problème de communication probablement lié à un défaut de calibration. Une procédure de calibration étendue sera disponible ultérieurement.

Application/modem, configuration à distance :

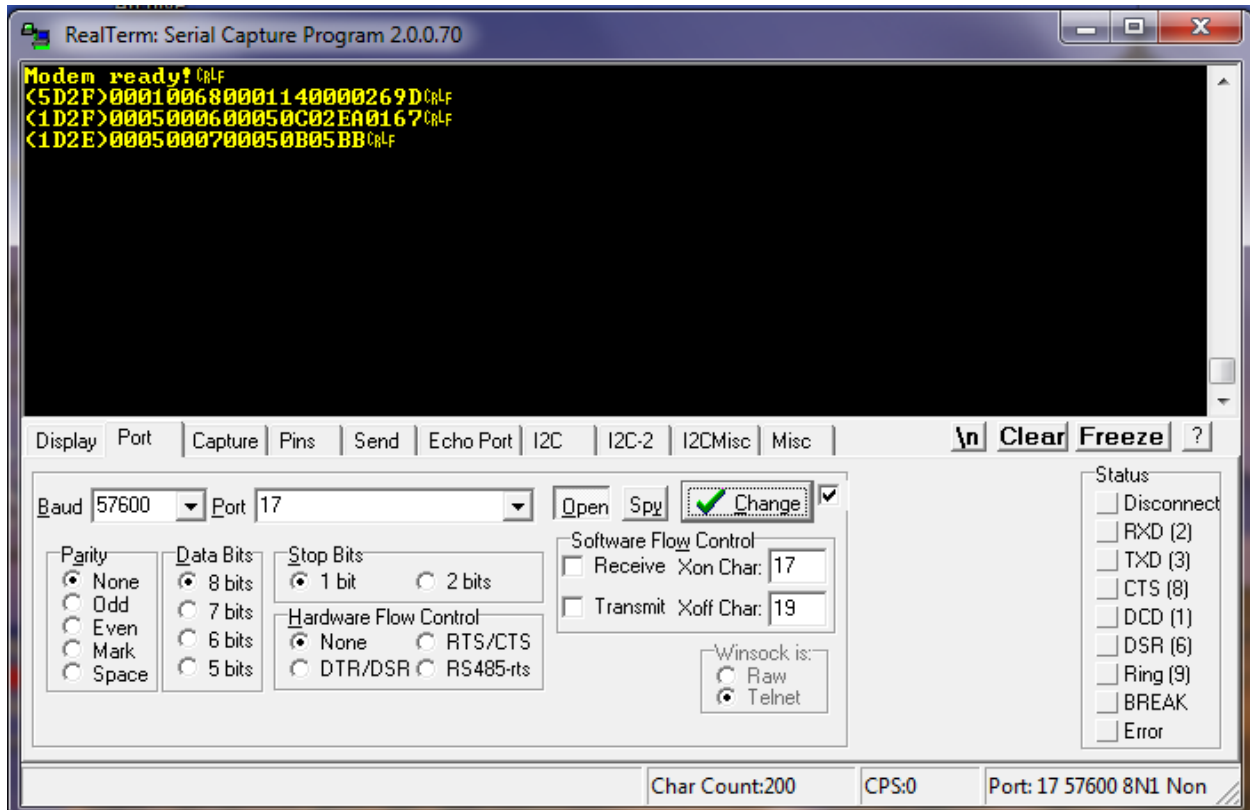
Pour pouvoir configurer le panMillenium à distance et pour vérifier que la communication est bien établie, il faut utiliser un second panMillenium relié à un convertisseur USB/Série, ou tout simplement un second panSTamp monté sur un panStick, et utiliser l'application « modemExtended » :

Dans Arduino, charger le fichier « exemple\panMillenium\modemExtended ».

Compiler ; éventuellement changer le port série pour communiquer avec le panstick ou le convertisseur USB/série. Télécharger l'application dans le panSTamp.

A ce stade il est préférable d'utiliser un logiciel « terminal série » par exemple « realterm »:

Lancer realterm :



Dans l'onglet *Display*, changer « rows » de 16 à 40 par exemple. Cocher Scrollback.

Dans l'onglet *Port* : Sélectionner 57600. Choisissez le port com correspondant au convertisseur USB ou au panStick. Cliquez le bouton « change ». Si le port n'apparaît pas, relancer realterm...

Instantanément le texte « Modem ready ! » doit apparaître dans la zone écran, et éventuellement les trames du protocole swap commencent à arriver, et s'affichent en hexadécimal.

Mode data :

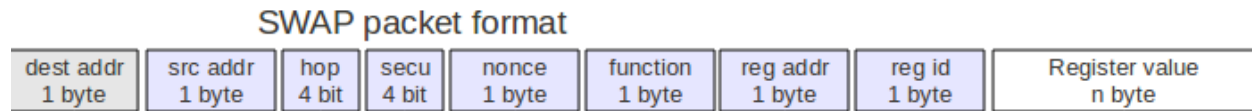
L'application modem fonctionne en mode « data » ou « commande » ou « mixed »

Dans le mode data, il est possible de taper directement (en aveugle) des chaînes hexadécimales qui seront envoyées sur le réseau swap après l'appui sur « enter ». Attention, un timeout de 1 seconde efface les précédents caractères tapés.

Dans ce mode, les trames reçues s'affichent au fil de l'eau.

Les 4 premiers caractères entre parenthèse représentent la force du signal pour l'émetteur de cette trame.

Les autres caractères représentent les éléments du protocole swap :



05 01 00 00 01 05 0B : permet de lire le registre 11 (0B) dans le point/nœud 5

05 01 00 00 02 05 0A 00 40 : permet d'écrire la valeur 64 (0040) dans le registre 10 (0A) « TXInterval»

Remarque : par défaut un nouveau point/nœud panMillenium aura l'adresse 255 (FF en hexadécimal) et donc probablement les chaines reçues commenceront par 00 FF.

De même pour le modem, qui devrait être à l'adresse 255 / FF au premier lancement.

Mode commande :

Pour entrer dans le mode commande, il faut taper successivement « +++ » sans appuyer sur « enter ».

(dans realterm, il faut d'abord positionner le curseur blanc clignotant à l'intérieur de l'écran noir en cliquant avec la souris...)

Après une seconde, le modem affiche « **OK-command mode** »

Dans ce mode il est possible de taper des commandes, commençant par les 2 lettres **AT**, qui permettront de lire l'état du modem ou de lire écrire des registres d'un panMillenium distant.

Commencer par activer l'écho des caractères tapés en tapant en aveugle « **ATE1** » « enter »

Lire le mot de synchronisation réseau et tapant : « **ATSW ?** » « enter »

La valeur **B547** devrait être retournée.

Vérifier l'adresse du modem sur le réseau avec : « **ATDA?** » « enter »

Choisissez de préférence l'adresse 1 pour le modem. Changer éventuellement la valeur en tapant : « **ATDA=1** » « enter »

Mode MIXED :

Pour pouvoir échanger des informations avec le panMillenium distant et voir les réponses renvoyées, il faut entrer dans le mode « mixed » en tapant : « **ATE2** » « enter »

Le modem affiche « **OK-command and data mode** » et un signe « > » s'affiche systématiquement en début de ligne. A partir de ce moment, toutes les trames reçues sont aussi affichées, et si une ligne de commande était en cours, elle est interrompue et recopiée sur la ligne inférieure.

Identifier les produits disponibles sur le réseau swap en tapant : « **ATQRY** » « enter »

Une réponse apparaît du genre :

```
>ATQRY00000CrLf
> QRY node=0 0 CRLF
>(5B33)>00 FF 00 0F 00 FF 00 00000000100000001 CRLF
>
```

La ligne 3B représente le registre « 59 » en décimal, qui est attribué à SLout26.

Commandes complémentaires utiles :

Lecture d'un registre « ATQRY XX YY » « enter »

XX correspondent à l'adresse du panMillenium, YY correspond à l'identifiant du registre « ID »

Les nombre doivent être écrit en hexadécimal. Un espace de séparation peut être présent mais n'est pas nécessaire. Un nombre décimal peut être saisi, en le précédant du symbole « % ». Il peut être utile d'insérer un espace pour faciliter la lecture/écriture de la commande, ou pour enlever toute ambiguïté si plusieurs chiffres se succèdent indépendamment les uns des autres.

Exemple :

Lecture du registre SLout26 : « **ATQRY 02 %59** » « enter »

Lecture du deuxième bit du mot SLin1 : « **ATQRY 02 %56** » « enter »

Lecture des 8 mots SLout25 à SLout 32 : « **ATQRY 02 0F** » « enter »

Ecriture d'un registre, octet par octet « ATCMD XX YY Zn..Zm » « enter »

XX correspondent à l'adresse du panMillenium, YY correspond à l'identifiant du registre « ID » Zn..Zm est une liste de valeur en hexadécimale correspondant impérativement à la taille du registre cible.

Exemple :

Ecriture de la valeur 1 dans le registre SLin1 : « **ATCMD 02 %18 00 01** » « enter »

Ecriture de la valeur 257 dans le registre SLin2 : « **ATCMD 02 13 01 02** » « enter »

Ecriture à 1 du troisième bit du registre SLin1 : « **ATCMD 02 %55 01** » « enter »

Ecriture d'un seul octet (0..255) dans un registre « byte » « **ATCMDDB XX YY ZZ** »

Ecriture d'un seul mot (0..65535) dans un registre « word » « **ATCMDW XX YY ZZZZ** »

Ecriture d'un seul mot 32 bits (0..2³²-1) dans un registre « long » « **ATCMDL XX YY ZZZZZZZZ** »

Liste des registres reconnus et mis à disposition par l'application panMillenium

description	Registre décimal(hexa)	Taille	Lecture/ecriture
Groupe SLin1 à SLin8	12 (0C)	8 mots = 16 octets	Lecture/ecriture
Groupe SLin9 à 16	13 (0D)	8 mots = 16 octets	Lecture/ecriture
Groupe SLin17 à 24	14 (0E)	8 mots = 16 octets	Lecture/ecriture
Groupe SLout25 à 32	15 (0F)	8 mots = 16 octets	Lecture seule
Groupe SLout 33 à 40	16 (10)	8 mots = 16 octets	Lecture seule
Groupe SLout 41 à 48	17 (11)	8 mots = 16 octets	Lecture seule

Registre SLin1 seul	18 (12)	1 mot = 2 octets	Lecture/ecriture
Registre SLin2 seul	19 (13)	1 mot = 2 octets	Lecture/ecriture
Registre SLin3 seul	20 (14)	1 mot = 2 octets	Lecture/ecriture
Registre SLin4 seul	21 (15)	1 mot = 2 octets	Lecture/ecriture
Registre SLin5 seul	22 (16)	1 mot = 2 octets	Lecture/ecriture
Registre SLin6 seul	23 (17)	1 mot = 2 octets	Lecture/ecriture
Registre SLin7 seul	24 (18)	1 mot = 2 octets	Lecture/ecriture
Registre SLin8 seul	25 (19)	1 mot = 2 octets	Lecture/ecriture
Registre SLin9 seul	26 (1A)	1 mot = 2 octets	Lecture/ecriture
Registre SLin10 seul	27 (1B)	1 mot = 2 octets	Lecture/ecriture
Registre SLin11 seul	28 (1C)	1 mot = 2 octets	Lecture/ecriture
Registre SLin12 seul	29 (1D)	1 mot = 2 octets	Lecture/ecriture
Registre SLin13 seul	30 (1E)	1 mot = 2 octets	Lecture/ecriture
Registre SLin14 seul	31 (1F)	1 mot = 2 octets	Lecture/ecriture
Registre SLin15 seul	32 (20)	1 mot = 2 octets	Lecture/ecriture
Registre SLin16 seul	33 (21)	1 mot = 2 octets	Lecture/ecriture
Registre SLin17 seul	34 (22)	1 mot = 2 octets	Lecture/ecriture
Registre SLin18 seul	35 (23)	1 mot = 2 octets	Lecture/ecriture
Registre SLin19 seul	36 (24)	1 mot = 2 octets	Lecture/ecriture
Registre SLin20 seul	37 (25)	1 mot = 2 octets	Lecture/ecriture
Registre SLin21 seul	38 (26)	1 mot = 2 octets	Lecture/ecriture
Registre SLin22 seul	39 (27)	1 mot = 2 octets	Lecture/ecriture
Registre SLin23 seul	40 (28)	1 mot = 2 octets	Lecture/ecriture
Registre SLin24 seul	41 (29)	1 mot = 2 octets	Lecture/ecriture
Registre Bit 16 de SLin1	42 (2A)	1 octet	Lecture/ecriture
Registre Bit 15 de SLin1	43 (2B)	1 octet	Lecture/ecriture
Registre Bit 14 de SLin1	44 (2C)	1 octet	Lecture/ecriture
Registre Bit 13 de SLin1	45 (2D)	1 octet	Lecture/ecriture
Registre Bit 12 de SLin1	46 (2E)	1 octet	Lecture/ecriture
Registre Bit 11 de SLin1	47 (2F)	1 octet	Lecture/ecriture
Registre Bit 10 de SLin1	48 (30)	1 octet	Lecture/ecriture
Registre Bit 9 de SLin1	49 (31)	1 octet	Lecture/ecriture
Registre Bit 8 de SLin1	50 (32)	1 octet	Lecture/ecriture
Registre Bit 7 de SLin1	51 (33)	1 octet	Lecture/ecriture
Registre Bit 6 de SLin1	52 (34)	1 octet	Lecture/ecriture
Registre Bit 5 de SLin1	53 (35)	1 octet	Lecture/ecriture
Registre Bit 4 de SLin1	54 (36)	1 octet	Lecture/ecriture
Registre Bit 3 de SLin1	55 (37)	1 octet	Lecture/ecriture
Registre Bit 2 de SLin1	56 (38)	1 octet	Lecture/ecriture
Registre Bit 1 de SLin1	57 (39)	1 octet	Lecture/ecriture
Registre SOut25	58 (3A)	1 mot = 2 octets	Lecture seule
Registre SOut26	59 (3B)	1 mot = 2 octets	Lecture seule
Registre Listener1	61 (3D)	1 quartet = 4 octet	Lecture/ecriture
Registre Listener2	62 (3E)	1 quartet = 4 octet	Lecture/ecriture
Registre Listener3	63 (3F)	1 quartet = 4 octet	Lecture/ecriture
Registre Listener4	64 (40)	1 quartet = 4 octet	Lecture/ecriture

Registre Listener5	65 (41)	1 quartet = 4 octet	Lecture/ecriture
Registre Listener6	66 (42)	1 quartet = 4 octet	Lecture/ecriture
Registre Listener7	67 (43)	1 quartet = 4 octet	Lecture/ecriture
Registre Listener8	68 (44)	1 quartet = 4 octet	Lecture/ecriture
Registre Listener9	69 (45)	1 quartet = 4 octet	Lecture/ecriture
Registre Broadcast1	71 (47)	1 quartet = 4 octet	Lecture/ecriture
Registre Broadcast2	72 (48)	1 quartet = 4 octet	Lecture/ecriture
Registre Broadcast3	73 (49)	1 quartet = 4 octet	Lecture/ecriture
Registre Broadcast4	74 (4A)	1 quartet = 4 octet	Lecture/ecriture
Registre Broadcast5	75 (4B)	1 quartet = 4 octet	Lecture/ecriture
Registre Broadcast6	76 (4C)	1 quartet = 4 octet	Lecture/ecriture
Registre Broadcast7	77 (4D)	1 quartet = 4 octet	Lecture/ecriture
Registre Broadcast8	78 (4E)	1 quartet = 4 octet	Lecture/ecriture
Registre Broadcast9	79 (4F)	1 quartet = 4 octet	Lecture/ecriture

Fonctionnement des registres « listener »

Ces registres permettent de définir une configuration d'écoute du réseau et de recopier une valeur d'un registre d'un point/nœud swap, vers un registre du panMillenium.

Un registre distant est identifié par l'adresse du point/nœud qui l'émet, et son identifiant « ID ».

Un registre distant peut être plus long que le registre cible, il faudra donc préciser un « offset » en octet qui définira l'index dans le registre source pour commencer la recopie vers le registre cible.

Jusqu'à 9 registres d'écoute sont supporté dans le model et dans l'application panMillenium. Pour des raisons de taille mémoire, la version v1.1 ne supporte que 4 registres. Ceci est modifiable en recompilant l'application panMillenium3 après avoir modifié la structure du fichier « SLinSLout.h »

Exemple :

Utilisation du listener1 pour recopier la température envoyée par le point panSTamp « tempHum » ayant l'adresse 05, et se trouvant au début de son registre 12 (offset 0), dans le registre SLin6 du panMillenium se trouvant à l'adresse 02 : « **ATCMD 02 %61 05 %12 00 %23** »

Utilisation du listener2 pour recopier l'humidité envoyée par le point panSTamp « tempHum » ayant l'adresse 05, et se trouvant en 3eme position de son registre 12 (offset 2), dans le registre SLin7 du panMillenium se trouvant à l'adresse 02 : « **ATCMD 02 %62 05 %12 02 %25** »

Utilisation du listener3 pour recopier le registre SLout25 envoyée par le panMillenium ayant l'adresse 03, dans le registre SLin8 du panMillenium ayant l'adresse 02 : « **ATCMD 02 %63 03 %58 00 %26** »

Utilisation du listener4 pour recopier le registre SLout27 envoyée par le panMillenium ayant l'adresse 03, dans le registre SLin9 du panMillenium ayant l'adresse 02 : « **ATCMD 02 %64 03 %15 04 %27** »

Fonctionnement des registres « Broadcast »

Ces registres permettent de définir une configuration de transmission automatique de registres en fonction du temps (périodiques), ou de changement de valeur ou de bits, selon 3 modes principaux :

« **mode 192 (C0)** » Transmission périodique : possibilité de transmettre un registre complet toute les XX secondes ($1..65535$) = 1 seconde à 18 heures environ.

« **mode 64 (40)** » Transmission sur changement d'état d'un ou plusieurs bit dans un registre « octet » ou « mot », avec un masque définit sur 16bits en fonction des bits à prendre en compte pour la comparaison avec l'état antérieur.

« **mode 128 (80)** » Transmission sur changement de valeur d'un registre octet ou mot, autour d'un seuil définit sur 16 bits, en valeur absolue.

Une confirmation du temps de changement d'état est paramétrable pour les modes 64 et 128 : possibilité de définir un temps nécessaire pour la prise en compte du changement, de 0.5secondes à 16heures, en ajoutant une valeur comprise entre 0 et 15 au mode, selon la table ci-dessous :

0	0.5sec	4	10sec	8	5min	12	2h
1	1sec	5	30sec	9	10min	13	4h
2	2sec	6	1min	10	30min	14	8h
3	5sec	7	2min	11	1h	15	16h

Il est aussi possible d'ajouter une demande de répétition de paquet 1 seconde après que le changement d'état ait été communiqué, en ajoutant 16 (10) au mode :

Exemple :

Transmission sur changement d'état binaire, confirmé pendant 2 secondes : mode = $64+2=66$

Transmission sur changement d'état binaire confirmé pendant 10secondes avec répétition du paquet : mode = $64+16+4 = 84$

Transmission sur changement de la valeur, confirmée sur 1 minute, avec répétition du paquet : mode = $128 + 16 + 6 = 150$

Exemple de construction de registre de broadcast :

Broadcast1 : Transmission toute les 5 secondes du registre SLout25: ATCMD 02 %71 %58 C0 0005

Broadcast2 : transmission sur changement de valeur du registre SLout26 à +/-3 digit, confirmée au bout de 10 secondes, avec répétition du paquet envoyé : ATCMD 02 %72 %59 94 0003

Broadcast3 : transmission du registre SLout25 sur changement du 10ieme bit confirmé après 1 seconde : ATCMD 02 %73 %58 41 0200

D'autres Listener et Broadcast peuvent être définis « en dur » dans l'application panMillenium en modifiant les tables « tabRegListen » et « tabBroadcast » avec la même codification.

Limite mémoire :

Il est possible de supprimer ou créer d'autres registres, dans les limites de la taille mémoire totale disponible dans le circuit atmega328p. Les modifications doivent être faites exclusivement dans le fichier SLinSLout.h et éventuellement SLinSLout.ino si les registres ont un fonctionnement spécifique.

Pour dépister les problèmes de mémoire, il est recommandé de laisser le panMillenium branché au PC et de lire les messages envoyés sur le port série à l'aide d'un realterm configuré en 115200 bauds.

La taille mémoire totale disponible au lancement de l'application s'affiche dans les 2 premières lignes d'information, ensuite, la taille restante dans la pile de la tâche modbus et dans la pile de la boucle principale s'affiche au fur et à mesure et la valeur réduit en fonction des trames et transactions à effectuer.... Donc il faut « solliciter » le programme avec des QRY et CMD et attendre que les valeurs de piles semblent stabilisées...

Dans tous les cas il est nécessaire de garder une sécurité minimale de 20 octet pour chaque « pile ».

Utilisation avec Swapdmt :

Swapdmt est un utilitaire écrit par l'équipe des développeurs du panSTamp. Il permet de « sniffer » un réseau swap et d'afficher tous les points/nœuds, en reconnaissant les points par leur numéros de produit, et leur fichier de description de registre au format xml associé.

Le fichier XML correspondant à la définition des registres gérés par le panMillenium s'appelle panMillenium1.xml. Il devrait être placé dans le répertoire config/device/panMillenium ou se trouve l'application swapdmt.

Pour être reconnu, le développeur « 29 » panMillenium doit être déclaré dans le fichier device.xml qui se trouve dans le répertoire config/device. Utiliser éventuellement le fichier fourni dans panMillenium.zip.

Utilisation avec smartapp :

L'application modemExtended ne supporte pas encore la reconnaissance des requêtes modbus Maitres venant de l'application smartapp. L'implémentation d'un mécanisme de bridge entre smartapp et le réseau swap est prévue et sera implémenté dans modemExtended ultérieurement.

Utilisation avec le logiciel Crouzet Logic.

Le logiciel Crouzet de programmation et de monitoring des automates Millenium utilise aussi un protocole modbus maitre, avec des registres et adresses complémentaires. Ces requêtes ne sont pas reconnues par l'application ModemExtended. Il est prévu dans un développement ultérieur pour transporter ces requêtes modbus entre le panMillenium et l'application ModemExtended, en les encapsulant dans des registres swap spécifique, avec un mécanisme « d'acknowledgement » pour garantir l'intégrité des transmissions. Ce développement est assez compliqué et devrait être réalisé, mais n'est pas encore planifié.

FIN DE LA DOCUMENTATION V1.1