UTILISATION DE LOGIPIC V2
LE BUS I2C POUR LES MÉMOIRES

Objectif : Il s'agit d'utiliser le bus I2C pour contrôler les mémoires 24Cxx.

1 - LA CIBLE

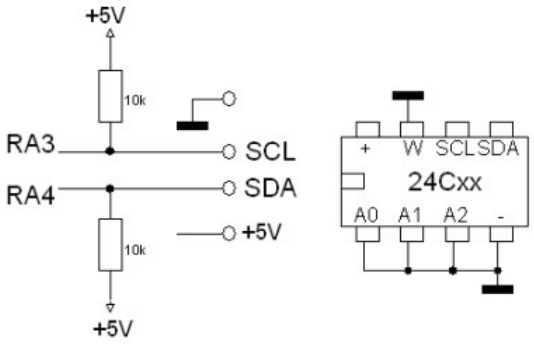
Les programmes sont réalisés sur la platine BootRS232. Cette cible est constituée par un microcontroleur 16F876A avec un oscillateur interne à 4MHz. Pour fonctionner en mode bootstrap, le pic devra être programmé une première fois avec un bootloader (16F876A_4M.hex), le test des programmes s'effectuera avec Tinybootloader dont l'appel est possible dans la version 2.07 de Logipic.

L'ensemble des explications ne s'attardera pas sur la prise en main de Logipic considérant que vous connaissez suffisamment ce formidable logiciel.

SCHEMA STRUCTUREL SIMPLIFIE

Le principe de lecture/écriture sur une liaison I2C rend indispensable l'utilisation pour SDA d'une ligne à collecteur ouvert. Il existe sur le 16F876 (ou le 16F628), une seule ligne possible : c'est RA4. Cette ligne est donc utilisée pour envoyer ou recevoir des données, elle alternera entre sortie et entrée grâce à la gestion de TRISA,4. La platine BootRS232 utilise pour SCL la ligne RA3, cependant il serait possible d'utiliser n'importe quelle autre sortie pour votre projet personnel...

N'importe quel périphérique I2C pourra être raccorder sur ce bus, ce coach ne traite que les mémoires.



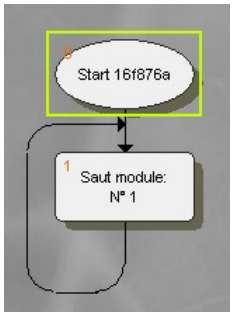
Dans cet exemple, les bits A0, A1 et A2 sont à 0. L'adresse fixe des mémoires est 1010 de A7 à A3 soit 1010 0000(B) ou A0(H)

2 - ECRITURE DES ROUTINES I2C

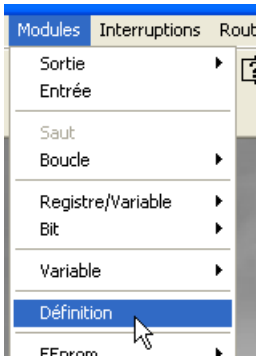
L'utilisation du bus I2C est expliqué dans le coach1, je ne reviendrais pas sur les explications indispensables à la compréhension pour l'écriture des routines.

La grande nouveauté c'est l'utilisation de la fonction Modules Définition qui permet de définir un nom facile à retenir comme SCL ou SDA au lieu de PORTA,3 ou PORTA,4.

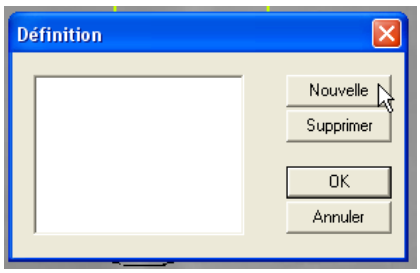
Mettre ne route Logipic, choisir votre circuit : dans mon exemple le 16f876A



puis Modules Définition

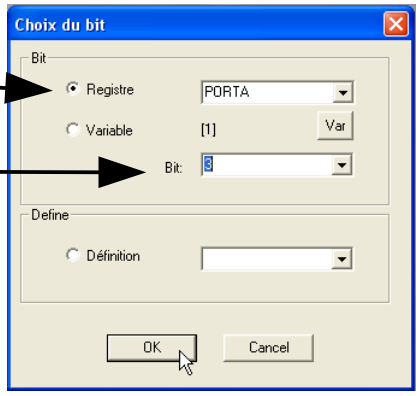


puis Nouvelle



choisir le Registre PORTA

Bit 3

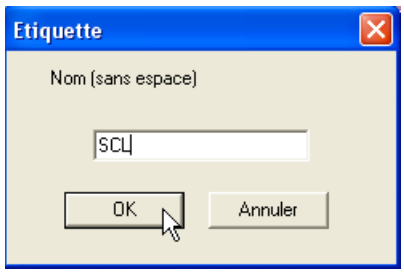


puis OK

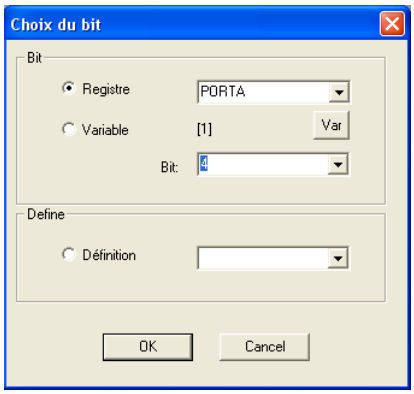
Choisir un nom d'étiquette

choisir (judicieusement) SCL

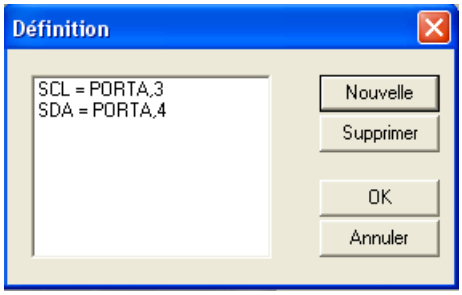
puis OK



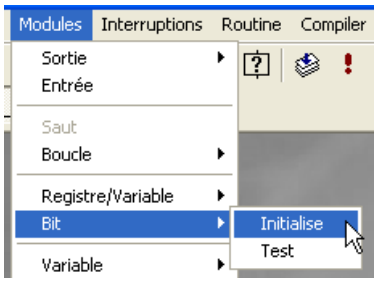
Faire de même avec PORTA,4 et donner le nom SDA.



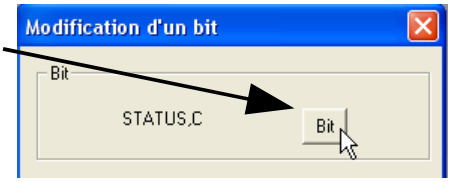
SCL et SDA sont maintenant définis :



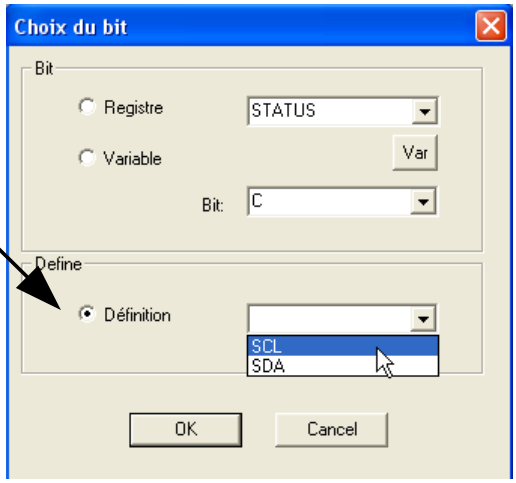
La routine Start_i2c est maintenant très facile à écrire avec Logipic :
Aller chercher Modules > Bit > Initialise



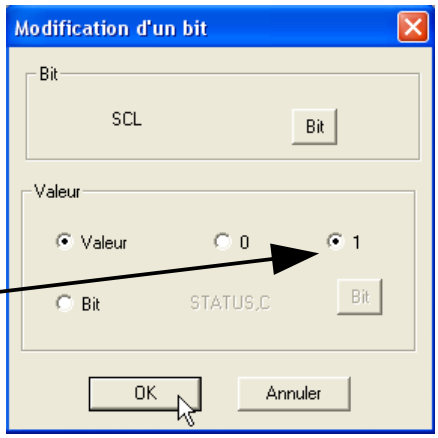
Demander Bit



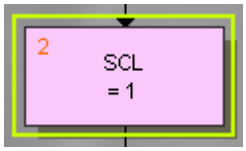
puis Define
Définition
choisir SCL



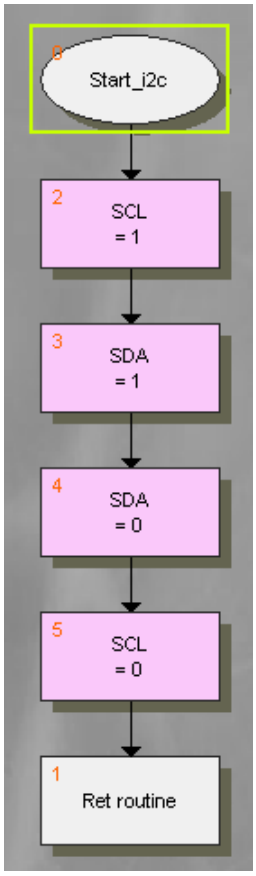
mettre
à 1



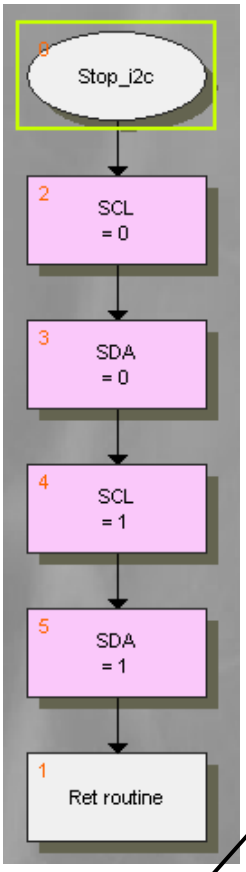
L'écriture est très claire :



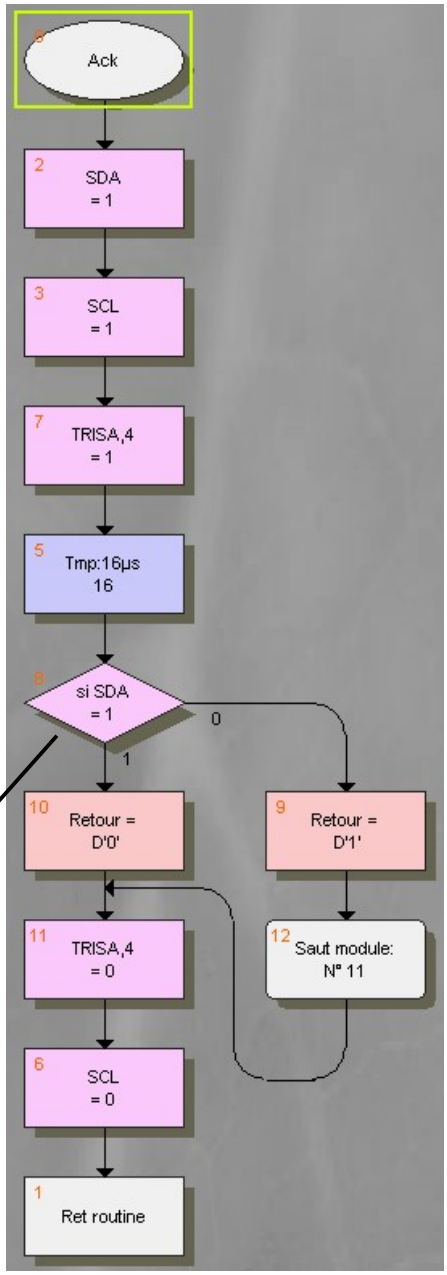
La routine Start_i2c



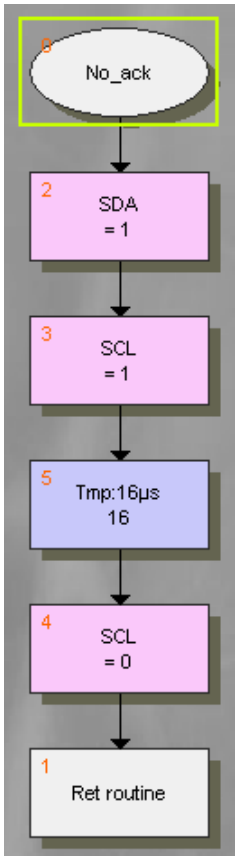
La routine Stop_i2c



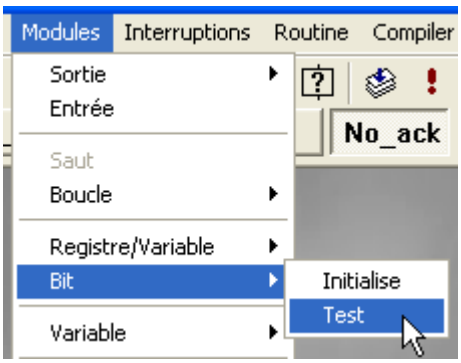
La routine Ack



La routine No_ack



Ce test s'écrit grâce à
Modules > Bit > Test



N'oubliez de définir la variable
Retour
et la temporisation de
16µs

La transmission d'une donnée

La réception d'un octet en série s'effectue grâce à la routine TX_i2c :
Il est nécessaire de déclarer les variables suivantes :

Compt
Dout **c'est la variable à transmettre...**
Valbit
Bit

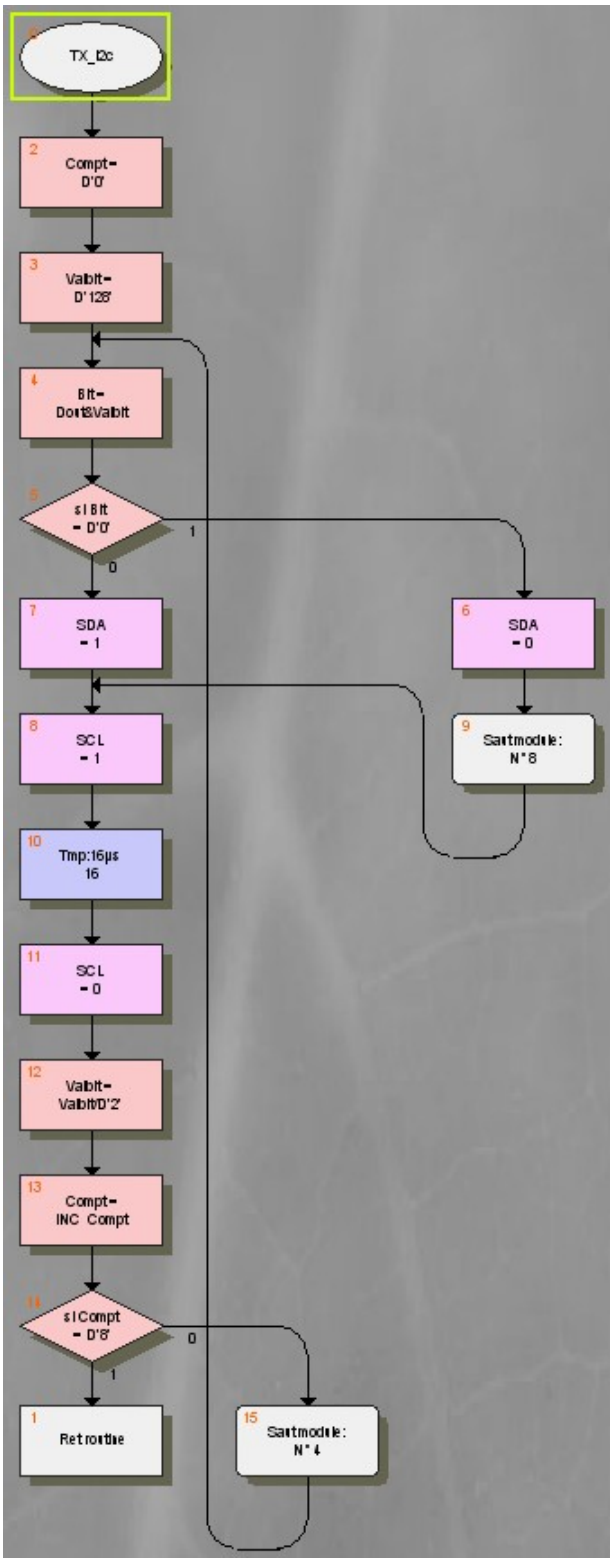
La réception d'une donnée

La réception d'un octet en série s'effectue grâce à la routine RX_i2c :
cette routine utilise les mêmes variables que la routine TX_i2c plus la variable :

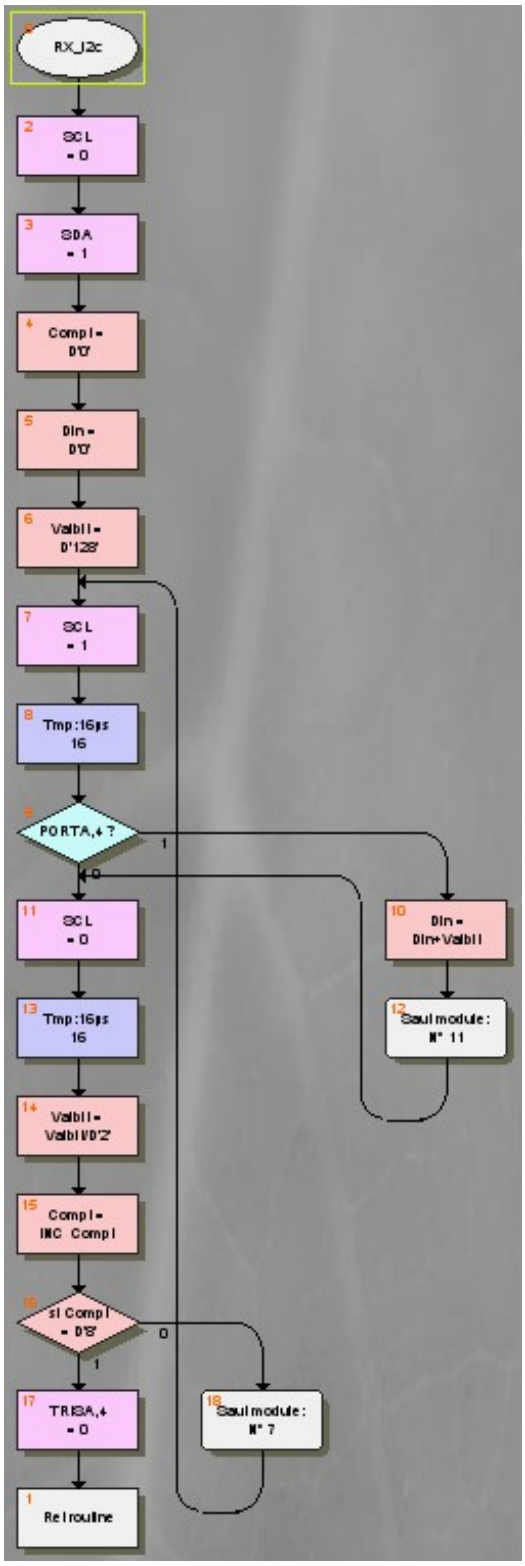
Din **c'est la variable lue...**

Je ne reviens pas sur la routine Test_ret qui n'appelle pas de commentaires.

TX_i2c



RX_i2c



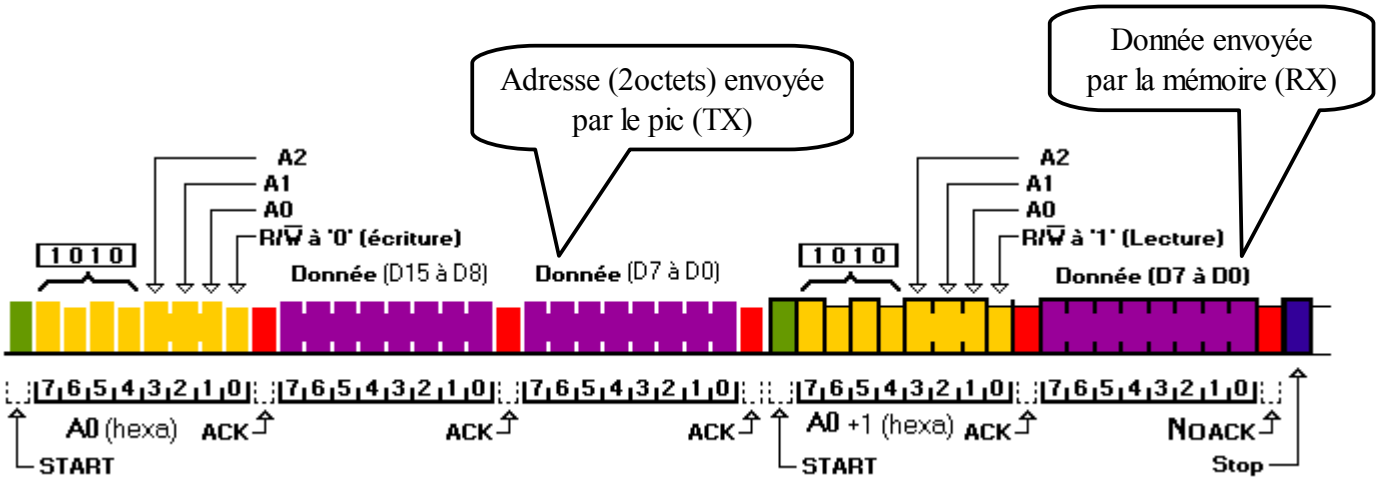
3 - MISE EN OEUVRE DES MEMOIRES 24Cxx

La mémoire 24C32 est capable de stocker 32768 bits, les informations sont stockées sous forme d'octets (un octet = 8 bits), chaque octet de la mémoire est défini par une adresse dont la taille est imposée par la capacité de la mémoire.

Pour une mémoire 24C32, l'adresse est définie par 2 octets (c'est à dire 16 bits), la lecture de la documentation technique (datasheet) est indispensable pour pouvoir exploiter d'autres mémoires...

Les routines nécessaires sont regroupées dans l'export : mem.rtn.
Le projet lecture permet de lire la mémoire 24C32.

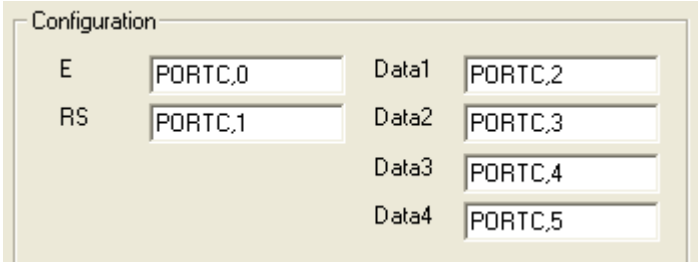
La documentation du 24C32 précise le protocole suivant pour la lecture :



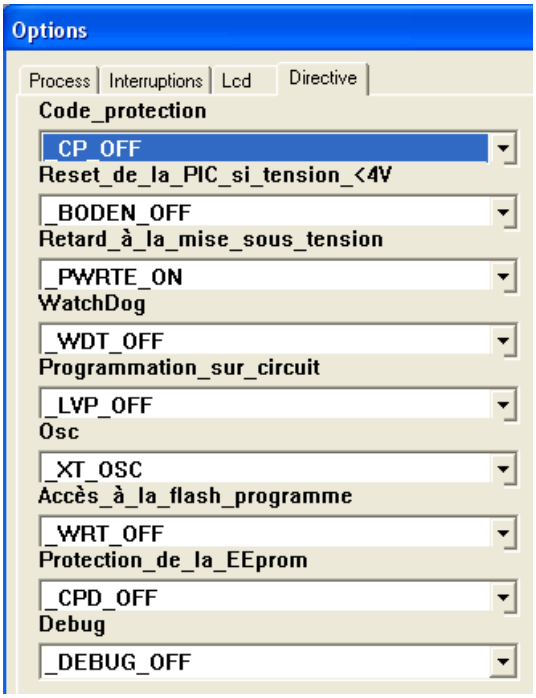
L'écriture du programme principal est maintenant très simple. Pour simplifier le programme, la lecture est limitée aux 255 premières adresses. L'afficheur LCD indique l'adresse en ligne 1 et la donnée en ligne 2.

Il est possible de lire plusieurs données les unes après les autres, dans ce cas chaque donnée est séparée par un Ack et seule la dernière donnée se termine par un No_ack. Pour plus de détails, il faut lire le datasheet et faire ses propres essais...

N'oubliez pas de configurer la configuration de l'afficheur LCD

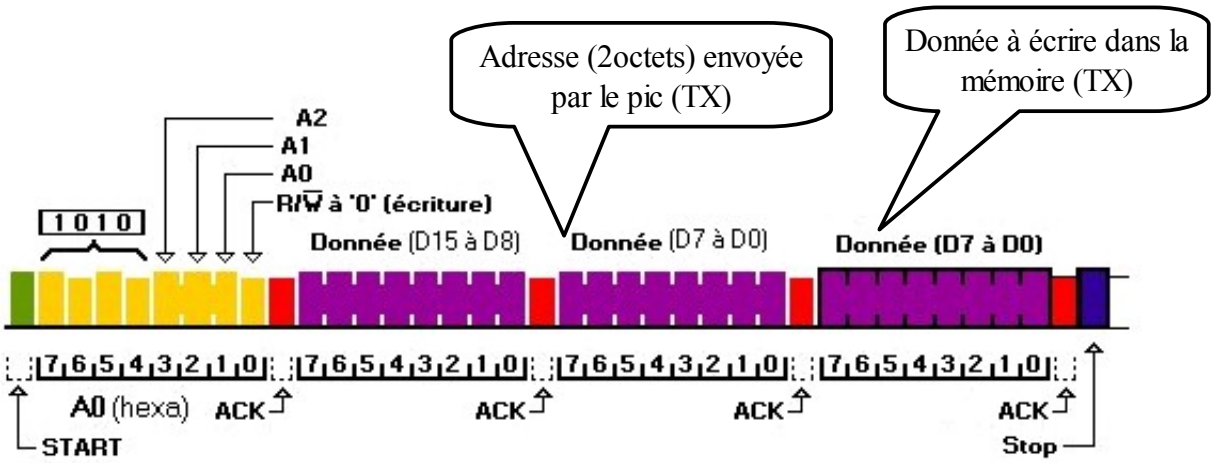


Et les bonnes directives :



Le projet écriture permet d'écrire dans la mémoire 24C32.

La documentation du 24C32 précise le protocole suivant pour l'écriture :



Le programme d'écriture ne sert à rien d'autres que d'écrire des données (255, 127, 63 ...) sur les 8 premières adresses puis des 0. Le but est essentiellement didactique mais nul doute que vous trouverez des applications pour utiliser cette mémoire de masse. Encore une fois la lecture du datasheet vous sera une aide préciseuse.