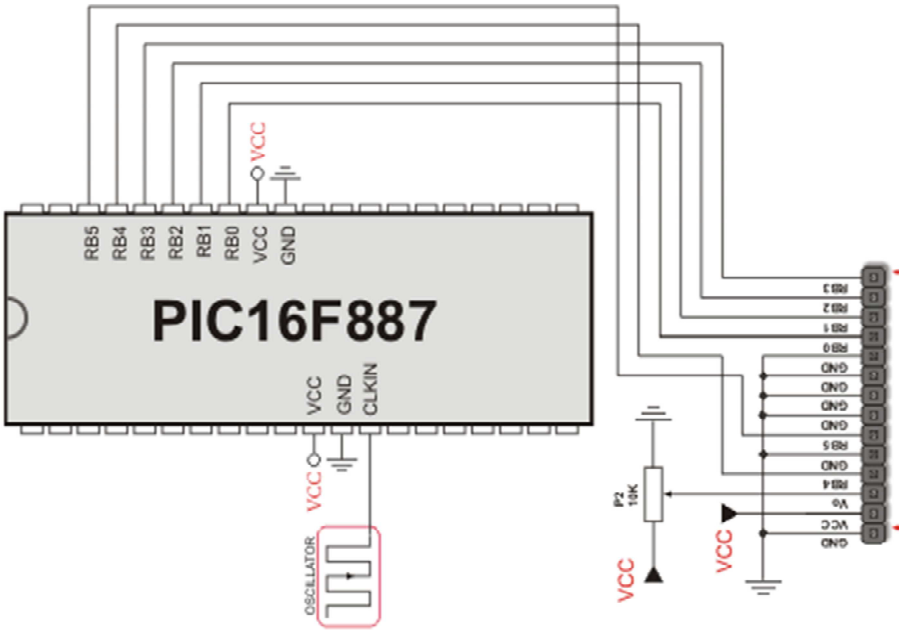
الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
وزارة التعليم العالي و البحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Aboubakr BELKAÏD – Tlemcen –
جامعة أبي بكر بلقايد – تلمسان –
Faculté de Technologie
كلية التكنولوجيا

Support de Travaux Pratiques

Microprocesseurs et Microcontrôleurs

(Programmation en mikroC. Application pour les microcontrôleurs de la famille PIC)

Filière de Génie Biomédical



Elaboré par :

Dr HAMZA CHERIF Lotfi

Année universitaire : 2019 – 2020

Table de matière

Avant-propos

Les microprocesseurs constituent le cœur de presque toutes les réalisations électroniques ; on en trouve dans tous les domaines, notamment : l'informatique (de la calculatrice à l'ordinateur), l'automobile (ABS, injection, ...), l'automatique (automates programmables, contrôle de processus, ...), l'électronique domestique (thermomètre, télécommande, carte à puce, ...)

Les performances des microprocesseurs sont liées aux possibilités offertes par la technologie, en terme de capacité (nombre de portes logiques intégrées) et de vitesse, et au choix d'architectures adaptées (ou imposées pour cause de compatibilité ascendante); à l'heure actuelle, on trouve sur le marché des microprocesseurs intégrant des millions de transistors, fonctionnant à plus de 3000 MHz et disposés dans des boîtiers de plusieurs centaines de broches, ils sont issus de différentes approches architecturales : CISC, RISC, DSP et VLIW.

En effet, la gestion des ressources internes d'un microcontrôleur impose de faire appel à de nombreuses fonctions nouvelles, spécifiques de ce type de circuit. En outre, si vous voulez pouvoir mettre au point votre application avec un maximum de chances de succès, le recours à un environnement de développement spécialisé, allant de l'éditeur de programme au simulateur en passant par le compilateur, est nécessaire.

Enfin, lorsque votre programme est prêt à être essayé en « vrai grandeur », il faut encore savoir comment le charger en mémoire du microcontrôleur.

Ce document constitue un support de Travaux Pratiques qui se propose de répondre à toutes ces questions avec des exemples d'applications sur des microcontrôleurs de type PIC en langage C. Ce document s'adresse aux étudiants Master1 (Master en instrumentation Biomédicale, département de Génie Biomédical, Faculté de technologie, Université de Tlemcen.

1. Introduction

Pour doter l'étudiant d'outils, lui permettant une révision et une assimilation de notions de base sur la programmation des Microcontrôleurs de type Pic :

Le compilateur **mikroC pour PIC Version 6** est présenté en premier lieu ; ce dernier a trouvé une large application pour le développement de systèmes embarqués sur la base de microcontrôleur. Il assure une combinaison de l'environnement de programmation avancée IDE (Integrated Development Environment), et d'un vaste ensemble de bibliothèques pour le matériel, de la documentation complète et d'un grand nombre des exemples.

Le logiciel de simulation **ISIS-PROTEUS Version 8** est présenté en deuxième lieu, ce dernier est largement utilisé ces dernières années comme outil pédagogique, en particulier pour sa simplicité due à son environnement graphique et interactif. Cet outil facilitera, par la suite la tache de l'étudiant durant la réalisation de ses projets.

Ces des outils des deux logiciels sont décrits au fur et à mesure, via une série d'exemples et d'applications, couvrant différents domaines de l'utilisations des Pic :

Certaines applications sont validées par des tests expérimentaux effectuées au niveau du laboratoire, afin de doter l'étudiant d'un esprit critique lui permettant d'analyser et de comparer les résultats obtenus, théoriquement, expérimentalement, et par simulation informatique.

2. Langage et compilateur

La programmation des microcontrôleurs se fait naturellement en langage assembleur. **Microchip** propose **MPLAB IDE**, un environnement de développement performant, téléchargeable gratuitement. Il permet d'éditer le code source (sous la forme d'un fichier texte avec extension **.asm**), de le simuler et de le débugger. Le compilateur fournit le code objet (fichier avec **extension**.**HEX**).

Notez que l'on peut aussi programmer les microcontrôleurs en :

- Langage C (*mikroC*) Utiliser dans Notre cas ;
- Pascal (*Pic Micro Pascal* ; *mikroPascal*) ;
- Basic (*Proton DS* ; *mikroBasic*).

Il existe aussi des langages de programmation graphique :

- *Flowcode* et *Niple*.

2.1 La carte de programmation (hardware)

Il existe des cartes de programmation qui se branchent sur le port parallèle, le port série ou bien le port USB d'un ordinateur. Plusieurs marque existe sur le marché, chaque société propose des cartes avec un logiciel pilote.

2.2 Le programmateur (logiciel)

IC-Prog (logiciel freeware développé par Bonny Gijzen), Serial Bootloader AN1310 (Microchip) et mikroProg Suite For PIC (Mikroelectronika) permettent le transfert du code objet (**fichier.HEX**) dans la mémoire Flash du microcontrôleur.

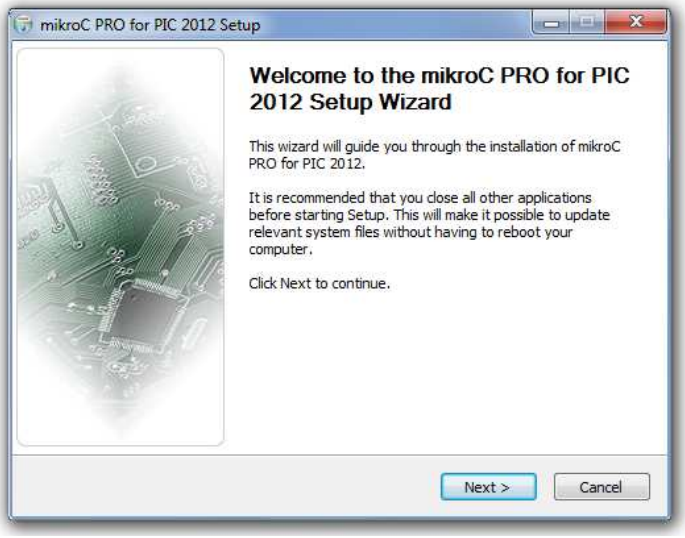
3. MikroC Pro for PIC

A chaque nouvelle version de **mikroC Pro for Pic** de très nombreuses améliorations du compilateur mikroC sont introduite : nouvelles variables utilisables, nouvelle interface IDE, amélioration des performances du linker et de l'optimisateur, cycle de compilation plus rapide, code machine généré plus compact, nouveaux PIC supportés, environnement de développement encore plus ergonomique, nouveaux exemples d'applications, etc...

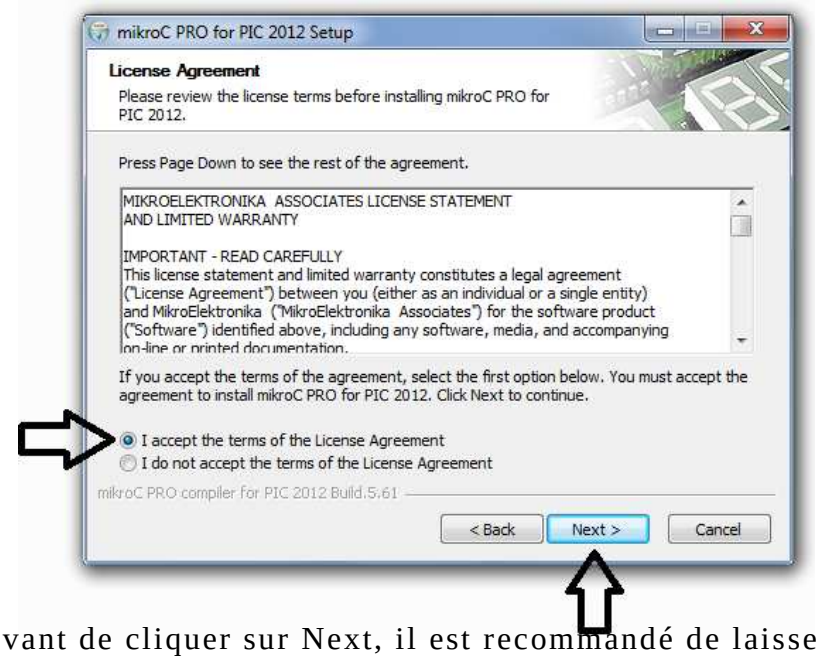
a. Installation du compilateur mikroC PRO

Etape 1 : Cliquer sur l'icône **mikroC_PRO_PIC_2012_Build.5.61.exe** (Si la fenêtre du Contrôle de compte d'utilisateur s'ouvre, cliquer sur oui), et attendre que les données de l'installation se décompressent.

Etape 2 : Cliquer sur **Next**

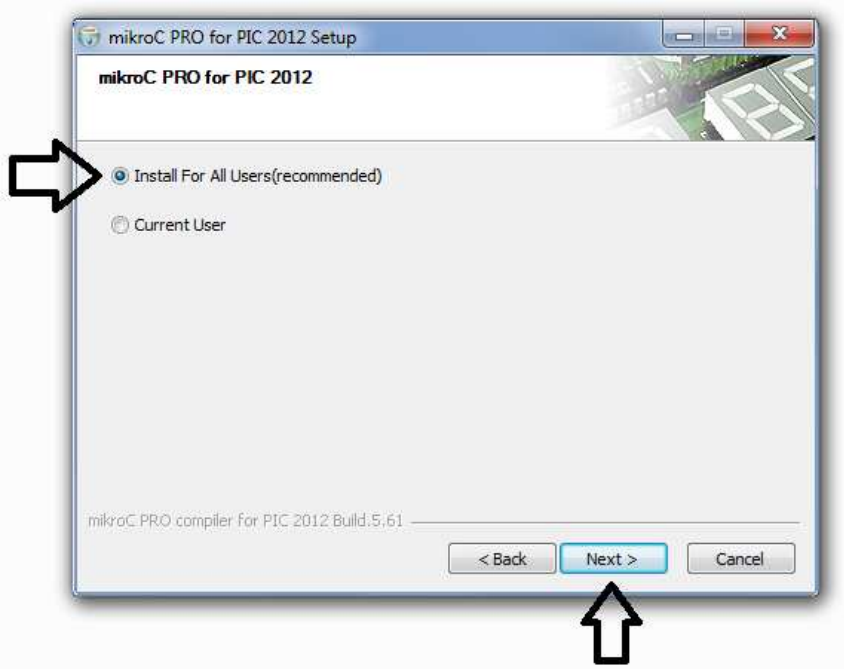


Etape 3: Cocher la case: **I accept the terms in the License Agreement** et cliquer sur **Next.**

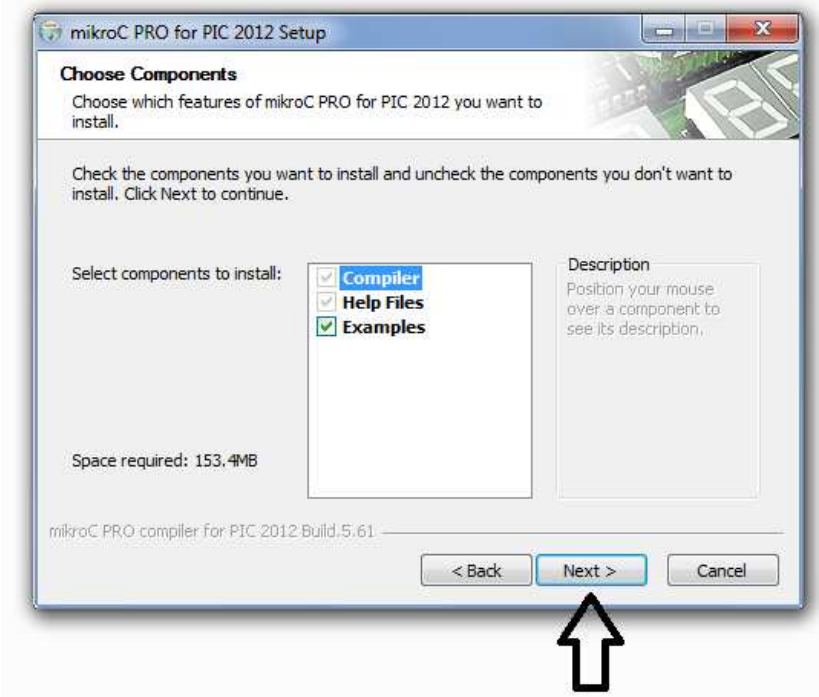


Etape 4 : Avant de cliquer sur Next, il est recommandé de laisser la case **Install For All Users** coché. Cliquer sur **Next.**

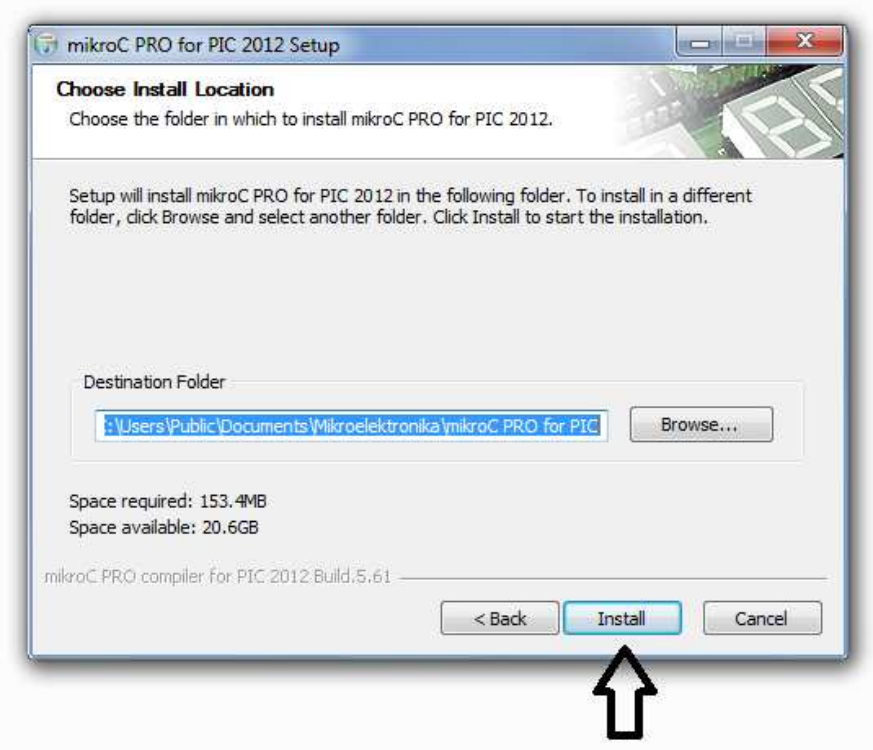
:



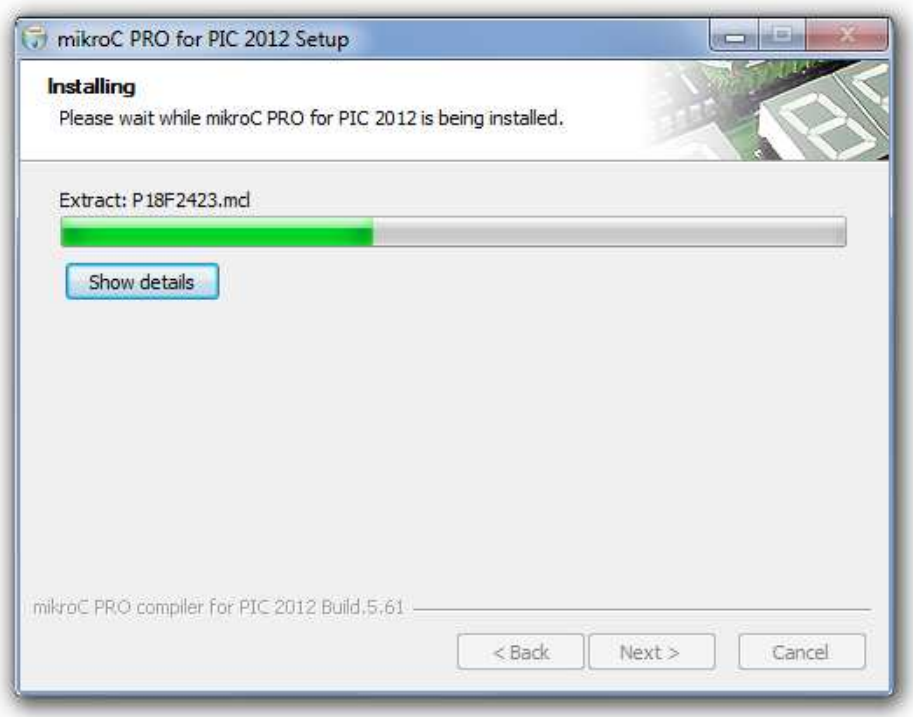
Etape 5 : Cliquer sur **Next.**



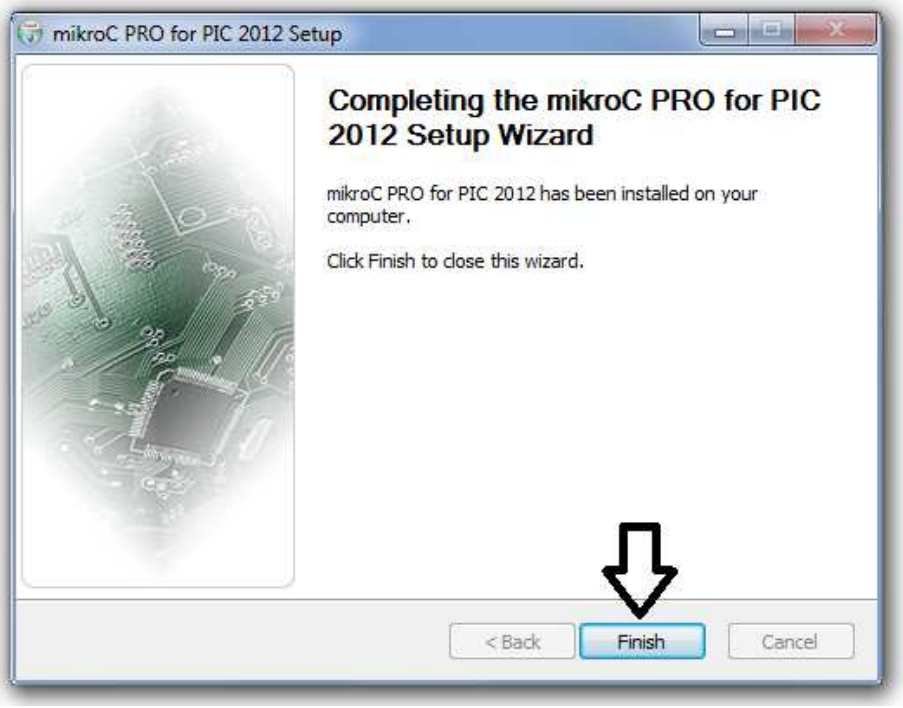
Etape 6 : Cliquer sur Install (Noter bien l'endroit d'installation).



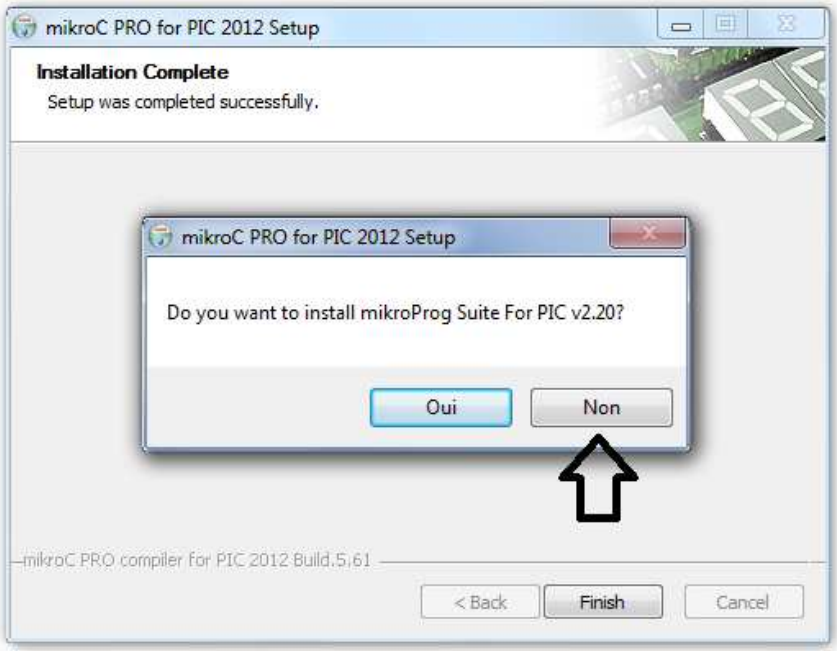
Etape 7 : Patienter pendant l'installation.

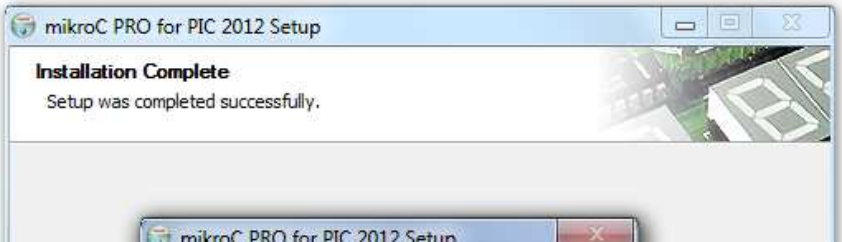


Etape 8 : Cliquer sur **Finish**



<u>Etape 9</u> : Installer **mikroProg Suit For PIC** et **mikroProg Suit drivers** si vous avez besoin, sinon continuer en cliquant sur **Non.**



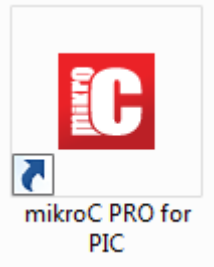


Etape 10 :

Avant de lancer **mikroC_PRO_PIC** ; Aller dans le répertoire M_License et copier le fichier qui s'y trouvent. Coller pour remplacer le fichier copié auparavant dans le répertoire (**Mikroelectronika)** ou **mikroC_PRO_PIC a** été installé.

Etape 11 :

Lancer le compilateur mikroC PRO en cliquant sur l'icône :



Maintenant, nous allons décrire seulement le processus d'écriture d'un programme en langage mikroC ; La description détaillé de toutes les options

disponibles dans ce compilateur prendre trop de temps, ce n'est pas le but de nos Travaux Pratiques ; Pour plus d'informations reportez-vous à l'aide [Help].

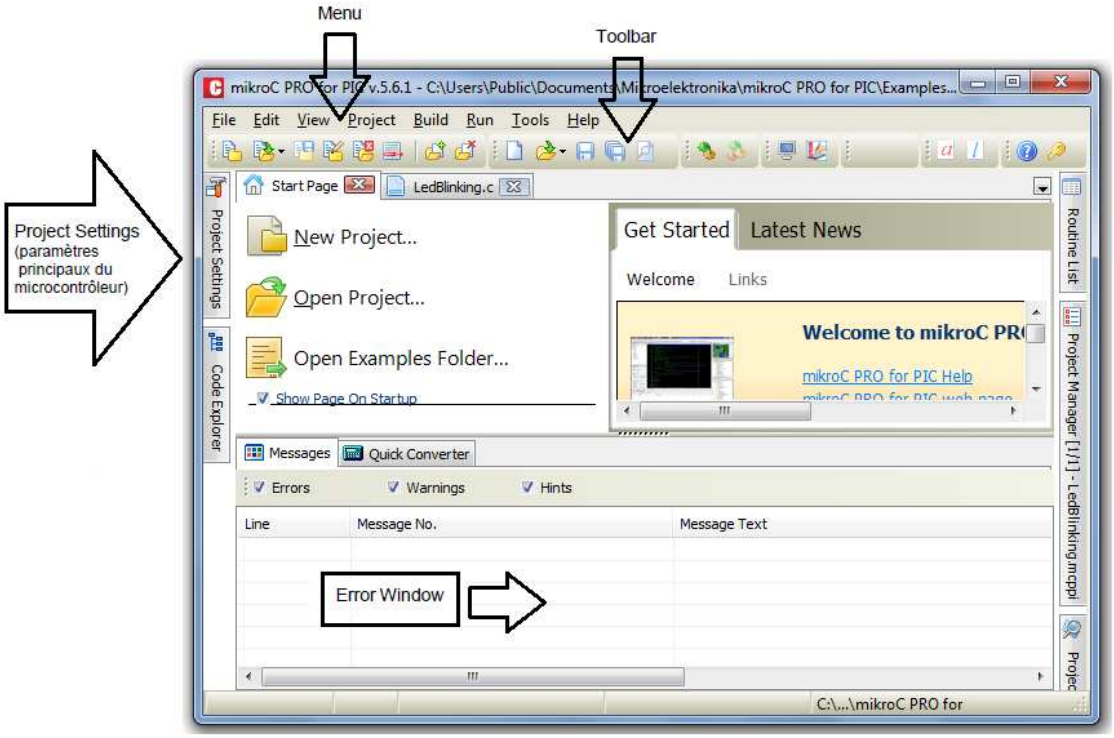
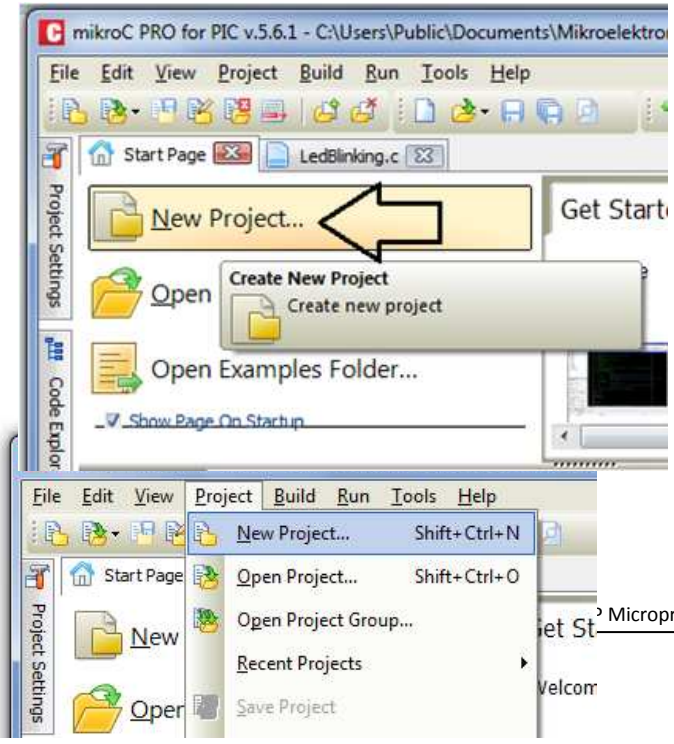


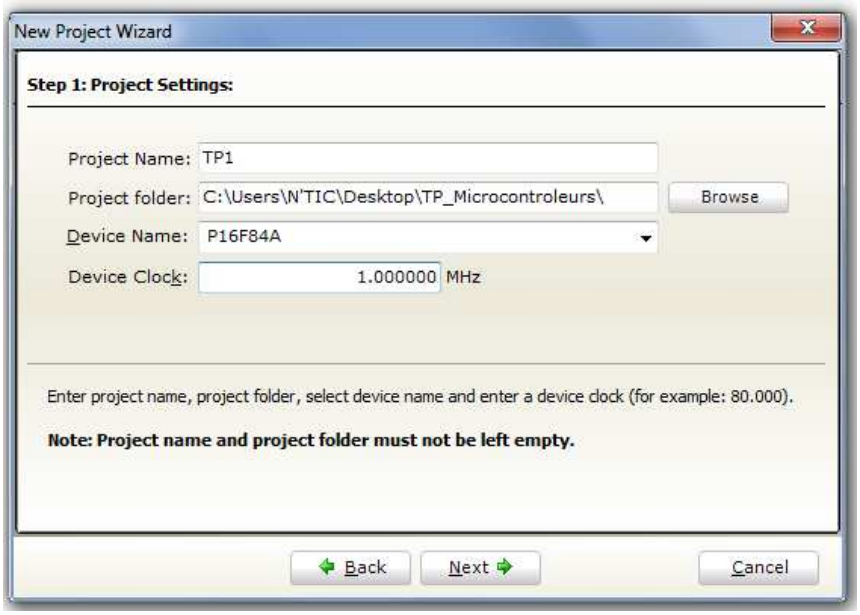
Figure.1. L'environnement **IDE** du compilateur mikroC PRO for PIC.

b. Création d'un nouveau projet

Pour créer un nouveau projet il faut cliquer à partir de la barre d'outils sur l'icône : **New project** ou à partir du **Menu** > Project > New Project).

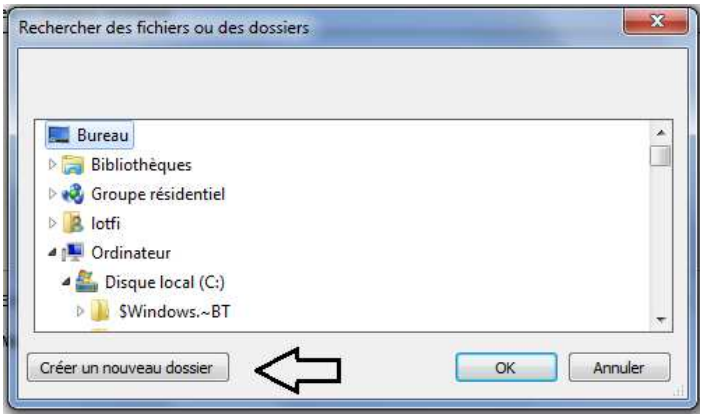


- Pour commencer la création de votre nouveau projet, cliquer sur le bouton **Next :**



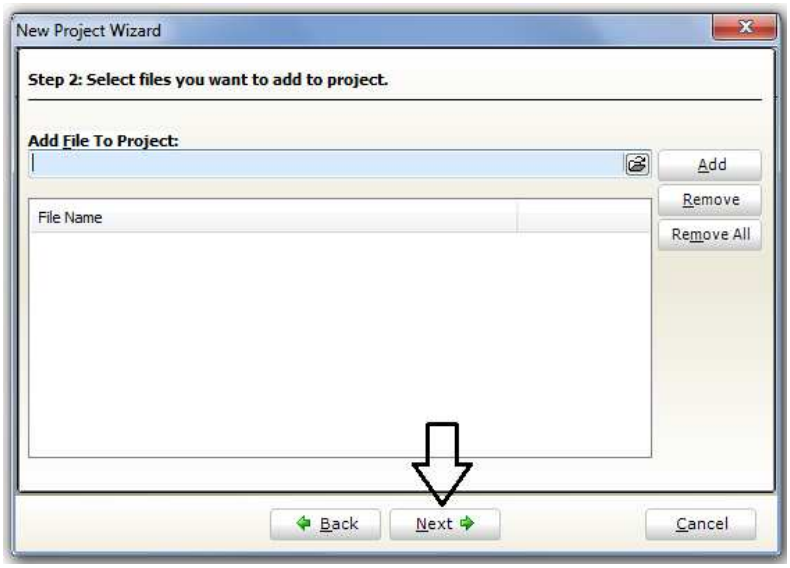
- Renommer votre projet sur **Project Name** : ex (TP1).

- Enregistrer votre projet sur un Dossier bien déterminer : ex (TP_Microcontroleurs) sur le bureau.



- Sélectionner le périphérique **16F84A** dans le périphérique dans la liste déroulante.

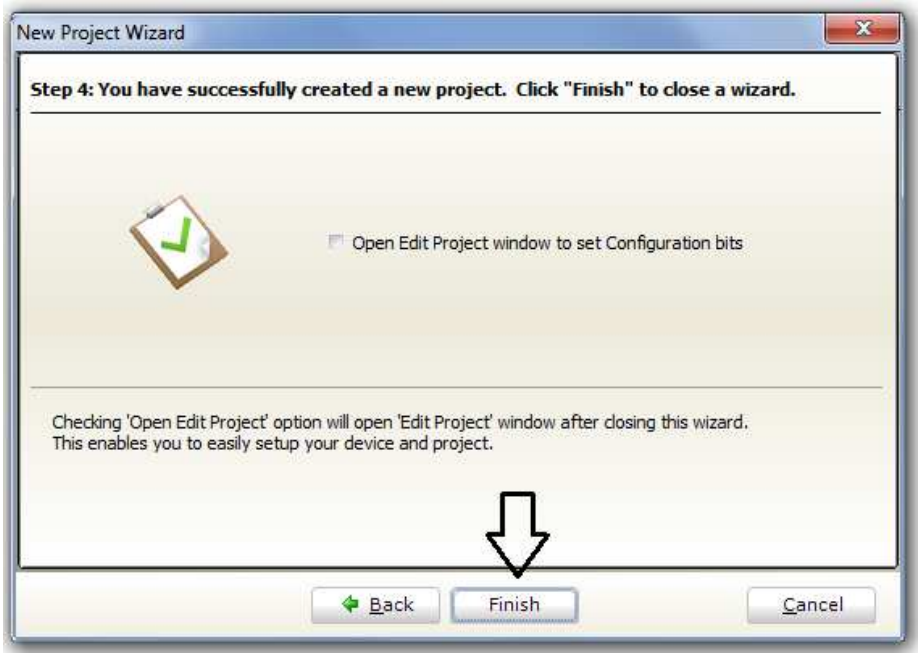
- Sélectionner la fréquence d'horloge **1MHz**.



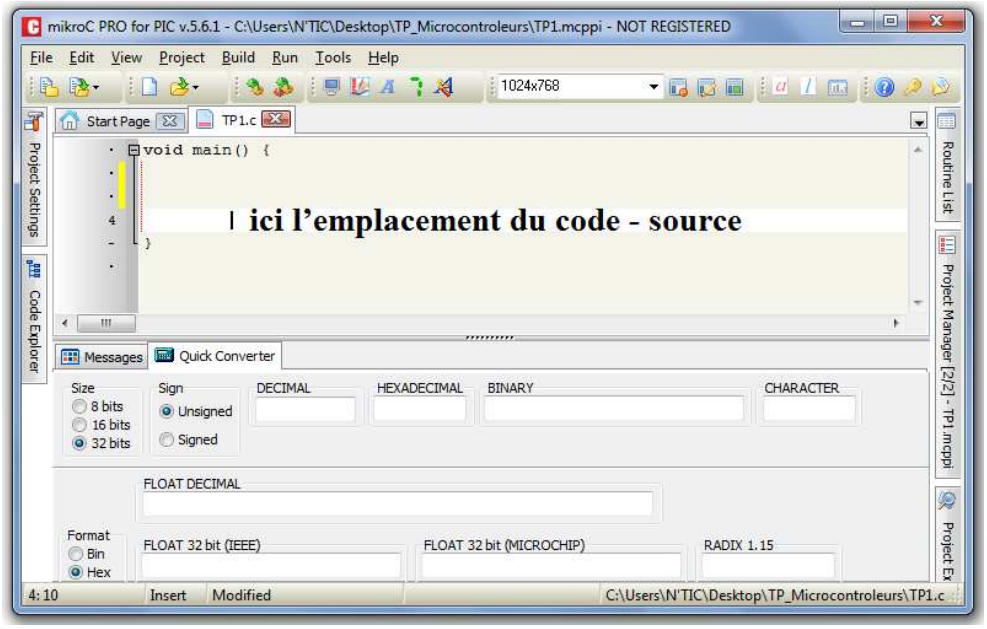
- Cliquer sur **Next.**

Cette fenêtre est utilisée si vous voulez ajouter un fichier à votre Project, vous pouvez le faire aussi en utilisant **Project Manager**.

Cliquez sur **Finish** pour créer votre nouveau projet.



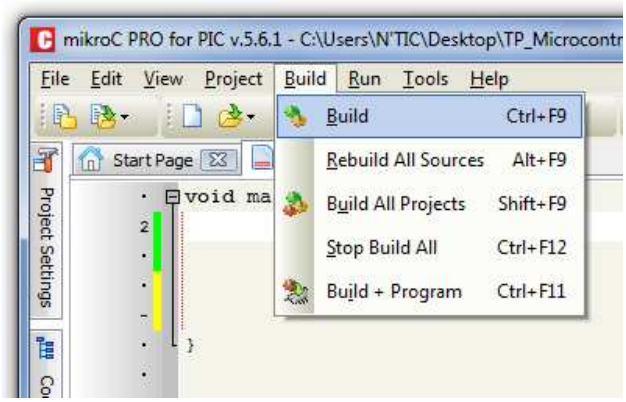
- Une fenêtre vide s'affiche afin de saisir le programme (code - source).

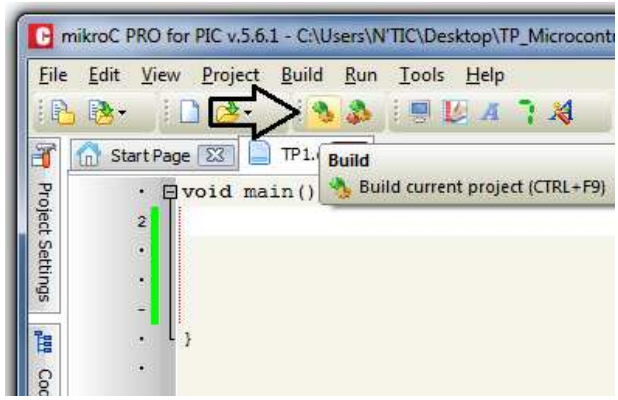


c. Compilation

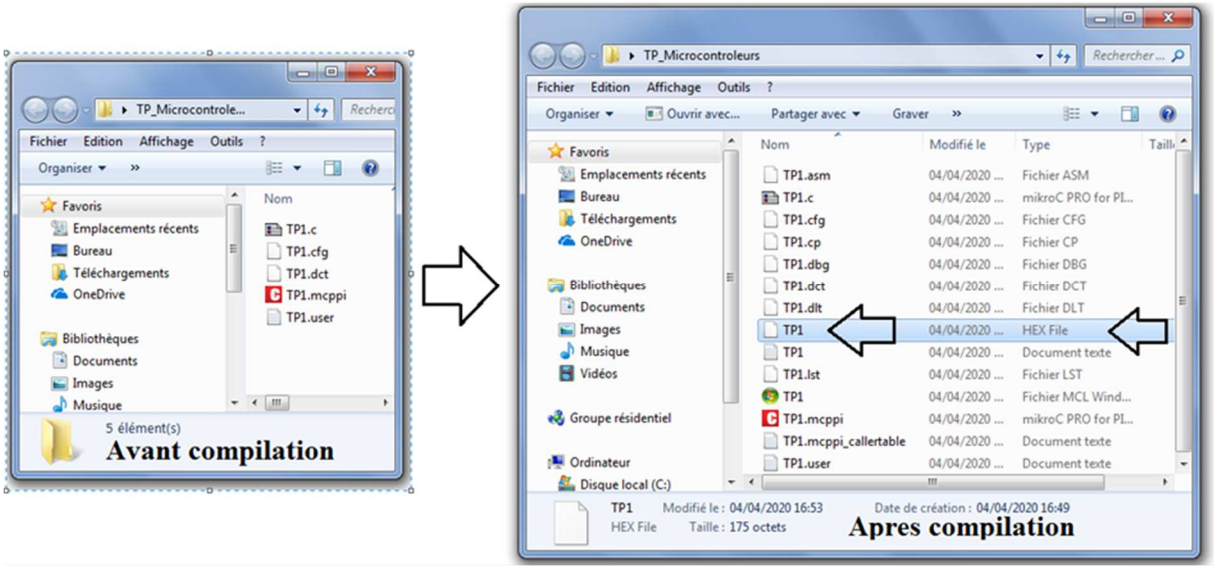
Après l'écriture du **code source**, la compilation est l'étape suivante pour avoir le fichier : **extension. Hex**

- Sélectionnez **Build** *à* partir du menu déroulant ou cliquez sur l'icône **Build** dans la barre d'outils du projet.





Le ***mikroC PRO* for PIC** organise des applications dans des projets, composé d'un seul fichier de projet (**extension. mcppi**) et un ou plusieurs fichiers sources (extension).

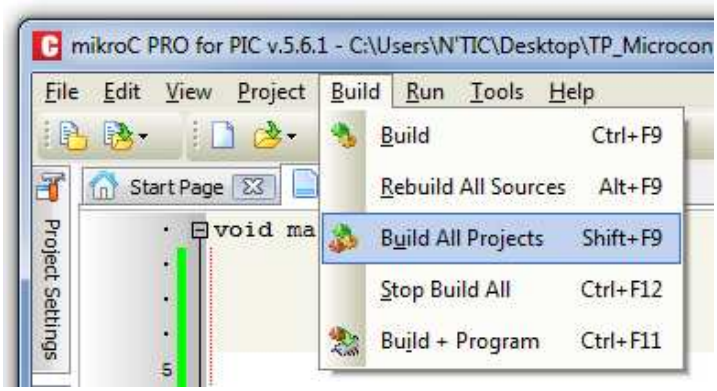


Les fichiers sources peuvent être compilés s'ils font partie d'un projet.

Le fichier projet contient les informations suivantes :

- Nom du projet et une description facultative
- Composant cible.
- Option du composant
- Fréquence d'horloge du composant
- La liste des fichiers source du projet avec les chemins
- Fichiers d'images
- Fichiers binaires (* mcl.)
- D'autres fichiers Après compilation on aura d'autre extensions dont **l'extension. Hex**, qui sera par la suite intégrée au niveau du **PIC**.

Si plus d'un projet est ouvert, vous pouvez les compiler tous avec **Build All Projects** en sélectionnant **Build.**



- La Barre de progression s'affiche pour vous informer sur l'état de la mémoire RAM et ROM, l'espace mémoire utilisé et de la compilation. S'il y'a des erreurs, vous en serez informé dans la fenêtre d'erreur (Figure. 2).

- Le convertisseur rapide est souvent utilisé pour une aide dans la programmation (Figure.2).

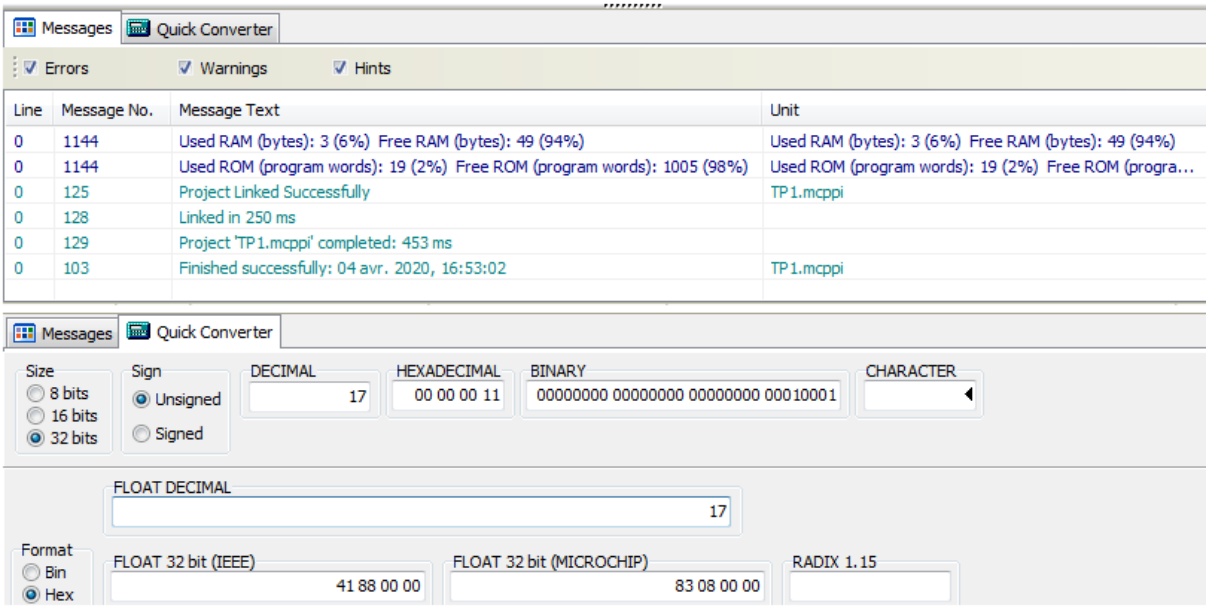


Figure.2. Message d'erreur et le convertisseur utilisé par L'environnement **IDE** du compilateur **mikroC PRO for PIC.**

4. Le langage de programmation miKroC

a. Structure d'un programme en mikroC

La structure la plus simple d'un programme en mikroC, c'est le programme représenté dans le code-source suivant :

```
void main() { // Début du programme
TRISB = 0;  // Configuration du PORTB  en sortie
PORTB = 0;  // Initialisation du PORTB
for(;;)  // Boucle sans fin
{ // Début de la boucle
PORTB.b0 = 0; // RB0 = 0
Delay_ms(1000);  // Pause d'une seconde
PORTB.b0 = 1;  // RB0 = 1
Delay_ms(1000);  // Pause d'une seconde
}  // Fin de la boucle
} // Fin du programme
```

Ce programme nous permettra de faire clignoter une LED connectée au PORTB (par exemple bit 0 du PORTB) du microcontrôleur PIC avec une période de 2 secondes (1 seconde allumée et une seconde éteinte).

b. Règle générale de l'écriture en mikroC

- Les instructions propres au langage microC doivent être écrites en minuscule. Ex : void main
- Les instructions particulières aux microcontrôleurs doivent être écrites en majuscule. Ex : PORTB
- Les retours à la ligne et les espaces servent uniquement à aérer le code
- Toutes instructions ou actions se terminent par un point-virgule « **;** ».
- Les commentaires sont utilisés pour préciser le fonctionnement du programme et pour une annotation du programme.
- Les lignes de commentaires sont ignorées et non compilé par le compilateur. Ils commencent par le caractère « // » et il n'a pas besoin d'un caractère de terminaison.

5. La simulation avec le logiciel ISIS de PROTEUS V8

Proteus est un logiciel de développement et de simulation d'application via un environnement graphique simple et interactif. Le lancement de PROTEUS donne un environnement classique de type Windows, constitué d'une fenêtre principale, et d'un ensemble de barres d'outils (Figure.3).

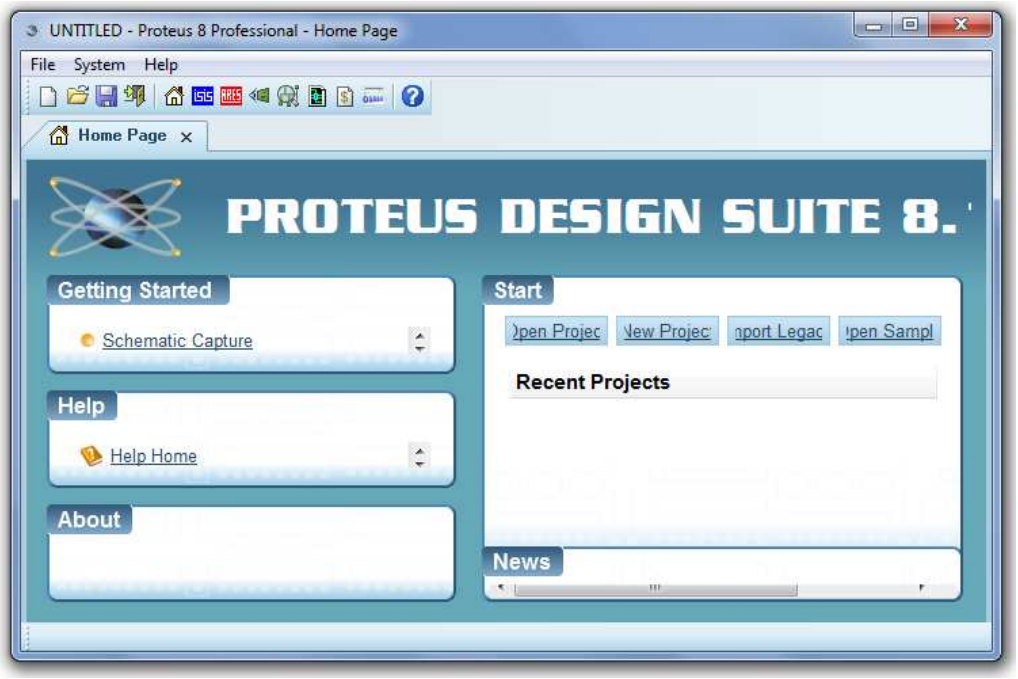


Figure.3. Le logiciel PROTEUS V8

En cliquant sur l'icône **ISIS** de PROTEUS une autre fenêtre apparaisse (Figure.4).

Outre le menu classique permettant la gestion des fichiers, de l'affichage, et des options des projets, la fenêtre principale comprend une Zone de travail destinée au développement des circuits à simuler et à tester.

Une Bibliothèque d'objets affiche la liste des objets (circuits électriques, électroniques…) utilisés dans l'application en cours.

Les différentes Touches magnétoscope constitues des raccourcis permettant le lancement de la simulation, ainsi que la mise en pause, l'exécution pas à pas, et l'arrêt de la simulation (Figure.4)

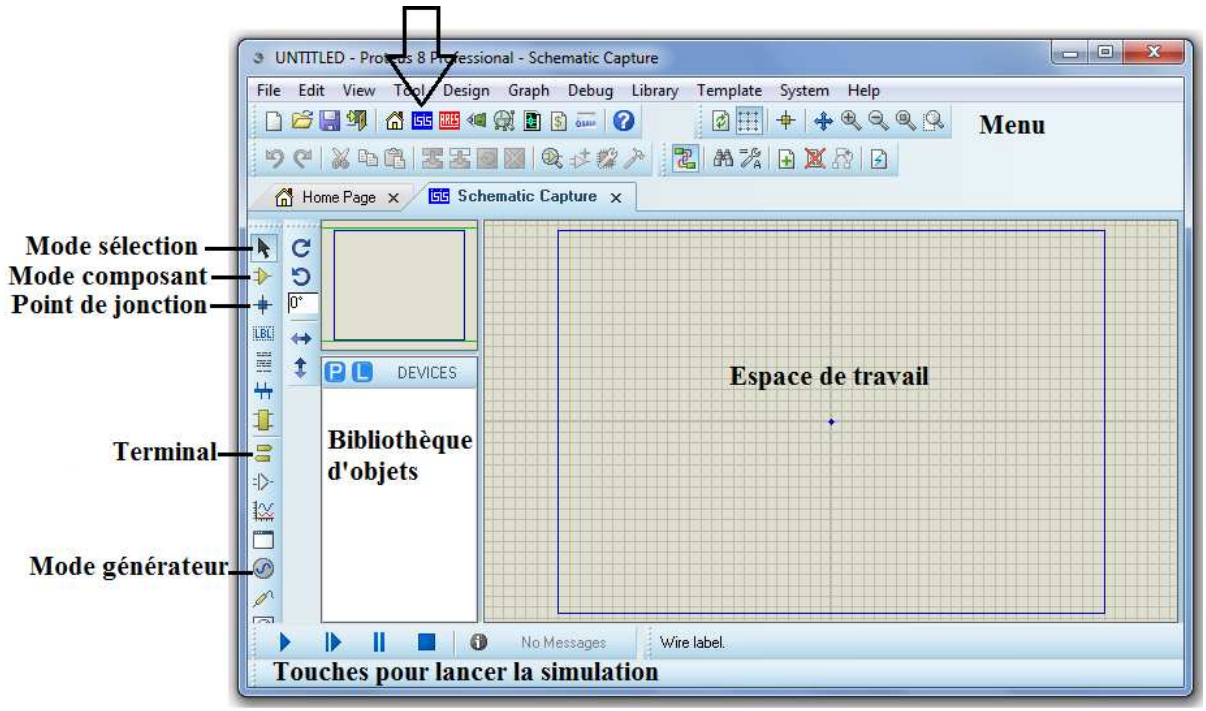


Figure.4. Fenêtre de gestion ISIS de PROTEUS.

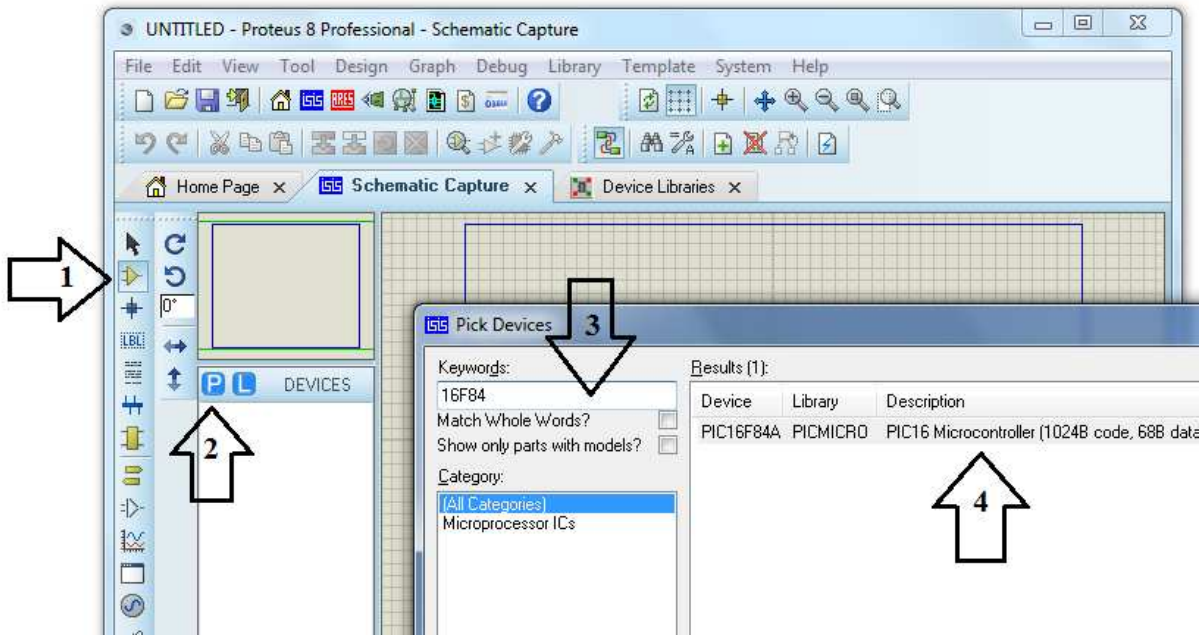
a. La simulation des Microcontrôleurs

Dans cette partie on s'intéresse à la simulation des à base de Microcontrôleur de Type PIC 16F84A. Le circuit de base d'un système à microcontrôleurs comprend généralement et en particulier :

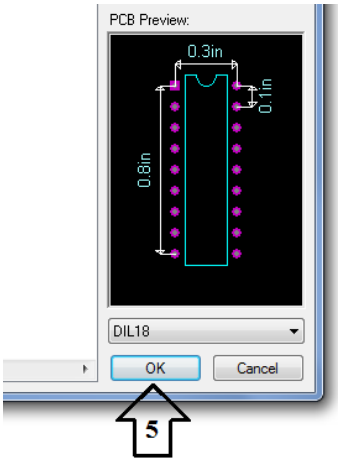
- Un circuit RESET, qui peut être un simple circuit RC.
- Un circuit d'horloge, qui est généralement un oscillateur à quartz.

Remarque : On n'a pas besoin lors de la simulation de ces deux circuits ; Le PIC une fois poser dans l'espace de travail fonctionne même sans alimentation.

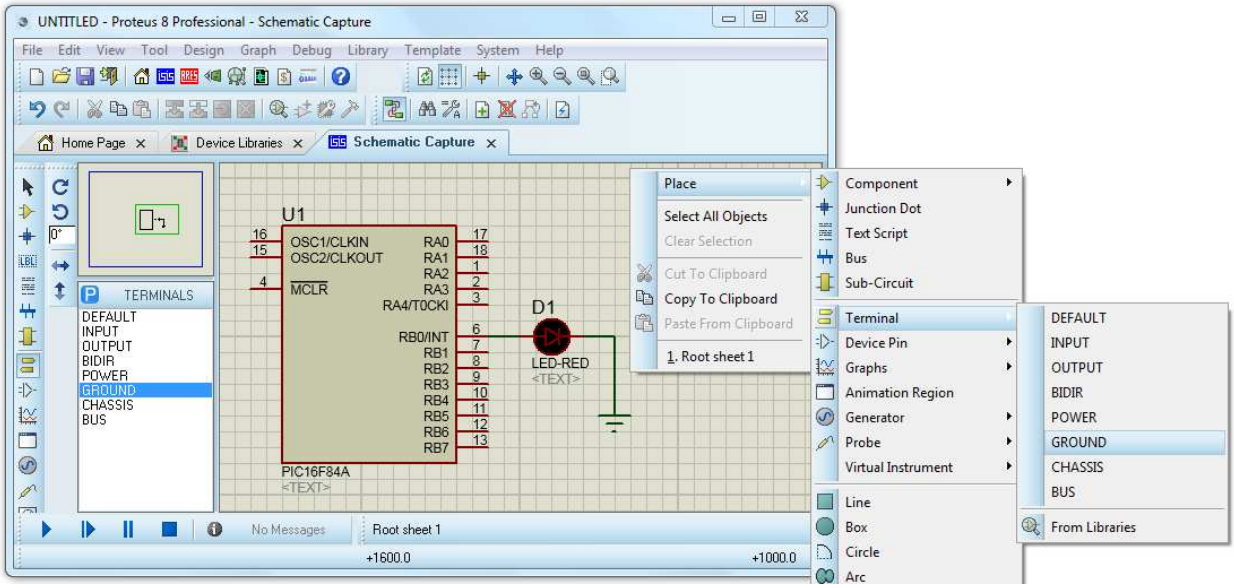
Etape 1 : Dans la barre d'outils principale, cliquer sur le **mode composant (1),** ensuite sur le **P** (Pick Devices), choisir des appareils ; Vous noter sur la case recherche 16F84 par exemple, et vous cliquer sur l'appareil que vous voulez.



Etape 2 : Approuver votre choix en cliquant sur **OK**

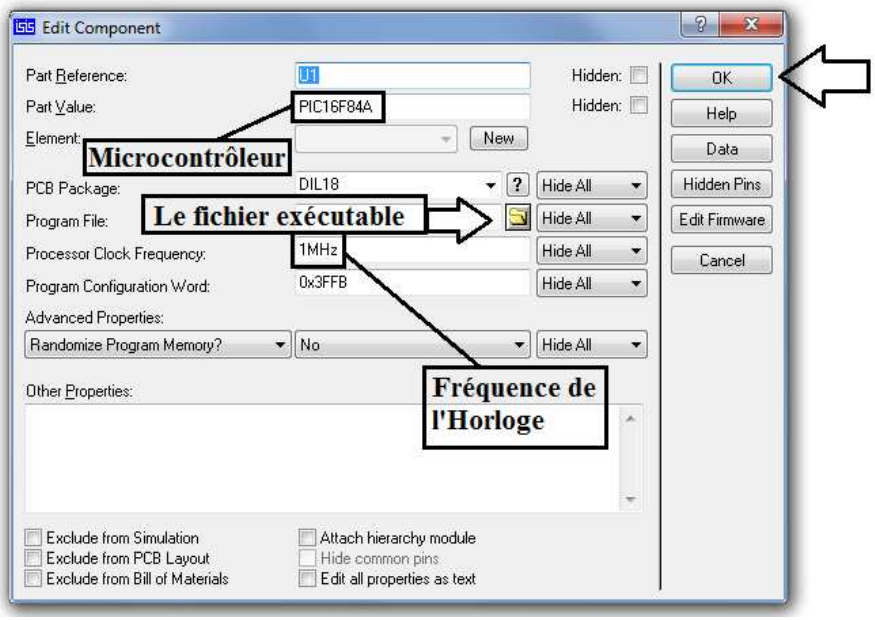


Etape 3 : Vous pouvez centrer le PIC, faire un Zoom et connecter autre éléments et appareils en procèdent de la même manière. Vous pouvez ajouter les appareils avec la souris (cliquer sur le bouton droit de la souris) ; pour ajouter la masse par exemple (GROUND), comme illustrer sur la figure suivante :

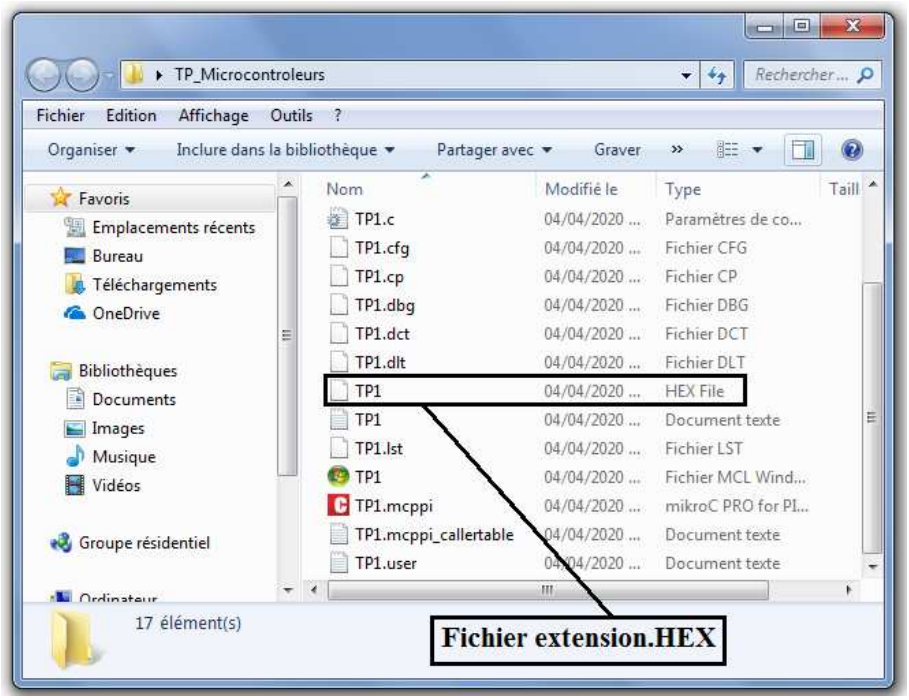


Etape 4 : Un double clic sur le microcontrôleur choisi 16F84A, permet la définition de sa fréquence de fonctionnement, et du nom du fichier contenant le code machine (fichier.hex) qui sera exécuté durant la simulation. 1.

Remarque : Pour une bonne synchronisation il faut choisir la même fréquence de fonctionnement (horloge) choisie auparavant sur le programme de mikroC.



Etape 5 : Une fois le fichier extension.HEX est chargé sur le PIC, vous pouvez lancer la simulation.



Etape 6 : Lancer de la simulation avec **PLAY** sur **ISIS.**

TP N° 1 : Gestion Des Sorties Et Multiplexage Avec Un Microcontrôleur

(Exemple : PIC 16F84) Utilisation des Afficheurs 7 segments.

1. Généralités :

Le PIC 16F84A est un µC (microcontrôleur) **8 bits** de la société américaine Microchip. Le 16F84A possède **13 entrées/sorties** (5 dans le port A et 8 dans le port B). Chaque entrée/sortie est configurable individuellement (en entrée ou bien en sortie). On peut par exemple configurer les broches RB0, RA2 et RA3 en entrée et les broches RB1, RB2, RB3, RA0 et RA1 en sortie.

Cas particulier de la broche RA4 configurée en sortie possède **une sortie de type drain ouvert.** Cela veut dire qu'elle ne peut pas fournir de courant. Cependant, elle peut en consommer.

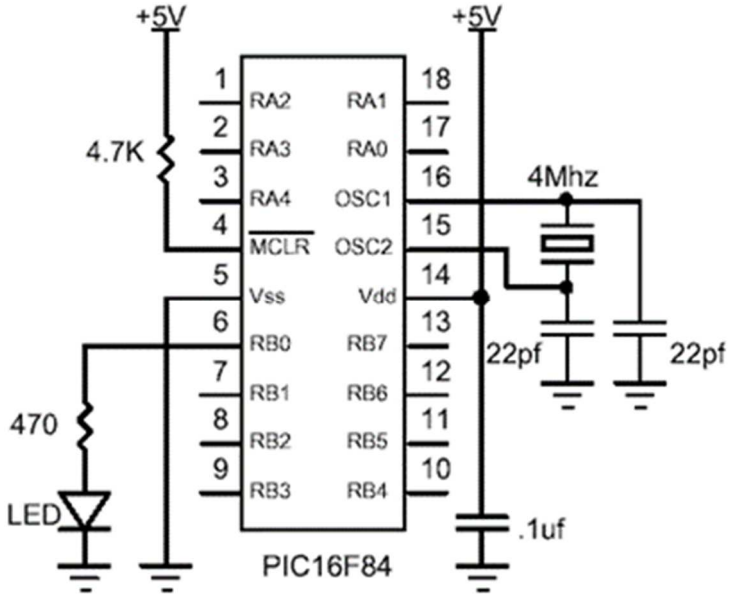


Figure. 5. Présentation du PIC 16F84.

2. Première partie :

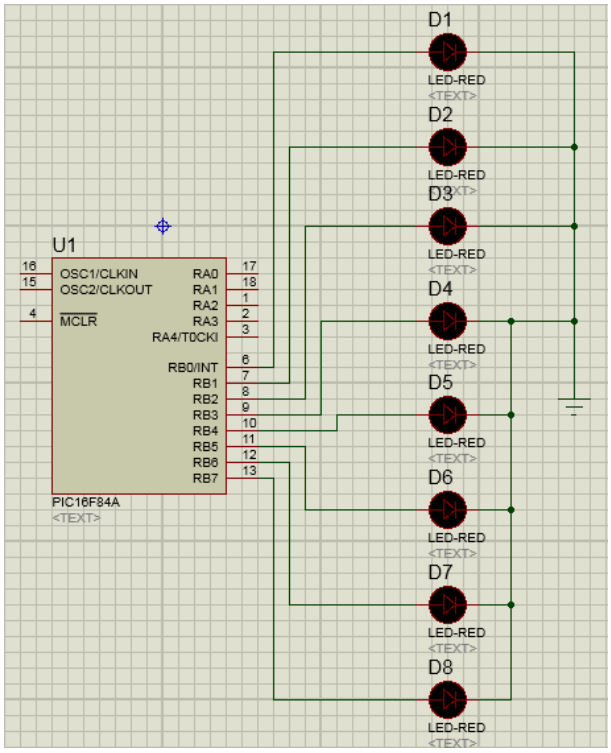
Objectifs : Vérifier le fonctionnement du **PIC 16F84** (Jeu de lumière et feu tricolore de circulation).

- Lancer ISIS de Proteus, et développer le circuit à base du 16F84 ci-contre.
- Lancer MikroC for Pic, puis éditer et compiler le programme en C suivant.
- Simuler le circuit précédent, en lui associant le programme donné.
- Modifier le programme pour allumer les LEDs deux secondes au lieu d'une seconde.
- Modifier le programme pour allumer deux ou trois LEDs au même temps.

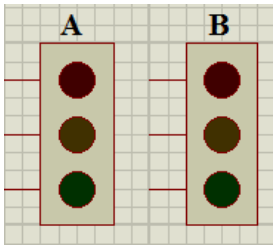
Designation: LED BLUE / RED / GREEN / YELLOW.

Programme1 :

```
void main()
{
TRISB=0x00;
// Registre de configuration
// ((TRISB=0 : sortie, TRISB=1 : entrée))
// TRISB=0b00000000; TRISB=0 ;
PORTB=0x00;
// registre de travail, pour initialisation

for(;;)
{
PORTB=0b00000001; delay_ms(1000);
PORTB=0b00000010; delay_ms(1000);
PORTB=0b00000100; delay_ms(1000);
PORTB=0b00001000; delay_ms(1000);
PORTB=0b00001000; delay_ms(1000);
PORTB=0b00010000; delay_ms(1000);
PORTB=0b00100000; delay_ms(1000);
PORTB=0b01000000; delay_ms(1000);
PORTB=0b10000000; delay_ms(1000);
}
}
```



- Modifier le programme pour réaliser deux feux tricolores.



Désignation: TRAFFIC LIGHTS

Rouge : Tout conducteur doit marquer l'arrêt absolu.
Vert : Autorisation de passage des véhicules.
Orange : Clignote avec le vert pour prévenir du passage au rouge.

A: ROUGE (5 secondes) >> VERT (3 s) >> ORANGE (2 s) >> ROUGE (5 s)
B: VERT (3 secondes) >> ORANGE (2 s) >> ROUGE (5 s) >> VERT (3 s)

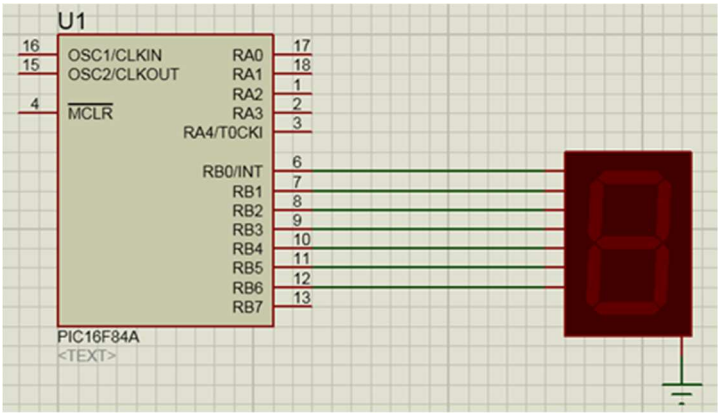
3. Deuxième partie :

Objectifs : Application du multiplexage : Utilisation des Afficheurs 7 segments et le décodeur 74ls48.

Désignation : 7SEG-COM-CATHODE

- Développer le circuit à base du 16F84 avec l'afficheur 7 segments ci-contre.
- Lancer MikroC for Pic, puis éditer et compiler le programme en C suivant.
- Simuler le circuit précédent, en lui associant le programme donné.
- Modifier le programme pour afficher la suite des nombres de 0 à 9 (utiliser un afficheur 7 Segments à cathode commune).
- Modifier le programme pour afficher la suite des nombres comme suite 5 à 9 ensuite de 0 à 9.

Programme2 :

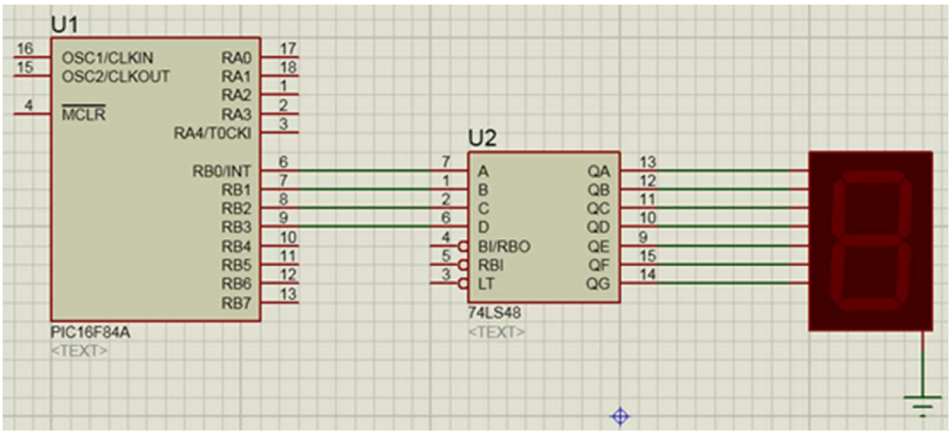



```
void main()
{
TRISB=0x00;  // registre de configuration.
PORTB=0x00; // registre de travail.
PORTB=0 ;
PORTB=1 ;
PORTB=3 ;
PORTB=4 ;
PORTB=5 ;
PORTB=6 ;
PORTB=7 ;
PORTB=8 ;
PORTB=9 ;
}
```

Ou avec une Boucle for :

```
void main()
{
TRISB=0x00;  // registre de configuration.
PORTB=0x00; // registre de travail.

for(;;)
{
  PORTB++;
  delay_ms(1000);
  if (PORTB==9)(PORTB=0);
  delay_ms(1000);
}}
```

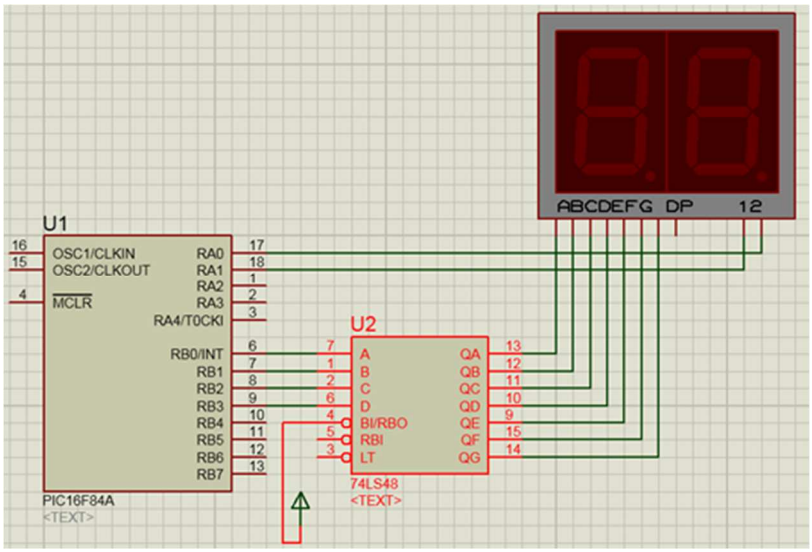
-Utiliser le décodeur 74ls48 avec l'afficheur 7 segments pour afficher la suite des nombres de 0 à 9.



-Modifier le schéma avec un décalage des lignes de connections (RB2 avec A, RB3 avec B, RB4 avec C et RB5 avec D) et modifier le programme pour afficher la suite des nombres de 0 à 9.

Application du multiplexage (Utilisation de deux afficheurs 7 segments).

- Développer le circuit à base du 16F84 avec deux afficheurs 7 segments ci-contre (Cathode commune).
- Déterminer le fonctionnement de l'afficheurs double avec RA0 et RA1.
- Modifier le programme pour afficher la suite des nombres de 00, 11, 22…
- Modifier le programme pour afficher le chiffre 15.



TP N° 2 : Gestion des entrées / sorties et utilisation de case mémoire RAM

(Application sur le microcontrôleur PIC 16F84A)

1. Généralités :

Le PORT A est un des deux Ports du PIC 16F84, et comprend 5 lignes Entrées/Sorties. Sa configuration et sa programmation passent par l'utilisation de deux registres qui sont PORTA et TRISA.

Le registre PORTA (Bank 0) est une copie des lignes RA0…RA4, tant en entrée qu'en sortie. En effet, lire le PORTA revient à lire l'état des pins alors qu'une écriture place le niveau correspondant sur les pins qui auront été configurées en sorties (dans une séquence interne au PIC de Read-modify-write).

Les lignes RA0 … RA3 sont des entrées à niveaux compatibles TTL et des sorties CMOS standards. La ligne RA4 est une entrée à Trigger de Schmitt et une sortie à drain ouvert qui est multiplexée avec l'entrée de Timer TMR0. Le registre TRISA (Bank 1) est le registre qui permettra de placer les pins indépendamment en entrée ou en sortie.

- Mettre un bit de TRISA à 1 placera le pin correspondant en entrée (et le circuit de sortie correspondant en haute impédance).

- Mettre un bit de TRISA à 0 placera le pin correspondant du PORTA en sortie.

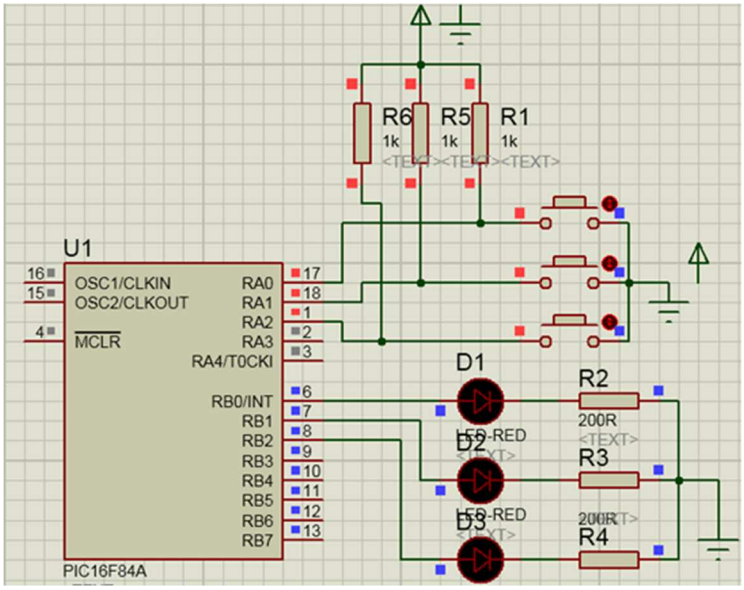
2. Première partie :

Objectifs : Utilisation des interrupteurs pour allumer les LEDs.

Désignation :

Symboles	Désignations
	RES
	BUTTON
	SW_SPST

- Lancer ISIS de Proteus, et développer le circuit à base du 16F84 ci-contre.
- Lancer MikroC for Pic, puis éditer et compiler le programme en C suivant.
- Simuler le circuit précédent, en lui associant le programme donné.



Programme1 :

```
void main()
{

TRISB=0x00;      // registre de configuration
TRISA=0xff ;  // Quick Converter mikroC //ou TRISA=0b00000111(RA0, RA1 et RA2 on sortie) ;

PORTB=0x00;  // registre de travail.
PORTA=0x00;  // registre de travail.

for(;;)
{
  If(PORTA.b0==0)PORTB.b0=1;else PORTB.b0=0;
    // b représente le bit, un seul '' = '' implique une erreur au niveau du simulateur ISIS
   //et pas au niveau du compilateur mikroC.

 If(PORTA.b1==0)PORTB.b1=1;
 else PORTB.b1=0;
  If(PORTA.b2==0)PORTB.b2=1;
else PORTB.b2=0;
}
}
```

- Procéder à un remplacement des masses et des alimentations et Modifier le programme1 pour allumer les différentes LEDs.

3. Deuxième partie :

Objectifs : Utilisation de case mémoire.

Une variable est une portion réservée d'une mémoire à laquelle on a donné un nom. On peut stocker ou lire les informations de ces variables. Toute variable utilisée dans un programme doit auparavant être définie.

La **définition** d'une variable consiste à la **nommer** et lui donnée un **type** et éventuellement lui donner une valeur initiale (**initialiser**). C'est cette définition qui réserve (**alloue**) la place mémoire nécessaire en fonction du type.

Certains noms sont réservés pour le compilateur lui-même et ne peut pas être utilisés comme noms de variables dans un programme :

Asm, enum, signed, auto, extern, sizeof, break, float, static, case, for, struct, char, goto, switch, const, if, typedef, continue int, union, default, long, unsigned, do, register, void, double, return, volatile, else, short, while.

Le langage mikroC prend en charge les types de variables suivant :

Type	Taille (bits)	Plage
unsigned char	8	0 à 255
unsigned short int	8	0 à 255
unsigned int	16	0 à 65535
unsigned long int	32	0 à 4294967295
signed char	8	-128 à 127
signed short int	8	-128 à 127
signed int	16	-32768 à 32767
signed long int	32	- 2147483648 à 2147483647
float	32	± 1.17549435082E-38 à ±6.80564774407E38
double	32	± 1.17549435082E-38 à ± 6.80564774407E38
long double	32	± 1.17549435082E-38 à ± 6.80564774407E38

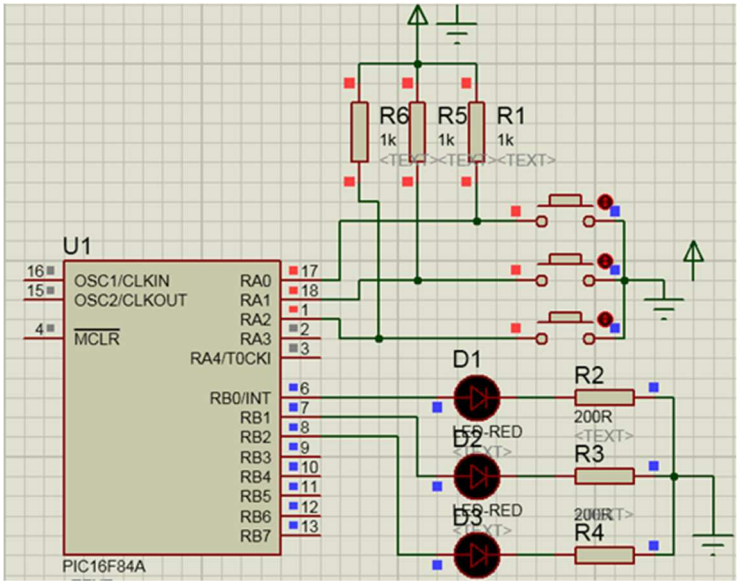
Tableau.1 Types de variables en mikroC.

```
bit a ;        //Crée une case de mémoire avec un seul bit soit 0 ou 1
char b ;     // Crée un octet dans la mémoire RAM
```

Le programme 2, présente une autre utilisation des interrupteurs.
-Donner la différence entre le fonctionnement dans le **programme1** et **programme2**.

Programme2 :

```
bit a1,a2,a3 ; //Crée trois case mémoire ;

void main()
{

TRISB=0x00 ;  // registre de configuration
PORTB=0x00 ; // registre de travail.
TRISA=0xff ;
a1=a2=a3=0;

for(;;)
{
if(PORTA.b0==0) { a1=~a1 ; delay_ms(300);  }
PORTB.b0=a1;
if(PORTA.b1==0)  {a2=~a2 ; delay_ms(300);  }
PORTB.b1=a2;
if(PORTA.b2==0) { a3=~a3 ; delay_ms(300); }
PORTB.b2=a3;
}
}
```



- Réaliser un compteur décompteur de 0 à 9 et de 9 à 0 sur un afficheur 7 segments ; à l'aide d'interrupteurs.
-Si un interrupteur A est enfoncé, on a un compteur.
-Si un interrupteur B est enfoncé, on a un décompteur.

TP N° 3 : Gestion des interruptions et utilisation des sous-programmes

(Application sur les afficheurs LCD)

1. Généralités :

Les afficheurs à cristaux liquides, appelés afficheurs **LCD** (**L**iquid **C**rystal **D**isplay), sont des modules compacts intelligents et nécessitent peu de composants externes pour un bon fonctionnement. Ils consomment relativement peu (de 1 à 5 mA).

Plusieurs afficheurs sont disponibles sur le marché et diffèrent les uns des autres, par leurs dimensions, (de 1 à 4 lignes de 6 à 80 caractères), et aussi par leurs caractéristiques techniques et leur tension de service. Certains sont dotés d'un rétro-éclairage. Cette fonction fait appel à des LED montées derrière l'écran du module.

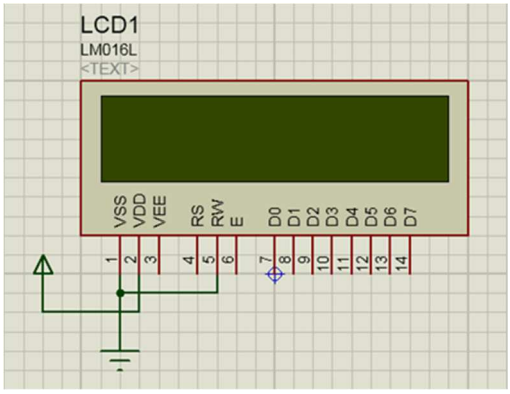
Deux modes de fonctionnement de l'afficheur sont disponibles, le mode 4 bits et le mode 8 bits :

Mode 8 bits : Dans ce mode 8 bits, les données sont envoyées à l'afficheur sur les broches D0 à D7.

Mode 4 bits : Il peut, dans certains cas, être nécessaire de diminuer le nombre de fils utilisés pour commander l'afficheur, comme, par exemple lorsqu'on dispose de très peu de broches d'entrées sorties disponibles sur un microcontrôleur. Dans ce cas, on peut utiliser le mode quatre bits de l'afficheur LCD. Dans ce mode, seuls les 4 bits de poids fort (D4 à D7) de l'afficheur sont utilisées pour transmettre les données et les lire. Les 4 bits de poids faible (D0 à D3) sont alors connectés à la masse. On a donc besoin, hors alimentation de sept fils pour commander l'afficheur. Les données sont alors écrites ou lues en envoyant séquentiellement les quatre bits de poids fort suivi des quatre bits de poids faible.

Un afficheur LCD est composé de plusieurs parties :

- Un contrôleur (driver LCD) qui communique avec l'extérieur et gère l'affichage.
- Led de rétro éclairage (pour activer le rétro éclairage ou non).
- Contraste : contraste de l'affichage.



Il existe plusieurs tailles d'écran ; Dans notre cas on utilise un 16 x 2 (2 lignes de 16 caractères).
- VEE ajuster l'éclairage avec un potentiomètre de 10 ou 20 k.
- RW =1 lecture à partir du LCD.
- RW=0 écriture dans le LCD.
- La communication du LCD (afficheur et registre) avec le PIC se fait via un bus de données de 4bits (D4,D5,D6 et D7 (data bit)) ou de 8bits.

RB0 avec RS (Registre entré select).
RB1 avec E (Entrée de validation).

Avant chaque connexion d'un LCD les variables suivantes doivent être défini avant d'utiliser cette fonction :

- LCD_D7: Data bit 7
- LCD_D6: Data bit 6
- LCD_D5: Data bit 5
- LCD_D4: Data bit 4
- LCD_RS: Register Select (data/instruction) signal pin
- LCD_EN: Enable signal pin
- LCD_D7_Direction: Direction of the Data 7 pin
- LCD_D6_Direction: Direction of the Data 6 pin
- LCD_D5_Direction: Direction of the Data 5 pin
- LCD_D4_Direction: Direction of the Data 4 pin
- LCD_RS_Direction: Direction of the Register Select pin

LCD_EN_Direction: Direction of the Enable signal pin

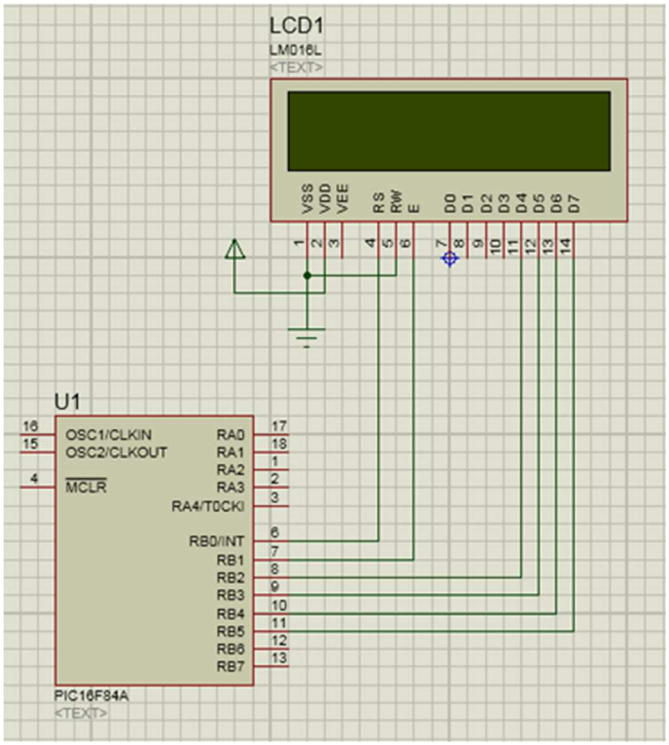
2. Première partie :

Objectifs : Utilisation des afficheurs LCD.

Désignation : LM016L ou LCD

- Développer le circuit à base du 16F84 avec l'afficheur LCD 16X2 ci-contre.
- Lancer MikroC for Pic, puis éditer et compiler le programme en C suivant.
- Simuler le circuit précédent, en lui associant le programme donné.

Programme1:



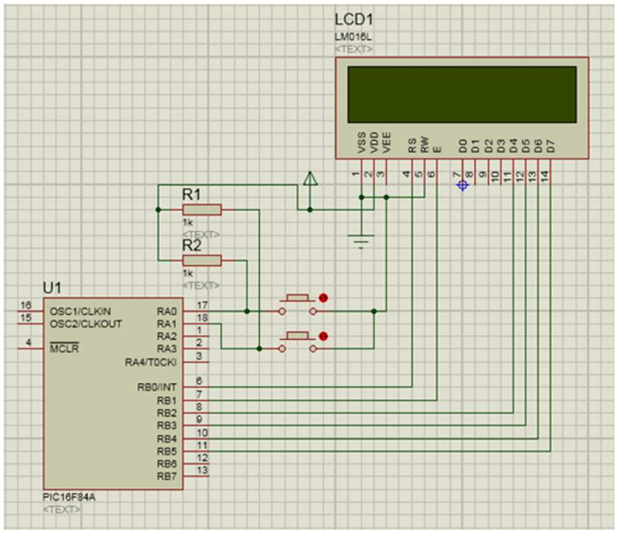

```
// Lcd pinout settings
sbit LCD_RS at RB0_bit;
sbit LCD_EN at RB1_bit;
sbit LCD_D7 at RB5_bit;
sbit LCD_D6 at RB4_bit;
sbit LCD_D5 at RB3_bit;
sbit LCD_D4 at RB2_bit;

// Pin direction
sbit LCD_RS_Direction at TRISB0_bit;
sbit LCD_EN_Direction at TRISB1_bit;
sbit LCD_D7_Direction at TRISB5_bit;
sbit LCD_D6_Direction at TRISB4_bit;
sbit LCD_D5_Direction at TRISB3_bit;
sbit LCD_D4_Direction at TRISB2_bit;

void main() {

      LCD_Init();
      Lcd_Cmd(_LCD_CURSOR_OFF);
      Lcd_Out(2, 2, "Hello!");
      Delay_ms(2000);
      Lcd_Cmd(_LCD_CLEAR);
}
```

Avec les instructions suivantes (Programme2) vous pouvez faire un affichage depuis le commencement de l'écran LCD.

Programme2 :

```
lcd_out_cp ("M1 "); //    CP Afficher depuis le commencement de l'écran.
lcd_out_cp ("INB");
lcd_out_cp (" GBM");
```

- Modifier le schéma ainsi que le programme1 pour l'utilisation des entrés pour un affichage au niveau du LCD.

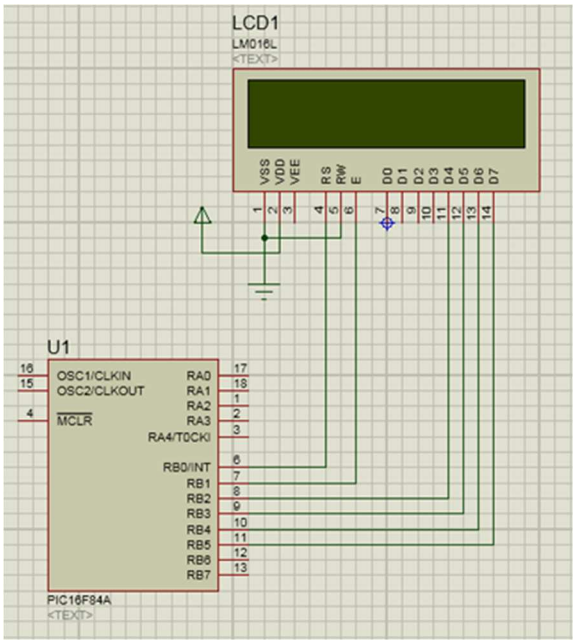


Programme3 :

```
TRISA=0xff;
LCD_Init();
Lcd_Cmd(_LCD_CURSOR_OFF);
for(;;){
while (PORTA.b0==0)  lcd_out(1,1,"ON");
while (PORTA.b1==0)  lcd_out(1,1,"Off");
Lcd_Cmd(_LCD_CLEAR);
```

- Le **programme4** permet de faire défiler un texte vers la droite avec quatre emplacements ensuite vers la gauche avec sept emplacements et a la fin vers la gauche avec sept emplacements, le temps du mouvement du texte est fixé avec un sous-programme **Void move delay.**
- Modifier le programme4 pour avoir un texte qui défile sans limite.

Programme4 :




```
char i;                          // Loop variable
 void Move_Delay() {        // Function used for text moving
  Delay_ms(500);     // You can change the moving speed here
}
void main() {
  LCD_Init();
  Lcd_Out(1,1,"M1 INB");            // Write text in first row
  Lcd_Out(2,5,"GBM");               // Write text in second row
   Delay_ms(2000);
  // Moving text
  for(i=0; i<4; i++) {            // Move text to the right 4 times
  Lcd_Cmd(_LCD_SHIFT_RIGHT);
  Move_Delay();
  }
while(1) {                         // Endless loop
   for(i=0; i<8; i++) {          // Move text to the left 7 times
     Lcd_Cmd(_LCD_SHIFT_LEFT);
     Move_Delay();
   }
for(i=0; i<8; i++) {           // Move text to the right 7 times
     Lcd_Cmd(_LCD_SHIFT_RIGHT);
     Move_Delay();
   }}}
```

3. Deuxième partie :

Objectifs : Gestion des interruptions et utilisation des sous-programme.

En informatique, une interruption est une suspension temporaire de l'exécution d'un programme informatique par le microprocesseur afin d'exécuter un programme prioritaire (appelé service d'interruption. Le PIC 16F84 utilise 3 types d'interruptions (Timer0, RB0/INT ou l'ensemble des RB qui restent).

Configuration du Pic pour utiliser les interruptions.

Pour Configurer le Pic pour qu'il utilise l'interruption RB0 il faut fermer GIE et INTE (mettre à 1) (Figure.6). D'où l'utilisation du registre INTCON :

-INTF représente le Flag par défaut elle est égale à 0 une fois on choisit RB0 elle devient 1.

	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (x)
INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Figure.6. Registre INTCON

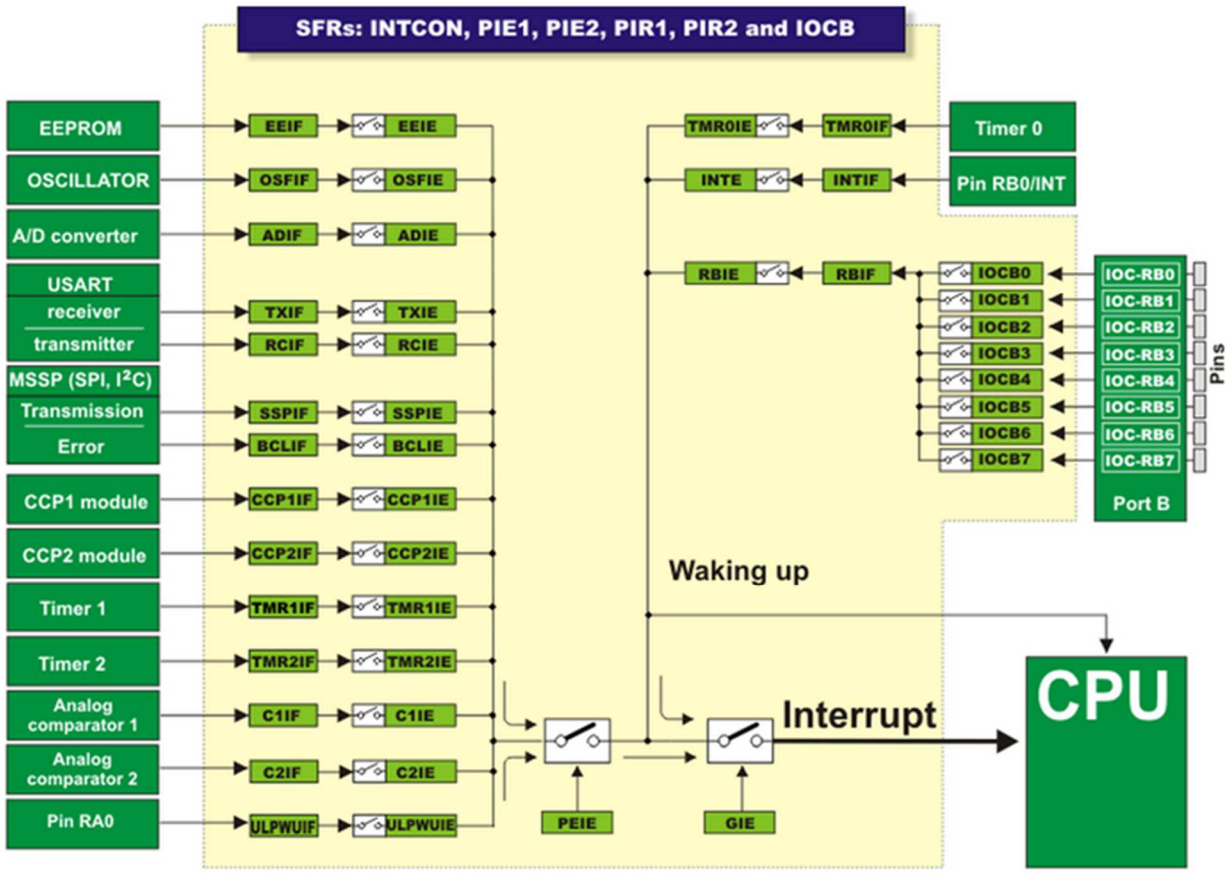
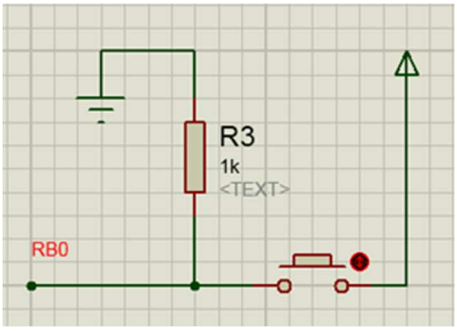


Figure.7. Interrupt System Registers

- Développer un circuit à base du PIC 16F84, afficheur LCD 16X2 et l'interrupteur ci-contre connecter au niveau de RB0.
- Lancer MikroC for Pic, puis éditer et compiler le programme en C suivant.
- Simuler le circuit précédant, en lui associant le programme donné.
- Modifier le programme5 en ajoutant un deuxième interrupteur pour le réglage des minutes.

Programme5 (Utilisation d'une interruption avec RB0):



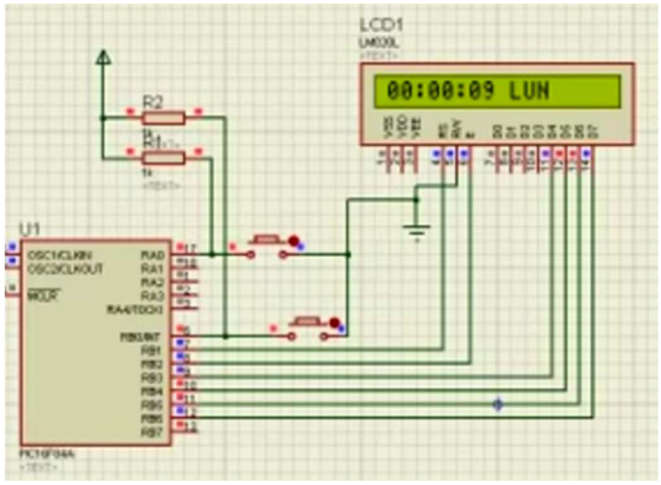

```
sbit LCD_RS at RB0_bit;
sbit LCD_EN at RB1_bit;
sbit LCD_D7 at RB5_bit;
sbit LCD_D6 at RB4_bit;
sbit LCD_D5 at RB3_bit;
sbit LCD_D4 at RB2_bit;
sbit LCD_RS_Direction at TRISB0_bit;
sbit LCD_EN_Direction at TRISB1_bit;
sbit LCD_D7_Direction at TRISB5_bit;
sbit LCD_D6_Direction at TRISB4_bit;
sbit LCD_D5_Direction at TRISB3_bit;
sbit LCD_D4_Direction at TRISB2_bit;

char s1, s2 ;

void interrupt ()
{
if ( INTCON.b1==1)  // pour ouvrire le FLAG.
     {
      s1++;
     }
INTCON.b1=0; // pour fermer le flag.
}
void main()
{
       INTCON=0b10010000 ;
       s1=s2=48; //code ascci de 0 initialisation
       LCD_Init();
       Lcd_Cmd(_LCD_CURSOR_OFF);
       Lcd_chr(1,8,s1); Lcd_chr(1,7,s2);
       for(;;)
{
delay_ms (1000);
s1++;
if(s1==58)   {s1=48;  s2++;}
if(s2==54)   {s1=48; s2=48;}
Lcd_chr(1,8,s1); Lcd_chr(1,7,s2);
}}
```

4. Troisième partie :

Objectifs : Utilisation du TIMER0, Application : Réalisation d'une montre numérique avec réglage des secondes, des minutes, des heurs et des jours.

- Développer un circuit à base du PIC 16F84, afficheur LCD 16X2 et des interrupteurs selon vos besoin de réglage.
- Lancer MikroC for Pic, puis éditer et compiler le **programme6** en C.
- éditer et compiler le **programme7** et donner la différence dans le fonctionnement entre les deux programmes 6 et 7.

Programme6 : (Montre numérique sans TIMER0)




```
sbit LCD_RS at RB1_bit;
sbit LCD_EN at RB2_bit;
sbit LCD_D7 at RB6_bit;
sbit LCD_D6 at RB5_bit;
sbit LCD_D5 at RB4_bit;
sbit LCD_D4 at RB3_bit;
sbit LCD_RS_Direction at TRISB1_bit;
sbit LCD_EN_Direction at TRISB2_bit;
sbit LCD_D7_Direction at TRISB6_bit;
sbit LCD_D6_Direction at TRISB5_bit;
sbit LCD_D5_Direction at TRISB4_bit;
sbit LCD_D4_Direction at TRISB3_bit;

 char s1,s2,p,m1,m2,h1,h2,j   ;
void interrupt ()
{
if ( INTCON.b1==1)  // pour ouvrir le FLAG. Pour détecter que la source de l'interruption est bien RB0
     {
        j++;
      }
INTCON.b1=0; // pour fermer le flag. Remettre le FLAG de la source RB0 à 0 pour pouvoir détecter une nouvelle interruption
}
void main()
{
      INTCON=0b10010000 ;
      S1=s2=m1=m2=h1=h2=48; //code ascci de 0 initialisation
      P=58 ; // code ascii de :
      J=1 ;
      LCD_Init();
      Lcd_Cmd(_LCD_CURSOR_OFF);
      Lcd_chr(1,8,s1); Lcd_chr(1,7,s2); Lcd_chr(1,6,p); Lcd_chr(1,5,m1); Lcd_chr(1,4,m2); Lcd_chr(1,3,p); Lcd_chr(1,2,h1); Lcd_chr(1,1,h2);// Affichage du début
      for(;;) // For san conditions
{
delay_ms (1000);
```

```
s1++;
if(s1==58)  {s1=48;  s2++;}
if(s2==54)  {s1=48; s2=48; m1++;}
if(m1==58)  { s1=48; s2=48;  m1=48; m2++;}
if(m2==54)  { s1=48; s2=48;  m1=48; m2=48; h1++;}
if(h1==58)  { s1= s2=m1=m2=h1=48; h2++;}
if(h2==50&&h1==52)  { s1=s2=m1=m2=h1=h2=48; j++;}
Lcd_chr(1,8,s1);  Lcd_chr(1,7,s2);  Lcd_chr(1,5,m1);  Lcd_chr(1,4,m2);  Lcd_chr(1,2,h1);
Lcd_chr(1,1,h2);
if (j==1) Lcd_out (1,10,"Lundi");
if (j==2) Lcd_out (1,10,"Mar");
if (j==3) Lcd_out (1,10,"Mer");
if (j==4) Lcd_out (1,10,"Jeu");
if (j==5)  Lcd_out (1,10,"Ven");
if (j==6) Lcd_out (1,10,"sam");
if (j==7)  Lcd_out (1,10,"Dim");
if (j==8) j=1;
}
}
```

Programme7 : (Montre numérique avec TIMER0)

```
sbit LCD_RS at RB1_bit;
sbit LCD_EN at RB2_bit;
sbit LCD_D7 at RB6_bit;
sbit LCD_D6 at RB5_bit;
sbit LCD_D5 at RB4_bit;
sbit LCD_D4 at RB3_bit;

// Pin direction
sbit LCD_RS_Direction at TRISB1_bit;
sbit LCD_EN_Direction at TRISB2_bit;
sbit LCD_D7_Direction at TRISB6_bit;
sbit LCD_D6_Direction at TRISB5_bit;
sbit LCD_D5_Direction at TRISB4_bit;
sbit LCD_D4_Direction at TRISB3_bit;
  char t;
 char s1,s2,p,m1,m2,h1,h2,j  ;

void interrupt ()
/*{
if ( INTCON.b1==1)  // pour ouvrire le FLAG.
    {
      h1++;  j++;
      }
INTCON.b1=0; // pour fermer le flag.
}*/
   if (INTCON.T0IF)
   {
```

```
   //TMR0=155;
   t++
   if (t==15){s1++;t=0}
   }
INTCON.T0IF=0;

void main()
{
      OPTION_reg=0b01010111;
      INTCON=0b10110100 ;
      TMR0=0;
         //  TMR0=155;
      t=0;
      S1=s2=m1=m2=h1=h2=48; //code ascci de 0 initialisation
      P=58 ; // code ascii de :
      J=1 ;
      LCD_Init();
      Lcd_Cmd(_LCD_CURSOR_OFF);
      Lcd_chr(1,8,s1); Lcd_chr(1,7,s2); Lcd_chr(1,6,p); Lcd_chr(1,5,m1); Lcd_chr(1,4,m2);
Lcd_chr(1,3,p); Lcd_chr(1,2,h1); Lcd_chr(1,1,h2);// Affichage du début
      for(;;) // For san conditions
{
//delay_ms (1000);
//s1++;
if(s1==58)  {s1=48;  s2++;}
if(s2==54)  {s1=48; s2=48; m1++;}
if(m1==58)  { s1=48; s2=48;  m1=48; m2++;}
if(m2==54)  { s1=48; s2=48;  m1=48; m2=48; h1++;}
if(h1==58)  { s1= s2=m1=m2=h1=48; h2++;}
if(h2==50&&h1==52)  { s1=s2=m1=m2=h1=h2=48; j++;}
Lcd_chr(1,8,s1);  Lcd_chr(1,7,s2);  Lcd_chr(1,5,m1);  Lcd_chr(1,4,m2);  Lcd_chr(1,2,h1);
Lcd_chr(1,1,h2);
if (j==1) Lcd_out (1,10,"Lun");
if (j==2) Lcd_out (1,10,"Mar");
if (j==3) Lcd_out (1,10,"Mer");
if (j==4) Lcd_out (1,10,"Jeu");
if (j==5)  Lcd_out (1,10,"Ven");
if (j==6) Lcd_out (1,10,"sam");
if (j==7)  Lcd_out (1,10,"Dim");
if (j==8) j=1;
}
}
```

TP N° 4 : Conversion analogique digitale avec le PIC 16F877

1. Généralités :

Le convertisseur analogique digitale (A/D) convertit le signal analogique présent sur une de les 8 entrées du PIC 16F877 en son équivalent numérique, codé sur 10 bits. Les broches AN2 et AN3 peuvent être utilisées comme références de tension ou comme entrées analogiques standard, les références de tension étant dans ce dernier cas prises sur les tensions d'alimentations du PIC : VDD et VSS. (VDD pour le + et VSS pour le -).

On peut donc numériser jusqu'à 8 signaux analogiques. Pas tous en même temps, bien sûr, étant donné qu'il n'y a qu'un seul module de conversion pour 8 signaux d'entrée multiplexés. Mais si vos signaux n'évoluent pas trop vite (fréquence basse), vous pouvez numériser le signal sur la patte AN0, puis celui sur AN1…

Les paramètres importants dont il faudra tenir compte sont :

- La résolution du convertisseur. Ici 10 bits, donc meilleur qu'un convertisseur 8 bits, mais moins précis qu'un 12 bits…
- Le temps de conversion.
- La rapidité d'évolution des signaux présents sur les entrées (leur fréquence pour des signaux périodiques).
- Le nombre de signaux à numériser.

En effet, pour un signal périodique, la fréquence d'échantillonnage doit être au moins deux fois supérieure à la fréquence du signal.

La conversion A/D fait appel au registre **ADCON1** (Figure.8) pour configurer les tensions de références, la sélection des entrées numériques et analogiques ainsi que au registre **ADCON0** pour le choix de la fréquence d'échantillonnage (Figure.8).

ADCON1 REGISTER (ADDRESS 9Fh)

R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	ADCS2	—	—	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 7 **ADFM:** A/D Result Format Select bit
1 = Right justified. Six (6) Most Significant bits of ADRESH are read as '0'.
0 = Left justified. Six (6) Least Significant bits of ADRESL are read as '0'.

ADCON1 REGISTER (ADRESS 9FH)

R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	ADCS2	—	—	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 3-0 **PCFG3:PCFG0:** A/D Port Configuration Control bits

PCFG <3:0>	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0	VREF+	VREF-	C/R
0000	A	A	A	A	A	A	A	A	VDD	VSS	8/0
0001	A	A	A	A	VREF+	A	A	A	AN3	VSS	7/1
0010	D	D	D	A	A	A	A	A	VDD	VSS	5/0
0011	D	D	D	A	VREF+	A	A	A	AN3	VSS	4/1
0100	D	D	D	D	A	D	A	A	VDD	VSS	3/0
0101	D	D	D	D	VREF+	D	A	A	AN3	VSS	2/1
011x	D	D	D	D	D	D	D	D	—	—	0/0
1000	A	A	A	A	VREF+	VREF-	A	A	AN3	AN2	6/2
1001	D	D	A	A	A	A	A	A	VDD	VSS	6/0
1010	D	D	A	A	VREF+	A	A	A	AN3	VSS	5/1
1011	D	D	A	A	VREF+	VREF-	A	A	AN3	AN2	4/2
1100	D	D	D	A	VREF+	VREF-	A	A	AN3	AN2	3/2
1101	D	D	D	D	VREF+	VREF-	A	A	AN3	AN2	2/2
1110	D	D	D	D	D	D	D	A	VDD	VSS	1/0
1111	D	D	D	D	VREF+	VREF-	D	A	AN3	AN2	1/2

A = Analog input D = Digital I/O
C/R = # of analog input channels/# of A/D voltage references

Figure.8. Configuration des entrées AN avec le registre ADCON1

ADCON0 REGISTER (ADDRESS 1Fh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/DONE	—	ADON
bit 7							bit 0

bit 5-3 **CHS2:CHS0:** Analog Channel Select bits
000 = Channel 0 (AN0)
001 = Channel 1 (AN1)
010 = Channel 2 (AN2)
011 = Channel 3 (AN3)
100 = Channel 4 (AN4)
101 = Channel 5 (AN5)
110 = Channel 6 (AN6)
111 = Channel 7 (AN7)

bit 2 **GO/DONE:** A/D Conversion Status bit
When ADON = 1:
1 = A/D conversion in progress (setting this bit starts the A/D conversion which is automatically cleared by hardware when the A/D conversion is complete)
0 = A/D conversion not in progress

bit 1 **Unimplemented:** Read as '0'

bit 0 **ADON:** A/D On bit
1 = A/D converter module is powered up
0 = A/D converter module is shut-off and consumes no operating current

ADCON0 REGISTER (ADDRESS 1Fh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/DONE	—	ADON
bit 7							bit 0

ADCON1 REGISTER (ADDRESS 9Fh)

R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	ADCS2	—	—	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 6 **ADCS2**: A/D Conversion Clock Select bit (ADCON1 bits in shaded area and in **bold**)

ADCON1 <ADCS2>	ADCON0 <ADCS1:ADCS0>	Clock Conversion
0	00	FOSC/2
0	01	FOSC/8
0	10	FOSC/32
0	11	FRC (clock derived from the internal A/D RC oscillator)
1	00	FOSC/4
1	01	FOSC/16
1	10	FOSC/64
1	11	FRC (clock derived from the internal A/D RC oscillator)

Figure.9. Configuration du registre ADCON0.

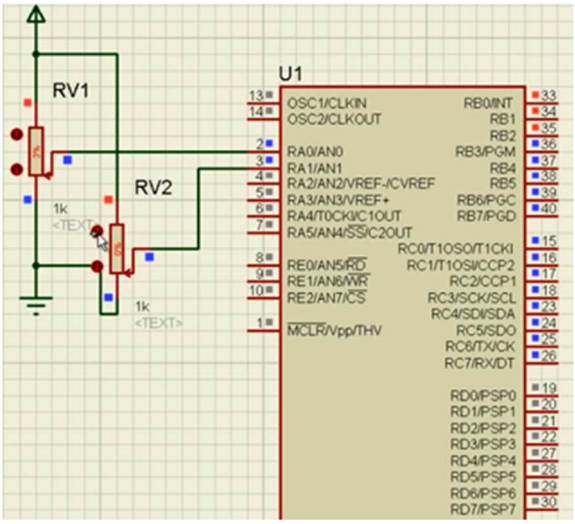
2. Première partie :

Objectifs : Utilisation du PIC 16f877 pour la conversion A/D.

- Développer un circuit à base du PIC 16F877 et d'un potentiomètre.
- Lancer MikroC for Pic, puis éditer et compiler le programme en C suivant.

- Simuler le circuit précédent, en lui associant le programme donné.

Désignation : POT HG (Pour la résistance variable)



Programme1 :

```
unsigned int a;  // déclarer une variable à 16 bits

void main() {
trisb=0x00;portb=0x00;
trisc=0x00;portc=0x00;
a=0;
ADC_Init();  // Initialiser module de ADC avec les paramètres par défaut (tous les RA sont des ADC)
//adcon1=0b10000001 ; //tension de référence + au niveau de RA3.

for(;;)
  {
a= ADC_Read(0);      // Obtenez des résultats 10 bits de conversion AD (0 entrée RA0)
 PORTB = a >> 2;       // Décalage à droite de 8 bits sur le portB

a= ADC_Read(1);     // Obtenez des résultats 10 bits de conversion AD (1 entrée RA1)
 PORTC = a >> 2;       // Décalage à droite de 8 bits sur le portC

}}
```

-Ajouter un troisième potentiomètre et donner le résultat de la conversion A/D sur les PORTD.

-Le convertisseur A/D réalisé est un convertisseur à 8 bits, programmer le pour qu'il soit à 4 bits.

-A jouter une tension de référence à l'entrée RA3 DE 10 V. et mettre le potentiomètre à 100%.

- Quelle est la valeur à la sortie du convertisseur A/D (PORTB). Si vous travaillez avec un ADC 10 bits.

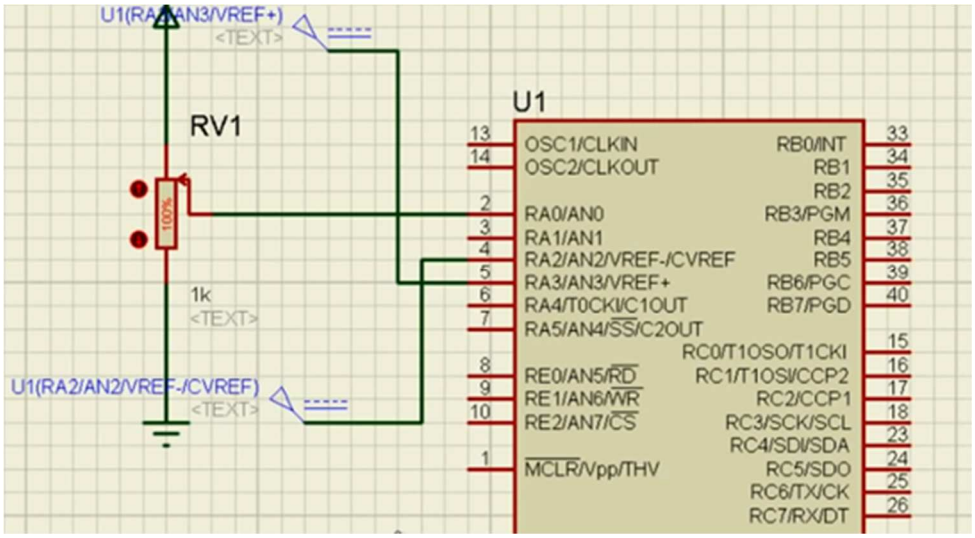
3. Deuxième partie :

Objectifs : Utilisation des tensions de références.

Ajouter deux tensions de référence (VREF- 4V et VREF+ 5V) avec l'instruction suivante :

//adcon1=0b10001000 ; // deux tension de référence VREF- < VREF+

- Quelle est la valeur à la sortie du convertisseur A/D (PORTB). Si vous travaillez avec un ADC 10 bits.



- Développer un circuit à base du PIC 16F877, un potentiomètre et un afficheur 7segments pour un affichage des tensions converties.

TP N° 5 : Gestion du contage et minuterie, sur la base de registre TMR0

(Application sur le microcontrôleur PIC 16F887)

1. Généralités :

Dans cette application de la minuterie, le registre TMR0, est utilisé comme un compteur. L'entrée de comptage est reliée à un bouton-poussoir **Input (**Figure. 10) de sorte que toute pression sur **le bouton poussoir** provoque un changement sur le registre TMR0 et commence à compter les impulsions.

Lorsque le nombre d'impulsions correspond au nombre stocké dans le registre nommé TEST, la valeur logique 1 (5V) apparaît sur le pin3 du PORTD. Cette tension active un relais électromécanique, par le bit « b3 du PORTD » dans le programme C (voire le programme1).

Dans le registre TEST est stocké un nombre de 5 pour cet exemple. Bien sûr, il peut être n'importe quel nombre défini comme une constante. Par ailleurs, le microcontrôleur peut activer un autre appareil au lieu de relais, tandis que le capteur peut être utilisé à la place du bouton-poussoir. Cet exemple illustre l'une des applications les plus courantes du microcontrôleur dans l'industrie ; quand quelque chose est effectué autant de fois que nécessaire, puis quelque chose d'autre doit être activé ou désactivé.

2. Application :

Objectifs : Utilisation des moteurs-DC

Désignation :

- **Microcontrôleur : 16F887**
- **Oscillateur : HS, 10.0000 Mhz**
- **Relai : G5C-1-DC5**
- **Battery : 12V**

- **Motor-DC**

- Développer le circuit à base du PIC 16F887.
- Lancer MikroC for Pic, puis éditer et compiler le programme en C suivant.
- Simuler le circuit précédent, en lui associant le programme donné.

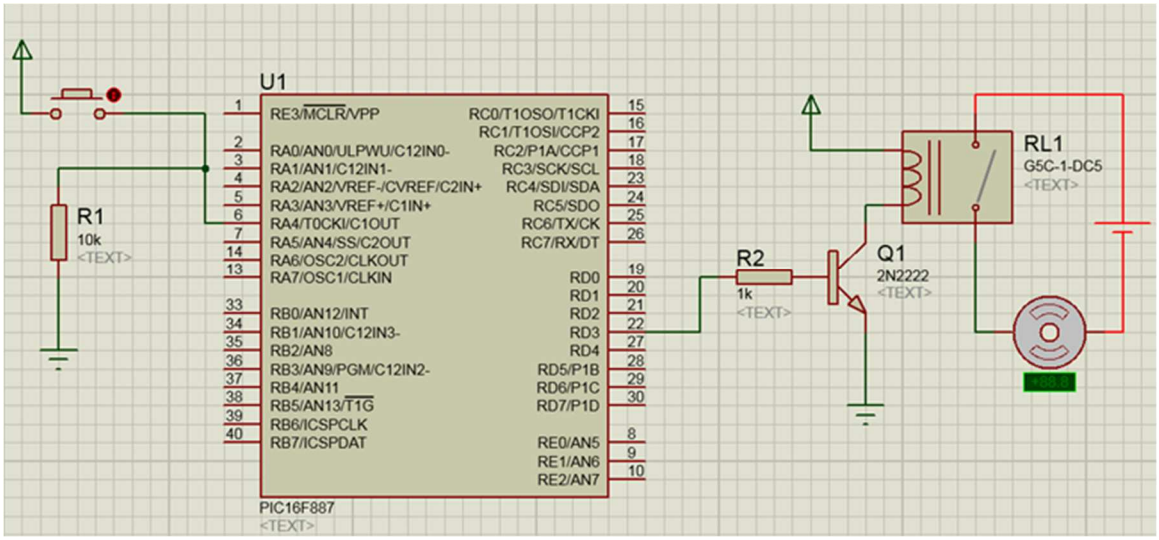


Figure. 10 : Utilisation des moteurs DC

Programme1 :

```
char TEST = 5;            // Constante TEST = 5
void main() {
PORTA = 0;               // Initialisation du porte A
TRISA = 0xFF;             // Porte A est configuré en entré
PORTD = 0;               // Initialisation du PORTD
TRISD = 0b11110111;          // Broche 3 du PORTD3 est configuré en sortie
OPTION_REG.b5 = 1;           // Compteur TMR0 reçoit des impulsions par la broche RA4
OPTION_REG.b3 = 1;           // Taux de pré-diviseur est de 1:1
TMR0 = 0;                // Initialisation  du compteur TMR0
for(;;) {                //  la boucle sans fin
if (TMR0 == TEST)  // Est-ce que le nombre d'impulsion TMR0 est égale à constante TEST?
*/
(PORTD.b3 = 1);  // Nombre est égale à TEST. La broche RD3 est en 1(RELAIS est activé) */
}}
```

- Modifier le programme pour arrêter le moteur DC.
- Réaliser une application avec le décodeur 74ls48 pour l'affichage sur un afficheur 7 segments après un comptage utilisant le TIMR0.

TP N° 6 : Génération des signaux carrés ; Utilisation du module CCP (CAPTURE COMPARE et PWM)

1. Généralités :

Les deux modules CCP sur le PIC 16F877. CCP pour « Capture, Compare, PWM ». En association avec les deux Timers, ils vont nous permettre de générer des signaux à modulation de largeur d'impulsion (PWM) pour, par exemple, faire varier la vitesse d'un moteur à courant continu, réguler le courant (et donc la luminosité) dans une ampoule…

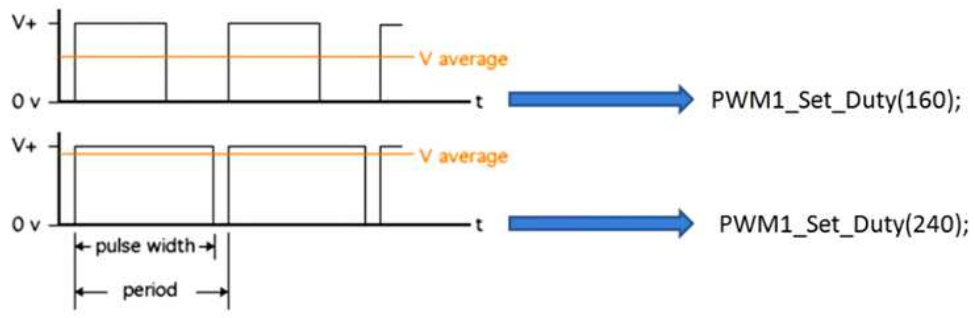
Ils vont également nous permettre de comparer l'occurrence d'un signal en entrée avec la valeur du compteur Timer1, réalisant ainsi un chronométrage de l'événement en question (par exemple : indication de la fréquence de rotation d'un moteur).

Ils vont encore nous permettre de générer des signaux carrés, et cela de manière quasi-indépendante du reste du microcontrôleur qui pourra continuer à vaquer à ses occupations.

2. Application :

Objectifs : Génération des signaux carrés.

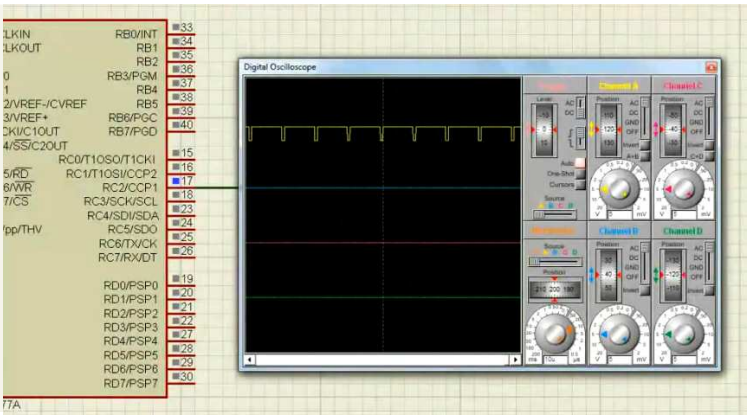
Un signal PWM est caractérisé par une période, et un temps de travail ou le signal est à "1". Ce temps est appelé **DUTY CYCLE**.



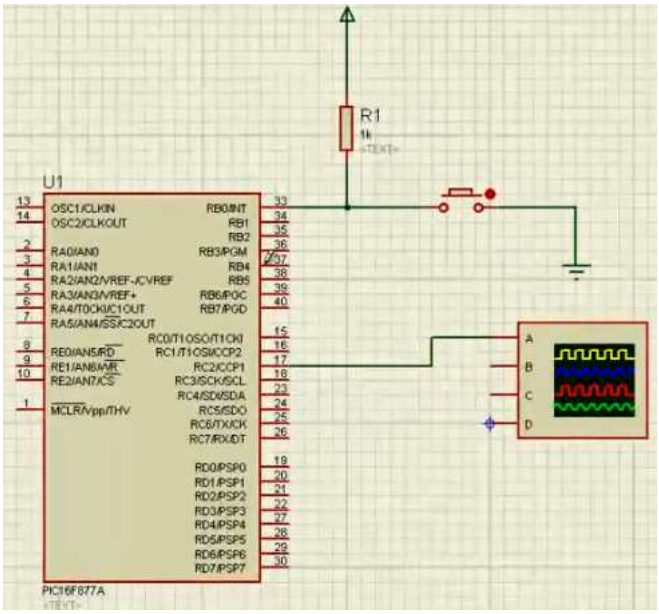
- Développer le circuit à base du PIC 16F877.
- Lancer MikroC for Pic, puis éditer et compiler le programme en C suivant.
- Simuler le circuit précédent, en lui associant le programme donné.

Programme1 :

```
char a;

void main() {
a=10;
trisb.b0=1;
PWM1_Init(37000);
PWM1_Set_Duty(a);
for(;;)
{
PWM1_Start();
if (portb.b0==0) {delay_ms (100); a=a+10};
PWM1_Set_Duty(a);
}
}
```

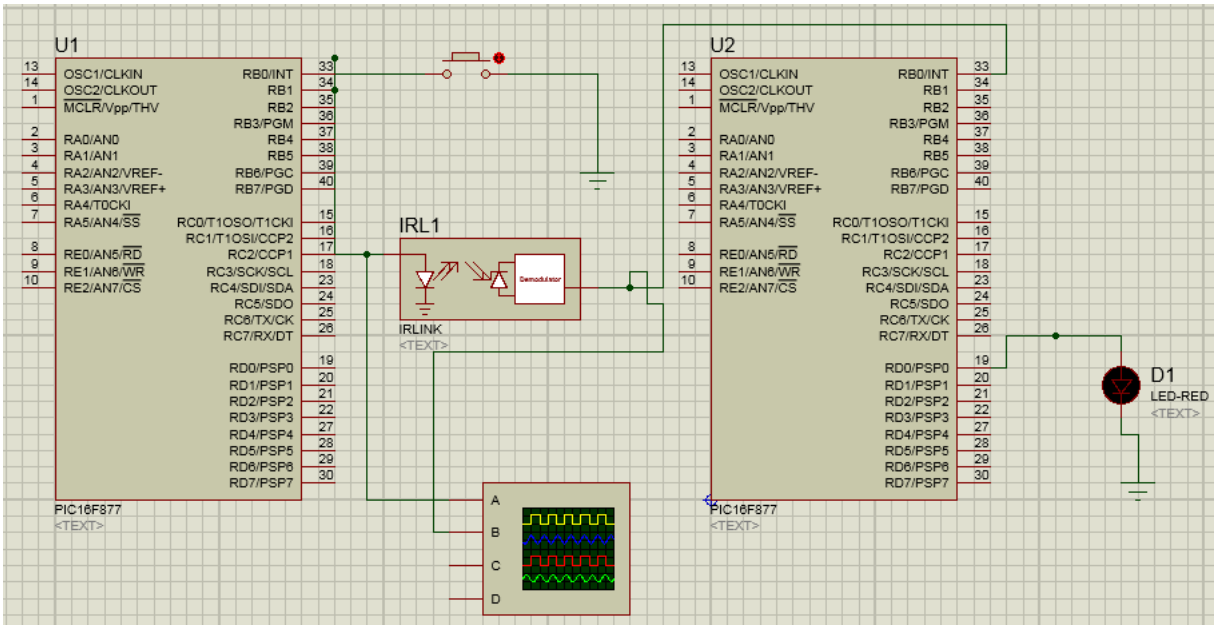


-Utilisation de l'instrument de simulation : **Oscilloscope**
- Affecter une sonde à un graphe du signal à visualiser.
- Puis glisser la sonde dans le graphe.
- Pour lancer la simulation, le graphe doit être pointé tout en appuyant sur louche **espace**.
- Un double clic sur le graphe permet de fixer le temps de simulation (200ms).
- Si l'oscilloscope ne s'affiche pas, Cliquer sur « debug / Oscilloscope »
-Modifier le programme précèdent pour générer un signal carré au niveau de CCP1 (RC2) avec un interrupteur comme illustre la figure suivante :



- Développer le circuit à base du PIC 16F877 pour l'émission/réception IR
- Lancer MikroC for Pic, puis éditer et compiler le programme en C.
- Simuler le circuit précédant, en lui associant le programme éditer.

Désignation : IRLINK (Module d'émission /réception IR)



Remerciements

Je tiens à remercier Mr BECHAR HASSENE et Mr DIB NABIL pour tout l'intérêt qu'ils ont témoigné pour examiner ce travail.

Je tiens en outre à exprimer ma reconnaissance et ma sympathie à tous ceux qui m'ont témoigné amitié et patience le long de mes années d'enseignement.

Enfin, un grand merci pour mes parents pour leurs soutiens durant toutes ces années. Je ne saurais être qu'infiniment reconnaissant quant aux sacrifices qu'ils ont consentis. Un merci spécial à ma femme, qui m'a encouragée, soutenue et motivée sans cesse pour arriver au bout de ce travail, un grand merci pour tout.

Références

- Christian Tavernier, Programmation en C des PIC, Edition Dunod, France, 2005.

- V. Tourtchine, Programmation en mikroC. Application pour les microcontrôleurs de la famille PIC, Travaux Pratiques, Département Physique, Université M'Hamed Bogara, Boumerdes, 2012.

- Jacques Weiss, Microcontrôleurs PIC (Cas du 16F628), Suplec Campus de Renes ; Février 2002.

- Christian Dupaty Système à Microcontrôleurs, Edition CMP, Georges Charpak, Ecole Nationale Supérieur des Mines, 2010.

- Sylvain Montagny, Microprocesseurs et Microcontrôleurs, Université de Savoie, France, 2014.

- V. Tourtchine, Microcontrôleur de la famille PIC, Support de cours et prise en main du logiciel MPLAB, Département Physique, Université M'Hamed Bogara, Boumerdes, 2009.

- Hassen Jedid et Hichem Bargaoui, Architecture des microcontrôleurs, Edition Esprit, Ecole Supérieur Privée d'ingénierie et de technologies. 2011.

- Jerome Vicente, Les microcontrôleurs, Dpt ME, Option SIIC 2ème année, Ecole polytechnique, Université de Merseille,2006

- Philippe Letenneur, Les microcontroleurs PIC 16F87x, GRANVILLE, 2003.

- F. Senny, Introduction aux microcontrôleurs et à leur assembleur Illustration par le PIC16F877, Université de Liège, Faculté des Sciences Appliquées, 2007.

Annexes

1- LCD Data Sheet

XIAMEN AMOTEC DISPLAY CO.,LTD

SPECIFICATIONS OF LCD MODULE

MODULE NO : ADM1602K-NSW-FBS/3.3V

DOC.REVISION: 00

	SIGNATURE	DATE
PREPARED BY (RD ENGINEER)	QIU	2008-10-29
CHECKED BY	Chen	2008-10-29
APPROVED BY	yfe	2008-10-29

1. Features

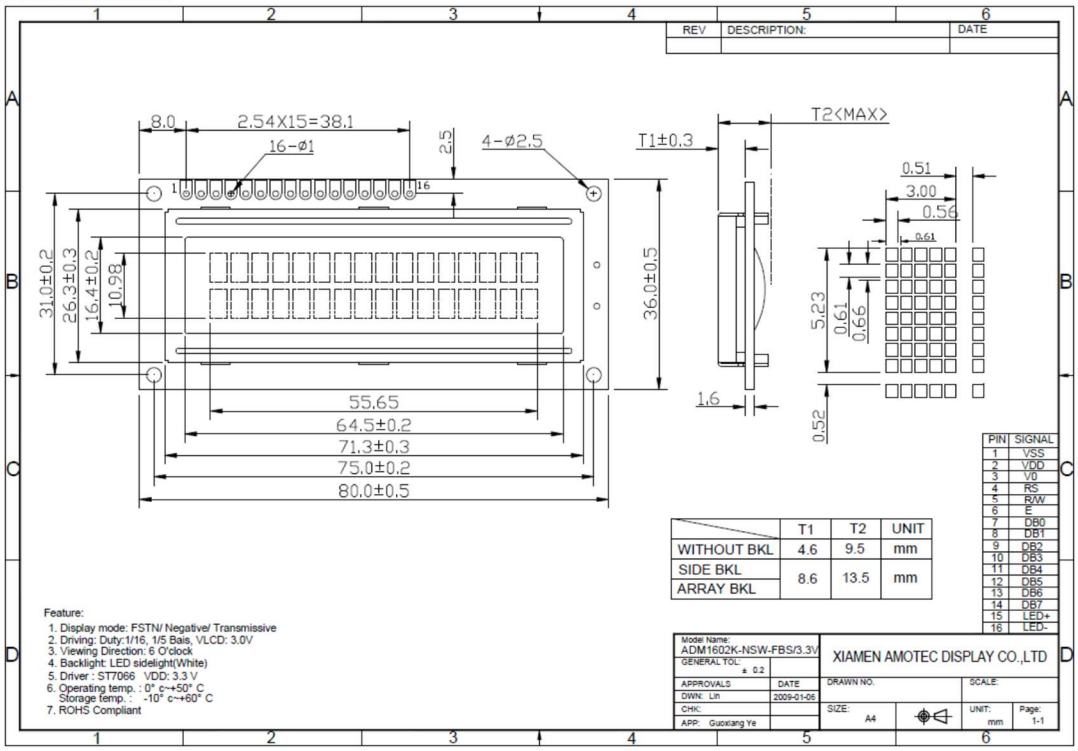
1. 5x8 dots with cursor
2. 16characters *2lines display
3. 4-bit or 8-bit MPU interfaces
4. Built-in controller (ST7066 or equivalent)
5. Display Mode & Backlight Variations
6. ROHS Compliant

LCD type	☐TN			
	☐FSTN	☑FSTN Negative		
	☐STN Yellow Green	☐STN Gray	☐STN Blue Negative	
View direction	☑6 O'clock	☐12 O'clock		
Rear Polarizer	☐Reflective	☐Transflective	☑Transmissive	
Backlight Type	☑LED	☐EL	☐Internal Power	☑3.3V Input
		☐CCFL	☑External Power	☐5.0V Input
Backlight Color	☑White	☐ Blue	☐ Amber	☐Yellow-Green
Temperature Range	☑Normal	☐Wide		☐Super Wide
DC to DC circuit	☐Build-in		☑Not Build-in	
Touch screen	☐With		☑Without	
Font type	☑English-Japanese	☐English-Europen	☐English-Russian	☐other

2. MECHANICAL SPECIFICATIONS

Module size	80.0mm(L)*36.0mm(W)* Max13.5(H)mm
Viewing area	64.5mm(L)*16.4mm(W)
Character size	3.00mm(L)*5.23mm(W)
Character pitch	3.51mm(L)*5.75mm(W)
Weight	Approx.

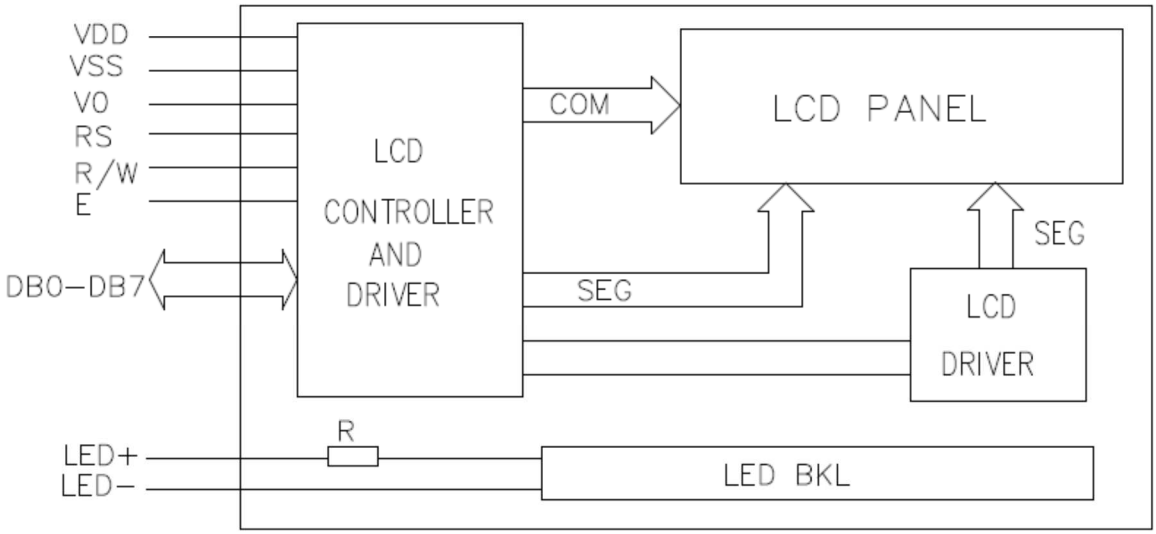
3. Outline dimension



4. Absolute maximum ratings

Item	Symbol	Standard			Unit
Power voltage	V_{DD}-V_{SS}	0	-	7.0	V
Input voltage	V_{IN}	VSS	-	VDD	
Operating temperature range	V_{OP}	0	-	+50	℃
Storage temperature range	V_{ST}	-10	-	+60	

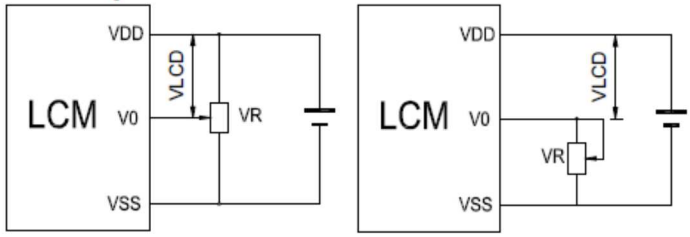
5. Block diagram



6. Interface pin description

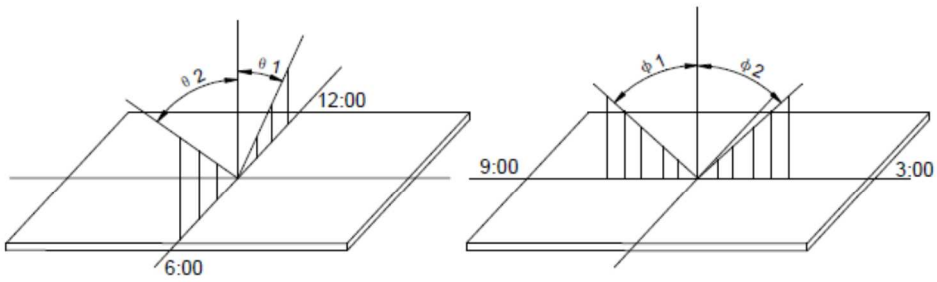
Pin no.	Symbol	External connection	Function
1	Vss	Power supply	Signal ground for LCM
2	V_{DD}		Power supply for logic for LCM
3	V_0		Contrast adjust
4	RS	MPU	Register select signal
5	R/W	MPU	Read/write select signal
6	E	MPU	Operation (data read/write) enable signal
7~10	DB0~DB3	MPU	Four low order bi-directional three-state data bus lines. Used for data transfer between the MPU and the LCM. These four are not used during 4-bit operation.
11~14	DB4~DB7	MPU	Four high order bi-directional three-state data bus lines. Used for data transfer between the MPU
15	LED+	LED BKL power supply	Power supply for BKL
16	LED-		Power supply for BKL

7. Contrast adjust



V_{DD}-V_0: LCD Driving voltage VR: 10k~20k

8. Optical characteristics



STN type display module (Ta=25℃, VDD=3.3V)

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Viewing angle	θ 1	$C_r \geqslant 3$		20		deg
	θ 2			40		
	Φ 1			35		
	Φ 2			35		
Contrast ratio	C_r		-	10	-	-
Response time (rise)	T_r	-	-	200	250	ms
Response time (fall)	T_r	-	-	300	350	

9. Electrical characteristics

DC characteristics

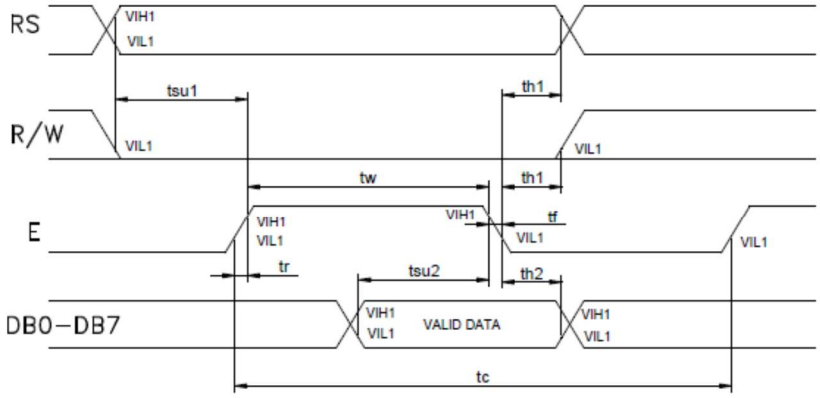
Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Supply voltage for LCD	V_{DD}-V_0	Ta =25℃	-	3.0	-	V
Input voltage	V_{DD}		3.1	3.3	3.5	
Supply current	I_{DD}	Ta=25℃, V_{DD}=3.3V	-	1.5	2.5	mA
Input leakage current	I_{LKG}		-	-	1.0	uA
"H" level input voltage	V_{IH}		2.2	-	V_{DD}	V
"L" level input voltage	V_{IL}	Twice initial value or less	0	-	0.6	
"H" level output voltage	V_{OH}	LOH=-0.25mA	2.4	-	-	
"L" level output voltage	V_{OL}	LOH=1.6mA	-	-	0.4	
Backlight supply voltage	V_F		-	3.0		
Backlight supply current	I_{LED}	V_{LED}=3.3 V R=25 Ω			16	mA

10. Timing Characteristics

Write cycle (Ta=25℃, VDD=3.3V)

Parameter	Symbol	Test pin	Min.	Typ.	Max.	Unit
Enable cycle time	t_c	E	500	-	-	ns
Enable pulse width	t_w		300	-	-	
Enable rise/fall time	t_r, t_f		-	-	25	
RS; R/W setup time	t_{su1}	RS; R/W	100	-	-	
RS; R/W address hold time	t_{h1}	RS; R/W	10	-	-	
Read data output delay	t_{su2}	DB0~DB7	60	-	-	
Read data hold time	t_{h2}		10	-	-	

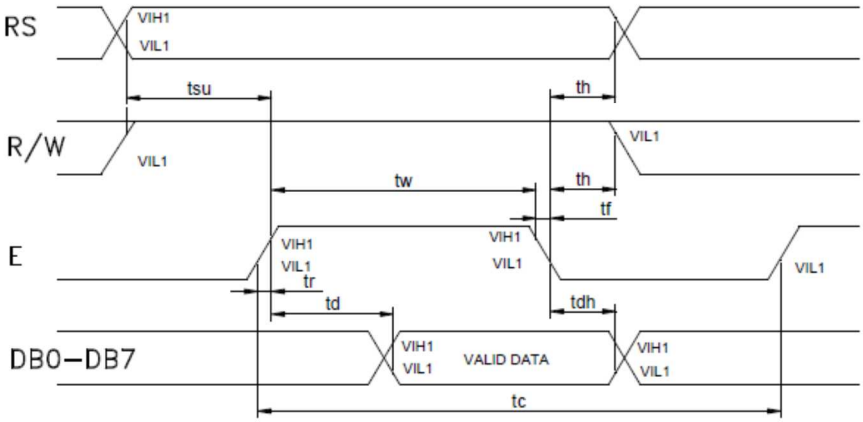
Write mode timing diagram



Read cycle (Ta=25℃, VDD=3.3V)

Parameter	Symbol	Test pin	Min.	Typ.	Max.	Unit
Enable cycle time	t_c	E	500	-	-	ns
Enable pulse width	t_w		300	-	-	
Enable rise/fall time	t_r, t_f		-	-	25	
RS; R/W setup time	t_{su}	RS; R/W	100	-	-	
RS; R/W address hold time	t_h	RS; R/W	10	-	-	
Read data output delay	t_d	DB0~DB7	60	-	90	
Read data hold time	t_{dh}		20	-	-	

Read mode timing diagram



11. FUNCTION DESCRIPTION

11.1 System Interface

This chip has all two kinds of interface type with MPU : 4-bit bus and 8-bit bus. 4-bit bus and 8-bit bus is selected by DL bit in the instruction register.

11.2 Busy Flag (BF)

When BF = "High", it indicates that the internal operation is being processed. So during this time the next instruction cannot be accepted. BF can be read, when RS = Low and R/W = High (Read Instruction Operation), through DB7 port. Before executing the next instruction, be sure that BF is not high.

11.3 Address Counter (AC)

Address Counter (AC) stores DDRAM/CGRAM address, transferred from IR. After writing into (reading from) DDRAM/CGRAM, AC is automatically increased (decreased) by 1. When RS = "Low" and R/W = "High", AC can be read through DB0 – DB6 ports.

11.4 Display Data RAM (DDRAM)

DDRAM stores display data of maximum 80 x 8 bits (80 characters). DDRAM address is set in the address counter (AC) as a hexadecimal number.

Display position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
DDRAM address	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
DDRAM address	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F

11.5 CGROM (Character Generator ROM)

CGROM has a 5 x 8 dots 204 characters pattern and a 5 x 10 dots 32 characters pattern. CGROM has 204 character patterns of 5 x 8 dots.

11.6 CGRAM (Character Generator RAM)

CGRAM has up to 5 . 8 dot, 8 characters. By writing font data to CGRAM, user defined characters can be used.

Character Code (DDRAM Data)									CGRAM Address						Character Patterns (CGRAM Data)							
b8	b7	b6	b5	b4	b3	b2	b1	b0	b5	b4	b3	b2	b1	b0	b7	b6	b5	b4	b3	b2	b1	b0
0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	-	-	-	1	1	1	1	1
						0	0	0				0	0	1				0	0	1	0	0
						0	0	0				0	1	0				0	0	1	0	0
						0	0	0				0	1	1				0	0	1	0	0
						0	0	0				1	0	0				0	0	1	0	0
						0	0	0				1	0	1				0	0	1	0	0
						0	0	0				1	1	0				0	0	1	0	0
						0	0	0				1	1	1				0	0	0	0	0
0	0	0	0	0	-	0	0	1	0	0	1	0	0	0	-	-	-	1	1	1	1	0
						0	0	1				0	0	1				1	0	0	0	1
						0	0	1				0	1	0				1	0	0	0	1
						0	0	1				0	1	1				1	1	1	1	0
						0	0	1				1	0	0				1	0	1	0	0
						0	0	1				1	0	1				1	0	0	1	0
						0	0	1				1	1	0				1	0	0	0	1
						0	0	1				1	1	1				0	0	0	0	0

Relationship between CGRAM Addresses, Character Codes (DDRAM) and Character patterns (CGRAM Data)

Notes:

1. Character code bits 0 to 2 correspond to CGRAM address bits 3 to 5 (3 bits: 8 types).
2. CGRAM address bits 0 to 2 designate the character pattern line position. The 8th line is the cursor position

and its display is formed by a logical OR with the cursor. Maintain the 8th line data, corresponding to the cursor display position, at 0 as the cursor display. If the 8th line data is 1, 1 bit will light up the 8th line regardless of the cursor presence.
3. Character pattern row positions correspond to CGRAM data bits 0 to 4 (bit 4 being at the left).
4. As shown Table, CGRAM character patterns are selected when character code bits 4 to 7 are all 0. However, since character code bit 3 has no effect, the R display example above can be selected by either character code 00H or 08H.
5. 1 for CGRAM data corresponds to display selection and 0 to non-selection.
"-": Indicates no effect.

11.7 Cursor/Blink Control Circuit
It controls cursor/blink ON/OFF at cursor position.

11.8 Outline
To overcome the speed difference between the internal clock of ST7066 and the MPU clock, ST7066 performs internal operations by storing control in formations to IR or DR. The internal operation is determined according to the signal from MPU, composed of read/write and data bus (Refer to Table7).
Instructions can be divided largely into four groups:

1) ST7066 function set instructions (set display methods, set data length, etc.)
2) Address set instructions to internal RAM
3) Data transfer instructions with internal RAM
4) Others

The address of the internal RAM is automatically increased or decreased by 1.

Note: during internal operation, busy flag (DB7) is read "High".
Busy flag check must be preceded by the next instruction.

12.Standard character pattern

Upper 4bit / Lower 4bit	LLLL	LLLH	LLHL	LLHH	LHLL	LHLH	LHHL	LHHH	HLLL	HLLH	HLHL	HLHH	HHLL	HHLH	HHHL	HHHH
LLLL	CG RAM (1)															
LLLH	(2)															
LLHL	(3)															
LLHH	(4)															
LHLL	(5)															
LHLH	(6)															
LHHL	(7)															
LHHH	(8)															
HLLL	(1)															
HLLH	(2)															
HLHL	(3)															
HLHH	(4)															
HHLL	(5)															
HHLH	(6)															
HHHL	(7)															
HHHH	(8)															

13.3 Reliability of LCM

Reliability test condition:

Item	Condition	Time (hrs)	Assessment
High temp. Storage	80°C	48	No abnormalities in functions and appearance
High temp. Operating	70°C	48	
Low temp. Storage	-30°C	48	
Low temp. Operating	-20°C	48	
Humidity	40°C/ 90%RH	48	
Temp. Cycle	0°C ← 25°C→ 50°C (30 min ← 5 min → 30min)	10cycles	

Recovery time should be 24 hours minimum. Moreover, functions, performance and appearance shall be free from remarkable deterioration within 50,000 hours under ordinary operating and storage conditions room temperature (20±8°C), normal humidity (below 65% RH), and in the area not exposed to direct sun light.

13.4 Precaution for using LCD/LCM

LCD/LCM is assembled and adjusted with a high degree of precision. Do not attempt to make any alteration or modification. The followings should be noted.

General Precautions:

1. LCD panel is made of glass. Avoid excessive mechanical shock or applying strong pressure onto the surface of display area.
2. The polarizer used on the display surface is easily scratched and damaged. Extreme care should be taken when handling. To clean dust or dirt off the display surface, wipe gently with cotton, or other soft material soaked with isoproply alcohol, ethyl alcohol or trichlorotriflorothane, do not use water, ketone or aromatics and never scrub hard.
3. Do not tamper in any way with the tabs on the metal frame.
4. Do not make any modification on the PCB without consulting AMOTEC
5. When mounting a LCM, make sure that the PCB is not under any stress such as bending or twisting. Elastomer contacts are very delicate and missing pixels could result from slight dislocation of any of the elements.
6. Avoid pressing on the metal bezel, otherwise the elastomer connector could be deformed and lose contact, resulting in missing pixels and also cause rainbow on the display.
7. Be careful not to touch or swallow liquid crystal that might leak from a damaged cell. Any liquid crystal adheres to skin or clothes, wash it off immediately with soap and water.

Static Electricity Precautions:

1. CMOS LSI is used for the module circuit; therefore operators should be grounded whenever he/she comes into contact with the module.
2. Do not touch any of the conductive parts such as the LSI pads; the copper leads on the PCB and the interface terminals with any parts of the human body.

3. Do not touch the connection terminals of the display with bare hand; it will cause disconnection or defective insulation of terminals.
4. The modules should be kept in anti-static bags or other containers resistant to static for storage.
5. Only properly grounded soldering irons should be used.
6. If an electric screwdriver is used, it should be grounded and shielded to prevent sparks.
7. The normal static prevention measures should be observed for work clothes and working benches.
8. Since dry air is inductive to static, a relative humidity of 50-60% is recommended.

Soldering Precautions:

1. Soldering should be performed only on the I/O terminals.
2. Use soldering irons with proper grounding and no leakage.
3. Soldering temperature: 280°C±10°C
4. Soldering time: 3 to 4 second.
5. Use eutectic solder with resin flux filling.
6. If flux is used, the LCD surface should be protected to avoid spattering flux.
7. Flux residue should be removed.

Operation Precautions:

1. The viewing angle can be adjusted by varying the LCD driving voltage Vo.
2. Since applied DC voltage causes electro-chemical reactions, which deteriorate the display, the applied pulse waveform should be a symmetric waveform such that no DC component remains. Be sure to use the specified operating voltage.
3. Driving voltage should be kept within specified range; excess voltage will shorten display life.
4. Response time increases with decrease in temperature.
5. Display color may be affected at temperatures above its operational range.
6. Keep the temperature within the specified range usage and storage. Excessive temperature and humidity could cause polarization degradation, polarizer peel-off or generate bubbles.
7. For long-term storage over 40 C is required, the relative humidity should be kept below 60%,and avoid direct sunlight.

Limited Warranty

AMOTEC LCDs and modules are not consumer products, but may be incorporated by AMOTEC 's customers into consumer products or components thereof, AMOTEC does not warrant that its LCDs and components are fit for any such particular purpose.

1. The liability of AMOTEC is limited to repair or replacement on the terms set forth below. AMOTEC will not be responsible for any subsequent or consequential events or injury or damage to any personnel or user including third party personnel and/or user. Unless otherwise agreed in writing between AMOTEC and the customer, AMOTEC will only replace or repair any of its LCD which is found defective electrically or visually when inspected in accordance with AMOTEC general LCD inspection standard . (Copies available on request)
2. No warranty can be granted if any of the precautions state in handling liquid crystal display above has been disregarded. Broken glass, scratches on polarizer mechanical damages as well as defects that are caused accelerated environment tests are excluded from warranty.
3. In returning the LCD/LCM, they must be properly packaged; there should be detailed description of the failures or defect.

2- PIC 16F84A Data sheet

PIC16F84A
Data Sheet

18-pin Enhanced FLASH/EEPROM
8-bit Microcontroller

 DS35007B

MICROCHIP

PIC16F84A

18-pin *Enhanced* FLASH/EEPROM 8-Bit Microcontroller

High Performance RISC CPU Features:

- Only 35 single word instructions to learn
- All instructions single-cycle except for program branches which are two-cycle
- Operating speed: DC - 20 MHz clock input
  DC - 200 ns instruction cycle
- 1024 words of program memory
- 68 bytes of Data RAM
- 64 bytes of Data EEPROM
- 14-bit wide instruction words
- 8-bit wide data bytes
- 15 Special Function Hardware registers
- Eight-level deep hardware stack
- Direct, indirect and relative addressing modes
- Four interrupt sources:
  - External RB0/INT pin
  - TMR0 timer overflow
  - PORTB<7:4> interrupt-on-change
  - Data EEPROM write complete

Peripheral Features:

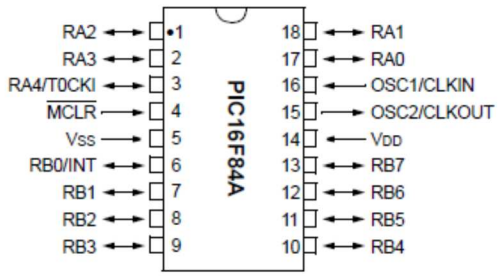
- 13 I/O pins with individual direction control
- High current sink/source for direct LED drive
  - 25 mA sink max. per pin
  - 25 mA source max. per pin
- TMR0: 8-bit timer/counter with 8-bit programmable prescaler

Special Microcontroller Features:

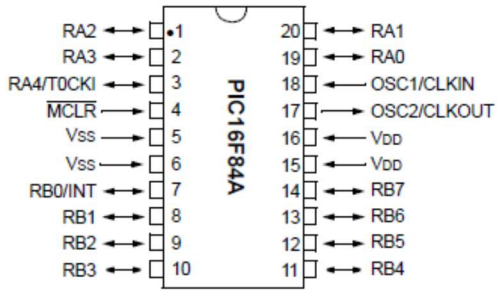
- 10,000 erase/write cycles *Enhanced* FLASH Program memory typical
- 10,000,000 typical erase/write cycles EEPROM Data memory typical
- EEPROM Data Retention > 40 years
- In-Circuit Serial Programming™ (ICSP™) - via two pins
- Power-on Reset (POR), Power-up Timer (PWRT), Oscillator Start-up Timer (OST)
- Watchdog Timer (WDT) with its own On-Chip RC Oscillator for reliable operation
- Code protection
- Power saving SLEEP mode
- Selectable oscillator options

Pin Diagrams

PDIP, SOIC



SSOP



CMOS Enhanced FLASH/EEPROM Technology:

- Low power, high speed technology
- Fully static design
- Wide operating voltage range:
  - Commercial: 2.0V to 5.5V
  - Industrial: 2.0V to 5.5V
- Low power consumption:
  - < 2 mA typical @ 5V, 4 MHz
  - 15 µA typical @ 2V, 32 kHz
  - < 0.5 µA typical standby current @ 2V

PIC16F84A

Table of Contents

TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@mail.microchip.com** or fax the **Reader Response Form** in the back of this data sheet to (480) 792-4150. We welcome your feedback.

Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Web site at:

http://www.microchip.com

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000A is version A of document DS30000).

Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

- Microchip's Worldwide Web site; http://www.microchip.com
- Your local Microchip sales office (see last page)
- The Microchip Corporate Literature Center; U.S. FAX: (480) 792-7277

When contacting a sales office or the literature center, please specify which device, revision of silicon and data sheet (include literature number) you are using.

Customer Notification System

Register on our web site at **www.microchip.com/cn** to receive the most current information on all of our products.

1.0 DEVICE OVERVIEW

This document contains device specific information for the operation of the PIC16F84A device. Additional information may be found in the PICmicro™ Mid-Range Reference Manual, (DS33023), which may be downloaded from the Microchip website. The Reference Manual should be considered a complementary document to this data sheet, and is highly recommended reading for a better understanding of the device architecture and operation of the peripheral modules.

The PIC16F84A belongs to the mid-range family of the PICmicro® microcontroller devices. A block diagram of the device is shown in Figure 1-1.

The program memory contains 1K words, which translates to 1024 instructions, since each 14-bit program memory word is the same width as each device instruction. The data memory (RAM) contains 68 bytes. Data EEPROM is 64 bytes.

There are also 13 I/O pins that are user-configured on a pin-to-pin basis. Some pins are multiplexed with other device functions. These functions include:

- External interrupt
- Change on PORTB interrupt
- Timer0 clock input

Table 1-1 details the pinout of the device with descriptions and details for each pin.

FIGURE 1-1: PIC16F84A BLOCK DIAGRAM

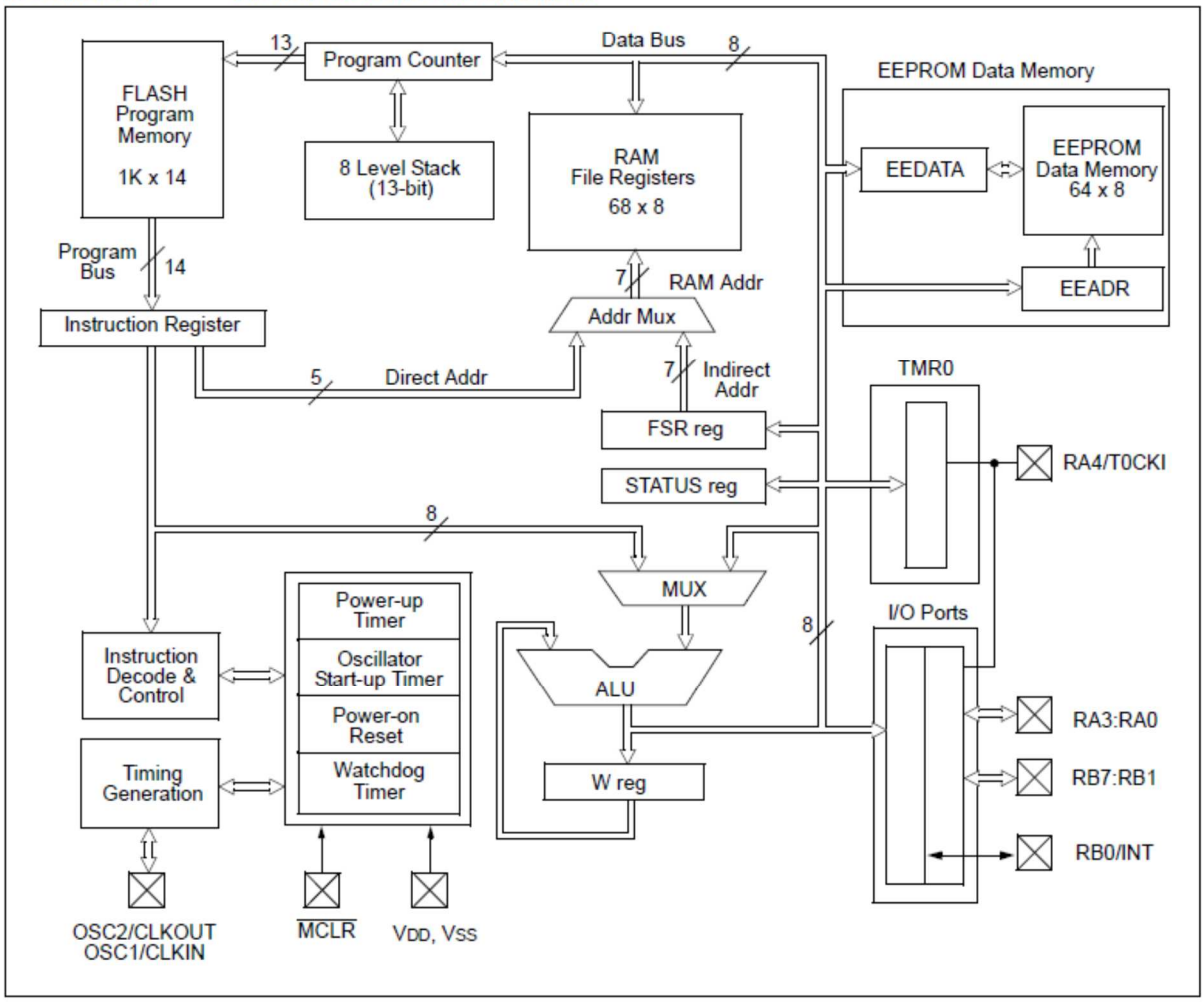


TABLE 1-1: PIC16F84A PINOUT DESCRIPTION

Pin Name	PDIP No.	SOIC No.	SSOP No.	I/O/P Type	Buffer Type	Description
OSC1/CLKIN	16	16	18	I	ST/CMOS(3)	Oscillator crystal input/external clock source input.
OSC2/CLKOUT	15	15	19	O	—	Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode. In RC mode, OSC2 pin outputs CLKOUT, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
$\overline{MCLR}$	4	4	4	I/P	ST	Master Clear (Reset) input/programming voltage input. This pin is an active low RESET to the device.
						PORTA is a bi-directional I/O port.
RA0	17	17	19	I/O	TTL	
RA1	18	18	20	I/O	TTL	
RA2	1	1	1	I/O	TTL	
RA3	2	2	2	I/O	TTL	
RA4/T0CKI	3	3	3	I/O	ST	Can also be selected to be the clock input to the TMR0 timer/counter. Output is open drain type.
						PORTB is a bi-directional I/O port. PORTB can be software programmed for internal weak pull-up on all inputs.
RB0/INT	6	6	7	I/O	TTL/ST(1)	RB0/INT can also be selected as an external interrupt pin.
RB1	7	7	8	I/O	TTL	
RB2	8	8	9	I/O	TTL	
RB3	9	9	10	I/O	TTL	
RB4	10	10	11	I/O	TTL	Interrupt-on-change pin.
RB5	11	11	12	I/O	TTL	Interrupt-on-change pin.
RB6	12	12	13	I/O	TTL/ST(2)	Interrupt-on-change pin. Serial programming clock.
RB7	13	13	14	I/O	TTL/ST(2)	Interrupt-on-change pin. Serial programming data.
VSS	5	5	5,6	P	—	Ground reference for logic and I/O pins.
VDD	14	14	15,16	P	—	Positive supply for logic and I/O pins.

Legend: I= input O = Output I/O = Input/Output P = Power
— = Not used TTL = TTL input ST = Schmitt Trigger input

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.
2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.
3: This buffer is a Schmitt Trigger input when configured in RC oscillator mode and a CMOS input otherwise.

2.0 MEMORY ORGANIZATION

There are two memory blocks in the PIC16F84A. These are the program memory and the data memory. Each block has its own bus, so that access to each block can occur during the same oscillator cycle.

The data memory can further be broken down into the general purpose RAM and the Special Function Registers (SFRs). The operation of the SFRs that control the "core" are described here. The SFRs used to control the peripheral modules are described in the section discussing each individual peripheral module.

The data memory area also contains the data EEPROM memory. This memory is not directly mapped into the data memory, but is indirectly mapped. That is, an indirect address pointer specifies the address of the data EEPROM memory to read/write. The 64 bytes of data EEPROM memory have the address range 0h-3Fh. More details on the EEPROM memory can be found in Section 3.0.

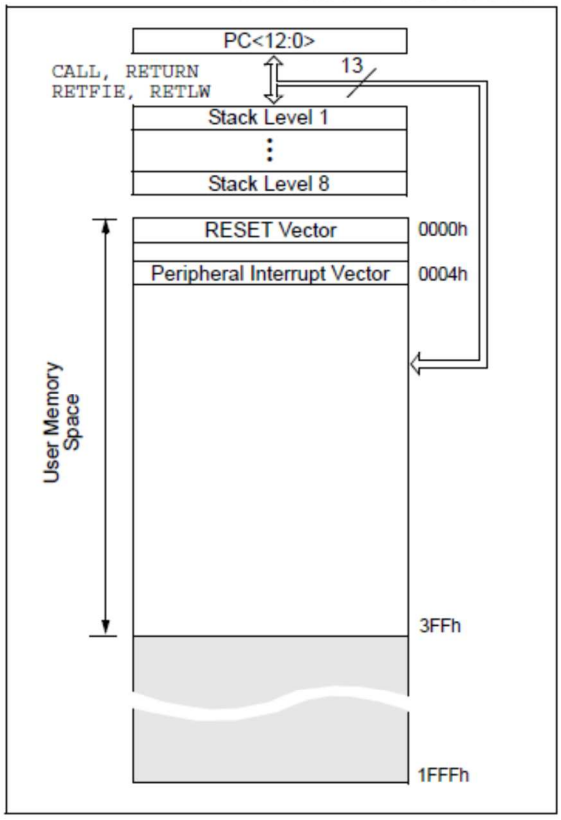
Additional information on device memory may be found in the PICmicro™ Mid-Range Reference Manual, (DS33023).

2.1 Program Memory Organization

The PIC16FXX has a 13-bit program counter capable of addressing an 8K x 14 program memory space. For the PIC16F84A, the first 1K x 14 (0000h-03FFh) are physically implemented (Figure 2-1). Accessing a location above the physically implemented address will cause a wraparound. For example, for locations 20h, 420h, 820h, C20h, 1020h, 1420h, 1820h, and 1C20h, the instruction will be the same.

The RESET vector is at 0000h and the interrupt vector is at 0004h.

FIGURE 2-1: PROGRAM MEMORY MAP AND STACK - PIC16F84A



2.2 Data Memory Organization

The data memory is partitioned into two areas. The first is the Special Function Registers (SFR) area, while the second is the General Purpose Registers (GPR) area. The SFRs control the operation of the device.

Portions of data memory are banked. This is for both the SFR area and the GPR area. The GPR area is banked to allow greater than 116 bytes of general purpose RAM. The banked areas of the SFR are for the registers that control the peripheral functions. Banking requires the use of control bits for bank selection. These control bits are located in the STATUS Register. Figure 2-2 shows the data memory map organization.

Instructions `MOVWF` and `MOVF` can move values from the W register to any location in the register file ("F"), and vice-versa.

The entire data memory can be accessed either directly using the absolute address of each register file or indirectly through the File Select Register (FSR) (Section 2.5). Indirect addressing uses the present value of the RP0 bit for access into the banked areas of data memory.

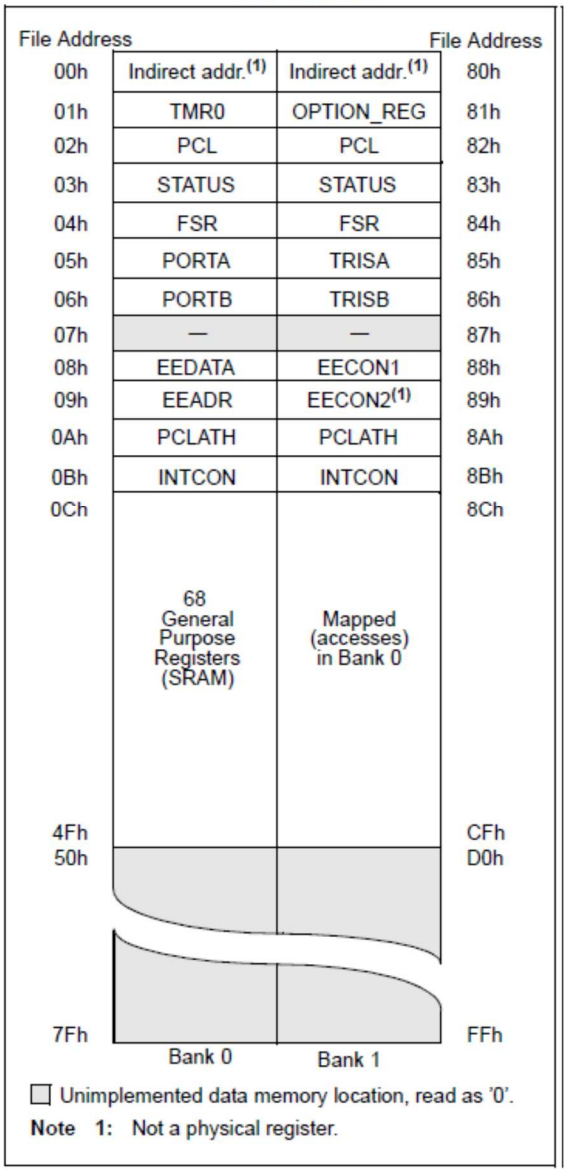
Data memory is partitioned into two banks which contain the general purpose registers and the special function registers. Bank 0 is selected by clearing the RP0 bit (STATUS<5>). Setting the RP0 bit selects Bank 1. Each Bank extends up to 7Fh (128 bytes). The first twelve locations of each Bank are reserved for the Special Function Registers. The remainder are General Purpose Registers, implemented as static RAM.

2.2.1 GENERAL PURPOSE REGISTER FILE

Each General Purpose Register (GPR) is 8-bits wide and is accessed either directly or indirectly through the FSR (Section 2.5).

The GPR addresses in Bank 1 are mapped to addresses in Bank 0. As an example, addressing location 0Ch or 8Ch will access the same GPR.

FIGURE 2-2: REGISTER FILE MAP - PIC16F84A



2.3 Special Function Registers

The Special Function Registers (Figure 2-2 and Table 2-1) are used by the CPU and Peripheral functions to control the device operation. These registers are static RAM.

The special function registers can be classified into two sets, core and peripheral. Those associated with the core functions are described in this section. Those related to the operation of the peripheral features are described in the section for that specific feature.

TABLE 2-1: SPECIAL FUNCTION REGISTER FILE SUMMARY

Addr	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on RESET	Details on page
Bank 0											
00h	INDF	Uses contents of FSR to address Data Memory (not a physical register)								---- ----	11
01h	TMR0	8-bit Real-Time Clock/Counter								xxxx xxxx	20
02h	PCL	Low Order 8 bits of the Program Counter (PC)								0000 0000	11
03h	STATUS(2)	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	8
04h	FSR	Indirect Data Memory Address Pointer 0								xxxx xxxx	11
05h	PORTA(4)	—	—	—	RA4/T0CKI	RA3	RA2	RA1	RA0	---x xxxx	16
06h	PORTB(5)	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0/INT	xxxx xxxx	18
07h	—	Unimplemented location, read as '0'								—	—
08h	EEDATA	EEPROM Data Register								xxxx xxxx	13,14
09h	EEADR	EEPROM Address Register								xxxx xxxx	13,14
0Ah	PCLATH	—	—	—	Write Buffer for upper 5 bits of the PC(1)					---0 0000	11
0Bh	INTCON	GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	10
Bank 1											
80h	INDF	Uses Contents of FSR to address Data Memory (not a physical register)								---- ----	11
81h	OPTION_REG	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	9
82h	PCL	Low order 8 bits of Program Counter (PC)								0000 0000	11
83h	STATUS (2)	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	8
84h	FSR	Indirect data memory address pointer 0								xxxx xxxx	11
85h	TRISA	—	—	—	PORTA Data Direction Register					---1 1111	16
86h	TRISB	PORTB Data Direction Register								1111 1111	18
87h	—	Unimplemented location, read as '0'								—	—
88h	EECON1	—	—	—	EEIF	WRERR	WREN	WR	RD	---0 x000	13
89h	EECON2	EEPROM Control Register 2 (not a physical register)								---- ----	14
0Ah	PCLATH	—	—	—	Write buffer for upper 5 bits of the PC(1)					---0 0000	11
0Bh	INTCON	GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	10

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0', q = value depends on condition

Note 1: The upper byte of the program counter is not directly accessible. PCLATH is a slave register for PC<12:8>. The contents of PCLATH can be transferred to the upper byte of the program counter, but the contents of PC<12:8> are never transferred to PCLATH.

2: The $\overline{TO}$ and $\overline{PD}$ status bits in the STATUS register are not affected by a $\overline{MCLR}$ Reset.

3: Other (non power-up) RESETS include: external RESET through $\overline{MCLR}$ and the Watchdog Timer Reset.

4: On any device RESET, these pins are configured as inputs.

5: This is the value that will be in the port output latch.

9.0 ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings †

Ambient temperature under bias ..... -55°C to +125°C

Storage temperature ..... -65°C to +150°C

Voltage on any pin with respect to V_{SS} (except V_{DD}, $\overline{MCLR}$, and RA4) ..... -0.3V to (V_{DD} + 0.3V)

Voltage on V_{DD} with respect to V_{SS} ..... -0.3 to +7.5V

Voltage on $\overline{MCLR}$ with respect to V_{SS} (1) ..... -0.3 to +14V

Voltage on RA4 with respect to V_{SS} ..... -0.3 to +8.5V

Total power dissipation (2) ..... 800 mW

Maximum current out of V_{SS} pin ..... 150 mA

Maximum current into V_{DD} pin ..... 100 mA

Input clamp current, I_{IK} (V_I < 0 or V_I > V_{DD}) ..... ± 20 mA

Output clamp current, I_{OK} (V_O < 0 or V_O > V_{DD}) ..... ± 20 mA

Maximum output current sunk by any I/O pin ..... 25 mA

Maximum output current sourced by any I/O pin ..... 25 mA

Maximum current sunk by PORTA ..... 80 mA

Maximum current sourced by PORTA ..... 50 mA

Maximum current sunk by PORTB ..... 150 mA

Maximum current sourced by PORTB ..... 100 mA

Note 1: Voltage spikes below V_{SS} at the $\overline{MCLR}$ pin, inducing currents greater than 80 mA, may cause latch-up. Thus, a series resistor of 50-100Ω should be used when applying a "low" level to the $\overline{MCLR}$ pin rather than pulling this pin directly to V_{SS}.

2: Power dissipation is calculated as follows: $P_{dis} = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD}-V_{OH}) \times I_{OH}\} + \sum(V_{OL} \times I_{OL})$.

† NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

FIGURE 9-1: PIC16F84A-20 VOLTAGE-FREQUENCY GRAPH

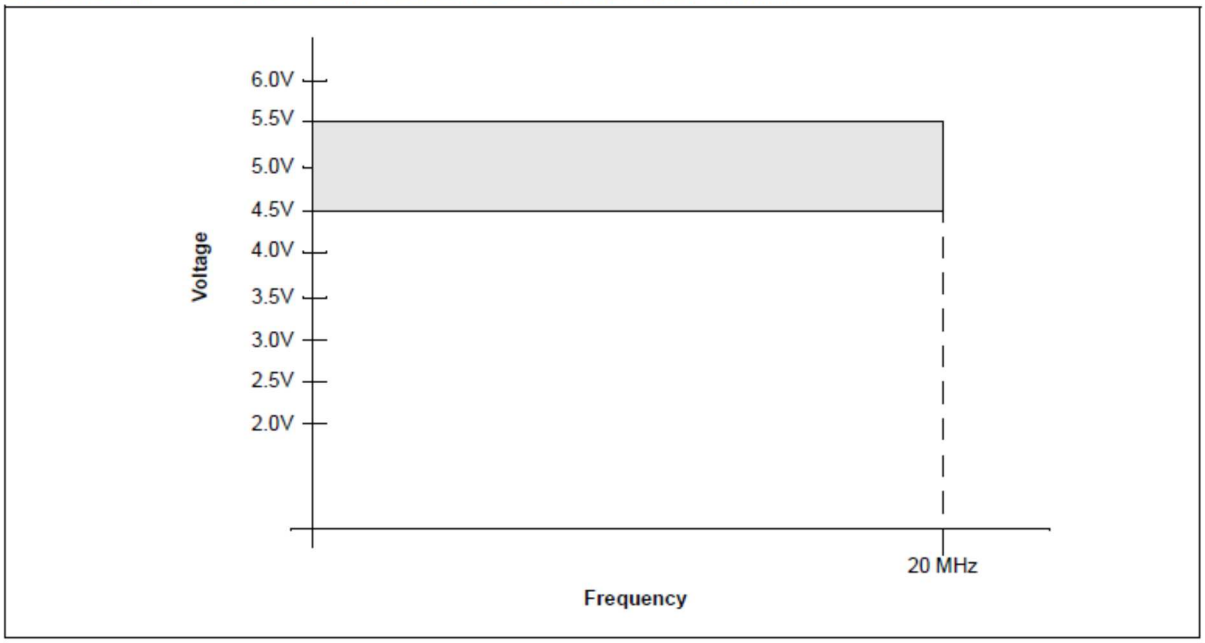
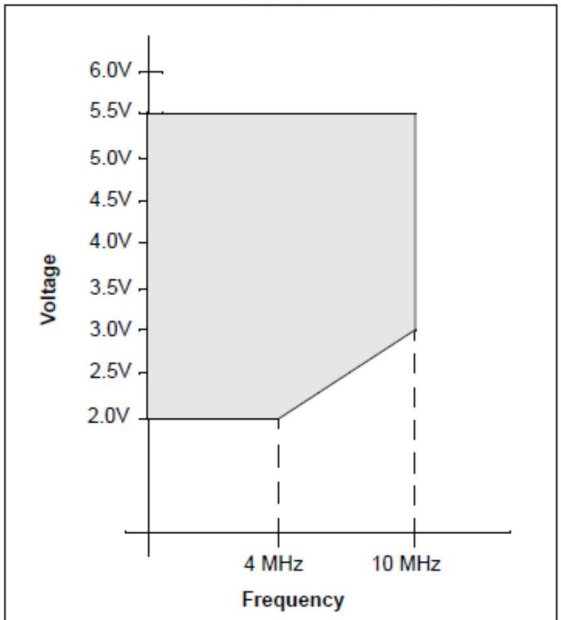


FIGURE 9-2: PIC16LF84A-04 VOLTAGE-FREQUENCY GRAPH

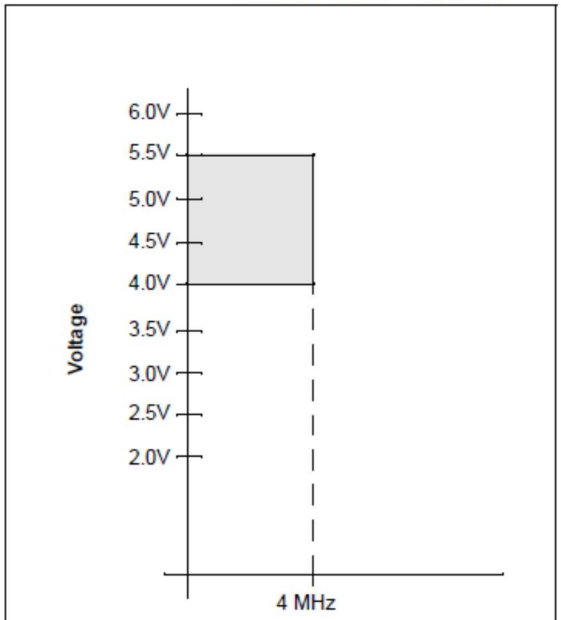


FMAX = (6.0 MHz/V) (VDDAPPMIN - 2.0V) + 4 MHz

Note 1: VDDAPPMIN is the minimum voltage of the PICmicro® device in the application.

2: FMAX has a maximum frequency of 10 MHz.

FIGURE 9-3: PIC16F84A-04 VOLTAGE-FREQUENCY GRAPH



9.1 DC Characteristics

PIC16LF84A-04 (Commercial, Industrial)			**Standard Operating Conditions (unless otherwise stated)** Operating temperature 0°C ≤ TA ≤ +70°C (commercial) -40°C ≤ TA ≤ +85°C (industrial) -40°C ≤ TA ≤ +125°C (extended)				
PIC16F84A-04 (Commercial, Industrial, Extended) **PIC16F84A-20** (Commercial, Industrial, Extended)			**Standard Operating Conditions (unless otherwise stated)** Operating temperature 0°C ≤ TA ≤ +70°C (commercial) -40°C ≤ TA ≤ +85°C (industrial) -40°C ≤ TA ≤ +125°C (extended)				
Param No.	**Symbol**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
	VDD	**Supply Voltage**					
D001		16LF84A	2.0	—	5.5	V	XT, RC, and LP osc configuration
D001		16F84A	4.0	—	5.5	V	XT, RC and LP osc configuration
D001A			4.5	—	5.5	V	HS osc configuration
D002	VDR	**RAM Data Retention Voltage (Note 1)**	1.5	—	—	V	Device in SLEEP mode
D003	VPOR	**VDD Start Voltage** to ensure internal Power-on Reset signal	—	Vss	—	V	See section on Power-on Reset for details
D004	SVDD	**VDD Rise Rate** to ensure internal Power-on Reset signal	0.05	—	—	V/ms	
	IDD	**Supply Current (Note 2)**					
D010		16LF84A	—	1	4	mA	RC and XT osc configuration **(Note 4)** FOSC = 2.0 MHz, VDD = 5.5V
D010		16F84A	—	1.8	4.5	mA	RC and XT osc configuration **(Note 4)** FOSC = 4.0 MHz, VDD = 5.5V
D010A			—	3	10	mA	RC and XT osc configuration **(Note 4)** FOSC = 4.0 MHz, VDD = 5.5V (During FLASH programming)
D013			—	10	20	mA	HS osc configuration (PIC16F84A-20) FOSC = 20 MHz, VDD = 5.5V
D014		16LF84A	—	15	45	µA	LP osc configuration FOSC = 32 kHz, VDD = 2.0V, WDT disabled

Legend: Rows with standard voltage device data only are shaded for improved readability.

† Data in "Typ" column is at 5.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

NR Not rated for operation.

Note 1: This is the limit to which VDD can be lowered without losing RAM data.

2: The supply current is mainly a function of the operating voltage and frequency. Other factors, such as I/O pin loading and switching rate, oscillator type, internal code execution pattern, and temperature also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD,
T0CKI = VDD, $\overline{MCLR}$ = VDD; WDT enabled/disabled as specified.

3: The power-down current in SLEEP mode does not depend on the oscillator type. Power-down current is measured with the part in SLEEP mode, with all I/O pins in hi-impedance state and tied to VDD and VSS.

4: For RC osc configuration, current through REXT is not included. The current through the resistor can be estimated by the formula IR = VDD/2REXT (mA) with REXT in kOhm.

5: The Δ current is the additional current consumed when this peripheral is enabled. This current should be added to the base IDD measurement.

9.3.3 TIMING DIAGRAMS AND SPECIFICATIONS

FIGURE 9-6: EXTERNAL CLOCK TIMING

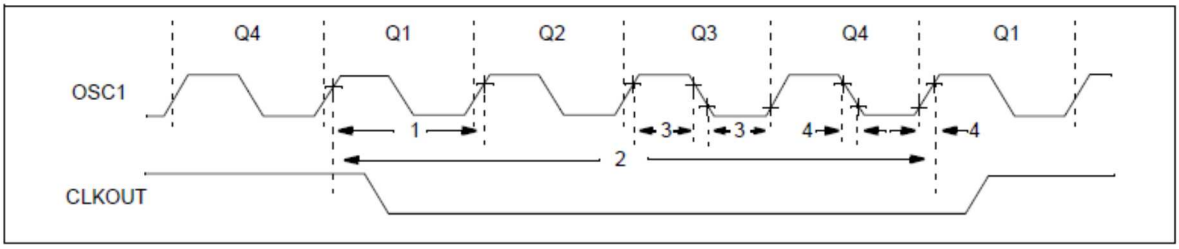


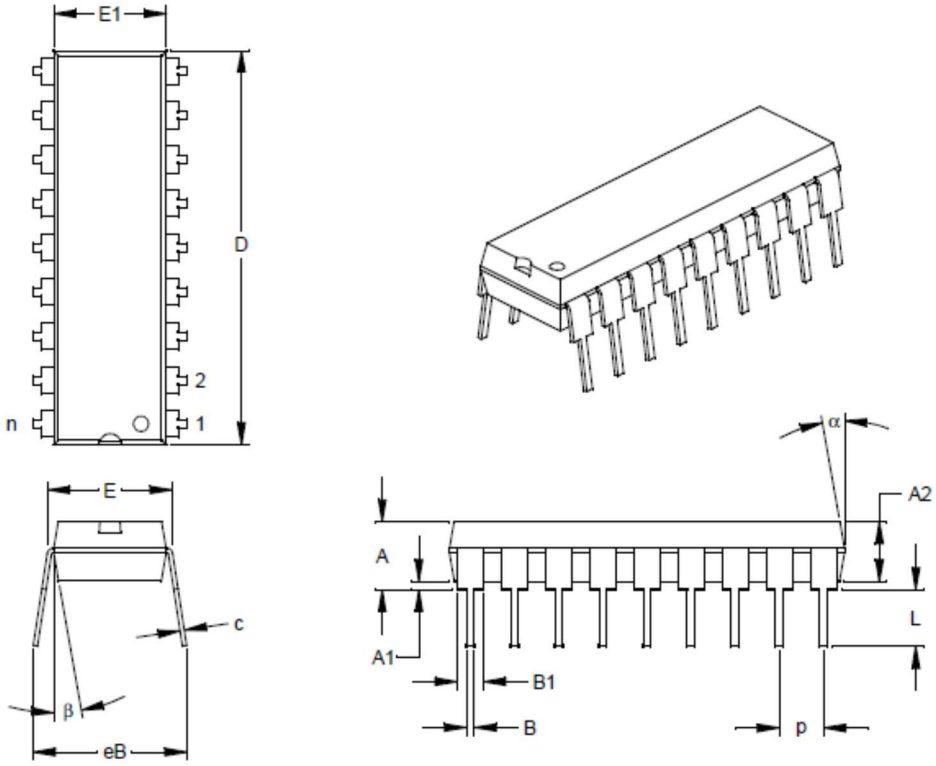
TABLE 9-2: EXTERNAL CLOCK TIMING REQUIREMENTS

Param No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions	
	FOSC	**External CLKIN Frequency(1)**	DC	—	2	MHz	XT, RC osc	(-04, LF)
			DC	—	4	MHz	XT, RC osc	(-04)
			DC	—	20	MHz	HS osc	(-20)
			DC	—	200	kHz	LP osc	(-04, LF)
		Oscillator Frequency(1)	DC	—	2	MHz	RC osc	(-04, LF)
			DC	—	4	MHz	RC osc	(-04)
			0.1	—	2	MHz	XT osc	(-04, LF)
			0.1	—	4	MHz	XT osc	(-04)
			1.0	—	20	MHz	HS osc	(-20)
			DC	—	200	kHz	LP osc	(-04, LF)
1	TOSC	**External CLKIN Period(1)**	500	—	—	ns	XT, RC osc	(-04, LF)
			250	—	—	ns	XT, RC osc	(-04)
			50	—	—	ns	HS osc	(-20)
			5.0	—	—	µs	LP osc	(-04, LF)
		Oscillator Period(1)	500	—	—	ns	RC osc	(-04, LF)
			250	—	—	ns	RC osc	(-04)
			500	—	10,000	ns	XT osc	(-04, LF)
			250	—	10,000	ns	XT osc	(-04)
			50	—	1,000	ns	HS osc	(-20)
			5.0	—	—	µs	LP osc	(-04, LF)
2	TCY	**Instruction Cycle Time(1)**	0.2	4/FOSC	DC	µs		
3	TosL, TosH	**Clock in (OSC1) High or Low Time**	60	—	—	ns	XT osc	(-04, LF)
			50	—	—	ns	XT osc	(-04)
			2.0	—	—	µs	LP osc	(-04, LF)
			17.5	—	—	ns	HS osc	(-20)
4	TosR, TosF	**Clock in (OSC1) Rise or Fall Time**	25	—	—	ns	XT osc	(-04)
			50	—	—	ns	LP osc	(-04, LF)
			7.5	—	—	ns	HS osc	(-20)

† Data in "Typ" column is at 5.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: Instruction cycle period (TCY) equals four times the input oscillator time-base period. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at "Min." values with an external clock applied to the OSC1 pin.
When an external clock input is used, the "Max." cycle time limit is "DC" (no clock) for all devices.

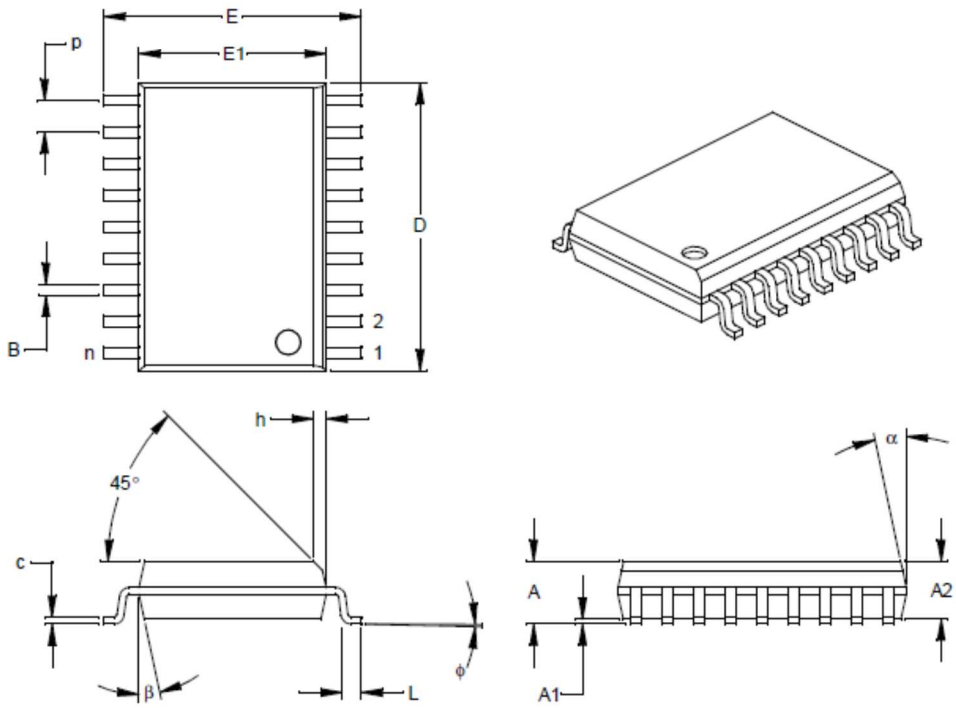
18-Lead Plastic Dual In-line (P) – 300 mil (PDIP)



Units		INCHES*			MILLIMETERS		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		18			18	
Pitch	p		.100			2.54	
Top to Seating Plane	A	.140	.155	.170	3.56	3.94	4.32
Molded Package Thickness	A2	.115	.130	.145	2.92	3.30	3.68
Base to Seating Plane	A1	.015			0.38		
Shoulder to Shoulder Width	E	.300	.313	.325	7.62	7.94	8.26
Molded Package Width	E1	.240	.250	.260	6.10	6.35	6.60
Overall Length	D	.890	.898	.905	22.61	22.80	22.99
Tip to Seating Plane	L	.125	.130	.135	3.18	3.30	3.43
Lead Thickness	c	.008	.012	.015	0.20	0.29	0.38
Upper Lead Width	B1	.045	.058	.070	1.14	1.46	1.78
Lower Lead Width	B	.014	.018	.022	0.36	0.46	0.56
Overall Row Spacing §	eB	.310	.370	.430	7.87	9.40	10.92
Mold Draft Angle Top	α	5	10	15	5	10	15
Mold Draft Angle Bottom	β	5	10	15	5	10	15

* Controlling Parameter
§ Significant Characteristic
Notes:
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.
JEDEC Equivalent: MS-001
Drawing No. C04-007

18-Lead Plastic Small Outline (SO) – Wide, 300 mil (SOIC)



Units		INCHES*			MILLIMETERS		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		18			18	
Pitch	p		.050			1.27	
Overall Height	A	.093	.099	.104	2.36	2.50	2.64
Molded Package Thickness	A2	.088	.091	.094	2.24	2.31	2.39
Standoff §	A1	.004	.008	.012	0.10	0.20	0.30
Overall Width	E	.394	.407	.420	10.01	10.34	10.67
Molded Package Width	E1	.291	.295	.299	7.39	7.49	7.59
Overall Length	D	.446	.454	.462	11.33	11.53	11.73
Chamfer Distance	h	.010	.020	.029	0.25	0.50	0.74
Foot Length	L	.016	.033	.050	0.41	0.84	1.27
Foot Angle	φ	0	4	8	0	4	8
Lead Thickness	c	.009	.011	.012	0.23	0.27	0.30
Lead Width	B	.014	.017	.020	0.36	0.42	0.51
Mold Draft Angle Top	α	0	12	15	0	12	15
Mold Draft Angle Bottom	β	0	12	15	0	12	15

* Controlling Parameter
§ Significant Characteristic

Notes:
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.
JEDEC Equivalent: MS-013
Drawing No. C04-051

APPENDIX B: CONVERSION CONSIDERATIONS

Considerations for converting from one PIC16X8X device to another are listed in Table 1.

TABLE 1: CONVERSION CONSIDERATIONS - PIC16C84, PIC16F83/F84, PIC16CR83/CR84, PIC16F84A

Difference	PIC16C84	PIC16F83/F84	PIC16CR83/ CR84	PIC16F84A
Program Memory Size	1K x 14	512 x 14 / 1K x 14	512 x 14 / 1K x 14	1K x 14
Data Memory Size	36 x 8	36 x 8 / 68 x 8	36 x 8 / 68 x 8	68 x 8
Voltage Range	2.0V - 6.0V (-40°C to +85°C)	2.0V - 6.0V (-40°C to +85°C)	2.0V - 6.0V (-40°C to +85°C)	2.0V - 5.5V (-40°C to +125°C)
Maximum Operating Frequency	10 MHz	10 MHz	10 MHz	20 MHz
Supply Current (I_{DD}). See parameter # D014 in the electrical specs for more detail.	I_{DD} (typ) = 60 µA I_{DD} (max) = 400 µA (LP osc, F_{OSC} = 32 kHz, V_{DD} = 2.0V, WDT disabled)	I_{DD} (typ) = 15 µA I_{DD} (max) = 45 µA (LP osc, F_{OSC} = 32 kHz, V_{DD} = 2.0V, WDT disabled)	I_{DD} (typ) = 15 µA I_{DD} (max) = 45 µA (LP osc, F_{OSC} = 32 kHz, V_{DD} = 2.0V, WDT disabled)	I_{DD} (typ) = 15 µA I_{DD} (max) = 45 µA (LP osc, F_{OSC} = 32 kHz, V_{DD} = 2.0V, WDT disabled)
Power-down Current (I_{PD}). See parameters # D020, D021, and D021A in the electrical specs for more detail.	I_{PD} (typ) = 26 µA I_{PD} (max) = 100 µA (V_{DD} = 2.0V, WDT disabled, industrial)	I_{PD} (typ) = 0.4 µA I_{PD} (max) = 9 µA (V_{DD} = 2.0V, WDT disabled, industrial)	I_{PD} (typ) = 0.4 µA I_{PD} (max) = 6 µA (V_{DD} = 2.0V, WDT disabled, industrial)	I_{PD} (typ) = 0.4 µA I_{PD} (max) = 1 µA (V_{DD} = 2.0V, WDT disabled, industrial)
Input Low Voltage (V_{IL}). See parameters # D032 and D034 in the electrical specs for more detail.	V_{IL} (max) = 0.2V_{DD} (OSC1, RC mode)	V_{IL} (max) = 0.1V_{DD} (OSC1, RC mode)	V_{IL} (max) = 0.1V_{DD} (OSC1, RC mode)	V_{IL} (max) = 0.1V_{DD} (OSC1, RC mode)
Input High Voltage (V_{IH}). See parameter # D040 in the electrical specs for more detail.	V_{IH} (min) = 0.36V_{DD} (I/O Ports with TTL, 4.5V ≤ V_{DD} ≤ 5.5V)	V_{IH} (min) = 2.4V (I/O Ports with TTL, 4.5V ≤ V_{DD} ≤ 5.5V)	V_{IH} (min) = 2.4V (I/O Ports with TTL, 4.5V ≤ V_{DD} ≤ 5.5V)	V_{IH} (min) = 2.4V (I/O Ports with TTL, 4.5V ≤ V_{DD} ≤ 5.5V)
Data EEPROM Memory Erase/Write cycle time (T_{DEW}). See parameter # D122 in the electrical specs for more detail.	T_{DEW} (typ) = 10 ms T_{DEW} (max) = 20 ms	T_{DEW} (typ) = 10 ms T_{DEW} (max) = 20 ms	T_{DEW} (typ) = 10 ms T_{DEW} (max) = 20 ms	T_{DEW} (typ) = 4 ms T_{DEW} (max) = 8 ms
Port Output Rise/Fall time (TioR, TioF). See parameters #20, 20A, 21, and 21A in the electrical specs for more detail.	TioR, TioF (max) = 25 ns (C84) TioR, TioF (max) = 60 ns (LC84)	TioR, TioF (max) = 35 ns (C84) TioR, TioF (max) = 70 ns (LC84)	TioR, TioF (max) = 35 ns (C84) TioR, TioF (max) = 70 ns (LC84)	TioR, TioF (max) = 35 ns (C84) TioR, TioF (max) = 70 ns (LC84)
$\overline{MCLR}$ on-chip filter. See parameter #30 in the electrical specs for more detail.	No	Yes	Yes	Yes
PORTA and crystal oscillator values less than 500 kHz	For crystal oscillator configurations operating below 500 kHz, the device may generate a spurious internal Q-clock when PORTA<0> switches state.	N/A	N/A	N/A
RB0/INT pin	TTL	TTL/ST* (*Schmitt Trigger)	TTL/ST* (*Schmitt Trigger)	TTL/ST* (*Schmitt Trigger)

PIC16F84A PRODUCT IDENTIFICATION SYSTEM

To order or obtain information (e.g., on pricing or delivery) refer to the factory or the listed sales office.

PART NO.	-XX	X	/XX	XXX
Device	Frequency Range	Temperature Range	Package	Pattern

Device	PIC16F84A[1], PIC16F84AT[2] PIC16LF84A[1], PIC16LF84AT[2]
Frequency Range	04 = 4 MHz 20 = 20 MHz
Temperature Range	- = 0°C to +70°C I = -40°C to +85°C
Package	P = PDIP SO = SOIC (Gull Wing, 300 mil body) SS = SSOP
Pattern	QTP, SQTP, ROM Code (factory specified) or Special Requirements . Blank for OTP and Windowed devices.

Examples:

a) PIC16F84A -04/P 301 = Commercial temp., PDIP package, 4 MHz, normal VDD limits, QTP pattern #301.

b) PIC16LF84A - 04I/SO = Industrial temp., SOIC package, 200 kHz, Extended VDD limits.

c) PIC16F84A - 20I/P = Industrial temp., PDIP package, 20 MHz, normal VDD limits.

Note 1: F = Standard VDD range
LF = Extended VDD range
2: T = in tape and reel - SOIC and SSOP packages only.

Sales and Support

Data Sheets
Products supported by a preliminary Data Sheet may have an errata sheet describing minor operational differences and recommended workarounds. To determine if an errata sheet exists for a particular device, please contact one of the following:

1. Your local Microchip sales office
2. The Microchip Corporate Literature Center U.S. FAX: (480) 792-7277
3. The Microchip Worldwide Site (www.microchip.com)

Please specify which device, revision of silicon and Data Sheet (include Literature #) you are using.

New Customer Notification System
Register on our web site (www.microchip.com/cn) to receive the most current information on our products.

3- PIC 16F87XA Data sheet

PIC16F87XA
Data Sheet

28/40/44-Pin Enhanced Flash Microcontrollers

DS39582B

MICROCHIP

PIC16F87XA

28/40/44-Pin Enhanced Flash Microcontrollers

Devices Included in this Data Sheet:

- PIC16F873A
- PIC16F874A
- PIC16F876A
- PIC16F877A

High-Performance RISC CPU:

- Only 35 single-word instructions to learn
- All single-cycle instructions except for program branches, which are two-cycle
- Operating speed: DC – 20 MHz clock input
  DC – 200 ns instruction cycle
- Up to 8K x 14 words of Flash Program Memory, Up to 368 x 8 bytes of Data Memory (RAM), Up to 256 x 8 bytes of EEPROM Data Memory
- Pinout compatible to other 28-pin or 40/44-pin PIC16CXXX and PIC16FXXX microcontrollers

Peripheral Features:

- Timer0: 8-bit timer/counter with 8-bit prescaler
- Timer1: 16-bit timer/counter with prescaler, can be incremented during Sleep via external crystal/clock
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Two Capture, Compare, PWM modules
  - Capture is 16-bit, max. resolution is 12.5 ns
  - Compare is 16-bit, max. resolution is 200 ns
  - PWM max. resolution is 10-bit
- Synchronous Serial Port (SSP) with SPI™ (Master mode) and I^2C™ (Master/Slave)
- Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) with 9-bit address detection
- Parallel Slave Port (PSP) – 8 bits wide with external $\overline{RD}$, $\overline{WR}$ and $\overline{CS}$ controls (40/44-pin only)
- Brown-out detection circuitry for Brown-out Reset (BOR)

Analog Features:

- 10-bit, up to 8-channel Analog-to-Digital Converter (A/D)
- Brown-out Reset (BOR)
- Analog Comparator module with:
  - Two analog comparators
  - Programmable on-chip voltage reference (V_{REF}) module
  - Programmable input multiplexing from device inputs and internal voltage reference
  - Comparator outputs are externally accessible

Special Microcontroller Features:

- 100,000 erase/write cycle Enhanced Flash program memory typical
- 1,000,000 erase/write cycle Data EEPROM memory typical
- Data EEPROM Retention > 40 years
- Self-reprogrammable under software control
- In-Circuit Serial Programming™ (ICSP™) via two pins
- Single-supply 5V In-Circuit Serial Programming
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Programmable code protection
- Power saving Sleep mode
- Selectable oscillator options
- In-Circuit Debug (ICD) via two pins

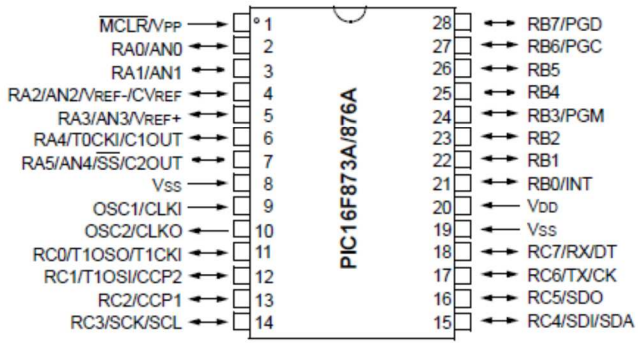
CMOS Technology:

- Low-power, high-speed Flash/EEPROM technology
- Fully static design
- Wide operating voltage range (2.0V to 5.5V)
- Commercial and Industrial temperature ranges
- Low-power consumption

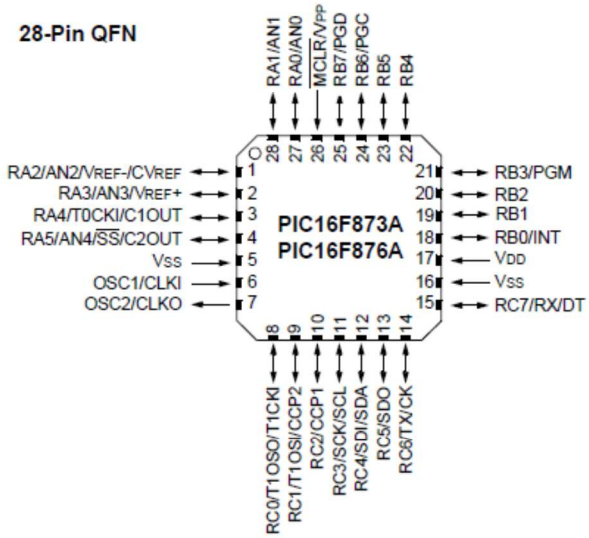
Device	Program Memory		Data SRAM (Bytes)	EEPROM (Bytes)	I/O	10-bit A/D (ch)	CCP (PWM)	MSSP		USART	Timers 8/16-bit	Comparators
	Bytes	# Single Word Instructions						SPI	Master I^2C			
PIC16F873A	7.2K	4096	192	128	22	5	2	Yes	Yes	Yes	2/1	2
PIC16F874A	7.2K	4096	192	128	33	8	2	Yes	Yes	Yes	2/1	2
PIC16F876A	14.3K	8192	368	256	22	5	2	Yes	Yes	Yes	2/1	2
PIC16F877A	14.3K	8192	368	256	33	8	2	Yes	Yes	Yes	2/1	2

Pin Diagrams

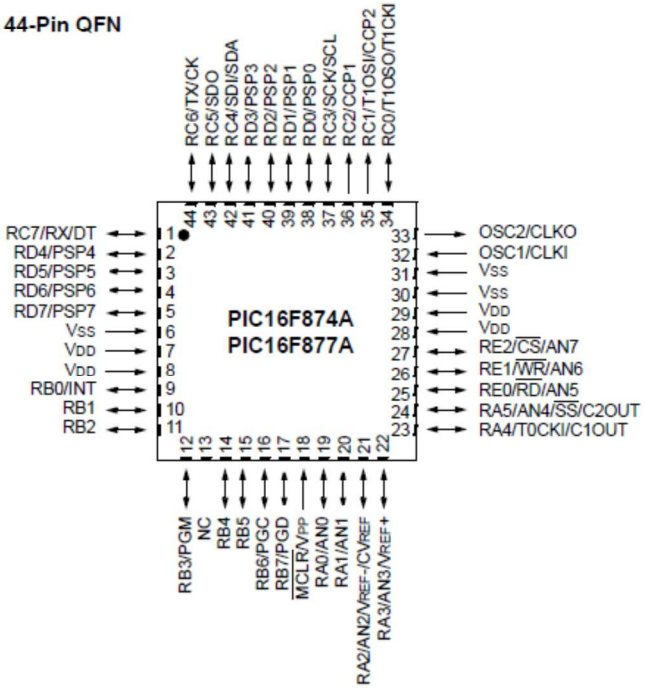
28-Pin PDIP, SOIC, SSOP



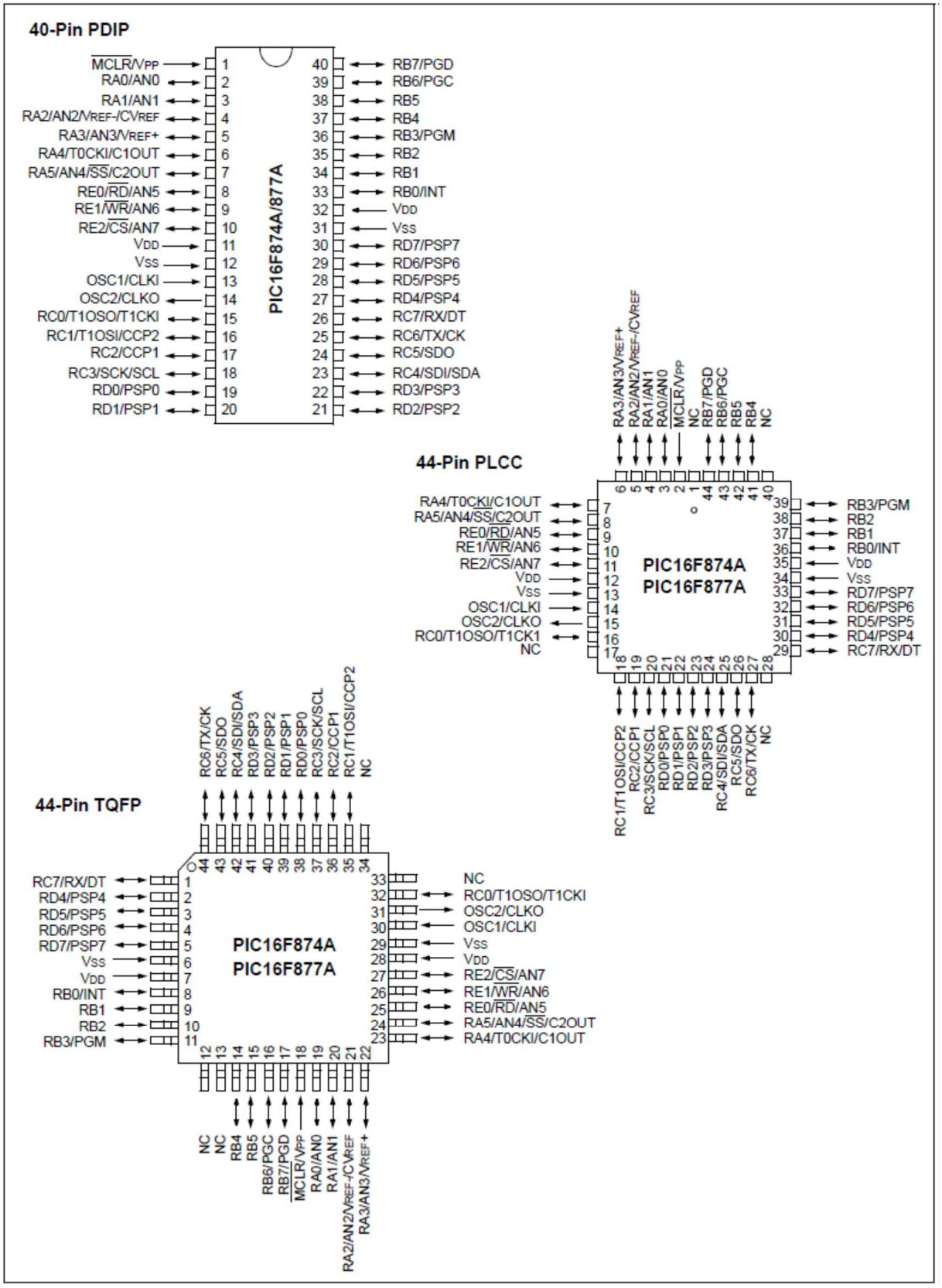
28-Pin QFN



44-Pin QFN



Pin Diagrams (Continued)



PIC16F87XA

Table of Contents

1.0 DEVICE OVERVIEW

This document contains device specific information about the following devices:

- PIC16F873A
- PIC16F874A
- PIC16F876A
- PIC16F877A

PIC16F873A/876A devices are available only in 28-pin packages, while PIC16F874A/877A devices are available in 40-pin and 44-pin packages. All devices in the PIC16F87XA family share common architecture with the following differences:

- The PIC16F873A and PIC16F874A have one-half of the total on-chip memory of the PIC16F876A and PIC16F877A
- The 28-pin devices have three I/O ports, while the 40/44-pin devices have five
- The 28-pin devices have fourteen interrupts, while the 40/44-pin devices have fifteen
- The 28-pin devices have five A/D input channels, while the 40/44-pin devices have eight
- The Parallel Slave Port is implemented only on the 40/44-pin devices

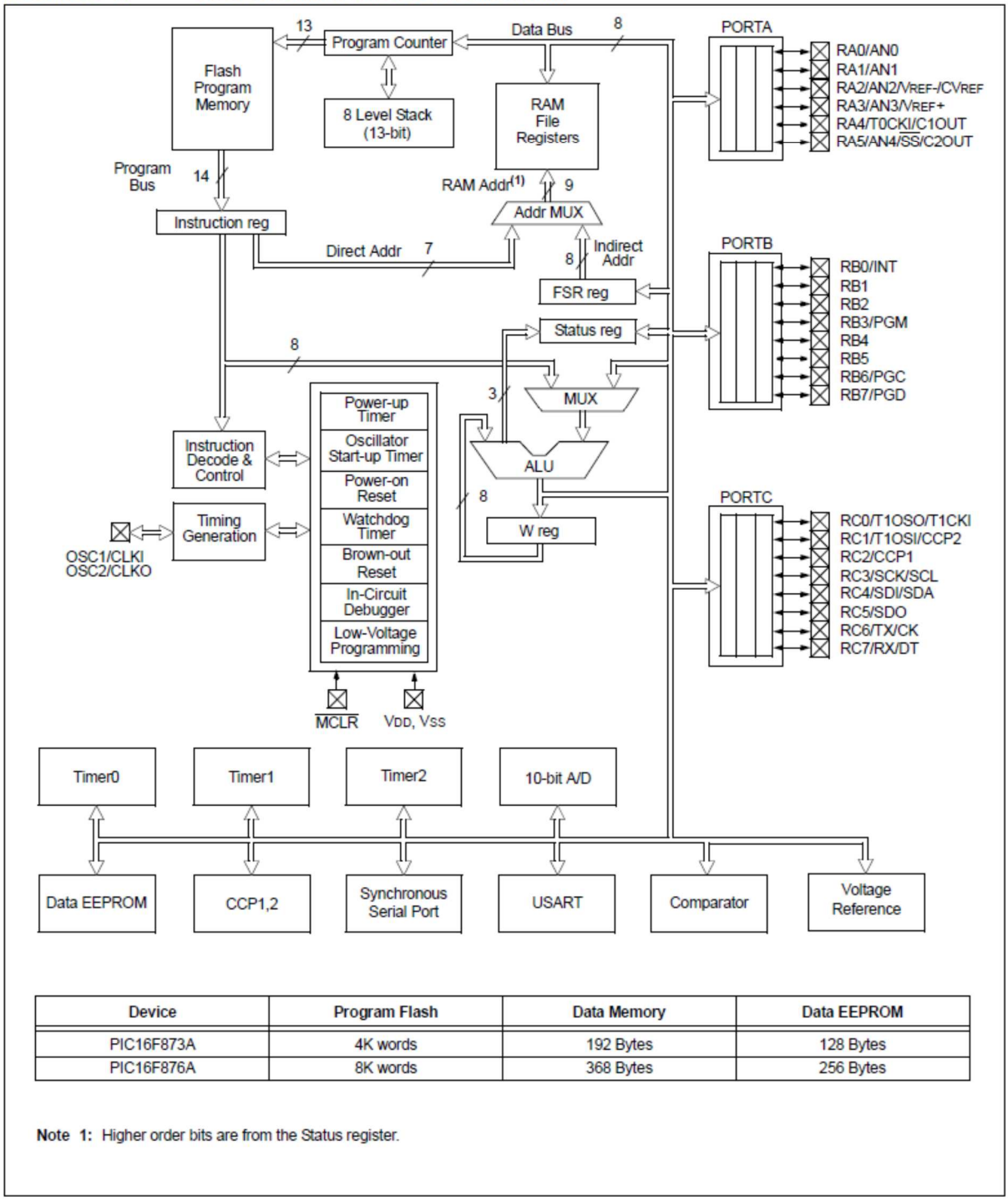
The available features are summarized in Table 1-1. Block diagrams of the PIC16F873A/876A and PIC16F874A/877A devices are provided in Figure 1-1 and Figure 1-2, respectively. The pinouts for these device families are listed in Table 1-2 and Table 1-3.

Additional information may be found in the PICmicro® Mid-Range Reference Manual (DS33023), which may be obtained from your local Microchip Sales Representative or downloaded from the Microchip web site. The Reference Manual should be considered a complementary document to this data sheet and is highly recommended reading for a better understanding of the device architecture and operation of the peripheral modules.

TABLE 1-1: PIC16F87XA DEVICE FEATURES

Key Features	PIC16F873A	PIC16F874A	PIC16F876A	PIC16F877A
Operating Frequency	DC – 20 MHz	DC – 20 MHz	DC – 20 MHz	DC – 20 MHz
Resets (and Delays)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)
Flash Program Memory (14-bit words)	4K	4K	8K	8K
Data Memory (bytes)	192	192	368	368
EEPROM Data Memory (bytes)	128	128	256	256
Interrupts	14	15	14	15
I/O Ports	Ports A, B, C	Ports A, B, C, D, E	Ports A, B, C	Ports A, B, C, D, E
Timers	3	3	3	3
Capture/Compare/PWM modules	2	2	2	2
Serial Communications	MSSP, USART	MSSP, USART	MSSP, USART	MSSP, USART
Parallel Communications	—	PSP	—	PSP
10-bit Analog-to-Digital Module	5 input channels	8 input channels	5 input channels	8 input channels
Analog Comparators	2	2	2	2
Instruction Set	35 Instructions	35 Instructions	35 Instructions	35 Instructions
Packages	28-pin PDIP 28-pin SOIC 28-pin SSOP 28-pin QFN	40-pin PDIP 44-pin PLCC 44-pin TQFP 44-pin QFN	28-pin PDIP 28-pin SOIC 28-pin SSOP 28-pin QFN	40-pin PDIP 44-pin PLCC 44-pin TQFP 44-pin QFN

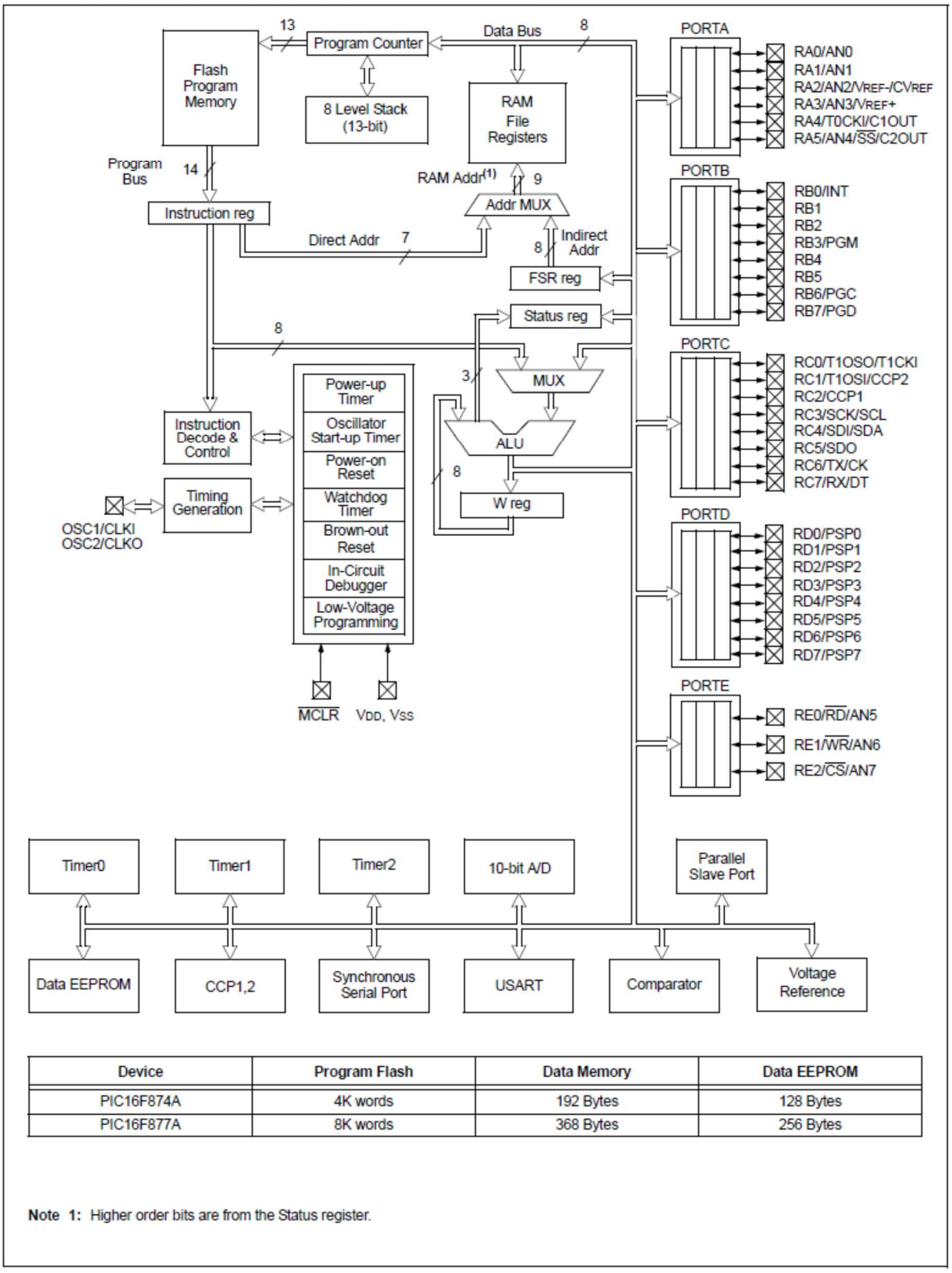
FIGURE 1-1: PIC16F873A/876A BLOCK DIAGRAM



Device	Program Flash	Data Memory	Data EEPROM
PIC16F873A	4K words	192 Bytes	128 Bytes
PIC16F876A	8K words	368 Bytes	256 Bytes

Note 1: Higher order bits are from the Status register.

FIGURE 1-2: PIC16F874A/877A BLOCK DIAGRAM



Device	Program Flash	Data Memory	Data EEPROM
PIC16F874A	4K words	192 Bytes	128 Bytes
PIC16F877A	8K words	368 Bytes	256 Bytes

Note 1: Higher order bits are from the Status register.

TABLE 1-3: PIC16F874A/877A PINOUT DESCRIPTION

Pin Name	PDIP Pin#	PLCC Pin#	TQFP Pin#	QFN Pin#	I/O/P Type	Buffer Type	Description
OSC1/CLKI	13	14	30	32		ST/CMOS(4)	Oscillator crystal or external clock input.
OSC1					I		Oscillator crystal input or external clock source input. ST buffer when configured in RC mode; otherwise CMOS.
CLKI					I		External clock source input. Always associated with pin function OSC1 (see OSC1/CLKI, OSC2/CLKO pins).
OSC2/CLKO	14	15	31	33		—	Oscillator crystal or clock output.
OSC2					O		Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
CLKO					O		In RC mode, OSC2 pin outputs CLKO, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
$\overline{MCLR}$/VPP	1	2	18	18		ST	Master Clear (input) or programming voltage (output).
$\overline{MCLR}$					I		Master Clear (Reset) input. This pin is an active low Reset to the device.
VPP					P		Programming voltage input.
							PORTA is a bidirectional I/O port.
RA0/AN0	2	3	19	19		TTL	
RA0					I/O		Digital I/O.
AN0					I		Analog input 0.
RA1/AN1	3	4	20	20		TTL	
RA1					I/O		Digital I/O.
AN1					I		Analog input 1.
RA2/AN2/VREF-/CVREF	4	5	21	21		TTL	
RA2					I/O		Digital I/O.
AN2					I		Analog input 2.
VREF-					I		A/D reference voltage (Low) input.
CVREF					O		Comparator VREF output.
RA3/AN3/VREF+	5	6	22	22		TTL	
RA3					I/O		Digital I/O.
AN3					I		Analog input 3.
VREF+					I		A/D reference voltage (High) input.
RA4/T0CKI/C1OUT	6	7	23	23		ST	
RA4					I/O		Digital I/O – Open-drain when configured as output.
T0CKI					I		Timer0 external clock input.
C1OUT					O		Comparator 1 output.
RA5/AN4/$\overline{SS}$/C2OUT	7	8	24	24		TTL	
RA5					I/O		Digital I/O.
AN4					I		Analog input 4.
$\overline{SS}$					I		SPI slave select input.
C2OUT					O		Comparator 2 output.

Legend: I = input O = output I/O = input/output P = power
— = Not used TTL = TTL input ST = Schmitt Trigger input

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.
2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.
3: This buffer is a Schmitt Trigger input when configured in RC Oscillator mode and a CMOS input otherwise.

2.0 MEMORY ORGANIZATION

There are three memory blocks in each of the PIC16F87XA devices. The program memory and data memory have separate buses so that concurrent access can occur and is detailed in this section. The EEPROM data memory block is detailed in **Section 3.0 "Data EEPROM and Flash Program Memory"**.

Additional information on device memory may be found in the PICmicro® Mid-Range MCU Family Reference Manual (DS33023).

2.1 Program Memory Organization

The PIC16F87XA devices have a 13-bit program counter capable of addressing an 8K word x 14 bit program memory space. The PIC16F876A/877A devices have 8K words x 14 bits of Flash program memory, while PIC16F873A/874A devices have 4K words x 14 bits. Accessing a location above the physically implemented address will cause a wraparound.

The Reset vector is at 0000h and the interrupt vector is at 0004h.

FIGURE 2-1: PIC16F876A/877A PROGRAM MEMORY MAP AND STACK

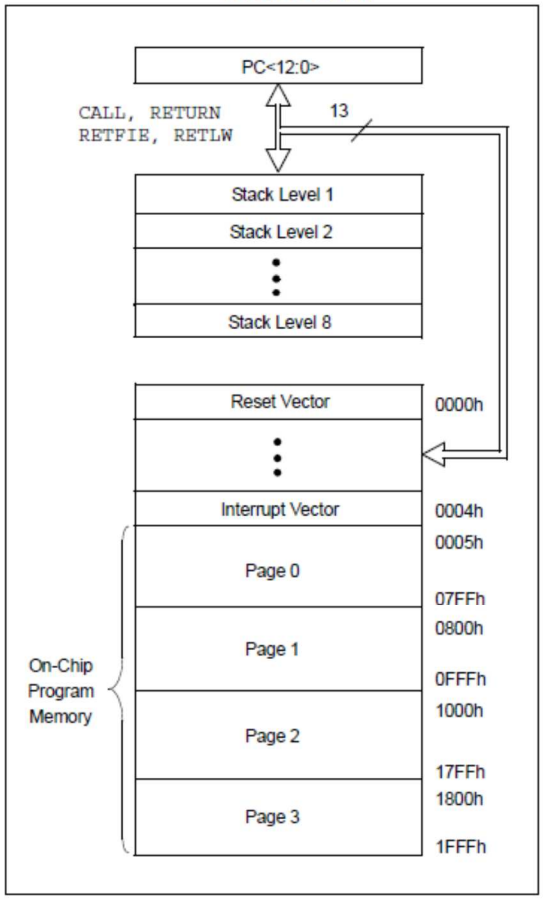


FIGURE 2-2: PIC16F873A/874A PROGRAM MEMORY MAP AND STACK

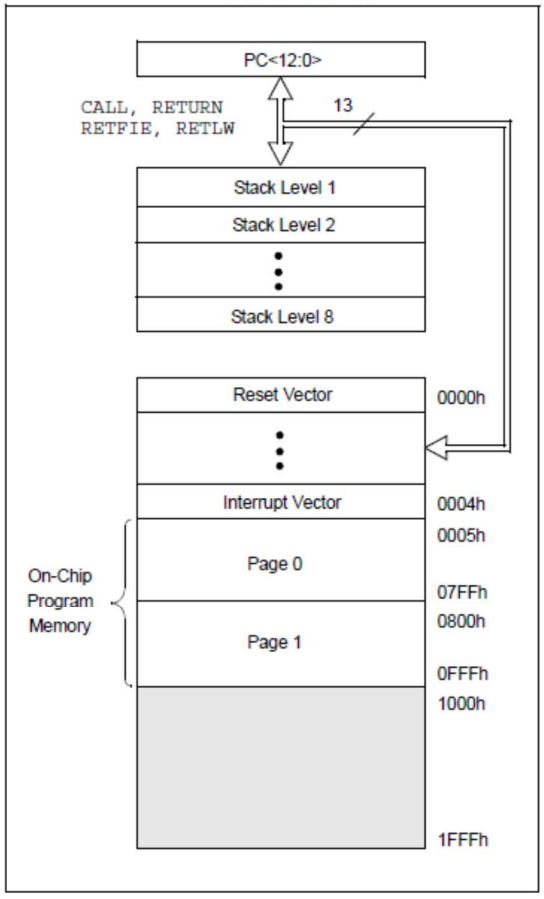


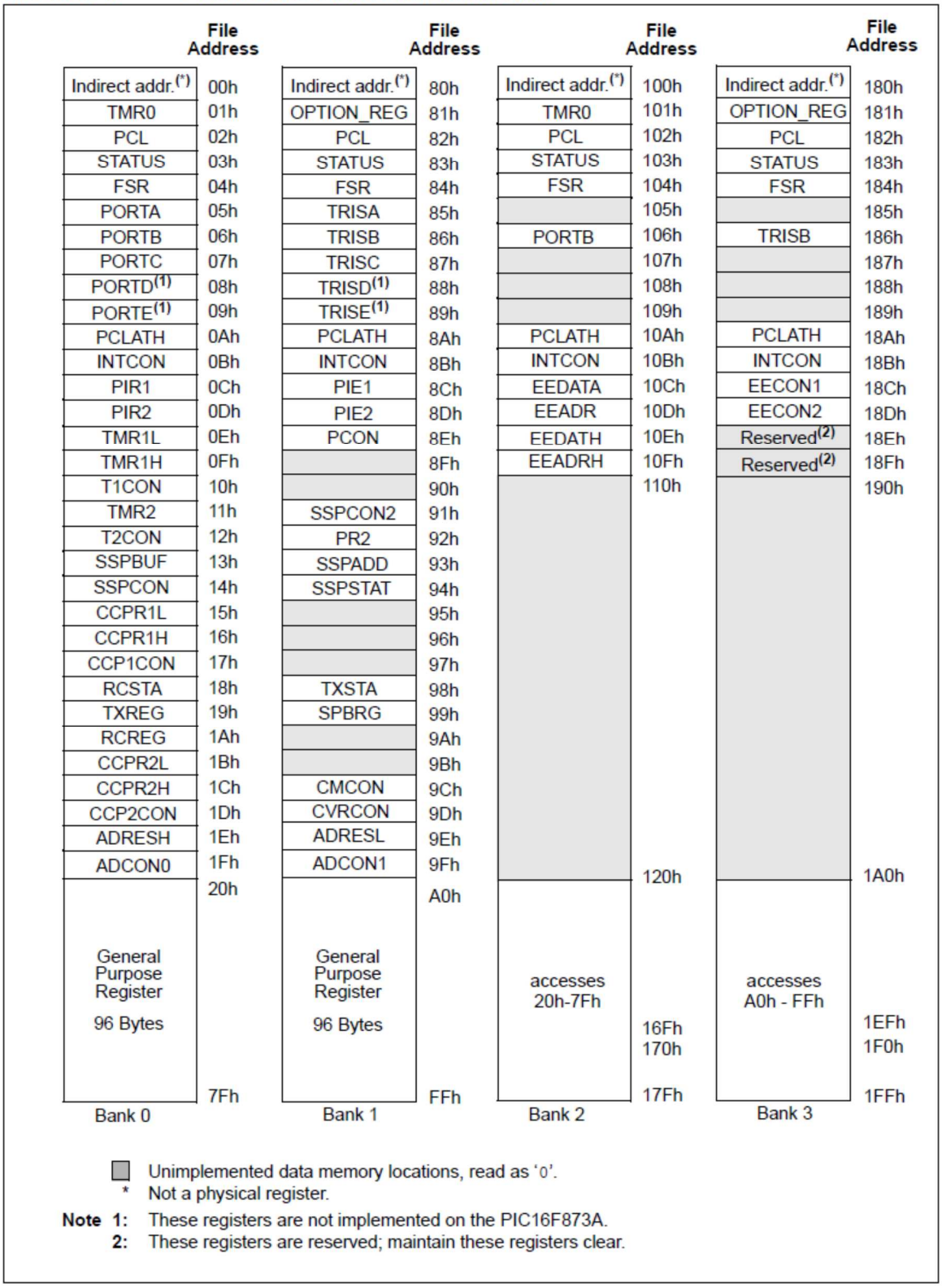
FIGURE 2-3: PIC16F876A/877A REGISTER FILE MAP

Bank 0	File Address	Bank 1	File Address	Bank 2	File Address	Bank 3	File Address
Indirect addr.(*)	00h	Indirect addr.(*)	80h	Indirect addr.(*)	100h	Indirect addr.(*)	180h
TMR0	01h	OPTION_REG	81h	TMR0	101h	OPTION_REG	181h
PCL	02h	PCL	82h	PCL	102h	PCL	182h
STATUS	03h	STATUS	83h	STATUS	103h	STATUS	183h
FSR	04h	FSR	84h	FSR	104h	FSR	184h
PORTA	05h	TRISA	85h		105h		185h
PORTB	06h	TRISB	86h	PORTB	106h	TRISB	186h
PORTC	07h	TRISC	87h		107h		187h
PORTD(1)	08h	TRISD(1)	88h		108h		188h
PORTE(1)	09h	TRISE(1)	89h		109h		189h
PCLATH	0Ah	PCLATH	8Ah	PCLATH	10Ah	PCLATH	18Ah
INTCON	0Bh	INTCON	8Bh	INTCON	10Bh	INTCON	18Bh
PIR1	0Ch	PIE1	8Ch	EEDATA	10Ch	EECON1	18Ch
PIR2	0Dh	PIE2	8Dh	EEADR	10Dh	EECON2	18Dh
TMR1L	0Eh	PCON	8Eh	EEDATH	10Eh	Reserved(2)	18Eh
TMR1H	0Fh		8Fh	EEADRH	10Fh	Reserved(2)	18Fh
T1CON	10h		90h		110h		190h
TMR2	11h	SSPCON2	91h		111h		191h
T2CON	12h	PR2	92h		112h		192h
SSPBUF	13h	SSPADD	93h		113h		193h
SSPCON	14h	SSPSTAT	94h		114h		194h
CCPR1L	15h		95h		115h		195h
CCPR1H	16h		96h		116h		196h
CCP1CON	17h		97h	General Purpose Register 16 Bytes	117h	General Purpose Register 16 Bytes	197h
RCSTA	18h	TXSTA	98h		118h		198h
TXREG	19h	SPBRG	99h		119h		199h
RCREG	1Ah		9Ah		11Ah		19Ah
CCPR2L	1Bh		9Bh		11Bh		19Bh
CCPR2H	1Ch	CMCON	9Ch		11Ch		19Ch
CCP2CON	1Dh	CVRCON	9Dh		11Dh		19Dh
ADRESH	1Eh	ADRESL	9Eh		11Eh		19Eh
ADCON0	1Fh	ADCON1	9Fh		11Fh		19Fh
General Purpose Register 96 Bytes	20h	General Purpose Register 80 Bytes	A0h	General Purpose Register 80 Bytes	120h	General Purpose Register 80 Bytes	1A0h
	7Fh	accesses 70h-7Fh	EFh F0h FFh	accesses 70h-7Fh	16Fh 170h 17Fh	accesses 70h - 7Fh	1EFh 1F0h 1FFh

Unimplemented data memory locations, read as '0'.
* Not a physical register.

Note 1: These registers are not implemented on the PIC16F876A.
2: These registers are reserved; maintain these registers clear.

FIGURE 2-4: PIC16F873A/874A REGISTER FILE MAP



Unimplemented data memory locations, read as '0'.
* Not a physical register.

Note 1: These registers are not implemented on the PIC16F873A.
2: These registers are reserved; maintain these registers clear.

2.2.2 SPECIAL FUNCTION REGISTERS

The Special Function Registers are registers used by the CPU and peripheral modules for controlling the desired operation of the device. These registers are implemented as static RAM. A list of these registers is given in Table 2-1.

The Special Function Registers can be classified into two sets: core (CPU) and peripheral. Those registers associated with the core functions are described in detail in this section. Those related to the operation of the peripheral features are described in detail in the peripheral features section.

TABLE 2-1: SPECIAL FUNCTION REGISTER SUMMARY

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Details on page:
Bank 0											
00h(3)	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	31, 150
01h	TMR0	Timer0 Module Register								xxxx xxxx	55, 150
02h(3)	PCL	Program Counter (PC) Least Significant Byte								0000 0000	30, 150
03h(3)	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	22, 150
04h(3)	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	31, 150
05h	PORTA	—	—	PORTA Data Latch when written: PORTA pins when read						--0x 0000	43, 150
06h	PORTB	PORTB Data Latch when written: PORTB pins when read								xxxx xxxx	45, 150
07h	PORTC	PORTC Data Latch when written: PORTC pins when read								xxxx xxxx	47, 150
08h(4)	PORTD	PORTD Data Latch when written: PORTD pins when read								xxxx xxxx	48, 150
09h(4)	PORTE	—	—	—	—	—	RE2	RE1	RE0	---- -xxx	49, 150
0Ah(1,3)	PCLATH	—	—	—	Write Buffer for the upper 5 bits of the Program Counter					---0 0000	30, 150
0Bh(3)	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	24, 150
0Ch	PIR1	PSPIF(3)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	26, 150
0Dh	PIR2	—	CMIF	—	EEIF	BCLIF	—	—	CCP2IF	-0-0 0--0	28, 150
0Eh	TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	60, 150
0Fh	TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	60, 150
10h	T1CON	—	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	--00 0000	57, 150
11h	TMR2	Timer2 Module Register								0000 0000	62, 150
12h	T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	61, 150
13h	SSPBUF	Synchronous Serial Port Receive Buffer/Transmit Register								xxxx xxxx	79, 150
14h	SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	82, 82, 150
15h	CCPR1L	Capture/Compare/PWM Register 1 (LSB)								xxxx xxxx	63, 150
16h	CCPR1H	Capture/Compare/PWM Register 1 (MSB)								xxxx xxxx	63, 150
17h	CCP1CON	—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	64, 150
18h	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	112, 150
19h	TXREG	USART Transmit Data Register								0000 0000	118, 150
1Ah	RCREG	USART Receive Data Register								0000 0000	118, 150
1Bh	CCPR2L	Capture/Compare/PWM Register 2 (LSB)								xxxx xxxx	63, 150
1Ch	CCPR2H	Capture/Compare/PWM Register 2 (MSB)								xxxx xxxx	63, 150
1Dh	CCP2CON	—	—	CCP2X	CCP2Y	CCP2M3	CCP2M2	CCP2M1	CCP2M0	--00 0000	64, 150
1Eh	ADRESH	A/D Result Register High Byte								xxxx xxxx	133, 150
1Fh	ADCON0	ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/$\overline{DONE}$	—	ADON	0000 00-0	127, 150

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations are unimplemented, read as '0'.

Note 1: The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<12:8>, whose contents are transferred to the upper byte of the program counter.

2: Bits PSPIE and PSPIF are reserved on PIC16F873A/876A devices; always maintain these bits clear.

3: These registers can be addressed from any bank.

4: PORTD, PORTE, TRISD and TRISE are not implemented on PIC16F873A/876A devices, read as '0'.

5: Bit 4 of EEADRH implemented only on the PIC16F876A/877A devices.

TABLE 2-1: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Details on page:
Bank 1											
80h(3)	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	31, 150
81h	OPTION_REG	$\overline{RBPU}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	23, 150
82h(3)	PCL	Program Counter (PC) Least Significant Byte								0000 0000	30, 150
83h(3)	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	22, 150
84h(3)	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	31, 150
85h	TRISA	—	—	PORTA Data Direction Register						--11 1111	43, 150
86h	TRISB	PORTB Data Direction Register								1111 1111	45, 150
87h	TRISC	PORTC Data Direction Register								1111 1111	47, 150
88h(4)	TRISD	PORTD Data Direction Register								1111 1111	48, 151
89h(4)	TRISE	IBF	OBF	IBOV	PSPMODE	—	PORTE Data Direction bits			0000 -111	50, 151
8Ah(1,3)	PCLATH	—	—	—	Write Buffer for the upper 5 bits of the Program Counter					---0 0000	30, 150
8Bh(3)	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	24, 150
8Ch	PIE1	PSPIE(2)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	25, 151
8Dh	PIE2	—	CMIE	—	EEIE	BCLIE	—	—	CCP2IE	-0-0 0--0	27, 151
8Eh	PCON	—	—	—	—	—	—	$\overline{POR}$	$\overline{BOR}$	---- --qq	29, 151
8Fh	—	Unimplemented								—	—
90h	—	Unimplemented								—	—
91h	SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000	83, 151
92h	PR2	Timer2 Period Register								1111 1111	62, 151
93h	SSPADD	Synchronous Serial Port (I^2C mode) Address Register								0000 0000	79, 151
94h	SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	79, 151
95h	—	Unimplemented								—	—
96h	—	Unimplemented								—	—
97h	—	Unimplemented								—	—
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	111, 151
99h	SPBRG	Baud Rate Generator Register								0000 0000	113, 151
9Ah	—	Unimplemented								—	—
9Bh	—	Unimplemented								—	—
9Ch	CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0111	135, 151
9Dh	CVRCON	CVREN	CVROE	CVRR	—	CVR3	CVR2	CVR1	CVR0	000- 0000	141, 151
9Eh	ADRESL	A/D Result Register Low Byte								xxxx xxxx	133, 151
9Fh	ADCON1	ADFM	ADCS2	—	—	PCFG3	PCFG2	PCFG1	PCFG0	00-- 0000	128, 151

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations are unimplemented, read as '0'.

Note 1: The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<12:8>, whose contents are transferred to the upper byte of the program counter.

2: Bits PSPIE and PSPIF are reserved on PIC16F873A/876A devices; always maintain these bits clear.

3: These registers can be addressed from any bank.

4: PORTD, PORTE, TRISD and TRISE are not implemented on PIC16F873A/876A devices, read as '0'.

5: Bit 4 of EEADRH implemented only on the PIC16F876A/877A devices.

TABLE 2-1: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Details on page:
Bank 2											
100h(3)	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	31, 150
101h	TMR0	Timer0 Module Register								xxxx xxxx	55, 150
102h(3)	PCL	Program Counter's (PC) Least Significant Byte								0000 0000	30, 150
103h(3)	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	22, 150
104h(3)	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	31, 150
105h	—	Unimplemented								—	—
106h	PORTB	PORTB Data Latch when written: PORTB pins when read								xxxx xxxx	45, 150
107h	—	Unimplemented								—	—
108h	—	Unimplemented								—	—
109h	—	Unimplemented								—	—
10Ah(1,3)	PCLATH	—	—	—	Write Buffer for the upper 5 bits of the Program Counter					---0 0000	30, 150
10Bh(3)	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	24, 150
10Ch	EEDATA	EEPROM Data Register Low Byte								xxxx xxxx	39, 151
10Dh	EEADR	EEPROM Address Register Low Byte								xxxx xxxx	39, 151
10Eh	EEDATH	—	—	EEPROM Data Register High Byte						--xx xxxx	39, 151
10Fh	EEADRH	—	—	—	—(5)	EEPROM Address Register High Byte				---- xxxx	39, 151
Bank 3											
180h(3)	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	31, 150
181h	OPTION_REG	$\overline{RBPU}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	23, 150
182h(3)	PCL	Program Counter (PC) Least Significant Byte								0000 0000	30, 150
183h(3)	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	22, 150
184h(3)	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	31, 150
185h	—	Unimplemented								—	—
186h	TRISB	PORTB Data Direction Register								1111 1111	45, 150
187h	—	Unimplemented								—	—
188h	—	Unimplemented								—	—
189h	—	Unimplemented								—	—
18Ah(1,3)	PCLATH	—	—	—	Write Buffer for the upper 5 bits of the Program Counter					---0 0000	30, 150
18Bh(3)	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	24, 150
18Ch	EECON1	EEPGD	—	—	—	WRERR	WREN	WR	RD	x--- x000	34, 151
18Dh	EECON2	EEPROM Control Register 2 (not a physical register)								---- ----	39, 151
18Eh	—	Reserved; maintain clear								0000 0000	—
18Fh	—	Reserved; maintain clear								0000 0000	—

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations are unimplemented, read as '0'.

Note 1: The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<12:8>, whose contents are transferred to the upper byte of the program counter.

2: Bits PSPIE and PSPIF are reserved on PIC16F873A/876A devices; always maintain these bits clear.

3: These registers can be addressed from any bank.

4: PORTD, PORTE, TRISD and TRISE are not implemented on PIC16F873A/876A devices, read as '0'.

5: Bit 4 of EEADRH implemented only on the PIC16F876A/877A devices.

17.0 ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings †

Ambient temperature under bias ..... -55 to +125°C
Storage temperature ..... -65°C to +150°C
Voltage on any pin with respect to V_{SS} (except V_{DD}, $\overline{MCLR}$, and RA4) ..... -0.3V to (V_{DD} + 0.3V)
Voltage on V_{DD} with respect to V_{SS} ..... -0.3 to +7.5V
Voltage on $\overline{MCLR}$ with respect to V_{SS} **(Note 2)** ..... 0 to +14V
Voltage on RA4 with respect to Vss ..... 0 to +8.5V
Total power dissipation **(Note 1)** 1.0W
Maximum current out of V_{SS} pin ..... 300 mA
Maximum current into V_{DD} pin ..... 250 mA
Input clamp current, I_{IK} ($V_I < 0$ or $V_I > V_{DD}$) ..... ± 20 mA
Output clamp current, I_{OK} ($V_O < 0$ or $V_O > V_{DD}$) ..... ± 20 mA
Maximum output current sunk by any I/O pin ..... 25 mA
Maximum output current sourced by any I/O pin ..... 25 mA
Maximum current sunk by PORTA, PORTB and PORTE (combined) **(Note 3)** ..... 200 mA
Maximum current sourced by PORTA, PORTB and PORTE (combined) **(Note 3)** ..... 200 mA
Maximum current sunk by PORTC and PORTD (combined) **(Note 3)** ..... 200 mA
Maximum current sourced by PORTC and PORTD (combined) **(Note 3)** ..... 200 mA

Note 1: Power dissipation is calculated as follows: $P_{dis} = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum (V_{OI} \times I_{OL})$

2: Voltage spikes below V_{SS} at the $\overline{MCLR}$ pin, inducing currents greater than 80 mA, may cause latch-up. Thus, a series resistor of 50-100Ω should be used when applying a "low" level to the $\overline{MCLR}$ pin rather than pulling this pin directly to V_{SS}.

3: PORTD and PORTE are not implemented on PIC16F873A/876A devices.

† NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

FIGURE 17-1: PIC16F87XA VOLTAGE-FREQUENCY GRAPH (INDUSTRIAL, EXTENDED)

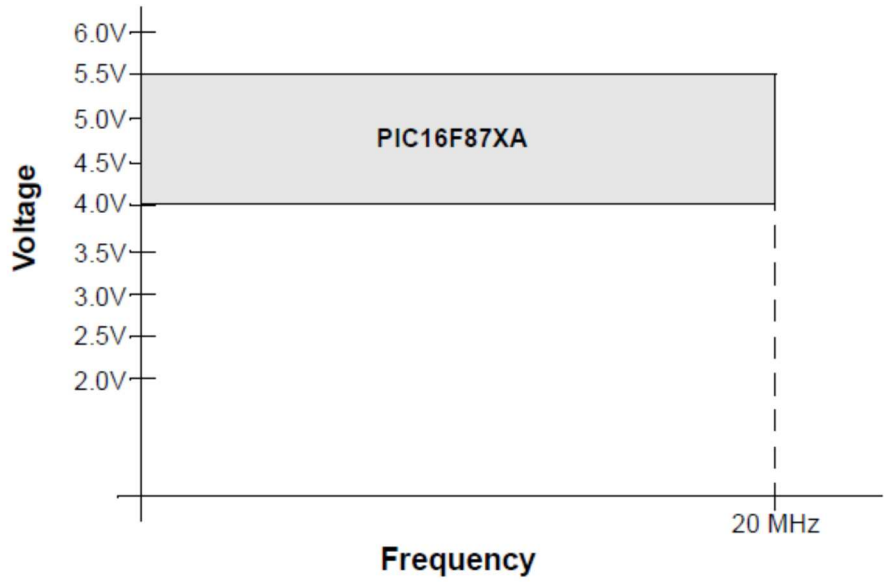
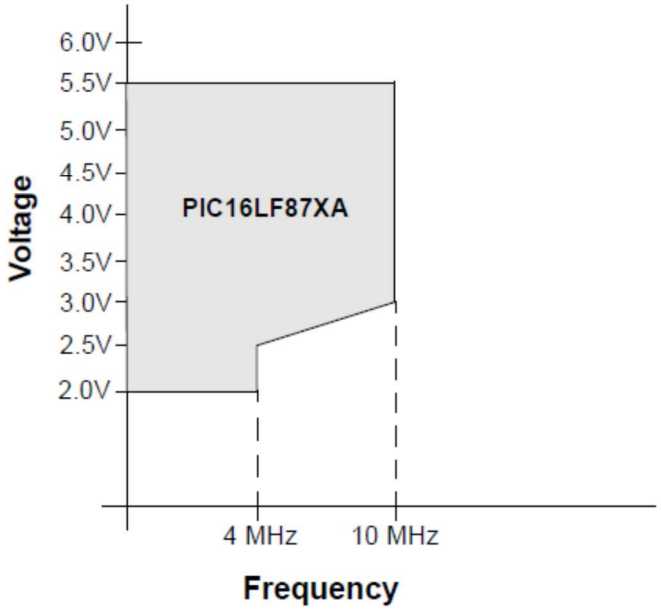


FIGURE 17-2: PIC16LF87XA VOLTAGE-FREQUENCY GRAPH (INDUSTRIAL)



FMAX = (6.0 MHz/V) (VDDAPPMIN – 2.0V) + 4 MHz

Note 1: VDDAPPMIN is the minimum voltage of the PICmicro® device in the application.

2: FMAX has a maximum frequency of 10 MHz.

17.1 DC Characteristics: PIC16F873A/874A/876A/877A (Industrial, Extended) PIC16LF873A/874A/876A/877A (Industrial)

PIC16LF873A/874A/876A/877A (Industrial)			**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial				
PIC16F873A/874A/876A/877A (Industrial, Extended)			**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended				
Param No.	**Symbol**	**Characteristic/ Device**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
	VDD	**Supply Voltage**					
D001		16LF87XA	2.0	—	5.5	V	All configurations (DC to 10 MHz)
D001		16F87XA	4.0	—	5.5	V	All configurations
D001A			VBOR		5.5	V	BOR enabled, FMAX = 14 MHz(7)
D002	VDR	**RAM Data Retention Voltage(1)**	—	1.5	—	V	
D003	VPOR	**VDD Start Voltage** to ensure internal Power-on Reset signal	—	VSS	—	V	See **Section 14.5 "Power-on Reset (POR)"** for details
D004	SVDD	**VDD Rise Rate** to ensure internal Power-on Reset signal	0.05	—	—	V/ms	See **Section 14.5 "Power-on Reset (POR)"** for details
D005	VBOR	**Brown-out Reset Voltage**	3.65	4.0	4.35	V	BODEN bit in configuration word enabled

Legend: Rows with standard voltage device data only are shaded for improved readability.

† Data in "Typ" column is at 5V, 25°C, unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: This is the limit to which VDD can be lowered without losing RAM data.

2: The supply current is mainly a function of the operating voltage and frequency. Other factors, such as I/O pin loading, switching rate, oscillator type, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
MCLR = VDD; WDT enabled/disabled as specified.

3: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD and VSS.

4: For RC osc configuration, current through REXT is not included. The current through the resistor can be estimated by the formula Ir = VDD/2REXT (mA) with REXT in kΩ.

5: Timer1 oscillator (when enabled) adds approximately 20 µA to the specification. This value is from characterization and is for design guidance only. This is not tested.

6: The Δ current is the additional current consumed when this peripheral is enabled. This current should be added to the base IDD or IPD measurement.

7: When BOR is enabled, the device will operate correctly until the VBOR voltage trip point is reached.

FIGURE 17-4: EXTERNAL CLOCK TIMING

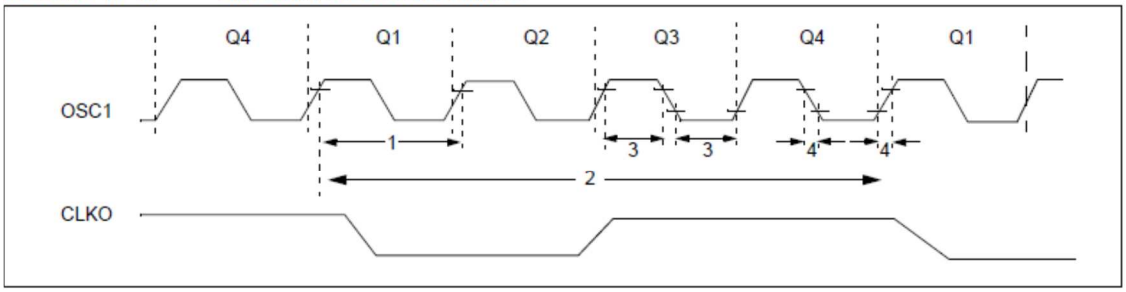


TABLE 17-3: EXTERNAL CLOCK TIMING REQUIREMENTS

Param No.	Symbol	Characteristic	Min	Typ†	Max	Units	Conditions
	FOSC	External CLKI Frequency (Note 1)	DC	—	1	MHz	XT and RC Osc mode
			DC	—	20	MHz	HS Osc mode
			DC	—	32	kHz	LP Osc mode
		Oscillator Frequency (Note 1)	DC	—	4	MHz	RC Osc mode
			0.1	—	4	MHz	XT Osc mode
			4	—	20	MHz	HS Osc mode
			5	—	200	kHz	LP Osc mode
1	TOSC	External CLKI Period (Note 1)	1000	—	—	ns	XT and RC Osc mode
			50	—	—	ns	HS Osc mode
			5	—	—	µs	LP Osc mode
		Oscillator Period (Note 1)	250	—	—	ns	RC Osc mode
			250	—	1	µs	XT Osc mode
			100	—	250	ns	HS Osc mode
			50	—	250	ns	HS Osc mode
			31.25	—	—	µs	LP Osc mode
2	TCY	Instruction Cycle Time (Note 1)	200	TCY	DC	ns	TCY = 4/FOSC
3	TOSL, TOSH	External Clock in (OSC1) High or Low Time	100	—	—	ns	XT oscillator
			2.5	—	—	µs	LP oscillator
			15	—	—	ns	HS oscillator
4	TOSR, TOSF	External Clock in (OSC1) Rise or Fall Time	—	—	25	ns	XT oscillator
			—	—	50	ns	LP oscillator
			—	—	15	ns	HS oscillator

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: Instruction cycle period (TCY) equals four times the input oscillator time base period. All specified values are based on characterization data for that particular oscillator type, under standard operating conditions, with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at "min." values with an external clock applied to the OSC1/CLKI pin. When an external clock input is used, the "max." cycle time limit is "DC" (no clock) for all devices.

TABLE 17-14: A/D CONVERTER CHARACTERISTICS:PIC16F873A/874A/876A/877A (INDUSTRIAL)
PIC16LF873A/874A/876A/877A (INDUSTRIAL

Param No.	Sym	Characteristic		Min	Typ†	Max	Units	Conditions
A01	NR	Resolution		—	—	10-bits	bit	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A03	EIL	Integral Linearity Error		—	—	< ± 1	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A04	EDL	Differential Linearity Error		—	—	< ± 1	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A06	EOFF	Offset Error		—	—	< ± 2	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A07	EGN	Gain Error		—	—	< ± 1	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A10	—	Monotonicity		—	guaranteed(3)	—	—	VSS ≤ VAIN ≤ VREF
A20	VREF	Reference Voltage (VREF+ – VREF-)		2.0	—	VDD + 0.3	V	
A21	VREF+	Reference Voltage High		AVDD – 2.5V		AVDD + 0.3V	V	
A22	VREF-	Reference Voltage Low		AVSS – 0.3V		VREF+ – 2.0V	V	
A25	VAIN	Analog Input Voltage		VSS – 0.3V	—	VREF + 0.3V	V	
A30	ZAIN	Recommended Impedance of Analog Voltage Source		—	—	2.5	kΩ	**(Note 4)**
A40	IAD	A/D Conversion Current (VDD)	PIC16F87XA	—	220	—	µA	Average current consumption when A/D is on **(Note 1)**
			PIC16LF87XA	—	90	—	µA	
A50	IREF	VREF Input Current **(Note 2)**		—	—	5	µA	During VAIN acquisition. Based on differential of VHOLD to VAIN to charge CHOLD, see **Section 11.1 "A/D Acquisition Requirements"**.
				—	—	150	µA	During A/D conversion cycle

* These parameters are characterized but not tested.

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

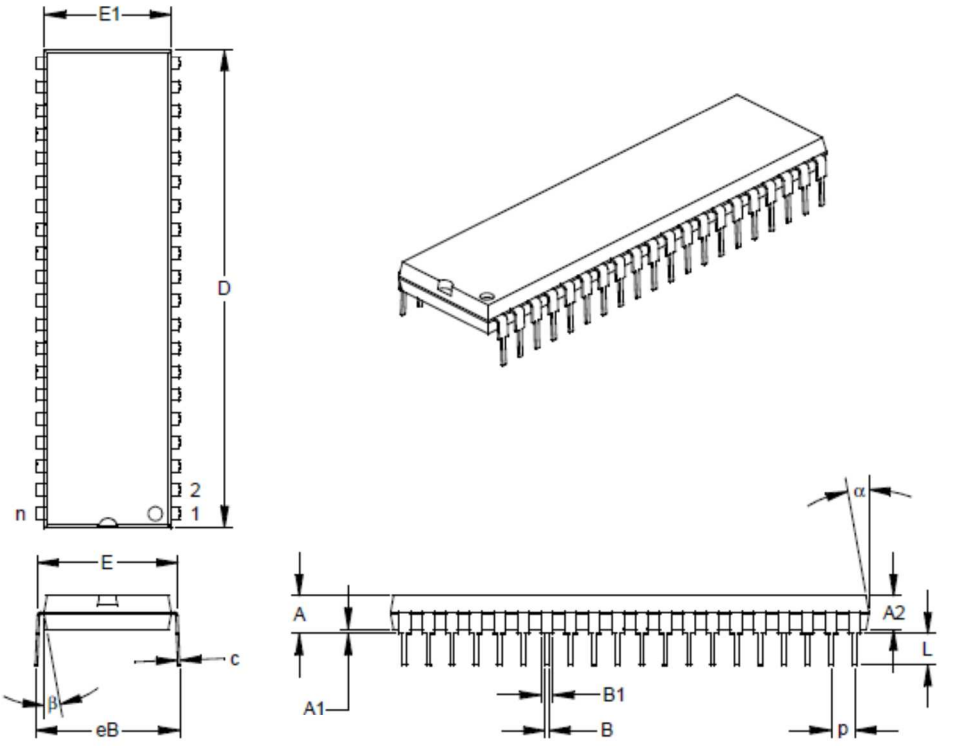
Note 1: When A/D is off, it will not consume any current other than minor leakage current. The power-down current spec includes any such leakage from the A/D module.

2: VREF current is from RA3 pin or VDD pin, whichever is selected as reference input.

3: The A/D conversion result never decreases with an increase in the input voltage and has no missing codes.

4: Maximum allowed impedance for analog voltage source is 10 kΩ. This requires higher acquisition time.

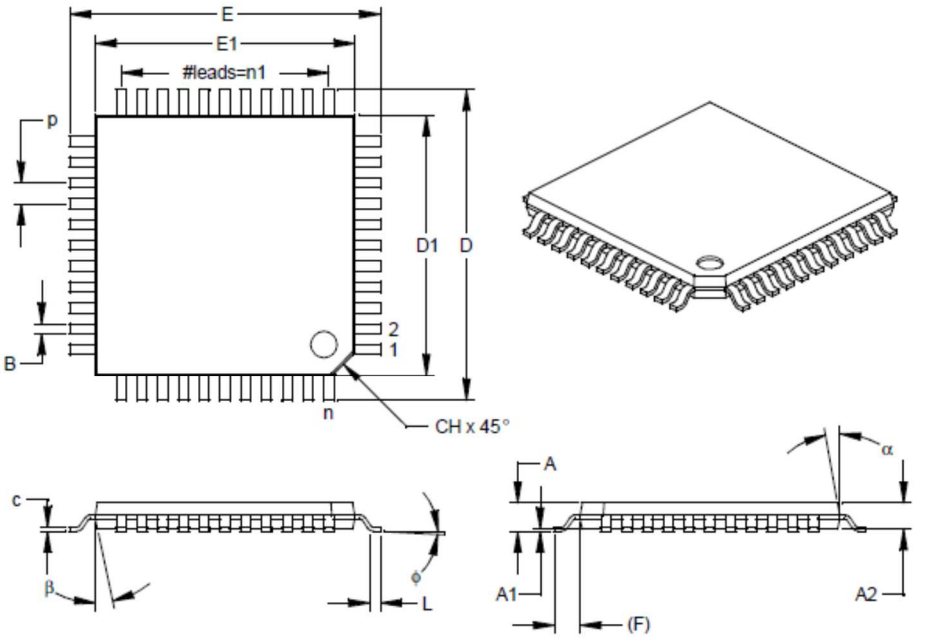
40-Lead Plastic Dual In-line (P) – 600 mil (PDIP)



Units		INCHES*			MILLIMETERS		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		40			40	
Pitch	p		.100			2.54	
Top to Seating Plane	A	.160	.175	.190	4.06	4.45	4.83
Molded Package Thickness	A2	.140	.150	.160	3.56	3.81	4.06
Base to Seating Plane	A1	.015			0.38		
Shoulder to Shoulder Width	E	.595	.600	.625	15.11	15.24	15.88
Molded Package Width	E1	.530	.545	.560	13.46	13.84	14.22
Overall Length	D	2.045	2.058	2.065	51.94	52.26	52.45
Tip to Seating Plane	L	.120	.130	.135	3.05	3.30	3.43
Lead Thickness	c	.008	.012	.015	0.20	0.29	0.38
Upper Lead Width	B1	.030	.050	.070	0.76	1.27	1.78
Lower Lead Width	B	.014	.018	.022	0.36	0.46	0.56
Overall Row Spacing §	eB	.620	.650	.680	15.75	16.51	17.27
Mold Draft Angle Top	α	5	10	15	5	10	15
Mold Draft Angle Bottom	β	5	10	15	5	10	15

* Controlling Parameter
§ Significant Characteristic
Notes:
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.
JEDEC Equivalent: MO-011
Drawing No. C04-016

44-Lead Plastic Thin Quad Flatpack (PT) 10x10x1 mm Body, 1.0/0.10 mm Lead Form (TQFP)



Units		INCHES			MILLIMETERS*		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		44			44	
Pitch	p		.031			0.80	
Pins per Side	n1		11			11	
Overall Height	A	.039	.043	.047	1.00	1.10	1.20
Molded Package Thickness	A2	.037	.039	.041	0.95	1.00	1.05
Standoff §	A1	.002	.004	.006	0.05	0.10	0.15
Foot Length	L	.018	.024	.030	0.45	0.60	0.75
Footprint (Reference)	(F)		.039		1.00		
Foot Angle	φ	0	3.5	7	0	3.5	7
Overall Width	E	.463	.472	.482	11.75	12.00	12.25
Overall Length	D	.463	.472	.482	11.75	12.00	12.25
Molded Package Width	E1	.390	.394	.398	9.90	10.00	10.10
Molded Package Length	D1	.390	.394	.398	9.90	10.00	10.10
Lead Thickness	c	.004	.006	.008	0.09	0.15	0.20
Lead Width	B	.012	.015	.017	0.30	0.38	0.44
Pin 1 Corner Chamfer	CH	.025	.035	.045	0.64	0.89	1.14
Mold Draft Angle Top	α	5	10	15	5	10	15
Mold Draft Angle Bottom	β	5	10	15	5	10	15

* Controlling Parameter
§ Significant Characteristic

Notes:
Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.
JEDEC Equivalent: MS-026
Drawing No. C04-076

APPENDIX C: CONVERSION CONSIDERATIONS

Considerations for converting from previous versions of devices to the ones listed in this data sheet are listed in Table C-1.

TABLE C-1: CONVERSION CONSIDERATIONS

Characteristic	PIC16C7X	PIC16F87X	PIC16F87XA
Pins	28/40	28/40	28/40
Timers	3	3	3
Interrupts	11 or 12	13 or 14	14 or 15
Communication	PSP, USART, SSP (SPI, I^2C Slave)	PSP, USART, SSP (SPI, I^2C Master/Slave)	PSP, USART, SSP (SPI, I^2C Master/Slave)
Frequency	20 MHz	20 MHz	20 MHz
Voltage	2.5V-5.5V	2.2V-5.5V	2.0V-5.5V
A/D	8-bit, 4 conversion clock selects	10-bit, 4 conversion clock selects	10-bit, 7 conversion clock selects
CCP	2	2	2
Comparator	—	—	2
Comparator Voltage Reference	—	—	Yes
Program Memory	4K, 8K EPROM	4K, 8K Flash (Erase/Write on single-word)	4K, 8K Flash (Erase/Write on four-word blocks)
RAM	192, 368 bytes	192, 368 bytes	192, 368 bytes
EEPROM Data	None	128, 256 bytes	128, 256 bytes
Code Protection	On/Off	Segmented, starting at end of program memory	On/Off
Program Memory Write Protection	—	On/Off	Segmented, starting at beginning of program memory
Other	—	In-Circuit Debugger, Low-Voltage Programming	In-Circuit Debugger, Low-Voltage Programming

PIC16F882/883/884/886/887
Data Sheet

28/40/44-Pin, Enhanced Flash-Based 8-Bit CMOS Microcontrollers with nanoWatt Technology

DS41291F

MICROCHIP

PIC16F882/883/884/886/887

28/40/44-Pin Flash-Based, 8-Bit CMOS Microcontrollers with nanoWatt Technology

High-Performance RISC CPU:

- Only 35 Instructions to Learn:
  - All single-cycle instructions except branches
- Operating Speed:
  - DC – 20 MHz oscillator/clock input
  - DC – 200 ns instruction cycle
- Interrupt Capability
- 8-Level Deep Hardware Stack
- Direct, Indirect and Relative Addressing modes

Special Microcontroller Features:

- Precision Internal Oscillator:
  - Factory calibrated to ±1%
  - Software selectable frequency range of 8 MHz to 31 kHz
  - Software tunable
  - Two-Speed Start-up mode
  - Crystal fail detect for critical applications
  - Clock mode switching during operation for power savings
- Power-Saving Sleep mode
- Wide Operating Voltage Range (2.0V-5.5V)
- Industrial and Extended Temperature Range
- Power-on Reset (POR)
- Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Brown-out Reset (BOR) with Software Control Option
- Enhanced Low-Current Watchdog Timer (WDT) with On-Chip Oscillator (software selectable nominal 268 seconds with full prescaler) with software enable
- Multiplexed Master Clear with Pull-up/Input Pin
- Programmable Code Protection
- High Endurance Flash/EEPROM Cell:
  - 100,000 write Flash endurance
  - 1,000,000 write EEPROM endurance
  - Flash/Data EEPROM retention: > 40 years
- Program Memory Read/Write during run time
- In-Circuit Debugger (on board)

Low-Power Features:

- Standby Current:
  - 50 nA @ 2.0V, typical
- Operating Current:
  - 11 µA @ 32 kHz, 2.0V, typical
  - 220 µA @ 4 MHz, 2.0V, typical
- Watchdog Timer Current:
  - 1 µA @ 2.0V, typical

Peripheral Features:

- 24/35 I/O Pins with Individual Direction Control:
  - High current source/sink for direct LED drive
  - Interrupt-on-Change pin
  - Individually programmable weak pull-ups
  - Ultra Low-Power Wake-up (ULPWU)
- Analog Comparator Module with:
  - Two analog comparators
  - Programmable on-chip voltage reference (CV_{REF}) module (% of V_{DD})
  - Fixed voltage reference (0.6V)
  - Comparator inputs and outputs externally accessible
  - SR Latch mode
  - External Timer1 Gate (count enable)
- A/D Converter:
  - 10-bit resolution and 11/14 channels
- Timer0: 8-bit Timer/Counter with 8-bit Programmable Prescaler
- Enhanced Timer1:
  - 16-bit timer/counter with prescaler
  - External Gate input mode
  - Dedicated low-power 32 kHz oscillator
- Timer2: 8-bit Timer/Counter with 8-bit Period Register, Prescaler and Postscaler
- Enhanced Capture, Compare, PWM+ Module:
  - 16-bit Capture, max. resolution 12.5 ns
  - Compare, max. resolution 200 ns
  - 10-bit PWM with 1, 2 or 4 output channels, programmable "dead time", max. frequency 20 kHz
  - PWM output steering control
- Capture, Compare, PWM Module:
  - 16-bit Capture, max. resolution 12.5 ns
  - 16-bit Compare, max. resolution 200 ns
  - 10-bit PWM, max. frequency 20 kHz
- Enhanced USART Module:
  - Supports RS-485, RS-232, and LIN 2.0
  - Auto-Baud Detect
  - Auto-Wake-Up on Start bit
- In-Circuit Serial Programming™ (ICSP™) via Two Pins
- Master Synchronous Serial Port (MSSP) Module supporting 3-wire SPI (all 4 modes) and I^2C™ Master and Slave Modes with I^2C Address Mask

Pin Diagrams – PIC16F882/883/886, 28-Pin PDIP, SOIC, SSOP

28-pin PDIP, SOIC, SSOP

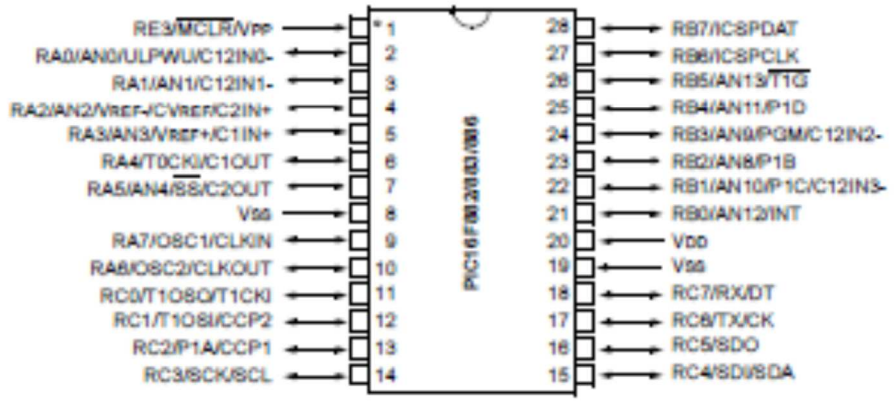


TABLE 1: PIC16F882/883/886 28-PIN SUMMARY (PDIP, SOIC, SSOP)

I/O	Pin	Analog	Comparators	Timers	ECCP	EUSART	MSSP	Interrupt	Pull-up	Basic
RA0	2	AN0/ULPWU	C12IN0-	—	—	—	—	—	—	—
RA1	3	AN1	C12IN1-	—	—	—	—	—	—	—
RA2	4	AN2	C2IN+	—	—	—	—	—	—	VREF-/CVREF
RA3	5	AN3	C1IN+	—	—	—	—	—	—	VREF+
RA4	6	—	C1OUT	T0CKI	—	—	—	—	—	—
RA5	7	AN4	C2OUT	—	—	—	SS	—	—	—
RA6	10	—	—	—	—	—	—	—	—	OSC2/CLKOUT
RA7	9	—	—	—	—	—	—	—	—	OSC1/CLKIN
RB0	21	AN12	—	—	—	—	—	IOC/INT	Y	—
RB1	22	AN10	C12IN3-	—	P1C	—	—	IOC	Y	—
RB2	23	AN8	—	—	P1B	—	—	IOC	Y	—
RB3	24	AN9	C12IN2-	—	—	—	—	IOC	Y	PGM
RB4	25	AN11	—	—	P1D	—	—	IOC	Y	—
RB5	26	AN13	—	T1G	—	—	—	IOC	Y	—
RB6	27	—	—	—	—	—	—	IOC	Y	ICSPCLK
RB7	28	—	—	—	—	—	—	IOC	Y	ICSPDAT
RC0	11	—	—	T1OSO/T1CKI	—	—	—	—	—	—
RC1	12	—	—	T1OSI	CCP2	—	—	—	—	—
RC2	13	—	—	—	CCP1/P1A	—	—	—	—	—
RC3	14	—	—	—	—	—	SCK/SCL	—	—	—
RC4	15	—	—	—	—	—	SDI/SDA	—	—	—
RC5	16	—	—	—	—	—	SDO	—	—	—
RC6	17	—	—	—	—	TX/CK	—	—	—	—
RC7	18	—	—	—	—	RX/DT	—	—	—	—
RE3	1	—	—	—	—	—	—	—	Y(1)	MCLR/VPP
—	20	—	—	—	—	—	—	—	—	VDD
—	8	—	—	—	—	—	—	—	—	VSS
—	19	—	—	—	—	—	—	—	—	VSS

Note 1: Pull-up activated only with external MCLR configuration.

Pin Diagrams – PIC16F882/883/886, 28-Pin QFN

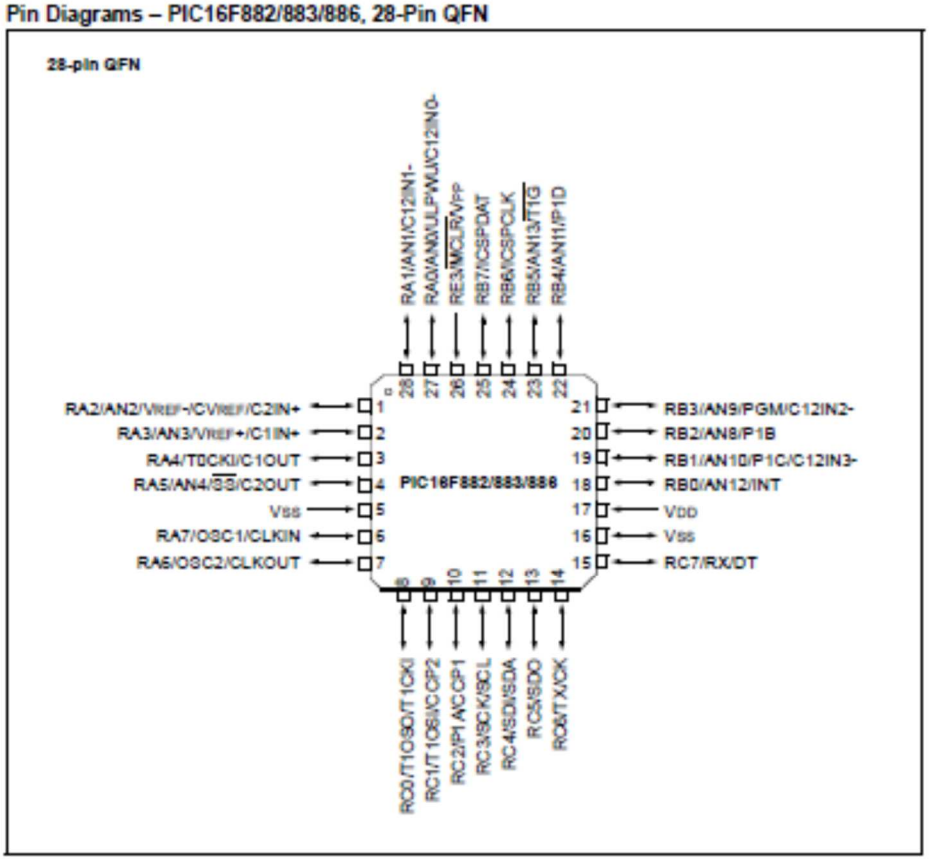
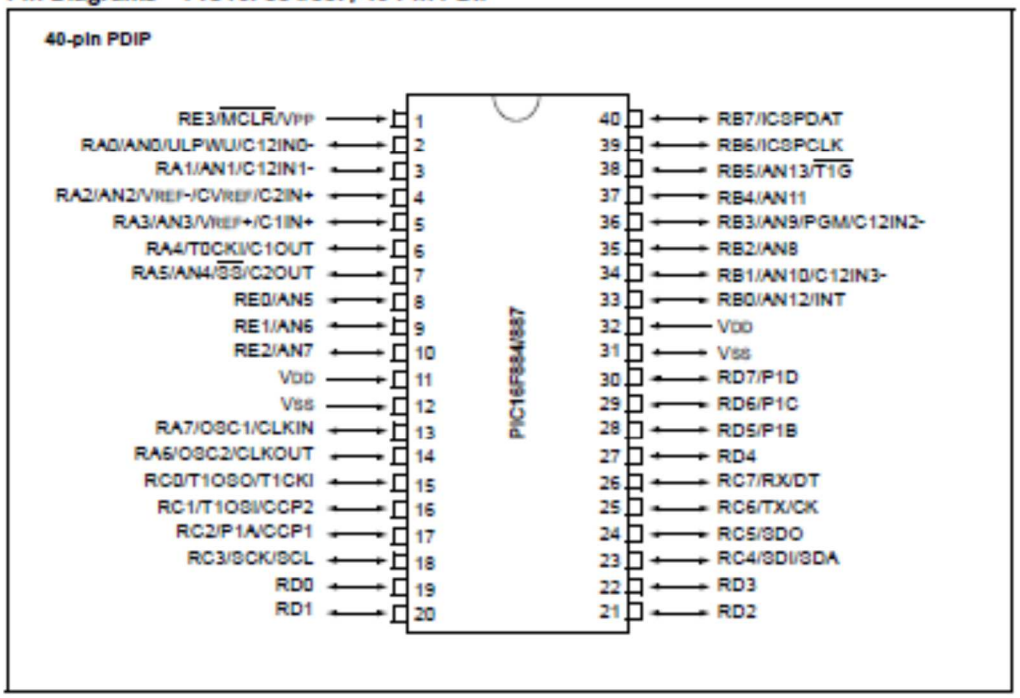


TABLE 2: PIC16F882/883/886 28-PIN SUMMARY (QFN)

I/O	Pin	Analog	Comparators	Timers	ECCP	EUSART	MSSP	Interrupt	Pull-up	Basic
RA0	27	AN0/ULPWU	C12IN0-	—	—	—	—	—	—	—
RA1	28	AN1	C12IN1-	—	—	—	—	—	—	—
RA2	1	AN2	C2IN+	—	—	—	—	—	—	VREF-/CVREF
RA3	2	AN3	C1IN+	—	—	—	—	—	—	VREF+
RA4	3	—	C1OUT	T0CKI	—	—	—	—	—	—
RA5	4	AN4	C2OUT	—	—	—	SS	—	—	—
RA6	7	—	—	—	—	—	—	—	—	OSC2/CLKOUT
RA7	6	—	—	—	—	—	—	—	—	OSC1/CLKIN
RB0	18	AN12	—	—	—	—	—	IOC/INT	Y	—
RB1	19	AN10	C12IN3-	—	P1C	—	—	IOC	Y	—
RB2	20	AN8	—	—	P1B	—	—	IOC	Y	—
RB3	21	AN9	C12IN2-	—	—	—	—	IOC	Y	PGM
RB4	22	AN11	—	—	P1D	—	—	IOC	Y	—
RB5	23	AN13	—	T1G	—	—	—	IOC	Y	—
RB6	24	—	—	—	—	—	—	IOC	Y	ICSPCLK
RB7	25	—	—	—	—	—	—	IOC	Y	ICSPDAT
RC0	8	—	—	T1OSO/T1CKI	—	—	—	—	—	—
RC1	9	—	—	T1OSI	CCP2	—	—	—	—	—
RC2	10	—	—	—	CCP1/P1A	—	—	—	—	—
RC3	11	—	—	—	—	—	SCK/SCL	—	—	—
RC4	12	—	—	—	—	—	SDI/SDA	—	—	—
RC5	13	—	—	—	—	—	SDO	—	—	—
RC6	14	—	—	—	—	TX/CK	—	—	—	—
RC7	15	—	—	—	—	RX/DT	—	—	—	—
RE3	26	—	—	—	—	—	—	—	Y(1)	MCLR/VPP
—	17	—	—	—	—	—	—	—	—	VDD
—	5	—	—	—	—	—	—	—	—	VSS
—	16	—	—	—	—	—	—	—	—	VSS

Note 1: Pull-up activated only with external MCLR configuration.

Pin Diagrams – PIC16F884/887, 40-Pin PDIP



Pin Diagrams – PIC16F884/887, 44-Pin QFN

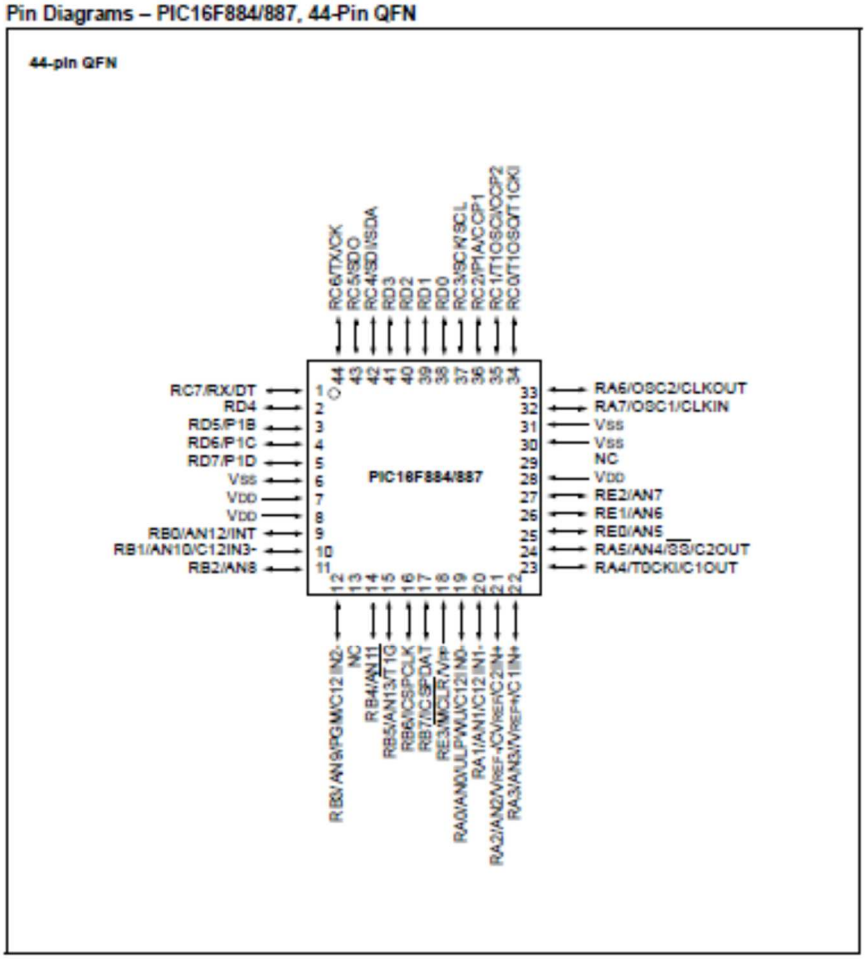


FIGURE 1-1: PIC16F882/883/886 BLOCK DIAGRAM

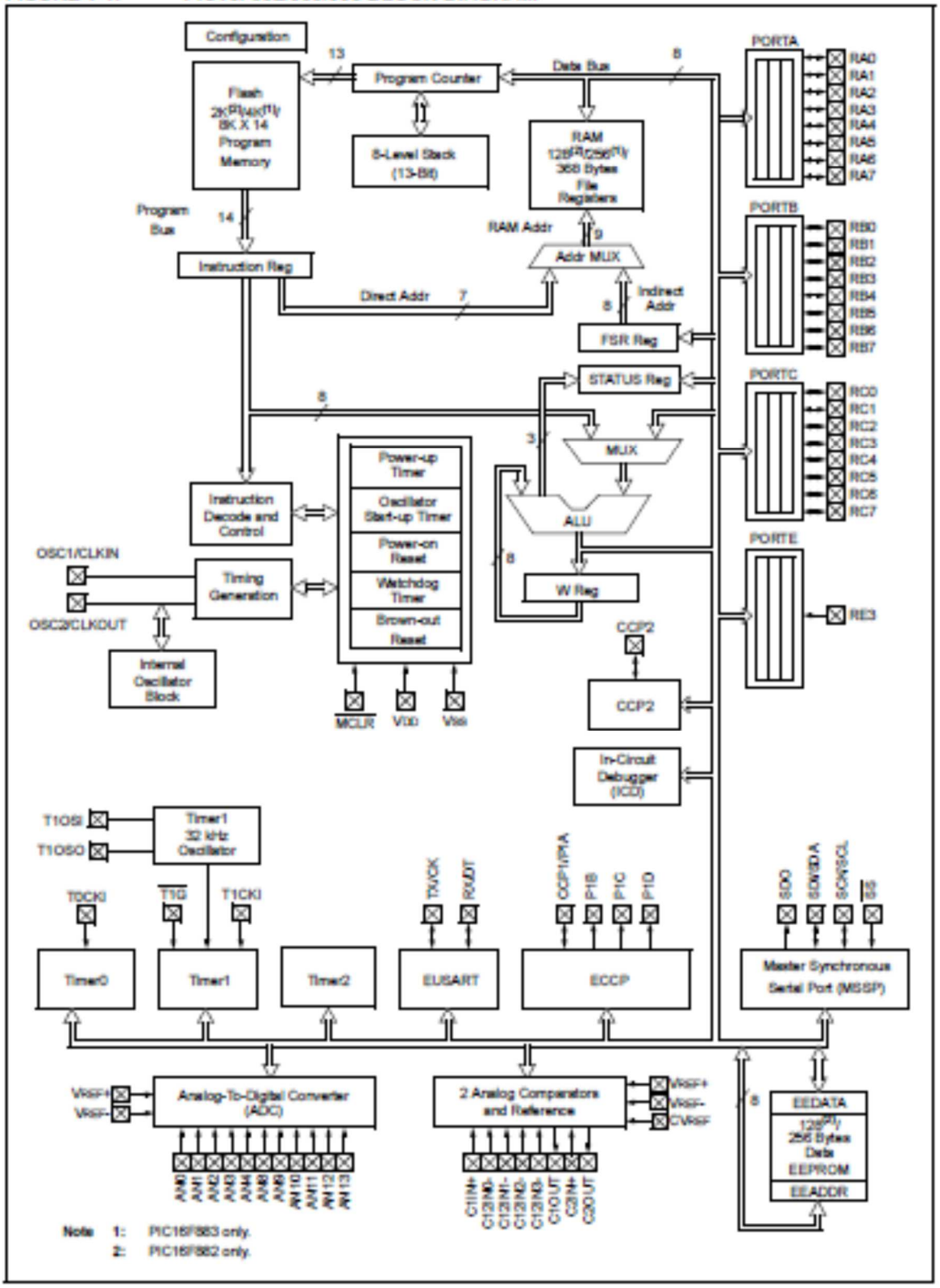


FIGURE 1-2: PIC16F884/PIC16F887 BLOCK DIAGRAM