MİKROİŞLEMCİLER VE MİKRODENETLEYİCİLER

Hazırlayan: Öğ. Gör. Sabri Uzuner

2. Hafta

PIC Programlama Teknikleri

Bir PIC'i programlamak için üç basit adım vardır.

- *Kaynak kodu yazılımı*
- *Kaynak kodu derlenerek makine diline çevrimi*
- *Makine diline çevrilmiş programı PIC'e yazdırma*

Bunu yapabilmek içinde bir çok farklı derleyiciler bulunmaktadır.

- *PIC Assembler*
- *Parallax*
- *Basic*
- *LET Basic*
- *Pic Basic Pro*
- *JAL*
- *CCS C*

PicBasic Pro Özellikleri

BASIC Advantages

- Very easy to learn and use.
- A BASIC compiler will produce code that runs fast as a C compiler.
- Many in built functions (depending on compiler).
- Very popular – large user base with many example programs.

BASIC Disadvantages

- Non standard language.
- The language is not standardized it will be difficult to move code to a new processor target type.

PicBasic Pro ile Kullanılan PICmikro'lar

PIC10 (prefix)	small, inexpensive devices that use the 12-bit or 14-bit instruction sets
PIC12 (prefix)	8-pin devices that may use the 12-bit, 14-bit, or enhanced 14-bit instruction sets
PIC16 (prefix)	devices ranging from 14-pin to 64-pin that may use the 12-bit, 14-bit, or enhanced 14-bit instruction sets
PIC18 (prefix)	high-performance devices that use the 16-bit instruction set
12-bit instruction set (denoted as "Baseline Architecture" by Microchip)	mostly attractive for low cost, these devices may not be compatible with all PBP commands, PBP LONG variables not available
14-bit instruction set (denoted as "Mid-Range Architecture" by Microchip)	compatible with all PBP commands, PBP LONG variables not available
Enhanced 14-bit instruction set (denoted as "Enhanced Mid-Range Architecture" by Microchip)	compatible with all PBP commands, some features in common with 16-bit instruction set, PBP LONG variables not available
16-bit instruction set (denoted as "PIC18 Architecture" by Microchip)	All devices with prefix PIC18, compatible with all PBP commands, PBP LONG variables supported, high-speed and generally better performance than the other families supported by PBP

PBP supports more than 500 microcontrollers

Compile Modes PBPW and PBPL

These represent compilation modes that control the maximum size variable type in PBP. WORD variables are 16-bits wide and can hold values 0 to 65535. LONG variables are 32-bits wide and can hold values -2147483648 to 2147483647.

Picbasic Pro Derleyicisinin Kullanılışı

PicBasic Pro derleyicisi DOS komut satırından aşağıdaki formatta yazılarak çalıştırılır;

PBP Seçenek dosya_adı

Örneğin;

PBP –p16C71 –ampasm yakson

Komut Satırı Seçenekleri

A Seçeneği

PBP, Mikrochip'in MPASM derleyicisini veya PicBasic Pro derleyicisi olan PM'yi kullanabilir. Aksi belirtilmedikçe otomatik olarak PM derleyicisi çalıştırılır. Eğer MPASM derleyicisi kullanılmak isteniyorsa komut satırında aşağıdaki örnekte olduğu gibi yazılmalıdır.

PBP -ampasm dosya_adı

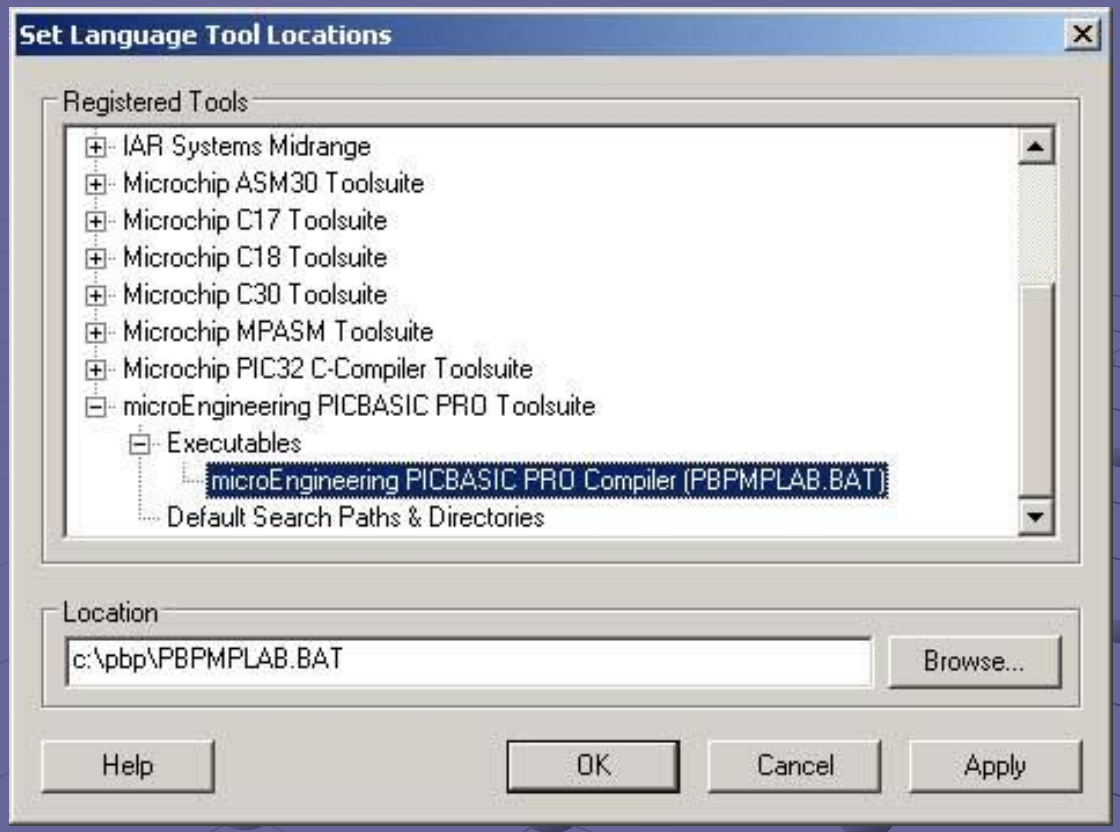
–C seçeneği

-C seçeneği assembly kaynak dosyası içerisinde PicBasic Pro kaynak dosyalarını açıklama satırları olarak yazılmasını sağlar. PicBasic Pro komutlarının yanına assembly komutlarının yazılması, PIC assembly PicBasic Pro'ya daha kolay geçiş yapabilmeleri yada hata arama esnasında kolaylık olması bakımından çok yararlıdır. Yazılış biçimi aşağıdaki gibidir.

PBP -C dosya_adı

Option	Description
A	Use a different Assembler
C	Insert source lines as Comments into assembler file
E	Output errors to a file
H or ?	Display Help screen
K+	Add source level debugging information (COFF)
K-	Add assembler level debugging information (COFF)
L	Use a different Library file
O	Pass Option to assembler
P	Specify target Processor
S	Skip execution of assembler after compilation
V	Verbose mode
Z	Add source level debugging information (COD)

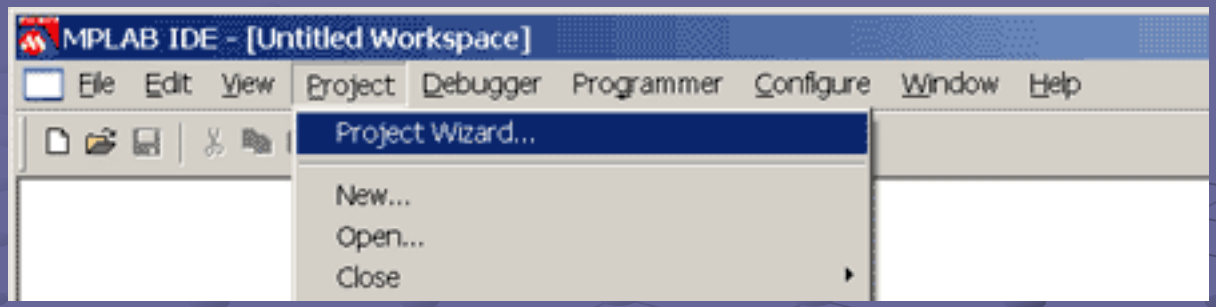
PBP'u MPLAB'a Dahil Etmek



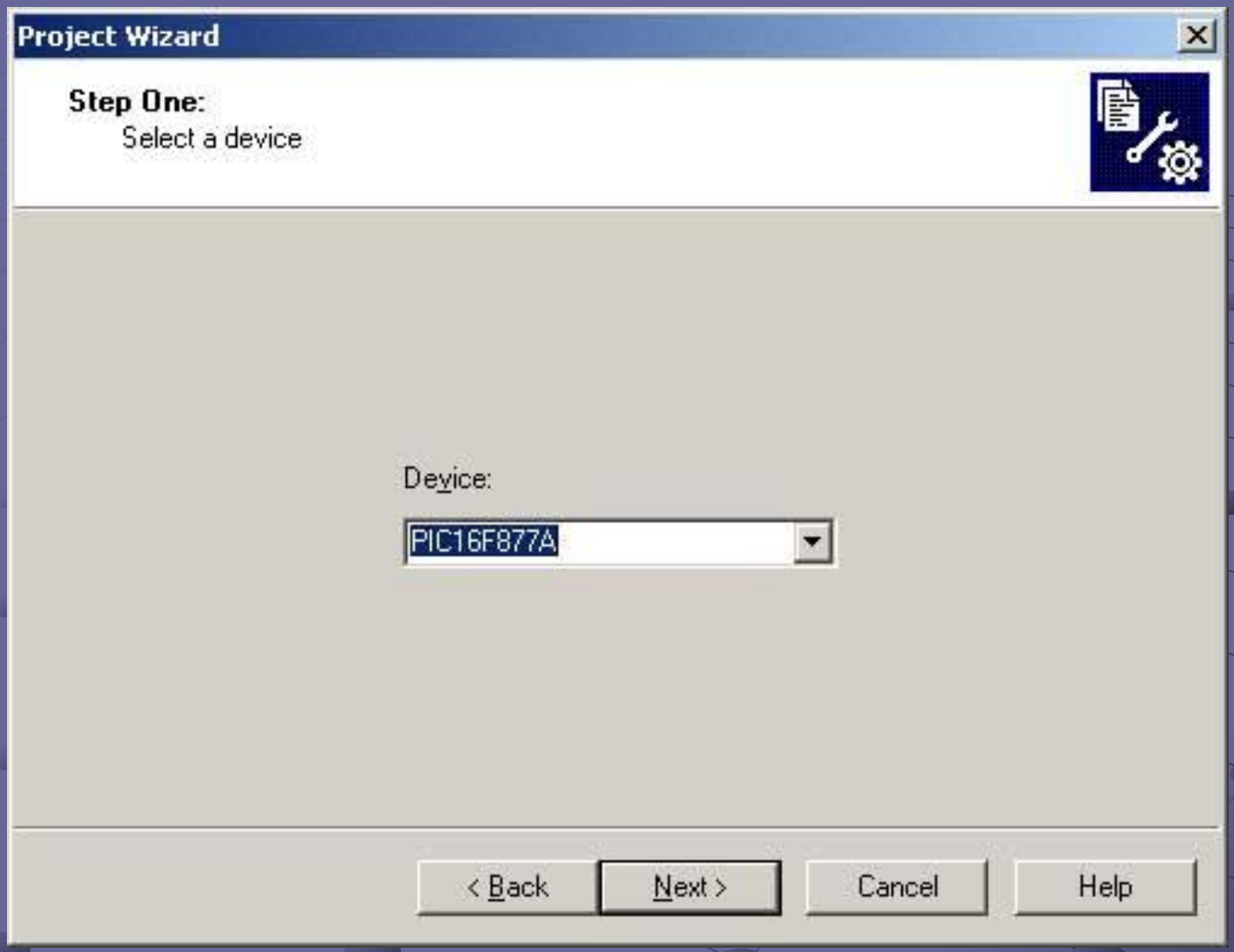
Start MPLAB and select Set Language Tool Locations under the Project menu. Find the "microEngineering PICBASIC PRO Toolsuite" and expand the levels until you can highlight the entry under "Executables" as shown.

Use the Browse button to locate and select the file **PBPMPLAB.BAT**, found in the PBP install folder (default location is shown). *Note that the browse dialog will only show .EXE files by default. Change the "Files of type" setting to "All Files" in order to select the .BAT file.*

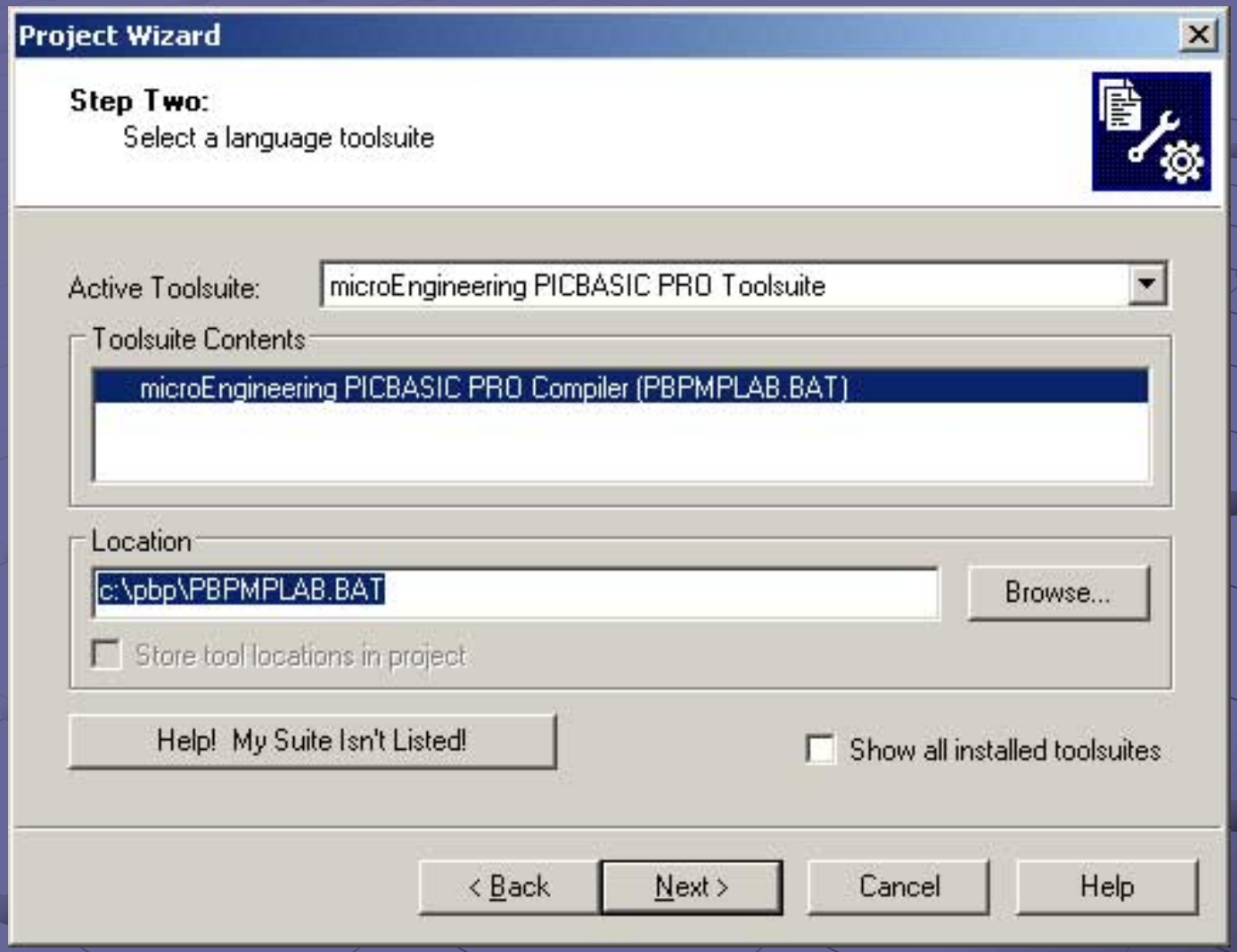
NOTE: If the PICBASIC PRO Toolsuite does not appear in MPLAB, close MPLAB and download and install theMPLAB Plugin.



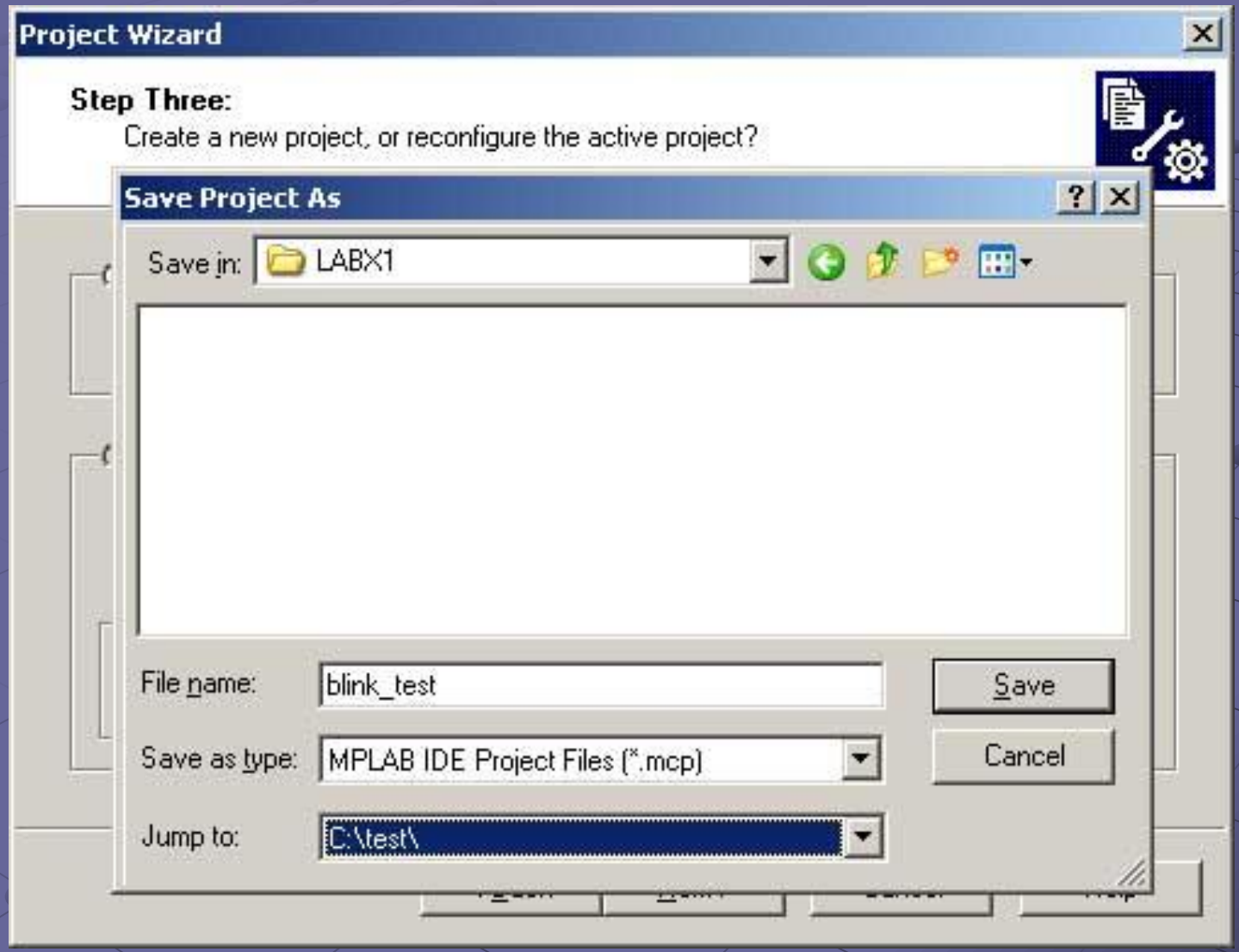
Step One: Select the desired target device, click Next:



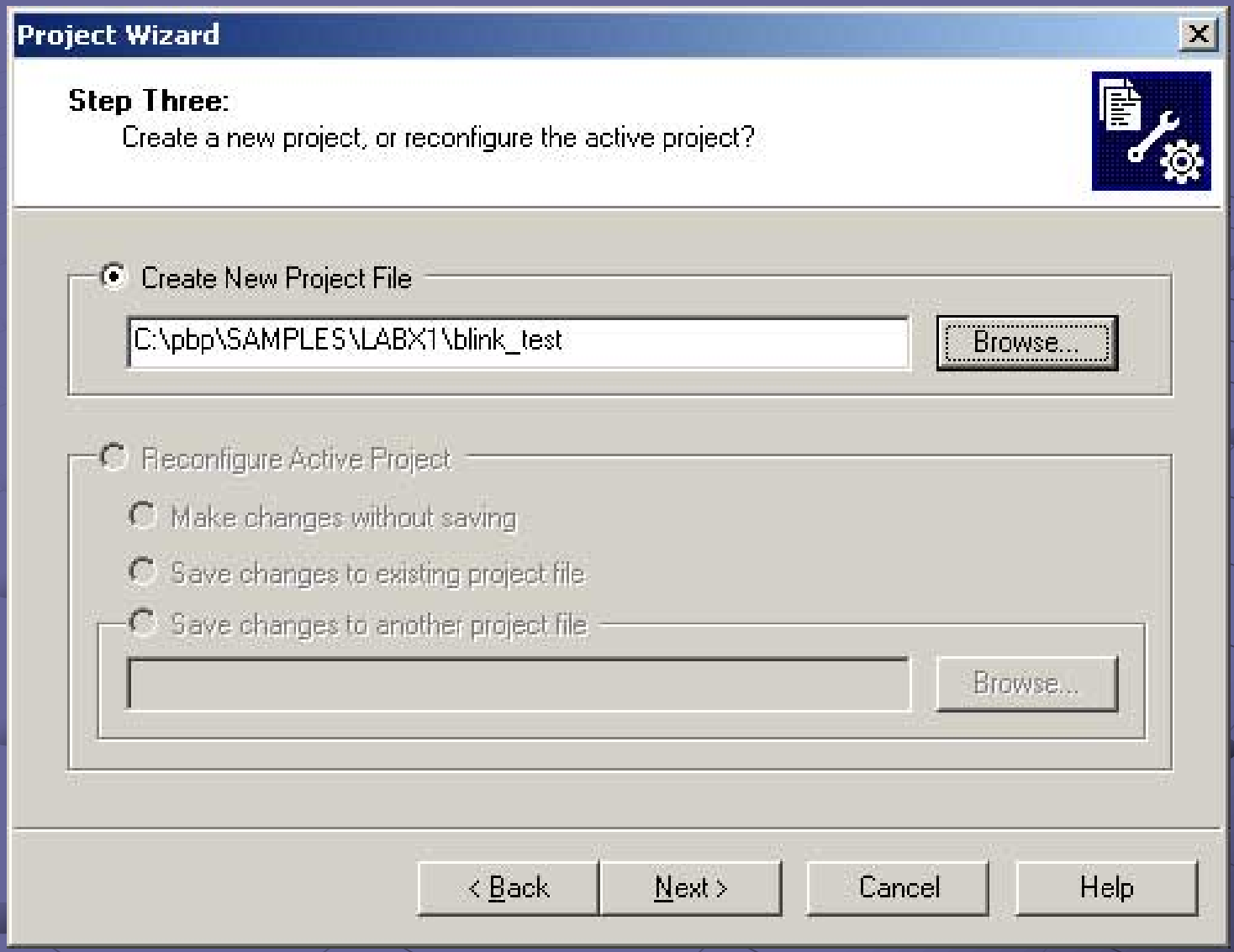
Step Two: Select the PICBASIC PRO Toolsuite, click Next:



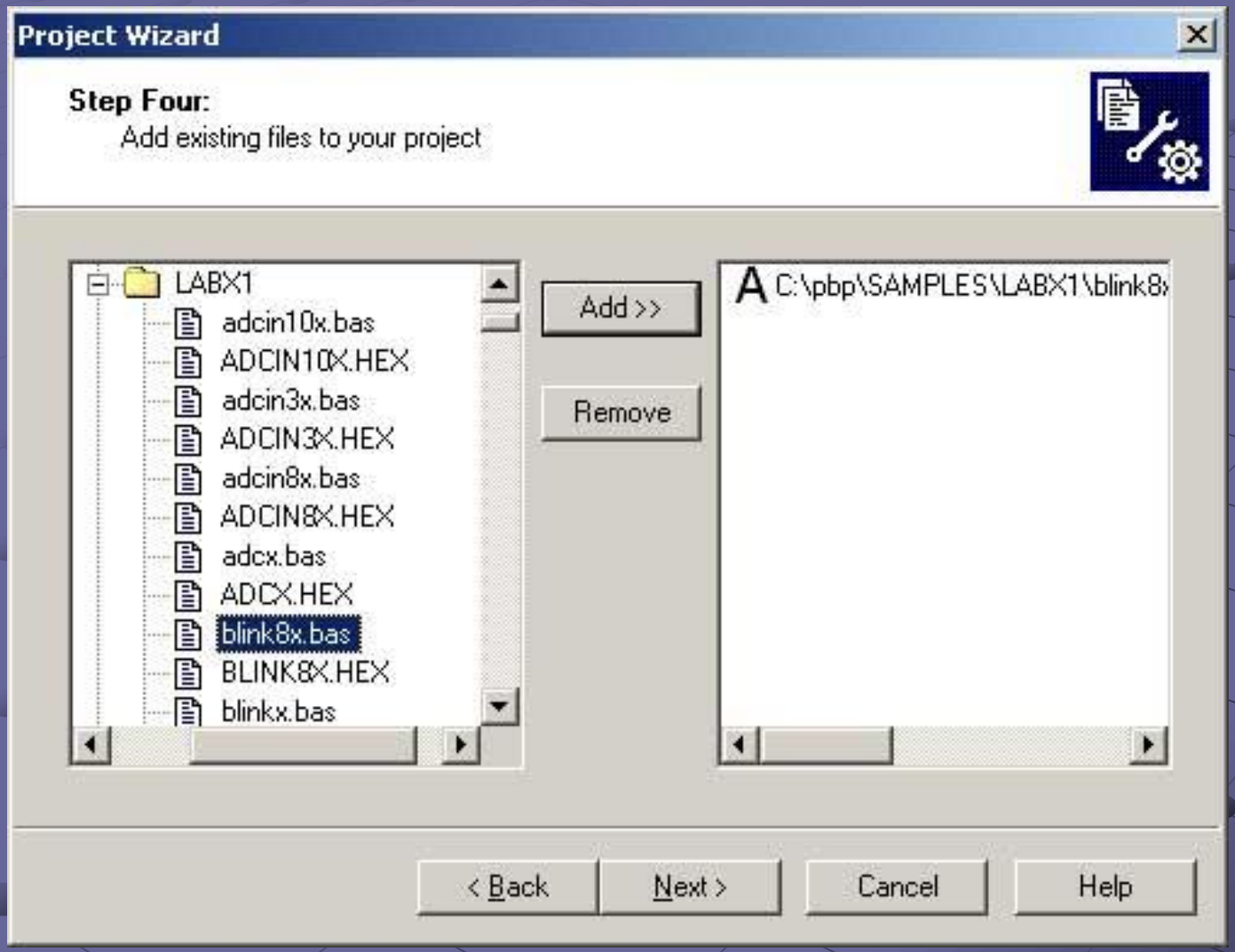
Step Three: Click the Browse button under Create New Project File, locate folder where the project will be located, type in a file name for the project, click Save:



Still on Step Three: Make sure the path and name are correct for the project, click Next:



Step Four: Pick one (only one) .BAS or .PBP file that represents the main program file for your project, click the Add button so it shows up on the right, then click Next:



Uygulamada Karşılaşılabilecek Bazı Problemler ve Çözümleri

Sorun: MCLR ucunun açık bırakılması gerilim seviyesinde dalgalanmalara sebep olur. Bu nedenle normal çalışan program aniden reset atabilir.

Çözüm: 5V'luk gerilim kaynağının + ucuna 4,7K lık bir dirençle doğrudan veya gerilim sınırlayıcı bir dirençle birlikte kullanılan reset butonu ile bağlanması gerekir.

Sorun: Osilatör devresinde kristal ile kondasatör uyumu önemlidir.Yüksek değerde kondansatör değeri seçmek osilatör devresinin çalışamamasına sebep olur.

Çözüm:

Osilatör Tipi	Frekans	Kondansator
LP	32KHz	33-68pF
	200KHz	15-47pF
	100KHz	47-100pF
XT	500KHz	20-68pF
	1MHz	15-68pF
	2MHz	15-47pF
	4MHz	15-33pF
HS	8MHz	15-47pF
	20MHz	15-47pF

Sorun: Güç kaynağından kaynaklanan problemler. Çıkışa bağlanan cihazlardan dolayı çok büyük gerilim düşmelerine sebep olur.
Çözüm: İyi regüle edilmiş güç kaynakları kullanılması gerekir.

Some devices have features that can interfere with expected pin operations. Many PIC MCUs have analogcomparators on PORTA or another port. When these chips start up,PORTA is set to analog mode. This makes the pin functions on PORTA work in an unexpected manner. To change the pins to digital, simply add the line:

CMCON = 7 (For example PIC16F628)
ADCON1=7 (For example PIC16F877)
ANSEL=0 (PIC12F675 and 16F767)
Another example of potential disaster is that PORTA, pin 4 exhibits unusual behavior when used as an output. This is because the pin has an open drain output rather than the usual bipolar stage of the rest of the output pins. This means it can pull to ground when set to 0, but it will simply float when set to a 1, instead of going high. To make this pin act in the expected manner, add a pull-up resistor between the pin and 5 volts. The value of the resistor may be between 1K and 33K, depending on the drive necessary for the connected input. This pin acts as anyother pin when used as an input.

Picbasic Pro Temel Kavramlar

Tanımlayıcılar : Tanımlayıcılar, değişken adı ve satır etiketlerini tanımlamakta kullanılan programcının verdiği isimlerdir. Tanımlayıcılar harflerden, rakamlardan ve alt çizgilerden (_) oluşabilir. Ancak bir rakamla başlayamazlar. Bir tanımlayıcı da küçük harf, büyük harf duyarlılığı yoktur. Yani "etiket" adında bir tanımlayıcıyla "ETIKET" tanılayıcısı arasında PBP açısından herhangi bir fark yoktur. Etiketlerde kullanılan karakter sayısında herhangi bir sınır olmamasına rağmen PBP sadece ilk 32 karakteri algılar.

Satır Etiketi: Bazı eski BASIC dillerinde olduğu gibi her satır için bir satır numarası kullanılmadığından GOTO ve GOSUB komutlarıyla programın dallanması istenen yere gidebilmesi için referans satırı gereklidir. Etiketler bu referans satırının yerini belirlemek için kullanılır. Etiket tanımlayıcılarından sonra muhakkak iki nokta üst üste (:) yazılmalıdır. Aşağıdaki örnekte etiket tanımlayıcısının adı " LCD_gönder "

```
LCD_gönder :
Serout  0,N2400, ["Merhaba, dünya! ",13,10]
Goto LCD_gönder
```

Değişkenler : Değişkenler, PicBasic Pro programları içerisinde geçici olarak veri saklamaya yarar. VAR anahtar sözcüğü kullanılarak oluşturulurlar. Değişkenler bit, byte,Word veya Long (PBPL modunda) tipinde tanımlanabilirler. Tanımlanan tipe göre mikro denetleyicinin RAM'ı içerisinde yeterli genişlik PBP tarafından otomatik olarak ayarlanır. Bir değişken adı tanımlama formatı aşağıdaki gibidir.

Değişken_adı VAR tip{.alt_eleman}

Değişken_adı, programcının seçtiği herhangi bir tanımlayıcı olabilir. Ancak anahtar kelime PBP komutlarından birisi olamaz. Alt_eleman kullanımı isteğe bağlıdır. Tip olarak BIT, BYTE ,WORD veya LONG (PBPL modunda) kelimeleri kullanılır. Aşağıda değişken adı oluşturmaya birkaç örnek verilmiştir.

Gecikme VAR byte

Saga_don VAR bit

Dur VAR word

Alias'lar (Bir Değişkene Başka Bir İsim Vermek)

Bazen bir değişkene başka bir isim vermek gerekebilir. Bu isimlere "alias" adı verilir. Alias kullanmak bir değişkenin alt_elemanlarını tanımlayarak onlara erişimde kolaylık sağlar.

```
Count  VAR  sayac        ' count , sayac değişkenine verilen başka bir  addır'
b0  VAR  W0.byte0        'b0, W0 kelimesinin (word) ilk byte'ıdır.
                                        (byte0alt_elemandır.)'
b1  VAR  W0.byte1        'b1 , W0 kelimesinin ikinci byte'ıdır
                                        (byte0 alt_elemandır.)'
ilk  VAR  sayac.0        'ilk, sayac'ın 0. bit'idir.'
```

Dizi Değişkenler (Arrays):

Array değişkenler, normal değişkenin oluşturulmasına benzer biçimde aşağıdaki formatta oluşturulurlar.

Değişken_adı VAR tip[Eleman_sayısı]

Değişken adı, programcının seçtiği ve tanımlayıcı belirleme kurallarına uygun olarak oluşturulmuş bir isimdir. Tip olarak BIT, BYTE, WORD veya LONG (PBPL modunda) kullanılabilir. Eleman sayısı ile dizi değişkenin kaç elemandan oluşacağı belirlenir. Aşağıda dizi değişken oluşturmaya birkaç örnek görülmektedir.

Oku VAR byte[14]
Sinyalsayisi VAR bit[8]

Dizi değişkenin ilk elemanının indisi 0 (sıfır) dır. Yukarıdaki oku dizi değişkeninin elemanları oku[0]'dan oku[13]'e kadardır. Toplam 14 eleman vardır. Mikro denetleyicilerdeki RAM'ın sınırlı oluşu nedeniyle her bir tip için aşağıdaki sınırlamalar vardır.

Size	# of bits	Range
BIT	1	0 to 1
BYTE	8	0 to 255
WORD	16	0 to 65535
LONG *	32	-2147483648 to 2147483647

* PBPL only.

Sabitler

Sabit isimleri de değişken isimlerine benzer bir biçimde tanımlanır. Tek farkla, VAR yerine CON anahtar kelimesi kullanılır. Program içerisine sabit değerleri yazmak yerine önceden tanımlı sabit ismini kullanmak iyi programcılık açısından yararlıdır. Bir sabit ismin içerisine program içerisinde değer atanmaz. Sabit ismi tanımlama aşağıdaki gibidir.

Sabit_ismi CON sabit_sayı
Sabit sayı oluşturmaya örnekler;
Adres CON 3
*Satir CON adres*100*

Semboller

Sembol, değişken veya sabit isimlerine alias adı vermenin başka bir yoludur. Sembol ile sabit yada değişken adı oluşturulmaz sadece başka bir isim vermek için kullanılır.

SYMBOL read=oku ‘oku değişkeni daha önceden VAR kullanılarak tanımlanmış olması gerekir’

SYMBOL adres=1 ‘adres CON 1 tanımın aynısıdır’

Sayısal Sabitler

PBP'de sayısal sabitler üç şekilde tanımlanabilir; ondalık, binary ve heksadesimal. Binary sayılar tanımlanırken önüne "%" işareti,heksadesimal sayıları tanımlanırken " $ " işareti konulur. Ondalık sayılar PBP'de kabul edilen sayı biçimidir ve herhangi ön işaret gerekmez.

100	'100 ondalık sayısı'
%100	'4 sayısının binary gösterilişi'
$100	'256 sayısının heksadesimal gösterilişi'

Çift tırnak içerisine yazılan bir karakter ASCII kodu karşılığına çevrilir.Tırnak içerisindeki karakter birden fazla olduğunda string sabit olarak ele alınır.

"A"	'65 sayısının ASCII değeridir.'
"d"	'100 sayısının ASCII değeridir.'

String sabitler

PBP'nin string sabit işleme yeteneği yoktur. Fakat bazı komutlarda kullanılabilirler. Bir string sabit bir veya daha fazla karakterin çift tırnak içerisine yazılmış biçimidir.

"Merhaba" *'"M","e","r","h","a","b","a"* karakterlerinin kısaltılmış biçimidir.'

PBP'de bir string içerisindeki karakterlerin her biri ayrı birer karakter sabit olarak ele alınır.

Pinler

Mikro denetleyici pinlerine erişmek ve kullanmak için birkaç farklı yol vardır. Bir programda pin'i tanımlamak için kullanılacak en kolay yöntem port adını ve bit numarasını birlikte kullanmaktır.

PORTB.1=1 'PORTB'nin 1 nolu ucunu 1 yap'

Pin'in ne amaçla kullanıldığını hatırlamayı kolaylaştırmak amacıyla VAR komutuyla bir isim atamak gerekir. Bundan sonra herhangi bir işlemde, bu isim kullanılabilir.

Led VAR PORTA.0 'PORTA'nın 0. pin'ini led olarak adlandır.'

High Led 'led'i (PORTA.0) high yap'

Açıklama satırı

REM, tek tırnak (') yada (;) ile başlayan tüm PBP satırları açıklama satırları olarak algılanır. REM veya (') veya (;) takip eden karakterler derlenmez.

REM Comments may be made after REM

PORTB.0 = 1 ' All of the characters after the ' are an in-line comment.

GOTO start ; All of the characters after the ; are an in-line comment.

Bir Satıra Birden Fazla Komut Yazmak

Uzun programlarda satır sayısını azaltmak veya birbiri ile ilgili komutları gruplamak amacıyla bir satıra birden fazla komut yazılabilir. Bu durumda komutlar arasına iki nokta üst üste (:) konulmalıdır. Aşağıdaki örnekler birbirinin aynısıdır.

W2 = W0
W0 = W1
W1 = W2

İle aşağıdaki satırlar aynıdır.

W2 = W0 : W0 = W1 : W1 = W2

İkinci örnekte satır sayısı azalmasına rağmen derleme sonunda üretilen makine dili komut sayısında herhangi bir azalma olmaz.

INCLUDE Komutu ile Dosya Dahil Etmek

INCLUDE komutu kullanılarak bir PBP programı içerisine başka BASIC kaynak dosyaları eklenebilir. Standart hale gelmiş bazı alt programları tanımlamaları veya başka bir program dosyalarını ana programımızdan ayrı bir yerde saklamak isteyebilirsiniz. Bu durumda bu program kodlarını kendi programımız içerisinde yazmak zorunda kalmadan sadece gerekli olduklarında INCLUDE komutu kullanarak programa dahil edebilirsiniz. INCLUDE dosyaları derlenmemiştir, yani ".bas" uzantılı kaynak dosyalardır.

INCLUDE komutundan hemen sonra yazılan dosyadaki program kodları oraya yazılmış gibi işlem görür.

INCLUDE "modedefs.bas"

DEFINE komutu

Clock osilatör frekansı ve LCD pin yerleşimi gibi bazı elemanlar PBP içerisinde tanımlıdır. DEFINE komutu istenirse bu tanımlamaların değiştirilmesini sağlar. Önceden tanımlı olan clock osilatörü değerinin değiştirilmesi, DEBUG pin'lerinin ve boud rate ve LCD pin yerleşiminin değiştirilmesi gibi işlemler DEFINE ile değiştirilebileceklerinden bazılarıdır. DEFINE ile yapılan tanımlamaların hepsi büyük harfle yazılmalıdır.

```
DEFINE  BUTTON_PAUSE  10          'Buton arkı söndürme gecikmesi(mS)'
DEFINE  CHAR_PACING  1000    ' Seraut karakter gönderme gecikmesi  (Ms) '
DEFINE  DEBUG_REG  PORTB                  'Debug pini PORTB'de'
DEFINE  DEBUG_BIT  0                      'Debug Pini 0. bit'
DEFINE  DEBUG_BAUD  2400                  'Debug baud rate 2400'
DEFINE  DEBUG_MODE  12                'Debug modu: 0=true(doğru)
DEFINE  LCD_RSBIT  4                   ' LCD registeri bitini seçer'
DEFINE  LCD_EREG  PORTB                    ' LCD enable portu'
DEFINE  LCD_EBIT  3                        ' LCD enable biti'
DEFINE  LCD_BITS  4              ' LCD data yolu biti (4 veya 8 )'
DEFINE  LCD_LINES  2                   ' LCD'deki satır sayısı'
DEFINE  OSC4  4                    ' 3, 4, 8, 10, 12, 16 veya 20'
DEFINE  OSCCAL_1K  1          ' OSCCAL'ı PIC12C671 için kur'
DEFINE  OSCCAL_2K  1          ' OSCCAL'I PIC16C672 için kur'
```

Aritmetik Operatörler

Aritmetik işlemler diğer BASIC dillerinde olduğu gibi hiyerarşik bir düzene göre yapılır. İşlem operatörlerinin birbirine göre önceliği vardır. Örneğin çarpma ve bölme işlemleri toplama ve çıkarmadan önce yapılır. İşlem önceliğini kendi istediğiniz sıraya göre yapmak istediğiniz sıraya göre yapmak istediğinizde işlemleri gruplamak için parantez içine alınız.

A= (B+C)(D-E)*

Burada önce parantez içindeki toplama ve çıkarma işlemleri yapılır, daha sonra çarpma işlemi yapılır. Aritmetik işlemlerde işlem önceliği aşağıdaki sıraya göredir:

( ) parantez içindeki ifadeler
** çarpma*
/ bölme
+ toplama
- çıkarma

Çarpma

PBP, 16x16 bit'lik çarpma yapar. "* " operatörü çarpma sonunda elde edilen 32 bitlik sonucun alt 16 bitini elde eder. Bu işlem çoğu programlama dillerindeki tipik çarpma işlemidir. "**" operatörü ise 32 bit'lik sonucun üst 16 bitini ifade eder. 16x16 bitlik çarpma sonucunda elde edilen 32 bitlik sonucun hem alt hem de üst 16 bitini elde etmek için bu iki operatör birlikte kullanılır.

*W1 =W0 * 1000* ' W0'ın içindeki sayıyı 1000 ile çarpar , elde edilen sonucun alt 16 bitini W1'in içine yerleştirir.'

*W2 =W0 ** 1000* ' W0'ın içindeki sayıyı 1000 ile çarpar , elde edilen sonucun üst 16 bitini W2'in içine yerleştirir.'

" */ " operatörü 32 bitlik sonucun ortasındaki 16 biti elde eder.

*W3 =W0 */ 1000* ' W1'le W0'ı çarpar, elde edilen 32bitlik sonucun ortasındaki 16 biti W3 içine İçerisine yerleştirir.'

Bölme

PBP, 16 bitlik bölme yapar. “ / ” operatörü 16 bitlik bölme sonucunu elde eder. “ // ” operatörü ise bölme sonucunda kalanı bulur. Buna bazen bir sayının modülünü de denir.

```
W1 = W0 / 1000      ‘ W0’ı 1000’e böler ve sonucu W1’e yerleştirir.’
W2 = W0 // 1000     ‘ W0’ı 1000’e böler ve kalanı W2’ye yerleştirir.’
```

Shift (Kaydırma)

“ << ” ve “ >> ” operatörü bir sayıyı sağa ve sola kaydırırlar. 16 bitlik bir saykı1-15 defa kaydırılabilir. Kaydırılmış bitlerin değeri sıfırdır.

```
B0 = B0 << 3        ‘B0 içindeki sayı 3 pozisyon sola kaydırılır ve sonuç
                      B0’ a yerleştirilir.(0,1 ve 2. bitler ” 0” değerini alır.)’
W1 = W0 >> 1        ‘W1 içindeki sayı 1 pozisyon sağa kaydırılır ve sonuç
                     W1 içerisine yerleştirilir. ( 15. bit “0” değerini alır.) ’
```

Math Operators	Description
+	Addition
-	Subtraction
*	Multiplication
**	Top 16 Bits of Multiplication
*/	Middle 16 Bits of Multiplication
/	Division
//	Remainder (Modulus)
<<	Shift Left
>>	Shift Right
ABS	Absolute Value*
ATN	Arctangent
COS	Cosine
DCD	2n Decode
DIG	Digit
DIV32	31-bit x 15-bit Divide
HYP	Hypotenuse
MAX	Maximum*
MIN	Minimum*
NCD	Encode
REV	Reverse Bits
SIN	Sine
SQR	Square Root
&	Bitwise AND
|	Bitwise OR
^	Bitwise Exclusive OR
~	Bitwise NOT
&/	Bitwise NOT AND
|/	Bitwise NOT OR
^/	Bitwise NOT Exclusive OR

PicBasic Pro Komutları

Komut	Açıklama
@	Bir satırlık assembly komutu PBP içerisine yazılır.
ASM..ENDASM	Birden fazla assembly satırı PBP içerisine yazılır.
BRANCH	Koşullu GOTO komutu (ON.. GOTO komutu benzeri)
BRANCHL	Koşullu GOTO komutu (2K'lık bellek dışına çıkıldığında)
BUTTON	Belirlenen bir pinden bir giriş almak için kullanılır.
CALL	Assembly dili alt programını çağırır.
CLEAR	Tüm değişkenlerin içeriğini sıfırlar.
COUNT	Bir pinden gelen pals sayısını sayar.
DATA	On-chip EEPROM belleğin içerisine sabit değer yazar.

DEBUG	Sabit bir pin ve baud'da asenkron seri çıkışı sağlar.
DISABLE	INTERRUPT kesmelerini geçersiz yapar.
DTMFOUT	Bir pinden telefon tuşu ton seslerini üretir.
EEPROM	On-chip EEPROM belleğin içerisine veri yazar.
ENABLE	INTERRUPT kesmelerini geçerli yapar.
END	Programı durdurup ve düşük güç moduna geçirir.
FOR..NEXT	Komut grubunu istenilen sayıda tekrar eder.
FREQOUT	Bir pinden iki farklı frekans üretir.
GOSUB	Bir PBP alt programını çağırır.
GOTO	Belirlenen bir etikete dallanmayı sağlar.
HIGH	Belirlenen bir pin çıkışını high (1) yapar.
HSERIN	Donanımı seri iletişimi destekleyen cihazlardan

	(USART) seri olarak veri almak için kullanılır.
HSEROUT	Donanımı seri iletişimi destekleyen cihazlara seri olarak veri göndermek için.
I2CREAD	Bir seri EEPROM bellekten bir byte'lık veriyi seri olarak okur.
I2CWRITE	Bir seri EEPROM belleğe bir byte'lık veriyi seri olarak yazar.
IF..THEN..ELSE..ENDIF	Bir koşula bağlı olarak farklı komut veya komut grupları çalıştırmak için kullanılır.
INPUT	Bir pini giriş yönünde yönlendirir.
{LET}	Bir ifadenin sonucunu değişkene atar. (Kullanımı isteğe bağlıdır.)
LCDOUT	Bir LCD'ye 4 bit veya 8 bit ile karakter gönderir.

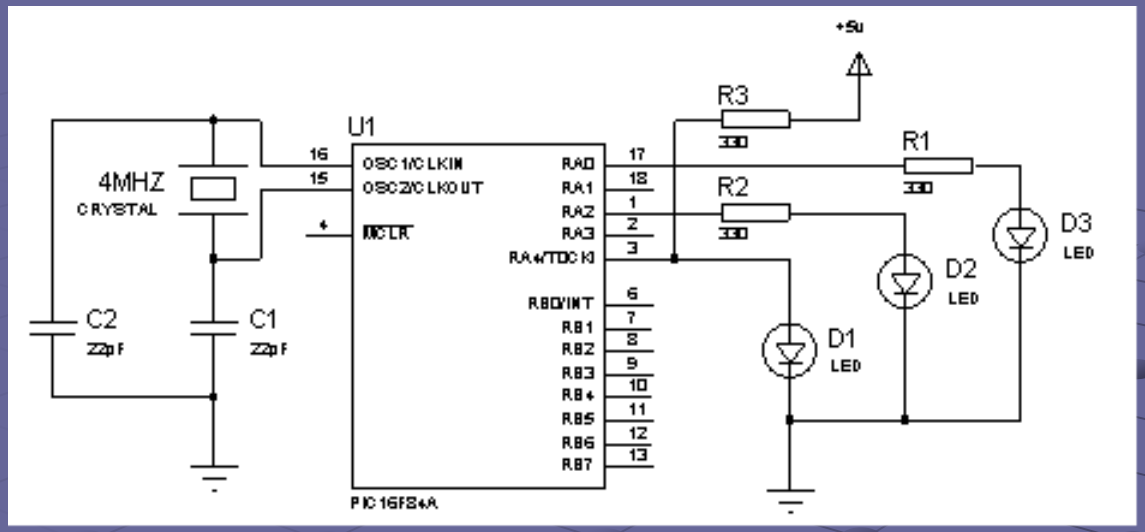
LOOKDOWN	Bir değer tablosundan sabit bir değer araştırmak için kullanılır.
LOOKDOWN2	Bir değer tablosundan sabit/değişken araştırmak için kullanılır.
LOOKUP	Bir değer tablosundan sabit değer almak için kullanılır.
LOOKUP2	Bir değer tablosundan sabit/değişken almak için kullanılır.
LOW	Bir pin çıkışını LOW(0) yapar.
NAP	İşlemciyi kısa süreyle durdurur.
ON INTERRUPT	Bir interrupt (kesme) oluştuğunda interrupt alt programın çağrılmasını sağlar.
OUTPUT	Bir pini çıkışa yönlendirir.

PAUSE	İstenilen bir zaman süresinde gecikme yapar.
PAUSEUS	İstenilen bir zaman süresinde gecikme yapar.
PEEK	Bir registerden 1 byte'lık veri okur.
POKE	Bir registere 1 byte'lık veri yazılır.
POT	Belirlenen bir pindeki potansiyometreyi okur.
PULSIN	Bir pindeki pals genişliğini ölçer.
PULSOUT	Bir pinden istenilen uzunlukta palsler üretilir.
PWM	Bir pinden PWM palsleri üretir.
RANDOM	Bir değişken içerisine 1-65535 arasında rastgele sayılar üretir.
RCTIME	Bir pindeki pals genişliğini ölçer.
READ	On-chip EEPROM bellekten 1 Byte' lık veri okur.
RESUME	INTERRUPT alt programı bittikten sonra programı kaldığı yerden devam ettirir.
RETURN	Programı en son kullanılan GOSUB komutundan sonraki komuttan itibaren devam ettirir.

REVERSE	Bir pinin giriş/çıkış yönlendirmesini tersine çevirir. Giriş olan Pinler çıkışa yönlenir.
SERIN	Asenkron seri giriş komutu (BS1 stilinde)
SERIN2	Asenkron seri giriş komutu (BS2 stilinde)
SEROUT	Asenkron seri çıkış komutu (BS1 stilinde)
SEROUT2	Asenkron seri çıkış komutu (BS2 stilinde)
SHIFTIN	Senkron seri giriş komutu
SHIFTOUT	Senkron seri çıkış komutu
SLEEP	İşlemciyi belirlenen bir süre ile düşük güç moduna sokar.
SOUND	Belirlenen bir pinden istenilen tonda ses üretir.
STOP	Programın çalışmasını durdurur.
SWAP	İki değişken içerisindeki değerleri değiştirir.
TOGGLE	Bir pini çıkışa yönlendirir ve toggle pals üretir. Yani pinin konumu "1"ise "0" yapar.
WHİLE..WEND	Bir koşul doğru (true) olduğu sürece bir grup komutu tekrarlar.
WRITE	On-chip EEPROM belleğe 1 byte'lık veri yazar.

Port Giriş-Çıkış İşlemleri

Proje 1)

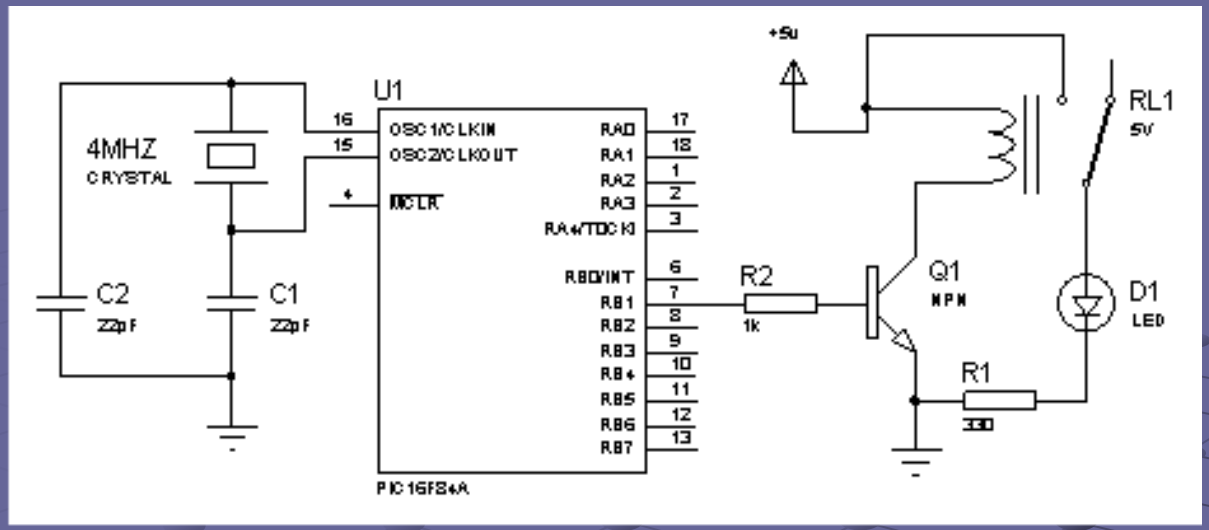


Birden fazla pine bağlı ledlerin aynı anda aktif edilmesi

TRISA=%00000
PortA=%00000
PortA.0=1
PortA.2=1
PortA.4=1
END

TRISA=%00000
PortA=0
PortA=%10101
END

Proje 2)

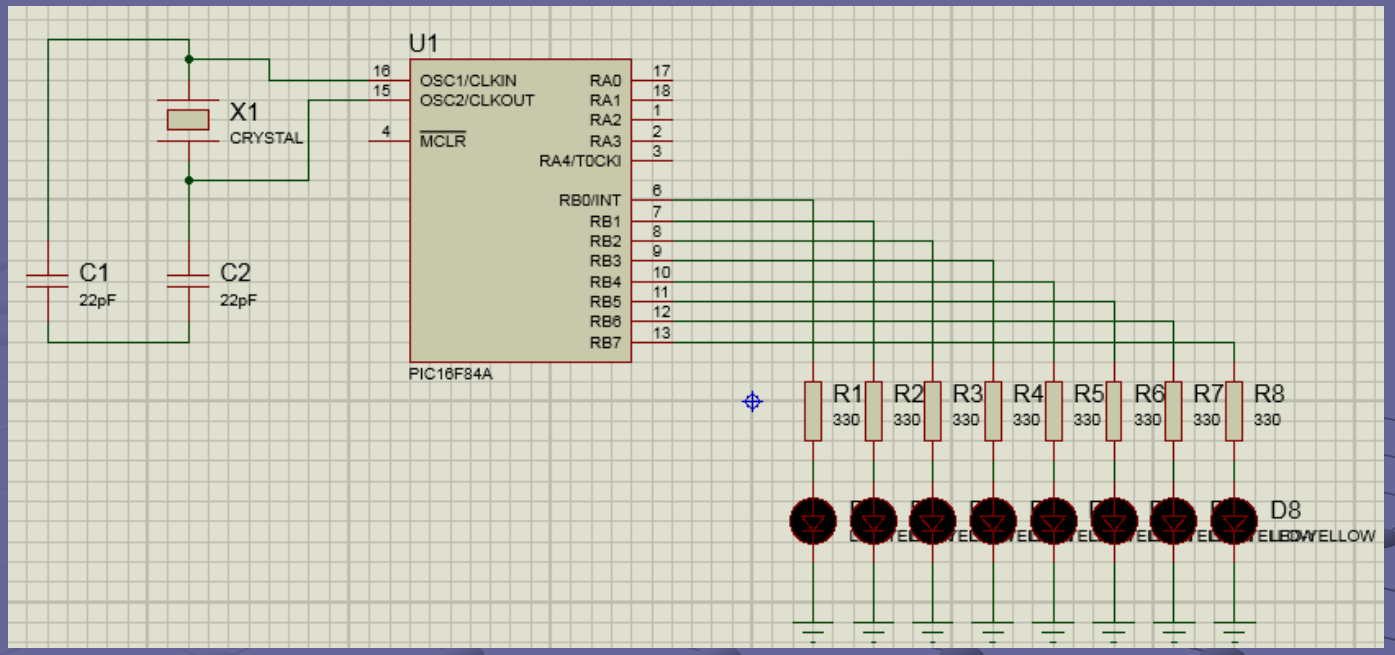


Röle'ye bağlı ledi 0,5sn aralıkla sürekli yakıp-söndüren program.

TRISB.1=0
LED VAR PORTB.1
LED=0
DONGU:
LED=1
PAUSE 500
LED=0
PAUSE 500
GOTO DONGU

TRISB=0
LED VAR PORTB.1
LED=0
DONGU:
TOGGLE LED
PAUSE 500
GOTO DONGU

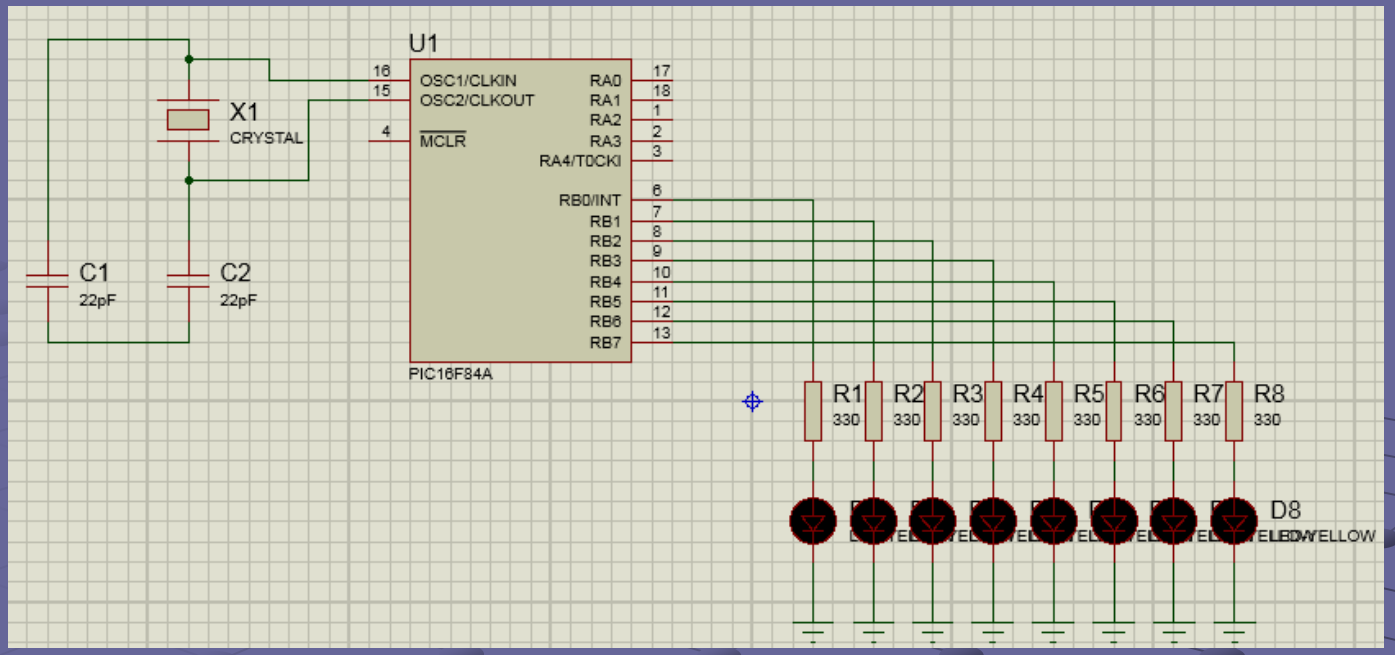
Proje 3)



PIC16F84'ün B protu üzerinde bir LED'i bit0'dan bit7'e kadar kaydıran program.

```
TRISB=0
LEDLER VAR PORTB
LEDLER=0
BASLA:
PAUSE 500
LEDLER=%00000001
SAG:
PAUSE 500
LEDLER=LEDLER << 1
IF LEDLER=%10000000 THEN BASLA
GOTO SAG
```

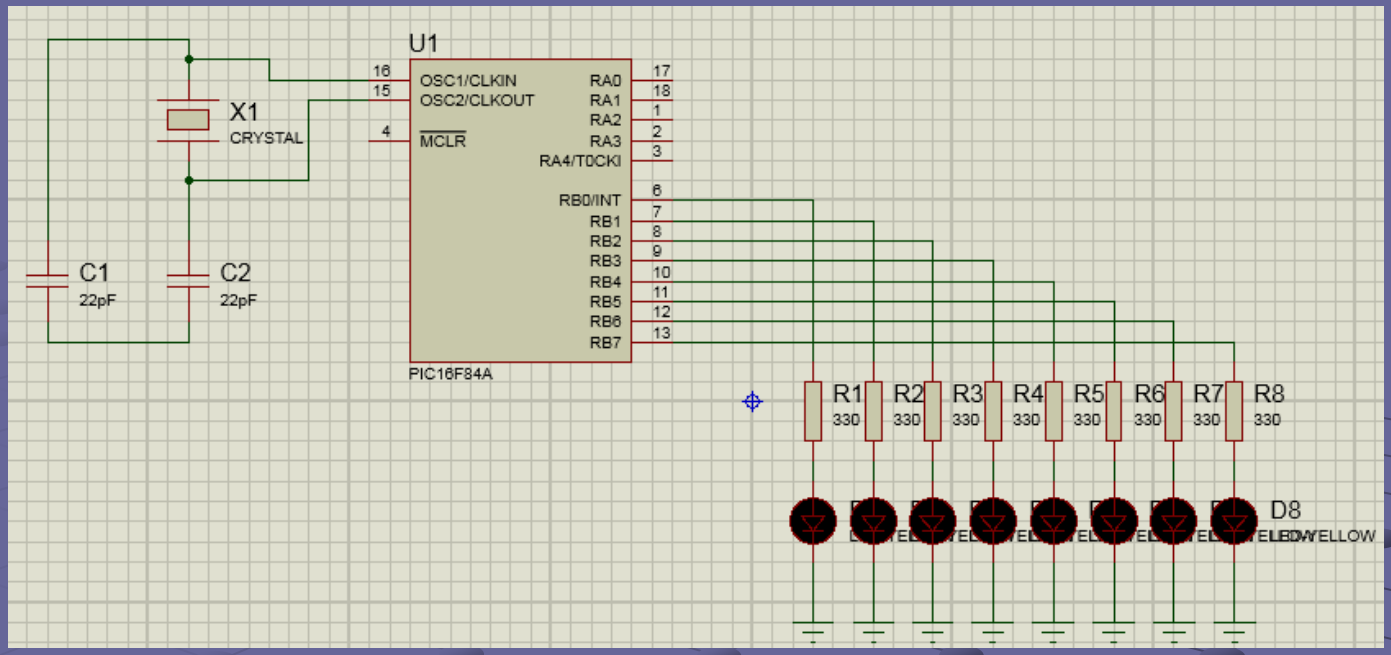
Proje 4)



PIC16F84'ün B protu üzerinde bir LED'i bit7'dan bit0'e kadar kaydıran program.

```
TRISB=0
LEDLER VAR PORTB
LEDLER=0
BASLA:
PAUSE 500
LEDLER=%10000000
SOL:
PAUSE 500
LEDLER=LEDLER >> 1
IF LEDLER=%00000001 THEN BASLA
GOTO SOL
```

Proje 5)



PIC16F84'ün B protu üzerinde bir LED'i sağa-sola kaydıran program.

```
TRISB=0
LEDLER VAR PORTB
LEDLER=0

SAG:
LEDLER=%00000001
SAGA_DEVAM:
PAUSE 500
LEDLER=LEDLER << 1
IF LEDLER=%10000000 THEN SOL
GOTO SAGA_DEVAM

SOL:
LEDLER=%10000000
SOLA_DEVAM:
PAUSE 500
LEDLER=LEDLER >> 1
IF LEDLER=%00000001 THEN SAG
GOTO SOLA_DEVAM
```