الباب الثالث CHAPTER (2)

برمجة أطراف ومداخل البيك الـميكروكونترلر

Input/Output Programming of PIC Microcontrollers

أول الأشياء التى يجب أن يتعلمها مبرمجوا الـ Microcontroller هى كيفية التعامل مع أطراف الدخل والخرج I/O pins الخاصة بالـ Microcontroller. لذلك فى هذه الجزئية سوف نتعرف على كيفية إستخدم برنامج الـ mikroC لبرمجة أطراف الدخل والخرج الخاصة بالـ Microcontroller. كما سنتعرف أيضاً على الدوائر الوسيطة Interface circuits التى يتم توصيلها بأطراف الـ Microcontroller لتشغيل الأجهزة المختلفة. ولكن قبل ذلك يجب أن نتعرف على كيفية التعامل مع أطراف الدخل والخرج بإستخدام الـ Registers. وهذا ما سوف يتم شرحه فى الجزء التالى.

3.1- المسجلان المسئولان عن التحكم بأطراف الـميكروكونترولر
TRIS and PORT registers

كما ذكرنا سابقاً ، تنقسم أطراف الدخل والخرج I/O Pins الخاصة بالـ Microcontroller إلى PORTs. وكل Pin فى الـ PORT ثنائى الإتجاه Bi-directional، وهذا يعنى أن أى Pin يمكن ضبطه كدخل Input أو كخرج Output.

عند التعامل مع أى PORT، فإننا قد نريد أن نفعل الآتى:

1. تحديد **إتجاه** Direction كل pin فى الـ PORT.
2. تحديد **قيمة الخرج** Output Value لكل pin من أطراف الـ PORT.
3. قراءة **قيمة الدخل** لأى pin على الـ PORT.

كما ذكرنا سابقاً، يوجد مجموعة من الـ Registers التى تسمى Special function registers والتى تستخدم للتحكم بالأشياء الداخلية للـ Microcontroller. هذه الـ Registers يمكن الوصول إليها والتحكم بها عن طريق البرنامج سواء بلغة الـ C أو Basic أو Assembly الذى يكتبة المستخدم والذى يتم تخزينه داخل الـ Microcontroller.

كل PORT داخل الـ Microcontroller يتحكم به Two registers وهم:

1. TRIS register.
2. PORT register.

فعلى سبيل المثال، إذا أردنا مثلاً التعامل مع أطراف الـ PORTB، فإننا نستخدم الـ PORTB register والـ TRISB register كما هو موضح فى شكل (3-1). ويستخدم الـ TRIS register بشكل عام للتحكم فى إتجاه كل Pin فى الـ PORT. فعلى سبيل المثال، إذا أردنا أن نصمم التركيبة المبينه فى شكل (3-1) على الـ Pins الخاصة بالـ PORTB فكل ما علينا فعله وضع '1' للـ pin الذى تريد أن تجعلة Input ووضع '0' للـ pin الذى تريد أن تجعله Output فى الـ TRISB register كما هو موضح فى نفس الشكل.

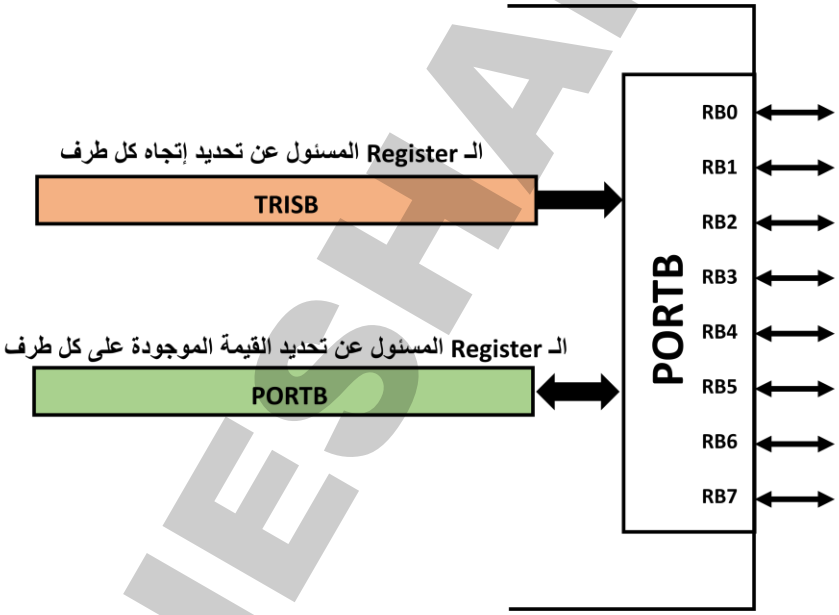

Fig. (3-1). Registers that are responsible for controlling I/O pins.

الـ Registers المسئولة عن التحكم بأطراف الدخل والخرج.

وإذا أردت أن تجعل كل الـ Pins الخاصة بالـ PORTB خرج Outputs، فإننا نجعل الـ TRISB=0 وإذا أردنا أن نجعل كل الأطراف Inputs، فإننا نجعل قيمة الـ TRISB=255 أو 0xff كما هو موضح فى شكل (3-2). ويتم تطبيق ذلك مع أى PORT أخر بنفس الطريقة.

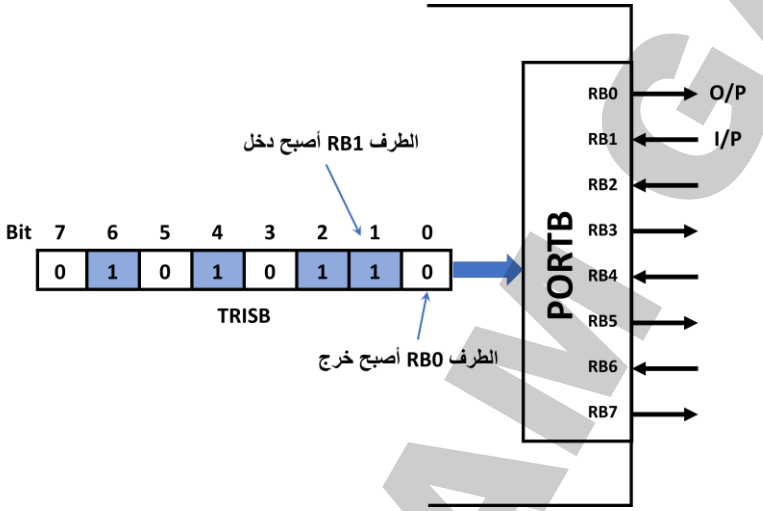

Fig. 3-2. Defining the direction of each pin in PORTB using TRISB.

تحديد إتجاه كل طرف من أطراف الـ PORTB عن طريق الـ TRISB

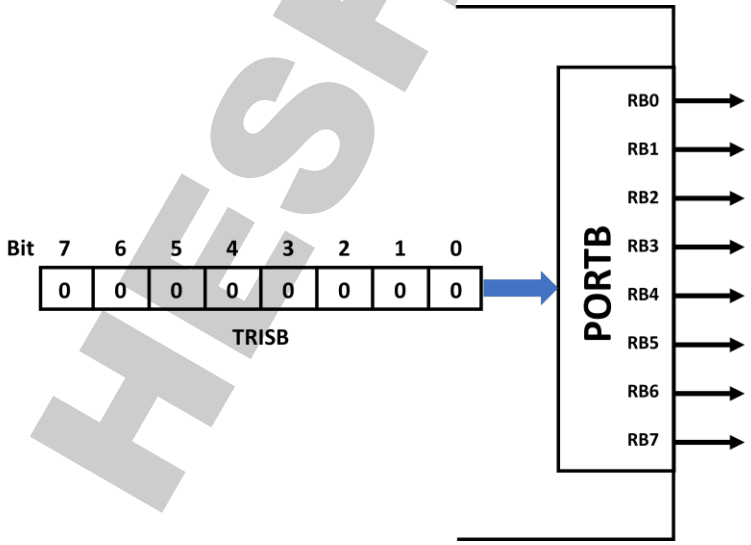

Fig. 3-3. Making all pins in PORTB as outputs.

جعل كل الأطراف فى الـ PORTB خرج.

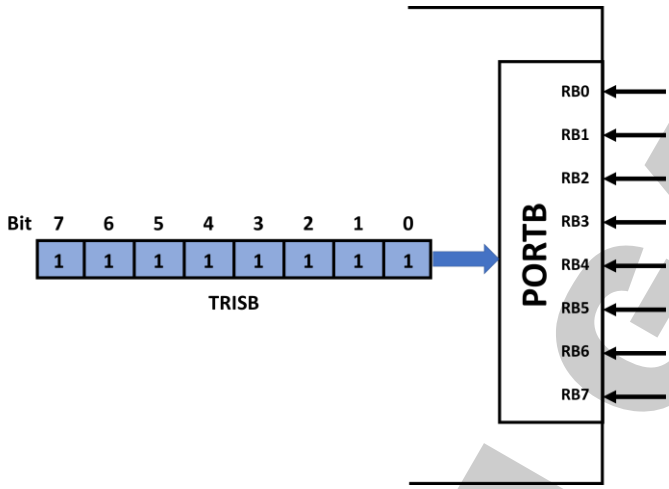

Fig. 3-4. Making all pins of PORTB inputs.

جعل كل الأطراف فى الـ PORTB دخل.

أما بالنسبة للـ PORT register، فإنه يستخدم للتحكم بالقيم Values الخاصة بكل طرف pin على الـ PORT. فعلى سبيل المثال، إذا أردنا تشغيل الـ LEDs كما هو موضح فى شكل (3-5)، فإن كل ما عليك فعله هو وضع '1' للطرف الذى تريد أن تجعله ON ووضع '0' للطرف الذى تريد أن تجعله OFF فى الـ PORTB register كما هو موضح فى نفس الشكل. وبذلك سوف تكون قيمة الـ PORTB=0x55.

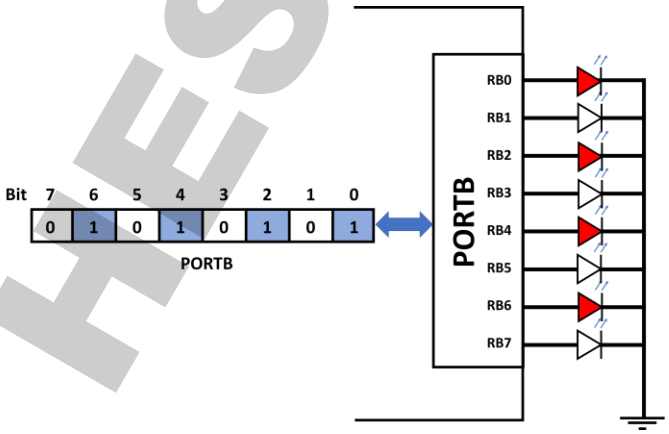

Fig. 3-5. Defining the values of PORTB pins.

تحديد القيم على أطراف الـ PORTB

3.2- دوائر الإدخال Input circuits

بعد أن تعرفنا على الـ Registers المستخدمة للتحكم بأطراف الـ microcontroller يجب أن نتعرف على الدوائر التى يجب توصيلها بهذه الأطراف، لذلك سوف نبدأ فى هذه الجزئية بكيفية توصيل المفاتيح Buttons بإستخدام دوائر الدخل Input circuits. توجد طريقتان لتوصيل الـ Button إلى أى pin خاص بالـ microcontroller كالآتى:

1. بإستخدام Pull-up resistor.
2. بإستخدام Pull-down resistor. (أكثر شيوعاً)

ولكن لماذا نحتاج إلى هذه المقاومات خاصةً عند وجود Button؟

لنقل أننا نريد توصيل مفتاح Button إلى الطرف RB0 كما هو موضح فى شكل (3-6). كما أننا نريد أن نجعل الـ Button يُعطى قيمة (5v) '1' للطرف RB0 عند الضغط عليه و (0v) '0' عند عدم الضغط عليه. فالطريقة المنطقية التى سوف تخطُر بتفكيرنا هى بتوصيل هذا الـ Button بالـ 5v والطرف الأخر بالـ RB0، بحيث أنه إذا تم الضغط Pressed على الـ Button، فإنه سوف يعمل على توصيل الـ 5V إلى الطرف RB0. ولكن تكمن المشكلة هنا أنه إذا لم يتم الضغط على الـ Button، فإن حالة State الـ Pin سوف تصبح مجهولة Floating state كما هو موضح فى شكل (3-6)، أى أنها يمكن أن تكون (5v) '1' أو (0v) '0' تبعاً للشوشرة Noise الموجودة على الطرف RB0 وهذا غير مقبول فى الدوائر الرقيمة Digital circuits.

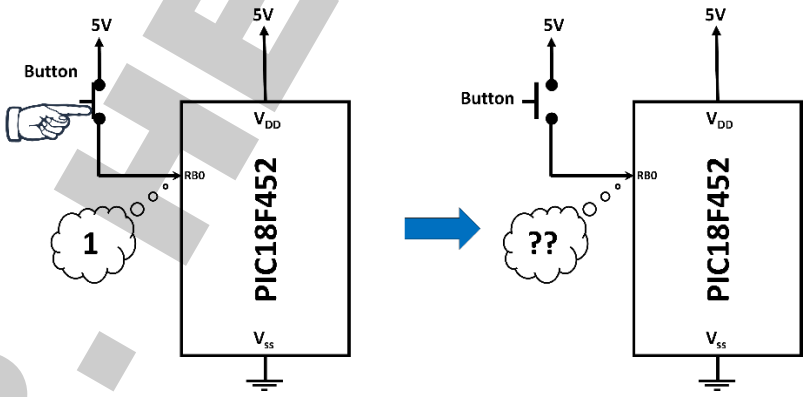

Fig. 3-6: When there is no pull-down resistor.

عندما لا توجد مقاومة على المفتاح

لذلك يمكن أن نقترح أنه لحل هذه المشكلة يتم توصيل الطرف RB0 للأرضى Gnd لكى يكون الوضع الطبيعى Normal state للطرف عند عدم الضغط على الـ Button يساوى الـ 0 كما هو موضح فى شكل (3-7). ولكن سوف يصبح لدينا مشكلة جديدة وهى أنه عند الضغط على الـ Button ، فإنه سوف يحدث Short-circuit بين الـ 5v والـ Gnd مما سيؤدى إلى تلف الـ Power supply كما هو موضح فى نفس الشكل. ولذلك يجب وضع مقاومة ولتكن قيمتها 1K وموصلة بالأرضى Gnd لكى يتم تقليل التيار المار من الـ 5V إلى الـ Gnd كما هو موضح فى شكل (3-8). وتسمى هذه الطريقة بإسم Pull-down resistor أو إستخدام المقاومة التى تسحب لأسفل. حيث أن المقاومة تعمل على جعل الحالة الطبيعية للطرف 0 وعند الضغط على المفتاح، فإن حالة الطرف سوف تصبح 1.

وتوجد طريقة أخرى وهى إستخدام الـ Pull-up resistor. حيث أنه فى هذه الطريقة يتم توصيل المقاومة بالـ 5V كما هو موضح فى شكل (3-9). حيث أنه فى هذه الطريقة تكون القيمة الطبيعية للطرف 1 وعندما يتم الضغط على الـ Button تكون قيمة الطرف 0. وفى هذا الكتاب سوف يتم إستعمال الـ Pull-down resistor نظراً لطبيعته المنطقية، حيث أنه يفضل عند عدم الضغط على الـ Button أن تكون القيمة الداخلة للـ pin تساوى 0 وعند الضغط على الـ Button فإن قيمة الـ Pin يفضل ان تكون 1. وبفضل أن تكون قيمة هذه المقاومة ما بين 1K إلى 10K. ولكن بعد إجراء تجارب كثيرةمن قبل مؤلف الكتاب وُجد أنه أفضل قيمة يمكن إستخدامها يجب أن تكون بين الـ 1K والـ 4.7K.

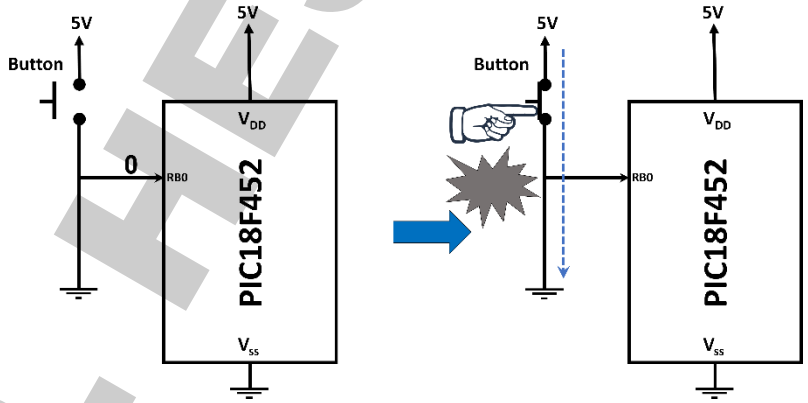

Fig. 3-7: When there is no resistor.

عندما لا توجد مقاومة على المفتاح

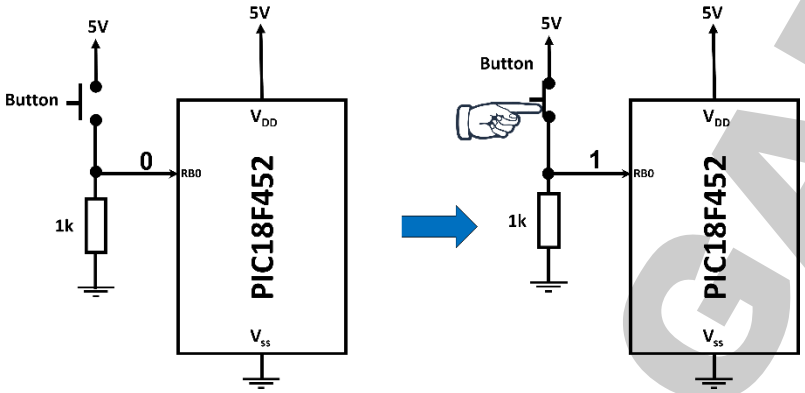

Fig. 3-8: Using pull-down resistor.

عند إستخدام مقاومة سحب إلى أسفل

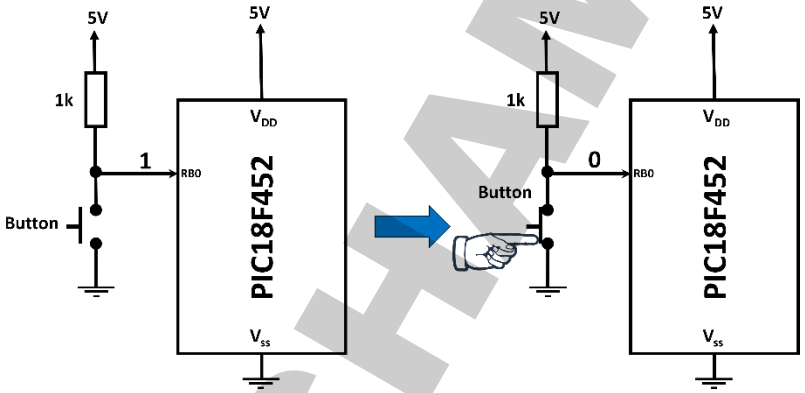

Fig. 3-9: Using Pull-Up resistor.

عند إستخدام مقاومة سحب إلى أعلى

3.2- دوائر الإخراج Output circuits

بعد أن عرفنا كيفية توصيل دوائر الدخل، يجب أن نتعرف على بعض أهم الدوائر المستخدمة فى الخرج. ومن أهم العناصر المستخدمة كخرج فى الدوائر الإلكترونية هى الـ LEDs والمعروفة بإسم (Light Emitting Diodes). ولتشغيل الـ LED، فإننا نحتاج إلى تسليط جهد محدد على أطرافة Terminals الخاصة به فى حدود من 1.7V إلى 2.3V تبعاً لنوع الـ LED. كما نحتاج أيضاً إلى تيار ثابت Constant current فى حدود 10 mA يمر بهذا الـ LED لتشغيله. وحيث أن الـ Current source أو التيار الذى يمكن توليده من أى Pin خاص بالـ PIC microcontroller يمكن أن يصل إلى 25 mA، فإن توصيل الـ LED مباشرة بأى طرف من هذه الأطراف يمكن أى يؤدى

إلى تلف الـ LED أو الطرف فوراً. لذلك يجب وجود مقاومة Limiting resistor تعمل على خفض التيار المار فى الـ LED إلى 10 mA. ويتم حساب قيمة المقاومة عن طريق المعادلة الآتية:

$$R_s = \frac{V_s - V_{led}}{I_{led}}$$

حيث أن الـ V_s هو جهد مصدر التغذية Power supply والذى سوف يعمل على تشغيل الـ LED والـ V_{led} هو الجهد المطلوب لتشغيل الـ LED أو Working voltage. كما أن الـ I_{led} هو التيار اللازم لتشغيل الـ LED. وحيث أننا سوف نوصل الـ LED بأى طرف من أطراف الـ Microcontroller، فإن الجهد الخارج من هذا الطرف والذى سوف يُغذى الـ LED سوف يساوى الـ 5v. وحيث أن جهد تشغل الـ LED يساوى الـ 1.7v والتيار المطلوب لتشغيله يساوى 10 mA، فإنه بعد التعويض فى المعادلة السابقة فإن قيمة المقاومة R_s سوف تساوى 330 ohm كما هو موضح فى شكل (3-10).

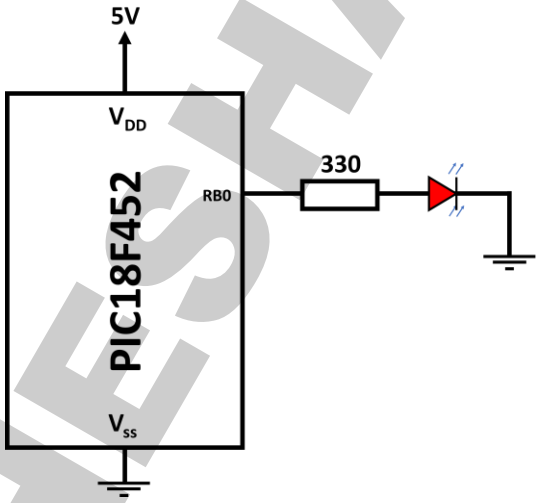

Fig. 3-10: Connecting a LED to a microcontroller.

وبما أن الـ Microcontroller يستخدم فى المقام الأول للتحكم بالأجهزة الكهربائية والمنزلية والروبوتات، فإنه لا يمكن توصيل أى Pin بشكل مباشر إلى هذه الأحمال Loads إلا عن طريق دائرة وسيطة والتى تسمى Interface circuit. والفائدة الأساسية لهذه الدائرة هى أنها تعمل على الإستفادة من الإشارة الخارجة من الـ Microcontroller لكى تصبح مناسبة لتشغيل الأحمال Loads الكبيرة مثل المواتير Motors أو لمبات الإنارة Lamps وغيرها من الأشياء التى تحتاج إلى جهود وتيارات عالية. ومن أشهر المكونات التى تستخدم فى دوائر الـ Interface هو الريلاى Relay وهو عبارة

عن مفتاح كهروميكانيكى Electromechanical switch يصبح Short circuit أو open circuit تبعاً للإشارة القادمة للملف Coil الخاص به كما هو موضح فى شكل (3-11).

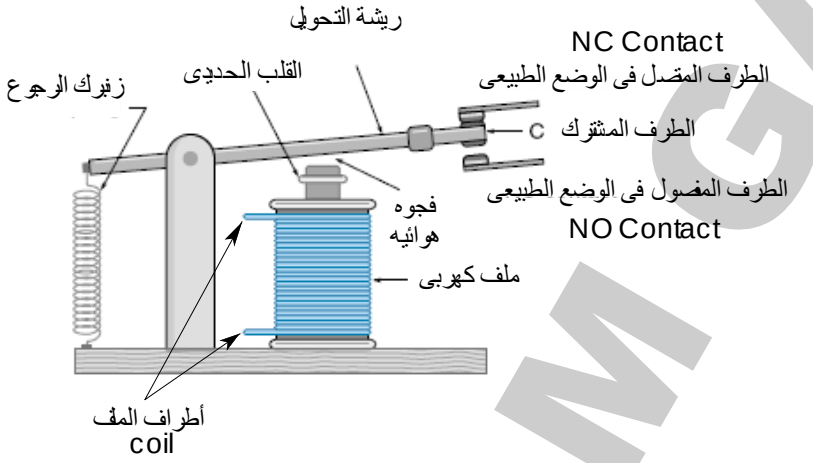

Fig. 3-11: Relay components. مكونات الريلاى

ويوجد فى السوق أنواع كثيرة من الـ Relay حيث أنه يتم شرائه تبعاً لجهد تشغيله والذى قد يكون DC أو AC. ولكن بشكل عام، من أشهر أنواع الـ Relays المستخدمة فى دوائر الـ Interface هو Relay الـ 12V أى أن الـ Coil الخاص به يحتاج إلى 12v DC لتشغيلة. لذلك سوف نستعمل هذا الـ Relay فى هذا الكتاب نظراً لرخصة وإنتشارة فى السوق.

ونظراً لأن أقصى جهد خرج لأى طرف من أطراف الـ Microcontroller فى حدود 5v، فإننا سوف نحتاج إلى ترانزيستور Transistor لكى يعمل كمفتاح إلكترونى لتشغيل الـ 12v Relay عن طريق الإشارة القادمة من الـ Microcontroller. ولهذا الـ Transistor فائدة أخرى هى أن الـ Relay قد يحتاج إلى تيارات قد تصل إلى 150 mA لكى يعمل وهذا يستحيل الحصول عليه من أى طراف من أطراف الـ Microcontroller لذلك يعمل الـ Transistor على تكبير الإشارة والتيار الخارج من الـ Microcontroller وذلك لتشغيل الـRelay كما هو موضح فى شكل (3-12). ويُلاحظ فى نفس الشكل أنه يوجد دايود Diode تم توصيله بالتوازى مع الـ Relay وذلك لمنع حدوث التيارات العكسية Reverse current وخاصة عند تشغيل وإغلاق الريلاى والذى يمكن أن يؤدى إلى تلف الـ Transistor أو الـ Microcontroller. لذلك يسمى هذا الـ Diode بإسم Reverse current protection diode. ولتشغيل ماتور Motor أو Lamp بجهد 220v فإنه يتم توصيل الريلاى بالحمل المراد تشغيله كما هو موضح فى شكل (3-13).

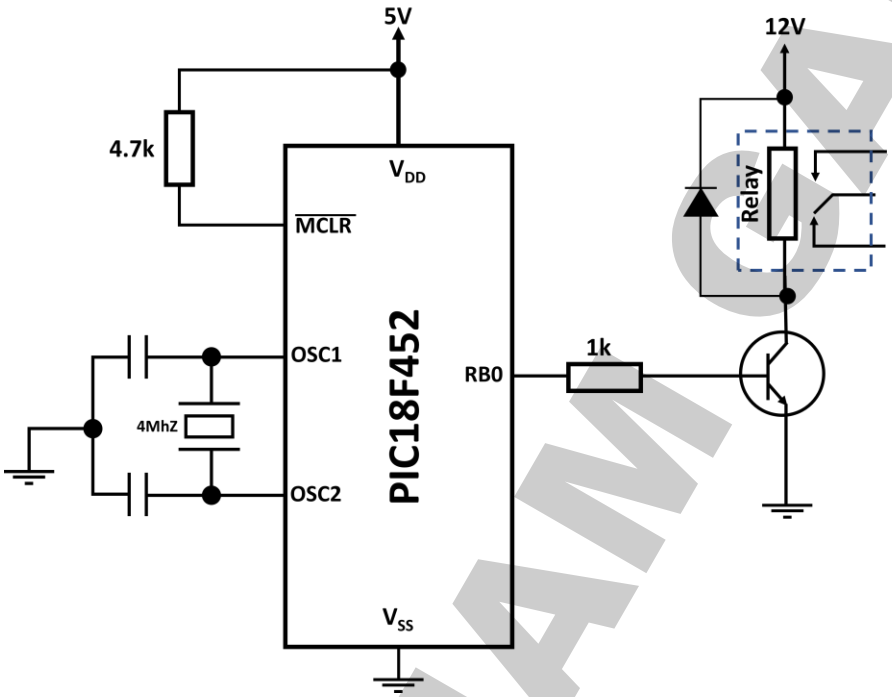

Fig. 3-12: Connecting a relay to the microcontroller.

توصيل الريلاى إلى الميكروكونترلر.

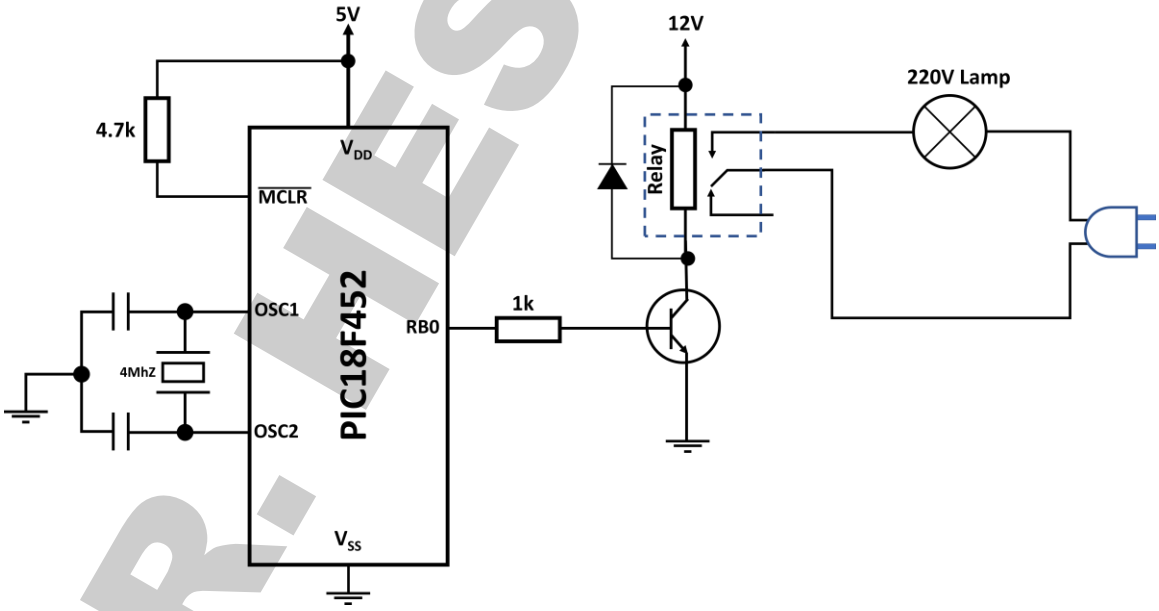

Fig. 3-13: Connecting a 220V lamp to a relay.

توصيل الريلاى إلى مصباح زو جهد 220 فولت

وفى حالة الأحمال الـ DC فقط مثل المواتير ذات التيار المستمر DC motors فإنه يفضل إستعمال الـ Transistor فقط بدلاً من الريلاى وذلك لمنع حدوث شوشرة كهرومغناطيسية Electromagnetic noise يمكنها أن تؤدى إلى تلف الـ Microcontroller نتيجه لتشغيل وإطفاء الـ Relay كما هو موضح فى شكل (3-14). كما أن الـ Transistor يتميز بسرعة تشغيله وإطفاءه على عكس الريلاى وهذه ميزه مهمة والتى سوف يتم إستخدامها لاحقاً للتحكم بسرعة المواتير الكهربائية عن طريق خاصية تسمى الـ PWM أو Pulse width modulation.

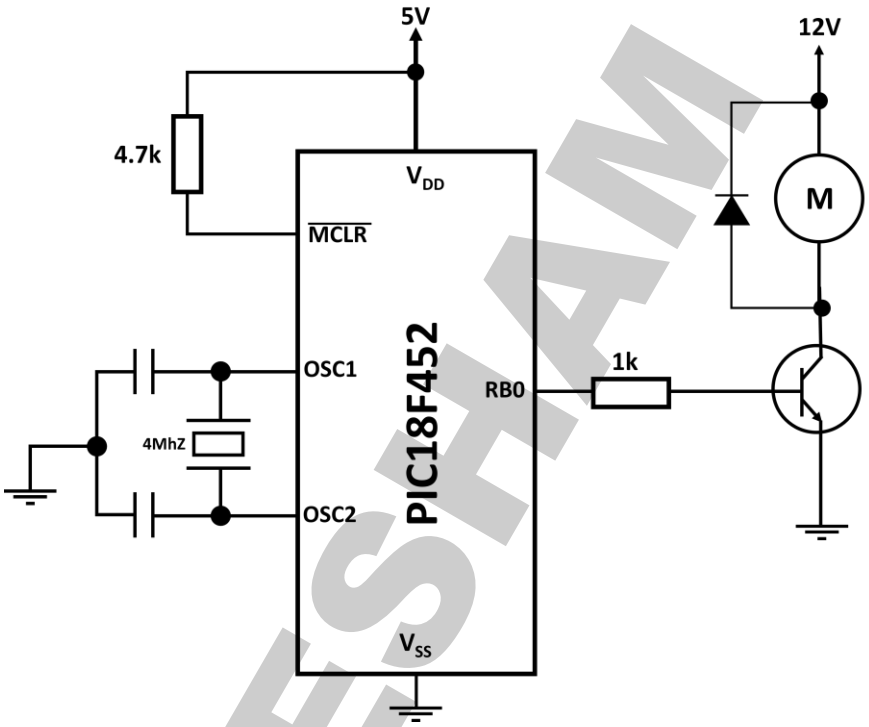

Fig. 3-14: Using a transistor to switch ON/OFF DC loads.

إستخدام الترانزستور لتشغيل وإطفاء الماتور ذو التيار المستمر.

Example (1)

Draw a circuit and write a microcontroller program by using PIC18F452 to turn ON eight LEDs connected to PORTB.

Solution:

فى هذا المثال يريد أن يشغل ثمانية ليدات تم توصيلهم بالـ PORTB.

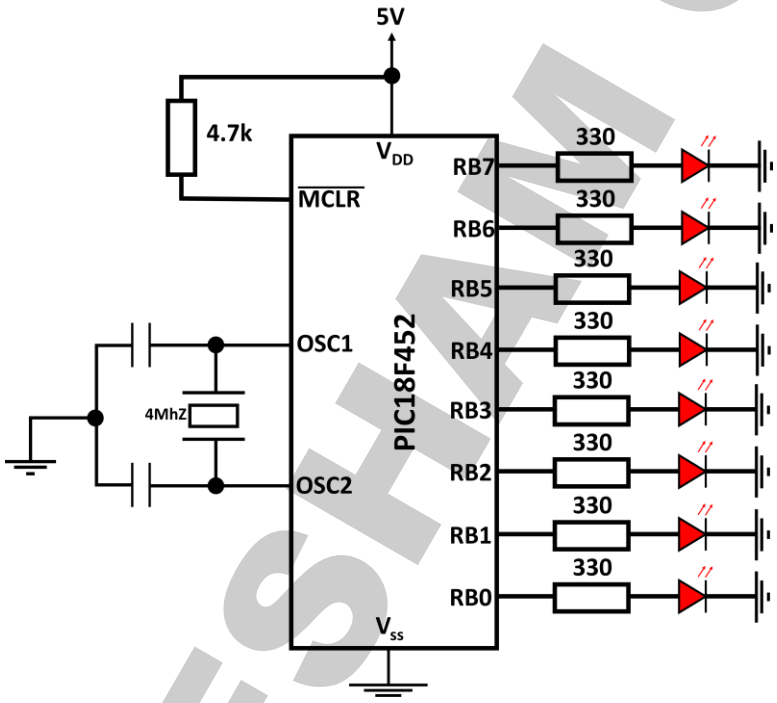

Fig. 3-15.

Program:

```
void main ()
{
 TRISB=0x00;                          //جعل كل أطراف البورت خرج
 PORTB=0;                             // تصفير الخرج فى البداية
 while (1)
      {
      PORTB=0xFF;                     //تشغيل الثمانية ليدات الموصلين بالبورت
      }}
```

Example (2)

Draw a circuit and write a microcontroller program by using PIC18F452 to turn ON eight LEDs connected to PORTB in the following fashion 01010101.

Solution:

فى هذا المثال يريد أن يشغل ثمانية ليدات تم توصيلهم بالـ PORTB بشكل محدد و هو 01010101

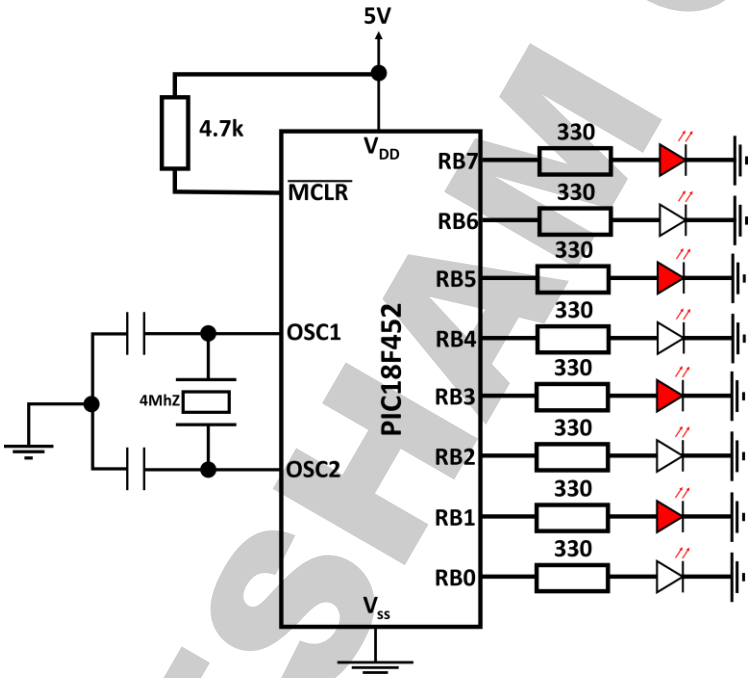

Fig. 3-16.

Program:

```
void main ()
{
 TRISB=0x00;                    //جعل كل أطراف البورت خرج
 PORTB=0;                       // تصفير الخرج فى البداية
 while (1)
      {
      PORTB=0xAA;               //تشغيل الثمانية ليدات الموصلين بالبورت
      }
}
```

Example (3)

Draw a circuit and write a microcontroller program by using PIC18F452 to flash eight LEDs connected to PORTB by turning them ON for 1 second and then turn them OFF for 1 second.

Solution:

فى هذا المثال يريد أن يشغل ثمانية ليدات تم توصيلهم بالـ PORTB ولكن لمدة 1 ثانية ثم إطفائهم لمدة 1 ثاينة.

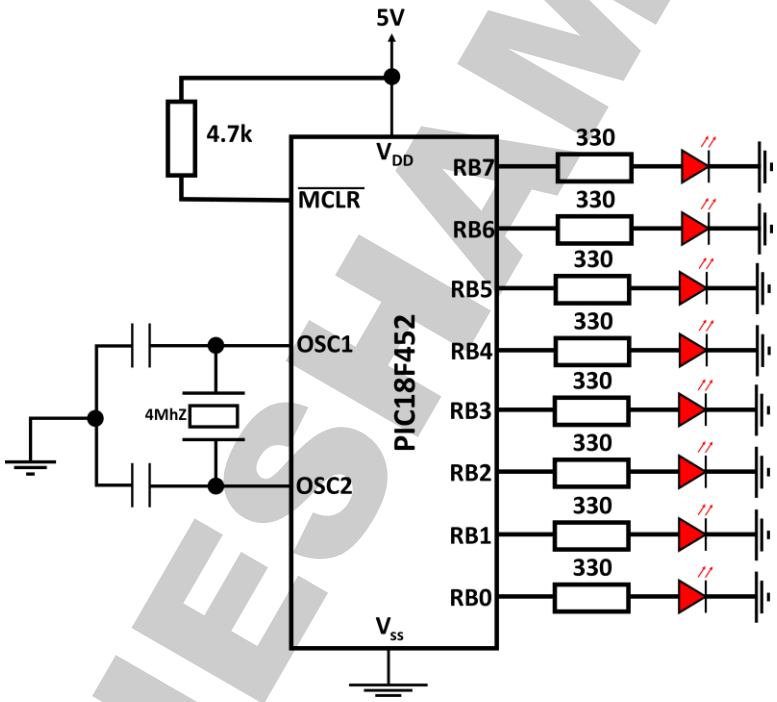

Fig. 3-17.

Program:

```
void main ()
{
 TRISB=0x00;                    //جعل كل أطراف البورت خرج
 PORTB=0;                       // تصفير الخرج فى البداية
 while (1)
      {
```

```
        PORTB=0xFF;          //تشغيل الثمانية ليدات الموصلين بالبورت
        delay_ms(1000);      //(توقف البرنامج لمدة 1000 ميللى ثانية (1 ثانيه
        PORTB=0;             //إطفاء الثمانية ليدات الموصلين بالبورت
        delay_ms(1000);      //(توقف البرنامج لمدة 1000 ميللى ثانية (1 ثانيه
        }
}
```

Example (4)

Draw a circuit and write a microcontroller program by using PIC18F452 to turn ON a LED connected to pin RB0 by pressing a button connected to RC0 pin. If the button is not pressed then turn OFF the LED.

Solution:

فى هذا المثال يريد أن يشغل LED واصل بالطرف RB0 عند الضغط على مفتاح متصل بالطرف RC0 بحيث يعمل الليد عند الضغط على المفتاح وينطفئ فور ترك المفتاح.

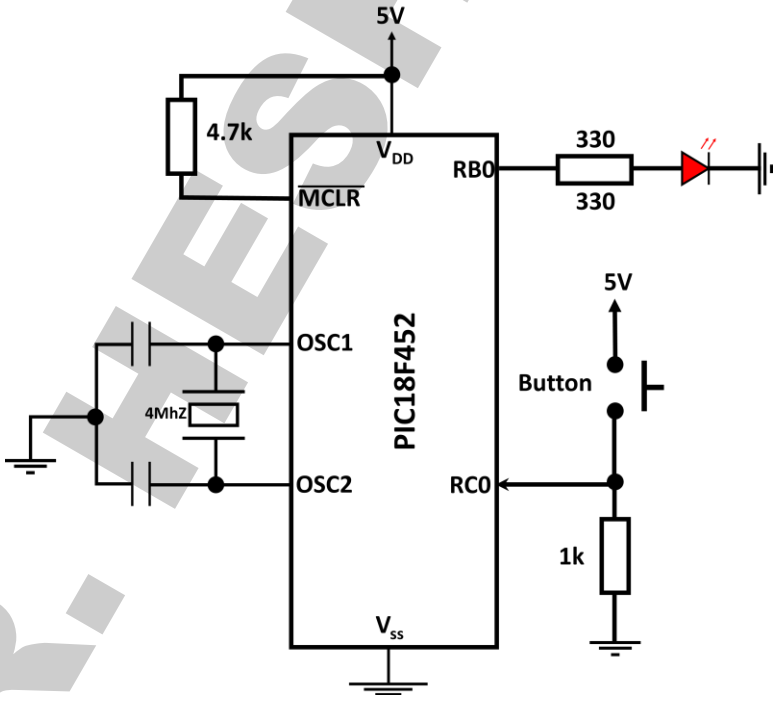

Fig. 3-18.

Program:

```
void main ()
{
 TRISB.b0=0;                    //جعل كل أطراف البورت خرج
 PORTB.b0=0;                    // تصفير الخرج فى البداية
 TRISC.b0=1;                    // جعل اول طرف فى البورت سى دخل
while (1)
        {
        if(PORTC.b0==1)
        PORTB.b0=1;
        else
        PORTB.b0=0;
        }
}
```

Example (5)

Draw a circuit and write a microcontroller program by using PIC18F452 to flash a LED connected to pin RB0 (0.5 sec ON and 0.5 sec OFF) as long as a button connected to RC0 pin is pressed. If the button is not pressed then turn OFF the flashing process.

Solution:

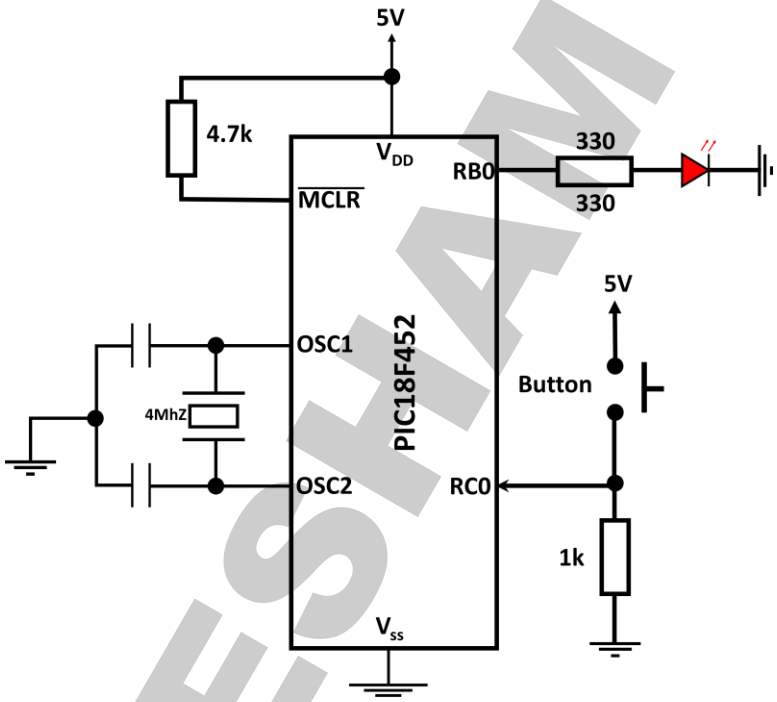

Fig. 3-19.

Program:

```
void main ()
{
 TRISB.b0=0;                    //جعل كل الطرف 0 فى البورت خرج
 PORTB.b0=0;                    // تصفير الخرج فى البداية
 TRISC.b0=1;                    // جعل اول طرف فى البورت سى دخل
while (1)
     {
```

```
        if(PORTC.b0==1)
        {
        PORTB.b0=1;
        delay_ms(500);
        PORTB.b0=0;
        delay_ms(500);
        }
        else
        PORTB.b0=0;
        }
}
```

Example (6)

Draw a circuit and write a microcontroller program by using PIC18F452 to count from 0 to 255 in binary form by using 8 LEDs connected to PORTB pins. The time between each count is 0.1 second.

Solution:

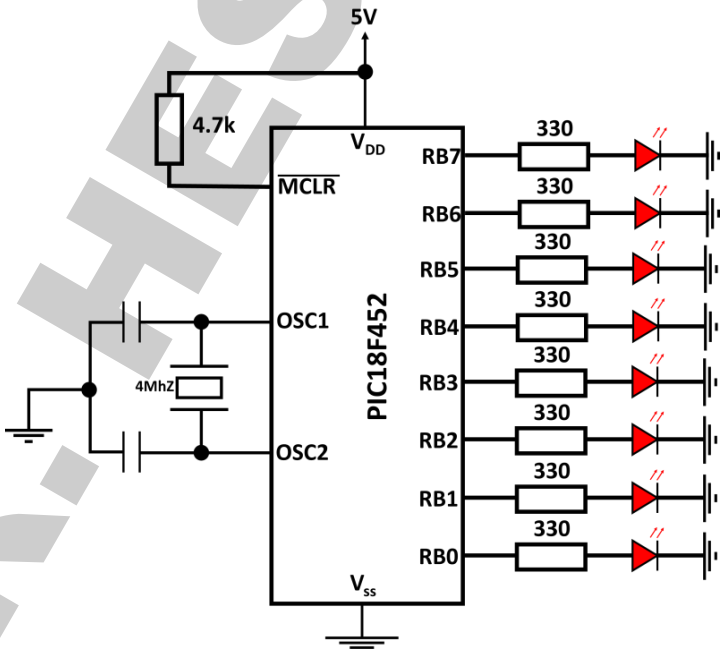

Fig. 3-20.

Program:

```
void main ()
{
char i;                          فتح مخزن للعداد//
 TRISB =0;                       جعل كل أطراف البورت خرج//
 PORTB =0;                       تصفير الخرج فى البداية //
 while (1)
        {
          for(i=0;i<=255;i++)    عداد من 0 إلى 255//
          {
                PORTB=i;                ضع قيمة العداد على البورت//
                delay_ms(500);          أنتظر لمدة 0.5 ثانية بين كل عدة//
          }
        }
}
```

Example (7)

Draw a circuit and write a microcontroller program by using PIC18F452 to display the numbers from 0 to 9 on a 7-segments display connected to PORTB pins. The time between each count is 0.5 second and the 7-segemsts type is common Anode.

Solution:

فى هذا المثال يريد أن يعرض الأرقام من 0 إلى 9 عن طريق شاشة رقمية 7-segmants بحيث يكون الفرق الزمنى بين كل عدة والعدة الأخرى نصف ثانية. المشكلة هنا أن الـ 7-segments يحتاج إلى سبعة أطراف على الأقل من الـ Microcontrollers مما سيؤدى إلى إضاعة أطراف كثيرة خاصة بالدخل والخرج كما أن ذلك سوف يؤدى إلى تعقيد البرنامج. لذلك يمكن إستخدام الشريحة 7447 حيث أن هذه الشريحة تعمل على تحويل القيمة الـ BCD إلى 7-segments بشكل مباشر. وحيث أن الـ BCD يحتاج فقط إلى 4 أطراف فبذلك تم توفير أطرف أخرى لكى يتم إستخدامها فى تطبيقات أخرى. لذلك سوف يتم توصيل الشريحة 7447 إلى أطراف الـ PORTB كما هو موضح فى الشكل التالى.

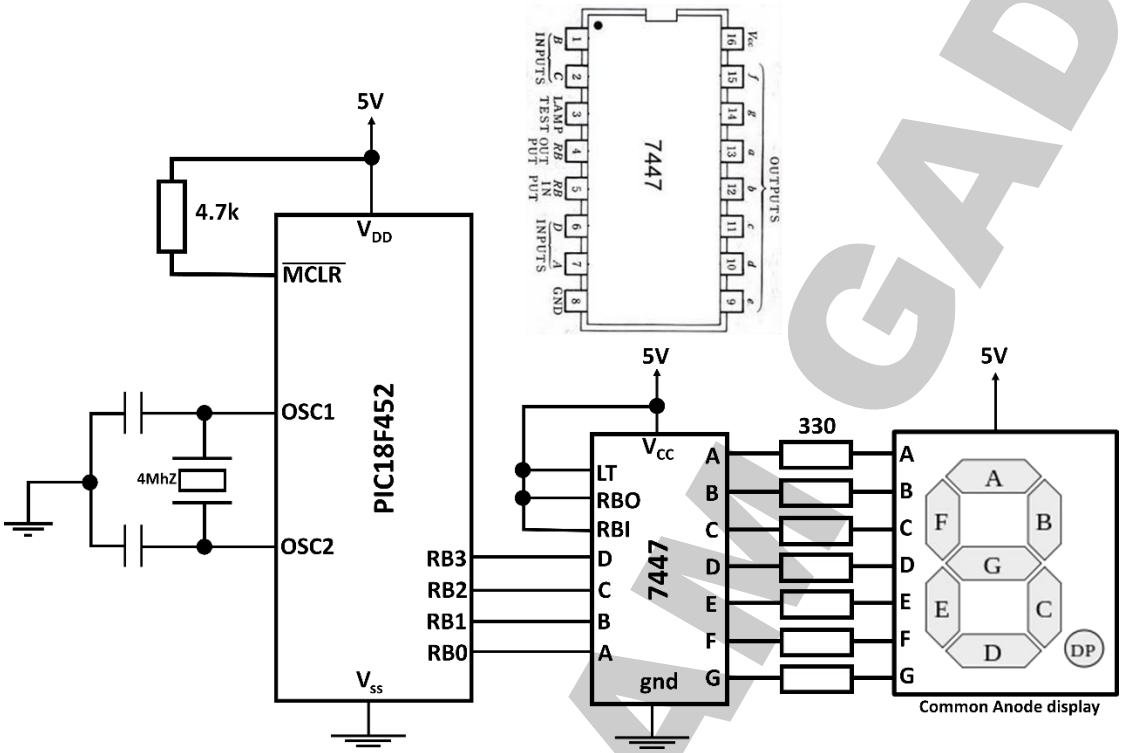

Fig. 3-21.

Program:

```
void main ()
{
char i;                          فتح مخزن للعداد//
 TRISB =0;                              جعل كل أطراف البورت خرج//
 PORTB =0;                        تصفير الخرج فى البداية  //
 while (1)
        {
          for(i=0;i<=9;i++)          عداد من 0 إلى 255//
          {
              PORTB=i;                    ضع قيمة العداد على البورت//
              delay_ms(500);              أنتظر لمدة 0.5 ثانية بين كل عدة//
          }
        }
}
```

Example (8)

Draw a circuit and write a microcontroller program by using PIC18F452 to display the numbers from 00 to 99 on two 7-segments displays connected to PORTB pins. The time between each count is 0.5 second and the 7-segemsts types are common Anode.

Solution:

فى هذا المثال يريد أن يعرض الأرقام من 00 إلى 99 عن طريق شاشة رقمية 7-segmants بحيث يكون الفرق الزمنى بين كل عدة والعدة الأخرى نصف ثانية. لذلك سوف يتم تنفيذ نفس الدائرة السابقة التى فى Example 7 مع توصيل شريحة 7447 أخرى لعمل الرقم الخاص بالعشرات ولكن من الأطراف RB4 إلى RB7 كما هو موضح فى الشكل (3-22).

وهناك طريقتان لعرض الرقيمين: الأولى وهى بإستخدام عدادان أحدهما للأحاد والأخر للعشرات. والأخرى بإستخدام عدادان أحدهما للأحاد والأخر للعشرات. ولكن بشكل عام فى كلا الطريقتين سوف تقابلنا مشكلة، وهى كيفية وضع الرقمين معاً (الأحاد والعشرات) على نفس الـ PORT وهو الـ PORTB. ولحل هذه المشكلة يتم ضرب الرقم الخاص بالعشرات فى 16 وذلك لإزاحته أربعة خانات، ثم يتم جمعه على رقم الأحاد كما هو موضح فى شكل (3-23). فعلى سبيل المثال، إذا كان رقم الأحاد j=2 ورقم العشرات i=4، فإن شكل الرقم الخارج من الـ PORTB سوف يكون كما هو موضح فى نفس الشكل.

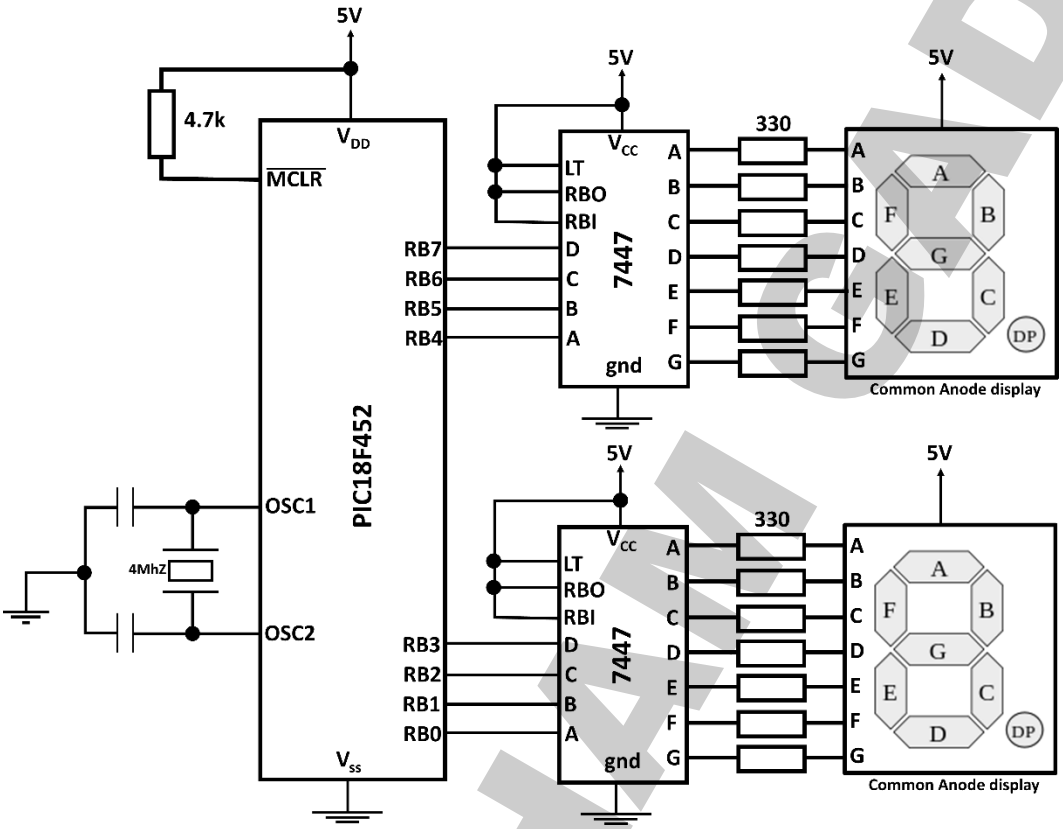

Fig. 3-22.

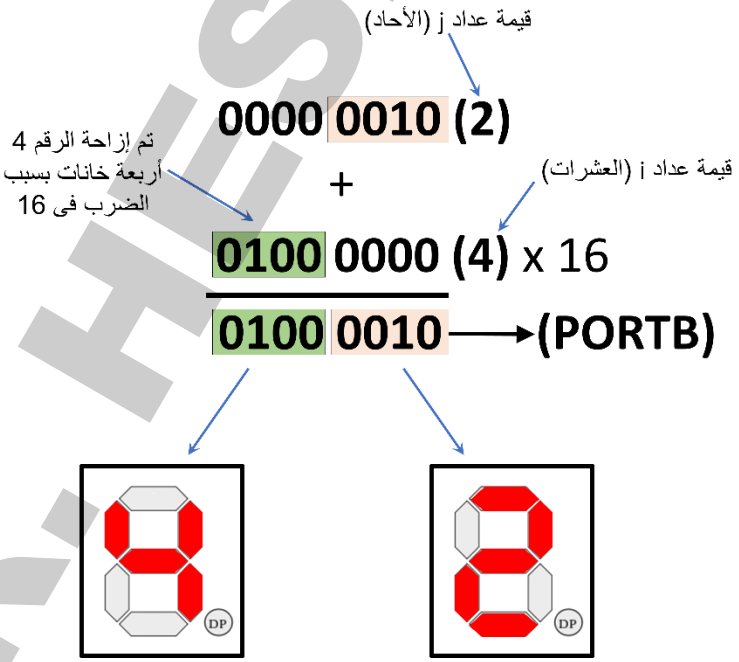

Fig. 3-23.

Program 1 (By using two counters):

```
void main ()
{
char x,y;                                   فتح مخزنين للعداداين (الأحاد والعشرات)//
 TRISB =0;                                  جعل كل أطراف البورت خرج//
 PORTB =0;                          // تصفير الخرج فى البداية
 while (1)
        {
          for(x=0;x<=9;x++)          عداد العشرات من 0 إلى 9 (الأبطأ)//
          {
                for(y=0;y<=9;y++)    عداد الأحاد من 0 إلى 9 (الأسرع)//
                {
                PORTB=x*16+y;               ضع قيمة المخزنين على البورت//
                delay_ms(500);              أنتظر لمدة 0.5 ثانية بين كل عدة//
                }
        }
}
```

Program 1 (By using one counters):

```
void main ()
{char i,x,y;                        فتح ثلاثة مخازن (العداد الرئيسى والأحاد والعشرات)//
 TRISB =0;                          جعل كل أطراف البورت خرج//
 PORTB =0;                  // تصفير الخرج فى البداية
 while (1)
        {
          for(i=0;i<=99;i++)        العداد سوف يعد من 0 إلى 99//
          {x=i/10;                  فصل قيمة العشرات عن العداد//
                y=i-x*10;                   فصل قيمة الأحاد عن العداد//
                PORTB=x*16+y;               ضع قيمة المخزنين على البورت//
                delay_ms(500);              أنتظر لمدة 0.5 ثانية بين كل عدة//
           }
        } }
```

Example (9)

Draw a circuit and write a microcontroller program by using PIC18F452 to control an AC fan that has three speeds. Each speed is selected using a SPEED button connected to pin RC0. Also, a STOP button is used to turn OFF the fan, which is connected to pin RC1. Each speed has specific wire from the fan as shown in Fig. (3-24). To set the speed, a 220V supply must be connected to the required speed terminal.

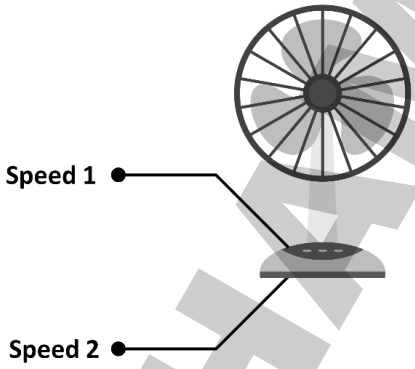

Fig. 3-24.

Solution:

فى هذا المثال يريد أن يتحكم فى سرعة مروجة كهربائية ذات ثلاثة سرعات عن طريق مفتاحين. المفتاح الأول يعمل على تغيير السرعة. والمفتاح الثانى يعمل على إطفاء المروحة. يتم تغيير سرعة المروحة عن طريق توصيل جهد 220 فولت إلى الطرف الخاص بسرعة المروحة كما هو موضح فى شكل (). بما أن المروحة تعمل بجهد 220 فولت، فإننا سوف نحتاج إلى ثلاثة Relays كل واحدةً مسئولةً عن سرعةُ محددةُ كما هو موضح فى شكل (3-25). ويتم التبديل بين الـ Relays عن طريق البرنامج بإستخدام المفتاح الموصل بالطرف RC0 مع العلم أنه يجب أن يعمل Relay واحد فقط. وإذا أردنا توقيف المروحة فكل ما علينا فعله هو إطفاء الـ Relays تماماً.

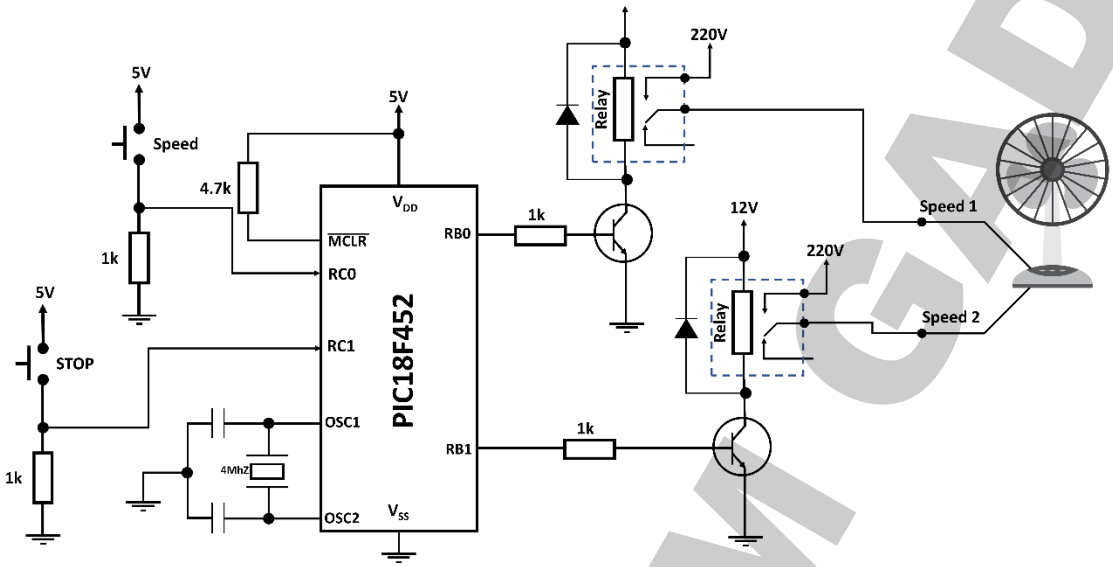

Fig. 3-25.

Program:

```
void main ()
{
TRISB =0;                    //جعل كل أطراف البورت خرج
 PORTB =0;                   // تصفير كل الخروج فى البداية
TRISC.B0=1;
TRISC.B1=1;
 while (1)
        {
        if(PORTC.B0==1)
        {
                if(PORTB==0)
                PORTB=1;
                else if(PORTB==1)
                PORTB=2;
                else
                PORTB=0;
                while(PORTC.B0==1);
        }
```

جعل الأطراف RC0 و RC1 دخول

1. تتحقق من مفتاح Speed الموصل بالطرفRC0
2. إذا تم الضغط على هذا المفتاح، يتم التحقق من قيمة الـ PORTB الموصل به الـ Relays
3. إذا كانت قيمة الـ PORTB=0 فهذا يعنى أنه لا يوجد أى ريلاى يعمل لذلك يتم تشغيل أول ريلاى عن طريق جعل قيمة الـ PORTB=1
4. وإذا كانت قيمة الـ PORTB=1 فهذا يعنى أن أول ريلاى يعمل ولذلك يجب رفع السرعة عن طريق الأنتقال إلى الريلاى الثانى وذلك عن طريق PORTB=2
5. وإذا كانت فيمة الـ PORTB أى قيمة أخرى يتم جعلها تساوى الصفر

السطر السطر يعنى أنه لن يتم الإنتقال الى إلى باقى البرنامج إلا إذا رفع المستخدم يده عن المفتاح الذى تم الضغط عليه

```
        if(PORTC.B1==1)
        {
                PORTB=0;
                while(PORTC.B1==1);
        }
        }
}
```

إذا تم الضغط على مفتاح الـ STOP الموصل بالطرف RC0 يتم تصفير الـ PORTB لإطفاء أى ريلاى