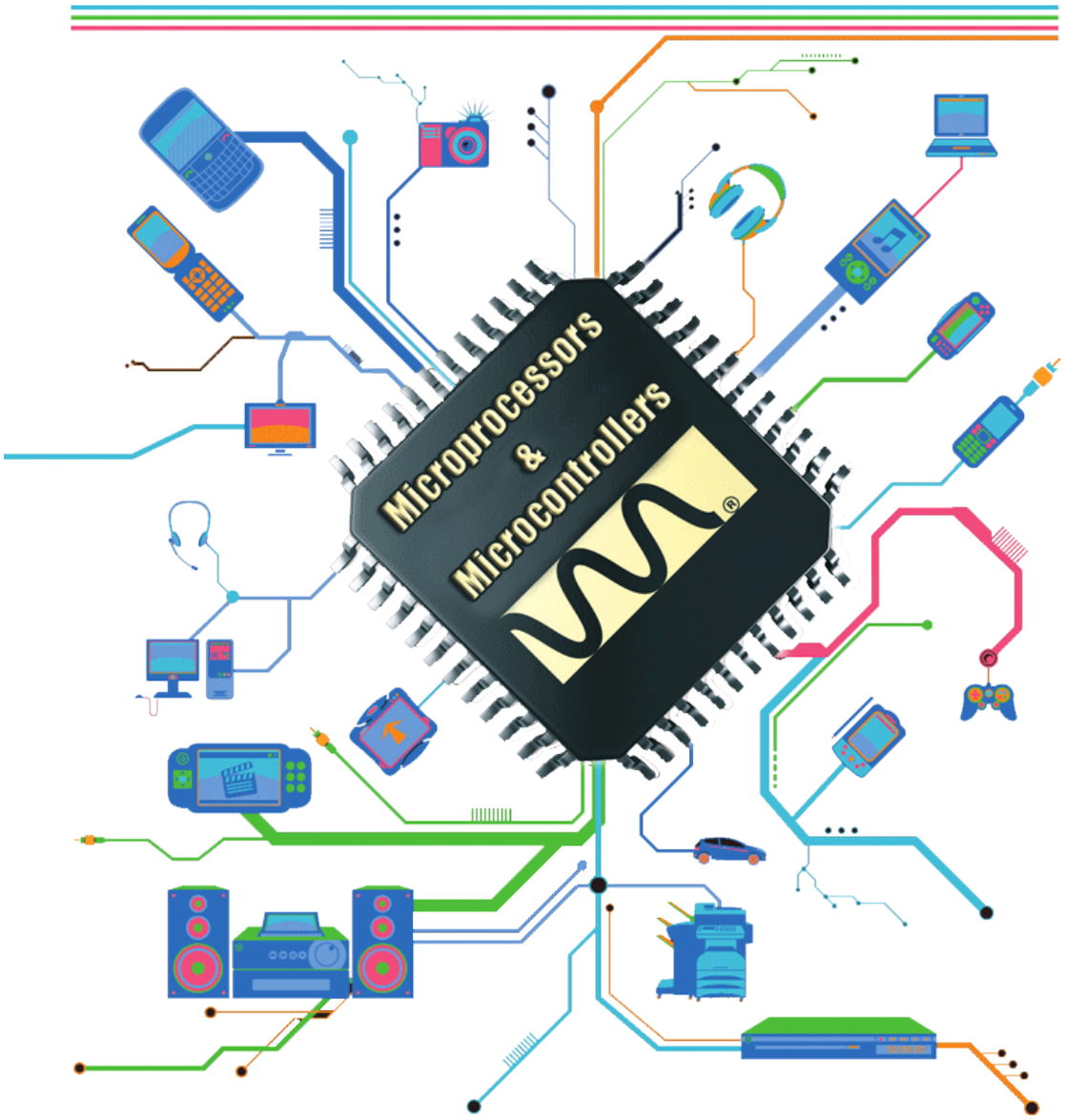
جامعـــــــــــة حلـــــــــــب
كلية الهندسة الكهربائية والإلكترونية
قسـم هندسـة التحكـم والأتمتـة

﴿الجلسات العملية لمادة المعالجات والمتحكمات المصغرة﴾

Microprocessors & Microcontrollers Lab Sessions

السنة الثالثة | قسم اتصالات

﴿الجلسة العملية الرابعة﴾



م. وليد بليد

Copyright © 2012 Walid Balid - All rights reserved.

Wednesday, March 28, 2012

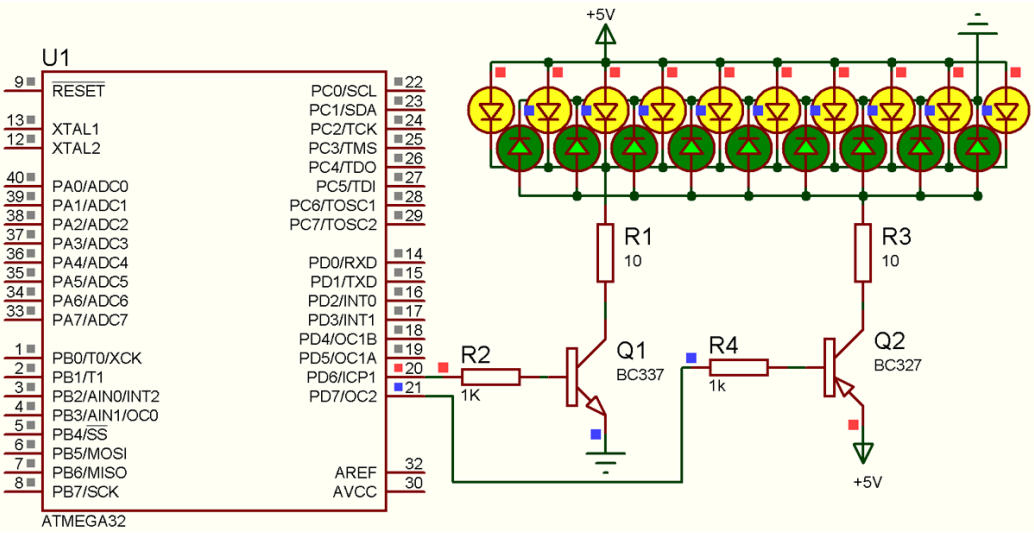
الجلسة العملية الرابعة

نظرة عامة (Overview):

هذه المحاضرة تقدم تطبيقاً عملياً لاستثمار أقطاب الدخل والخرج لمتحكمات AVR وبرمجتها في البيئة BASCOM-AVR ومحاكاتها في البيئة Proteus. حيث تقدم مجموعة من التجارب العملية التي تتضمن توصيل الثنائيات الضوئية وتصميم المفاتيح الترانزستورية لربط أقطاب المتحكم مع الأحمال. وكذلك ربط الواصلات الميكانيكية ولمفاتيح اللحظية.

1-4 برمجة بوابات الدخل والخرج في متحكمات AVR (Programming AVR MCUs GPIOs):

التجربة الرابعة: تم وصل عشرة ثنائيات ضوئية (LEDs) إلى متحكم ATmega32A على القطب PIND.6 ("1")، وعشرة أخرى إلى القطب PIND.7 ("0")، والمطلوب: كتابة برنامج خفقان لكلا المجموعتين معاً كل 0.5S وتصميم دارة المفتاح الترانزستوري.



الشكل4-1 توصيل الثنائيات مع المتحكم عن طريق مفاتيح تحكم ترانزستورية للتجربة 4

أولاً: إن الترانزستور الذي قمنا باختياره BC337 له المواصفات التالية:

I_{Cmax}=800mA, $V_{BE\_saturate}$=0.65V, $V_{CE\_saturate}$=0.2V, h_{FE} = 100, V_{CE_max}=50V

ثانياً: نحسب تيار الحمل (IC) الكلي علماً أن: V_{LED} = 2.2V & I_{LED} = 10mA:

$$I_C = 10 \times 10mA = 100mA$$

ثالثاً: نحسب مقاومة تحديد التيار RC واستطاعتها:

$$Rc = \frac{V_{cc} - V_{LED}}{I_C} = \frac{5 - 2.2}{100} = 28\Omega$$

$$P_{RC} = (V_{cc} - V_{LED}) \times I_C = (5 - 2.2) \times 100 = 280mW$$

رابعاً: نحسب قيمة التيار الأصغري اللازم لقيادة الترانزستور :

$$I_C = hfe \times I_B \rightarrow I_B = \frac{I_C}{hfe} = \frac{100}{100} = 1.0mA$$

$$Pc_{max} = U_{CE} \times I_C = 0.2 \times 100 = 20mW$$

خامساً: الآن يمكن حساب قيمة مقاومة القاعدة واستطاعتها من العلاقة التالية:

$$R_B = \frac{V_P - V_{BE}}{I_B} = \frac{5 - 0.7}{1} = 4.3K\Omega$$

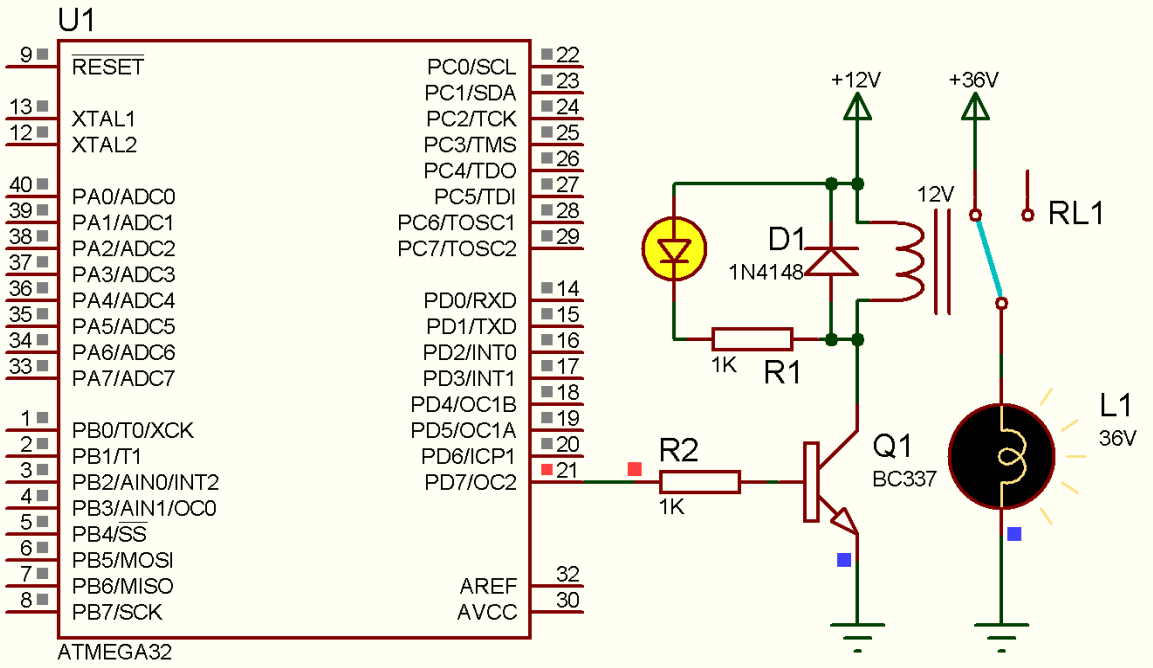
$$P_{RC} = (V_P - V_{BE}) \times I_B = (5 - 0.7) \times 1 = 4.3mW$$

البرنامج Exp.04.bas في بيئة BASCOM-AVR:

```
' ********************************************************************************
' * Title          : Exp.04.bas                                                  *
' * Target MCU     : ATMega32A                                                   *
' * Author         : Walid Balid                                                 *
' * IDE            : BASCOM AVR 2.0.7.3                                          *
' * Peripherals    : LEDs                                                        *
' * Description    : GPIOs as Outputs                                            *
' ********************************************************************************
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'-----------------------[Definitions]
$regfile = "m32def.dat"
$crystal = 8000000
'-----------------------
'-----------------------[GPIO Configurations]
Config Pind.6 = Output
Leds_y Alias Portd.6 : Reset Leds_y

Config Pind.7 = Output
Leds_g Alias Portd.7 : Set Leds_g
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'--->[Main Program]
Do
   '>[Turn Leds on]
   Set Leds_y : Set Leds_g : Waitms 500
   '>[Turn Leds off]
   Reset Leds_y : Reset Leds_g : Waitms 500
Loop
End
'---<[End Main]
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

التجربة الخامسة: التحكم بمصباح كهربائي 36V تياره 5A عن طريق متحكم ATmega32A على القطب PIND.7، وكتابة برنامج بحيث يخفق المصباح كل 1Sec علماً أن جهد تغذية ملف الريليه هو 5V وتيار تشغيل ملف هو 100mA.



الشكل4-2 توصيل الريليه مع المتحكم عن طريق مفتاح تحكم ترانزستوري للتجربة 5

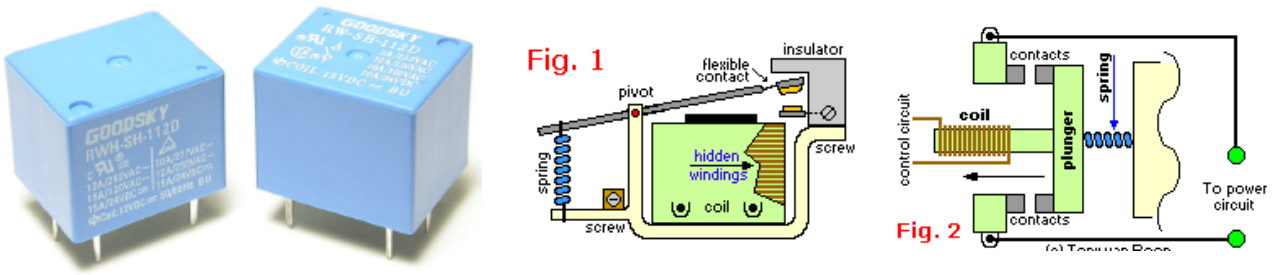
البرنامج Exp.05.bas في بيئة BASCOM-AVR:

```
' ****************************************************************************
' * Title          : Exp.05.bas                                              *
' * Target MCU     : ATMega32A                                               *
' * Author         : Walid Balid                                             *
' * IDE            : BASCOM AVR 2.0.7.3                                      *
' * Peripherals    : LEDs                                                    *
' * Description    : GPIOs as Outputs                                        *
' ****************************************************************************
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'-----------------------[Definitions]
$regfile = "m32def.dat"
$crystal = 8000000
'-----------------------
'-----------------------[GPIO Configurations]
Config Pind.7 = Output : Relay Alias Portd.7
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'--->[Main Program]
Do
   Toggle Relay : Waitms 1000
Loop
End
'---<[End Main]
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

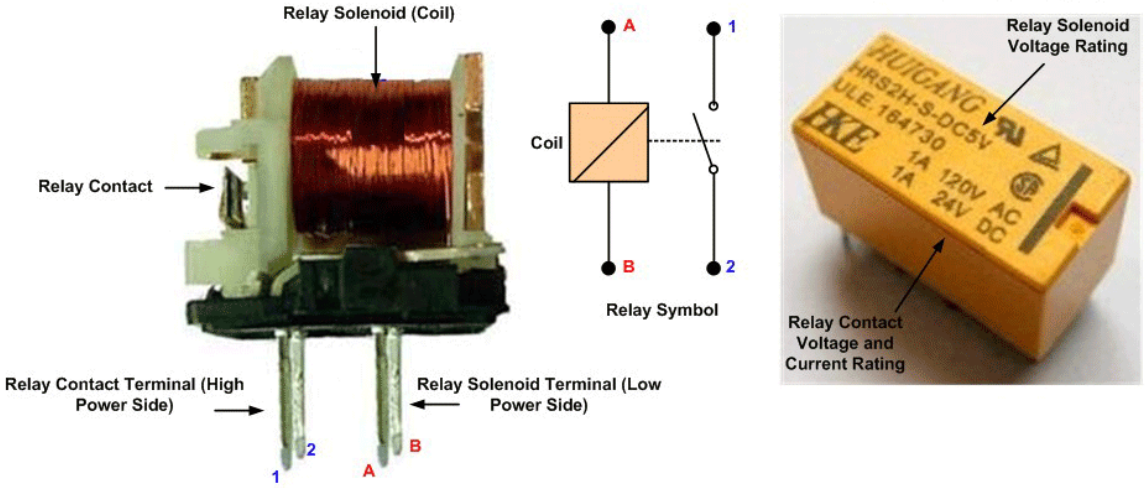
2-4 التحكم بالريليه باستخدام مفتاح ترانزستوري ثنائي القطبية (Driving Relays using BJT Control Switch):

تستخدم الريليه أو الوصل الميكانيكي (Relay) للتحكم بأحمال التيار المستمر والمتناوب التي لا تتطلب تحكماً سريعاً بالوصل والفصل – أي أن الريليه تستخدم كقاطع ميكانيكي متحكم به كهربائياً. تتوفر الريليه تجاري بجهود تحكم ذات مجال واسع نسبياً ومن هذه الجهود نذكر: 3V, 5V, 6V, 9V, 12V, 15V, 24V, 36V, 48V, 60V. كما أن التيار الذي يستجره ملف تشغيل الريليه يتراوح من 30mA ~ 300mA وذلك حسب حجم واستطاعة الريليه ويتناسب مع ذلك تناسباً طردياً.

تتألف الريليه من ملف (Coil) وهو مسؤول عن فصل ووصل الريليه (التحكم)، ومن تماسات استطاعية لقيادة الحمل واستطاعتها تختلف من ريليه لأخرى، ولكن معظم الريليهات المستخدمة في الدارات الإلكترونية تكون تماساتها قادرة على قيادة حمل بتيار من 3~10A عند جهد تشغيل الشبكة 220V. يبين الشكل4-3 والشكل4-4 رسماً تفصيلياً للبنية الداخلية للريليه حيث أنه عندما يتم تغذية ملف الريليه فإن الزراع الذي يحمل التماس المتحرك سوف ينجذب ويلامس التماس الثابت مؤدياً إلى وصل الدارة، وعندما يفقد الملف تهييجه تؤثر قوة النابض العكسية على الذراع وتعيده إلى وضعيته الأساسية.



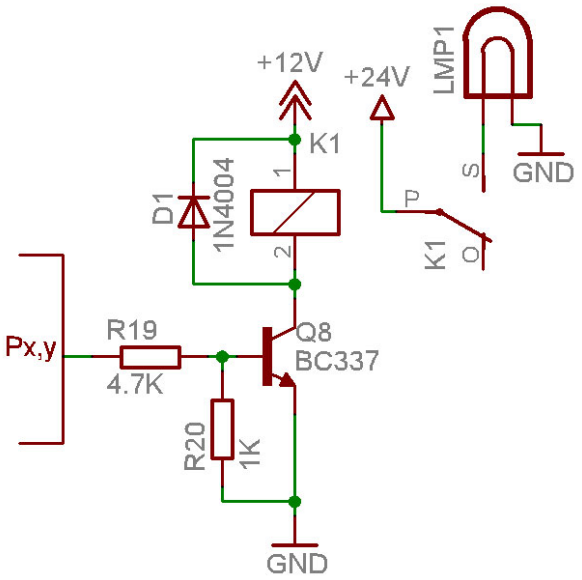
الشكل4-3 رسم تمثيلي للبنية الداخلية للريليه



الشكل4-4 رسم تفصيلي للبنية الداخلية للريليه

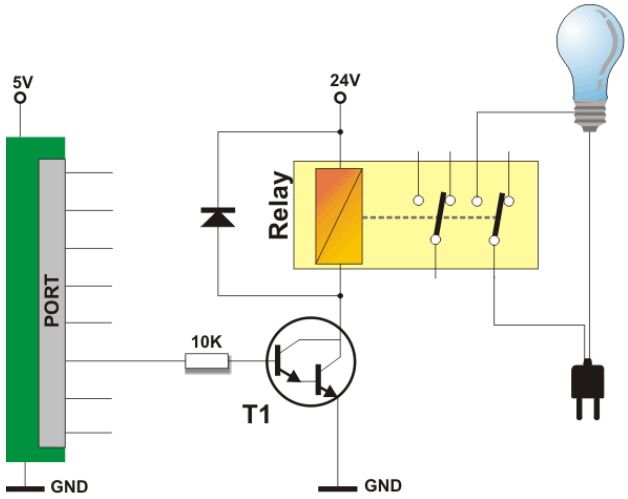
يبين الشكل4-5 دارة تحكم بريليه يعمل ملفها على جهد 12V وتتحكم بدورها بعمل مصباح تيار مستمر جهد تشغيله 24V، حيث أنه عند تطبيق "1" على قطب بوابة المتحكم فإن الترانزستور سوف يغلق مؤدياً إلى وصل النقطة الأرضية إلى الطرف الثاني من ملف الريليه، فيتهيج الملف مؤدياً بدوره إلى جذب تماس الريليه K1 وإغلاق النقطتين P, S، وعندها يضيء المصباح الكهربائي.

ملاحظة: من أجل حماية الترانزستور من أن يتم تدميره (حرقه) بسبب تيار التفريغ العكسي (Electromotive Force) لملف الريليه عند فصل الترانزستور، يتم إضافة ديود على التوازي مع ملف الريليه (D1) يسمى ديود المسار الحر والذي بدوره يشكل حلقة مغلقة لتفريغ تيار الملف عند قطع الترانزستور.



الشكل4-5 التحكم بحمل باستخدام ريليه

يبين الشكل4-6 دارة عملية لقيادة مصباح كهربائي متناوب ذو جهد 220V وتيار 1A عن طريق ريليه متحكم بها من متحكم مصغر.



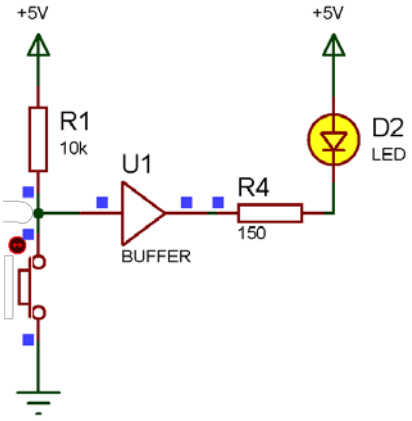
والشكل4-6 قيادة مصباح كهربائي متناوب عن طريق ريليه متحكم بها من متحكم مصغر

3-4 وصل المفاتيح اللحظية مع المتحكم المصغر (Interfacing Switches with MCUs):

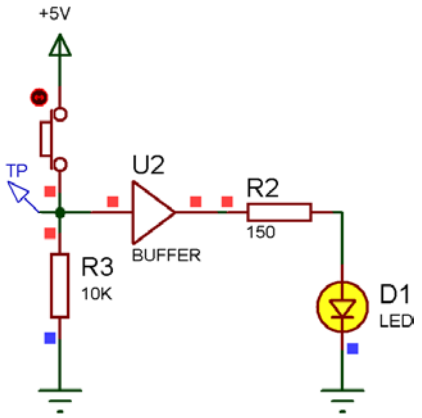
يوجد طريقتين لوصل المفاتيح مع أقطاب المتحكم المصغر:

1) المفتاح يطبق على قطب المتحكم القيمة المنطقية "0" عند ضغطه – الشكل4-7.
2) المفتاح يطبق على قطب المتحكم القيمة المنطقية "1" عند ضغطه – الشكل4-8.

على الشكل4-7 وعند ضغط المفتاح يتم توصيل النقطة الأرضية إلى قطب المتحكم، اما عند تحرير المفتاح فيتم تطبيق التغذية 5V+ على مدخل قطب المتحكم. وبالعكس تماماً تكون الحالة في طريقة التوصيل المبينة على الشكل4-8. وفي كلا الحالتين يقوم المتحكم في برنامجه الرئيسي بفحص حالة التغير على القطب المتصل مع المفتاح.

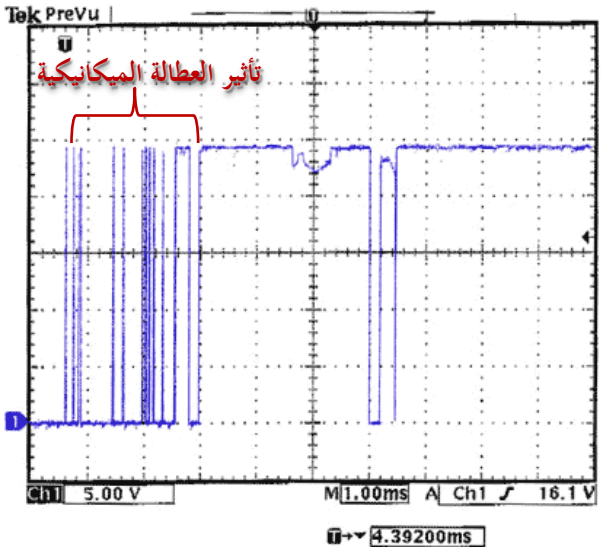


الشكل4-7 توصيل المفتاح ليكون فعال عند "0"



الشكل4-8 توصيل المفتاح ليكون فعال عند "1"

إن المفاتيح الميكانيكية لها تأثير سلبي عند ضغطها وتحريرها يسمى بالعطالة الميكانيكية للمفتاح والتي بدورها تسبب نشوء تغيرات سريعة في الإشارة على قطب المتحكم، هذه التغيرات ناتجة عن الاهتزاز الميكانيكي للمفتاح قبل أن تستقر الإشارة على الحالة المنطقية الحقيقية كما هو مبين على الشكل4-9.

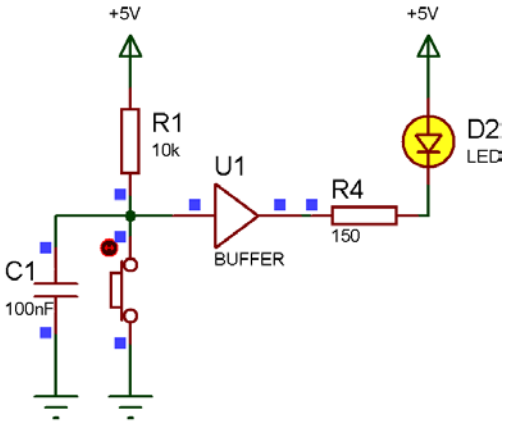


الشكل4-9 أثر العطالة الميكانيكية للمفتاح عند ضغطه وتحريره

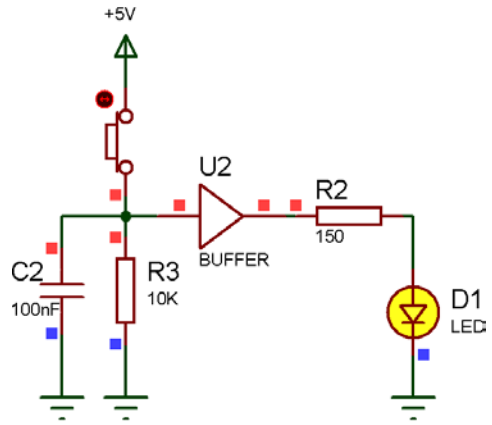
بشكل عام يوجد طريقتين للتخلص من العطالة الميكانيكية للمفاتيح وهما:

1) استخدام مكثف بقيمة تتراوح من 100nF – 1uF على التوازي مع المفتاح والذي سيقوم بدوره على تأخير التذبذب الناشئ كما هو مبين على الشكل4-10 والشكل4-11.

2) معالجة هذه الحالة برمجياً في برنامج المتحكم بفحص حالة المفتاح، وعند تحقق الشرط يتم توليد تأخير زمني 25~100ms وبعدها يتم فحص الحالة من جديد، فإذا بقيت الحالة مستقرة على الشرط المطلوب فيتم التنفيذ.



الشكل4-10 التخلص من الاهتزاز الميكانيكي للمفتاح ("0")



الشكل4-11 التخلص من الاهتزاز الميكانيكي للمفتاح ("1")

4-4 مقاومات الرفع والسحب (Pull-up & Pull-down Resistors):

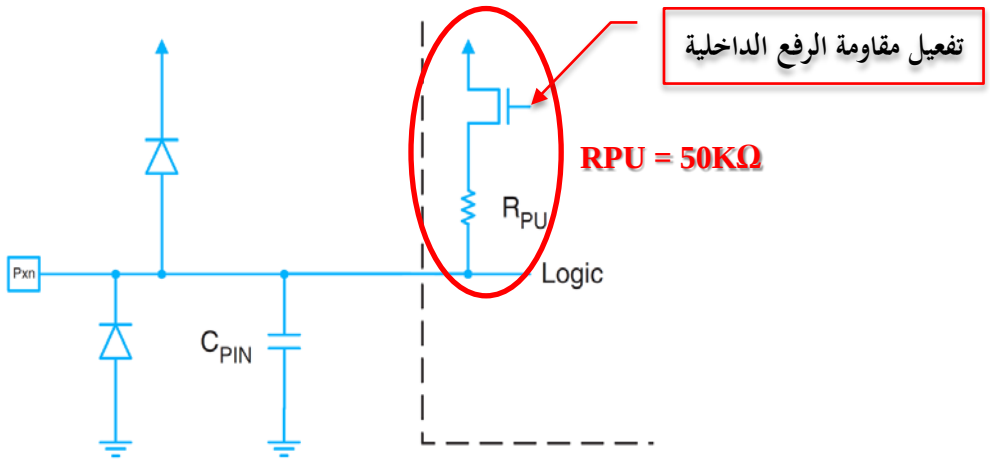
بالعودة إلى الشكل4-7 أو الشكل4-10 والذي تم فيهما توصيل المفتاح ليكون فعالاً عند "0" – أي بالضغط على المفتاح سيتم تطبيق صفر منطقي على قطب الدخل – فإن المقاومة R1 تمثل مقاومة رفع وظيفتها تأمين القيمة المنطقية "1" (+5V) على مدخل القطب عندما يكون المفتاح غير مضغوط، وبدونها ستكون الحالة على قطب الدخل غير معرّفة.

بالعودة إلى الشكل4-8 أو الشكل4-11 والذي تم فيهما توصيل المفتاح ليكون فعالاً عند "1" – أي بالضغط على المفتاح سيتم تطبيق واحد منطقي على قطب الدخل – فإن المقاومة R3 تمثل مقاومة سحب وظيفتها تأمين القيمة المنطقية "0" (GND) على مدخل القطب عندما يكون المفتاح غير مضغوط، وبدونها ستكون الحالة على قطب الدخل غير معرّفة.

قيمة مقاومات الرفع أو السبح تتراوح عادة بين القيمة 10KΩ ~ 50KΩ.

هل يمكن الاستغناء عن مقاومة الرفع الخارجية في الشكل4-7 والشكل4-10 عند وصل المفتاح إلى قطب متحكم AVR؟؟

تمتلك أقطاب متحكمات AVR عند استخدامها كأقطاب دخل مقاومات رفع داخلية قيمتها R_{PU} = 50KΩ كما هو مبين على الشكل4-12 وهو مخطط البنية الداخلية لقطب دخل/خرج لمتحكم AVR. يمكن تفعيل أو إلغاء تفعيل هذه المقاومة لكل قطب دخل على حدى وبشكل افتراضي تكون هذه المقاومات غير مفعلة. ملاحظة: لا تملك متحكمات AVR مقاومات سحب داخلية وبالتالي يجب وضع مقاومات سحب خارجية من أجل التوصيلات في الشكل4-8 والشكل4-11.

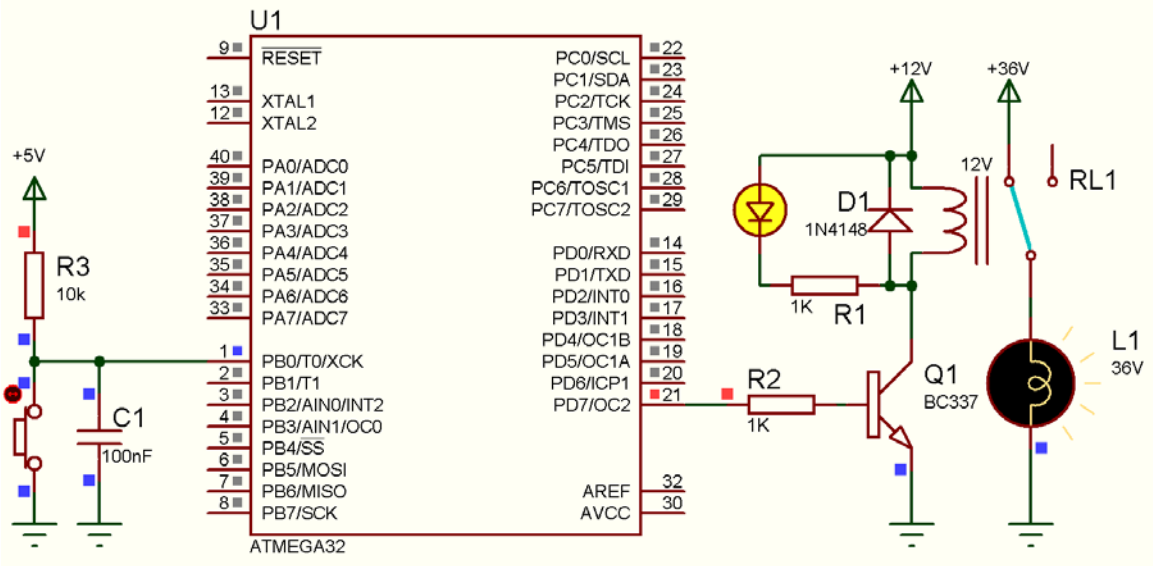


الشكل4-12 مخطط البنية الداخلية لقطب دخل/خرج لمتحكم AVR

4-5 تعليمات تعريف الأقطاب كمداخل في Bascom-AVR (Input Configuration Instructions in Bascom):

وظيفة التعليمة	شكل التعليمة
تعريف البوابة C كبوابة دخل	`Config PORTC = Input`
تعريف القطب رقم 5 من البوابة C كقطب دخل	`Config PINC.5 = Input`
تفعيل مقاومات الرفع الداخلية للبوابة C كاملةً	`PORTC = 255`
إلغاء تفعيل مقاومات الرفع الداخلية للبوابة C كاملةً	`PORTC = 0`
تفعيل مقاومة الرفع الداخلية للقطب رقم 5 فقط من البوابة C	`PINC.5 = 1`
إلغاء تفعيل مقاومة الرفع الداخلية للقطب رقم 5 فقط من البوابة C	`PINC.5 = 0`
تفعيل بعض مقاومات الرفع الداخلية للبوابة C (PIN.4,5,6,7)	`PORTC = &B11110000`
يمكن استخدام هذا الشكل لتعريف الأقطاب من البوابة كدخل/خرج حيث أن (0) تعني قطب دخل، والـ (1) تعني قطب خرج.	`Config PORTC = &B11110000`
يصرح إلى أن PINC سوف يشار إليها أثناء البرنامج بالاسم (SWs)	`SWs Alias PINC`
يصرح إلى أن القطب PINC.5 سوف يشار إليه بالاسم (SW)	`SW Alias PINC.0`

التجربة السادسة: المطلوب التحكم بتشغيل وفصل الريليه في التجربة الخامسة باستخدام مفتاح لحظي موصول إلى القطب PINB.0، بحيث أنه عند ضغط المفتاح تعمل الريليه وعند تحرير المفتاح تفصل الريليه – باستخدام مقاومة رفع خارجية.



الشكل4-13 التحكم بالريليه عن طريق مفتاح لحظي

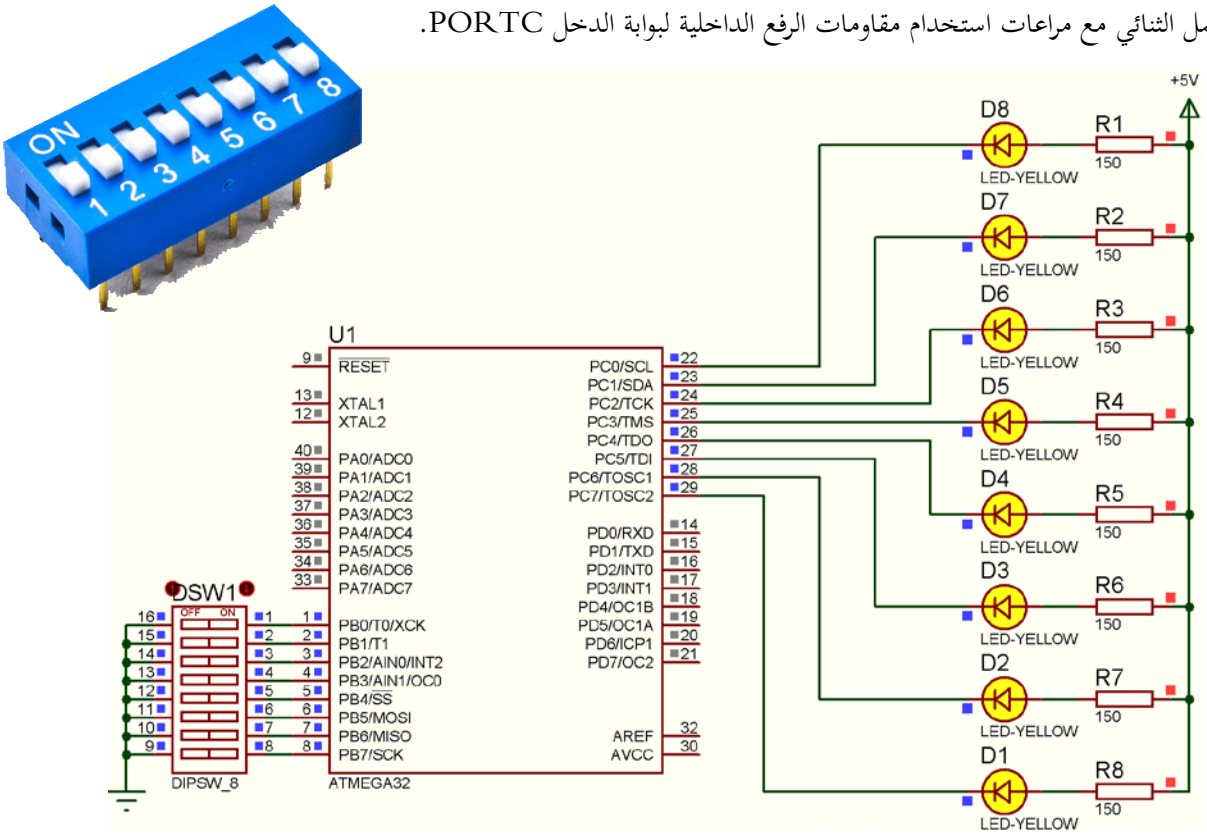
البرنامج Exp.06.bas في بيئة BASCOM-AVR:

```
' ********************************************************************************
' * Title          : Exp.06.bas                                                  *
' * Target MCU     : ATMega128A                                                  *
' * Author         : Walid Balid                                                 *
' * IDE            : BASCOM AVR 2.0.7.3                                          *
' * Peripherals    : Relay - Switch                                              *
' * Description    : GPIOs as Output/Input                                       *
' ********************************************************************************
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'-----------------------[Definitions]
$regfile = "m32def.dat"
$crystal = 8000000
'-----------------------
'-----------------------[GPIO Configurations]
Config Pind.7 = Output : Relay Alias Portd.7
Config Pinb.0 = Input : Switch Alias Pinb.0
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'--->[Main Program]
Do
   If Switch = 0 Then Set Relay Else Reset Relay
Loop
End
'---<[End Main]
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

ملاحظة: من أجل الاستغناء عن مقاومة الرفع الخارجية وتفعيل مقاومة الرفع الداخلية فإن كل ما نحتاجه هو إضافة تعليمة تفعيل مقاومة الرفع الداخلية للقطب PINB.0 بعد تعليمة تفعيل القطب وهي:

```
PORTB.0 = 1
```

التجربة السابعة: المطلوب التحكم بتشغيل وفصل ثمانية ثنائيات ضوئية موصولة إلى البوابة PORTC باستخدام مفتاح DIP-Switch موصول إلى البوابة PINB، بحيث أنه عند تفعيل المفتاح (on) يعمل الثنائي الموافق لرقم المفتاح وعند إلغاء تفعيل المفتاح (off) سوف يتوقف عمل الثنائي مع مراعات استخدام مقاومات الرفع الداخلية لبوابة الدخل PORTC.



الشكل4-14 توصيل المفاتيح الانزلاقية DIP-Switch

البرنامج Exp.07.bas في بيئة BASCOM-AVR:

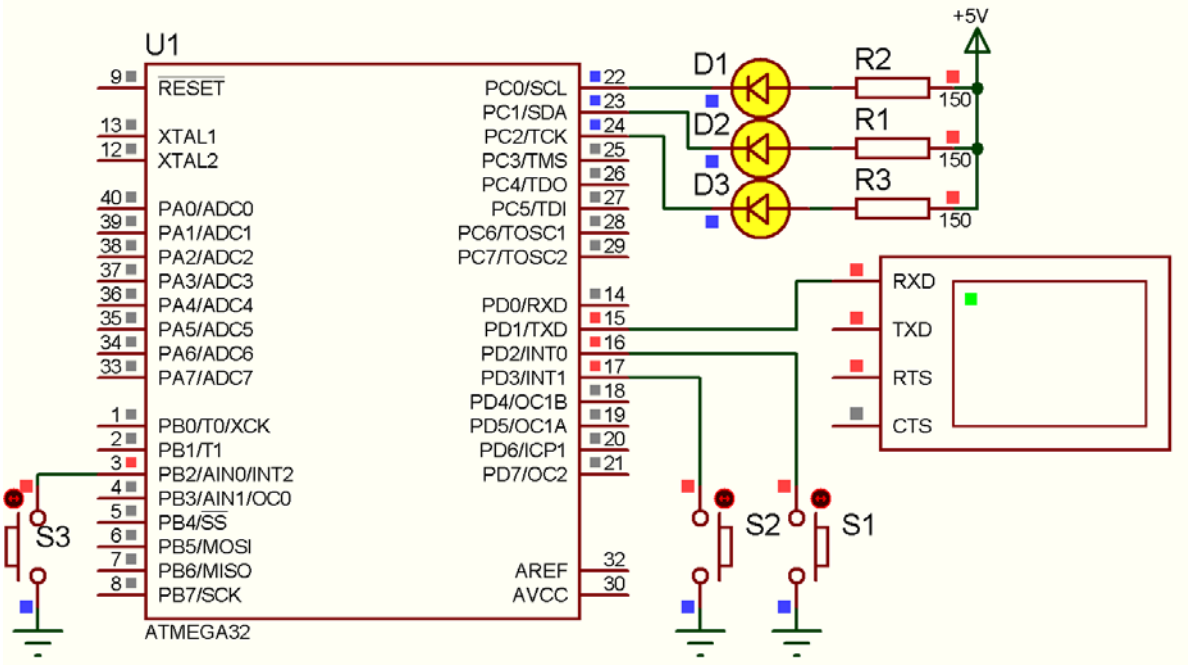
```
' ****************************************************************************
' * Title          : Exp.07.bas                                              *
' * Target MCU     : ATMega128A                                              *
' * Author         : Walid Balid                                             *
' * IDE            : BASCOM AVR 2.0.7.3                                      *
' * Peripherals    : DIP-Switch                                              *
' * Description    : GPIOs as Output/Input                                   *
' ****************************************************************************
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'-----------------------[Definitions]
$regfile = "m32def.dat"
$crystal = 8000000
'------------------------
'-----------------------[GPIO Configurations]
Config Portc = Output : Leds Alias Portc
Config Portb = Input : Switchs Alias Pinb : Portb = &B11111111
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'--->[Main Program]
Do
   Leds = Switchs
Loop
End
'---<[End Main]
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

6-4 تعليمات قراءة حالة مفتاح موصول إلى قطب دخل في Bascom (Reading a Port PIN connected to a switch):

إن التجربة السادسة تضمنت قراءة حالة مفتاح لحظي من خلال قراءة القيمة الموجودة على القطب المتصل معه المفتاح، ومبدأ التخلص من العطالة الميكانيكية استخدم مكثف خارجي مع القطب. وتضمنت الفقرة 4-3 طريقتان للتخلص من العطالة الميكانيكية للمفاتيح، حيث أن الطريقة الثانية تستخدم تأخير زمني برمجي. التعليمات التالية يمكن استخدامها كتعليمات برمجية جاهزة في البيئة Bascom تسهل إلى حد كبير التعامل مع المفاتيح اللحظية وإضافة تأخير زمني.

وظيفة التعليمة	شكل التعليمة
يراقب حالة القطب المحدد في Px.y كلما مر عليه، وعندما تصبح حالته موافقة للحالة المحددة في state، سوف يقفز إلى البرنامج الفرعي عند اللافتة label وينفذ البرنامج ويعود.	`Debounce Px.y , state , label , Sub` `Ex. Debounce Key1 , 0 , Sw1 , Sub`
تهيئة زمن تأخير (ميلي ثانية) عن استعمال تعليمة Debounce للتخلص من العطالة الميكانيكية للمفتاح.	`Config Debounce = time`
سوف يقف البرنامج عند هذه التعليمة وينتظر أن تصبح حالة القطب صفر (reset) أو واحد (set) منطقي ليكمل البرنامج.	`Bitwait x , Set/reset` `Ex. Bitwait Pinb.7 , reset`

التجربة الثامنة: يوجد على اللوحة التعليمية Mini-Phoenix ثلاث مفاتيح لحظية موصولة إلى الأقطاب Pind2, Pind3, Pinb.2، كما يوجد ثمانية ثنائيات ضوئية (Leds) موصولة إلى البوابة PortC، والمطلوب: باستخدام التعليمة الشرطية If تغيير حالة عمل (Toggle) الثنائي D1 عن الضغط على المفتاح S1، وتغيير حالة D2 عند الضغط على S2، وتغيير حالة D3 عند الضغط على S3.



الشكل4-15 توصيل المفاتيح اللحظية الثنائيات مع المتحكم ATmega32A على اللوحة Mini-Phoenix

البرنامج Exp.08.bas في بيئة BASCOM-AVR:

```
' ********************************************************************************
' * Title          : Exp.08.bas                                                  *
' * Target Board   : Mini-Phoenix - REV 1.00                                     *
' * Target MCU     : ATMega32A                                                   *
' * Author         : Walid Balid                                                 *
' * IDE            : BASCOM AVR 2.0.7.3                                          *
' * Peripherals    : Pull-Up Resistors                                           *
' * Description    : GPIOs as Input; Active Low (GND)                            *
' ********************************************************************************
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'-----------------------[Definitions]
$regfile = "m32def.dat"
$crystal = 8000000
$baud = 9600
'-----------------------
'-----------------------[GPIO Configurations]
Config Portc = &B00000111
Led1 Alias Portc.0 : Led2 Alias Portc.1 : Led3 Alias Portc.2
Set Led1 : Set Led2 : Set Led3

Config Pind.2 = Input : Sw_1 Alias Pind.2 : Portd.2 = 1  'PU Internal Resistor
Config Pind.3 = Input : Sw_2 Alias Pind.3 : Portd.3 = 1
Config Pinb.2 = Input : Sw_3 Alias Pinb.2 : Portb.2 = 1
'-----------------------
'-----------------------[Variables]
Dim Count1 As Byte , Count2 As Byte , Count3 As Byte
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'--->[Main Program]
Print "Hello!"
Do
   If Sw_1 = 0 Then Gosub Sw_r1
   If Sw_2 = 0 Then Gosub Sw_r2
   If Sw_3 = 0 Then Gosub Sw_r3
Loop
End
'---<[End Main]
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'--->[Print]
Sw_r1:
   Toggle Led1 : Count1 = Count1 + 1
   Print "Sw1 has Pressed! > " ; Count1
Return
'---<
Sw_r2:
   Toggle Led2 : Count2 = Count2 + 1
   Print "Sw2 has Pressed! > " ; Count2
Return
'---<
Sw_r3:
   Toggle Led3 : Count3 = Count3 + 1
   Print "Sw3 has Pressed! > " ; Count3
Return
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

التجربة التاسعة: المطلوب تعديل البرنامج في التجربة الثامنة باستبدال التعليمة الشرطية If بالتعليمة Debounce.

البرنامج Exp.09.bas في بيئة BASCOM-AVR:

```
' ********************************************************************************
' * Title          : Exp.09.bas                                                  *
' * Target Board   : Mini-Phoenix - REV 1.00                                     *
' * Target MCU     : ATMega32A                                                   *
' * Author         : Walid Balid                                                 *
' * IDE            : BASCOM AVR 2.0.7.3                                          *
' * Peripherals    : Pull-Up Resistors                                           *
' * Description    : GPIOs as Input; Active Low (GND)                            *
' ********************************************************************************
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'-----------------------[Definitions]
$regfile = "m32def.dat"
$crystal = 8000000
$baud = 9600
'-----------------------
'-----------------------[GPIO Configurations]
Config Portc = &B00000111
Led1 Alias Portc.0 : Led2 Alias Portc.1 : Led3 Alias Portc.2
Set Led1 : Set Led2 : Set Led3

Config Pind.2 = Input : Sw_1 Alias Pind.2 : Portd.2 = 1    'PU Internal Resistor
Config Pind.3 = Input : Sw_2 Alias Pind.3 : Portd.3 = 1
Config Pinb.2 = Input : Sw_3 Alias Pinb.2 : Portb.2 = 1

Config Debounce = 50
'-----------------------
'-----------------------[Variables]
Dim Count1 As Byte , Count2 As Byte , Count3 As Byte
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'--->[Main Program]
Print "Hello!"
Do
   Debounce Sw_1 , 0 , Sw_r1 , Sub
   Debounce Sw_2 , 0 , Sw_r2 , Sub
   Debounce Sw_3 , 0 , Sw_r3 , Sub
Loop
End
'--->[End Main]
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'--->[Print]
Sw_r1:
   Toggle Led1 : Count1 = Count1 + 1
   Print "Sw1 has Pressed! > " ; Count1
Return
'--->
Sw_r2:
   Toggle Led2 : Count2 = Count2 + 1
   Print "Sw2 has Pressed! > " ; Count2
Return
'--->
Sw_r3:
   Toggle Led3 : Count3 = Count3 + 1
   Print "Sw3 has Pressed! > " ; Count3
Return
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

التجربة العاشرة: المطلوب تعديل البرنامج في التجربة الثامنة باستبدال التعليمة الشرطية If بالتعليمة Bitwait.

البرنامج Exp.10.bas في بيئة BASCOM-AVR:

```
' ************************************************************************
' * Title          : Exp.10.bas                                          *
' * Target Board   : Mini-Phoenix - REV 1.00                             *
' * Target MCU     : ATMega32A                                           *
' * Author         : Walid Balid                                         *
' * IDE            : BASCOM AVR 2.0.7.3                                  *
' * Description    : GPIOs as Input; Active Low (GND)                    *
' ************************************************************************
'-----------------------[Definitions]
$regfile = "m32def.dat"
$crystal = 8000000
$baud = 9600
'-----------------------
'-----------------------[GPIO Configurations]
Config Portc = &B00000111
Led1 Alias Portc.0 : Led2 Alias Portc.1 : Led3 Alias Portc.2
Set Led1 : Set Led2 : Set Led3

Config Pind.2 = Input : Sw_1 Alias Pind.2 : Portd.2 = 1    'PU Internal Resistor
Config Pind.3 = Input : Sw_2 Alias Pind.3 : Portd.3 = 1
Config Pinb.2 = Input : Sw_3 Alias Pinb.2 : Portb.2 = 1
'-----------------------
'-----------------------[Variables]
Dim Count1 As Byte , Count2 As Byte , Count3 As Byte
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'--->[Main Program]
Do
   Print "PC is waiting for Sw1"
   Bitwait Sw_1 , Reset : Gosub Sw_r1

   Print "PC is waiting for Sw2"
   Bitwait Sw_2 , Reset : Gosub Sw_r2

   Print "PC is waiting for Sw3"
   Bitwait Sw_3 , Reset : Gosub Sw_r3
Loop
End
'---<[End Main]
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'--->[Print]
Sw_r1:
   Toggle Led1 : Count1 = Count1 + 1
   Print "Sw1 has Pressed! > " ; Count1
Return
'---<
Sw_r2:
   Toggle Led2 : Count2 = Count2 + 1
   Print "Sw2 has Pressed! > " ; Count2
Return
'---<
Sw_r3:
   Toggle Led3 : Count3 = Count3 + 1
   Print "Sw3 has Pressed! > " ; Count3
Return
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

ملاحظة: شرح البرامج موجود في ملف الفيديو للمحاضرة الخامسة.

. . . ﴿انتهت الجلسة العملية الرابعة﴾ . . .

وليد بليد - دمتم بخير ومودة ونور -