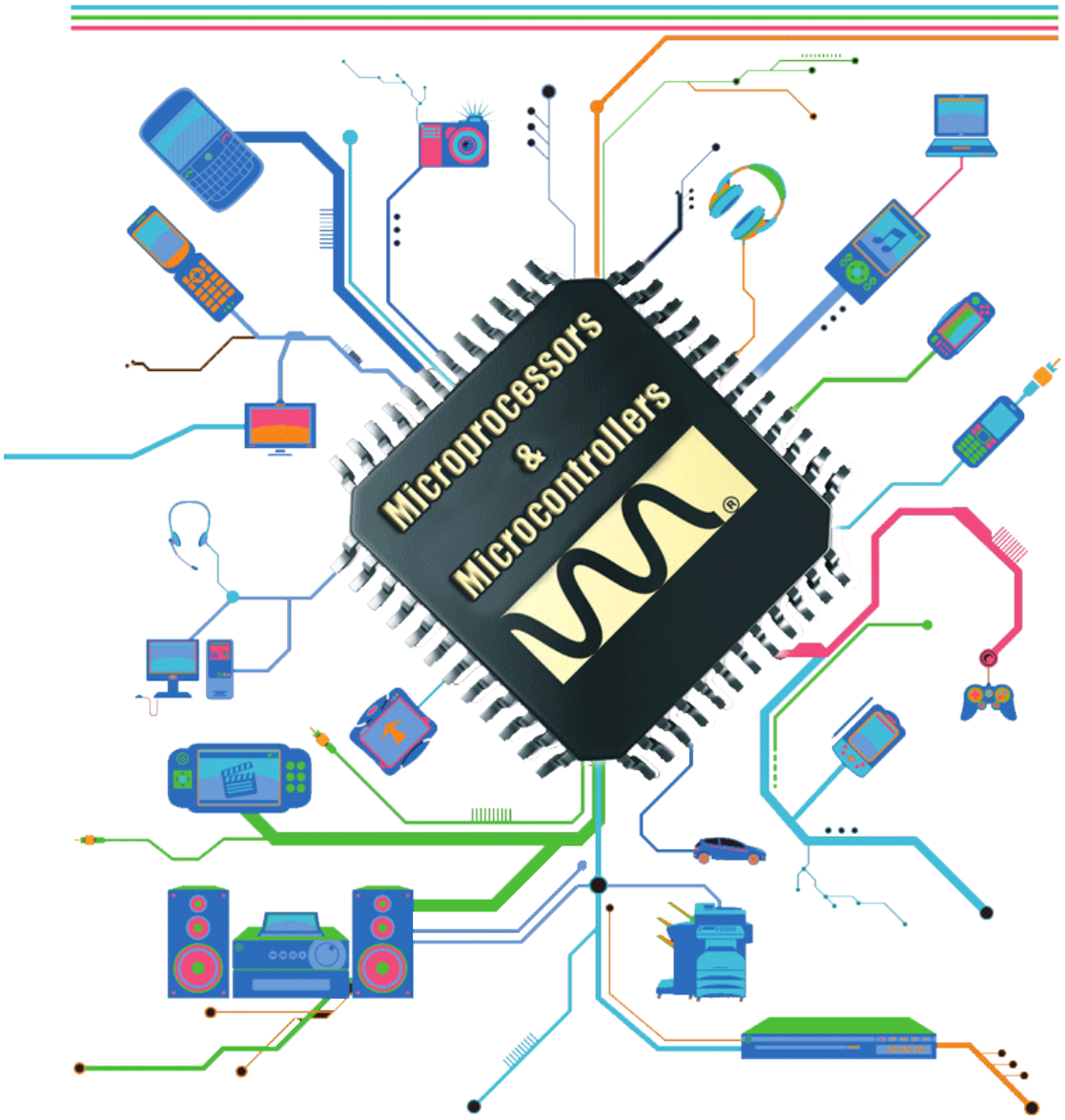
جامعـــــــــــة حلـــــــــــب
كلية الهندسة الكهربائية والإلكترونية
قسـم هندسـة التحكـم والأتمتـة

﴿ الجلسات العملية لمادة المعالجات والمتحكمات المصغرة ﴾

Microprocessors & Microcontrollers Lab Sessions

السنة الثالثة | قسم اتصالات

﴿ الجلسة العملية الثامنة ﴾



م. وليد بليد

Wednesday, April 25, 2012

الجلسة العملية الثامنة

نظرة عامة (Overview):

هذه المحاضرة تشرح مبادئ الاتصالات التسلسلية والنافذة التسلسلية اللامتزامنة UART.ثم برمجة النافذة UART في تطبيقات عدة منها: ربط متحكمات في شبكة سلكية، إرسال البيانات لاسلكياً باستخدام الأشعة تحت الحمراء، إرسال البيانات لاسلكياً باستخدام الليزر، إرسال البيانات لاسلكياً باستخدام الترددات الراديوية، وأخيراً ربط موديول GPS مع النافذة UART واستحصال الوقت والتاريخ والإحداثيات الجغرافية.

1-8 بروتوكولات الاتصال (Communication Protocols):

تتفرع بروتوكولات الاتصال بشكل عام إلى فرعين رئيسيين:

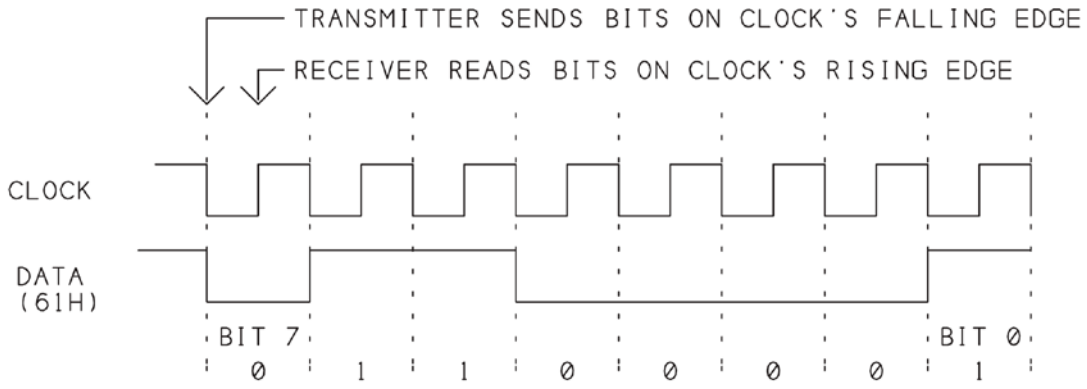
1) اتصالات تفرعية.. 2) اتصالات تسلسلية.

يختصر استخدم الاتصالات التفرعية من أجل نقل البيانات بسرعات عالية جداً ولمسافات قصيرة جداً، والسبب في محدودية المسافة هو تشكل السعات الطفيلية والضجيج العالي على مسارات خطوط النقل التفرعية عند ازدياد طول الناقل، كما أن حجم الناقل سيكون كبير وبالتالي فإن كلفة الناقل ستكون كبيرة أيضاً. في حين تستخدم الاتصالات التسلسلية على نطاق أوسع بكثير من الاتصالات التفرعية وتمتاز بمناعة عالية ضد الضجيج ونقل لمسافات بعيدة، كما أن حجم الناقل سيكون صغير وكلفته ضئيلة نسبياً مقارنة مع الناقل التفرعية.

Serial Communications		Parallel Communications
Asynchronous	Synchronous	
• Morse code telegraphy • RS-232 (COM Port) • RS-423 • RS-485 • Universal Serial Bus (USB) • FireWire • Ethernet • Fiber Channel • InfiniBand • MIDI • DMX512 • Serial ATA • SpaceWire • PCI Express • SONET and SDH • T-1, E-1	o I2C o SPI o PS2	▪ LPT ▪ ISA ▪ EISA ▪ VESA ▪ ATA ▪ SCSI ▪ PCI ▪ PCMCIA ▪ IEEE-1284 ▪ IEEE-488

2-8 مفاهيم أساسية في الاتصالات التسلسلية غير المتزامنة (Asynchronous):

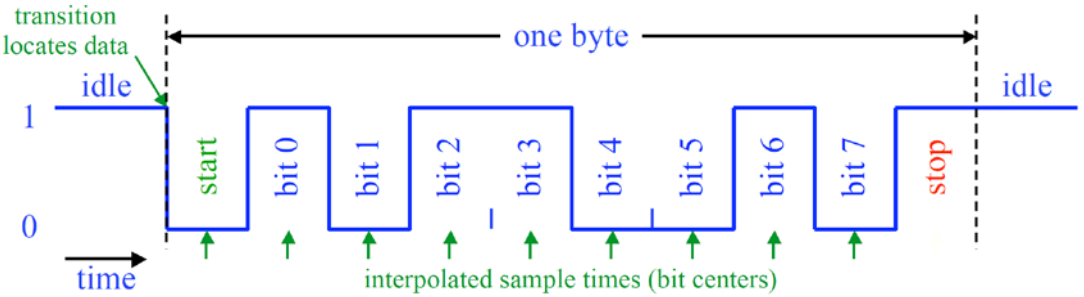
أولاً: الاتصالات المتواقتة (المتزامنة): يكون فيها بروتوكول الإرسال مؤلف من خطين على الأقل أحدهما خط التزامن (clock)، وبالتالي فإن سرعة إرسال البيانات تتحدد من خلال تردد إشارة التزامن بحيث يتم إرسال كل بت من البتات تسلسلياً عند جبهة التزامن (صاعدة أو هابطة).



الشكل1 إشارة البيانات وإشارة التزامن في بروتوكول إرسال متزامن

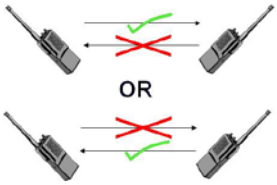
ملاحظة: بازدياد المسافة بين الطرفيتين فإنه يحصل انحراف\انزياح بين إشارة التواقت وبين إشارة البيانات مما يؤدي إلى فشل عملية النقل.

ثانياً: اتصالات غير متواقتة (غير متزامنة): لا تحوي على خط تزامن وإنما يتم بدء عملية الإرسال بإرسال بت بدء الإرسال (Start Bit) والذي بدوره يعلم المستقبل أن الذي يليه هو بايت البيانات، وبعدها يتم إرسال البايت المطلوب وتنتهي عملية إرسال البايت بإرسال بت التوقف (Stop Bit) والذي بدوره يعلم المستقبل أن عملية إرسال البايت قد انتهت ويجب تخزين البايت في مسجل نافذة الاستقبال والتحضر لاستقبال البايت التالي إن وجد.



الشكل2 إشارة البيانات في بروتوكول إرسال غير متزامن

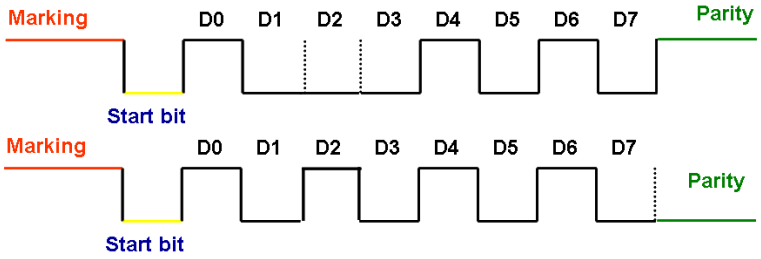
ملاحظة: بخلاف الاتصالات المتواقتة فإن ازدياد المسافة بين الطرفيتين لا يؤدي إلى فشل عملية النقل، كما أن هذه الطريقة أقل كلفة وأبسط بنية وأسهل برمجة.



الإرسال أحادي الاتجاه (Half-Duplex): تتم فيه عملية الاتصال بين الطرفيتين باتجاه واحد فقط في نفس اللحظة الزمنية، فإما أن تكون في حالة إرسال أو استقبال.

الإرسال ثنائي الاتجاه (Full-Duplex): يمكن أن تكون الوحدة الطرفية في حالة إرسال واستقبال في نفس اللحظة الزمنية.

خانة الإيجابية (Parity Bit): خانة يضيفها المرسل ويستخدمها المستقبل لضمان عدم ضياع المعلومات، وتتعلق خانة الإنجابية بعدد الواحدات في البايت المرسل.



الشكل3 توضع خانة الإيجابية في إشارة البيانات لبروتوكول إرسال غير متزامن

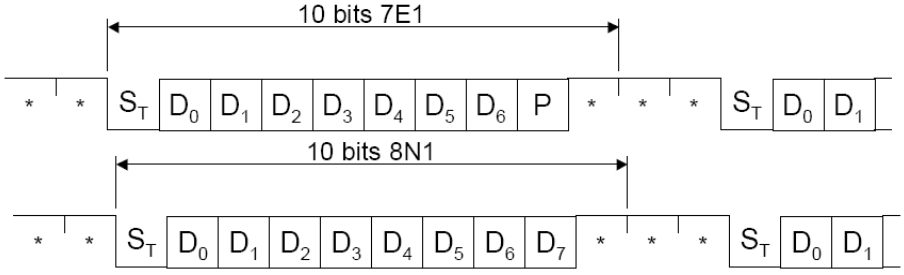
في حال كون خانة الإيجابية "Even" فإن هذه الخانة تملك القيمة "0" إذا كان عدد الواحدات في البايت المرسل زوجي وإلا فستصبح "1". الأمثلة التالية توضح ذلك.

10110010 > Parity Bit = 0 | 10110110 > Parity Bit = 1

في حال كون خانة الإيجابية "Odd" فإن هذه الخانة تملك القيمة "0" إذا كان عدد الواحدات في البايت المرسل فردي وإلا فستصبح "1". الأمثلة التالية توضح ذلك.

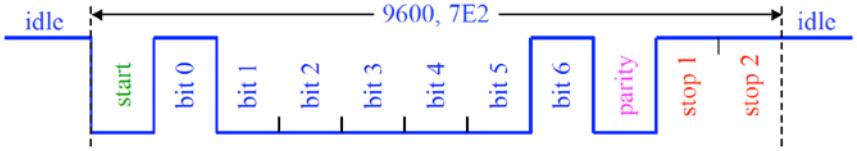
10110010 > Parity Bit = 1 | 10110110 > Parity Bit = 0

عدد البتّات لكل محرف (N): يتم فيها التصريح عن عدد البتات لبايت البيانات التي سيتم إرسالها، فإما أن تكون 5, 6, 7 or 8bit، ولكن يجب الانتباه مثلاً: في حال إرسال N=7bit فإن قيم العظمى ASCII=127.



الشكل4 مثال عن عدد بتات مختلف 8|7 في إشارتي بيانات لبروتوكول إرسال غير متزامن

خانة بت التوقف (Stop Bit): يعلم المرسل من خلالها المستقبل بانتهاء عملية الإرسال. 2 or 1.5 ,1 بت.



الشكل5 توضع خانة بت التوقف في نهاية إشارة البيانات لبروتوكول إرسال غير متزامن

معدل سرعة النقل (Baud Rate): وهو عدد البتات المرسلة خلال ثانية واحد على خط اتصال تسلسلي، وهناك قيم قياسية متعارف عليها لمعدلات النقل وهي: 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200, etc…

إن الزمن اللازم لإرسال بت واحد يعطى بالعلاقة التالية:

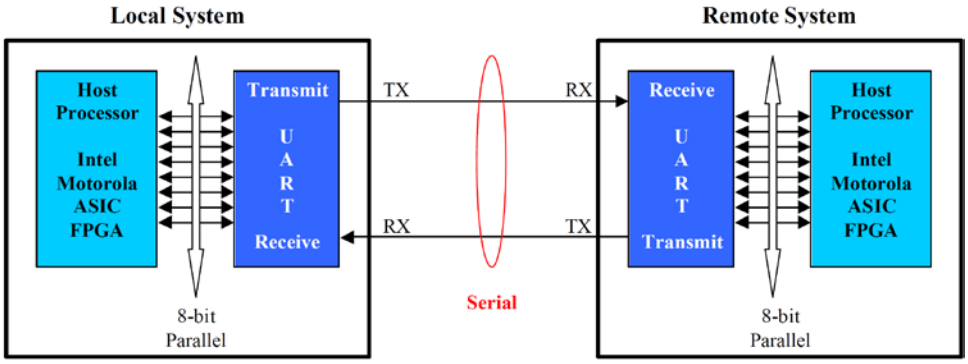
$$Bit_{Time} = \frac{1}{Baud\ Rate}$$

إن عدد البايتات التي يمكن إرسالها خلال ثانية واحدة يمكن حسابها من العلاقة التالية:

$$Bytes_{Num/1sec} = \frac{Baud\ Rate}{8}$$

3-8 النافذة التسلسلية UART (Universal Asynchronous Receiver and Transmitter Interface):

تعتبر هذه النافذة من أكثر نوافذ الاتصال التسلسلي استخداماً في الأنظمة الرقمية ومبدأ عملها وكذلك بروتوكولها متوافق تماماً مع البروتوكول RS232 إلا أن المستويات المنطقية فيها وفق المنطق TTL، وتتميز بسهولة وبساطة استخدامها بالإضافة إلى الكلفة المنخفضة للربط بين متحكمين (MCU-MCU)، أو الربط بين حاسب ومتحكم (MCU-PC).



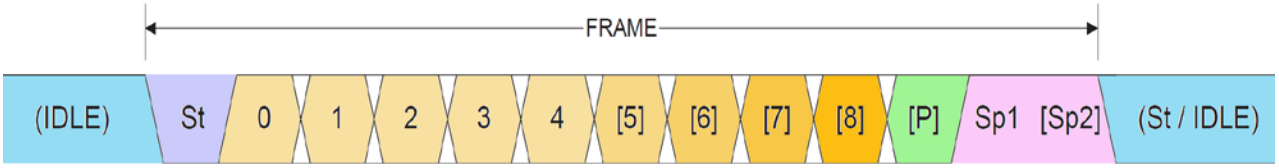
الشكل6 الربط بين متحكمين من خلال النافذة UART

تملك النافذة التسلسلية في متحكمات العائلة AVR على ميزات عديدة وهي تعمل في نمطين مستقلين:

- ✓ UART: نافذة تسلسلية عامة للإرسال والاستقبال اللامتزامن عبر القطبان TXD, RXD.
- ✓ USART: نافذة تسلسلية عامة للإرسال والاستقبال المتزامن عبر القطبان TXD, RXD بالإضافة إلى القطب XCK كقطب تزامن.

بنية إطار البيانات (UART Frame Format):

إن تشكيل إطار البيانات المرسلة أو المستقبلة للنافذة UART مشابه تماماً لبنية إطار البروتوكول RS232 باختلاف وحيد وهو المستوى المنطقي المعكوس.



الشكل7 بنية إطار البيانات المرسلة أو المستقبلة للنافذة UART

St: Start bit, always low.

Data bits: (0 to 8).

P: Parity bit (Can be odd or even)

Sp: Stop bit, always high.

IDLE: No transfers on the communication line (RxD or TxD), IDLE line is high.

حساب قيمة مسجل معدل النقل (Baud Rate Register):

من أجل تحديد معدل سرعة النقل للنافذة التسلسلية يتم شحن المسجل UBRR بقيمة تحسب وفقاً للعلاقات في الشكل8.

Operating Mode	Equation for Calculating Baud Rate[1]	Equation for Calculating UBRR Value
Asynchronous Normal Mode (U2X = 0)	$BAUD = \frac{f_{OSC}}{16(UBRR+1)}$	$UBRR = \frac{f_{OSC}}{16BAUD} - 1$
Asynchronous Double Speed Mode (U2X = 1)	$BAUD = \frac{f_{OSC}}{8(UBRR+1)}$	$UBRR = \frac{f_{OSC}}{8BAUD} - 1$
Synchronous Master Mode	$BAUD = \frac{f_{OSC}}{2(UBRR+1)}$	$UBRR = \frac{f_{OSC}}{2BAUD} - 1$

الشكل8 معادلات حساب قيمة المسجل UBRR الموافقة لمعدل النقل

حيث أنّ: UBRR هي محتوى المسجل UBRRH and UBRR وتتراوح 4095 – 0.

مثال: أحسب قيمة المسجل UBRR من أجل تردد هزاز كريستالي 1Mhz ومعدل نقل 9600bps ونمط عمل عام غير متواقت.

$$UBRR_{H,L} = \frac{f_{osc}}{16 \times Baud} - 1 = \frac{1000000}{16 \times 9600} - 1 = 5.510416 \approx 6$$

كما هو ملاحظ فإن القيمة غير دقيقة أي أن هناك خطأ في قيمة معدل النقل ولن تكون القيمة تماماً 9600، وبالتالي إذا كانت دارة المستقبل تعتمد تردد عمل مختلف وكان الخطأ مختلف فإنه ربما يحصل تشوه في البيانات بسبب عدم التزامن الدقيق في معدل النقل.

لذلك يوصى بمعدلات نقل قياسية وترددات هزازات كريستالية قياسية لتفادي الأخطاء الكبيرة في حساب معدلات النقل، بحيث أن الخطأ يجب أن لا يتجاوز 0.5% من أجل الحصول على وثوقية عمل عالية؛ لكن يمكن أن يعمل النظام بدون مشاكل حتى خطأ 5%.

يمكن حساب الخطأ من العلاقة التالية:

$$ERROR_{[\%]} = \left(\frac{BaudRate_{CloseMatch}}{BaudRate_{Calculated}} - 1\right) \times 100\%$$

مثال: من أجل نفس المثال السابق، نعوض في العلاقة السابقة:

$$ERROR_{[\%]} = \left(\frac{9600}{8928.571} - 1\right) \times 100\% = 7.52\%$$

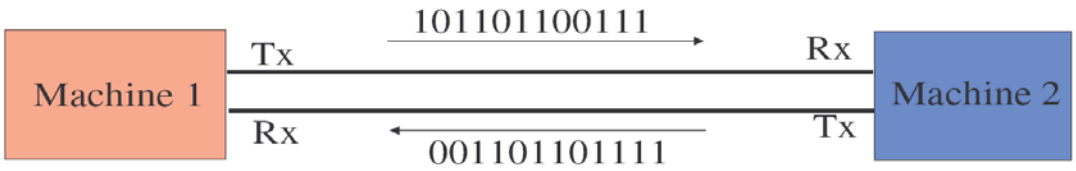
ملاحظة: من أجل تفادي مشكلة أخطاء معدل النقل قم باختيار تردد الهزاز الكريستالي بحيث يكون من مضاعفات معدل النقل.

8-4 تحقيق اتصال بين طرفيتين في برتوكول UART:

هناك بارامترات يجب تحديدها بين المرسل والمستقبل قبل إرسال البيانات في الاتصالات غير المتواقتة وهي:

- ✓ تحديد نمط الإرسال: أحادي الاتجاه (Half-Duplex) أو ثنائي الاتجاه (Full-Duplex).
- ✓ تحديد عدد البتات لكل محرف: 6, 7 or 8 bit.
- ✓ تحديد معدل سرعة الإرسال (Baud Rate).
- ✓ تحديد استخدام أو عدم استخدام خانة فحص الإيجابية (Parity Bit)، وفي حال الاستخدام يجب تحديد نمط فحص خانة الإيجابية (Even or Odd).
- ✓ تحديد عدد بتات التوقف (1, 1.5 or 2).

عموماً، فإنه من أجل تحقيق اتصال بين طرفيتين بدون مصافحة يكفي توصيل قطب الإرسال "**TxD**" والاستقبال "**RxD**" على التوازي المتعاكس كما في الشكل التالي:



الشكل9 تحقيق اتصال بين طرفيتين من خلال النافذة UART

8-5 هناك نمطين للبيانات في الاتصالات التسلسلية وهما:

1) **نمط الآسكي** (Ascii Mode): يتم تمثيل كل خانة على أنها محرف مستقل ويتم إرسال قيمة الآسكي لهذا المحرف. **مثال**: التعليمة "**Print 123**" ستقوم بإرسال الأرقام (1,2,3) على أنها محارف، وبالتالي سترسل الآسكي لكل منها [51][50][49] – بالنتيجة سترسل ثلاث بايتات.

2) **النمط الثنائي** (BIN Mode): يتم تمثيل البيانات على أنها قيمة عديدة وليس محرفية ويتم إرسال القيمة الثنائية لهذا العدد. **مثال**: التعليمة "**Printbin 123**" ستقوم بإرسال القيمة (123) على أنها بايت واحد، وبالتالي سترسل [1111011] – بالنتيجة سترسل بايت واحد فقط.

8-6 التعامل مع النافذة UART في Bascom-AVR:

1) تعليمات التهيئة (Configuration).
2) تعليمات الإرسال (Sending over TXD).
3) تعليمات الاستقبال (Receiving over RXD).

التعليمة البرمجية	شرح التعليمة
`$baud = Var`	تحديد معدل النقل العام للنافذة التسلسلية UART0.
`Print Var ; "const"`	إرسال البيانات عبر النافذة التسلسلية UART0.
`Printbin Var [; Varn]`	إرسال البيانات بصيغة ثنائية عبر النافذة التسلسلية UARTx. [; Varn]: خيار من أجل تحديد عدد البايتات المراد إرسالها (مصفوفة).
`Input ["prompt"] , Var [,Varn]`	قراءة البيانات الواردة على النافذة التسلسلية UART0. ["prompt"]: خيار يقوم بإرسال رسالة نصية قبل قراءة محتوى النافذة. Var: المتحول الذي سيتم إدخاله (رقمي، محرفي). [,Varn]: خيار من أجل إدخال أكثر من متحول بنفس التعليمة (n=1,2…).
`Inputbin Var1 [,Var2]`	قراءة البيانات الواردة على النافذة UART0 بصيغة ثنائية. [,Var2]: خيار من أجل تحديد عدد البايتات المراد إدخالها (مصفوفة).
`Inputhex ["prompt"],Var[,Varn]`	قراءة البيانات الواردة على النافذة UART0 بالصيغة HEX.
`var = INKEY()`	تعود بقيمة الـ Ascii لأول محرف في مسجل buffer النافذة UART0.
`var = WAITKEY()`	ينتظر وصول أول محرف إلى مسجل buffer النافذة التسلسلية UART0 ويعود بقيمة الـ Ascii له.
`Var = Iscarwaiting()`	يفحص محتوى buffer مسجل النافذة UART0 ويعود بالقيمة "1" إذا كان هناك أي محرف، وإلا فسوف يعود بالقيمة "0"، مع العلم أن هذه التعليمة تفحص محتوى المسجل ولا تؤثر على محتواه!

ملاحظة: من أجل إرسال أكثر من متحول على نفس السطر يمكن استخدام (;) للفصل بين المتحولات (Print A ; B ; C).

ملاحظة: إن التعليمة **Printbin** مكافئة تماماً للتعليمة **Print Chr (var);**.

ملاحظة: يمكن استخدام التعليمة **Printbin** من أجل إرسال عدة متحولات مخزنة في مصفوفة؛ كما في المثال التالي سوف يتم إرسال عشر بايتات موجودة في المتحول (مصفوفة) Arr.

```
Printbin Arr(1) ; 10
```

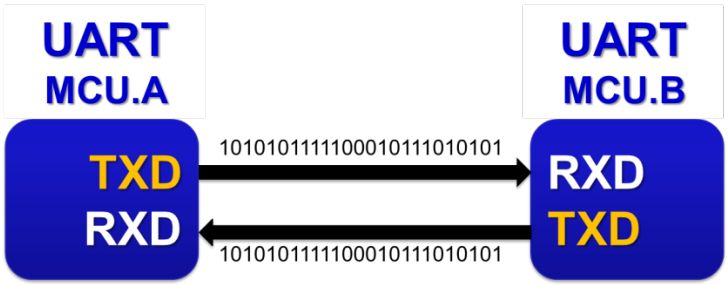
ملاحظة: يمكن استخدام التعليمة **Inputbin** من أجل إدخال عدة متحولات وإسنادها إلى مصفوفة؛ كما في المثال التالي سوف يتم استلام عشر بايتات ووضعها في المصفوفة Arr.

```
Inputbin Arr(1) , 10
```

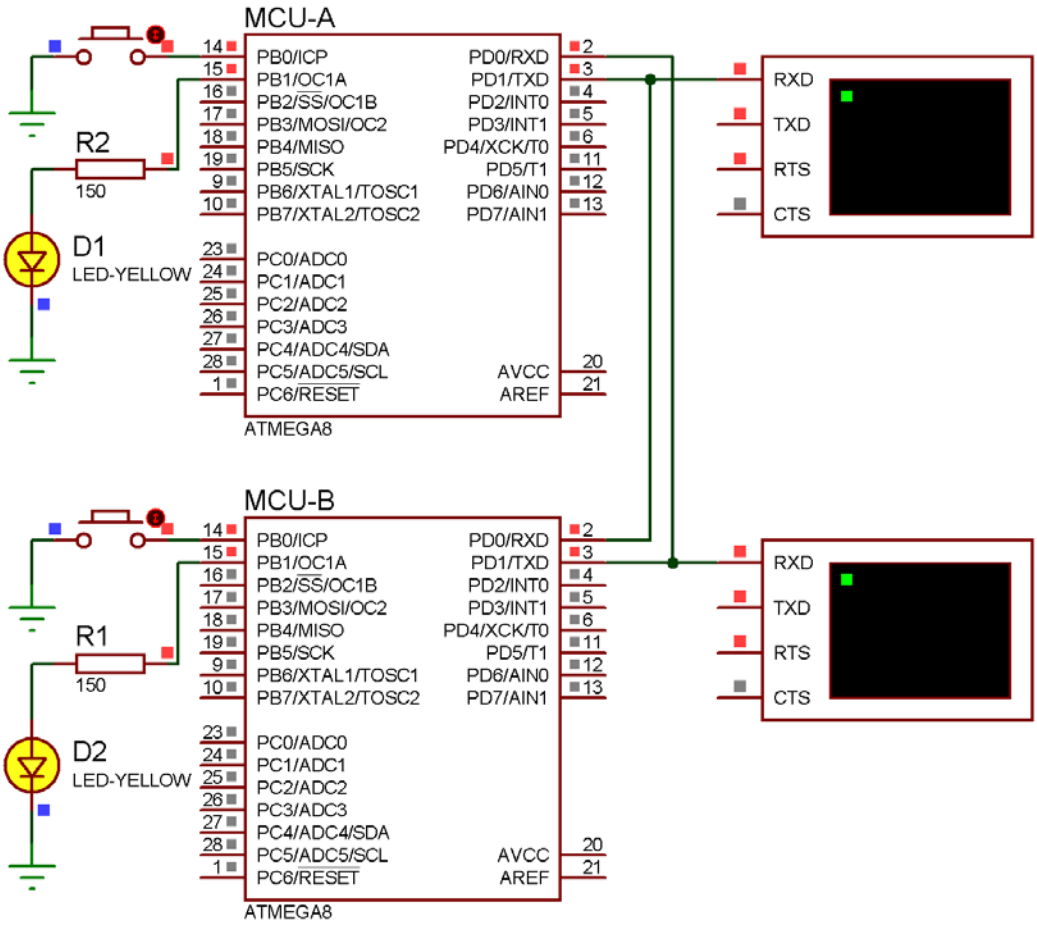
ملاحظة: إن التعليمة **Inputbin** سوف تنتظر حتى تستلم جميع البايتات المحددة في متحولاتها!

7-8 تطبيق: ربط متحكمي AVR من خلال النافذة التسلسلية UART . . .

المطلوب وصل متحكمي AVR من خلال النافذة التسلسلية UART بحيث يتم إرسال أوامر تحكم بينهما على الشكل التالي: عند الضغط على المفتاح الموصول مع المتحكم MCU-A سيتم إرسال المحرف "A" من MCU-A إلى MCU-B، وعندما يستلم المتحكم MCU-B المحرف "A" سيقوم بتغيير حالة الثنائي D2. وبالمثل تماماً: عند الضغط على المفتاح الموصول مع المتحكم MCU-B سيتم إرسال المحرف "B" من MCU-B إلى MCU-A، وعندما يستلم المتحكم MCU-A المحرف "B" سيقوم بتغيير حالة الثنائي D1.



الشكل10 المخطط التمثيلي لربط المتحكمين من خلال النافذة UART



الشكل11 يبين طريقة الوصل للنافذة التسلسلية بين المتحكمين

البرنامج "Exp.18-A.bas" للمتحكم MCU-A في بيئة BASCOM-AVR:

```
'-----------------------[Definitions]
$regfile = "m8def.dat"
$crystal = 8000000
$baud = 9600

'-----------------------[GPIO Configuration]
Config Pinb.0 = Input : Switch Alias Pinb.0 : Portb.0 = 1
Config Pinb.1 = Output : Led Alias Portb.1
'
'-----------------------[Variables]
Dim Var As Byte
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

'--->[Main Program]
Do
   If Ischarwaiting() = 1 Then
      Var = Inkey()
      If Var = "B" Then Toggle Led
   End If

   If Switch = 0 Then
      Print "A" : Waitms 200
   End If
Loop
End
'---<[End Main]
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

البرنامج "Exp.18-B.bas" للمتحكم MCU-B في بيئة BASCOM-AVR:

```
'-----------------------[Definitions]
$regfile = "m8def.dat"
$crystal = 8000000
$baud = 9600

'-----------------------[GPIO Configuration]
Config Pinb.0 = Input : Switch Alias Pinb.0 : Portb.0 = 1
Config Pinb.1 = Output : Led Alias Portb.1

'-----------------------[Variables]
Dim Var As Byte
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

'--->[Main Program]
Do
   If Ischarwaiting() = 1 Then
      Var = Inkey()
      If Var = "A" Then Toggle Led
   End If

   If Switch = 0 Then
      Print "B" : Waitms 200
   End If
Loop
End
'---<[End Main]
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

8-8 حلول التطوير باستخدام منفذ الاتصالات التسلسلي USB:

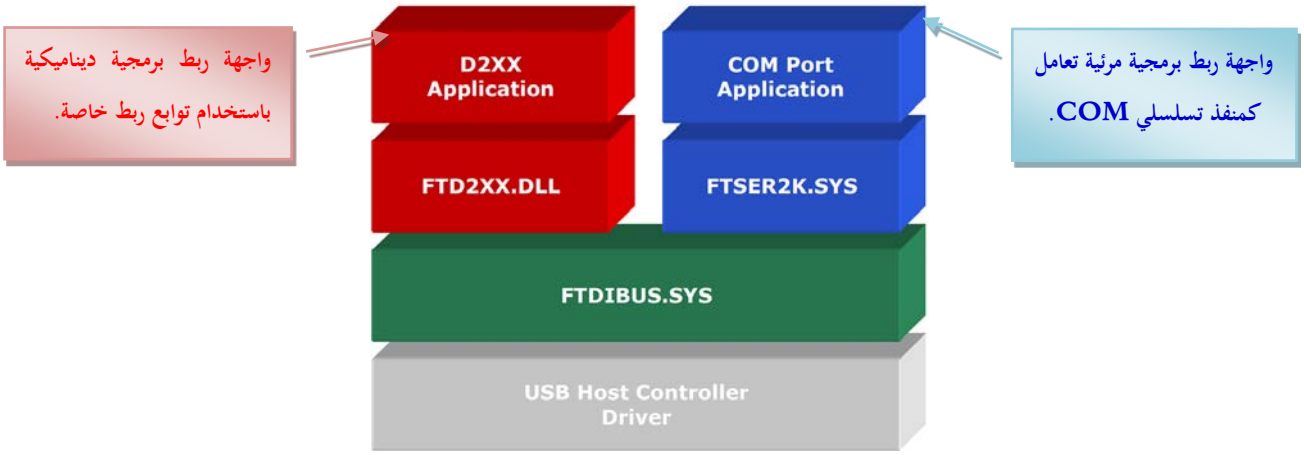
تعتبر تقنية USB في الوقت الحالي من التقنيات المعقدة حيث أن تضمين منفذ USB في النظام الإلكتروني وكتابة برنامج القيادة الخاص به على الحاسب أمر شديدة التعقيد، وذلك لأنه يتوجب على المصمم تحقيق أمرين:

1. تصميم عتاد الكتروني (Hardware) يحقق معايير البروتوكول USB.
2. كتابة برنامج التعريف الخاص بقيادة هذا العتاد.

لذلك وبسبب الطلب المتزايد على هذه التقنية واقتحامها للسوق العالمية فإن هنالك الكثير من الشركات التي وفرت على المصممين عناء تصميم العتاد الالكتروني لينصّب اهتمامهم على كتابة برامج القيادة، لذلك كل ما يتوجب على المصمم هو الاطلاع على معايير USB بغرض فهم كيفية التعامل مع هذا العتاد الالكتروني.

تقدم بعض الشركات حلولاً للتعامل مع المنفذ USB باستخدام شرائح متكاملة تقوم على تحويل البروتوكول USB إلى نافذة تسلسلية UART تمكن المستخدم من توصيل المتحكم المصغر بشكل مباشرة مع هذه النافذة، بالإضافة إلى ذلك توفر هذه الشرائح حلولاً برمجية من خلال مكتبات ربط ديناميكية من أجل ربط نظام مع الحاسب عن طريق البروتوكول USB ومعالجة بارامترات النظام أو إرسال أوامر التحكم إلى النظام. من أشهر وأكثر الشرائح انتشاراً واستخداماً هي الدارة المتكاملة FT232 التي هي عبارة عن دارة تحويل USB<>UART التي تنتجها شركة FTDI. حيث أن عملية تحويل البروتوكول USB تم بنائها في داخل هذه الشريحة ككيان صلب (Hardware) دون الحاجة إلى برمجة الشريحة، حيث تؤمن هذه الشريحة واجهتي ربط ديناميكي للتعامل برمجياً مع المنفذ باستخدام توابع خاصة وجاهزة موجودة في مكتبات الربط الديناميكي للشريحة دون الحاجة إلى بناء البروتوكول USB بشكل برمجي من البداية أو حتى فهم مبدأ عمله.

إن واجتهي الربط (D2XX driver & VCP driver) التي تؤمنها هذه الشريحة هي على الشكل التالي:



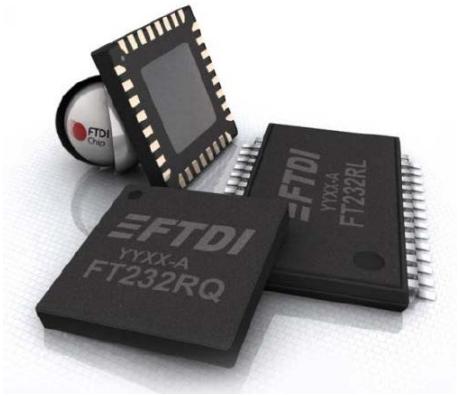
الشكل12 واجهتي الربط (نمطي العمل) للشريحة FT232R المخصصة للتحويل UART<>USB

فيما يلي جدول مقارنة بين واجتهي الربط (D2XX driver & VCP driver) للشريحة FT232R:

	VCP Driver	D2XX.DLL Driver
بساطة البرنامج	برنامج بسيط	برنامج معقد
السرعة	سرعة ثابتة لا يمكن تغيرها 300 KB/s	سرعة قابلة للتغيير تصل إلى 3MB
التحكم بالشريحة	لا يمكن التحكم بالشريحة	تحكم كامل ومباشر بالشريحة

- VCP (Virtual Com Port): يعرف منفذ USB كمنفذ COM تسلسلي إضافي، مما يسمح لنا بالتخاطب مع منفذ USB كمنفذ Com معياري.
- D2XX.DLL: يسمح هذا التعريف بالوصول المباشر إلى كامل مميزات هذه الشريحة عن طريق أوامر موجودة ضمن مكتبة ربط ديناميكية DLL.

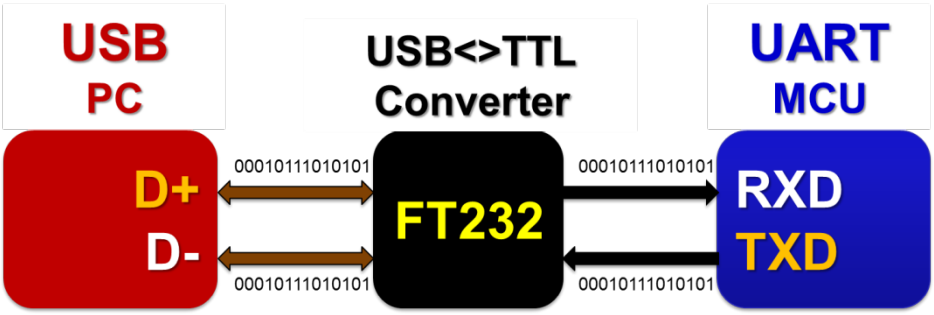
9-8 الشريحة FT232R – دارة متكاملة مخصصة للتحويل UART<>USB:



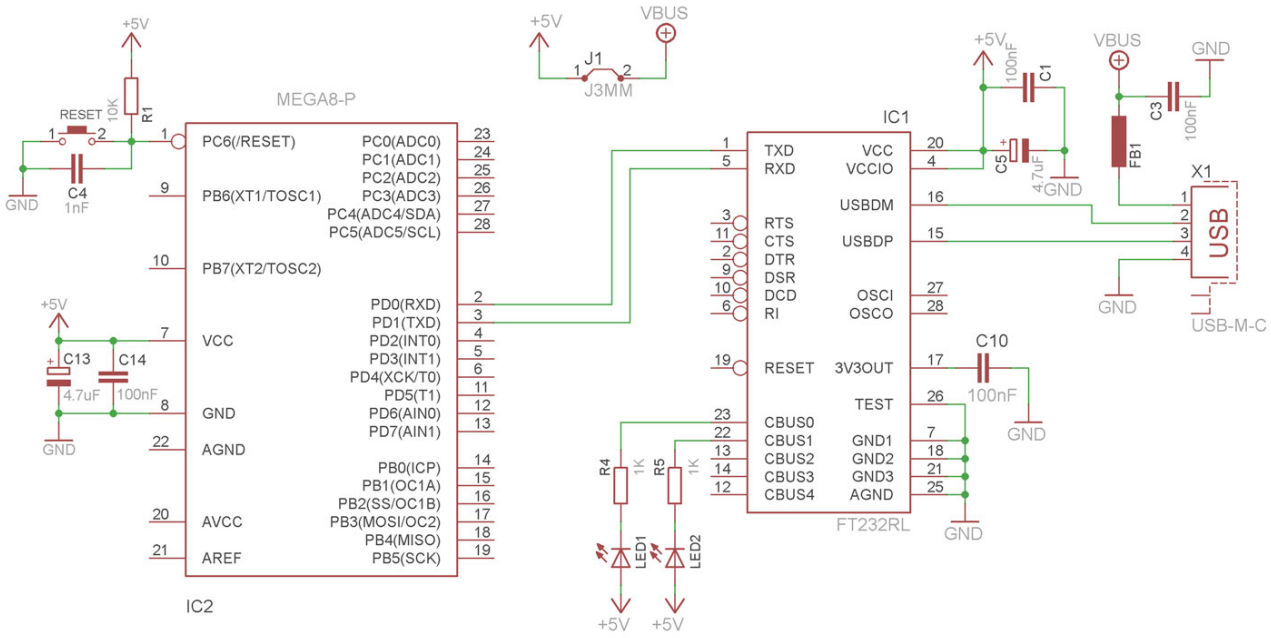
- ✓ توفر الشركة الصانعة برنامج القيادة لهذه الشريحة بشكل مجاني متوافق مع معظم أنظمة التشغيل.
- ✓ تقدم شركة FTDI برنامجي قيادة لشرائحها (VCP & D2XX.DLL).
- ✓ متوافقة مع المعيارين USB1.1, USB2.0.
- ✓ تدعم هذه الشريحة ملائمة كاملة لنظم الاتصالات التسلسلية.
- ✓ سرعة اتصال 300kb~3Mb بحسب نوع برنامج القيادة.
- ✓ ذاكرة استقبال وسيطية من نوع FIFO بطول 256 بايت.
- ✓ ذاكرة إرسال وسيطية من نوع FIFO بطول 128 بايت.
- ✓ رقمي VID, PID ورقم تسلسلي للمنتج ووصف لهذا الجهاز.
- ✓ توفر العديد من المقالات التقنية من الشركة المصنعة تقدم معلومات مفصلة عن طرق استخدام هذه الشريحة.

تلعب هذه الشريحة دور الملائم بين منفذ USB وبين النظام حيث تقوم باستقبال بيانات منفذ USB وتستخلص منها البيانات المطلوبة، كما تقوم بإرسال البيانات من المتحكم بشكلها التسلسلي إلى منفذ USB بعد إضافة الحقول اللازمة لتحقيق بروتوكول USB.

10-8 ربط متحكم AVR من خلال النافذة UART (TTL) مع منفذ USB (Differential) .

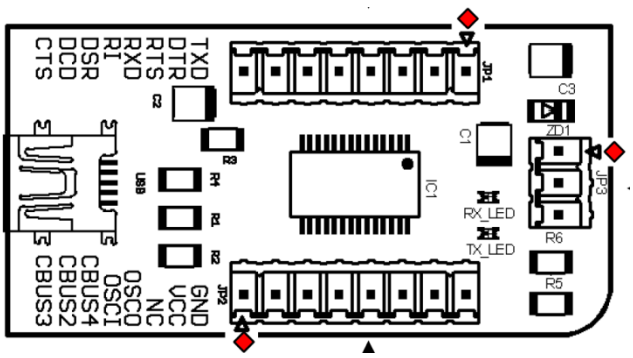


الشكل13 المخطط التمثيلي لربط متحكم AVR مع منفذ USB من خلال الشريحة FT232R



الشكل14 مخطط التوصيل (Schematic) لربط متحكم AVR مع منفذ USB من خلال الشريحة FT232R

إن التعامل فيزيائياً مع الشريحة FT232R يعتبر أمراً صعباً لعدم توفرها في غلاف فيزيائي من النوع DIP وهي فقط متوفرة كعنصر SMD، لذلك يمكن استخدام موديول التحويل UART<>USB الجاهز "Nawatt neXus" أو أي موديول آخر مشابه.



الشكل15 موديول "Nawatt neXus"

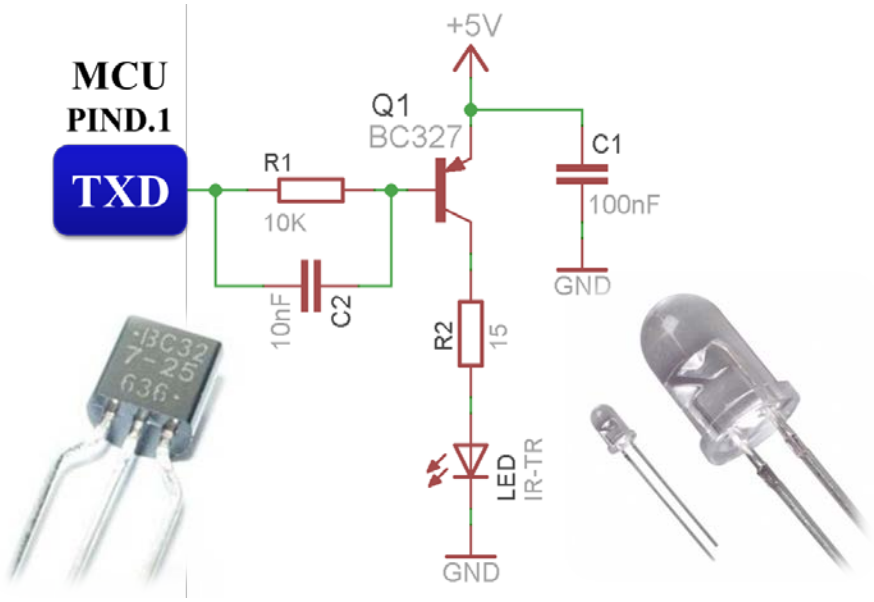
يمتلك الموديول العديد من الأقطاب، ولكن يلزمنا فقط الأقطاب التالية:

- ✓ TXD: قطب الإرسال من الحاسب.
- ✓ RXD: قطب الاستقبال من الحاسب.
- ✓ GND: قطب الأرضي 0V من الحاسب.
- ✓ +5V: قطب التغذية 5V+ من الحاسب (يستخدم فقط عندما يراد الحصول على تغذية من USB من أجل تغذية المتحكم).

11-8 تطبيق: إرسال البيانات بين متحكم AVR والحاسب باستخدام الأشعة تحت الحمراء (IR Data Link):

إن عملية إرسال البيانات بين متحكم AVR من خلال النافذة UART (TTL) والحاسب عبر منفذ USB (Differential) ستتم في هذا التطبيق باستخدام الأشعة تحت الحمراء، وبالتالي سيتضمن التصميم دارتين:

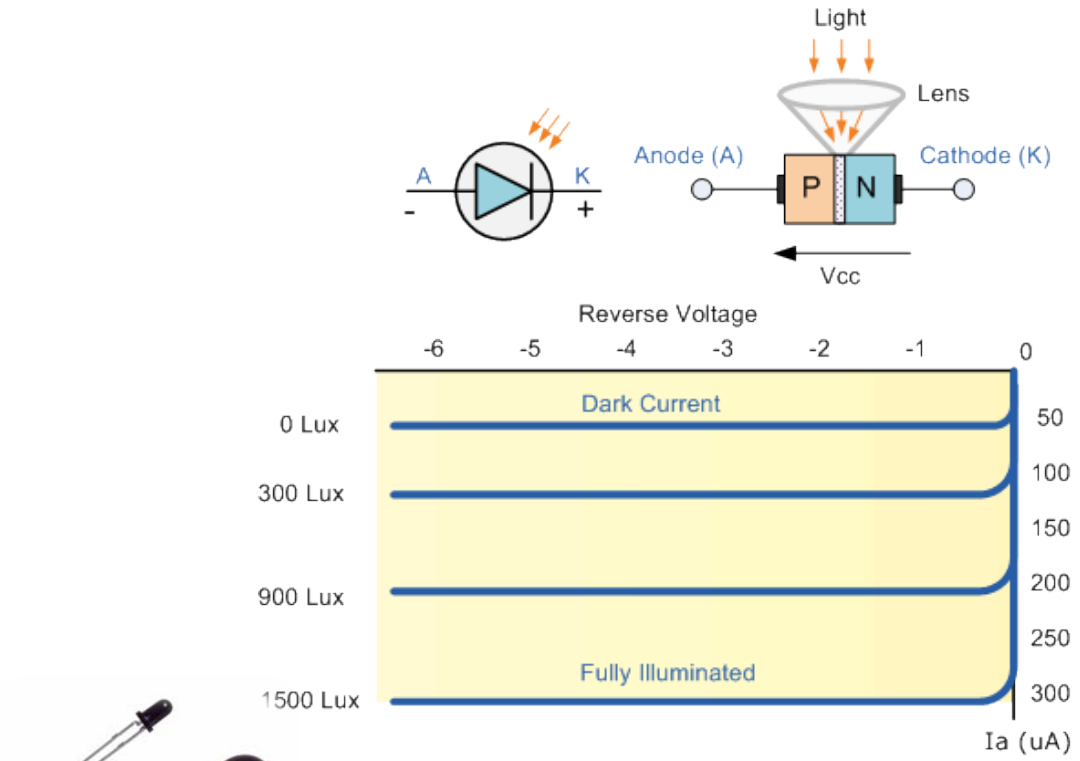
1) **دارة الإرسال للأشعة تحت الحمراء (IR Data Sender)**: وهي عبارة عن مرسل أشعة تحت الحمراء (IR LED) متحكم به عن طريق مفتاح إلكتروني ترانزستوري (Q1). إن التيار الاسمي للثنائي LED يتراوح بين 25~100mA وكلما ازدادت قيمة التيار ازدادت استطاعة الإرسال وجهد العمل للثنائي 2V (R2 = 30Ω). تم توصيل مدخل دارة الإرسال إلى قطب الإرسال للنافذة التسلسلية UART للمتحكم. وبالتالي فإن جميع البيانات الصادرة من النافذة على القطب TXD سوف ترسل على شكل ثنائي (0,1) على المرسل IR-LED.



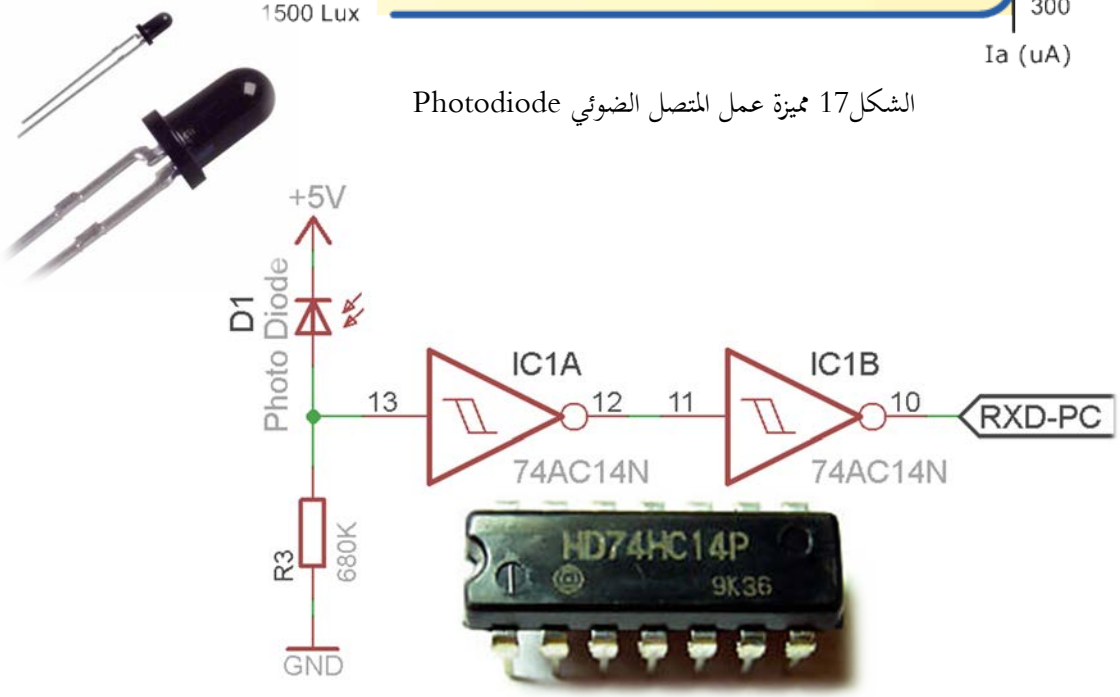
الشكل16 مخطط التوصيل (Schematic) لدارة الإرسال بالأشعة تحت الحمراء ووصلها مع القطب TXD للمتحكم

2) دارة الاستقبال للأشعة تحت الحمراء (IR Data Receiver): وهي عبارة عن متصل ضوئي (Photodiode) محيز عكسياً بحيث أنه عندما يتم تسليط ضوء على نافذة الثنائي التي تمثل المنطقة الفاصلة بين المتصل P/N يقوم على تمرير كمية أكبر من

التيار كما هو مبين على مميزة العمل في الشكل17. عندما يكون الثنائي في الظلام فإن مقاومة الثنائي تكون كبيرة جداً (بالميغا أوم)، وعندما يتم تسليط الضوء على الثنائي تصبح مقاومته بضع كيلو أوم، كما أن تغير شدة الضوء الساقط على الثنائي سيؤدي إلى تغير مطال الخرج على طرفي المقاومة R3، وبالتالي سنستخدم قادح شميث (74HC14) لتثبيت المطال بحيث تتأرجح إشارة الخرج بين القيمة "0" (عندما يرسل المرسل القيمة المنطقية "0") والقيمة "1" (عندما يرسل المرسل القيمة المنطقية "1"). الشكل18 يبين مخطط دارة الاستقبال وتوصيلها مع القطب RXD لموديول الوصل مع الحاسب neXus.



الشكل17 مميزة عمل المتصل الضوئي Photodiode

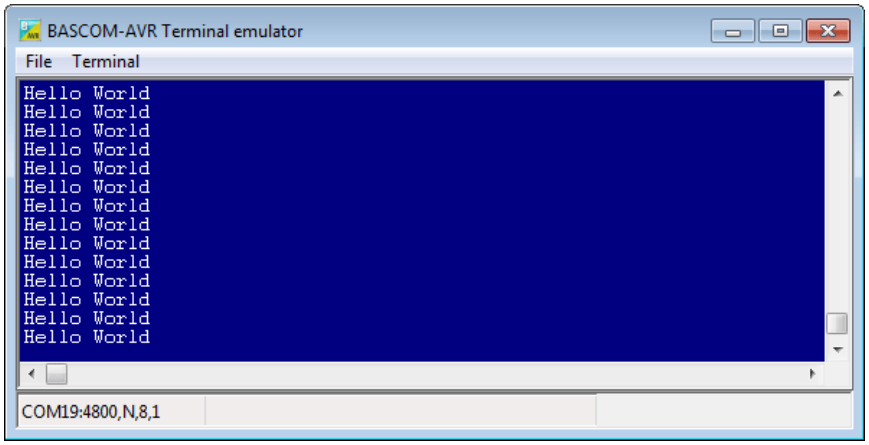


الشكل18 مخطط التوصيل (Schematic) لدارة الاستقبال بالأشعة تحت الحمراء ووصلها مع القطب RXD للموديول neXus

البرنامج "Exp.19.bas" في بيئة BASCOM-AVR:

```
'-----------------------[Definitions]
$regfile = "m32def.dat"
$crystal = 8000000
$baud = 4800
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'--->[Main Program]
Do
   Print "Hello World" : Waitms 50
Loop
End
'---<[End Main]
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

سيقوم البرنامج بإرسال (TXD) العبارة "Hello World" كل 50 ميلي ثانية على النافذة التسلسلية (UART) بشكل مستمر. على الطرف الآخر سيكون المستقبل (Photodiode) موصل مع منفذ USB من خلال الموديول neXus وبالتالي يمكن عرض القيم المستقبلة من خلال النافذة Terminal – الشكل19.



الشكل19 خرج دارة الاستقبال في نافذة Terminal في الحاسب

في حال طلب استقبال البيانات من خلال متحكم آخر بدل الحاسب وعرضها على شاشة إظهار LCD فيتم ذلك بتوصيل خرج دارة الاستقبال إلى القطب RXD للمتحكم الآخر وسيكون البرنامج على الشكل التالي:

البرنامج "Exp.20.bas" في بيئة BASCOM-AVR:

```
'-----------------------[Definitions]
$regfile = "m8def.dat"
$crystal = 8000000
$baud = 4800

'-----------------------[LCD Configuration]
Config Lcdpin = Pin , Db4 = Portc.2 , Db5 = Portc.3 , Db6 = Portc.4 , Db7 =
Portc.5 , E = Portd.3 , Rs = Portd.4
Config Lcd = 16 * 2
```

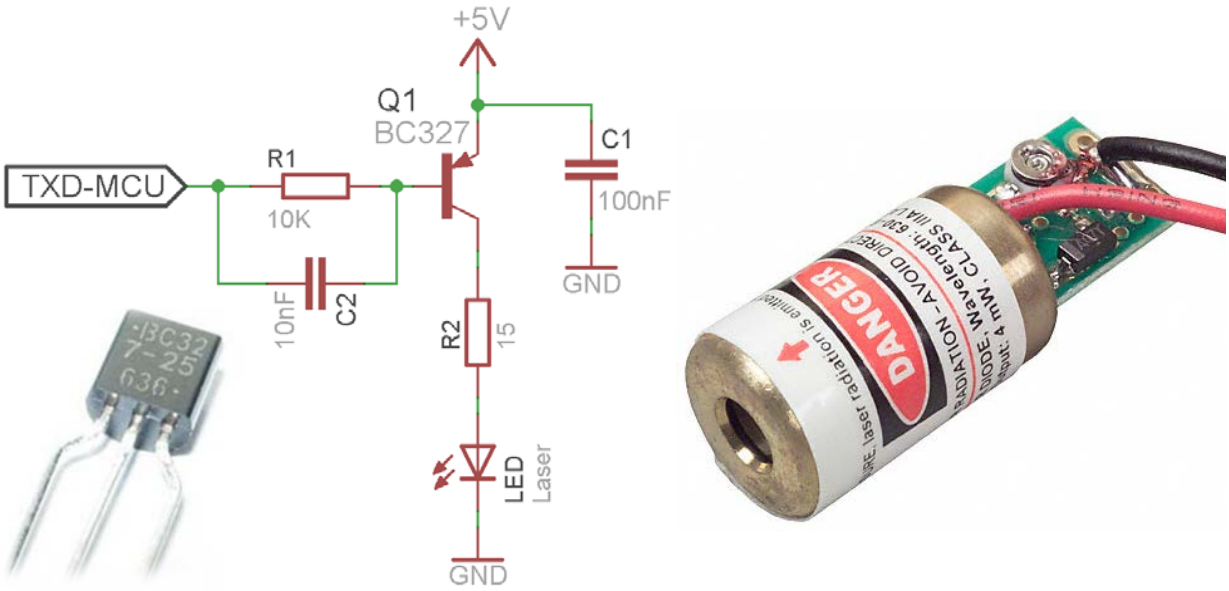
```
'-----------------------[Variables]
Dim Var As Byte
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'--->[Main Program]
Do
   If Ischarwaiting() = 1 Then
      Var = Inkey() : LCD Var
   End If
Loop
End
'---<[End Main]
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

سوف يقوم التابع "**Ischarwaiting()**" بفحص محتوى مسجل الدخل للنافذة التسلسلية (UART) وفي حال ورود بيانات سيتحقق الشرط (**Ischarwaiting() = 1**) ويتم قراءة البيانات الواردة (**Var = Inkey()**) وعرضها على شاشة LCD.

12-8 تطبيق: إرسال البيانات بين متحكم AVR والحاسب باستخدام الليزر (Laser Data Link):

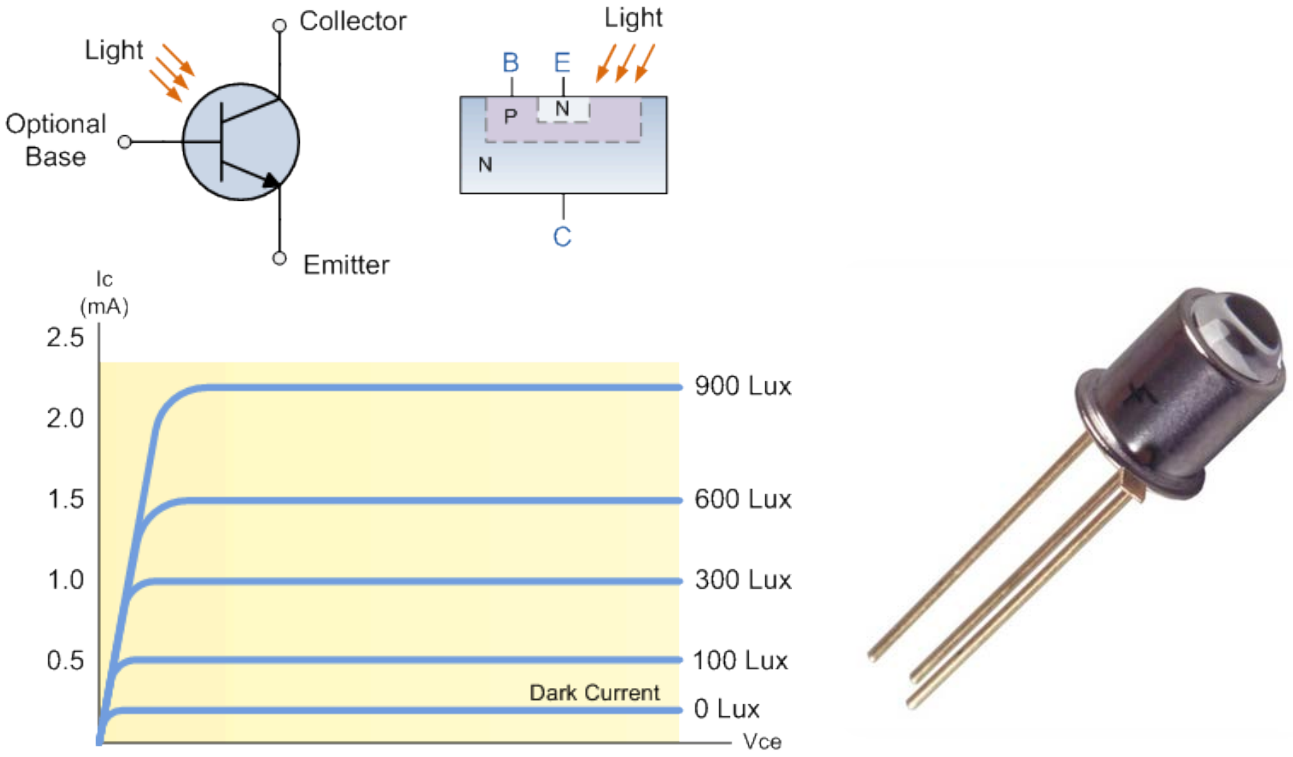
إن عملية إرسال البيانات بين متحكم AVR من خلال النافذة UART (TTL) والحاسب عبر منفذ USB (Differential) ستتم في هذا التطبيق باستخدام أشعة الليزر، وبالتالي سيتضمن التصميم دارتين:

1) **دارة الإرسال لأشعة الليزر (Laser Data Sender):** وهي عبارة عن مرسل ليزري (Laser LED) متحكم به عن طريق مفتاح إلكتروني ترانزستوري (Q1). إن التيار الاسمي للثنائي LED يتراوح بين 25~100mA وكلما ازدادت قيمة التيار ازدادت استطاعة الإرسال وجهد العمل للثنائي 2V (R2 = 30Ω). تم توصيل مدخل دارة الإرسال إلى قطب الإرسال للنافذة التسلسلية UART للمتحكم. وبالتالي فإن جميع البيانات الصادرة من النافذة على القطب TXD سوف ترسل على شكل ثنائي (0,1) على المرسل Laser-LED.

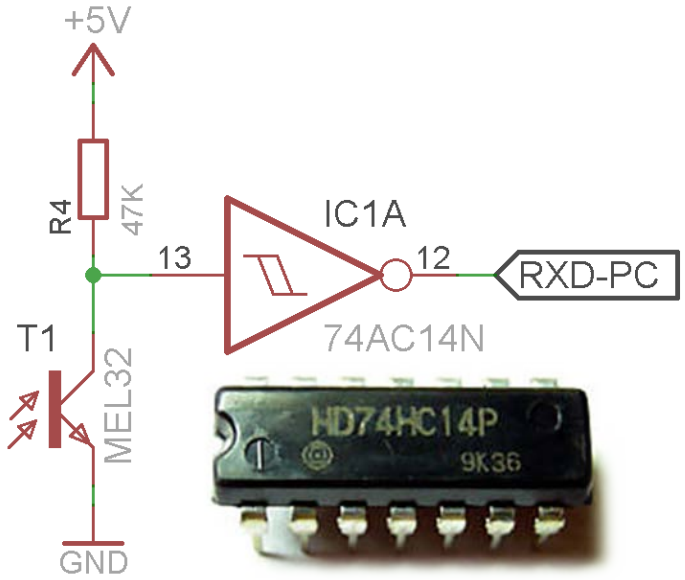


الشكل20 مخطط التوصيل (Schematic) لدارة الإرسال بأشعة الليزر ووصلها مع القطب TXD للمتحكم

2) **دارة الاستقبال لأشعة الليزر (Laser Data Receiver):** وهي عبارة عن ترانزستور ضوئي (Phototransistor) محيز أمامياً بحيث أنه عندما يتم تسليط ضوء على نافذة الترانزستور التي تمثل القاعدة فسوف يقوم الترانزستور بتمرير كمية أكبر من التيار كما هو مبين على مميزة العمل في الشكل21. عندما يكون الترانزستور في الظلام فإن مقاومة الترانزستور تكون كبيرة جداً (بالميغا أوم) وسيكون في حالة القطع، وعندما يتم تسليط الضوء سوف يفتح الترانزستور، كما أن تغير شدة الضوء الساقط على الترانزستور سيؤدي إلى تغير مطال الخرج على طرفي الترانزستور، وبالتالي سنستخدم قادح شميث (74HC14) لتثبيت المطال بحيث تتأرجح إشارة الخرج بين القيمة "0" (عندما يرسل المرسل القيمة المنطقية "0") والقيمة "1" (عندما يرسل المرسل القيمة المنطقية "1"). الشكل22 يبين مخطط دارة الاستقبال وتوصيلها مع القطب RXD لموديول الوصل مع الحاسب neXus.



الشكل21 مميزة عمل الترانزستور الضوئي Phototransistor



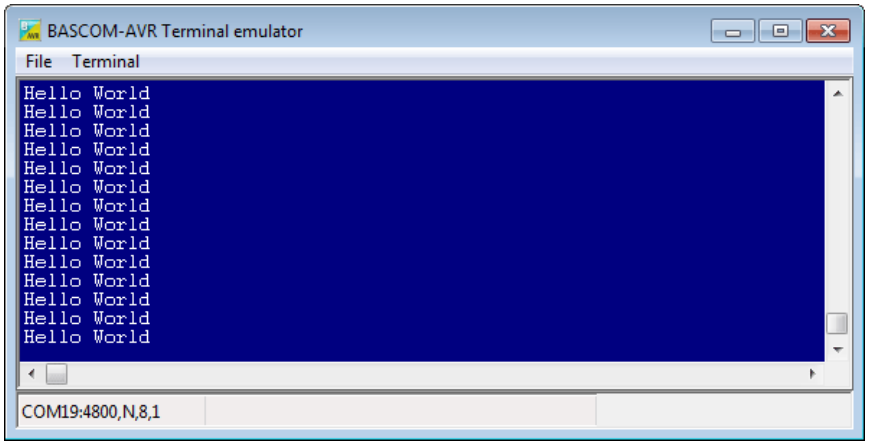
الشكل22 مخطط التوصيل (Schematic) لدارة الاستقبال بالأشعة تحت الحمراء ووصلها مع القطب RXD للموديول neXus

البرنامج "Exp.19.bas" في بيئة BASCOM-AVR:

```
'-----------------------[Definitions]
$regfile = "m32def.dat"
$crystal = 8000000
$baud = 4800
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

```
'--->[Main Program]
Do
   Print "Hello World" : Waitms 50
Loop
End
'---<[End Main]
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

سيقوم البرنامج بإرسال (TXD) العبارة "Hello World" كل 50 ميلي ثانية على النافذة التسلسلية (UART) بشكل مستمر. على الطرف الآخر سيكون المستقبل (Phototransistor) موصل مع منفذ USB من خلال الموديول neXus وبالتالي يمكن عرض القيم المستقبلة من خلال النافذة Terminal – الشكل23.



الشكل23 خرج دارة الاستقبال في نافذة Terminal في الحاسب

في حال طلب استقبال البيانات من خلال متحكم آخر بدل الحاسب وعرضها على شاشة إظهار LCD فيتم ذلك بتوصيل خرج دارة الاستقبال إلى القطب RXD للمتحكم الآخر وسيكون البرنامج هو نفسه البرنامج "Exp.20.bas".

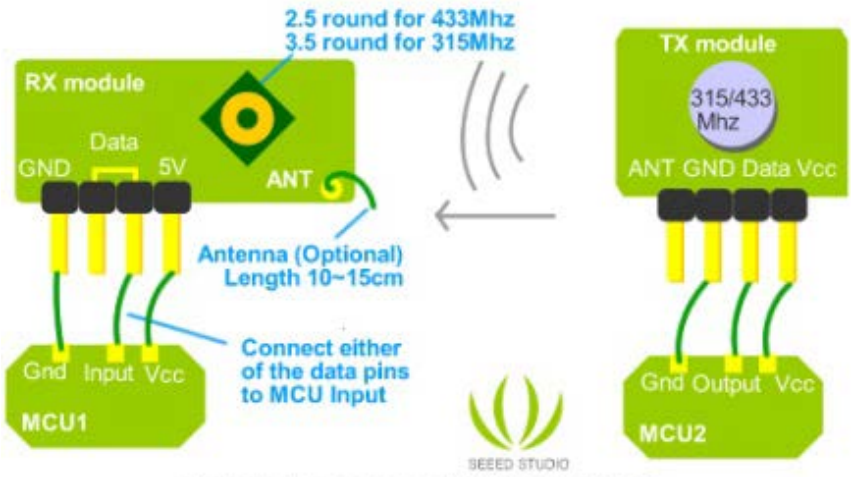
بالنتيجة فإن مشروع إرسال واستقبال البيانات باستخدام الأشعة تحت الحمراء مشابه تماماً لمشروع إرسال واستقبال البيانات باستخدام الليزر والاختلاف الوحيد هو باستبدال مرسل الأشعة تحت الحمراء بمرسل ليزري واستبدال الثنائي الضوئي بترانزستور ضوئي.

13-8 تطبيق: إرسال البيانات بين متحكم AVR والحاسب باستخدام الأمواج الراديوية (RF Data Link):

إن عملية إرسال البيانات بين متحكم AVR من خلال النافذة UART (TTL) والحاسب عبر منفذ USB (Differential) ستتم في هذا التطبيق باستخدام الأمواج الراديوية RF، وبالتالي سيتضمن التطبيق دارتين:

1) **دارة الإرسال للأمواج الراديوية (RF Data Sender)**: وهي عبارة عن مرسل راديوي (RF Transmitter) على شكل موديول جاهز يعمل بجهد 5V ويملك أربعة اقطاب (+5V, GND, ANT, DI). تم توصيل مدخل البيانات لموديول الإرسال (DI) إلى قطب الإرسال للنافذة التسلسلية UART للمتحكم. وبالتالي فإن جميع البيانات الصادرة من النافذة على القطب TXD سوف ترسل على شكل ثنائي (0,1) على قطب إرسال البيانات للموديول RF.

2) **دارة الاستقبال للأمواج الراديوية (RF Data Receiver)**: وهي عبارة عن مستقبل راديوي (RF Receiver) على شكل موديول جاهز يعمل بجهد 5V ويملك أربعة اقطاب (+5V, GND, ANT, DO).. تم توصيل مخرج البيانات المستقبلة (DO) لموديول الاستقبال إلى قطب الاستقبال للنافذة التسلسلية UART للموديول neXus .الشكل22 يبين مخطط دارة الاستقبال وتوصيلها مع القطب RXD لموديول الوصل مع الحاسب neXus.



Demostration scheme of 433/315Mhz RF kit

الشكل24 مخطط التوصيل (Schematic) لدارة الإرسال والاستقبال بالأمواج الراديوية – الDatasheet مرفقة في مجلد المشروع

البرنامج هو نفسه البرنامج "Exp19.bas"، وبالنتيجة فإن مشروع إرسال واستقبال البيانات باستخدام الأمواج الراديوية مشابه تماماً لمشروع إرسال واستقبال البيانات باستخدام الأشعة تحت الحمراء والاختلاف الوحيد هو باستبدال مرسل الأشعة تحت الحمراء بمرسل راديوي واستبدال الثنائي الضوئي بمستقبل راديوي. وفي حال طلب استقبال البيانات من خلال متحكم آخر بدل الحاسب وعرضها على شاشة إظهار LCD فيتم ذلك بتوصيل خرج دارة الاستقبال إلى القطب RXD للمتحكم الآخر وسيكون البرنامج هو نفسه البرنامج "Exp.20.bas".

14-8 تطبيق: ربط موديول GPS مع متحكم AVR من خلال النافذة UART:

يستخدم هذا الجهاز لتحديد موقع أي نقطة على الأرض من حلال مجموعة من الحسابات على البيانات المستقبلة من الأقمار الصناعية وقد تقدم أثناء عرض المحاضرة الثامنة (Session_08_CE_2012.wmv) مبدأ عمل نظام الGPS وكذلك حزم البيانات التي يتم بثها من الأقمار الصناعية المخصصة لنظام الملاحة العالمي وكيف يتم استقبالها من خلال موديول استقبال "GPS Receiver Module".

```
$GPGGA,092750.000,5321.6802,N,00630.3372,W,1,8,1.03,61.7,M,55.2,M,,*76
$GPGSA,A,3,10,07,05,02,29,04,08,13,,,,,1.72,1.03,1.38*0A
$GPGSV,3,1,11,10,63,137,17,07,61,098,15,05,59,290,20,08,54,157,30*70
$GPGSV,3,2,11,02,39,223,19,13,28,070,17,26,23,252,,04,14,186,14*79
$GPGSV,3,3,11,29,09,301,24,16,09,020,,36,,,*76
$GPRMC,092750.000,A,5321.6802,N,00630.3372,W,0.02,31.66,280511,,,A*43
$GPGGA,092751.000,5321.6802,N,00630.3371,W,1,8,1.03,61.7,M,55.3,M,,*75
$GPGSA,A,3,10,07,05,02,29,04,08,13,,,,,1.72,1.03,1.38*0A
$GPGSV,3,1,11,10,63,137,17,07,61,098,15,05,59,290,20,08,54,157,30*70
$GPGSV,3,2,11,02,39,223,16,13,28,070,17,26,23,252,,04,14,186,15*77
$GPGSV,3,3,11,29,09,301,24,16,09,020,,36,,,*76
$GPRMC,092751.000,A,5321.6802,N,00630.3371,W,0.06,31.66,280511,,,A*45
```

(reference: http://www.gpsinformation.org/dale/nmea.htm)

الشكل25 مثال عن الحزم المستقبلة على خرج موديول GPS.

في هذا التطبيق سوف نتعامل مع إحدى حزم البيانات المستقبلة من خلال الموديول وهي الحزمة $GPZDA وهي تحوي على الوقت والتاريخ فقط؛ وبالتالي من أجل استخلاص البيانات من الحزمة فإنه يجب معرفة نوع وشكل البيانات التي يستقبلها موديول الـG.P.S والتي تعتمد البروتوكول NMEA (National Marine Electronics Association) الذي يعد أشهر برتوكولات هذا النظام.

الحزمة ZDA - Time and Date:

شكل الحزمة هو: **$GPZDA,hhmmss.ss,DD,MM,YYYY,ltzh,ltzn*cs<CR><LF>**

Name	ASCII String		Units	Description	
	Format	Example			
$GPZDA	string	$GPZDA		Message ID	ZDA Protocol header
hhmmss.ss	hhmmss.ss	082710.00		UTC time	hours, minutes, seconds, seconds
day	dd	16	day	UTC time: day	01 ... 31
month	mm	09	month	UTC time: month	01 ... 12
year	yyyy	2002	year	UTC time: year	4 digit year
ltzh	xx or -xx	00		Local zone hours	Not supported (fixed to 00)
ltzn	zz	00		Local zone minutes	Not supported (fixed to 00)
cs	hexadecimal	*64		Checksum	
<CR> <LF>					End of message

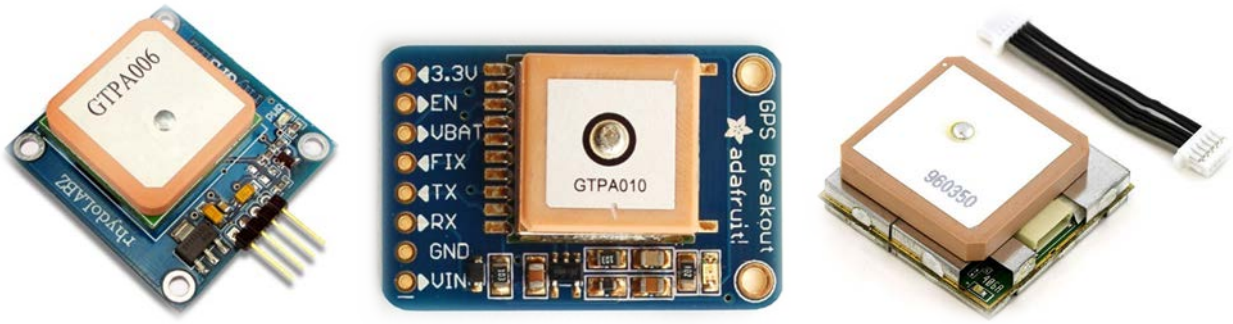
مثال: **$GPZDA,071802.00,29,10,2008,00,00*6A**

الحزمة في المثال تشير إلى أن التاريخ هو: 29/10/2008 والوقت هو: 07:18:02 بتوقيت غرينتش.

ربط موديول GPS مع المتحكم المصغر:

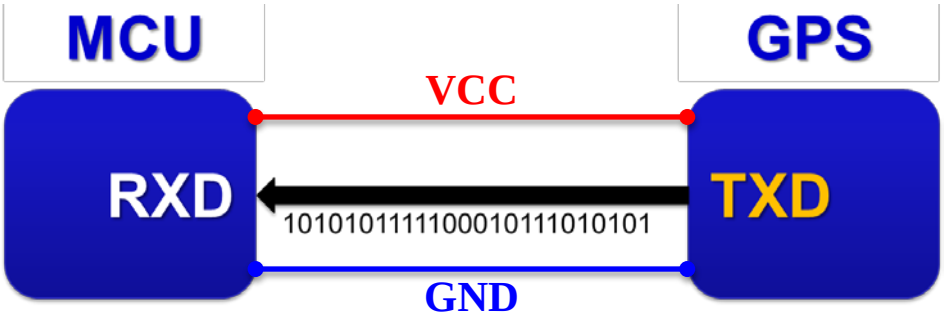
يتوفر تجارياً العديد من موديولات GPS ($20 ~ $50) وجميع موديولات الGPS تملك نافذة تسلسلية UART. بشكل عام يمتلك الموديول من 4~6 أقطاب لها الوظائف التالية:

- القطب VCC: قطب التغذية الرقمية للموديول (+3V ~ +5V).
- القطب GND: قطب النقطة الأرضية للتغذية.
- القطب TXD: قطب خرج البيانات المستقبلة من قبل الموديول (يجب أن يوصل مع القطب RXD للمتحكم المصغر).
- القطب RXD: قطب دخل من أجل ضبط بارامترات الموديول.
- القطب PPS: قطب توليد نبضة تزامن بدور 1sec.



الشكل26 بعض موديولات الGPS التجارية

من أجل هذا التطبيق فإنه يكفي تغذية الموديول (VCC; GND) ووصل القطب TXD من الموديول مع القطب RXD للنافذة التسلسلية UART للمتحكم المصغر كما في الشكل27.



البرنامج "GPS_ZDA.bas" في بيئة BASCOM-AVR:

```
' ********************************************************************************
' * Title          : GPS_ZDA.bas                                                 *
' * Target Board   : Mini-Phoenix - REV 1.00                                     *
' * Target MCU     : ATMega32A                                                   *
' * Author         : Walid Balid                                                 *
' * IDE            : BASCOM AVR 2.0.7.3                                          *
' * Peripherals    : LCD - GPS - LED - Buzzer                                    *
' * Description    : Acquiring Time/Date/Coordinates from GPS Module             *
' ********************************************************************************
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'-----------------------[Definitions]
$regfile = "m32def.dat"
$crystal = 8000000
$baud = 4800
'-----------------------
'-----------------------[LCD Configurations]
Config Lcdpin = Pin , Db4 = Portc.2 , Db5 = Portc.3 , Db6 = Portc.4 , Db7 =
Portc.5 , E = Portd.3 , Rs = Portd.4
Config Lcd = 16 * 2
'-----------------------
'-----------------------[Variables]
Dim Uart_var As Byte , Pps_f As Bit
Dim Temp_str As String * 2, Identifier As String * 6, Data_stream As String * 27
Dim Hour_val As String * 2 , Min_val As String * 2 , Sec_val As String * 2
Dim Day_val As String * 2 , Month_val As String * 2 , Year_val As String * 2
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'--->[Main Program]
Do
   If Ischarwaiting() = 1 Then Gosub Gps_isr

   If Pps_f = 1 Then
      Reset Pps_f : Cls
      Locate 1 , 1 : Lcd "Time: " ; Hour_val ; ":" ; Min_val ; ":" ; Sec_val
      Locate 2 , 1 : Lcd "Date: " ; Day_val ; "/" ; Month_val ; "/20" ; Year_val
   End If
Loop
End
'---<[End Main]
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'--->[UART]
Gps_isr:
   Uart_var = Inkey()
   If Uart_var = "$" Then
      $timeout = 100000 : Input Data_stream
      Identifier = Mid(data_stream , 1 , 6)
      If Identifier = "GPZDA," Then
         '->[Time]
         Hour_val = Mid(data_stream , 07 , 2)
         Min_val = Mid(data_stream , 09 , 2)
         Sec_val = Mid(data_stream , 11 , 2)
         '->[Date]
         Day_val = Mid(data_stream , 17 , 2)
         Month_val = Mid(data_stream , 20 , 2)
         Year_val = Mid(data_stream , 25 , 2)
         Set Pps_f
      End If
   End If
Return
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

إن جميع حزم البيانات الواردة على خرج موديول GPS تبدأ بالمحرف "$" وبالتالي فإن البرنامج "GPS_ZDA.bas" سيقوم بمايلي:

1) أولاً بانتظار ورود بيانات على النافذة التسلسلية UART حتى يتحقق الشرط (**If Iscarwaiting()=1**).
2) سوف يقوم بقراءة المحرف الوارد على النافذة (Uart_var = **Inkey()**) والتأكد فيما إذا كان المحرف هو "$" (**If** Uart_var = "$" **Then**) إشارة إلى بداية حزمة بيانات.
3) في حال كانت بداية حزمة بيانات ("$") فسيتم قراءة كامل الحزمة (**Input** Data_stream) إلى سلسلة محرفية "Data_stream" والذي تم تعريفه "Data_stream **As String** * 27" بـ27 محرف ممثلة بالمتحول "Data_stream".
4) سنحتاج الآن إلى التأكد من أن الحزمة التي تم وضعها في المتحول "Data_stream" هي حزمة البيانات "GPZDA" المطلوبة. لذلك سيتم استخدام تعليمة الاقتطاع من سلسلة محرفية (**Mid**) من أجل اقتطاع المحارف الستة الأولى وفحصها للتأكد فيما إذا كانت هي للحزمة "GPZDA".

شكل التعليمة هو: `String_var = Mid(String , Start , Num_of_char)`

حيث أن المتحول "String_var" هو الذي سيتم فيه وضع المحارف المقتطعة من السلسلة ويجب أن يكون حجمه معرفاً بحيث يتسع للمحارف المطلوب اقتطاعها. المتحول "String" هو السلسة المحرفية الأصلية المطلوب أن يتم الاقتطاع منها. المتحول "Start" هو نقطة بداية الاقتطاع. المتحول "Num_of_char" هو عدد المحارف المطلوب اقتطاعها.

5) في حال كانت المحارف الستة الأولى المقتطعة من السلسلة المحرفية هي للحزمة المطلوبة (GPZDA,)، فعندها يتم إكمال عملية تجزيئ السلسلة المحرفية من أجل الحصول على البيانات المطلوبة وهي الوقت والتاريخ حيث أن لكل قيمة موضع محدد في السلسلة المحرفية كما هو مبين أدناه بين الأقواس...

```
'$GPZDA,hhmmss.ss,DD,MM,YYYY,00,00*cs<CR><LF>
'hh(7,8) : mm(9,10) : ss(11,12) : DD(17,18) : MM(20,21) : YYYY(xx,xx,25,26)
```

6) المتحول "Pps_f" يستخدم كعلم من أجل عرض القيم الجديدة كلما توفرت على شاشة الإظهار LCD.

ملاحظة هامة: تمتلك الحزمة GPZDA مواضع ثابتة للمحارف ضمن السلسلة، أي: قيمة الثواني تتوضع دائماً في السلسلة عند المحرفين 11,12 (SS) وقيمة الشهر تتوضع عند المحرفين 20,21 (MM) وهكذا... إلا أن بعض الحزم الأخرى وأهمها الحزمة GPRMC لا تمتلك مواضع ثابتة للمحارف إذا يمكن أن تتغير تبعاً لعدد القيم بعد الفاصلة العشرية لبعض متحولات خطوط الطول والعرض. من أجل ذلك سنضع هنا فكرة برمجية من أجل استخلاص قيم الإحداثيات والوقت والتاريخ والارتفاع والسرعة من الحزمة GPRMS فيما يلي.

الحزمة RMC (RMC - Recommended Minimum Data):

شكل الحزمة هو:

$GPRMC,hhmmss.000,status,latitude,N,longitude,E,spd,cog,ddmmyy,mv,mve,mode*cs<CR><LF>

Name	ASCII String		Description	
	Format	Example		
$GPRMC	string	$GPRMC	Message ID	RMC protocol header
hhmmss	hhmmss.sss	083559.00	UTC Time	Time of position fix
status	character	A	Status	V = Navigation receiver warning A = Data valid.
latitude	ddmm.mmmm	4717.11437	Latitude	User datum latitude degrees, minutes, minutes
N		N	N/S Indicator	N=north or S=south
longitude	ddmm.mmmm	00833.91522	Longitude	User datum latitude degrees, minutes, minutes
E	character	E	E/W indicator	E=east or W=west
Spd	numeric	0.004	Speed (knots)	Speed Over Ground
cog	numeric	77.52	COG (degrees)	Course Over Ground
ddmmyy	ddmmyy	091202	Date	Current Date in Day, Month Year
mv	numeric		Magnetic variation	Not being output by receiver
mvE	character		Magnetic variation E/W indicator	Not being output by receiver
mode			Mode Indicator	See Position Fix Flags in NMEA Mode
cs	hexadecimal	*53	Checksum	
<CR> <LF>				End of message

مثال عن الحزمة:

$GPRMC,071802.00,A,4717.11437,N,00833.91522,E,0.004,77.52,14072011,,,A*57

البرنامج "GPS_RMC.bas" في بيئة BASCOM-AVR:

```
' ********************************************************************************
' * Title          : GPS_RMC.bas                                                 *
' * Target Board   : Mini-Phoenix - REV 1.00                                     *
' * Target MCU     : ATMega32A                                                   *
' * Author         : Walid Balid                                                 *
' * IDE            : BASCOM AVR 2.0.7.3                                          *
' * Peripherals    : LCD - GPS - LED - Buzzer                                    *
' * Description    : Acquiring Time/Date/Coordinates from GPS Module             *
' ********************************************************************************
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'-----------------------[Definitions]
$regfile = "m32def.dat"
$crystal = 8000000
$baud = 4800
'-----------------------
'-----------------------[LCD Configurations]
Config Lcdpin = Pin , Db4 = Portc.2 , Db5 = Portc.3 , Db6 = Portc.4 , Db7 =
Portc.5 , E = Portd.3 , Rs = Portd.4
Config Lcd = 16 * 2
'-----------------------
'-----------------------[Variables]
Dim Pps_f As Bit , Uart_byte As Byte , Pos(9) As Byte , J As Byte , I As Byte
Dim Hour_val As String * 2 , Min_val As String * 2 , Sec_val As String * 2
```

```
Dim Day_val As String * 2 , Month_val As String * 2 , Year_val As String * 2
Dim Identifier As String * 6 , Data_stream As String * 66 , Pos_i As Byte
Dim Latitude As String * 10 , Longitude As String * 11
Dim N_s As String * 1 , E_w As String * 1
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'--->[Main Program]
Do
   If Ischarwaiting() = 1 Then Gosub Gps_isr

   If Pps_f = 1 Then
      Reset Pps_f : Cls
      Locate 1 , 1 : Lcd "Time: " ; Hour_val ; ":" ; Min_val ; ":" ; Sec_val
      Locate 2 , 1 : Lcd "Date: " ; Day_val ; "/" ; Month_val ; "/20" ; Year_val
      Locate 3 , 1 : Lcd Latitude ; "  - " ; N_s
      Locate 4 , 1 : Lcd Longitude ; " - " ; E_w
   End If
Loop
End
'---<[End Main]
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'--->[UART]
Gps_isr:
   Uart_byte = Inkey()
   If Uart_byte = "$" Then
      $timeout = 100000 : Input Data_stream
      Identifier = Mid(data_stream , 1 , 6)
      If Identifier = "GPRMC," Then
         '->[Looking for ',' Positions]
         J = 1
         For I = 1 To 9
            Pos(i) = Charpos(data_stream , "," , J)
            J = Pos(i)
         Next I
         '->[Time]
         Pos_i = Pos(1) + 1 : Hour_val = Mid(data_stream , Pos_i , 2)
         Pos_i = Pos(1) + 3 : Min_val = Mid(data_stream , Pos_i , 2)
         Pos_i = Pos(1) + 5 : Sec_val = Mid(data_stream , Pos_i , 2)
         '->[Date]
         Pos_i = Pos(9) + 1 : Day_val = Mid(data_stream , Pos_i , 2)
         Pos_i = Pos(9) + 3 : Month_val = Mid(data_stream , Pos_i , 2)
         Pos_i = Pos(9) + 5 : Year_val = Mid(data_stream , Pos_i , 2)
         '->[Location]
         Pos_i = Pos(3) + 1 : Latitude = Mid(data_stream , Pos_i , 9)
         Pos_i = Pos(4) + 1 : N_s = Mid(data_stream , Pos_i , 1)
         Pos_i = Pos(5) + 1 : Longitude = Mid(data_stream , Pos_i , 10)
         Pos_i = Pos(6) + 1 : E_w = Mid(data_stream , Pos_i , 1)
         Set Pps_f
      End If
   End If
Return
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

البرنامج "GPS_RMC.bas" يعتمد نفس المبدأ في البرنامج "GPS_ZDA.bas"، إلا أننا هنا لا نعتبر موقع المحارف ثابت وإنما نوجد مواقع الفاصلة "," التي تفصل بين البيانات ونوضح هذا فيما يلي:

$GPRMC,071802.00,A,4717.11437,N,00833.91522,E,0.004,77.52,14072011,,,A*57

بالنظر إلى الحزمة السابقة فإننا سنجد أن مواقع الفواصل "," هي: [7, 17, 19, 30, 32, 44, 46, 52, 58, 67, ...]. ومن الواضح تماماً أنه بعد الفاصلة الأولى يأتي قيمة الوقت (071802) وبعد الفاصلة الثالثة يأتي قيمة خط الطول (4717.11437) وبعد الفاصلة الرابعة تأتي قيمة محدد الاتجاه (N) وبعد الفاصلة الخامسة يأتي قيمة خط العرض (00833.91522) وبعد الفاصلة السادسة تأتي قيمة محدد الاتجاه (E) وبعد الفاصلة السابعة تأتي قيمة السرعة (0.004) وبعد الفاصلة التاسعة تأتي قيمة التاريخ (14072011)...

وبالتالي تمكنا من معرفة بدايات توضع كل صنف من البيانات والآن يمكننا اقتطاعها ابتداءً من هذا العنوان وانتهاءً بعنوان الفاصلة التالية. ويتم تحديد مواقع الفواصل من خلال تعليمة البحث عن موضع محرف ضمن سلسلة محرفية المتمثلة بالتعليمة "**Charpos**". يمكن الاطلاع على بارامترات التعليمة في برنامج BASCOM-AVR/Help.

... ﴿انتهت الجلسة العملية الثامنة والأخيرة﴾ ...

- تمنياتي لكم بمستقبل مشرق وحياة طيبة كريمة هانئة -

وليد بليد