

سوف نعالج من خلال دراستنا في هذه الأطروحة جميع النقاط التي يمر بها المشروع أثناء دراسته وتنفيذها من خلال الفصول التالية :

الفصل الأول :

المقدمة و المخطط الصندوقي .

الفصل الثاني :

شرح في هذا الفصل عن لغة البرمجة الرسومية وكيفية التصميم بلغة تدفق المعطيات ، وكيفية التعامل مع برنامج ال Labview بقسمييه الرسومي و الصندوقي و تفصيل مكتباته وصولا لكيفية إجراء المحاكاة المطلوبة و كيفية الربط مع الوسط الخارجي للحاسب .

الفصل الثالث :

تتم فيه الدراسة التطبيقية للمتحكم المصغر ATMEGA 128 ، مع المميزات الأساسية له و مخطط البنية الداخلية ووصف الأقطاب ، كما نستعرض خوارزمية البرنامج Bascom-AVR والكود البرمجي.

الفصل الرابع :

شرح مفصل عن بروتوكولات الاتصال التسلسلي وكيفية استثمارها وقراءتها للحساسات المدمجة على اللوحة و تفصيل أنواعها و طرق التعامل معها مع المقارنة الشاملة بينها .

الفصل الخامس :

تعريف بلوحة التطوير phoenix AVR ، وشرح مفصل عن مكوناتها ، و كيفية استخدامها في تطبيقنا لهذا المشروع .

كما نستعرض المخططات النظرية و العملية للنظام المدروس بكمال تفاصيلها و جزئياتها ..



دراسة و تنفيذ نظام قياس إفتراضي لمعطيات لوحة التطوير Phoenix-AVR باستخدام البرمجة الرسومية Labview لتدفق المعطيات

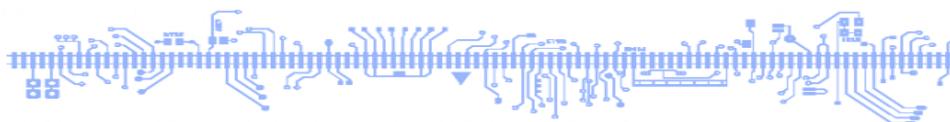
1. المقدمة :

يتسم القرن الواحد والعشرون بتسارع في وتيرة التطور التكنولوجي ، حيث يشهد العالم اليوم تفجراً معرفياً وتطوراً عاصفاً في مجالات العلوم التكنولوجية و التقنية ، إلى حد جعلت البعض يطلقون عليه عصر الثورة العلمية التكنولوجية ، إذ أنه فاق كل تصوراتنا في أبعاده وتأثيره ، وبات من الصعب جداً مواكبة التطورات المتلاحقة في عالم التكنولوجيا المتقدمة و تقنيات المعلومات والاتصالات ، إذ أصبح المدى الزمني الفاصل بين الابتكارات والاختراعات وتطبيقاتها العملية يسبق بزوج فجر يومنا هذا.

في الوقت الحاضر أصبحت المتحكمات الرقمية هي القلب النابض في الأنظمة الالكترونية بشكل عام وازدادت النظم تعقيداً في بنيتها بسبب تعقيد الوظائف المطلوبة من هذه النظم ، وبقدر ما تزداد الوظائف المطلوبة من النظام يزداد تعقيد النظام وبالتالي تعقيد الكود البرمجي للمتحكم الذي يقود هذا النظام .

لقد بات من الصعب جداً - بل من المستحيل - برمجة المعالجات و المتحكمات المصغرة بلغة التجميع (Assembly) ، وأصبح تسعار الوقت و حاجة السوق وعامل الزمن سبباً أساسياً لابتكار لغات برمجية عالية المستوى لبرمجة المتحكمات الرقمية ، وأصبحت معالجة القضايا البرمجية تحل بلغة رسومية أيضاً نشرحها من خلال بحثنا هذا .

نعرض في هذه الأطروحة ملخصاً عن المشروع الذي أعد لنيل درجة الإجازة في هندسة النظم الالكترونية ، حيث يقوم هذا المشروع على تصميم و تنفيذ نظام قياس يعتمد على استخلاص المعطيات من لوحة التطوير المخبرية (Phoenix AVR) .. حيث أن هذا النظام يقوم بوظيفتي المراقبة و التحكم (SCADA) (Supervisory Control and Data Acquisition) للحساسات التشابهية المدمجة على اللوحة وذلك عن طريق تجميع البيانات من الحساسات و إرسالها إلى الحاسوب الذي يقوم بعرض القيم وإشارات التحكم بالمشغلات الممثلة على اللوحة بثنائيات ضوئية متصلة مع أحد منافذ المعالج المدمج على اللوحة ذاتها .



ولقد تم تصميم لوحة التطوير خصيصاً بحيث تخدم المبتدئ و المتقدم في تعلم برمجة المتحكمات المصغرة من العائلة AVR، وتضم أكثر من 55 وحدة محبوطة على نفس اللوحة لتغطي ما يقارب 100 تجربة أساسية ، وقد تصل إلى أكثر من 200 تجربة بالدمج بين الوظائف المحبوطة ، إضافةً إلى إمكانية ربط وحدات خارجية عن طريق وحدات التوسعة الموزعة على أطراف اللوحة.

ومن نهاية الأسس النظرية التي تستند إلى حقائق مدروسة ومجربة ، تبدأ مرحلة التنفيذ العملي ، و بالنتيجة فإن النظام المحصل هو ثمرة البحث المعمق والفهم الدقيق لجزئيات النظام ولوحة التطوير التي تعتبر ثمرة هذا المشروع و سر نجاحه .

إن الغاية من تحقيق النظام المذكور هو استحصال عدة قيم تشابهية من الحساسات المدمجة على اللوحة و الموصولة مع المعالج وإجراء تخطاب بطريقة الوصل التسلسلي بين الحاسوب و لوحة التطوير ، حيث تتم ترجمة هذا التخطاب عن طريق تصميم واجهة رسومية بلغة البرمجة الصندوقية وتدفق المعطيات بواسطة برنامج Labview بالإضافة لعمل محاكاة كاملة للنظام مع شرح مفصل عن لوحة التطوير و مخططاتها النظرية و العملية و كيفية التعامل معها و مع ما تدعم من تجارب ..

تم تقسيم مراحل العمل على المشروع إلى مراحلتين :

1. الدراسة النظرية .. و تتتألف من أربع مراحل:

- ❖ دراسة مفصلة عن كيفية التعامل مع لوحة التطوير و مكوناتها .
- ❖ شرح مفصل عن عمل المعالجات و المتحكمات المصغرة و كيفية استثمارها للتحكم بالنظام وفق التطبيق المطلوب .
- ❖ دراسة مفصلة للتحويل التشابهي الرقمي و كيفية التعامل مع منافذ المعالج .
- ❖ دراسة مفصلة عن كيفية استثمار بروتوكولات الاتصال التسلسلي و كيفية ربط اللوحة مع الحاسوب وإجراء البرمجة الرسومية .

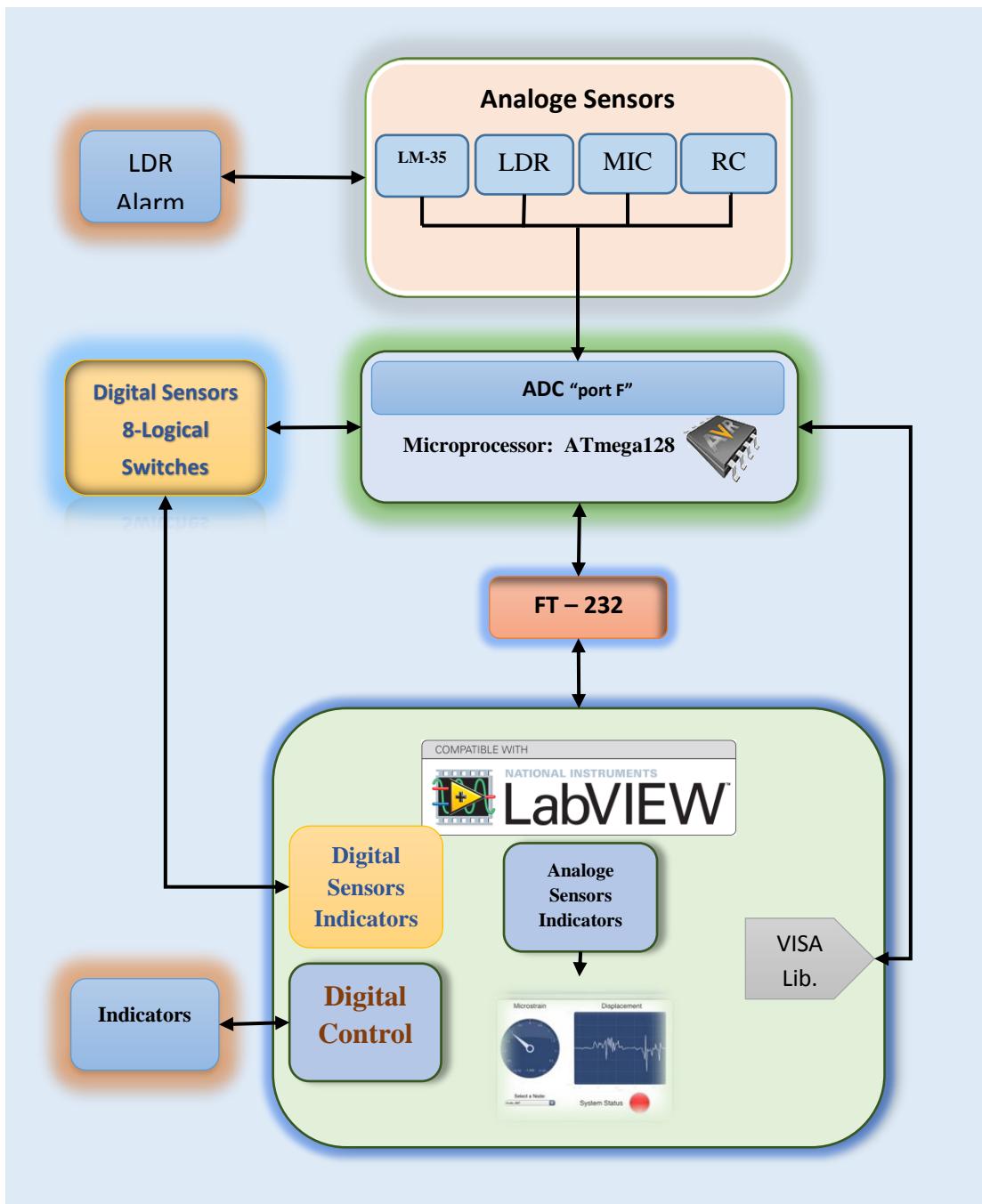
2. التنفيذ العملي ويشمل:

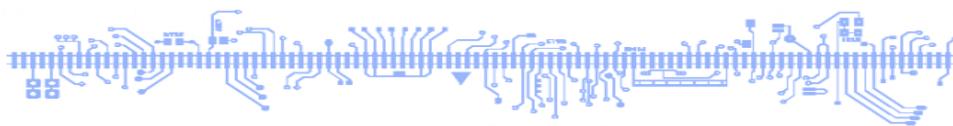
- ❖ برمجة المعالج AT-mega128 بلغة ال Bascom-AVR لضمان تأمين قراءة الحساسات التشابهية المرتبطة مع المعالج و المدمجة على لوحة التطوير ذاتها مع الأخذ بعين الاعتبار تأمين الارسال بمعدل متواافق بين اللوحة والحاسوب .
- ❖ استخدام دارة ال FT-232 المدمجة على لوحة التطوير ، و ذلك للتحويل بين بروتوكول ال RS-232 إلى البروتوكول USB الذي سيتم الوصل من خلاله إلى الحاسوب .

❖ تصميم الواجهة الرسومية باستخدام لغة البرمجة الصندوقية لتدفق المعطيات مع الأخذ بعين الاعتبار ضرورة الربط باستخدام مكتبة VISA لتأمين القراءة الصحيحة لخوارزمية التحويل للربط التسلسلي بين الحاسوب واللوحة عن طريق البروتوكول USB.

وفي نهاية الاختبارات العملية أثبتت هذا النظام فعاليته ، حيث أن جميع الاشارات المأخوذة من الحساسات التشابهية قد تم تحويلها عن طريق المبدلات التشابهية الرقمية للمعالج وتأمين وصولها لبرنامج المحاكاة عن طريق بروتوكولات الاتصال التسلسلي التي سيتم تفصيلها في هذه الأطروحة.

2. المخطط الصنديقي للنظام :





شرح المخطط الصندوقي :

تعتمد فكرة هذا المشروع على نظام استحصال المعطيات التشابهية من لوحة التطوير Phoenix .. حيث يقوم المعالج المدمج على اللوحة بإجراء القراءة الدقيقة للحساسات التشابهية والتي هي حساس الحرارة LM-35 .. حيث أن المعالج يقوم بتأمين قراءة درجات الحرارة الموجبة و السالبة باستخدام طريقة الوصل التفاضلي للحساس ، و حساس شدة الإضاءة والمتمثل بالمقاومة الضوئية LDR ، ودارة قياس الحساسية الصوتية MIC ، وأخيراً الحساس الأومي السعوي RC .. تدخل هذه القراءات التشابهية إلى المعالج عن طريق المبدلات التشابهية الرقمية ثم يتم تأمين الاتصال التسلسلي باستخدام الشريحة MAX/FT-232 و المدمجة على لوحة التطوير ذاتها وذلك لإجراء التخاطب بين بروتوكول RS 232 إلى USB .

ويتم وصل لوحة التطوير مع الحاسوب عن طريق بروتوكول الإتصال USB وذلك بعد تأمين ضبط معدلات الإرسال ذاتها بين الحاسوب والكود البرمجي للمعالج .. حيث أن الحاسوب يعتبر هذا المنفذ عبارة عن منفذ COM يتعامل معه برنامج Labview كمنفذ USB بعد تحميل مكتبة VISA للبرنامج والتي تعد المسؤولة عن إجراء الوصل التسلسلي بين الحاسوب و النظم الذي سيتم إستحصال المعطيات منه .

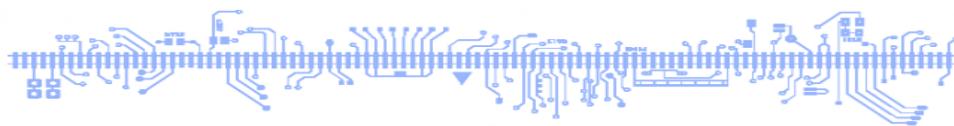
أما بالنسبة للغة البرمجة الصندوقيّة لتدفق المعطيات Labview فقد تم تصميم واجهة مكونة من قسم الحساسات التشابهية وقسم الحساسات الرقمية وقسم التحكم الرقمي حيث أن كل حساس مربوط بمؤشر إنذار على الواجهة يتم تحديد بaramيتراته من قبل المستخدم .. وبالتالي فإن النظم الكلي مكون من الأجزاء التالية :

الجزء الأول : Hardware

ويتمثل بلوحة التطوير Phoenix-AVR التي تقوم بتجميع المعلومات من الحساسات المدمجة معها بالإضافة إلى جهاز الحاسوب .

الجزء الثاني : Software

ويتمثل بلغة البرمجة الصندوقيّة لتدفق المعطيات Labview الذي يقوم بالإتصال باللوحة واستقبال البيانات وتمثيلها للمستخدم على شكل منحنيات وقيم وإصدار تقارير Reports توضح سير النظام وفق التطبيق المطلوب حيث يتم تمثيل الحساسات التشابهية عن طريق عدادات ويتم تمثيل الحساسات الرقمية عن طريق ثنائيات ضوئية مع توافر ثمانية مفاتيح في واجهة البرنامج للتحكم في ثمانية

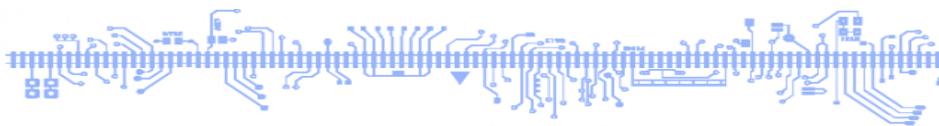


أجهزة ممثلة بثنائيات موصولة مع خرج المعالج ، بالإضافة إلى إنذارات Alarms عند حدوث خطأ معين أو تجاوز مجال القراءة للحساس وفق حد معين يحدده المستخدم .

الجزء الثالث : Communication

ويمثل وسيلة الاتصال التي تقوم بنقل المعلومات من لوحة التطوير إلى الحاسب وتعتمد على شريحة MAX232 للتحويل بين بروتوكول RS232-USB والتي سيتم شرحها بالتفصيل من خلال الفصل الرابع .





مدخل إلى البرمجة الرسومية في البيئة LABVIEW

1. لغات برمجة الكيان الصلب الرسومية (Graphical Hardware Programming Language)

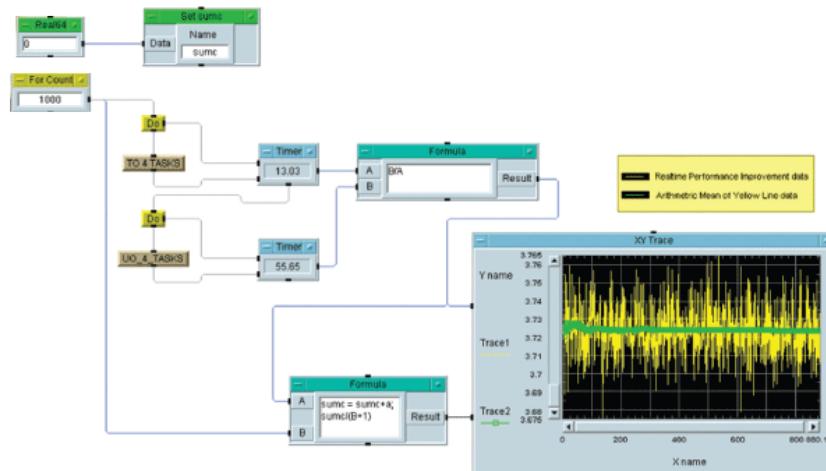
على مدى سنوات عديدة تطورت لغات البرمجة الرسومية واتسعت دائرة التطبيقات التي تشملها لتطبيقات البرمجية الحاسوبية وتطبيقات الأنظمة المدمجة وغيرها. حالياً يوجد العديد من لغات البرمجة الرسومية .. من أشهرها:

.Agilent VEE -

.National Instruments LabVIEW -

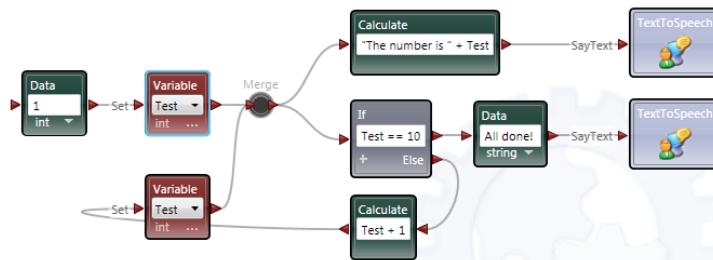
.Microsoft Visual Programming Language (MVPL) -

تعتبر البيئة "Agilent VEE" من لغات البرمجة الرسومية التدفقية التي تستخدم في تصميم تطبيقات القياسات والاختبارات المؤتمتة وتحليل البيانات، وهي تعتبر لغة برمجية رسومية موجهة للتطبيقات الصناعية ، الشكل (2,1) يبين أحد التطبيقات البرمجية باستخدام Agilent VEE Pro 9.2



الشكل (2.1) مخطط برمجي باستخدام البيئة Agilent VEE Pro 9.2

البيئة البرمجية "Microsoft Visual Programming Language" تعتبر أيضاً من لغات البرمجة الرسومية التدفقية، إلا أنها موجهة بشكل خاص لبرمجة تطبيقات الروبوت. الشكل(2,2) يبين أحد تطبيقات الأوامر الصوتية في البيئة MVPL.

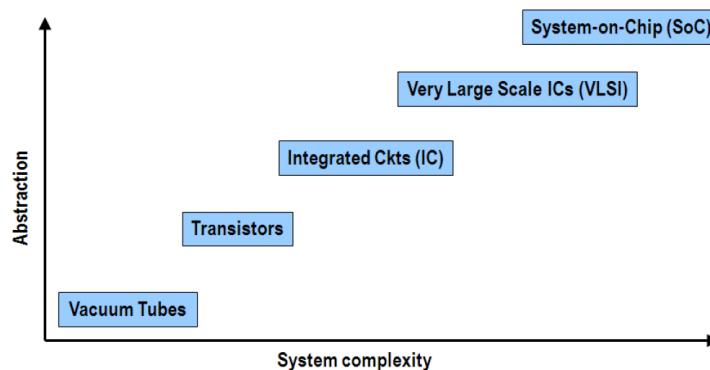


الشكل (2.2) تحويل النص إلى الصوت في البيئة البرمجية MVPL

البيئة البرمجية LabVIEW تعتبر من أقوى وأكثر البيئات البرمجية الرسومية استخداماً وانتشاراً وتطبيقاً، وهي نقطة محورية في هذا البحث سنفصل فيها في ما يأتي في هذا الفصل.

2. أهمية لغات البرمجة الرسومية (The Importance of Graphical Programming)

في الوقت الذي تزداد فيه كثافة الترانزستورات على شريحة سيليكونية وحيدة - وفقاً لقانون Moor فإن كلفة الترانزستورات على المستوى السيليكوني بانحدار، وبالتالي فإن العناصر المتكاملة المعقدة (FPGAs, Multi-core MPUs, SoCs) أصبحت أكثر استخداماً وشيوعاً في التطبيقات، وهذا بدوره أدى إلى حجم تعقيد برمجي أكبر بكثير ودورة تصميم أطول بكثير. الشكل (2.3) يبين منحنى تطور العناصر المتكاملة ودرجة تعقيد النظام.



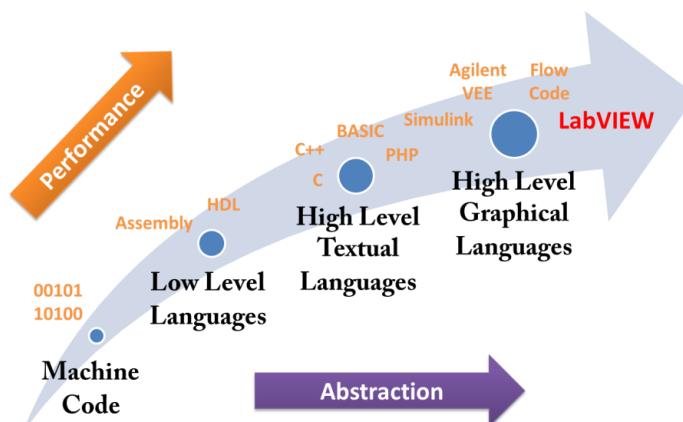
الشكل (2.3) درجة التعقيد للنظام ومستوى التجريد لتطور تقنيات الدارات المتكاملة

عند استخدام لغات البرمجة التقليدية النصية وبعد الانتهاء من كتابة البرنامج، فإن على المهندسين المصممين المرور بالعديد من المراحل المرهقة قبل توليد الملف البرمجي النهائي الذي يتم برمجته، فيجب إعادة كتابة أو تعديل البرنامج بحيث يكون قابلاً للترجمة (Synthesizable) من قبل المترجم المحدد. بالإضافة إلى أنه لكل مترجم متطلبات خاصة تختلف عن غيره من المترجمات الأخرى تبعاً للشركة المطورة للمترجم، لهذا السبب فإن المصممين أو الباحثين يصرفون وقتاً كبيراً في دراسة

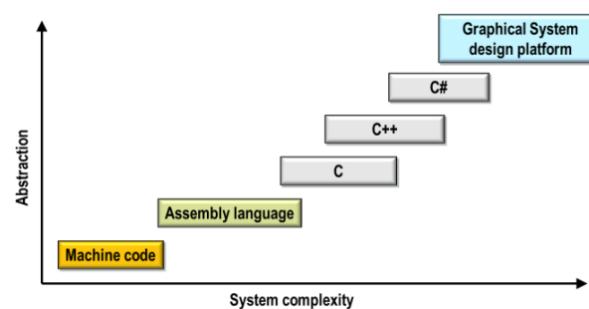
المتطلبات الخاصة للأدوات البرمجية التي سيعملون عليها بدلاً من صرف الوقت على متطلبات التصميم نفسه.

تؤكد الأبحاث على ضرورة تطوير وتبني بيئات برمجية جديدة على مستوى جديد، وذلك بعيداً عن اللغات النصية لأن حجم البرنامج يزداد طولاً وتعقيداً - مثل: البيئات الرسومية - إضافةً إلى البيئة الأساسية بلغة C حيث يمكن البرمجة بكل المنهجيات بنفس الوقت و ضمن بيئه برمجية واحدة، بما في ذلك مراحل التحليل والفحص والتنفيذ .

البحث يشير إلى أنه من أجل برمجة الأنظمة المدمجة عموماً، فإنه من الضروري جداً وجود تحول أو انتقال جذري في المنهجية البرمجية المتبعة. كما تؤكد الأبحاث على أن لغات البرمجة الرسومية مناسبة بشكل كبير لتصميم وبرمجة الأنظمة المدمجة، نظراً لارتكازها على منهجية تدفق البيانات (Dataflow). الشكل (2.4) يبين تطور الأنظمة البرمجية على المستوى البنوي. والشكل (2.5) يبين مخطط تطور اللغات البرمجية المخصصة لبرمجة الأنظمة المدمجة.



الشكل (2.4) مخطط تطور البرمجة الحاسوبية الموافق لتطور الكيان الصلب



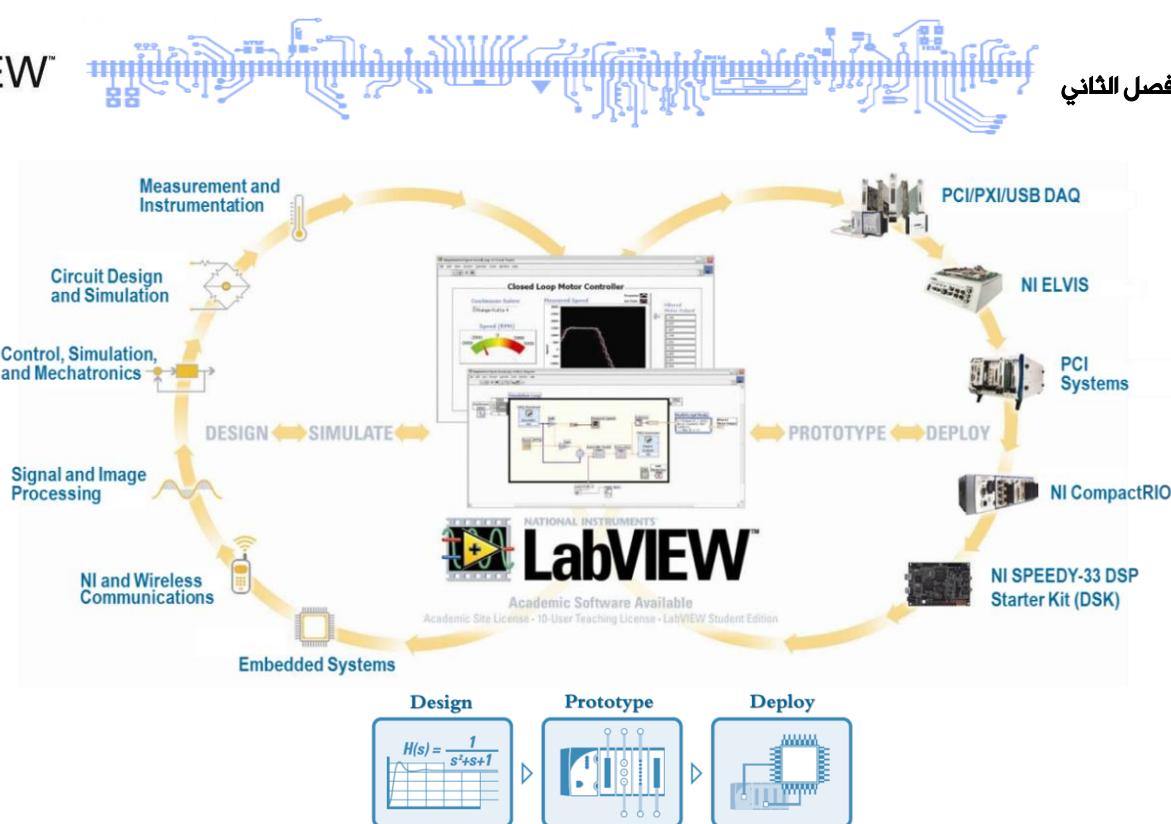
الشكل (2.5) مخطط تطور اللغات البرمجية الموافق لدرجة تعقيد الكيان الصلب

لقد أثبتت لغات البرمجة الرسومية فعاليتها على لغات البرمجة النصية، كما أنها أسرع بخمس مرات من اللغات النصية في تطوير التطبيقات. علاوةً على ذلك فإن لغات البرمجة الرسومية تعزز الإنتاجية لدى الباحثين وتطور التطبيقات بغض النظر عن مستوى خبرتهم البرمجية، وذلك لأن اللغات الرسومية تعطي تنظيماً بدھياً، وتجعل المعلومات واضحة ومرئية، الأمر الذي يجعل عملية كتابة أو تحويل الخوارزمية البرمجية من مخطط تدفق (Flowchart) إلى برنامج أمراً بدھياً.

3. البيئة البرمجية (LabVIEW Programming Environment) LabVIEW

البيئة LabVIEW – اختصاراً لـ "Laboratory Virtual Instrumentation Engineering" – عبارة عن لغة برمجية رسومية تم تطويرها من قبل شركة National Instruments Workbench في بدايات 1980 بهدف إيجاد أداة برمجية فعالة وتفاعلية لتطوير البرامج الخاصة بأنظمة استحصال البيانات وتجهيزات القياسات .

تمكن هذه البيئة البرمجية الطلاب والمهندسين والباحثين في مختلف الفروع الهندسية والمتخصصين في فروع العلوم، من التصميم التفاعلي (Design) وبناء النماذج الأولية (Prototype) والتطبيق العملي (Deploy) للأنظمة المدمجة ب مختلف تقنياتها وتطبيقاتها (MCUs, MPUs, Multi-core, FPGAs, ...) (PLC, Vision, Communications, DSPs,...) والأنظمة الصناعية والقياسات وتطبيقاتها (Control, Measurements, Mechatronics) نموذجية إضافةً إلى تضمين مكتبات أو برامج خارجية جاهزة (C, HDL, .m file) لبناء تطبيق موثوق يتم برمجته على الكيان الصلب مباشرةً كما في مشروعنا هذا دون أي مراحل تصميم كيان صلب مسبقة . الشكل (2.6) يبين بعض التطبيقات الأساسية للبيئة البرمجية LabVIEW

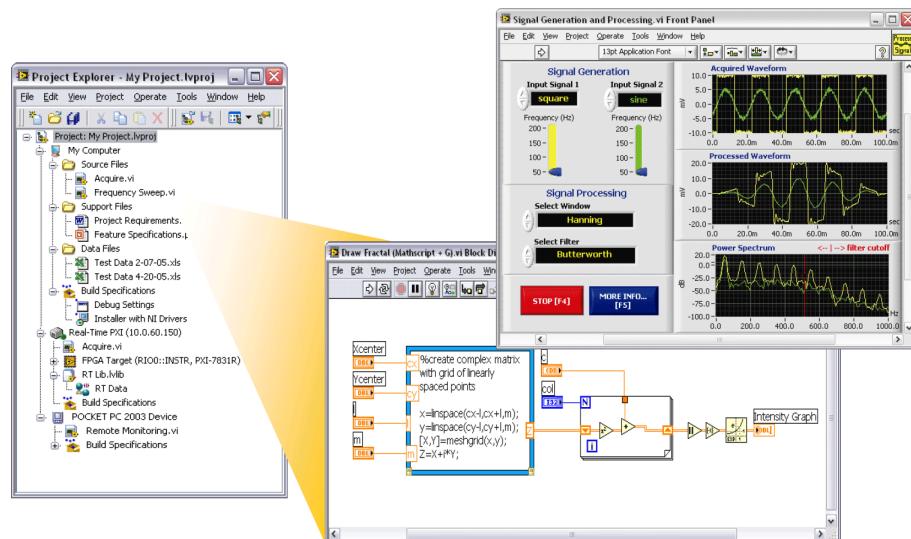


الشكل (2.6) بعض التطبيقات الأساسية للبيئة البرمجية LabVIEW

4. عناصر البيئة البرمجية LabVIEW

تتألف البيئة البرمجية الأساسية للبرنامج من:

- 1- واجهة المستخدم الرسومية (Front Panel)
- 2- واجهة البرمجة الرسومية (Block Diagram)



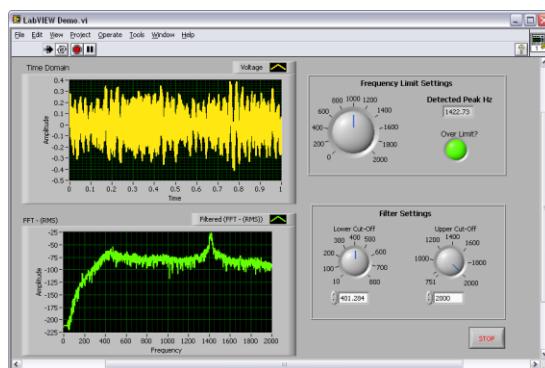
الشكل (2.7) الواجهات الأساسية والمستعرض للبيئة البرمجية LabVIEW



الشكل (2.8) مجموعة منتجات من أكبر الشركات العالمية التي تستخدم البيئة LabVIEW

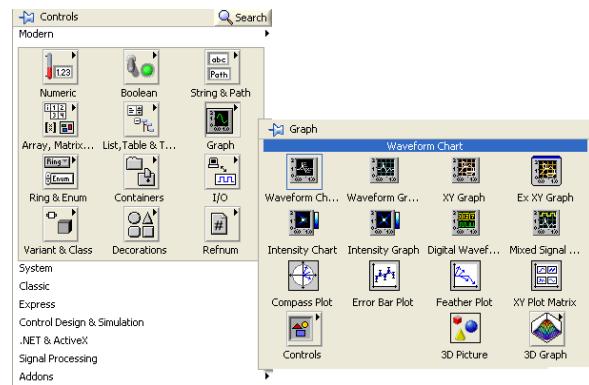
1.4. واجهة المستخدم الرسومية (Front Panel)

وهي واجهة تحكم تفاعلية مرئية للمستخدم (GUI)، تضم مجموعة عناصر تحكم وإظهار وظيفية تدعى بـ "Controls" (عناصر دخول وخرج وإظهار مرئية) يتم إضافتها من لوحة عناصر التحكم (Controls Palette). الشكل (2.9) يبين مثلاً لواجهة المستخدم للتطبيقات البرمجية في بيئة LabVIEW.



الشكل (2.9) واجهة المستخدم للتطبيقات البرمجية في البيئة LabVIEW

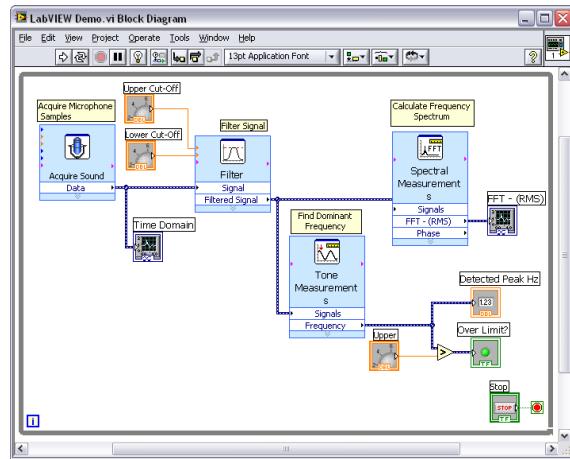
إن عناصر التحكم في واجهة المستخدم مرتبطة بشكل مباشر بالعناصر الوظيفية في الواجهة البرمجية، حيث أنه بإضافة أي عنصر في واجهة التحكم، سيتم إضافة العنصر الوظيفي له في الواجهة البرمجية آنذاك، وبالتالي يمكن بناء واجهة المستخدم بالكامل ثم توصيل عناصر التحكم الوظيفية في الواجهة البرمجية. الشكل (2.10) يبين لوحة "Controls Palette".



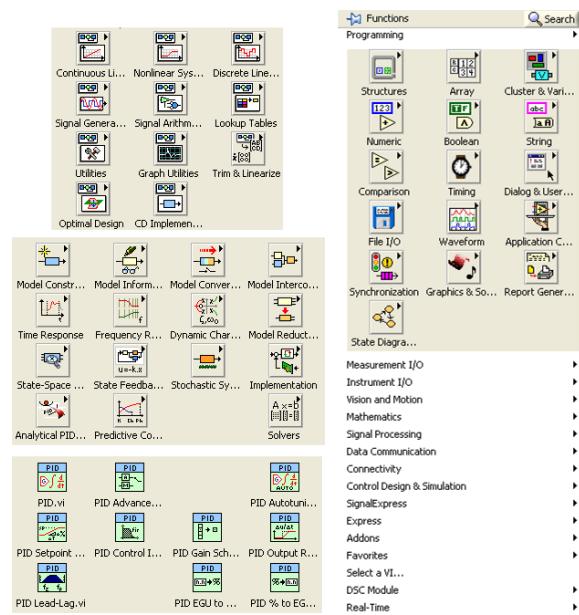
الشكل 0 (2.10) لوحة عناصر التحكم "Controls Palette" في واجهة المستخدم في البيئة LabVIEW

2.4. واجهة البرمجة الرسومية (Block Diagram)

وهي الواجهة البرمجية الرسومية (مشابهة للمحرر البرمجي النصي في لغات البرمجة التقليدية، C، C++, Java)، تضم العناصر والمكتبات البرمجية الوظيفية التي يتم إضافتها من لوحة العناصر الوظيفية (Functions Palette)، هذه العناصر والمكتبات تم بناؤها باستخدام العناصر الرسومية ولكن عند مستوى برمجي أخفض. الشكل (2.11) يبين مثلاً لوحة البرنامج للتطبيقات البرمجية في بيئة LabVIEW . الشكل (2.12) يبين لوحة العناصر الوظيفية "Functions Palette".

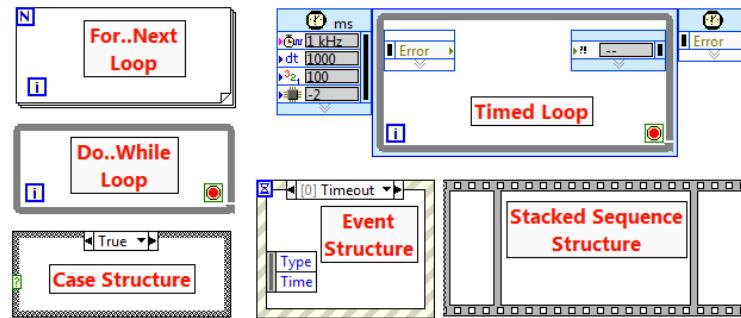


الشكل (2.11) واجهة البرنامج للتطبيقات البرمجية في البيئة LabVIEW



الشكل (2.12) لوحة العناصر الوظيفية "Functions Palette" في الواجهة البرمجية في البيئة LabVIEW

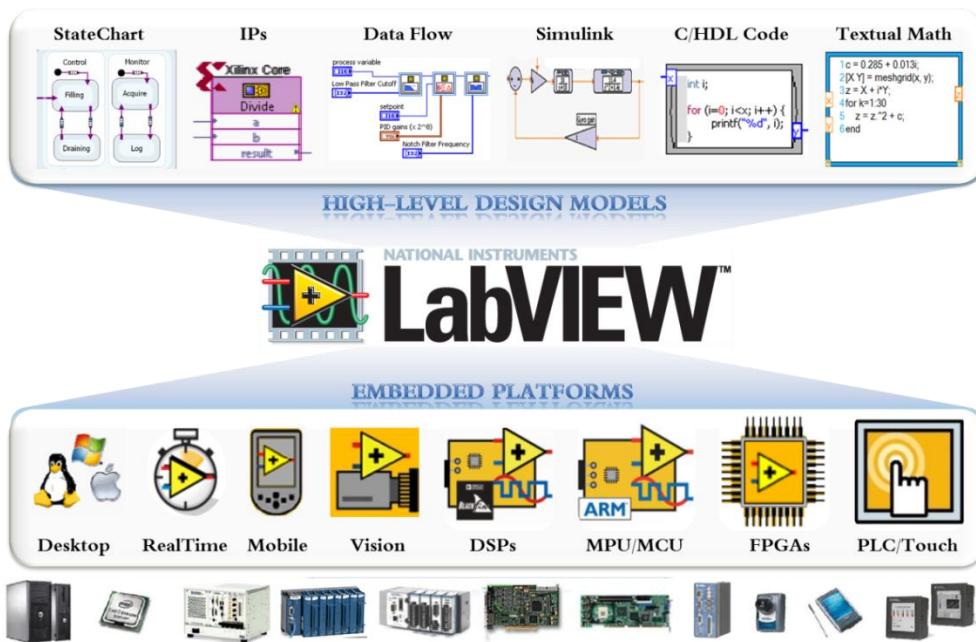
بشكل مشابه للغات النصية فإن البيئة LabVIEW تستخدم الحلقات (مثل: For..Next، Do..While) بشكل رسومي للتحكم بالعمليات التكرارية، كما تستخدم التوابع الشرطية (مثل: If...Then لمقارنة الشروط، إضافةً إلى العديد من الحلقات المتزامنة وعناصر تنفيذ متسلسل والعديد من العناصر الأخرى.



الشكل (2.13) العناصر الوظيفية للحلقات الشرطية في البيئة LabVIEW

إن التصميم والبرمجة في البيئة LabVIEW لا يقتصر فقط على استخدام لغة G الرسومية، وإنما يوجد العديد من الطرق عالية المستوى لبناء التصميم (High-level Design Models)، حيث يمكن بناء التطبيق باستخدام Matlab-Simulink المشابه لبيئة البرنامج Simulation-Module، كما يمكن استخدام ملفات برمجية نصية خارجية (HDL, C/C++, .m file)، أو تضمين ملفات برمجية نصية خارجية StateChart Module

باستخدام العقد المخصصة لذلك. الشكل (2,14) يبين طيفاً واسعاً من الطرق البرمجية التي يمكن استخدامها في تصميم وبرمجة حلول الكيان الصلب المبنية على الشكل.



الشكل (2,14) الحلول البرمجية وحلول الكيان الصلب في البيئة LabVIEW

5. (LabVIEW, "G" Dataflow Programming) تختلف بيئة LabVIEW عن معظم لغات البرمجة الأخرى في كونها تستخدم لغة برمجة رسومية تدعى بـ "G" (Graphical) تقوم على مبدأ توصيل أيقونات رسومية على شكل مخطط، الذي يترجم مباشرةً إلى لغة الآلة حتى تستطيع المعالجات الموجودة في الحاسوب تنفيذه ، وعلى الرغم من كونها تمثل بشكل رسومي عوضاً عن الشكل النصي؛ فإن لغة "G" تمتلك نفس المبادئ البرمجية المتّبعة في معظم لغات البرمجة التقليدية. على سبيل المثال، تمتلك لغة البرمجة "G" جميع البنية النظامية التي تحتويها لغات البرمجة مثل: أنماط البيانات، الحلقات، المتغيرات، العودية (recursion)، إدارة الأحداث (event handling)، والبرمجة غرضية التوجّه (object-oriented programming).

الصفة المهمة الأخرى التي تميّز البيئة البرمجية LabVIEW عن غيرها من لغات البرمجة التقليدية، هي كون لغة G المطورة فيها تُنفذ وفقاً لقواعد تدفق المعطيات (Dataflow) عوضاً عن الطريقة التقليدية الإجرائية التي تعتمد على تنفيذ عددٍ من الأوامر (الإجراءات) المتسلسلة كما في معظم لغات البرمجة النصية كلغة C++ & C.

تعتمد لغة البرمجة G على منهجية البرمجة التدفقيّة (Dataflow) التي فيها يكون خرج كل عقدة برمجية حسابية محسوب عندما تكون جميع القيم محددة على مداخل العقدة؛ حيث أن تدفق البيانات بين عقد البرنامج -وليس أسطر التعليمات المتسلسلة- هو ما يحدد أولوية التنفيذ، كما أن العمليات الحسابية يمكن أن تكون مزامنة للعقد التي لا تكون مداخلها متعلقة بمخارج عقد آخر.

ربما تبدو هذه الصفة ضئيلة الأثر للوهلة الأولى، ولكنها في الحقيقة ذات تأثير استثنائي لأنها تجعل من المسارات التي تسلكها البيانات بين أجزاء البرنامج المختلفة موضوع الاهتمام الأول للمبرمج.

تمتلك العقد (التوابع، البنى كالحلقات، البرامج الفرعية، وغيرها) في بيئة LabVIEW مدخلاً لقراءة البيانات، وحالما تحتوي جميع مداخل عقدة ما على بياناتٍ مناسبة، تقوم هذه العقدة بتنفيذ العمليات المنطقية المنسقة بها، ثم تولد البيانات المناسبة على مخارجها، وتمرر هذه البيانات إلى العقدة التالية في مسار تدفق البيانات؛ إن العقدة التي تستقبل بيانات ما من عقدة أخرى، تستطيع تنفيذ تعليماتها فقط بعد أن تنتهي تلك العقدة تنفيذ تعليماتها بشكلٍ كامل.

6. فوائد لغة البرمجة الرسومية "G": (Benefits of G Programming)

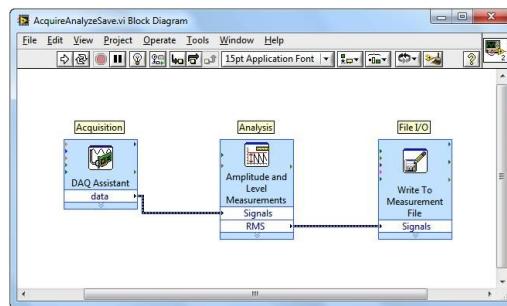
فيما يلي نلخص بإيجاز الميزات والفوائد الهامة للغات البرمجة الرسومية.

1.6. البرمجة الرسومية حدسيّة بديهية (Intuitive Graphical Programming):

إن معظم لغات البرمجة عامة الاستخدام تتطلب من المبرمج جهداً إضافياً ليتعلم التعليمات النصية المحددة الخاصة بهذه اللغة، ومن ثم عليه إسقاط بنية هذه اللغة على المسألة المدرستة.

إن البرمجة الرسومية باستخدام لغة G تتيح للمبرمج تجربة أكثر بديهية وأكثر انسجاماً مع تفكيره، ذلك لأنها تعتبر أكثر سهولةً للفهم والاستيعاب على اعتبار أنه مختلف كلياً مع التمثيل الرسومي ونمذجة العمليات بشكل مخططات منهجية أو تدفقيّة (والتي تتبع قواعد تدفق البيانات)، بالإضافة إلى ذلك، وبما أنَّ لغات البرمجة المقادمة بالبيانات تتطلب من المبرمج أن يجعل تدفق هذه البيانات المحور الرئيس في البرنامج، فإنَّ هذا يشجع المبرمج بالتفكير في المسألة التي يحلُّها عوضاً عن التفكير في أسلوب برمجتها؛ على سبيل المثال، قد يبدأ برنامجٌ نموذجيٌّ مكتوب بلغة G بتحصيل عدة قنوات تحمل بيانات عن درجة الحرارة، ثم يقوم بتمرير هذه البيانات إلى تابع معالجة وحساب، وأخيراً يقوم

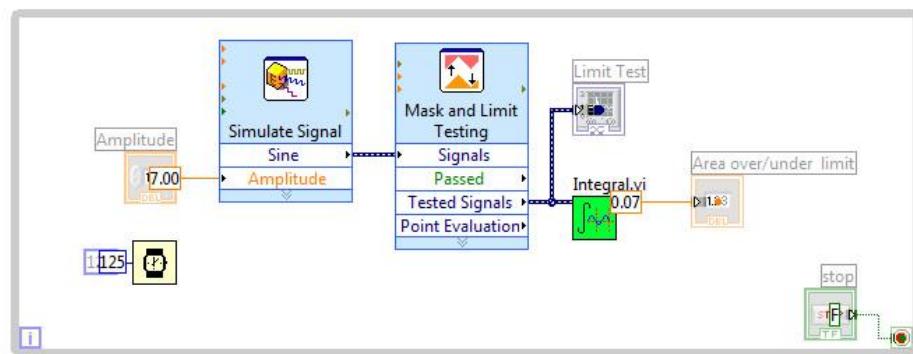
بتخزين البيانات المعالجة على القرص، كما هو مبين على الشكل (2,15) فإن تدفق البيانات والخطوات التي يتضمنها هذا البرنامج تعتبر سهلة الفهم إجمالاً ضمن مخطط بيئه LabVIEW.



الشكل(2,15) المخطط البرمجي لاستحصلان بيانات ومعالجتها وتتخزينها في البرنامج LabVIEW

2.6. أدوات التنقيح والفحص التفاعلي (Interactive Debugging Tools)

بما أن لغة البرمجة الرسومية G في بيئة LabVIEW سهلة الفهم، فإن هذا يجعل من المهام البرمجية الشائعة كتنقيح الأخطاء أمراً روتينياً وبديهياً أيضاً. على سبيل المثال، تقدم بيئة LabVIEW أدوات تنقيح فريدة من نوعها تتيح للمبرمج مشاهدة البيانات بشكلٍ تفاعلي وهي تنتقل عبر الأسلاك من عقدة لأخرى (Execution Highlighting).



الشكل(2,16) استخدام خاصية التنقيح "Execution Highlighting" لمراقبة تدفق البيانات بين العقد في البيئة LabVIEW

تحتوي البيئة LabVIEW أيضاً على أدوات تنقيح لغة G مماثلة لتلك الموجودة في بيئات البرمجة التقليدية الأخرى، تتضمن الأدوات: نقاط مراقبة (Probes)، نقاط توقف (Break Points)، تشغيل خطوة بخطوة (Step-by-Step).

تمكّن أدوات التنقيح الخاصة بلغة G المبرمج من استحصلان البيانات من عدة أجزاء في البرنامج بنفس الوقت، كما تعطيه إمكانية الإيقاف الآني والدخول إلى برنامج فرعى بدون الحاجة إلى تعليمات برمجية

معقدة ، ورغم أن هذه الإمكانيات متوفرة في لغات البرمجة الأخرى، إلا أن بيئة LabVIEW تجعل من السهل جداً تصوّر حالة البرنامج والعلاقات بين الأجزاء التفرعية فيه بسبب الطبيعة الرسومية، كما تعتبر أداة المترجم الآني إحدى أبرز أدوات تنقية الأخطاء المستخدمة في بيئة LabVIEW، حيث أنه أثناء قيام المبرمج بتطوير البرنامج، تقوم هذه الأداة بتفحص الأخطاء بشكل آني، وتقدم اقتراحات للمبرمج حول الأخطاء البرمجية وطريقة حلها.

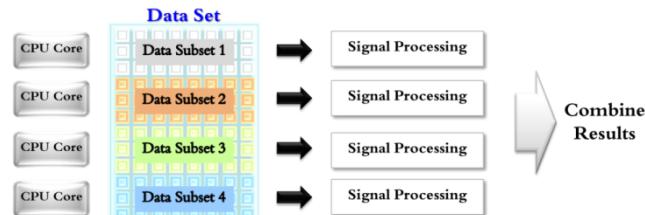
3.6. التوزيع التلقائي لمهام التنفيذ والأداء (Automatic Parallelism and Performance):

تسمح لغات البرمجة المُقادمة بالبيانات كما في بيئة LabVIEW بالحصول بشكلٍ تلقائي على تفّرعٍ في التنفيذ. وبعكس لغات البرمجة التسلسليّة كلغة C++ ولغة C، فإنَّ البرامج الرسومية تحتوي بشكلٍ أساسي على معلومات عن أجزاء البرنامج التي تحتاج إلى التنفيذ على التوازي مع أجزاء أخرى.

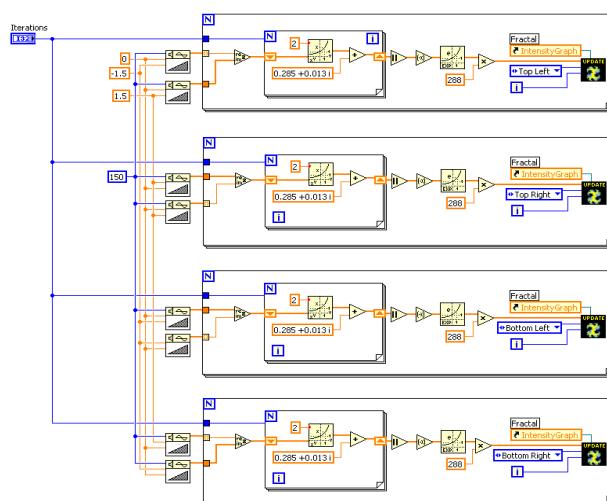
إن خاصية التفرعية تُعتبر أمراً بالغ الأهمية في برامج الحاسوب، وذلك لكونها قادرة على تخفيض حدود الأداء الناتجة عن ضعف البرامج التسلسليّة في التعامل مع التطورات الأخيرة في تصاميم معالجات الحواسيب. على مدى أكثر من 40 عام، قام مصنفو المعالجات الحاسوبية بزيادة تردد عمل المعالج لزيادة أدائه، في أيامنا هذه لم يعد هذا الأمر ممكناً نتيجةً للضوابط التي تحذر من الاستطاعة المستهلكة والطاقة الحرارية المبددة في هذه المعالجات، ونتيجةً لهذا قام مصنفو المعالجات بالانتقال إلى تصاميمٍ جديدة تستخدم عدة نوii معالجة على شريحة واحدة.

حتى يستفيد المبرمج من الأداء الكبير الذي تقدمه المعالجات متعددة النوى، يجب أن يكون قادرًا على استخدام التقنيات البرمجية المتقدمة (Pipelining, Task and Data Parallelism) وتوزيع المهام (Multithreading) في برنامجه (أي بمعنى آخر تقسيم البرنامج إلى مقاطع منفصلة يمكن أن تُنفذ بشكلٍ مستقلٍ عن بعضها البعض). وبالتالي فإنه عند استخدام لغات البرمجة النصية التقليدية، سيصبح المبرمج مسؤولاً بشكلٍ مباشر عن إنشاء المسارات وإدارتها من أجل الحصول على مزايا التفرعية، وهو ما يُعتبر تحدياً كبيراً للمبرمجين المحترفين وغير المحترفين. على العكس تماماً، فإن خواص التفرعية الطبيعية في لغة البرمجة G تبسط استخدام تعدد المهام (Multitasking) في البرامج، حيث تُقوم البيئة LabVIEW آنِيَاً أثناء التنفيذ الأجزاء التفرعية من البرنامج، وكلما صادفت تفرعاً في أحد الأسلاك، أو توضعاً متوازياً للعقد، تُنفذ البرنامج بشكلٍ تفريعي عبر استخدام عددٍ من المسارات التي تتحكم بها.

الشكل (2,17) يبيّن مبدأ توزيع البيانات المتوازي على نوى المعالجات وهذا المبدأ يستخدم عندما يراد معالجة ونقل كميات كبيرة من البيانات. الشكل (2,18) يبيّن تطبيق "Data Parallelism" على معالج Quad-core في LabVIEW.

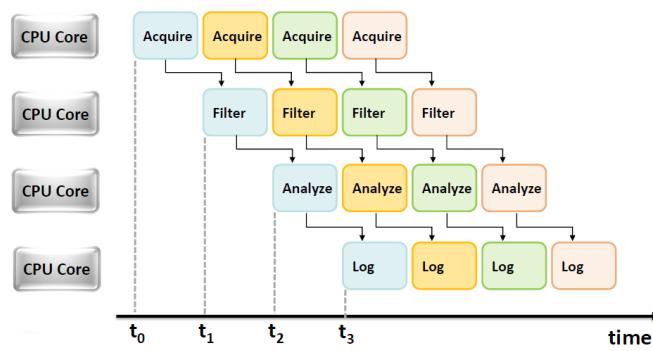


الشكل (2,17) مبدأ "Data Parallelism" على معالج Quad-core

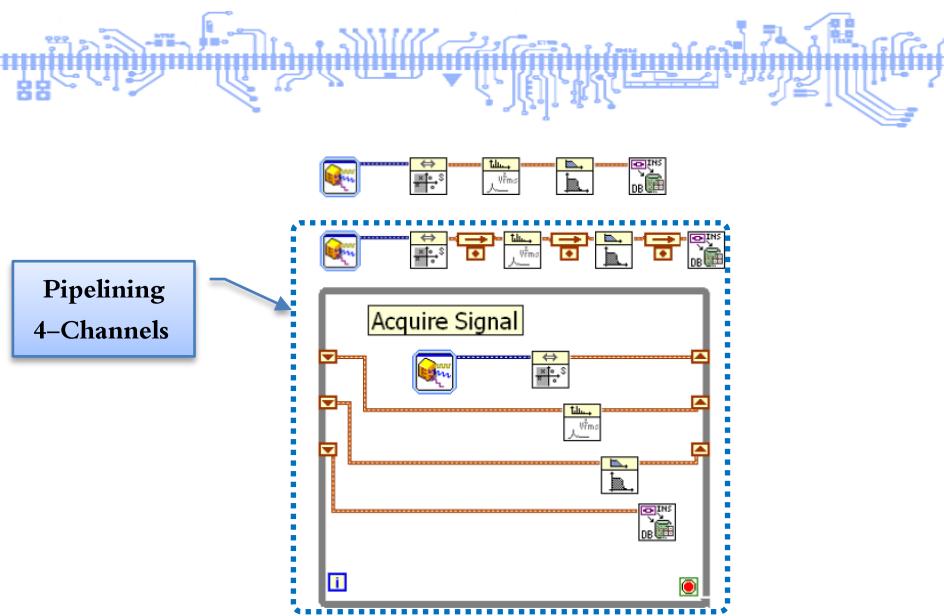


الشكل (2,18) تطبيق "Data Parallelism" على معالج Quad-core في LabVIEW

الشكل (2,19) يبيّن مبدأ المعالجة المتزامنة "Pipelining" في توزيع المهام البرمجية على معالج ذو نواة وحيدة تدعم أربع مستويات متزامنة. الشكل (2,20) يبيّن تطبيق مبدأ "Pipelining" بأربع مستويات في بيئة البرنامج LabVIEW.

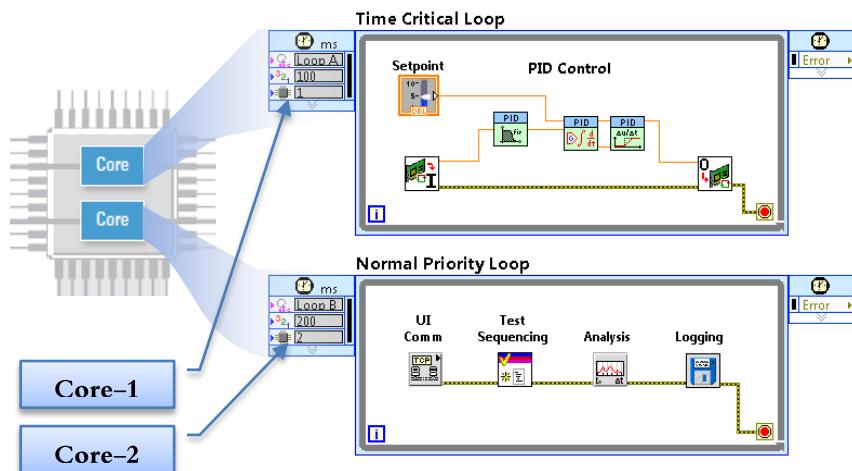


الشكل (2,19) تمثيل المبدأ العام للمعالجة المتزامنة "Pipelining" بأربع مستويات



الشكل (2.20) مقارنة بين المعالجة التسلسليّة (Sequential) والمعالجة المتزامنة (Pipelining) في بيئة البرنامج LabVIEW (4L.Pipelining)

الشكل (2.21) يبيّن البرمجة والتوزيع المتوازي في الزمن الحقيقـي (Real-time) للمعـالجات متعدـدة النوى باستخدـام الحلـقات المتـزامـنة. وفيـها يـمـكـن تحـديـد نـوـاـةـ المـعـالـجـةـ المـعـنـيـةـ بـتـنـفـيـذـ الـحـلـقـةـ فيـ إـعـادـاتـ الـحـلـقـةـ بـإـسـنـادـ رـقـمـ الـنـوـاـةـ (1,2,...n).



الشكل (2.21) البرمـجةـ فـيـ الزـمـنـ الحـقـيقـيـ لـالـمـعـالـجـاتـ مـتـعـدـدـةـ الـنـوـىـ فـيـ الـبـيـئـةـ LabVIEW

4.6. اختصار المهام والعمليات منخفضة المستوى (Abstraction of Low-Level Tasks)

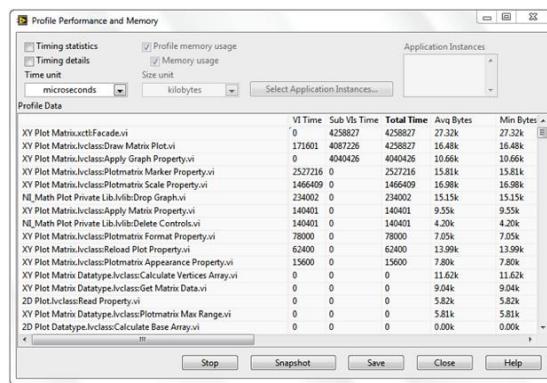
تُعد عملية الاختصار والتجريد إحدى المزايا الأساسية في اللغات عالية المستوى، حيث أنّها تعبر عن البرامج بطرق أخرى أكثر عفوية وأقرب إلى فطرة المبرمج وتفكيره. تقوم لغة البرمجة G تلقائياً بأداء الكثير من المهام التي يتوجب على المبرمج القيام بها في لغات البرمجة النصية (كالتعامل مع الذاكرة مثلاً)، حيث يتوجب على المبرمج في لغات البرمجة النصية حجز الموضع الذاكرة قبل التعامل معها، كما يتوجب عليه إنهاء حجز هذه الموضع عندما تنتهي الحاجة إليها. على المبرمج أيضاً أن يكون حذراً بحيث لا يتجاوز الموضع الذاكرة الممحورة عند الكتابة على الذاكرة. إن الفشل في حجز الموضع

المطلوبة في الذاكرة، أو حجز مساحة غير كافية، يُعد من أكبر الأخطاء الشائعة والصعبة للتنقیح في لغات البرمجة النصية.

تعتبر خاصية التعامل الآلي مع الذاكرة من أهم مزايا البرمجة باللغة G، حيث لا يحتاج المبرمج إلى حجز المتغيرات أو التصريح عنها، كما لا يحتاج إلى الكتابة إلى هذه المتغيرات أو القراءة منها، وإنما تقوم العقد التي تولد البيانات في بيئة LabVIEW تلقائياً بحجز الأماكن الذاكرة لهذه البيانات، وعندما تنتهي الحاجة إلى استخدامها يتم إلغاء حجز المواقع الذاكرة بشكل آلي. كذلك عند إضافة معلومات جديدة إلى مصفوفة أو سلسلة معرفية، يتم حجز مقدار إضافي من الذاكرة بشكل تلقائي ليتسع لهذه المعلومات المضافة.

إن رفع مهام وسائل إدارة الذاكرة منخفضة المستوى عن عاتق المبرمج، يحرره من دراسة القواعد المعقدة اللازمة لمنع حدوث أخطاء تشغيل في البرنامج، من أجل أن يركز اهتمامه على المسألة التي يقوم بحلها.

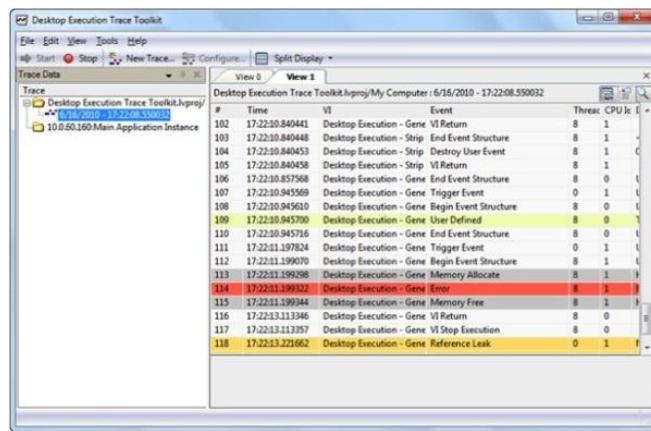
على الرغم من ذلك، فإن المبرمج يستطيع الوصول إلى تحكم دقيق باستخدام الذاكرة في اللغة G عند المستوى الأدنى باستخدام أدوات إدارة الذاكرة "Profile Performance & Memory" المدمجة ضمن بيئة LabVIEW؛ فإذا قرر المبرمج أن استهلاك الذاكرة يشكل عاملًا مهمًا في بيئة LabVIEW، يمكنه أن يتدخل ليخفض كمية الذاكرة المستهلكة عبر استخدام عدة تقنيات برمجية متقدمة. الشكل(2,22) لوحة التحكم بالأداء وموارد الذاكرة في البيئة LabVIEW.



الشكل(2,22) لوحة التحكم بالأداء وموارد الذاكرة في البيئة LabVIEW

عندما تُظهر لغة البرمجة G سلوكاً غير متوقع لا يمكن حله بسهولة باستخدام أدوات التنقیح المذكورة سابقاً، فعندها بإمكان المبرمج استخدام أدوات تنقیح أكثر تطوراً "LabVIEW Desktop Execution"

"Trace Toolkit". تقدم هذه الأدوات إمكانيات أقوى للمبرمجين المحترفين الذين يحتاجون تحليلًا ديناميكيًا للبرنامج عند مستويات منخفضة، مثل: كشف التسربات في الذاكرة، عزل المصدر المسئّب لحدث معين أو سلوك غير مرغوب، تحديد البرامج بحثًا عن المواقع التي تتمكن من تطوير الأداء، إيجاد آخر عملية نداء حصلت قبل وقوع خطأ معين، التأكّد من كون أداء برنامج معين هو نفسه على أنظمة تشغيل ومنصات عمل مختلفة. **الشكل (2.23) لوحدة الأداة "Execution Trace"**

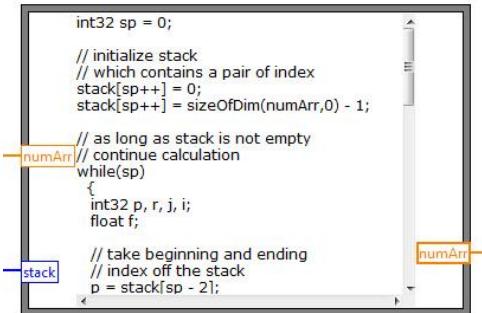


الشكل (2.23) أداة متقدمة للفحص وتتبع الأخطاء "Execution Trace" في البيئة LabVIEW

5.6. الجمع بين لغة G ولغات البرمجة الأخرى (Combining G with Other Languages)

بالرغم من أنَّ لغة البرمجة G تقدّم تمثيلًا ممتازًا للعمليات التفرعية، وتحرر المبرمج من تعقيدات فهم ذاكرة الحاسب والتعامل معها، إلا أنَّها غير مناسبة بالضرورة لأداء جميع المهام. بشكلٍ خاص، يمكن للعلاقات والصيغ الرياضية أن تمثل نصيًّا بإيجاز وسهولة أكبر في بعض الأحيان، لهذا السبب، تتيح بيئه LabVIEW إمكانية الجمع بين البرمجة الرسومية وبين عدة أنواع من لغات البرمجة النصية، إذ يستطيع المبرمج في بيئه LabVIEW الاختيار بين البرمجة النصية والبرمجة الرسومية أو الجمع بينهما.

على سبيل المثال، تتيح بيئه LabVIEW استخدام ما يُسمى بعقد الصيغ الرياضية (Formula Node)، والتي تمكن المبرمج من كتابة صيغ رياضية نصية شبيهة بتلك المستخدمة في لغة C ضمن المخطط الصندوقي للبرنامج، بإمكان تلك الصيغ الرياضية أن تتفَّذ جنبًا إلى جنب وبشكل متكامل مع الوحدات البرمجية (الرسومية) في بيئه LabVIEW. **الشكل (2.24) يبيّن العنصر C-node المخصص لكتابة برامج بلغة C/C++ ضمن بيئه LabVIEW**.



```

int32 sp = 0;
// initialize stack
// which contains a pair of index
stack[sp++] = 0;
stack[sp++] = sizeOfDim(numArr,0) - 1;

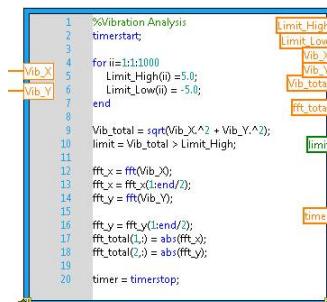
// as long as stack is not empty
// continue calculation
while(sp)
{
    int32 p, r, j, i;
    float f;

    // take beginning and ending
    // index off the stack
    p = stack[sp - 2];
    numArr
    stack
    numArr
}
    
```

الشكل (2.24) كتابة برمج بلغة C ضمن بيئه LabVIEW باستخدام العنصر البرمجي C-node

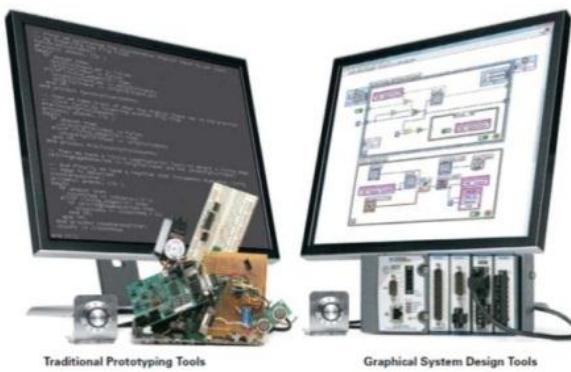
بشكل مشابه، تضيف عقدة النصوص الرياضية (MathScript Node) البرمجة النصية الرياضية إلى بيئه LabVIEW، وهي متواقة بشكل عام مع صيغة الملفات ".m file" (Matlab) شائعة الاستخدام.

الشكل (2.25) يبين العنصر MathScript-Node المخصصة للتعامل مع صيغ الملفات من النوع ".m". إضافةً إلى ذلك يمكن تضمين برنامج وصف كيان صلب "HDL file" ضمن بيئه LabVIEW. باستخدام العقدة البرمجية .HDL-Node.



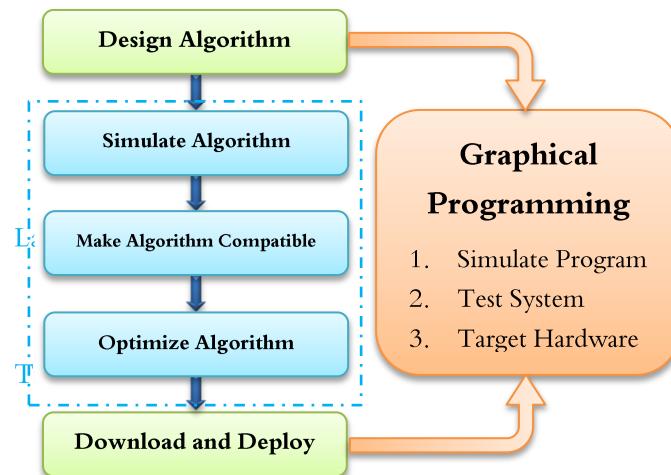
الشكل (2.25) التعامل مع الملفات ".m file" ضمن بيئه LabVIEW باستخدام العنصر MathScript-Node

7. مقارنة بين لغات البرمجة النصية والرسومية (Textual vs. Graphical Programming)

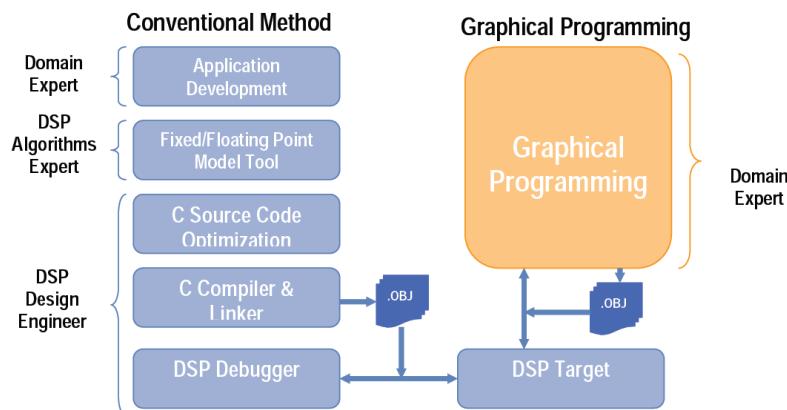


الشكل (2.26) مقارنة بين البرمجة النصية والرسومية

كما هو مبين على الشكل (2.26) فإن على المبرمج الخوض في العديد من المراحل للوصول إلى مرحلة تشغيل البرنامج على الكيان الصلب، وهذه المراحل تتضمن كل منها منصة عمل مستقلة تحتاج إلى خبرة مرتبطة بالوظيفة البرمجية. في حين أنه وباستخدام البيئة LabVIEW فإن كامل عملية البرمجة والتحليل والتطوير تتم على منصة عمل وحيدة، وأما تفاصيل ومراحل توليد الملف البرمجي للكيان الصلب فتتم بشكل مؤتمت من خلال تجريدها إلى مستوى البناء الألخضن الذي يتم آلياً.



الشكل(2.27) مقارنة الخطوات البرمجية بين لغات البرمجة النصية واللغات الرسومية - مستوى تجريد أعلى باستخدام لغات البرمجة الرسومية



الشكل (2.28) مقارنة الخطوات البرمجية بين لغات البرمجة الرسومية واللغات النصية لبرمجة شريحة DSP.

الشكل السابق يبيّن مقارنةً بين البيئة LabVIEW-DSP واللغات التقليدية النصية للخطوات المطلوبة لبرمجة تطبيق عملي لشريحة معالجات الإشارة الرقمية، باستخدام البيئة LabVIEW يمكن تصميم التطبيق بدون الحاجة إلى كون المصمم متخصص في خوارزميات معالجة وتحليل الإشارة الرقمية؛ وذلك لأن معظم هذه الخوارزميات ستكون مبنية بالكامل على شكل صناديق وظيفية في بيئة

وكذلك ما سيحتاجه المبرمج هو ضبط البارامترات الوظيفية لهذه العناصر، كما أن يحتاج المبرمج الخوض في تعقيدات توليد الملف البرمجي للكيان الصلب ومسائل تبع الأخطاء.

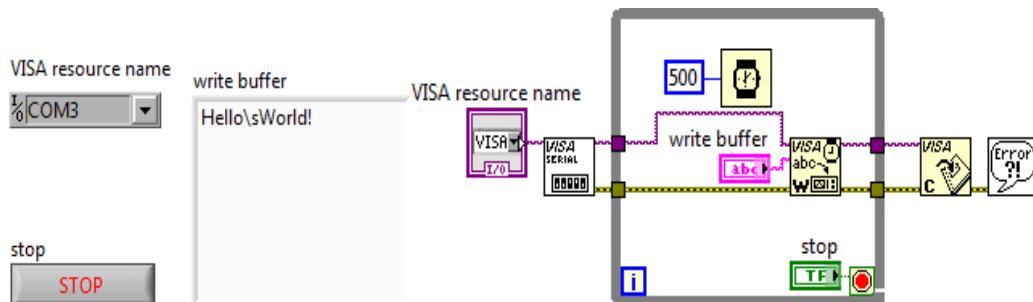
8. برمجة المنفذ التسلسلي في البيئة LabVIEW

من أجل برمجة المنفذ التسلسلي في البيئة LabVIEW فإنه يجب تنصيب الموديولات التالية:

- NI LabVIEW 2011
- NI VISA 511full
- NI-VISA_Runtime

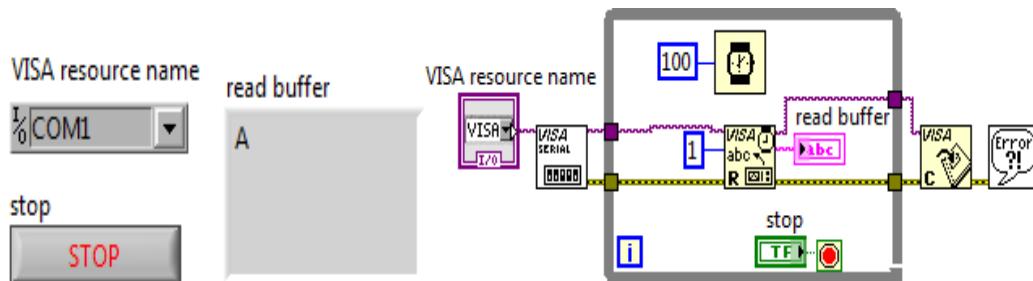
حيث تم إنشاء برنامج الكتابة (Write.vi) إلى المنفذ التسلسلي COM وبرنامج آخر للقراءة (Read.vi) من المنفذ التسلسلي COM أيضاً.

واجهة المستخدم وبرنامج الكتابة إلى المنفذ التسلسلي في البيئة LabVIEW2011



الشكل(2.29) واجهة المستخدم وبرنامج الكتابة إلى المنفذ التسلسلي في البيئة LabVIEW2011

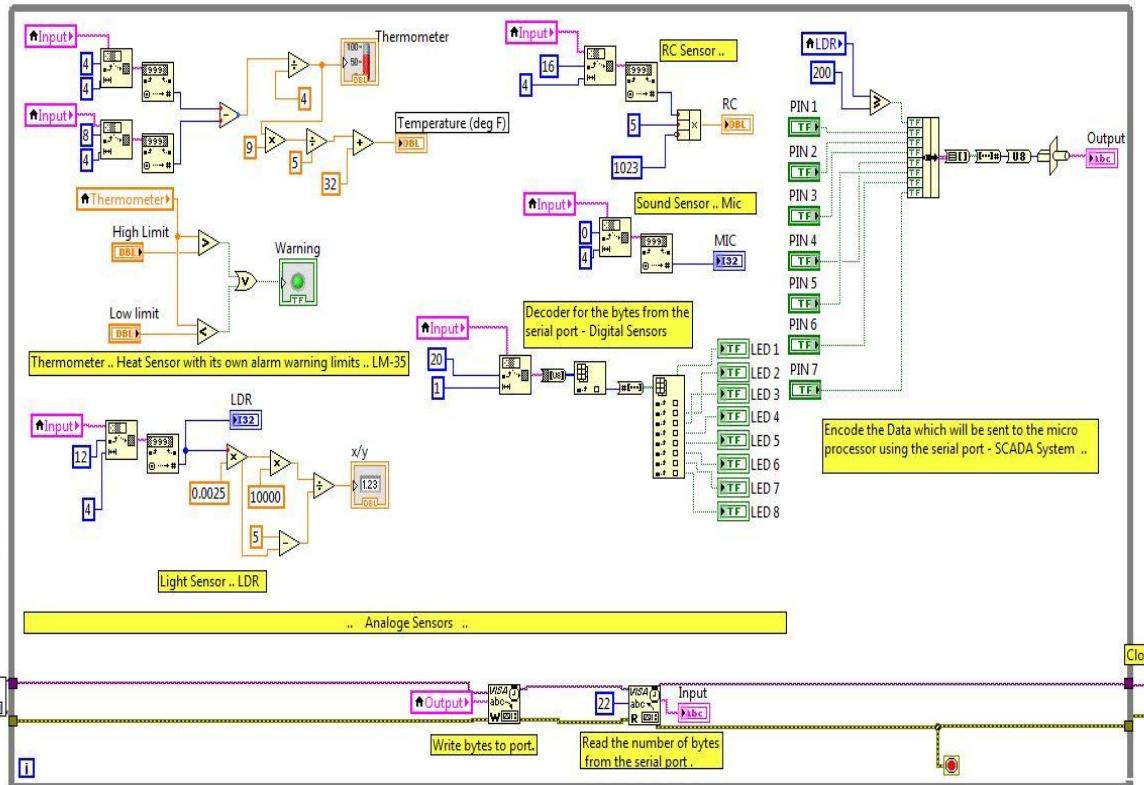
واجهة المستخدم وبرنامج القراءة من المنفذ التسلسلي في البيئة LabVIEW2011



الشكل(2.30) واجهة المستخدم وبرنامج القراءة من المنفذ التسلسلي في البيئة LabVIEW2011

٩. تصميم المخططات الصندوقية للحساسات :

بعد تأمين فتح المنفذ التسلسلي عن طريق مكتبة LabVIEW ووضع الحلقات المطلوبة للبرمجة وبتحديد باراميترات الإتصال التسلسلي (رقم المنفذ المستخدم و معدل إرسال البيانات ..) يتم في الشكل التالي (2.31) عرض مخطط واجهة البرمجة الصندوقية للمشروع ..



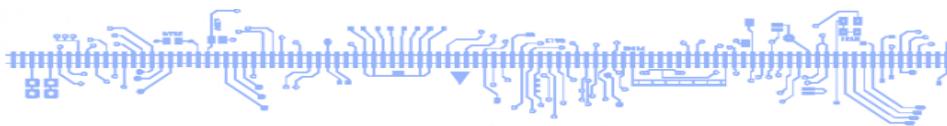
الشكل 2.31 (الجزء الأساسي من مخطط البرمجة الصندوقية

ينقسم البرنامج في الأعلى إلى ثلاثة أجزاء :

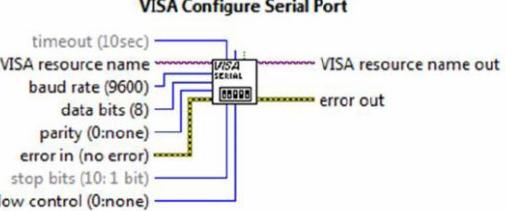
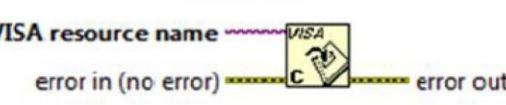
الجزء الأول :

إنشاء الإتصال وتهيئته "استقبال - إرسال" البيانات وهو الجزء الموجود بأسفل الصورة ويتم تخزين البيانات المستقبلة في متغير يسمى **Input** ويتم تخزين البيانات المراد إرسالها في متغير يسمى **. Output**.

حيث يتم إنشاء الاتصال مع أي جهاز خارجي عن طريق NI-VISA وهي عبارة عن لغة برمجة مخصصة للتalking مع الأجهزة الخارجية أيًّا كان طريقة اتصالها بالكمبيوتر ويتم تنصيبها بشكل منفصل ولإنشاء الاتصال مع المنفذ التسلسلي يلزم أربعة مكونات رئيسية ويتم الحصول عليهم من مكتبة .. (Instrument I/O)



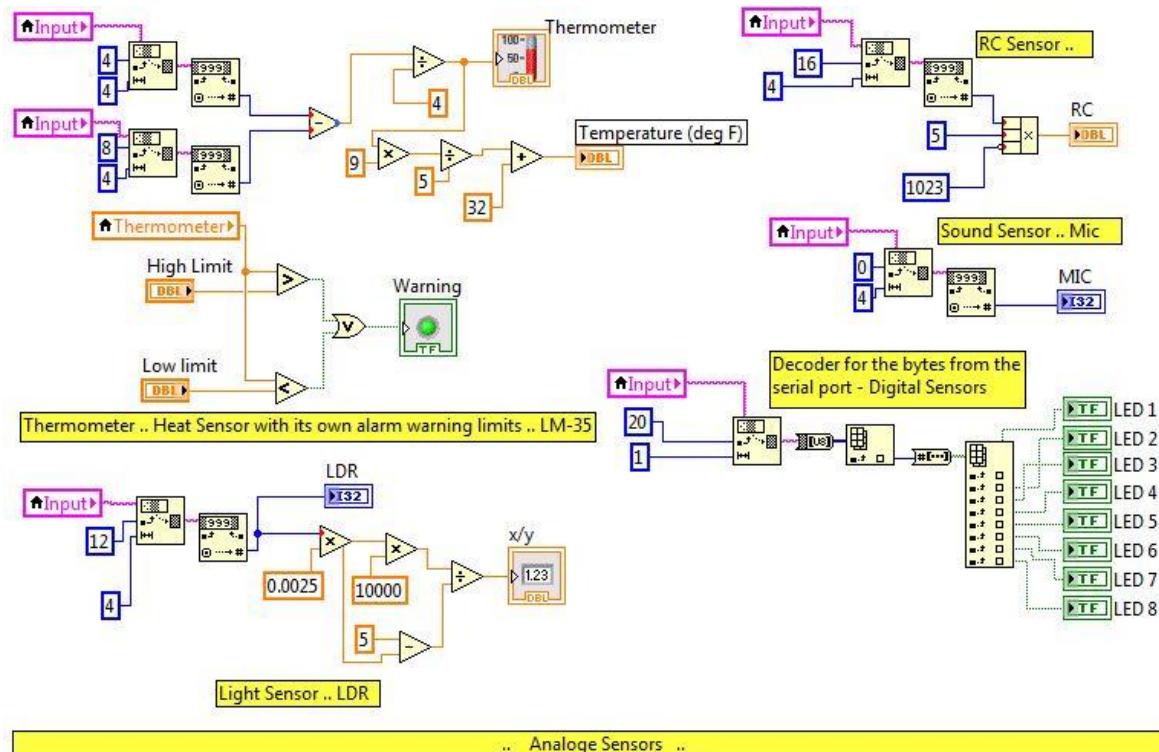
و الجدول التالي يبين الشرح التفصيلي لوظيفة و بaramيترات كل من التابع المستخدمة :

 <p>VISA Configure Serial Port</p> <p>Initializes the serial port specified by VISA resource name to the specified settings. The VISA class you wire to the VISA resource name input determines the polymorphic instance to use.</p>	<p>يقوم هذا التابع بفتح منفذ الاتصال التسلسلي .. ويتم وضعه ضمن حلقة While .</p>
 <p>VISA Write</p> <p>Writes the data from write buffer to the device or interface specified by VISA resource name.</p>	<p>يقوم هذا التابع بإرسال النص الموصى بالـ Write Buffer إلى المعالج .</p>
 <p>VISA Read</p> <p>Reads the specified number of bytes from the device or interface specified by VISA resource name and returns the data in read buffer.</p>	<p>يقوم هذا التابع باستقبال النص القادم من المعالج علىـ Read Buffer .</p>
 <p>VISA Close</p> <p>Closes a device session or event object specified by VISA resource name.</p>	<p>يقوم هذا التابع بإغلاق منفذ الاتصال التسلسلي .. ويتم وضعه بعد حلقة While وتنفذ مرة واحدة فقط في حالة تحقق الشرط وذلك في حالةـ error While .</p>

الجزء الثاني :

ويتم فيه عملية فك التشفير للبيانات المستقبلة من المعالج والمخزنة في المتغير **Input** بداخلـ Labview وذلك بتقسيمها إلى 4 عدادات للحساسات التشابهية و8 ليدات للحساسات الرقمية . حيث يتم استقبال البيانات فيـ Labview بشكل نصي يتكون من 26 حرف مرتبة حسب كود المعالج بحيث ترسل قيم الحساسات التشابهية ثم الحساسات الرقمية ويتم تفكيك النص بنفس الصورة بداخلـ Labview للحصول على قيم الحساسات في نهاية الإرسال ، و الشكل (2,32) يوضح ذلك .

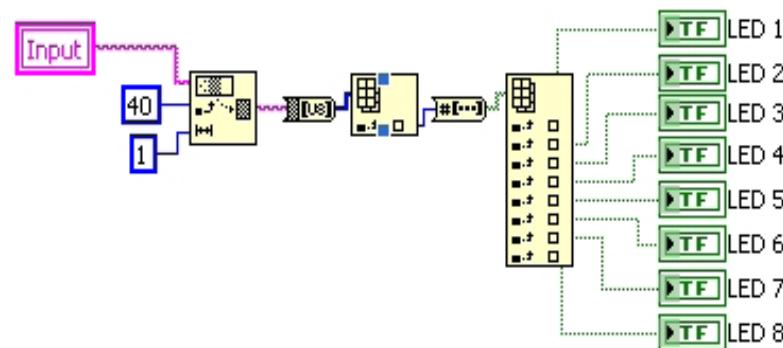
LM-35 (1)	LM-35 (2)	LDR	RC	MIC	Digital Sensor	End Charachter
4 Byte	4 Byte	4 Byte	4 Byte	4Byte	1 Byte	1 Byte



الشكل(2.32) مخطط البرمجة الصندوقية للحساسات التشابهية

String Subset Returns the substring of the input string beginning at offset and containing length number of characters.	يقوم هذا التابع بقطع جزء من النص الأساسي .. ويتحدد مكان بدء القطع حسب قيمة الـ offset ويتحدد طول النص المقطوع حسب طول الـ length Programming > String
Decimal String To Number Converts the numeric characters in string , starting at offset , to a decimal integer and returns it in number .	Character يقوم هذا التابع بتحويل الأرقام من صيغة Int .. بنفس الشكل يمكن استخدامها في عمليات الجمع والطرح . Programming > String > str/num conversion

ثم نقوم بتكرار التقاطع مع تغيير **offset** في كل مرة للحصول على قراءة حساس مختلف كما في الشكل (2.33).



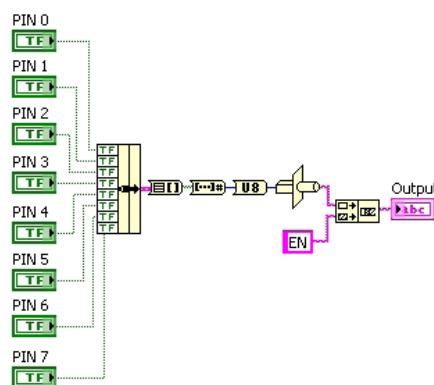
الشكل (2.33)

<p>String Subset</p> <p>Returns the substring of the input string beginning at offset and containing length number of characters.</p>	<p>تابع تقطيع النص :</p> <p>Programming > String</p>
<p>String To Byte Array</p> <p>Converts a string into an array of unsigned bytes.</p>	<p>تقوم بتحويل النص إلى مصفوفة ويتم تخزين الحرف كعنصر في المصفوفة (مثال : لو كان النص يتكون من ثلاثة حروف .. ستحتوي المصفوفة على ثلاثة عناصر مع العلم بأن القيم تخزن بداخل المصفوفة بالصيغة Decimal ، حرف الـ A سيسجل داخل المصفوفة 65).</p> <p>Programming > String > conversion</p>
<p>Index Array</p> <p>Returns the element or subarray of n-dimension array at index.</p>	<p>يستخدم هذا التابع للحصول على قيمة عنصر معين داخل المصفوفة و سيكون الخرج منها . Decimal</p> <p>Programming > Array</p>
<p>Number To Boolean Array</p> <p>Converts an integer or fixed-point number to a Boolean array.</p>	<p>تقوم بتحويل البيانات من صورة Decimal إلى صورة Binary (مثال: سيتم تحويل عنصر 255 إلى مصفوفة لها 8 عناصر وقيمة كل عنصر true وهو ما يمكن استخدامه في إضافة ليد ويتم استخدام التابع مرة أخرى لاستخراج عناصر المصفوفة .</p> <p>Programming > Numeric > Conversion</p>

الجزء الثالث :

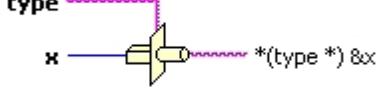
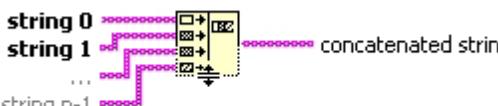
ويتم فيه تشفير البيانات المراد إرسالها إلى المعالج حيث أننا نريد إرسال قيمة 8 مفاتيح فيتم تشفيرهم لما يقابلهم من ASCII Code ويتم تخزين قيمته في المتغير Output ومن ثم توصيلها بـ VISA Write .

سنقوم الآن بتجميع قيمة 8 مفاتيح في Byte حيث تمثل قيمة كل مفتاح 1Bit وأيضاً إلهاق رمز مثل نهاية الارسال والذي تم تحديده في كود المعالج .. و الشكل التالي يوضح ذلك :



(2.34) الشكل

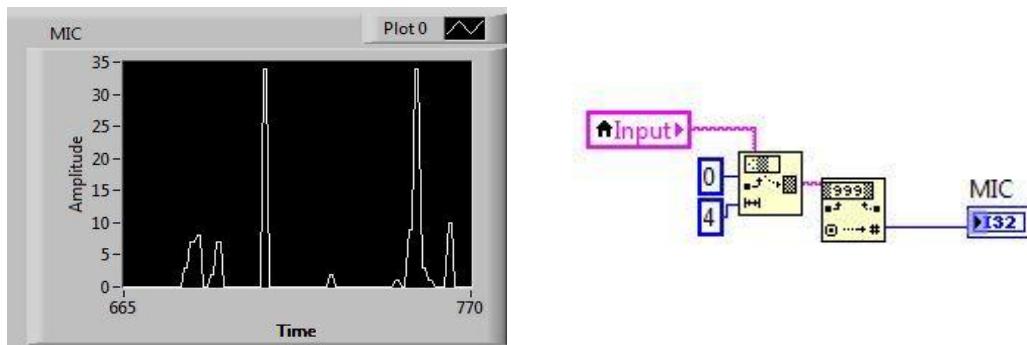
Bundle Assembles a cluster from individual elements.	تقوم بتجميع قيم 8 مفاتيح في Cluster Programming > Cluster
Cluster To Array Converts a cluster of elements of the same data type to a 1D array of elements of the same data type.	تقوم بالتحويل من ال Cluster إلى Array مع ملاحظة أن المصفوفة من نوع Boolean Programming > Cluster
Boolean Array To Number Converts a Boolean array to a 32-bit unsigned integer by interpreting the array	يتم تحويل المصفوفة والتي تحتوي بداخلها 8 بت إلى ما يقابلها من رقم Decimal ، مثلاً لو كانت جميع المفاتيح on سيكون الخرج 255. Programming > Numeric > Conversion

<p>To Unsigned Byte Integer</p>  <p>Converts a number to an 8-bit unsigned integer in the range 0 to 255.</p>	<p>تقوم بتحويل نوع خرج التابع السابق من 32 بت إلى 8 بت :</p> <p>Programming > Numeric > Conversion</p>
<p>Type Cast</p>  <p>Casts x to the data type,</p>	<p>تقوم بالتحويل من صيغة ASCII إلى Decimal .. وكمثال الرقم 65 سيصبح حرف A.</p> <p>Programming > Numeric > Data Manipulation</p>
<p>Concatenate Strings</p>  <p>Concatenates input strings and 1D arrays of strings into a single output string. For array inputs, this function concatenates each element of the array.</p>	<p>تقوم بوصل أكثر من string وستستخدم لالحاق رمز نهاية الكود ويمكن استخدامها في حال أردنا الكتابة على شاشة LCD.</p> <p>Programming > String</p>

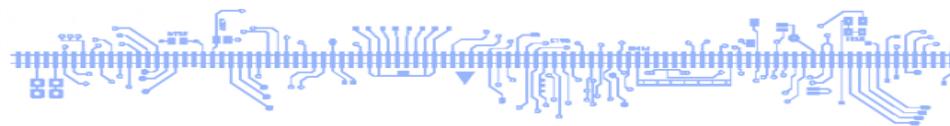
10. البرمجة الصندوقية للحساسات :

1.10. الحساس الصوتي:

ويتمثل بمدخل حساس إشار صوتية باستخدام ميكروفون إلكتروليتي مدمج على اللوحة ، حيث يقطع أربع محارف من النص المرسل من المعالج و عبر المنفذ التسلسلي ليتم في البرمجة الصندوقية تمثيله بمتحول ويأخذ أربع عدات (من 0 إلى 4) .

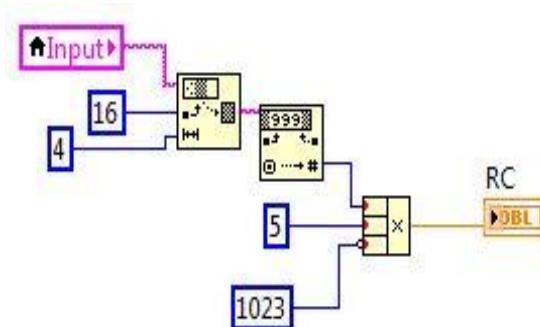
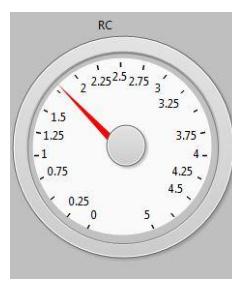


الشكل (2.35) الواجهة الرسومية و البرمجة الصندوقية للحساس الصوتي



2.10. الحساس السعوي :

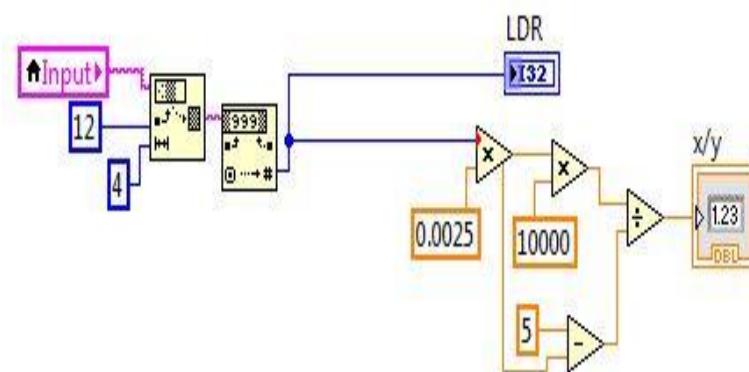
و يتمثل على لوحة التطوير بدارة قياس معامل المقاومة - السعة (R-C) ، و يقطع أربع محارف من النص المرسل من المعالج و عبر المنفذ التسلسلي ليتم في البرمجة الصندوقية تمثيله بمتحول ويأخذ أربع عدات (من 16 إلى 20) .. و الشكل التالي يوضح ذلك :



الشكل (2.36) الواجهة الرسمية و البرمجة الصندوقية للحساس السعوي

3.10. حساس شدة الإضاءة LDR :

و يتمثل على لوحة التطوير بدارة قياس شدة الضوء باستخدام المقاومة الضوئية LDR ، و يقطع أربع محارف من النص المرسل من المعالج و عبر المنفذ التسلسلي ليتم في البرمجة الصندوقية تمثيله بمتحول ويأخذ أربع عدات (من 12 إلى 16) .. و الشكل التالي يوضح ذلك :

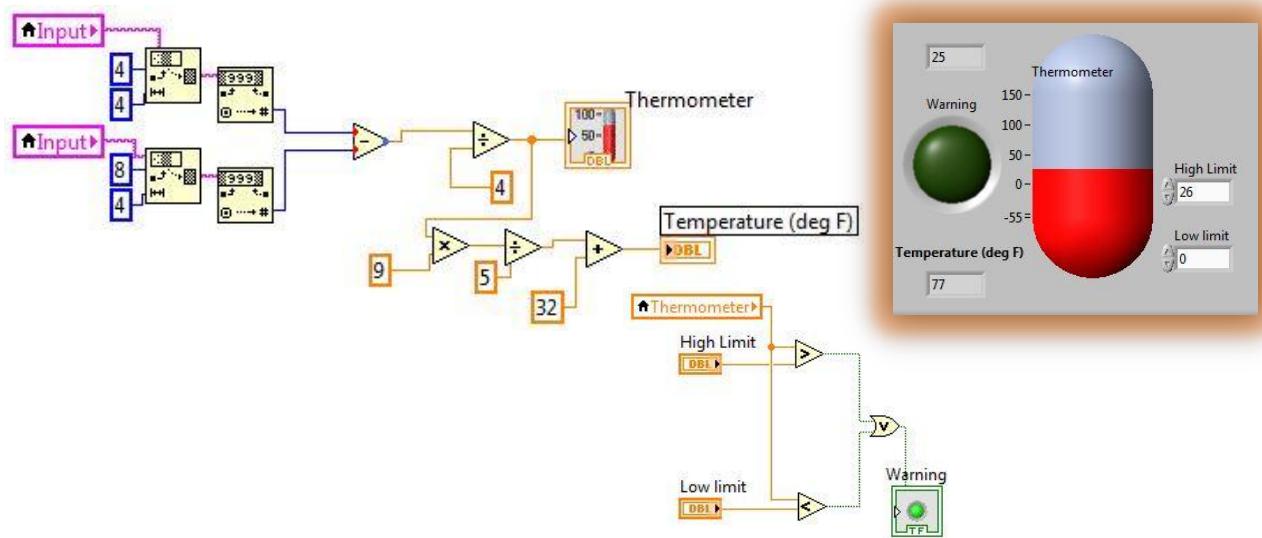


الشكل (2.37) الواجهة الرسمية و البرمجة الصندوقية للحساس الضوئي

4.10 . حساس الحرارة LM-35

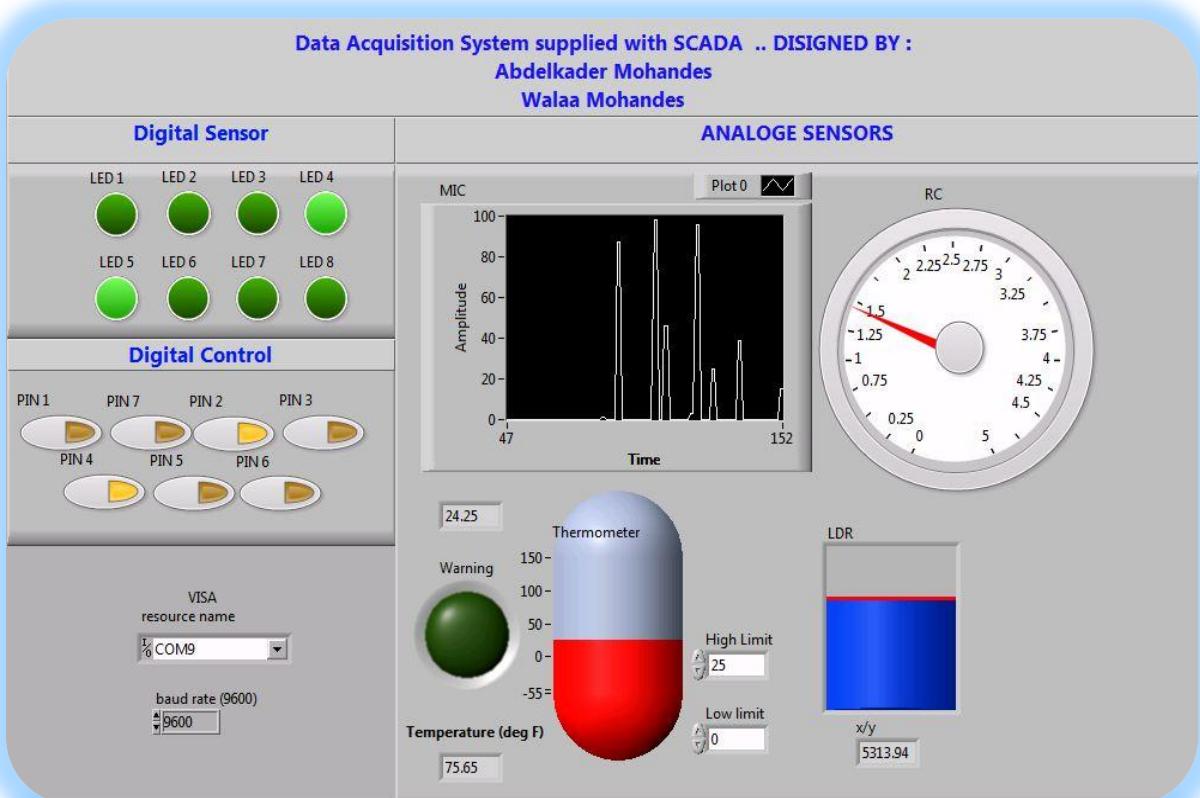
ويتمثل على لوحة التطوير بدارة قياس شدة الحرارة باستخدام الحساس التشابهي LM-35 .. وبما أنّ الحساس يعتمد الخاصة التفاضلية بين قطبين من أجل قياس درجات الحرارة الموجبة والسلبية .. فإننا نعتمد طرح القيم المقروءة والواصلة إلى Labview ومقارنتها مع قيمة مرجعية مدخلة عن طريق المستخدم من ثم نقوم بتحويل درجة الحرارة المقاسة إلى فهرنهايت .

أما بالنسبة لتحديد القيم العليا والدنيا لدرجة الحرارة فيتم في هذه الحالةأخذ thermometer وأدخاله مع دارة مقارن بحيث تكون القيمة المرجعية عبارة عن Control يحدده المستخدم ، وبذلك يقوم بتحديد مجال درجة الحرارة المرغوبة .. ويكون جهاز إنذار حساس الحرارة عبارة عن Indicator على الواجهة الرسومية للبرنامج .. والشكل التالي يوضح ذلك :



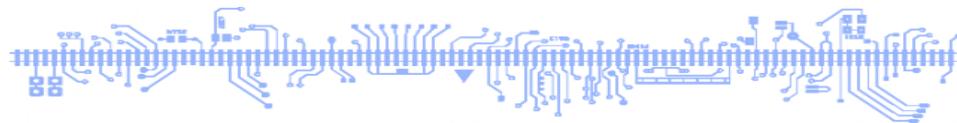
الشكل (2.38) الواجهة الرسومية والبرمجة الصندوقية لحساس الحرارة

والشكل التالي يبيّن الواجهة الرسومية النهائية للبرنامج Labview متضمناً الحساسات الرقمية والتشابهية إضافة إلى التحكم الرقمي وإعداد بaramيترات المنفذ التسلسلي :



الشكل (2.39) الواجهة الرسومية للنظام





الدراسة التطبيقية للمعالج ATMEGA-128

1. مقدمة إلى متحكمات AVR : (Introduction to AVR Microcontrollers)

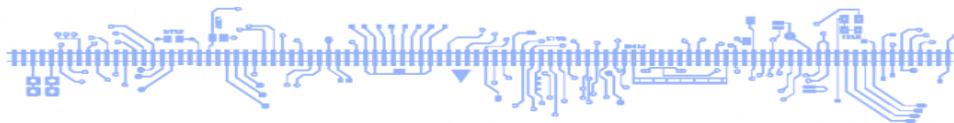
تعتبر متحكمات AVR إحدى منتجات شركة ATMEL الأمريكية، وقد تم تطويرها في مختبرات الشركة الموجودة في النرويج في أواخر التسعينيات، وتعتبر من أكثر المتحكمات المصغرة انتشاراً لما تتميز به من العديد من الميزات التي جعلتها مناسبة لكثير من التطبيقات.

لقد أحدثت شركة ATMEL ثورة في عالم المتحكمات المصغرة بإنجاحها لمتحكمات AVR التي تفوقت بشكل كبير على العديد من نظيراتها من متحكمات 8-bit، حيث تم استخدام البنية RISC التي تتميز بالأداء العالي وبالطاقة المنخفضة، واحتوت قائمة التعليمات في متحكمات AVR على 132 تعليمة - ينفذ معظمها خلال دورة آلة واحدة (1-cycle) – وبالتالي عند وصل هرّاز 16MHz إلى المتحكم فإنه سينفذ حوالي 16MIPS (مليون تعليمة في الثانية الواحدة)، كما زوّدت هذه المتحكمات بذاكرة برنامج قابلة للمسح والكتابة لأكثر من 100000 مرة، وضمنت شركة ATMEL أن يبقى البرنامج داخل المتحكم يعمل بشكل صحيح حتى 25 سنة، كما تملك متحكمات AVR وحدات محبوكة مدمجة متعددة الوظائف الأمر الذي يوفر استخدام دارات متكاملة خارجية، كذلك زوّدت معظم متحكمات AVR بمبدل تشابهي رقمي متعدد الأقنية مدمج داخل المتحكم، إضافة إلى إمكانية برمجة المتحكم دون فصله عن النظام (In-system Programming)، وكذلك تتوفر في الأسواق بكميات كبيرة وسعرها منخفض مقارنة مع ميزاتها.

2. عائلات متحكمات AVR : (AVR MCUs Families)

تقسم عائلات متحكمات AVR ذات عرض ناقل 8-bit إلى أربع مجموعات أساسية، إضافة إلى مجموعات أخرى ذات وظائف خاصة، تمتلك جميعها نفس البنية وتختلف عن بعضها البعض بالميزات والخصائص الموجودة في كل نوع:

✓ العائلة AT90Sxxxx: العائلة الكلاسيكية التي كان منها الانطلاق الأولي لمتحكمات AVR في عام 1997 وقد توقفت صناعتها.



العائلة ATtinyxx: وهي العائلة الصغرى لمتحكمات AVR المطورة والتي ظهرت في أوائل عام 2000، وهي تملك عدد أقطاب قليل (6~32pin) وحجم ذاكرة برنامج صغير نسبياً (0.5~16KB) وموارد محدودة على الشريحة الأمر الذي يجعل سعرها منخفض مقارنة مع متحكمات ATmega.

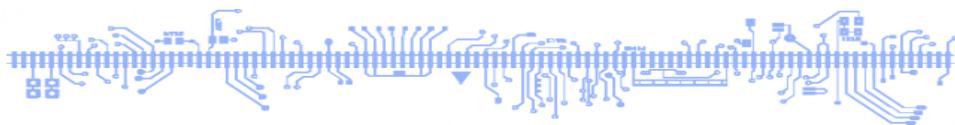
العائلة ATmegaxxx: وهي العائلة الكبرى لمتحكمات AVR المطورة والتي ظهرت في أوائل عام 2000 وهي تملك عدد أقطاب كبير (28~100pin) وحجم ذاكرة برنامج كبير نسبياً (8~256KB) وموارد متنوعة على الشريحة.

العائلة ATxmegaxxx: وهي العائلة المتطرفة والأحدث لمتحكمات AVR وقد ظهرت في عام 2008 وهي تملك ميزات متنوعة وسعة معالجة كبيرة نسبياً وتعمل بترددات أعلى من سابقاتها ... كما أنها تملك عدد أقطاب كبير (32MHz) (44~100pin) وحجم ذاكرة برنامج كبيرة (384KB~16) إضافة إلى ميزات جديدة لا تتوفر في سابقاتها ، والشكل التالي يبين ملخصاً للعائلات والخصائص الأساسية لكل منها...

BatteryM AVR	Lighting AVR	USB AVR	megaAVR	tinyAVR
18 ~ 48 Pin	24 ~ 32 Pin	32 ~ 64 Pin	28 ~ 100 Pin	8 ~ 32 Pin
MAX I/O 4~18	MAX I/O 19~27	MAX I/O 22~48	MAX I/O 23~86	MAX I/O 6~28
4KB~40KB Flash	8KB~16KB Flash	8KB~128KB Flash	4KB~256KB Flash	1KB~8KB Flash
256B~1KB EPROM	512B EPROM	512B~4KB EPROM	512B~4KB EPROM	64B~512B EPROM
512B~2KB SRAM	512B~1KB SRAM	512B~8KB SRAM	512B~16KB SRAM	32B~512B SRAM
Up To 8MIPS	Up To 16MIPS	Up To 16MIPS	Up To 20MIPS	Up To 20MIPS
1.8V – 25V	2.7V – 5.5V	2.7V – 5.5V	1.8V – 5.5V	1.8V – 5.5V

AVR® 8-Bit متحكمات العائلة				
				
Automotive AVR	CAN AVR	LCD AVR	AVR Z-Link	xmegaAVR
14 ~ 64 Pin	64 Pin	64 ~ 100 Pin	MCU Wireless chipset for: IEEE 802.15.4 and ZigBee applications.	44 ~ 100 Pin
MAX I/O 6~54	32KB~128KB Flash	MAX I/O 54~69		MAX I/O 36~78
2KB~128KB Flash	1KB~4KB EPROM	16KB~64KB Flash		16KB~384KB Flash
128B~4KB EPROM	1K~4KB SRAM	512B~2KB EPROM		1KB~4KB EPROM
128B~4KB SRAM	Up To 16MIPS	1KB~4KB SRAM		2KB~32KB SRAM
Up To 16MIPS	2.7V – 5.5V	Up To 20MIPS		Up To 32MIPS
2.7V – 5.5V		1.8V – 5.5V		1.8V – 3.6V

الشكل (3.1) الخصائص العامة لعائلات متحكمات AVR



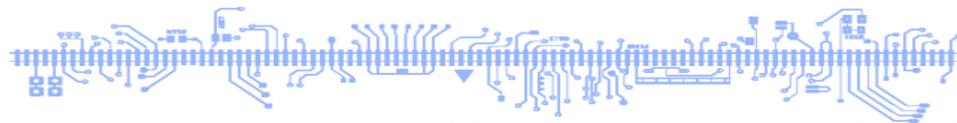
3. مقارنة بين أشهر عائلات المتحكمات المصغرة (Comparison between most famous μC)

الجدول التالي يبيّن مقارنة بين أشهر عائلات متحكمات 8-bit.

HC11 (Motorola)	PIC (Microchip)	8051 (Intel)	AVR (Atmel)	الميزة / العائلة
Von-Neumann	Harvard	Von-Neumann	Harvard	البنية الأساسية
CISC	RISC	CISC	RISC	تقنية النواة
8MHz	20MHz	24MHz	20MHz	تردد التشغيل الأعظمي
8	4	12	1	نبضة لكل تعليمية
1MIPS	5MIPS	2MIPS	16MIPS	تعليمية في الثانية
200	32	215	132	عدد التعليمات
32KB	64KB	32KB	256KB	حجم ذاكرة البرنامج
8-bit	12-bit	8-bit	16-bit	عرض ناقل التعليمات

من خلال قراءة الجدول نستنتج أفضلية متحكمات AVR للأسباب التالية:

- ✓ متحكمات AVR أسرع من متحكمات PIC بأربع مرات، وأسرع من متحكمات 8051 بثمانية مرات.
- ✓ متحكمات AVR تملك ذاكرة برنامج ذات حجم أكبر من باقي العائلات مما يمكن من كتابة برامج ضخمة.
- ✓ متحكمات AVR مبنية بالاعتماد على تقنية Harvard التي تقوم على الفصل بين ذاكرة البيانات وذاكرة التعليمات بحيث يكون لكل من الذاكيتين خطوط عنونة منفصلة (عناوين فизيائية مستقلة) وكذلك الأمر بالنسبة لخطوط التحكم وممر المعطيات، الأمر الذي يمكن من أن تحدث عملية قراءة التعليمات مع قراءة أو كتابة البيانات في نفس اللحظة، وكذلك يتاح لطول كلمة البيانات أن يكون مختلفاً عن طول كلمة التعليمات بسبب عدم اشتراك البيانات والتعليمات في نفس الذاكرة. بالمقارنة مع البنية von-Neumann فإن هذه الأخيرة منظمة بحيث لا يوجد فصل بين ذاكرة التعليمات وذاكرة البيانات ولهم نفس خطوط العنونة ونفس ممر المعطيات، وبالتالي فإن الفائدتين اللتان تم ذكرهما سابقاً لا توفرهما البنية von Neumann مما يجعل بنية Harvard ذات أداء أعلى من حيث سرعة المعالجة وتنفيذ البرنامج.
- ✓ متحكمات AVR تملك نواة من التقنية RISC التي تمكن من إنجاز تعليمية خلال دورة هرزاً واحدة بخلاف التقنية CISC التي تحتاج عدة دورات هرزاً لتنفيذ تعليمية واحدة. كذلك فإن البنية RISC أقل تكلفة من البنية CISC.



4. الميزات الأساسية للمتحكم : (ATmega128A Features) ATmega128A

الميزات الأساسية تأتي دائمًا في الصفحة الأولى من الوثيقة الفنية لأي دارة متكاملة... في ما يلي

ميزات المعالج :

❖ متحكم 8-bit بـأداء عالٍ واستهلاك منخفض للطاقة.

❖ بنية متطرفة من النوع RISC (أقل عدد ممكن من التعليمات):

▪ 133 تعليمة معظمها تنفذ بدورة آلة واحدة

▪ 32 x 8 مسجلات أغراض عامة + ومسجلات تحكم محيطية

▪ عمل مستقر ومناعة ضد الضجيج

▪ قادر على تنفيذ 16 مليون تعليمة في الثانية عند تردد 16MHz

▪ يحوي على مضاعف دورة العمل

❖ ذاكرة معطيات دائمة:

▪ 128KB ذاكرة برنامج يمكن برمجتها بدون فصل المعالج عن الدارة، قابلة للمسح والكتابة 10,000

مرة

▪ أقفال برمجية مستقلة مع قطاع مخصص للكود إقلاع.

▪ 4KB ذاكرة معطيات دائمة EEPROM قابلة للمسح والكتابة 100,000 مرة

▪ 4KB ذاكرة وصول عشوائي مؤقتة SRAM

▪ إمكانية عنونة 64KB (وصل) ذاكرة برنامج خارجية

▪ أقفال برمجية من أجل حماية البرنامج على الشريحة

▪ واجهة ربط تسلسليّة (SPI) من أجل برمجة المعالج دون فصله

❖ واجهة اختبار (JTAG):

▪ قابلية مسح المسجلات الداخلية للمعالج وقراءة حالاتها

▪ دعم متخصّص بأخطاء (Debug) شامل للشريحة

▪ إمكانية برمجة ذاكرة البرنامج وذاكرة المعطيات.

❖ الميزات المحيطية:

▪ اثنان مؤقت/عداد 8-bit مزود بأنماط مقسم ترديي وحادية مقارنة

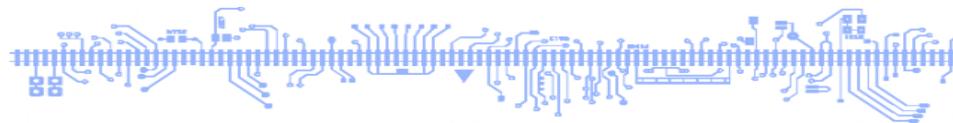
▪ اثنان مؤقت/عداد 16-bit موسع مزود بأنماط مقسم ترديي وحادية مقارنة وحادية مسائّ

▪ عدد الزمن الحقيقي مع هرزاً مستقل

▪ قناتي خرج (PWM) تتعديل عرض النبضة 8-bit

▪ ستة قنوات خرج (PWM) تتعديل عرض النبضة 16-bit مع إمكانية التحكم بالدقة من 2 وحتى 16 بت

▪ سبع قنوات تبديل تشابهي/رقمي بدقة 10-bit



- نافذة اتصال تسلسلي ثنائية (I2C)
- نافذتي اتصال تسلسلي (USARTs) قابلة للبرمجة
- نافذة اتصال تسلسلي (SPI) بنمطي عمل قائد/تابع
- مؤقت مراقبة قابل للبرمجة مع هرزاً مستقل
- نافذة مقارن تشابهي

❖ **الميزات الخاصة للمعالج:**

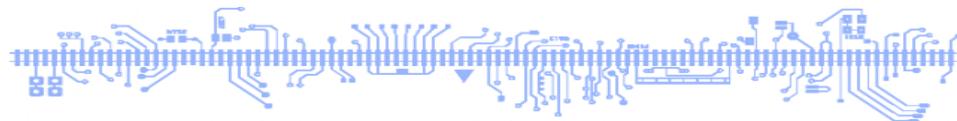
- تصفيير عند وصل التغذية وكاشف انخفاض جهد التغذية للشريحة
- هرزاً داخلي معاير
- مصادر مقاطعة خارجية وداخلية
- ستة أنماط لتخفيض الطاقة ولتخفيض ضجيج المبدل
- إمكانية تحديد تردد الهرزاً الداخلي برمجياً
- نمط تلاؤمي مع المعالج ATmega103 يحدد عن طريق الفيوزات
- إلغاء شامل لمقاومات الرفع الداخلية للبوابات

❖ **عدد أقطاب الدخل/الخرج وشكل الغلاف الخارجي للمعالج:**

- 53 قطب دخل/خرج قابل للبرمجة متوفّر من أجل شريحة 64 قطب بغلاف TQFP أو QFN/MLF.

❖ **جهود العمل للشريحة في المجال 2.7 - 5.5V**

❖ **تردد عمل أعظمي حتى 0 - 16MHz**

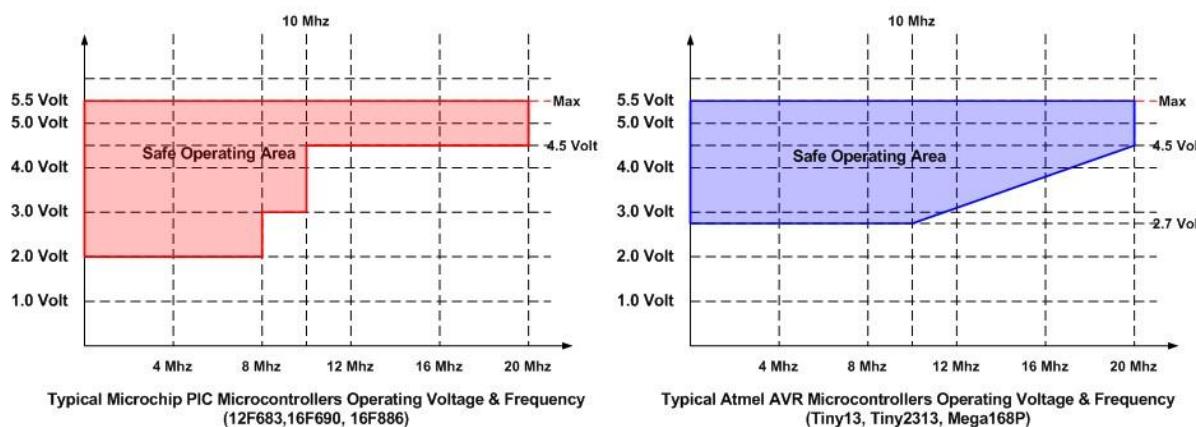


5. تغذية المتحكم المصغر :

بقدر ما تكون التغذية الرئيسية - لأي دارة إلكترونية - مصممة بشكل جيد وفق اعتبارات تصميمية قياسية ، بقدر ما يكون عمل العناصر الإلكترونية في الدارة مستقرًا وقريباً من منحني العمل الأمثل.

إن استهلاك التغذية في المتحكم يتعلق مباشرة بسرعة عمل المتحكم المصغر، حيث أنه كلما ازداد تردد عمل المعالج، ازداد استهلاك التغذية في المعالج.

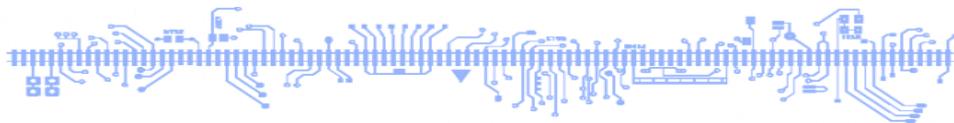
الشكل التالي بين منحني العمل الآمن للمعالج نسبة إلى التغذية المطبقة من أجل كل تردد عمل.



الشكل (3.2) منحني العمل الآمن للمعالج نسبة إلى التغذية المطبقة

من أجل متحكم مصغر من العائلة "AVR" فإن التغذية 4.5V ستؤمن عمل آمن للمعالج عند كامل مجال تردد الهراء الكريستالي، أما من أجل جهد تغذية "3V" فإن أقصى سرعة عمل للمتحكم يجب أن لا تزيد عن "8MHZ" لكي يبقى المعالج ضمن منطقة العمل الآمنة.

أحد أهم الاعتبارات التي يجب أن تؤخذ بعين الاعتبار عن ربط أقطاب المتحكم إلى الأحمال هو التيار الأعظمي المستهلاً من قطب المتحكم (Vcc to Gnd). إن قيمة التيار التي يمكن سحبها أو تصريفها لقطب دخل - خرج من أقطاب المتحكم تتراوح عادة من 20-40mA حسب المواصفات الكهربائية للمتحكم المصغر. كما أن التيار الأعظمي الذي يمكن سحبه أو تصريفه عن طريق المتحكم بشكل كلي هو 200mA. الشكل التالي يوضح المواصفات الكهربائية لمتحكمات العائلة AVR.

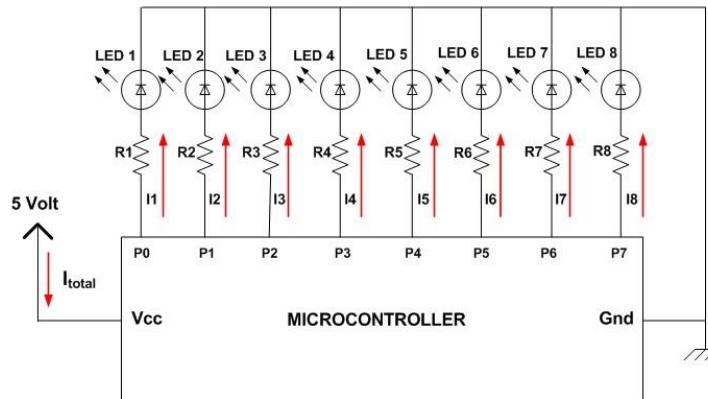


Absolute Maximum Ratings*

Operating Temperature	-55°C to +125°C
Storage Temperature	-65°C to +150°C
Voltage on any Pin except $\overline{\text{RESET}}$ with respect to Ground	-0.5V to $V_{CC}+0.5V$
Voltage on $\overline{\text{RESET}}$ with respect to Ground.....	-0.5V to +13.0V
Maximum Operating Voltage	6.0V
DC Current per I/O Pin	40.0 mA
DC Current V_{CC} and GND Pins.....	200.0 mA

الشكل (3.3) الموصفات الكهربائية لمحكمات العائلة AVR

إن التيار الأعظمي الذي يمكن استجراره من المتحكم هو مجموع تيارات الأقطاب وتيار التشغيل للمتحكم، وإن زيادة التيار فوق الحدود العظمى سوف يؤدي إلى عطل دائم في المتحكم ويجب بعدها تغييره. في الشكل التالي تم استخدام ثمانية أقطاب من متحكم مصغر كأقطاب خرج لتشغيل الثنائيات الضوئية.



Typical LEDs Display on the Microcontroller I/O Ports

الشكل (3.4)

إن التيار الأعظمي المستجر من المتحكم هو مجموع تيارات الثنائيات الثمانية بالإضافة لتيار عمل المتحكم ويمكن حسابه بالشكل التالي:

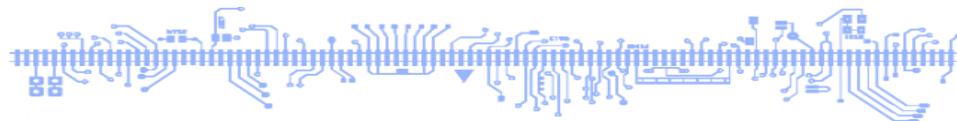
$$I_{\text{total}} = I_{\text{operating_current}} + (8 \times I_{\text{LED}})$$

بافتراض أن جهد عمل الثنائي الضوئي هو "2V" وقيمة المقاومة التسلسالية (مقاومة تحديد تيار عمل الثنائي الضوئي) هي "150Ω" ، فيمكن حساب قيمة التيار المستجر من كل قطب من العلاقة التالية:

$$I_{\text{LED}} = V / R = (5 - 2) / 150 = 20\text{mA}$$

كما أن تيار عمل المتحكم هو 2.4mA وبالتالي يمكن حساب التيار الكلي من العلاقة:

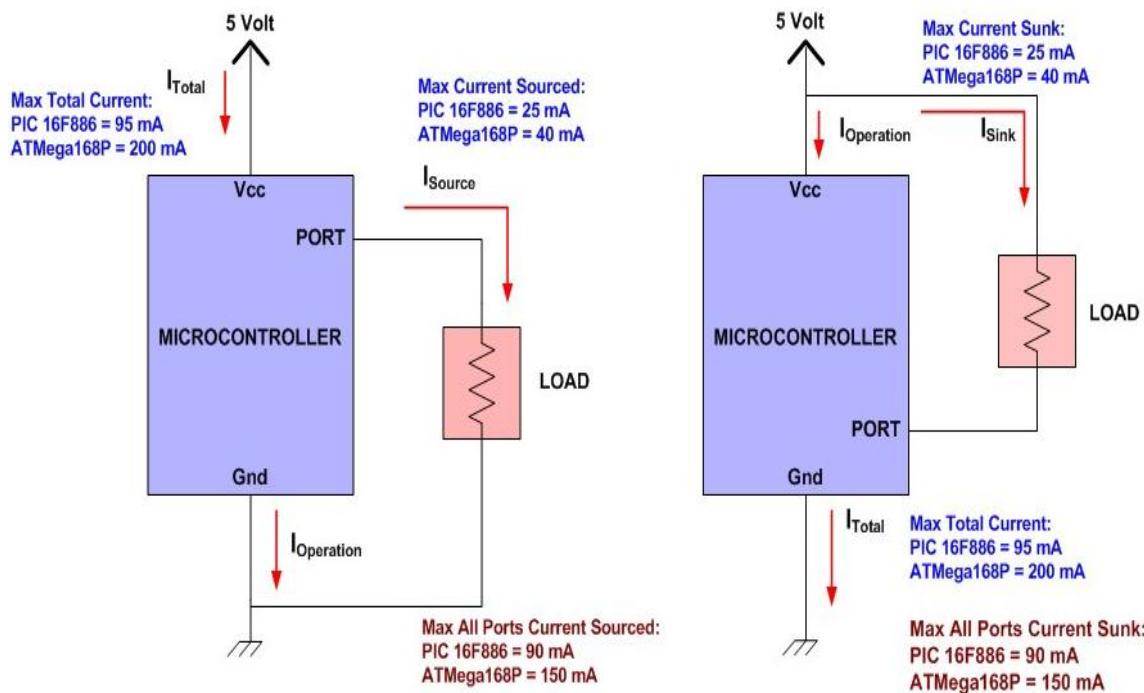
$$I_{\text{total}} = 2.5\text{mA} + 8 \times I_{\text{LED}} = 8 \times 20\text{mA} = 162.5\text{mA}$$



كما هو واضح فإن هذه القيمة تقترب من القيمة العظمى للتيار المسموح استجراره من متحكمات العائلة AVR والذي هو 200mA، بينما تفوق القيمة العظمى للتيار المسموح استجراره من متحكمات العائلة PIC والذي هو 90mA.

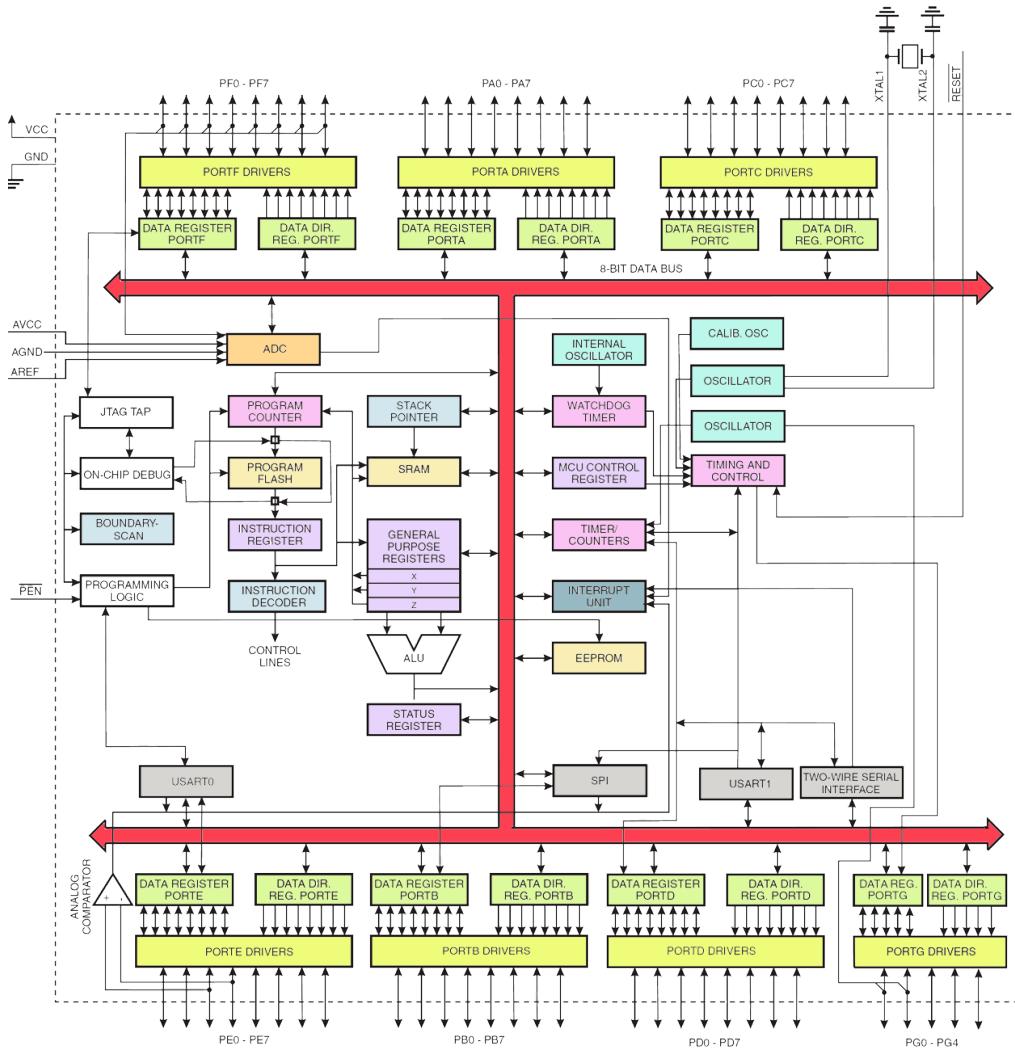
وبالتالي فإن حساب التيارات المسحوبة من أقطاب المتحكم المصغر يعتبر من أهم الأمور التي يجب دراستها في بداية أي مشروع يعتمد على المتحكم المصغر وهو ما سوف نناقشه فيما يأتي. عملياً، لا يتجاوز التيار المسحوب من المتحكم نصف قيمة التيار الأعظمي المسموح به لتخفيض ضجيج العمل وللتتأكد من أن المتحكم قادر على تيار لعمل الأحمال الموصولة معه بشكل جيد. إن وصل الأحمال مع أقطاب المتحكم يكون بطريقتين:

- ✓ القطب يعمل كمنبع لتيار تشغيل الحمل (Source).
- ✓ القطب يعمل كمصرف لتيار تشغيل الحمل (Sink).



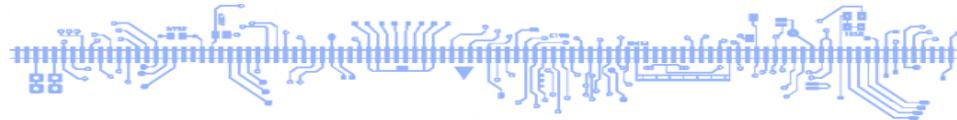
الشكل (3.5) وصل الأحمال مع أقطاب المتحكم

6. مخطط البنية الداخلية للمتحكم .(ATmega128A Block Diagram)



الشكل (3.6) المخطط الصندوقي للبنية الداخلية للمتحكم ATmega128A

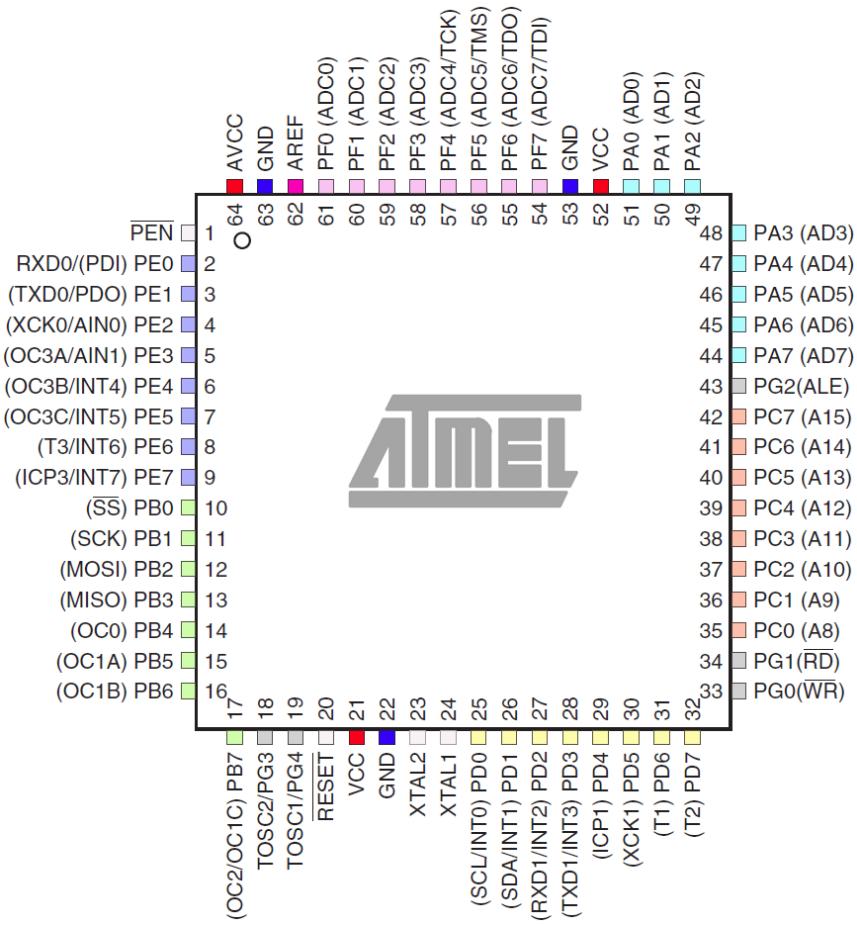
تجمع نواة العائلة AVR بين مجموعة التعليمات القوية والمسجلات الـ32 المستخدمة للأغراض العامة، حيث تربط المسجلات الـ32 مباشرةً مع وحدة الحساب والمنطق ALU، الأمر الذي يسمح بالوصول إلى أي مسجل من هذه المسجلات المستقلة عند تنفيذ تعليمات واحدة، فالبنية الناتجة أعطت فعالية أكبر لشيفرة التعليمية وأصبحت سرعة التنفيذ أكبر بعشرين مرات من بنية متحكمات CISC التقليدية. لقد تم صناعة الشريحة ATmega128A باستخدام تقنية ذواكر ATMEtal الغير قابلة للزوال ذات الكثافة العالية، مع إمكانية برمجة ذاكرة البرنامج الوميضية (Flash) المبنية على شريحة المتحكم إما من خلال الوصلة التسلسليّة SPI أو باستخدام مبرمجة تفرعية أو باستخدام برنامج إقلاع موجود على الشريحة



(Boot program) حيث تستطيع البرمجية المخزنة في جزء الإقلاع (Bootloader) في الذاكرة الوميضية RISC متابعة عملها أثناء تحديث القسم الرئيسي في ذاكرة البرنامج. لقد أدى الجمع ما بين معالجات ذات 8-bit مع ذاكرة البرنامج القابلة لإعادة البرمجة إلى إنتاج المتحكم ATmega128A الذي يتمتع بالقوة و المرونة العالية وبالكلفة المنخفضة للعديد من تطبيقات التحكم المتطورة. الشكل 3.7 يبين المخطط الصنودوقي للبنية الداخلية للمتحكم ATmega128 وهو يبين طريقة ربط الوحدات المحيطة والمسجلات مع وحدة المعالجة المركزية.

7. وصف أقطاب المتحكم (ATmega128A Pin Description)

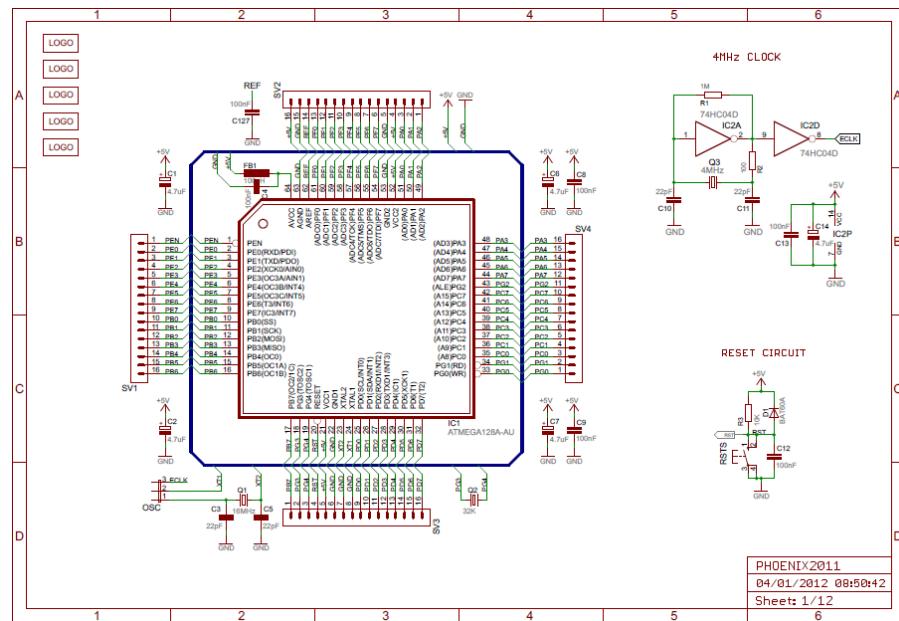
يملك المتحكم ATmega128A مجموعة من الأقطاب عددها 64 قطب موزعة على الأطراف الفيزيائية لشريحة المتحكم وهي:



الشكل 3.7 (3) توزيع الأقطاب على الشريحة ATmega128A

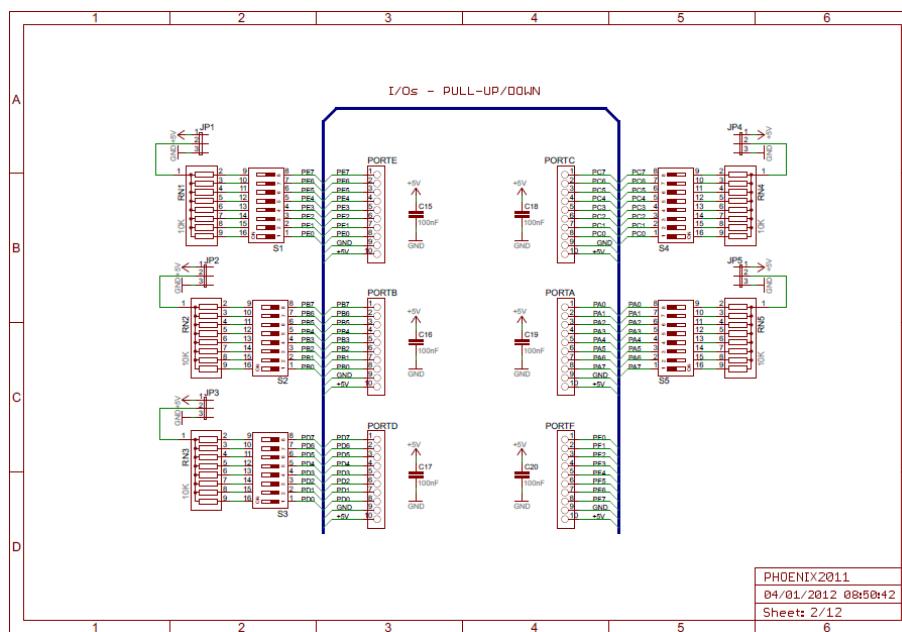


الشكل (3.8) يبين المخطط النظري للمعالج ATmega128A في لوحة التطوير المستخدمة : و الشكل (3.8) يبين المخطط النظري لتوزيع أقطاب المعالج ATmega128A في لوحة التطوير المستخدمة :

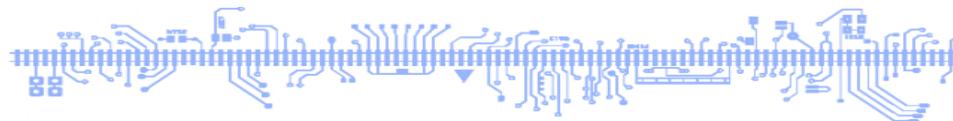


الشكل (3.8) المخطط النظري لأقطاب للمعالج

و الشكل (3.9) يبين المخطط النظري لتوزيع أقطاب المعالج ATmega128A في لوحة التطوير المستخدمة :

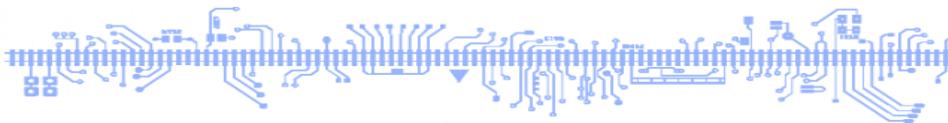


الشكل (3.9) المخطط النظري لتوزيع الأقطاب



والجدول التالي يوضح الاسم و الوظيفة لكل قطب من أقطاب المعالج :

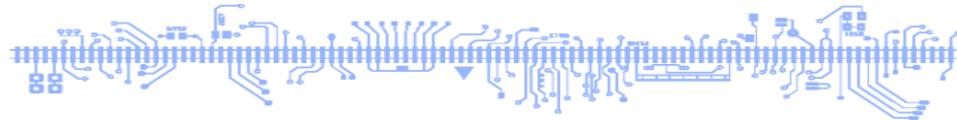
الاسم	الوظيفة
VCC	قطب جهد التغذية الموجب $VCC = 2.5 - 5.5V$
GND	قطب جهد التغذية الصفرى (الأرضي) $GND = 0V$
Port A	البوابة A: وهي عبارة عن بوابة دخل/خرج ذات ثمانية أقطاب ثنائية الاتجاه، وقد زودت الأقطاب بمقاومات رفع داخلية (Pull-up) مع إمكانية اختيار مقاومة الرفع لكل قطب على حدي. تمتلك البوابة A وظيفة ثانوية أخرى حيث تعمل كواجهة ربط مع ذاكرة خارجية وتمثل القسم السفلي لبوابة العناوين والبيانات (AD7 – AD0).
Port B	البوابة B: وهي عبارة عن بوابة دخل/خرج ذات ثمانية أقطاب ثنائية الاتجاه، وقد زودت الأقطاب بمقاومات رفع داخلية (Pull-up) مع إمكانية اختيار مقاومة الرفع لكل قطب على حدي، كذلك تمتلك البوابة B وظائف ثانوية أخرى وهي: واجهة اتصال تسلسلي SPI (MISO, MOSI, SCK, SS), وأقطاب توليد إشارات PWM (OC0, OC1A, OC1B, OC1C/OC2), وأقطاب المقارن التشابهية (AIN0, AIN1).
Port C	البوابة C: وهي عبارة عن بوابة دخل/خرج ذات ثمانية أقطاب ثنائية الاتجاه، وقد زودت الأقطاب بمقاومات رفع داخلية (Pull-up) مع إمكانية اختيار مقاومة الرفع لكل قطب على حدي. تمتلك البوابة C وظيفة ثانوية أخرى حيث تعمل كواجهة ربط مع ذاكرة خارجية وتمثل القسم العلوي لبوابة العناوين (A15 – A8) فقط.
Port D	البوابة D: وهي عبارة عن بوابة دخل/خرج ذات ثمانية أقطاب ثنائية الاتجاه، وقد زودت الأقطاب بمقاومات رفع داخلية (Pull-up) مع إمكانية اختيار مقاومة الرفع لكل قطب على حدي. تمتلك البوابة D وظائف ثانوية أخرى وهي: المقاطعات الخارجية (INT0-INT3), النافذة التسلسليه USART1 (RXD1, TXD1, XCK1), الموقتات/العدادات (T1, T2, ICP1).
Port E	البوابة E: وهي عبارة عن بوابة دخل/خرج ذات ثمانية أقطاب ثنائية الاتجاه، وقد زودت الأقطاب بمقاومات رفع داخلية (Pull-up) مع إمكانية اختيار مقاومة الرفع لكل قطب على حدي. تمتلك البوابة E وظائف ثانوية أخرى وهي: المقاطعات الخارجية (INT4-INT7), النافذة التسلسليه USART0 (RXD0, TXD0, XCK0), الموقتات/العدادات (T3, ICP3), وأقطاب توليد إشارات PWM (OC3A, OC3B, OC3C).
Port F	البوابة F: وهي عبارة عن بوابة دخل/خرج ذات ثمانية أقطاب ثنائية الاتجاه، وقد زودت الأقطاب بمقاومات رفع داخلية (Pull-up) مع إمكانية اختيار مقاومة الرفع لكل قطب على حدي. تمتلك البوابة F وظائف ثانوية أخرى وهي: المبدلات التشابهية الرقمية (ADC0-ADC7), والنافذة JTAG (TDI, TDO, TMS, TCK), والنافذة ADC (ADC0-ADC7).



البوابة G. وهي عبارة عن بوابة دخل/خرج ذات خمسة أقطاب ثنائية الاتجاه، وقد زودت الأقطاب بمقاومات رفع داخلية (Pull-up) مع إمكانية اختيار مقاومة الرفع لكل قطب على حدي. تمتلك البوابة G وظائف ثانوية أخرى وهي: أقطاب تحكم بالذاكرة الخارجية (RD, WR, ALE)، أقطاب هرزاً وحدة RTC (TOSC1,2).	Port G
مدخل تصفير الشريحة، عند تطبيق إشارة كهربائية ذات منطق منخفض على القطب RESET لمدة دورتي آلة، فإن دارة التصفير الداخلية تعمل على تصفير المتحكم - عدد البرنامج $.PC = 0$.	RESET
مدخل الهرزاً الخارجي: وهو عبارة عن مدخل دارة مضخم الهرزاً العاكس.	XTAL1
مدخل الهرزاً الخارجي: وهو عبارة عن خرج دارة مضخم الهرزاً العاكس.	XTAL2
قطب التغذية للمبدل التشابهي الرقمي (ADC)، إذا كان المبدل ADC غير مستخدم فإن هذا القطب يجب وصله إلى القطب V_{CC} ، أما إذا كان المبدل ADC مستخدماً فيوصل هذا القطب مع V_{CC} عن طريق مرشح تمرير منخفض.	AVCC
إذا كانت الدارة التي نقوم بتصميمها لها أرضي تشابهي مستقل، فيجب ربط هذا القطب مع هذا الأخير، وإلا يربط هذا القطب مع القطب الأرضي العام GND.	AGND
وهو قطب دخل الجهد المرجعي التشابهي للمبدل التشابهي الرقمي ويجب أن تتراوح قيمة هذا الجهد عند عمل المبدل ما بين $.2V - A_{V_{CC}}$.	AREF

ولقد تم استخدام عدة منافذ من المعالج المدمج على اللوحة وفق ما سبق .. منها :

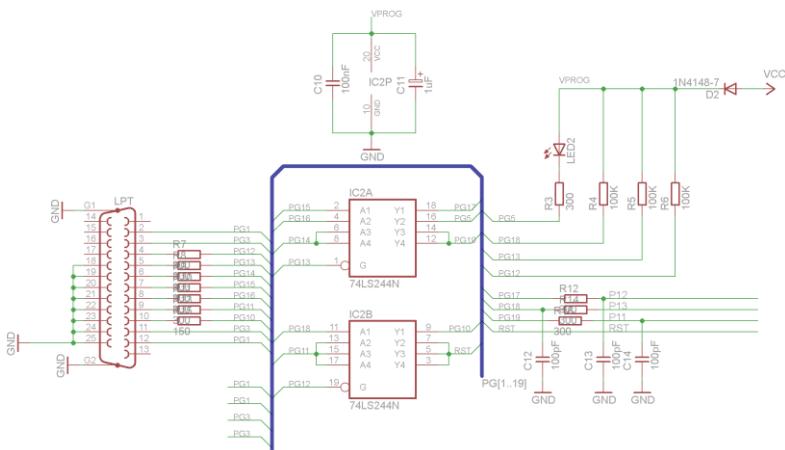
1. المنفذ A: استخدم لتمثيل الحساسات الرقمية . Digital sensors
2. المنفذ B: استخدم كمنفذ خرج مرتبط بثمانى ليدات ممثلة بـ Digital output indicators للتأكد من استجابة التحكم عن طريق Labview .
3. المنفذ C: استخدم للربط مع شاشة LCD المدمجة على لوحة التطوير والتي تتنبه عند حدوث إنذار الحساس الضوئي .
4. المنفذ D: استخدم قطب واحد من هذا المنفذ وهو القطب السادس الموصول مع Buzzer المدمج على لوحة التطوير والذي يعمل كإنذار للحساس الضوئي عند تعرض المقاومة الضوئية للضوء الشديد ، عندها تنهار المقاومة ويعمل الإنذار بتزامن مع تغير حالة الشاشة الكريستالية وظهور رسالة الإنذار .
5. المنفذ F: استخدم للتحويل التشابهي الرقمي لتأمين قراءة الحساسات التشابهية .



8. برمجة المعالج :

برمجة المعالج ستكون عن طريق مبرمج USB-ASP مدمجة على اللوحة ذاتها، و الشكل

التالي يبين المخطط النظري التصميمي للمبرمج المدمجة على اللوحة :



الشكل (3.10) المخطط النظري لدارة المبرمجة

سوف نستخدم البرنامج Bascom-AVR في برمجة اللوحة وذلك لما يوفره هذا البرنامج من بيئة برمجية قوية بالإضافة إلى المكتبات الأساسية الشاملة، ويحوي البرنامج على:

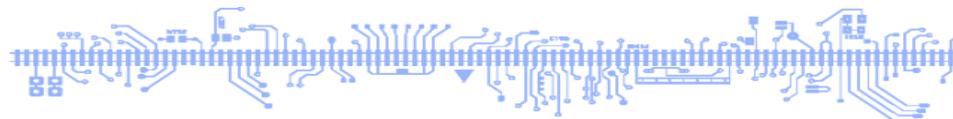
- الواجهة البرمجية الرئيسية: وهي محرر التعليمات والأوامر البرمجية.

```

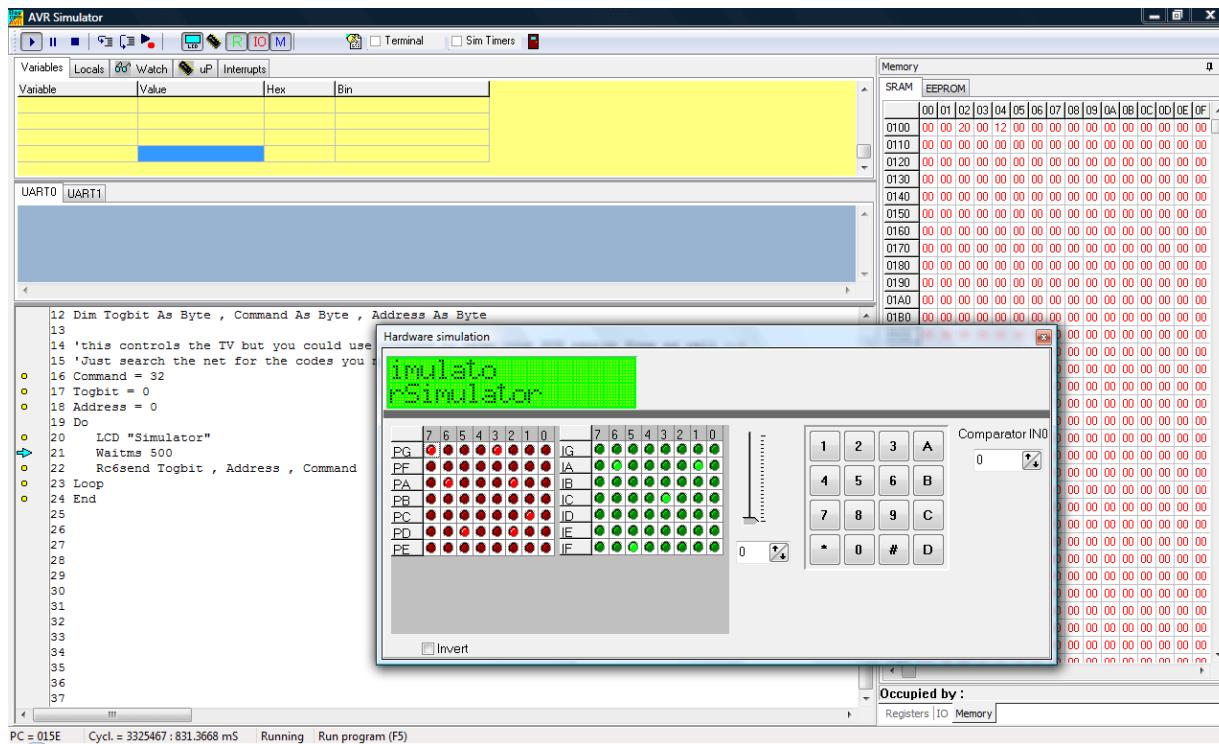
BASCOM-AVR IDE [1.11.9.0] - [D:\Program Files\MCS Electronics\BASCOM-AVR\SAMPLES\JR\sendrc6.bas]
File Edit View Program Tools Options Window Help
Sub Label
1 ' SENDRC6.BAS
2 ' (c) 2003 MCS Electronics
3 ' code based on application note from Ger Langezaal
4 ' +5V <--[A Led K]-> [220 Ohm]---> Pb.3 for 2313.
5 ' RC6SEND is using TIMER1, no interrupts are used
6 ' The resistor must be connected to the OC1(A) pin , in this case PB.3
7 '
8 '$regfile = "2313def.dat"
9 $crystal = 4000000
10
11 Dim Togbit As Byte , Command As Byte , Address As Byte
12
13 'this controls the TV but you could use rc6send to make your DVD region free as well :-
14 'Just search the net for the codes you need to send. Do not ask me for info please.
15 Command = 32
16 ' channel next
17 Togbit = 0
18 Address = 0
19 Do
20   Waitms 500
21   Rc6send Togbit , Address , Command
22 Loop
23 End
24
25
26
27
28
29
30
31
32

```

الشكل (3.11) الواجهة البرمجية

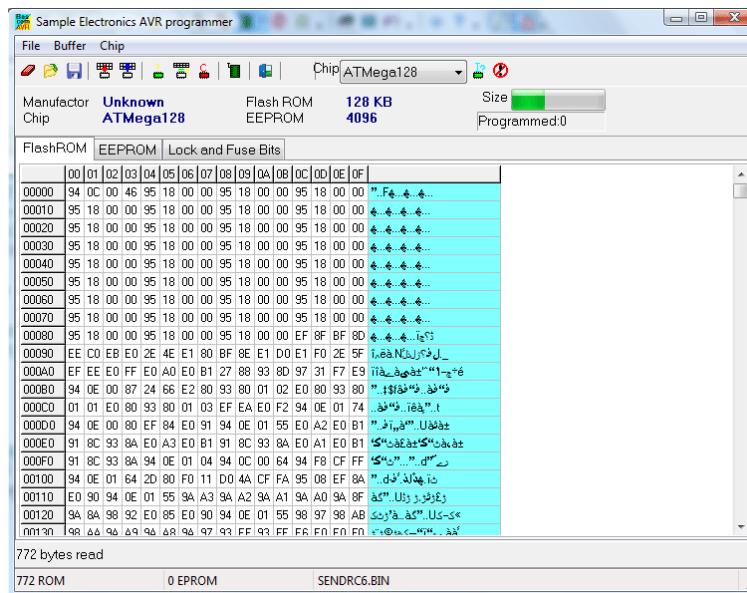


- واجهة المحاكاة: وفيها يتم تشغيل البرنامج خطوة خطوة ومراقبة حالة المسجلات الداخلية والذاكرة.

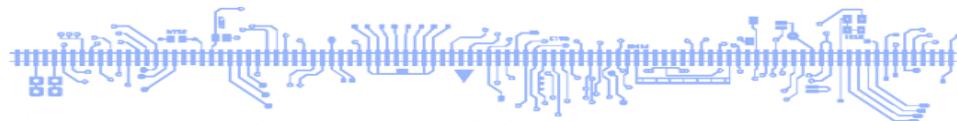


الشكل (3.12) واجهة المحاكاة

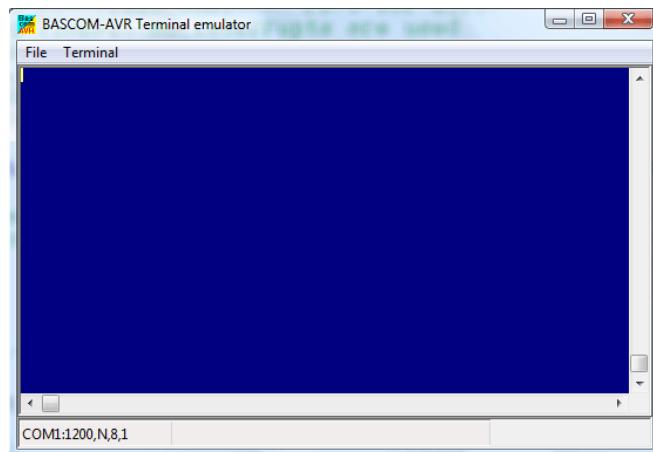
- واجهة المبرمج: وفيها يتم برمجة المعالج بعد إجراء عملية توليد الملف البرمجي بالأمر : Compile



الشكل (3.13) واجهة المبرمج لحقن المعالج



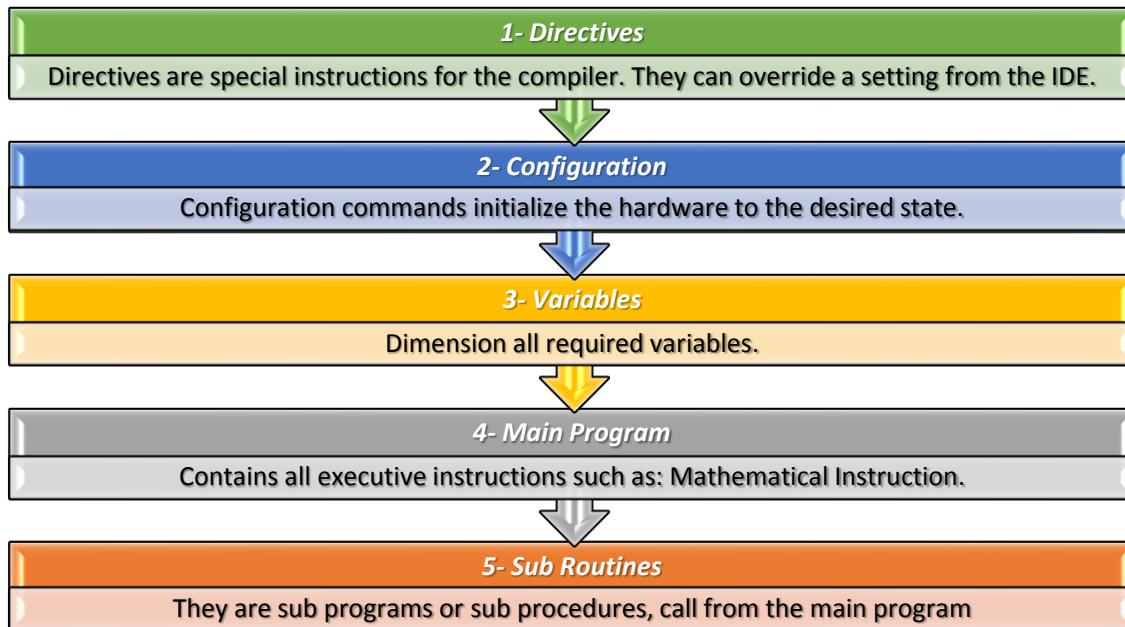
- واجهة الربط البيني: وفيها يتم عرض المعلومات المرسلة والمتعلقة بين المعالج والحاسوب بهدف مراقبة بaramترات النظام بشكل آني.



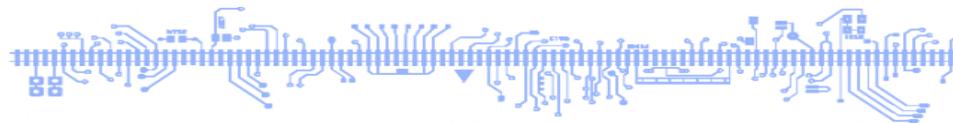
الشكل (3.14) واجهة الربط البيني

خطوات كتابة الكود البرمجي في البرنامج Bascom-AVR وبرمجه على شريحة المتحكم المصغر:

1. تشغيل البرنامج من قائمة البرامج واختيار ملف جديد.
2. من أجل كتابة كود برمجي متماستك ومفهوم مع إمكانية تعديله أو تطويره بسهولة مستقبلًا، فإنه يجب الالتزام بالهيكلية التالية في مراحل كتابة البرنامج:

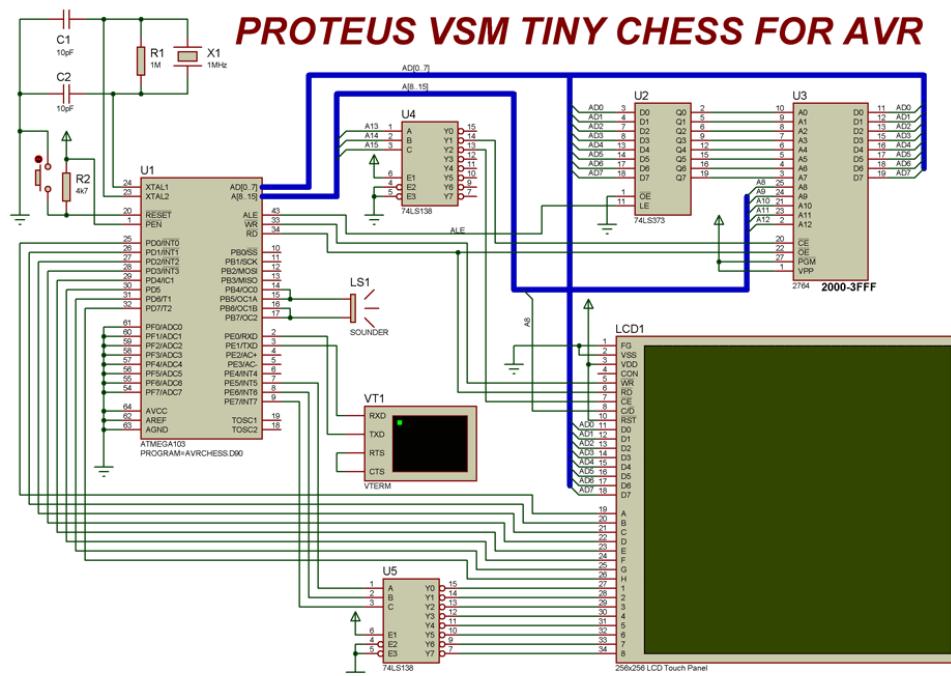


الشكل (3.15) مراحل كتابة البرنامج



3. بعد الانتهاء من كتابة الكود قم باختيار أمر تفحص الأخطاء (Syntax Check) من القائمة (Program).
4. في حال وجود خطأ برمجي سوف تشير نافذة الأخطاء (أسفل الواجهة الرئيسية) إلى موقع الخطأ وسببه.
5. بعد الانتهاء من تفحص الأخطاء، نقوم باختيار أمر الترجمة (Compile) من القائمة (Program)، عدتها سيقوم البرنامج بـتوليد الملفات البرمجية اللازمة للمبرمجة، ثم نقوم باختيار أمر الإرسال إلى المبرمج (Send to programmer) من القائمة (Program).

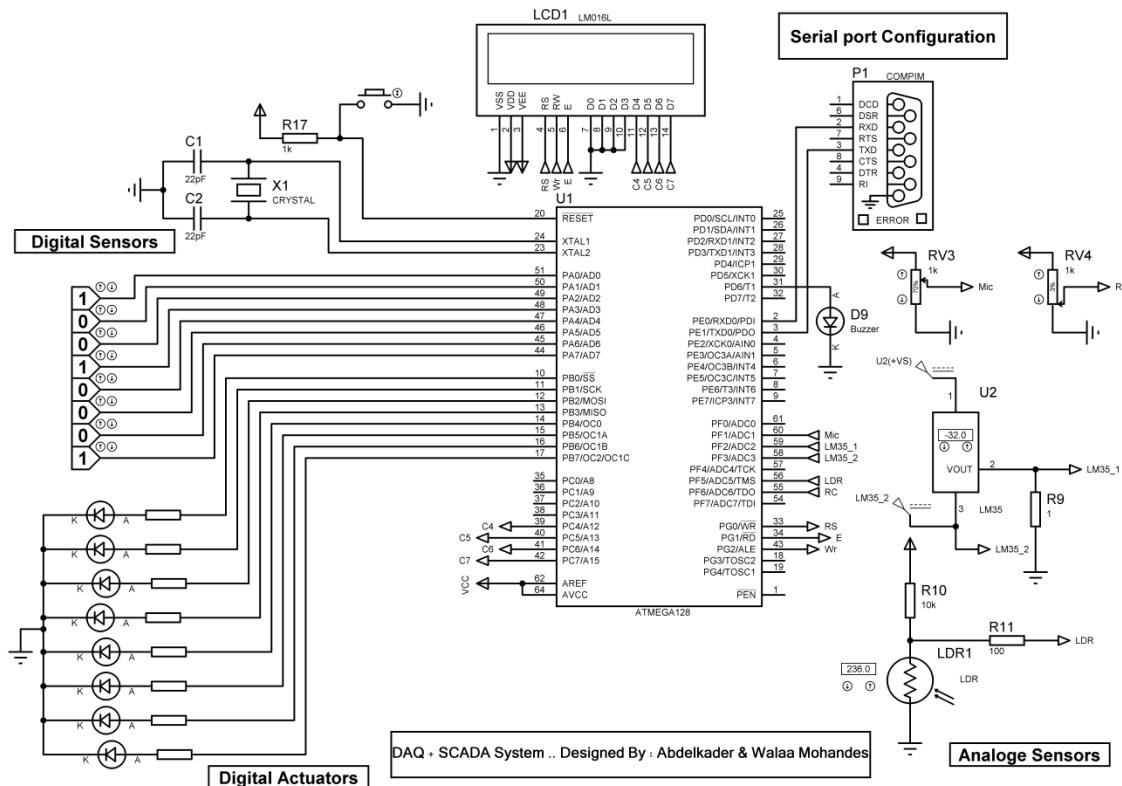
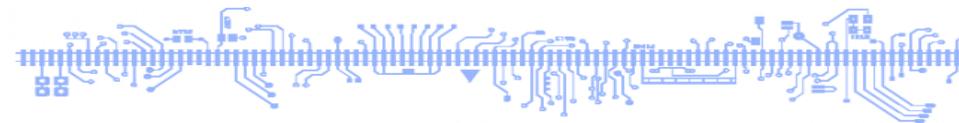
9. بيئة المحاكاة : ISIS Proteus



الشكل (3.16) بيئة محاكاة البرنامج Proteus

يعتبر برنامج Proteus من أقوى برامج المحاكاة لدارة المتحكمات المصغرة وهو يملئ العديد من المكتبات التي تغطي جميع أنواع المحيطيات التي يمكن وصلها مع المتحكم المصغر بالإضافة إلى أدوات القياس العديدة .. سوف نستخدم هذا البرنامج لتصميم دارات النظام المدروسة وتجريبيه على الحاسب بدون اللوحة .. حيث يتم الربط مع Labview في هذه الحالة عن طريق برامج الاتصال التسلسلي الوهمية Virtual Serial Port Kit ، و الشكل التالي يوضح مخطط النظام على البرنامج

: Proteus

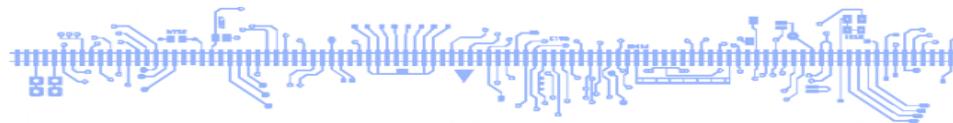


الشكل (3.17) مخطط برنامج المحاكاة

10. التحويل التشابهي - الرقمي : (Analog to Digital Conversion)

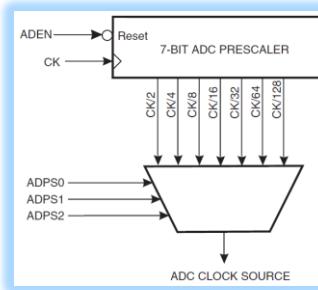
إن معظم التعليمات الجديدة لها علاقة بتشغيل وتعريف المبدل التشابهي الرقمي.

التعليمية البرمجية	شرح التعليمية
Config Adc= Single/Free, Prescaler = Auto, Reference = off Avcc internal	Single يبدأ التحويل بعد التعليمية (ADC) Start ADC وكلما مر على التعليمية Getadc. Free يبدأ التحويل بعد التعليمية Start ADC ويستمر بشكل متتابع في أحد العينات (ADCH and ADCL). والتحويل وتحديث القيم في مسجل المبدل (Prescaler).
Start Adc	تحديد المقسم الترددى للمبدل.
Stop Adc	تحديد الجهد المرجعى للمبدل.
var = Getadc (channel)	بدء عملية التحويل (تغذية المبدل). إيقاف عملية التحويل (فصل تغذية المبدل). قراءة القيمة التي تم تبديلها على قناة المبدل المحددة بـ channel.



1.10. المقسم الترددی للمبدل (Prescaler):

سوف يقوم بتقسيم تردد المهرز الكريستالي للحصول على تردد عمل المبدل، ومن أجل الحصول على أعلى دقة للمبدل يجب أن يكون تردد عمل المبدل يتراوح $~200\text{ KHz}$ ، وأماماً من أجل سرعة تبديل أكبر، فإنه يمكن زيادة تردد التبديل إلى قيم أكبر من 200 KHz ولكن ذلك سيكون على حساب الدقة، فكلما زاد تردد التبديل نقصت دقة المبدل.



(3.18) الشكل

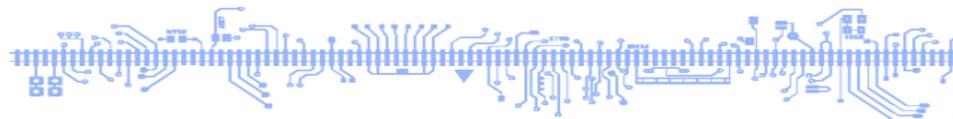
2.10. الجهد المرجعي (Reference):

يمكن اختيار الجهد المرجعي الداخلي للمبدل "Internal" وهو $2.56V$ بشرط أن لا يكون مجال إشارة القياس أكبر من الجهد المرجعي، وإلا فإنه يجب استخدام الجهد "AVCC" كجهد مرجعي. في بعض الأحيان يكون مجال إشارة القياس أصغر بكثير من الجهد المرجعي الداخلي، أو يتطلب عملية قياس دقة وبالتالي الحاجة إلى جهد مرجعي دقيق، فعندها يمكن استخدام القطب V_{ref} كمدخل جهد مرجعي للمبدل "off".

3.10. ميزات المبدلات التشابهية الرقمية في عائلة المتحكمات AVR:

تتميز المبدلات التشابهية الرقمية لمتحكمات العائلة AVR بالميزات التالية:

- طول المبدلات 10-bit.
- مجال القياس $0 - VCC$.
- خطأ القياس $\pm 0.5\text{ LSB}$.
- خطأ عدم الخطبية 0.05 LSB .
- زمن أخذ العينة $13 - 260\text{ Micro Sec}$.
- سرعة أخذ العينات 15KSPS عند أعلى دقة.
- تحوي على جهد مرجعي داخل اختياري $2.56V$.
- تملك نمطي عمل (Single, Free).
- تملك مقاطعة اكتمال عملية التحويل.



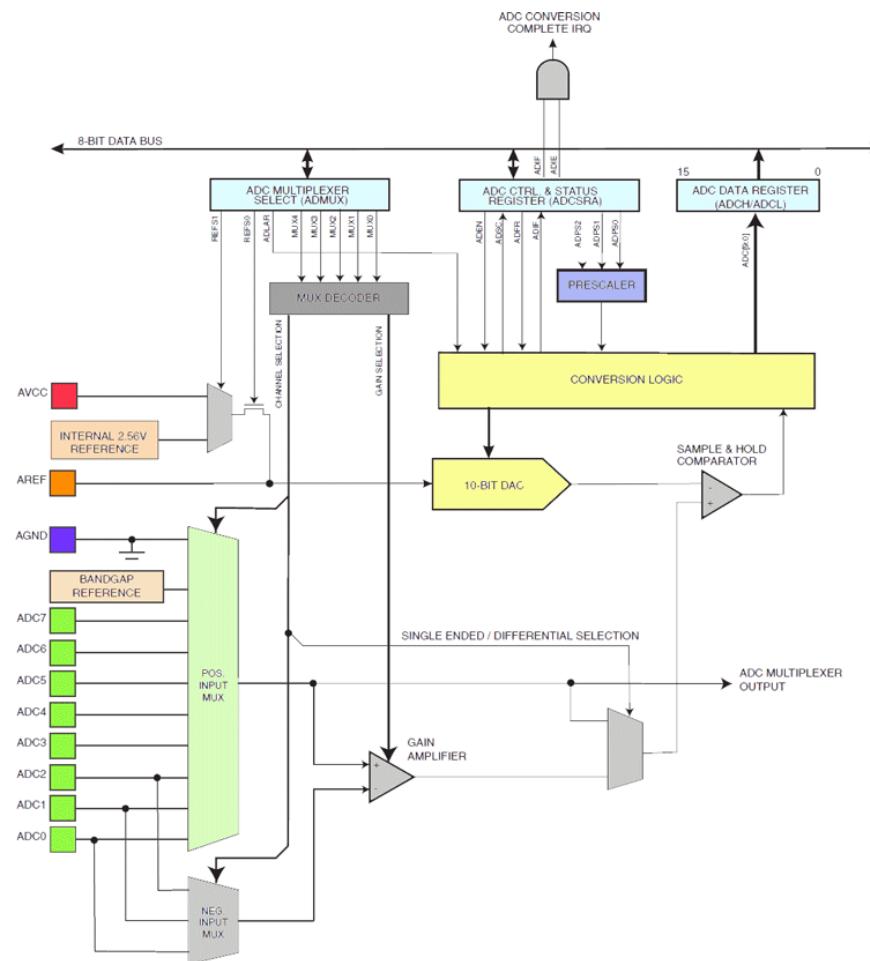
- تملك نمط تخفيض ضجيج المبدل.
- تردد عمل المبدل يتراوح $50 \sim 1000 \text{ KHz}$.
- تستخدم كمدخل مبدل إشارة ذات دخل وحيد عن طريق موزع.
- يمكن استخدامها كمدخل مبدل إشارة ذات دخل تفاضلي.
- يمكن استخدامها كمدخل مبدل إشارة ذات دخل تفاضلي مع عامل ربح $x10 - x200$.

4.10. اعتبارات هامة لتخفيض ضجيج المبدلات التشابهية الرقمية واستقرار عملها في عائلة المتحكمات AVR:

- ❖ يجب أن لا يتم وصل التغذية التشابهية للمبدل مع التغذية الرقمية للدارة لمنع انتقال الضجيج التشابهي على خطوط التغذية الرقمية، وإنما يتم الفصل بينهما عن طريق مرشح تمرير منخفض من نوع LC ($L=100\mu\text{H}$, $C=100\text{nF}$) أو RC ($R=100\Omega$, $C=100\text{nF}$).
- ❖ يجب أن لا يكون مجال التغير في الجهد بين التغذية الرقمية والتغذية التشابهية أكبر من $\pm 0.3\text{V}$.
- ❖ يجب أن تكون المسارات التشابهية الموصولة مع مداخل المبدلات على مخطط الدارة المطبوعة أقصر ما يمكن لتفادي تراكب الضجيج على هذه المسارات وتفادى انتقال الضجيج على مسارات الإشارات الرقمية.
- ❖ يجب أن تكون الإشارات التشابهية الداخلة إلى المبدل محاطة بمساحة مصممة من الأرضي التشابهي على هو مبين على الشكل جانباً.
- ❖ يجب إدخال المتحكم في نمط البطالة (Idle mode) أثناء عملية التبديل وتفعيل مقاطعة انتهاء عملية التبديل للخروج من نمط البطالة من أجل دقة قياس أكبر وعدم تأثر المبدل بضجيج عمل المعالج، وذلك كما يلي:
 - .1. يجب اختيار نمط العمل "Single".
 - .2. يجب تفعيل مقاطعة اكتمال عملية التحويل "Enable ADC".
 - .3. قم بتمكين المبدل "Start ADC" وتأكد من أنه لا يقوم بأي عملية تحويل.
 - .4. أدخل المتحكم في نمط البطالة "Idle".
- .5. سوف يبدأ المبدل بالتحويل حالما يتوقف المعالج عند الدخول إلى نمط البطالة، ومع انتهاء عملية التحويل سوف تحدث مقاطعة اكتمال عملية التحويل للمبدل والتي بدورها تخرج المتحكم من حالة الخمول إلى حالة العمل الطبيعي وسوف يقوم المتحكم بتنفيذ برنامج المقاطعة (قراءة القيمة المبدلة).



5.10. مخطط البنية الداخلية للمبدل ADC :



الشكل (3,19) بنية المبدل التشابهي- الرقمي

6.10. حساب الجهد على دخل المبدل:

بما أن المبدلات التشابهية الرقمية لمحكمات العائلة AVR هي بطول 10-bit أي 1024^2 فإن القيمة التي سيعطيها المبدل ستكون $1023 - 0$ من أجل مجال جهد دخل المبدل $V_{CC} - 0$.

تعطى العلاقة التي تحسب القيمة في مسجل المبدل (قيمة التبديل) بالشكل التالي:

$$ADC_{val} = \frac{V_{in} \times 1024}{V_{ref}}$$

حيث أن:

هو الجهد على مدخل قطب المبدل.

: هو الجهد المرجعي للمبدل.



الفصل الثالث

من العلاقة السابقة يمكن إيجاد الجهد على دخل المبدل بالشكل:

$$V_{in} = \frac{ADC_{val} \times V_{ref}}{1024}$$

في حال استخدام المبدل ADC كمبدل إشارة تفاضلية فإن العلاقة تصبح على الشكل التالي:

$$ADC_{val} = \frac{(V_{Pos} - V_{Neg}) \times GAIN \times 512}{V_{ref}}$$

حيث أن:

V_{Pos} : هو الجهد الموجب على مدخل القطب الموجب للمبدل.

V_{Neg}: هو الجهد السالب على مدخل القطب السالب للمبدل.

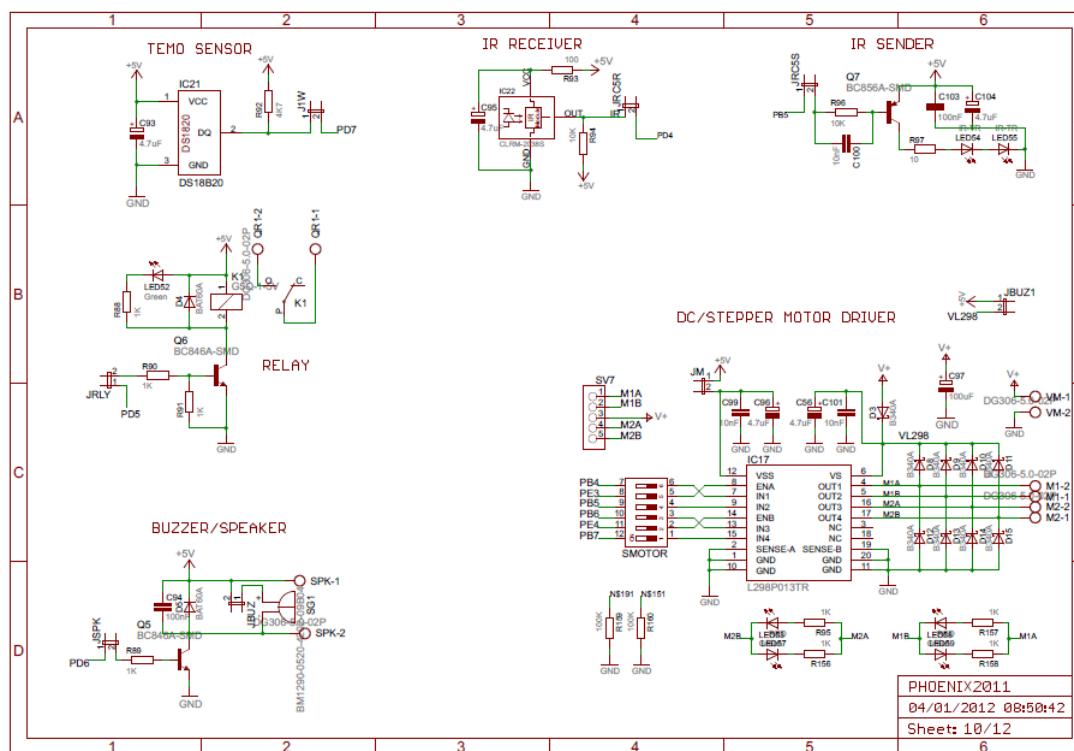
GAIN: هو عامل الربح (الضرب) المختار للمبدل.

في هذه الحالة سيتم تمثيل القيمة ADC_{val} بالشكل التالي:

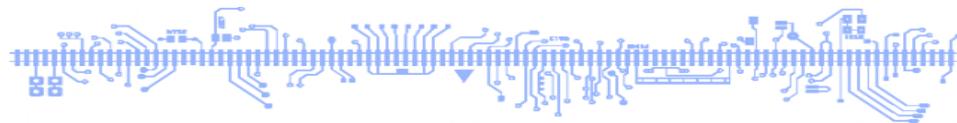
$0 \rightarrow 511 \gg Positive$
 $512 \rightarrow 1023 \gg Negative$

١١. دراسة الحساسات التشايرية المستخدمة من اللوحة:

الشكل التالي يوضح المخطط النظري لبعض الحساسات التشابهية المستخدمة من لوحة التطوير:

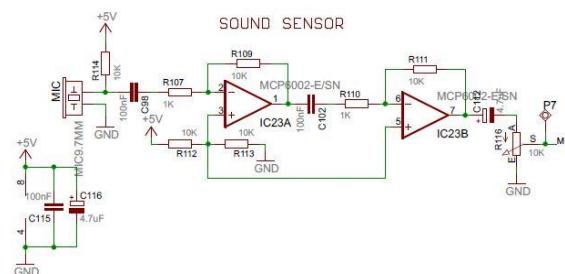


الشكل (3.20) المخطط النظري لبعض الحساسات التشابهية من لوحة التطوير



1.11. الحساس الصوتي:

ويتمثل بمدخل حساس إشارات صوتية باستخدام ميكروفون إلكتروليتي مدمج على اللوحة ، يستقبل الإشارات الصوتية من الوسط المحيط ، ثم عن طريق المبدل التشابهي الرقمي للمعالج .. تتم قراءة التغيرات الصوتية و إظهارها وفق إشارة متغيرة على شاشة رسومية ضمن البيئة البرمجية Labview . الشكل التالي يبين المخطط النظري للحساس الصوتي .

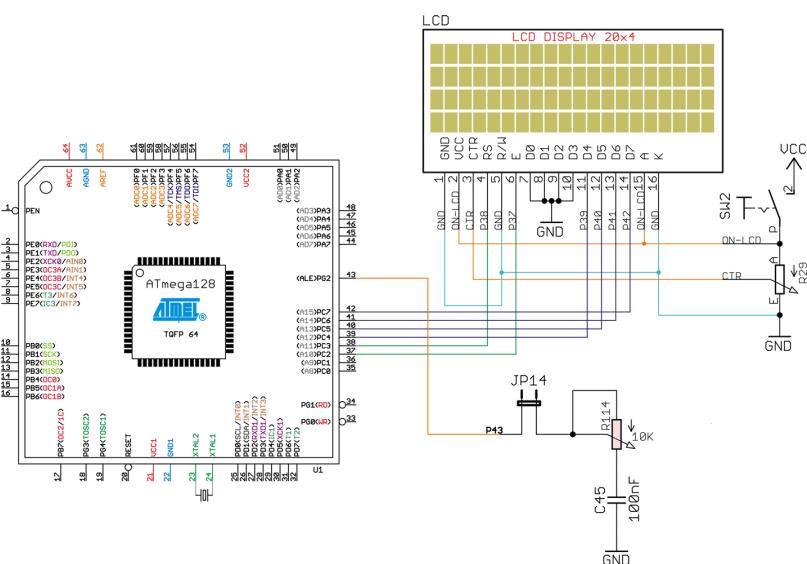
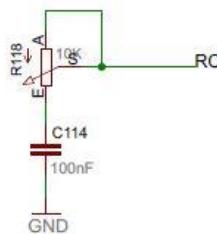


الشكل (3.21) المخطط النظري لدارة الحساسية الصوتية

2.11. الحساس السعوي :

ويتمثل على لوحة التطوير بدارة قياس معامل المقاومة - السعة (R-C) . سوف نقوم بتوصيل مقاومة متغيرة ومكثف متغير مع أحد أقطاب المتحكم المصغر وقراءة قيمة تغير الثابت الزمني لشحن المقاومة والمكثف معاً .

RC CIRCUIT



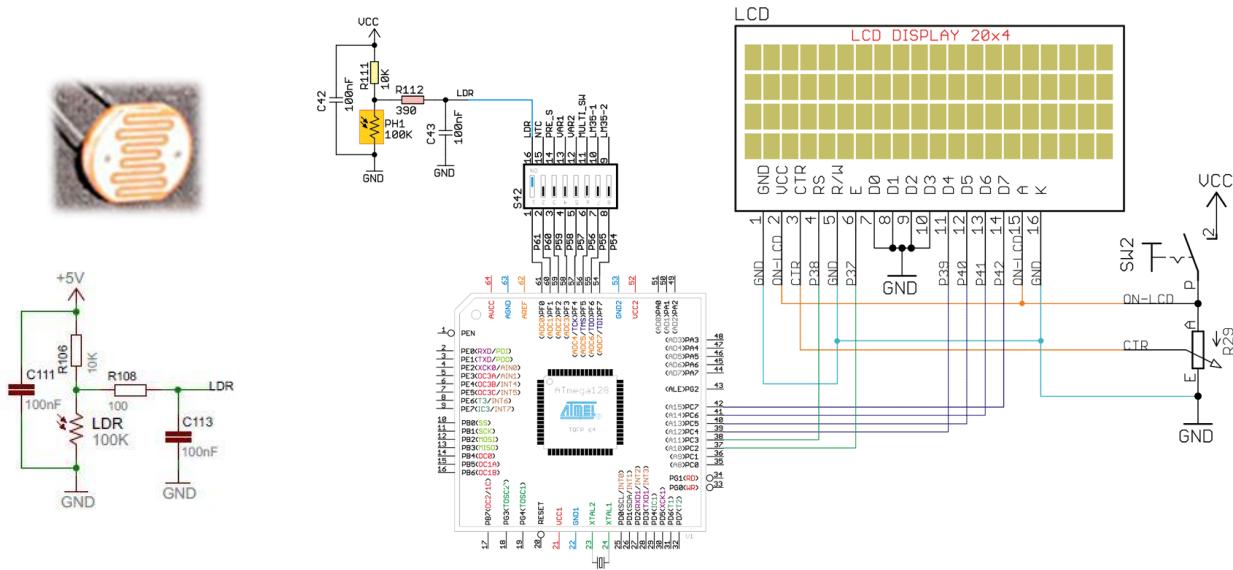


الفصل الثالث

الشكل(3,22) مخطط الحساس السعوي

3.11. حساس شدة الإضاءة:

ويتمثل على لوحة التطوير بدارة قياس شدة الضوء باستخدام المقاومة الضوئية LDR ، حيث يعتمد الحساس على توصيل وبرمجة المقاومة الضوئية إلى قطب المبدل التشابهي الرقمي بهدف قياس شدة الإضاءة للوسط المحيط.



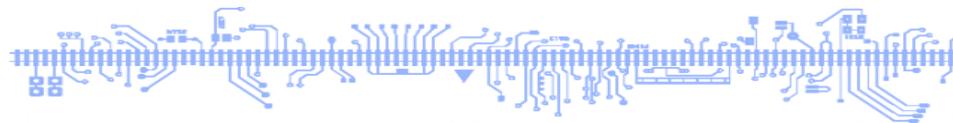
الشكل (3,23) مخطط الحساس الضوئي

شرح عمل الدارة:

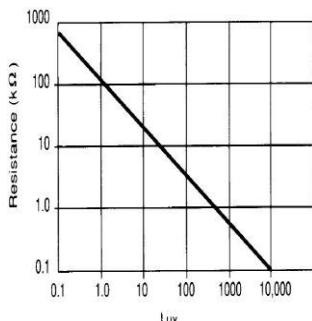
سوف نقوم باستخدام مقاومة ضوئية لقياس شدة الضوء المحيط، وهي تقوم على تحويل الضوء إلى مقاومة. تصنع هذه المقاومات من سلفيد الكاديوم (CDS) حيث تنخفض قيمتها الألومية عند ازدياد شدة الإضاءة حيث تصل قيمتها في الضوء الشديد إلى (100Ω)، وتزداد قيمتها عند انخفاض شدة الإضاءة حيث أنّ قيمتها الأعظمية في الظلام حوالي ($2M\Omega$), وتنستطيع قياس شدة الإضاءة للموجات في المجال $\sim 800\text{nm}$ ~ 350nm ، ولقد تم في البيئة البرمجية وصل الحساس الضوئي بحيث أنه عند تعرض المقاومة للضوء الشديد فإنـ الـ Buzzer يعمل بتزامن مع ظهور رسالة إنذار (LDR Alarm) على شاشة الأظهار، إن العلاقة بين شدة الإضاءة (illumination) وقيمة المقاومة الضوئية (LDR)

$$R_{LDR} = A \times L^{-0.85}$$

تعطى بالشكل التالي:



الجدول التالي يوضح المعطيات والثوابت الأساسية للمقاومة الضوئية :



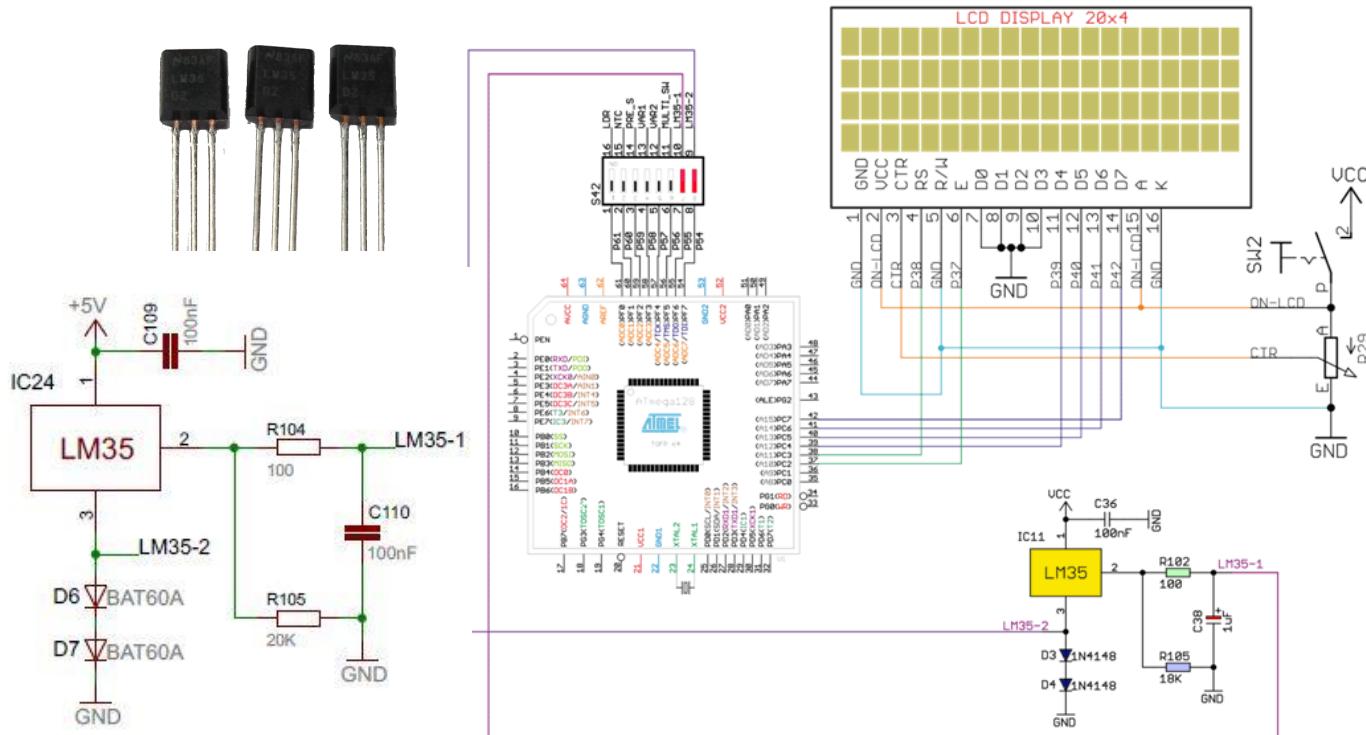
LDR Technical Specifications	
Dark Resistance	1MΩ
Resistance @ 10 Lux	10 – 20 kΩ
Resistance @ 100 Lux	2 – 4 kΩ
Peak Spectral response	540nm
Maximum Voltage	150V peak AC or DC
Power Dissipation	100mW
Operating Temperature	-30°C to + 70°C

الشكل (3.24) الثوابت الأساسية للمقاومة الضوئية

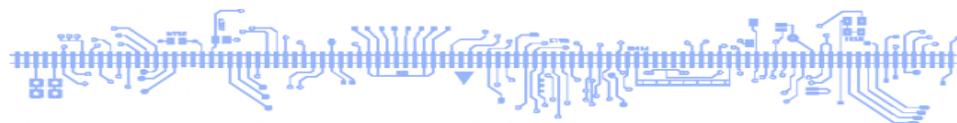
4.11. حساس الحرارة :

ويتمثل على لوحة التطوير بدارة قياس شدة الحرارة باستخدام الحساس التشابهي LM-35 ، استخدامة في تطبيقات قياس درجات الحرارة في المجال من (-45°C ~ +145°C) .

مخطط التوصيل:



الشكل (3.25) مخطط توصيل حساس الحرارة



شرح عمل الدارة:

سوف نقوم باستخدام الحساس التشابهي LM35 لقياس درجات الحرارة وذلك بعد إزاحة النقطة الصفرية للحساس بمقدار (1.2 V) بواسطة ثنائين موصولين على التسلسل بهدف قياس درجات الحرارة السالبة.

إن العلاقة بين درجة الحرارة وجهد خرج الحساس تعطى بالشكل التالي:

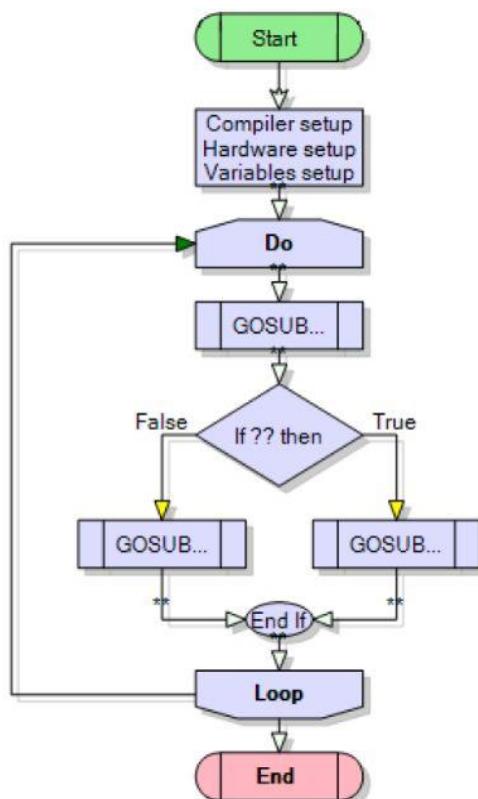
$$1^{\circ}\text{C} \rightarrow 10\text{mV}$$

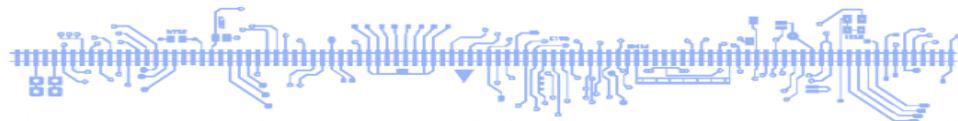
وبالتالي يمكن إيجاد معادلة الحساس والمبدل بالتعويض في معادلة المبدل التشابهي. من أجل درجة مؤوية واحدة يكن الجهد على دخل المبدل 10mV وبالتالي فإن كل درجة حرارة يقابلها:

$$ADC_{val} = \frac{V_{in} \times 1024}{V_{ref}} = \frac{0.01 \times 1024}{2.56} = 4$$

وهذا يعني أنه يكفي أن نقسم قيمة قراءة المبدل على 4 لنحصل على درجة الحرارة الحقيقية.

12. الكود البرمجي : وله الخوارزمية التالية :





▪ تعریف المعالج و الهراءز الكريستالي :

```
[Definitions]
$regfile = "m128def.dat"
$crystal = 8000000
$baud = 9600
```

▪ تعریف أقطاب شاشة الإظهار ..

```
[LCD Configuration]
Config Lcdpin = Pin , Db4 = PORTC.4 , Db5 = PORTC.5 , Db6 = PORTC.6 , Db7 = PORTC.7 , E = PORTG.1 , Rs = PORTG.0 , Wr = PORTG.2
Config Lcd = 16 * 2
Dim I As Integer
```

▪ ضبط باراميترات المبدل التشابهي الرقمي :

```
[ADC Configuration]
Config ADC = Single , Prescaler = Auto , Reference = Internal
Start ADC
Config PORTA = Input
Config PORTB = Output
PORTA = 255
Config PIND.6 = Output : Buzzer Alias PORTD.6 : Reset Buzzer
```

▪ المتحوّلات ..

```
[Variables]
Analog_sensors Alias PINF
Digital_sensors Alias PINA
Dim Dig_input As Byte
Dim Led As Byte
Dim Mic As Word , Lm35_1 As Word , Lm35_2 As Word , Ldr As Word , Rc As Word
Dim Adc1 As String * 4 , Adc2 As String * 4 , Adc3 As String * 4 , Adc5 As String * 4 , Adc6 As String * 4
Dim Flag As Bit
Dim Temp_flag As Bit
```

▪ البرنامج الرئيسي .. و يكتب ضمن حلقة لانهائية :

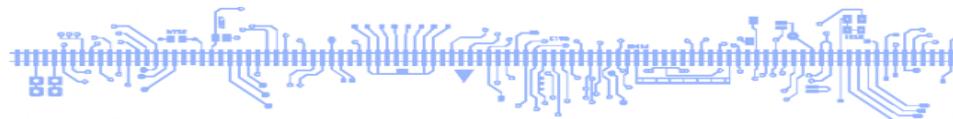
```
>[Main Program]
Cursor Off Noblink
Gosub Display_authors
Do
Waitms 200
Gosub Get_analog
Gosub Get_digital
Gosub Send
Gosub Ready
Loop
End
```

--<[End Main]

▪ اسم المشروع وهو النص الدائم على الشاشة الكريستالية ، حيث يتغيّر النص عند حدوث إنذار للحساس

▪ الضوئي :

```
-->[Display Authors]
Display_authors:
Cls
Locate 1 , 1 : Lcd "DAQ+SCADA system"
Locate 2 , 1 : Lcd "Abdelkader&WALAA"
'Gosub Shift2right
'Gosub Shift2left
Return
```



■ قراءة الحساسات التشابهية :

```
'-----[Get analog Sensor]-----
Get_analog:
Mic = Getadc(1)
Adc1 = Str(mic)
Adc1 = Format(adc1 , "    ")
Lm35_2 = Getadc(2)
Adc2 = Str(lm35_2)
Adc2 = Format(adc2 , "    ")
Lm35_1 = Getadc(3)
Adc3 = Str(lm35_1)
Adc3 = Format(adc3 , "    ")
Ldr = Getadc(5)
Adc5 = Str(ldr)
Adc5 = Format(adc5 , "    ")
Rc = Getadc(6)
Adc6 = Str(rc)
Adc6 = Format(adc6 , "    ")
'
```

■ إرسال البيانات إلى واجهة البرمجة الرسومية :

```
'-----[Send Data to Labview]-----
Send:
Print Adc1 ; Adc3 ; Adc2 ; Adc5 ; Adc6 ;
Printbin Dig_input
Printbin 10
Return
'
```

■ قراءة البيانات من برنامج الواجهة الرسومية :

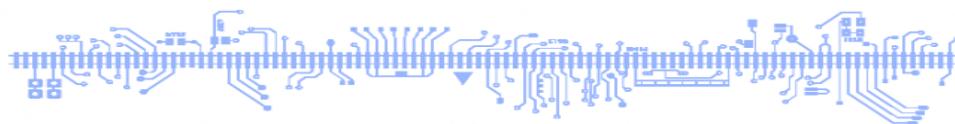
```
'-----[Read Data form Labview]-----
Ready:
Inputbin Led
PORTB = Led
Flag = Led.0
If Flag = 0 Then
  Set Buzzer
  If Temp_flag = 0 Then
    Temp_flag = 1
    Cls : Lcd "      Alarm" : Lowerline : Lcd "      LDR"
  End If
Else
  Reset Buzzer
  If Temp_flag = 1 Then
    Temp_flag = 0
    Cls : Lcd "DAQ SCADA system" : Lowerline : Lcd "Abdelkader&WALAA"
  End If
End If
Return
'
```

■ إرسال قراءة الحساسات الرقمية :

```
'-----[Digital_sensors]-----
Get_digital:
Dig_input = Digital_sensors
Return

Shift2right:
For I = 1 To 8
  Shiftlcd Right : Waitms 500
Next I
Return
'
```





بروتوكولات الاتصال التسلسلي

1. مقدمة عن بروتوكولات الاتصال :

تتفرع بروتوكولات الاتصال بشكل عام إلى فرعين رئيسيين:

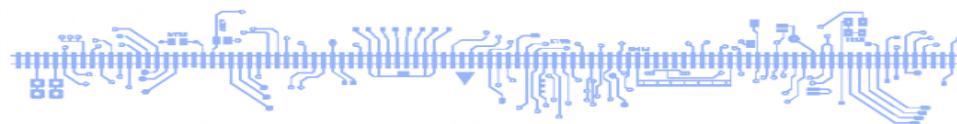
- اتصالات تفرعية.
- اتصالات تسلسلية.

يختصر استخدام الاتصالات التفرعية من أجل نقل البيانات بسرعات عالية جداً ولمسافات قصيرة جداً، والسبب في محدودية المسافة هو تشكيل السعات الطفيلية والضجيج العالي على مسارات خطوط النقل التفرعية عند ارتفاع طول الناقل، كما أن حجم الناقل سيكون كبير وبالتالي فإن كلفة الناقل ستكون كبيرة أيضاً.

تستخدم الاتصالات التسلسليه 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.

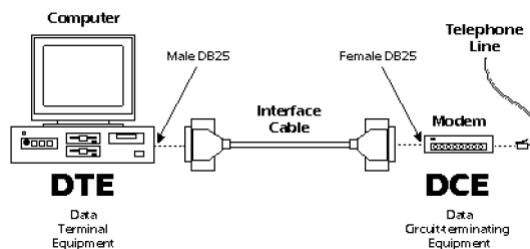
Serial Communications		Parallel Communications
Asynchronous	Synchronous	
<ul style="list-style-type: none"> • Morse code telegraphy • RS-232 (COM Port) • RS-423 • RS-485 • Universal Serial Bus (USB) • FireWire • Ethernet • Fiber Channel • InfiniBand • MIDI • DMX512 • Serial ATA • SpaceWire • PCI Express • SONET and SDH • T-1, E-1 	<ul style="list-style-type: none"> ◦ I2C ◦ SPI ◦ PS2 	<ul style="list-style-type: none"> ▪ LPT ▪ ISA ▪ EISA ▪ VESA ▪ ATA ▪ SCSI ▪ PCI ▪ PCMCIA ▪ IEEE-1284 ▪ IEEE-488

سنوضح فيما يلي أهم مزايا وخصائص كل من بروتوكولات الاتصال التسلسلي المستخدمة في مشروعنا هذا ..



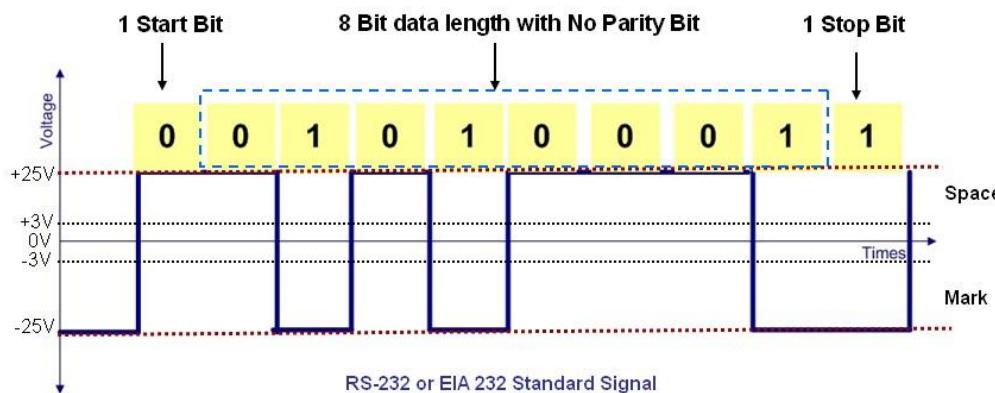
2. بروتوكول الاتصال التسلسلي : RS232

هو عبارة عن بروتوكول اتصال تسلسلي غير متواقيت يستخدم من أجل الربط بين طرفيتين، تسمى الأولى DCE وتشمل الثانية DTE.



الشكل (4.1) بنية البروتوكول RS232

يتم إرسال كل بايت كحزمة مؤلفة من مجموعة برات على الشكل التالي:

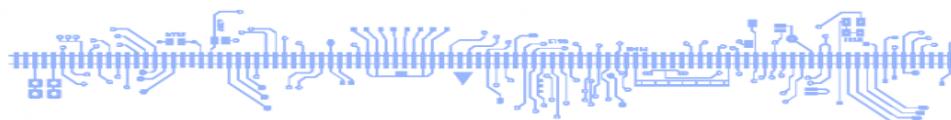


الشكل (4.2) حزمة البروتوكول RS232

كما هو ملاحظ فإن المستويات المنطقية لهذا المعيار مختلفة تماماً عن المعيار TTL حيث أن:

- “0”: المستوى المنطقي المنخفض ويسمى بـ "Space" ويترافق بين $+25V \sim +3V$.
- “1”: المستوى المنطقي العالي ويسمى بـ "Mark" ويترافق بين $-25V \sim -3V$.
- “X”: مستوى منطقي غير معروف ويترافق بين $-3V \sim +3V$.

ملاحظة: إن جهد الدارة المفتوحة يجب أن لا يتجاوز $25V \pm$ بالنسبة للنقطة الأرضية "GND" ، كما أن تيار الدارة القصيرة يجب أن لا يتجاوز $500mA$.



1.2. الميزات والمساوئ لبروتوكول الاتصال RS232

المحاسن (Advantages)	المساوئ (Disadvantages)
بروتوكول اتصال شائع الاستخدام في كثير من التطبيقات ومعتمد من قبل العديد من الشركات.	مناسب فقط من أجل الربط بين System-to-System أكثر من كونه قابلاً للربط بين Chip-2-Chip أو من أجل تطبيقات Chip-2-Sensor.
مسافة الاتصال طويلة نسبياً حوالي 50 قدم عند معدل إرسال منخفض، ويمكن زيادة المسافة باستخدام معدلات نقل منخفضة وتصحيح أخطاء.	معدل نقل بيانات منخفض جداً من أجل مسافة اتصال كبيرة.
مناعة ضد الضجيج بسبب الجهد المرتفع نسبياً (± 25 للمستويات المنطقية ("1", "0").).	يحتاج إلى وحدة تبديل المستوى المنقطي RS232<>TTL.
سهل البناء والبرمجة ومتوفر برمجياً وكبيان صلب.	مخصص للربط بين Single Master/Single Slave غير قابل للتتوسيع.

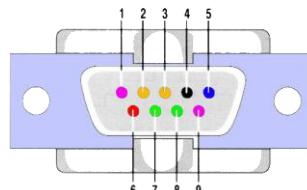
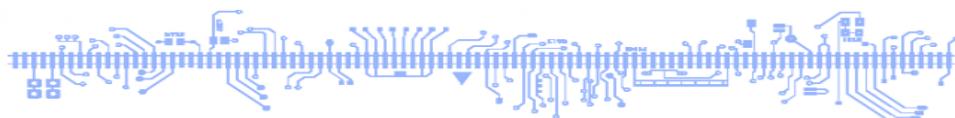
2. عناوين بوابات الاتصال التسلسلي RS232 في الحاسب

يوجد في الحاسب منفذ اتصال تسلسلي وفق المعيار RS232 وتسمى "COM" Serial Port، الجدول التالي يوضح عناوين هذه المنافذ.

Port	Address
COM1	0x3F8
COM2	0x2F8
COM3	0x3E8
COM4	0x2E8

يتوضع منفذ الاتصالات التسلسلي COMx على الوجه الخلفي للحاسب وهو من النوع DB-9Pin كما في الشكل:





الشكل (4.3) منفذ الاتصالات التسلسلي RS 232

يحتوي المنفذ على تسع نقاط (9, 2, 3, ..., 1) وظائفها مبينة في الجدول التالي:

Pin	Name	Direction	Function	Description
1	CD	In	Control	Carrier Detect
2	RXD	In	Data	Receive Data
3	TXD	Out	Data	Transmit Data
4	DTR	Out	Control	Data Terminal Ready
5	GND	---	Ground	System Ground
6	DSR	In	Control	Data Set Ready
7	RTS	Out	Control	Request to Send
8	CTS	In	Control	Clear to Send
9	RI	In	Control	Ring Indicator

:CD – Carrier Detect (Control sent from DCE to DTE)

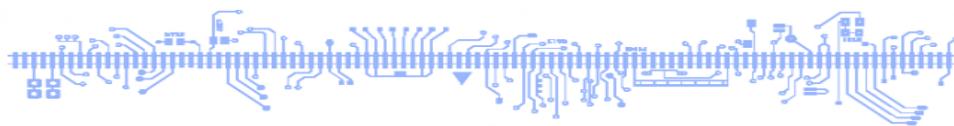
قطب كشف حامل إشارة الرنين ويستخدم فقط في حال استخدام البوابة من أجل ربط بين حاسب وجهاز مودم.

:RxD – Receive Data (Data sent from DCE to DTE)

قطب مدخل استقبال البيانات المرسلة من الطرفية الثانوية (DCE) إلى الطرفية الرئيسية (DTE). فعال (Idle State, “0 or Positive”) عند استقبال البيانات، ويعود إلى نمط البطالة (1 or Negative) عند انتهاء استلام البيانات.

:TxD – Transmit Data (Data sent from DTE to DCE)

قطب خرج البيانات المرسلة من الطرفية الرئيسية (DTE) إلى الطرفية الثانوية (DCE). فعال (Mark state, “1 or Negative”) خلال إرسال البيانات، ويعود إلى نمط البطالة (0 or Positive) عند انتهاء إرسال البيانات.



DTR – Data Terminal Ready (Control sent from DTE to DCE)

قطب تحكم يشير إلى أن الطرفية (DTE) جاهزة للاتصال مع الطرفية الأخرى، فإذا كانت الطرفية الثانية (DCE) في نمط البطالة يقوم بإخراجها إلى النمط الفعال.

DSR – Data Set Ready (Control sent from DCE to DTE)

قطب تحكم يشير إلى أن الطرفية (DCE) في حالة اتصال مع الطرفية الرئيسية (DTE). فعال (‘0’) عند وجود الاتصال، ويعود إلى نمط البطالة (‘1’) فور انتهاء الاتصال.

RTS – Request To Send (Control sent from DTE to DCE)

قطب تحكم يقوم بإعلام الطرفية (DCE) أن البيانات جاهزة للإرسال من الطرفية الرئيسية (DTE)، وبالتالي يمكن استخدام هذه الإشارة من أجل تفعيل دارة الاستقبال قبل إرسال أي إشارة. فعال (‘0’) عندما تكون الطرفية الرئيسية جاهزة لإرسال البيانات، ويعود إلى نمط البطالة (‘1’) فور انتهاء إرسال البيانات.

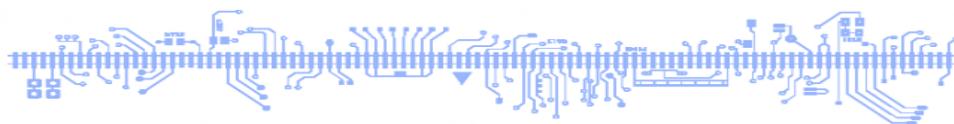
CTS – Clear To Send (Control sent from DCE to DTE)

قطب تحكم يقوم بإعلام الطرفية الرئيسية (DTE) أنه استلم إشارة الإعلام بإرسال البيانات السابقة ويمكنها الآن أن تبدأ بإرسال البيانات إلى الطرفية الثانوية (DCE)، وبالتالي يمكن استخدام هذه الإشارة من أجل تفعيل دارة الاستقبال قبل إرسال أي إشارة. فعال (‘0’) عندما تكون الطرفية الثانوية جاهزة لاستلام البيانات، ويعود إلى نمط البطالة (‘1’) فور انتهاء استلام البيانات.

RI – Ring Indicator (Control sent from DCE to DTE)

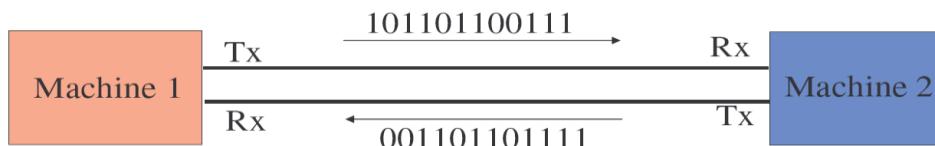
قطب تحكم يقوم بإعلام الطرفية الرئيسية (DTE) بوجود رنين من أجل فتح الخط، ويستخدم فقط في حال استخدام البوابة من أجل ربط بين حاسب وجهاز مودم.





3.2. تحقيق اتصال بين طرفيتين في RS232

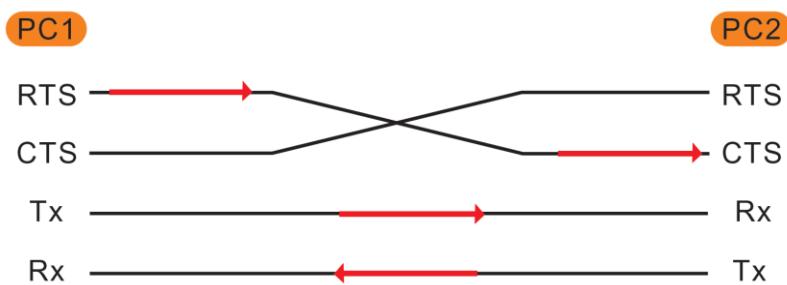
عموماً، فإن من أجل تحقيق اتصال بين طرفيتين بدون مصافحة يكفي توصيل قطب الإرسال "Tx" و/or الاستقبال "Rx" على التوازي المتعاكسي كما في الشكل التالي:



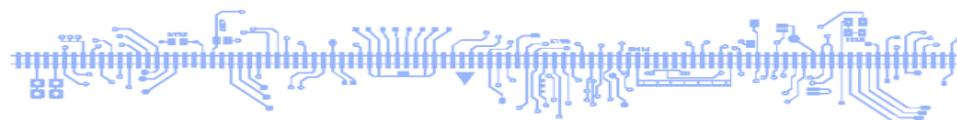
الشكل (4.4) تحقيق إتصال بين طرفيتين في RS 232

أما في حال وجود مصافحة (Hardware handshaking) بين الطرفيتين فإنه يجب توصيل قطبي التحكم المتناوبين (RTS, CTS) بالإضافة لقطبي الإرسال والاستقبال (Tx, Rx)، ويتم التخاطب بين الطرفيتين:

- تقوم الطرفية الأولى بتفعيل أمر التحكم على القطب CTS من أجل إعلام الطرفية الثانية بأنها سوف ترسل بيانات.
- تقوم الطرفية الثانية بالرد على الطرفية الأولى بتفعيل القطب RTS إذا كانت جاهزة لاستقبال البيانات، وإلا يبقى القطب RTS في حالة عدم تفعيل (نمط البطالة).
- في حال كانت الطرفية الثانية مشغولة ولم ترد على طلب الطرفية الأولى فيوجد لدينا حالتين:
 - إما أن تقوم الطرفية الأولى بإعادة الطلب مرة ثانية بعد زمن محدد حتى تحصل على أذن الإرسال.
 - أو أن تقوم الطرفية الثانية بتفعيل القطب RTS فور انتهائها من العملية التي كانت تشغليها، وخلال هذا الوقت تبقى الطرفية الأولى في حالة انتظار رد الطرفية الثانية.



الشكل (4.5) تحقيق إتصال بين طرفيتين في RS 232



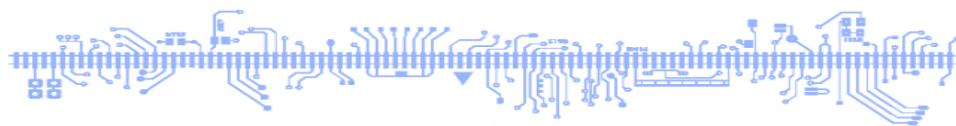
4.2. الموصفات الفنية للبروتوكول RS232

SPECIFICATIONS		RS232	RS423
Mode of Operation		SINGLE-ENDED	SINGLE-ENDED
Total Number of Drivers and Receivers on One Line		1 DRIVER / 1 RECVR	1 DRIVER / 10 RECVR
Maximum Cable Length		50 FT	4000 FT
Maximum Data Rate		20kb/s	100kb/s
Maximum Driver Output Voltage		$\pm 25V$	$\pm 6V$
Driver Output Signal Level (Loaded Min.)	<i>Loaded</i>	$\pm 5V$ to $\pm 15V$	$\pm 3.6V$
Driver Output Signal Level (Unloaded Max)	<i>Unloaded</i>	$\pm 25V$	$\pm 6V$
Driver Load Impedance (Ohms)		3k to 7k	$>= 450$
Max. Driver Current in High Z State	<i>Power On</i>	N/A	N/A
Max. Driver Current in High Z State	<i>Power Off</i>	$\pm 6mA$ @ $\pm 2V$	$\pm 100\mu A$
Slew Rate (Max.)		30V/ μ s	Adjustable
Receiver Input Voltage Range		$\pm 15V$	$\pm 12V$
Receiver Input Sensitivity		$\pm 3V$	$\pm 200mV$
Receiver Input Resistance (Ohms)		3k to 7k	4k min

إن الموصفات القياسية لبروتوكولات الاتصال المذكورة أعلاه توصي باستخدام كابل مزدوج مجدول ويحوي علىield محيط بالعزل الداخلي، ذو سعة نقل 16PF/FT ومانعة مميزة 100Ω 24AWG.



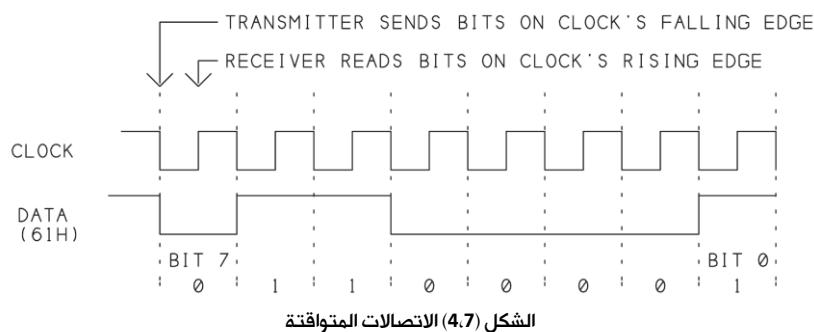
الشكل (4.6) تحقيق إتصال بين طرفيتين في RS 232



5.2 مفهوم الاتصالات التسلسليّة المتزامنة (Synchronized) وغیر المتزامنة (Asynchronous)

أولاً: الاتصالات المتواقة (المتزامنة):

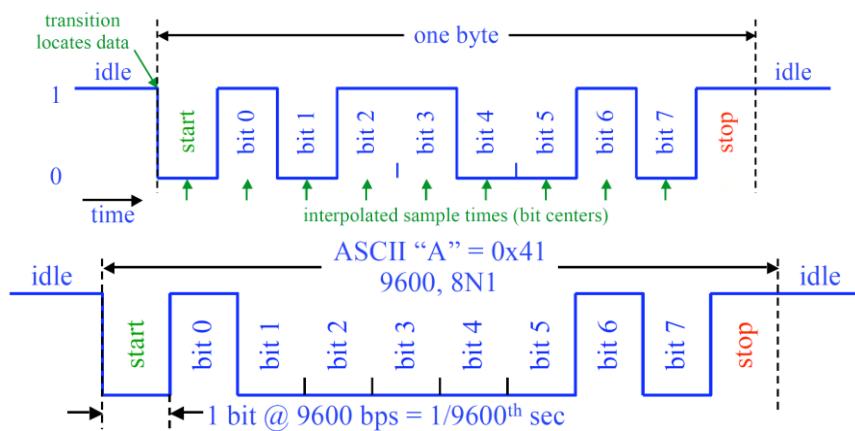
يكون فيها بروتوكول الإرسال مؤلف من خطين على الأقل أحدهما خط التزامن (clock or strobe)، وبالتالي فإن سرعة إرسال البيانات تتحدد من خلال تردد إشارة التزامن بحيث يتم إرسال كل بت من البيانات تسلسلياً عند جبهة التزامن (صاعدة أو هابطة).



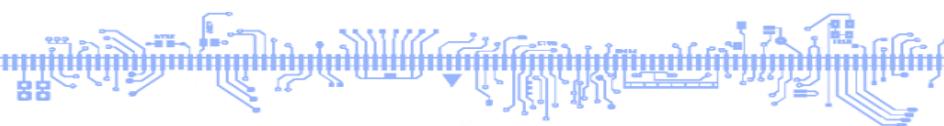
الشكل (4.7) الاتصالات المتواقة

ثانياً: اتصالات غير متواقة (غير متزامنة):

لا تحوي على خط تزامن وإنما يتم بدء عملية الإرسال بـإرسال 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.



الشكل (4.8) الاتصالات غير المتواقة



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

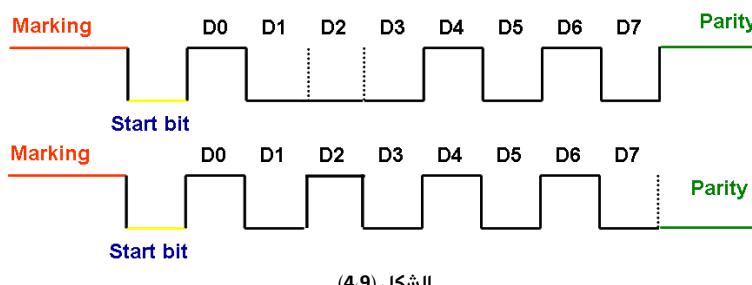
هناك بارامترات يجب تحديدها بين المرسل والمستقبل قبل إرسال البيانات في الاتصالات غير المتواقة وهي:

- ✓ تحديد نمط الإرسال: أحادي الاتجاه (Half-Duplex) أو ثنائي الاتجاه (Full-Duplex).
- ✓ تحديد عدد البتات لكل حرف: 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 برات التوقف (1, 1.5 or 2).

الإرسال أحادي الاتجاه (Half-Duplex): تتم فيه عملية الاتصال بين الطرفيتين باتجاه واحد فقط في نفس اللحظة الزمنية، فإذاً 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، وترتبط خانة الإيجابية بعدد الوحدات في البايت المرسل.



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

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



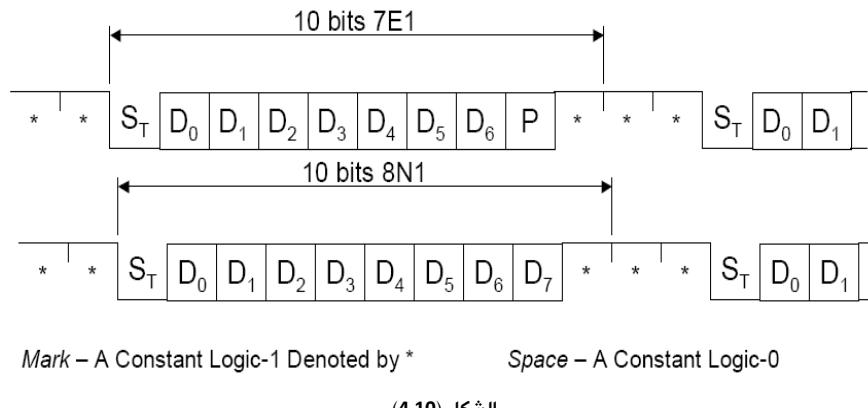
الفصل الرابع

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

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

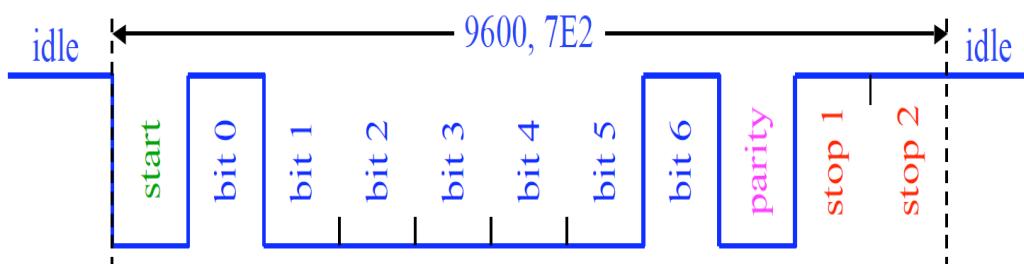
عدد البتات لكل محرف (N):

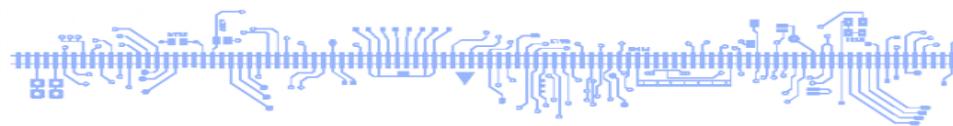
يتم فيها التصريح عن عدد البتات لبيانات التي سيتم إرسالها، فإذاً أن تكون 5, 6, 7 or 8bit ولكن يجب الانتباه مثلاً في حال إرسال $N=7$ bit فإن قيمة العظمى ASCII = 127.



خاتمة بit التوقف (Stop Bit)

يعلم المرسل من خلالها المستقبل بانتهاء عملية الارسال. 1.5 or 2, 1 بت.





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...

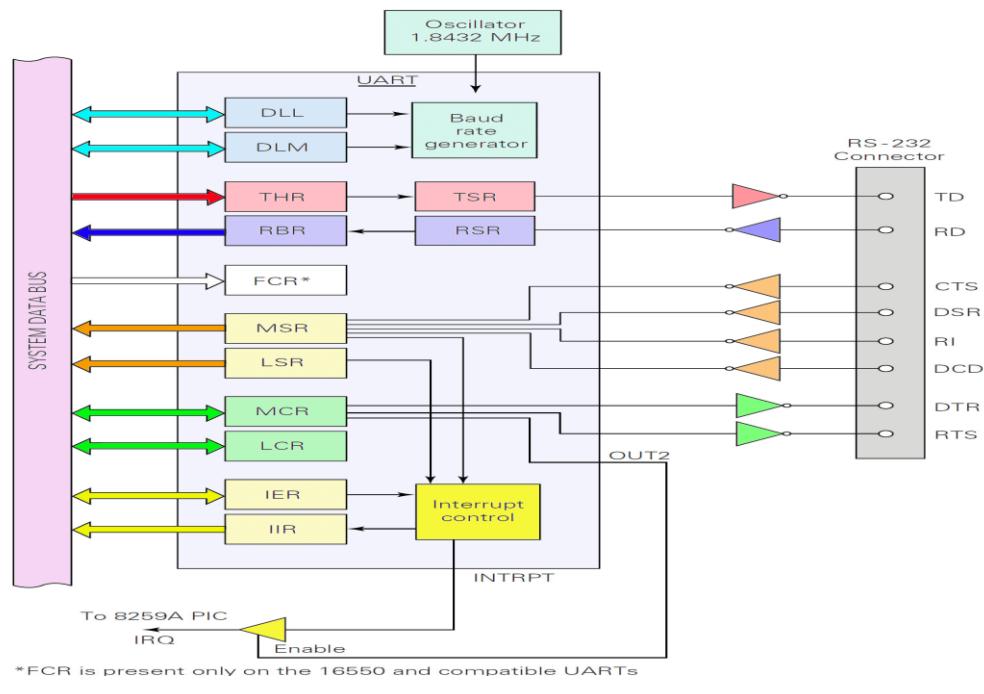
إن الزمن اللازم لإرسال بت واحد يعطى بالعلاقة التالية:

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

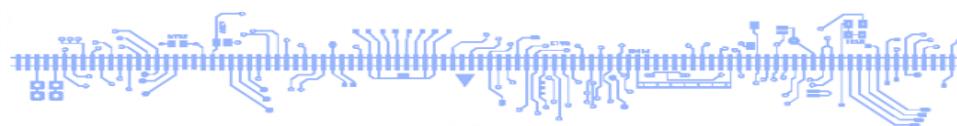
إن عدد البايتات التي يمكن إرسالها خلال ثانية واحدة يمكن حسابها من العلاقة التالية:

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

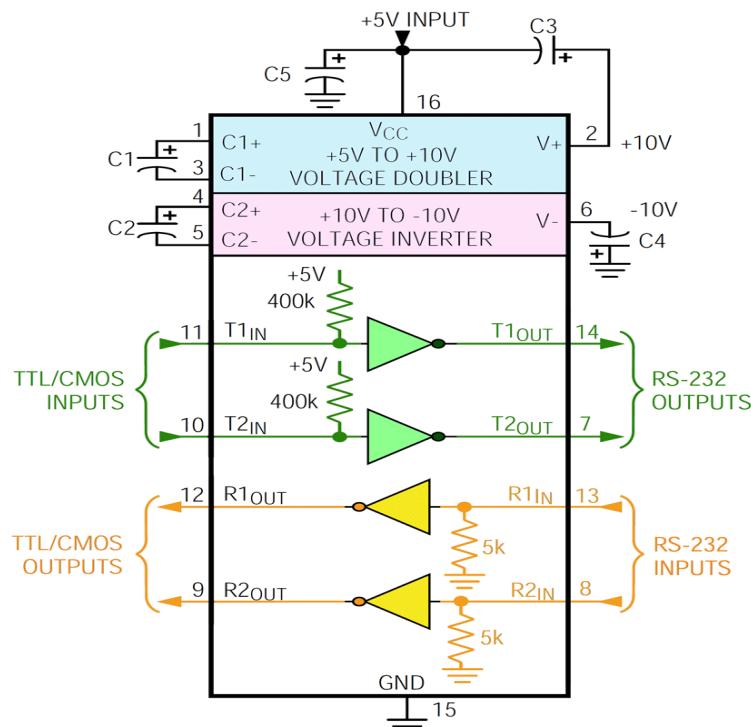
6.2. بنية النافذة التسلسلي RS232 في الحاسب:



(4.12) الشكل



7.2 دارات الملائمة :TTL <> RS232

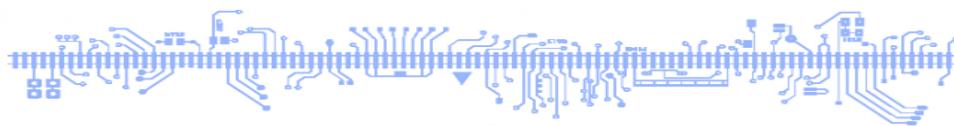


(4.13) الشكل

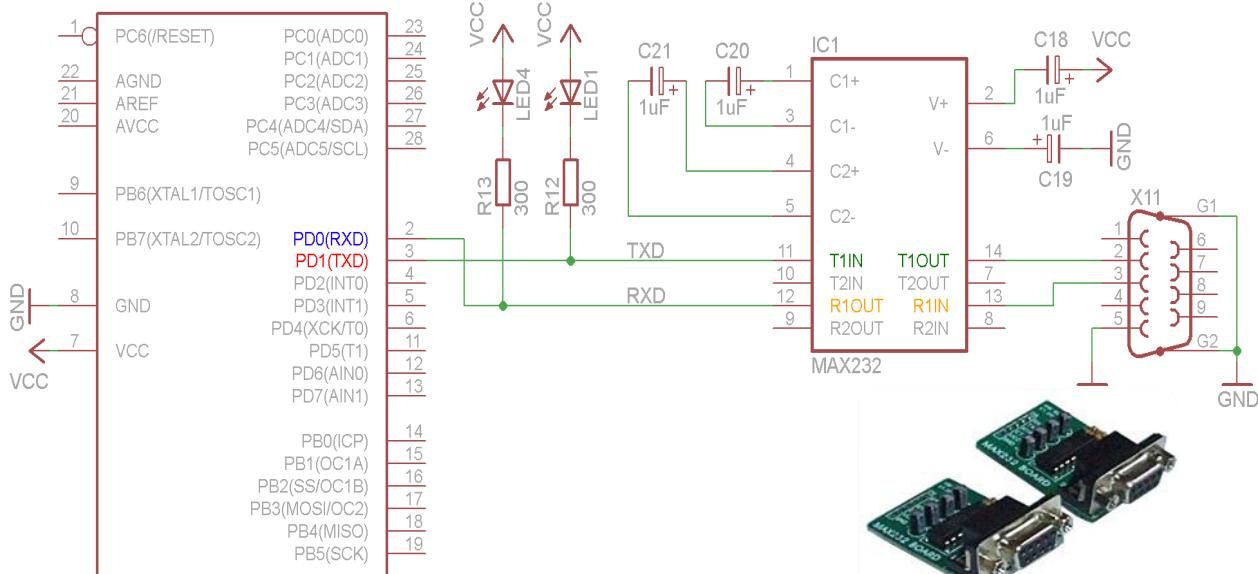
إن المستويات المنطقية للبروتوكول RS232 تختلف عن المستويات المنطقية للمتحكمات المصغرة وللدارات الرقمية الأخرى التي تعتمد المنطق TTL في عملها، وبالتالي نحتاج إلى دارة وسيطية (Adapter) من أجل الملائمة بين الطرفين.

تستخدم الدارة المتكاملة Max232 كدارة تحويل وعزل .TTL<>RS232

الشكل التالي (4.14) يبين طريقة تحقيق دارة ملائمة TTL<>RS232 بين منفذ الحاسب التسلسلي (RS232) وبين نافذة تسلسليه (UART) لمتحكم مصغر باستخدام الدارة المتكاملة Max232.



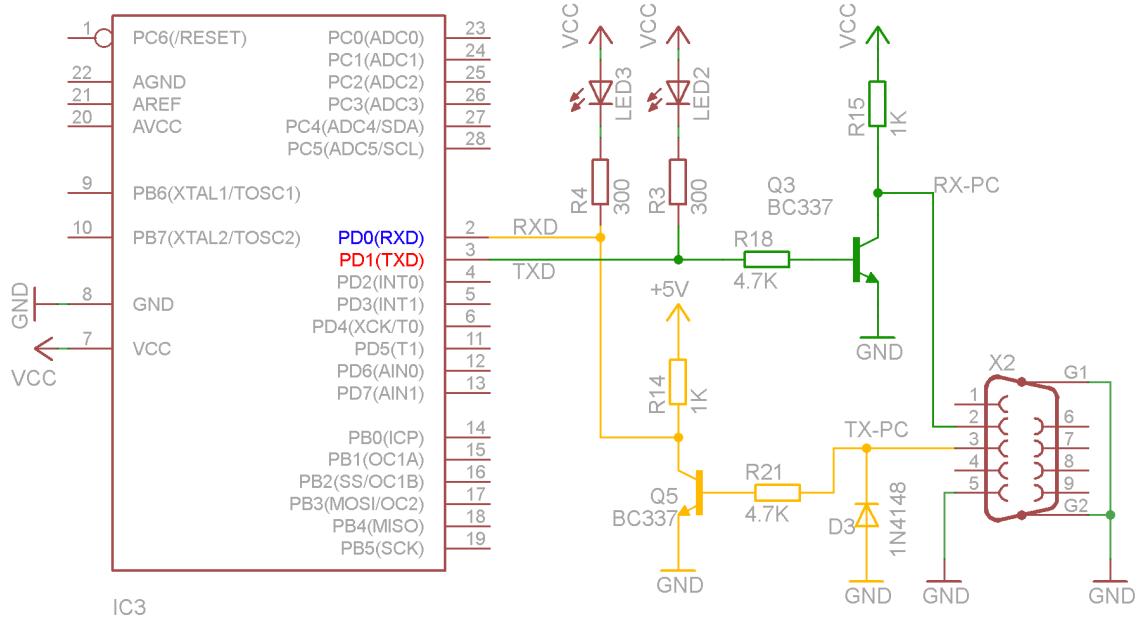
MEGA8-P



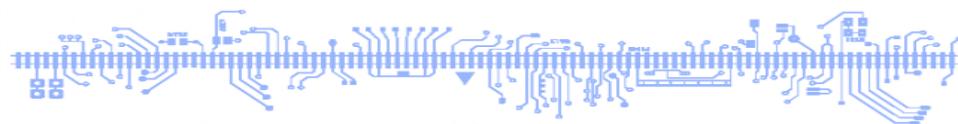
(4.14)

الشكل التالي يبين طريقة تحقيق دارة ملائمة TTL \leftrightarrow RS232 بين منفذ الحاسب التسلسلي (RS232) وبين نافذة تسلسليه (UART) لمتحكم مصغر باستخدام وصلة مفاتيح ترانزستورية.

MEGA8-P



(4.15)



3. بروتوكول الاتصال التسلسلي USB

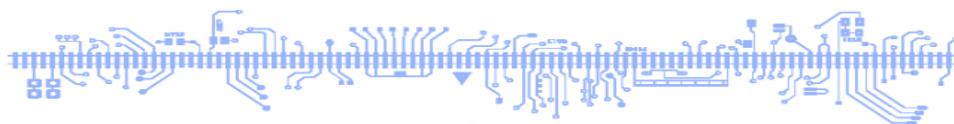
معظم الأجهزة الطرفية الحديثة أصبحت تستخدم منفذ USB كطريقة ربط مع الحاسب لما له من مميزات على الأنواع الأخرى.

1.3. مقارنة بين بروتوكولات الاتصال التسلسلي الشائعة الاستخدام:

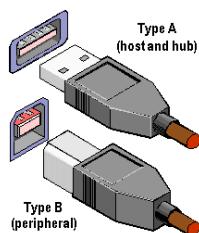
الاستخدامات	صيغة الاتصال	السرعة الأعظمية bits/sec	طول قناة الاتصال (قدم)	عدد الأجهزة على الممر	البروتوكول
لوحة المفاتيح، الفأرة، المودم	تسلسلي غير متوازن	1.5M, 12M, 480M	16	127	USB
الفأرة، المودم تجهيزات أخرى	تسلسلي غير متوازن	20k - 115k	50-100	2	RS-232
الشبكات الصناعية	تسلسلي غير متوازن	10M	4000	-	RS-485
الطبعات، الأجهزة الكافية	تسلسلي غير متوازن	115k	6	2	IrDA
المتحكمات المصغرة	تسلسلي متوازن	2M	10	8	Microwire
المتحكمات المصغرة	تسلسلي متوازن	2.1M	10	8	SPI
المتحكمات المصغرة	تسلسلي متوازن	3.4M	18	40	I²C
الفيديو، وحدات التخزين	تسلسلي	400M-3.2G	15	64	FireWire
الشبكات الصناعية	تفرعي	8M	60	15	IEEE-488
شبكات الحاسب	تسلسلي	10M/100M/1G	1600	1024	Ethernet
أجهزة موسيقية	تسلسلي	31.5k	50	2	MIDI
طبعات، ماسحات ضوئية	تفرعي	8M	10-30	2	LPT

2.3. ميزات منفذ الاتصالات التسلسلي USB:

- ✓ السرعة العالية التي تصل إلى 480Mb/s.
- ✓ دعمه لتقنية Plug & Play حيث يمكن وصل الجهاز دون الحاجة لإعادة إقلاع الجهاز.
- ✓ عدد أقل من الخطوط، لأن التقنية تسلسليّة تستخدم فقط أربع خطوط.
- ✓ إمكانية ربط عدة طرفيات حتى 127 طرفية على نفس الممر.



3.3. الموصفات الميكانيكية والكهربائية لمنفذ USB:



تحدد معايير USB نوعين من المأخذ :

- مأخذ نوع A: يوجد على الحاسوب المضيف.
- مأخذ نوع B: يوجد على الطرفية.

يمنع هذا الاختلاف في الشكل بين المأخذين من وصل طرفيتين أو جهازي حاسب مع بعضهما البعض.

يستخدم ممر USB أربع خطوط: خطٌ تغذية **+5v** & **GND**, خطٌ معطيات **D-** & **D+**.

		<table border="1"> <tr> <td>VCC (أحمر)</td><td>1</td></tr> <tr> <td>D- (أبيض)</td><td>2</td></tr> <tr> <td>D+ (أخضر)</td><td>3</td></tr> <tr> <td>GND (أسود)</td><td>4</td></tr> </table>	VCC (أحمر)	1	D- (أبيض)	2	D+ (أخضر)	3	GND (أسود)	4
VCC (أحمر)	1									
D- (أبيض)	2									
D+ (أخضر)	3									
GND (أسود)	4									

يستخدم ممر USB الخطين **D+&D-** لإرسال البيانات بشكل تفاضلي، فمثلاً لإرسال '1' منطقي بشكل تفاضلي على الممر يجب وضع '1' منطقي على القطب **D+**، '0' منطقي على القطب **D-** وذلك في حال السرعة الكاملة والمنخفضة للممر، وينعكس هذا التشفير في حال السرعة العالية.

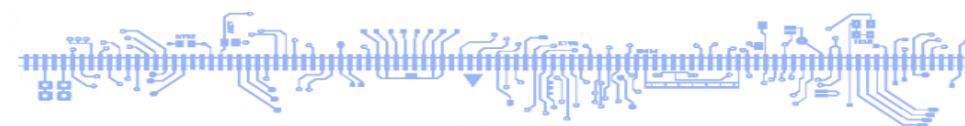
4.3. طريقة عمل البروتوكول USB:

يتبادل منفذ USB البيانات من خلال إطارات (Frame)، لكل إطار فترة زمنية ثابتة تتغير بحسب نوع المنفذ (USB1.1, USB2.0) والطرفية المستخدمة، فمثلاً يكون طول الإطار (1ms) في حالة كانت السرعة المستخدمة هي سرعة منخفضة أو كاملة، في حين يكون عند السرعة العالية (125μs).

يحوي كل إطار عدد من عمليات تداول البيانات (Transaction) وكل عملية تداول تتتألف من مجموعة من رزم البيانات.

تقسم هذه الرزم إلى الأنواع التالية:

ترسل في بداية كل عملية تداول للبيانات وتحوي معلومات عن عملية التداول المراد إجراؤها وتقسم إلى أربعة أنواع: **Token Packet**



In Token ♦: تقوم هذه الرزمة بإخبار الطرفية بأن المضيف يريد قراءة بيانات، وتتألف من

خمسة حقول:

1. **Synchronization**: تتألف من ثمانية بتات وتوجد في بداية كل أنواع الرزم، وتستخدم لمواقة المرسل مع المستقبل.

2. **PID**: يتألف من ثمانية بتات تستخدم للدلالة على نوع الرزمة المرسلة، وتكون ثاني أربع خانات هي متممة لأول أربع خانات وذلك للتأكد من أن البيانات صحيحة.

3. **Address**: يحوي هذا الحقل عنوان الطرفية المطلوبة ويتألف من سبع خانات مما يسمح بعنونة 128 جهاز.

4. **Endpoint**: يتألف من أربع خانات تدل على رقم النقطة النهائية المراد مخاطبتها.

5. **CRC**: بطول 5 بت من أجل كشف الأخطاء.

Out Token ♦: تقوم هذه الرزمة بإخبار الطرفية بأن المضيف يريد إرسال بيانات، وتتألف من

نفس حقول الرزمة السابقة (In).

Control Transfer : **Setup Token** ♦: تستخدم هذه الرزمة للدلالة على بداية عملية تهيئة

.In وهي تتألف من نفس حقول رزمة

Start of frame Token ♦: يقوم المضيف بإرسال هذه الرزمة في بداية كل إطار أي كل

، وتحتوي هذه الرزمة على رقم الإطار ضمن حقل (Frame number) المؤلف من

11 خانة.

In	SYNC 0x01	PID 0x96	Address 7 bits	End point 4 bits	CRC 5 bits
Out	SYNC 0x01	PID 0xE	Address 7 bits	End point 4 bits	CRC 5 bits
Setup	SYNC 0x01	PID 0xD2	Address 7 bits	End point 4 bits	CRC 5 bits
Start of frame	SYNC 0x01	PID 0x5A	Frame number 11 bits		CRC 5 bits

(4.16) الشكل

(Data Packet): وتقسم إلى نوعين:

1. رزمة بيانات من نوع Data0

2. رزمة بيانات من نوع Data1



لكل نوعي رزم البيانات حقل بيانات بطول 1024bit وحقل CRC بطول 16bit وذلك بسبب كبر هذه الرزمة.

Data 0	SYNC 0x01	PID 0x3C	Data 0-1023 bits	CRC 16 bits
Data 1	SYNC 0x01	PID 0xB4	Data 0-1023 bits	CRC 16 bits

(4.17)

رزمة المصافحة (Handshake Packet): ولها ثلاثة أنواع:

Acknowledge: وتدل على أن عملية التداول قد تمت بنجاح وأنه تم استقبال البيانات بشكل صحيح.

Not acknowledge: ترسل هذه الرزمة للدلالة على أن الجهاز مشغول لا يمكنه بشكل مؤقت إرسال أو استقبال البيانات، وترسل أيضاً للمضيف خلال عمليات التداول الخاصة بالمقاطعة لإخباره بعدم وجود بيانات.

Stall: تدل هذه الرزمة على أن النقطة النهائية في حالة توقف بسبب مشكلة ما وتحتاج تدخل المضيف أو أن الأمر المرسل غير مدعوم.

Ack	SYNC 0x01	PID 0x2D
Nak	SYNC 0x01	PID 0xA5
Stall	SYNC 0x01	PID 0xE1

(4.18)

5. 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 الحاسوب 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.



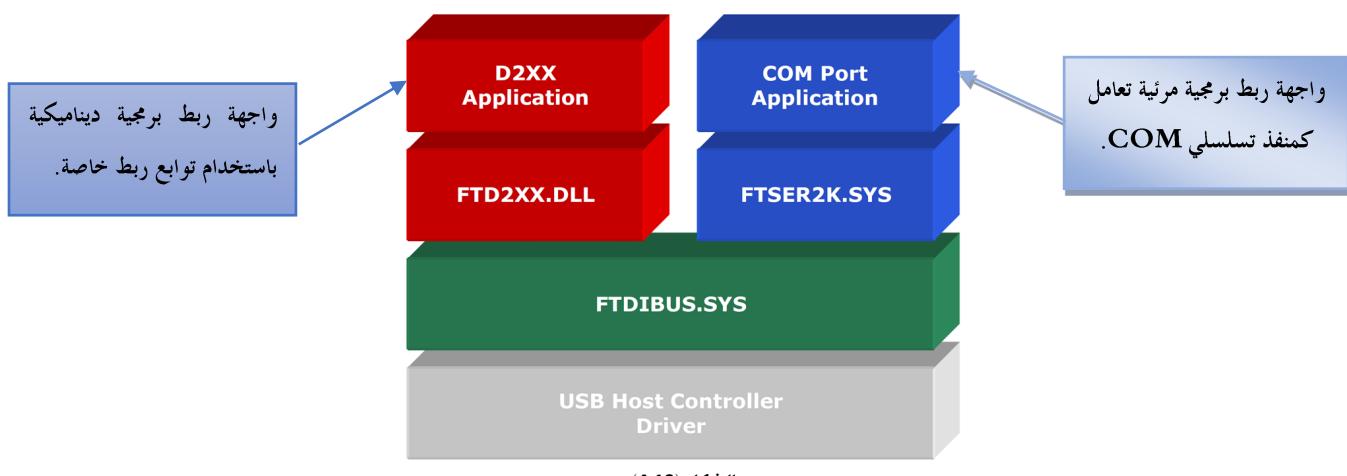
لذلك وبسبب الطلب المتزايد على هذه التقنية واقتحامها للسوق العالمية فإن هناك الكثير من الشركات التي وفرت على المصممين عناء تصميم العتاد الالكتروني ليت椿 اهتمامهم على كتابة برامج القيادة، لذلك كل ما يتوجب على المصمم هو الإطلاع على معايير USB بعرض فهم كيفية التعامل مع هذا العتاد الالكتروني.

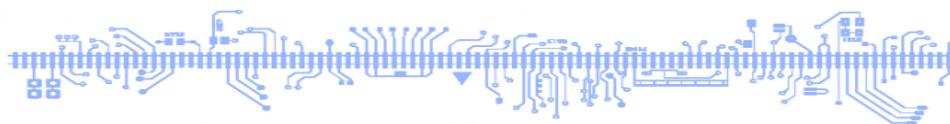
تقدم بعض الشركات حلولاً للتعامل مع المنفذ USB باستخدام شرائح متكاملة تقوم على تحويل البروتوكول USB إلى نافذة تسلسلية UART تمكن المستخدم من توصيل المتحكم المصغر بشكل مباشر مع هذه النافذة، بالإضافة إلى ذلك توفر هذه الشرائح حلولاً برمجية من خلال مكتبات ربط ديناميكية من أجل ربط نظام مع الحاسب عن طريق البروتوكول USB ومعالجة بaramترات 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 شركة FTDI.

إن عملية تحويل البروتوكول USB تم بنائهما 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 الرابط الديناميكي للشريحة دون الحاجة إلى بناء البروتوكول USB بشكل برمجي من البداية أو حتى فهم مبدأ عمله.

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





فيما يلي جدول مقارنة بين واجهتي الربط : (D2XX driver & VCP driver)

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

▶ **(Virtual Com Port) VCP :** يعرف منفذ USB كمنفذ Com تسلسلي إضافي، مما يسمح لنا

بالاتصال 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.

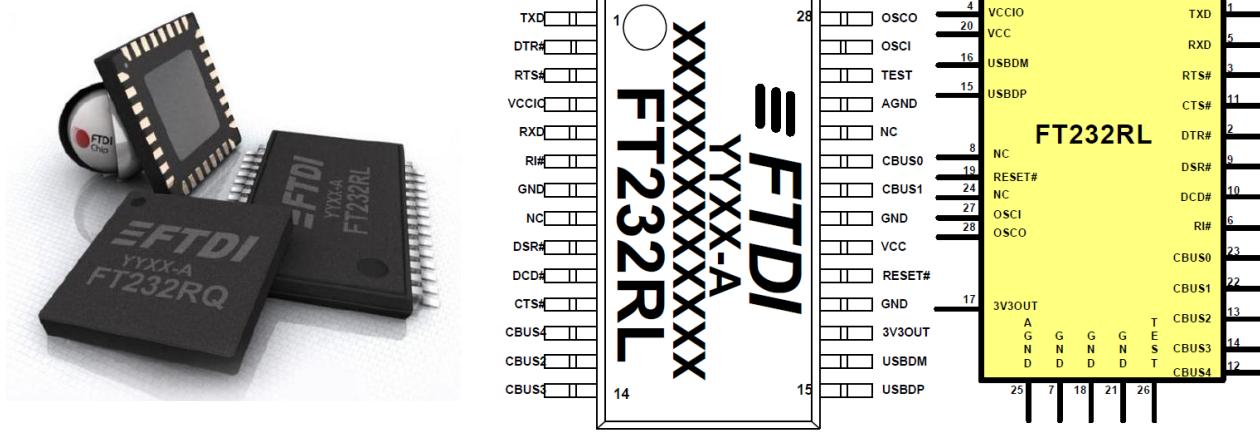
6.3 الشريحة FT232BM

- ✓ توفر الشركة الصانعة برنامج القيادة لهذه الشريحة بشكل مجاني متواافق مع معظم أنظمة التشغيل.
- ✓ تقدم شركة FTDI برنامجي قيادة لشرائحتها (VCP & D2XX.DLL).
- ✓ متواقة مع المعايير USB1.1, USB2.0.
- ✓ تدعم هذه الشريحة ملائمة كاملة لنظم الاتصالات التسلسلي.
- ✓ 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.

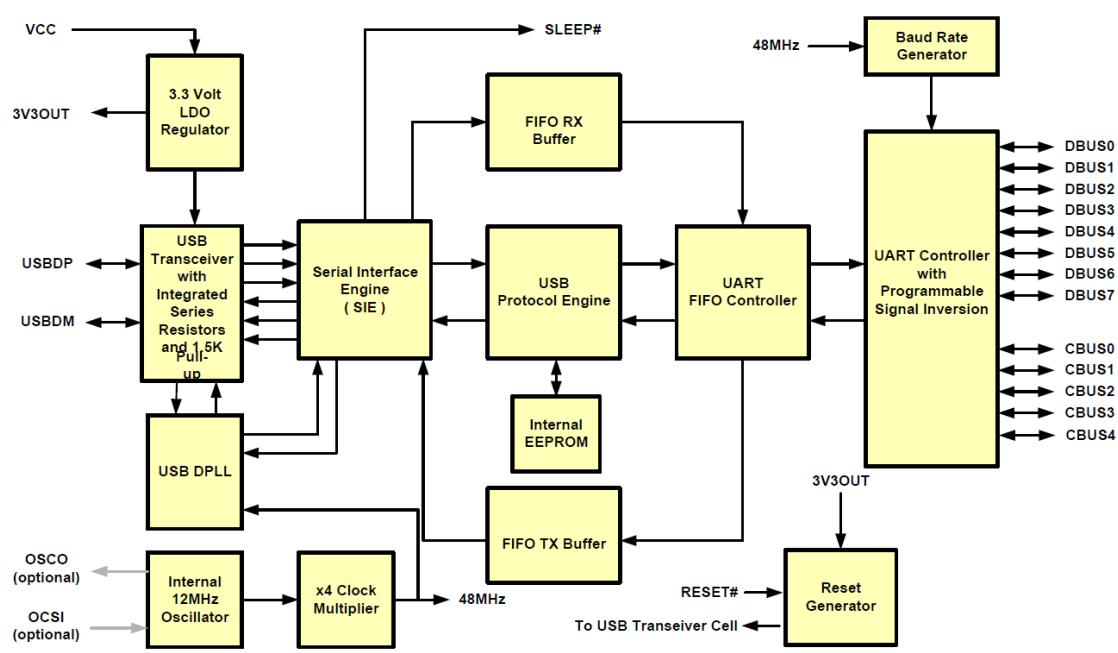


تلعب هذه الشريحة دور الملائم بين منفذ USB وبين النظام حيث تقوم باستقبال بيانات منفذ USB وتنقلها البيانات المطلوبة، كما تقوم بإرسال البيانات من المتحكم بشكلها التسلسلي إلى منفذ USB بعد إضافة الحقول اللاحقة لتحقيق بروتوكول USB.

7.3. توزيع أقطاب الشريحة : FT232BM



8.3. المخطط العام لبنية هذه الشريحة مع العناصر الأساسية :





تمتلك هذه الشريحة 11 قطب للوصل مع النظام بحيث تؤمن تبادل البيانات مع الحاسب باستخدام بروتوكول المصادفة.

عند وصل الشريحة مع الحاسب وبعد أن يقوم نظام التشغيل بتحميل برنامج القيادة لها، تقوم هذه الشريحة بإعطاء صفر منطقي على القطب #PWREN وبالتالي يمكن استخدام هذا القطب لتشغيل الدارة الخارجية عند وصل النظام بالحاسب.

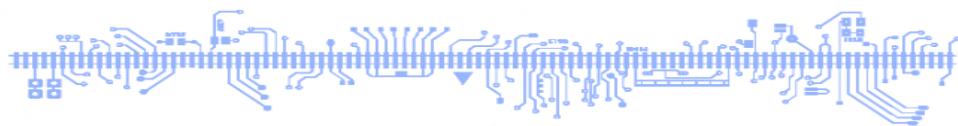
تدخل الشريحة في نمط الطاقة المنخفضة (Sleep mode) إذا لم تكن هناك عملية تبادل بيانات لمدة 3ms (أي بطول ثلاثة إطارات)، في هذه الحالة تصبح حالة القطب "Sleep=1" ، وبالتالي فإن ربط هذا القطب مع المتحكم بطريقة مناسبة يمكن أن يدخله في نمط الطاقة التحتية أيضاً.

تمتلك هذه الشريحة قطب Wake up يسمح للنظام بإخراج الحاسب من الوضع الاحتياطي عند ورود جبهة صاعدة وذلك في حال كان النظام في حالة وضع احتياطي، أما إذا لم يكن كذلك فإن هذه الجبهة الصاعدة تؤدي إلى إرسال البيانات الموجودة في ذاكرة الاستقبال إلى الحاسب.

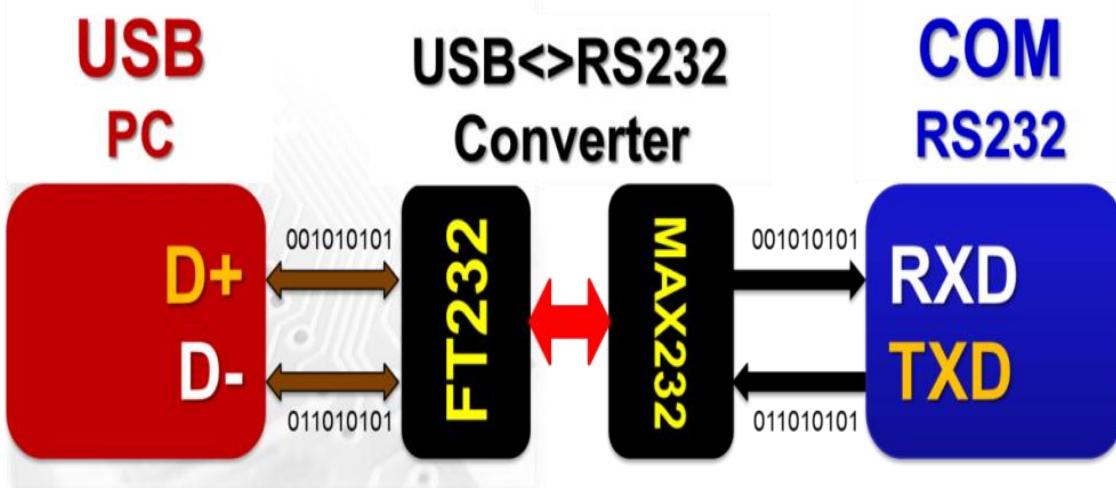
يتم تحقيق المتطلبات الخاصة لبروتوكول التخاطب مع منفذ USB في الشريحة من قبل وحدة الملائمة التسلسليّة (Serial Interface Engine) SIE التي تقوم بالمهام التالية:

- كشف الرزم المستقبلة وإرسال الرزم من وحدة بروتوكول USB إلى مرسل USB.
- فحص و توليد قيمة CRC.
- كشف و توليد إشارات Start Of Packet, End Of Packet, Resume, Reset.
- تشفير وفك تشفير البيانات على الممر بحسب تقنية NRZI.
- فك تشفير وتوليد معرفات الرزم (PID).

تقوم وحدة USB Transceiver (USB Transceiver) بملائمة الشريحة مع الممر وفقاً للشروط الكهربائية المحددة للمعايير USB2.0, USB1.1، بما في ذلك تعريف سرعة الشريحة على الممر باستخدام مقاومة شد على القطب D+ لأن سرعة الشريحة تصنف كسرعة كاملة.



9.3 ربط منفذ USB (Differential) مع منفذ RS232 COM .(Differential) USB (RS232 COM) مع منفذ



الشكل (4.22)

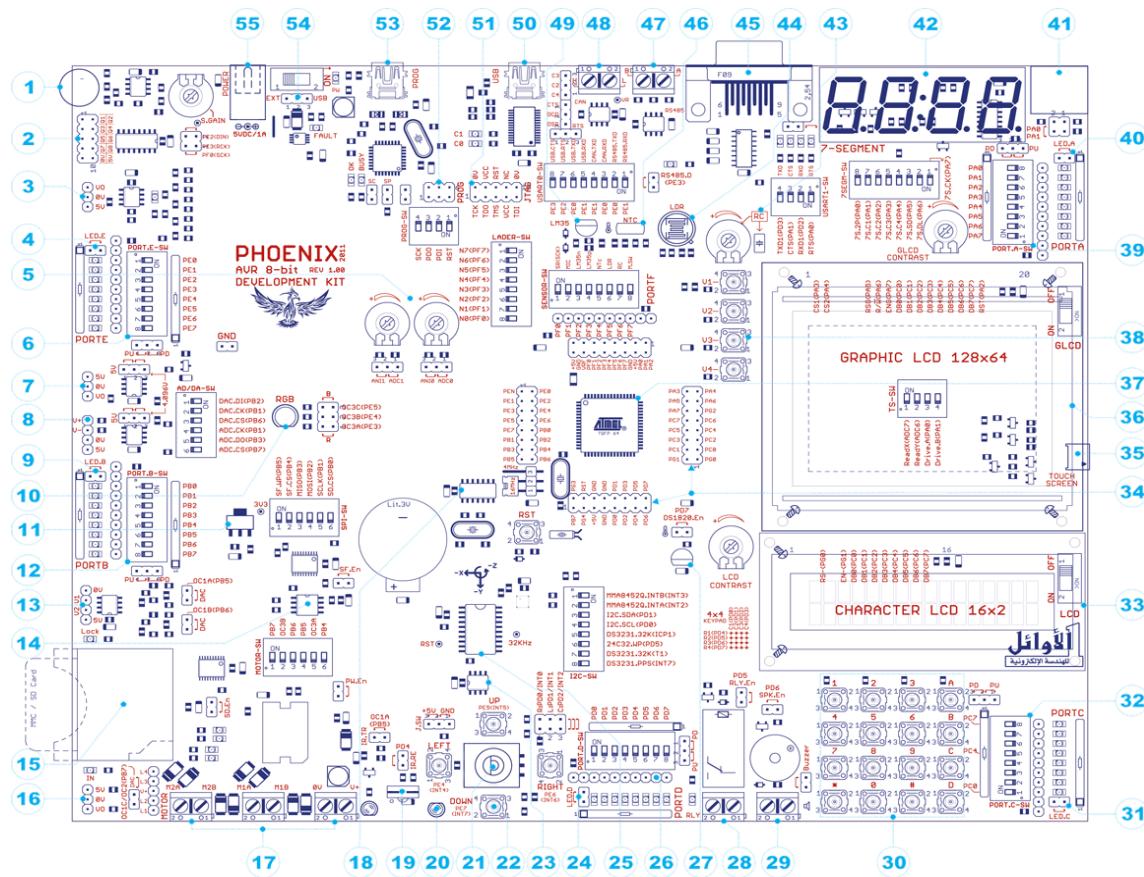




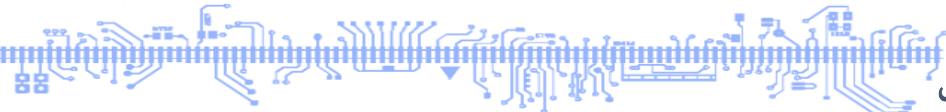
لوحة التطوير Phoenix-AVR

1. الهدف الرئيسي من لوحة التطوير:

على الرغم من وجود العديد من لوحات التطوير التجارية المتوفرة عالمياً - هو تعزيز الجانب العملي التطبيقي في مختبراتنا الجامعية والطلابية وذلك من خلال توفير موارد عملية تطبيقية ملائمة - تم تصميمها وإنتاجها محلياً - ترافق وتعزز الأسس النظرية، الأمر الذي سينشأ عنه جيل جديد من المهندسين متعدد المعرفة والإمكانات قادر على تحمل أعباء الثورة التكنولوجية الحديثة. لقد تم تصميم هذه اللوحة خصيصاً بحيث تخدم المبتدئ والمتقدم في تعلم برمجة المتحكمات المصغرة من العائلة AVR، حيث تضم أكثر من 55 وحدة محيطية على نفس اللوحة لتغطي ما يقارب 100 تجربة أساسية، وقد تصل إلى أكثر من 200 تجربة بالدمج بين الوظائف المحيطية، إضافة إلى إمكانية ربط وحدات خارجية عن طريق وحدات التوسعة الموزعة على أطراف اللوحة.



الشكل (5.1) توزيع المحيطيات على لوحة التطوير



لقد تم تصميم هذه اللوحة بعد دراسة استطلاعية شاملة بحيث تضم اللوحة جميع المحيطيات على لوحة واحدة وبسعر زهيد مقارنة مع لوحات التطوير المتوفرة عالمياً والتي صممته من منطلق تجاري وسعيرها ثلاثة أضعاف سعر لوحة التطوير Phoenix-AVR والتي تعتبر اللوحة الأشمل التي تنافس بجودتها عالمياً بين لوحات التطوير AVR المتوفرة عالمياً.

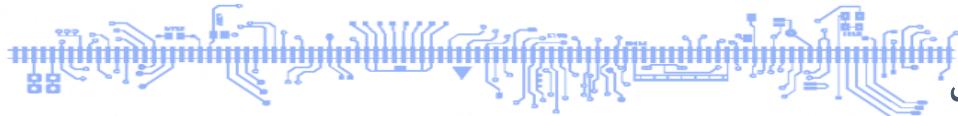
تتميز لوحة التطوير Phoenix-AVR بأنها تحوي على مبرمجة USB-ASP على نفس اللوحة، كما أنها يمكن أن تعمل من خلال تغذيتها عن طريق منفذ USB دون الحاجة إلى وصل المحول الخارجي، الأمر الذي يمكن من استخدامها خلال قاعة الدرس بربطها مباشرة مع الحاسب المحمول، كما أنها تملأ دائرة متكاملة للحماية من تيار القصر عند تغذيتها من منفذ USB وذلك لحماية المنفذ والهاسب. كذلك تملك اللوحة العديد من الوحدات المحيطية الغنية مثل: الشاشة الرسومية المرفقة بشاشة لمس أومية - حساس التسارع ثلاثي المحاور - دائرة قيادة محركات خطوية ومحركات تيار مستمر - حساس موضع دوار - مبدلات رقمية تشابهية وتشابهية رقمية - ذواكر وحساسات تغطي جميع بروتوكولات الاتصال التسلسلي (1-wire, I2C, SPI, USART, CAN, RS485, USB, RS232...) والعديد من الوحدات الأخرى موضحة على توزع المحيطيات على لوحة التطوير Phoenix AVR.

2. الوحدات المحيطية للوحة التطوير :

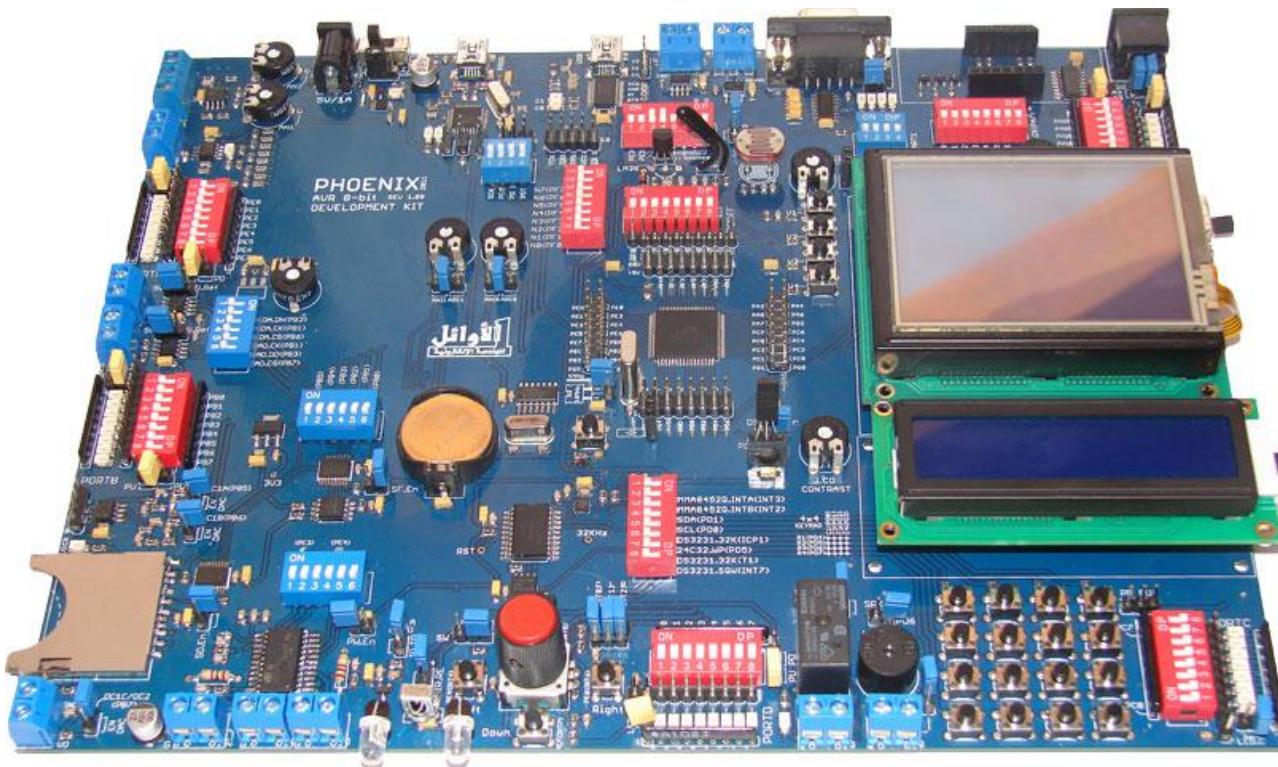
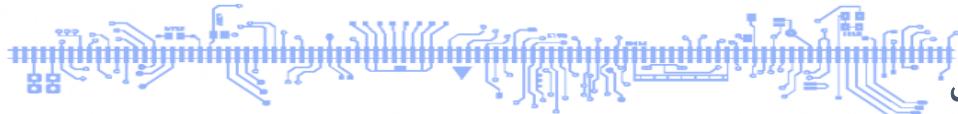
- (1) مدخل حساس إشارة صوتية لقياس شدة الصوت باستخدام ميكروفون إلكترونطي.
- (2) مخرج قيادة 8-bit عن طريق مسجل إزاحة مع دائرة قيادة ترانزستورية مدمجة باستخدام الشريحة STPIC6C595.
- (3) مبدل رقمي تشابهي (ADC) باستخدام شبكة لادر 8-bit.
- (4) ثنائيةات ضوئية (8-LEDs) متصلة مع البوابة PortE.
- (5) دارتي توليد جهد خطى تشابهي لدائرة المبدل ADC ودائرة المقارن التشابهي (AIN).
- (6) منفذ توسيعة دخل وخرج (8-I/O) للبوابة PortE مزود بمقاومات رفع و مقاومات سحب.
- (7) مبدل رقمي تشابهي بدقة 12-bit ذو خرج تفاضلي أو مفرد باستخدام الشريحة MCP4921.
- (8) مبدل تشابهي رقمي ذو دخل تفاضلي أو مفرد بدقة 12-bit باستخدام الشريحة MCP3201.
- (9) ثنائيةات ضوئية (8-LEDs) متصلة مع البوابة PortB.



- (10) ثنائى ضوئي ثلاثي الألوان (RGB).
- (11) دارة تنظيم جهد تغذية 3V3.
- (12) منفذ توسيعة دخل وخرج (8-I/O) للبوابة PortB مزود بمقاييس رفع ومقاييس سحب.
- (13) مبدل رقمي-تشابهي (DAC) بقناتين باستخدام خرج PWM بدقة 12-bit مع مرشح من الدرجة الثالثة ودارة مضخم عازل.
- (14) ذاكرة من نوع Flash بسعة 1Mb (W25Q80).
- (15) بطاقة متعددة الوسائط الرقمية (MMC/SD).
- (16) مبدل رقمي-تشابهي باستخدام خرج PWM مع دارة ترشيح RC.
- (17) دارة قيادة محرك خطوي / محركين تيار مستمر باستخدام الشريحة L298P.
- (18) دارة توليد تردد خارجي للمعالج.
- (19) وحدة استقبال (Receiver) أشعة تحت الحمراء (IR) وفق البروتوكول RC5.
- (20) ثنائيات إرسال (Transmitter) أشعة تحت الحمراء (IR) وفق البروتوكول RC5.
- (21) مشفر دوار (Rotary Encoder) مع مفتاح ضاغط.
- (22) أربع مفاتيح اتجاه (Top, Bottom, Left, Right).
- (23) ذاكرة EEPROM بسعة 32KB (AT24C32).
- (24) ثنائيات ضوئية (8-LEDs) متصلة مع البوابة PortD.
- (25) دارة توليد الزمن الحقيقي (RTC) DS3232.
- (26) منفذ توسيعة دخل وخرج (8-I/O) للبوابة PortD مزود بمقاييس رفع ومقاييس سحب.
- (27) حساس حرارة رقمي 1-wire (DS18B20).
- (28) مخرج تحكم نوع ريليه (Relay).
- (29) مخرج صوتي (Speaker/Buzzer).
- (30) لوحة مفاتيح ست عشرية (Hexadecimal 16-key Keypad).
- (31) ثنائيات ضوئية (8-LEDs) متصلة مع البوابة PortC.
- (32) منفذ توسيعة دخل وخرج (8-I/O) للبوابة PortC مزود بمقاييس رفع ومقاييس سحب.
- (33) شاشة إظهار كريستالية (LCD) بأبعاد 16x2 مع إضاءة خلفية زرقاء.
- (34) منفذ توسيعة دخل وخرج (52-I/O) مباشرة.

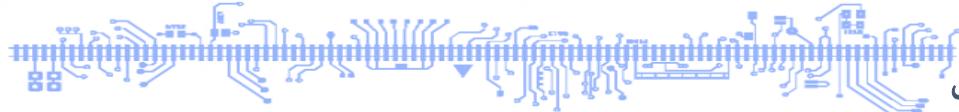


- (35) دارة قيادة شاشة لمس (Touch-screen) بـ 128x64.
- (36) شاشة إظهار رسومية (GLCD) بـ 128x64 مع إضاءة خلفية زرقاء.
- (37) المعالج الأساسي للوحة ATMega128A مع إمكانية التبديل بمعالجات أخرى متوافقة.
- (38) وصل عدة مفاتيح لحظية إلى قطب وحيد (ADC-Pin).
- (39) منفذ توسيعة دخل وخرج (8-I/O) للبوابة PortA مزود بمقاومات رفع و مقاومات سحب.
- (40) ثنائيات ضوئية (8-LEDs) متصلة مع البوابة PortA.
- (41) واجهة اتصال تسلسلي عبر منفذ PS2 – لوحة مفاتيح أو فأرة حاسوبية.
- (42) شاشة إظهار سباعية بأربع خانات (Quad 7-segment).
- (43) دارة قياس معامل المقاومة / السعة (RC).
- (44) دارة قياس شدة الضوء باستخدام مقاومة LDR.
- (45) واجهة اتصال تسلسلي عبر RS232 باستخدام الشريحة MAX232 مع مصافحة.
- (46) دارة قياس الحرارة باستخدام مقاومة ذات معامل حراري ثابت (NTC).
- (47) واجهة اتصال تسلسلي عبر RS485 باستخدام الشريحة MAX485.
- (48) واجهة اتصال تسلسلي عبر CAB باستخدام الشريحة MCP5251.
- (49) دارة قياس درجة الحرارة باستخدام حساس حرارة تشابهي LM35.
- (50) واجهة اتصال تسلسلي عبر USB باستخدام الشريحة FT232RL.
- (51) واجهة تتبع وفحص JTAG (Debugger).
- (52) وصلة برمجة مدمجة من أجل برمجة تطبيقات خارجية.
- (53) وحدة برمجة مدمجة على اللوحة AVR-MKII أو USB-ASP.
- (54) دارة تغذية من خلال منفذ USB مزودة بدارة حماية من ازدياد التيار أو القصر.
- (55) دارة تغذية خارجية مع حماية من انعكاس القطبية.
- (56) حساس تسارع رقمي ثلاثي المحاور (3-axis).



الشكل (5.2) لوحة التطوير Phoenix-AVR

يمثل المتحكم المصغر ATmega128A قلب النظام ويرتبط مع باقي المحيطيات الموزعة إما عن طريق بوابات الدخول والخرج، كارتباشه مع شاشات الإظهار المحرافية والرسومية، أو عن طريق نوافذ الاتصال التسلسلي (RS232, USB, RS485, CAN, PS2, I2C)، أو عن طريق أقطاب المبدلات التشابهية الرقمية (ADCs)، كارتباشه مع حساسات الحرارة والضوء وتوليد الجهد الخطي، أو عن طريق أقطاب المقاطعات الخارجية، كارتباشه مع المفاتيح اللحظية وحساس الدوران، أو عن طريق أقطاب توليد إشارة تعديل عرض النبضة (PWM) من أجل التحكم بالتطبيقات التي تحتاج إلى جهد خرج مستمر ذو قيمة وسطية متغيرة والتحكم بالمحركات ...



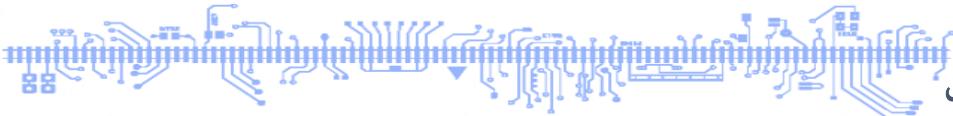
3. خطة العمل العامة للوحدة التطوير :

- 1 البحث المكتبي عن جميع الشركات المصنعة للوحات التطوير المتخصصة في برمجة المتحكمات .AVR
- 2 جمع الميزات والوظائف الأساسية لهذه اللوحات ومقاطعتها مع بعضها للحصول على ميزات جديدة متكاملة مستندة إلى تلك الميزات.
- 3 دراسة الحاجة إلى تجارب عملية تغطي معظم الاختصاصات في الهندسة الالكترونية بأقسامها (تحكم، اتصالات، قيادة، إلктرونون).
- 4 جمع هذه الدراسة الأكاديمية وتنفيذها على شكل لوحة تطوير عملية تراعي في تصميمها جميع الاعتبارات التصميمية الاختصاصية (EMC,EMR,ESD) في تصميم لوحات الأنظمة المضمنة بالإضافة إلى مرونة التعامل مع محاطياتها. Embedded Systems
- 5 بناء التجارب الأساسية ليتم برمجتها على النظام أو محاكماتها باستخدام برنامج Proteus

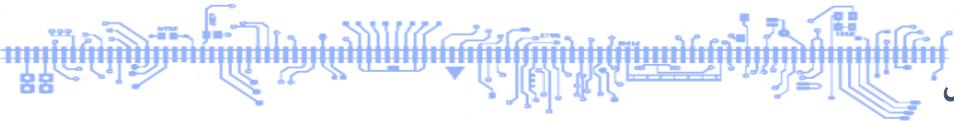
4. الاعتبارات التصميمية للوحدة :

تم تصميم هذا النظام وفقاً للاعتبارات التالية:

- (1) متوافقة مع جميع البيئات البرمجية compilers المستخدمة في برمجة المتحكمات المصغرة من العائلة AVR والمذكورة آنفاً.
- (2) يمكن برمجة المعالج دون الحاجة إلى فصله عن النظام وهو ما يسمى بـ System Programming.
- (3) تم وضع المحاطيات لتغطي التجارب العملية لمعظم الاختصاصات الهندسية.
- (4) تحوي على أكثر من 60 تجربة أساسية مستقلة ويمكن أن تصل إلى أكثر من 100 تجربة بالدمج.
- (5) تم بناؤها استناداً إلى خبرة عملية كبيرة تزيد عن ستة سنوات في مجال تصميم الأنظمة المضمنة (Embedded Systems Design) واستناداً إلى خبرة في مجال التدريس لمدة تزيد عن ثلاثة سنوات وهذا بدوره كان عاملًا أساسياً لاختيار التجارب التي تفيid المبرمج.



- (6) يستطيع الطالب التعامل مع اللوحة بشكل مباشر دون الحاجة إلى موجه لتشغيلها (self learners)، كل ما يحتاج هو اتباع الخطوات المنشورة في الدليل.
- (7) إن عامل الكلفة هو ما يميز لوحة التطوير هذه حيث أن الطالب يستطيعون بناء لوحتهم الخاصة بما لا يزيد عن USD120، بينما إذا أردنا الحصول على نفس المحيطيات والوظائف الموجودة في هذه اللوحة فإننا ربما نحتاج إلى أكثر من لوحتي تطوير بالإضافة إلى العديد من الموديلات الإضافية وبالتالي فإن الكلفة ربما تصل إلى USD700.
- (8) بالإضافة لكون البرامج (التجارب) يمكن كتابتها وإرسالها إلى النظام بشكل مباشر، فإنه تم بناء المخططات الأساسية للدارة في بيئة برنامج المحاكاة Lab-Center Proteus-7.2 الذي يعد البرنامج الأقوى في محاكاة دارات المعالجات والمتحكمات المصغرة وبالتالي يمكنك تشغيل البرامج التي يتم كتابتها في بيئة البرنامج Bascom-AVR دون الحاجة إلى لوحة التجارب وهذا غير موجود في لوحتات التطوير الأخرى!
- (9) تم تزويد لوحة التطوير بمنفذ مراقبة (Debugger) حيث يمكن وصل اللوحة مع نافذة الاتصالات التسلسليّة للحاسوب (RS232 Interface) ومراقبة عمل النظام وقراءة المتحولات.
- (10) بالإضافة إلى التجارب الأساسية للوحة التطوير فإنه تم وضع نقاط توسيعة لجامعة بوابات المتحكم المصغر على أطراف البوard (I/O 48) ليسهل عملية توسيعة البوارد أو ربط تجارب (موديولات) خارجية يتم بناؤها مستقبلاً.
- (11) إن هذه البوارد متواقة مع جميع متحكمات العائلة AVR بالإضافة إلى جميع المتحكمات التي تعتمد على البروتوكول SPI في برمجتها حيث تم مراعاة وضع المعالج على موديول منفصل من أجل تغييره.
- (12) تم اختبار لوحة التجارب هذه عملياً في متناول أيدي طلاب المرحلة الجامعية وقد أثبتت قدرتها على التجاوب المباشر ومرنة العمل عليها.
- (13) الآن هناك خطة لتسويق لوحة التطوير هذه على مستوى الشرق الأوسط لأغراض تدريسيّة.
- (14) أخيراً، ما يميز هذه البوارد هو تصميم النظام بأقل عدد ممكن من الطبقات في الدارة المطبوعة (يسهل على الطالب تنفيذ لوحتات لمخابرهم الإلكترونيّة المنزليّة) ووضع ملفات التصميم والمقرر التدريسي (التجارب العملية) في متناول أيدي الطلاب (Open Source) لإتاحة الفرصة لهم في بناء لوحة التطوير ووضعها في مخبر منزلي مصغر بحيث يستطيع أن يطور نفسه بشكل مستمر.
- (15) إمكانية التخاطب مع البوارد من خلال البيئات البرمجية... LabVIEW, VB6, Matlab, etc...

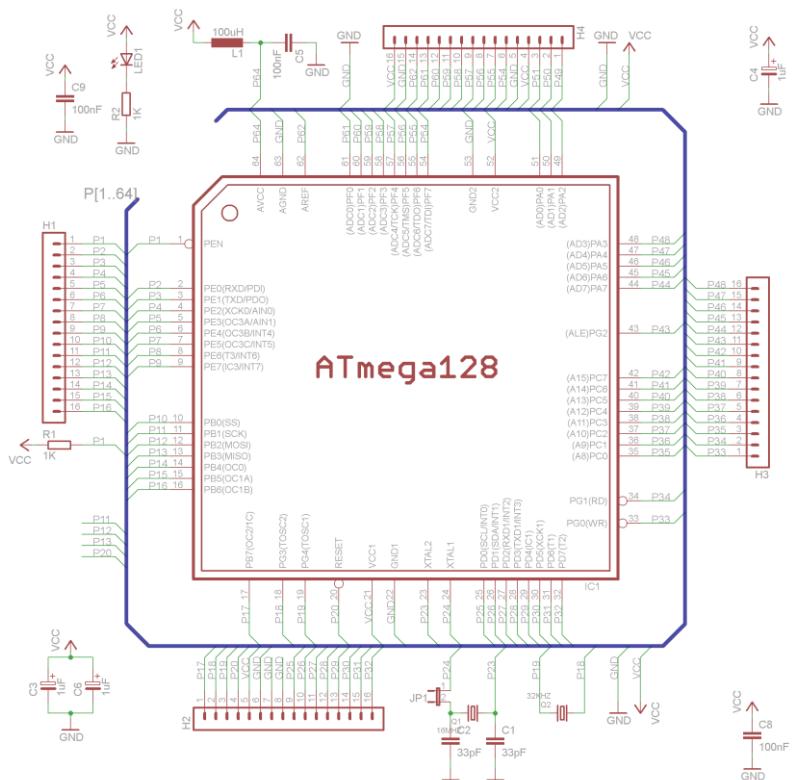


5. المخططات التصميمية المستخدمة من لوحة التطوير في هذا المشروع

(Development Board Schematic & Board Design)

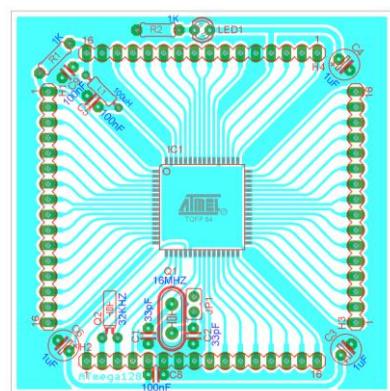
تحوي هذه اللوحة على العديد من المخططات و التوسعات الإضافية كما لاحظنا مسبقا .. و فيما يلي عرض بعض المخططات النظرية التصميمية المستخدمة في هذا البحث ..

الشكل التالي يبين المخطط التصميمي لموديول المعالج الخاص بلوحة التطوير:



الشكل (5.3)

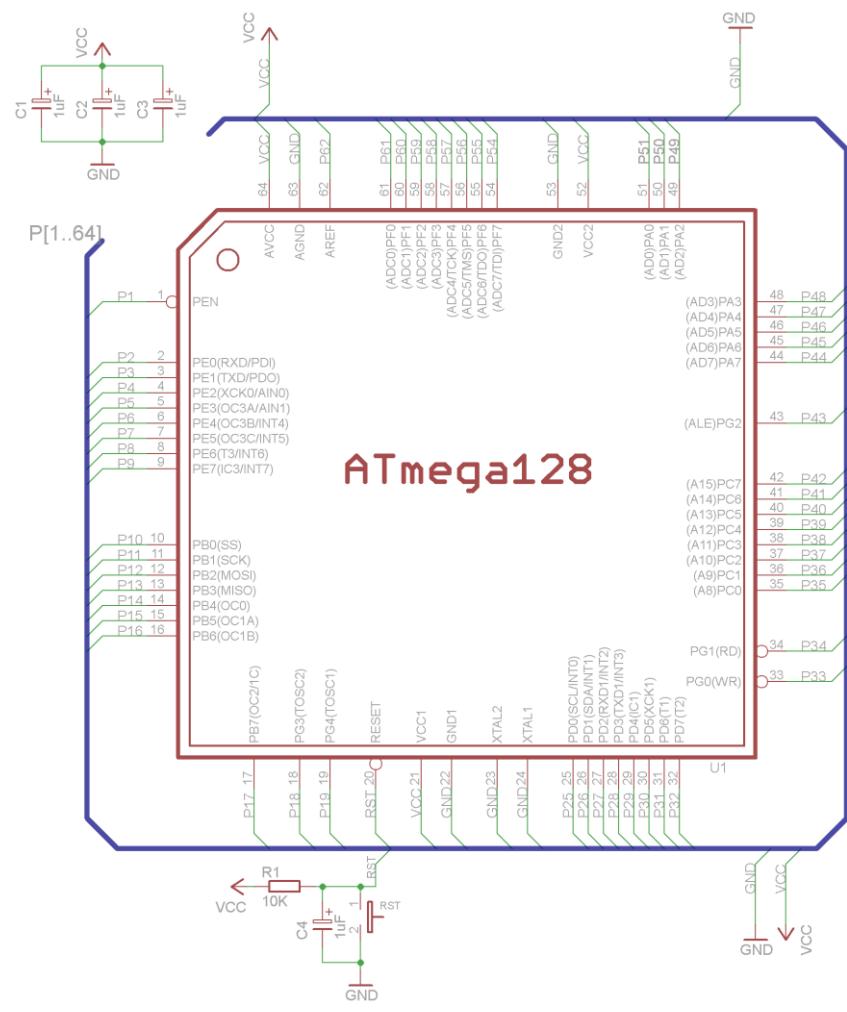
الشكل التالي يبين مخطط الدارة المطبوعة لموديول المعالج الخاص بلوحة التطوير:



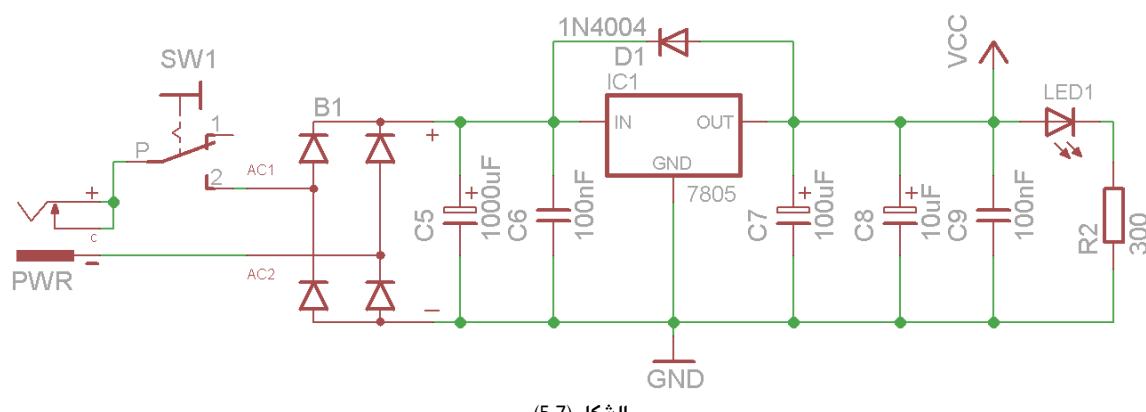
الشكل (5.4)

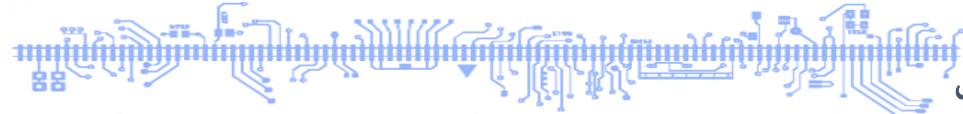


المخطط التصميمي لقاعدة موديل المعالج في لوحة التطوير:



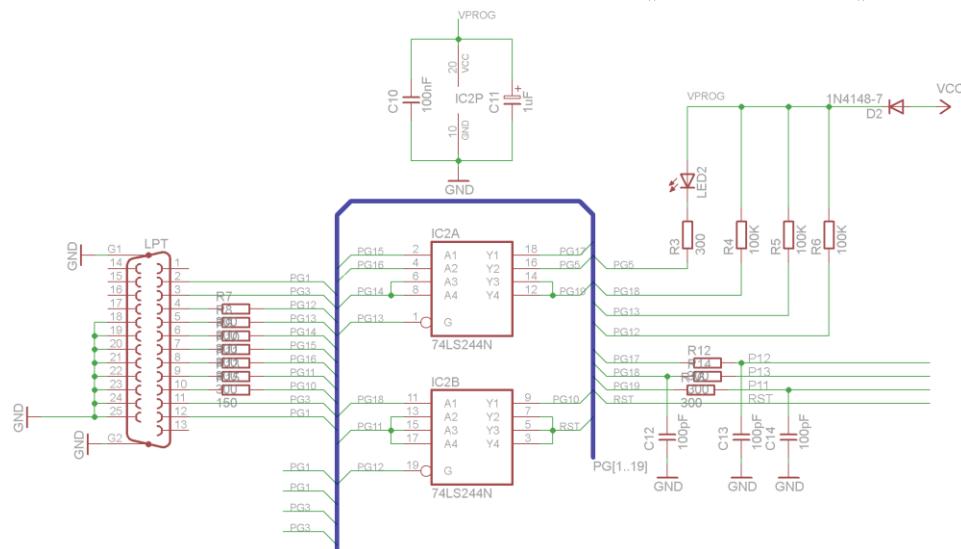
المخطط التصميمي لوحدة التغذية الرئيسية في لوحة التطوير:





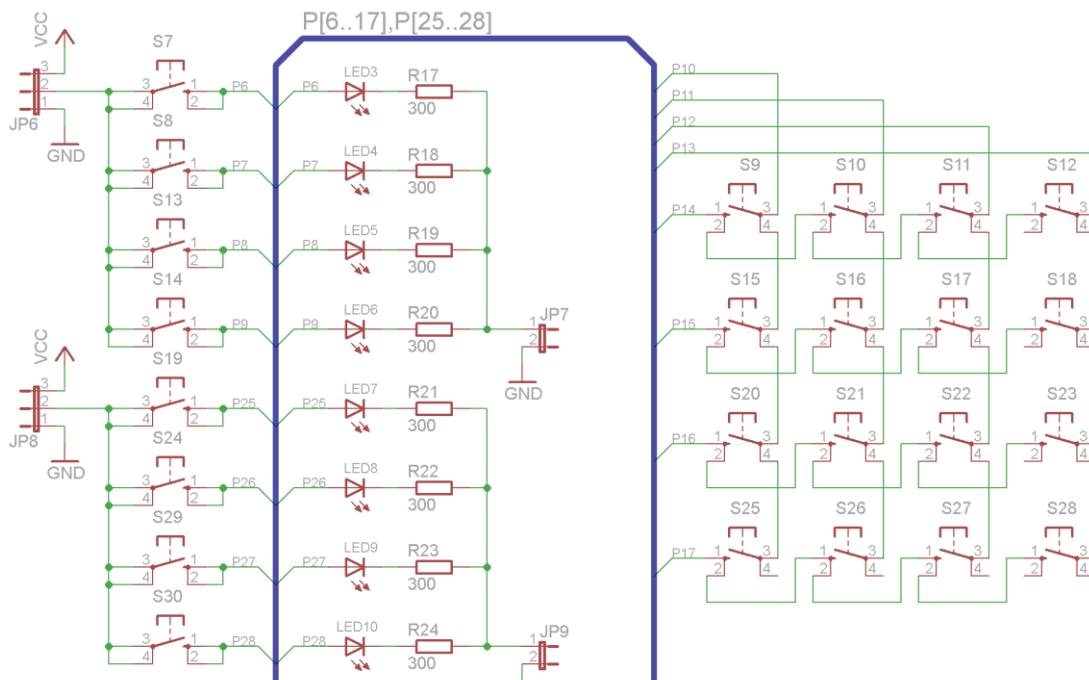
الفصل الخامس

المخطط التصميمي لدارة المبرمج في لوحة التطوير:



الشكل (5.8)

المخطط التصميمي للمفاتيح الاحادية والثنائية الضوئية في لوحة التطوير:

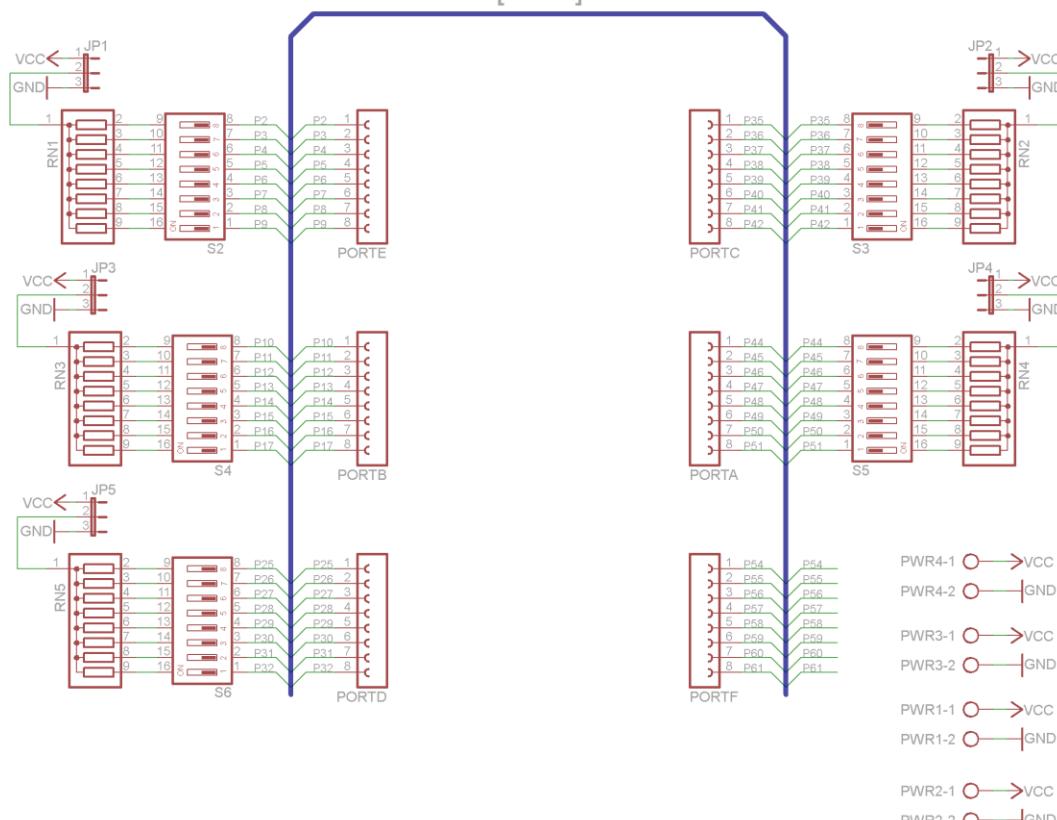


الشكل (5.9)



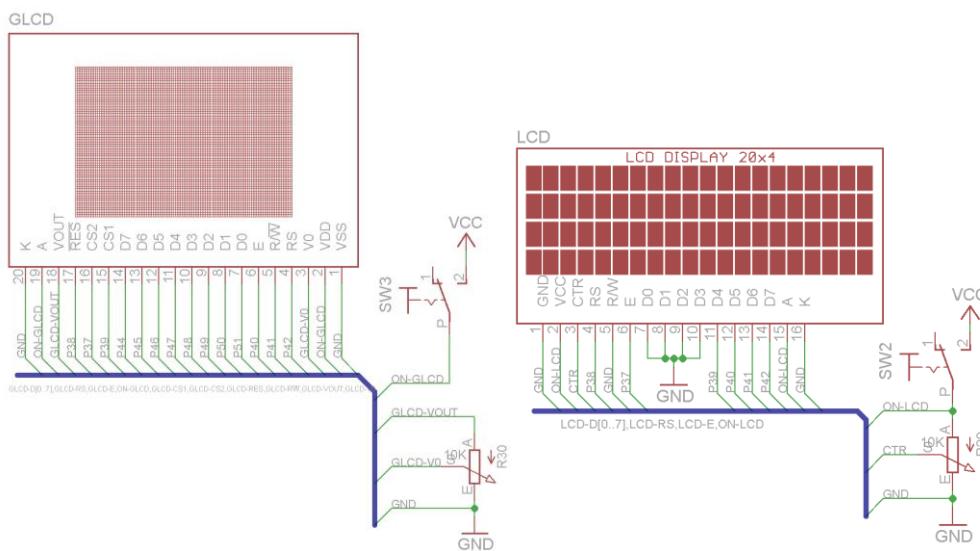
المخطط التفصيلي للتوسعات الرئيسية في لوحة التطوير:

P[1..61]



الشكل (5.10)

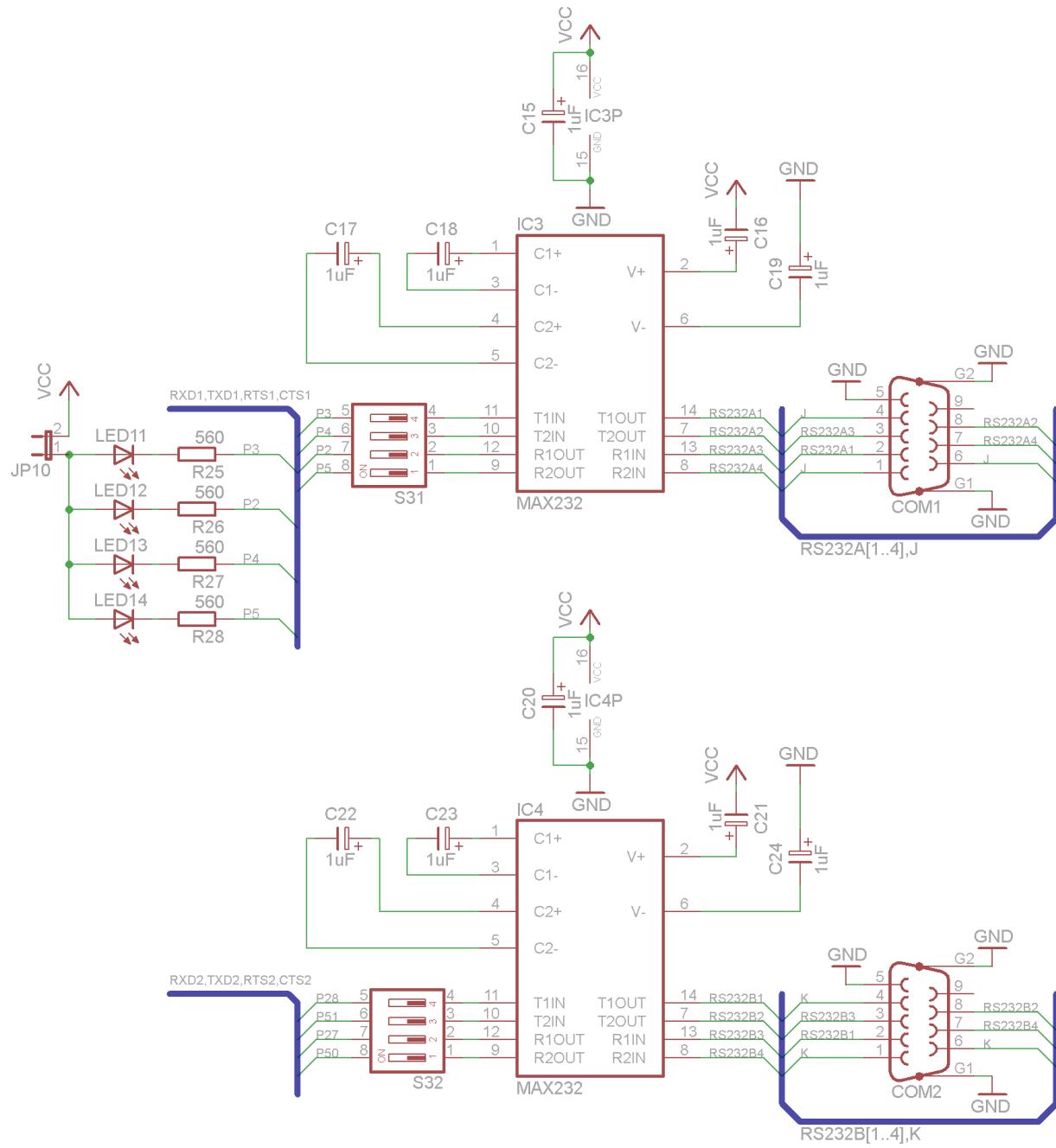
المخطط التفصيلي لشاشة الرسومية والحرفية في لوحة التطوير:



الشكل (5.11)



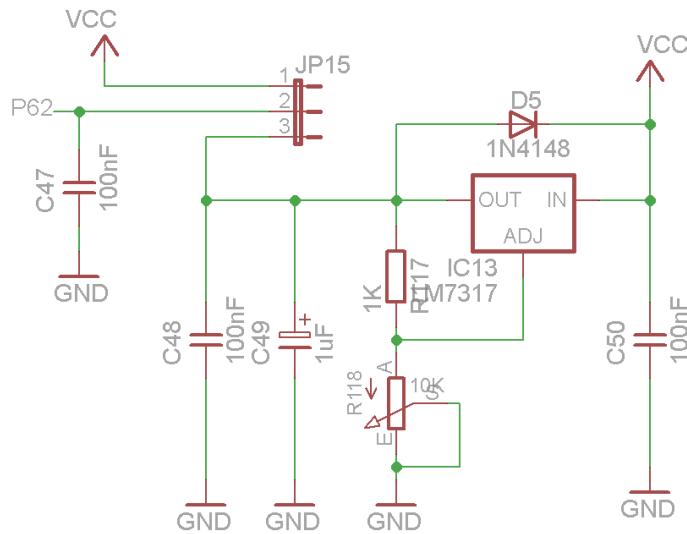
المخطط التفصيلي لนาفذتي الاتصال التسلسلي USART في لوحة التطوير



الشكل (5.12)

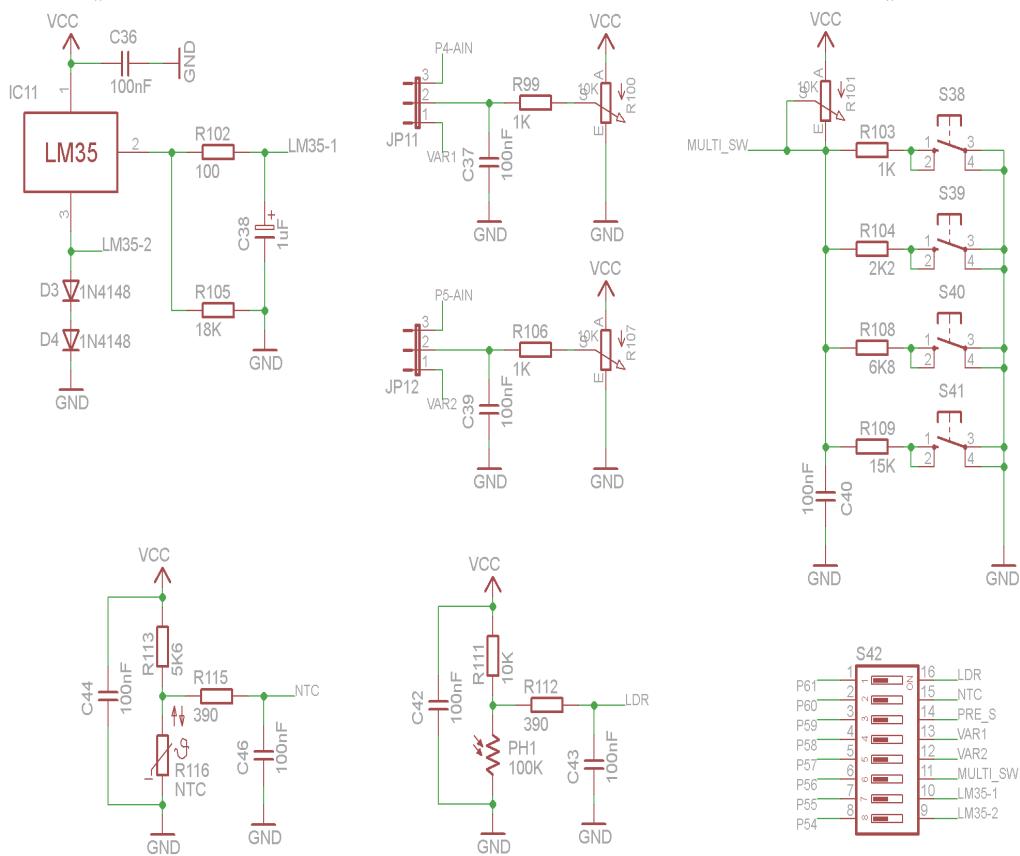


**المخطط التصميمي لدارة الجهد المرجعي القابل للتعديل من أجل المبدلات التشابهية الرقمية
في لوحة التطوير:**

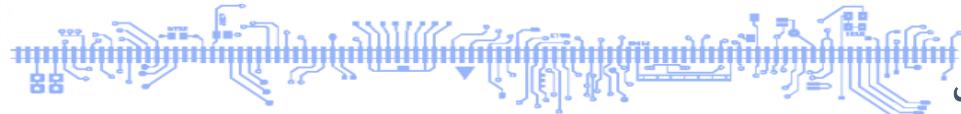


الشكل (5.13)

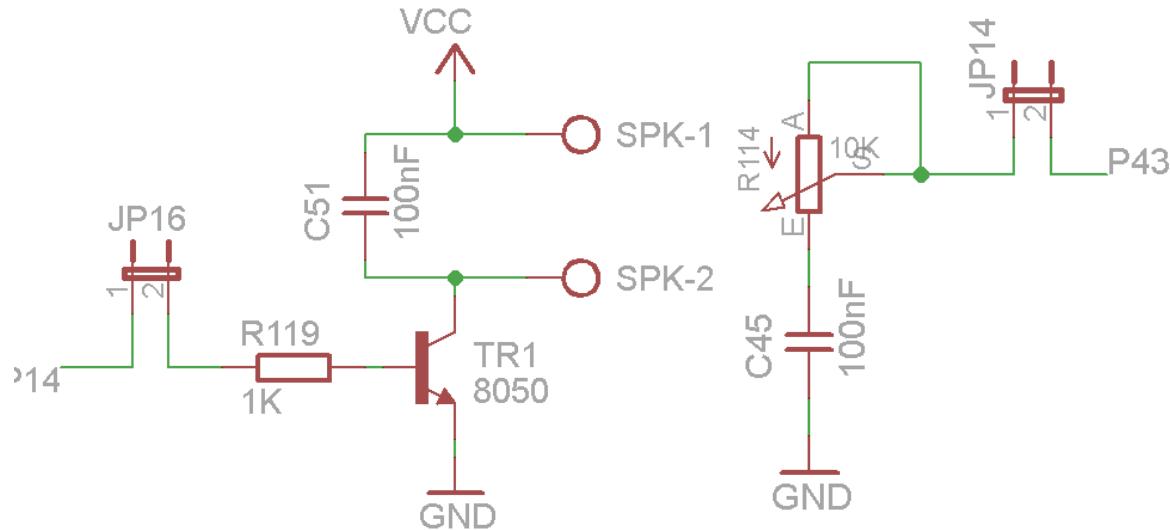
المخطط التصميمي لارات الحساسات الموصلة مع أقطاب المبدلات التشابهية الرقمية في اللوحة :



الشكل (5.14)



المخطط التفصيلي لدارة قياس السعات والمقاومات ودارة مخرج المجهار الصوتي في لوحة التطوير:

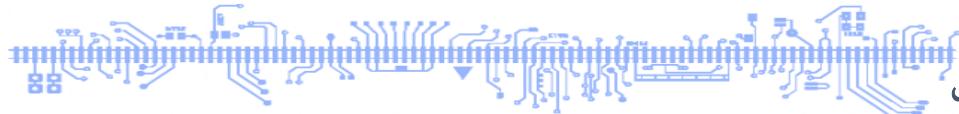


الشكل (5.15)

الجدول التالي يبين توزيع الوظائف على أقطاب المعالج:

	Px.0	Px.1	Px.2	Px.3	Px.4	Px.5	Px.6	Px.7
Port E	INT4~7		RS485 Interface		PWM>DAC			Four Buttons/Leds ¹
	UART1 AIN		UART1 with Hand-checking and LEDs Indicators					
	OC3A,B,C							
	T3				8-bit DAC Interface			
	ICP3				External Port Connector for further connecting and can be set to Pull Up/Down Resistor			
Port B	SPI		Programmer					
	OC1A,B		MMC/SD Card SPI Interface		Speaker	IR Sender	PWM	IR Receiver
	OC0,2				Hexadecimal Keypad			
	OC1C				External Port Connector for further connecting and can be set to Pull Up/Down Resistor			

¹ Buttons for Interrupt 4~7 can be set to VCC or GND by Jumper, connected with led indicators.

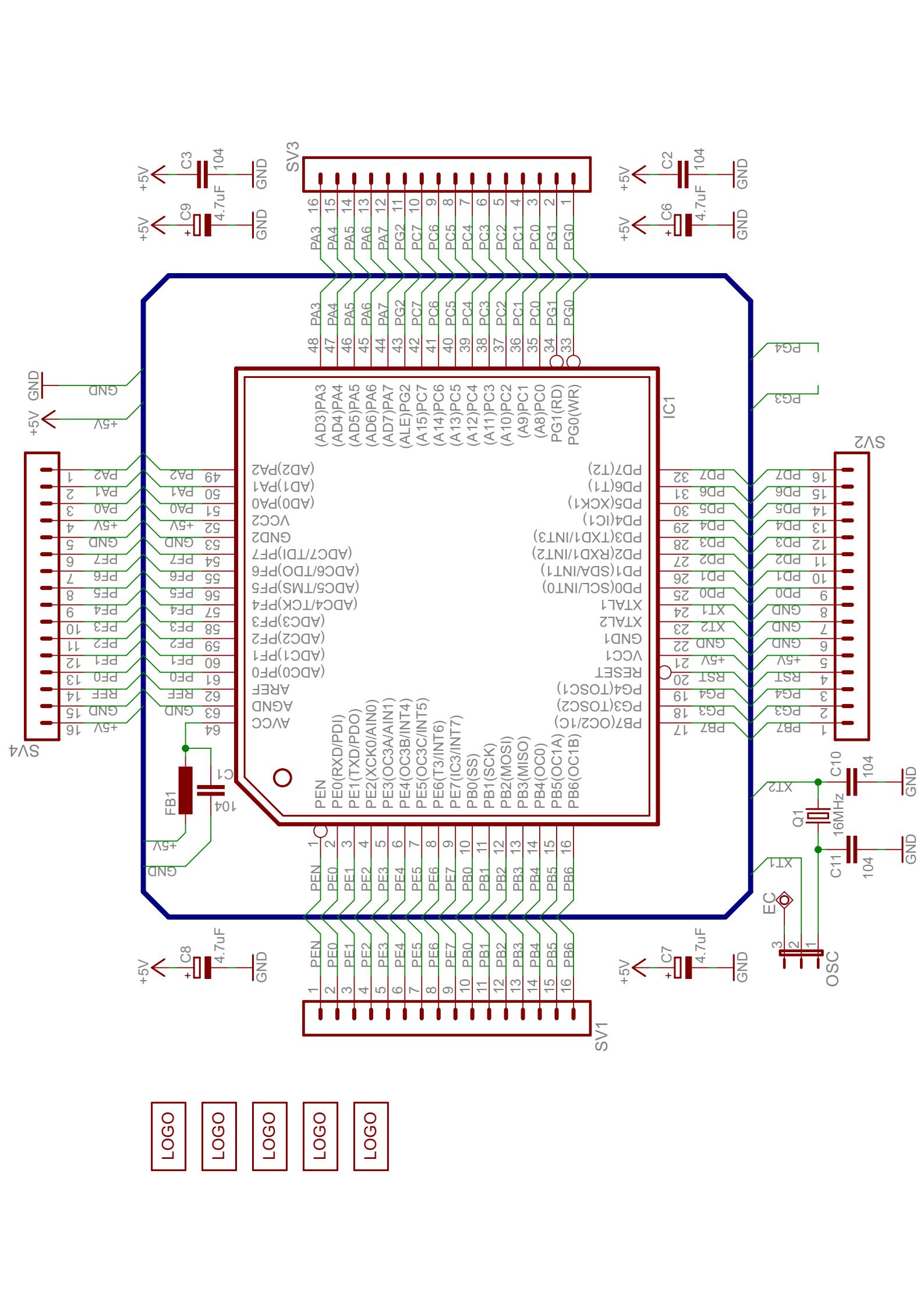


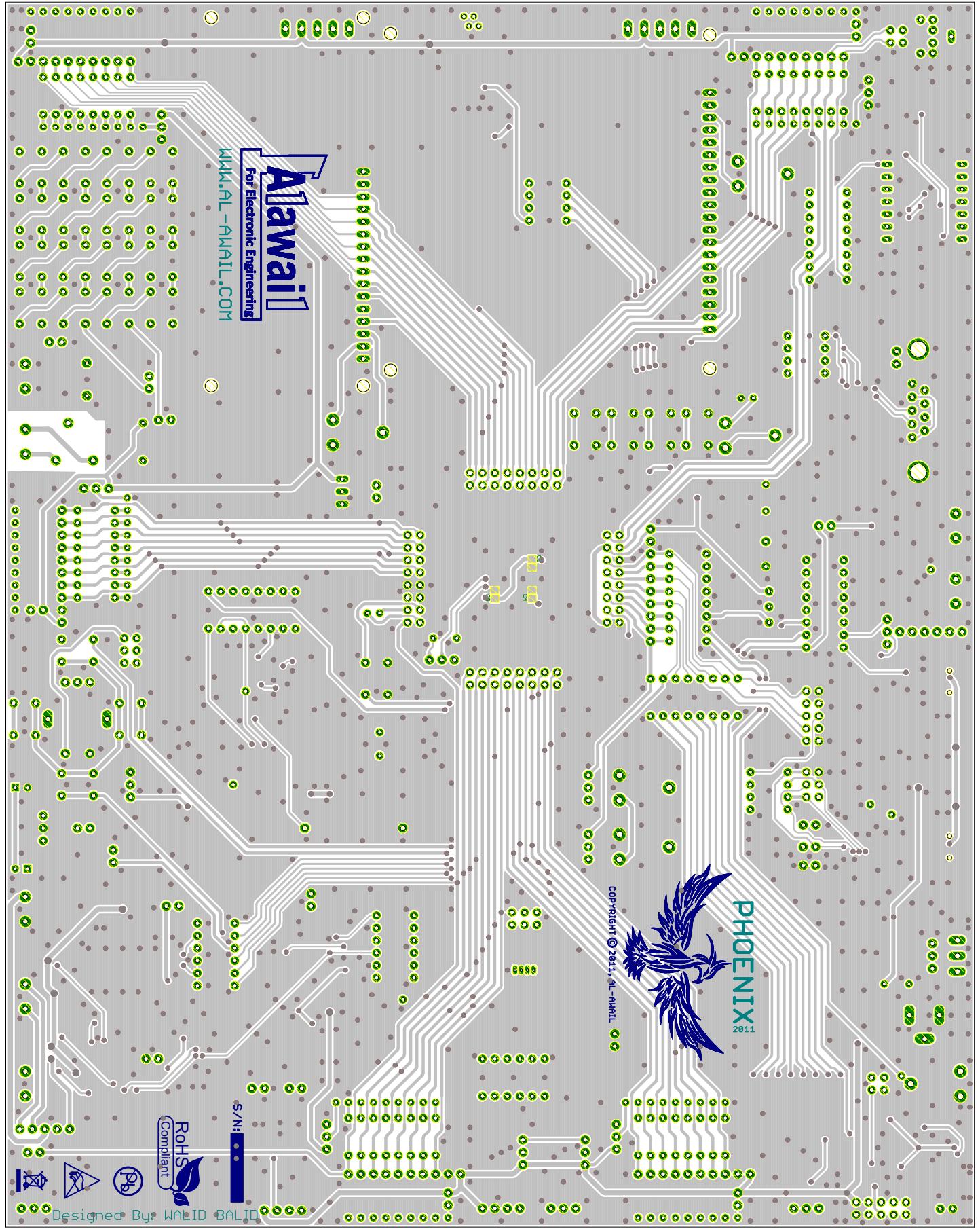
الفصل الخامس

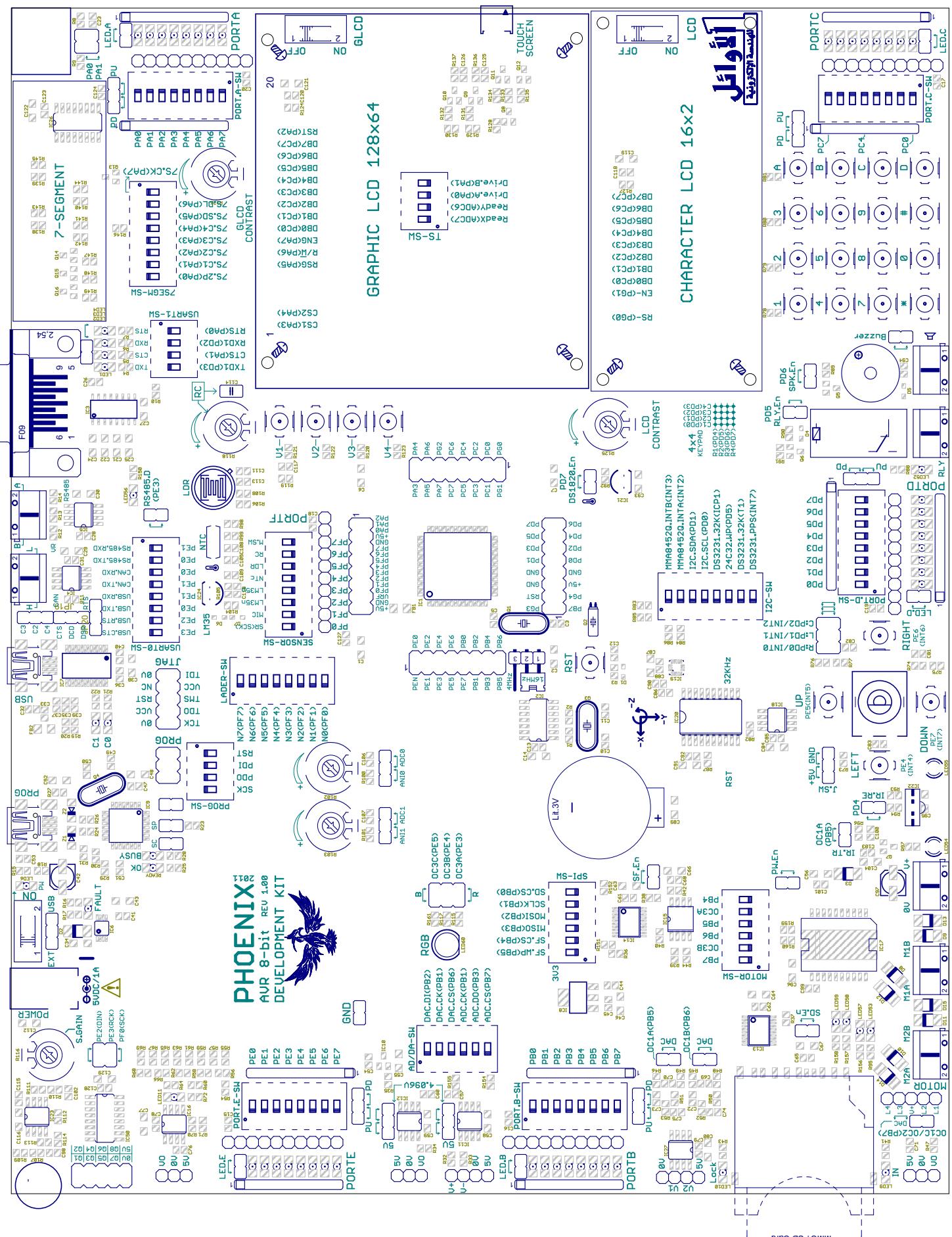
Port D	INT0~3	Four Buttons/Leds ²			ICP1	Relay1	T1	Relay2					
	UART2	PS2 SCK	UART2										
	TWI												
	T1~2	RTC & EEPROM											
	ICP1	External Port Connector for further connecting and can be set to Pull Up/Down Resistor											
Port C	Ex.MI-H	DS1820		GLCD Control Bus									
				LCD									
				Quad Seven Segment Control Lines									
		Dual Led-Matrix Display Data Bus											
		External Port Connector for further connecting and can be set to Pull Up/Down Resistor											
Port A	Ex.MI-L	GLCD Data Bus											
		UART2 Hand-checking		Basic Card									
		Quad Seven Segment Data Bus											
		External Port Connector for further connecting and can be set to Pull Up/Down Resistor											
Port F	ADC0~7 JTAG	LDR	NTC	Pressure	Variable	Variable	4 Switches	Temperature Sensor					
		Resistor	Resistor	sensor	Resistor	Resistor	On a line	JTAG Interface					
Port G	TOSC WR/RD	Dual Led-Matrix Display Control Lines		RC Circuit	23KHZ Crystal		X	X					
		ASK TR	ASK RE	PS2 Data	X	X	X	X					

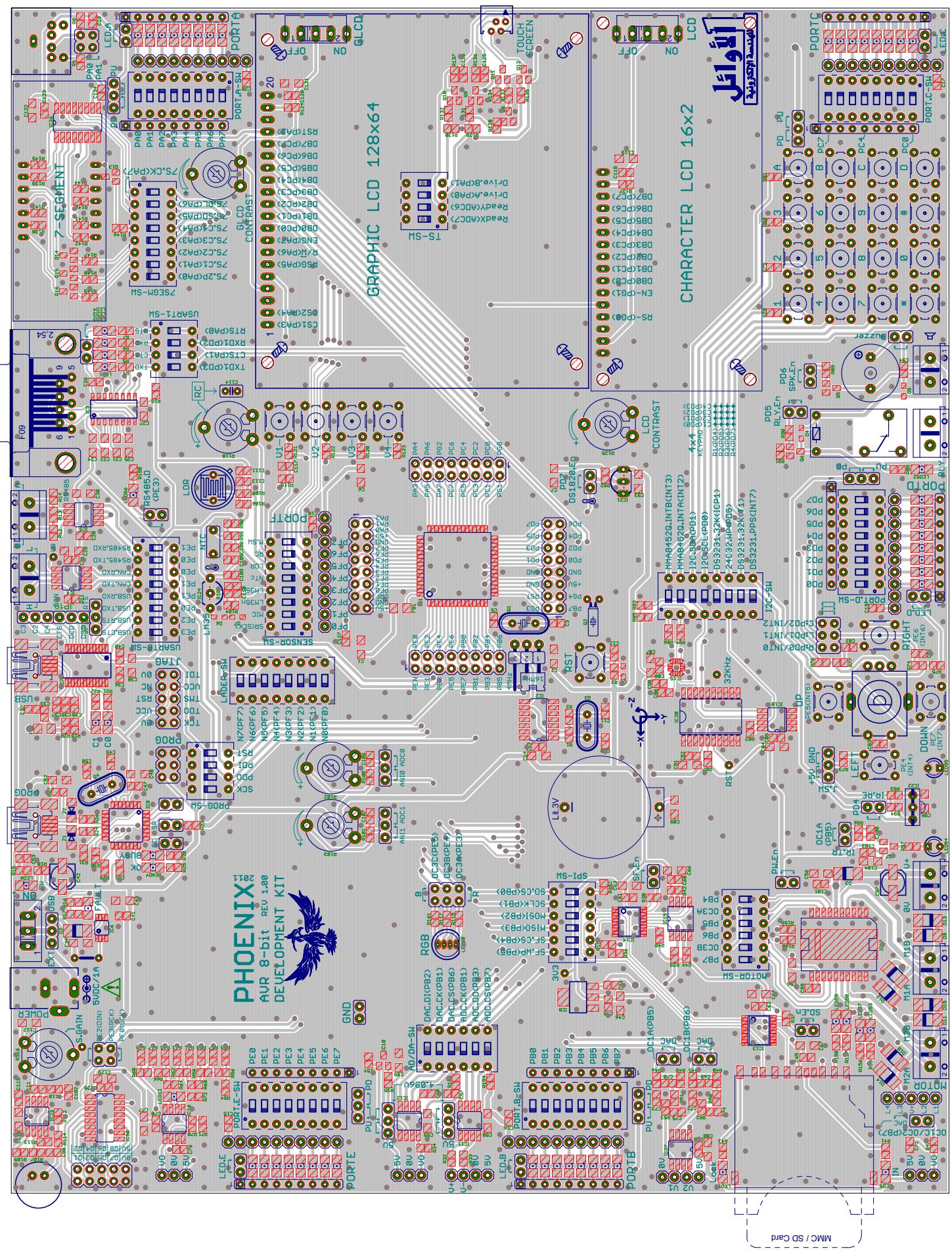


² Buttons for Interrupt 0~3 can be set to VCC or GND by Jumper, connected with led indicators.









الفهرس

الفصل	المحتوى	رقم الصفحة
الفصل الأول	المقدمة المخطط الصندوقي للنظام	1 3
الفصل الثاني	لغات برمجة الكيان الصلب الرسومية أهمية لغات البرمجة الرسومية البيئة البرمجية LabVIEW عناصر البيئة البرمجية LabVIEW 1.4. واجهة المستخدم الرسومية 2.4. واجهة البرمجة الرسومية LabVIEW والبرمجة الرسومية التدفقيه فوائد لغة البرمجة الرسومية 1.6. البرمجة الرسومية حدسية بدئيهية 2.6. أدوات التنجيح والفحص التفاعلية 3.6. التوزيع التقائى لمهام التنفيذ والأداء 4.6. اختصار المهامات والعمليات منخفضة المستوى 5.6. الجمع بين لغة G ولغات البرمجة الأخرى	6 7 9 10 14 15
الفصل الثالث	مقارنة بين لغات البرمجة الرسومية والنصية برمجة المنفذ التسلسلي في البيئة LabVIEW تصميم المخططات الصندوقية للحساسات البرمجة الصندوقية للحساسات 1.10. الحساس الصوتي 2.10. الحساس السعوي 3.10. حساس شدة الإضاءة LDR 4.10. حساس الحرارة LM-35	22 24 25 30
الفصل الثالث	مقدمة إلى متحكمات AVR عائلات متحكمات AVR مقارنة بين أشهر عائلات المتحكمات المصغرة الميزات الأساسية للمتحكم ATmega128A تغذية المتحكم المصغر مخطط البنية الداخلية للمتحكم وصف أقطاب المتحكم برمجة المعالج بيئة المحاكاة ISIS Proteus 10. التحويل التشابهي - الرقمي 1.10. المقسم الترددى للمبدل (Prescaler) 2.10. الجهد المرجعى (Reference)	34 34 36 37 39 42 43 47 50 51
	3.10. ميزات المبدلات التشابهية الرقمية في عائلة المتحكمات AVR	

	4.10. اعتبارات هامة لتخفيض ضجيج المبدلات التشابهية الرقمية	
55	5.10. مخطط البنية الداخلية للمبدل ADC	
	6.10. حساب الجهد على دخل المبدل	
	11. دراسة الحساسات التشابهية المستخدمة من اللوحة	
59	1.11. الحساس الصوتي	1. الكود البرمجي
62	1.11. الحساس السعوي	2. مقدمة عن بروتوكولات الاتصال
63	1.11. حساس شدة الإضاءة	3. بروتوكول الاتصال التسلسلي RS232
	1.11. حساس الحرارة	1.2. الميزات والمساوئ لبروتوكول الاتصال RS232
		2. عناوين بوابات الاتصال التسلسلي RS232 في الحاس
		3. تحقيق اتصال بين طرفيتين في RS232
		4. المواصفات الفنية للبروتوكول RS232
		5.2. مفهوم الاتصالات التسلسلي المتزامنة وغير المتزامنة
		6.2. بنية النافذة التسلسلي RS232 في الحاس
		7.2. دارات الملائمة TTL <→ RS232
75	4. بروتوكول الاتصال التسلسلي USB	الفصل الرابع
	1.3. مقارنة بين بروتوكولات الاتصال التسلسلي	
	2.3. ميزات منفذ الاتصالات التسلسلي USB	
	3.3. المواصفات الميكانيكية والكهربائية لمنفذ USB	
	4.3. طريقة عمل البروتوكول USB	
	5.3. حلول التطوير باستخدام منفذ الاتصالات التسلسلي USB	
	6.3. الشريحة FT232BM	
	7.3. توزع أقطاب الشريحة FT232BM	
	8.3. المخطط العام لبنية هذه الشريحة مع العناصر الأساسية	
	9.3. ربط منفذ COM (RS232) مع منفذ USB (Differential)	
84	1. الهدف الرئيسي من لوحة التطوير	الفصل الخامس
85	2. الوحدات المحيطية للوحدة التطوير	
89	3. خطة العمل العامة للوحدة التطوير	
89	4. الاعتبارات التصميمية للوحدة	
91	5. المخططات التصميمية المستخدمة من لوحة التطوير في هذا المشروع	

.. الملحقات ..

.. الفهرس ..