برمجة متحكمات AVR بلغة C

إعداد : اسماعيل الطرودي

TAB ELECTRONICS

برمجة متحكمات

AVR بلغة C

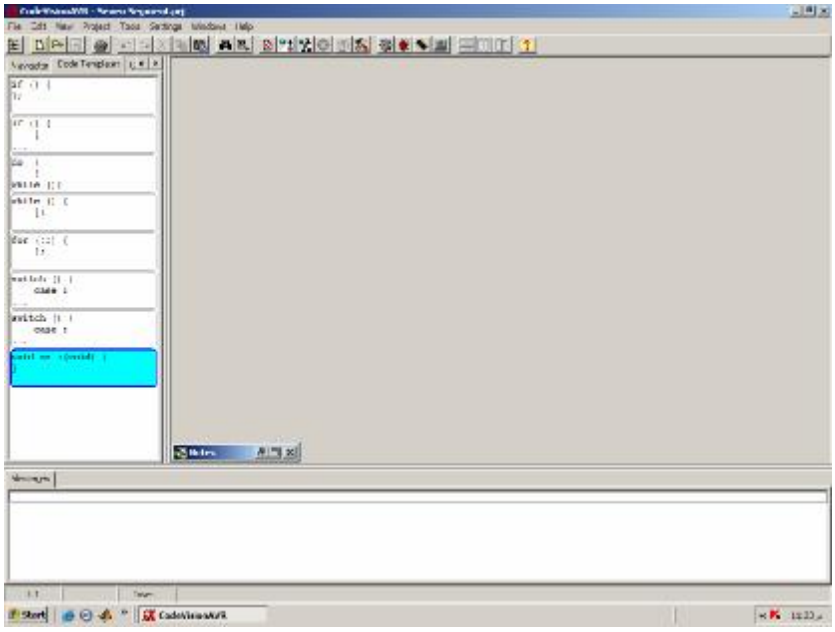
PROGRAMMING MICROCONTROLLER AVR WITH C

إعداد: اسماعيل الطرودي

الفارس لتقنيات الحاسوب والإتصالات
سوريا – حماه- هـ-229619 33 963+

بسم الله الرحمن الرحيم

المترجم الذي سوف نستخدمه من أجل بناء التطبيقات بلغة الـــC هو CodeVision
الشكل (١) يظهر لنا الواجهة الأساسية لهذا المترجم .



الشكل(١) الواجهة الأساسية لبرنامج CodeVisionAVR

وأول خطوة لكتابة البرنامج هو إنشاء مشروع جديد ، يوجد طريقتين لإنشاء المشروع.

الطريقة الأولى :

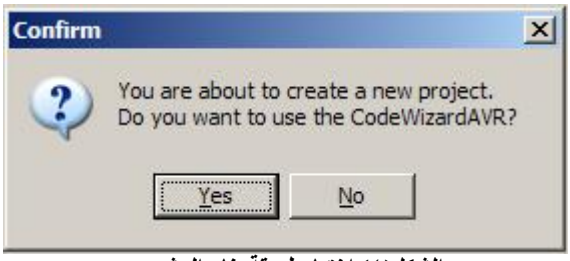
١: من File نختار New ويظهر لدينا مربع الحوار الشكل (٢) نختار Source ونحفظـــه بإســـم المـــشروع وليكن Led وسوف يقوم المترجم بحفظه تلقائياً بامتداد Led.c .

الشكل(٢) إختيار المصدر

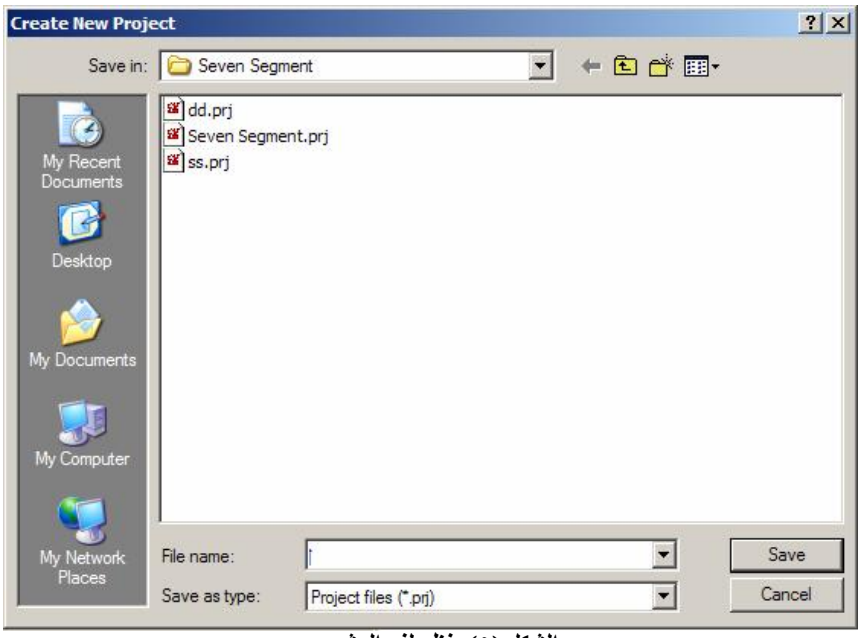
بعد ذلك نقوم بتكرار العملية السابقة" من File نختار New " ولكن نختار هنا Project الشكل(٣) ويظهـــر بعد ذلك مربع حوار يتـــسائل فيمـــا إذا كنـــا نريـــد اســـتخدام خاصـــية توليـــد الكـــود تلقائيـــاً باســـتخدام CodeWizardAVR نختار هنا No الشكل (٤) ، فيطلب منا حفظ المشروع ونحفظه بأي اسم ولكن يفضل أن يحتوي على دلالة للمشروع الذي يتم تطبيقه وليكن "Led" الشكل (٥)، بعد ذلك يطلب من ملف المـــصدر

للمشروع المراد تطبيقه الشكل(٦) وبالضغط على Add يظهر لنا مربع حوار نختار منه الملـف المـصدر الذي قد قمنا بإنشائه مسبقاً الشكل (٧) وهو "Led.c " **ونضغط** Open

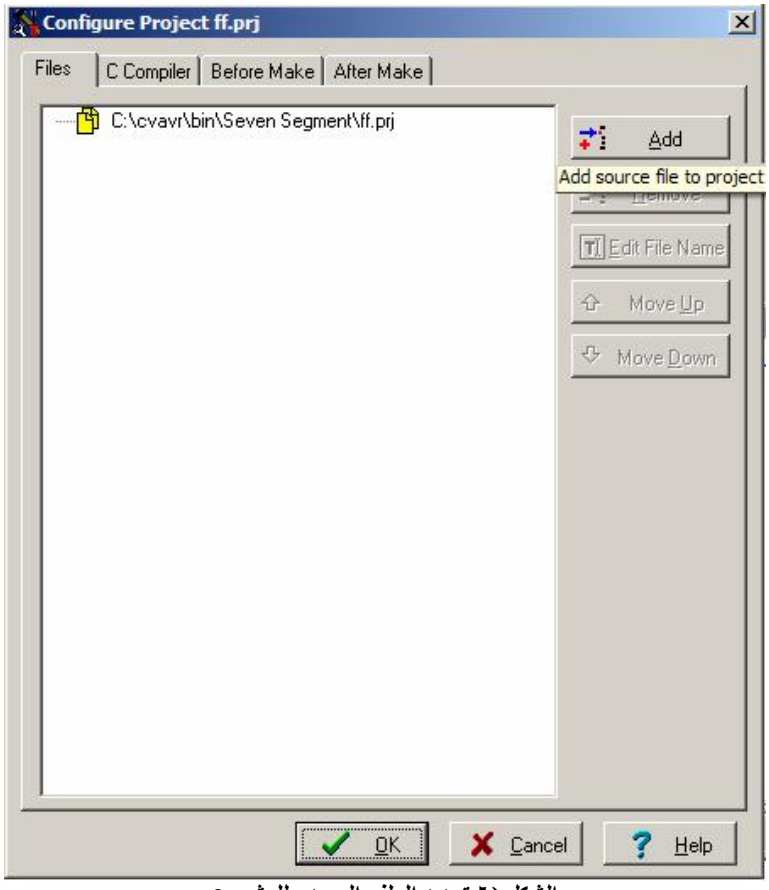
الشكل(٣) إختيارالمشروع



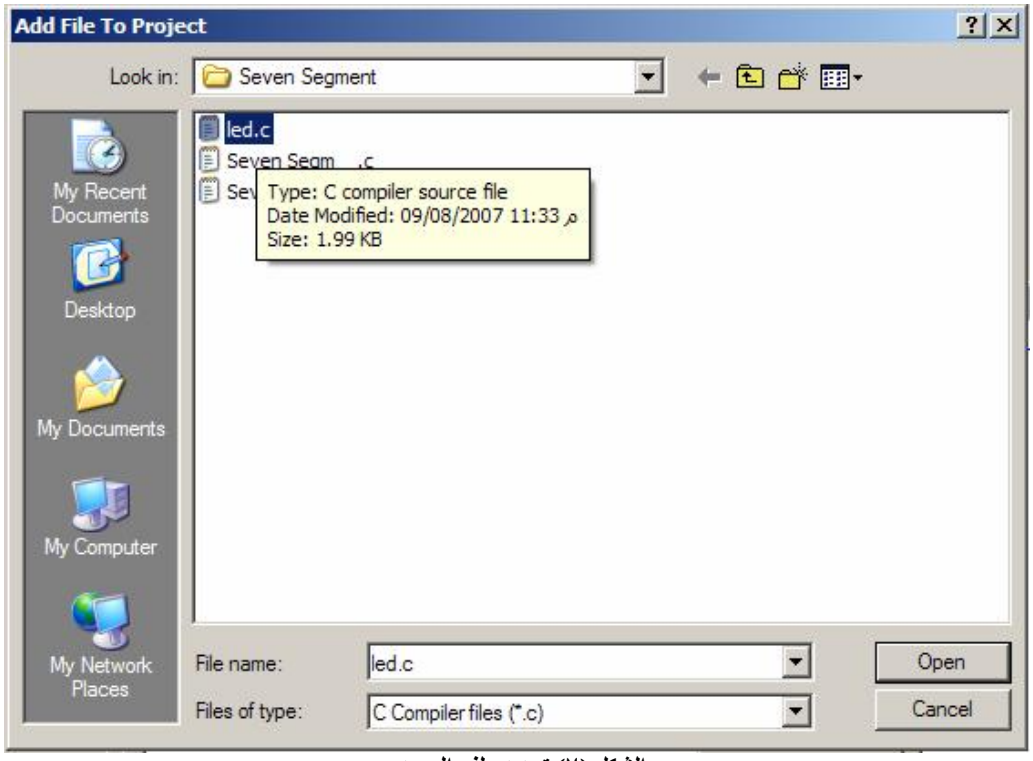
الشكل(٤) إختيارطريقة بناء المشروع



الشكل (٥)حفظ ملف المشروع



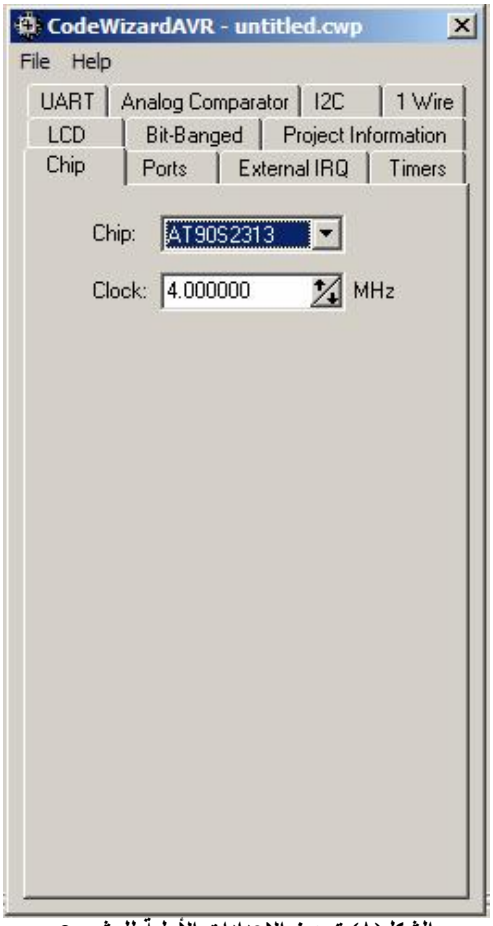
الشكل (٦ تحديد الملف المصدر للمشروع



الشكل (٧) تحديد ملف المصدر

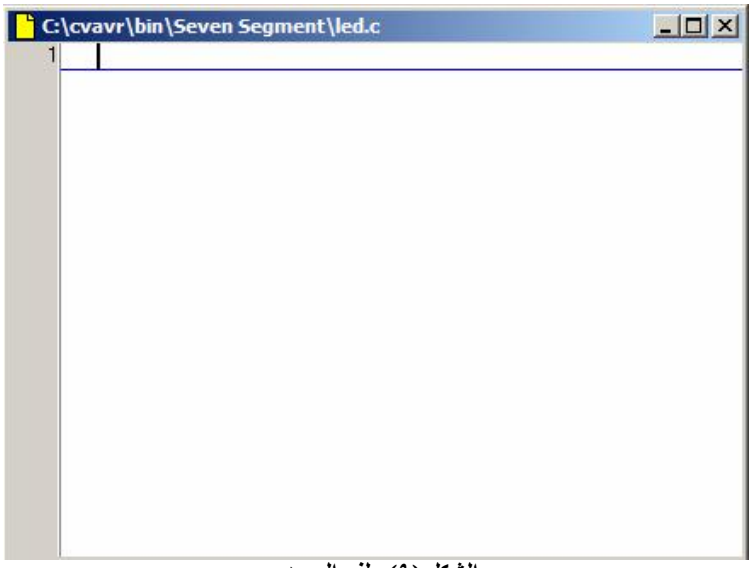
الطريقة الثانية :

من File نختار New ثم نختار Project فيظهر مربع الحوار السابق الشكل(٤) ونختار Yes فيظهر لنا مربع الحوار الشكل (٨) وضمنه نستطيع أن نقوم بتحديد كثير من الأمور مثل الحالة البدائية للمداخل الدخل/خرج ونستطيع تحديد نمط عمل المؤقتات أو تحديد البوابة التي سوف نقوم بوصلها مع شاشة الـ LCD وكثير من الأمور التي سوف نتطرق لها لاحقاً ، بعد أن نختار هذه الأمور نختار File "من مربع الحوار " ونختار بعد ذلك Generate Save and Exit فيطلب منا في البداية تحديد اسم ملف المصدر ثم اسم ملف المشروع وبعدها اسم ملف التوليد الأوتومانيكي لهذا المشروع ويفضل أن سكون جميع هذه الأسماء متشابهة .

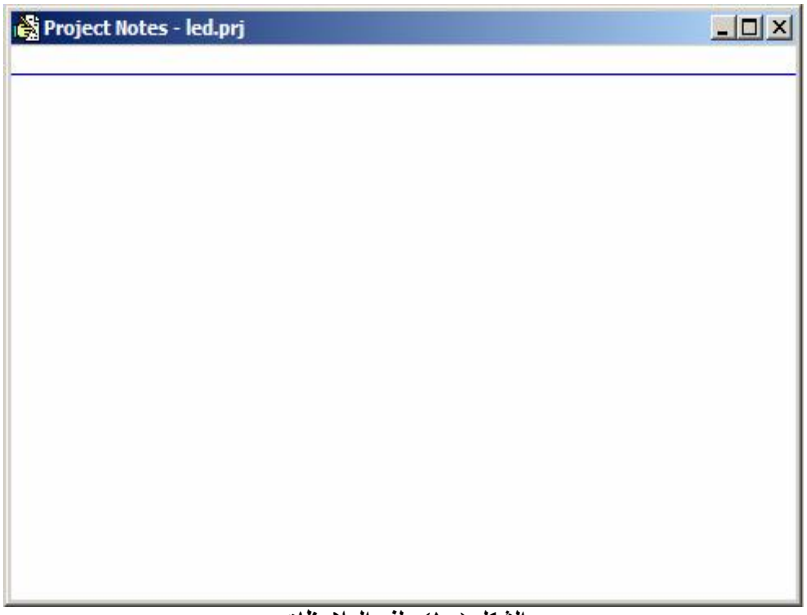


الشكل(٨) تجهيز الإعدادات الأولية للمشروع

ونلاحظ بعد ذلك في كلتا الطريقتين وجود ملفين الأول هو ملف المصدر واسمه كما تم اختياره Led.c الشكل(٩) والثاني ملف الملاحظات حول المشروع الذي نحن بصدد كتابة برنامجه الشكل (١٠) مثل عدد المداخل والمخارج وأشياء تتعلق بالمشروع وليس لهذ المعلومات أي دور في كتابة البرنامج.



الشكل (٩) ملف المصدر

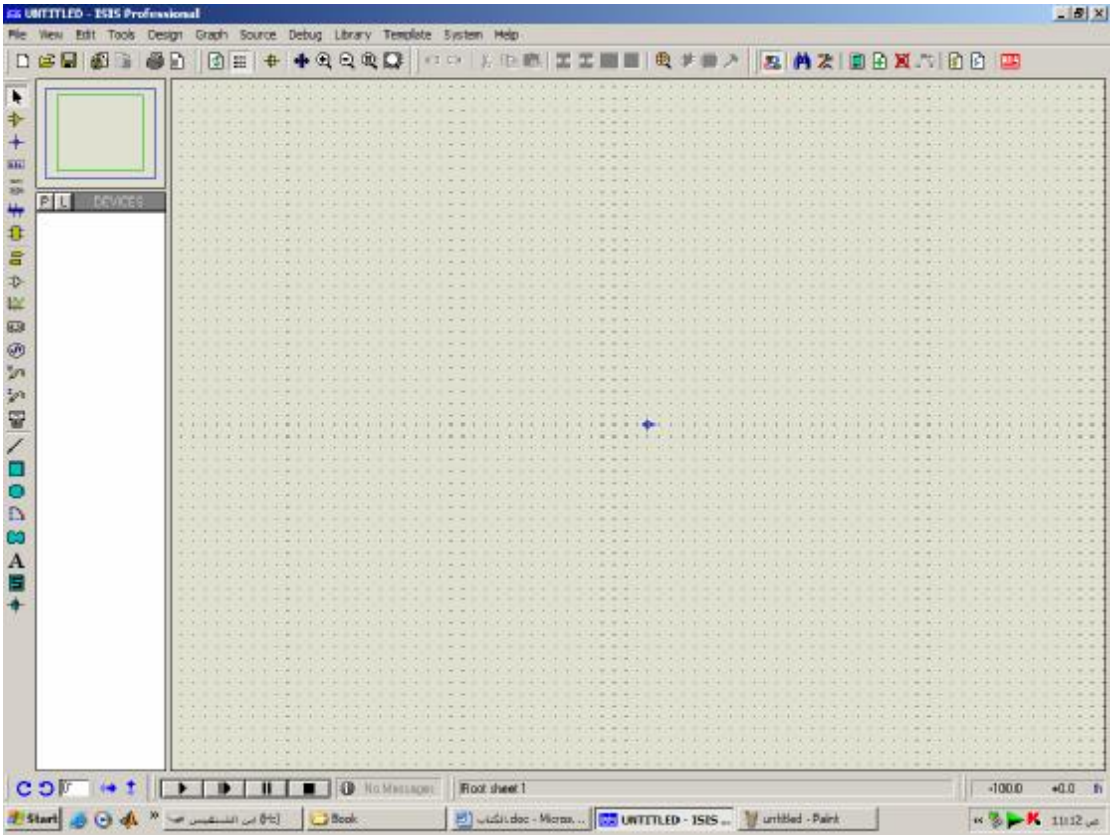


الشكل (١٠) ملف الملاحظات

برنامج المحاكاة
Proteus

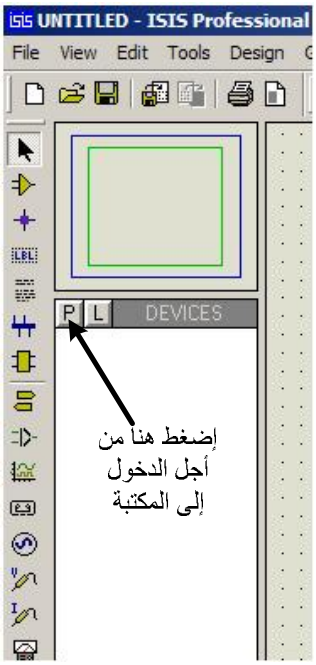
يوجد هناك العديد من برامج المحاكاة المستخدمة لمحاكاة الدارات الإلكترونية ولكن الأفضل والأروع في هذا المجال هو برنامج **Proteus** الذي يتصف بالدقة والسهولة في محاكاة المشاريع وخصوصاً أنه يحتوي على مكاتب كثيرة لكافة القطع الإلكترونية وهو يستطيع أن يحاكي الدارات التي تحتوي على المتحكمات Microcontroller بسهولة جداً مما يتيح للمصمم بأن يختبر عمل المتحكم قبل أن يتم حقن البرنامج في المتحكم مما سهل عملية التطوير في البرامج بسهولة وإضافة لذلك يستطيع المتدرب أو الذي يدرس برمجة المتحكمات أن يتم إختبار برامجه ويعدل عليها دون أن يتم تطبيقها والتي تكلف بالنسبة للطالب .

في البداية سوف نتعرف على واجهة البرنامج :



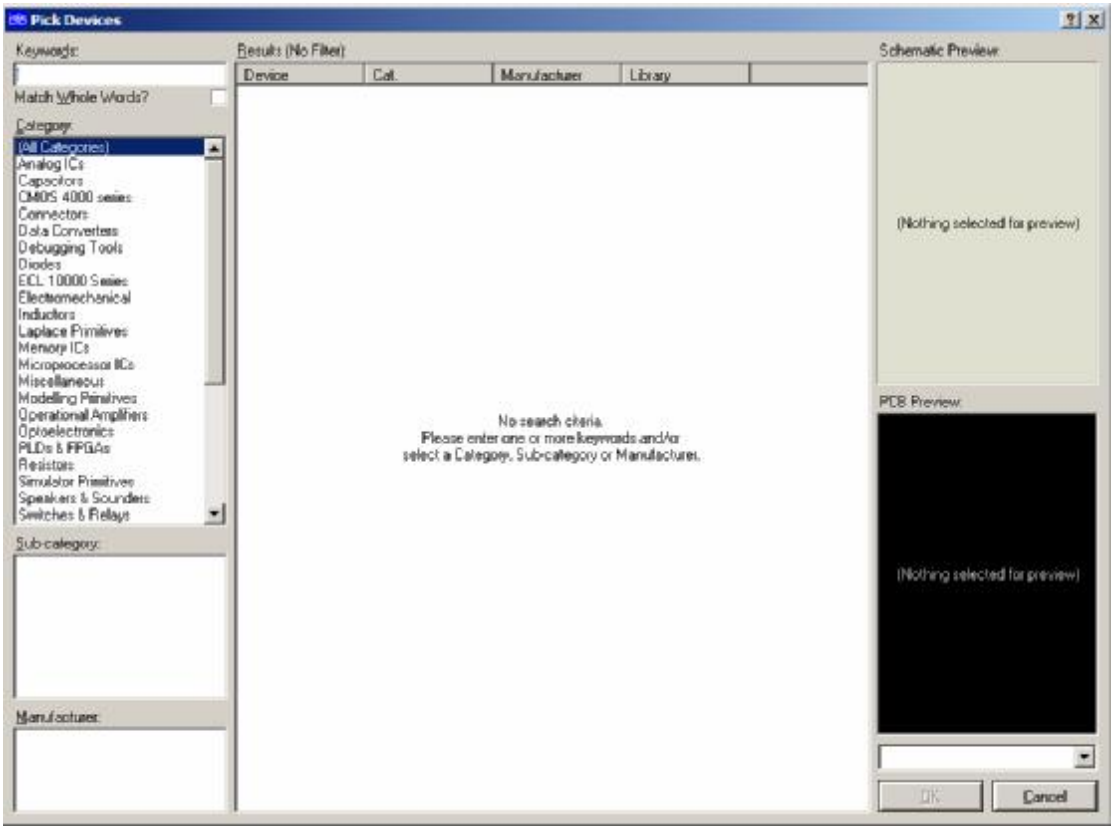
الواجهة الرئيسية للبرنامج

وسوف نتعلم على أهم المكتبات الموجودة في البداية من أجل جلب العناصر التي نريدها في المشروع ننقر بالمؤشر على الأيقونة الموضحة بالشكل التالي :



في حال عدم ظهور هذه الإشارة يجب أن يكون مضغوط على الأيقونة  الموجودة في أقصى اليسار .

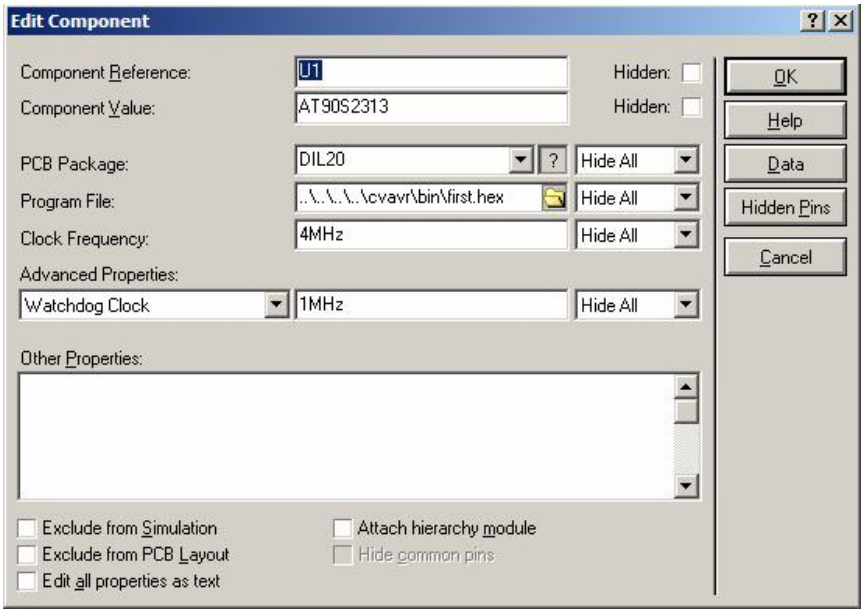
وبعدها تظهر لنا واجهة المكتبة



ونلاحظ وجود مربع للبحث عن العنصر الإلكتروني الذي نريده فنختار العناصر من هذه المكتبة ثـــم نغلقهـــا ونلاحظ وجود هذه العناصر في قائمة الأجهزة .

حقن البرنامج في المتحكم :

من أجل حقن البرنامج في المتحكم نضغط مرتين على المتحكم فتظهر لنا مربع الحوارالتالي :



وبالضغط على المجلد يظهر لنا مربع يطلب منا مسار ملف المشروع وهو بامتداد hex. والذي يتولـــد أثنـــاء عملية الترجمة Compile للمشروع ويكون بنفس المسار الذي حفظنا فيه هذا المشروع .

ملاحظة : يفضل حفظ المشروع على السواقة C من أجل سهولة عملية المحاكاة .

المتحكم AT90S2313

المزايا Features

يتمتع المتحكم (MCU) AT90S2313 بالمزايا و المواصفات التالية :

- استخدمت عائلة AVR مزايا بنية RISC المحسنة .
- تتمتع عائلة AVR بالأداء العالي وبطاقة بنية RISC المنخفضة .
- تحتوي قائمة التعليمات على 120 تعليمة ، ينفذ معظمها خلال دورة آلة واحدة .
- ذاكرة برنامج وميضية مبنية داخل الشريحة حجمها 2Kbyte قابلة لإعادة البرمجة .

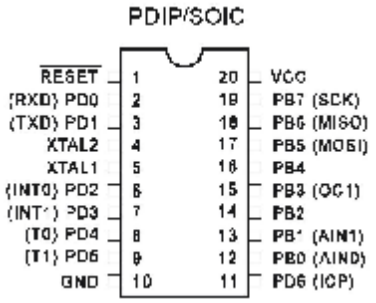
- نافذة تسلسلية SPI لتحميل البرنامج .
- الديمومة : 1000 دورة كتابة/مسح .

- ذاكرة معطيات EEPROM بطول 128bytes .
- الديمومة :100، 000 دورة كتابة / مسح .

- ذاكرة معطيات داخلية RAM بطول 128byte .
- اثنان و ثلاثون مسجل عمل للإغراض العامة .
- خمسة عشرة قطب I/O قابلة للبرمجة .
- جهد التغذية : Vcc = 2.7 – 6.0V .
- مجال عمل الهزاز : Fck = 0 – 20MHz .
- زمن دورة التعليمة : 50ns عند 20MHz .
- مؤقت/عداد بطول 8-bit بمقسم prescaler منفصل .
- مؤقت/عداد بطول 16-bit بمقسم prescaler منفصل .
- بأنماط المقارنة و المسك capture .
- نافذة تسلسلية ثنائية الاتجاه UART .
- خرج PWM قابل للبرمجة بثمان أو تسع أو عشرة خانات .
- مصادر المقاطعة داخلية و خارجية.
- مؤقت مراقبة watchdog قابل للبرمجة دخله من الهزاز RC الداخلي .
- مقارن تشابهي مبني داخل الشريحة.
- أنماط لتوفير الطاقة : نمط البطالة ، و نمط الطاقة التحتية .
- أقفال برمجية لحماية البرنامج .
- شريحة ذات عشرين قطباً.

أقطاب التحكم Pin Configuration

يبين الشكل التالي أقطاب المتحكم AT90S2313



أقطاب التحكم AT90S2313

شرح أقطاب المتحكم Pin Description

Vcc:

قطب جهد التغذية الموجب Vcc = 2.7 – 6.0V

GND:

قطب جهد التغذية الصفري GND = 0V

النافذة Port B (PB7 ..... PB0) :

وهي عبارة عن نافذة I/O ذات ثمانية أقطاب ثنائية الاتجاه مزودة بترانزيستور رفع داخلي ، وحيث أننا نستطيع استخدام القطبين PB0 و PB1 أيضاً كمدخل موجب (AIN0) و مدخل سالب (AIN1) على الترتيب للمقارن التشابهي المبني على شريحة المتحكم MCU .

تتمتع النافذة B بمزايا أخرى خاصة بعمل المتحكم MCU وهي مشروحة في فقرة " نوافذ الدخل/الخرج " .

النافذة Port D (PD6 ..... PD0) :

وهي عبارة عن نافذة I/O ذات سبعة أقطاب ثنائية الاتجاه مزودة بترانزيستور رفع داخلي ، وتتمتع النافذة D بمزايا أخرى خاصة بعمل المتحكم AT90S2313 وهي مشروحة في فقرة "نوافذ الدخل/الخرج".

قطب التصفير RESET :

مدخل التصفير . عند تطبيق إشارة كهربائية ذات منطق منخفض على القطب RESET لمدة دورتي آلة أثناء عمل هزاز الشريحة ، فإن دارة التصفير الداخلية تعمل على تصفير المتحكم MCU .

مدخل الهزاز XTAL1 :

هو عبارة عن مدخل لمضخم الهزاز العاكس وهو أيضاً مدخل لدارة تشغيل الساعة الداخلية .

مدخل الهزاز XTAL2 :

هو عبارة عن مخرج لمضخم الهزاز العاكس .

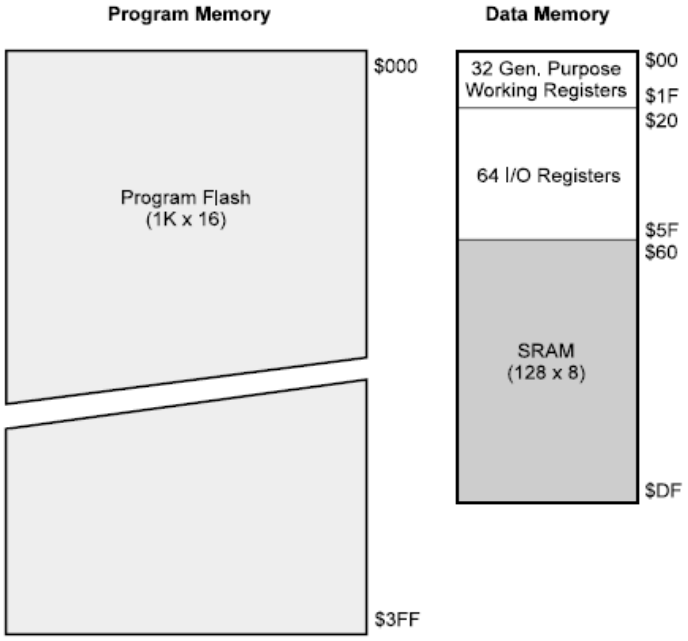
لمحة بنيوية عن المتحكم AT90S2313 :

لقد اعتمدت متحكمات AVR مفهوم بنية هارفارد Harvard التي تعنون ذاكرة المعطيات و ذاكرة البرنامج بخطوط منفصلة (على خلاف البنية التي وضعها العالم فون نويمان في مذكرته) . حيث يتم الولوج لذاكرة البرنامج بدورة ممر واحدة . فعندما تبدأ وحدة المعالجة المركزية CPU بتنفيذ التعليمة الأولى ، فإنه يتم إحضار شيفرة التعليمة التالية من ذاكرة البرنامج ، و بالتالي أدت هذه البنية إلى تنفيذ التعليمة بدورة ساعة واحدة . ونذكر هنا أن ذاكرة البرنامج هي ذاكرة وميضية Flash قابلة لإعادة البرمجة مبنية على شريحة المتحكم MCU .

تدعم وحدة الحساب و المنطق ALU العمليات الحسابية و المنطقية بين المسجلات أو بين عدد فوري ثابت و مسجل . كما تقوم وحدة الحساب و المنطق بتنفيذ بعض العمليات على مسجل وحيد .

ويقصد بالولوج السريع لملف المسجلات الذي يحتوي على اثنين و ثلاثين مسجل عمل للأغراض العامة ، على أنه الولوج الذي يستغرق زمن قدره دورة ساعة واحدة . هذا يعني أنه خلال دورة ساعة واحدة تقوم وحدة الحساب و المنطق ALU (Arithmetic Logic Unit) بتنفيذ عملية واحدة ، فهي تقوم أولاً بإخراج المعاملين من ملف السجلات ، ومن ثم تنفذ العملية ، ومن ثم تعيد تخزين النتيجة في ملف المسجلات. وكل ذلك يتم خلال دورة ساعة واحدة.

يبين الشكل (٢) خريطة ذاكرة البرنامج و ذاكرة المعطيات :

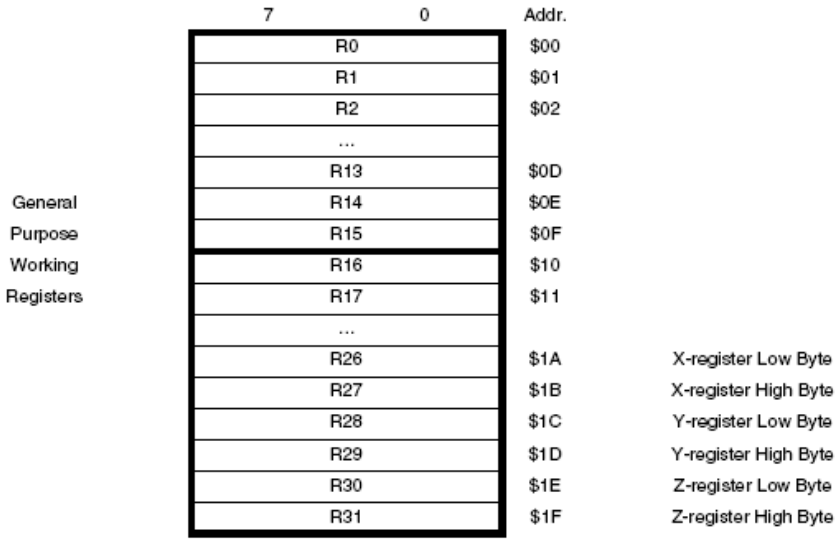


بالإضافة لفعالية المسجل ، فإننا نستطيع استخدام أنماط عنونة الذاكرة التقليدية لعنونة ملف المسجلات أيضاً . و هذا متاح على اعتبار أن ملف المسجلات معين باثنين و ثلاثين مسجلاً متوضعة عند حيز عنونة المعطيات السفلي ($00 - $1F) مما يسمح لنا بالولوج إلى ملف المسجلات كمواقع ذاكرية .كما يحتوي حيز ذاكرة I/O على أربع وستين عنواناً مخصصة لوظائف وحدة المعالجة المركزية CPU المحيطية peripheral ، مثل مسجلات التحكم ، المؤقتات/العدادات ، المبدلات A/D ، و وظائف I/O الأخرى . ونستطيع الولوج إلى حيز I/O بشكل مباشر ، أو من خلال عناوين حيز المعطيات التالي لملف المسجلات ($20 - $5F)

يتوضع المكدس فعلياً في ذاكرة المعطيات العامة SRAM ، و بالتالي فإن حجم المكدس يتحدد فقط بحجم الذاكرة SRAM الإجمالي . ويقع على عاتق المبرمج تهيئة مؤشر المكدس SP في روتين خدمة التصفير reset وذلك قبل تنفيذ البرامج الفرعية و المقاطعات . ومؤشر المكدس هو عبارة عن مسجل بعرض 8–bit نستطيع قراءته أو الكتابة عليه ، ويتم الوصول إليه بعنوانه المتوضع في حيز ذاكرة I/O .

ملف المسجلات ذات الأغراض العامة :

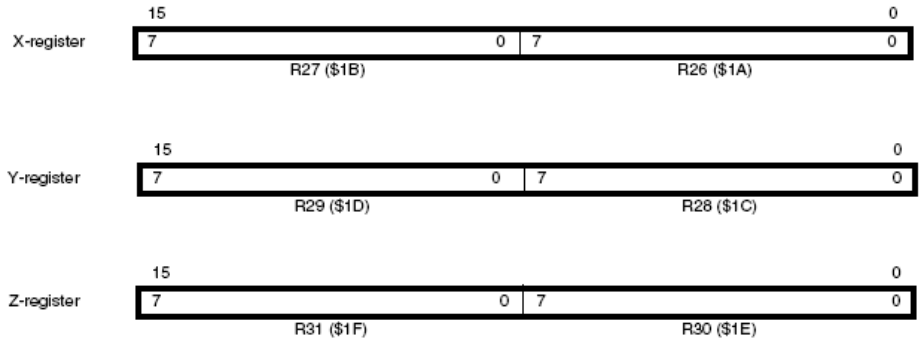
يُبيّن الشكل (٣) تركيبة مسجلات الأغراض العامة الاثنين و الثلاثين في وحدة المعالجة المركزية CPU .



يتألف ملف المسجلات من اثنين و ثلاثين مسجل عمل ، تشكل آخر ستة من هذه المسجلات ثلاث مسجلات مؤشر بطول 16-bit تستخدم في العنونة غير المباشرة لعنونة حيز المعطيات ، والتي تسهل حساب العنوان الفعال ، كما يستخدم أحد هذه المسجلات الثلاث كمؤشر عنونة للجداول المنشأة في ذاكرة البرنامج Flash . نرمز لهذه المسجلات الثلاث بالرموز التالية : المسجل X ، المسجل Y ، المسجل Z .

إن تعليمات مسجل العمل في مجموعة التعليمات لها دورة ساعة واحدة و ذلك بالنسبة لكامل ملف المسجلات . يستثنى من ذلك فقط خمس تعليمات حسابية و منطقية : SBCI ، SUBI ، CPI ، ANDI ، ORI بين عدد ثابت ومسجل ، وكذلك بالنسبة لتعليمة تحميل المعطيات الثابتة الفورية LDI . و تطبق هذه التعليمات على مسجلات النصف الثاني لملف المسجلات R16...R13 .كما تطبق التعليمات SBC، SUB، CP، AND، OR وكل التعليمات الأخرى بين مسجلين أو على مسجل وحيد على كامل ملف المسجلات. و نلاحظ كما هو مبين في الشكل (٢) ، أن لكل مسجل عنوان في حيز عنونة ذاكرة المعطيات ، وتحتل هذه المسجلات على خريطة الذاكرة أول اثنان و ثلاثون موقعاً .

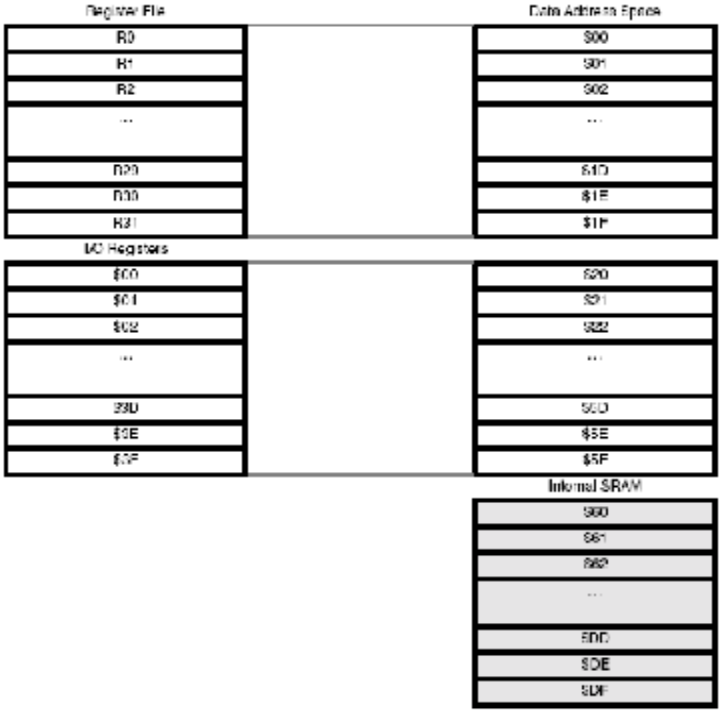
ومما يجدر ذكره هنا أن ملف المسجلات ليس منجزاً فيزيائياً كمواقع الذاكرة SRAM ، ولقد منح هذا التنظيم للذاكرة مرونة كبيرة في الولوج للمسجلات ، مثل المسجلات X، Y، Z التي تعمل كمؤشرات لأي مسجل في ملف المسجلات .



حيز ذاكرة I/O :

تتوضع مختلف الوظائف المحيطية و وظائف الدخل/الخرج للمتحكم AT90S2313 في حيز ذاكرة I/O . و يتم الولوج إلى مختلف مواقع I/O باستخدام التعليمتين IN و OUT اللتين تحولان المعطيات ما بين مسجلات العمل الاثنين والثلاثين ذات الأغراض العامة و حيز ذاكرة I/O . ونستطيع الولوج إلى خانات مسجلات حيز I/O الواقعة ضمن مجال العناوين ($20 - $5F) بشكل مباشر باستخدام التعليمتين CBI و SBI، وكذلك يمكننا فحص أي خانة من هذه المسجلات باستخدام التعليمتين SBIS و SBIC .

ملاحظة : عندما نستخدم أوامر محددة على حيز ذاكرة I/O ، فإنه يجب استخدام التعليمات : IN، OUT، SBIS ، SBIC مع عناوين حيز I/O ($3F - $00) . وعند عنونة مسجلات حيز I/O كذاكرة معطيات SRAM فإنه يجب إضافة القيمة $20 إلى هذا العنوان .



نوافذ الدخل/الخرج I/O Ports

النافذة B Port B

النافذة B هي عبارة عن نافذة دخل/خرج ثنائية الاتجاه ذات ثمانية أقطاب 8-bit bi-directional .

تتوضع عناوين مسجلات النافذة B الثلاث في حيز ذاكرة I/O وهي : مسجل المعطيات PORTB الذي يتوضع عند العنوان (38$)18$ ، مسجل اتجاه المعطيات DDRB الذي يتوضع عند العنوان (37$)17$ ، عنوان أقطاب دخل النافذة PINB – B الذي يتوضع عند العنوان (36$)16$. ويجب الانتباه إلى أن عنوان أقطاب النافذة B يقرأ فقط ، بينما تنفذ عمليات القراءة والكتابة على مسجل المعطيات و مسجل اتجاه المعطيات . تتم عملية اختيار ترانزيستور الرفع MOS pull-up بشكل مستقل لكل قطب من أقطاب النافذة B . وباستطاعة دارة قيادة خرج النافذة B تقديم تيار قدره 20mA قادر على قيادة وحدة إظهار LED بشكلٍ مباشر . وعند استخدام أقطاب النافذة B كأقطاب دخل ، فإنها تصبح منبعاً للتيار (IIL) عند ربطها خارجياً مع المنطق المنخفض و ذلك إذا كان ترانزيستور الرفع الداخلي مُفعّلاً .

لأقطاب النافذة B وظائف أخرى مبينة في الجدول (16) التالي :

قطب النافذة	الوظيفة الخاصة
PB0	AIN0 (المدخل الموجب للمقارن التشابهي)
PB1	AIN1 (المدخل السالب للمقارن التشابهي)
PB3	OC1 (قطب خرج مقارنة نظير المؤقت/العداد ١)
PB5	MISO (قطب دخل المعطيات أثناء برمجة الذاكرة)
PB6	MISO (قطب خرج المعطيات أثناء قراءة الذاكرة)
PB7	SCK (قطب دخل الساعة التسلسلية)

الجدول (16) : الوظائف الخاصة لأقطاب النافذة B

عند استخدام أقطاب هذه النافذة لخدمة الوظائف الخاصة الأخرى التي تتمتع بها ، فإنه يجب ضبط قيم مسجل اتجاه المعطيات DDRB ومسجل النافذة PORTB بما يتلاءم مع شرح هذه الوظائف .

Bit	7	6	5	4	3	2	1	0	
$18 ($38)	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	PORTB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

مسجل معطيات النافذة B – PORTB The Port B Data Register-PORTB

Bit	7	6	5	4	3	2	1	0	
$17 ($37)	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	DDRB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

مسجل اتجاه معطيات النافذة B – DDRB The Port B Direction Register-DDRB

عنوان دخل أقطاب دخل النافذةB - PINB **The Port B Input PINS Address-PINB**

Bit	7	6	5	4	3	2	1	0	
$16 ($36)	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	PINB
Read/Write	R	R	R	R	R	R	R	R	
Initial value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

إن عنوان أقطاب النافذة PINB - B ليست مسجلاً فيزيائياً . حيث نستطيع الوصول بهذا العنوان إلى القيم الفيزيائية الموجودة على كل قطب من أقطاب النافذة B . فعند قراءة مسجل النافذة PORTB – B فإننا نقرأ معطيات ماسك latch النافذة B . أما عندما نقرأ عنوان أقطاب دخل النافذة B فإننا نقرأ القيم المنطقية الحالية الموجودة على أقطاب هذه النافذة .

النافذة B كنافذة دخل/خرج عامة **Port B as General Digital I/O**

التعليق	ترانزيستور الرفع	I/O	**PORTBn**	**DDBn**
ممانعة عالية	بدون رفع	دخل	0	0
يصبح PBn منبعاً للتيار IIL إذا اتصل خارجياً مع المنطق المنخفض	مع رفع	دخل	1	0
خرج مرفوع إلى الصفر	بدون رفع	خرج	0	1
خرج مرفوع إلى الواحد	بدون رفع	خرج	1	1

الجدول (17) : تأثير الخانة DDRBn على أقطاب النافذة B

حيث أن : n،7 : 0...6 رقم القطب

النافذة D **Port D**

لأقطاب النافذة D وظائف أخرى مبينة في الجدول (18) التالي :

قطب النافذة	الوظيفة الخاصة
PD0	RXD (مدخل استقبال وحدة UART)
PD1	TXD (قطب ارسال وحدة UART)
PD2	INT0 (مدخل المقاطعة الخارجية 0)
PD3	INT1 (مدخل المقاطعة الخارجية 1)
PD4	T0 (مدخل العداد 0)
PD5	T1 (مدخل العداد 1)

الجدول (18) : الوظائف الخاصة لأقطاب النافذة B

عند استخدام أقطاب النافذة D بوظائفها الأخرى فإنه يجب تحميل المسجلين DDRD و PORTD بما يتناسب مع هذه الوظائف .

مسجل معطيات النافذة D – PORTD **The Port D Data Register-PORTD**

Bit	7	6	5	4	3	2	1	0	
$12 ($32)	–	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	PORTD
Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

مسجل اتجاه معطيات النافذة D - DDRD **The Port D Data Direction Register-DDRD**

Bit	7	6	5	4	3	2	1	0	
$11 ($31)	–	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	DDRD
Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

عنوان أقطاب دخل النافذة D - PIND **The Port D Input Pins Address-PIND**

Bit	7	6	5	4	3	2	1	0	
$10 ($30)	–	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	PIND
Read/Write	R	R	R	R	R	R	R	R	
Initial value	0	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

إن عنوان النافذة D – PIND ليس مسجلاً ، حيث يُمكن هذا العنوان عملية الوصول إلى القيم الفيزيائية الموجودة على كل قطب من أقطاب النافذةD . فعند قراءة مسجل معطيات النافذة D – PORTD فإننا نقرأ معطيات ماسك latch النافذة D . أما عند قراءة عنوان أقطاب دخل النافذة D – PIND فإننا نقرأ القيم المنطقية الحالية الموجودة على أقطاب النافذة D .

التعليق	ترانزيستور الرفع	I/O	**PORTDn**	**DDRDn**
ممانعة عالية	بدون رفع	دخل	0	0
يصبح PDn منبعاً للتيار IIL إذا اتصل خارجياً مع المنطق المنخفض	مع رفع	دخل	1	0
خرج مرفوع إلى الصفر	بدون رفع	خرج	0	1
خرج مرفوع إلى الواحد	بدون رفع	خرج	1	1

الجدول (19) : تأثير الخانات DDRDn على أقطاب النافذة D

حيث أن : n،7 : 0...6 رقم القطب

المشروع الأول:
حركة أضواء LEDS .

نوافذ الدخل/الخرج (I/O Ports) :

أولاً : النافذة B
النافذة B هي عبارة عن نافذة دخل/خرج ثنائية الاتجاه ذات ثمانية أقطاب. تتوضع عناوين مسجلات النافذة B الثلاث في حيز ذاكرة I/O وهي :
- مسجل المعطيات PORTB الذي يتوضع عند العنوان ($18($38 .
- مسجل اتجاه المعطيات DDRB الذي يتوضع عند العنوان ($17($37 .
- عنوان أقطاب دخل النافذة B – PINB الذي يتوضع عند العنوان ($16($36 .

ويجب الانتباه إلى أن عنوان أقطاب النافذة B يقرأ فقط ، بينما تنفذ عمليات القراءة والكتابة على مسجل المعطيات و مسجل اتجاه المعطيات . تتم عملية اختيار ترانزيستور الرفع MOS pull-up بشكل مستقل لكل قطب من أقطاب النافذة B .

مسجل معطيات النافذة B – PORTB :

Bit	7	6	5	4	3	2	1	0	
$18 ($38)	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	PORTB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

مسجل اتجاه معطيات النافذة B – DDRB :

Bit	7	6	5	4	3	2	1	0	
$17 ($37)	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	DDRB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

عنوان دخل أقطاب دخل النافذة PINB-B :

Bit	7	6	5	4	3	2	1	0	
$16 ($36)	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	PINB
Read/Write	R	R	R	R	R	R	R	R	
Initial value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

إن عنوان أقطاب النافذة PINB-B ليست مسجلاً فيزيائياً . حيث نستطيع الوصول بهذا العنوان إلى القيم الفيزيائية الموجودة على كل قطب من أقطاب النافذة B . فعند قراءة مسجل النافذة PORTB – B فإننا نقرأ معطيات ماسك latch النافذة B . أما عندما نقرأ عنوان أقطاب دخل النافذة B فإننا نقرأالقيم المنطقية الحالية الموجودة على أقطاب هذه النافذة .

يبين الجدول التالي تأثير الخانة DDRBn على أقطاب النافذة B (حيث أن : ٦ ... n = 0 رقم القطب)

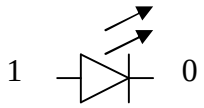
DDRBn	PORTBn	I/O	ترانزيستور الرفع	التعليق
0	0	دخل	بدون رفع	ممانعة عالية
0	1	دخل	مع رفع	يصبح PBn منبعاً للتيار IIL إذا اتصل خارجياً مع المنطق المنخفض
1	0	خرج	بدون رفع	خرج مرفوع إلى الصفر
1	1	خرج	بدون رفع	خرج مرفوع إلى الواحد

ويوجد مثل هذه المسجلات للمنفذ D و هي: DDRD,PORTD and PIND وجميع هذه المسجلات موجودة في المسجلات الخاصة الـ64 " SFR "Special Function Registers
على سبيل المثال إذا أردنا أن يكون نصف المنفذ B دخل مع مقاومة رفع ونصفه خرج وذلك بالتعاقب أي PIN دخل ثم الذي يليه خرج فيجب أن نكتب:

PORTB=0x0F ("00001111")
DDRB=0xAA ("10101010")

ونلاحظ هنا أنه تم تعريف PINB.1 PINB.3 PINB.5 PINB.7 على أنها جميعها خرج.
و PINB2. PINB.0 دخل ولكن مع وجود مقاومة رفع "أي أن قيمة الدخل الإفتراضية 1 ".
وPINB.4 PINB.6 دخل ولكن بدون مقاومة رفع . "أي أن قيمة الدخل الإفتراضية 0 ".

- يعرف الـ Led بأنه عبارة عن ثنائي ضوئي يمرر التيار الكهربائي عند تطبيق 1 منطقي على مصعده و 0 منطقي على مهبطه وأثناء مرور التيار يقوم بإصدار ضوء معين ويرمز له .

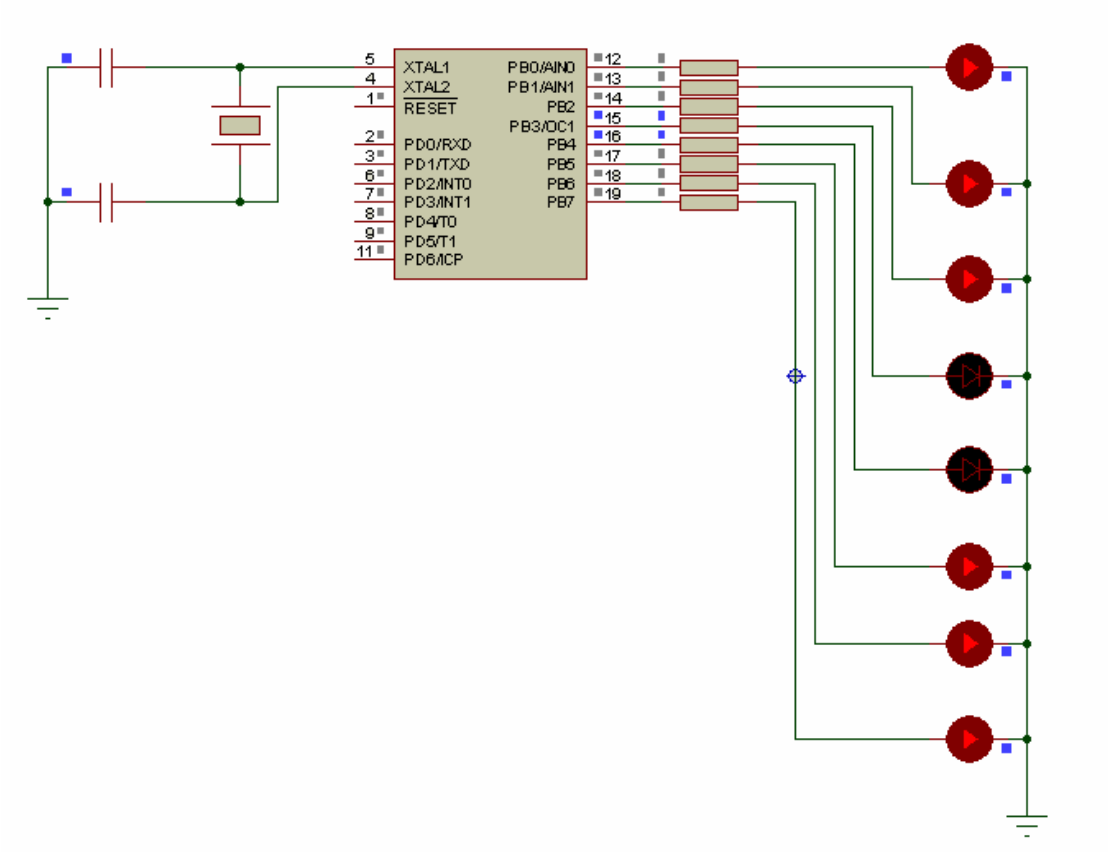


رمز الثنائي الضوئي Led

يوصل عادة مع الثنائي الضوئي على التسلسل مقاومة لحماية الثنائي من الإنهيارفي حال تطبيق جهود عالية ، وتقدر قيمتها في حال تطبيق جهد 5 فولط بـ 220 أوم .

سوف نقوم بكل المشاريع بالتكلم عن العناصر التي سوف نستخدمها في المشروع ثم شرح التعليمات الجديدة في المشروع ثم شرح المشروع بالكامل .

دارة المشروع :



مشروع حركة أضواء الليدات

برنامج المشروع :

```
#include <90s2313.h>          توجيه من أجل إدراج المكتبة المتعلقة بالمعالج ٢٣١٣

void main(void)          // التابع الرئيسي
{
PORTB=0x00;          // تهيئة المنفذ كله خرج
DDRB=0xff;

while (1)              // حلقة لانهائية
    {
     PORTB=0xff;          // إضاءة جميع الليدات
     delay();          // استدعاء التابع الفرعي المسؤول عن عملية التأخير الزمني
```

```
PORTB=0x00;
delay();

PORTB=0x0f;
delay();

PORTB=0xf0;
delay();

PORTB=0x81;
delay();

};          // نهاية الحلقة الانهائية
}           // نهاية التابع الرئيسي
```

```
void delay(void)        // تابع فرعي من أجل عملية التأخير
{
int t=999;        // تعريف متحول حقيقي وإسناد القيمة ٩٩٩له
while (t!=0)        // حلقة تستمر مادام العدد المتحول لايساوي الصفر
t--;                // إنقاص قية المتحول بمقدار ١
}                   // نهاية التابع الفرعي والعودة إلى مكان الإستدعاء
```

شرح تعليمات المشروع :

- #include <90s2313.h>

هذه التعليمة عبارة عن توجيه للمترجم بأن هذا البرنامج مخصص للمتحكم At90s2313 لذلك يقوم المتحكم بإدراج المكتبة المخصصة لهذا المعالج "كل تعليمة تبدأ بـ # فهي عبارة عن توجيه".

- التابع الرئيسي

```
void main(void) //
{
.....
.....
}
```

يعد هذا التابع هو الأساسي في كتابة البرنامج والذي منه يبدأ المترجم بتنفيذ عملية الترجمة وله الصيغة السابقة.
و(void) تعني بأن هذا التابع لا يستقبل أي قيمة و void التي في البداية تعني أن التابع لا يرجع أي قيمة وذلك لأن هذا التابع لايستدعى من قبل أي تابع آخر .

PORTB=0x00; -
DDRB=0xff; -

هاتان التعليمتان يستخدمان من أجل تهيئة المداخل والمخارج فالتعليمة PORTB=0x00; تعني أن أخرج على المنفذ B القيمة الست عشرية 00 حيث تدل الدالة 0X على أن الرقم مكتوب بالصيغة الست عشرية . والتعليمة DDRB=0xff; تعني بأن النافذة B كلها مخارج .
ففي بداية أي مشروع يجب تهيئة المسجلات التي سوف نستخدمها في مشروعنا قبل تنفيذ أي خطوة .

- حلقة لانهائية

```
while (1)
    {
    …….
    …….
    };
```

تستخدم هذه الحلقة من أجل استمرار عمل البرنامج دائماً والشرط (1) يعني أن هذا الشرط دائماً محقق .
ملاحظة لاتنسى الفاصلة المنقوطة في نهاية الحلقة .

ضمن هذه الحلقة نكتب البرنامج الذي سوف يقوم المعالج بتنفيذه دائماً .

```
PORTB=0xff;
```

أخرج القيمة الست عشرية 0xff على المنفذ B أي أضئ جميع الليدات الموصولة مع هذا المنفذ

```
delay();
```

وهي تعليمة استدعاء لتابع فرعي يدعى delay يقوم هذا التابع بعملية التأخير الزمني بين كل عمليتين إظهار وذلك حتى يتثنى لنا ملاحظة الحركة قبل أن تأتي الحركة التالية وكما نعلم بأن سرعة المعالج كبيرة جداً لذا فبدون هذه العملية لن نلاحظ هذه الحركات .

بعد ذلك نقوم بكتابة حركات جديدة ونضع بعد كل حركة التأخير الزمني delay .

تابع فرعي من أجل عملية التأخير

```
void delay(void)  //
{
int t=999;
while (t!=0)
t--;
}
```

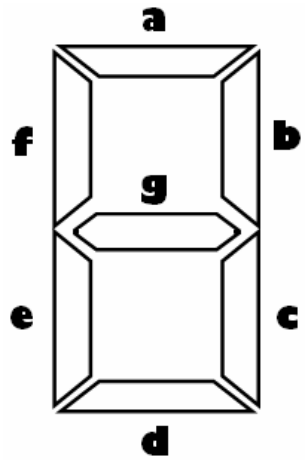
هذا التابع لايستقبل أي قيمة ولايعيد أي قيمة وإنما يقوم بعمليات تمتص نبضات المعالج من أجل التأخير الزمني ولتحقيق ذلك تم تعريف متحول حقيقي t يحتوي على القيمة ٩٩٩ ثم ندخل بحلقة while وتبقى هذه الحلقة مادام الشرط بين القوسين محقق وهو أن t لايساوي الصفر(t!=0) وضمن هذه الحلقة نقوم بإنقاص العدد t (t--) وتبقى الحلقة حتى تصبح قيمة العدد tمساوية للصفر وعندها يرجع المعالج إلى مكان استدعاء هذا التابع الفرعي في البرنامج الأساسي ويقوم بمتابعة تنفيذ التعليمات التالية.
ملاحظة : إذا كان ضمن حلقة while لايوجد إلا تعليمة واحدة فليس هناك حاجة للأقواس للحلقة .

المشروع الثاني
الإظهار على شاشات SEVEN SEGMENT

شاشات الإظهار ذات السبع قطع (.7-Seg) :

شاشة الإظهار ذات السبع قطع (.7-Seg) هي عبارة عن أداة الكترونية تحتوي على سبع قطع متوضعة بشكل يسمح بإظهار جميع الأرقام عبرها ، وخلف كل قطعة يوجد ليد ضوئي (LED) وتُضاء هذه الليدات وتُطفئ بشكل يسمح بإظهار الرقم المطلوب ، ويمكن عن طريق هذه الشاشة إظهار جميع الأرقام كما يمكن إظهار بعض الأحرف كأحرف النظام الست عشري مثلاً .

ويختلف الضوء الصادر عنها باختلاف نوعها ، فهناك الشاشات الحمراء والخضراء والصفراء وفي بعض الأحيان تأتي هذه الشاشات باللون البرتقالي ، ولكنها على اختلاف نوعها ولونها تمتلك جميعها سبع قطع مرتبة وفق الشكل السابق ويُرمز لكل قطعة فيها بحرف من الأحرف المبينة بالشكل التالي :



في الشكل المجاور إذا أردنا إظهار الرقم (0) مثلاً ، يجب علينا إضاءة القطع (a,b,c,d,e,f) وإطفاء القطعة (g) أما إظهار الرقم (1) فيتطلب إضاءة القطعتين (b,c) وإطفاء القطع (a,d,e,f,g) ، ولإظهار الرقم (2) يجب إضاءة القطع (a,b,g,e,d) وإطفاء القطعتين (c,f) وهكذا بالنسبة لبقية الأرقام .

تمتلك الخانة الواحد من شاشة الإظهار (.7-Seg) عادةً عشرة أقطاب متوضعة على طرفين متقابلين من الخانة إما في الأعلى والأسفل أو على الجانبين ، تأخذ القطع السبع سبعة أقطاب من الأقطاب العشرة بحيث أن كل قطب يتصل مع القطعة المقابلة له ، وهناك قطبين للتغذية كل قطب في طرف بحيث تعمل الخانة عند وصل التغذية السالبة أو الموجبة (بحسب نوع الخانة) إلى **أحد** قطبي التغذية ، والقطب العاشر متصل مع النقطة المتوضعة بجانب الخانة وبالتالي هو مسؤول عن إضاءتها .

ولشاشة الإظهار ذات السبع قطع (.7-Seg) نوعين رئيسيين :

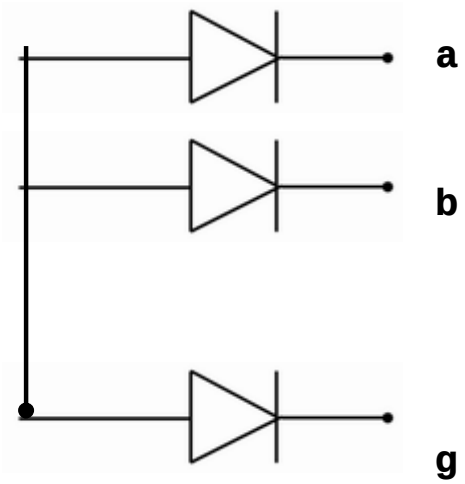
1 . الشاشة ذات المصاعد المشتركة .

2 . الشاشة ذات المهابط المشتركة .

1 . **الشاشة ذات المصاعد المشتركة** :

في هذه الحالة تكون مصاعد الليدات السبعة متصلة مع بعضها البعض وموصولة إلى قطب التغذية ويكون مهبط كل ليد متصل مع القطب المسؤول عن إضاءته ، حيث أن الليد في هذه الحالة يُضاء عندما يرد على مهبطه (0) منطقي ويُطفئ عندما يرد على مهبطه (1) منطقي مع الأخذ بعين الاعتبار أن يكون قطب التغذية للخانة موصول إلى جهد (1) منطقي ويُسمى هذا النوع عادةً في السوق **مُظهر رقمي موجب مشترك** .

ويُبيّن الشكل التالي الوصل الداخلي للّيدات في هذا النوع من شاشات الإظهار :

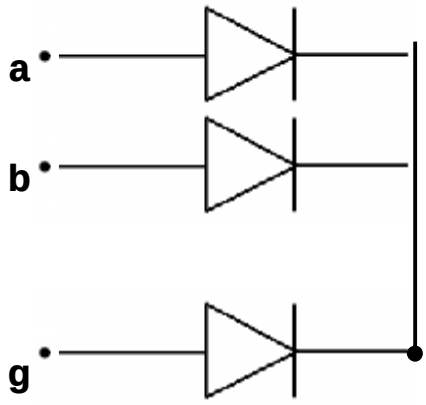


(0) منطقي (1) منطقي

2 . <u>الشاشة ذات المهابط المشتركة</u> :

في هذه الحالة تكون مهابط الليدات السبعة متصلة مع بعضها البعض وموصولة إلى قطب التغذية ويكون مصعد كل ليـد متصل مع القطب المسؤول عن إضاءته ، حيث أن الليد في هذه الحالة يُضاء عندما يرد على مصعده (1) منطقي ويُطفئ عندما يرد على مصعده (0) منطقي مع الأخذ بعين الاعتبار أن يكون قطب التغذية للخانة موصول إلى جهد (0) منطقي ويُسمى هذا النوع عادةً في السوق **مُظهر رقمي سالب مشترك** .

ويُبيّن الشكل التالي الوصل الداخلي للّيدات في هذا النوع من شاشات الإظهار :



الآن وبفرض أن لدينا شاشة إظهار ذات السبع قطع (7-Seg.) بخانة واحدة *<u>ذات مهابط</u>* <u>مشتركة</u> (سـالب مـشترك) و سنكتب فيما يلي الحالات المنطقية لأقطاب هذه الخانة المطلوبة لتمثيل الأرقام المختلفة :

الرقم	a	b	c	d	e	f	g	شكل الرقم على الشاشة
0	1	1	1	1	1	1	0	
1	0	1	1	0	0	0	0	
2	1	1	0	1	1	0	1	
3	1	1	1	1	0	0	1	
4	0	1	1	0	0	1	1	
5	1	0	1	1	0	1	1	
6	1	0	1	1	1	1	1	
7	1	1	1	0	0	0	0	
8	1	1	1	1	1	1	1	
9	1	1	1	1	0	1	1	

شرح دارة المشروع :

إنّ الدارة المطلوبة مُبيّنة في الشكل التالي :

ونلاحظ في هذه الدارة أنه تمّ وصل الأقطاب (a,b,c,d,e,f,g) للخانات الأربع مع أقطاب المتحكم (PB0,PB1,PB2,PB3,PB4,PB5.PB6) على الترتيب ، وبالتالي فإن النافذة (B) في المتحكم يجب أن تكون نافذة خرج ، كما نلاحظ مع أن أقطاب التفعيل للخانات الأربع تم وصلها مع أقطاب المتحكم (PD2,PD1,PD0) عن طريق أربعة ترانزستورات من النوع (NPN) والتي تعمل كمفتاح الكتروني ، إذ يقوم هذا الترانزيستور بوصل قطب التفعيل للخانة إلى الأرض (تفعيل هذه الخانة) عندما يرد (1) منطقي على قاعدته ، ويقوم بفصل قطب التفعيل للخانة عن الأرض (حجب هذه الخانة) عندما يرد (0) منطقي على قاعدته .

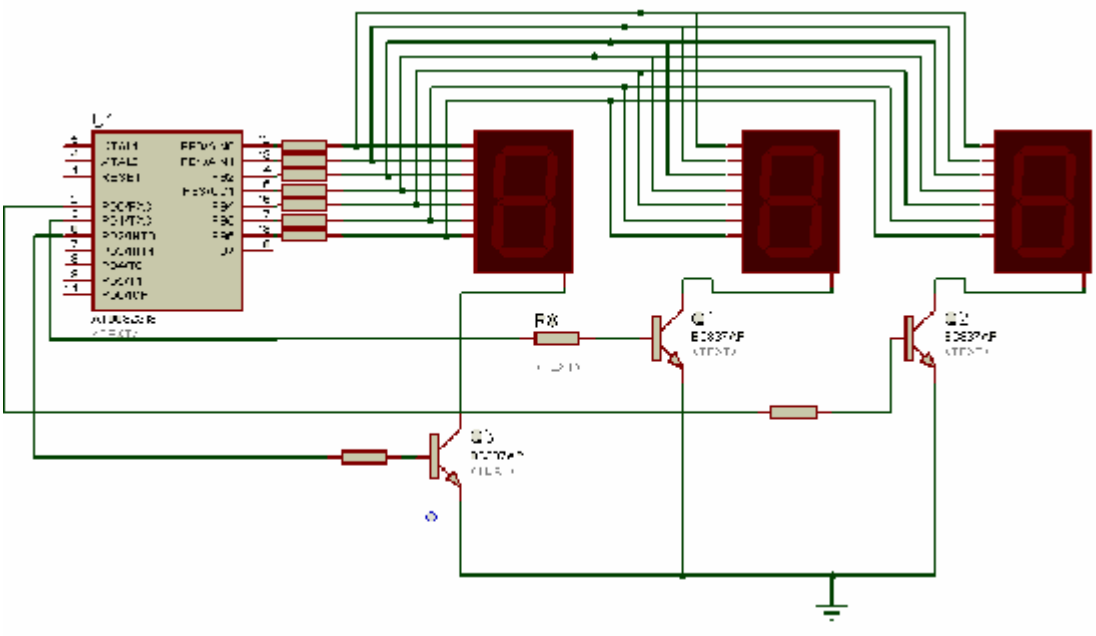
إذا دققنا في الرسم التخطيطي للدارة نجد أن مخارج المتحكم التالية تُقابل مداخل شاشة الإظهار التالية :

PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
*	g	f	e	d	c	b	a

وبالتالي فإن أي رقم نريد إظهاره يجب أن نعرف الشيفرة المقابلة له والتي تؤدي إلى إظهاره على شاشة الإظهار حتى نُخرّجها على أقطاب النافذة (B) ، فمثلاً لإظهار الرقم (1) يجب أن نُخرّج على النافذة (B) الرقم الثنائي التالي (00000110) والذي يساوي ست عشريّاً العدد (0x06) .

والآن لنكتب جدول يُبين شيفرات الأرقام المختلفة وفقاً للترتيب السابق :

الرقم عشرياً	PB7 *	PB6 g	PB5 f	PB4 e	PB3 d	PB2 c	PB1 b	PB0 a	الرقم الست عشري المكافئ
0	0	0	1	1	1	1	1	1	0x3f
1	0	0	0	0	0	1	1	0	0x06
2	0	1	0	1	1	0	1	1	0x5B
3	0	1	0	0	1	1	1	1	0x4f
4	0	1	0	1	0	1	1	0	0x66
5	0	1	1	0	1	1	0	1	0x6d
6	0	1	1	1	1	1	0	0	0x7c
7	0	0	0	0	0	1	1	1	0x07
8	0	1	1	1	1	1	1	1	0x7f
9	0	1	1	0	1	1	1	1	0x6f



دارة المشروع

البرنامج :

نريد إظهار الرقم 012 على المظهرات حيث الأحاد وهو 2 مخزن في متحول I والعشرات 1 في المتحول j والمئات 0 في المتحول o .

```
#include<90s2313.h>
void delay(void);
main()
{                                                    تعريف مصفوفة من أجل التشفير لإظهار الأرقام
unsigned char i=2,j=1,o=0;
unsigned char decode[10]={0x3f,0x06,0x5B,0x4f,0x66,0x6d,0x7c,0x07,0x7f,0x6f};


PORTD=0;
DDRD=255; // 255=0xff;          تهيئة المنفذ D كله خرج

PORTB=0;                  تهيئة المنفذ B كله خرج
DDRB=0XFF;

while (1)          حلقة لانهائية
  {
   PORTB=decode[i];                                  طبق على المنفذB تشفير عدد الأحاد i
   PORTD=1;  // تفعيل ترانزستور الأحاد
   delay();
```

```
   PORTD= 0;
   PORTB= decode[j];
   PORTD= 2;
   delay();


   PORTD = 0;
   PORTB = decode[o];
   PORTD = 4;
   delay();
   PORTD = 0;
  };   // end while

}  // end main

void delay(void)
{
int x=999;
while (x !=0)
x=x-1;
}
```

أنواع المتحولات
Data Types

Typeالنوع	Size (Bits)الحجم	Rangeالمجال
bit	1	0 , 1
char	8	-128 to 127
unsigned char	8	0 to 255
signed char	8	-128 to 127
int	16	-32768 to 32767
short int	16	-32768 to 32767
unsigned int	16	0 to 65535
signed int	16	-32768 to 32767
long int	32	-2147483648 to 2147483647
unsigned long int	32	0 to 4294967295
signed long int	32	-2147483648 to 2147483647
float	32	±1.175e-38 to ±3.402e38
double	32	±1.175e-38 to ±3.402e38

توابع التأخير الزمني
Delay Functions

تقوم هذه التوابع بالقيام بعملية التأخير الزمني في البرامج الكتوبة بلغة الـــ C وهذه التوابع مضمنة في المكتبة (delay.h) لذا يجب إدراج هذه المكتبة قبل استخدام أي تابع تأخير زمني .
وهذه التوابع هي :

void delay_us(unsigned int n)

يقوم هذا التابع بإجراء عملية التأخير الزمني في رتبة المايكرو ثانية حيث n يمثل زمن التأخير ويجب أن يكون من النمط integer أي صحيح .

void delay_ms(unsigned int n)

يقوم هذا التابع بإجراء عملية التأخير الزمني في رتبة الميلي ثانية حيث n يمثل زمن التأخير ويجب أن يكون من النمط integer أي صحيح .

ملاحظة عند إدراج هذه التوابع يجب أن لا نضع void وإلا سوف يظهر المترجم بأن هناك خطأ .

إدراج لغة الأسمبلي في البرنامج .

يمكن إدراج لغة الأسمبلي في كتابة البرنامج في أي منطقة وذلك باستخدام التوجيه #asm في بداية التصريح عن ان البرنامج بعده بلغة الأسمبلي ويجب أن ننتهي بالتوجيه #endasm دالاً على انتهاء لغة الأسمبلي .
على سبيل المثال :

```
#asm
        nop
        nop
 #endasm
```

أو يكننا ان نكتب التوابع بلغة الأسمبلي في نفس السطر .
Inline assembly may also be used.
Example:

```
#asm("sei") /* enable interrupts */
```

وفي حال كان هناك أكثر من تعليمة أسمبلي وأردنا كتابتها بنفس السطر فيجب عندها أن نفصل بين كل تعليمة بالإشارة \ . مثال :

```
#asm("nop\nop\nop")
```

التوابع الرياضية :

Mathematical Functions

توجد هذه التوابع في المكتبة math.h وهذه التوابع هي :

unsigned char cabs(signed char x)

إرجاع القيمةالمطلقة للعدد X من النمط المحرفي .

unsigned int abs(int x)

إرجاع القيمة المطلقة للعدد X من النمط الصحيح .

unsigned long labs(long int x)

إرجاع القيمة المطلقة للعدد X من النمط الصحيح الطويل .

float fabs(float x)

إرجاع القيمة المطلقة للعدد X من النمط الحقيقي ذو الفاصلة العائمة .

signed char cmax(signed char a, signed char b)

إرجاع القيمة الكبرى بين البايتين a و b .

int max(int a, int b)

إرجاع القيمة الكبرى بين العددين الصحيحين a و b .

long int lmax(long int a, long int b)

إرجاع القيمة الكبرى بين العددين الصحيحين الطويلين a و b .

float fmax(float a,float b)

إرجاع القيمة الكبرى بين العددين الحقيقين ذو الفاصلة العائمة a و b .

signed char cmin(signed char a, signed char b)

إرجاع القيمة الصغرى بين البايتين a و b .

int min(int a, int b)

إرجاع القيمة الصغرى بين العددين الصحيحين a و b .

long int lmin(long int a, long int b)

إرجاع القيمة الصغرى بين العددين الصحيحين الطويلين a و b .

float fmin(float a, float b)

إرجاع القيمة الصغرى بين العددين الحقيقين ذو الفاصلة العائمة a و b .

signed char csign(signed char x)

ارجاع القيمة 1- في حال كان البايت x سالب والقيمة 0 إذا كانت قيمته صفر والقيمة 1 إذا كان موجب .

signed char sign(int x)

ارجاع القيمة 1- في حال كان العدد الصحيح x سالب والقيمة 0 إذا كانت قيمته صفر والقيمة 1 إذا كان موجب .

signed char lsign(long int x)

ارجاع القيمة 1- في حال كان العدد الصحيح الطويل x سالب والقيمة 0 إذا كانت قيمته صفر والقيمة 1 إذا كان موجب .

signed char fsign(float x)

ارجاع القيمة 1- في حال كان العدد الحقيقي x سالب والقيمة 0 إذا كانت قيمته صفر والقيمة 1 إذا كان موجب .

unsigned char isqrt(unsigned int x)

ارجاع الجذر التربيعي للعدد x من النمط الصحيح غير المؤشر .

unsigned int lsqrt(unsigned long x)

ارجاع الجذر التربيعي للعدد x من النمط الصحيح الطويل غير المؤشر .

float sqrt(float x)

ارجاع الجذر التربيعي للعدد x من النمط الحقيقي الموجب .

float floor(float x)

ارجاع أصغر قيمة صحيحية للعدد الحقيقي ذو الفاصلة العائمة.

float ceil(float x)

ارجاع أكبر قيمة صحيحية للعدد الحقيقي ذو الفاصلة العائمة.

float fmod(float x, float y)

إرجاع باقي القسمة لناتج قسمة العدد الحقيقي x على العدد الحقيقي y .

float modf(float x, float *ipart)

يقوم هذا التابع بقسم العدد الحقيقي x ذو الفاصلة العائمة إلى قسمين قسم صحيح يتم تخزينه في العدد الحقيقي ذو الفاصلة العائمة ipart وإلى قسم الأجزاء أي التي تلي الفاصلة (0,45.....) والتي يتم إرجاعها بواسطة هذا التابع .

float ldexp(float x, int expn)

إرجاع القيمة x مرفوع للقوة 2*expn .

float frexp(float x, int *expn)

إرجاع الجزء العشري من اللوغاريتم والقوة للعدد x الحقيقي ذو الفاصلة العائمة .

float exp(float x)

إرجاع العدد e مرفقع للقيمة x.

.

float log(float x)

إرجاع اللوغاريتم الطبيعي للعدد الحقيقي ذو الفاصلة العائمة x .

float log10(float x)

إرجاع اللوغاريتم العشري للعدد الحقيقي ذو الفاصلة العائمة x .

float pow(float x, float y)

إرجاع القيمة x مرفوع للقوة y.

float sin(float x)

إرجاع sine(x) حيث x عدد حقيقي ذو فاصلة عائمة تمثل الزاوية مقدرة بالراديان .

float cos(float x)

إرجاع cosine (x) حيث x عدد حقيقي ذو فاصلة عائمة تمثل الزاوية مقدرة بالراديان .

float tan(float x)

إرجاع tangent (x) حيث x عدد حقيقي ذو فاصلة عائمة تمثل الزاوية مقدرة بالراديان .

float sinh(float x)

إرجاع hyperbolic sine(x) الجيب القطعي للعدد x حيث x عدد حقيقي ذو فاصلة عائمة تمثل الزاوية مقدرة بالراديان .

float cosh(float x)

إرجاع hyperbolic cosine(x) التجيب القطعي للعدد x حيث x عدد حقيقي ذو فاصلة عائمة تمثل الزاوية مقدرة بالراديان .

float tanh(float x)

إرجاع hyperbolic tangent(x) الظل القطعي للعدد x حيث x عدد حقيقي ذو فاصلة عائمة تمثل الزاوية مقدرة بالراديان .

float asin(float x)

إرجاع arc sine(x) الجيب العكسي ويتراوح العدد المرجع.(- PI/2 to PI/2) العدد x يجب أن يتراوح بين (1- و 1) .

float acos(float x)

إرجاع arc sine(x) التجيب العكسي ويتراوح العدد المرجع(0 وPI) العدد x يجب أن يتراوح بين (1- و 1)

float atan(float x)

إرجاع arc tangent(x) الظل العكسي ويتراوح العدد المرجع.(PI/2 to PI/2 -) .

float atan2(float y, float x)

إرجاع arc tangent(y/x) الظل العكسي ويتراوح العدد المرجع.(PI/2 to PI/2 -) .

المشروع الثالث
زيادة ونقصان عدد وإظهاره على شاشات SEVEN SEGMENT

شرحنا في الجلسة السابقة عن شاشة الإظهار 7-Seg بنوعيها (مهابط مشتركة ومصاعد مشتركة) وكيفية ربطها مع المتحكم الصغري ، والتحكم بها باستخدامه لإظهار رقم معين على الشاشة .

ولكن في هذه الجلسة المباركة . . سنتكلم عن كيفية ربط مفاتيح (Switches) مع المتحكم الصغري ، واستخدام هذه المفاتيح من أجل التحكم بشاشة الإظهار (كزيادة عدد أو إنقاصه) .
هناك أنواع كثيرة من المفاتيح التي يمكن ربطها ومنها :

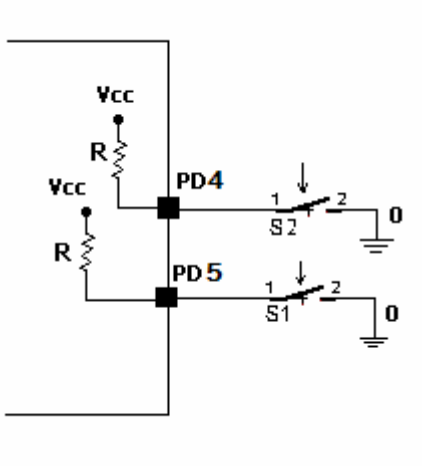
- المفاتيح اللحظية : وهي من النوع الكباس ، أي يتم الوصل في حال الضغط ويتم الفصل في حال رفع اليد عن المفتاح ، مثل أزرار لوحة المفاتيح .
- المفاتيح ON/OFF : وهي مفاتيح لها موضعين إما أن تكون ON أو OFF ، (مفاتيح الإنارة ...الخ) .
- Keypad : وهي مجموعة من المفاتيح اللحظية والمفاتيح ON/OFF ...

وهناك أنواع عديدة وكثيرة ، ولكن في هذا اليوم سنتعامل مع المفاتيح اللحظية .
سيتم ربط هذه المفاتيح مع المتحكم وحكماً إلى أحد أقطابه (النافذة B أو النافذة D) من أجل التحكم بالعدد الموجود على شاشة الإظهار وذلك إما لزيادته أو لإنقاصه ، وبالتالي سنستخدم اثنين من المفاتيح اللحظية أحدهما للزيادة والآخر للإنقاص .
سنستخدم قطبان من أقطاب المتحكم AT90S2313 وليكن PD4 و PD5 من النافذة D لأوصل إليهما المفتاحان S1 للزيادة و S2 للإنقاص على الترتيب .
بما أنني وصلت هذان المفتاحان إلى أقطاب المتحكم ، فإن الوظيفة التي يقومان بها هي إدخال قيم إلى المتحكم ("٠" منطقي أو "١" منطقي) وبالتالي يجب برمجة القطبين PD2 و PD3 كأقطاب دخل ورفع مقاومة الرفع الداخلية لكل منهما .
مقاومة الرفع الداخلية :
وهي مقاومة موصولة على التسلسل مع منبع جهد Vcc داخل المتحكم وذلك لكل قطب من أقطاب النافذتين B و D في المتحكم AT90S2313 كما في الشكل المجاور ..
وينم رفع هذه المقاومة برمجياً ، وتفيد هذه العملية فقط في الحالة التي يكون فيها أحد أقطاب النوافذ مبرمجاً على أنه دخل ، وبالتالي يتوضع على القطب القيمة "1" منطقي .
ويتم ذلك كمايلي :

```
PORTD=0x30; //0b00110000
DDRD=0xCF; //0b11001111
```

أي أننا نقوم بوضع "1" منطقي في الخانة التي نريدها دخل مع مقاومة رفع في المسجل PORTD ونضع "0" منطقي في الخانة التي نريدها أن تكون دخل في المسجل الذي يحدد إتجاه المنفذ وهو المسجل DDRD ٠

فعند وصل أحد طرفي مفتاح إلى أحد الأقطاب والطرف الآخر إلى التغذية (الأرضي) بدون الضغط على المفتاح تكون القيمة الابتدائية على القطب هي "1" منطقي ، أما عند الضغط عليه فيتم تمرير القيمة "0" منطقي (الأرضي) وتتحول القيمة على القطب من "1" إلى "0" وهذا يدل على ضغط المفتاح .



حالة المفتاح	القيمة على القطب
عدم الضغط	" 1 " منطقي
في حال الضغط	" 0 " منطقي

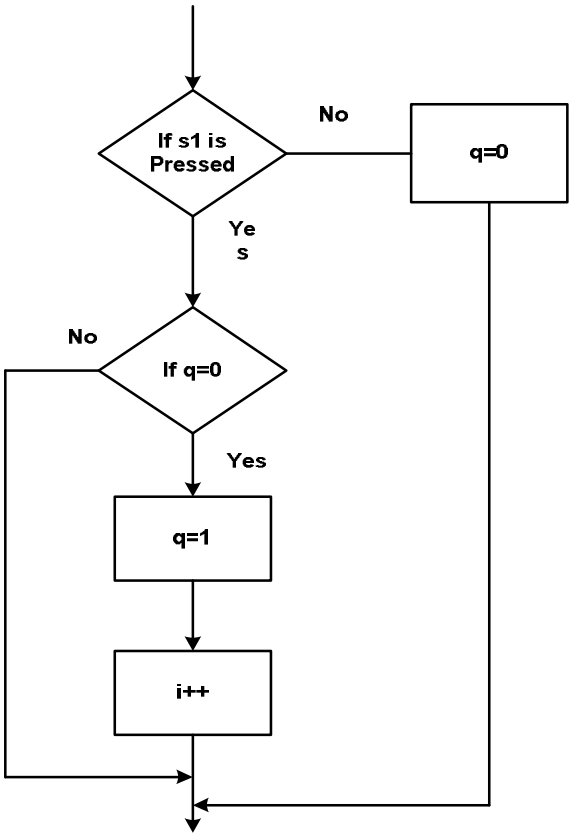
يتم تخزين القيمة الفيزيائية الموجودة على القطب في مسجل عنوان أقطاب الدخل للنافذة (PIN)
فيجب أخذ القيمة الموجودة على القطب (المخزنة في المسجل PIN) وفحصها هل هي"1" منطقي (عدم الضغط على المفتاح) أم "0" منطقي (حالة الضغط على المفتاح)

ولكشف عملية الضغط نقرأ المسجل المسؤول عن الدخل وهو PIND ونختبر الخانة رقم 4 لعملية الزيادة والخانة رقم 5 لعملية الإنقاص وبفرض العدد I هو العدد الذي يجب أن أزيده أو أنقصه ويتم ذلك برمجياً كمايلي:

```
While(1)
{
If(PIND.4==0)
{
I++;
}

If(PIND.4==0)
{
I++;
}
---------- عمليات الإظهار
------
------------------
------------

};// end while
```

ولكن كما نعلم أن سرعة المعالج كبيرة جداً وانه عندما نقوم بضغطة واحدة فإن عملية الزيادة أو النقصان سوف تتم بألاف المرات لذلك علينا استخدام حيلة أو طريقة لكي نتجاوز ذلك بأن يتم الزيادة أو النقصان مرة واحد لكل عملية ضغط والطريقة المعروفة لذلك هي عملية استخدام القفل البرمجي ويعمل كالتالي .

نقوم بتعريف متغير وليكن q=0 ونقول إذا تم الضغط على أي من الكباسين نقوم بفحص المتغير q فإذ كانت قيمته مساوية للصفر تتم عملية الزيادة أو النقصان ويتم وضع القيمة q=1 وإلا لاتتم العملية وطالما الكباس مضغوط تبقى قيمة المتغير q مساوية للواحد أما إذا رفعنا يدنا نقوم بتصفير المتغير q وهكذا ضمنا أنه لن تتم العملية حتى نقوم برفع يدنا والضغط مرة أخرى والمخطط النهجي التالي يفسر هذه العملية وذلك لعملية الزيادة .



وبنفس الطريقة تتم لعملية النقصان .

ولكي نقوم بإظهار العدد على الشاشات يجب أن نفصل العدد إلى الأحاد والعشرات والمئات ويتم ذلك كمايلي

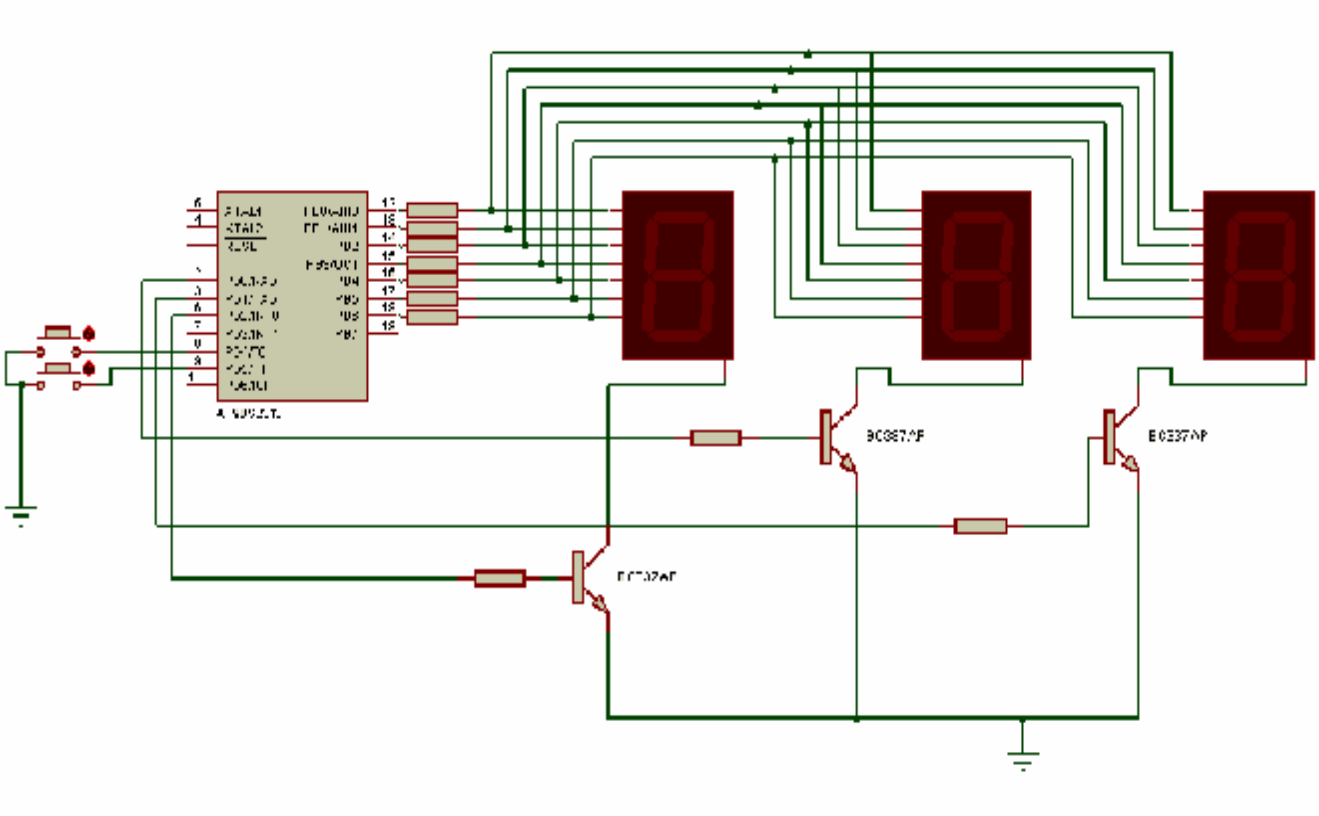
a=i%10; هذه التعليمة تقول إنه أسند للمتحول a قيمة باقي القسمة للعدد i على القيمة 10 أي قيمة الأحاد أصبحت في المتغيير a.

```
b=i/10;
c=b%10;
```

بعد ذلك نقوم بالقسمة الصحيحة للعدد iعلى عشرة أي اسند فقط القيمة الصحيحة للمتحول b وبذلك أصبح في المتحول b قيمة العشرات والمئات فقط فإذا ماقسمنا باقي القسمة للعدد b على عشرة نكون قد حصلنا على قيمة العشرات وأسندناها للمتحول c وولحصول على المئات ماعلينا إلى أن نقوم بالقسمة الصحيحة للمتحول i على مئة أي

```
b=i/100;
```

دارة المشروع :



البرنامج :

```
// This program shows number i and decrees - increases it by
 // Buttons connected to pind.4 and pind.5 respectively


#include<90s2313.h>
void delay(void);                              العدد i هو العدد الذي سوف نظهره على الشاشة


main()
{
int i=235,a,b,c;
unsigned char decode[10]={0x3f,6,0x5B,0x4f,0x66,0x6d,0x7c,0x07,0x7f,0x6f};
unsigned char q=0;

PORTD=0xf0;
DDRD=0x0f;

PORTB=00;
DDRB=0XFF;
```

```
while(1)
  {                                   إذا تم الضغط على كباس الإنقاص وكانت قيمة
  if(PIND.5==0 && q==0)               المتغير q مساوي للصفر
  {
  i--;
  q=1;
  }
  if(PIND.4==0 && q==0)
   {
  i++ ;
  q=1;
  }

  if (PIND.5==1 && PIND.4==1)         إذا كان كلا الكباسين غير مضغوطين صفر
  q=0;                                المتغيير q .


   a=i%10;
   PORTB=decode[a];
   PORTD|=0x01;
   delay();

   b=i/10;
   c=b%10;
   PORTD&=0xf0;
   PORTB=decode[c];
   PORTD|=0x02;
   delay();

   b=i/100;
   PORTD&=0xf0;
   PORTB=decode[b];
   PORTD|=0x04;
   delay();
   PORTD&=0xf0;
  };

}

void delay(void)
{
int x=999;
while (x !=0)
x=x-1;
}
```

المشروع الرابع
استخدام المقاطعات الخارجية INT0,INT1 لزيادة وإنقاص عدد وإظهاره على شاشات SEVEN SEGMENT

المقاطعات في المتحكم (**AT90S2313**) :

المقاطعة هي عبارة عن حادثة تُسبّب توقف المتحكم عن تنفيذ البرنامج الرئيسي ليذهب لتنفيذ برنامج خدمة المقاطعة وبعد الانتهاء منه يعود لتنفيذ البرنامج الرئيسي ابتداءاً من التعليمة التي توقف عندها عند حدوث المقاطعة ، ويختلف نوع المقاطعة باختلاف مصدرها ، فهناك المقاطعة الداخليّة والتي تنتج عن الوحدات الداخلية للمتحكم ، كمقاطعة طفحان المؤقت والعداد ، ومقاطعة المقارن التشابهي ، ومقاطعة نظير المؤقت والعداد .....الخ . وهناك المقاطعة الخارجية والتي تحدث نتيجة ورود إشارة مقاطعة من خارج المتحكم على قطب المقاطعة للمتحكم ، وعند ورود إشارة المقاطعة يدفع المتحكم عنوان التعليمة التي يُنفّذها حالياً إلى المُكدّس ثم يقوم بوضع عنوان بداية خدمة المقاطعة في مسجل عداد البرنامج (PC) ليقوم بتنفيذ برنامج المقاطعة وعند الوصول في هذا البرنامج إلى التعليمة (RETI) (وهي تعليمة العودة إلى البرنامج الرئيسي) يقوم المتحكم بإخراج (POP) العنوان الموجود في قمة المكدس

ويضعه في مسجل عداد البرنامج ليقوم المتحكم بمتابعة تنفيذ تعليمات البرنامج الرئيسي ابتداءاً من التعليمة التي توقف عندها عند ورود إشارة المقاطعة .

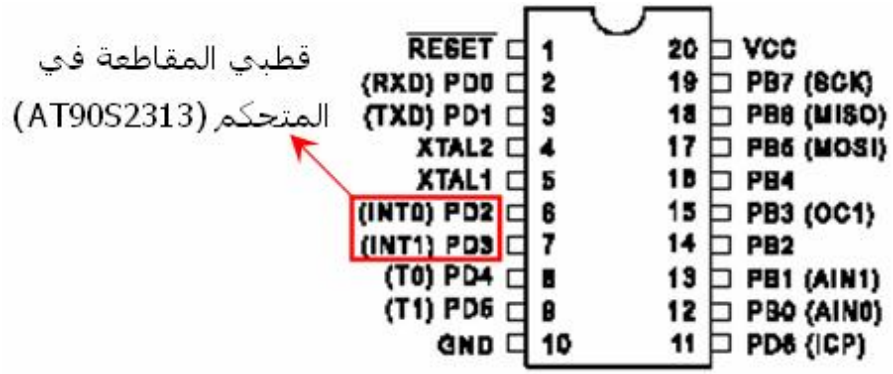
وسندرس في هذه الجلسة المقاطعات الخارجية عند المتحكم (AT90S2313) :

هناك نوعين من المقاطعة الخارجية في المتحكم (AT90S2313) وهما :

المقاطعة من النوع (0) (Int 0)

المقاطعة من النوع (1) (Int 1)

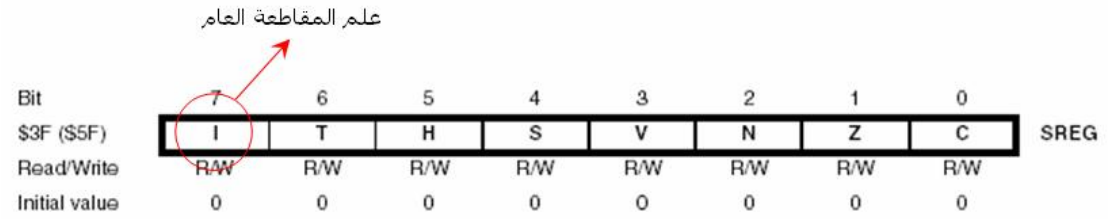
والشكل التالي يُبيّن توضع قطبي المقاطعة في المتحكم (AT90S2313) :



حيث نلاحظ في الشكل السابق أن الأقطاب (6,7) من المتحكم (AT90S2313) لها وظيفتين :
الوظيفة الأولى هي كون هذه الأقطاب جزء من النافذة (D) حيث يُشكّل هذين القطبين الخانتين (PD2 , PD3) من النافذة (PORT D) .
الوظيفة الثانية لهذين القطبين هي أنهما يُشكلان قطبي المقاطعة عند المتحكم ، بحيث يُشكل القطب (PD2) قطب المقاطعة (Int 0) ، ويُشكل القطب (PD3) قطب المقاطعة (Int 1) ، وهذا يعني أنه عند تفعيل المقاطعات الخارجية في المتحكم يجب التعامل حصراً مع هذين القطبين للمقاطعتين (Int 0,Int 1) .
ذكرنا أنه عند ورود مقاطعة للمتحكم فإنه سيتوقف عن تنفيذ البرنامج الرئيسي مؤقتاً وسيتجه لتنفيذ برنامج خدمة المقاطعة ابتداءاً من عنوان خدمة المقاطعة الموافق ، وهناك في المتحكم (AT90S2313) إحدى عشر مقاطعة داخلية وخارجية وهي مُبيّنة مع عنوان خدمة كل مقاطعة في الجدول التالي :

رقم شعاع المقاطعة	عنوان برنامج خدمة المقاطعة	مصدر المقاطعة	تعريف المقاطعة
1	$000	RESET	تصفير القطب الخارجي
2	$001	INT0	طلب مقاطعة خارجية (Int 0)
3	$002	INT1	طلب مقاطعة خارجية (Int 1)
4	$003	Timer1 CAPT1	حادثة مسك المؤقت / العداد 1
5	$004	Timer1 COMP1	مقارنة نظير المؤقت / العداد 1
6	$005	Timer1 OVF1	طفحان المؤقت / العداد 1
7	$006	Timer0 OVF0	طفحان المؤقت / العداد 0
8	$007	UART,RX	اكتمال استقبال وحدة (UART)
9	$008	UART,UDRE	فراغ مسجل معطيات وحدة (UART)
10	$009	UART,TX	اكتمال إرسال وحدة (UART)
11	$00A	ANA_COMP	المقارن التشابهي

الآن وبعد أن تعرّفنا على المقاطعات الخارجية وأنواعها في المتحكم (AT90S2313) سنبحث فيما يلي بكيفية ورود إشارات المقاطعة إلى المتحكم وكيف يستجيب لها أو يقوم بحجبها .
في البداية يجب أن نعلم أن هناك علم (Flag) في مسجل الحالة (SREG) (وهو أحد مسجلات التحكم) يُدعى بعلم المقاطعة العام (I) وهو متوضع في الخانة الثامنة من مسجل الحالة :



هذا العلم إذا وضعنا فيه القيمة (1) منطقي فهذا يعني أننا فعّلنا خدمة المقاطعات وبالتالي عند ورود أي مقاطعة للمتحكم سوف يستجيب لها بشكل طبيعي .

إما إذا وضعنا في هذا العلم القيمة (0) منطقي فهذا يعني أننا حجبنا المقاطعات بأكملها وبالتالي فإن أي مقاطعة ترد إلى المتحكم سوف يتم حجبها ولن يستجيب لها المتحكم أبداً إلى أن يتم تفعيل العلم (I) من جديد .
إذاً حتى نضمن استجابة المتحكم للمقاطعات الخارجية يجب أولاً وقبل كل شيء تفعيل علم المقاطعة العام (I) . يتم وضع القيمة (1) منطقي في علم المقاطعة العام (I) عن طريق التعليمة SEI
ويتم وضع القيمة (0) منطقي في علم المقاطعة العام (I) عن طريق التعليمة CLI

وبالإضافة إلى علم المقاطعة العام (I) هناك خانة تمكين منفصلة لكل مقاطعة من المقاطعات الخارجيّة ، إذا وضعنا في هذه الخانة (1) منطقي نكون قد فعّلنا المقاطعة الخارجية المقابلة وإذا وضعنا فيها (0) منطقي نكون قد حجبنا المقاطعة الخارجية المقابلة لهذه الخانة ، وخانتي تمكين المقاطعتين (Int 0,Int 1) موجودتين في أحد مسجلات التحكم والذي يُسمى مسجل قناع المقاطعة العام (General Interrupt MASK Register) (GIMSK) والذي له الشكل التالي :

Bit	7	6	5	4	3	2	1	0	
$3B ($5B)	INT1	INT0	–	–	–	–	–	–	GIMSK
Read/Write	R/W	R/W	R	R	R	R	R	R	
Initial value	0	0	0	0	0	0	0	0	

ونلاحظ توضّع خانتي التفعيل في الخانتين ذوات الرقمين (6,7) من المسجل (GIMSK)
وبالتالي فإذا أردنا تفعيل المقاطعة (Int 0) فقط علينا أن نكتب في المسجل (GIMSK) القيمة (0100 0000) = ($40) ، وإذا أردنا تفعيل المقاطعة (Int 1) فقط علينا أن نكتب في المسجل (GIMSK) القيمة (1000 0000) = ($80) ، وإذا أردنا تفعيل المقاطعتين (Int0,Int1) فقط علينا أن نكتب في المسجل (GIMSK) القيمة (1100 0000) = ($C0) وفي جميع الحالات يجب أن يكون علم المقاطعة العام (I) مفعّلاً.

مسجل أعلام المقاطعة العام (**GIFR**) :

مسجل أعلام المقاطعة العام (General Interrupt Flags Register) (GIFR) هو أحد مُسجلات التحكم والذي له الشكل التالي :

Bit	7	6	5	4	3	2	1	0	
$3A ($5A)	INTF1	INTF0	–	–	–	–	–	–	GIFR
Read/Write	R/W	R/W	R	R	R	R	R	R	
Initial value	0	0	0	0	0	0	0	0	

نلاحظ في هذا المسجل وجود خانتين : الخانة رقم (6) والخانة رقم (7) .

الخانة رقم (6) : تُمثّل علم المقاطعة الخارجية (Int 0) ، حيث أنّ ورود إشارة مقاطعة خارجية على قطب المتحكم (Int 0) يُؤدي إلى طلب مقاطعة فيصبح عندها العلم (INTF0) متوضعاً بالقيمة (1) منطقي ويقفز المتحكم إلى شعاع المقاطعة المُتوضّع عند العنوان ($001) في ذاكرة البرنامج ، وعند ورود تعليمة (RETI) في نهاية برنامج المقاطعة يُعيد المتحكم قيمة العلم (INTF0) إلى القيمة (0) منطقي استعداداً لتلقي مقاطعة أخرى من النمط (Int 0) ، وهذا يقودنا إلى أنه لا يُمكن حدوث مقاطعة أخرى من نفس النوع أثناء تنفيذ برنامج مقاطعة ما .

الخانة رقم (7) : تُمثّل علم المقاطعة الخارجية (Int 1) ، حيث أنّ ورود إشارة مقاطعة خارجية على قطب المتحكم (Int 1) يُؤدي إلى طلب مقاطعة فيصبح عندها العلم (INTF1) متوضعاً بالقيمة (1) منطقي ويقفز المتحكم إلى شعاع المقاطعة المُتوضّع عند العنوان ($002) في ذاكرة البرنامج ، وعند ورود تعليمة (RETI) في نهاية برنامج المقاطعة يُعيد المتحكم قيمة العلم (INTF1) إلى القيمة (0) منطقي استعداداً لتلقي مقاطعة أخرى من النمط (Int 1) ، وبالتالي لا يُمكن حدوث مقاطعة أخرى من نفس النوع أثناء تنفيذ برنامج مقاطعة ما .

ويُمكن التعامل مع مسجل أعلام المقاطعة العام (GIFR) بشكل مباشر بأن نكتب القيمة (0) منطقي أو (1) منطقي في خانتي أعلام المقاطعة .

أنواع إشارات المقاطعات الخارجيّة :

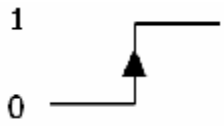
يُمكن ورود ثلاثة أنواع من الإشارات على أقطاب المقاطعة في المتحكم (AT90S2313) وهذه الأنواع مبينة فيما يلي :

(1) إشارة مستوي منخفض (0) منطقي :

ولها الشكل التالي :

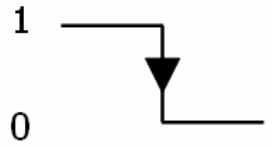
(2) إشارة جبهة صاعدة :

ولها الشكل التالي :



(3) إشارة جبهة هابطة :

ولها الشكل التالي :



إذاً عند ورود إحدى الإشارات الثلاث السابقة على قطب المقاطعة للمتحكم سوف تحدث المقاطعة ، ولكن كيف يُمكننا إعلام المتحكم بنوع إشارة المقاطعة التي يجب أن يستجيب لها عند ورودها على قطب المقاطعة ؟؟؟...

إن هذا يتم عن طريق أحد مسجلات التحكم والذي يُدعى مسجل التحكم بالمتحكم (MCUCR) والذي له الشكل التالي :

Bit	7	6	5	4	3	2	1	0	
$35 ($55)	-	-	SE	SM	ISC11	ISC10	ISC01	ISC00	MCUCR
Read/Write	R	R	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

ويتم التحكم بنوع الإشارة الواردة على قطب المقاطعة للمُتحكم عن طريق :

الخانتين 0 و 1 (ISC00 , ISC01) بالنسبة للمقاطعة (Int 0) .

الخانتين 2 و 3 (ISC10 , ISC11) بالنسبة للمقاطعة (Int 1) .

وذلك وفق الجدولين التاليين :

ISC00	ISC01	الحالة
0	0	إن تطبيق إشارة ذات مستوي منخفض ((0) منطقي) على القطب (Int 0) سوف يُولّد طلب المقاطعة
1	0	محجوزة
0	1	إن تطبيق إشارة جبهة هابطة على القطب (Int 0) سوف يولّد طلب المقاطعة
1	1	إن تطبيق إشارة جبهة صاعدة على القطب (Int 0) سوف يولّد طلب المقاطعة

ISC10	ISC11	الحالة
0	0	إن تطبيق إشارة ذات مستوي منخفض ((0) منطقي) على القطب (Int 1) سوف يُولّد طلب المقاطعة
1	0	محجوزة
0	1	إن تطبيق إشارة جبهة هابطة على القطب (Int 1) سوف يولّد طلب المقاطعة
1	1	إن تطبيق إشارة جبهة صاعدة على القطب (Int 1) سوف يولّد طلب المقاطعة

أما الخانات (4,5) من هذا المسجل فهي خاصة بنمط النوم للمتحكم وسيتم شرحها لاحقاً .

أولويّات المقاطعات :

عند ورود طلب مقاطعة من النمط (Int 0) إلى المتحكم أثناء تنفيذ برنامج مقاطعة من النمط (Int 0) (من نفس النمط) فإن المتحكم سوف يُهمل هذا الطلب ولن يستجيب لهذه المقاطعة (لأن خانة علم المقاطعة (INTF0) تحمل القيمة (1) منطقي) ، ونفس الأمر بالنسبة للمقاطعة (Int 1) .

أما إذا ورد طلب مقاطعة من نمط مختلف أثناء تنفيذ برنامج مقاطعة ما فإن استجابة المتحكم لتلك المقاطعة مرتبطة بأولوية المقاطعة الأولى على الثانية ، وتتحدد أولويات المقاطعات المختلفة وفق القاعدة التالية :

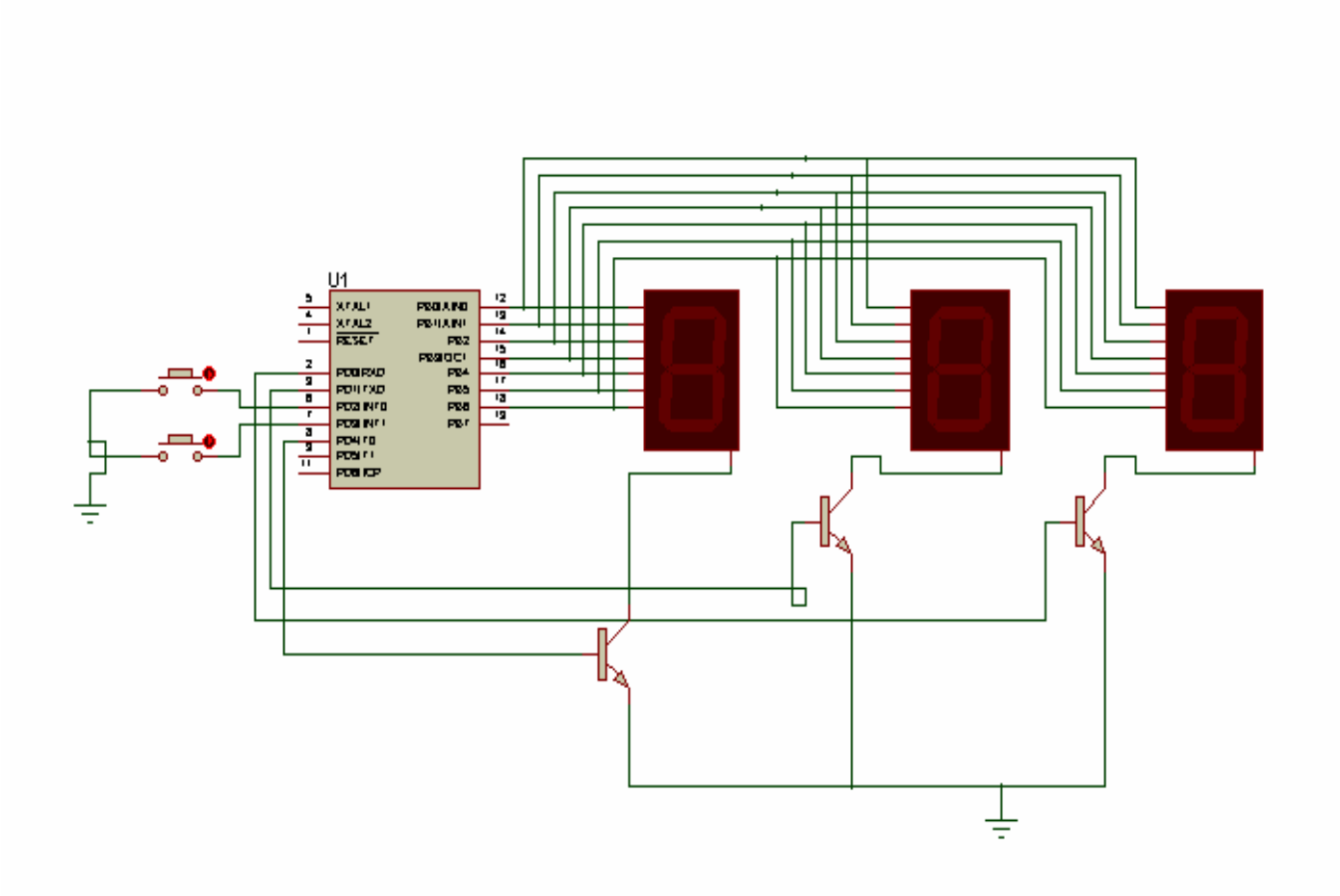
المقاطعة ذات العنوان الأصغر تمتلك الأولوية على المقاطعة ذات العنوان الأكبر.

فمثلاً المقاطعة (RESET) ذات العنوان ($000) لها أولوية على المقاطعة (Int 0) ذات العنوان ($001) والتي لها الأولوية هي الأخرى على المقاطعة (Int 1) ذات العنوان ($002) وهكذا بالنسبة لباقي المقاطعات.

وبالتالي إذا ورد طلب مقاطعة من النمط (Int 0) أثناء تنفيذ برنامج مقاطعة من النمط (Int 1) فإن المتحكم سوف يقف عند التعليمة التي ورد عندها طلب المقاطعة (Int 0) في برنامج المقاطعة (Int 1) وينتقل إلى العنوان ($001) ليُنفذ برنامج المقاطعة (Int 0) ثمّ يعود بعد الانتهاء منه ليُنفذ بقية برنامج المقاطعة (Int 1) ليعود بعد الانتهاء منه إلى البرنامج الرئيسي .

أما إذا حدث العكس أي إذا ورد طلب مقاطعة من النمط (Int 1) أثناء تنفيذ برنامج مقاطعة من النمط (Int 0) فإنّ المتحكم عندها سيُهمل طلب المقاطعة (Int 1) لأن المقاطعة (Int 0) لها أولوية على المقاطعة (Int 1) .

دارة المشروع



البرنامج :

```
#include <90s2313.h>
#include <delay.h>
 int i=235;                    //  Global Variable
                              // External Interrupt 0 service routine
interrupt [EXT_INT0] void ext_int0_isr(void)
{
i++;                          // INT0 subroutine
}
                                    // External Interrupt 1 service routine
interrupt [EXT_INT1] void ext_int1_isr(void)
{
i--;                      // INT1 subroutine
}

void main(void)
{

unsigned char decode[10]={0x3f,6,0x5B,0x4f,0x66,0x6d,0x7c, 0x07,0x7f,0x6f};
unsigned char a,b,c;
```

```
PORTB=0x00;             // I/O ports initialization
DDRB=0xFF;

PORTD=0xfc;
DDRD=0x13;
                                  // External Interrupt(s) initialization
                                 // INT0: On
                                // INT0 Mode: Falling Edge
                               // INT1: on
                              // INT1 Mode: Falling Edge
GIMSK=0xC0;
MCUCR=0x0A;
GIFR=0xC0;
#asm("sei")          // Global enable interrupts

 while(1)
   {

    PORTD.4=0;
    delay_ms(100);
    a=i%10;
    PORTB=decode[a];
    PORTD.0=1;

    b=i/10;
    c=b%10;
    delay_ms(100);
    PORTD.0=0;
    PORTB=decode[c];
    PORTD.1=1;

    b=i/100;
    delay_ms(100);
    PORTD.1=0;
    PORTB=decode[b];
    PORTD.4=1;
    delay_ms(100);

   }; // end while

}  // end main
```

المشروع الخامس
التخزين في ذاكرة EEPROM الداخلية

دورة القراءة/الكتابة لذاكرة المعطيات EEPROM EEPROM Read/Write Access

نستطيع الولوج إلى مسجلات التحكم بذاكرة المعطيات EEPROM الموجودة في حيز ذاكرة I/O .
إن زمن الولوج access عند الكتابة على الذاكرة يقع ضمن المجال 4ms – 2.5 . وهذا يعتمد على جهد التغذية ، ووظائف التزامن الداخلي . و على كلٍ يستطيع المبرمج كتابة برمجيات صغيرة لاكتشاف جاهزية كتابة بايت آخر . و تستمر عملية الكتابة على الذاكرة EEPROM حتى لو حدثت إحدى حالات التصفير .و لمنع حدوث عملية كتابة غير مقصودة على ذاكرة المعطيات EEPROM ، فإنه يجب علينا إتباع إجراء الكتابة المشروح في فقرة " مسجل التحكم بالذاكرة EEPOM " .
تتوقف وحدة المعالجة المركزية CPU لمدة دورتي ساعة قبل تنفيذ التعليمة التالية ، عند الكتابة على الذاكرة EEPROM . بينما تتوقف وحدة المعالجة المركزية CPU لمدة أربع دورات ساعة قبل تنفيذ التعليمة التالية ، عند القراءة من الذاكرة EEPROM .
و عند القراءة من أو الكتابة على ذاكرة المعطيات EEPROM . فإن وحدة المعالجة المركزية CPU تتوقف لمدة دورتي ساعة قبل تنفيذ التعليمة التالية .

مسجل عنوان ذاكرة المعطيات EEPROM - EEAR

The EEPROM Address Register – EEAR

Bit	7	6	5	4	3	2	1	0	
$1E ($3E)	EEAR7	EEAR6	EEAR5	EEAR4	EEAR3	EEAR2	EEAR1	EEAR0	EEAR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	X	X	X	X	X	X	X	X	

يقوم هذا المسجل EEAR بعنونة حيز ذاكرة المعطيات EEPROM التي يبلغ طولها 128/256 bytes .و يتم عنونة بايتات الذاكرة EEPROM بشكل خطي بين 128/256…0 .و القيمة البدائية لهذا المسجل غير معروفة . ويجب علينا كتابة عنوان البايت قبل تنفيذ الولوج للذاكرة EEPROM .

مسجل معطيات ذاكرة المعطيات EEPROM – EEDR

Bit	7	6	5	4	3	2	1	0	
$1D ($3D)	MSB							LSB	EEDR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

The EEPROM Data Register – EEDR

الخانات EEDR7…0 – 7…0 : خانات معطيات ذاكرة المعطيات EEPROM :

الفارس لتقنيات الحاسوب والإتصالات
سوريا – حماه- هـ-229619 33 963+

عند الكتابة على الذاكرة EEPROM ، فإن المسجل EEDR سيحتوي على المعطيات المكتوبة للذاكرة EEPROM عند العنوان المعطى في المسجل EEDR . أما عند القراءة من الذاكرة EEPROM فإن المسجل EEDR ستحتوي على المعطيات المقروءة من الذاكرة EEPROM عند العنوان المعطى في المسجل EEAR .

مسجل التحكم بذاكرة المعطيات EEPROM – EECR

The EEPROM Control Register – EECR

Bit	7	6	5	4	3	2	1	0	
$1C ($3C)	-	-	-	-	EERIE	EEMWE	EEWE	EERE	EECR
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

الخانات Res – 7...4: خانات محجوزة :

هذه الخانات هي خانات محجوزة في المتحكم AT90S4433 وتقرأ دائماً صفر منطقي .

الخانة 3 - EERIE : خانة تمكين مقاطعة جاهزية الذاكرة EEPROM :

تؤهل مقاطعة جاهزية الذاكرة EEPROM عند تأهيل كلٍ من الخانتين : خانة تمكين المقاطعة العامة I=1 الواقعة في مسجل الحالة SREG و الخانة EERIE=1 . و تولد مقاطعة جاهزية الذاكرة EEPROM مقاطعة ثابتة عند تصفير خانة تمكين الكتابة EEWE=0 .

الخانة 2 – EEMWE : خانة تمكين قيادة الكتابة على الذاكرة EEPROM :

تحدد الخانة EEMWE هل بإمكان الخانة EEWE إذا ما فُعّلت "١" أن تحرر الكتابة على الذاكرة EEPROM. فعندما نُفعّل الخانة القائدة EEMWE=1 ، عندئذ تستطيع خانة تمكين الكتابة عند تفعيلها EEWE=1 أن تكتب المعطيات على الذاكرة EEPROM عند العنوان المحدد في المسجل EEAR . أما إذا صفرت الخانة القائدة للكتابة EEMWE=0 ، فإن خانة تمكين الكتابة EEWE تصبح عديمة التأثير . وعند تفعيل الخانة EEMWE برمجياً ، فإن الكيان الداخلي للمتحكم MCU يقوم بتصفير هذه الخانة بعد أربع دورات ساعة . أنظر شرح الخانة EEWE عند إجراء الكتابة على ذاكرة المعطيات EEPROM .

الخانة 1 – EEWE : خانة تمكين الكتابة على ذاكرة المعطيات EEPROM :

تقدح إشارة تمكين الكتابة (EEWE) عملية الكتابة على ذاكرة المعطيات EEPROM عند تفعيلها EEWE=1 فعند الكتابة على الذاكرة EEPROM يجب علينا أولاً تحميل المعطيات والعنوان بشكل صحيح للمسجلين EEDR و EEAR ، ومن ثم نفعل الخانة EEWRE=1 . كما يجب أن تكون الخانة EEMWE مُفعّلة عند كتابة المنطق واحد على الخانة EEWE ، وفي كل الحالات الأخرى لن تحدث عملية الكتابة على ذاكرة المعطيات EEPROM . يجب إتباع الإجراء التالي عند الكتابة على الذاكرة EEPROM (وقد تكون الخطوتين ٢،٣ غير ضروريتين) :

١- أنتظر حتى تصبح الخانة EEWE = 0 .

٢- اكتب عنوان حجرة الذاكرة EEPROM في مسجل العنوان EEAR (اختياري) .

٣- اكتب معطيات الذاكرة EEPROM في مسجل المعطيات EEDR (اختياري) .

٤- اكتب المنطق واحد "١" على الخانة القائدة EEMWE الموجودة في المسجل EECR .

٥- خلال دورات الساعة الأربعة بعد تفعيل الخانة EEMWE اكتب المنطق واحد "١" على خانة تمكين الكتابة EEWE=1 .

الفارس لتقنيات الحاسوب والإتصالات
سوريا – حماه- هـ-229619 33 963+

تحذير : إذا حدثت المقاطعة بين الخطوة 4 و 5 ، فإن ذلك سيؤدي إلى فشل دورة الكتابة على الذاكرة EEPROM ، ويتم تجاهل خانة تمكين قيادة الكتابة EEMWE . و من ناحية ثانية ، سيؤدي روتين المقاطعة الذي قاطع عملية الولوج للذاكرة EEPROM إلى تعديل قيمة مسجلي العنوان و المعطيات الذاكرة EEPROM بقيم عشوائية . ونحن نوصي دائماً في مثل هذه الحالات بحجب جميع المقاطعات بتصفير خانة تمكين المقاطعة العامة I=0 أثناء تنفيذ الخطوات الأربعة الأخيرة من إجرائية الكتابة على ذاكرة المعطيات EEPROM ، وذلك لتجنب الوقوع في مشكلة فشل عملية الكتابة .

يقوم كيان Hardware المتحكم MCU الداخلي بتصفير خانة تمكين الكتابة EEWE=0 عند انقضاء زمن ولوج الكتابة (يقدر هذا الزمن بشكل نموذجي بحوالي 2.5ms عند الجهد Vcc = 5.0V . وبزمن قدره 4ms عند الجهد Vcc = 2.7V) . ويقع على عاتق المبرمج كتابة برنامج بسيط يفحص فيه رجوع الخانةEEWE إلى الصفر ، حتى يتمكن من كتابة بايت المعطيات التالي . و عند تفعيل خانة تمكين الكتابة EEWE تتوقف وحدة المعالجة المركزية CPU لمدة دورتي ساعة قبل تنفيذ التعليمة التالية .

الخانة 0 – EERE : خانة تمكين القراءة من ذاكرة المعطيات EEPROM :

تقدح إشارة تمكين القراءة EERE عملية القراءة من ذاكرة المعطيات EEPROM عند تفعيلها EERE=1 . فعند القراءة من الذاكرة EEPROM . يجب علينا أولاً تحميل العنوان الصحيح في مسجل العنوان EEAR ، ومن ثم نفعل خانة تمكين القراءةEERE=1 . نستطيع الحصول على المعطيات المطلوبة من مسجل معطيات الذاكرة EEDR ، وذلك عندما يقوم كيان المتحكم MCU بتصفير خانة تمكين القراءة EERE . و تستهلك عملية الولوج عند القراءة من الذاكرة EEPROM زمن تعليمة واحدة فقط ، لذلك ليست هناك أي ضرورة لفحص الخانةEERE . كما أن وحدة المعالجة المركزية CPU تتوقف لمدة دورتي ساعة قبل تنفيذ التعليمة التالية ، وذلك عند تفعيل خانة تمكين القراءة EERE=1.

يقع على عاتق المبرمج فحص خانة تمكين الكتابة EEWE قبل بداية عملية القراءة من ذاكرة المعطيات EEPROM ، فإذا كانت عملية الكتابة في حالة تقدم (معالجة) ، وكتبت المعطيات الجديدة والعنوان الجديد على مسجلات الذاكرة EEPROM الموجودين في حيز ذاكرة I/O ، فإنه يتم مقاطعة عملية الكتابة . والنتيجة تكون غير معروفة .

تجنب إفساد الذاكرة EEPROM **Prevent EEPROM Corruption**

قد تفسد معطيات الذاكرةEEPROM بسبب جهود التغذية VCC المنخفضة غير الملائمة لعمل كلٍ من وحدة المعالجة المركزية CPU و الذاكرة EEPROM . و يرجع ذلك إلى مستوى الأنظمة التي تتوضع فيها CPU و الذاكرة EEPROM و الحلول التصميمية المطبقة على كامل التطبيق . و يعود سبب إفساد معطيات الذاكرة EEPROM عند انخفاض جهد التغذية إلى حالتين. الأول ، تتطلب عمليات الكتابة المنتظمة على الذاكرة EEPROM جهود ضمن الحدود الدنيا المسموح بها كي يتم العمل بشكل صحيح . و الثاني ، قد تنفذ وحدة المعالجة المركزية CPU التعليمات بشكل خاطئ إذا انخفض جهد التغذية أثناء تنفيذ التعليمات .و بإمكاننا تجنب إفساد معطيات الذاكرة EEPROM بسهولة بإتباع الوصايا التصميمية التالية (و يكفي إتباع الوصية الأولى) :

١. الحفاظ على فعالية تصفير المتحكم MCU (التصفير عند المنطق المنخفض) أثناء دورات الجهود المنخفضة غير الكافية . و يتحقق ذلك بتفعيل دارة كاشف حالة Brown-out الداخلية BOD إذا كان سرعة

عند الكتابة على الذاكرة EEPROM ، فإن المسجل EEDR سيحتوي على المعطيات المكتوبة للذاكرة EEPROM عند العنوان المعطى في المسجل EEDR . أما عند القراءة من الذاكرة EEPROM فإن المسجل EEDR ستحتوي على المعطيات المقروءة من الذاكرة EEPROM عند العنوان المعطى في المسجل EEAR .

مسجل التحكم بذاكرة المعطيات EEPROM – EECR

The EEPROM Control Register – EECR

Bit	7	6	5	4	3	2	1	0	
$1C ($3C)	-	-	-	-	EERIE	EEMWE	EEWE	EERE	EECR
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

الخانات Res – 7...4: خانات محجوزة :

هذه الخانات هي خانات محجوزة في المتحكم AT90S4433 وتقرأ دائماً صفر منطقي .

الخانة 3 - EERIE : خانة تمكين مقاطعة جاهزية الذاكرة EEPROM :

تؤهل مقاطعة جاهزية الذاكرة EEPROM عند تأهيل كلٍ من الخانتين : خانة تمكين المقاطعة العامة I=1 الواقعة في مسجل الحالة SREG و الخانة EERIE=1 . و تولد مقاطعة جاهزية الذاكرة EEPROM مقاطعة ثابتة عند تصفير خانة تمكين الكتابة EEWE=0 .

الخانة 2 – EEMWE : خانة تمكين قيادة الكتابة على الذاكرة EEPROM :

تحدد الخانة EEMWE هل بإمكان الخانة EEWE إذا ما فُعّلت "١" أن تحرر الكتابة على الذاكرة EEPROM. فعندما نُفعّل الخانة القائدة EEMWE=1 ، عندئذ تستطيع خانة تمكين الكتابة عند تفعيلها EEWE=1 أن تكتب المعطيات على الذاكرة EEPROM عند العنوان المحدد في المسجل EEAR . أما إذا صفرت الخانة القائدة للكتابة EEMWE=0 ، فإن خانة تمكين الكتابة EEWE تصبح عديمة التأثير . وعند تفعيل الخانة EEMWE برمجياً ، فإن الكيان الداخلي للمتحكم MCU يقوم بتصفير هذه الخانة بعد أربع دورات ساعة . أنظر شرح الخانة EEWE عند إجراء الكتابة على ذاكرة المعطيات EEPROM .

الخانة 1 – EEWE : خانة تمكين الكتابة على ذاكرة المعطيات EEPROM :

تقدح إشارة تمكين الكتابة (EEWE) عملية الكتابة على ذاكرة المعطيات EEPROM عند تفعيلها EEWE=1 فعند الكتابة على الذاكرة EEPROM يجب علينا أولاً تحميل المعطيات والعنوان بشكل صحيح للمسجلين EEDR و EEAR ، ومن ثم نفعل الخانة EEWRE=1 . كما يجب أن تكون الخانة EEMWE مُفعّلة عند كتابة المنطق واحد على الخانة EEWE ، وفي كل الحالات الأخرى لن تحدث عملية الكتابة على ذاكرة المعطيات EEPROM . يجب إتباع الإجراء التالي عند الكتابة على الذاكرة EEPROM (وقد تكون الخطوتين ٢،٣ غير ضروريتين) :

١- أنتظر حتى تصبح الخانة EEWE = 0 .

٢- اكتب عنوان حجرة الذاكرة EEPROM في مسجل العنوان EEAR (اختياري) .

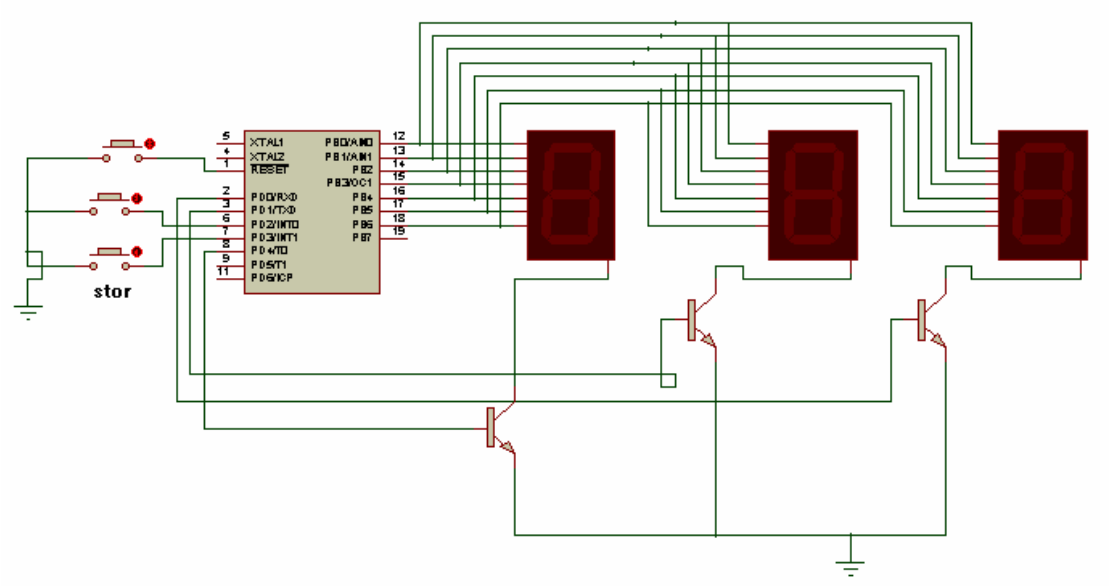
٣- اكتب معطيات الذاكرة EEPROM في مسجل المعطيات EEDR (اختياري) .

٤- اكتب المنطق واحد "١" على الخانة القائدة EEMWE الموجودة في المسجل EECR .

٥- خلال دورات الساعة الأربعة بعد تفعيل الخانة EEMWE اكتب المنطق واحد "١" على خانة تمكين الكتابة EEWE=1 .

الفارس لتقنيات الحاسوب والإتصالات
سوريا – حماه- هـ-229619 33 963+

نقوم في هذا المشروع بعرض الرقم i وزياته وتخزينه بواسطة المقاطعتين الخارجيتين int 0 ، int 1 علـــى الترتيب .



دارة المشروع

البرنامج :

```
#include <90s2313.h>
#include <delay.h>
eeprom int alfa=0;          // write alfa to eeprom

int i;                                  // global variable
                                        // External Interrupt 0 service routine
interrupt [EXT_INT0] void ext_int0_isr(void)
{
i++;
}
interrupt [EXT_INT1] void ext_int1_isr(void)
{
alfa=i;                     // store value of number i in EEPROM
}

void main(void)
{

unsigned char decode[10]={0x3f,6,0x5B,0x4f,0x66,0x6d, 0x7c,0x07,0x7f,0x6f};
unsigned char a,b,c;
PORTB=0x00;
DDRB=0xFF;
```

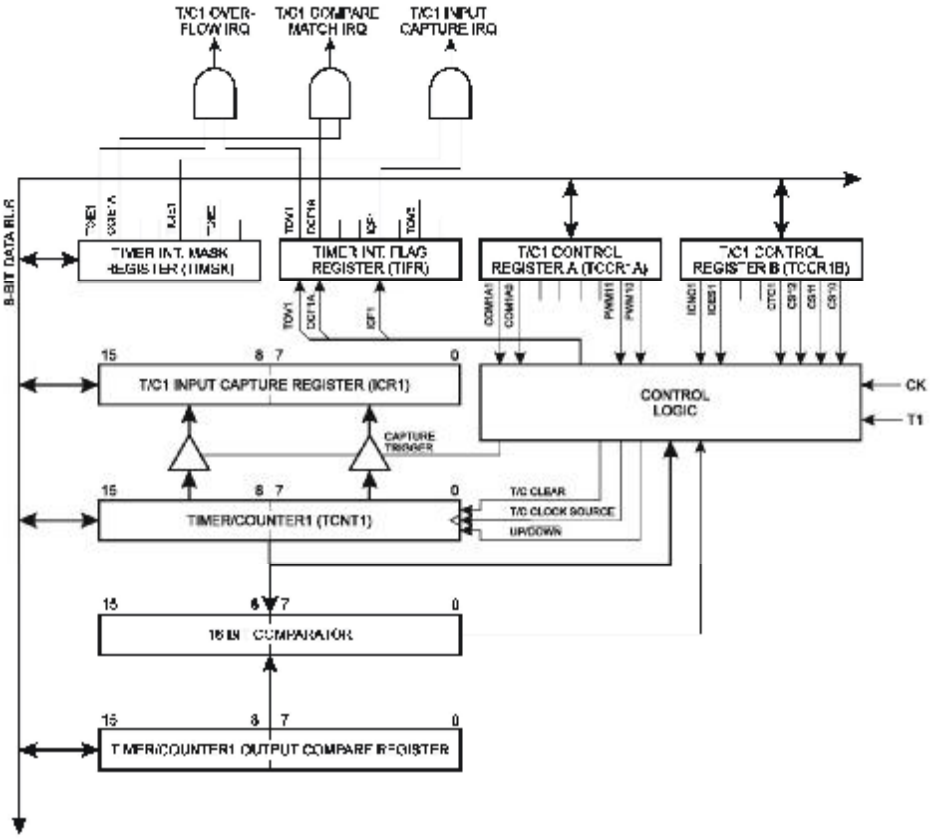

الفارس لتقنيات الحاسوب والإتصالات
سوريا – حماه- هـ-229619 33 963+

```
PORTD=0xfc;
DDRD=0x13;

                                                  // External Interrupt(s) initialization
                                                 // INT0: On
                                                // INT0 Mode: Falling Edge
                                                // INT1: on
                                                // INT1 Mode: Falling Edge
GIMSK=0xC0;
MCUCR=0x0A;
GIFR=0xC0;
#asm("sei")
   i=alfa;
 while(1)
   {
    a=i%10;
    PORTD.4=0;
    delay_ms(100);
    PORTD.0=1;
    PORTB=decode[a];

    b=i/10;
    c=b%10;
    delay_ms(100);
    PORTD.0=0;
    PORTB=decode[c];
    PORTD.1=1;

    b=i/100;
    delay_ms(100);
    PORTD.1=0;
    PORTB=decode[b];
    PORTD.4=1;
    delay_ms(100);

   };                              // end while

}                                 // end main
```

المشروع السادس
الساعة الإلكترونية

المؤقت/العداد ١ ذو ١٦ bit - 6 **The 16–Bit Timer/Counter 1**

يبين الشكل (31) المخطط الصندوقي للمؤقت/العداد ١



شكل (31) : المخطط الصندوقي للمؤقت/العداد١

يمكن اختيار مصدر ساعة المؤقت/العداد١ إما من هزاز الساعة CK و إما من تقسيمات هزاز الساعة CK و إما من القطب الخارجي (العمل كعداد) . وكل هذه الحالات مشروحة في مسجل التحكم بالمؤقت /العداد TCCR1A . تتوضع أعلام الحالة المختلفة (علم الطفحان و علم مقارنة النظير وعلم حادثة المسك) مع إشارات التحكم في مسجل أعلام مقاطعة المؤقت/العداد - TIFR . و تتوضع خانة تمكين/حجب مقاطعة المؤقت/العداد١ في مسجل قناع مقاطعة المؤقت/العداد TIMSK .

عندما نطبق إشارة ساعة خارجية على القطب الخارجي للمؤقت/العداد١ – T1(PD5) ، فإنه على هذه الإشارة أن تتزامن مع تردد هزاز وحدة المعالجة المركزية CPU . ولضمان ملائمة العينات sampling الساعة

الخارجية ، فيجب أن يكون الزمن الأصغري لكل انتقالين من انتقالات الساعة الخارجية أقل من زمن دورة داخلية لساعة وحدة المعالجة المركزية CPU .

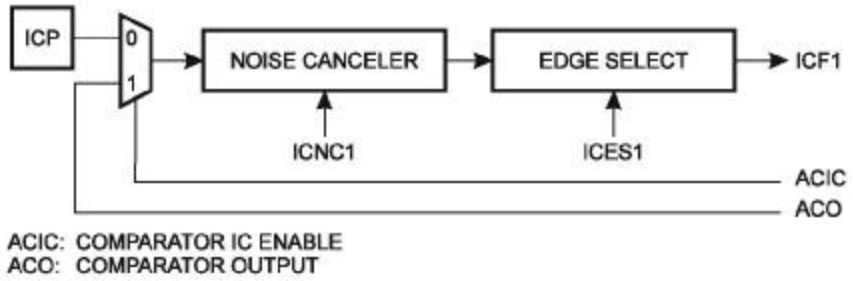
زود المؤقت/العداد١ بوظيفة مقارنة الخرج التي تستخدم مسجل مقارنة الخرج OCR1A كمسجل مصدر للمعطيات التي سوف تقارن مع المحتويات الآنية لمسجل المؤقت/العداد TCNT1 . تتضمن وظيفة مقارنة الخرج خيار تصفير العداد عند تحقق المساواة بين مسجل مقارنة الخرج ومسجل المؤقت/العداد ،كما أن قطب مقارنة الخرج OC1 يُفعّل عند تحقق هذه المساواة .

يستخدم المؤقت/العداد١ أيضاً كمُعدل عرض نبضة PWM بثمان أو تسع أو عشر خانات . في هذا النمط يوظّف كلٍ من العداد والمسجل OCR1 لتوليد نبضات PWM مركزية مع بعض النبضات الخاطئة glitchالناتجة عن عمليات الكتابة الغير متزامنة على المسجل OCR1 . ولمزيد من المعلومات حول هذه الوظيفة راجع فقرة " المؤقت/العداد١ في نمط معدل عرض النبضة PWM " .

تمنح وظيفة مدخل المسك للمؤقت/العداد ١ إمكانية مسك محتويات المؤقت/العداد ١ وتخزينه في مسجل مدخل المسك – ICR1 ، وذلك عندما تقدح الحادثة الخارجية قطب مدخل المسك (ICP) . وقد تم تعريف حادثة المسك الفعلية في مسجل التحكم بالمؤقت/العداد١ – TCCR1 .

يبين الشكل (32) المخطط الصندوقي لرفض ضجيج قطب مدخل المسك للمؤقت/العداد١ .

إذا تم تمكين وظيفة رفض الضجيج noise ، فإن ملاحقة شرط القدح الفعلي لحادثة المسك capture تتم بأربع عينات sample 4 من نبضات الساعة وذلك قبل تفعيل المسك . بحيث تؤخذ العينات من إشارة قطب الدخل عند كل نبضة من نبضات الساعة أي عند كل XTAL .



الشكل (32) : رفض ضجيج قطب مدخل المسك

مسجل التحكم A بالمؤقت/العداد ١ – TCCR1A

Bit	7	6	5	4	3	2	1	0	
$2F ($4F)	COM1A1	COM1A0	–	–	–	–	PWM11	PWM10	TCCR1A
Read/Write	R/W	R/W	R	R	R	R	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

The Timer/Counter 1 Control Register A

الخانتان 7, 6 - COM1A1 ، COM1A0 : خانتا خرج المقارنة :

إن كلا خانتي التحكم COM1A1،COM1A0 تحدد ماذا سيحدث على قطب الخرج OC1 ، وذلك عند تحقق المساواة بين مسجل مقارنة الدخل OCR1 ومسجل المؤقت/العداد١ . وهذه الوظيفة هي وظيفة إضافية لعمل القطب PB3(OC1) كقطب دخل/خرج . ويجب علينا تفعيل خانة التحكم باتجاه المعطيات المطابقة لهذا

القطب لإعداده كقطب خرج و أي يجب تفعيل الخانة DDB3=1 الموجودة في مسجل اتجاه المعطيات DDRB .بين الجدول (7) التالي أنماط التحكم لهاتين الخانتين :

COM1A0	COM1A1	الشرح
0	0	المؤقت/العداد١ لا يتصل مع قطب الخرج OC1
1	0	عقدة المنتصف لقطب الخرج OC1
0	1	تصفير قطب الخرج OC1 (للصفر)
1	1	تفعيل قطب الخرج OC1 (للواحد)

الجدول (7) : اختيار نمط المقارنة ١

أما عند عمل المؤقت/العداد١ في نمط PWM ، فإن لهاتين الخانتين وظيفة أخرى مختلفة ، وقد شرحت هذه الوظيفة بالتفصيل في الجدول (11) .

عندما نرغب في تغيير قيم الخانتين COM1A0،COM1A1 فإنه يتوجب علينا أولاً حجب مقاطعة مقارنة الخرج A١ ، وذلك بتصفير خانة تمكين المقاطعة OCIE1=0 الموجودة في المسجل TIMSK . وقد تحدث المقاطعة في كل الحالات الأخرى .

الخانات Res – 5 ... 2 : خانات محجوزة :

هذه الخانات هي خانات محجوزة في المتحكم AT90S2313 وتقرأ دائماً صفر منطقي .

الخانتان PWM11، PWM10 – 1، 0 : خانتي اختيار معدل عرض النبضة :

تحدد هاتان الخانتان عملية اختيار PWM للمؤقت/العداد ١ كما هو مبين في الجدول (8) . وقد شرح هذا النمط في فقرة " المقت/العداد١ في نمط PWM ".

PWM10	PWM11	الشرح
0	0	حجب عمل PWM للمؤقت/العداد١
1	0	معدل عرض نبضة PWM بثمان خانات
0	1	معدل عرض نبضة PWM بتسع خانات
1	1	معدل عرض نبضة PWM بعشر خانات

الجدول(8) : اختيار نمط PWM

مسجل التحكم B بالمؤقت/العداد ١ - TCCR1B

The Timer/Counter1 Control Register B – TCC1B

Bit	7	6	5	4	3	2	1	0	
$2E ($4E)	ICNC1	ICES1	–	–	CTC1	CS12	CS11	CS10	TCCR1B
Read/Write	R/W	R/W	R	R	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

الخانة ICNC1 – 7 : خانة رفض ضجيج مدخل المسك ١ (4CKs) :

تحجب وظيفة رفض الضجيج noise لحادثة قدح مدخل المسك capture عند تصفير الخانة ICNC1=0 ، عندئذ يمكن قدح وظيفة مدخل المسك عند أول جبهة صاعدة /هابطة مطبقة على قطب مدخل المسك ICP وذلك حسب قيمة الخانة ICES1 . أما عندما نفعل الخانة ICNC1=1 ، فإن ذلك يؤدي إلى تفعيل وظيفة رفض ضجيج مدخل المسك ، فيتم مراقبة الإشارة المطبقة على مدخل المسك ICP وانتظار تتابع أربع عينات ذات

مستوى منطقي عالي أو منخفض وذلك حسب قيمة الخانة ICES1 حتى تقبل الإشارة القادحة لمدخل المسك . إن تردد أخذ العينات الفعلي يتزامن مع تردد الساعة XTAL.

الخانة 6 – ICES1 :خانة اختيار جبهة مدخل المسك ١ :

عندما نُصفر الخانة ICES1=0 ، فإن محتويات المؤقت/العداد١ تحول إلى مسجل مدخل المسك ICR1 عند الجبهة الهابطة للإشارة المطبقة على مدخل المسك ICP . بينما عندما نُفعّل الخانة ICES1=1 فإن محتويات المؤقت/العداد١ تحول إلى مسجل مدخل المسك عند الجبهة الصاعدة للإشارة المطبقة على مدخل المسك ICP .

الخانتان 5 – 4،Res : خانتان محجوزتان :

هاتان الخانتان هما خانتان محجوزتان في التحكم AT90S2313 وتقرأان دائماً صفر منطقي .

الخانة 3 - CTC1 : تصفير المؤقت/العداد١ عند تحقق مساواة مقارنة النظير:

عند تفعيل الخانة CTC1=1 فإن المؤقت/العداد١ يُصفر (أي تصبح محتوياته $0000) في دورة الساعة بعد تحقق المساواة بين محتويات مسجل المؤقت/العداد TCNT1 و محتويات مسجل مقارنة الخرج OCR1 . أما في حالة تصفر الخانة CTC1=0 فإن المؤقت/العداد١ يتابع العد حتى يتوقف ، أو يصفر ، أو يطفح ، أو يغير اتجاهه. وليس لهذه الخانة أي تأثير على نمط PWM .

الخانات CS12، CS11 ، CS10 ، 2 - ، 1 ،0 : خانات اختيار الساعة للمؤقت/العداد ١ :

إن الخانات CS12، CS11 ،CS10 تُعرّف نسبة تقسيم مصدر نبضات ساعة المؤقت/العداد١ . كما هو مبين في الجدول التالي :

CS10	CS11	CS12	الشرح
0	0	0	نمط توقف المؤقت/العداد ١
1	0	0	CK
0	1	0	CK/8
1	1	0	CK/64
0	0	1	CK/256
1	0	1	CK/1024
0	1	1	عد الحوادث الخارجية المطبقة على القطب T1 عند الجبهات الهابطة
1	1	1	عد الحوادث الخارجية المطبقة على القطب T1 عند الجبهات الصاعدة

جدول (9) : اختيار نسبة تقسيم نبضات الساعة للمؤقت١.

يقدم لنا شرط التوقف stop condition وظيفة تمكين/حجب المؤقت . إن أنماط تقسيمات الساعة الأقل من CK تعاير مباشرة من ساعة الهزاز CK ، وعند استخدام المؤقت/العداد١ كعداد للحوادث الخارجية المطبقة على قطبه T1(PD5) ، فيجب علينا تحقيق التهيئة المطابقة لهذا القطب في مسجل التحكم باتجاه المعطيات الفعلي actual (عند تصفير العلم المطابق يصبح القطب قطب دخل)

مسجل قناع مقاطعة المؤقت / العداد TIMSK

Timer/Counter Interrupt Mask Register – TIMSK

Bit	7	6	5	4	3	2	1	0	
$39 ($59)	TOIE1	OCIE1A	–	–	TICIE1	–	TOIE0	–	TIMSK
Read/Write	R/W	R/W	R	R	R/W	R	R/W	R	
Initial value	0	0	0	0	0	0	0	0	

الخانة 7 – TOIE1 : خانة تمكين مقاطعة طفحان المؤقت/العداد ١ :

يتم تفعيل مقاطعة المؤقت/العداد١ عند تفعيل كلٍ من الخانتين TOIE1=1 وخانة تمكين المقاطعة العامة I=1 الموجودة في مسجل الحالة SREG . ويتم الانتقال إلى روتين خدمة مقاطعة المؤقت/العداد١ الواقع عند الشعاع $005 في ذاكرة البرنامج الوميضية Flash عند طفحان مسجل المؤقت/العداد١. ويفعل عندها أيضاً علم طفحان المؤقت/العداد١ أي TOV1=1 الواقع في مسجل أعلام مقاطعة المؤقت/العداد TIFR . و عند استخدام المؤقت/العداد١ في نمط PWM ، فإن علم طفحان المؤقت يُفعل عندما يغير العداد اتجاه العد عند القيمة $0000

الخانة 6 – OCIE1A : خانة تمكين مقاطعة خرج مقارنة النظير للمؤقت/العداد ١ :

يتم تفعيل مقاطعة خرج مقارنة النظير للمؤقت/العداد ١ عند تفعيل كلٍ من الخانتين OCIE1=1 وخانة تمكين المقاطعة العامة I=1 الموجودة في مسجل الحالة SREG . و ينفذ المتحكم روتين خدمة مقاطعة خرج مقارنة النظير للمؤقت/العداد ١ الواقع عند الشعاع $004 في ذاكرة البرنامج الوميضية Flash وذلك عندما تتساوى القيمة الموجودة في مسجل مقارنة الخرج OCR1AH/OCR1AL مع القيمة الآنية لمسجل المؤقت/العداد TCNT1 ، فيصبح علم خرج المقارنة للمؤقت/العداد١ -OCF1A=1 الواقع في مسجل أعلام مقاطعة المؤقت/العداد١.

الخانات 5 – Res،4 : خانات محجوزة :

هذه الخانات هي خانات محجوزة في المتحكم AT90S2313 وتقرأ دائماً صفر منطقي .

الخانة 3 – TICIE1 : علم مقاطعة مدخل المسك للمؤقت/العداد ١ :

تُفعّل مقاطعة حادثة المسك للمؤقت/العداد١ عند تفعيل كلٍ من الخانتين TICE1=1 وخانة تمكين المقاطعة العامة I=1 الموجودة في مسجل الحالة SREG . وينفذ المتحكم MCU شعاع المقاطعة الواقع عند العنوان $003 في ذاكرة المتحكم الوميضية إذا حدثت حادثة المسك عند قطب المتحكم ١١ ، (ICP) PD6 . فيصبح علم مدخل المسك للمؤقت/العداد١- ICF1=1 الواقع في مسجل أعلام المؤقت/العداد TIFR .

الخانات 2 – Res : خانة محجوزة :

هذه الخانة هي خانة محجوزة في المتحكم AT90S2313 وتقرأ دائماً صفر منطقي .

الخانة 0 – TOIE0 : خانة تمكين مقاطعة طفحان المؤقت/العداد 0 :

يتم تفعيل مقاطعة المؤقت/العداد٠ عند تفعيل كلٍ من الخانتين TOIE0=1 وخانة تمكين المقاطعة العامة I=1 الموجودة في مسجل الحالة SREG . ويتم الانتقال إلى روتين خدمة مقاطعة المؤقت/العداد٠ الواقع عند الشعاع $006 في ذاكرة البرنامج الوميضية Flash عند طفحان مسجل المؤقت/العداد٠. ويفعل عندها أيضاً علم طفحان المؤقت/العداد ٠ أي TOV0=1 الواقع في مسجل أعلام مقاطعة المؤقت/العداد TIFR .

الخانات 0 – Res : خانة محجوزة :

هذه الخانة هي خانة محجوزة في المتحكم AT90S2313 وتقرأ دائماً صفر منطقي .

مسجل أعلام مقاطعة المؤقت/العداد TIFR

Bit	7	6	5	4	3	2	1	0	
$38 ($58)	TOV1	OCF1A	–	–	ICF1	–	TOV0	–	TIFR
Read/Write	R/W	R/W	R	R	R/W	R	R/W	R	
Initial value	0	0	0	0	0	0	0	0	

Timer/Counter Interrupt Flag Register – TIFR

الخانة 7 – TOV1 : علم طفحان المؤقت/العداد ١ :

يُفعّل علم طفحان المؤقت/العداد١ أي TOV1=1 عندما يحدث طفحان في مسجل المؤقت/العداد١- TCNT1 . و يصفر هذا العلم TOV1=0 من الكيان Hardware الداخلي للمتحكم عند تنفيذ روتين خدمة مقاطعة طفحان المؤقت/العداد١ . وبشكل مشابه يمكن تصفير هذا العلم بكتابة المنطق واحد عليه ، فعندها ينفذ أيضاً روتين خدمة مقاطعة طفحان المؤقت/العداد١ . أما عند استخدام المؤقت ١ في نمط PWM فإن هذا العلم يُفعّل TOV1=1 عندما يغير المؤقت/العداد١ اتجاه عده عند القيمة $0000 .

الخانة 6 – OCF1A : علم خرج المقارنة 1A :

يُفعّل العلم OCF1A=1 عندما تحدث المساواة بين القيمة الموجودة في مسجل المؤقت/العداد١ - TCNT1 و المعطيات الموجودة في مسجل مقارنة الخرج OCR1A – 1A . و يتم تصفير هذا العلم OCR1A=0 من الكيان Hardware الداخلي للمتحكم عند تنفيذ روتين خدمة المقاطعة المطابق . وبشكل مشابه يمكن تصفير هذا العلم OCF1A=0 بكتابة المنطق واحد عليه . و نذكر هنا أن مقاطعة مقارنة نظير المؤقت/العداد١ تُؤهل عند تفعيل كلٍ من الخانات : خانة تمكين المقاطعة العامة I=1 الواقعة في مسجل الحالة SREG وخانة تمكين مقاطعة خرج مقارنة نظير المؤقت/العداد ١ - OCIE1A=1 الواقعة في مسجل قناع مقاطعة المؤقت/العداد TIMSK وعلم خرج المقارنة1A - OCF1A=1 .

الخانات 5 – Res،4 : خانتان محجوزتان :

هذه الخانات هي خانات محجوزة في المتحكم AT90S2313 وتقرأ دائماً صفر منطقي .

الخانة 3 – ICF1 : علم مدخل الدخل ١ :

تُفعّل الخانة ICF1=1 عند تطبيق إشارة قادحة على مدخل المسك (القطب 11(PD6))، وتشير إلى أن قيمة مسجل المؤقت/العداد١ – TCNT1 قد انتقلت إلى مسجل مدخل المسك – ICR1 . وعندما تنفذ وحدة المعالجة المركزية CPU روتين خدمة المقاطعة المطابق ، فان الكيان الداخلي للمتحكم MCU يُصفر هذا العلم ICF1=0. وبشكل مشابه يمكن تصفير هذا العلم بكتابة المنطق واحد عليه .

الخانات 2 – Res : خانة محجوزة :

هذه الخانة هي خانة محجوزة في المتحكم AT90S2313 وتقرأ دائماً صفر منطقي .

الخانة - TOV0 : علم طفحان المؤقت /العداد٠ :

يُفعّل علم طفحان المؤقت/العداد0 أي TOV0=1 عندما يحدث طفحان في مسجل المؤقت/العداد٠ - TCNT0. و يُصفر هذا العلم TOV0=0 من الكيان Hardware الداخلي للمتحكم عند تنفيذ روتين خدمة مقاطعة طفحان

المؤقت/العداد0 . وبشكل مشابه يمكن تصفير هذا العلم بكتابة المنطق واحد عليه ، يتم الانتقال إلى تنفيذ روتين خدمة مقاطعة المؤقت/العداد٠ الواقع عند العنوان 006$ في ذاكرة البرنامج الوميضية Flash فقط إذا كانت الخانات الثلاث التالية مُفعّلة : خانة تمكين المقاطعة العامة I=1 الواقعة في مسجل الحالة SREG وخانة تمكين مقاطعة طفحان المؤقت/العداد0 - TOIE0=0 الواقعة في مسجل قناع مقاطعة المؤقت/العداد TIMSK وعلم الطفحان TOV0=1 الواقعة في هذا المسجل TIFR .

الخانات 0 – Res : خانة محجوزة :

هذه الخانة هي خانة محجوزة في المتحكم AT90S2313 وتقرأ دائماً صفر منطقي .

مسجل المؤقت/العداد١ – TCNT1H و TCNT1L

The Timer/Counter1 – TCNT1H and TCNT1L

Bit	15	14	13	12	11	10	9	8	
$2D ($4D)	MSB								TCNT1H
$2C ($4C)								LSB	TCNT1L
	7	6	5	4	3	2	1	0	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

يحتوي هذا المسجل الذي طوله ١6-bit على القيمة الآنية التي وصل إليها المؤقت/العداد١ الذي طوله أيضاً ١6-bit . وحتى نضمن عملية القراءة من أو الكتابة على كلا البايتين العلوي والسفلي في نفس اللحظة الزمنية عند ولوج وحدة المعالجة المركزية CPU لهذه المسجلات . فإنه يتوجب عليتا إنجاز الولوج باستخدام المسجل اللحظي TEMP .

الكتابة في مسجل المؤقت /العداد ١ – TCNT1 TCNT1 Timer/Counter1 Write

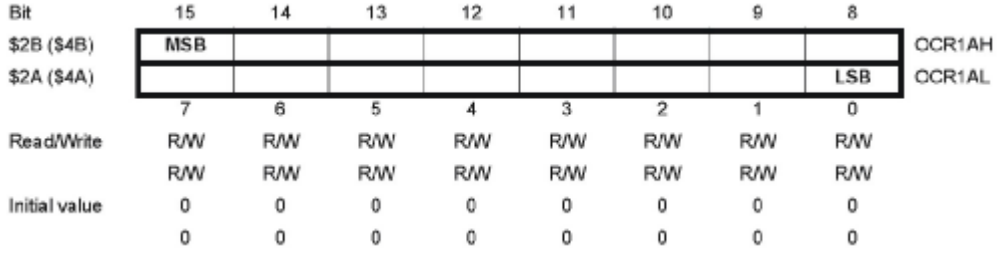
عندما تكتب وحدة المعالجة المركزية CPU البايت العالي على المسجل TCNT1H ، فإن المعطيات المكتوبة تتوضع في المسجل TEMP ، ومن ثم عندما تكتب وحدة المعالجة المركزية CPU البايت السفلي على المسجل TCNT1L ، فإن هذا البايت يتحد مع بايت المعطيات في المسجل TEMP ، وتكتب كل الخانات الست عشرة على مسجل المؤقت /العداد١ .ونستنتج من ذلك ، أنه يجب الولوج للبايت العلوي TCNT1H أولاً عند عملية الكتابة على كامل المسجل ١Bit – 6 .

القراءة من مسجل المؤقت/العداد١ – TCNT1 TCNT1 Timer/Counter1 Read

عندما تقرأ وحدة المعالجة المركزية CPU البايت السفلي من المسجل TCNT1L ، فإن معطيات البايت السفلي TCNT1L ترسل إلى وحدة المعالجة المركزية CPU و تتوضع معطيات البايت العلوي TCNT1H في المسجل TEMP . و عندما تقرأ وحدة المعالجة المركزية CPU المعطيات الموجودة في البايت العلوي TCNT1H فإن وحدة المعالجة المركزية CPU تستقبل المعطيات الموجودة في المسجل TEMP. ونستنتج من ذلك ، أنه يجب الولوج للبايت السفلي TCNT1L أولاً عند عملية القراءة من كامل المسجل ١Bit – 6 .

لقد أنجز المؤقت/العداد١ كعداد صاعد/هابط (في نمط PWM) مع إمكانية الولوج له عند القراءة والكتابة .إذا تمت عملية الكتابة على المؤقت/العداد١ وكان مصدر الساعة قد حدد له ، فإن المؤقت /العداد١ يتابع عمله في دورة ساعة المؤقت بعد شحنه preset بالقيمة المكتوبة

مسجل مقارنة الخرج A للمؤقت /العداد١ – OCR1AH , OCR1AL



Timer/Counter1 Output Compare Register A – OCR1AH and OCR1AL

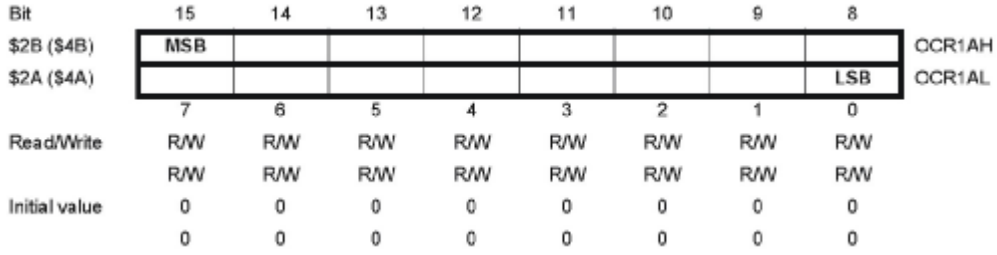
مسجل مقارنة الخرج هو عبارة عن مسجل ذو ١Bit – 6 يمكن القراءة منه أو الكتابة عليه .

تقارن محتويات مسجل مقارنة الخرج للمؤقت/العداد١ بشكل مستمر مع محتويات مسجل العداد/المؤقتTCNT1. تحدد الأفعال عند مقارنة النظير في مسجل الحالة والتحكم بالمؤقت/العداد TCCR1A

إن مسجل مقارنة الخرج OCR1A هو عبارة عن مسجل ذو ١Bit–6 ، يتم استخدام المسجل اللحظي TEM – temporary عند الكتابة على المسجل OCR1A كي نضمن أن كلا البايتين قد حُدّثا updated في نفس اللحظة الزمنية . عندما تكتب وحدة المعالجة المركزية CPU البايت العالي في المسجل OCR1AH ، فإن المعطيات تخزن في المسجل اللحظي TEMP . و عندما تكتب وحدة المعالجة المركزية CPU البايت السفلي في المسجل OCR1AL،فإن محتويات المسجلTEMP تُكتب في نفس اللحظة الزمنية في المسجلOCR1AH. فنستنتج من ذلك ،أنه يجب كتابة البايت العالي OCR1AH أولاً عند عملية الكتابة على كامل المسجل١Bit –6.

مسجل مدخل المسك للمؤقت /العداد١ – ICR1H , ICR1L

The Timer/Counter1 Input Capture Register – ICR1H and ICR1L



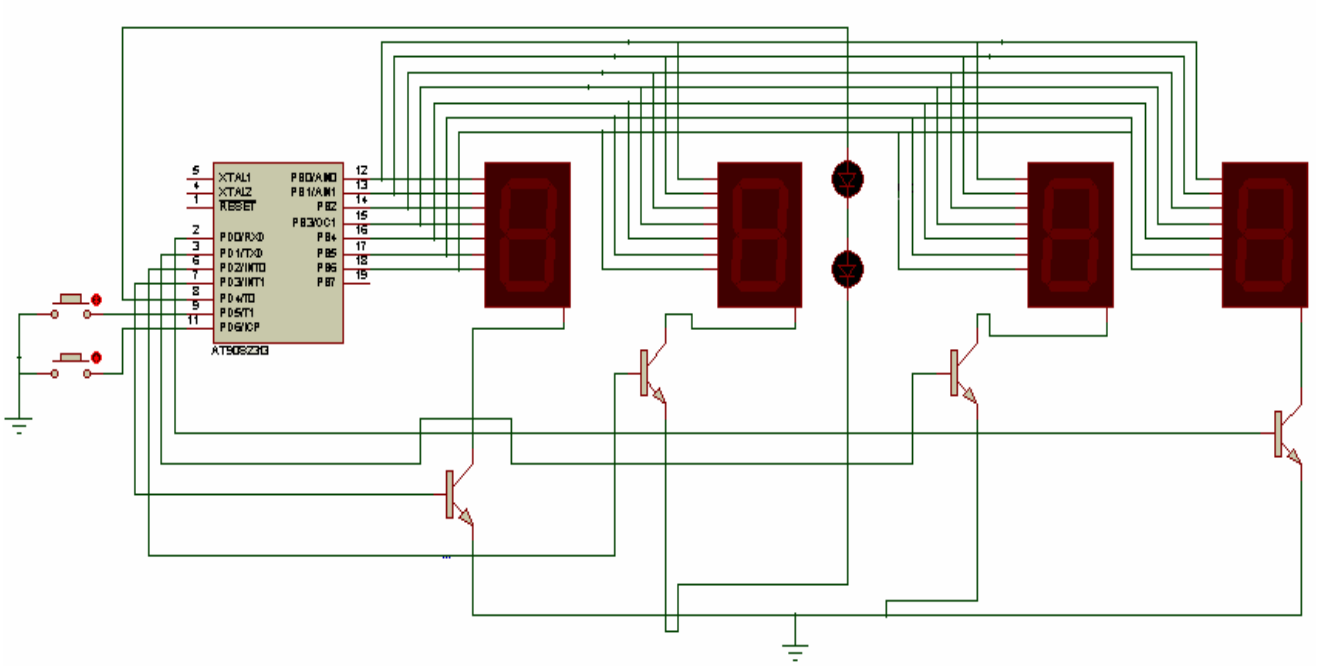
مسجل مدخل المسك هو عبارة عن مسجل ذو ١bit - 6 قابل للقراءة فقط .

عندما اكتشاف إشارة ذات جبهة صاعدة أو هابطة (بما يتطابق مع حالة خانة جبهة مدخل المسك – ICES1) على قطب مدخل المسك – ICP ، فإن القيمة الحالية للمؤقت/العداد١ تحول إلى مسجل مدخل المسك – ICR1 . وفي نفس اللحظة الزمنية يُفعّل علم مدخل المسك ICF1=1 . إن مسجل مدخل المسك – ICR1 هو عبارة عن مسجل ذو١6-bit . ويتم استخدام المسجل اللحظي TEMP عند قراءة المسجل ICR1 كي نضمن أن كلا البايتين قد تم قراءتهم في نفس اللحظة الزمنية . فعندما تقرأ وحدة المعالجة المركزية CPU البايت المنخفض من المسجل ICR1L ، فإن المعطيات ترسل إلى وحدة المعالجة المركزية CPU و تتوضع معطيات البايت العالي للمسجل ICR1H في المسجل اللحظي TEMP . وعندما تقرأ وحدة المعالجة المركزية CPU المعطيات في البايت العاليICR1H ، فإن وحدة المعالجة المركزية CPU تستقبل المعطيات الموجودة في

المسجل اللحظي TEMP . ونستنتج من ذلك، أنه يجب الولوج للبايت المنخفض ICR1L أولاً عند عملية القراءة لكامل المسجل ١6-bit .

§ **نحتاج في مشروعنا أن نزيد عداد الثواني مرة عند مرور كل ثانية من الوقت لذا يجب تعيير المؤقت على قيمة ثانية ويتم ذلك باستخدام مقاطعة المقارنة، في البداية تم اختيار مقسم التردد على نسبة 64 و تردد الساعة الذي نستخدمه هو 4000000 Hz أي كل 4MHz يمر من الزمن 1 ثانية فإذا ماتم اختيار نسبة التقسيم 64 يصبح عدد حالات التي يجب أن نزيد فيها الثواني كالتالي : 4000000/64= 62500 أي إذا قارنا قيمةالمؤقت مع القيمة 62500 عند المساواة نعلم أنه قد مر من الوقت 1ثانية فإذا وضعنا القيمة 62500=0xF424 في مسجلي مقارنة النظير وباستخدام مقاطعة مقارنة النظير عندها نقوم بزيادة عداد الثواني ونقوم بنفس الوقت بتصفير مسجل العداد مؤقت لعد قيمة جديدة وثانية جديدة .**

الدارة التالية تظهر لنا كيفية وصل شاشات الـــSeven Segment مع المتحكم ونلاحظ وجود كباسين يستخدمان من أجل زيادة الدقائق وزيادة الساعات .



البرنامج :

```
/*****************************************************
Project : Clock
Chip type           : AT90S2313
Clock frequency     : 4.000000 MHz
*****************************************************/

#include <90s2313.h>
#include <delay.h>
unsigned char s=59,m=19,h=12;
```

```
                // Timer 1 output compare interrupt service routine
interrupt [TIM1_COMP] void timer1_comp_isr(void)
{
PORTD.4=!PORTD.4;
if (s==59)
 {
  s=0;
  if(m==59)
    {
     m=0;
     if(h==12)
     h=0;

     h++;
     }
  else
     m++;
}
else
s++;
}

void main(void)
{
 unsigned char decode[10]={0x3f,6,0x5B,0x4f,0x66,0x6d,0x7c,0x07,0x7f,0x6f};
unsigned char a,b,c;
bit q=0;

PORTB=0x00;
DDRB=0xFF;

PORTD=0x60;
DDRD=0x1f;

                        // Timer/Counter 1 initialization
                        // Clock source: System Clock
                        // Clock value: 4000.000 kHz
                       // Compare Match Interrupt: On
TCCR1A=0x00;
TCCR1B=0x0b;   //    اختيار نسبة التقسيم٦٤وتصفير المؤقت عن مساواة المقارنة
OCR1H=0xF4;    //  القيمة التي يجب أن تقارن مع قيمة العداد
OCR1L=0x24;
                       // Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x40;                 //تفعيل مقاطعة مقارنة النظير
                      // Global enable interrupts
#asm ("sei")
 while (1)
  {
```

```
 if(PIND.5==0 && q==0 )
 {
 if (m==59)
 {
  m=0;
  }
   else
m++;
q=1;
}

if(PIND.6==0 && q==0 )
{
  if (h==12)
    {
     h=0;
     }
h++;
q=1;
}
if (PIND.5==1 && PIND.6==1)
 {
 q=0;
  }
  PORTD.3=0;
 delay_ms(10);
 a=m%10;
 PORTB=decode[a];
 PORTD.0=1;

 b=m/10;
 c=b%10;
 delay_ms(10);
 PORTD.0=0;
 PORTB=decode[c];
 PORTD.1=1;

 delay_ms(10);
 PORTD.1=0;
 a=h%10;
 PORTB=decode[a];
 PORTD.2=1;
 delay_ms(10);

 delay_ms(10);
 PORTD.2=0;
 b=h/10;
 c=b%10;
 PORTB=decode[c];
```

```
    PORTD.3=1;
    delay_ms(10);

   };              // end while
}              // end main
```

المشروع السابع
ساعة إلكترونية باستخدام شاشات الـ LCD

كيف يتم التعامل مع شاشة الـــ LCD :

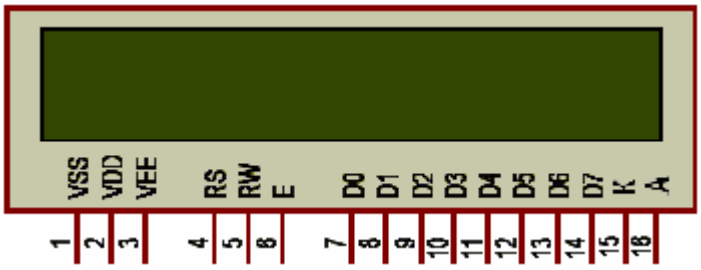
شاشة الـــ (LCD) هي عبارة عن شاشة مؤلفة من سطر أو أكثر يحتوي كل سطر على عدد من الخانات ، والخانة هي عبارة عن مربع صغير يتم إظهار المحرف عليه ، أي أن كل خانة تستطيع إظهار محرف واحد فقط ، وأكثر الشاشات شيوعاً هي الشاشات ذات القياسات التالية :

١٦×١ ، ٢٠×١ ، ٢٤×١ ، ١٦×٢ ، ٢٠×٢ ، ١٦×٤ ، ٢٠×٤

تملك شاشة الإظهار الكريستالية معالج إظهار خاص بها، بحيث توفر على المستثمر القيام بعمليات عديدة شاقة و معقدة ، حيث تزود شاشة الـــــ LCD بذاكرة داخلية خاصة تقسم بدورها إلى قسمين : ذاكرة المعطيات DD-RAM و ذاكرة مولد الرمز CG-RAM ، و تقوم هذه الذواكر بالاحتفاظ بالرموز المراد إظهارها وتُمكن المبرمج من إعادة إظهارها مرة أخرى بدون الحاجة إلى إرسالها مرة أخرى ، ومن الممكن إظهار جميع رموز الأسكي عليها (189) رمز مختلف .

تملك شاشات الـــ LCD **بشكل عام** نفس أقطاب التحكم ولكن تختلف باختلاف ما تم ذكره سابقاً .

ويبين الشكل التالي أقطاب التحكم بشاشة الـــ LCD ذات السطرين و الـــ ١٦ عمود :



والجدول التالي يوضح ماهية هذه الأقطاب ووظيفتها :

Pin No.	Symbol	Name	Description
1	VSS	Power Supply	0V (GND)
2	VDD	Power Supply	Power supply for logic circuit and LCD (+4.5V ~ +5.5V)
3	VEE	LCD Supply Voltage	Bias voltage level for LCD driving
4	RS	Register Select	Register select input. , Data register is High" When RS =" selected. , Instruction register is " Low When RS =" selected.
5	R/W	Read/Write	Read/Write selection input. , Read operation. " High When RW = " , Write operation. " Low When RW = "
6	E	Read Write Enable	Start enable signal to read or write the data
7 8 9 10	DB0 DB1 DB2 DB3	Data Bus 0-7	DB0 - DB3 , In 8-bit bus mode, used as low order bi-directional data bus. During 4- , Open these pins bit bus mode
11 12 13 14	DB4 DB5 DB6 DB7		DB4 - DB7, In 8-bit bus mode, used as . In case high order bi-directional data bus used as both high and , of 4-bit bus mode low order. DB7 used for Busy Flag output
15	K		light cathode Back-
16	A		light anode Back-

Interface Pin Connections

أقطاب شاشة الــ LCD :

القطب V_{SS} :

هو قطب التغذية لشاشة الــ LCD ، وهو جهد الأرضي (٠) منطقي (GND) .

القطب V_{DD} :

هو أيضاً قطب التغذية لشاشة الــ LCD ، ولكن ذو القيمة (+5V) .

القطب V_{EE} :

ويدعى أيضاً بــ V_O وهو قطب جهد التباين ، ويقصد بالتباين هو حدة ظهور الرمز على الشاشة وأقل تباين أن لا نرى شيئاً على الشاشة يكون عند تطبيق (+5V) على هذا القطب ، وأعلى تباين للشاشة يكون عند تطبيق (0 V) على هذا القطب .

ويمكن التحكم بتباين الشاشة عن طريق وصل قطب التباين (V_O) إلى مقاومة متغيرة .

القطب RS :

وهو مسجل اختيار الدخل لشاشة الــ LCD وذلك في حال طبق عليه (٠) منطقي عندها نريد إرسال كلمة تحكم ، بينما في حال طبق (١) منطقي فعندها نريد إرسال معطيات إلى الشاشة ، وفي الحالتين عن طريق أقطاب المعطيات .

القطب R/W :

وهو للقراءة أو الكتابة إلى الشاشة ، نطبق (٠) منطقي على هذا القطب عندما نريد كتابة (**إرسال**) المعلومات إلى شاشة الــ LCD ، ونطبق (١) منطقي عندما نريد قراءة (استقبال) المعلومات من شاشة الــ LCD .

القطب E :

وهو قطب تمكين شاشة الــ LCD ، فكل معلومة يتم كتابتها أو قراءتها من شاشة الــ LCD يجب إرفاقها بنبضة تمكين على هذا القطب ، ونبضة التمكين هذه تحدث عند الجبهة الهابطة ، أي تتم هذه النبضة برفع القطب إلى الــ (١) منطقي وإنزاله إلى الــ (٠) منطقي بعد تأخير مناسب .

الأقطاب DB0 . . . DB7 :

وهي أقطاب المعطيات (DATA) ، حيث يتم كتابة المعطيات أو كلمات التحكم عبر هذه الخطوط إلى شاشة الــ LCD وكذلك قراءة المعطيات منها . . .

القطبين K و A :

تزود بعض الشاشات بهذين القطبين ، وهما على الترتيب قطبي المهبط و المصعد للّيد المسطّح المستطيل الموجود خلف شاشة الـــ LCD ، والذي يضاء عند تطبيق الـــ (١) منطقي على المصعد (A) و الـــ (٠) منطقي على المهبط (K) .

وظيفة هذا الليد هو تأمين الإضاءة الكافية لشاشة الـــ LCD ليتمكن المستخدم من رؤية العبارات المكتوبة عليها في **ظلمة الليل** . . .

وبالتالي من أجل جعل شاشة الـــ LCD تقوم بعمل معين لابد من وجود كلمات تحكم (**تعليمات**) يتم تزويد هذه الشاشة بها عن طريق أقطاب التحكم ، يوضح الجدول التالي أوامر التعامل مع المعالج الخاص بشاشة الإظهار الكرستالية :

Instructions	**Code**									
	RS	R\W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Clear Display	٠	٠	٠	٠	٠	٠	٠	٠	٠	1
Return Home	٠	٠	٠	٠	٠	٠	٠	٠	1	X
Entry Mode Set	٠	٠	٠	٠	٠	٠	٠	1	I\D	S
Display ON\OFF	٠	٠	٠	٠	٠	٠	1	D	C	B
Cursor\Display Shift	٠	٠	٠	٠	٠	1	S\C	R\L	X	X
Function Set	٠	٠	٠	٠	1	DL	N	F	X	X
Set CGRAM Address	٠	٠	٠	1	CG-RAM Address					
Set DDRAM Address	٠	٠	1	DD-RAM Address						
Ready Busy & Address	٠	1	BF	Address						
Write Data To CG\DD RAM	1	٠	Write Data							
Read Data To CG\DD RAM	1	1	Read Data							

مجموعة التعليمات الخاصة بشاشة الـــ LCD :

١ – Clear Display :

يتم كتابة الرقم 20h (رقم الأسكي للفراغ) إلى جميع حجرات الذاكرة DD-RAM ويعود المؤشر إلى الخانة رقم صفر التي عنوانها 80h ، وإذا كان قد تم إزاحة خانات الإظهار فستعود الإزاحة إلى الموقع الأصلي . بكلمات أخرى يمكن القول إن تنفيذ هذه التعليمة يعيد المؤشر إلى الموقع الأصلي في أول خانة من طرف اليسار .

Db7	- -	Db1	Db0	
			RS	R\W
0	0	0	0	1

٢ – Return Home (Cursor) :

يعود المؤشر إلى العنوان صفر (80h) ، ولكن في هذه الحالة لا تتغير قيم الذاكرة DD-RAM حيث تبقـــى محافظة على قيمها السابقة .

Db7	- -	Db1	Db0	
			RS	R\W
0	0	0	1	X

٣ – Entry Mode Set :

يتم استخدام في كلمة التحكم هذه :

I\D : Increment / Decrement of DDRAM address (cursor or blink)

يقوم بالتحكم بعداد عنوان الذاكرة DD-RAM ، فعندما يكون ("1"=I\D) يؤدي إلى زيادة عداد عنوان الذاكرة ، بينما في حال كون ("0"=I\D) فإنه يؤدي إلى نقصانه .

S : Shift of entire display

عندما ("1"=S) تنزاح كامل لوحات الإظهار لليمين أو لليسار حسب قيمة I\D فإذا كانت ("1"=I\D) تتم الإزاحة لليسار ، وإذا كانت ("0"=I\D) تتم الإزاحة إلى اليمين ، وبالتالي فإن المؤشر يكون ثابتاً عند إزاحة خانات الإظهار .

Db7	- -	Db1	Db0	
			RS	R\W
0	0	0	I\D	S

٤ – Display ON\OFF Control :

الفارس لتقنيات الحاسوب والإتصالات
سوريا – حماه- هـ-229619 33 963+

يكون الإظهار في حالة ON عندما (D="1") - Display ON/OFF control bit -
ويكون في حالة OFF عندما (D="0") .

يظهر المؤشر في حال كون (C="1") - Cursor ON/OFF control bit -
بينما لا يظهر عندما (C="0") .

يظهر المؤشر بشكل وامض بتردد 2.5hz عندما (B="1") - Cursor Blink ON/OFF
- control bit ويكون ثابت بدون خفقان عندما (B="0") .

Db7 - Db1 Db0

								RS	R\W
٠	٠	٠	٠	٠	٠	1	D	C	B

٥ – Cursor Or Display Shift :

يتم إزاحة المؤشر أو لوحة الإظهار وذلك حسب قيم S\C و R\L ووفقا للجدول التالي :

EXPLANATION	S\C	R\L
يزيح المؤشر إلى اليسار (يزداد عداد العناوين بمقدار واحد)	0	0
يزيح المؤشر إلى اليمين (ينقص عداد العناوين بمقدار واحد)	0	1
يزيح كامل لوحة الإظهار لليمين (تبدو العبارة متحركة إلى اليسار)	1	0
يزيح كامل لوحة الإظهار لليسار (تبدو العبارة متحركة إلى اليمين)	1	1

Table. Shift Patterns According To S/C And R/L Bits

Db7 - Db1 Db0

								RS	R\W
٠	٠	٠	٠	٠	1	S\C	R\L	X	X

٦ – Function set :

DL : Interface data length control bit

نمط الملائمة بـ 8-Bit عندما (DL="1") . نمط الملائمة بـ 4-Bit عندما (DL="0") ، حيث يتم إرسال البايت الواحد على مرحلتين :

النيبل العلوي ثم النيبل السفلي ، وبين النيبلين نبضة تمكين .

N : Display line number control bit

لوحة الإظهار ذات سطر واحد في حال (N="0") . لوحة الإظهار أكثر من سطر في حال (N="1") .

F : Display font type control bit

كل رمز عبارة عن (5×7) نقطة عندما (F="0") . كل رمز عبارة عن (5×10) نقطة عندما (F="1") .

Db7 - Db1 Db0

RS R\W

٠	٠	٠	٠	1	DL	N	F	X	X

: Set CGRAM Address – ٧

يضع الرقم الثنائي (AAAAAA) إلى عداد العناوين (AC) ، وبالتالي فإن كل المعطيات التي ستأتي لاحقاً ستوضع في CGRAM .

Db7 - Db1 Db0

RS R\W

٠	٠	٠	1	A	A	A	A	A	A

: Set DDRAM Address – ٨

يضع الرقم الثنائي (AAAAAA) إلى عداد العناوين (AC) ، وبالتالي فإن كل المعطيات التي ستأتي لاحقــاً ستوضع في DDRAM .

نلاحظ أنه من أجل (AAAAAAA= 0000000) تكون التعليمة 80h = 128 ، وهو عنوان أول بايــت في ذاكرة الإظهار DDRAM ، ومن أجل (AAAAAAA= 1111111) تكــون التعليمــة = FFh 256 ، وهو عنوان آخر بايت في ذاكرة الإظهار DDRAM . وبالتالي فهناك = FFh - 80h = 80h 128 بايت (حجرة ذاكرة) .

Db7 - Db1 Db0

RS R\W

٠	٠	1	A	A	A	A	A	A	A

: Ready Busy Flag And Address -٩

تتم قراءة علم المشغولية (BF) ومحتويات عداد العناوين (AC) ، عندما يكون (BF="1") فهذا يعنــي أن هناك تعليمة قيد الإنجاز ولم ينتهي تنفيذها بعد ، وبالتالي لا يمكن إرسال تعليمة جديدة حتـــى يـــصبح (BF="0") . والتعليمة الثنائية المقروءة هي قيمة عداد الــ DDRAM أو الــ CGRAM ، ويعتمد ذلك على آخر تعليمة هل كانت Set DDRAM أم Set CGRAM .

Db7 - Db1 Db0

RS R\W

٠	1	BF	A	A	A	A	A	A	A

: Write Data To CGRAM Or DDARM – 10

كتابة بايت كامل إلى الذاكرة DDRAM أو إلى CGRAM ، وهذا يتحدد بحسب آخر تعليمة.. هــل كانــت Set CGRAM Address أم Set DDRAM Address .

Db7 - Db1 Db0

RS R\W

1	•	D	D	D	D	D	D	D	D

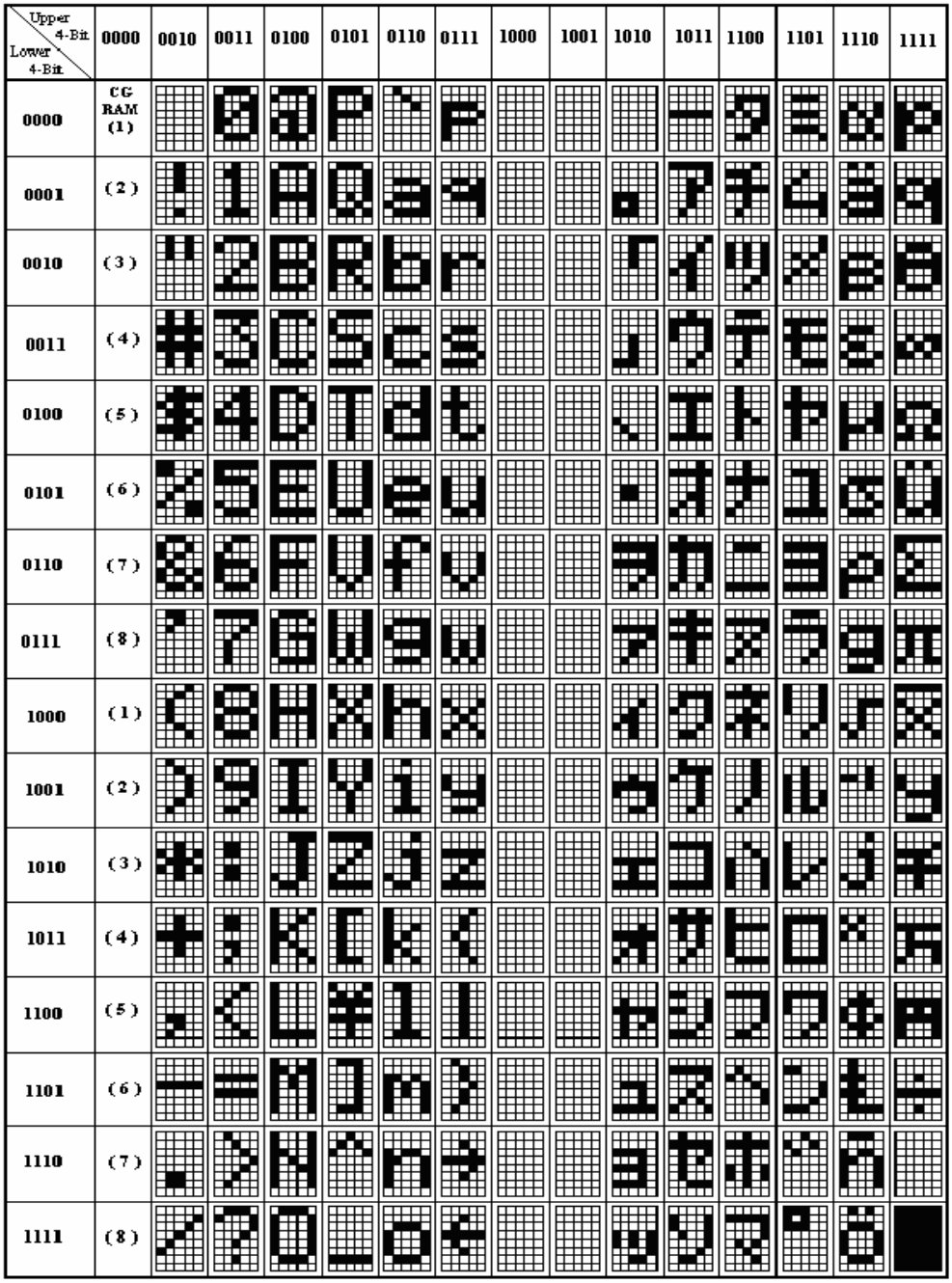
: Read Data From CGRAM Or DDRAM – 11

نفس التعليمة السابقة ولكن يتم قراءة بايت من CGRAM أو من DDRAM .

Db7 - Db1 Db0

								RS	R\W
1	1	D	D	D	D	D	D	D	D

تملك شاشة الــ LCD شيفرة (ASCII) خاصة لمحارفها ، حيث أن كل محرف في هذه الشيفرة مشفّر بــ 8-Bit وبالتالي عند إرسال البايت الخاص بمحرف ما إلى الــ LCD سيتم إظهاره ، وفق الجدول التالي :



الفارس لتقنيات الحاسوب والإتصالات
سوريا – حماه- هـ-229619 33 963+

تعليمات شاشة الـــ LCD

قبل استخدام أي من هذه التوابع يجب في البداية ادراج المكتبة "LCD.h"
وفي البداية يجب ان يتم التصريح عن المنفذ الموصول مع شاشة الـــ LCD ويتم ذلك بالتعبير عن عنوان هذا المنفذ والعناوين هي كالتالي :

PORTA=0x1B
PORTB=0x18
PORTC=0x15
PORTD=0x12
PORTE=0x03
PORTF=0x11

وهذه المكتبة تدعم الشاشات التالية 1x8, 2x12, 3x12, 1x16, 2x16, 2x20, 4x20, 2x24 and 2x40.

مثال :
بفرض تم وصل الشاشة على المنفذ C .

```
#asm
.equ __lcd_port=0x15
#endasm
```

بعد ذلك يتم إدراج المكتبة

```
#include <lcd.h>
```

ويجب أن يتم وصل أقطاب الشاشة على أقطاب المنفذ المخصص للشاشة كمايلي (حيث bit 0 هي الخانة الآولى من المنفذ الذي خصصناه للشاشة)

bit 0 RS (pin4) ------
bit 1 RD (pin 5) ------
bit 2 EN (pin 6) ------
bit 4 DB4 (pin 11) ---
bit 5 DB5 (pin 12) ---
bit 6 DB6 (pin 13) ---
bit 7 DB7 (pin 14) ---

كما ينبغي عليك أن تقوم بوصل جهد الطاقة وجهد التباين في الشاشة حسب تعليمات الشاشة .

تعليمات شاشة الــ LCD ذات المستوى المنخفض low level LCD Functions are

void _lcd_ready(void)

تقوم هذه التعليمة بالإنتظار حتى تصبح الشاشة جاهزة لإستقبال البيانات ويجب استدعاء هذا التابع قبل كتابة البيانات باستخدام التابع _lcd_write_data

void _lcd_write_data(unsigned char data)

يقوم هذا التابع بكتابة البيانات لمسجل التعليمات للشاشة ولهذ التعليمة إرتباط بالتعليمة السابقة وفي حال أردنا إرسال أي بيانات فيجب أن نتبع الخطوات التالية حيث المثال التالي يقوم بتشغيل المؤشر :

```
_lcd_ready();
_lcd_write_data(0xe);
```

void lcd_write_byte(unsigned char addr, unsigned char data)

يقوم هذ التابع بكتابة بايت إلى ذاكرة RAM المسؤولة عن توليد المحارف .

unsigned char lcd_read_byte(unsigned char addr)

قراءة بايت من ذاكر RAM المسؤولة عن توليد الرموز .

تعليمات ذات المستوى العالي : high level LCD Functions

unsigned char lcd_init(unsigned char lcd_columns)

يقوم هذا التابع يتهيئة الشاة حيث يقوم بمسح الشاشة ويضع المؤشر في البداية أي في العمود رقم 0 والسطر رقم 0 . ويجب أن يتم تعريف عدد أعمدة الشاشة lcd_columns ويكون في هذه الحالة المؤشر غير ظاهر والتابع يرجع رقم 1 إذا تم تهيئة الشاشة و0 إذا لم يتم التهيئة لخطأ ما .

void lcd_clear(void)

يقوم هذا التابع بمسح الشاشة ويضع المؤشر على السطر 0 والعمود 0.

void lcd_gotoxy(unsigned char x, unsigned char y)

يضع هذا التابع المشر في العمود رقم x والسطر رقم y حيث يتم العد من الصفر للأسطر والأعمدة ومن اليسار إلى اليمين ومن الأعلى والأسفل .

void lcd_putchar(char c)

إظهار المحرف c في الموضع الحالي .

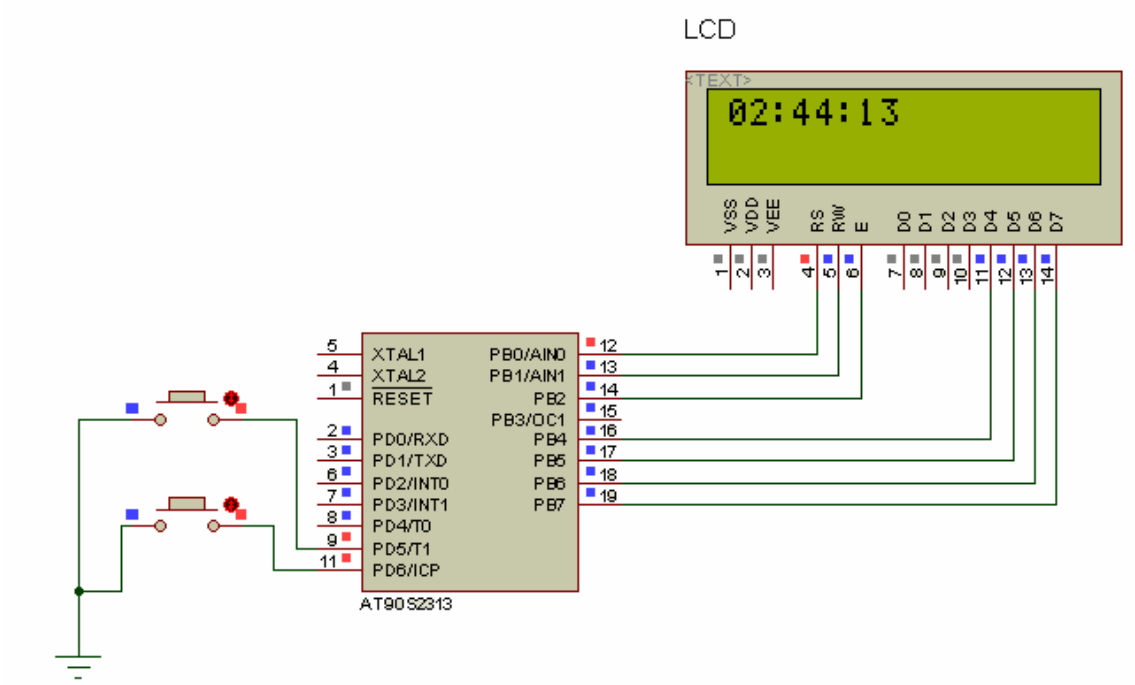
void lcd_puts(char *str)

إظهار السلسلة المحرفية str في الموضع الحالي المعنونة في ذاكرة RAM.

void lcd_putsf(char flash *str)

إظهار السلسلة المحرفية str في الموضع الحالي المعنونة في ذاكرة FLASH.

المشروع عبارة عن ساعة إلكترونية قابلة للضبط للدقائق والساعات عبر الكباسن الموصولين عبر القطبين PD5 ، PD6 .



دارة المشروع

البرنامج

```
#include <90s2313.h>
#include <delay.h>
#include <lcd.h>
#include <stdlib.h>
#asm
  .equ __lcd_port=0x18 ;PORTB
#endasm
unsigned char s=59,m=59,h=12;
unsigned char a,b,*str;

interrupt [TIM1_COMP] void timer1_comp_isr(void)
{
if (s==59)
 {
  s=0;
  if(m==59)
    {
     m=0;
     if(h==12)
     h=0;
```

```
    h++;
    }
  else
    m++;
}
else
s++;

}

void main(void)
{

bit q=0;

PORTB=0x00;
DDRB=0xFF;

PORTD=0x60;
DDRD=0x1f;

lcd_init(16);
                                // Timer/Counter 1 initialization
                                                // Clock source: System Clock
                                  // Clock value: 4000.000 kHz


                                // // Compare Match Interrupt: On
TCCR1A=0x00;
TCCR1B=0x0b;
OCR1H=0xF4;
OCR1L=0x24;
                        // // Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x40;
                                    // Global enable interrupts
#asm("sei")
while(1)
  {

   if(PIND.5==0 && q==0 )
  {
  if (m==59)
   {
    m=0;
   }
  m++;
  q=1;
```

```
  }

  if(PIND.6==0 && q==0 )
  {
    if (h==12)
      {
       h=0;
       }
  h++;
  q=1;
  }

  if (PIND.5==1 && PIND.6==1)
  {
  q=0;
   }

   lcd_gotoxy(0,0);
   b=h/10;
  itoa(b,str);
   lcd_puts(str);
   a=h%10;
   itoa(a,str);
   lcd_puts(str);
   lcd_putchar(':');

   lcd_gotoxy(3,0);
   b=m/10;
   itoa(b,str);
   lcd_puts(str);
   a=m%10;
   itoa(a,str);
   lcd_puts(str);
   lcd_putchar(':');

   lcd_gotoxy(6,0);
    b=s/10;
   itoa(b,str);
   lcd_puts(str);
   a=s%10;
   itoa(a,str);
   lcd_puts(str);

};                                    // end while

}                                     // end main
```

المشروع الثامن
قيادة محرك تيار مستمر باستخدام تقنية تعديل عرض الموجة PWM

من العلاقة العامة لمحركات التيار المستمر نجد أن حتى نستطيع تغير سرعة محرك التيارالمستمر يوجد ثلاث طرق:

$$w = \frac{V}{k\Phi} - \frac{R_a}{(k\Phi)^2}T$$

(المعادلة العامة للمحرك التيار المستمر)

1: تغير السرعة عن طريق تغير المقاومة الموصولة على التسلسل مع المتحرض وهي طريقة قديمة وغير مجدية.

2: تغير الفيض المغناطيسي Φ في ملف التهيج وذلك إما بتغيير جهد ملف التهييج أوتغيير مقاومته ولكن هذه الطريقة لايمكن تطبيقها عند كل مجالات السرعة حيث أننا نحاول ان لا ندخل في منطقة الإشباع لنواة المحرك .

3: تغير الجهد المطبق على طرفي المتحرض V وهذه الطريقة هي الأكثر استخداماً وهي التي سوف نتطرق إليها.

طريقة تغير جهد المتحرض:

السؤال الذي يطرح نفسه الآن كيف يمكننا أن نغير الجهد .....؟

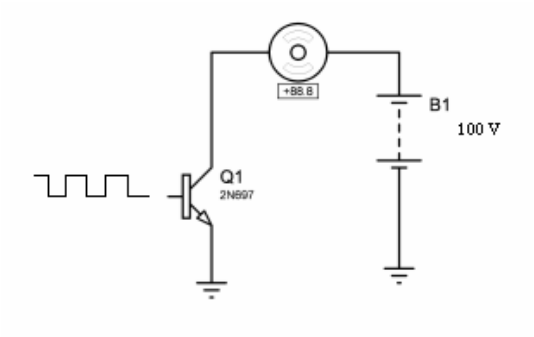
يوجد عدة طرق لتغير الجهد وهي :

1: عن طريق محولة ذاتية ونصل على الخرج مقوم جسري ونصل خرج المقوم مع المحرك وعن طريق تغيير ذراع المحولة يتغيرالجهد المطبق على المحرك.

2: عن طريق المحولة العادية يوصل على خرجها مبدلة ثايرستورية ويمكن التحكم بجهد الخرج المطبق على المحرك بتغير زاوية القدح للثايريستور.

3: طريقة الجهد المتقطع والتي سوف نستخدمها هنا إن شاء الله...

بفرض لدينا محرك مستمر جهده الإسمي هو V 100 وهو الجهد الازم تطبيقه ليدور المحرك بسرعته الإسمية ولكي نستطيع تغييرسرعته من الصفر إلى السرعة الإسمية يجب أن نغير الجهد من الصفرإلى V 100 وسوف نستخدم لهذا الغرض عملية تقطيع الجهد للحصول على قيمة وسطية متغيرة بمطال ثابت الشكل (1,8) .....



الشكل (1,8)

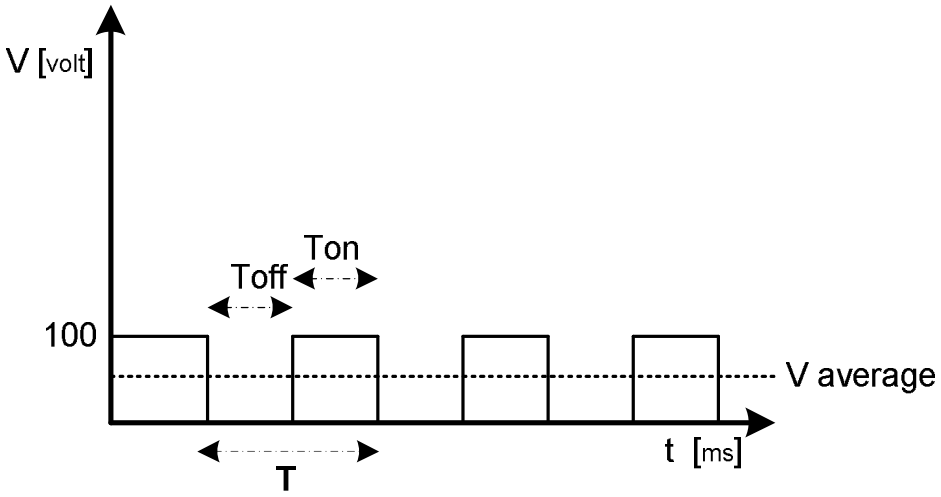
فعندما نطبق نبضة موجبة على قاعدة الترنزيستور فإنه يطبق الجهد الإسمي كاملاً على المحرك 100 v وعندما نطبق منطق منخفض فإنه يتوقف مرور التيار في المحرك I=0 .

ولكن عندما نطبق على التارنزيستور نبضات ذات عرض متغير فإن المحرك سوف يعمل ويتوقف بما يتناسب مع عرض النبضات المطبقة .

فإذا اعتبرنا أن زمن تطبيق النبضة هو Ton وزمن تطبيق المنطق المنخفض هو Toff

وقمنا بإعطاء قطار من النبضات بتردد عالي فإن ذلك سوف يؤدي إلى تقطيع التيار في المحرك وبالتالي فإن سرعة المحرك سوف تتغيروتعطى قيمة الجهد الوسطية المطبقة على المحرك بالعلاقة :

$$V_a = V\frac{T_{on}}{T} = V\frac{T_{on}}{T_{on}+T_{off}}$$

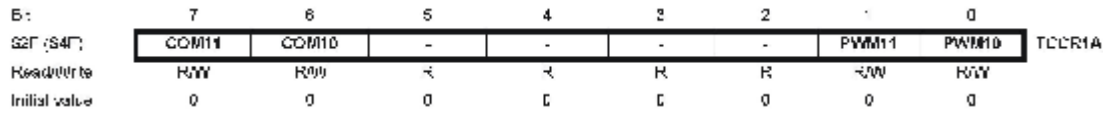


حيث Vaهو الجهد الوسطي بفرض أن Ton=Toff فإن Va=1/2 V

ولتحقيق هذه النبضات فإننا سوف نستخدم المؤقت العداد في نمط PWM.

المؤقت/العداد ١ في نمط PWM The Timer/Counter1 in PWM Mode

مسجل التحكم A بالمؤقت/العداد ١ TCCR1A



The Timer/Counter 1 Control Register A

الخانتان 0 , 1 – PWM10 , PWM11 : خانتي اختيار معدل عرض النبضة :

تحدد هاتان الخانتان عملية اختيار PWM للمؤقت/العداد١ كما هو مبين في الجدول (11) . وقد شرح هذا النمط بشكل مفصل في فقرة " المؤقت/العداد١ في نمط PWM " .

الشرح	PWM11	***PWM10***
حجب عمل PWM للمؤقت/العداد١	0	0
معدل عرض نبضة PWM بثمان خانات	0	1
معدل عرض نبضة PWM بتسع خانات	1	0
معدل عرض نبضة PWM بعشر خانات	1	1

الجدول(11) : اختيار نمط PWM

عند اختيار نمط PWM ، فإن المؤقت /العداد١ مع مسجل مقارنة الخـرج١ – OCR1A ، يـشكلان مولـداً لمعدل عرض النبضة PWM بثمان أو تسع أو عشر خانات ،فأثناء العمل ، ستظهر النبضات الخاطئـة glitch و نبضات PWM الصحيحة على قطب المتحكم PB1(OC1) . يعمل المؤقت/العداد١ كعداد صاعد/هابط . و يعد صاعداً من $0000 إلى القمة TOP (انظر الجدول ٣١) ، و عندما يصل إلى القمة يصبح كعداد هـابط إلى أن يصل للصفر $0000 قبل تكرار الدورة . و عندما تصبح محتويات العداد مماثلة لمحتويات الخانـات ٨ أو ٩ أو ١٠ الأقل أهمية في المسجل OCR1A ، فإن القطب PB1 (OC1) يأخذ القيمة إما واحـد أو صـفر منطقي وذلك بما يتطابق مع إعدادات الخانتين COM1A1 , COM1A0 الموجودتان فـي مـسجل الـتحكم بالمؤقت/العداد١ – TCCR1 . و الجدول (14) يشير إلى شرح هاتين الخانتين .

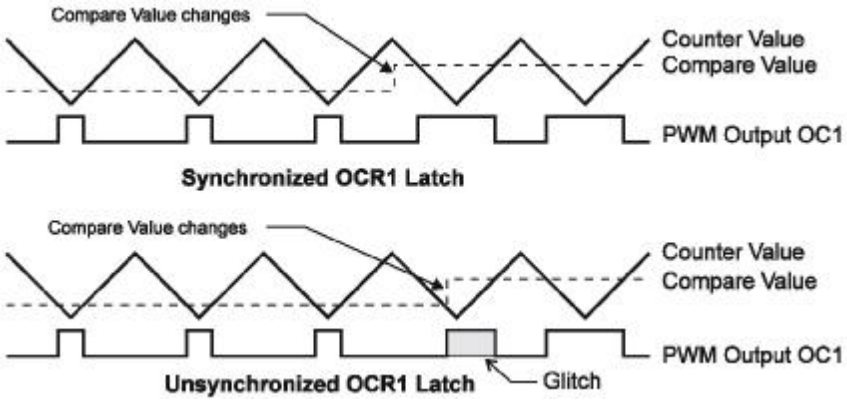
التردد	أعلى قيمة عد للمؤقت	دقة PWM
Ftck1/510	$00FF(255)	8-bit
Ftck1/1022	$01FF(511)	9-bit
Ftck1/2046	$03FF(1023)	10-bit

الجدول (13) : قيم القمة TOP للمؤقت وتردد PWM

التأثير على القطب OC1	COM11	COM10
لا يوجد اتصال	0	0
لا يوجد اتصال	0	1
التصفير عند نظير المقارنة في العد الصاعد . التفعيل عند نظير المقارنة فـي	1	0
التصفير عند نظير المقارنة في العد الهابط . التفعيل عند نظير المقارنـة فـي	1	1

الجدول (14) : اختيار نمط المقارنة١ في نمط PWM

نلاحظ أنه في نمط PWM ، أن الخانات العشرة الأقل أهمية في المسجل OCR1 تنتقل عند الكتابة إلى الموقع اللحظي temporary location . وتمسك إلى أن يصل المؤقت/العداد١ للقمة . وهذا يمنع حدوث نبضات PWM فردية الطول (نبضة خاطئة glitch) في حادثة الكتابة غير المتزامنة على المسجل OCR1A . وكمثال على ذلك أنظر الشكل(34) .



الشكل (34) : تأثيرات حوادث المسك غير المتزامنة للمسجل OCR1

و في الفترة الزمنية بين الكتابة و عملية المسك ، فإن القراءة من المسجل OCR1 ستقرأ محتويات الموقع اللحظي . هذا يعني أن معظم القيم الجديدة المكتوبة دائماً ستقرأ خارج المسجل OCR1 .

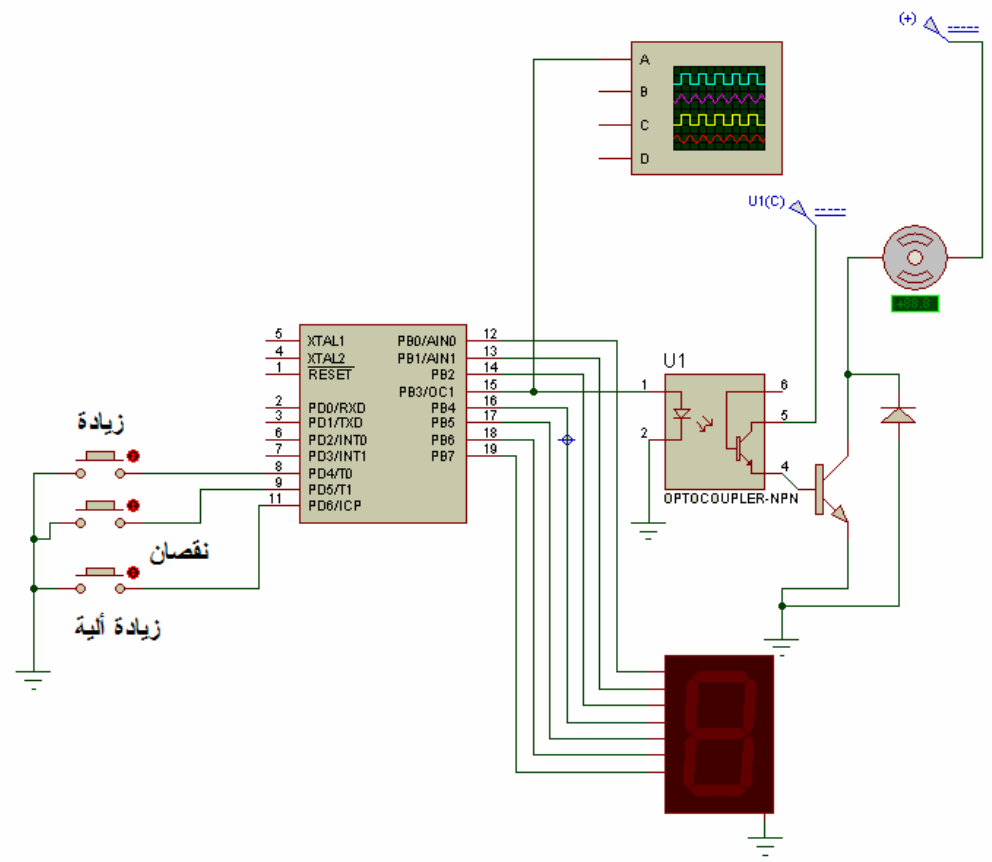
عندما تكون محتويات المسجل OCR1 تساوي $0000 أو القمة TOP ، فإن الخرج OC1 يمسك على المنطق العالي أو المنخفض وذلك بما يتلاءم مع قيم الخانتين COM11 , COM10 . كما هو مبين في الجدول (15).

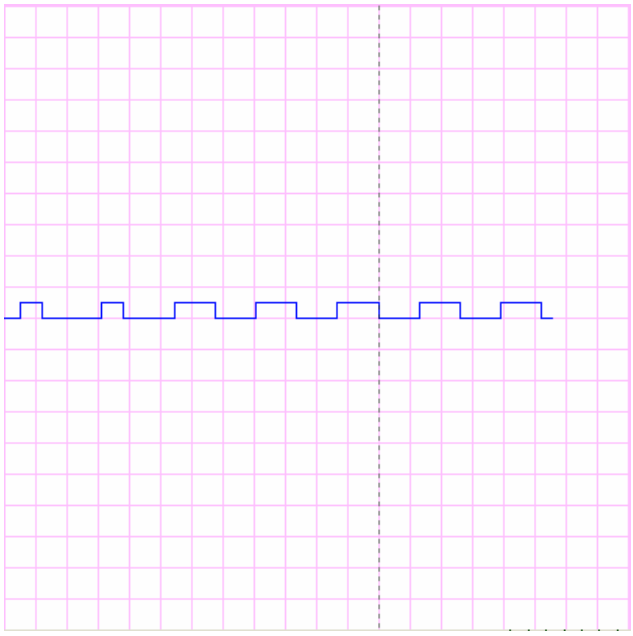
COM10	COM11	OCR1	خرج OC1
0 0	1 1	$0000 TOP	L H
1 1	1 1	$0000 TOP	H L

الجدول(15) : خرج PWM عند القيم : OCR = $0000 or TOP

في نمط PWM ، يصبح علم طفحان المؤقت TOV1=1 عندما يغير العداد اتجاه عده عند القيمة $0000 . و تعمل مقاطعة طفحان المؤقت١ بشكل فعلي exactly كما في نمط المؤقت/العداد العادي . مثلاً: ينفذ المتحكم MCU روتين خدمة المقاطعة عندما يصبح علم الطفحان TOV1=1 مع كون خانة تمكين المقاطعة العامة I=1 . وهذا يطبق أيضاً على علم مقارنة الخرج١ والمقاطعة .

الدارة التالية تظهر لنا دارة المشروع سوف نتحكم بسرعة الدوران عن طريق الكباس بالضغط علـــى الكبـــاس الموصول على PIND.4 تزداد السرعة والكباس PIND.5 للنقصان و PIND.6 لزيـــادة الـــسرعة أليـــاً وتظهر لنا شاشة الإظهار رقم السرعة الآنية المطبقة .





البرنامج:

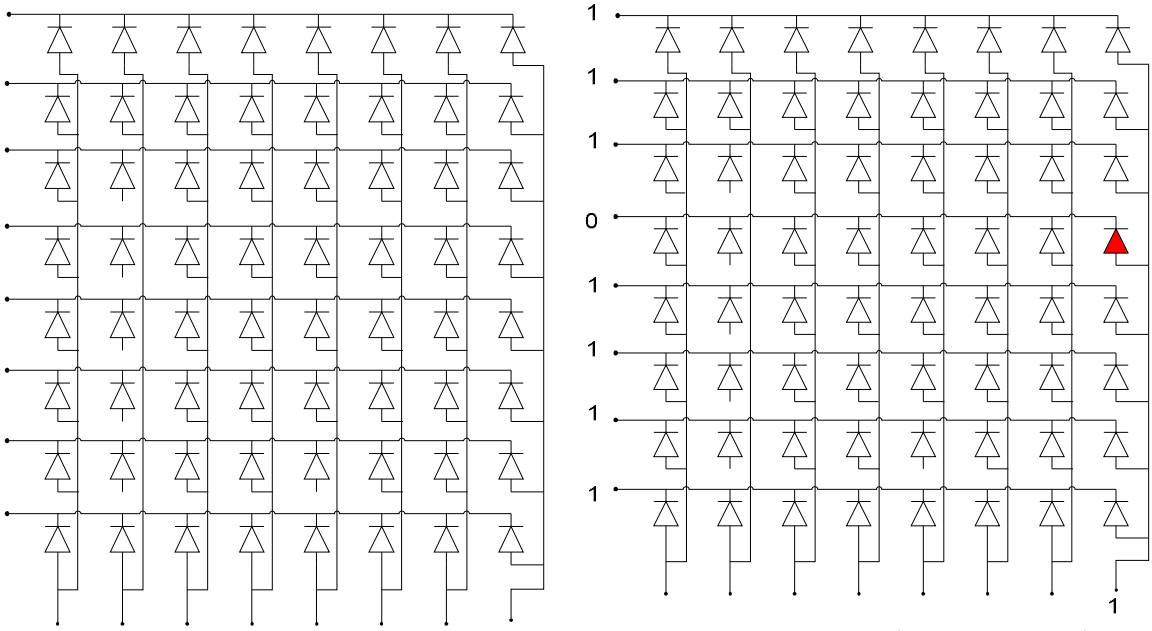
```
#include <90s2313.h>
#include <delay.h>


void main(void)
{
unsigned char decode[10]={0x77,0x06,0xb3,0x97,0xc6,0xD5,0xf5,0x07,0xf7,0xD7};
unsigne char OCR1H_[10]={0x03,0x03,0x03,0x03,0x03,0x02,0x02,0x01,0x01,0x00};
unsigned char OCR1L_[10]={0xff,0xfa,0xf4,0xf0,0x80,0xf0,0x00,0xf0,0x00,0x00};
unsigned char q=0,t=0;
PORTB=0x00;
DDRB=0xFF;

PORTD=0x7f;
DDRD=0x00;

TCCR0=0x00;
TCNT0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: 15.625 kHz
// Mode: Ph. correct PWM top=03FFh
// OC1 output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare Match Interrupt: Off

TCCR1A=0xc3;
TCCR1B=0x04;
TCNT1H=0x00;
TCNT1L=0x00;
OCR1H=0x03;
OCR1L=0xff;
PORTB |=decode[t];



while (1)
 {
   if(PIND.5==0 && q==0 && t!=0 )
    {
    t--;
    OCR1H= OCR1H_[t];
    OCR1L= OCR1L_[t] ;
    PORTB &=8;          // تصفير المنفذ عدا القطب الثالث
    PORTB |= decode [t];
```

```
    q=1;
    }

  if(PIND.4==0 && q==0 && t!=9 )
    {
     t++;
    OCR1H= OCR1H_[t];
    OCR1L= OCR1L_[t] ;
    PORTB &=8;
    PORTB |=decode[t];
    q=1;
    }


   if(PIND.6==0 && q==0 && t!=9 )
    {
     while(t!=9)
      {
       t++;
       OCR1H= OCR1H_[t];
       OCR1L= OCR1L_[t] ;
       PORTB &=8;
       PORTB |=decode[t];
       delay_ms(1000);
      }
    }

  if (PIND.5==1 && PIND.4==1)
      q=0;

 };
}
```

المشروع التاسع

الجريدة الإلكترونية

تتألف الجريدة الإلكترونية من مصفوفة من الليدات الموصولة مع بعضها البعض حيث يتم وصل مهابط الأسطر مع بعضها البعض ويتم وصل مصاعد العمود الواحد مع بعضها البعض كما في الشكل (1-9) حيث يتألف الشكل من ثمانية أسطر ومثلهما للأعمدة ولأجل إضاءة Led ماعلينا إلا أن نطبق 0 منطقي على سطره و 1 منطقي على عموده أما الليدات التي تقع على نفس العمود فيجب أن يطبق على مهبطهم 1 منطقي .



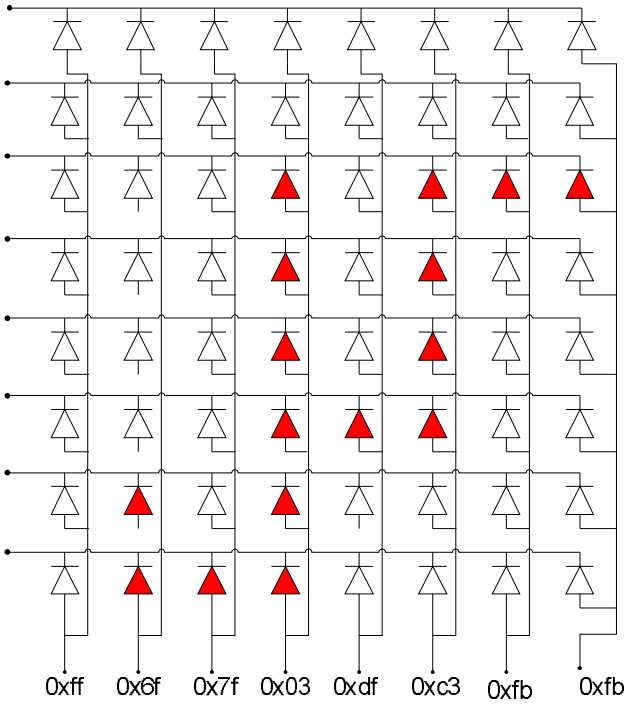
الكتابة على المصفوفة :

من أجل الكتابة على المصفوفة نقوم بتعين الليدات التي يجب أن تضئ والليدات التي يجب أن تنطفئ في كل عمود وكل سطر وبعد ذلك نقوم بتخزين شيفرة كل عمود وذلك بقراءة القيمة التي تطبق على الأسطر من أجل إضاءة ليدات هذا العمود في المثال السابق نلاحظ أن القيمة هي (11110111=0xf7) .

الشكل (2-9) يظهر لنا كلمة (كل) ونلاحظ أسفل كل عمود الشيفرة المقابلة لليدات المضاءة في كل عمود وتتم عملية الإظهار لهذه العبارة كالتالي :

في البداية نضع 1 منطقي على العمود الأول ثم نطبق على الأسطر الشيفر (0xfb) بعدها نضع صفر منطقي على هذا العمود ونضع 1 منطقي على العمود الثاني ثم نطبق شيفرة العمود الثاني (0xfb) وهكذا حتى نقوم بمسح جميع الأعمدة وعندها نعود للعمود الأول ونستمر بذلك مع الملاحظة أن هناك عملية تأخير زمني بين كل عمليتي إظهار حتى يتسنى لنا رؤية الليدات المضيئة، قد يخطر ببال أحدكمكأنه عندما ننتقل لإضاء العمود التالي بأنه

سوف ينطفئ العمود الأول وهذا يؤثر على استمرارية الكلمة الظاهرة على المصفوفة ، نقول لـــه هـــذا صـــحيح فالليدات العمود السابق سوف تنطفئ ولكن سرعة مسح الأعمدة وبسبب ظاهرة الإنطباع البصري سوف لن نلاحظ ذلك وهذا يتطلب إختيار مناسب لقيمة التأخير الزمني ين كل عمليتي إظهار.



تحريك الكلام على الشاشة :

نلاحظ أنه في العملية السابقة أن الكلام لايتحرك وإنما يظهر على شاشة الليدات فقط ومن أجـــل تحريكـــه نتبـــع الخوارزمية التالية :

في البداية نتبع نفس الطريقة السابقة في عملية إظهار الكلام على الشاشة ولكن بعد عدد معين من عملية مسح كل الأعمدة نقوم بنقل الشيفرة التي تظهر في العمود الثاني إلى العمود الأول والثالث إلى العمود الثـــاني وهكـــذا .... وبالطبع فإن شيفرة العمود الأول أصبحت خارج عملية الإظهار كما أنه تم إدخال شيفرة جديد على العمود الأخيـــر بعد عملية الإزاحة هذه نقوم بتكرار عملية المسح للأعمدة مثل المرحلة السابقة وتسمى هذه المرحلة بعملية مـــسح الأعمدة والشاشة متوقفة وعند إنتهاء ذلك العدد من عملية المسح نقوم بإجراء عملية الإزاحة من جديد . يتحدد عدد عملية الإزاحة بعدد بايتات الجملة التي نريد عرضها على الشاشة وبعد الوصول إلى أخر بايت نعود إلى البايـــت الأول في الشيفرة ونعرضة على العمود الأخير وهكذا يظهر الكلام بأنه يتحرك .

ملاحظة :

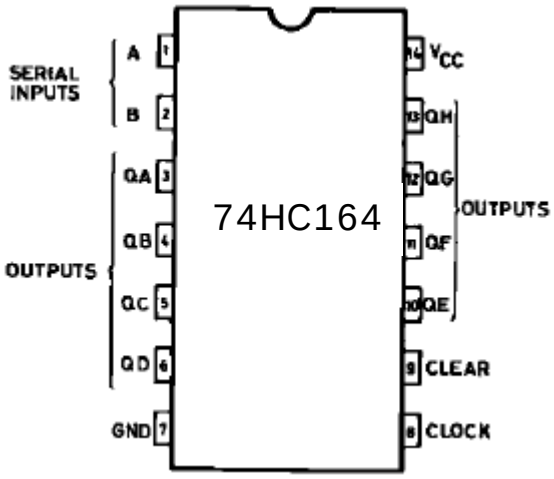
بما أن اللغة العربية تبدأ بالكتابة من اليمين إلى اليسار لذلك معملية المسح يجب أن تبدأ من العمـــود الأيمـــن إلـــى العمود الأيسر أما في اللغات الأخرى بالعكس .

كما انه يتم إضافة شيفرات إطفاء لجميع الليدات (0xff) في بداية تشفير الجملة من أجل دخول الكلمة في عمليـــة الإظهار من أقصى اليسار وليست تظهر مباشرة في العمود الأول وكذلك الأمر يتم إضافة شيفرت إطفاء في نهاية تشفير الجملة من أجل ترك فاصل بين نهاية الجملة وبداية الجملة.

عملية المسح باستخدام المايكرو AT90s2313 :

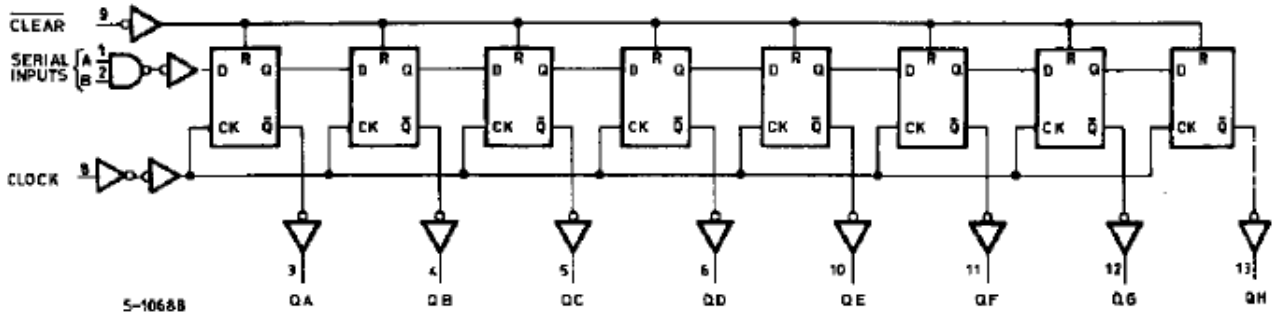
نلاحظ أنه كلما كان عدد الأعمدة أطول كلما إزدادت طول الجملية التي تظهر على الشاشة ويعطي ذلك جمالية للشاشة إلا أن زيادة عدد الأعمد يعني ذلك زياة عدد الأقطاب التي سوف تتحكم بهذه الأعمد وكما نعلم أن المتحكم AT90s2313 لايملك سوى 15 قطب وفي حال استخدمنا مصفوفة بطول 16 عمود و8 أسطر وذلك يعني أنه يجب أن يكون هناك 24 قطب من أجل عملية المسح وذلك غير ممكن في هذا المتحكم ولذلك تم الإستعانة بالدارة المتكاملة 74HC164 وماهي إلى عبارة عن مسجل إزاحة يقوم بتحويل البايانات من شكلها التسلسلي إلى التفرعي ويتم ذلك كمايلي :

تحتوي الدارة المتكاملة 74HC164 على ثمانية أقطاب خرج وعلى قطبين لإستقبال البيانات التسلسلية حيث يوجد بين هذين القطبين دارة and داخلية لذا يجب قصر هذين القطبين مع بعضهما البعض كما تحتوي على قطب استقبال نبضات الساعة و التي تستخدم من أجل تحويل القيمة المطبقة على الدخل إلى أول قطب من أقطاب الخرج وتحول القيمة السابقة على القطب الأول للخرج إلى القطب الثاني من أقطاب الخرج وهكذا ..... .



وفي مثالنا الحالي نلاحظ أنه لدينا 16 عمود لذلك فهذ الدارة لن تكفي بالغرض وإنما نحن بحاجة إلى إثنتين من هذه الدارة حيث نقوم بوصل آخر قطب خرج في الدارة الأولى مع أقطاب الدخل للدارة الثانية أما نبضات الإزاحة فيجب أن يتم تطبيقها بشكل متزامن على كل مدخلي الساعة في كلتا الدارتين .

في البداية يكون خرج الأقطاب في كلتا الدارتين غير محدد ومن بداية عملية الإظهار نضع 1 منطقي على أقطاب الدخل لأول دارة ثم نطبق نبضة ذات طول معين حتى تستجيب هذه الدارة عندها ينتقل هذا الـ 1 إلى القطب الأول في خرج الدارة الأولى والموصول إلى العمود الأول في الشاشة وعندها نطبق على الأسطر الشيفرة المقابلة للعمود الأول وبعد فترة زمنية معينة نضع 0 منطقي على مدخل الدارة الأولى ونطبق نبضة على مدخل الساعة عندها يتم تطبيق 0 على القطب الأول الموصول مع العمود الأول وينزاح الـ 1 إلى القطب الثاني إي إلى العمود الثاني وعندها يتم تطبيق شيفرة العموج الثاني ونستمر بيتطبق 15 صفر منطقي على مدخل الدارة الأول وعندها يكون الـ1 منطقي وصل إلى العمود الأخير بعدعها نعيد تطبيق 1 منطقي على الدخل وبعده 15 صفر منطقي .

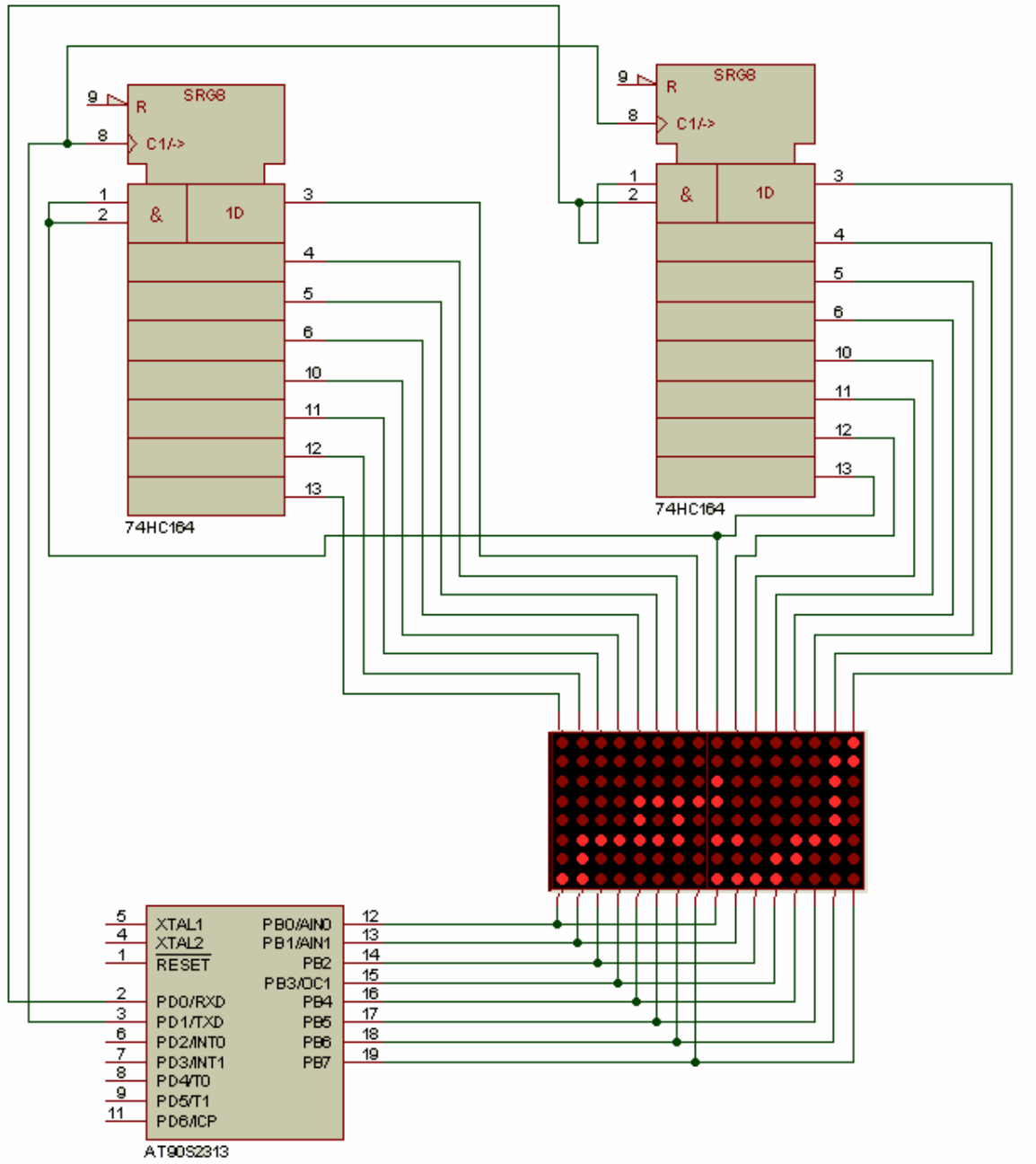


ملاحظة :

نلاحظ أن قطب الخرج للدارة الأولى يجب أن يطبق على العمود الذي في أقصى اليمين في حال كان الكلام المراد إظهاره باللغة العربية .

وظيفة القطب Clear في الدارة المتكاملة 74HC164 هو وضع صفر منطقي على كل المخارج عند وصله إلـــى الصفر منطقي .

دارة المشروع



البرنامج:

تم وصل الأسطر إلى الثمانية على أقطاب المنفذ B وتم تخصيص القطب PORTD.0 خرج للبيانات التسلسلية والقطب PORTD.1 خرج لنبضات الساعة .

يتألف البرنامج من ثلاث حلقات : الأولى وهي الأكبر وتحتوي على عدد البايتات للجملة التي سوف يتم عرضها أما الثانية تحتوي على عدد مرات المسح والشاشة متوقفة وهذه الحلقة هي المسؤولة عن سرعة تحريك الكلام فكلما إزداد عدد مرات المسح نقصت السرعة ، والحلقة الثالثة مسؤولة عن عدد الأعمدة .

والجملة المشفرة هي "رمضان كريم " .

الفارس لتقنيات الحاسوب والإتصالات
سوريا – حماه- هـ-229619 33 963+

```
#include <90s2313.h>
void delay(void);


 const unsigned char  m[74]={0x00,0x00,                    // 74 is a number of words, this array
                                stored in flash memory because its type is const and it coded "رمضان كريم"
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x10,0x18,0x0C,0x07,0x01,0x01,
0x1C,0x14,0x1C,0x04,0x04,0x3C,0x2C,0x24,
0x34,0x1C,0x44,0x04,0xFC,0x00,0x3C,0x04,
0x84,0x3C,0x00,0x00,0x80,0x80,0x80,0xC0,
0xC0,0x7C,0x04,0x06,0x03,0x01,0x05,0x35,
0x10,0x1C,0x14,0x1C,0x04,0x04,0x07,0x01,
0x00,0x00, 0x00,0x00,0x00,0x00,0x00,0x00
};

void main(void)
{
int i;  // number of words in the Sentence
int w; // number of columns
int q; // number of scanning  Times

PORTB=0x00;
DDRB=0xFF;

PORTD=0x00;
DDRD=0x7F;

while(1)
{

for(i=0;i<74 ;i++)
for (q=0;q<7;q++)           //Number of scanning times
  {
   w=0;
   PORTD=1;
   delay();       //delay for clock
   while(w!=16)    //Number of columns
    {
    PORTB=0xff;
     PORTD|=2;
     delay();
     PORTD=0;
     PORTB=~m[i+w];
     w+=1;
     delay();
     delay();
    }
```

```
  }
};

}


void delay(void)
{
int e=99;
while (e!=0)
e--;
}
```

المشروع العاشر
النافذة التسلسلية UART

النافذة التسلسلية

جُهز المتحكم AT90S2313 بمرسل ومستقبل عام غير متزامن UART (Universal synchronous Receiver Transmitter)يتمتع بالمزايا الرئيسية التالية:

١- مولد معدل بود يولد أي معدل بود.

٢- معدلات بود Baud عالية عند الترددات XTAL المنخفضة .

٣- خانات المعطيات المرسلة أو المستقبلة 8 أو 9 خانات .

٤- ترشيح الضجيج noise .

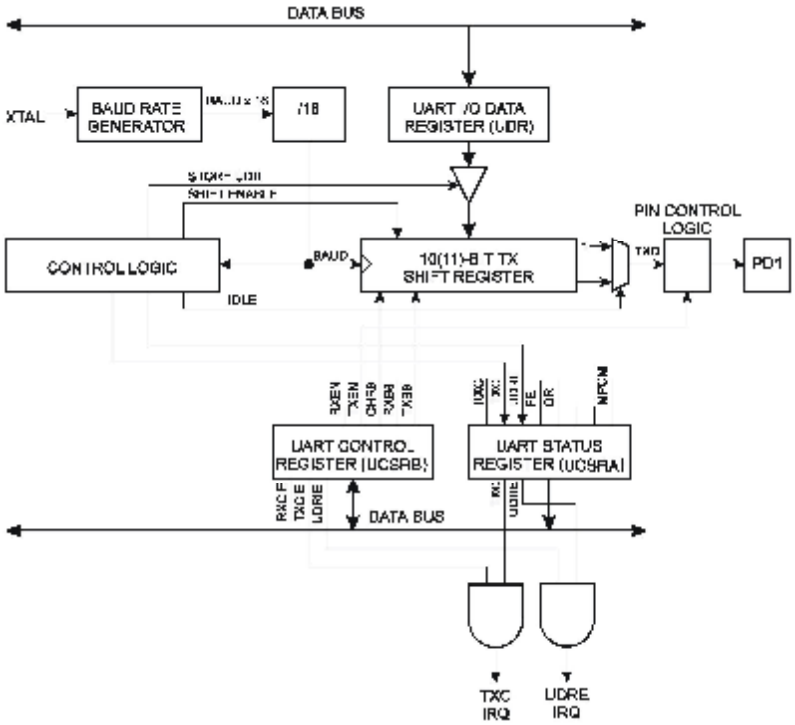
٥- اكتشاف التجاوز .

٦- اكتشاف خطأ هيكلية الاستقبال .

٧- اكتشاف خانة البدء الوهمية في الاستقبال .

٨- ثلاث مقاطعات منفصلة عند : اكتمال الإرسال TX،فراغ مسجل المعطيات TX، اكتمال الاستقبالRX.

إرسال المعطيات Data Transm يبين الشكل (40) المخطط الصندوقي لمرسل UART .



الشكل(40) : مرسل وحدة UART

الفارس لتقنيات الحاسوب والإتصالات
سوريا – حماه- هـ-229619 33 963+

يبتدأ الإرسال عند كتابة المعطيات المراد إرسالها إلى مسجل معطيات وحدة UART – UDR الموجود في حيز ذاكرة I/O . وتحول المعطيات من المسجل UDR إلى مسجل إزاحة الإرسال TX في إحدى حالتين :

- عند كتابة معطيات الرمز character الجديد للمسجل UDR بعد إزاحة خانة توقف الرمز السابق خارج المرسل TX . فإن مسجل الإرسال سيحمل بشكل فوري .

- أما عند كتابة معطيات الرمز character الجديد للمسجل UDR قبل إزاحة خانة توقف الرمز السابق خارج المرسل TX . فإن مسجل الإزاحة TX سوف لن يحمل بمعطيات الرمز الجديد إلا بعد إزاحة خانة التوقف خارج المرسل TX .

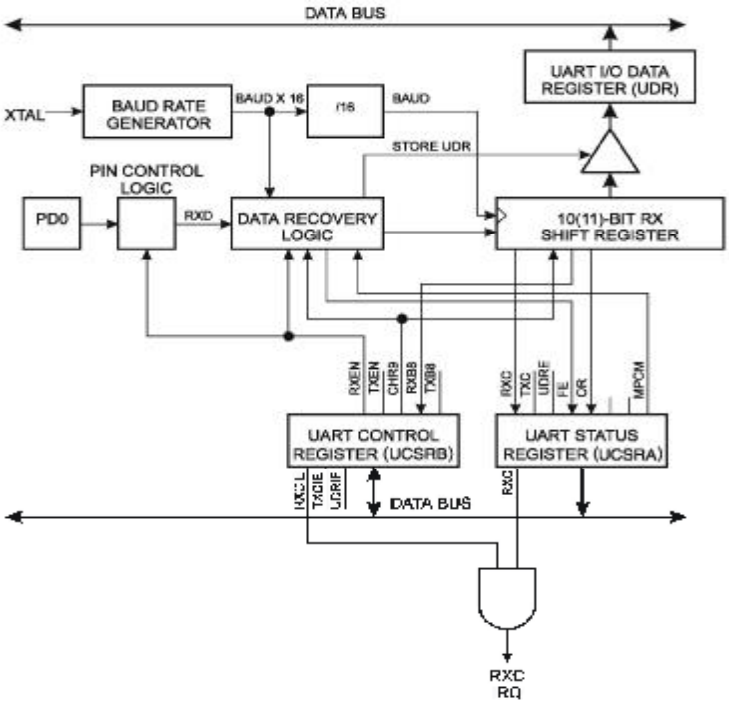
عند نقل المعطيات من المسجل UDR إلى مسجل الإزاحة ، يتم تفعل خانة فراغ مسجل معطيات وحدة UART أي UDRE=1 الموجودة في مسجل حالة وحدة USR – UART . فإذا أصبحت الخانة UDRE=1 فهذا يدل على أن وحدة UART جاهزة لاستقبال رمز جديد . وعند نفس الزمن تحول المعطيات من المسجل UDR إلى خانات مسجل الإزاحة bits (11)10 . بحيث يكون دائماً خانة مسجل الإزاحة bit0=0 (خانة بدء) وتكون الخانة 1= bit9(10) (خانة توقف) . فإذا تم اختيار كلمة المعطيات بطول bit–9 (أي أن الخانة CHR9=1 الواقعة في مسجل تحكم وحدة UART – UCR) عندها تنتقل الخانة TXB8 الموجودة في المسجل UCR إلى الخانة التاسعة في مسجل إزاحة الإرسال .

تتم عملية تحويل المعطيات إلى مسجل الإزاحة TX عند ساعة معدل بود التالية ، وتزاح خانة البدء start bit خارجاً عند القطب TXD (PD1) . ومن ثم تتدفق خانات المعطيات مع الحانة الأولى ذات الأهمية الأقل LSB . وعندما تزاح حانة التوقف خارج المسجل TX ، فإن مسجل الإزاحة سيحمل إذا ما كتبت أي معطيات جديدة على المسجل UDR خلال الإرسال . و أثناء التحميل ، تُفعّل الخانة UDRE=1 . وإذا لم توجد أي معطيات جديدة في المسجل UDR كي ترسل عند إزاحة خانة التوقف خارج مسجل الإرسال TX ، فإن هذا العلم يبقى UDRE=1 إلى أن تكتب المعطيات على المسجل UDR من جديد . فعند عدم كتابة أي معطيات جديدة ، فإن خانة التوقف تبقى ظاهرة على قطب الإرسال TXD (PD1) مما يجعل قيمة القطب TXD=1 ، ويتم تفعيل علم اكتمال الإرسال أي TXC=1 الواقع في مسجل الحالة USR .

يتم تمكين إرسال وحدة UART عند تفعيل خانة تمكين الإرسال XEN=1 . وعند تصفير هذه الخانة TXEN=0 ، فإن قطب النافذة PD1 يستخدم كقطب دخل/خرج للأغراض العامة . وعند تفعيل الخانة TXEN=1 فإن مرسل وحدة UART يتصل مع القطب PD1 وتهمل خانة تحديد الاتجاه DDD1 الموجودة في مسجل اتجاه المعطيات DDRD .

استقبال المعطيات Data Reception

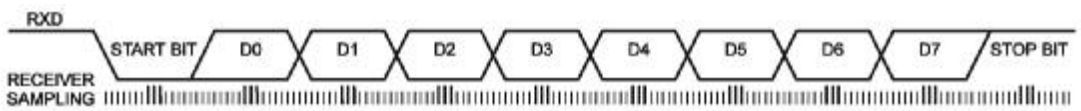
يبين الشكل (41) المخطط الصندوقي لمستقبل UART .



الشكل (41) :**مستقبل وحدة** UART

تؤخذ العينات من الإشارة المطبقة على قطب المستقبل receiver RXD (PD0) عند كل ١٦ نبضة من تردد معدل بودbaud rate . فعندما يكون خط الاستقبال في حالة بطالة idle ، فإن عينة واحدة صفرية المنطق سوف تقاطع هذا النمط وتعتبر كجبهة هابطة لخانة البدء start bit ، وتبدأ عندئذ سلسلة اكتشاف خانة البدء . تشير العينة 1 sample إلى عينة الصفر الأولى أي إلى الانتقال التالي 0 1 ، وتأخذ عينات قطب المستقبل RXD عند أزمنة أخذ العينات 8 , 9 , 10 . فإذا كان عينتان أو أكثر من هذه العينات الثلاث ذات منطق عالي ، تهمل عندئذ خانة البدء وتعتبر كتشويش شوكي spike noise و يبدأ المرسل مرة أخرى بمعاينة انتقال جديد 1إلى 0 على القطب RXD (PD0) .

وعلى كلٍ إذا اكتشفت خانة بدء شرعية VAILD ، فإنه يتم استقبال خانات المعطيات التالية لخانة البدء وتؤخذ العينات من هذه الخانات أيضاً عند أزمنة أخذ العينات 8 , 9 , 10 . وتؤخذ قيمة الخانة إذا وجدت عينتان من ثلاث على الأقل ذات نفس القيمة المنطقية. تزاح كل الخانات إلى مسجل إزاحة الاستقبال ، يبين الشكل (42) هيكلية استقبال الرمز character مع عملية أخذ العينات .



الـ

شكل (37) : **نمذجة المعطيات المستقبلة**

عندما تدخل خانة التوقف إلى المستقبل ، فإنه يجب أن تكون أغلبية العينات الثلاث ذات منطق عالي حتى يتم قبولها كخانة توقف . فإذا كانت عينتان أو أكثر ذات منطق منخفض . يفعل عندئذٍ علم خطأ هيكلية الاستقبال أي FE=1 الموجود في مسجل حالة وحدة USR - UART . ويجب على المبرمج دائماً قبل قراءة مسجل المعطيات UDR فحص العلم FE ليكتشف خطأ هيكلية الاستقبال .

سواءً أكانت خانة التوقف المكتشفة عند نهاية دورة استقبال الرمز شريعة أم لا. فإن المعطيات المستقبلة تحول إلى مسجل المعطيات UDR ويفعل علم اكتمال الاستقبال RXC=1 . في الحقيقة إن المسجلUDR هو عبارة عن مسجلين منفصلين فيزيائياً ، إحداهما للمعطيات المرسلة . والآخر للمعطيات المستقبلة . فعند قراءة مسجل المعطياتUDR ، فإننا نلج accessed لمسجل معطيات المستقبل ، أما عندما نكتب على المسجلUDR فإننا نلجaccessed إلى مسجل معطيات المرسل . إذا تم اختيار طول كلمة المعطيات 9-bit (أي CHR9=1 الموجودة في مسجل تحكم وحدة UART – UCR) ، عندئذ تحمل الخانة RXB8 الموجودة في المسجل UCR بالخانة التاسعة في مسجل إزاحة المستقبل عند تحويل المعطيات إلى المسجل UDR .

بعد استقبال الرمز ، إذا لم نقرأ مسجل المعطياتUDR للاستقبال السابق ، فإن علم التجاوز يُفعل OR=1 و الموجود في مسجل التحكم UCR . هذا يعني أن إزاحة بايت المعطيات السابق إلى مسجل الإزاحة لم يحول إلى مسجل UDR وتم فقده lost . تكون الخانة OR معزولة ومحدثة updated عند قراءة بايت المعطيات الشرعي في مسجل المعطيات UDR . و هكذا فإنه يجب على المبرمج دائماً فحص الخانة OR بعد قراءة المسجل UDR كي يكتشف أي تجاوزات overruns .

يحجب المستقبل عند تصفير خانة تمكين الاستقبال RXEN=0 الموجودة في مسجل التحكم UCR . هذا يعني أن القطب PD0 يستخدم كدخل/خرج للأغراض العامة . وعند تفعيل خانة الاستقبال RXEN=1 ، فإن مستقبل UART يتصل مع القطب PD0 وتهمل وضعية خانة تحديد اتجاه المعطيات DDD0 الموجودة في مسجل اتجاه المعطيات DDRD .

التحكم بوحدة UART UART Control

مسجل المعطيات UDR-UART UART I/O Data Register – UDR

Bit	7	6	5	4	3	2	1	0	
$0C ($2C)	MSB							LSB	UDR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

يتألف مسجل المعطيات UDR فعلياً من مسجلين منفصلين فيزيائياً لهما نفس العنوان في حيز ذاكرة I/O . فعند الكتابة على هذا المسجل ، فإننا نكتب على مسجل معطيات مرسل وحدة UART . أما عندما نقرأ من المسجل UDR فإننا نقرأ مسجل معطيات مستقبل وحدة UART .

مسجل الحالة لوحدة UART USR – UART Status Register–USR

Bit	7	6	5	4	3	2	1	0	
$0B ($2B)	RXC	TXC	UDRE	FE	OR	–	–	–	USR
Read/Write	R	R/W	R	R	R	R	R	R	
Initial value	0	0	1	0	0	0	0	0	

الخانة7 – RXC : خانة اكتمال استقبال وحدة UART :

تصبح هذه الخانة مُفعّلة RXC=1 عند تحويل الرمز المستقبل من مسجل إزاحة المستقبل إلى مسجل المعطيات UDR . وتفعل هذه الخانة بغض النظر عن اكتشاف أي خطأ في هيكلية الاستقبال . و عند تفعيل خانة تمكين مقاطعة اكتمال الاستقبال RXCIE=1 الموجودة في مسجل التحكم UCR . فإن مقاطعة اكتمال استقبال وحدة UART سينفذ عندما RXC=1 . وتصفر خانة اكتمال الاستقبال RXC=0 عند قراءة مسجل المعطياتUDR.

وعند استخدام المعطيات المستقبلة في المقاطعة ، فإنه يجب قراءة مسجل المعطيات UDR في روتين خدمة مقاطعة الاستقبال حتى تصفر الخانة RXC=0 ، وقد تحدث مقاطعة جديدة في نهاية روتين خدمة المقاطعة في الحالات الأخرى .

الخانة 6 – TXC : خانة اكتمال إرسال وحدة UART :

تصبح هذه الخانة TXC=1 عند دخول الرمز character (بما في ذلك خانة التوقف) في مسجل إزاحة الإرسال، وإزاحته نحو الخارج ولا يوجد معطيات جديدة مكتوبة لمسجل المعطيات UDR . وهذا العلم نافعاً بشكل خاص في ملاءمة الاتصالات النصف مضاعفة half-duplex ،حيث تطبيق الإرسال يجب أن يدخل في نمط المستقبل ويحرر ممر الاتصالات بشكل فوري بعد اكتمال الإرسال .

عند تفعيل خانة تمكين مقاطعة اكتمال الإرسال TXCIE=1 الموجودة في مسجل التحكم UCR فإن مقاطعة اكتمال الإرسال ستنفذ عندما TXC=1 . ويقوم كيان المتحكم بتصفير هذه الخانة TXC=0 عند تنفيذ شعاع خدمة المقاطعة المطابق . وبشكل مشابه تصفر الخانة TXC=0 بكتابة المنطق العالي فيها .

الخانة 5 – UDRE : خانة فراغ مسجل المعطيات لوحدة UART :

تفعل هذه الخانة UDRE=1 عند انتقال الرمز المكتوب لمسجل المعطيات UDR إلى مسجل إزاحة الإرسال . و تشير حالة هذه الخانة إلى أن المرسل transmitter جاهزاً لاستقبال رمز جديد لإرساله .

عند تفعيل خانة تمكين مقاطعة فراغ مسجل معطيات وحدة UART أي UDRIE=1 ، فإن روتين مقاطعة اكتمال الإرسال سينفذ طالما أن الخانة UDRE=1 . وتصفر الخانة UDRE=0 عند الكتابة على مسجل المعطيات UDR . عند استخدام المعطيات المرسلة في المقاطعة ، فإنه يجب الكتابة على المسجل UDR في روتين مقاطعة فراغ مسجل معطيات وحدة UART ، وفي كل الحالات الأخرى ، قد تحدث مقاطعة جديدة في نهاية روتين المقاطعة . وخلال تصفير المتحكم MCU تصبح الخانةUDRE=1 لتشير إلى أن المرسل جاهزاً لاستقبال المعطيات .

الخانة 4 – FE : خانة خطأ الهيكلية :

تصبح هذه الخانة FE=1 إذا اكتشفت حالة خطأ في هيكلية الاستقبال . مثلاً : عندما تكون خانة توقف الرمز الوارد صفراً .

تصفر خانة خطأ الهيكلية FE=0 عندما تكون خانة التوقف للمعطيات المستقبلة واحداً .

الخانة : 3 – OR : خانة التجاوز overrun :

تصبح هذه الخانة OR=1 إذا اكتشفت حالة التجاوز ، مثلاً : عند عدم قراءة الرمز السابق الموجود في المسجل UDR قبل إزاحة الرمز الجديد إلى مسجل إزاحة الاستقبال . إن الخانة OR هي عبارة عن ذاكرة و يعني ذلك أنها ستظل واحداً طالما أن المعطيات الشرعية المستقبلة ليست محولة للمسجل UDR .

الخانات0.. 1... 2 – Res : خانات محجوزة :

هذه الخانات هما خانات محجوزة في المتحكم AT90S2313 تقرأ دائماً صفر منطقي .

مسجل التحكم لوحدة UART – UCR

UART Control Register - UCR

Bit	7	6	5	4	3	2	1	0	
$0A ($2A)	RXCIE	TXCIE	UDRIE	RXEN	TXEN	CHR9	RXB8	TXB8	UCR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	W	
Initial value	0	0	0	0	0	0	1	0	

الخانة 7 – RXCIE : خانة تمكين مقاطعة اكتمال الاستقبال RX :

عند تفعيل هذه الخانة RXCIE=1 ، فإن روتين اكتمال الاستقبال سينفذ عندما تصبح الخانة RXC=1 الموجودة في المسجل USR مع كون خانة تمكين المقاطعة العامة I=1 الموجودة في مسجل الحالة SREG .

الخانة 6 – TXCIE : خانة تمكين مقاطعة اكتمال الإرسال TX :

عند تفعيل هذه الخانة TXCIE=1 ، فإن روتين اكتمال الإرسال سينفذ عندما تصبح الخانة TXC=1 الموجودة في المسجل USR مع كون خانة تمكين المقاطعة العامة I=1 الموجودة في مسجل الحالة SREG .

الخانة 5 – UDRIE : خانة تمكين مقاطعة فراغ مسجل المعطيات :

عند تفعيل هذه الخانة UDRIE=1 ، فإن روتين مقاطعة فراغ مسجل المعطيات لوحدة UART سينفذ عندما تصبح الخانة UDRE=1 الموجودة في المسجل USR مع كون خانة تمكين المقاطعة العامة I=1 الموجودة في مسجل الحالة SREG .

الخانة 4 – RXEN : خانة تمكين المستقبل :

إن هذه الخانة تمكن مستقبل وحدة UART عند تفعيلها RXEN=1 . وعند حجب المستقبل ، فإن أعلام الحالة FE , OR , RXC سوف لن تُفعّل للواحد ، أما إذا كانت هذه الأعلام مُفعّلة قبل حجب المستقبل ، فإن عملية حجب المستقبل RXEN=0 سوف لن تؤدي إلى تصفير هذه الأعلام.

الخانة 3 – TXEN : خانة تمكين المرسل :

إن هذه الخانة تمكن مرسل وحدة UART عند تفعيلها أي TXEN=1 . إذا تمت عملية حجب المرسل أثناء إرسال الرمز ، فإن المرسل لن يُحجب إلا بعد إرسال الرمز الموجود في مسجل الإزاحة مع أي رمز آخر موجود في المسجل UDR .

الخانة 2 – CHR9 : خانة الرمز التاسع :

عند تفعيل هذه الخانة CHR9=1 يصبح طول الرمز المرسل و المستقبل 9-bit إضافةً إلى خانة البدء و خانة التوقف . نستخدم عند قراءة أو كتابة الخانة التاسعة كُلٌ من الخانتين RXB8 , TXB8 على الترتيب . كما يمكن استخدام خانة المعطيات التاسعة كخانة توقف إضافية أو كخانة تكافئ parity .

الخانة 1 – RXB8 : خانة المعطيات التاسعة bit8 المستقبلة :

عند تفعيل الخانة CHR9=1 ، فإن الخانة RXB8 تحتوي خانة المعطيات التاسعة للرمز المستقبل .

الخانة 0 – TXB8 : خانة المعطيات التاسعة bit8 المرسلة :

عند تفعيل الخانة CHR9=1 ، فإن الخانة TXB8 تحتوي خانة المعطيات التاسعة للرمز التي سوف ترسل .

مولد معدل بود The BAUD Rate Generator

يعطى معدل بود بالعلاقة التالية : حيث أن : $$BAUD = \frac{fCK}{16(UBRR+1)}$$

- BAUD = معدل بود
- fCK = تردد ساعة الكرستالة
- UBRR = محتويات مسجل معدل بود لوحدة UART، (0-4096) UBRR .

يوضح الجدول (19) تردد الكرستالات القياسية ، لأكثر معدلات بود شيوعاً و استخداماً و المولـدة باسـتخدام المسجل UBRR . إن قيم المسجل UBRR التي تنتج معدل بود الفعلي قد يختلف عن معـدل بـود المطلـوب بحوالي 2% كما هو مبين في الجدول أدناه . و نحن ننصح دائماً باستخدام معدلات بود التي لا يزيد فيها الخطأ عن ١% .

Baud Rate	1 MHz		%Error	1.8432 MHz		%Error	2 MHz		%Error	2.4576 MHz		%Error
2400	UBRR=	25	0.2	UBRR=	47	0.0	UBRR=	51	0.2	UBRR=	63	0.0
4800	UBRR=	12	0.2	UBRR=	23	0.0	UBRR=	25	0.2	UBRR=	31	0.0
9600	UBRR=	6	7.5	UBRR=	11	0.0	UBRR=	12	0.2	UBRR=	15	0.0
14400	UBRR=	3	7.8	UBRR=	7	0.0	UBRR=	8	3.7	UBRR=	10	3.1
19200	UBRR=	2	7.8	UBRR=	5	0.0	UBRR=	6	7.5	UBRR=	7	0.0
28800	UBRR=	1	7.8	UBRR=	3	0.0	UBRR=	3	7.8	UBRR=	4	6.3
38400	UBRR=	1	22.9	UBRR=	2	0.0	UBRR=	2	7.8	UBRR=	3	0.0
57600	UBRR=	0	7.8	UBRR=	1	0.0	UBRR=	1	7.8	UBRR=	2	12.5
76800	UBRR=	0	22.9	UBRR=	1	33.3	UBRR=	1	22.9	UBRR=	1	0.0
115200	UBRR=	0	84.3	UBRR=	0	0.0	UBRR=	0	7.8	UBRR=	0	25.0

Baud Rate	3.2768 MHz		%Error	3.6864 MHz		%Error	4 MHz		%Error	4.608 MHz		%Error
2400	UBRR=	84	0.4	UBRR=	95	0.0	UBRR=	103	0.2	UBRR=	119	0.0
4800	UBRR=	42	0.8	UBRR=	47	0.0	UBRR=	51	0.2	UBRR=	59	0.0
9600	UBRR=	20	1.6	UBRR=	23	0.0	UBRR=	25	0.2	UBRR=	29	0.0
14400	UBRR=	13	1.6	UBRR=	15	0.0	UBRR=	16	2.1	UBRR=	19	0.0
19200	UBRR=	10	3.1	UBRR=	11	0.0	UBRR=	12	0.2	UBRR=	14	0.0
28800	UBRR=	6	1.6	UBRR=	7	0.0	UBRR=	8	3.7	UBRR=	9	0.0
38400	UBRR=	4	6.3	UBRR=	5	0.0	UBRR=	6	7.5	UBRR=	7	6.7
57600	UBRR=	3	12.5	UBRR=	3	0.0	UBRR=	3	7.8	UBRR=	4	0.0
76800	UBRR=	2	12.5	UBRR=	2	0.0	UBRR=	2	7.8	UBRR=	3	6.7
115200	UBRR=	1	12.5	UBRR=	1	0.0	UBRR=	1	7.8	UBRR=	2	20.0

Baud Rate	7.3728 MHz		%Error	8 MHz		%Error	9.216 MHz		%Error	11.059 MHz		%Error
2400	UBRR=	191	0.0	UBRR=	207	0.2	UBRR=	239	0.0	UBRR=	287	-
4800	UBRR=	95	0.0	UBRR=	103	0.2	UBRR=	119	0.0	UBRR=	143	0.0
9600	UBRR=	47	0.0	UBRR=	51	0.2	UBRR=	59	0.0	UBRR=	71	0.0
14400	UBRR=	31	0.0	UBRR=	34	0.8	UBRR=	39	0.0	UBRR=	47	0.0
19200	UBRR=	23	0.0	UBRR=	25	0.2	UBRR=	29	0.0	UBRR=	35	0.0
28800	UBRR=	15	0.0	UBRR=	16	2.1	UBRR=	19	0.0	UBRR=	23	0.0
38400	UBRR=	11	0.0	UBRR=	12	0.2	UBRR=	14	0.0	UBRR=	17	0.0
57600	UBRR=	7	0.0	UBRR=	8	3.7	UBRR=	9	0.0	UBRR=	11	0.0
76800	UBRR=	5	0.0	UBRR=	6	7.5	UBRR=	7	6.7	UBRR=	8	0.0
115200	UBRR=	3	0.0	UBRR=	3	7.8	UBRR=	4	0.0	UBRR=	5	0.0

الجدول (19) : إعداد المسجل UBR عند مختلف الترددات الكرستالية .

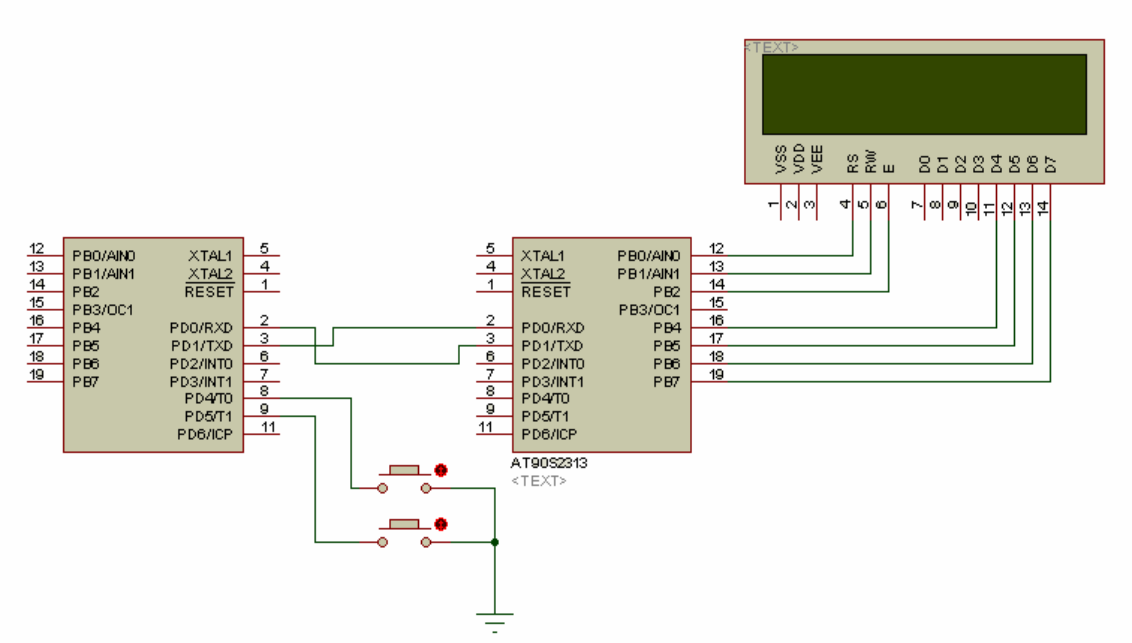
مسجل معدل بود لوحدة UART – UBRR UART Baud Rate Register UBRR – UBRR

Bit	7	6	5	4	3	2	1	0	
$09 ($29)	MSB							LSB	UBRR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

إن مسجل معدل بود UBRR هو عبارة عن مسجل بطول 8-bit نستطيع قراءته و الكتابة عليه ، ومن خلاله نحدد معدل بود لوحدة UART وذلك بما يتطابق مع المعادلة الموجودة أعلاه .

المشروع :

مشروعنا عبارة عن دارتين الأولى تحتوي على المايكرو الأول وعلى كباسين لزيادة ونقصان العدد i ويقوم هذا المايكرو بإرسال العدد i للمايكرو الثاني عبر وصلة الـــUART والذي بدوره يقوم بإظهار هذا الرقم على شاشة الـــ LCD :



البرنامج :

برنامج المعالج الأول:

```
/*****************************************************
Project: UART Transmitter
Chip type           : AT90S2313
Clock frequency     : 4.000000 MHz
*****************************************************/

#include <90s2313.h>
#include <delay.h>

void main(void)
{
unsigned char q , i=0;
PORTB=0x00;
DDRB=0xFF;

PORTD=0x70;
DDRD=0x0E;

// UART initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// UART Receiver: Off
// UART Transmitter: On
// UART Baud rate: 9600
UCR=0x08;
UBRR=0x19;
delay_ms(600);
  UDR=i;

while(1)
 {

 if (PIND.5==0 && q==0)
   {
     i--;
     UDR=i;
     q=1;
   }

  if(PIND.4==0 && q==0)
    {
      i++ ;
      UDR=i;
```

```
    q=1;
  }

  if (PIND.5==1 && PIND.4==1)
  q=0;
 }; // end while

} // end main function
```

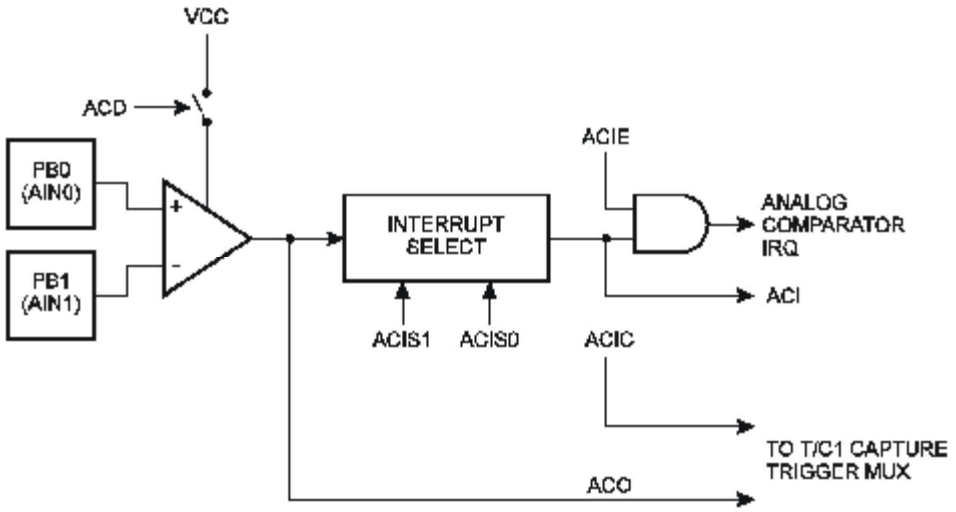
برنامج المعالج الثاني :

```
#include <90s2313.h>
#include <lcd.h>
#include <stdlib.h>
#asm
   .equ __lcd_port=0x18 ;PORTB
#endasm

interrupt [UART_RXC] void uart_rx_isr(void)
{
unsigned char t,*str;
t=UDR;
itoa(t,str);
lcd_clear();
lcd_puts(str);
}


void main(void)
{

PORTB=0x00;
DDRB=0xFF;

PORTD=0x00;
DDRD=0x7E;
lcd_init(16);
                        // UART initialization
                        // Communication Parameters: 8 Data, 1 Stop, No Parity
                       // UART Receiver: On
                       // UART Transmitter: Off
                       // UART Baud rate: 9600
UCR=0x90;
UBRR=0x19;

                        // Global enable interrupts
#asm("sei")

while (1)
    {
                // Place your code here

    };
}
```

المشروع الحادي عشر
المقارن التشابهي

المقارن التشابهي The Analog Comparator

يقوم المقارن التشابهي Analog comparator بمقارنة الدخل المطبقة على قطبيه الموجب PB0(AIN0) والسالب PB1(AIN1) . ويصبح خرج المقارن مرتفعاً AC0=1 عندما يصبح الجهد المطبق على قطبه الموجب AIN0 أكبر من الجهد المطبق على قطبه السالب AIN1 . و بإمكان خرج المقارن أن يقدح وظيفة مدخل المسك للمؤقت/العداد١ . وللمقارن التشابهي شعاع مقاطعة منفصل يتوضع عند العنوان $00A في ذاكرة البرنامج Flash .تحدد طبيعة إشارة خرج المقارن القادحة للمقاطعة من برنامج software المبرمج ، فيمكن تحقيق المقاطعة إما عند الجبهة الهابطة أو الصاعدة أو عند عقدة المنتصف لجهد خرج المقارن . يبن



الشكل (38) المخطط الصندوقي للمقارن التشابهي مع توابعه المنطقية .

الشكل (38) : المخطط الصندوقي للمقارن التشابهي

مسجل الحالة و التحكم بالمقارن التشابهي – ACSR

The Analog Comparator Control and Status Register-ACSR

Bit	7	6	5	4	3	2	1	0	
$08 ($28)	ACD	–	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0	ACSR
Read/Write	R/W	R	R	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	N/A	0	0	0	0	0	

الخانة 7 – ADC : خانة حجب المقارن التشابهي :

يتم قطع الطاقة عن المقارن التشابهي عند تفعيل خانة حجب المقارن التشابهي ACD=1 . و باستطاعتنا تفعيل هذه الخانة في أي لحظة زمنية لقطع التغذية عن المقارن . ويستخدم تفعيل هذه الخانة في أغلب الحالات عندما تكون الطاقة المستخدمة في نمط البطالة حرجة critical . لذلك فإن استيقاظ المتحكم MCU من مقاطعة المقارن التشابهي غير مرغوب فيها . وعندما نرغب في تغيير قيمة الخانة ACD فيجب حجب مقاطعة المقارن التشابهي بتصفير الخانة ACIE=0 الواقعة في المسجل ACSR . و قد تحدث المقاطعة في كل الحالات الأخرى عند تغيير قيمة هذه الخانة ACD .

الخانة 6 - Res : خانة محجوزة :

هذه الخانة هي خانة محجوزة في المتحكم AT90S2313 وتقرأ دائماً صفر منطقي .

الخانة 5 - ACO : خرج المقارن التشابهي :

تتصل هذه الخانة بشكل مباشر مع خرج المقارن التشابهي .

الخانة 4 - ACI : علم مقاطعة المقارن التشابهي :

تصبح هذه الخانة ACI=1 عندما يُحدث خرج المقارن قدح لإحدى أنماط المقاطعة المعرفة حسب قيم الخانتين ACIS0 و ACIS1 . و ينفذ روتين خدمة مقاطعة المقارن التشابهي عند تفعيل كل من الخانتين : ACIE=1 وخانة تمكين المقاطعة العامة I=1 الموجودة في مسجل الحالة SREG . يقوم كيان المتحكم الداخلي بتصفير هذه الخانة ACI=0 عندما ينفذ شعاع خدمة مقاطعة المقارن . وبشكل مشابه تصفر هذه الخانة ACI=0 بكتابة المنطق العالي فيها برمجياً .

الخانة 3 – ACIE : خانة تمكين مقاطعة المقارن التشابهي :

تمكن مقاطعة المقارن التشابهي عندما نفعل كلٍ من خانة تمكين مقاطعة المقارن التشابهي ACIE=1 مع خانة تمكين المقاطعة العامة I=1 الموجودة في مسجل الحالة SREG . وتحجب مقاطعة المقارن التشابهي عند تصفير الخانة ACIE=0 .

الخانة 2 – ACIC : خانة تمكين مدخل مسك المقارن التشابهي :

تقدح وظيفة مدخل المسك في المؤقت/العداد١ من المقارن التشابهي عند تفعيل خانة مدخل مسك المقارن التشابهيACIC=1 . فخرج المقارن في هذه الحالة يتصل بشكل مباشر مع مدخل المسك . وهذا يجعل المقارن يستفيد من رفض التشويش ومزايا اختيار الجبهة لمقاطعة مدخل مسك المؤقت/العداد١ . و عندما نصفر هذه الخانة ACIC=0 فإنه لا يوجد أي اتصال بين المقارن التشابهي وظيفة مدخل المسك . لجعل المقارن يقدح مقاطعة مدخل مسك المؤقت/العداد١ ، فإنه يجب تفعيل خانة تمكين مقاطعة مدخل مسك المؤقت/العداد١ - TICIE1=1 الموجودة في مسجل قناع مقاطعة المؤقت TIMSK .

الخانتان ACIS1، ACIS0 – 1، 2 : خانتي اختيار نمط مقاطعة المقارن التشابهي :

تحدد قيم هاتين الخانتين حوادث خرج المقارن التي ستقدح مقاطعة المقارن التشابهي . والحالات المختلفة لهاتين الخانتين مبنية في الجدول (15) .

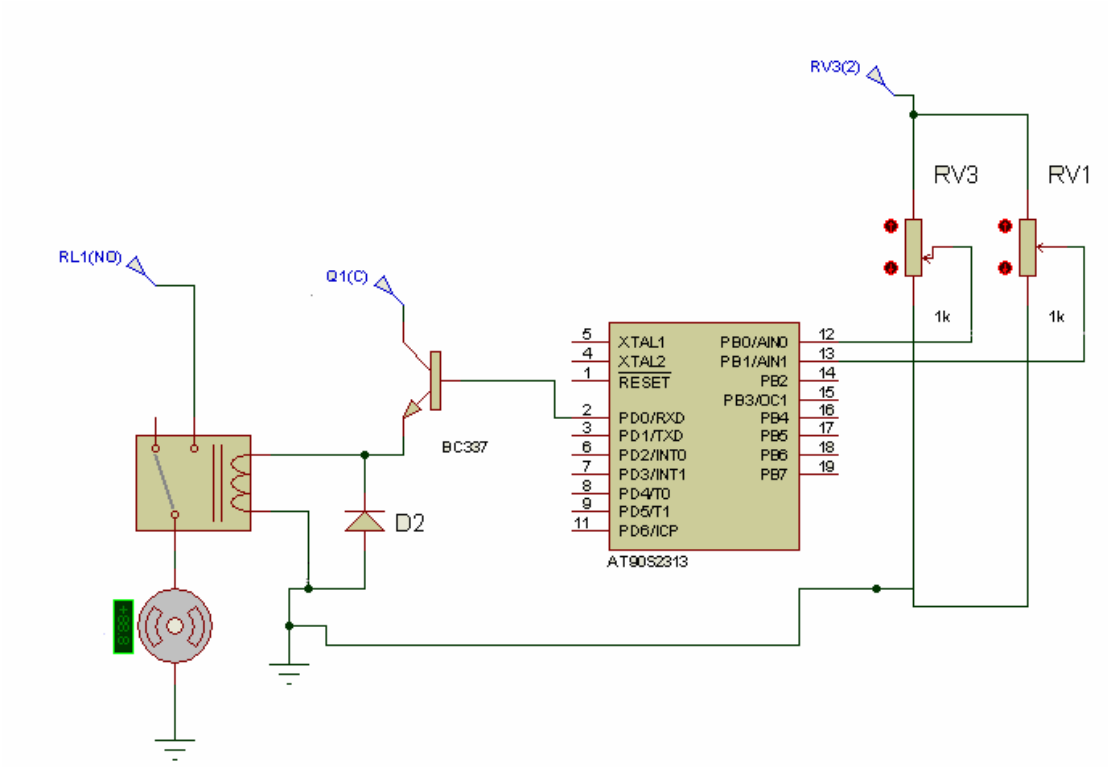
عندما نرغب في تغيير قيم هاتين الخانتين فعلينا أولاً حجب مقاطعة المقارن التشابهي بتصفير خانة تمكين مقاطعة المقارن ACIE=0 الواقعة في هذا المسجل ACSR . وقد تحدث المقاطعة في كل الحالات الأخرى عند تغيير قيم هاتين الخانتين .

ACIS0	ACIS1	نمط المقاطعة
0	0	مقاطعة المقارن عند عقدة المنتصف للخرج
1	0	محجوز
0	1	مقاطعة المقارن عند جبهة الخرج الهابطة
1	1	مقاطعة المقارن عند جبهة الخرج الصاعدة

الجدول (15) : أنماط المقاطعة لقيم الخانتين ACIS1/ACIS0

المشروع :

بفرض أن لدينا مقسمين للجهد موصولين على مداخل المبدل التشابهي الرقمي وأردنا يعمل المحرك ويتوقف كلما تغيرت حالة المبدل التشابهي الرقمي .



دارة المشروع

البرنامج :

```
/*****************************************************
Project: Analog Comparator
Chip type           : AT90S2313
Clock frequency     : 4.000000 MHz
```

```
**************************************************/
#include <90s2313.h>
// Analog Comparator interrupt service routine
interrupt [ANA_COMP] void ana_comp_isr(void)
{
PORTD.0=!PORTD.0;
}

void main(void)
{
PORTB=0x00;
DDRB=0x0C;

PORTD=0x00;
DDRD=0x7F;
// Analog Comparator initialization
// Analog Comparator: On
// Interrupt on Output Toggle
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x08;

// Global enable interrupts
#asm("sei")

while (1)
    {

    };
}
```

المتحكم المصغر ATMEGA8L

المزايا :

يتمتع المتحكم ATMEGA08 (MCU) بالمزايا والمواصفات التالية :

- استخدمت عائلة AVR مزايا بنية RISC المحسنة.
- تتمتع عائلة AVR بالأداء العالي وبطاقة بنية RISC المنخفضة.
  - تحتوي قائمة التعليمات على 130 تعليمة، ينفذ معظمها خلال دورة آلة واحدة.
  - اثنان وثلاثون مسجل عمل تستخدم للأغراض العامة.
  - سرعة التنفيذ تصل إلى 8 مليون تعليمة في الثانية عند تردد قدره 8MHz.
- ذاكرة معطيات وذاكرة برنامج غير قابلة للزوال.

- ذاكرة برنامج وميضية مبنية داخل الشريحة حجمها 4Kbyte قابلة لإعادة البرمجة ضمن النظام .

- الديمومة : 1000 دورة كتابة / مسح.

-ذاكرة معطيات داخلية RAM بطول 128byte.

-ذاكرة معطيات EEPROM بطول 256 bytes .

- أقفال برمجية لحماية كل من ذاكرة البرنامج الوميضية وذاكرة المعطيات .

- **المزايا المحيطية** :
  - مؤقت/عداد بطول 8-bit عدد 2 بمقسم تردد prescaler منفصل بنمط مقارنة واحد .
  - مؤقت/عداد بطول 16-bit بمقسم تردد prescaler منفصل ونمط مقارنة ونمط المسك .
  - ثلاث قنوات PWM .
  - مقارن تشابهي مبني على شريحة المتحكم MCU .
  - نافذة تسلسلية ثنائية الاتجاه UART .
  - نافذة تسلسلية ثنائية الأسلاك 2 Wire I2C .
  - القيادة بالوصلة التسلسلية SPI قائد/ مقاد .
  - ست قنوات للتحويل التشابهي الرقمي ADC بدقة 10-BIT.

مزايا المتحكم الخاصة :

- مزايا المتحكم الخاصة:

- نمط البطالة ذو الطاقة المنخفضة و نمط الطاقة التحتية .

- مصادر المقاطعة داخلية و خارجية.

- دارة تصفير عند وصل الطاقة.

الفارس لتقنيات الحاسوب والإتصالات
سوريا – حماه- هـ-229619 33 963+

- دارة تصفير عند اكتشاف حالة Brown-Out .
- إمكانية اختيار هزاز الشريحة RC الداخلي كمصدر ساعة للمتحكم .

- **الطاقة المستهلكة عند تردد قدره 4MHz و جهد 3V و درجة حرارة 25 C .**
  - في نمط العمل الطبيعي : 3.4mA .
  - في نمط البطالة : 1.4mA .
  - في نمط الطاقة المنخفضة :1µA .
- **أقطاب الدخل و الخرج و الهيكل الخارجي :**
  - ثلاث وعشرون قطب I/O قابلة للبرمجة .
  - شريحة ذات 28 رجلاً موزعة على شكل PDIP و 32 رجلاً موزعة على شكل TQFP .
- **جهود العمل :**
  - يترواح جهد تغذية المتحكم ATMEGA08 ضمن المجال : 4.0 – 6.0V
- **سرعة عمل الهزاز :**
  - للمتحكم ATMEGA08 ضـــمن المجـــال: 0 – 8MHz
- **الطاقة المستهلكة عند تردد قدره 4MHz و جهد 3V و درجة حرارة 25 C :**
  - في نمط العمل الطبيعي : 3.4mA .
  - في نمط البطالة: 1.4mA .
  - في نمط الطاقة التحتية أقل من 1µA .

6.4.2 وصف المتحكم:

إن الشريحة ATMEGA08 هي عبارة عن متحكم Microcontroller ذو 8-bit مصنع بتقنية CMOS ذات الطاقة المنخفضة ، المبني على أساس بنية AVR بمزايا بنية RISC المحسنة . و بالنظر إلى قائمة التعليمات القوية التي يُنفذ معظمها خلال دورة ساعة واحدة ، فإن المتحكم ATMEGA08 ينجز تقريباً مليون تعليمة في الثانية 1MIPS عند تردد قدره 1MHz ، مما يسمح لمصمم النظام باستهلاك الطاقة الأمثلية optimize مقابل سرعة المعالجة.

لقد جمعت نواة AVR ما بين مجموعة التعليمات القوية واثنان و ثلاثون مسجل عمل تستخدم للإغراض العامة . حيث ربطت المسجلات الاثنان والثلاثون مباشرة مع وحدة الحساب و المنطق ALU ، مما سمح بالولوج إلى أي مسجلين من هذه المسجلات المستقلة عند تنفيذ تعليمة واحدة . فالبنية الناتجة أعطت فعالية أكبر لشيفرة التعليمة حيث أصبحت سرعة التنفيذ أكبر بعشرة مرات من بنية متحكمات CISC التقليدية conventional.

زود المتحكم ATMEGA08 بالمزايا التالية : ذاكرة برنامج وميضية Flash قابلة لإعادة التحميل بطول 2K/4K bytes ، ذاكرة معطيات EEPROM بطول 128/256 bytes ، ذاكرة معطيات داخلية SRAM بطول 128 bytes ، عشرون قطب I/O للأغراض العامة ، اثنان و ثلاثون مسجل عمل للأغراض العامة ، مؤقتان/عدادان مرنان بأنماط المقارنة والمسك ، مقاطعات داخلية و خارجية ، نافذة تسلسلية UART قابلة

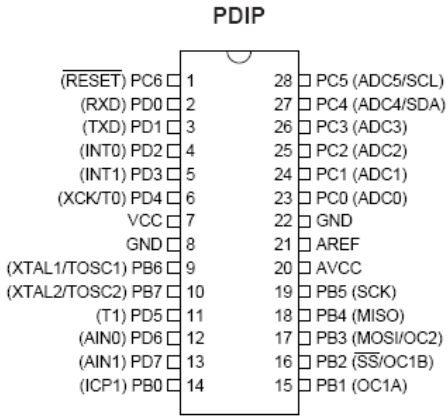
للبرمجة، ست قنوات للتحويل التشابهي الرقمي ADC بدقة 10-bit ، مؤقت مراقبة Watchdog Timer دخله من الهزاز الداخلي للشريحة ، نافذة تسلسلية SPI مخصصة لتحميل البرنامج مع إمكانية اختيار أحد نمطين من أنماط توفير الطاقة برمجياً ، ففي نمط البطالة Idle Mode تتوقف وحدة المعالجة المركزية CPU عن العمل بينما تبقى ذاكرة المعطيات SRAM و المؤقتات/العدادات و النافذة التسلسلية SPI ونظام المقاطعة في حالة عمل . بينما يقوم نمط الطاقة المنخفضة بحفظ محتوى المسجلات ، ولكنه يوقف عمل الهزاز و بالتالي تتشل جميع وظائف الشريحة الأخرى إلى أن تحدث مقاطعة خارجية أو يتم تصفير المتحكم MCU.

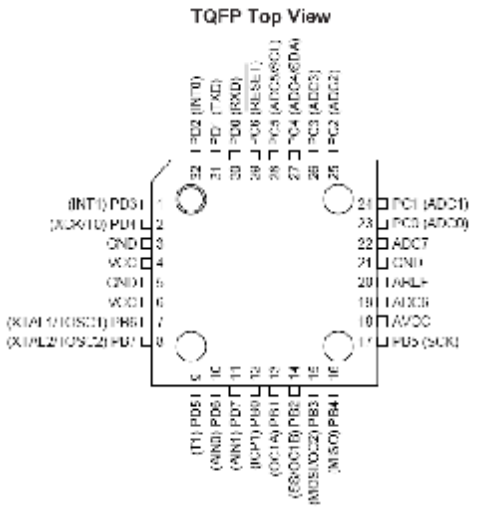
صُنّعت هذه الشريحة باستخدام تقنية ذاكرات Atmel Memory الغير قابلة للزوال ذات الكثافة العالية . مع إمكانية برمجة ذاكرة البرنامج الوميضية Flash المبنية على شريحة المتحكم إما من خلال الوصلة التسلسلية SPI للشريحة ، وإما باستخدام مبرمجة ذاكرات تقليدية . وقد أدى الجمع ما بين معالجات RISC ذات 8-bit مع ذاكرة البرنامج الوميضية Flash القابلة لإعادة البرمجة إلى إنتاج المتحكم ATMEGA08 الذي يتمتع بالقوة و المرونة العالية وبالكلفة المنخفضة للعديد من تطبيقات التحكم المتطورة.

لقد دعم المتحكم AVR ATMEGA08 بشكل كامل من ناحية البرامج و أدوات تطوير النظام مثل : مترجم AVR Studio .

6.4.3 أقطاب المتحكم:

يبين الشكل التالي الشكل (6.1) أقطاب المتحكم ATMEGA08





شرح أقطاب المتحكم PIN Description :

Vcc:قطب جهد التغذية الموجب

GND:قطب جهد التغذية الصفري

النافذة (PB5 ... PB0) Port B :

وهي عبارة عن نافذة دخل/خرج ذات ستة أقطاب ثنائية الاتجاه . وقد زودت أقطاب هذه النافذة بمقاومة رفع داخلية . و بإمكان دارة قيادة خرج النافذةB تقديم تيار قدره 20mA . و عند استخدامها كنافذة دخل ، فإنها ستصبح منبعاً للتيار إذا كانت مقاومات الرفع الداخلية مُمكنة ، وذلك عند ربط أقطاب هذه النافذة خارجياً مع

المنطق المنخفض. و تتمتع أقطاب النافذة B بمزايا خاصة أخرى بعمل المتحكم ATMEGA08 ، وهي مشروحة بالتفصيل في فقرة " نوافذ الدخل/الخرج " ، وعند حدوث إحدى حالات التصفير ، فإن أقطاب هذه النافذة تصبح في حالة الممانعة العالية ، حتى لو كانت الساعة متوقفة عن العمل .

النافذة (PC5 ... PC0) Port C :

وهي عبارة عن نافذة دخل/خرج ذات ستة أقطاب ثنائية الاتجاه . وقد زودت أقطاب هذه النافذة بمقاومة رفع داخلية . و بإمكان دارة قيادة خرج النافذةC تقديم تيار قدره 20mA . و عند استخدامها كنافذة دخل ، فإنها ستصبح منبعاً للتيار إذا كانت مقاومات الرفع الداخلية مُمكنة ، وذلك عند ربط أقطاب هذه النافذة خارجياً مع المنطق المنخفض . و لأقطاب هذه النافذة أيضاً وظائف خاصة أخرى ، كاستخدامها كمداخل تشابهيه للمبدل التشابهي الرقمي ADC . وعند حدوث إحدى حالات التصفير ، فإن أقطاب هذه النافذة تصبح في حالة الممانعة العالية ، حتى لو كانت الساعة متوقفة عن العمل .

النافذة (PD7 ... PD0) Port D :

وهي عبارة عن نافذة I/O ذات ثمانية أقطاب ثنائية الاتجاه مزودة بمقاومة رفع داخلية . و بإمكان دارة قيادة خرج النافذة D تقديم تيار قدره 20mA ، و عند استخدام هذه النافذة كنافذة دخل فإنها تصبح منبعاً للتيار (IIL) عند وصلها خارجياً مع المنطق المنخفض إذا كانت مقاومات الرفع الداخلية مُفعّلة .و تتمتع أقطاب النافذة D بمزايا خاصة أخرى بعمل المتحكم ATMEGA08 ، وهي مشروحة بالتفصيل في فقرة " نوافذ الدخل/الخرج " ، وعند حدوث إحدى حالات التصفير ، فإن أقطاب هذه النافذة تصبح في حالة الممانعة العالية ، حتى لو كانت الساعة متوقفة عن العمل .

قطب التصفير RESET :

مدخل التصفير . يولد التصفير الخارجي عند تطبيق نبضات كهربائية أطول من 50 ns ذات منطق منخفض على القطب RESET . وذلك حتى إن لم تكن الساعة في حالة عمل . و النبضات الأقصر من ذلك قد لا تؤدي بالضرورة إلى توليد التصفير .

مدخل الهزاز XTAL1 :

هو عبارة عن مدخل مضخم الهزاز العاكس وهو أيضاً مدخل لدارة تشغيل الساعة الداخلية .

مدخل الهزاز XTAL2 :

هو عبارة عن مخرج مضخم الهزاز العاكس .

:AVCC

وهو قطب جهد تغذية التغذية للمبدل التشابهي الرقمي ADC . ويجب وصله خارجياً مع القطب VCC من خلال مرشح تمرير منخفض . راجع فقرة " عمل المبدل ADC " .

: AREF

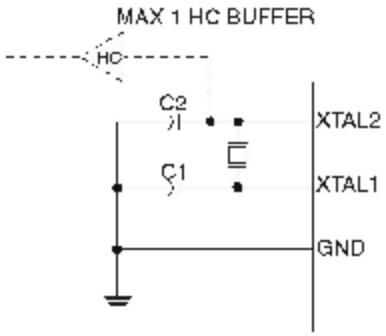
و هو قطب دخل الجهد المرجعي التشابهي للمبدل ADC . و يجب أن تتراوح قيمة هذا الجهد عند عمل المبدل ADC ما بين (2.7V...AVCC) .

:AGND

إذا كانت الدارة التي تقوم بتصميمها لها أرضي تشابهي مستقل ، فيجب ربط هذا القطب مع هذا الأرضي . و في الحالات نربط هذا القطب مع أرضي شريحة المتحكم GND .

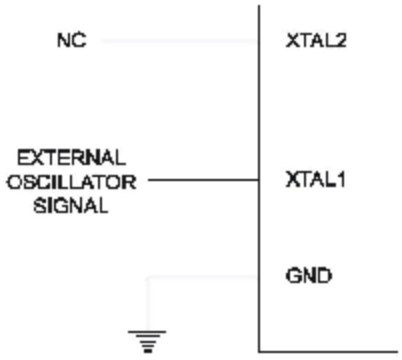
الهزاز الكرستالي Crystal Oscillator :

إن القطبين XTAL1 وXTAL2 هما مدخل ومخرج الهزاز العاكس على الترتيب . فعندما نربط بلورة كرستالية أو مهتز سيراميكي مع هذين القطبين ، فإنه يتم استخدام



الشكل (6.2):توصيلات الهزاز الكريستالي

الهزاز الداخلي لتأمين نبضات الساعة للمتحكم MCU ، كما هو مبين في الشكل (٦-٢) . وعندما نرغب في قيادة الشريحة من مصدر ساعة خارجي ، فإننا نهمل القطب XTAL2 و نقوم بوصل الساعة الخارجية مع القطب XTAL1 ، كما هو مبين في الشكل (٦-٣) :



الشكل (6.3) : قيادة المتحكم من ساعة خارجية

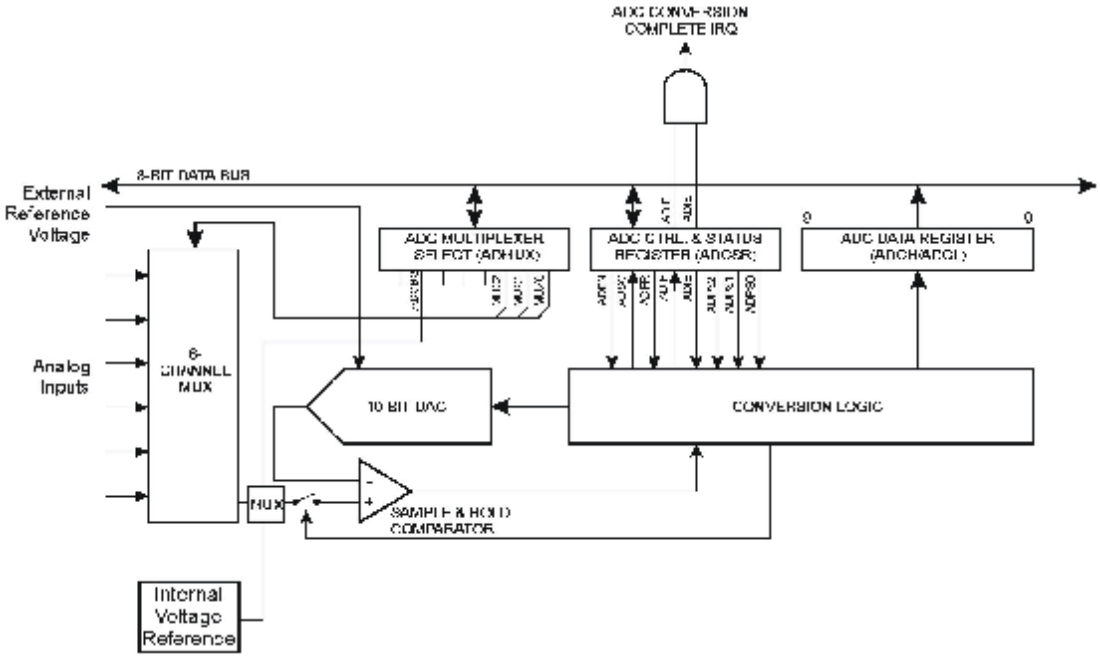
ملاحظة : عند استخدم هزاز المتحكم كساعة لجهاز خارجي آخر ، فإن إشارة الساعة المأخوذة من القطب XTAL2 يجب أن توصل مع دارة قيادة عالية السرعة من النوع HC .

المشروع الثاني عشر
المبدل التشابهي الرقمي

المبدل التشابهي الرقمي Analog to Digital Converter

يتمتع المبدل التشابهي الرقمي ADC بالمزايا التالية :

- محول بطول 10-bit
- الدقة المطلقة ± 2 LSB
- اللاخطية التكاملية 0.5 LSB
- زمن التحويل 65-260µs
- أكثر من 15 KSPS عينة في الثانية
- ناخب لست قنوات دخل تشابهية
- مجال الدخل من القمة إلى القمة
- نمط العمل الحر و نمط التحويل المفرد
- مقاطعة عند اكتمال تحويل ADC



- رفض ضجيج نمط النوم

رفض ضجيج نمط الشكل (44) : المخطط الصندوقي لدارة المبدل التشابهي الرقمي ADC

يتميز المتحكم ATMEGA8L بأنه مزود بمبدل تشابهي رقمي ADC يعمل عل طريقة التقريب المتتالي بدقة 10-bit . و يسمح الناخب التشابهي بربط المبدل ADC مع أقطاب النافذة C الست فنحصل بالتالي على ست قنوات تشابهيه . كما يحتوي المبدل التشابهي الرقمي ADC على آخذ عينات و مضخم ماسك لضمان مسك جهد دخل المبدل عند مستوى ثابت أثناء عملية التحويل . و قد بينا على الشكل (44) المخطط الصندوقي لدارة المبدل ADC . و للمبدل ADC قطبي تغذية تشابهيان مستقلان هما AVCC و AGND . حيث يربط القطب AGND مع أرضي المتحكم العام GND و يجب أن لا يختلف جهد القطب AVCC أكثر من 0.3V ± عند جهد التغذية VCC . راجع فقرة " تقنية رفض ضجيج المبدل ADC " كي تتعرف على كيفية وصل هذه الأقطاب . و للمبدل ADC جهد مرجعي خارجي يطبق على القطب AREF و يقع مجال هذا الجهد ما بين AGND-AVCC .

العمل Operation

يعمل المبدلADC بأحد نمطي عمل و هما : نمط التحويل المفرد ، و نمط العمل الحر . ففي نمط التحويل المفرد سيهيئ المبرمج كل عملية تحويل . أما في نمط العمل الحر ، فيتم أخذ عينات من إشارة الدخل التشابهيه و تحويلها إلى قيمة رقمية بشكل ثابت و يتم أيضاً تحديث قيم مسجلي معطيات المبدل ADC . وتقوم الخانة ADFR الموجودة في المسجل ADCSR باختيار أحد نمطي العمل .

يقوم المسجل ADMUX بتحديد أيٍ من قنوات المداخل التشابهية الستة ستوصل كمدخل للمبدل ADC . و يتم تأهيل المبدل ADC بكتابة المنطق العالي (1) في خانة تمكين المبدل ADC ، أي تفعيل الخانة ADEN=1 الموجودة في مسجل التحكم و الحالة ADCSR . إن التحويل الأول هو التحويل الذي يأتي بعد بداية تمكين المبدل ADC مباشرة ، و سوف يسبق بتبديل زائف dummy . و بالنسبة للمبرمج ، فإن الاختلاف الوحيد في هذا التبديل هو أنه يأخذ أكثر من ١٢ دورة ساعة عن عملية التحويل الطبيعية .

تبدأ عملية التحويل بكتابة المنطق العالي (1) على خانة بدء التحويل للمبدل ADC ، أي تفعيل الخانة ADSC=1 الموجودة في مسجل التحكم و الحالة ADCSR . و ستبقى هذه الخانة في المنطق العالي طالما أن عملية التحويل مستمرة ، و يقوم الكيان الداخلي بتصفير هذه الخانة ADSC=0 عند اكتمال عملية التحويل . و إذا اختلفت قناة المعطيات المختارة أثناء تقدم عملية التبديل ، فإن المبدل سينهي عملية التبديل الحالية قبل إنجاز تغيير القناة .

يتوضع ناتج عملية التحويل للمبدل ADC الذي طوله 10-bit في المسجلين ACDH و ACDL . و يجب علينا قراءة هذين المسجلين للحصول على الناتج عند اكتمال عملية التحويل . وعلينا استخدام منطق خاص لحماية المعطيات حتى نضمن أن محتويات مسجلي المعطيات ACDH و ACDL تعود إلى نفس عملية التحويل عند القراءة . و عمل هذه التقنية هو كالتالي :

عند قراءة المعطيات ، فيجب قراءة المسجل ADCL أولاً ، لأن الولوج لمسجلات معطيات ADC يكون ككتلة واحدة أثناء قراءة المسجل ADCL . هذا يعني إذا تم قراءة المسجل ADCL و اكتملت عملية التحويل قبل قراءة المسجل ADCH ، فلن تُحدث المسجلات و سنفقد ناتج عملية التحويل . وعند قراءة المسجل ADCH ، فإن الولوج لمسجلات المبدل ACDH و ACDL يتم بإعادة التأهيل re-enabled .

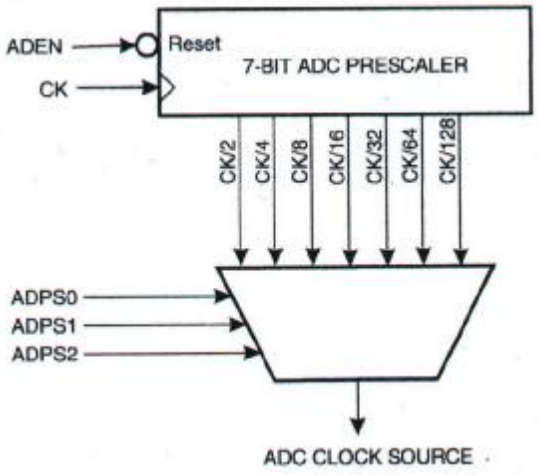
و للمبدل ADC علم مقاطعة ADIF ، الذي يقدح المقاطعة عند اكتمال عملية التحويل . و تحجب عملية الولوج لمسجلي معطيات المبدل ADC بين قراءة المسجل ADCH و ADCL ،و ستقدح المقاطعة حتى لو خسرنا الناتج.

المقسم الزمني للمبدل ADC ADC Prescaler

يحتوي المبدل ADC على مقسم زمني prescaler ، الذي يقوم بتقسيم ساعة النظام إلى ترددات دخل مقبولة كساعة للمبدل ADC . ويقبل المبدل ADC ترددات ساعة ضمن المجال 50-200 kHz . و تطبيق ترددات ساع أعلى من ذلك سينتج دقة أقل . راجع جدول " خصائص المبدل ADC " .

تستخدم الخانات ADPS0, ADPS1, ADPS2 الواقعة في مسجل التحكم والحالة ADCSR لتوليد ترددات الدخل المناسبة لساعة المبدل ADC من ترددات الساعةXTAL الأعلى من 100 kHz . و يبدأ المقسم بالعد من لحظة تشغيل المبدل ADC أي عند تفعيل الخانة ADEN=1 الموجودة في المسجل ADCSR . ويبقى المسجل في حالة عمل طالما أن ADEN=1 ، و يصفر في اللحظة التي تصبح فيها الخانة ADEN=0 .

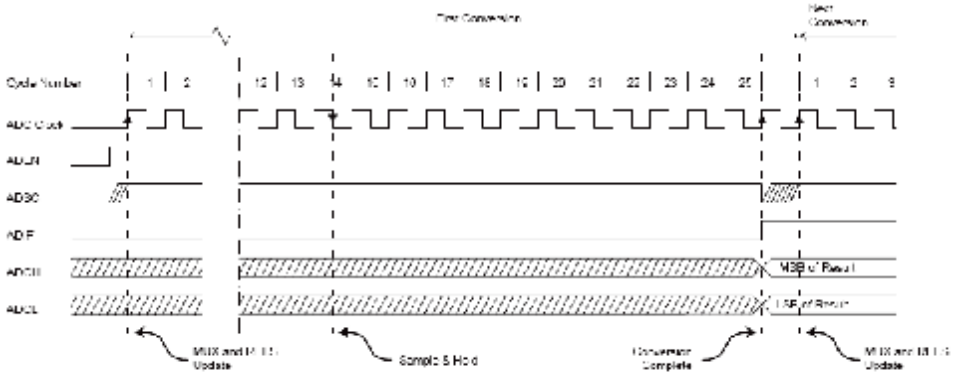
تبدأ عملية التحويل عند تفعيل الخانة ADSC=1 الموجودة في المسجل ADCSR ، و يبدأ التحويل ند الجبهة الصاعدة التالية لدورة ساعة المبدل ADC . و تأخذ عملية أخذ العينة من إشارة الدخل و مسكها فترة 1.5 من دورة ساعة المبدل من بدء التحويل .و تصبح النتيجة جاهزة و مكتوبة على مسجل معطيات ADC بعد 13 دورة. وفي نمط التحويل المفرد ، فإن المبدل ADC يحتاج إلى دورة ساعة إضافية قبل أن يبدأ بعملية تحويل جديدة .أنظر الشكل (47) .فإذا أصبحت الخانة ADSC=1 في هذه الدورة ، فإن المبدل ADC سيبدأ عملية تحويل جديدة مباشرة .أما في نمط العمل الحر ، فإن عملية التحويل الجديدة ستبدأ مباشرة بعد كتابة النتيجة على مسجلي معطيات المبدل ADC . و عندما نستخدم تردد ساعة مقداره 200 kHz للمبدل في نمط العمل الحر ، فإن ذلك سيعطي أقل زمن تحويل 56 µs و يكافئ ذلك أخذ 15.4 kSPS عينة في الثانية . وقد بينا بشكل مختصر أزمنة عملية التحويل في الجدول (21) .



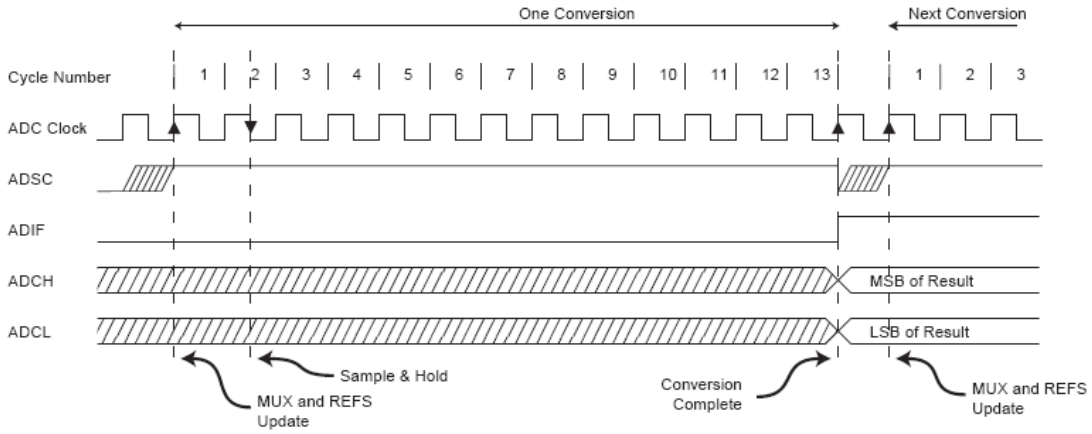
الشكل (45) : مقسم المبدل ADC

الحالة	عدد دورات العينة	جاهزية الناتج (عدد الدورات)	زمن التحويل الإجمالي (الدورات)	زمن التحويل الإجمالي (µs)
دورة تحويل في النمط الحر	14	25	25	125-500
دورة التحويل المفرد	14	25	26	130-520
تحويل النمط الحر	2	13	13	65-260
تحويل النمط المفرد	2	13	13	70-280

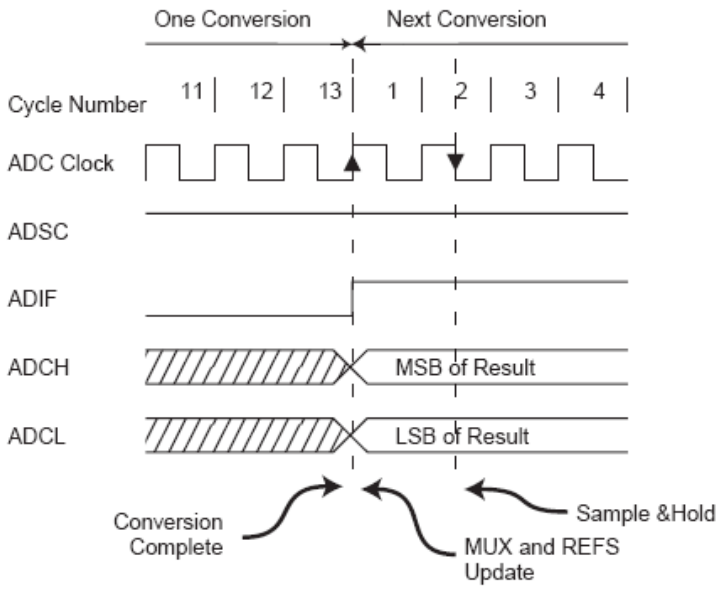
الجدول (21) : أزمنة تحويل المبدل ADC



الشكل (46) : مخطط تزامن المبدل ADC ، التحويل الأول (نمط التحويل المفرد)



الشكل (47) : مخطط تزامن المبدل ADC ،نمط التحويل المفرد



الشكل (48) : مخطط تزامن المبدل ADC ، نمط التحويل العمل الحر

وظيفة رافض ضجيج المبدل ADC ADC Noise Canceller Function

يتميز رافض ضجيج المبدل ADC بأنه يسمح بإجراء عملية التحويل أثناء نمط البطالة idle ليقلل الضجيج الناتج عن نواة وحدة المعالجة المركزية CPU . فإذا أردنا استخدام هذه المميزة ، فيجب علينا إتباع الإجراء التالي :

١. لضمان أن المبدل ADC مؤهل و غير مشغول بالتحويل ، فيجب اختيار نمط التحويل المفرد ، وكذلك تأهيل مقاطعة اكتمال تحويل المبدل ADC . وذلك :

ADEN=1
ADSC=0
ADFR=0
ADIE=1

٢. الدخول في نمط البطالة . يبدأ المبدل ADC بالتحويل في حين تتوقف وحدة المعالجة المركزية CPU .

٣. إذا لم تحدث أي من المقاطعات الأخرى قبل إتمام تحويل المبدل ADC ، فإن مقاطعة المبدل ADC ستوقظ المتحكم MCU من نومه كي ينفذ روتين خدمة مقاطعة إتمام تحويل ADC .

مسجل اختيار ناخب المبدل ADC ADC Multiplexer Select Register-ADMUX

Bit	7	6	5	4	3	2	1	0	
$07 ($27)	-	ADCBG	-	-	-	MUX2	MUX1	MUX0	ADMUX
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

الخانة 7 – Res : خانة محجوزة

هذه الخانة هي خانة محجوزة في المتحكم ATMEGA8L و تقرأ دائماً صفر منطقي .

الخانة 6 – ADCBG : خانة اختيار حزمة جهد لدخل المبدل ADC

عند تفعيل هذه الخانة ADCBG=1 و تأهيل الخانة BOD (برمجة خانة الدمج BODEN) ، فإن حزمة جهد ثابتة قيمتها 0.05V ± 1.22 ستحل محل جهد الدخل الطبيعي للمبدل ADC . و عند تصفير هذه الخانة ADCBG=0 فإن جهد أحد أقطاب الدخل الخارجية (حسب قيمة الخانات MUX2…MUX0) سيطبق على مدخل المبدل ADC .

الخانة Res – 5…3 : خانات محجوزة

هذه الخانات هي خانات محجوزة في المتحكم ATMEGA8L و تقرأ دائماً صفر منطقي .

الخانات MUX2…MUX0 - 2…0 : خانات اختيار قناة الدخل التشابهية

تحدد قيم هذه الخانات الثلاث أيٍ من المداخل التشابهيه الست 0-5 سيتصل مع دخل المبدل ADC .

مسجل التحكم والحالة بالمبدل ADC ADC Control and Status Register-ADCSR

Bit	7	6	5	4	3	2	1	0	
$06 ($26)	ADEN	ADSC	ADFR	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

الخانة ADEN - 7 : خانة تأهيل المبدل ADC

يتم تأهيل المبدل ADC عند كتابة المنطق العالي على الخانة ADEN=1 ، و يتوقف المبدل ADC عن العمل بكتابة المنطق المنخفض على الخانة ADEN=0 . و عند إيقاف المبدل و هو في حالة تحويل ، فإنه سينهي عملية التحويل هذه .

الخانة ADSC – 6 : خانة بداية عملية التحويل للمبدل ADC

في نمط التحويل المفرد ، يجب كتابة المنطق العالي على الخانة ADSC=1 عند بداية كل عملية تحويل . أما في نمط العمل الحر ، فيجب كتابة المنطق العالي على الخانة ADSC=1 عند بداية عملية التحويل الأولى . ففي المرة الأولى للكتابة على الخانة ADSC بعد تأهيل المبدل ADC ، أو عند الكتابة على الخانة في نفس لحظة تأهيل المبدل ADC ، فإن تحويل زائف dummy سيسبق التحويل الأول الحقيقي ، و يقوم التحويل الزائف بتحقيق التهيئة للمبدل ADC .

تبقى الخانة ADSC=1 أثناء عملية التحويل . وتصبح الخانة ADSC=0 عند إتمام عملية التحويل ، ولكن قبل كتابة الناتج على مسجلي معطيات المبدل ADC . و هذا يسمح ببداية ملية تحويل جديدة قبل بداية عملية التحويل الحالية . و تبدأ عملية التحويل الجديدة بالعمل مباشرة بعد إتمام عملية التحويل الحالية . و كما ذكرنا سابقاً أن عملية تحويل زائفة ستسبق أول عملية تحويل حقيقية ، فخلال ذلك ستبقى الخانة ADSC=1 إلى أن تكتمل عملية التحويل الحقيقية .

ليس لكتابة الصفر على هذه الخانة أي تأثير .

الخانة ADFR 5 : خانة اختيار نمط العمل الحر للمبدل ADC

يعمل المبدل ADC في نمط العمل الحر عند كتابة المنطق العالي على الخانة ADFR=1 ، حيث تؤخذ العينات من الإشارة التشابهية المطبقة على دخل المبدل ADC وتحدث معطيات المسجلين ADCH و ADCL بشكل مستمر. و ينتهي نمط العمل الحر بكتابة المنطق المنخفض على الخانة ADFR=0 .

الخانة ADIF – 4 : علم مقاطعة المبدل ADC

يصبح هذا العلم فعالاً ADIF=1 عند إتمام عملية تحويل المبدل ADC و تحديث مسجلي معطياته . و يتم الانتقال إلى تنفيذ روتين خدمة مقاطعة إتمام تحويل ADC إذا كان كلٍ من الخانتين فعالاً : خانة تمكين مقاطعة المبدل ADIE=1 و خانة تمكين المقاطعة العامة I=1 الموجودة في مسجل الحالة SREG . و يقوم الكيان الداخلي بتصفير هذه الخانة ADFI=0 عند معالجة شعاع المقاطعة المطابق . و بشكل مشابه يمكن تصفير هذا العلم بكتابة المنطق العالي عليه .

الخانة 3 – ADIE : خانة تمكين مقاطعة المبدل ADC

تؤهل مقاطعة إتمام تحويل المبدل ADC عند تفعيل كلٍ من الخانتين : خانة تمكين مقاطعة المبدل ADIE=1 مع خانة تمكين المقاطعة العامة I=1 .

الخانات 2...0 – ADPS2, ADPS1, ADPS0 : خانات اختيار مقسم المبدل ADC

تحدد هذه الخانات معامل التقسيمة بين تردد ساعة النظام XTAL و ساعة دخل المبدل ADC . كما هو مبين في الجدول التالي :

معامل التقسيم	ADPS2	ADPS1	ADPS0
2	0	0	0
2	0	0	1
4	0	1	0
8	0	1	1
16	1	0	0
32	1	0	1
64	1	1	0
128	1	1	1

الجدول (22) : اختيار نسبة التقسيم للمبدل ADC

مسجل معطيات المبدل ADC - ADCL and ADCH

Bit	15	14	13	12	11	10	9	8	
$05 ($25)	-	-	-	-	-	-	ADC9	ADC8	ADCH
$04 ($24)	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0	ADCL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
Initial value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

ADC Data Register-ADCL and ADCH

يتوضع الناتج عند إتمام عملية التحويل في هذين المسجلين ، و من الأمور الأساسية في نمط العمل الحر قراءة هذين المسجلين ، على أن نقرأ المسجل ADCL قبل قراءة المسجل ADCH .

مسح مضاعف القنوات Scanning Multiple Channels

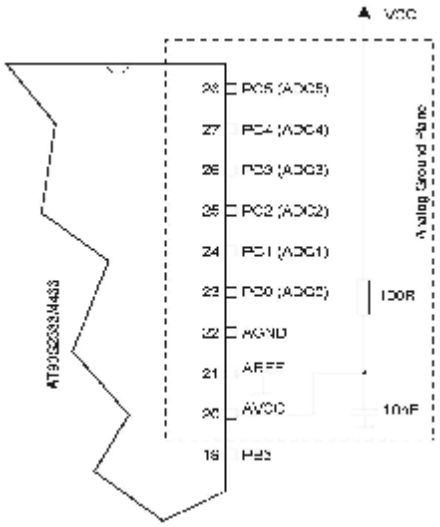
إن تغيير القناة التشابهية دائماً يتأخر حتى تنتهي عملية التحويل للقناة الحالية . و نستطيع استخدام نمط العمل الحر لمسح عدد مضاعف للقنوات من دون مقاطعة عمل المبدل ADC . و يتم ذلك باستخدام مقاطعة إتمام عملية تحويل المبدل ADC لإنجاز إزاحة القناة . و على كلٍ على المبرمج أخذ الحقيقة التالية بعين الاعتبار :

كما نعلم أن مقاطعة المبدل ADC تُقدح عندما يصبح ناتج عملية التحويل جاهز للقراءة . و في نمط العمل الحر تبدأ عملية التحويل التالية مباشرة بعد قدح المقاطعة . فإذا قمنا بتغيير محتوى المسجل ADMUX بعد قدح المقاطعة، فإن عمليات التحويل التالية ستأخذ بعين الاعتبار القيمة الجديدة في هذا المسجل و التي تعبر عن قناة دخل تشابهية جديدة ستطبق على المبدل التشابهي الرقمي ADC .

تقنية رفض ضجيج المبدل ADC ADC Noise Canceling Techniques

تولد الدارة الرقمية الداخلية الخارجية للمتحكم ATMEGA8L حقل EMI . و قد يؤدي ذلك إلى التأثير على دقة القياس التشابهي . فإذا كانت دقة القياس مهمة بالنسبة لنا ، فإننا نستطيع تقليل بتطبيق التقنية التالية :

١. إن القسم التشابهي الخاص بالمتحكم ATMEGA8L و كل العناصر التشابهية في التطبيق يجب أن يتمتع بأرضي تشابهي منفصل على الدارة المطبوعة PCB . و يجب وصل هذا الأرضي مع الأرضي الرقمي من خلال نقطة وحيدة على الدارة المطبوعة PCB .
٢. يجب أن نكون حريصين أن يكون مسار الإشارة التشابهية أقصر ما يمكن .
٣. يجب وصل قطب جهد التغذية للمبدل AVCC مع قطب جهد التغذية الرقمي VCC من خلال شبكة RC كما هو مبين في الشكل (49) .
٤. استخدام وظيفة رافض ضجيج المبدل ADC لتقليل ضجيج وحدة المعالجة المركزية CPU .
٥. إذا استخدم بعض أقطاب النافذة C كمخارج رقمية ، فمن الأمور الأساسية عندئذ عدم تبديل هذه الأقطاب عندما عملية التحويل تكون في حالة معالجة .



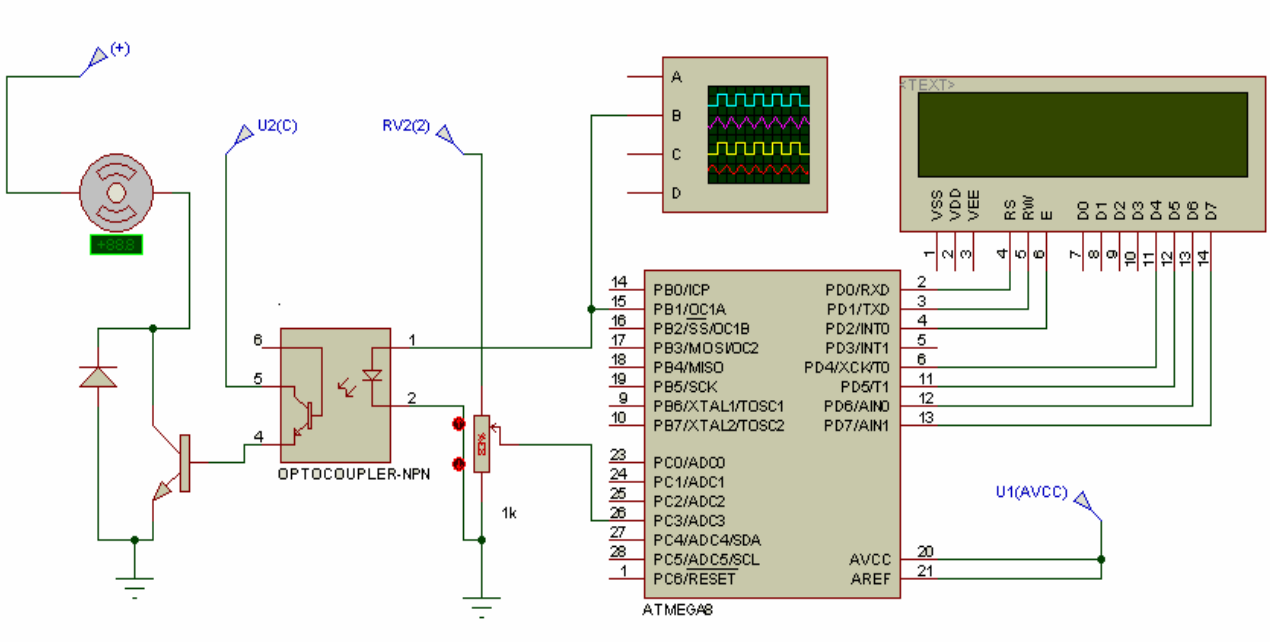
الشكل (49) : توصيلات الطاقة للمبدل ADC

الوحدة	Max	Typ	Min	الحالة	البارامتر	الرمز
Bits		10			التصميم	
LSB	2	1		VREF=4V ADCclock=200kHz	الدقة المطلقة	
LSB		4		VREF=4V ADCclock=1MHz	الدقة المطلقة	
LSB		16		VREF=4V ADCclock=2MHz	الدقة المطلقة	
LSB		0.5		VREF>2V	اللاخطية التكاملية	
LSB		0.5		VREF>2V	اللاخطية التفاضلية	
LSB		1			الخطأ الصفري (الإزاحة)	
µs	260		65		زمن التحويل	
kHz	200		50		تردد الساعة	
V	VCC+0.3		VCC -0.3		جهد التغذية التشابهي	AVCC
V	AVCC		AGN D		الجهد المرجعي	VREF
kΩ	13	10	6		مقاومة الدخل المرجعية	RREF
MΩ		100			مقاومة الدخل التشابهية	RAIN

خصائص المبدل ADC عند TA= -40 C TO 85 C

المشروع :

قيادة محرك تيار مستمر باستخدام تقنية الـــ PWM باستخدام المبدل التشابهي الرقمي لتغيير السرعة وإظهار ناتج عملية التبديل على شاشة LCD.



البرنامج :

```
#include <mega8.h>
#include <stdlib.h>
#include<delay.h>
// Alphanumeric LCD Module functions
#asm
   .equ __lcd_port=0x12 ;PORTD
#endasm
#include <lcd.h>

#define ADC_VREF_TYPE 0x03
unsigned int adc_data;
unsigned char *str;

// ADC interrupt service routine
interrupt [ADC_INT] void adc_isr(void)
{
// Read the AD conversion result
adc_data=ADCW;
}

void main(void)
```

الفارس لتقنيات الحاسوب والإتصالات
سوريا – حماه- هـ-229619 33 963+

```
{
PORTB=0x00;
DDRB=0xFF;

PORTC=0x00;
DDRC=0x00;

PORTD=0x00;
DDRD=0xFF;
// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: 62.500 kHz
// Mode: Fast PWM top=03FFh
// OC1A output: Inverted
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0xC3;
TCCR1B=0x0B;

OCR1AH=0x03;
OCR1AL=0xff;

OCR1BH=0x00;
OCR1BL=0x00;

// ADC initialization
// ADC Clock frequency: 250.000 kHz
// ADC Voltage Reference: AVCC pin
ADMUX=ADC_VREF_TYPE;
ADCSRA=0xAC;
ADCSRA.6=1;
// LCD module initialization
lcd_init(16);
// Global enable interrupts
#asm("sei")
while (1)
      {
       lcd_clear();
       itoa(adc_data, str);
       lcd_puts(str);
       OCR1AH=ADCH;
       OCR1AL=ADCL;
       delay_ms(100);
            };
}
```

المشروع الثالث عشر

النافذة المحيطية التسلسلية SPI في عائلة المتحكمات AVR

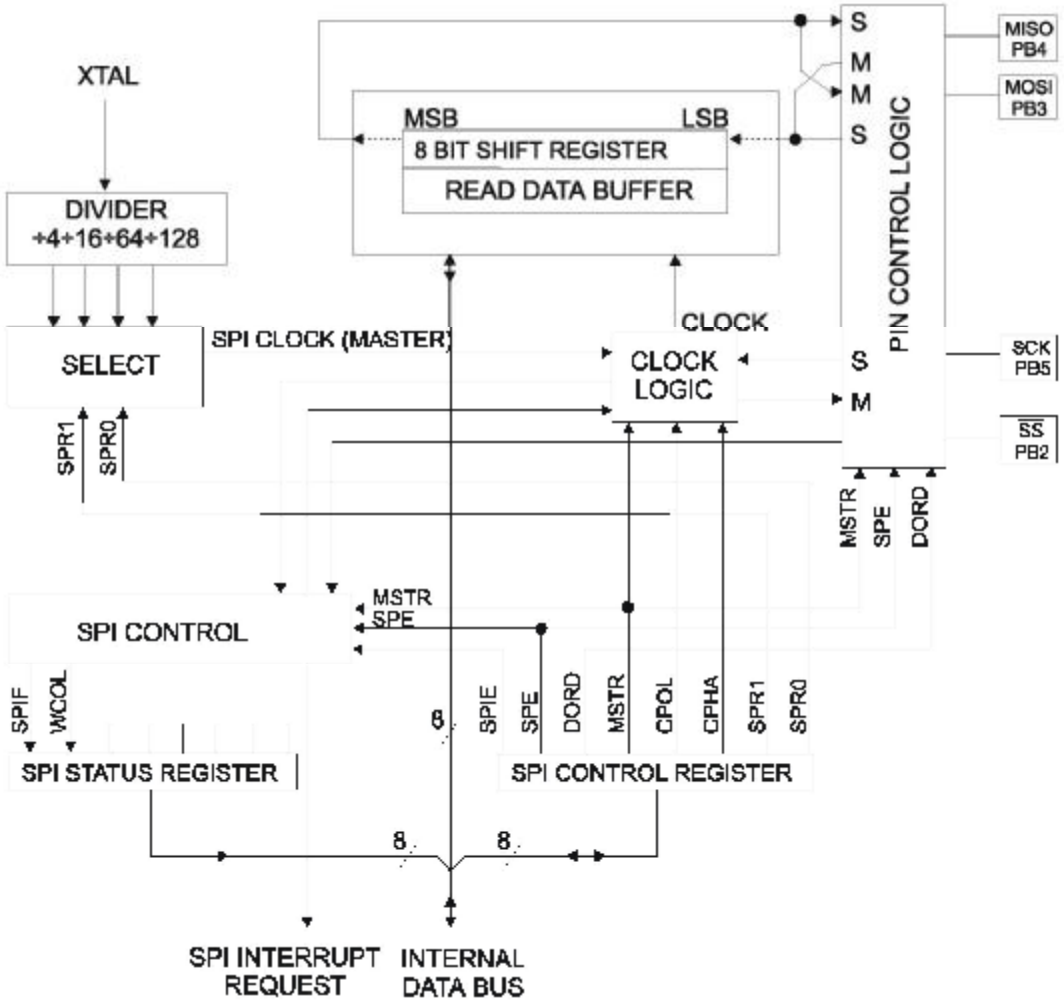
AVR Serial Peripheral Interface-SPI

تسمح النافذة المحيطية التسلسلية SPI بنقل المعطيات بشكل متزامن بسرعة عالية بين المتحكم و الأجهزة المحيطية ، أو بين عدة شرائح AVR . وتتمتع النافذة بالمزايا التالية :

- نقل المعطيات بشكل متزامن بالاتجاهين عن طريق ثلاث خطوط.
- العمل كوصلة قائدة أو تابعة.
- أربع معدلات لنقل المعطيات.
- التحكم بترتيب النقل لخانات المعطيات مع أول LSB أو أول MSB .
- علم لحماية تعارضات الكتابة.
- توليد مقاطعة عند نهاية الإرسال.
- النهوض من نمط البطالة.

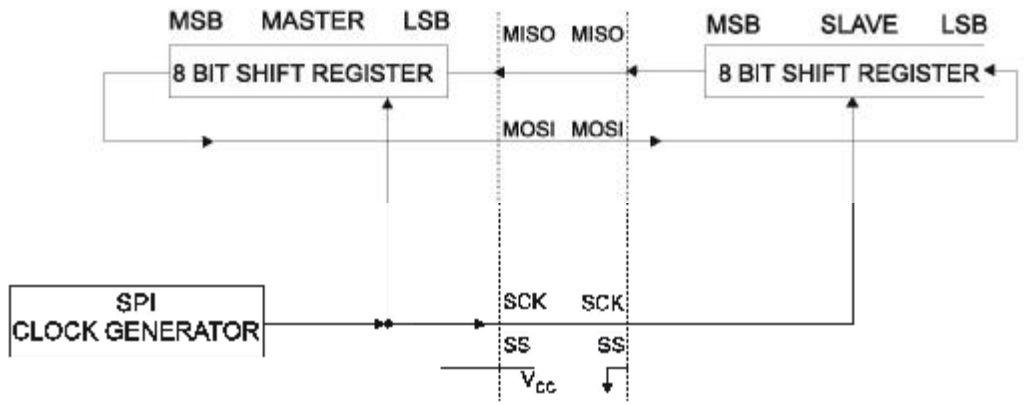
تمكن النافذة التسلسلية SPI من ربط متحكمين MCUs أحدها قائد والآخر تابع، كما هو مبين في الشكل ( ) للنافذة SPI ثلاث مسجلات: مسجل معطيات النافذة التسلسليةSPDR ، مسجل التحكم SPCR ، ومسجل الحالة SPSR. يتم تبادل المعطيات بشكل أساسي عبر ثلاثة أقطاب من البوابةB وهي PB3(MOSI) ، PB4(MISO)، PB5(SCK) .

في الوصلة القائدة يكون القطب PB5(SCK) خرجاً لنبضات الساعة في حين يكون مدخلاً لنبضات الساعة في الوصلة التابعة . إن عملية الكتابة إلى مسجل معطيات النافذة SPDR في المتحكم القائد تمثل بداية توليد نبضات ساعة النافذة التسلسلية SPI ، و عندها تتم إزاحة المعطيات المكتوبة خارجة من القطب PB3(MOSI) في الصلة القائدة إلى القطب PB3(MOSI) في الوصلة التابعة . و يتوقف مولد ساعة وصلة SPI عند إزاحة أول بايت ، و يفعل عندها علم نهاية الإرسال SPIF=1 . فإذا كانت خانت تمكين مقاطعة وصلة SPI مؤهلة SPIE=1 الواقعة في المسجل SPCR ، فإن ذلك سيؤدي إلى طلب مقاطعة . و يمكن اعتبار مسجلي الإزاحة في الوصلة القائدة و التابعة كمسجل إزاحة دائري واحد موزع بطول 16-bit ، كما هو مبين في الشكل (37) . فعندما تزاح المعطيات من الوصلة القائدة إلى التابعة ، فإن المعطيات ستزاح أيضاً بالاتجاه المعاكس في نفس الوقت . هذا يعني أن المعطيات ستتغير خلال دورة إزاحة واحدة في كلا الوصلتين القائدة و التابعة .



الشكل (36) : المخطط الصندوقي لوصلة SPI

و يمكن ربط وحدتي معالجة مركزية CPUs إحداهما قائدة والأخرى تابعة عبر الوصلة التسلسلية SPI ، كما هو مبين في الشكل (37) . بحيث يصبح القطب PB3(MOSI) خرجاً للساعة في نمط الوصلة القائدة و مدخل للساعة في نمط الوصلة التابعة . و تعني الكتابة على مسجل معطيات SPI للوحدة CPU القائدة بداية توليد ساعة الوصلة التسلسلية SPI ، و إزاحة المعطيات المكتوبة خارج القطب PB3(MOSI) إلى القطب PB3(MOSI) في الوصلة التابعة . و يتوقف مولد ساعة وصلة SPI عند إزاحة أول بايت ، و يفعل عندها علم نهاية الإرسال SPIF=1 . فإذا كانت خانت تمكين مقاطعة وصلة SPI مؤهلة SPIE=1 الواقعة في المسجل SPCR ، فإن ذلك سيؤدي إلى طلب مقاطعة . و يمكن اعتبار مسجلي الإزاحة في الوصلة القائدة و التابعة كمسجل إزاحة دائري واحد موزع بطول 16-bit ، كما هو مبين في الشكل (37) . فعندما تزاح المعطيات من الوصلة القائدة إلى التابعة ، فإن المعطيات ستزاح أيضاً بالاتجاه المعاكس في نفس الوقت . هذا يعني أن المعطيات ستتغير خلال دورة إزاحة واحدة في كلا الوصلتين القائدة و التابعة .



الشكل (37) : توصيل وحدتي SPI قائدة-تابعة

وفي الحقيقة لوصلة SPI موقع تخزين واحد في طرف المرسل و وموقعي تخزين في طرف المستقبل . و يعني ذلك أن البايت المرسل يجب أن لا يكتب على مسجل معطيات SPI قبل إتمام اكتمال دورة الإزاحة . و عند استقبال المعطيات ، يتوجب علينا قراءة المعطيات من مسجل معطيات SPI قبل اكتمال إزاحة البات التالي . وسوف نخسر في كل الحالات الأخرى البايت الأول.

عند تمكين وصلة SPI ، يجب ضبط اتجاه معطيات الأقطاب MOSI, MISO, SCK, SS بما يتناسب مع الجدول التالي :

القطب	الاتجاه المهيمن، نمط SPI القائدة	الاتجاه المهيمن، نمط SPI التابعة
MOSI	تابع لتعريف المبرمج	دخل
MISO	دخل	تابع لتعريف المبرمج
SCK	تابع لتعريف المبرمج	دخل
SS	تابع لتعريف المبرمج	دخل

الجدول (17) :هيمنة اتجاه قطب وصلة SPI

ملاحظة : راجع فقرة "الوظائف الخاصة للنافذة B" التي تتضمن شرحاً وافياً حول كيفية تعريف اتجاه أقطاب SPI.

توظيف القطب SS **SS Pin Functionality**

عند تهيئة وصلة SPI كوصلة قائدة (أي تفعيل الخانة MSTR=1 الموجودة في مسجل التحكم SPCR) ، فإنه باستطاعة المبرمج تحديد اتجاه القطب SS . فإذا هيئ القطب SS كقطب خرج ، فيصبح القطب SS قطب خرج عام و لا يؤثر على نظام SPI . أما إذا هيئ القطب SS كقطب دخل ، فإنه يجب أن يمسك على المنطق العالي لضمان عمل الوصلة SPI القائدة . و إذا قدنا هذا القطب SS للمنطق المنخفض من دارة خارجية عند تهيئة الوصلة SPI كوصلة قائدة بقطب SS معرف كقطب دخل ، فإن نظام SPI سيقاطع الخيار الأخير للوصلة القائدة كوصلة SPI تابعة و يبدأ بإرسال معطياته .و لتجنب مثل حالات التشابك في الممر ، فيجب أخذ بعين الاعتبار الإجراءات التالية :

١. عند تصفير الخانة MSTR=0 الموجودة في مسجل التحكم SPCR ، يصبح نظام SPI وصلة تابعة . ونتيجة لذلك تصبح الأقطاب MOSI و SCK أقطاب دخل .

٢. عندما يُفعّل علم مقاطعة الوصلة التسلسلية SPIF=1 الموجود في مسجل الحالة SPSR ، فإن روتين مقاطعة SPI سينفذ عند تفعيل كلٍ من خانة تمكين مقاطعة الوصلة التسلسلية SPIE=1 إلى جانب خانة تمكين المقاطعة العامة I=1 .

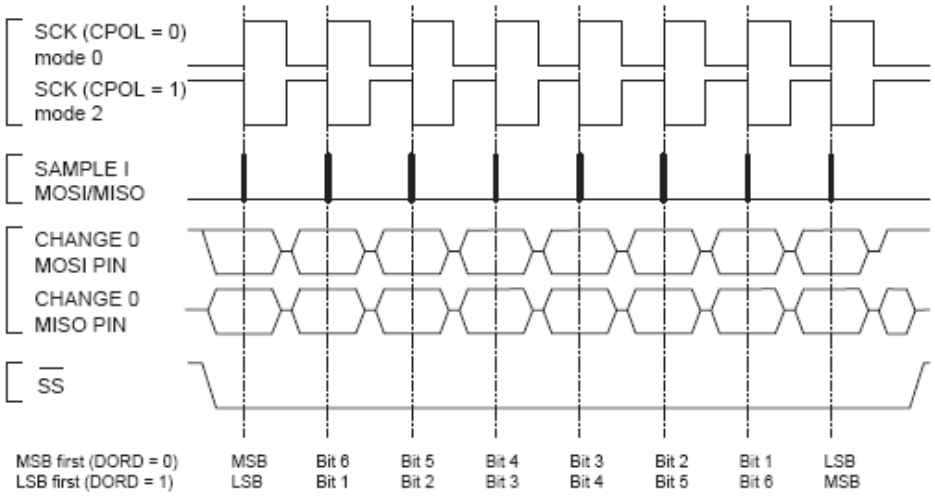
و هكذا ، عند استخدام مقاطعة إرسال SPI في نمط الوصلة القائدة ، فيوجد إمكانية للخروج حتى لو كان القطب SS مسحوب للمنطق المنخفض . و بشكل دائم تقوم المقاطعة بفحص بقاء فعالية الخانة MSTR=1 . وحين تصفير الخانة MSTR=0 عند اختيار الوصلة التابعة ، فإنها يجب أن تؤهل من قبل المبرمج لإعادة تمكين نمط وصلة SPI القائدة .

و عند تهيئة وصلة SPI كوصلة تابعة . فإنه يجب علينا تهيئة القطب SS دائماً كقطب دخل . و عند مسك القطب SS على المنطق المنخفض . يؤدي ذلك إلى تفعيل وصلة SPI ويصبح القطب MIOS قطب خرج إذا ما هيئ لذلك من قبل المبرمج . أما عند قيادة القطب SS للمنطق العالي ، فتصبح كل الأقطاب الخارجية كمداخل إذا ما هيئت لذلك من قبل المبرمج ، و وصلة SPI غير فعالة . هذا يعني أنه لن يتم استقبال المعطيات الواردة . مع ملاحظة أن منطق وصلة SPI سيصبح صفرياً عند فرض المنطق العالي على القطب SS . فإذا ما فرض المنطق العالي على القطب SS أثناء الإرسال ، فإن وحدة SPI ستوقف إرسالها و استقبالها بشكل فوري ، وكلا المعطيات المرسلة و المستقبلة تؤخذ بعين الاعتبار على أنها معطيات مفقودة .

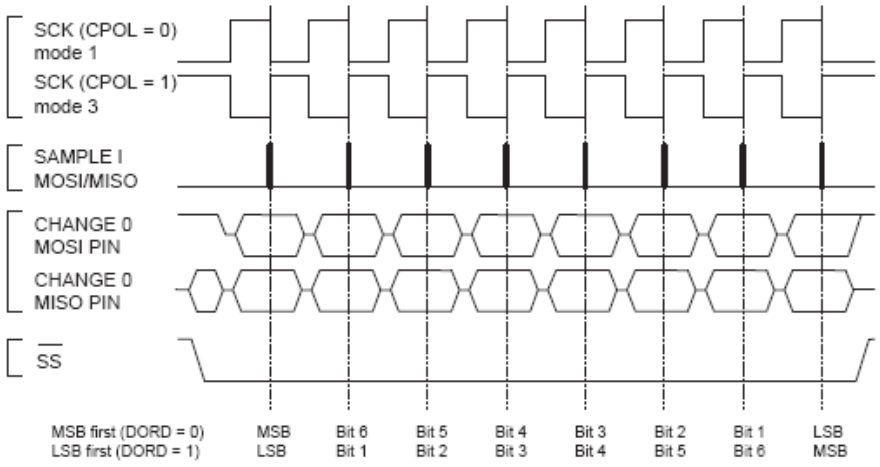
أنماط المعطيات Data Modes

يوجد أربع مجموعات لقطبية و طور الساعة SCK المتعلقة بالمعطيات التسلسلية . و المحددة من خانات التحكم CPHA و CPOL . و نبين على الشكلين (38) و (39) صيغة إرسال معطيات وصلة SPI والجدول التالي يوضح لنا وظيفة الخانتان CPHA و CPOL.

نمط SPI	الحافة المتأخرة	الحافة المتقدمة	
0	تجهز عند الجبهة الهابطة	أخذ العينة عند الجبهة الصاعدة	CPOL=0 CPHA=0
1	أخذ العينة عند الجبهة الهابطة	تجهز عند الجبهة الصاعدة	CPOL=0 CPHA=1
2	تجهز عند الجبهة الصاعدة	أخذ العينة عند الجبهة الهابطة	CPOL=1 CPHA=0
3	أخذ العينة عند الجبهة الصاعدة	تجهز عند الجبهة الهابطة	CPOL=1 CPHA=1



الشكل (38) : صيغة إرسال المعطيات عند CPHA=0 و DORD=0



الشكل (39) : صيغة إرسال المعطيات عند CPHA=1 و DORD=0

مسجل التحكم بوصلة SPI – SPCR

SPI Control Register-SPCR

7	6	5	4	3	2	1	0	
SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0	SPCR
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
0	0	0	0	0	1	0	0	

الخانة 7 - SPIE : خانة تمكين مقاطعة SPI

تسبب هذه الخانة عند تفعيلها SPIE=1 تنفيذ مقاطعة SPI ، وذلك عند تفعيل كلٍ من العلم SPIF=1 الموجود في مسجل الحالة SPSR و خانة تمكين المقاطعة العامة I=1 .

الخانة 6 - SPE : خانة تمكين وصلة SPI

لتمكين عمل وصلة النافذة التسلسلية ، فإنه يجب تأهيل خانة تمكين هذه الوصلة SPE=1 .

الخانة 5 - DORD : خانة اختيار وزن نقل المعطيات

عند DORD=1 ، فإن كلمة المعطيات سترسل مع أول خانة ذات الأهمية الأقل LSB .

أما DORD=0 ، فإن كلمة المعطيات سترسل مع أول خانة ذات الأهمية الأعلى MSB .

الخانة 4 - MSTR : خانة اختيار الوصلة القائدة التابعة

تقوم هذه الخانة بتحديد نمط الوصلة SPI القائدة عند تفعيلها MSTR=1 ، و نمط الوصلة SPI التابعة عند تصفيرها MSTR=0 . و إذا ما هيئ القطب SS كقطب دخل و تم سحبه للمنطق المنخفض عند تفعيل الخانة MSTR=1 ، فإن الخانة MSTR ستصفر ، ويصبح العلم SPIF=1 الموجود في مسجل الحالة SPSR . و على المبرمج إعادة تأهيل الخانة MSTR=1 كي يعيد تمكين نمط وصلة SPI القائدة .

الخانة 3 - CPOL : خانة قطبية الساعة

عند تفعيل هذه الخانة CPOL=1 ، فإن إشارة القطب SCK تصبح منخفضة عند البطالة . أما عند تصفير هذه الخانة CPOL=0، فإن إشارة القطب SCK تصبح عالية عند البطالة . لمزيد من المعلومات راجع الشكلين (38) و (39)

الخانة 2 - CPHA : خانة طور الساعة

حتى تتعرف على وظيفة هذه الخانة راجع الشكلين (38) و (39)

الخانتان 1,0 – SPR1, SPR0 : خانتي اختيار معدل ساعة وصلة SPI

تحدد خانتي التحكم SPR1 و SPR0 معدل ساعة القطب SCK عند تهيئة الوصلة التسلسلية كوصلة قائدة .و ليس لهاتين الخانتين أي تأثير في نمط الوصلة التابعة . ونبين في الجدول التالي العلاقة بين معدل ساعة القطب SCK و تردد ساعة الهزاز Fcl :

SPR0	SPR1	تردد SCK
0	0	Fcl /4
1	0	Fcl /16
0	1	Fcl /64
1	1	Fcl /128

الجدول (18) : العلاقة بين SCK و تردد الهزاز

مسجل حالة وصلة SPI – SPSR **SPI Status Register – SPSR**

Bit	7	6	5	4	3	2	1	0	
$0E ($2E)	SPIF	WCOL	-	-	-	-	-	-	SPSR
Read/Write	R	R	R	R	R	R	R	R	
Initial value	0	0	0	0	0	0	0	0	

لخانة 7 – SPIF : علم مقاطعة وصلة SPI

يُفعّل هذا العلم SPIF=1 عند اكتمال نقل معطيات الوصلة التسلسلية . وينفذ روتين مقاطعة SPI إذا كان كلٍ من الخانتين SPIE=1 و خانة تمكين المقاطعة العامة I=1 . و إذا تم سحب القطب SS نحو المنطق المنخفض في نمط الوصلة SPI القائدة ، فإن ذلك سيؤدي أيضاً إلى تفعيل العلم SPIF=1 . و يتم تصفير هذا العلم SPIF=0 من الكيان الداخلي للمتحكم عند تنفيذ شعاع المقاطعة المطابق . و كذلك يمكن تصفير هذا العلم

SPIF=0 عند القراءة الأولى لمسجل حالة SPI مع تفعيل العلم SPIF=1 ، وكذلك عند الولوج لمسجل معطيات SPI .

الخانة 6 – WCOL : علم تعارض الكتابة

يصبح هذا العلم WCOL=1 عند كتابة المعطيات على المسجل SPDR أثناء الإرسال . و يتم تصفير هذا العلم WCOL=0 (مع العلم SPIF=0) عند القراءة الأولى لمسجل حالة SPI مع تفعيل العلم WCOL=1 ، وكذلك عند الولوج لمسجل معطيات SPI .

الخانات 5...0 – Res : خانات محجوزة

هذه الخانات هي خانات محجوزة في المتحكم AT90S4433 و تقرأ دائماً صفر منطقي .

كما تستخدم الوصلة التسلسلية SPI أيضاً في المتحكم AT90S4433 لتحميل و قراءة معطيات كلٍ من ذاكرة البرنامج الوميضية و ذاكرة المعطيات EEPROM .

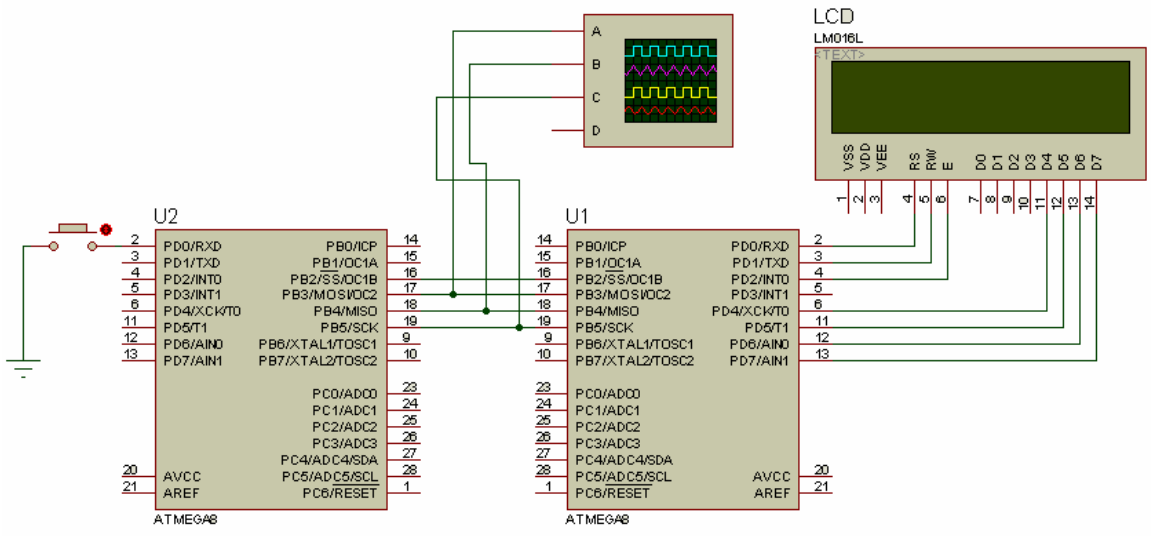
مسجل معطيات الوصلة التسلسلية SPDR – SPI SPI Data Register - SPDR

Bit	7	6	5	4	3	2	1	0	
$0F ($2F)	MSB							LSB	SPDR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

مسجل معطيات SPI هو مسجل قابل للقراءة و الكتابة يستخدم لنقل المعطيات ما بين ملف المسجلات مسجل إزاحة SPI .تؤدي عملية الكتابة على هذا المسجل بدء عملية نقل المعطيات . و تؤدي عملية قراءة هذا المسجل إلى قراءة مسجل إزاحة الاستقبال .

المشروع :

يقوم المعالج الأول (السيد) بإرسال قيمة العدد c إلى المعالج الثاني عبر وصلة SPI ويمكن زيادة هذا العدد بواسطة الكباس الموصول مع القطب PD.0 الذي يقوم باستقبال هذا العدد وطباعته على شاشة LCD.



برنامج المعالج السيد :

```
#include <mega8.h>
#include <delay.h>
#include <spi.h>

void main(void)
{
unsigned char c=0;
bit q=0;
PORTB=0x00;
DDRB=0xFe;

PORTC=0x00;
DDRC=0x00;

PORTD=0x01;
DDRD=0xf0;

// SPI initialization
// SPI Type: Master
// SPI Clock Rate: 2*62.500 kHz
// SPI Clock Phase: Cycle Half
// SPI Clock Polarity: Low
// SPI Data Order: MSB First
SPCR=0x54;
SPSR=0x00;
SPDR=c;
while (1)
    {
    if(PIND.0==0 && q==0)
    {
    q=1;
    SPDR=c;
    c++;
    }
    if(PIND.0==1)
    q=0;
    };
}
```

برنامج المعالج العبد :

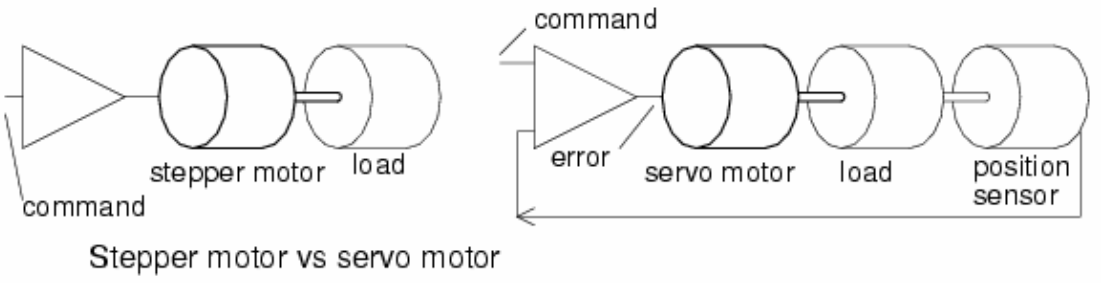
```
#include <mega8.h>
#include <spi.h>
#include <delay.h>
#include <stdlib.h>
#asm
   .equ __lcd_port=0x12 ;PORTD
#endasm
#include <lcd.h>

unsigned char data;
interrupt [SPI_STC] void spi_isr(void)
{
data=SPDR;
}

void main(void)
{
unsigned char *str;
PORTB=0x00;
DDRB=0xd0;

PORTC=0x00;
DDRC=0xff;

PORTD=0x00;
DDRD=0xff;

// SPI initialization
// SPI Type: Slave
// SPI Clock Rate: 31.250 kHz
// SPI Clock Phase: Cycle Half
// SPI Clock Polarity: Low
// SPI Data Order: MSB First
SPCR=0xC4;
SPSR=0x00;
// Global enable interrupts
#asm("sei")
delay_us(9);
lcd_init(16);

while (1)
```

```
        {
        itoa(data,str);
        lcd_gotoxy(0,0);
        lcd_puts(str);
        };
}
```

المشروع الرابع عشر
قيادة محرك خطوي

المقدمة :

يعتمد مبدأ المحركات الخطوية على دوران جسم الدوار المعدني عند تهييج أحد ملفات الثابت بحيث يشكل أصغر ممانعة مغناطيسية للفيض المغناطيسي بين الثابت والدوار، ولكن يمتاز المحرك الخطوي بأنه يمكننا من التحكم بمقدار زاوية الدوران كما أن تركيبه وقيادته بسيط ولا يملك مجمعات أو مسفرات.
وهذا ما يحققه محركات السيرفو من أجل التحكم بزاوية الدوران لكن محركات السيرفو تحتاج لتغذية عكسية و مقسّم جهد من أجل تحويل الموضع أو زاوية الدوران إلى الجهد وهذا الجهد يقارن مع الجهد المرجعي الذي تم ضبطه على دخل وحدة التحكم وبالتالي فإن التحكم به يكون بواسطة نظام الحلقة المغلقة . لذلك سوف نقارن بين المحركين من أجل توضيح مزايا كل منهما.
وبما أن الإشارة المرجعية هي عبارة عن جهد فهي قابلة للتعرض لأي تشويش وبالتالي عدم وجود دقة في زاوية الدوران.
أما المحركات الخطوية فإن التحكم بها يتم بواسطة التحكم بتردد النبضات المطبقية على مداخله (أي ملفات الثابت) حيث أن كل نبضة تجعله يدور بزاوية معينة وعند التحكم بعدد النبضات نكون قد تحكمنا بزاوية الدوران، وبالتالي فهو ليس بحاجة إلى تغذية عكسية ولذلك يمكن أن يعمل في نظام الحلقة المفتوحة كما يمتاز بدقة عالية لاحظ الشكل (1).
كما أن تكرارية الموضع في محركات السير فو إنها تعتمد على استقرارية مقسّم الجهد والعناصر المتعلقة بدارة التغذية العكسية.
أما تكرارية الموضع في الخطوية فإنها تعتمد على هندسة القسم الدوار الموجود في المحرك وبعد معرفة هندسته نقوم بتطبيق النبضات على مداخله بطريقة معينة(والتي سوف نتطرق إليها لاحقاً) نستطيع تكرار الموضع.



الشكل (1)

أنواع المحركات الخطوية حسب بنية الدوار:

يوجد تقريبياً ثلاثة أنواع للمحركات الخطوية:

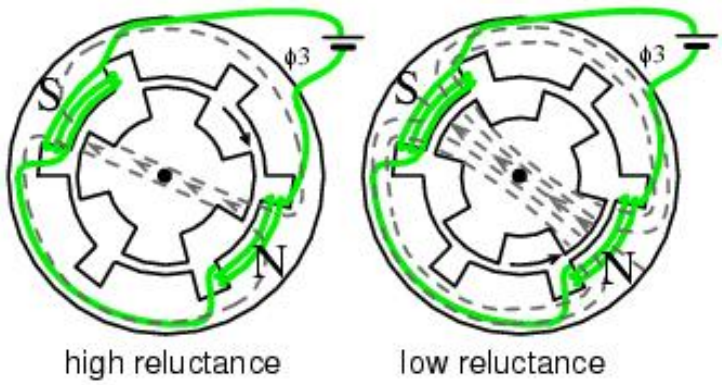
- المحركات ذات الممانعة المتغيرة (Switched reluctance).
- محركات الخطوة ذات المغناطيس الدائم (Permanent-magnet stepping motor).
- محركات الخطوة الهجينة (Hybrid stepping motor).

المحركات ذات الممانعة المتغيرة:

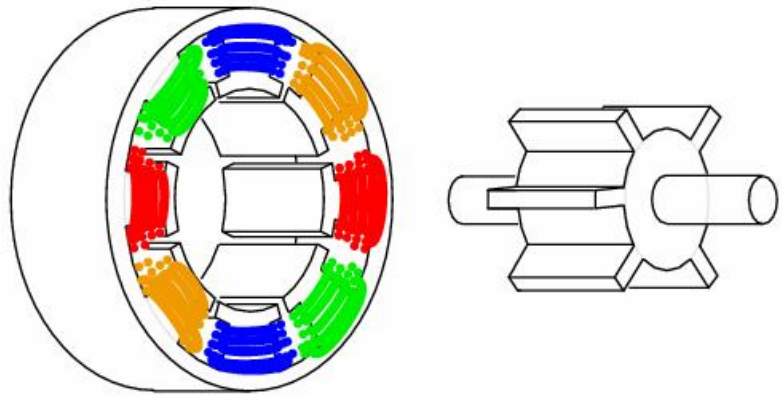
في هذه المحركات يكون الدوار مصنوع من مواد قابلة للمغنطة مثل الحديد الطري ويكون الدوار عبارة عن مسنن ومحاط بأسنان الثابت كما في الشكلين (2,3).

وعندما تتغذى إحدى و شائع الثابت بالتيار الكهربائي المستمر فإنها تمغنط الدوار وتجذب السن القريب منها بحيث تتقابل أسنان الدوار مع القطب المغذى من الثابت، ويجب أن يكون عدد أسنان الدوار مختلف عنه في الثابت وفي معظم المحركات يكون عدد أسنان الدوار أقل بسنين حيث يستفاد من الفروق الزاوية بين الثابت والدوار من أجل التحكم بزاوية الخطوة .

يتم عكس اتجاه دورانها بتغيير تسلسل تغذية ملفات الثابت.



الشكل (2)

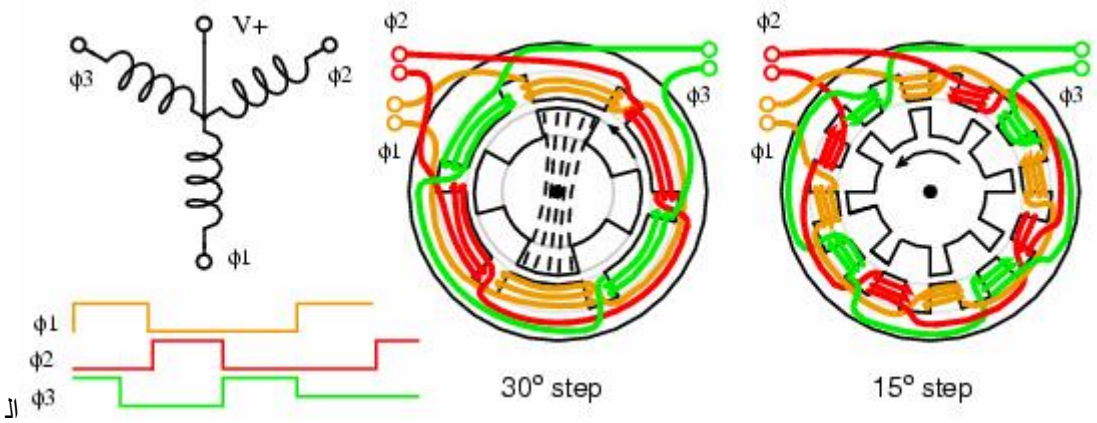


الشكل (3)

تعتمد زاوية الدوران أو طول الخطوة على عدد أسنان الدوار (Z) وعدد أطوار الثابت (N) وتعطى بالعلاقة:

$$Step\ length = 360/NZ$$

كما يلاحظ بالشكل (4):



شكل (4)

وتكون المحركات في هذه الحالة إما ذات قطاع واحد (Single stack) أو متعددة القطاعات (Multi stacks) تكون كل منها معزولة مغناطيسيا.

في المحركات المتعددة القطاعات يعتبر كل قطاع طور واحد ويغذى كل طور (قطاع) بالتتالي الشكل (5)، ويكون محور كل قطاع على الثابت فقط مزاح بزاوية عن الآخر، كما يتساوى عدد الأسنان في كل من الثابت والدوار. ويتوفر في الحياة العملية محركات بسبعة قطاعات وسبعة أطوار لاحظ الشكل في أسفل الصفحة.

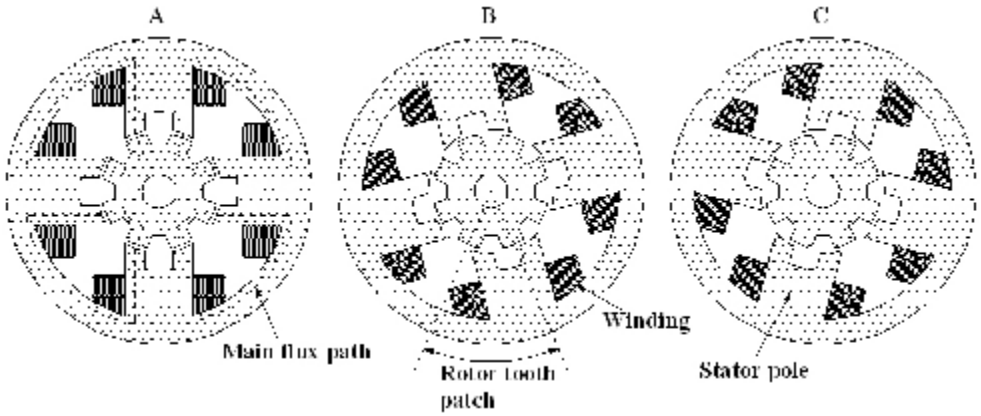
يتألف ثابت كل قطاع من عدد من الأقطاب ولكل قطب عدد من الأسنان ويلف ملف كل طور حول كل قطب، وتلف الأقطاب المتجاورة بشكل معاكس للقطب المجاور، أي ينشأ الفيض في أحد أقطاب الثابت وينتقل إلى الدوار عبر الثغرة الهوائية ثم يكمل مساره إلى القطب المجاور على الثابت عبر الثغرة أيضا.

تعطى زاوية الخطوة (Step length) لمحرك ذو N قطاع و Z سن للدوار بالعلاقة:

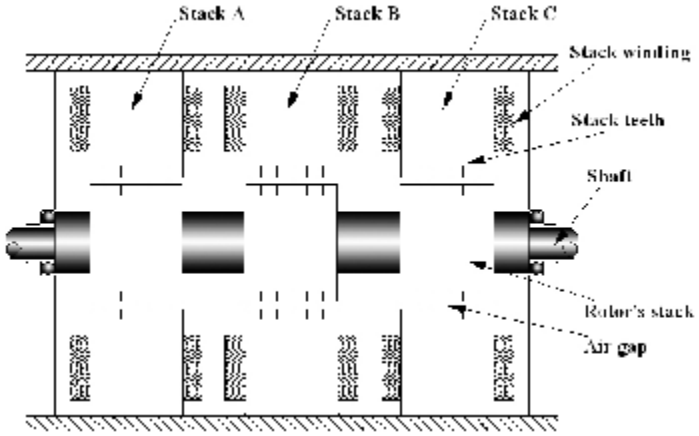
$$Step\ length = 360/NZ$$

تلخيص:

- الدوار عبارة عن اسطوانة مسننة من معدن طري.
- الأقل تعقيدا والأرخص ثمنا.
- زاوية خطوة كبيرة.
- المحرك الوحيد الذي لا يملك عزم إعاقة (Detent torque) عند وضعية الراحة (من دون تغذية ملفات الثابت).



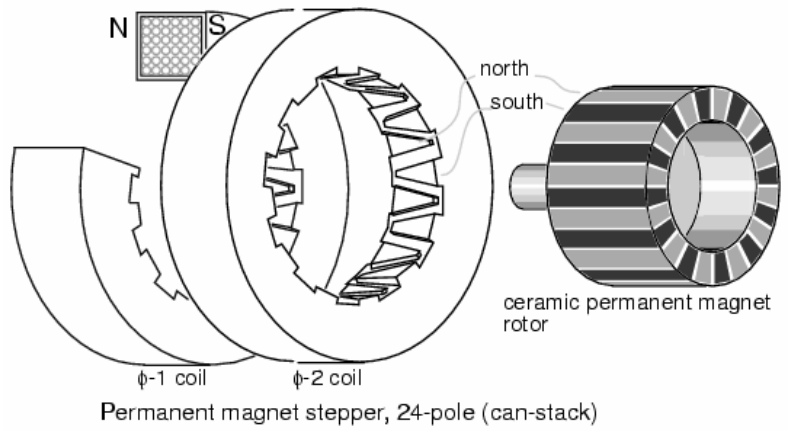
الشكل (5-a)



الشكل (5-b)

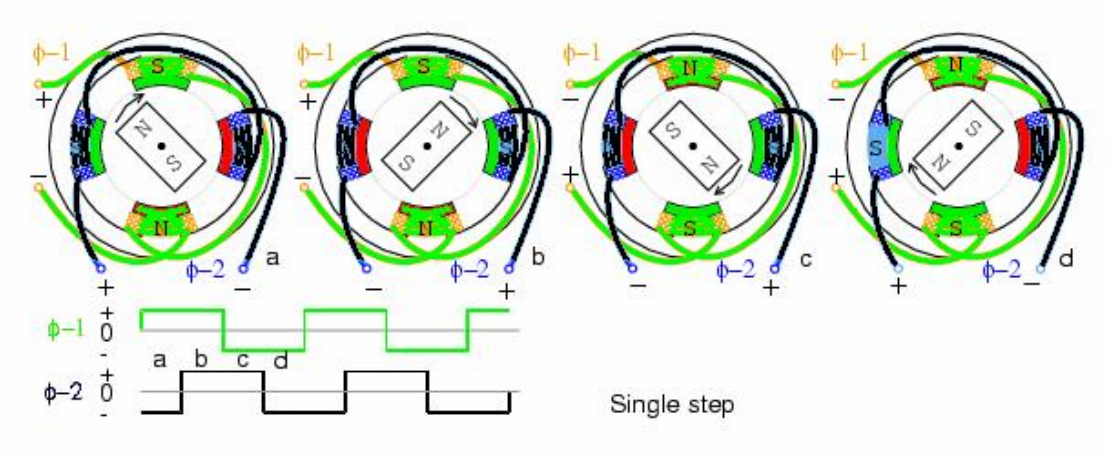
المحركات ذات المغانط الدائمة:

ويكون فيها الدوار عبارة اسطوانة غير مسننة وتتألف من مجموعة من المغانط الدائمة التي تتابع فيها أقطابها بشكل متناوب كما في الشكل(6). وبما أنه من الصعب تصنيع مغانط بأحجام صغيرة وبعدد كبير من الأقطاب لذلك فإن زاويتها الخطوية كبيرة نسبياً حوالي 30 إلى 90 درجة إلا أن مميزة السرعة- عزم جيدة.

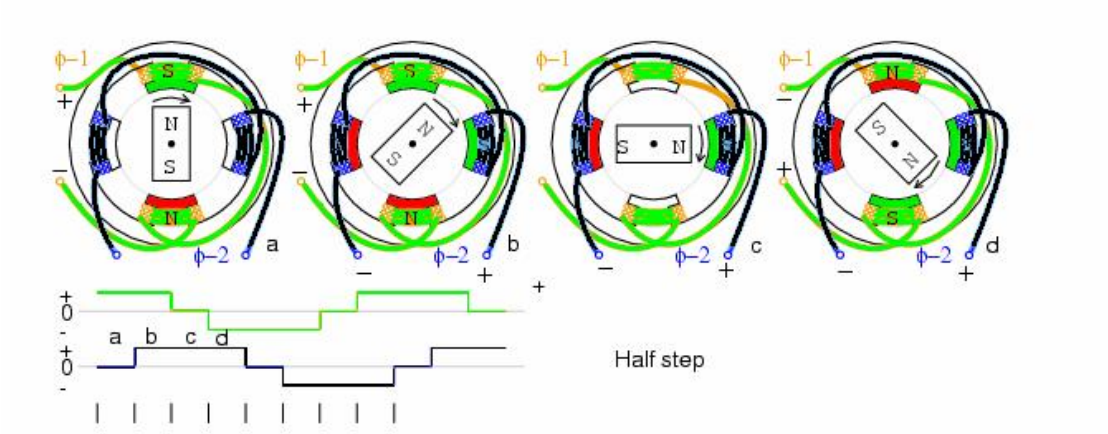


الشكل (6)

قطبية التيار المار ضرورية بالنسبة لهذا النوع من المحركات فهي تحدد طريقة قيادة المحرك حيث يوضح الشكل (7) عملية تغذية ملفات الثابت وذلك من أجل خطوة كاملة أما الشكل (8) فيظهر قيادة المحرك من أجل نصف خطوة:



الشكل (7)



الشكل (8)

تلخص:

- الدوار مصنوع من المغناطيس الدائم وعادة من مغناطيس الفيرايت (Ferrite) وبأقطاب متعددة.
- تؤمن زاوية خطوة جيدة (متوسطة).
- تستخدم عادة في طابعات الحاسب من أجل تقديم الورق.

المحركات الهجينة:

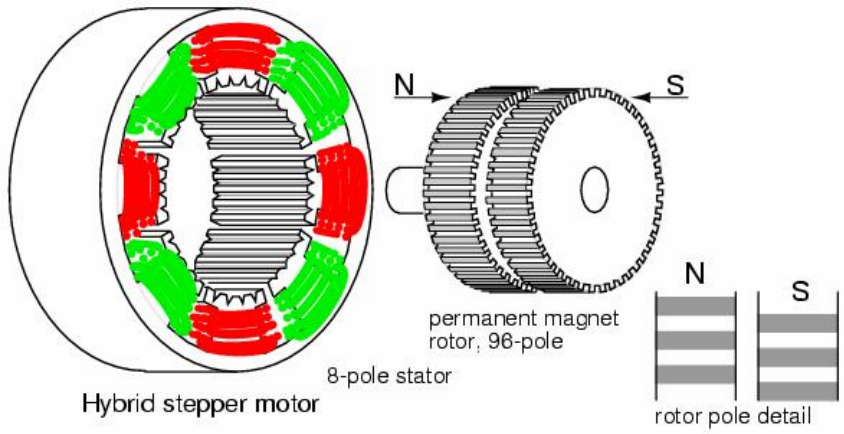
تملك هذه المحركات مغناطيس دائم متوضع ضمن الدوار وهي محركات ثمنها أغلى من محركات المغانط الدائمة ولكنها تتميز بأن زاويتها الخطوية صغيرة جداً تتراوح (٠,٩ -٣,٦) درجة حيث يلزمنا تطبيق مابين ١٠٠ إلى ٤٠٠ نبضة عمل من أجل إتمام دورة كاملة وهذا يعطينا دقة عالية في العمل.

ويكون فيها الدوار مسنن الشكل (9) لذلك فهي هجينة بين محركات الدائمة والممانعة المتغيرة.

يكون فيها محور الدوران عبارة عن مغناطيس دائم وتقوم الأسنان بتوجيه الفيض المغناطيسي في الثغرة الهوائية بين الثابت والدوار.

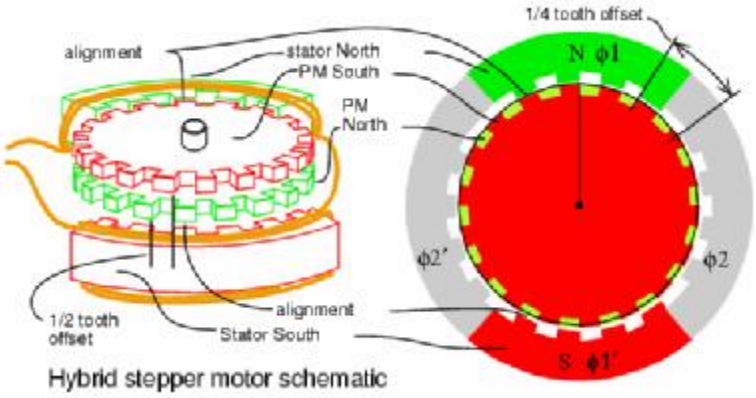
وتلف الملفات المتتالية (التي تملك نفس اللون في الشكل (9)) بعكس بعضها البعض.

ونلاحظ أن الدوار يملك مقطعين مختلفين في المحور (الأسنان غي متطابقة كما يظهر الشكل (10)) والثابت يملك مقطع أو مقطعين.



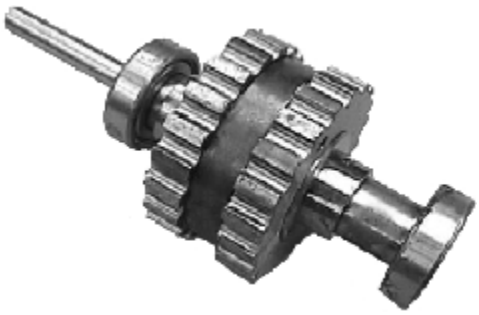
الشكل (9)

يتألف أقطاب الثابت والدوار من أسنان، فمثلا المحرك الموضح في الشكل (10) له قطبان لكل منهما ٦ أسنان مشكلا بذلك ١٢ سن للثابت والثابت يملك ١٨ سن ويلاحظ أيضا أن أسنان الثابت في كلا المقطعين متقابلين لبعضهما البعض بينما الدوار غير متقابلين.



الشكل (10)

تلخص:



- زاوية الخطوة الأصغر من الأنواع السابقة.
- يملك الدوار أيضا مغناطيسا وأسنان ناعمة. وتكون الإزاحة بين المقطعين للدوار نصف خطوة سنية للحصول على أصغر زاوي خطوة سنية.
- أسنان الثابت كما الدوار تكون ناعمة.

مقارنة بين أنواع المحركات:

يتم اختبار المحرك وفق التطبيق المطلوب، إذ ليس هناك حالة مطلقة لأفضلية إحداهما على الآخر في جميع المواقع.

تملك المحركات الهجينة زاوية خطوة الأصغر (1.8°) وذات دقة أكبر في وضعية الدوران والتوضع.

كذلك بينت الدراسات أنه من أجل حجم معين فإن العزم الذي يعطيه المحرك الهجين أكبر من المحركات ذات الأنواع الأخرى.

وعندما لا تهيج المحركات الثلاث السابقة فإن المحرك الهجين وذات المغناطيس الدائم ينتج عزم مقاوم والذي يحفظ الدوار عند خطوة معينة، وعلى الرغم من كون هذا العزم أصغر من العزم الذي يولده المحرك أثناء التغذية إلا أنه يحافظ على موقعه عند انقطاع التغذية (لعطل ما مثلا).

تمتاز محركات الخطوة ذات الممانعة المغناطيسية المتغيرة بميزتين أساسيتين عندما يطلب منها تحريك الحمل لمسافات قصيرة:

- § طول الخطوة النموذجي 15° أول منها في المحرك الهجيني وبالتالي فهو يحتاج لعدد خطوات أقل لكي يتحرك لمسافة معينة. واختصار عدد الخطوات يعني تغيرات أقل في التهييج.
- § العطالة الميكانيكية لدوار المحرك الخطوي ذو الممانعة المغناطيسية المتغيرة أقل منها في المحرك الهجين لأنه لا يوجد مغناطيس دائم على الدوار والذي يسهم إلى حد ما في عطالة الحمل الكلية على المحرك. وتخفيض هذه العطالة يسمح بتسارع أكبر للمحرك.

يعتبر العزم الذي ينتجه المحرك الخطوي ذو المغناطيس الدائم من أجل وحدة حجم معينة ضعيفا.

وهناك نوع من محركات الخطوة يدعى بالمحرك الخطوي الكهروهيدروليكي (Electro hydraulic stepping motor) ويمتاز بتأمين عزم خرج عالي جدا.

قيادة المحركات الخطوية:

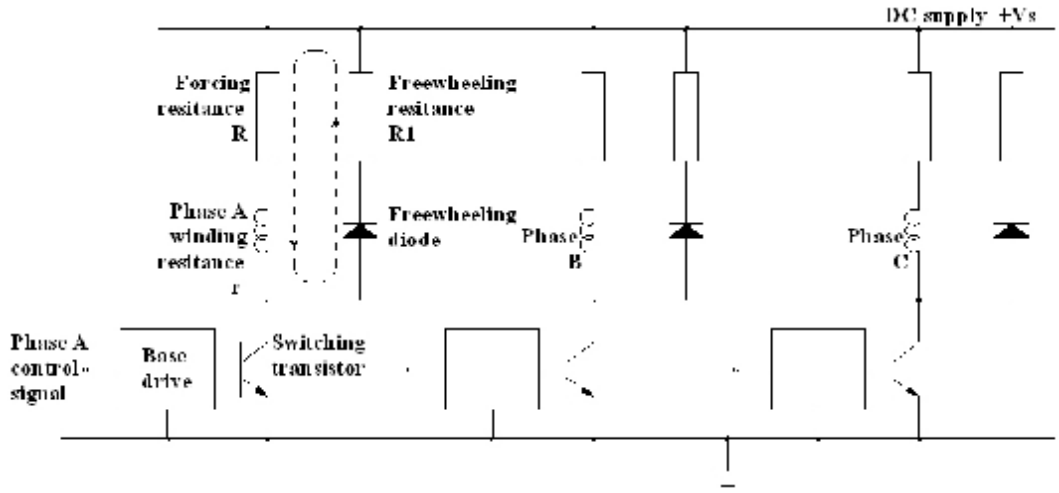
بما إن إشارات التحكم اللازمة لقيادة المحرك الخطوي أصغر بكثير من إشارات التحكم للعديد من الدارات المتكاملة وخاصة دارات ألـ TTL لذلك لابد من تضخيمها عدة مراحل قبل ربطها مع المحرك.

كما تتعلق قيادة المحركات الخطوية ببنية المحرك (أي ذات مغانط دائمة أو ممانعة متغيرة).

إن المحركات ذات الممانعة المغناطيسية المتغيرة تحوي ثلاثة ملفات على اطوار على الأقل، ويحتاج تيار كل طور لفصل ووصل مع العلم أن قطبية التيار لا علاقة لها بإنتاج العزم.

سنناقش دارة قيادة بسيطة ووحيدة القطبية لمحرك خطوي ذو ممانعة مغناطيسية متغيرة والتي تدعى بالدارة زحيدة القطبية:

قيادة المحرك ذو الممانعة المتغيرة (دارة وحيدة القطبية):



وضح الشكل التالي دارة بسيطة لقيادة المحرك خطوي ذو ممانعة مغناطيسية متغيرة:

يلاحظ بأن ملف كل طور بحاجة لدارة تهييج خاصة به.

كما أن إشارات التحكم بقاعدة الترانزستورات الطورية تكون صغيرة وغير كافية لوضعها في حالة الإشباع لذلك لا بد من تضخيمها بواسطة مفاتيح الكترونية قبل تطبيقها على قاعدة الترانزستورات.

عندما يمرر تيار كافي لإشباع الترانزستور عبر قاعدته تتطبق كامل التغذية المستمرة على الترانزستور وملف الطور والمقاومة المرغمة (Forcing resistor) باعتبار أن جهد التحييز الأمامي للترانزستور صغير.يتم اختيار جهد التغذية بحيث يحقق التيار الاسمي وفق المعادلة:

$$Vs = I\,(r + R)$$

حيث r المقاومة الداخلية لملف الطور الواحد.

R مقاومة الإرغام.

حيث يملك ملف الطور الواحد تحريضية ذات ثابت زمني L / R كبير. لذلك فإن وصول التيار للقيمة الاسمية سيكون بطيئا وغير مرضي وخاصة عند السرعات العالية حيث التردد العالي لن يسمح للتيار بالوصول للقيمة الاسمية لذلك زيادة المقاومة R يقلل من الثابت الزمني ويحقق السرعات العالية.

كما أن التحريضية لها أثر مهم آخر وهو استمرار مرور التيار بالرغم من قطع إشارة التحكم عن الترانزستور وجعله في حالة الفتح مما يحرض جهد عالي بين المجمع والباعث قد يسبب تلفه.

ولتجنب هذه المشكلة يوضع مسار حر عبر الديود لمرور التيار التحريضي للملف عبره وعبر مقاومة المسار الحر Rf.

عند لحظة تحول الترانزستور لحالة القطع يمر في تلك اللحظة التيار الاسمي I عبر المسار الحر مسببا هبوط جهد أعظمي على المقاومة Rf والذي بدوره هو الجهد العكسي على الترانزستور، ويساوي بإهمال هبوط الجهد على الديود إلى:

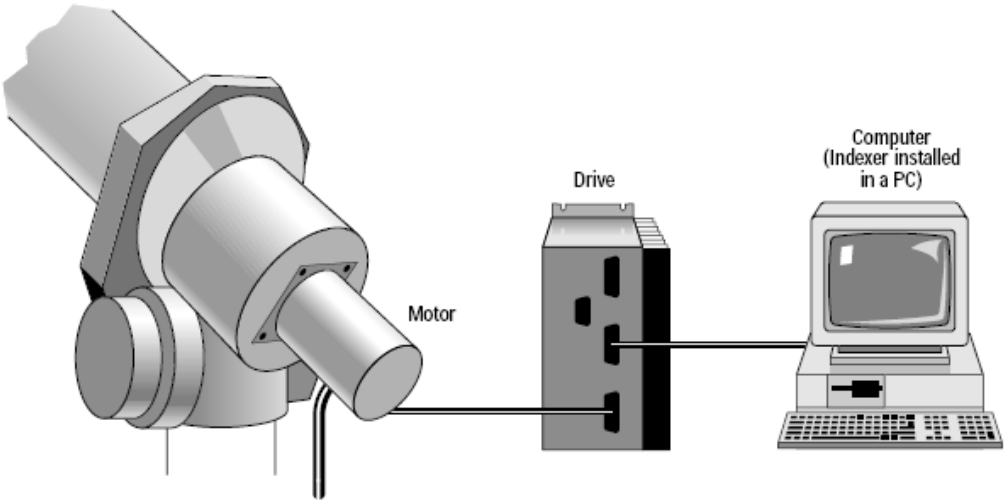
$$Vce\ max = Vs + Rf\ .\ I$$

أي يجب اختيار ترانزستور يتحمل قيمة الجهد السابقة.

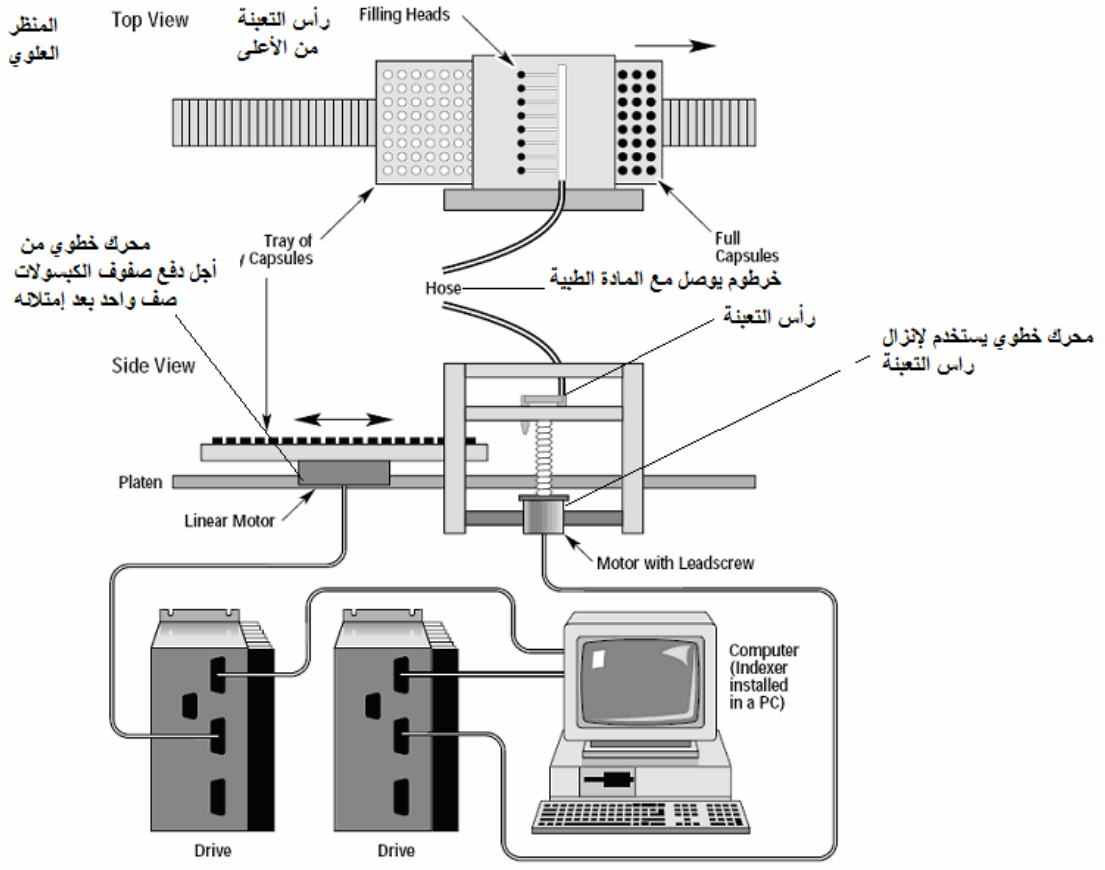
وهكذا يتبدد تيار الطور يتخامد في مقاومة المسار الحر والمقاومة المرغمة والمقاومة الداخلية لملف الطور.

تطبيقات المحركات الخطوية :

يستخدم المحرك الخطوي في التطبيقات التي تحتاج إلى دقة عالية في الموضع مثال على ذلك المحركات الموجودة في الطابعات الليزرية والراسمات والروبوتات وفي التطبيقات الصناعية والشكل التالي يبين لنا استخدام المحرك الخطوي للتوجيه ال TELESCOPE باتجاه معين ودقيق عن طريق الكومبيوتر.

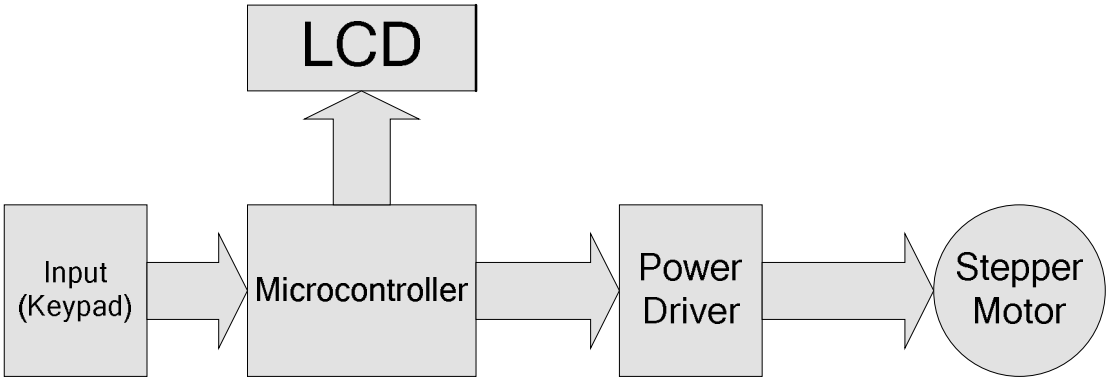


أما الشكل التالي يظهر لنا بوضوح استخدام المحرك الخطوي في آلة لتعبئة كبسولات الأدوية.



Capsule Machine

المخطط الصندوقي للمشروع :



يتكون المخطط الصندوقي من الأجزاء الرئيسة التالية :

١- **وحدة إدخال Input Unit** : بحيث يدخل المستخدم كل من السرعة و نوع الزاوية و اتجاه الدوران للمحـــرك ، وذلك عن طريق لوحة مفاتيح Keypad .

٢- **وحدة المعالجة المركزية Microcontroller** : تكون مسؤولة عن معالجة البيانات المدخلة لتحولها إلى إشـــارات رقمية لقيادة المحرك الخطوي .

٣- **المحرك الخطوي Output Unit** .

٤- **وحدة الملائمة Power Driver Unit** : تقوم بملائمة إشارات التحكم الصغيرة الصادرة عن المتحكم المـــصغر إلى إشارات استطاعية مكافئة ، وتقوم بعزل دارة التحكم Microcontroller عـــن دارة القـــدرة Stepper Motor .

٥- **شاشة عرض LCD** : للتواصل مع المستخدم .

مواصفات التشغيل للمحرك الخطوي :

الجهد :

يعمل المحرك الخطوي على جهد مستمر تختلف قيمته حسب حجم المحرك المـــستخدم ، وبالنـــسبة للمحركات الموجودة في سواقات الأقراص المرنة فهي تعمل على جهد 12V . أما المحرك المـــستخدم فـــي دارتنا فهو يعمل على جهد 24V DC .

التيار :

لملفات المحرك الخطوي مقاومة صغيرة لا تتجاوز 100 Ohm لكل ملف ، والجهد المطبق عليها لنفرض أنه 24V فيكون التيار المار في كل ملف I=0.25A وهو تيار كبير نسبياً ، لـــذلك عنـــدما نقـــوم بالتحكم بملفات المحرك ، فالتيار المار في بوابات الـــ Microcontroller سيكون كـــافي لعطبهـــا جميعاً !!!!.

المقاومة و الممانعة لملفات المحرك الخطوي :

تمتلك الملفات مقاومة لا تتجاوز 100 Ohm كما ذكرنا ، أما الممانعة فهي صغيرة أيـــضاً و بحـــدود 100-250 mH .

الخطوة :

تأتي المحركات الخطوية بزوايا دوران متعددة ، وعادة تكتب على الجسم الخارجي للمحرك. و تقاس بالدرجات.

و قد نصادف أحياناً محركات خطوية تكتب عليها عدد الخطوات في الدورة الواحدة، فببساطة نستطيع حساب الخطوة الواحدة : Step = 360/n .

التردد الأعظمي :

إذا أردت أن تدور محور المحرك بسرعة معينة ، ما عليك إلا أن تزيد من تردد الإشارة التي تحدد الانتقال بين الخطوات ، ولكن هذا التردد لا يجوز أن يتجاوز السرعة الأعظمية المحددة من قبل الشركة الصانعة ، فمثلاً تكون 200-300Hz و بالتالي فإن أعظم سرعة للدوران هي بين 300-400rpm .

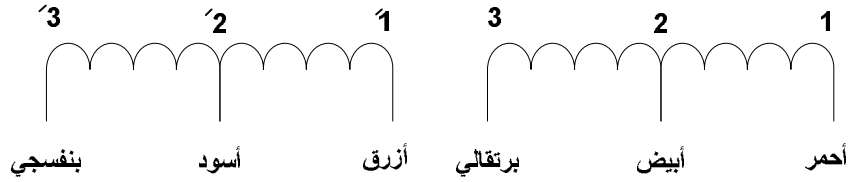
عزم المسك الذي يتحمله المحور :

بما أن المحرك الخطوي قد صمم لتحريك الأجسام بدقة إلى موضع محدد ، لذا كان من الضروري أخذ بعين الاعتبار مسؤولية تحمل محور المحرك للحمولات الزائدة عندما لايكون محور المحرك في حالة دوران ، وعادة تحدد الشركات المصنعة قيمة العزم الكافي لقيادة الأجسام و لكن مع الحافظة على ثبات الزاوية ، وهذه القيمة تعطى بالميللي نيوتن- متر (m Nm).

وتكون بالنسبة لمحركنا المستخدم (70 m Nm) .

وصل المحرك :

في المحركات الأحادية القطبية المتواجدة في الأسواق لها ستة أسلاك ألوانها أحمر أبيض برتقالي و أزرق أسود بنفسجي حيث الألوان الثلاثة الأولى عبارة عن وشيعة الثلاثة الآخرى عبارة عن وشيعة ثانية كما في الشكل () :



نوصل السلكان الأبيض والأسود مع بعضهما البعض ونوصلهما إلى القطب الموجب ونطبق على الأطراف البقية جهد منخفض أثناء عمل المحرك .

دارة القدرة :

تكلمنا في الفقرة الأولى عن التيار الذي يمر في ملفات المحرك الخطوي عندما يطبق جهد على أحد أطواره (ملفاته) وهو كبير نسبياً . لذلك لا يجوز قيادة المحرك مباشرة من أقطاب المتحكم المصغر ، بل لابد من تصميم نظام قيادة (Driver) تكون وظيفته : استقبال أوامر التحكم الرقمية ذات الجهد الصغير ليتم تحويلها إلى جهد و تيار كافي لعمل المحرك .

حديثاً يأتي المحرك الخطوي مع نظام قيادة جاهز خاص به و ما عليك سوى أن تهتم فقط بإشارات التحكم التي سيتم إرسالها من المتحكم المصغر .

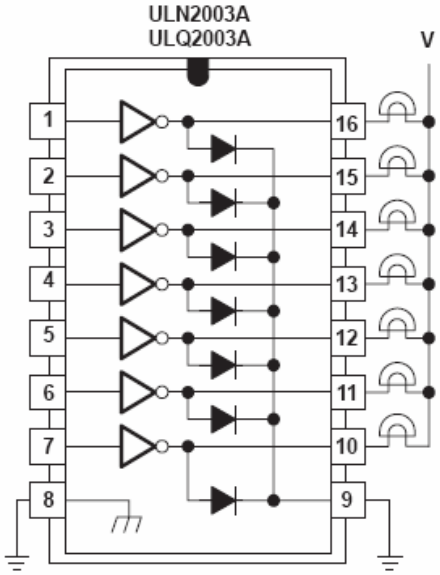
قدمت شركة Allegro Microsystems شريحة قد صممت لهذا الغرض، فمهمتها الأساسية قيادة الأحمال ذات التيار العالي بإشارات تحكم ذات الجهد والتيار الصغير ، وهي من عائلة ULN200x.

وهناك بديل عنها مثل : DS200x من تصنيع National Semiconductor و MC1413 من إنتاج Motorola . والتي سيتم استخدامها في دارتنا هي: ULN2003.

الشريحة ULN2003 :

وهي عبارة عن مصفوفة ترلنزيستورات DARLINGTON ، تستقبل هذه الشريحة على مداخلهاكإشارات تحكم جهود : CMOS – TTL و أما تيار الخرج يصل إلى 0.5A لكل مخرج . أما جهود الخرج يمكن أن تصل إلى 95V .

تحتوي هذه الشريحة على سبعة مداخل(1…7) وسبعة مخارج (10..16) ، وقطب أرضي (8) وقطب مشترك(9) . الشكل التالي يوضح لنا بنية الدارة وكيفية وصل الأجهزة مع خرجها .



وصل لوحة المفاتيح مع المتحكم :

الشكل التالي يظهر لنا كيفية وصل لوحة المفاتيح داخلياً ونلاحظ أن كل مفتاح يتم وصل أحد أطرافه بأطراف المفاتيح التي تقع في نفس العمود أما الطرف الآخر من المفتاح فيتم وصله مع أطراف المفاتيح التي تقع في نفس السطر وفي الشكل نلاحظ أن عدد المفاتيح هو 4*4 ويوجد مقاييس عديدة وأحجام مختلفة .

كيفية الكشف عن ضغط المفتاح :

يتم استخدام تقنية المسح من أجل الكشف عن الفتاح الذي تم ضغطه وتعتمد هذه التقنية على الطريقة التالية :

يتم في البداية وصل الأسطر مع منافذ المتحكم ونعرفها على أنها مداخل في حالة مقاومة رفع وبالتالي فإن القيمة الإفتراضية على هذه المداخل هي '1' منطقي . أما الأعمدة فيتم وصلها مع منافذ المتحكم والتي تهيئ على أنها مخارج . وبالتالي نلاحظ أنه يلزمنا فقط ثمانية منافذ للتحكم بستة عشر مفتاح ولكن المشكلة في كيفة معرفة الفتاح الذي تم ضغطه ...! في الحقيقة المسألة بسيطة ففي البداية نقوم بتطبيق القيمة "0111" على الأعمدة ونقرأ حالة الأسطر.

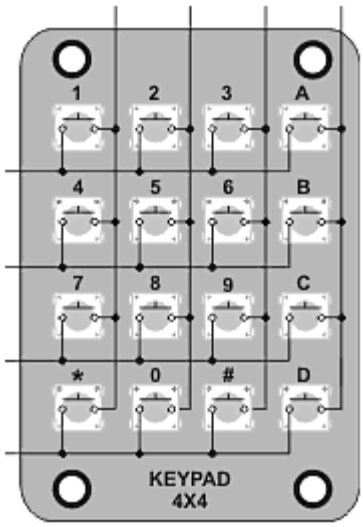
فإذا كانت القيمة "0111" فذلك يعني أن الرقم(1) تم ضغطه

أما إذا كانت القيمة "1011"" فذلك يعني أن الرقم(4) تم ضغطه

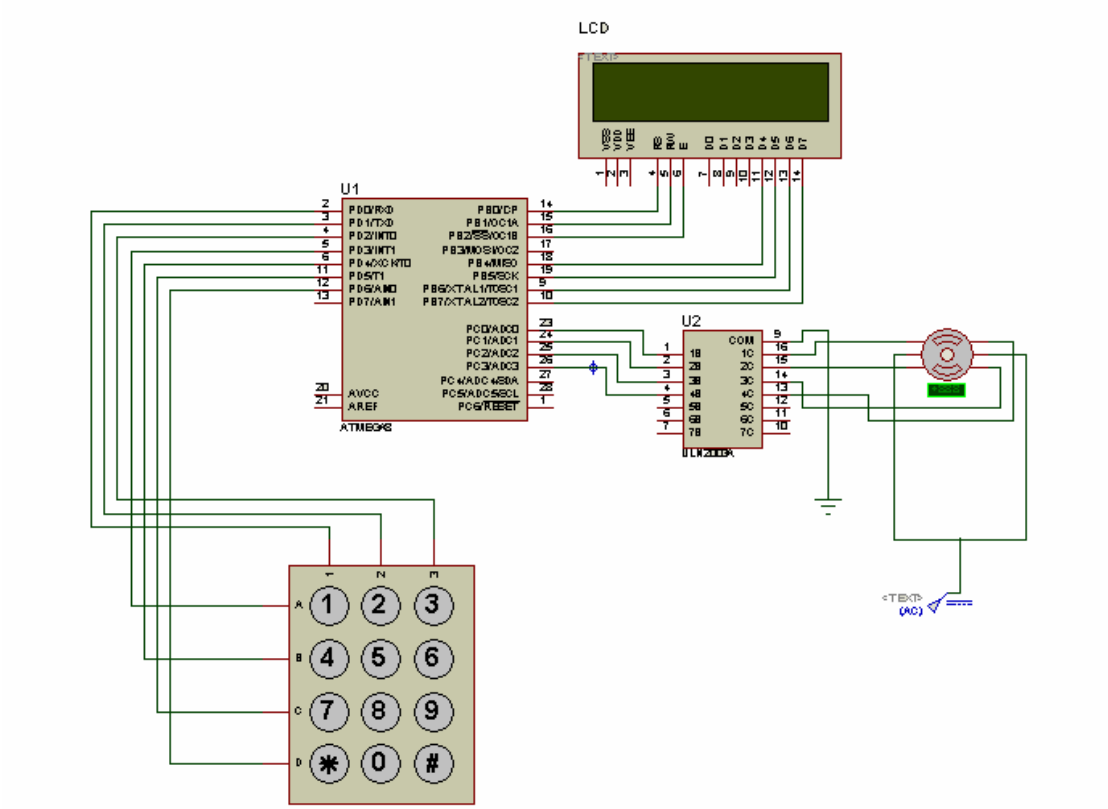
أما إذا كانت القيمة "1011"" فذلك يعني أن الرقم (7) تم ضغطه

أما إذا كانت القيمة "1011"" فذلك يعني أن الرقم(*) تم ضغطه

أما إذا كانت القيمة "1111"" فذلك يعني أنه لم يتم ضغط أي مفتاح وبعد ذلك نقوم بنفس العملية للعمود الثاني(1011) ثم الثالث(1101) والرابع(1110) وهكذا .



دارة المشروع :



يقوم هذا المشروع بالتحكم بالمحرك الخطوي أحادي القطبية بعدة إمكانيات أول خيار يظهر لنا على الــشاشة هو في أي اتجاه تريد للمحرك أن يدور في اتجاه الساعةCW أو بعكس اتجاه الساعة CCW . بعد ذلك يخيرنا بين نريد التحكم بعدد الخطوات أو بالسرعة ، إذا اخترنا عدد الخطوات فسوف يطب منــا عــدد الخطــوات المطلوب تنفيذها أما إذا اخترنا السرعة فيخيرنا ببين السرعة العالية أو المتوسطة أو المنخفضة .

البرنامج :

```
/*****************************************************
Project: Stepper motor
Author: Ismail Al-troudi
*****************************************************/
#include <mega8.h>
#include <delay.h>
#include <stdlib.h>
#asm
  .equ __lcd_port=0x18 ;PORTB
#endasm
#include <lcd.h>
int keypad(void);
char o=0;
```

```
void main(void)
{
char q,r;
int d,s,steps;
char *str;

PORTB=0x00;
DDRB=0xFF;

PORTC=0x00;
DDRC=0x7F;

PORTD=0xff;
DDRD=0x07;

// LCD module initialization
lcd_init(16);
lcd_putsf(" Stepper motor");
delay_ms(400);
lcd_clear();

lcd_putsf("chose Direction");
delay_ms(400);

wrong:
      lcd_gotoxy(0,1);
      lcd_putsf("1 cl 2 clw");
      lcd_gotoxy(14,1);
      lcd_putsf(" ");
      d=keypad();
      itoa(d,str);
      lcd_gotoxy(14,1);
      lcd_puts(str);
      delay_ms(500);
      if(d==1 || d==2)
      goto cont;
      else
      {
      lcd_gotoxy(0,1);
      lcd_putsf("   Wrong  ");
      delay_ms(500);
      goto wrong;
      }
cont:
   delay_ms(300);
   lcd_clear();
   lcd_gotoxy(0,0);
   lcd_putsf("Speed or steps");
false:lcd_gotoxy(0,1);
     lcd_putsf("1 sp 2 st");
```

```
      lcd_gotoxy(13,1);
      lcd_putsf(" ");
      q=keypad();
      itoa(q,str);
      lcd_gotoxy(13,1);
      lcd_puts(str);
      delay_ms(500);
      if(q==1 || q==2)
      goto cont1;
      else
      {
      lcd_gotoxy(0,1);
      lcd_putsf("   Wrong  ");
      delay_ms(500);
      goto false;
       }

cont1:
        if(q==1)
        {
        lcd_clear();
        lcd_putsf("1 h 2 m 3 l ");
again:
        lcd_gotoxy(0,1);
        lcd_putsf("                ");
        lcd_gotoxy(10,1);
        s=keypad();
        itoa(s,str);
        lcd_puts(str);
        delay_ms(500);
        if(s==1 || s==2 || s==3)
        goto speed;
        else
        {
        lcd_gotoxy(0,1);
        lcd_putsf(" Wrong  ");
        delay_ms(500);
        goto again;
        } // end else
        } //end if


if(q==2)
{
lcd_clear();
lcd_putsf("enter NO of steps");
keytest:
        r=keypad();
        if (r==10)
        goto steps_start;
```

```
        else
        {
        steps=steps*10;
        steps+=r;
        lcd_gotoxy(3,1);
        itoa(steps,str);
        lcd_puts(str);
        delay_ms(500);
        goto keytest;
        }  // end else
} // end if q==2


steps_start:
        if (d==1)
          {
          while(1)
          {
           if(steps!=0)
           {delay_ms(500);
           PORTC=1;
           steps--;
           }
           if(steps!=0)
           {delay_ms(500);
           PORTC=2;
           steps--;
           }
           if(steps!=0)
           {delay_ms(500);
           PORTC=4;
           steps--;
           }
           if(steps!=0)
           {delay_ms(500);
           PORTC=8;
           steps--;
           if(steps==0)
           goto end_main;
           }
           };
          } // end if
        if(q==2)
        {
        while(1)
          {
           if(steps!=0)
           {delay_ms(500);
           PORTC=8;
           steps--;
```

```
        }
        if(steps!=0)
        {delay_ms(500);
        PORTC=4;
        steps--;
        }
        if(steps!=0)
        {delay_ms(500);
        PORTC=2;
        steps--;
        }
        if(steps!=0)
        {delay_ms(500);
        PORTC=1;
        steps--;
        if(steps==0)
        goto end_main;
        }
        };
       } // end if



speed:
{
if(s==1)
{
lcd_clear();
lcd_putsf("High Speed ");
}
if (s==2)
{
lcd_clear();
lcd_putsf("Medium Speed ");
}
if(s==3)
{
lcd_clear();
lcd_putsf("Low Speed ");
}

 s=s*10;
 if(d==1)
  {
   while(1)
    {
    PORTC=1;
    delay_ms(s);
    PORTC=2;
```

```
    delay_ms(s);
    PORTC=4;
    delay_ms(s);
    PORTC=8;
    delay_ms(s);
    };
   }

else
  {
  s=s*10;
  while(1)
  {
  PORTC=8;
  delay_ms(s);
  PORTC=4;
  delay_ms(s);
  PORTC=2;
  delay_ms(s);
  PORTC=1;
  delay_ms(s);
   };
   }
} //end speed

end_main:
}// end main



int keypad()
{
while(1)
{
if(PIND.3==1 && PIND.4==1 && PIND.5==1 && PIND.6==1 )
o=0;

if(o==1)
goto end;

PORTD.0=0;
PORTD.1=1;
PORTD.2=1;
if(PIND.3==0 &&o!=1)
{
o=1;
return  (1);
}
else if(PIND.4==0 &&o!=1)
{
```

```
o=1;
return  (4);
}
else if(PIND.5==0 &&o!=1)
{
o=1;
return  (7);
}
else if(PIND.6==0 && o!=1)
{
o=1;
return  (10);
}
PORTD.0=1;
PORTD.1=0;
PORTD.2=1;
if(PIND.3==0 &&o!=1)
{
o=1;
return  (2);
}
else if(PIND.4==0 &&o!=1)
{
o=1;
return  (5);
}
else if(PIND.5==0 &&o!=1)
{
o=1;
return  (8);
}
else if(PIND.6==0 && o!=1)
{
o=1;
return  (0);
}

PORTD.0=1;
PORTD.1=1;
PORTD.2=0;
if(PIND.3==0 &&o!=1)
{
o=1;
return  (3);
}
else if(PIND.4==0 &&o!=1)
{
o=1;
return  (6);
}
```

```
else if(PIND.5==0 &&o!=1)
{
o=1;
return  (9);
}
else if(PIND.6==0 && o!=1)
{
o=1;
return  (11);
}
end:
};
}
```

المشروع الخامس عشر
وصل المتحكم مع الحاسب عبر المنفذ التسلسلي RS-232

لقد كانت المنافذ التسلسلية منذ البداية جزءا من الحاسب الشخصي وكل منفذ COM أو منفذ في حاسب شخصي ما هو الا منفذ تسلسلي غير متزامن مضبود بوحدة (UART) وقد يمتلك المنفذ COM وصلة RS-232 التقليدية أو وصلات قريبة مثل RS-485 ، او قد يكون المنفذ مخصصا للاستخدام من قبل مودم خارجي أو جهاز اخر ، كما يمكن للحاسب الشخصي ان يحتوي على انواع أخرى من المنافذ التسلسلية ايضا مثل FIREWARE وUSB و I^2C لكن هذه المنافذ تستخدم بروتوكولات مختلفة وتتطلب مكونات مختلفة .

إن الوصلات التسلسلية الاحدث مثل USB و FIREWARE تكون اسرع وذات مميزات أخرى ، وفي الحقيقة مع أن توصيات Microsoft's PC 98 تسمح بمنافذ RS-232 إلا انها تنصح بإستخدام USB عوضا" عن المنافذ التسلسلية RS-232 عندما يكون ذلك ممكنا" في التصاميم الحديثة ، وبالنسبة للعديد من المحيطيات تعتبر الوصلات الحديثة مناسبة أكثر .

ولكن شهرة RS-232 و وصلات الربط المشابهة ستستمر في التطبيقات مثل أنظمة المراقبة والتحكم ، فهي رخيصة وسهلة البرمجة وتسمح بإستخدام كابلات طويلة جدا" سهلة الاستخدام في المتحكمات الصغرية Microcontroller والحواسيب القديمة .

وعندما تصبح USB أكثر شيوعا" ستصبح المحولات Converter متواجدة بوفرة وأسعار مناسبة لتحول من USB الى RS-232 أو RS-485 فالمحول سيوصل الى منفذ USB في الحاسب الشخصي ويترجم بين USB و الوصلات الاخرى ، وسيسهل هذا الترتيب إضافة منفذ RS-232 أو RS-485 الى اي نظام .

تتحكم وحدة UART باستخدام الشريحة 8250 في حواسيب IBM الاصلية بالمنفذ التسلسلي بسرعة قصوى تصل الى bit/sec 57600 ومنذ ذالك الوقت لاتزال UART في الحواسيب الشخصية مستمرة في تقليد هذه الشريحة 8250 الاصلية .

ومع ذلك تضيف الوحدات الاحدث ذواكر مؤقتة وسرعات أعلى بإلإضافة الى مميزات أخرى .

في هذه الايام تكون UART في الغالب في الحواسيب الشخصية جزءا" من الرقاقة متعددة الوظائف التي تحتوي على وحدة أو أكثر من هذه الوحدات مع دعم الParallel port ، ومحركات الاقراص و أجزاء أخرى من النظام .

تقوم UART بالترجمة المتبادلة بين المعطيات المتوازية والمعطيات التسلسلية. و في اتجاه واحد تقوم هذه الوحدات بتحويل المعطيات المتوازية على ناقل النظام System Bus في الحاسب الشخصي الى معطيات تسلسلية بهدف الإرسال وتقوم UART في الاتجاه الاخر بتحويل المعطيات التسلسلية التي تم استقبالها الى معطيات متوازية تقرؤها وحدة المعالجة المركزية CPU على ناقل النظام System Bus .

تدعم هذه الوحدات كل الاتصالات المزدوجة (بإتجاهين) والمزدوجة التناوبية ذات الاتجاه الواحد بنفس الوقت ، وتستطيع الارتباطات مزدوجة البث أن ترسل وتستقبل في نفس الوقت ، أما الاتصالات المزدوجة التاوبية فيستطيع جهاز واحد فقط أن يرسل كل مرة ، بوجود إشارات تحكم أو شيفرات لتحديد الوقت الذي يمكن فيه لكل جهاز ان يرسل .

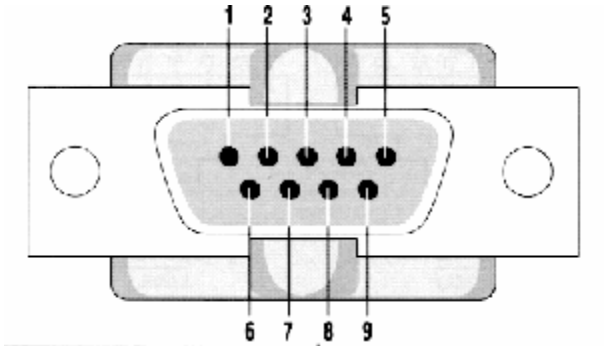
ويكون الازدواج التناوبي مطلوبا" عندما يشترك كلا الاتجاهان في مسلك واحد ، أو عندما يكون هنالك مسلكان ولكن يستطيع حاسب أو حاسبين الاتصال في اتجاه واحد في نفس الوقت .

وتستطيع UART أيضا" أن تدعم الاتصالات المفردة او ذات الاتجاه الواحد فقط

وتدعم UART بالإضافة الى خطوط المعطيات ، مصافحة RS-232 القياسية ، وإشارات تحكم مثل (DCR و DTR و CTS و RTS و RI و CD) .

مهام أقطاب المنفذ:

يمتلك المنفذ تسعة أقطاب موزعة حسب الشكل التالي يتطابق المنفذ التسلسلي الموجود في الحاسب الشخصي من كافة النواحي مع المعيار RS-232 وتنشر المؤسسة الصناعية للإتصالات Telecommunication Industry Association (TIA) وثائق تعرف وظائف الأقطاب والمميزات الاخرى لهذا المعيار والروابط .



Pin	Signal	Pin	Signal
1	Data Carrier Detect	6	Data Set Ready
2	Received Data	7	Request to Send
3	Transmitted Data	8	Clear to Send
4	Data Terminal Ready	9	Ring Indicator
5	Signal Ground		

الجدول ادناه يوضح الاشارات الموجودة على المنفذ حسب المعيار RS-232 بالاضافة الى الوظائف لكل اشارة .

وصف القطب	النوع	المصدر	الاشارة	9 PIN
كشف الحمل	تحكم	DCE	CD	1
المعطيات المستقبلة	معطيات	DCE	RD	2
المعطيات المرسلة	معطيات	DTE	TD	3
جاهزية طرفية المعطيات	تحكم	DTE	DTR	4
أرضي الاشارة	-	-	GND	5
جاهزية مجموعة المعطيات	تحكم	DCE	DSR	6
طلب الارسال	تحكم	DTE	RTS	7
المسح من اجل الارسال	تحكم	DCE	CTS	8
مؤشر الرنين	تحكم	DCE	RI	9

إن الإشارات الاساسية الثلاثة في اتصالات RS-232 ذات الاتجاهين : -

1/ TD يحمل الإشارة المعطيات من DTE الى DCE ويدعى أيضا" TX و TXD .

2/ RD يحمل إشارة من DCE الى DTE ويدعى ايضا" RX و RXD .

3/ SG أرضي الإشارة ويدعى أيضا" GND أو SGND .

أما الاشارات الاخرى فهي إشارات تحكم إختيارية معدة للإتصالات لوظائف مختلفة مثل إستعداد الجهـــاز أو وجود رنين أو إشارات حاملة على خط الهاتف .

وهنالك زوجين من إشارات المصافحة وهما RTS/CTS و DTR/DSR وكل زوج له استخدامات يحـــددها المعيار .

الجهود والتيارات التي يدعمها المنفذ RS-232

يدعم هذا المنفذ جهود غير متوافقة مع منطق TTL حيث يعبر عن 1 منطقي بجد يتراوح بين (-25 to -3) والمنطق المنخفض Logic "0" بجهد يتراوح بين (+25 to +3) ونلاحظ أن المنطق معكوس أيضاً وطبعا هذه الجهود الكبيرة لها فائدة في نقل الإشارة إلى مسافات بعيدة دون ضياع يذكر في قيمة الجهد وكمـــا أنـــه يتحمل هذا المنفذ تيار أعظمي مقداره 500 mA .

وكمـــا نعلـــم تمتلـــك العديـــد مـــن التحكمـــات الـــصغرية Microcontroller منافـــذا" تسلـــسلية غيـــر متزامنة UART/USART وتستخدم مداخل ومخارج هذه المتحكمات جهود المنطق (5V) بدلا" جهود RS-232 ، وحتى نستطيع أن نصل بين المتحكمات والمنفذ التسلسلي يتطلب الوصل بين المنطق (5V) والمنفـــذ RS-232 التحويل من والى مستويات RS-232 .

التحويل من وإلى منطق TTL :

يوحد عدة طرق من أجل مالئمة الإشارات بين المنفذ التسلسلي وبين الدارات التي تعتمد في عملها على منطق TTL واهمها وأشهرها الدارة المتكاملة MAX232 .

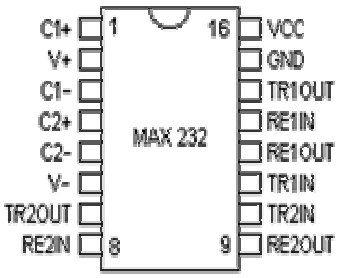
لقد كانت Maximum Semiconductor من أولى الشركات التي وفرت شرائح التحويل من و الى RS-232 والتي تتطلب جهد تغذية (+5V) في حين لم تستطع الشركات الاخرى مثل Dallas و Harris وغيره تصميم هذه الشرائح .

تتضمن الشريحة محولي جهد يعملان كوحدتي تغذية بسيطتين وغير منتظمتين ، تعمل هذه المحولات على تمكين مخارج RS-232 المحملة من جهد (+7) و (-7)

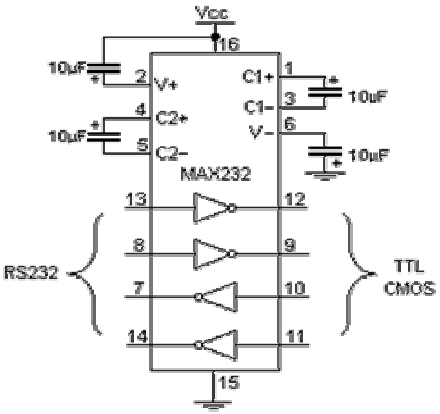
أو أفضل ، وتتضمن أيضا" أربع مكثفات خارجية تستخدم لحفظ القدرة من أجل وحدات التغذية .

أن قيمة سعة هذه المكثفات المقترحة من المعايير القياسية تساوي (10µF) أو أكبر.

الشكل ادناه يوضح شكل الشريحة MAX232 و توزيع PINS الخاصة بها :-



الشكل ادناه يوضح الطريقة المتبعة لتوصيل المكثفات الخارجية :-



وهنالك مجموعة أخرى من شرائح الربط مع RS-232 .

ضمن هذه الشرائح شريحة MAX232A القديمة ، وهي قادة على التعامل مع سرعات عالية بإستخدام مكثفة بسعة (0.1µF) وتعتبر MAX233 اكثر ملاءمة لأنها لا تحتاج الى مكثفات خارجية إطلاقا" (ولكنها تكلف اكثر) .

تعتبر MAX3221 أحد خيارات الارتباط RS-232 التي تستخدم دارة قيادة واحدة ومستقبل واحد لانه من النادر وجود شرائح ملاءمة لهذا النوع من الارتباطات .

تشتمل العديد من الشرائح الحديثة ميزة الاغلاق توفير للطاقة ، حيث أن هذه الميزة تجعل الشريحة تعمل في وضع توفير إنقاص إستخدام الطاقة .

تمتلك بعض الشرائح دخل تمكين منفصل يعمل على تمكين المستقبل في إستقبال المعطيات عندما تكون الشريحة تعمل في نمط الإغلاق توفيرا" للطاقة .

تمتلك الشريحة MAX3212 ميزتين إضافتين في توفير الطاقة ، فعندما يكون الجهد على الدخل جهد RS-232 غير صحيح (الشي الذي يحدث عندما تكون دارة القيادة البعيدة غير فعالة) تنتقل الشريحة اليا" الى نمط الطاقة المنخفضة ، أما اذا كانت المداخل صحيحة ، ولكن خاملة فإنه يتم إكتشاف الخرج عندما تحدث عملية الإرسال على المداخل . تفيد هذه الاشارة في إستنهاض المتحكم الصغري MICROCONTROLLER في نمط SLEEP MODE من أجل معالجة المعطيات القادمة اليه .

تقوم الدارات DS275 و DS276 المصنعة من قبل شركة Dallas Semiconductor بإستعارة الجهد من النهاية العكسية للوصلة بدلا" من توليد جهود سلبية من مصادرخارجية .

كانت وصلات RS-232 قبل ظهور MAX232 تستخدم زوجا" من المستقبلات وزوجا" من القيادة MC148819 الثنائية حيث تحتاج عائلة 1488 جهود تغذية سالبة و موجبة . وتعمل عائلة 1489 جهود تغذية تبدأ من 5V وتقبل دخول حتى (+30V) و (-30V) .إذا تم تامين الدارة بجهود التغذية السالبة و الموجبة المتوفرة ، وخصوصا" إذا كان مطلوبا" أربع دارات قيادة و أربع مستقبلات ، فإن استخدام زوج العائلات 1488/89 هو الحل البديل .

ان البديل الاخرللمحولات الخاصة بك (التي يبنيها المصمم بنفسه) هو وحات قياسية متوفرة يمكن الاعتماد عليها .

تأخير الكوابل :

أن زمن الكابل هو مفهوم اخر لفهم الخطوط الطويلة والقصيرة ، التأخير وحيد الاتجاه هو الزمن الذى تحتاجه الاشارة لتجتاز الكابل بطوله الكامل ، ويساوي هذا الزمن الطول الفيزيائي للكابل مقسوما" على معدل أو سرعة إنتشار الإشارة في الكابل.

بالطبع تكون معدلات الانتشار سريعة جدا" فالضوء ينتشر في الفراغ بسرعة تصل الى 300 مليون متر في الثانية (186.000 ميلا" في الثانية أو 12 بوصة في النانو ثانية) .

تنتشر الإشارة الكهربائية في السلك النحاسي بسرعة تصل الى 2/3 الى 3/4 من هذه السرعة أي 200 مليون متر في الثانية (124.000 ميل في الثانية ،8 بوصة في النانو الثانية) و 225 مليون مترفي الثانية (140.000 ميل في الثانية أو 9 بوصة في النانو ثانية) كما يسمى معدل الإنتشار ايضا" بسرعة الإرسال .

ولان التاخيرات صغيرة ، فلن يكون لها ايق نتائج على الإطلاق عندما يكون الكابل قصيرا" . وزمن الصعود Rise time بطيئا" (زمن الصعود هو الزمن المطلوب لتحويل الخرج من 10% الى 90% من كامل المجال كما يعرف ايضا" بزمن الانتقال ويشير الى معدل القفز معدل تغير الاشارة) ولكن بالنسبة للكابلات الطويلة الحاملة للإشارات ذات سرعة الإنتقال الكبيرة ، يمكن للتاخيرات ان تكون طويلة بما يكفي لينتج عنها إنعكاسات تؤثر على مستويات المنطق LOGIC التي يقرؤها المستقبل .

أن العبارة الأخيرة ذات الصلة بالموضوع هي تاخير الانتشار (Propagation Delay) أو الطول الكهربائي (electrical length) ، وهو يساوي (1 على معدل الانتشار) ويعبر عنه بالزمن على وحدة الطول . فتأخر انتشار الضوء في الخلاء يساوي 85 بيكو ثانية/ البوصة ، وعند 3/2 من هذه السرعة يكون تأخير إنتشار الإشارة في سلك نحاسي (125 بيكو ثانية/البوصة) .

هناك طريقة اخرى لإجاد التاخير وحيد الإتجاه ، وهي القيام بضرب تأخير الإنتشار بطول الكابل .

بنية الإشارة التسلسلية في وصلة RS-232:

تعم وصلة RS-232 ارسال البيانات بطول 8 bits وعندما نقوم بارسال بايت المعطيات عبر المنفذ التسلسلي باستخام وصلة RS-232 يتم اضافة بعض البيتات من أجل إتمام عملية الإرسال بنجاح وتتألف إشارة الإرسال الواحدة من:

1: بت البداية (Start Bit):

عندما لايكون هناك ارسال للمعلومات "أي في حالة البطالة" يكون قيمة القطب المسؤول عن ارسال المعلومات TX مساوي "1" وعنما يتم الإرسال يصبح قية هذا القطب مساوي للصفر ليخبر الطرف المستقبل على ان الإرسال قد بدأ ويقوم المستقبل بإرسال البايت المرسل بعد خانة البداية .

3:بتات المعطيات (Data bit):

وهي عبارة عن 8bits تشكل القيمة المراد إرسالها وترسل بشكل تسلسلي بسرعة يتم تحديدها بواسطة معدل بود حيث يتم إرسال البت الأقل أهمية أولاً (LSB) .

2: بت الإنجابية (Parity bit) :

يستخدم هذا البيت من أجل ضمان أن المعطيات لم تتعرض للضياع أثناء الإرسال حيث يأخذ القيمة "1" إذا كان عدد الواحدات المنطقية في البايت المرسل زوجي ويأخذ القيمة "0" إذا كان فردي ويمكن أن يتم إهمال هذا القطب أثناء الإرسال

4: بت التوقف (Stop bit) :

وهوبت يدل على إنهاء عملية الإرسال ويأخذ القيمة "1" وتيقى هذه القيمة على القطب TXD حتى يتم إرسال بايت أخر .

Stop	Parity	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Start
1	1/0/None	*	*	*	*	*	*	*	*	0

معدل بود Baud Rate :

ستخدم معدل بود من أجل تحديد سرعة نقل المعلومات بين طرف الإرسال والإستقبال ويجب أن يكون لكلا الطرفين نفس السرعة وإلا فإن المعلومات سوف تنتقل بشكل خاطئ . وهو بمعنى آخر يحدد عرض بت الإرسال من أجل فحصه عند طرف الإرسال وهناك سرعات عدة وأشهرها:

300-1200-2400-2800-9600-19200-38400-57600-115200(bps - bit per second)

وهذا التنوع جاء على حساب المسافة حيث كلما إزدادت السرغة نقصت المسافة وبالعكس والجدول التالي يحدد علاقة السرعة بالمسافة :

Baud Rate	Maximum Distance (in feet)
2400	3000
4800	1000
9600	500
19200	50

التخاطب مع المنفذ بلغة C++:

يوجد ضمن مكتبة bios.h توابع جاهزة تمكننا من التعامل مع منفذ COM بسهولة أما التوابع التي نستخدمها من اجل التخاطب فهي :

1: bioscom (int cmd, char byte, int port);
2: bios_serialcom(int cmd, char byte, int port);

ففي بداية الإرسال يجب أن يتم تحديد السرعة التي سوف يتم نقل البيانات بواسطتها و رقم المنفذ وعدد بتات التوقف وبت الإنجابية ويتم ذلك كما يلي :

cmd : عبارة عن متحول من النوع int يأخذ أربعة قيم وكل قيمة لها وظيفة خاصة :

cmd	الوظيفة
0	تهيئة بارمترات المنفذ
1	إرسال بايت عبر المنفذ
2	إستقبال بايت عبر المنفذ
3	إرجاع حالة الإتصال عبر المنفذ

اما port فهي تدل على المنفذ الذي سوف يتم الإرسال/ الإستقبال بواسطته ويأخذ القيم :

Port: 0=com1 1=com2 2=com3 3=com4

أما byte فهي القيمة المراد إرسالها عبر المنفذ وفي حالة الإستقبال يتم تجاهل هذه القيمة .

وإذا كانت cmd=0 عنده قيمة البايت يتم تحديد بعملية OR بين البايتات التالية :

وصف التعليمة	قيمة البايت byte
طول بايت المعطيات 7bits طول بايت المعطيات 8bits	0x02 0x03
بت واحد للتوقف بتان لخانة التوقف	0x00 0x04
بدون خانة الإنجابية خانة الإنجابية للفردي خانة الإنجابية للزوجي	0x00 0x08 0x10
معدل بود 110	0x00
معدل بود 150	0x20
معدل بود 300	0x40
معدل بود 600	0x60
معدل بود 1200	0x80
معدل بود 2400	0xA0
معدل بود 4800	0xC0
معدل بود 9600	0xE0

مثلاً إذا أردنا أن يعمل المنفذ التسلسلي COM1 بنظام 8bits وخانة واحدة للتوقف وإهمال بت الإنجابية وبمعدل بود 2400 عندها يكون البايت المرسل مساوي :

Byte= 0x03|0x04|0x00|0xA0= 0xA7;

مسجل الحالة :

يوجد مسجل للحالة بطول 16 bit يتألف من بايت سفلي يدعى مسجل التحكم بالمودم وبايت علوي يدعى مسجل التحكم بالخط البايت العلوي ومن إسمه فهو يتعلق بالمودم لذلك سوف لن نتطرق إليه أما البايت السفلي فإن موضح بالشكل التالي :

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
R	R	R	R	R	R	R	R

إن مسجل الحالة هو مسجل قابل للقراءة فقط .

الخانة 15 - خطأ في الإستقبال FIFO: Error in Received FIFO

تصبح هذه الخانة 1 عند إكتشاف خطأ في عملية الإستقبال كخطأ في خانة الإنجابية أو في المرسل كإستخدام معدل بود غير المحدد في الجهاز المستقبل .

الخانة 14 - فراغ مسجلات المعطيات :Empty Data Holding Registers

عندما تكون هذه الخانة1 فذلك يعني أن كل من مسجل الإرسال ومسجل الإزاحة فارغيين

الخانة 13 - خانة فراغ مسجل الإرسال :Empty Transmitter Holding Register

تكون هذه الخانة 1 عندما يكون مسجل الإرسال فارغ فقط أما مسجل الإزاحة فيكون مشغول .

الخانة 12 - خانة إلغاء المقاطعة Break Interrupt

تكون هذه الخانة 1 عندما يكون خط إستقبال المعطيات ممسوك إلى المنطق المنخفض '0' لمدة أطول من المدة الازمة لإرسال كلمة كاملة مع ماتتضمن هذه الكلمة من خانة البدء و الإنجابية و خانات التوقف .

الخانة 11 - خطأ في الهيكلية :Framing Error

تكون هذه الخانة 1 عنما تكون الخانة الأخير ليست خانة التوقف ويحدث ذلك عندما يكون هناك خطأ في عملية المزامنة بين المرسل والمستقبل .

الخانة 10 - خطأ في خانة فحص الإنجابية : Parity Error

تكون هذه الخانة 1 عندما عند إكتشاف خطأ في خانة فحص الإنجابية .

الخانة 9 - خانة خطأ التجاوز :Overrun Error

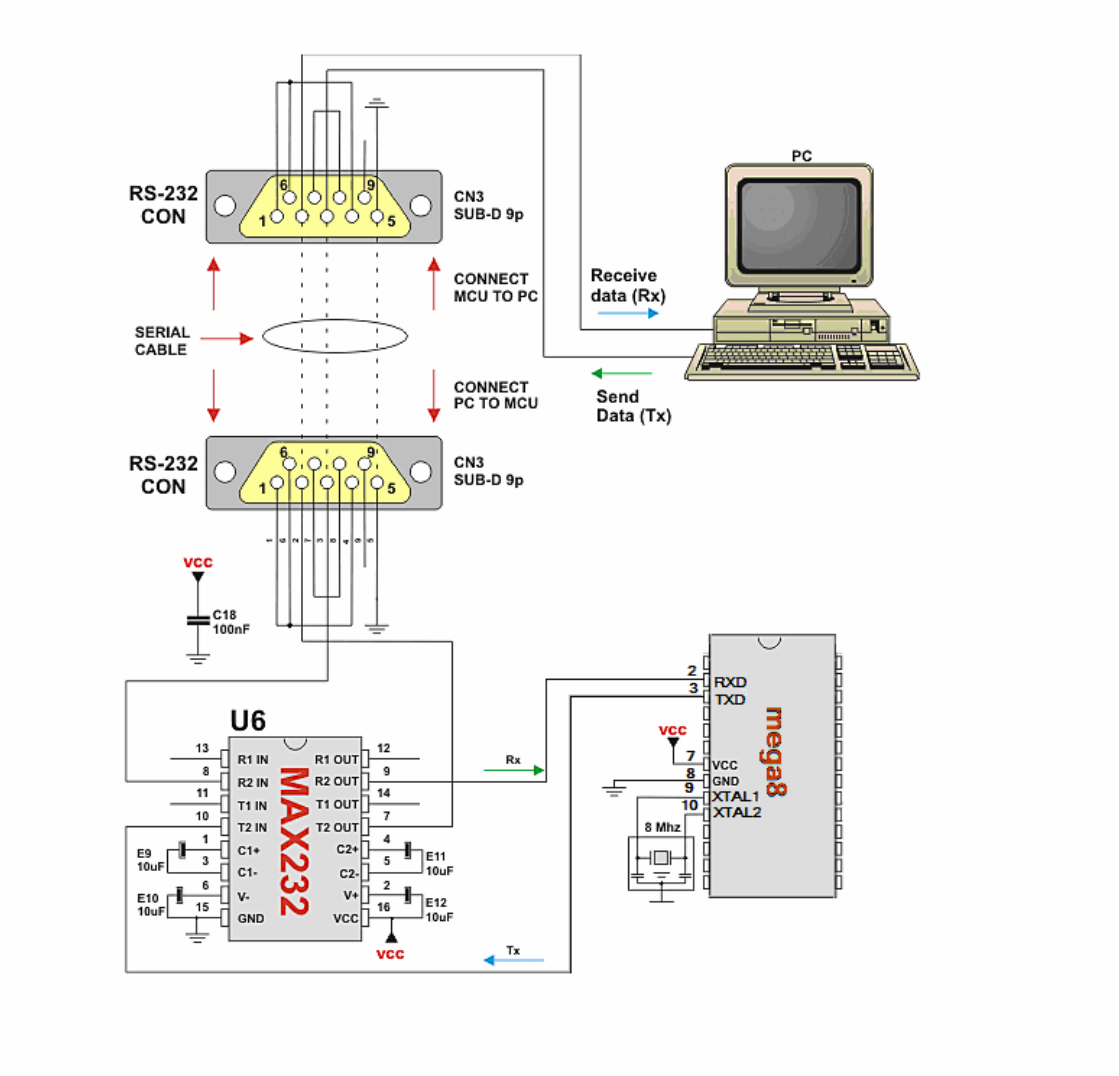
تكون هذه الخانة 1 عندما يتم إرسال بايت معطيات جديد ولم نقم في برنامجنا بقراءة البايت المرسل السابق حيث يتم ضياع معلومات البايت السابق.

الخانة 8 - خانة جاهزية المعطيات :Data Ready

عنما تكون هذه الخانة 1 يعني ذلك بأنه قد تم استقبال بايت وهو الآن موجود في مسجل الإستقبال وهو جاهز الآن للقراءة .

وصل المتحكم مع المنفذ التسلسلي :

الشكل التالي يوضح لنا طريقة وصل المتحكم مع المنفذ مع وجود دارة ملائمة بين المـــتحكم والمنفـــذ وهـــي الدارة المتكاملة MAX232



البرنامج التالي يقوم بإستقبال وإرسال مجموعة من الأحرف بين جهازي الإستقبال والإرسال :

شرح بعض التوابع :

int Putch(int c)

يقوم هذا التابع بإرجاع الشيفرة المقابلة لقيمة c في شيفرة الأسكي .

int kbhit(void)

يقوم هذا التابع بإرجاع 1 عند كشف ضغط على أي زر من أزرار لوحة المفاتيح .

int getch(void)

يقوم هذا التابع بإرجاع القيمة العددية المقابلة للزر المضغوط من أزرار لوحة المفاتيح بناءً على شيفرة الأسكي أي أنه يقوم بعكس وظيفة التابع Putch.

وجميع هذه التوابع محتواة في المكتبة conio لذلك يجب إدراجها في بداية البرنامج .

```
#include <bios.h>
#include <conio.h>
#define COM1        0
#define DATA_READY 0x100
#define SETTINGS ( 0x30 | 0x04 | 0x00 | 0xA0)
int main(void)
{
   int in, out, status;
   bioscom(0, SETTINGS, COM1); /*initialize the port*/
   cprintf("Data sent to you:  ");
   while (1)
   {
      status = bioscom(3, 0, COM1); /*wait until get a
data*/

      if (status & DATA_READY) // If any previous
unwanted data has been received

           if ((out = bioscom(2, 0, COM1) & 0x7F) != 0)
/*input a data*/
              putch(out); /*print a character of input
data */
           if (kbhit())   /* if any button has been
pressed */
           {
              if ((in = getch()) == 27)   /*if the
pressed button was Esc (27) ASCII of Esc*/
                 break; /* out of while */
              bioscom(1, in, COM1);   /*output a data*/
           }
   }
   return 0;
}
```

المشروع السادس عشر
الربط باستخدام البوابات التفرعية
PARALLEL PORT

مقدمة :

هناك العديد من طرائق الاتصال بالحاسب مع الأجهزة المحيطة وتختلف هذه الطرائق من حيث تقنية التخاطب وسرعة الارسال والاستقبال وكذلك حجم المعلومات المنتقلة وطريقة نقلها .

ومن هذه الطرائق المستخدمة لدينا مثلاً:

- بوابة الـــCOM (الاتصال التسلسلي).
- بوابة الـــLPT (الاتصال التفرعي).
- كرت الـــISA:ولكن هذا النوع من الاتصال يتم انشاؤه من قبل المستخدم.
- بوابة الألعابGame Port.
- حديثاً وصلة الـــUSB.

لكل طريقة لها محاسنها ولها مساوئها ولكن يمكن للمبرمج استخدام الاتصال الذي يفي بغرضه وبمشروعه .

وفي مشروعنا (PWM) استخدمنا بوابة LPT في الاتصال مع دارتنا وذلك من أجل برمجة الـــCTC.

واليكم الآن الشرح المفصل عن هذه البوابة وطرائق التعامل معها.....

٥-٢ عناوين الـــLPT:

قبل البدء في الشرح يجب أن ننوه الى أنه يجب عند التعامل مع أي بوابة أن نكون على علم بعناوينها وذلك من أجـــل أن نستطيع برمجتها(أي ارسال واستقبال البيانات).

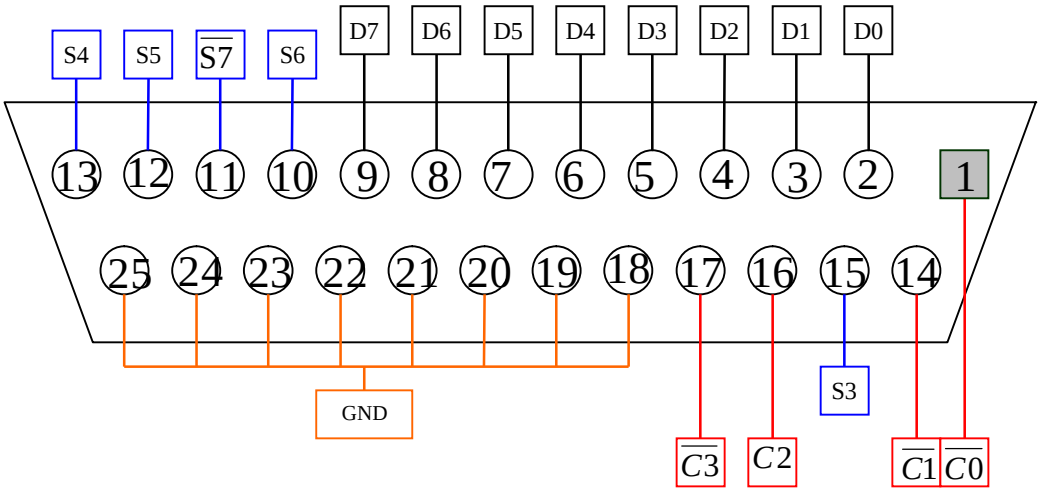
وبما أن بوابة الـــ LPT تأتي مع ال Motherboard مدمجة لذلك قامت جميع الشركات بوضع عناوين موحدة لهذه البوابة من أجل تلافي الوقوع في الأخطاء وفيما يلي الجدول الذي يبين عناوين الادخال والاخراج والمعلومات :

	HEX			DECIMAL		
	DATA	IN	OUT	DATA	IN	OUT
LPT1	378	379	37A	888	889	890
LPT2	278	279	27A	632	633	634
LPT3	3BC	3BD	4BE	956	957	958

ولكن حالياً تم تنسيق الـــLPT2 & LPT3 وبقي الـــLPT1 بسبب تطور مداخل الـــUSB .

٥ - ٣ مخطط البوابة التفرعية LPT :

تتألف البوابة LPT من ٢٥ رجل موزعة بين دخل وخرج ومعطيات وخطوط أرضي كما يبينه الشكل التالي:



الشـــــــــــكـل(٥ - ١)

تتألف بوابة الـــLPT من الأرجل التالية :

- ٨ أرجل من أجل المعطيات (D0~D7).
- ٥ أرجل للدخل (S3~S7).
- ٤ أرجل للخرج (C0~C3).
- ٨ أرجل للأرضي (GND).
-

٥ - ٤ أنماط عمل بوابة الـــ LPT:

- النمط العادي :يكون في هذه الحالة جميع خطوط المعطيات تعمل كمخارج فقط.
- النمط المطور : يكون في هذه الحالة خطوط المعطيات كدخل وكخرج معاً أي أنهـا ثنائيـة الاتجـاه ويـتم الحصول على النمط المطور بارسال واحد منطقي على المخرج الخامس أي OUT5=1 ولكـن كمـا هـو ملاحظ أن هذا المخرج غير موجود في أرجل الـــLPT ولكن داخلياً يتواجد هذا المخـرج ، حيـث يمكـن برمجته بواسطة الحاسب بأحد لغات البرمجة المعروفة مثل (c++ أو VB6).

٥ - ٥ برمجة الـــLPT:

يمكن أن تبرمج الـــ LPT بكثير من لغات البرمجة منها C++ أو الـــASSMBLY أو VB6 ....ولكن جميع هـذه اللغات تستخدم نفس التعليمات من أجل الاخراج أو الادخال ولكن بطرائق مختلفة :

من أجل الاخراج→ OUT(الرقم المراد ارساله ، عنوان الاخراج الخاص بالبوابة)

من أجل الادخال→ متحول=IN(عنوان الادخال الخاص بالبوابة)

مثال (على لغة الــVB6):

سوف نأخذ مثال على اضاءة ليدات بشكل تتابعي :

```
For i = 0 To 7
OUT "&H378", 2 ^ i

حلقة تأخير زمني '

   For J = 1 To 10000
   Next J
i = i + 1
Next i
```

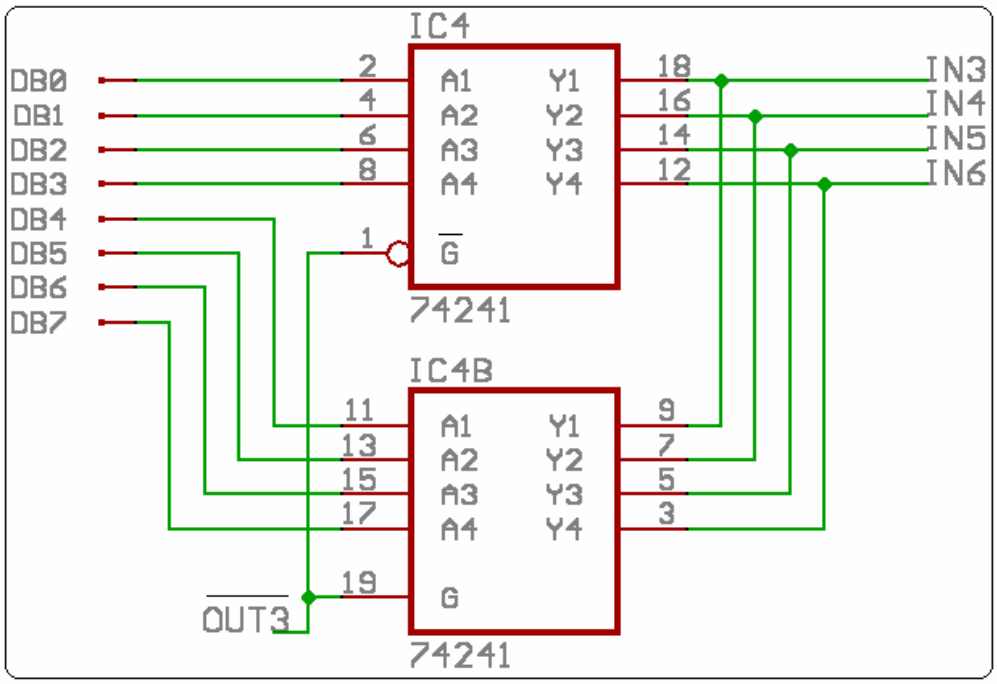
هذا يعني أننا أرسلنا المعلومات الى أرجل المعطيات وذلك واضح من العنوان(&H378).

ملاحظات هامة من أجل استخدامات الــLPT:

- يجب الانتباه أولاً الى نوع المعلومات المستقبلة فاذا كانت من النوع ذو التغير البطيء(مثــل تغيــر درجــات الحرارة) فإننا يمكن التعامل مع البوابة بشكل طبيعي أما اذا كانت المعلومات المستقبلة ســريعة جــداً (مثــل تغيرات الصوت)فإنه يجب استخدام الماسكات(LS74245) من أجل مسك المعلومات قبل ادخالها وخاصــة عند ادخال أكثر من بايت على الــLPT .
- يجب الانتباه الى أن الــ LPT تستطيع استقبال دخل منطقي واخراج خرج منطقي وذلك بقيمة (0-5)فولط لذلك يجب الانتباه عند ادخال جهود كبيرة أن نستخدم محددات جهد أو مقسمات جهد وذلك من أجل المحافظة على الــ LPT من العطب.
- يجب الانتباه الى سحب التيار الكبير الذي قد يحدث نتيجة الدارات القصيرة أو نتيجة التحميــل الزائــد علــى البوابة لذلك ينصح باستخدام الــPhotocoupler أي العوازل الضوئية .
- يجب الانتباه الى وصل الأرضي الى الــLPT الى أحد أرجل الأرضي لأنه لايولد أرضي بشكل تلقائي .

إدخال بايت كامل: Full byte input

من الواضح ظأنه لا يمكن إدخال بايت كامل دفعة واحدة عبر خطــوط الــدخل IN3-IN7(لأن عــددها أقــل مــن ثمانية).ويمكن عاى سبيل المثال استخدام أحد خطوط الخرج out x كخط انتخاب ، حيــث يــتم انتخــاب وإدخــال الخانات الأربعة الدنيا عندما يكون خط الإنتخاب مساوياً للصفر مثلاً ويتم في المرحلة الثانية انتخاب وادخال الخانــات الأربعة العليا بجعل خط الانتخاب مساوياً للواحد كما هو مبين في الشكل .



الشكل استخدام البوابة التفرعية كبوابة دخل رقمية ثمانية الخانة

نضع دارتي عازل ثلاثي الحالات(Tristate Buffer)،كل منها بعرض أربعة خانات، ونصل مخارج هذه العازلين إلى خطوط الدخل للبوابة التفرعية . ويقوم أحد خطوط الخرج x out بقيادة مدخل التأهيل (Enable) لأحد العزلين ، بينما تتم قيادة مدخل تأهيل العازل الآخر عبر نفس خط الخرج ولكن بعد مروره بعاكس (inverter) . هكذا نضمن أنه لن يتم تأهيل إلاّ أحد العازلين في نفس اللحظة ، ويمكن اختيار تأهيل أحدهما أو الآخر بإخراج القيمة المناسبة على خط الخرج المستخدم .

وهكذا يتم إدخال البايت الكامل بإدخال الخانات الأربعة الدنيا D0-D3 عبر العازل الأول ، والخانات الأربعة العليا D4-D7 عبر العازل الثاني ، ومن ثم يقوم البرنامج المكتوب بتشكيل القيمة الصحيحة للبايت الكامل D0-D7 ابتداءً من القيمتين المقروءتين. من الواضح أن هذه الطريقة ليست مباشرة وسريعة كما لو كنا نقرأ البايت دفعة واحدة ، ولكن حتى مع الحواسيب البطية نسبياً فإن معدلات سرعة النقل بهذه الطريقة يمكن أن تبلغ عدة مئات الكيلو بايتات في الثانية KBPS.

لا بد من الانتباه لوجود مشكلة بسيطة: وهي احتمال تغير قيمة المعطيات في الفترة الواقعةبين قراءة النصف الأول والنصف الثاني . وهذه المشكلة ستكون موجودة كلما كان عرض بوابة الدخل أقل من عرض بوابة المعطيات التي نود قراءتها.

في كثير من التطبيقات يمكن تجاوز هذه المشكلة دون إضافة أي دارات متكاملة إضافية ، وذلك عندما تكون سرعة تبدل المعطيات قليلة نسبياً، ولكن مع ذلك يفضل اتخاذ بعض التدابير لضمان صحة قراءة المعطيات . فمثلاً يمكن اللجوء إلى تنفيذ القراءة مرتين أو أكثر ومقارنة هذه القراءات ، فإذا كانت النتيجة هي نفسها في القراءات المتتالية فهذا يعني أن المعطيات لم تتغير أثناء إجراء هذه القراءات (باحتمال كبير).

أما إذا أردنا ضمان صحة القراءة 100% فلا بد من استخدام ماسك (Latch) للبايت بكامله قبل قراءته . فبواسطة أحد خطوط الخرج Out x نقوم بمسك البايت كاملاً ، ومن ثم قراءة البيت الممسوك على دفعتين من خرج الماسك .

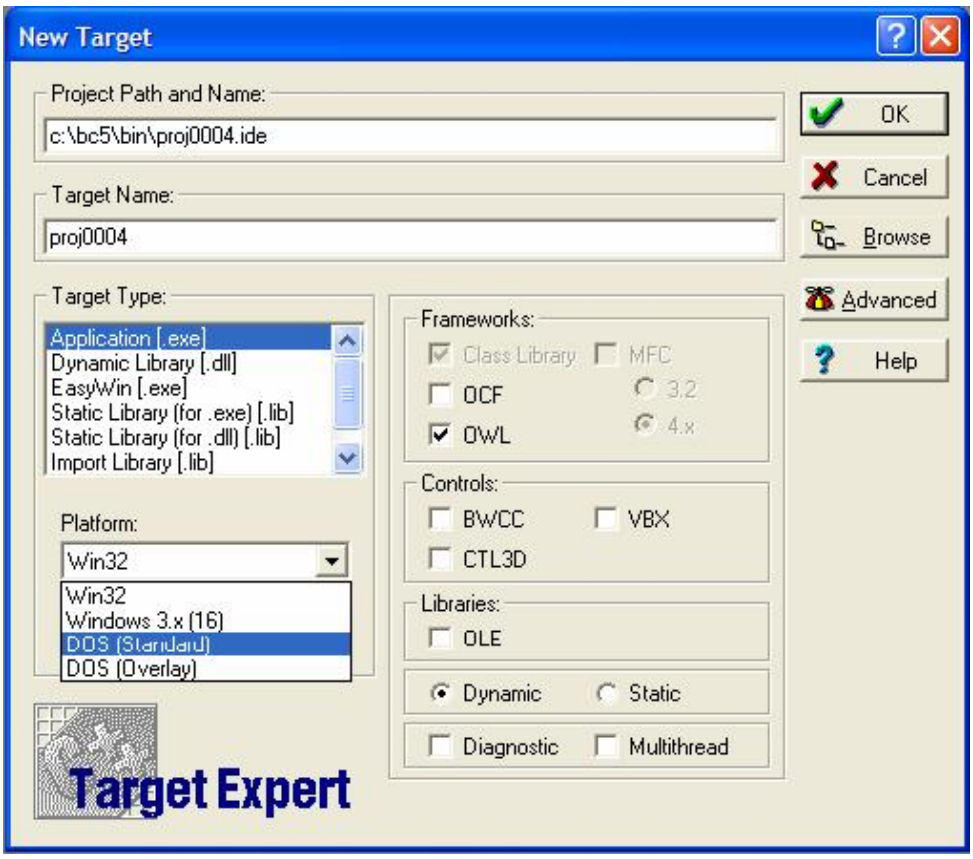
إن ضمان صحة القراءة 100% سيكون على حساب الكلف الزائدة ثمناً للماسك، وعلى حساب انخفاض معدل سرعة نقل المعطيات قليلاً بسبب الحاجة لتنفيذ تعليمتين إضافيّتين : تعليمة مسك المعطيات قبل إجراء القراءة ، وتعليمة العودة لمتابعةتغير المعطيات بعد إجراء عملية القراءة .

بعد هذه البداية التي تعرفنا من خلالها على بنية البوابة التفرعية بقي السؤال كيف نستطيع استخدامه عملياً وهوالشيء المهم لأن علم بلا عمل لاشيء وإنما مضيعةً للوقت .

استخدام لغة c من أجل التخاطب مع المنفذ التفرعي :

تعتبر لغة C من أهم الغات العالية المستوى والتي تمتاز بقوتها ومرونتها وتعتبر أيضاً لغة غرضية التوجه كما أنها تحتوي على مكاتب عديدة تسهل لنا إجراء عمليات الإخراج والإظهار والعمليات الحسابية والتي اتلزممنا خلال كتابة البرنامج .

سوف نستخدم هنا مترجم شركة Borland المشهورة الإصدار رقم ٥,٢ عند فتح البرنامج يظهر لنا نافذة العمل فنختار من File العنوان New ومنه نختار الأمر project وعند ذلك يظهر لنا نافذة صغيرة نكتب فيها إسم المشروع وهناك أمر مهم إذ يجب إختيار platform الخاص الذي يمكننا من التخاطب مع منفذ الحاسب وهو (dos standard) كما في الشكل التالي :



يتألف أي برنامج في لغة C++ من ثلاث مناطق وهي منطقة التصريحات العامة ومنطقة المتحولات ومنطقة البرنامج وتحتوي منطقة التصريحات العامة على المكتبات المراد استخدامها في البرنامج وهنا في مشروعنا يوجد ثلاث مكتبات وهي :

#include<iostream.h>

وهي المسؤلة عن عمليات الإخراج والإدخال وبها نستطيع إدخال الإرقام من لوحة المفاتيح لإستخدامها في البرنامج .

#include<dos .h>

وهي المسؤلة عن عملية التخاطب بين البرنامج والمنفذ التفرعي .

#include<conio.h>

وهي مسؤلة عن عملية الإظهار على شاشة الحاسب وإجراء بعض تنسيقات الإظهار .

كيفية الإخراج على المنفذ التفرعي :

البرنامج التالي يخرج لنا القيمة ١٢٨ على المنفذ التفرعي وهذه القيمة تساوي FF بالنظام الستة عشري .
التابع getch() يستخدم من أجل مسك حالة الخرج عند أخر قيمة تم إخراجها .

```
#include<iostream.h>
#include<dos .h>
#include<conio.h>
Void main ()
{
Outport (0x378, 0xff);
getch ();
}
```

عملية الإدخال :

كما شرحنا يوجد هناك خمسة مداخل في البوابة التفرعية مهي من (IN3----IN7) حيث IN7 يكون منفي وغالباً لايستخدم للإدخال ولكن في الحقيقة يوجد هناك مسجل بطول ٨ bit داخل الكومبيوتر ولكننا لانستطيع أن نستخدم إلا خمسة فقط أما (IN0-IN1-IN2) فهي داخل الحاسب وتأخذ قيم عشوائية لذا عملية قراءة عنوان الدخل يجب التخلص من هذه القيم والطريقة هي كالتالي : نقرأ المدخل ثم نعمل عملية & مع القيمة ٠١١١١٠٠٠ حيث نكون قد تخلصنا من القيم العشوائية الثلاث ومن الخانة السابعة المنفية وذلك بعملية تصفير هذه الخانات بقي هناك مشكلة إزاحة العدد ثلاث خانات إلى اليمين من أجل إظهار الرقم صحيح لذلك نقوم بقسمة العدد على ٨ أما البرنامج فيكون كالتالي :

```
#include<iostream.h>
#include<dos .h>
#include<conio.h>
Void main ()
```

```
{
Int x;
x=intport (0x379);
x=x&120;
x=x/8;
cout<<”x=”<<x;

}
```

تشكيل المؤقت الزمني :

لتشكيل المؤقت الزمني يوجد هناك تعليمة DELAY(X) حيث تقوم هذه التعليمة بتأخير 0.001 SECOND وقيمة x هي التي تدل على قيمة التأخير مثلاً x=5000 اي يتم التأخير ٥ثوان .

يوجد هناك طريقة أخرى وهي عن طريق تشكيل حلقة كبيرة يتم المعالج بالدخول فيها وبالتالي تأخذ وقتاً منه حتى يتممها ولكن هذه الطريقة ليست دقيقة بمعرفة الزمن .

المنفذ التفرعي في النمط المطور:

في النمط المطور يكون البوابة DATA إما بوابة دخل (Input) أو بوابة خرج (Output) ويتم اختيار أحد النمطين من مسجل بوابة التحكم (Control) .

فكما معلم أن المسجل هو حجرة ذاكرية في المعالج وهذه الحجرة تكون بطول 1بايت بالنسبة لبوابة التحكم وهي موزعة كما نعلم :

x	x	x	x	$\overline{Out3}$	$Out2$	$\overline{Out1}$	$\overline{Out0}$
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

حتى الآن كنا نستخدم الخطوط من Out0 وحتى Out3 وقلنا أن الخطوط الأربعة الباقية موجودة ولكن ليس على المنفذ ولكن على اللوحة الأم وهذه الخطوط يمكن أن تأخذ القيمة أيضاً ولكن بطريقة برمجية

وفي النمط المطور سيكون لهذا المسجل دور في تفعيل أو إلغاء تفعيل جهة بوابة المعطيات DATA والبت الذي سيتولى ذلك هو Out5 .

x	x	$Out5$	x	$\overline{Out3}$	$Out2$	$\overline{Out1}$	$\overline{Out0}$
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

فعندما يأخذ القيمة "1" واحد تكون البوابة كدخل وإذا كانت"0" تكون خرج وينم ذلك :

```
Outport(0x37a,(1<<5)) ;  // data now is Input
Outport(0x37a,(0<<5)) ;  // data now is Output
```

المراجع العربية :

[1] برمجة متحكمات AVR محمد عبد المعطي شد
[2] برمجة متحكمات AVR بلغة الأسمبلي تميم الأمير ، حيان شحيمة
[3] قيادة محرك خطوي باستخدام متحكمات AVR أحمد سكري ، أنس دباغ

المراجع الأجنبية :

[1] John Morton, AVR an Introductory course.
[2] DHANANJAY V.GADRE, Programming and Customizing the AVR Microcontroller.
[3] VAN SICKLE, T. (2001). Programming Microcontrollers in C (2nd ed.)