بسم الله الرحمن الرحيم

وزارة التعليم العالي والبحث العلمي

جامعة الشيخ عبد الله البدري

كلية التكنلوجيـــا

قسم الهندسه الكهربائيه

بحث تكميلي لنيل درجة الدبلوم في تقنية الهندسة الكهربائية تخصص إلكترونيـــــــات

بعنوان :-

تصميم منظم مروحة يتم التحكم فيه عن بعد

Design of Remote controlled fan Regulator

إعداد الطلاب :-

i. محمد علي أبشر محمد أحمد .
ii. صالح عبد الله صالح .
iii. عزام هاشم الحاج .
iv. تقوي عثمان إبراهيم .
v. جيهان محمد بلوله .

إشراف :-

أ/مهند كمال

مايو 2012

بِسْمِ اللهِ الرَّحْمنِ الرَّحِيمِ

اقْرَأْ بِاسْمِ رَبِّكَ الَّذِي خَلَقَ (1) خَلَقَ الْإِنسَانَ مِنْ عَلَقٍ

(2) اقْرَأْ وَرَبُّكَ الْأَكْرَمُ (3) الَّذِي عَلَّمَ بِالْقَلَمِ

(4) عَلَّمَ الْإِنسَانَ مَا لَمْ يَعْلَمْ (5)

صدق الله العظيم

الإهداء

(قل إعملوا فسيرى الله عملكم ورسوله والمؤمنون)

صدق الله العظيم

إلهي لايطيب الليل إلا بشكرك ولايطيب النهار إلا بطاعتك .. ولاتطيب اللحظات إلا بذكرك .. ولا تطيب الآخرة إلا بعفوك .. ولا تطيب الجنة إلا برؤيتك الله جل جلاله

إلى من بلغ الرسالة وأدى الأمانة .. ونصح الأمة .. إلى نبي الرحمة ونور العالمين ..

سيدنا محمد صلى الله عليه وسلم

إلى من كلله الله بالهيبة والوقار .. إلى من علمني العطاء بدون انتظار .. إلى من أحمل أسمه بكل افتخار .. أرجو من الله أن يمد في عمرك لترى ثماراً قد حان قطافها بعد طول انتظار وستبقى كلماتك نجوم أهتدي بها اليوم وفي الغد وإلى الأبد ..

والدي العزيز

إلى ملاكي في الحياة .. إلى معنى الحب وإلى معنى الحنان والتفاني .. إلى بسمةالحياة وسر الوجود

إلى من كان دعائها سر نجاحي وحنانها بلسم جراحي إلى أغلى الحبايب

أمي الحبيبة

الشكر والعرفان

إلى الشموع التي ذابت في كبرياء.....

لتنير كل خطوة في دربنا .....

لتذلل كل عائق أمامنا .....

فكانوا رسلاً للعلم والأخلاق .....

شكراً لكم جميعاً .....

الشكر موصول الي :-

i. أ/ مهند كمال .
ii. أ/ إبراهيم الريشابي .
iii. أ/ أحمد صلاح .

الملخص

منهجية التصميم التي تناولها البحث تعمل علي تقسيم التصميم الـــي كتل وظيفيــه function block وكل كتله وظيفيه تعتبر دائره لهـــا وظيفتها الخاصه .

يهتم هذا البحث ببرتوكول NEC وذلك لكثرة استخدامه ، وبساطــة التصميم ، معظم المبرمجين يستخدمون هذا البرتوكول ولكل مبرمج شفرته الخاصة .

Abstract

The Design methodology In This research Is division Into design into function block ,each block Show Electronic circuit and its special function

The Communication protocol which is used in this research is NEC protocol , because of common use from many manufacture and simple design.

All manufacture are agree in the protocol , but are different in coding .

الفهرس

فهرس الأشكال والجداول :-

الفصل الأول

المقدمـــة Introduction

نجـد ان التحكم عن بعد ييسر من تشغيل منظم المروحــه في المنزل او المكتب كمـا يتميز ببساطـه التشغيـل , الحفـــاظ علي الطاقــه , نظــام موثوق به , طول العمر الافتراضي , إنعـدام الضجيج , بالاضافـه الي المزيد من الراحه الي الحياه اليوميه من خلال ازاله إزعاج الاضطرار للتنقل لتشغيل منظم المروحه.

نسبةً للخصائص والمميزات التي ذكرت يهدف البحث الي تصميم منظم مروحه فعال من حيث التكلفه والكفاءه العاليه .

نجد ان اول تحكم عن بعد يدعي lazy bones نظام كسول تم انتاجه من قبل هيئه zenith electronics في عام 1950 والتي انتقلت فيما بعد الي هيئه zenith radio.

أما اليوم اصبح التحكم عن بعد (التحكم بالرموت) للمعايير القياسيه للمنتجات الاستهلاكيه الالـكترونيــه لاجهـزة VCR , satellite boxes , digital video disk players , audio players .

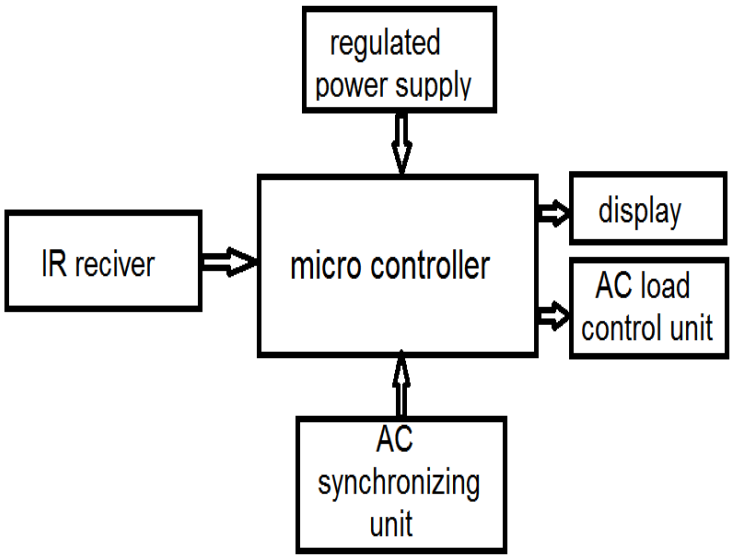
في عام 2000 أصبحت نسبه 99% من أجهزه TV , VCR , DVD ضمن تقنيه التحكم عن بعد .

في الاساس جهــاز التحكم عن بعد يعمل علي النحو التالي , يتم الضغط علي زر وهذا يكمل اتصــال معين والتي تعمل علي إنتاج شفرة هورس تبعـاً لـذلك الزر يعمـل الترانزستور علي تضخيم الاشاره وإرسـالها الي الثنـائي LED الـذي يترجم الاشاره الـي اشعة تحت الحمراء IR ، يعمـل المحساس الموجود علي الجهاز المراد التحكم فيه علي إستقبال أشعه IR ويتفاعـل معها بشكل مناسب .

وظيفه وحدة التحكم عن بعد هي أنتظـــار المستخدم حتي يتم الضغط علي أحـد المفـاتيح ثم ترجمـة ذلك الــي أشعـــه تحت الحمراء IR (إشاره ضوء) والتي يتم إستقبالهـــا أو أكتشافها من قبل الاداة , نجد أن التردد الحامـل لإشاره IR قـد يصـل الـي 36 kHz .

منهجيه التصميم التي سوف يتناولها البحث تعمل علي تقسيم التصميم الي كتل وظيفيه function block, حيث كل كتله وظيفيه تعتبر دائره لها وظيفتها الخاصه .

يحتوي النظام علي 6 كتل وظيفيه كما موضح في الشكل (1 - 1) :



شكل (1 - 1) مخطط المنظومه system block diagram

-:IR protocols (1 - 1)

يوجد عده برتكولات قياسيه لمنظومات الاتصال التي تستخدم IR من عدة مصنعين منها :

* RC – 5 و RECS - 80 لشركة فليبس Philips .

* NCS.

* Sony.

* Sharp .

يهتم البحث ببرتكول NEC وذلك نسباً لكثره أستخدامه كما أنه مستخدم من قبل عدة مصنعين مختلفين وكـذلك توفر بيانات البرتكول علي الشبكه العنكبوتيه .

-: NEC Protocol (1 - 2)

برتكول الناقل NEC IR يستخدم encoding a space impulse تشفير مسافه النبضه لبيانات الرسالة . حيث فترة النبضه هي :- 562.5 µs بتردد حامل 38 KHz (26.3 µs) .

ترسل البت المنطقية كلأتي :-

- الرقم " 0 " – 562.5 µs للنبضة يتبعها فراغ 562.5 µs

  بزمن ارسال كلي قدره 1.125 µs .

- الرقم " 1 " – 562.5 µs للنبضة يتبعها فراغ 1.687 µs

  بزمن إرسال كلي قدره 2.25 ms .

عند الضغط علي إحدي مفاتيح وحده التحكم عن بعد فإن محتوي الرسالة كما موضح أدناه علي الترتيب :-

- ❖ تتولد نبضه القيادة لمدة 9 ms .
- ❖ 8 بت للعنونة .
- ❖ 8 بت للعنونة بمنطق عكسي .
- ❖ 8 بت للتوجيهات .
- ❖ 8 بت للتوجيهات بمنطق عكسي .
- ❖ أخيراً نبضه لمدة 562.5 µs لتدل علي نهاية الرسالة .

الشكل (2 - 1) يوضح التكوين لبرتكول الناقل NEC IR لعونة ooh وتوجيه ADH (command) :

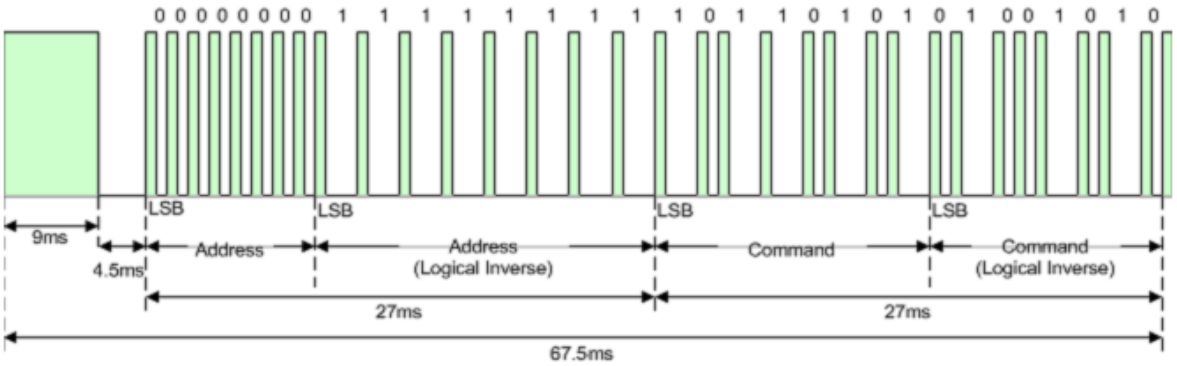


Figure 1. Example message frame using the NEC IR transmission protocol.

Fig (1 - 2) message frame

يلاحظ من الشكل (2 - 1) الأتي :-

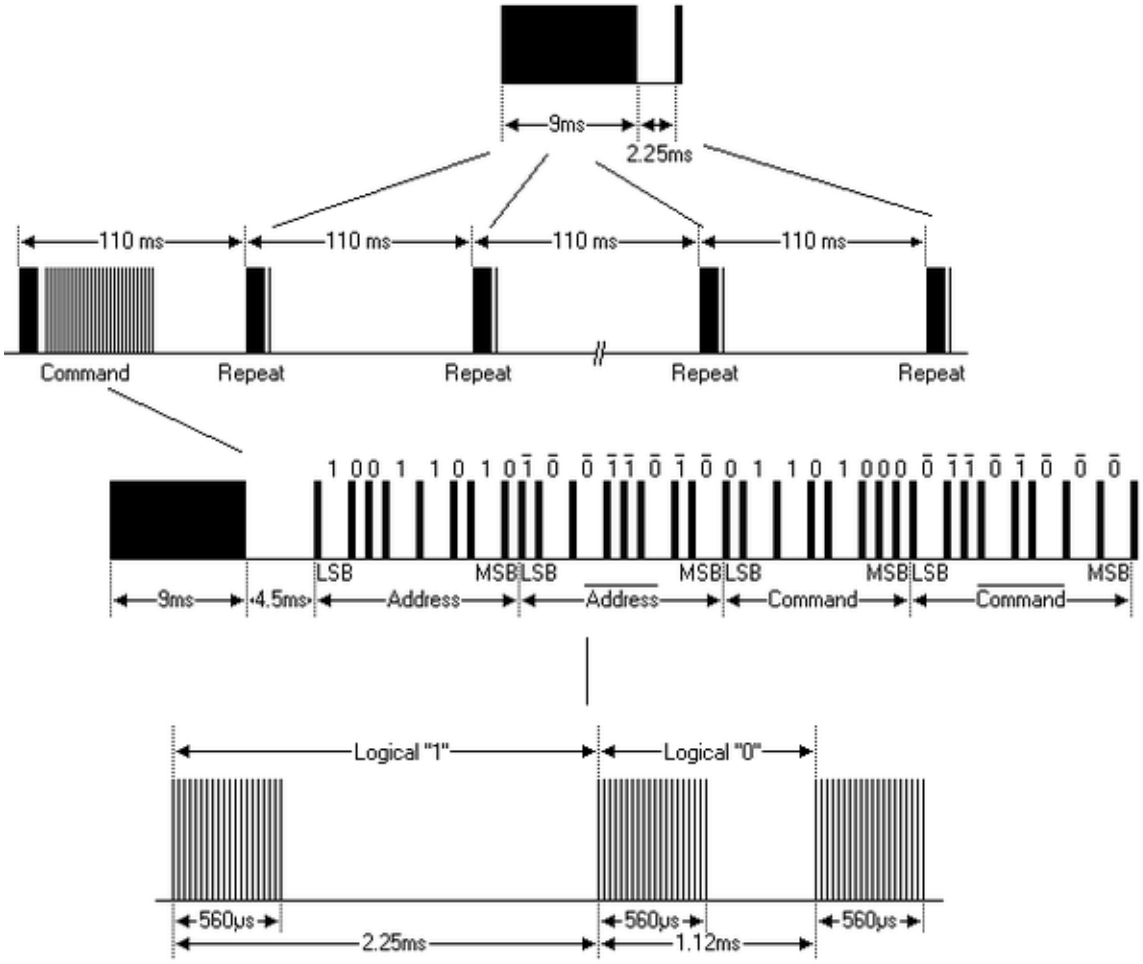
- ✓ يلاحظ أن فترة إرســـــال 16 bits للعنونه تحتــاج الي 27 ms (العنونه + المعكوس) وكذلك لإرسال 16 bits للتوجيهـــات (التوجيهـــات + المعكوس) . كلاهمـــا يحوي ثمانيه أصفــار وثمانيه للمنطق واحد $(8 \times 2.25\ ms) + (8 \times 1.25\ ms)$
- ✓ الزمـن الكـلي لإرســـال إطـــار الرســـالـة هو 67.5 ms .

(3 - 1) إعادة الشفرة Re-code :-

أذا تم الضغط علي مفتاح وحدة التحكم عن بعد والحفاظ عليه مضغوطاً هذا يجعل الوحـدة تعمل علي إعــادة إرســـال الشفرة مـره أخـري بعـد فترة زمنيه قدرهـــا 40 ms وبالتالي يتم أرسال الرسالة بدوره 108 ms مادام أستمرار الضغط علي المفتاح .

نجد أن شفرة الأعاده كالاتي :-

- 9 ms لنبضة القيادة .
- 2.25 ms فراغ .
- 562.5 µs للدلالة علي نهاية أرسال البيانات .



شكل (3 - 1) يوضح أرسال لشفرة تمت إعادتها مرتين

الفصل الثاني

تصميم الـدائره

(1 - 2) تصميم المنظم :-

يمكـن أستخـدام المنظم للتحكم في قيمة الجهد والحفاظ عليه ثابتاً عند قيمه محدده .

وحـــدة الـتحكـم Remote control تستخـــدم للتحكـم في نظـــم الـتشبـع لـبرتكـول NEC Format Remote عادة ماتتوفر مع أجهزة مشغلات DVD حيث يتم أستخدام ثلاث مفاتيح في وحدة التحكم لتوجيه الدائره بحيث :

- ❖ Up key يستخدم لزيادة السرعه .
- ❖ Down key يستخدم لتخفيض السرعه .
- ❖ Enter key للتشغيل ON أو OFF .

يستخـدم المنظـم للتحكم فـي سرعـة المروحـه بتدرج قـدرهـــا 10 way speed control (من 0 الي 9) ، حيث يتم عرض السرعة بعارض ذي سبعة وسمات 7 segment Display . كما يوجد دايود green LED في PCB وذلك لبيان حاله الحمل (on أو off) .

الأجزاء الرئيسية المستخدمة بدائرة المنظم :-

i. عارضه ذي سبعة وسمات 7 segment display لتوضيح مستوي السرعه الحاليه .

ii. محساس Tsop 1738 IR يستخدم لأستقبال التوجيهات في منظم المروحه .

iii. Green LED لبيان حالة التشغيل .

iv. الخرج out يتم توصيل الحمل علي التوالـي والذي يتمثل فـي 220 v .

v. الدخـــل IN – مصدر القـــدرة يطبق من خلال محول خافـض للجهـد 12 v – 0 – 12 v .

vi. Mcu – وهي Atmega8 AVR 8-bit microcontroller .

vii. مفتاح (Switch) للتحكم manual لتشغيل المروح بدون وحدة التحكم عن بعد Remote control .

Fig (2-1) Figure practical voltage Regulator

.

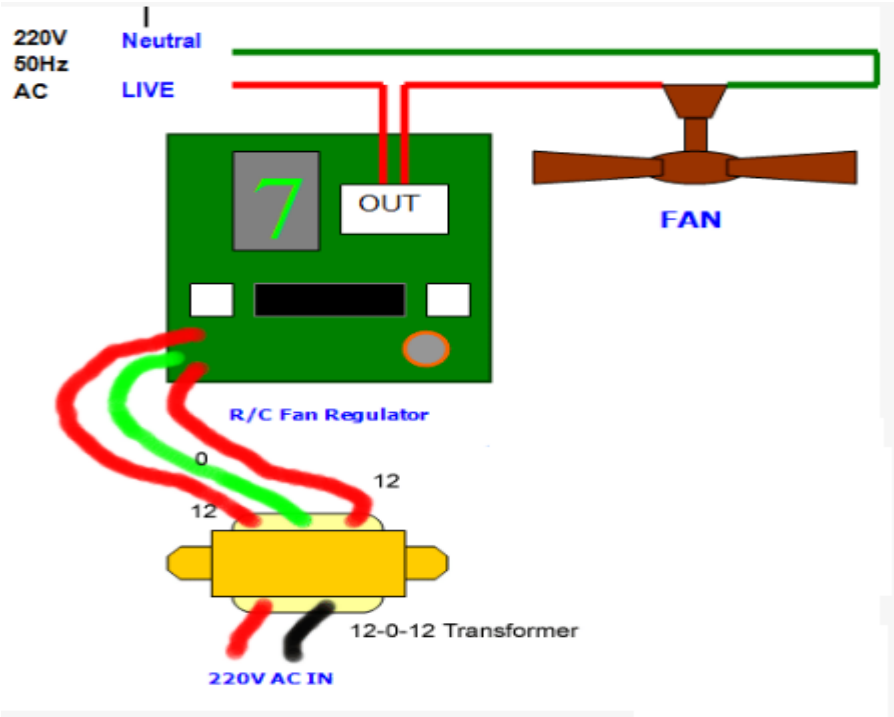


Fig (2-2) Circle process to the organizer fan

(2 - 2) **التركيب Structure:-**

Part list for Remote Controlled Fan Regulator.

Part List		
01	330 ohm resistor (8 Nos)	R2-R9
02	4k7 Resistor (2 Nos)	R1, ,R11
03	1K Resistor	R12
04	39 ohm Resistor	R13
05	1K5 Resistor	R15
06	22pF Ceramic Disk Capacitor (2 Nos)	C1,C2
07	0.1uF Ceramic Disk Capacitor (3 Nos)	C3,C5,C6
08	470uF 50v Electrolytic Capacitor	C4
09	16 MHz Crystal Half Size	X1
10	1N4007 Diode (6 Nos)	D2,D3,D4,D5,D6,D7
11	LED 5mm Any Colour	D1
12	MCT2E Opto Coupler	U4
13	MOC3021 Opto Triac Driver	U2
14	ATmega8-16PU General purpose 8 bit MCU	U1
15	Triac BT136	U3
16	7805 Voltage Regulator	U5
17	Common Anode Display	DISP11
18	TSOP1738 IR Sensor	X2
19	220V AC to 12-0-12 Centre Tapped Transformer (NOT Included in KIT)	
20	Hobby Remote Control (NEC)	

جدول (1 – 2) يوضح العناصر الألكترونيه المستخدمه

بدائرة المنظم

(2 - 3) دائرة القدرة power supply circuit:-

يستخدم محول خافض للجهد ذو نقطة تفرعيه 12 – 0 – 12 center-tap transformer ثم دائرة تقويم الجهد المتردد AC الي تيار مستمر DC عن طريق دائرة قنطرة ، ثم يستخدم منظم جهد Lm7805 لضمان ثبات الفولتية عند 5 v ، يجب علي مصدر القدرة أن يفي بمواصفات القدرة للمايكروكنترولة atmega8 حيث تعمل بمستوي فولتيه يتراوح من :-

- 2.7 – 5.5 لـ atmega8L .
- 4.5 – 5.5 لـ atmega8 .

(2 - 4) منظم الجهد :Voltage Regulators-

هو عبارة عن دائره متكامله تتكون من ثلاث أرجل ويعمل علي تثبيت الجهد عند 5 v وقيمة الجهد التي يعمل بها مابين (8 – 30)v والتيارات التي يمكن أن يتحملها المنظم مابين (0.5 – 0.75)A لكنه يولد درجة حراره عاليه لذلك يفضل أستخدام مشتتات الحراره ، تم تصميم منظمات الجهد الثابت مع حماية التحميل الزائد الحراري والتي تعمل علي إغلاق الدائرة عندما يتعرض الى حالة الجهد المفرط الزائد ، ويتم أستخدام المكثفات لضمان التشغيل المستمر في ظل جميع ظروف التحميل .

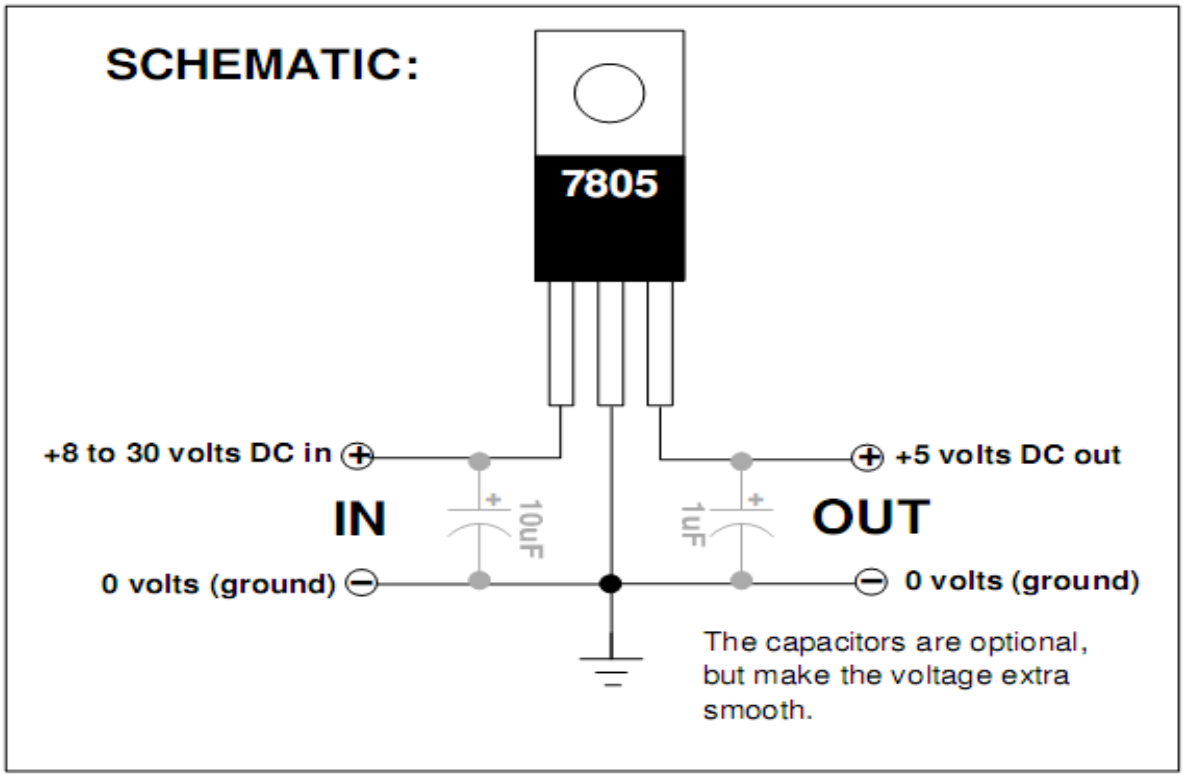


Fig (2 – 3) Figure illustrates the use of capacitors with voltage

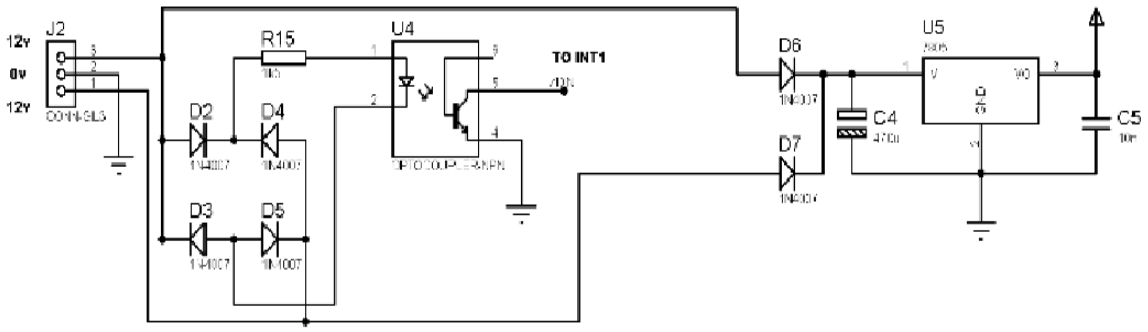


fig (2 – 4) power supply circuit

(2 - 5) دائرة عبور الفولتية الصفر zero crossing circuit:-

يستخدم القارن الضوئي moc 3021opto coupler بدائرة عبور الصفر حيث تعمل علي مقاطعة mcu في كل لحظه تعبر فيها الموجه الجيبيه AC الصفر حتي يتم تزامن نبضات البوابه gate pluse للترياك .

دائرة عبور الصفر هي الدائرة الكهربائية التي تبدأ مع عملية حمل التيار الكهربائي المتردد على مقربة من مرحلة الصفر ، هذا هو فيما يتعلق بمرحلات الحالة الصلبة ، مثل الترياك والمقومات ، والغرض من هذه الدائرة هو لبدء الترياك سيطرته في أقرب وقت ممكن ، هذا مفيد عندما يتم استخدام الترياك للسيطرة على المنافذ ، أو غيرها من الأحمال حيث هبوط التيار الكهربائي أو القطع الموجي يمكن أن يسبب آثار سيئة .

عند توصيل الترياك في أبسط أشكاله، فإنه يمكن قطع بداية منحنى التيار الكهربائي ، وذلك بسبب جهد بوابة الحد الأدنى في الترياك .

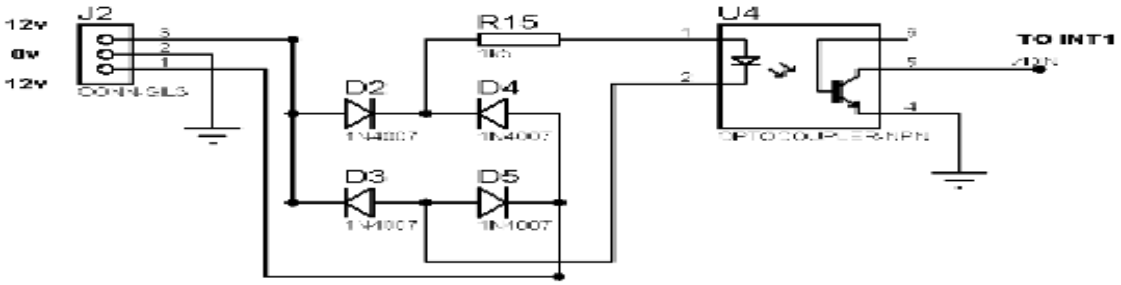


FIG (2 – 5) zero crossing circut

(2 - 6) 7 Segment Display :-

تستخدم العارضه DISP11 ذات السبعه وسمات لبيان الأرقام من 0 – 9 لبيان السرعه ، حيث

تستخدم Common Anode Display .

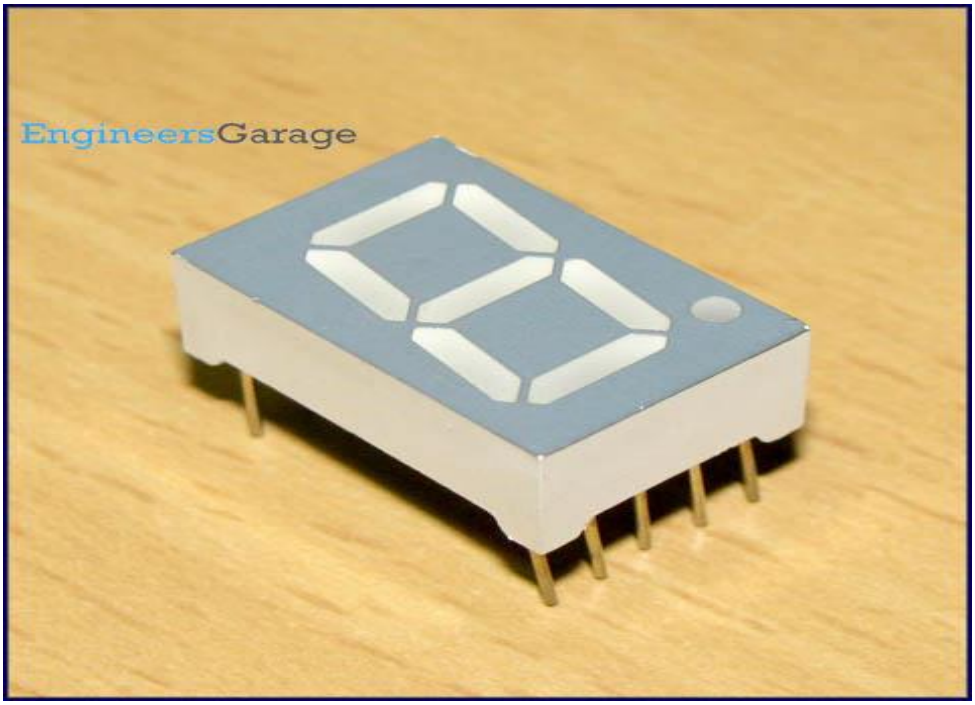


Fig (2 - 6) 7 Segment Display

(2 - 7) المحساس IR Sensor:-

يستخدم محساس TSOP1738 IR Sensor (X2) وذلك لأستقبال التوجيهات الأتيه من وحدة التحكم عن بعد Remote control .

وهو عبارة عن ترانزستور ضوئي يوجد علي شكل ترانزستور ولكن الهيكل الخارجي مختلف قليلاً ويوجد له ثلاث أرجل .

المميزات :

i. فلاتر داخليه لتردد PCM .
ii. تحسين التدريج ضد تردد المجال الكهربي .
iii. إنخفاض إستهلاك الطاقه .
iv. أرتفاع المناعه ضد الضوء المحيط .

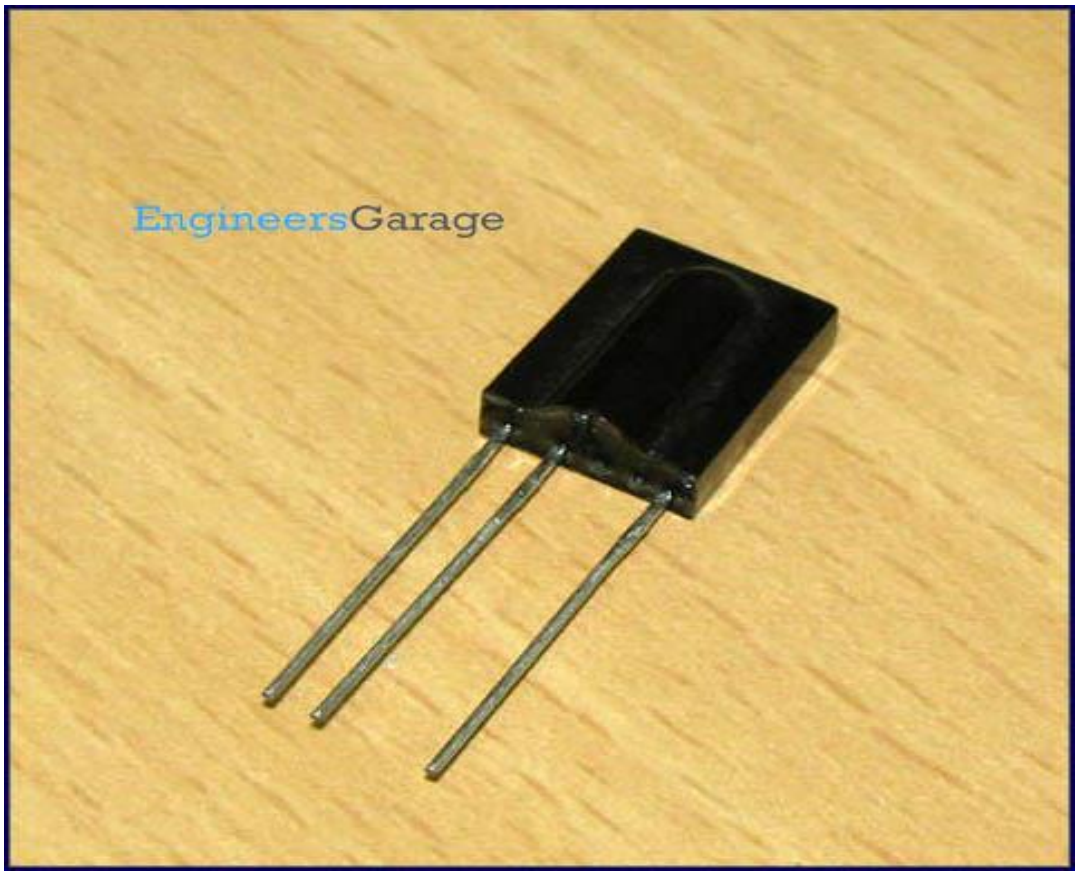


Fig (2 – 7) IR Sensor

يتكون المحساس من صمام ثنائي ومكبر ، عندما يستقبل المحساس الأشعه تحت الحمراء IR المرسله من جهاز التحكم عن بعد Remote control فإنه يعطي في الخرج جهداً قدره 5v بتردد 38 KHz ، أن الأضواء القادمة من ضوء الشمس أو مصابيح الفلورسنت ألخ قد تسبب أضطراباً له ويؤدي إلى إخراج غير مرغوب فيه حتى عندما يكون المصدر لم يبث الإشاره تحت الحمراء ، لذلك يستخدم مرشح ممرر الموجة إلى مرحلة التكامل والتحكم التلقائي لقمع هذه الاضطرابات .

وحدة Tsop 1738 لديها دائرة تحكم تحمل في ثناياها عوامل لتضخيم النبضات المرمزة التي يتم أستقبالها من جهاز إرسال الأشعة تحت الحمراء ، يتم إنشاء إشارة عندما يتلقى الثنائي الضوئي PIN الاشارات ثم يتم تلقي إشارة الدخل عن طريق التحكم التلقائي (AGC) لمجموعة من المدخلات وتعمل علي ضبط الكسب إلى مستوى مناسب يتم تمرير الإشارة من دائرة التحكم AGC إلى أمرار نطاق محدد Band Bass Filter لتصفية الترددات غير المرغوب فيها و بعد ذلك تمر الأشاره الي وحدة فك التعديل demodulated لفك الأشاره المشفره ويكون خرج دائرة فك التعديل ضعيف لذلك يكبر الخرج الناتج بواسطة ترانزستور NPN ، يتم الحصول على الخرج النهائي من طرف

الجامع collector للترانزستور كما موضح بالشكل التالي :-

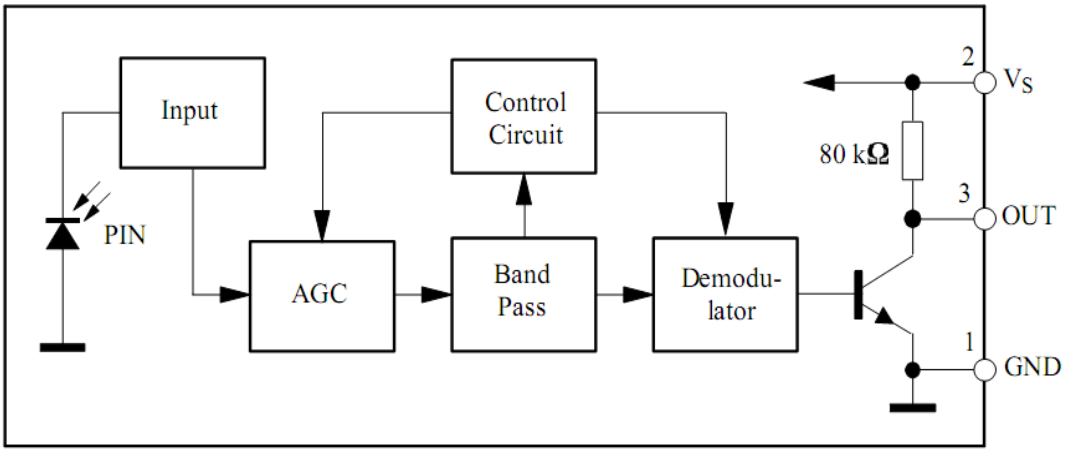


Fig (2 – 8) block diagram

-: Atmaga8 MCU (2 - 8)

تم أختراع المايكروكنترولة من قبل المهندس Marcian Hoff وهو أحد مهندسي شركة أنتل وبعده بدأت تتنافس الشركات في تتطوير هذا الأختراع وأدخاله في أجهزتها حتي أصبح كما نراها اليوم .

المايكروكنترولة هي عباره عن جهاز كمبيوتر مدمج في قطعه صقيره جداً ولها أنواع عديده حسب الأستخدام ، وتدخل تقريباً في تركيب جميع الأجهزه الحديثه .

تحتوي علي وحدة معالج صغري وبداخله ذاكره داخليه قابله للبرمجه ، كما أنها تحتوي علي بوابات أدخال وأخراج البيانات والأوامر .

تحتوي المايكروكنترولة علي ثلاث منافزهي :-

a. Port b .

b. Port c .

c. Port d .

وهي تحتوي علي 28 بت أي مدخل (28 Pin) .

أسهمت بدرجه كبيره في تصغير حجم الأجهزه مما أدي الي سهولـة نقلهـا وأستخدامهـا وكـذلك أسهمت المــايكروكنترولـه في أنخفــاض ثمن الأجهـزه وجعلهـا في متنـاول الكثيـر مـن الأشخاص .

مكونات الــ Atmaga8 :-

1. وحدة معالج صغري بداخلها ذاكره قابله للبرمجه .
2. المسجلات REGISTER .
3. ذاكره عشوائيه Ram .
4. مذبذبات OSCILLATOR .
5. العدادات COUNTER .

كما موضح بالشكل التالي :-

Figure 1. Block Diagram

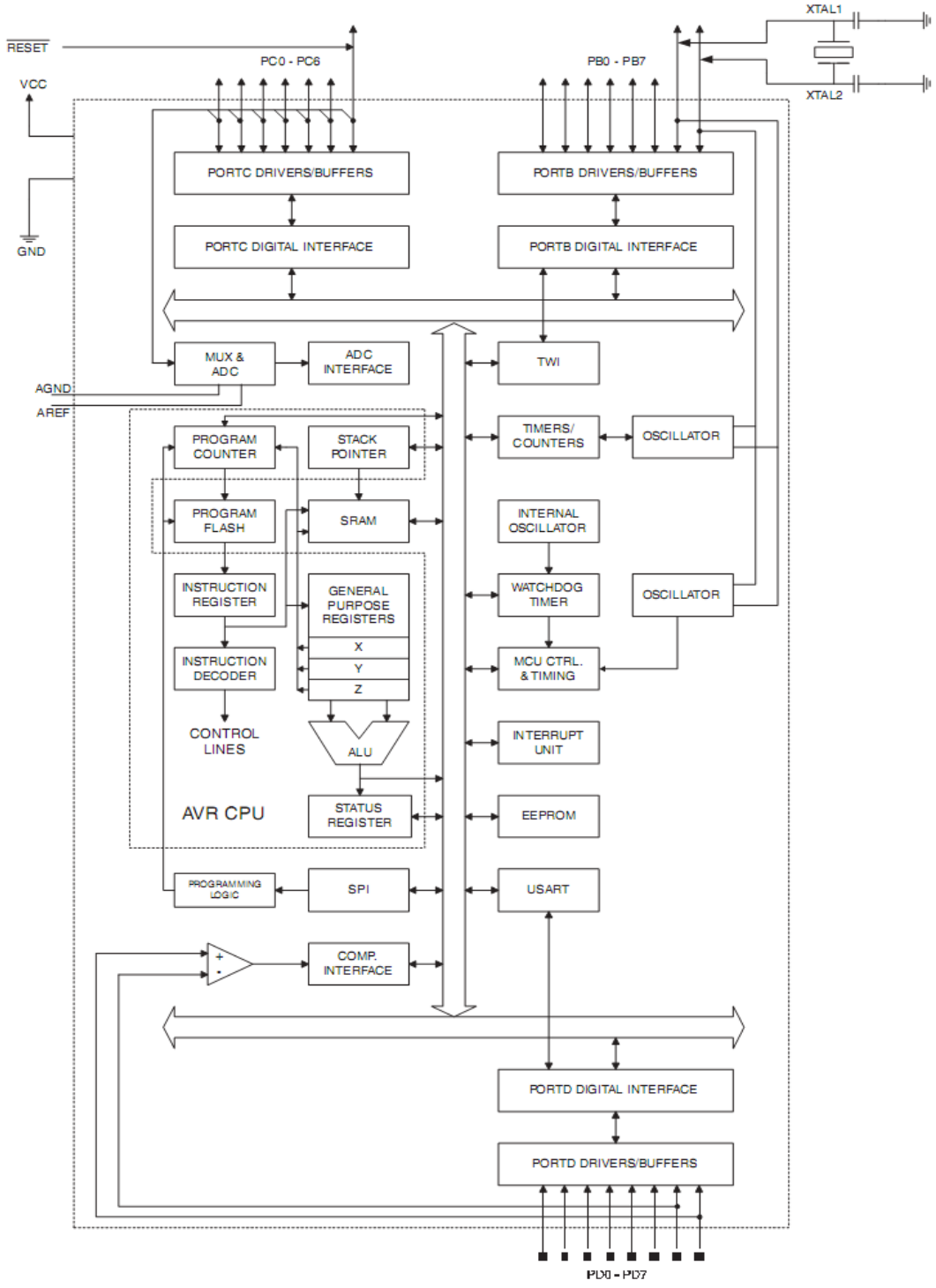


Fig (2 - 9) MCU Block Diagram

إستخدامات المايكرو كنتروله :-

a. تستخدم في جهاز الأنزار ضد السرقه .

b. تستخدم في إشارات المرور .

c. تستخدم في دوائر الأثاره الأوتماتيكيه .

d. تستخدم في الغسالات الزكيه وأفران المايكروويف .

e. تستخدم في الروبوتات وأجهزه القياس المختلفه .

المميزات :-

- عالية الاداء ومنخفضة الطاقه .
- قفل برمجة برامج الأمن .

مميزات طرفيه :-

- ✓ ثلاث قنوات PWM .
- ✓ 8 قناة ADC في حزمة TQFP وQFN / MLF .

(9 - 2) دائرة الخرجOutput circuit:-

يستخدم خرج التعديل النبضي PWM في المايكروكنتروله وذلك للتحكم ببواب التريـاك حتي يتم التحكم بجهد الحمل .

تسلط نبضات PWM الي القـــارن الضوئي ذو التريـاك moc 3021 opto triac coupler { V2 } بمستـوي فولتيـه 5v ، بينمـــا يوصـــل مستوي فولتيـة 220v AC للتريـاك الموجـود بالقـارن وذلك للتحكم في مستوي فولتيـه التريـاك BT136 بواسطـة نبضـات ذات مستوي فولتيه 220v AC تسلط علي بوابته .

كمـــا يستخدم مرشح ترددات RC Filter يوصـل علي التـوازي مع التريـاك BT136 وذلك لتجنب الحـالات العـابره (حالات فتح التريـاك) نتيجه لطبيعه الحمـــل الحثيه وكذلك لمنع أنتشـار التشوهات نتيجه لعملية الفتح بالتريـاك والتي قد تسبب في تداخلات راديويه .

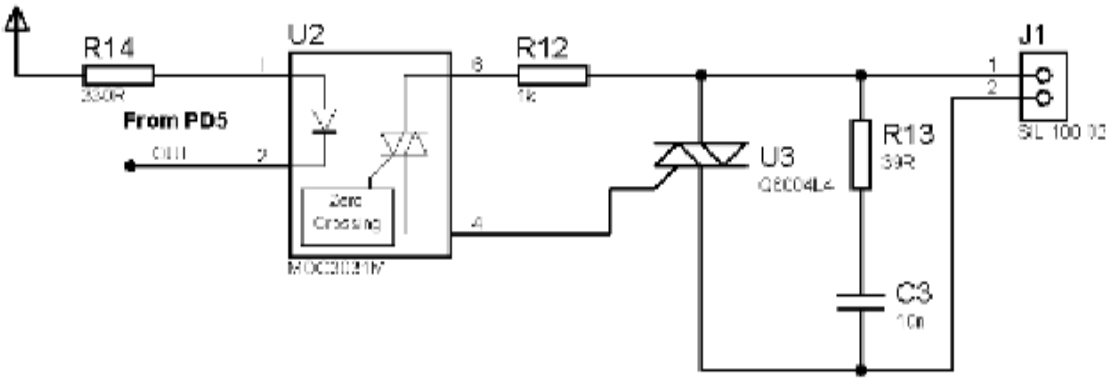


Fig (2 – 10) Output circuit

الفصل الثالث

الـبرمجيـــــات

يستخدم برنامج AVR gcc لكتابة وتنفيز البرنامج ، حيث يكتب البرنامج بلغة C .

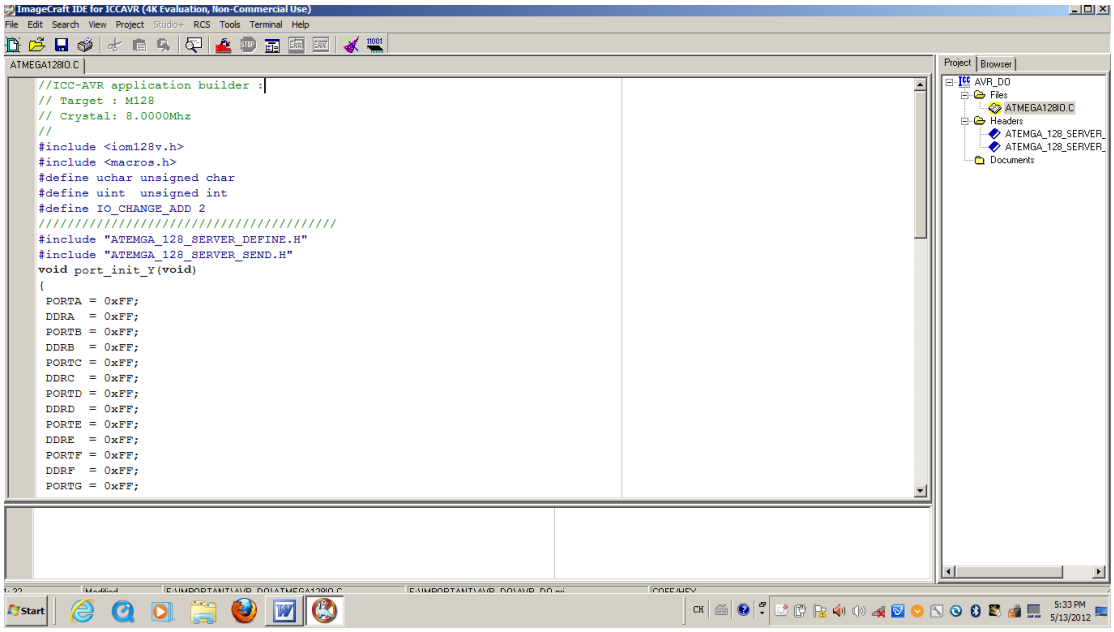


Fig (3 – 1) AVR- gcc software

يتكون البرنامج من الملفات الأتيه :-

Fan.c , rckeys.h , remote.c , remote.h

Fan.c يحوي البرنامج الرئيسي حيث يتم أستدعاء كل الملفات الفرعيه rckeys.h , remote.c من خلاله وكذلك يتم تعريف بعض المتغيرات لنقاط الأدخال والأخراج مثل :-

#define Fan-Power-LED-PortD

ياخذ Portd أسم أخر وهو :-

Fan-Power-LED-Port

#define Fan-Power-LED-Bit 7

يأخذ الرقم 7 الأسم الجديد :-

Fan-Power-LED-Bit 7

كذلك يتم تعريف بعض الدوال مثل :-

#define Power-LED-ON ()

Fan-Power-LED-Port

& = (~(1<<Fan-Porb-LED-BTT))

عند أستدعاء هذه الدالة فتعمل علي نفي المتغير بمعني اذا كان يأخذ قيمة 1 فيصبح 0 والعكس .

كذلك تعريف المصفوفه بالقيم :-

Unit 8 – T table[10]={141,125,109,94,78,62,47,31,16,1}

هذه القيم تستخدم للتحكم بالسرعه أي التحكم في التعديل PWM .

كذلك تعريف المتغير speed بأنه حقيقي بدون أشارة عرضها 8bit يأخذ القيمه صفر :-

Uint 8 – T speed = 0 ;

(3 - 1) الداله الرئيسه main( ) :-

- تعمل علي تهيئة المايكروكنتروله بإستدعاء الداله الفرعيه ()Initialize ثم إدخال البرنامج في مسار مغلق while(1) :-

داخل هذا المسار تعمل علي جلب توجيهات الريموت وتضعها بالمتغير CMD ثم مقارنة التوجيهات فإذا كان يتبع المفتاح RC-UP والذي يأخذ الرقم أو الشفره تعمل ، تعمل علي زيادة السرعه ++speed أما إذا كان يتبع للمفتاح RC-Down فتعمل علي تخفيض السرعه .

أما إذا كان مطابق RC-Enter فهذا يتوقف علي حالة المتغير fan-on فإذا كان يحمل القيمه 1 فتصبح Zero والعكس .

ثم أخيراً تعمل علي عرض قيمة السرعه علي العارضه بإستدعاء الداله Display(speed**);**

(3 - 2) الداله الفرعيه void Initialize:-

- تعمل علي تشغيل Power LED عند النقطه 8 من المنفز port D من خلال الأمر :
  fan-power-LED-DDR|=0B1000000

- ثم تعمل علي إطفاء power LED من خلال داله power-LED-off

- ثم عرض قيمة zero علي العارضه .
- ثم تهيئة الريموت بالداله RemoteInit() الموجوده بالملف remote.c .
- كذلك تعمل علي تهيئة المقاطعه int1 الخاصه بدائرة عبور الصفر .
- ضبط الخرج لل PWM الذي يتحكم بالترياك بالمنفز port D بالبت رقم PD5 بالتوجيه

```
DDRD|= (1<<PD5);
```

وبنفس الوقت تعمل علي إطفائه بالتوجيه

```
PortD|= (1<<PD5);
```

- ضبط المؤقت Timer2 الذي يعمل علي توليد نبضات PWM ووضعه في حالة المقارنه بالقيم التي تم تعريفها بالمصفوفه Table .

(3 - 3) دالة المقاطعه ISR (INT1-Vect):-

تستخدم نظام المقاطعه لدي المايكروكنتروله لتحسين عبور موجه AC الجيبيه منطقة الصفر (دائرة عبور zero) ، تعمل فقط في حالة أن fan-on ، اذا تحقق هذا الشرط وكانت السرعه في أقصي قيمه لها وهي 9 فإن الخرج لدائرة الترياك يكون LOW بمعني أن يتم قدح الترياك كلياً .

في حالة fan-off فإن الخرج high بمعني لا يقدح الترياك وبالتالي حاله off ، أما أذا كانت السرعه تأخذ قيمه أخري fan-on فيتم تحميل قيم المصفوفه المعرفه بـــ table وتوضح بسجل المقارنه OCR2 للمؤقت Timer2 وذلك تبعاً لقيم السرعه المطلوبه من خلال التوجيه

```
OCR2 = Table [speed];
```

(3 - 4) دالة مقاطعه المؤقت في حالة المقارنه ((ISR(Timer2-(cmd-Ved :-

في حالة تزامن قيمة المؤقت الحاليه مع القيمه المضبوطه في الجدول ، تعمل هذه الدائره علي أن يكون الترياك ON وإيقاف المؤقت .

(3 - 5) دالة التأخير الزمني (Void wait) :-

عند استدعاء هذه الداله فإنها تعمل علي تاخير زمن النبضه باستخدام الاوامر:

```
Char i ;

For (i=0; i<100; i++)

- delay – loop – 2 (0);
```

(3 - 6) دالة العرض display (unit 8-t num) :-

تعمل علي إرسال البيانات الرقميه لوحدة العرض (حيث يتم توصيل وحدة العرض مع المايكروكنتروله بثمانيه خطوط ، أي أنتقال البيانات توازي) أو لتعمل علي فك الشفره decoder حتي يتم عرض الرقم (السرعه) بشكل صحيح علي العارضه ذات السبعه وسمات ، حيث تستخدم خوارزميه

```
Switch, case
```

فمثلاً اذا أريدنا عرض الرقم 9 علي العارض فتكون الداله كالأتي :

```
Switch(num)

Case 9:

\\ ×× fedcba

portC = 0B00010000;

portb&=(~(0B00000010));

Break;
```

علمً بأن العارضه توصل مع المنفذ portC لدي المايكروكنتروله .

(3 - 7) الملف Remote.c :-

لـقد تم تنزيـل الملف من الشبكـه العنكبوتيـه حيث يعمـــل كمكتبه Libaray للبرنامج الرئيسي ويتم تنفيز بعض دوالـــه عنـد الضروره من الملف الـرئيسي .

يعمـل هذا الملف علي فـك شفرات إشـــاره IR القـادمـه من الرموت وذلك بالمؤقت الزمني لطـول فتره النبضه وفقاً لبرتكول NEC ثم ترجمتها الي رقم حتي يتم إستخدامـه من قبل البرناج الرئيسي كما يلاحظ أن هذا الملف يتميز بالتعقيد ودقـة الزمـن ، لذلك تم تنزيله في الأنترنت نسبـة لتوفره في الشبكه وحفاظـاً علي عدم هدر الزمن .

(3 - 8) الملف Remot.h :-

وهو ملف heeder يتبع الملف Remote.c ويحوي علي تعـــاريف الدوال المستخدمه في الملف Remote.c .

(3 - 9) الملف rckeys.h :-

في هـذا الملف يتم تعريف الشفرات (في شكل أرقـــام) لمفـــاتيح الـرموت ، يلاحظ أنه توجد عـدة ريمـوت مـن مصنعين مختلفين تستخـدم نفس الـبرتكولNEC ولكنها تختلف في شفرات المفاتيح ، في هذا البحث تم إستخدام شفرات Car mp3 remote .

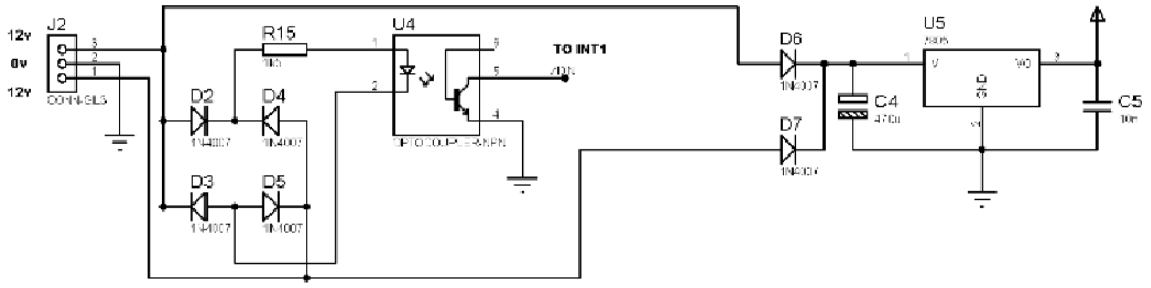
الفصل الرابع

منـاقشــة الدائــرة

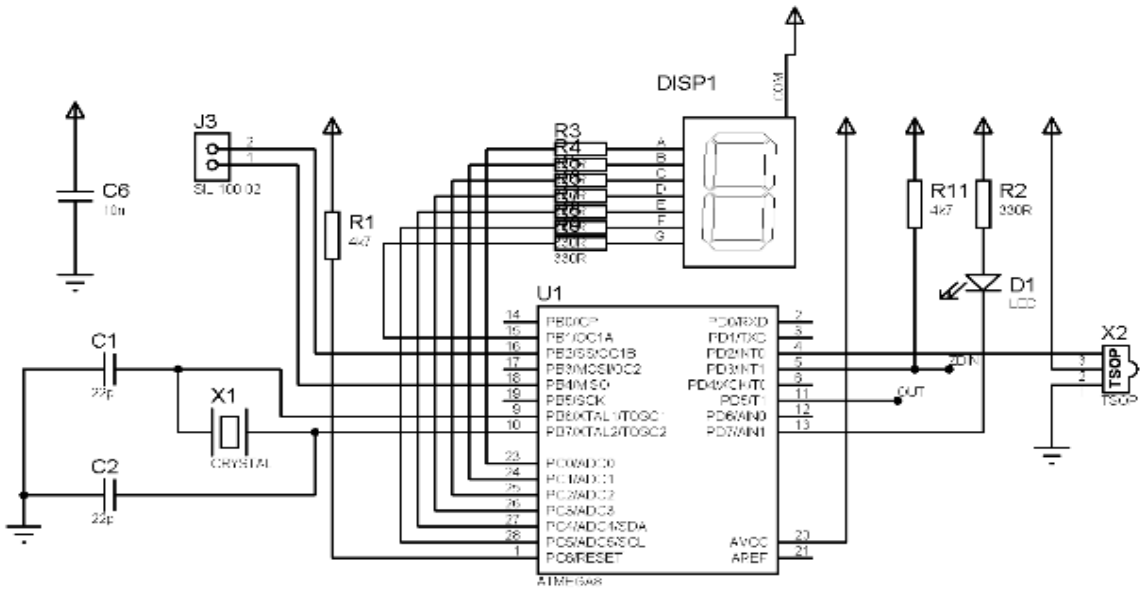
(4 - 1) مناقشه الدائره عملياً :-

تم تقسيم الدائره الي ثلاث أجزاء رئيسه لتسهيل عملية التركيب والأختبار وهي :-

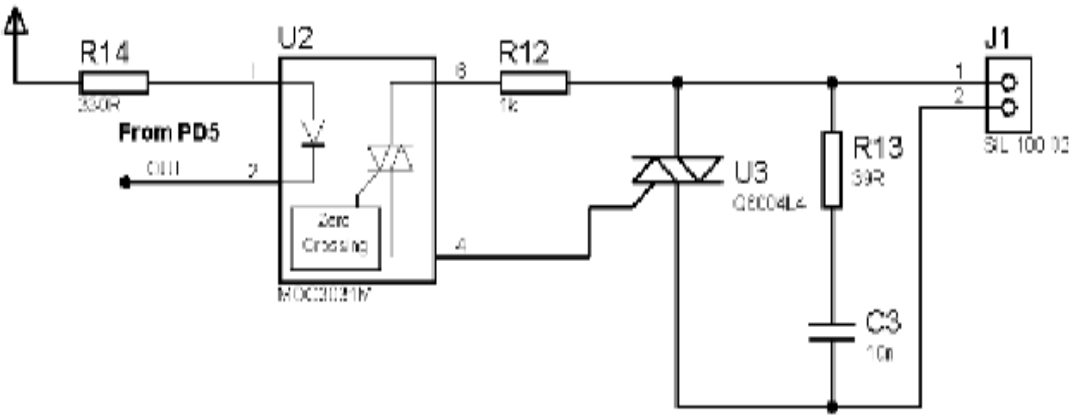
(4 - 1 - 1) دائرة القدرة power supply .



(4 - 1 - 2) دائرة التحكم control circuit .



(3 - 1 - 4) دائرة الخرج OutPut Circuit .



(4 - 2) **مناقشة النتائج :-**

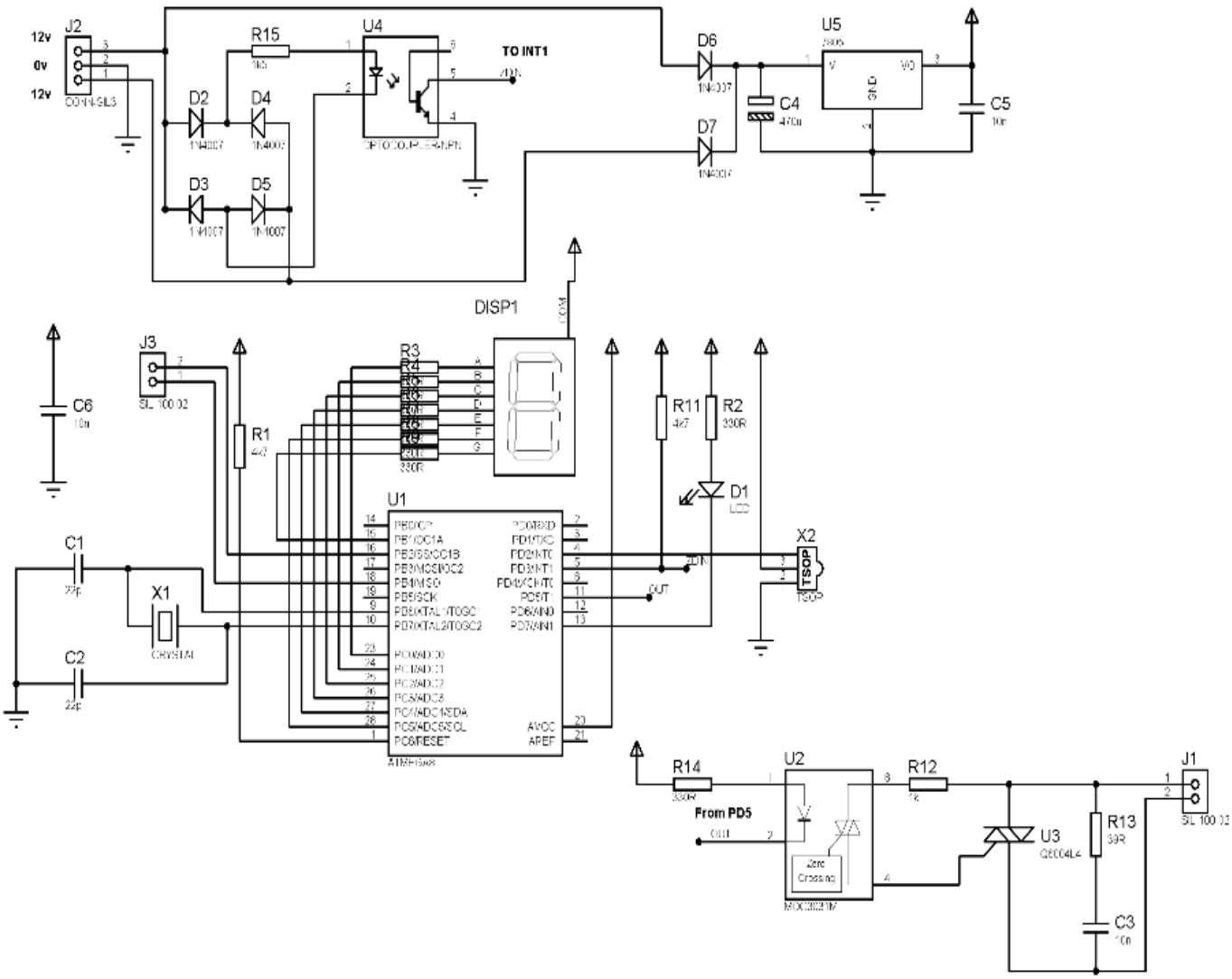
تم التاكد من سلامة الدائره وعدم وجود قصر Short أو فتح Open في الدائر ثم تم أختبار الدائره عملياً وصُلت الجهد عليها ويتم التحكم في جهد الخرج بواسطة الريموت كنترول وقياس الخرج بواسطة جهاز الأفوميتر وسُجلت النتائج في الجدول التالي :-

Scale	Volts
OF	**7**
ON	**17**
1	**39**
2	**69**
3	**104**
4	**141**
5	**177**
6	**204**
7	**225**
8	**234**
9	**234**

جدول (1 - 4) يوضح الخرج مع تدرج السرعه

الملحق

(4.4) الدائره :-


12v
0v
12v
J2
CONN-SIL3
R15
U4
TO INT1
D2
D4
D3
D5
1N4007
OPTOCOUPLER-NPN
D6
D7
U5
C4
C5
DISP1
J3
SIL 100-02
C6
R1
R3
R11
R2
D1
U1
X2
TSOP
C1
X1
CRYSTAL
C2
OUT
R14
U2
R12
J1
From PD5
Zero Crossing
U3
R13
C3

البرنامج الرئيسي Fan.c :-

```
/*

Remote Controlled Fan Regulator.
********************************

Compiler: avr-gcc (WinAVR-20090313)
Project Manager/IDE: AVR Studio 4.17 Build 666
Other Lib: eXtreme NEC Decoder.

Hardware:
        INT0 - IR Receiver
        INT1 - Zero Crossing Detector.
        PD5 - Triac Control.
        PB2 - Manual Switch

For
        ATmega8 @ 16MHz

Fuse:
        HIGH=0xC9 LOW=0xFF



*/


#include <avr/io.h>
#include <util/delay_basic.h>
#include <avr/interrupt.h>

#include "remote.h"
#include "rckeys.h"

#define FAN_POWER_LED_PORT PORTD
#define FAN_POWER_LED_DDR DDRD
#define FAN_POWER_LED_BIT 7
```

```
#define POWER_LED_ON()
	FAN_POWER_LED_PORT&=(~(1<<FAN_POWER_LED_BIT))
#define POWER_LED_OFF()
	FAN_POWER_LED_PORT|=(1<<FAN_POWER_LED_BIT)

uint8_t table[10]={141,125,109,94,78,62,47,31,16,1};

uint8_t speed=0;
uint8_t fan_on=0;

void Initialize()
{
	FAN_POWER_LED_DDR|=0B10000000;

	POWER_LED_OFF();

	DDRC|=0b00111111;	//Seven segment
	DDRB|=0b00000010;	//Middle segment G

	Display(0);

	RemoteInit();

	//Initialize the zero crossing detector INT(1)

	MCUCR|=((1<<ISC11)|(1<<ISC10));		//INT in Rising edge
	GICR|=(1<<INT1);					//Enable INT1

	//Output
	DDRD|=(1<<PD5);
	PORTD|=(1<<PD5);	//High = TRIAC Off


	//Set Timer 2
	TCCR2|=(1<<WGM21); //CTC
	TIMSK|=(1<<OCIE2);	//Enable OCI2

	sei();

}
```

```
/*

Zero Crossing Detect.

*/
ISR(INT1_vect)
{
	if(!fan_on)
	{
		PORTD|=(1<<PD5);	//High = TRIAC Off
		return;
	}

	if(speed==9)
	{
		PORTD&=(~(1<<PD5)); //low = TRIAC ON
		return;
	}

	PORTD|=(1<<PD5);	//High = TRIAC Off

	OCR2=table[speed];

	TCNT2=0x00;

	TCCR2|=((1<<CS22)|(1<<CS21)|(1<<CS20));	//Start Timer
prescaler =1024

}

/*

Timer2 Compare ISR

*/
ISR(TIMER2_COMP_vect)
{
	PORTD&=(~(1<<PD5)); //low = TRIAC ON

	TCCR2&=(~((1<<CS22)|(1<<CS21)|(1<<CS20)));//Stop Timer
```

```
}

/*

Simple Wait Function

*/
void Wait()
{
	char i;

	for(i=0;i<100;i++)
		_delay_loop_2(0);
}

/*

Displays a number in Seven Seg Display

*/
void Display(uint8_t num)
{
	if(num>9)
		return;

	switch (num)
	{
		case 0:
			PORTC=0B00000000;
			PORTB|=0B00000010;
			break;
		case 1:
		       //	xxfedcba
			PORTC=0B00111001;
			PORTB|=0B00000010;
			break;
		case 2:
			//	xxfedcba
			PORTC=0B00100100;
			PORTB&=(~(0B00000010));
```

```
                break;
            case 3:
                //          xxfedcba
                PORTC=0B00110000;
                PORTB&=(~(0B00000010));
                break;
                break;
            case 4:
                //          xxfedcba
                PORTC=0B00011001;
                PORTB&=(~(0B00000010));
                break;
            case 5:
                //          xxfedcba
                PORTC=0B00010010;
                PORTB&=(~(0B00000010));
                break;
            case 6:
                //          xxfedcba
                PORTC=0B00000010;
                PORTB&=(~(0B00000010));
                break;
            case 7:
                //          xxfedcba
                PORTC=0B00111000;
                PORTB|=0B00000010;
                break;
            case 8:
                //          xxfedcba
                PORTC=0B00000000;
                PORTB&=(~(0B00000010));
                break;
            case 9:
                //          xxfedcba
                PORTC=0B00010000;
                PORTB&=(~(0B00000010));
                break;
        }
}
```

```
void main()
{

	uint8_t cmd; //Command received from remote

	Initialize();

	while(1)
	{
		//Get Command For the Remote Control
		cmd=GetRemoteCmd(1);

		//Now process the command

		//UP Key
		if(cmd==RC_UP)
		{
			if(speed<9)
				speed++;

		}

		//DOWN Key
		if(cmd==RC_DOWN)
		{
			if(speed>0)
				speed--;
		}

		//Enter Key
		if(cmd==RC_ENTER)
		{
			if(fan_on)
			{
				POWER_LED_OFF();
				fan_on=0;    //Turn Off
			}
			else
			{
				POWER_LED_ON();
```

```
                    fan_on=1;    //Turn On
                }
            }

            Display(speed);

        }
}
```

(5 - 2) البرنامج الفرعي remote.c :-

```
/*

IR remote interfacing library for AVR series of microcontrollers.
Target MCU ATmege8 @ 8MHz/12MHz/16MHz

Sensor: TSOP1738 IR receiver module must be connected to INT0 Pin.
        This is PIN4 in ATmega8

          -----
         |  _  |
         | | | |
         | | | |
         -------
         | |  |
         | |  |
         | |  |
   (GND) | (To INT0)
     (5v)

         **********************
         *TSOP 1738 Front View*
         **********************


Copyright Avinash Gupta 2008
extremeelectronics.co.in



Resource Usage:
-Timer1
-INT0 (PD2)


Copyright Avinash Gupta 2008-2010
extremeelectronics.co.in

MUST BE DISTRIBUTED WITH THIS COPYRIGHT INTACT
```

```
*/

#include <avr/interrupt.h>
#include <util/delay.h>

#include "remote.h"

//************************************************************
//                    Environment Checkup
//                    -------------------
//Test if user has correct clock setting.
//************************************************************

#ifndef F_CPU
#error "IR Remote Lib : F_CPU not defined"
#endif

#if ((F_CPU !=8000000) && (F_CPU !=12000000) && (F_CPU
!=16000000))
#error "IR Remote Lib : Unsupported CPU frequency"
#error "IR remote Lib : Pls use F_CPU = 8MHz,12Mhz or 16MHz"
#endif

//Now check the CPU
#ifndef __AVR_ATmega8__
#error "IR Remote Lib : This version is only for ATmega8"
#error "IR Remote Lib : Pls see www.eXtremeElectronics.co.in for other
CPU models"
#endif

#if F_CPU==8000000
        #define TIMER_COMP_VAL 80
        #elif F_CPU==12000000
                #define TIMER_COMP_VAL 120
                        #elif F_CPU==16000000
                                #define TIMER_COMP_VAL 160
                                        #endif
```

```
//Globals
volatile unsigned int Time;             //Main timer, stores time in 10us,
                                                    //Updated by
ISR(TIMER0_COMP)
volatile unsigned char BitNo;           //Pos of next BIT
volatile unsigned char ByteNo;          //Pos of current Byte

volatile unsigned char IrData[4];       //The four data Bytes of Ir Packet
                                                    //2-Byte Address
2-Byte Data
volatile unsigned char IrCmdQ[QMAX];//Final Command Received
(Buffer)

volatile unsigned char PrevCmd;                 //Used for repeat

//Variables used for start repeating only after a key is pressed for certain
time

volatile unsigned char Repeat;          //1=yes 0=no
volatile unsigned char RCount;          //Repeat count

volatile char QFront=-1,QEnd=-1;

volatile unsigned char State;           //State of receiver

volatile unsigned char Edge;            //Edge of interrupt [ RISING=1 OR
FALLING=0 ]

volatile unsigned int stop;
```

```
/*****************************************************************
*********************************/
/*                          F U N C T I O N S    S T A R T S
                                                                  */
/*****************************************************************
*********************************/

void RemoteInit()
{


        char i;
        for(i=0;i<4;i++) IrData[i]=0;

        stop=0;
        State=IR_VALIDATE_LEAD_HIGH;
        Edge=0;
        Repeat=0;

        //Setup Timer1
        //------------
        TCCR1B|=((1<<CS10)|(1<<WGM12));       //Prescaler : Fcpu
Mode : CTC

        TIMSK|=(1<<OCIE1A); //Enable Output Compare Interrupt

        OCR1A=TIMER_COMP_VAL;                       //Set Compare
Value


        //Set Up INT1
        //------------
        MCUCR|=(1<<ISC01);   //INT ON Falling Edge

        GICR|=(1<<INT0);        //Enable INT1

        //Enable Interrupts
        //------------------

        SREG|=(1<<7);
```

```
}

ISR(TIMER1_COMPA_vect)
{
	Time++;
}
ISR(INT0_vect)
{
	GICR&=(~(1<<INT0));  //Disable INT0
	sei();

	if(stop) return;
	unsigned int TempTime=Time;
	Time=0;
	TCNT0=0;
	switch(State)
	{
	case IR_VALIDATE_LEAD_HIGH:
	{
			if(Edge)
			{
				//Rising
				if((TempTime>(900-(900*TOL))) &&
(TempTime<(900+(900*TOL))))
				{

					//Lead High Correct
					State=IR_VALIDATE_LEAD_LOW;
					//INT ON FALLING EDGE
					MCUCR&=(~((1<<ISC01)|(1<<ISC00)));
					MCUCR|=(1<<ISC01);
					Edge=0;
				}
				else
				{
				/*
				stop=1;
				LCD_Clear();
				LCD_Write_String("Err in LeadHigh");
				LCD_GotoXY(0,1);
```

```
                        LCD_Write_Int(TempTime,-1);
                        */

                        ResetIR();

                        }
                }
                else
                {
                        //Falling
                        MCUCR|=((1<<ISC01)|(1<<ISC00)); //Set INT on
Rising Edge
                        Edge=1;
                }
                break;
        }
        case IR_VALIDATE_LEAD_LOW:
        {
                if((TempTime>(450-(450*TOL))) &&
(TempTime<(450+(450*TOL))))
                {
                //Got a valid leader
                State=IR_RECEIVE_BITS;
                BitNo=0;
                ByteNo=0;
                MCUCR|=((1<<ISC01)|(1<<ISC00)); //Set INT on Rising Edge
                Edge=1;


                }
                else if((TempTime>200) && (TempTime<245))
                {
                        if(Repeat)
                        {
                        //Got a repeat pulse
                        if((QEnd==(QMAX-1) &&
QFront==0)||((QEnd+1)==QFront))
                                        {
                                                QFront++;
                                                if(QFront==(QMAX))
```

```
                                        QFront=0;
                                }

                                if(QEnd==(QMAX-1))
                                        QEnd=0;
                                else
                                        QEnd++;

                                IrCmdQ[QEnd]=PrevCmd;

                                if(QFront==-1) QFront=0;
                        }
                        else
                        {
                                RCount++;
                                if(RCount==4) Repeat=1;
                        }

                                ResetIR();
                }else
                {/*
                stop=1;
                                LCD_Clear();
                                LCD_Write_String("Err in LeadLow");
                                LCD_GotoXY(0,1);
                                LCD_Write_Int(TempTime,-1);
                 */
                ResetIR();

                }
break;
}
case IR_RECEIVE_BITS:
{
        if(Edge)
        {
                //Rising
                if((TempTime>50) && (TempTime<69))
                {
                        //Correct Beg of BIT found
```

```
                        //INT ON FALLING EDGE
                                MCUCR&=(~((1<<ISC01)|(1<<ISC00)));
                                MCUCR|=(1<<ISC01);
                                Edge=0;
                }
                else
                {
                /*      stop=1;
                        LCD_Clear();
                        LCD_Write_String("Err in Bit Beg");
                        LCD_Write_StringXY(1,0,"Bit");
                        LCD_Write_Int(BitNo,-1);
                        LCD_Write_String("Byt");
                        LCD_Write_Int(ByteNo,-1);
                        LCD_GotoXY(13,1);
                        LCD_Write_Int(TempTime,-1);*/
                        ResetIR();
                }
        }
        else
        {
                //Falling
                if((TempTime>41) && (TempTime<58))
                {
                        //We got a '0' here
                        BitNo++;
                        if(BitNo==8)
                        {
                                BitNo=0;
                                ByteNo++;
                                if(ByteNo==4)
                                {

                                        State=IR_WAIT_STOP_BIT;

                                }
                        }
                        MCUCR|=((1<<ISC01)|(1<<ISC00)); //Set INT on
Rising Edge
                        Edge=1;
```

```
                    }else if((TempTime>(169-(169*TOL))) &&
(TempTime<(169+(169*TOL))))
                    {
                        //We Have got a '1' here
                        IrData[ByteNo]|=(1<<BitNo);
                        BitNo++;
                        if(BitNo==8)
                        {
                            BitNo=0;
                            ByteNo++;
                            if(ByteNo==4)
                            {

                                State=IR_WAIT_STOP_BIT;

                            }
                        }
                        MCUCR|=((1<<ISC01)|(1<<ISC00)); //Set INT on
Rising Edge
                        Edge=1;

                    }else
                    {
                    /*Invalid Bit
                        stop=1;
                        LCD_Clear();
                        LCD_Write_String("Err in Bit Low/Sp");
                        LCD_Write_StringXY(1,0,"Bit");
                        LCD_Write_Int(BitNo,-1);
                        LCD_Write_String("Byt");
                        LCD_Write_Int(ByteNo,-1);
                        LCD_GotoXY(13,1);
                        LCD_Write_Int(TempTime,-1);
                    */

                    ResetIR();
                    }
                }
            break;
            }
```

```
	case IR_WAIT_STOP_BIT:
	{
		if(Edge)
		{
			//Check for integrity
			if(IrData[2]==((unsigned char)~IrData[3]))
			{
				//Now We Have Got a packet
				//Add its Cmd to Queue

					//Step1:Check of Q full
					if((QEnd==(QMAX-1) &&
QFront==0)||((QEnd+1)==QFront))
					{
						QFront++;
						if(QFront==(QMAX))
							QFront=0;
					}

					if(QEnd==(QMAX-1))
						QEnd=0;
					else
						QEnd++;

					IrCmdQ[QEnd]=IrData[2];
					PrevCmd=IrData[2];

					if(QFront==-1) QFront=0;
				//Prevent repeating immediatly
				Repeat=0;//It will be enabled after 4 repeat pulses
				RCount=0;


				ResetIR();
			}


		}
	}
	break;
```

```
	}
	GICR|=(1<<INT0);	//Enable INT0
}

void ResetIR()
{
	char i;
	for(i=0;i<4;i++) IrData[i]=0;
	State=IR_VALIDATE_LEAD_HIGH;
	//INT ON FALLING EDGE
	MCUCR&=(~((1<<ISC01)|(1<<ISC00)));
	MCUCR|=(1<<ISC01);
	Edge=0;
}


unsigned char GetRemoteCmd(char wait)
{
	unsigned char cmd;

	if(wait)
		while(QFront==-1);
	else
		if(QFront==-1) return (RC_NONE);

	cmd=IrCmdQ[QFront];

	if(QFront==QEnd)
		QFront=QEnd=-1;
	else
	{
		if(QFront==(QMAX-1))
			QFront=0;
		else
			QFront++;
	}

	return cmd;
}
```

الفصل الخامس

الخاتمة

التوجيهات والمقترحات :-

صمم هذا المشروع للتحكم في مروحة عن طريق الريموت كنترول بإستخدام Atmaga8 ومن المقترح إستخدام المشروع للتحكم بعدة أجهزه مختلفه ويتم ذلك عن طريق تنزيل البرنامج الخاص بالجهاز المعني في المايكروكنتروله ، إضافة الي ذلك يمكن أستخدام Atmaga16,32 للحصول علي دقه أكبر ومدي أعلي .

المراجع

1. Atmel atmega8L microcontroller datasheet, Rev.2486Z–AVR–02/11
2. http://en.wikipedia.org/wiki/Consumer_IR
3. http://extremeelectronics.co.in
4. C Programming for Microcontrollers Featuring ATMEL's AVR Butterfly and the Free WinAVR Compiler Joe Pardue SmileyMicros.
5. http://wiki.altium.com/display/ADOH/NEC+Infrared+Transmission+Protocol
6. AVR Studio Assembler/SimulatorTutorial, Last update: May 24, 2007
7. http://www.aaroncake.net/circuits/irremote.asp
8. http://en.wikipedia.org/wiki/Remote_control
9. Design and Construction of a Remote Controlled Fan Regulator Mahmud Shehu AHMED, Abubakar Sadiq MOHAMMED, Temitope George ONIMOLE, Paul Obafemi ATTAH *Department of Electrical and Computer Engineering, Federal University Technology*.
10. http://www.vishay.com/docs/80071/dataform.pdf