**LỜI MỞ ĐẦU**

Sự phát triển của công nghiệp hóa hiện đại hóa đất nước là một trong những chỉ tiêu để đánh giá sự phát triển của đất nước đó. Với sự phát triển của khoa học công nghệ và những yêu cầu của con người hiện nay và cả trong tương lai. Nghành robot tự động ra đời giải quyết các yêu cầu cần thiết của con người trong lĩnh vực đời sống và trong lao động.

Ở nước ta, mặc dù nghành tự động mới hình thành và phát triển nhưng đã thu hút rất nhiều bạn trẻ đam mê, tham gia sáng tạo và ứng dụng rất nhiều vào thực tế. Ngày nay với sự phát triển nhanh chóng của máy tính điện tử, công nghệ thông tin và những thành tựu của nghành điều khiển tự động đã làm cơ sở và hỗ trợ cho sự phát triển tương xứng trong lĩnh vực tự động hóa.

Các nước đã và đang phát triển đã ứng dụng thành tựu của nghành tự động hóa trong việc chế tạo robot thay thế con người làm việc trong các công việc nặng nhọc, môi trường làm việc nguy hiểm mà con người không thể làm viêc. Một trong những ứng dụng đó là **robot tự động (automatic)**. Robot còn tham gia các cuộc thi robocon do các nước Châu Á- Thái Bình Dương tổ chức hằng năm.

Với đề tài: “**Thiết kế và chế tạo Robot con tham gia Robocon 2014**” em đã học hỏi và vận dụng được rất nhiều kiến thức, đúc kết được nhiều kinh nghiệm quí báu qua thời gian thực hiện đề tài.

Với sự giúp đỡ nhiệt tình của các thầy trong bộ môn, trong khoa và đặc biệt là sự hướng dẫn, tận tình giúp đỡ của thầy chỉ đạo viên Ths Nguyễn Đắc Lựcđã giúp em hoàn thành tốt đề tài này. Do kiến thức và kinh nghiệm còn hạn chế nên bài báo cáo chắc chắn sẽ còn thiếu sót và em còn phải học hỏi rất nhiều.

Em xin chân thành cảm ơn các thầy cô trong nhà trường, khoa Cơ Khí, các thầy cô trong bộ môn, đặc biệt là thầy hướng dẫn Ths Nguyễn Đắc Lựcđã luôn quan tâm, tạo điều kiện thuận lợi nhất giúp em có thể hoàn thành đề tài này.

**Sinh viên thực hiện**

***Tống Thanh Thuận***

***Nguyễn Quang Tiên***

MỤC LỤC

[LỜI MỞ ĐẦU 1](#_Toc385088144)

MỤC LỤC  [2](#_Toc385088144)

[GIỚI THIỆU ĐỀ TÀI 5](#_Toc385088144)

[CHƯƠNG 1: 6](#_Toc385088162)

[1.1 Tóm tắt nhiệm vụ của Robot con 6](#_Toc385088145)

[1.2 Cấu trúc và các thông số kỹ thuật 6](#_Toc385088146)

[1.2.1 Giới thiệu về sân thi đấu ……6](#_Toc385088147)

[1.3 Thông số kỹ thuật yêu cầu của Robot con 10](#_Toc385088148)

[1.3.1. Giới hạn thông số kỹ thuật 10](#_Toc385088149)

[1.3.2. “Khởi động lại” Robot cha mẹ và Robot con 11](#_Toc385088150)

[1.3.3. Nguồn điện điều khiển 11](#_Toc385088152)

[1.3.4. Khối lượng 11](#_Toc385088153)

[1.3.5. Các đặc điểm kỹ thuật của Robot 11](#_Toc385088154)

[1.3.6 Thời gian thi đấu 11](#_Toc385088155)

[1.3.7 Thể thức thi đấu 12](#_Toc385088156)

[1.3.8 Cách tính điểm 13](#_Toc385088157)

[1.3.9 Phạm luật, trừ điểm 14](#_Toc385088158)

[1.5 Một số quy định khác 14](#_Toc385088159)

[1.5.1 Truất quyền thi đấu 14](#_Toc385088160)

[1.5.2 Độ an toàn 15](#_Toc385088161)

[CHƯƠNG 2: 16](#_Toc385088162)

TÍNH TOÁN THIẾT KẾ ROBOT CON [16](#_Toc385088163)

[2.1 Nhiệm vụ và giới hạn của Robot con 16](#_Toc385088164)

[2.1.1 Nhiệm vụ 16](#_Toc385088165)

[2.1.2 Giới hạn chung 16](#_Toc385088166)

[***2.2* Phương án thiết kế Robot con** 16](#_Toc385088167)

[2.2.1 **Nhiệm vụ** đi bộ trên cột 16](#_Toc385088168)

[2.2.2 **Nhiệm vụ** leo cầu thang](#_Toc385088168) 17

[2.2.3 **Nhiệm vụ** cầu bập bênh và xích đu 18](#_Toc385088168)

[2.2.4 **Phương** án giao tiếp giữa robot con va robot mẹ](#_Toc385088168) 19

[2.2.5 Robot còn phải được thiết kế trong các giới hạn sau](#_Toc385088168) 19

[2.3 Tính toán các thông số hình học - động học của Robot 19](#_Toc385088169)

[2.3.1 Phân tích cơ cấu 19](#_Toc385088170)

[2.3.2 Tính toán động học của các cơ cấu 20](#_Toc385088171)

[2.4 Hình ảnh robot tự động thiết kế bằng Solidworks 2010](#_Toc385088175) 25

2.5 [Hình ảnh robot tự động đã thi công.](#_Toc385088175) 27

[CHƯƠNG 3: 29](#_Toc385088162)

[THIẾT KẾ MẠCH ĐIỀU KHIỂN ROBOT CON 29](#_Toc385088163)

[3.1 Hệ thống điều khiển của Robot con](#_Toc385088177) 29

[3.1.1 Thống kê các tín hiệu vào ra](#_Toc385088180) 29

[3.1.2 Sơ đồ khối hệ thống điều khiển](#_Toc385088181) 30

[3.2 Thiết kế và tính toán mạch](#_Toc385088177) 29

[3.2.1 **Tổng quan về vi điều khiển PIC**](#_Toc385088180) 31

[3.2.2 Khối nguồn](#_Toc385088180) 45

[3.2.3 Khối điều khiển](#_Toc385088177) 46

[3.2.4 Khối công suất cho hai động cơ quay cánh tay](#_Toc385088180) 47

[3.2.5 Khối công suất cho hai động cơ leo cầu thang](#_Toc385088180) 48

[3.2.6 Khối mạch kích van](#_Toc385088180) 49

[CHƯƠNG 4: 50](#_Toc385088162)

[LƯU ĐỒ THUẬT TOÁN VÀ CHƯƠNG TRÌNH 50](#_Toc385088163)

[4.1 **Tổng Quan Về CCS**](#_Toc385088177) 50

[4.2 Lưu đồ thuật toán Robot con](#_Toc385088177) 53

[4.3 Chương trình](#_Toc385088177) 56

[CHƯƠNG 5: 50](#_Toc385088162)

[KẾT LUẬN 50](#_Toc385088163)

[TÀI LIỆU THAM KHẢO](#_Toc385088211)

**GIỚI THIỆU ĐỀ TÀI**

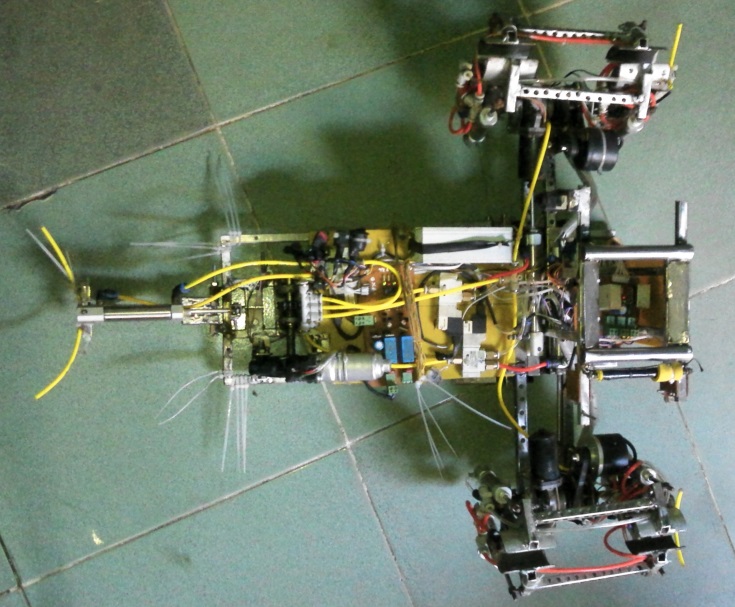
**Tên đề tài: Thiết kế và chế tạo Robot con tham gia Robocon 2014**

Với sự phát triển của nền kinh tế thế giới và nhu cầu cần thiết trong sinh hoạt và lao động, nghành sản xuất tự động ra đời nhằm đáp ứng các nhu cầu đó. Đặc biệt nghành sản xuất robot ra đời đáp ứng các nhu cầu thiết yếu trong lao động như robot thay thế con người làm việc trong những môi trường độc hại, làm những công việc nặng nhọc, ứng dụng robot trong quân sự, y tế, truyền thông,….

Với sự tìm tòi và ham học hỏi, sáng tạo, các sinh viên đã và đang tạo ra các chú robot tham gia các cuộc thi robocon do các nước Châu Á- Thái Bình Dương tổ chức hằng năm. Robot tự động là robot làm việc theo các chương trình do con người lập trình sẵn, làm việc theo ý muốn của người lập trình.

Với đồ án tổng hợp cơ điện tử đã giúp chúng em nắm lại các kiến thức đã học về lập trình vi điều khiển, tính toán truyền động cơ khí khí nén,…

Robot vận chuyển tự động ứng dụng vào thực tế rất nhiều ở các nhà máy, công xưởng. Vì vậy chúng em quyết định thực hiện đề tài **Thiết kế và chế tạo Robot con tham gia Robocon 2014.**

Do robot làm việc theo chương trình nên phụ thuộc rất nhiều vào kiến thức vào kĩ năng người lập trình.

Robot tự động

**CHƯƠNG 1:**

**GIỚI THIỆU CHUNG VỀ ROBOT CON**

# 

# 1.1 Tóm tắt nhiệm vụ của Robot con:

Với chủ đề "ASALUTE TO PARENTHOOD" (Tri ân đấng sinh thành) trong thời gian 3 phút của một trận đấu robot con phải cùng robot bằng tay thực hiện 4 nhiệm vụ là:

* Seesaw (cầu bập bênh)
* Swing (xích đu)
* Pole walk (đi bộ trên cột)
* Jungle gym (leo cầu thang)

Khi Robot con hoàn thành 3 nhiệm vụ là Seesaw (cầu bập bênh), Pole walk (đi bộ trên cột) và Swing (xích đu), nó có thể thực hiện nhiệm thứ 4 là Jungle gym (leo cầu thang), được đặt giữa công viên. Đội hoàn thành nhiệm vụ thứ 4 đầu tiên, sẽ đạt được “SHABAASH”và là đội thắng cuộc.

Nếu không đội nào hoàn thành nhiệm vụ chiến thắng trên, đội thắng cuộc sẽ được được quyết định dựa trên luật.

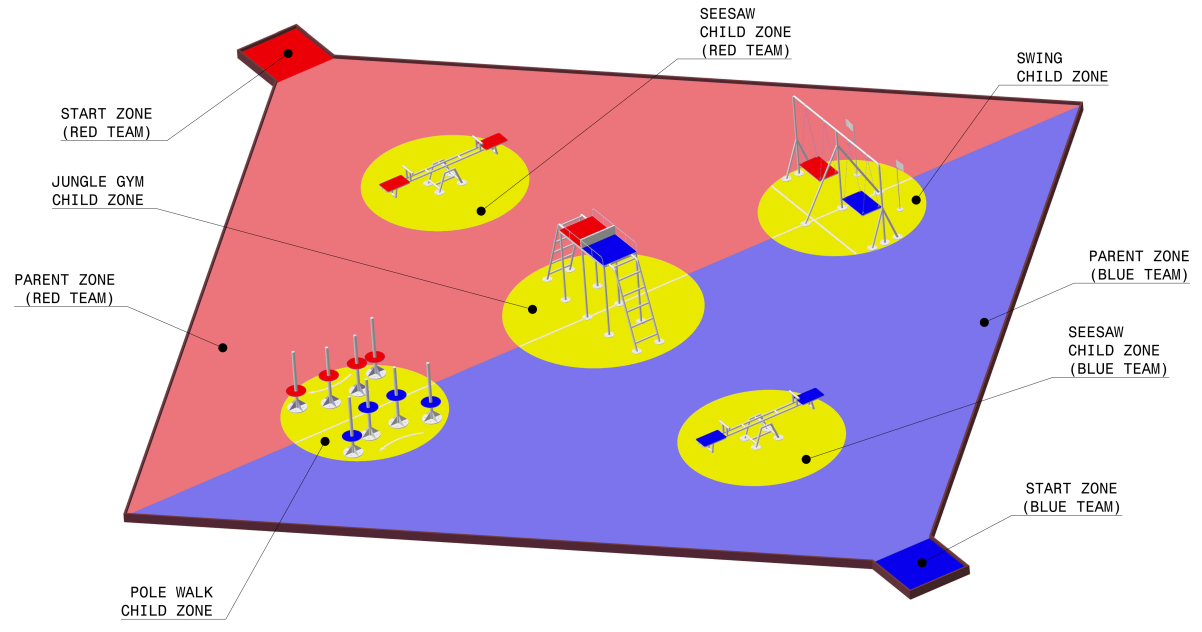
# 1.2 Cấu trúc và các thông số kỹ thuật của sân thi đấu:

1.2.1 Giới thiệu về sân thi đấu:

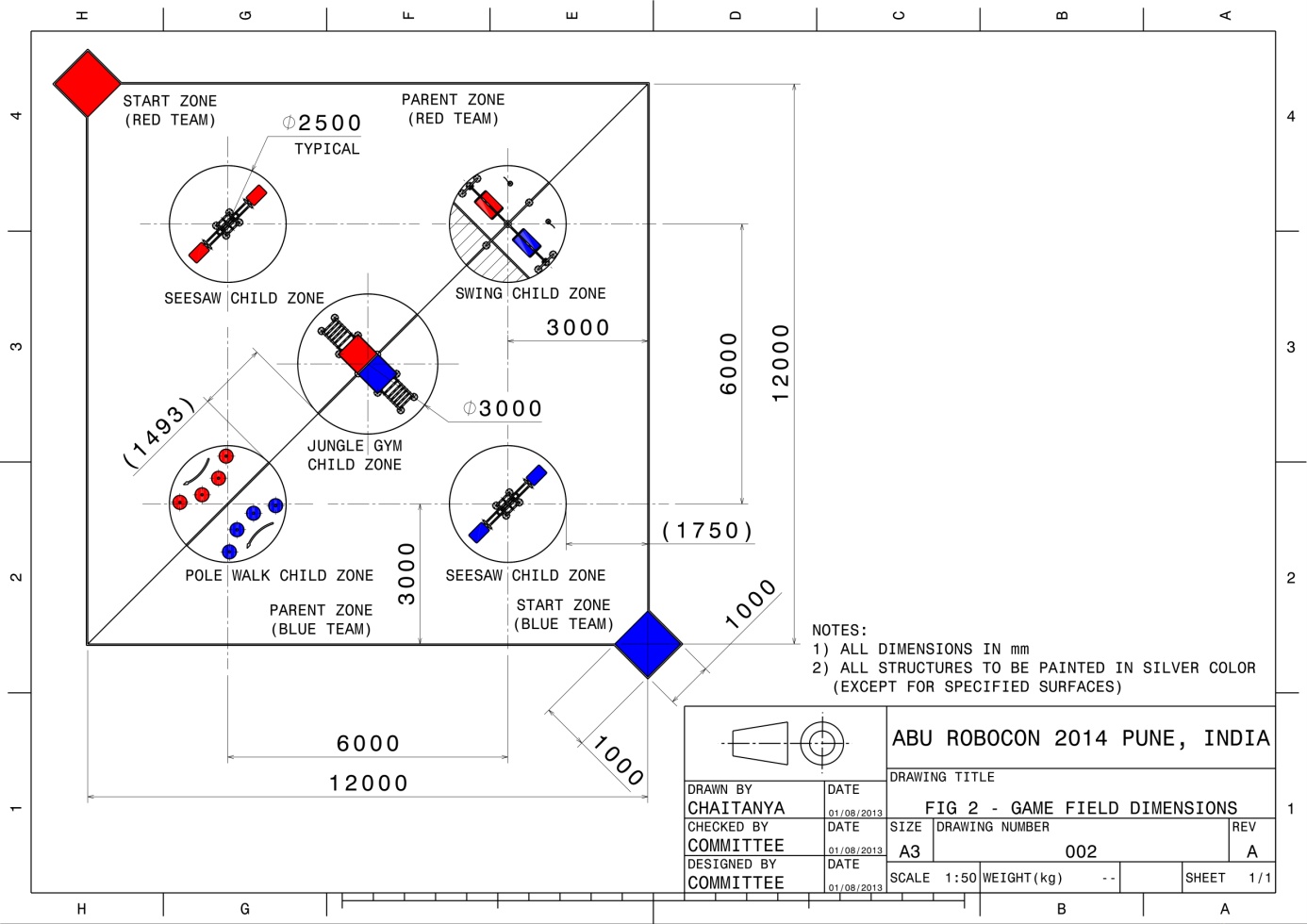
Sân thi đấu là khu vực có kích thước 12000 mm x 12000 mm được bao quanh bởi hàng rào làm bằng gỗ cao 50 mm và độ dày 30 mm.

Các đội thi đấu gồm đội đỏ và đội xanh. Sân thi đấu có hai khu vực dành cho Robot cha mẹ một cho đội đỏ, một cho đội xanh và năm khu vực trò chơi dành cho trẻ em các khu vực trò chơi. Robot con sẽ thực hiện nhiệm vụ trong các khu vực dành cho trẻ em. Nhiệm vụ đầu tiên bập bênh dành cho cả 2 đội đỏ và xanh đều giống nhau và mỗi đội sẽ thực hiện nhiệm vụ này ở khu vực sân chơi dành cho mỗi đội. Những nhiệm vụ còn lại của 2 đội sẽ được bố trí đối xứng nhau qua đường băng dính trắng không bóng rộng 30mm, đây cũng chính là ranh giới giữa hai đội. Mỗi đội sẽ phải thực hiện những nhiệm vụ của mình ở khu vực sân chơi của mỗi đội.

Sân thi đấu cũng bao gồm một khu xuất phát cho đội xanh và một khu xuất phát cho đội đỏ. Mỗi khu xuất phát có kích thước 1000mm x 1000mm.



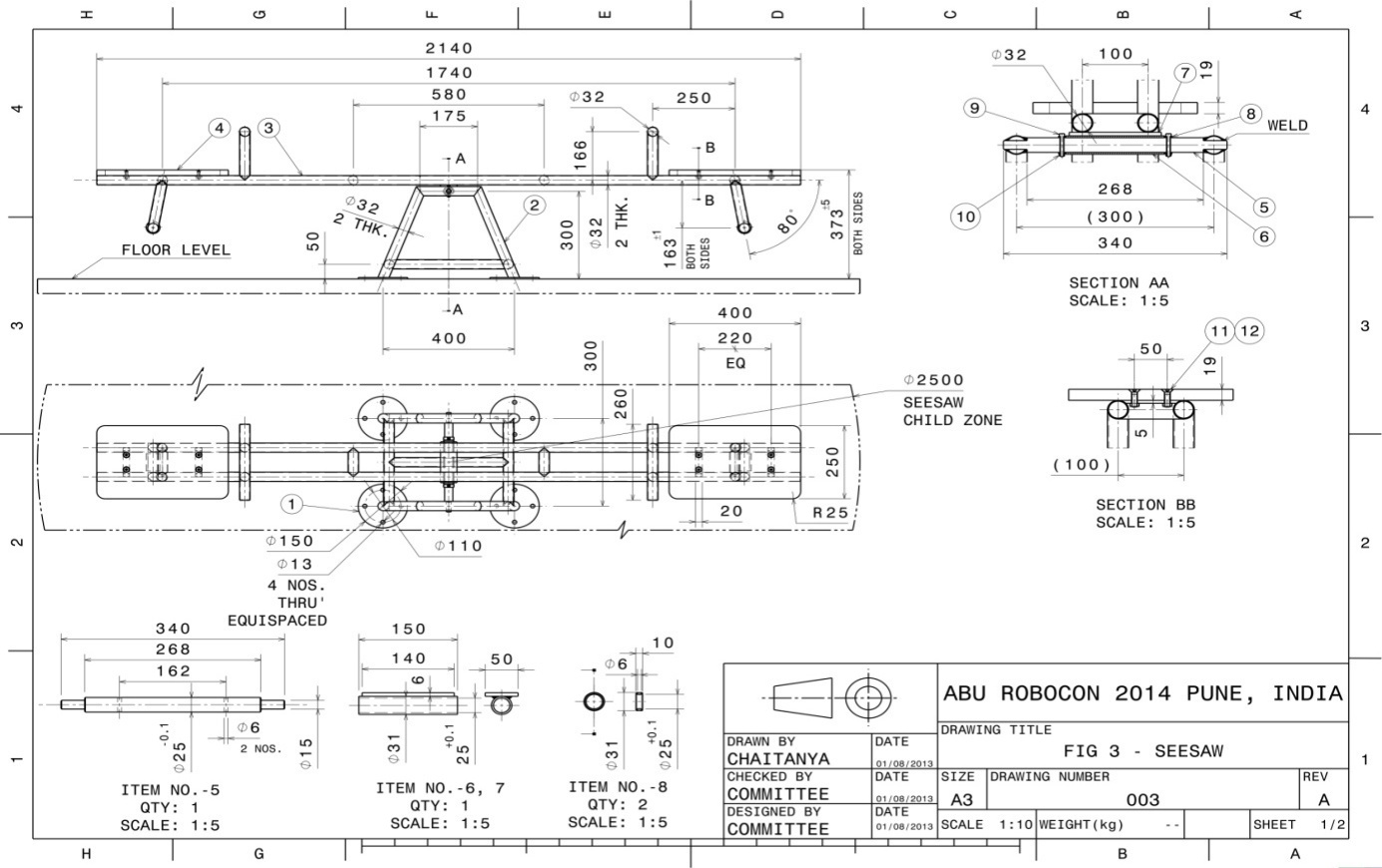
***Hình 1.1: Tổng thể sân thi đấu***

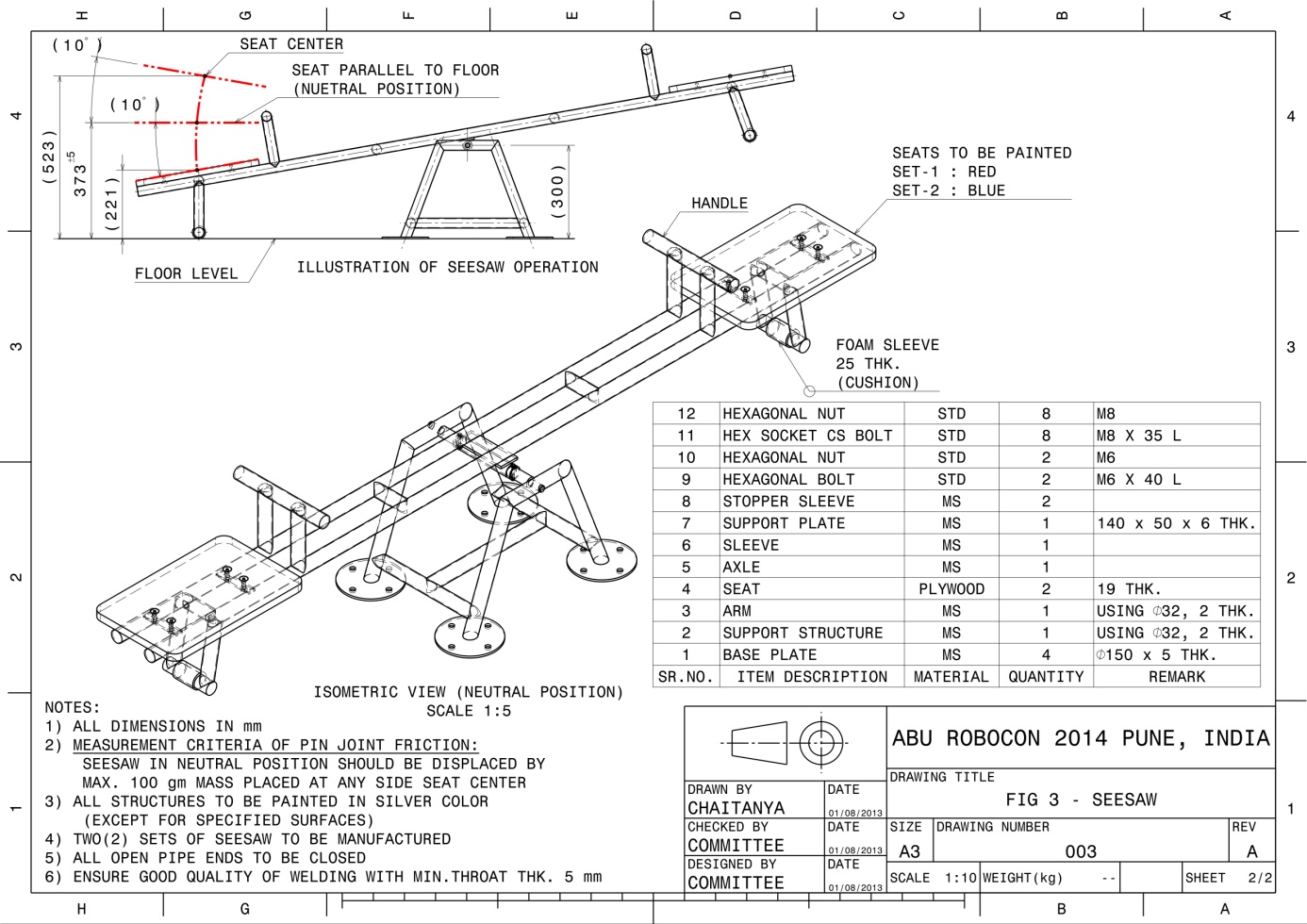
******

***Hình 1.2: Kích thước sân thi đấu***

Các hoạt động trò chơi được diễn ra trong khu trẻ em theo danh sách sau:

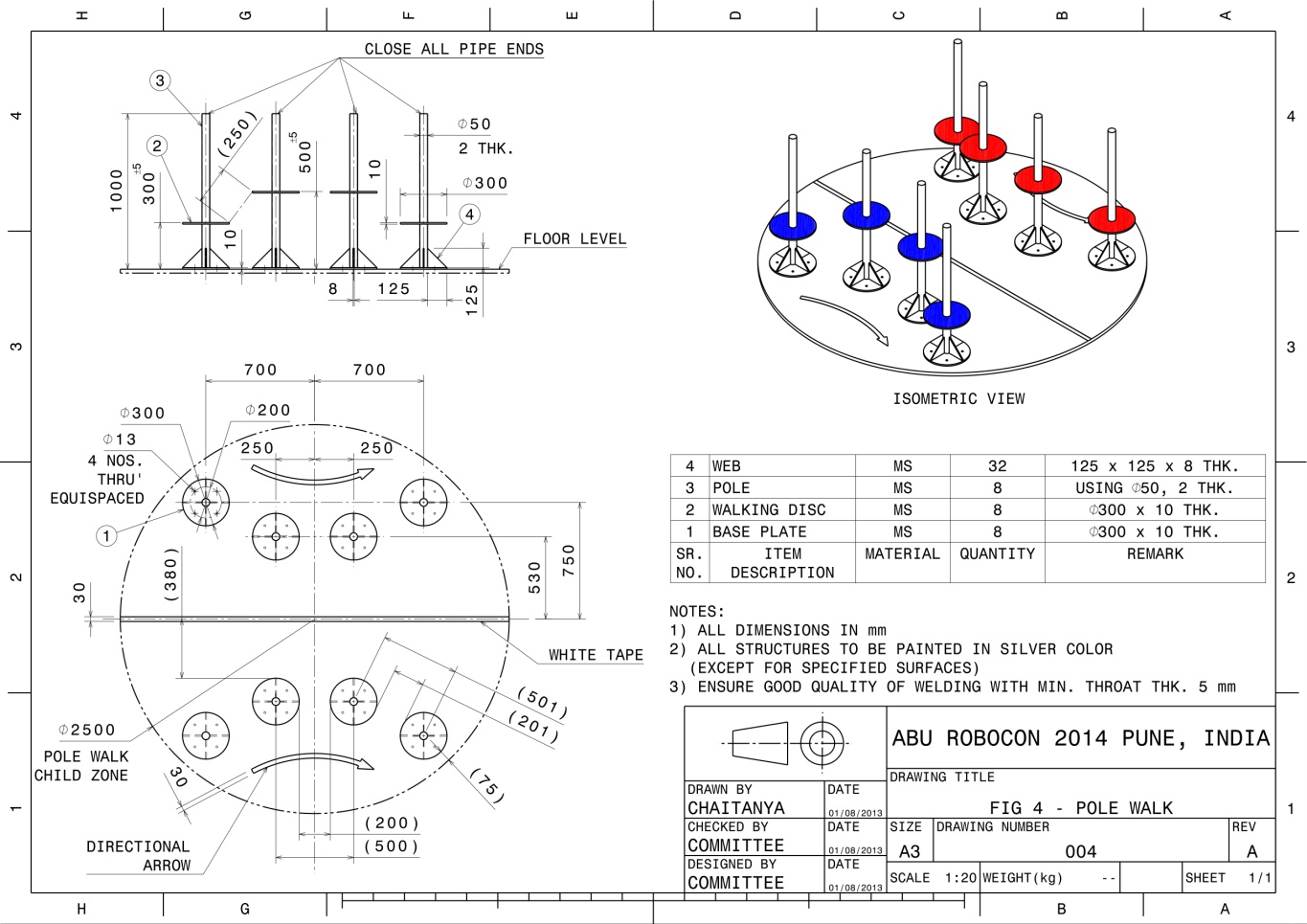
****1.2.2** *Khu vực trò chơi Seesaw (cầu bập bênh):***

**

**

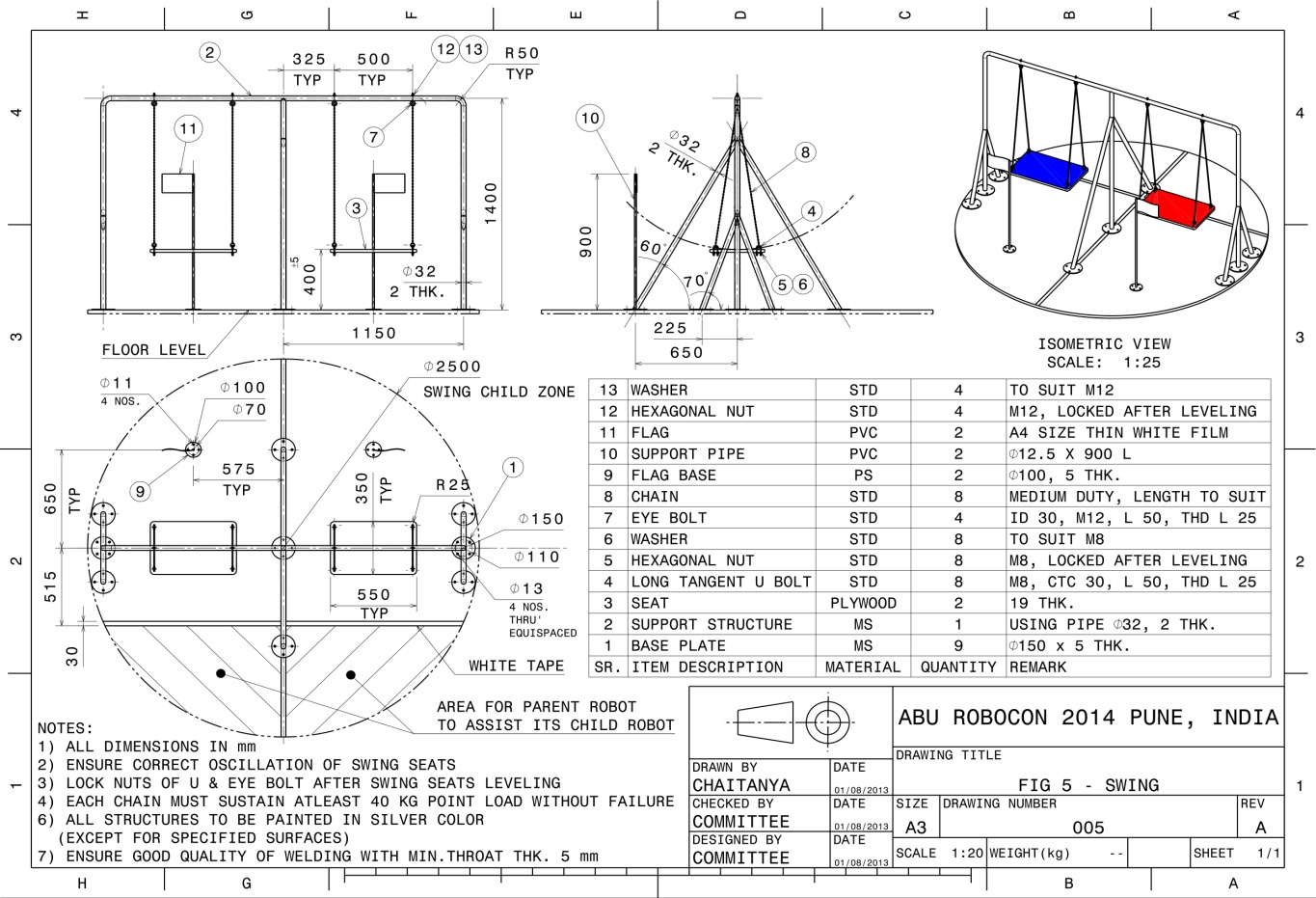
***Hình 1.3: Khu vực trò chơi Seesaw (cầu bập bênh).***

****1.2.3 Khu vực trò chơi Pole walk (đi bộ trên cột):****

****

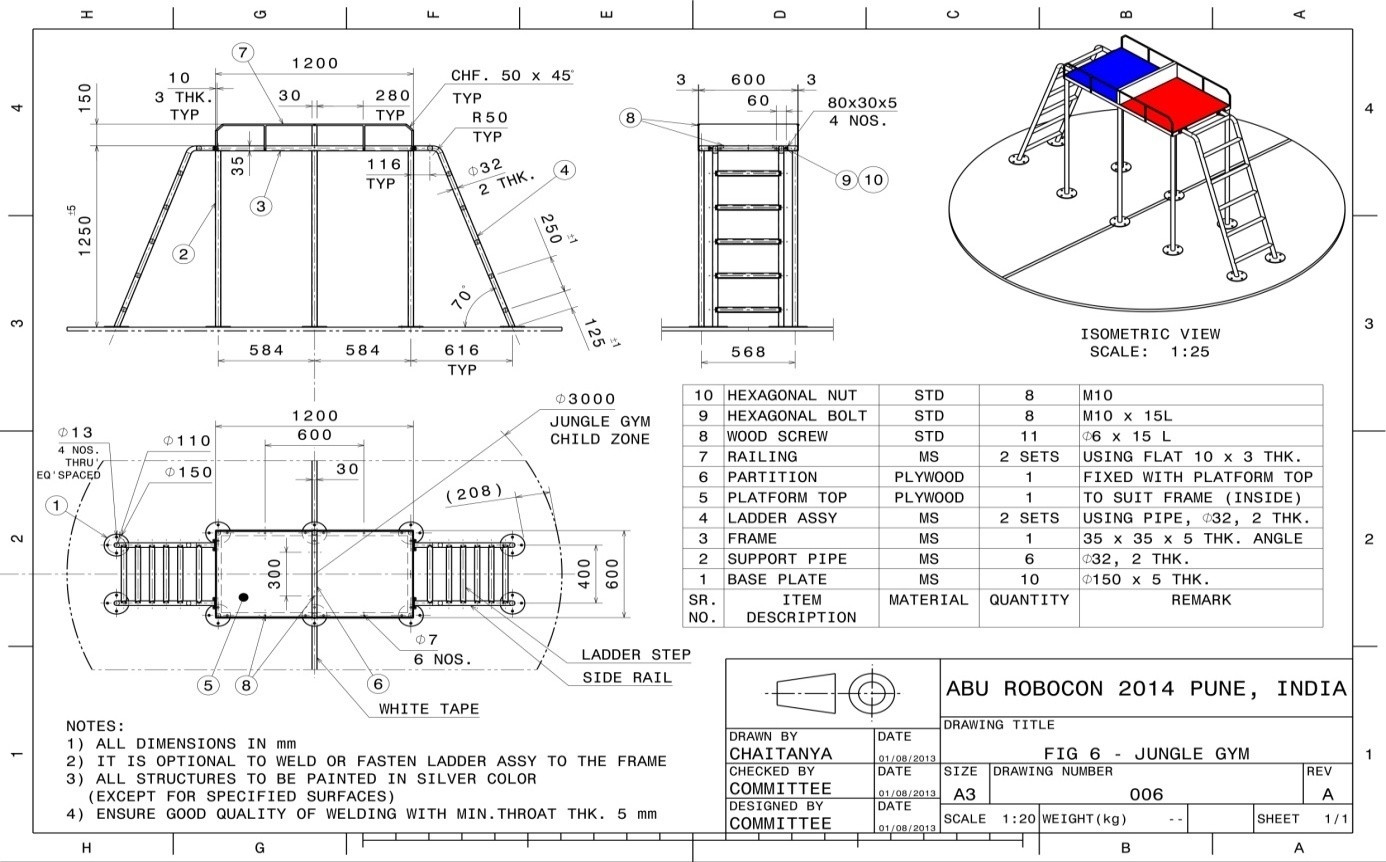
***Hình 1.4: Khu vực trò chơi Pole walk (đi bộ trên cột).***

****1.2.4 Khu vực trò chơi Swing (xích đu):****

****

***Hình 1.5: Khu vực trò chơi Swing (xích đu).***

****1.2.5 Khu vực trò chơi Jungle gym (cầu thang vận động**):**



***Hình 1.6: Khu vực trò chơi Jungle gym (cầu thang vận động).***

# 1.3 Thông số kỹ thuật yêu cầu của Robot con:

1.3.1 Giới hạn thông số kỹ thuật:

- Robot con sẽ không được có bánh xe hay bánh xích để chuyển động. Nó có thể có những cánh tay hoặc chân.

- Robot con không được phép chạm xuống mặt sàn của vùng khởi động và khu vực của Robot cha mẹ .

- Robot con phải hoạt động một cách tự động.

- Sau khi đặt Robot con vào khu trẻ em, Robot cha mẹ khởi động Robot con thông qua một tín hiệu.

- Khi Robot con hoạt động, không thành viên của đội được phép chạm vào các Robot trừ trường hợp khởi động lại.

- Kích thước và hình dạng của Robot con có thể thay đổi trong suốt trận đấu, nhưng nó phải nằm trong khối lập phương 500 mm tại mọi thời điểm.

- Robot con không được phép chia hoặc tách thành 2 hay nhiều khối.

- Phải có khoảng trống để dán stickers/tags (bởi ban tổ chức cuộc thi) trên Robot con. Khoảng trống này có kích thước 150 mm x 100 mm.

1.3.2 “Khởi động lại” Robot cha mẹ và Robot con:

- Không hạn chế số lần “khởi động lại “ cho Robot cha mẹ và Robot con.

- Trong quá trình “khởi động lại”, Robot con phải khởi động lại bằng cách đặt lên Robot cha mẹ trong khu vực cha mẹ.

- Sau khi xin “khởi động lại” Robot con và được cho phép, Robot con phải được khởi động lại trên Robot cha mẹ. Điểm khởi động lại chính là nơi Robot con xin được khởi động lại.

- Các chiến thuật dựa trên việc khởi động lại thì không được phép.

- Trong khi khởi động lại, không bộ phận nào của Robot được tháo ra; nguồn của Robot không được nạp lại. Kể cả thêm nguồn vào cho Robot cũng không được phép.

1.3.3 Nguồn điện điều khiển:

-Mỗi đội sẽ tự chuẩn bị nguồn cho Robot đội mình.

- Điện áp nguồn nuôi các Robot phải nhỏ hơn hoặc bằng 24VDC.

- Nếu sử dụng bình khí nén thì áp suất tối đa không được quá 6 Bars.

- Nguồn nuôi được xem là nguy hiểm hoặc không hợp lệ sẽ không được phép sử dụng.

1.3.4 Khối lượng:

Robot cha mẹ và Robot con kể cả nguồn nuôi, cable, bộ điều khiển, và các bộ phận khác của mỗi robot phải được cân trước khi thi đấu. Tổng trọng lượng của các Robot và các phụ kiện trên cho mỗi đội không vượt quá 40kg. Tổng trọng lượng 40kg không bao gồm pin thay thế và các thành phần có đặc điểm kỹ thuật tương tự.1.4 Nhiệm vụ thi đấu và nội quy chung.

1.3.5 Các đặc điểm kỹ thuật của Robot:

Kích thước và trọng lượng của mỗi Robot sẽ được kiểm tra trước khi trận đấu.Những Robot được chế tạo mà không phù hợp với luật lệ sẽ không được phép tham gia vào cuộc thi.

1.3.6 Thời gian thi đấu:

- Việc chuẩn bị các Robot cần phải hoàn thành trong vòng 1 phút sau khi nhận được tín hiệu chuẩn bị.

- Trận đấu kéo dài trong vòng 3 phút.

- Khi một đội đạt được chiến thắng tuyệt đối “SHABAASH”thì trận đấu ngay lập tức được kết thúc bất kể đội bạn ghi được bao nhiêu điểm.

1.3.7 Thể thức thi đấu:

- Robot cha mẹ sẽ bế Robot con xuất phát từ khu vực khởi động, đến khu vực cầu bập bênh của đội mình và hoàn thành trò chơi như sau:

- Cầu bập bênh :

+ Robot cha mẹ đặt Robot con ngồi lên cầu bập bênh và rời khỏi Robot con.

+ Robot con phải ngồi trên ghế của cầu bập bênh. Robot con được phép chạm vào ghế ngồi và phần tay cầm cạnh ghế để ngồi vững trên cầu bập bênh.

+ Robot cha mẹ phải di chuyển đến đầu đối diện của cầu bập bênh và không được ngồi lên cầu bập bênh, nó phải chơi với Robot con và hoàn thành ít nhất ba (3) lần bập bênh liên tục.

+ Robot cha mẹ đón lại Robot con.

+ “Hoàn thành một lần bập bênh” nghĩa là: ghế ngồi cầu bập bênh của Robot con và chỗ ngồi đối diện phải chạm vào mặt sàn một lần cho mỗi lần bập bênh.

+ Số lần bập bênh bắt đầu tính khi ghế ngồi ở phía Robot cha mẹ chạm vào mặt sàn lần đầu tiên sau khi đặt thành công Robot con lên ghế ngồi.

+ Trong khi chơi bập bênh, Robot con không được chạm vào mặt sàn của khu vực trò chơi bập bênh.

- Sau khi hoàn thành nhiệm vụ ở khu bập bênh, đội đủ điều kiện sẽ tiếp tục chơi ở khu đi bộ trên cột hoặc khu xích đu.

- Đi bộ trên cột

+ Robot cha mẹ đặt Robot con lên đĩa đi bộ của đội mình và rời khỏi Robot con.

+ Robot con phải đứng trên đĩa đi bộ của đội mình.

+ Robot con được phép chạm vào đĩa đi bộ và phần cột phía trên đĩa trong khi thực hiện quá trình đi bộ trên cột.

+ Robot con phải đi bộ trên đĩa và hoàn thành phần chơi bằng cách di chuyển từ đĩa này sang đĩa khác mà không bỏ sót bất cứ đĩa nào của đội mình và cũng không chạm vào mặt sàn trong khu vực đi bộ trên cột.

+ Robot cha mẹ đón lại Robot con

- Xích đu

+ Robot cha mẹ đặt Robot con lên ghế xích đu của đội mình và rời khỏi Robot con.

+ Robot con được phép chạm vào ghế xích đu và hai dây xích đu.

+ Robot cha mẹ có thể bắt đầu đẩy ghế xích đu đung đưa lần thứ nhất mà không chạm vào Robot con, 2 lần đung đưa tiếp theo do Robot con tự thực hiện. Tuy nhiên, Robot cha mẹ có thể hỗ trợ Robot con ngừng đung đưa.

+ Robot cha mẹ đón lại Robot con.

+ “Hoàn thành 1 lần đung đưa” chỉ được tính: khi bất cứ bộ phận nào của Robot con chạm vào lá cờ cắm ở trước xích đu và phải hoàn thành 3 lần đung đưa liên tiếp như vậy.

+ Robot con không được chạm xuống mặt sàn của khu vực xích đu trong khi đung đưa.

- Chỉ sau khi hoàn thành 3 trò chơi, đội mới đủ điều kiện chơi ở khu cầu thang vận động.

- Cầu thang vận động

+ Robot cha mẹ chuyển Robot con đến chính xác hoặc gần vị trí bậc thang đầu tiên của cầu thang và rời khỏi Robot con.

+ Robot con leo lên cầu thang mà không được phép chạm vào 2 thành cầu thang trong quá trình trèo lên trên đỉnh của cầu thang vận động.

+ Sau khi leo đến đỉnh cầu thang, Robot con phải vẫy lá cờ được đặt sẵn bên trong robot. Đội nào vẫy được lá cờ trước trong thời gian 3 phút của trận đấu sẽ dành được chiến thắng tuyệt đối và chiến thắng này được gọi là “SHABAASH”. Mỗi đội phải tự trang bị cờ “SHABAASH” của đội mình.

+ Để đạt được “SHABAASH”, lá cờ phải được vẫy ở trên điểm cao nhất của Robot con.

+ Đội nào hoàn thành nhiệm vụ tại khu cầu thang vận động trước, sẽ đạt được “SHABAASH”, và là đội thắng cuộc, khi đó trận đấu ngay lập tức được kết thúc.

1.3.8 Cách tính điểm:

- Chỉ sau khi hoàn thành tất cả các nhiệm vụ của mỗi nhiệm vụ; trận đấu sẽ được xem là hoàn tất và đạt được điểm số dành cho mỗi nhiệm vụ đặt ra.

- Để có thêm điểm, phần chơi cầu bập bênh có thể thực hiện tối đa 3 lần (mỗi lần 3 lượt bập bênh).Để có thêm điểm ở phần chơi đi bộ trên cột và xích đu, mỗi đội có thể thực hiện tối đa 2 lần.

- Trong trường hợp “khởi động lại” ở phần chơi cầu bập bênh và xích đu, số lần thực hiện sẽ tính lại từ đầu. Những lần thực hiện thành công trong trường hợp này sẽ không được tính.

- Điểm số:

+ Hoàn thành trò chơi đầu tiên (cầu bập bênh) được 10 điểm cho mỗi lần hoàn thành (có thể thực hiện tối đa 3 lần để có thêm điểm).

+ Hoàn thành trò chơi thứ 2 (đi bộ trên cột hay xích đu) được 20 điểm cho mỗi lần hoàn thành (có thể thực hiện tối đa 2 lần để có thêm điểm).

+ Hoàn thành trò chơi thứ 3 (trò chơi còn lại của đi bộ trên cột hay xích đu) được 50 điểm cho mỗi lần hoàn thành (có thể thực hiện tối đa 2 lần để có thêm điểm).

+ Hoàn thành trò chơi thứ 4 (cầu thang vận động) đội sẽ đạt được “SHABAASH”. Nhiệm vụ cuối cùng này sẽ không tính điểm và đội chơi nào dành được sẽ là đội chiến thắng tuyệt đối.

1.3.9 Phạm luật, trừ điểm:

Khi trận đấu bắt đầu, những hành động sau được cho là phạm luật và sẽ bị trừ 2 điểm cho một lần vi phạm.

- Bất cứ bộ phận nào của robot chạm vào robot đối phương.

- Robot cha mẹ hoặc thành viên trong đội chạm vào sàn của khu trò chơi trẻ em, ngoại trừ trường hợp “khởi động lại”.

- Người điều khiển Robot cha mẹ xâm phạm vùng không gian của khu trò chơi trẻ em, ngoại trừ trường hợp “khởi động lại”.

- Robot cha mẹ hoặc Robot con của đội xâm phạm phần sân đối phương hoặc vùng không gian phía trên nó.

- Robot con chạm xuống sàn của vùng khởi động, và khu vực Robot cha mẹ.

- Robot con cố ý chạm xuống sàn của khu cầu bập bênh, xích đu, và đi bộ trên cột.

- Thực hiện các hành động bị cấm đã nêu trong luật chơi và FAQs

# 1.5 Một số quy định khác:

1.5.1 Truất quyền thi đấu:

Những hành vi sau sẽ được trọng tài xem xét để truất quyền thi đấu của đội trong trận đó. Đội sẽ không được tiếp tục trận đấu đó và số điểm ghi được của trận đấu đó sẽ không được ghi nhận.

- Các hành vi nhằm phá hỏng sân thi đấu, các trang thiết bị sân hoặc robot đối phương.

- Các thành viên của đội cố ý chạm vào các Robot đội mình trong trận đấu.

- Robot hay thành viên đội cố tình cản phá, va chạm hay tấn công các Robot đối phương trực tiếp hoặc gián tiếp. Tuy nhiên, nếu Robot của đội nào di chuyển hay rớt vào vùng giới hạn do tai nạn thì phải lập tức được lấy ra theo hướng dẫn của trọng tài. (Trọng tài được quyền xác định lỗi do tai nạn hay không). Đội được phép khởi lại.

- Hai lỗi xuất phát trong một trận đấu (Robot cha mẹ xuất phát trước hiệu lệnh của trọng tài).

- Có bất cứ hành động nào trái với tinh thần fairplay.

- Cố ý dùng cable để lái hoặc kéo Robot cha mẹ.

- Không tuân theo hướng dẫn hoặc cảnh báo của trọng tài.

- Truyền thông tin qua lại giữa các Robot (như IR, RF, LASER, ...).

1.5.2 Độ an toàn:

- Tất cả các Robot phải được thiết kế an toàn, không gây nguy hiểm tới người điều khiển, các trọng tài hoặc khán giả và trang thiết bị trên sân.

- Nếu sử dụng tia la-ze thì phải yếu hơn tia la-ze cấp 2, và phải bảo đảm không gây nguy hiểm cho người điều khiển, trọng tài hoặc khán giả.

- Robot phải có cấu trúc để dễ kiểm tra độ an toàn trong đoạn phim kiểm tra Robot hay quá trình chạy thử.

- Các loại động cơ chạy bằng xăng, chất dễ cháy, nổ và bình nén khí áp suất cao, năng lượng tái tạo từ hóa chất đều bị cấm khi sử dụng cho các chuyển động hay mở rộng kích cỡ của Robot.

**CHƯƠNG 2:**

**TÍNH TOÁN THIẾT KẾ ROBOT CON**

**2.1 Nhiệm vụ và giới hạn của Robot con:**

**2.1.1 Nhiệm vụ:**

Robot con có nhiệm vụ thực hiện các trò chơi : cầu bập bênh, đi bộ trên cột, xích đu và khu cầu thang vận động. Nhưng muốn thực hiện chơi các trò chơi này robot con đều cần sự trợ giúp của robot cha mẹ ( như thực hiện bập bênh, xích đu, vận chuyển robot con đến các khu vực…).

### **2.1.2 Giới hạn chung:**

***-*** Robot con phải thực hiện nhiệm vụ hoàn toàn tự động.

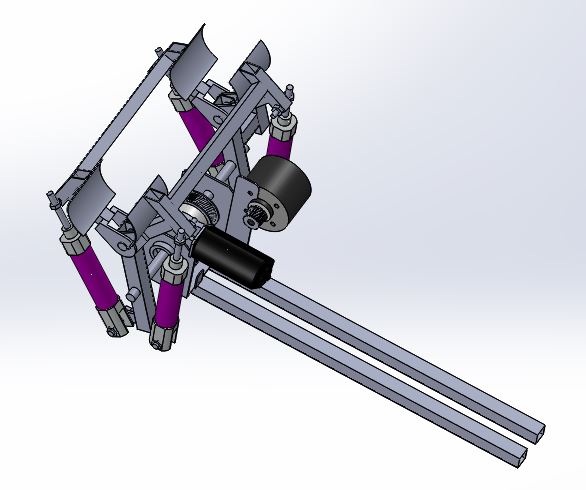
***-*** Trong khu vực xuất phát của Robot con và trong quá trình hoạt động thực hiện nhiệm vụ của mình thì kích thước robot phải phù hợp với hình lập phương kích thước 500x500x500mm

# 2.2 Phương án thiết kế Robot con:

### 2.2.1 **Nhiệm vụ** đi bộ trên cột:

* ***Phương án 1*:** Thiết kế robot kẹp đĩa và kết hợp với thanh trượt để di chuyển robot qua các đĩa.
* *Ưu điểm :*
* Robot mẹ giao tiếp với robot con dễ dàng, không yêu cầu độ chính xác cao khi giao tiếp.
* Giải quyết được vấn đề do luật đề ra là bắt buộc chạm đĩa.
* *Nhược điểm:*
* Cơ khí phức tạp.
* Robot cần độ cứng vững cao.
* ***Phương án 2*:** Thiết kế robot kẹp trụ và xoay quanh trụ để di chuyển robot sang các trụ.
* *Ưu điểm :*
* Cơ khí đơn giản, dễ chế tạo.
* Độ an toàn cao.
* Di chuyển nhanh và dễ dàng.
* *Nhược điểm :*
* Đòi hỏi giao tiếp phải chính xác.
* Chạm đĩa cần phải qua bộ phận trung gian để đúng luật.

Sau khi phân tích 2 phương án ta thấy rằng mỗi phương án đều có một ưu thế nhất định.Tuy nhiên do điều kiện kinh tế và cơ sở vật chất còn hạn chế nên ta chọn phương án 2 để thiết kế.

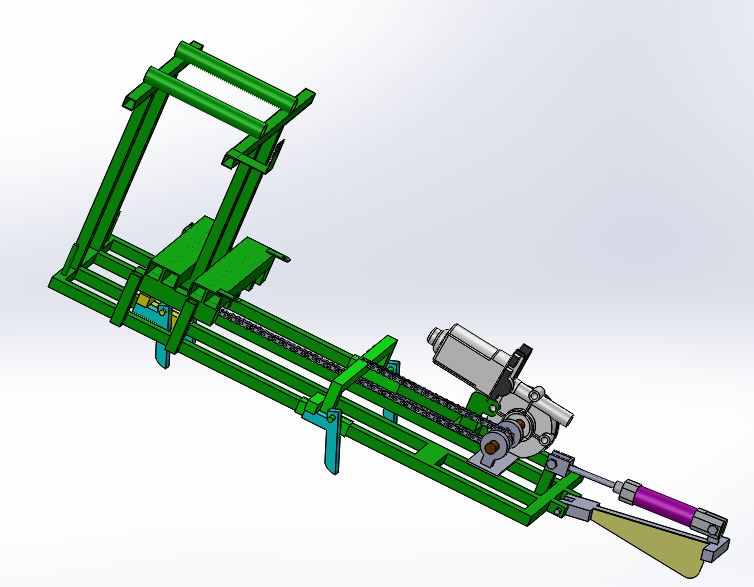


**Hình 2.1: Bản vẽ kết cấu của tay kẹp.**

### 2.2.2 **Nhiệm vụ** leo cầu thang:

* ***Phương án 1:***Sử dụng cơ cấu di chuyển trụ để leo cầu thang.
* *Ưu điểm :*
* Robot gọn nhẹ
* Không phải thiết kế thêm cơ cấu để thực hiện nhiệm vụ này.
* *Nhược điểm :*
* Cần dùng động cơ có công suất lớn.
* Một số vấn đề phát sinh theo với đông cơ làm tăng trọng lượng robot con.
* Di chuyển chậm
* ***Phương án 2:*** Sử dụng một xilanh và các lẩy tự lựa để di chuyển liên tục trên các bậc cầu thang.
* *Ưu điểm:*
* Thiết kế gọn, nhẹ.
* Di chuyển nhanh.
* *Nhược điểm:*
* Phụ thuộc nhiều vào áp suất khí nén.
* Chi phí cao.
* ***Phương án 3:*** Sử dụng động cơ và các lẩy tự lựa để di chuyển liên tục trên các bậc cầu thang.
* *Ưu điểm:*
* Di chuyển chắc chắn.
* Nguồn cung cấp ổn định.
* *Nhược điểm:*
* Khó gá đặt, nặng, cồng kềnh
* Di chuyển chậm.

Sau khi phân tích so sánh cả 3 phương án, kết hợp với điều kiện kinh tế thì ta thấy phương án 3 là phù hợp với các điều kiện đặt ra nên em chọn phương án này.

 **Hinh vẽ 2.1: Bản vẽ cơ cấu leo thang.**

### 2.2.3 **Nhiệm vụ** cầu bập bênh và xích đu:

Ta nhận thấy rằng ở hai nhiệm vụ này chủ yếu phụ thuộc vào hoàn toàn robot mẹ. Nên ta tận dụng chính trọng lượng của robot để nằm yên trên bập bênh và xích đu. Ta không cần phải thêm cơ cấu kẹp chặt robot ở hai nhiệm vụ này. Dể tiết kiệm kinh tế và thời gian.

### 2.2.4 **Phương** án giao tiếp giữa robot con và robot mẹ:

* Tay kẹp nằm trên robot bố mẹ để đảm bảo khối lượng nhỏ cho robot con.
* Giao tiếp bằng công tắc hành trình đảm bảo độ ổn định và an toàn.

Từ những nhiệm vụ của robot con và kết cấu các modum sân thi đấu chúng ta phải thiết kế kết cấu robot sao cho trọng tâm của robot phải ở giữa cân bằng khối lượng của robot khi thực hiện nhiệm vụ.Trong các nhiệm vụ của robot con ta nhận thấy nhiệm vụ đi bộ trên cột và nhiệm vụ leo cầu thang là 2 nhiệm vụ phức tạp nhất, nhiều cơ cấu nhất và quan trọng nhất nên phải thiết kế robot dựa trên cơ sở của các cơ cấu thực hiện nhiệm vụ này.

**2.2.5**  **Robot còn phải được thiết kế trong các giới hạn sau:**

* Robot phải có khối lượng <10kg.
* ***Ưu điểm*:**
  + - * Kết hợp và giao tiếp với Robot bằng tay dễ dàng, nhanh, hiệu quả.
      * Đảm bảo an toàn cho phương án leo cột và phương án leo cầu thang.
      * Robot nhỏ, nhẹ thực hiện vận chuyển dễ dàng
      * Giao tiếp với robot cha mẹ bằng công tắc hành trình nên đảm bảo độ an toàn và ổn định cao khi thực hiện nhiệm vụ.
      * Cơ cấu chủ yếu là xilanh khí nén làm cho việc thực hiện nhanh, hiệu quả và có độ chính xác cao hơn.
* ***Nhược điểm*:**
* Cơ cấu leo cột hơi phức tạp, cần tính toán và gia công chắc chắn.

# 2.3 Tính toán các thông số hình học - động học của Robot:

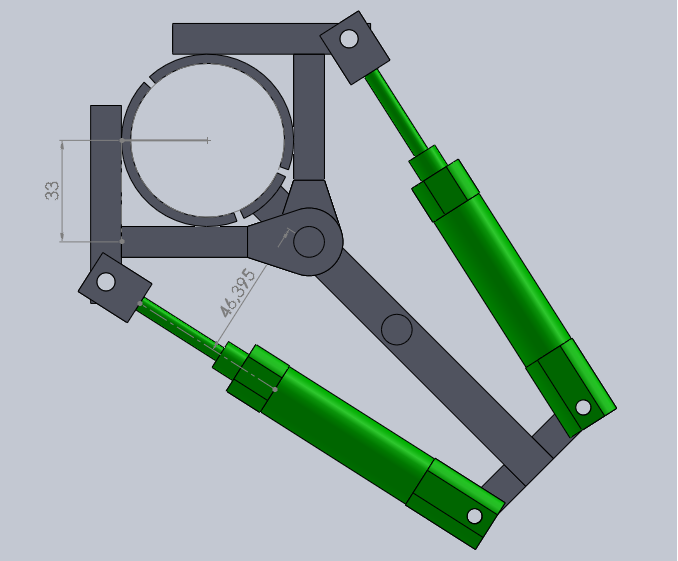
2.3.1 Phân tích cơ cấu :

- Để đảm bảo phương án hoạt động thì Robot con phải được mang tại vị trí robot nằm thẳng đứng.

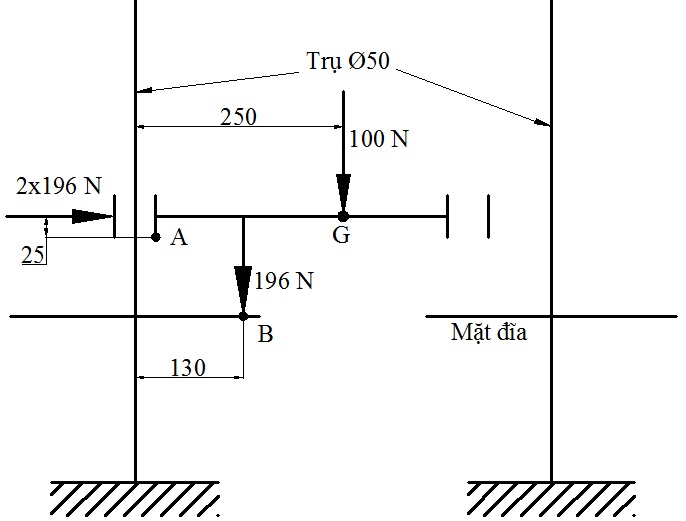
- Robot cần được thiết kế đối xứng để đảm bảo tính ổn định của Robot và đảm bảo phương án hoạt động khi đổi sân.

- Robot phải giao tiếp trong thời gian ngắn, robot bố mẹ phải kẹp chặt và ổn định robot con trong khi di chuyển.

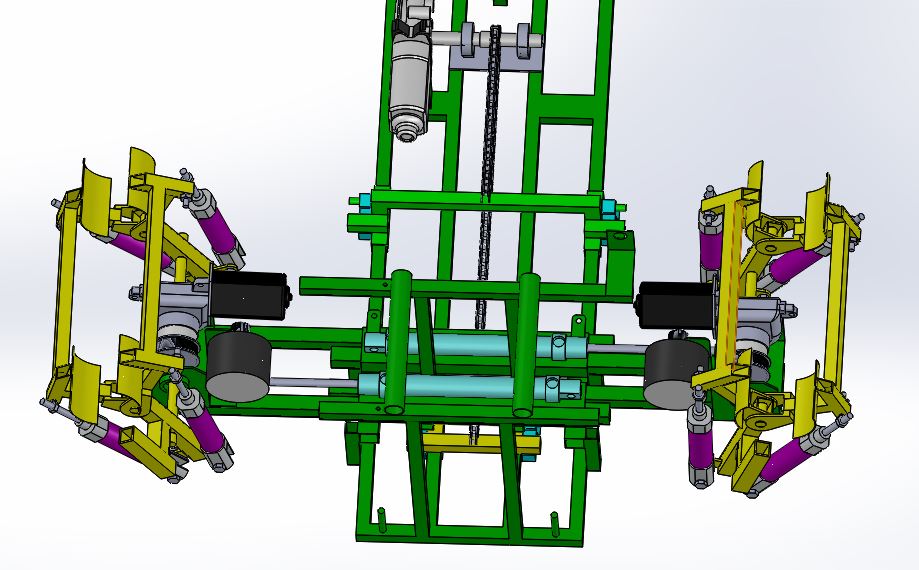
2.3.2 Tính toán động học của các cơ cấu :

* **Cơ cấu kẹp trụ:** 

***Hình 3.2: Sơ đồ tính hành trình xylanh kẹp trụ***

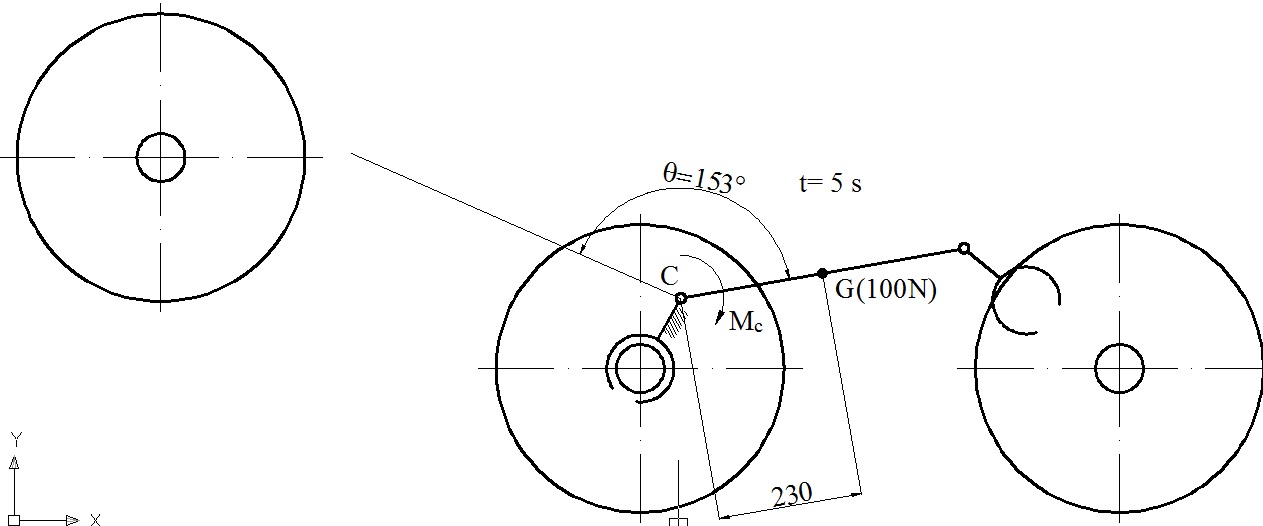
* Mỗi tay kẹp trụ được chế tạo dùng 4 xilanh mỗi má kẹp được đẩy bằng 2 pittông.
* Hành trình xilanh L= 30 mm
* Đường kính xilanh D= 16 mm
* Áp suất của bình khí : P = 7 bar = 0,7 N/mm2
* Ta có lực kẹp xilanh: 
* Momen kẹp trụ mỗi má kẹp tác dụng: 
* Lực kẹp do mỗi má kẹp tác dụng: 
* 

***Hình 3.3 Sơ đồ tính toán lực kẹp***

* Tay kẹp dùng 2 má kẹp nên sẽ có 2 lực F, trên thân robot có sử dụng 1 xilanh ∅16 tác dụng lực F tại B.
* Điều kiện để robot bám chắc tại vị trí đã kẹp mà không bị rơi là momen kẹp phải lớn hơn momen do trọng lượng robot sinh ra tại điểm A. Cụ thể :
  + - * + 2F.25 + F.130 > 250.100 (1)
* Với F = 196 N thì điều kiện (1) thỏa mãn.
* **Bộ phận xoay cột:**

***Hình 3.6: Cơ cấu quay tay kẹp.***

* Hai tay kẹp robot quay được nhờ hai động cơ gạt nước ôtô.
* Hai cánh tay robot vương ra nhờ hai pittông có hành trình L= 200mm.
* Dựa vào kích thước của của các trụ và khoảng cách giữa các trụ ta tính toán thiết kế như hình 3.4.



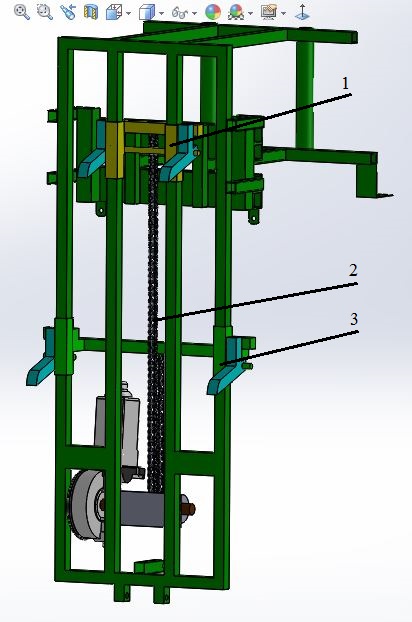
***Hình 3.4: Sơ đồ tính công suất động cơ xoay.***

* Hành trình góc quay : 153o
* Động cơ : Động cơ led
* Nguồn chạy của động cơ : 24v
* Ta có công suất động cơ:



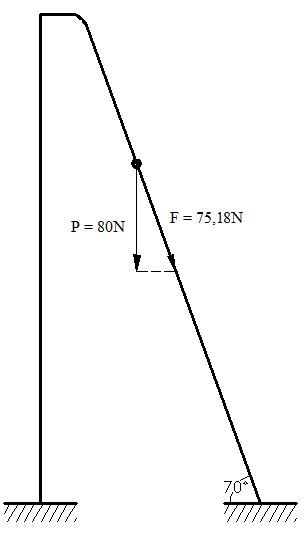
Ta chọn động cơ gạt nước ôtô loại nhỏ để chế tạo cơ cấu quay tay kẹp của robot.

* **Cơ cấu leo thang:**

****

***Hình 3.6: Cơ cấu leo cầu thang.***

* Cơ cấu leo cầu thang bằng cơ cấu trượt và lẫy để móc giữ robot trên bậc thang. Cơ cấu trượt gồm bốn thanh inox đặt song song nhau có nhiệm vụ trượt trên các bậc thang và các ống inox được lồng vào các thanh inox trên có nhiệm vụ mang theo các lẫy móc vào các bậc thang kéo robot đi lên. Các cơ cấu chuyển động tương đối với nhau bằng động cơ gạt nước ô tô thông qua bộ truyền xích và cơ cấu dây đai ròng rọc.
* Cơ cấu trượt trong **(1)** và cơ cấu trượt ngoài **(3)** chuyển động tương đối với nhau thông qua bộ tuyền xích **(2)**.

****

***Hình 3.7: Sơ đồ tính toán lực leo cầu thang***

( hệ số ma sát = 0,2)

Công suất động cơ cần chọn:

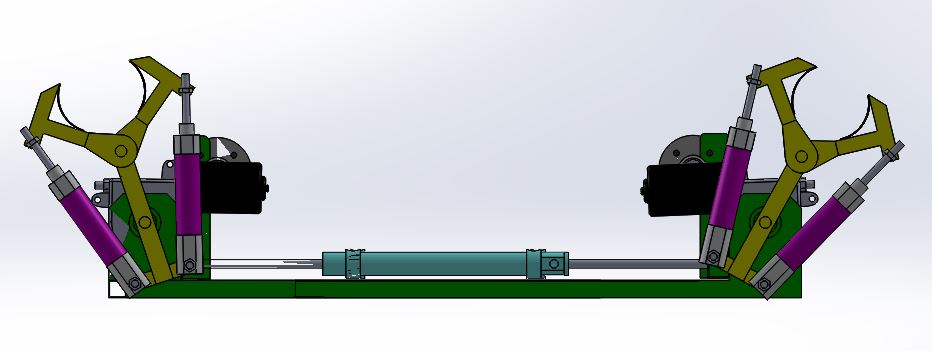
(vận tốc leo cầu thang 0,2m/s)

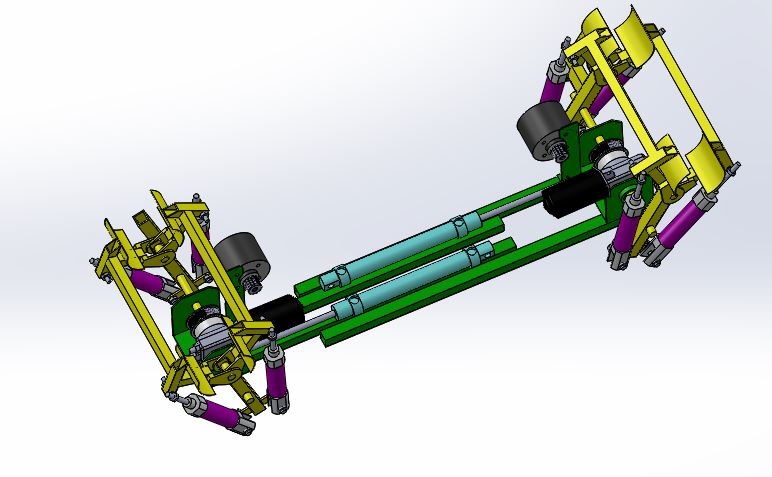


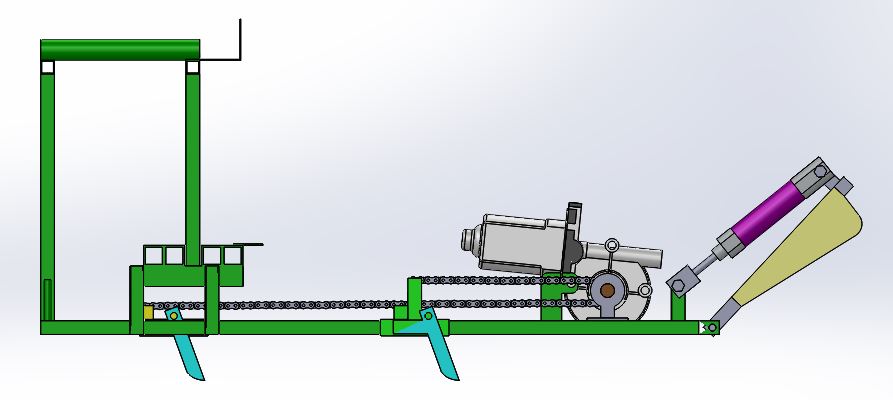
Ta chọn động cơ lép cỡ vừa và bộ truyền xích để chế tạo cơ cấu leo thang.

# 2.4 Hình ảnh robot tự động thiết kế bằng phần mềm Solidworks 2010

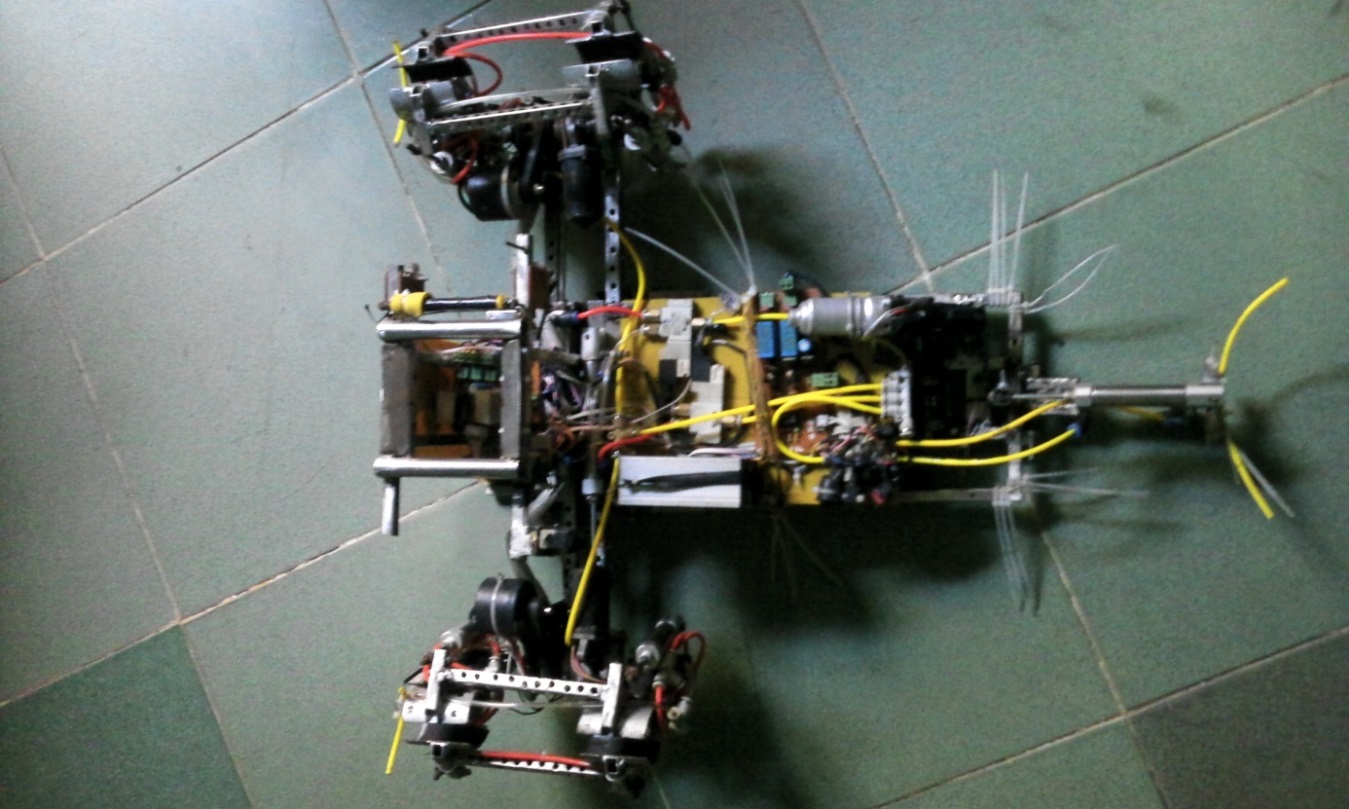


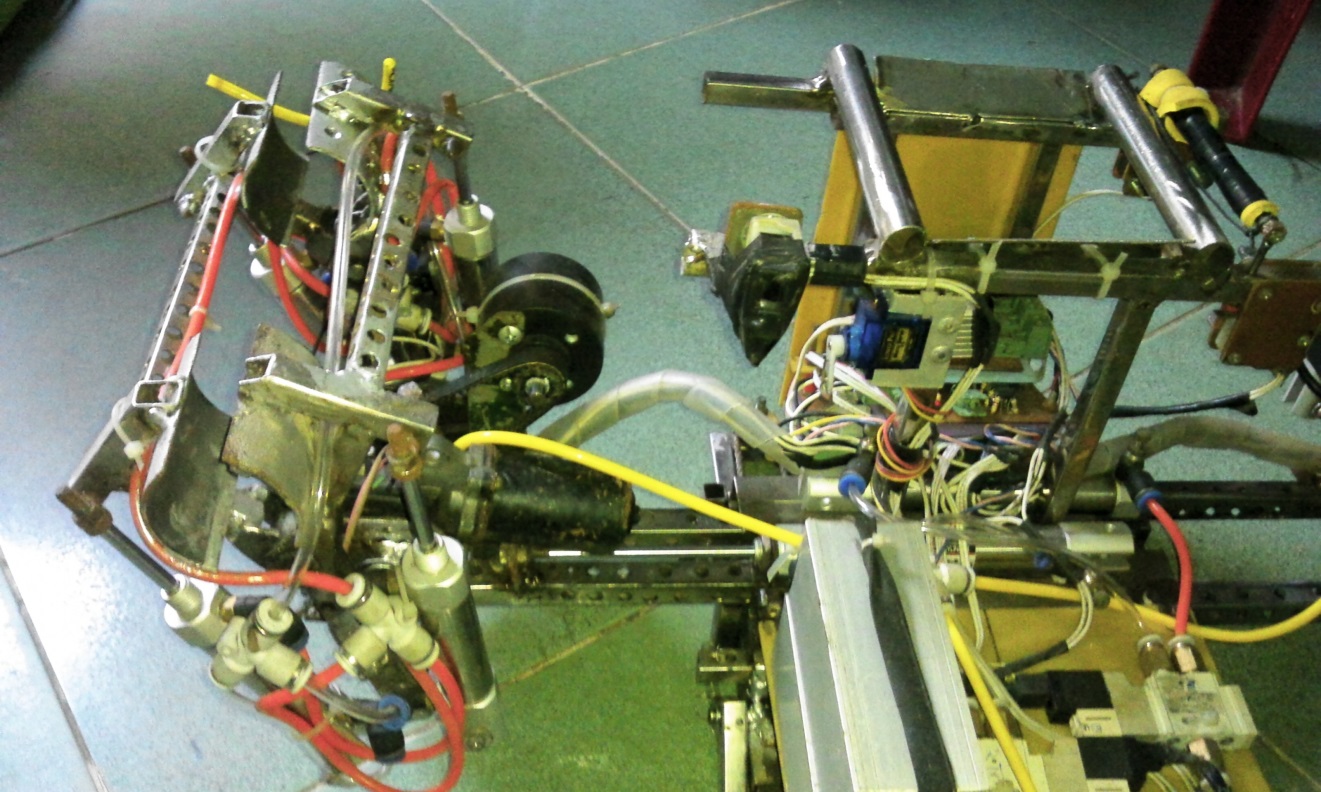
****

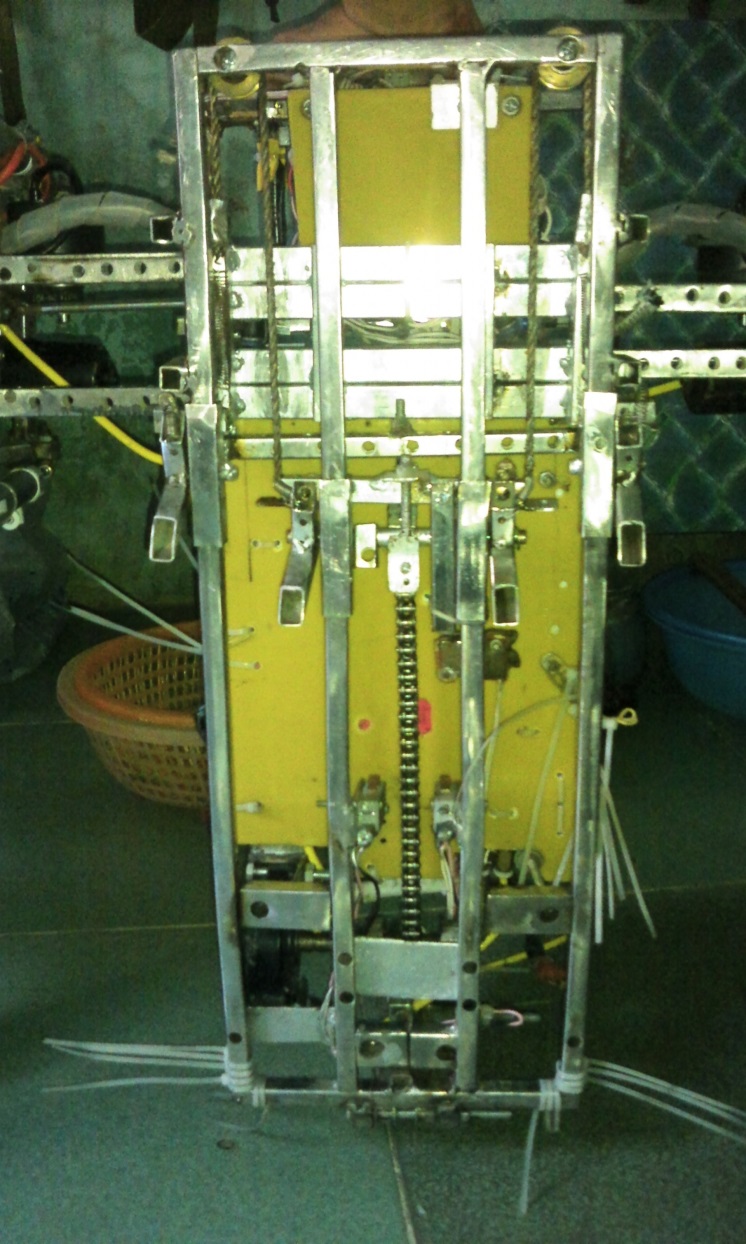
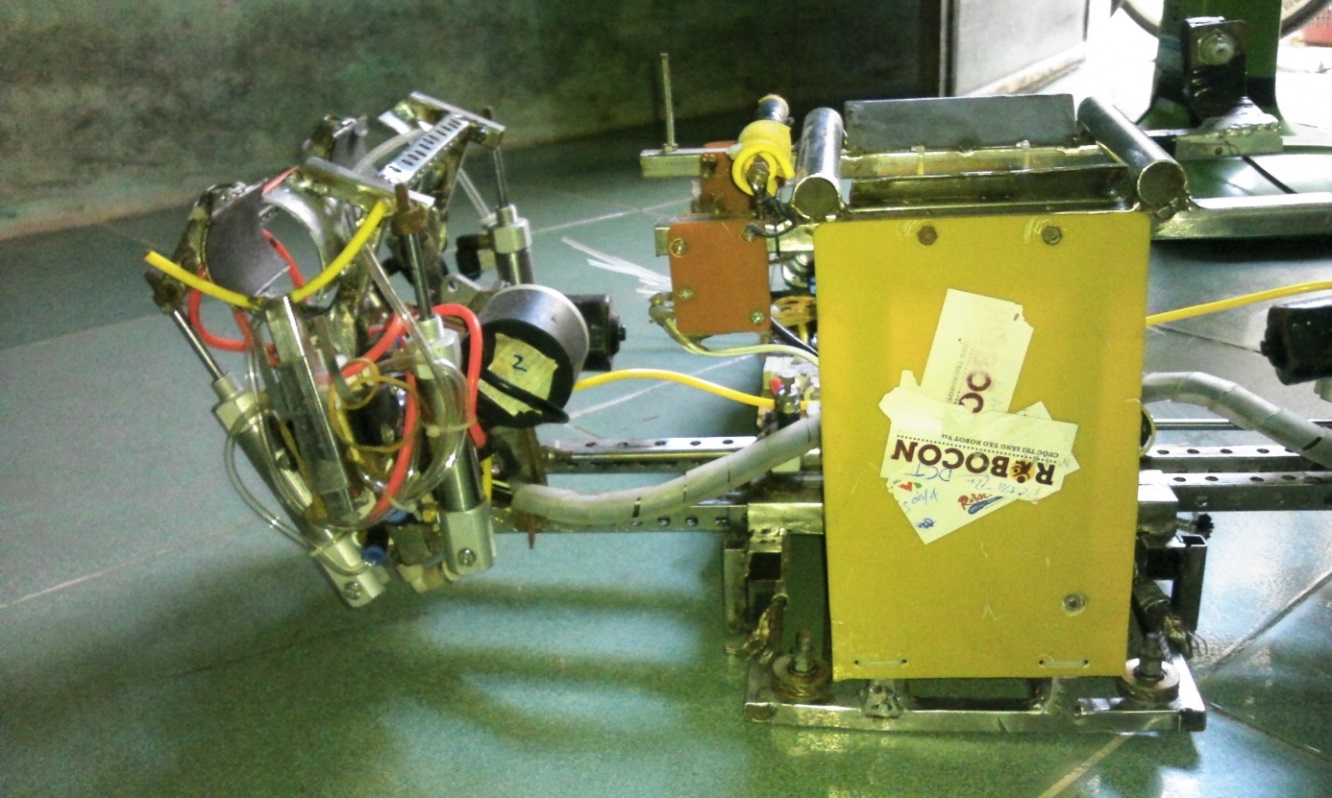
****

****

**2.5 Hình ảnh robot tự động đã thi công.**







**CHƯƠNG 3:**

**THIẾT KẾ MẠCH ĐIỀU KHIỂN ROBOT CON**

**3.1 Hệ thống điều khiển của Robot con:**

**3.1.1 Thống kê các tín hiệu vào ra****:**

**Bảng 3.1. Các tín hiệu vào của hệ thống điều khiển Robot con:**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên gọi** | **Số lượng tín hiệu** | **Chức năng** |
| 1 | Công tắc hành trình | 2 | Xác định góc ban đầu của hai tay kẹp |
| 2 | Công tắc hành trình | 2 | Xác định vị trí cột |
| 3 | Công tắc hành trình | 2 | Xác định cử hành trình của cơ cấu leo thang |
| 4 | Encorder | 4 | Xác định góc quay tay kẹp |
| 5 | Công tắc hành trình | 2 | Giao tiếp giữa robot mẹ và robot con |

Bảng 3.2. Các tín hiệu ra của hệ thống điều khiển Robot con:

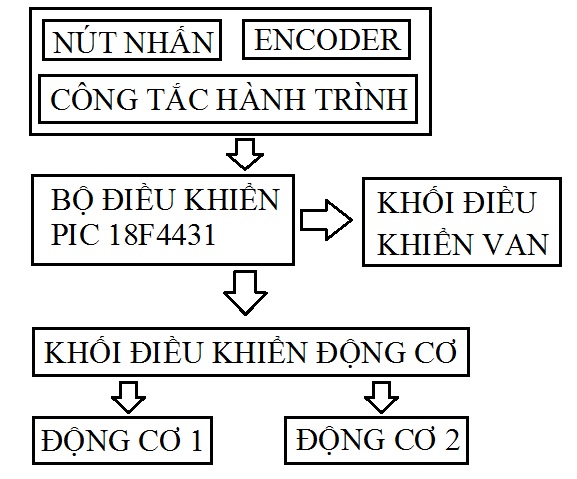
|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên gọi** | **Số lượng tín hiệu** | **Chức năng** |
| 1 | Tín hiệu điều khiển động cơ | 6 | Điều khiển động cơ quay tay kẹp và động cơ leo thang |
| 2 | Tín hiệu điều khiển Van khí nén | 5 | Điều khiển cơ cấu kẹp, cơ cấu vương cánh tay và đuôi của robot |

**Thống kê: có 12 tín hiệu vào và 11 tín hiệu ra.**

### 

### 3.1.2 Sơ đồ khối hệ thống điều khiển

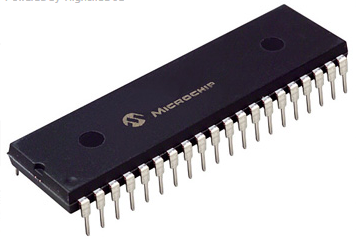
* **Phân tích thiết kế mạch**
* Các tín hiệu vào: bao gồm có 12 tín hiệu vào. Trong đó, các tín hiệu từ công tắc hành trình, nút nhấn là các tín hiệu xác định trạng thái mức cao hay thấp bao gồm 8 tín hiệu, tín hiệu từ encorder để xác định chiều và góc quay bao gồm 4 tín hiệu.
* Phần mạch công suất: bao gồm 1 mạch điều khiển 2 động cơ lép.
* Ngoài ra còn có module điều khiển van điện từ.
* Phần mạch điều khiển trung tâm: có nhiều phương án để lựa chọn bộ điều khiển, ở đây ta chọn vi điều khiển PIC18F4431 làm bộ điều khiển trung tâm. Họ này có nhiều tính năng vượt trội so với họ vi điều khiển 8051 thường dùng như tốc độ xử lý nhanh, có tích hợp ADC, Eeprom…
* **Sơ đồ khối mạch điều khiển**



***Hình 3.1 Sơ đồ khối mạch điều khiển Robot con.***

## Thiết kế và tính toán mạch:

****3.2.1 Tổng quan về vi điều khiển PIC:****

* ****Giới thiệu chung:****

**PIC là viết tắt của “Programble Intelligent Computer”, có thể tạm dịch là máy tính thông minh khả trình” do hãng Genenaral Instrument đặt tên cho vi điều khiển đầu tiên của họ: PIC1650 được thiết kế để dùng làm các thiết bị ngoại vi cho vi điều khiển CP1600. Vi điều khiển này sau đó được nghiên cứu phát triển thêm và từ đó hình thành nên dòng vi điều khiển PIC ngày nay.**

**Vì sao phải dùng PIC:**

**Họ vi điều khiển này có thể tìm mua dễ dàng tại thị trường.**

**Giá thành không quá đắt.**

**Có đầy đủ các tính năng của một vi điều khiển khi hoạt động độc lập.**

**Là một sự bổ sung rất tốt về kiến thức cũng như về ứng dụng cho vi điều khiển mang tính truyền thống.**

**PIC được sử dụng khá rộng rãi, điều này tạo nhiều thuận lợi cho quá trình tìm hiểu và phát triển các ứng dụng.**

**Sự hổ trợ của nhà sản xuất về trình biên dịch, các công cụ lập trình, nạp chương trình từ đơn giản đến phức tạp,…Các tính năng đa dạng của PIC không ngừng được phát triển.**

**Các dòng PIC được lựa chọn:**

**PIC12xxxx: độ dài lệnh 12 bit**

**PIC16xxxx: độ dài lệnh 14 bit**

**PIC18xxxx: độ dài lệnh 16 bit**

**Ngoài ra còn có thêm một dòng vi điều khiển PIC mới là dsPIC**

* + ****Cấu trúc của PIC****

**ở đây em dùng PIC18F4431 do hãng Microchip sản xuất có 14kB Flash và 768bytes bộ nhớ dữ liệu RAM, 40 chân.**

**+Là CPU sử dụng tập lệnh RISC và có tốc độ xử lý cao, công suất thấp nhờ sử dụng công nghệ CMOS FLASH/EEPROM.**

**+Một chu kỳ lệnh bằng 4 chu kỳ xung. Sử dụng bộ dao động 20Mhz thì chu kỳ lệnh 0,1 us.**

**+Tần số bộ dao động cho phép tới 20Mhz.**

**+8K x 14 word bộ nhớ FLASH lập trình.**

**+768 byte bộ nhớ RAM, trong đó bộ nhớ EEPROM lên đến 256 byte.**

**+Trang bị tới 34 ngắt với 8 cấp độ ngắt.**

**+5 port I/O. ( port A,B,C,D,E)**

**+Trang bị 3 bộ định thời: 2 bộ 8 bit, 1 bộ 16 bit.**

**+2 module Capture/Compare/PWN.**

**+Bộ chuyển đổi 10 bit ADC với tốc độ 5-10us.**

**+Cổng serial đồng bộ với chế độ SPI(Master) và 12C (Master/Slave) thức hiện bằng phần cứng.**

**+Chế độ chuyển nhận đồng bộ/bất đồng bộ với 9 bit địa chỉ kiểm tra.**

**+Cổng song song (PSP) 8bit.**

**+Các chế độ định địa chỉ: trực tiếp, gián tiếp và tương đối.**

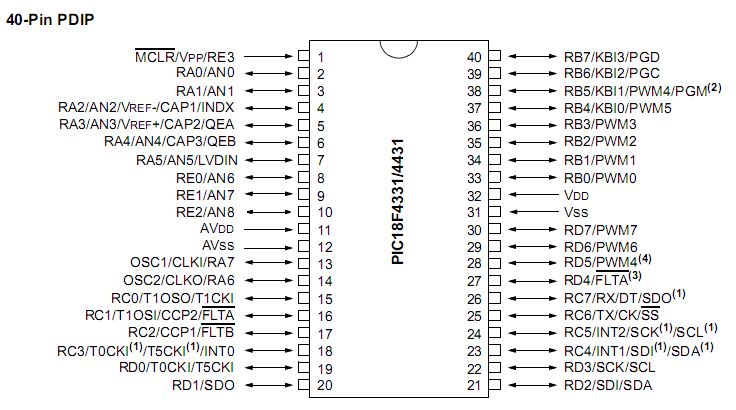
**+Cho phép đọc ghi bộ nhớ trương trình.**

**+2 chân cho phép gở rối hoạt động của vi điều khiển.**

**+Lập trình qua cổng serial với điện thế 5 V.**

**+Tầm điện thế hoạt động: từ 2-5.5 V. Dòng cấp khoảng 25mA.**

****Sơ đồ chân****



**Đây là vi điều khiển thuộc họ PIC18Fxxx với tập lệnh gồm 35 lệnh có độ dài 14 bit. Mỗi lệnh đều thực hiện được thực thi trong một chu kì xung clock. Tốc độ hoạt động tối đa cho phép là 20MHz với một chu kì lệnh là 200ns. Bộ nhớ chương trình 8Kx14 bit, bộ nhớ dữ liệu EEPROM với dung lượng 256x8 byte. Số PORT I/O là 5 với 33 pin I/O.**

**Các đặc tính ngoại vi bao gồm các khối chức năng sau:**

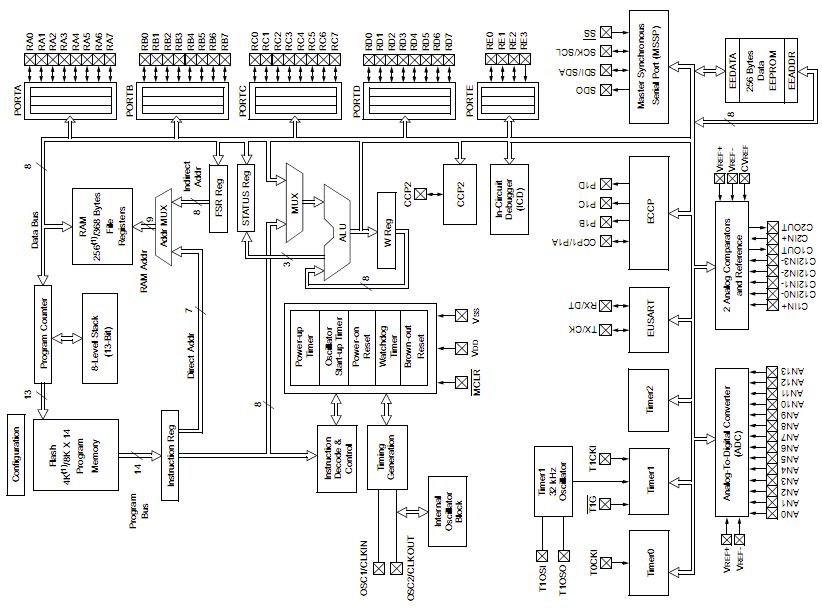
**Timer0: bộ đếm 8 bit với bộ chia tần số 8 bit.**

**Timer1: bộ đếm 16 bit với bộ chia tần số, có thể thực hiện chức năng đếm dựa vào xung clock ngoại vi ngay khi vi điều khiển hoạt động ở chế độ sleep.**

**Timer2: bộ đếm 8 bit với bộ chia tần số, bộ postcaler. Hai bộ Capture/so sánh/điều xung.**

**Các chuẩn giao tiếp nối tiếp SSP (synchronous Serial Port), SPI và I2C.**

**Các chuẩn giao tiếp nối tiếp USART với 9 địa chỉ.**

**Cổng giao tiếp song song PSP (parallel Slave Port) với các chân điều khiển RD, WR, CS ở bên ngoài.**

**Các đặc tính Analog:**

**8 kênh chuyển đổi ADC 10 bit.**

**Hai bộ so sánh.**

**Bên cạnh đó là một vài đặt tính khác của vi điều khiển như:**

**Bộ nhớ flash.**

**Bộ nhớ EEPROM.**

**Watchdog Timer với bộ dao động trong.**

**Chức năng bảo mật chương trình.**

**Chế độ sleep.**

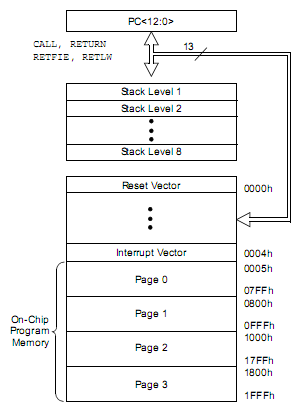
**Sơ đồ khối (hình bên) vi điều khiển PIC18F4431.**

Tổ chức bộ nhớ:

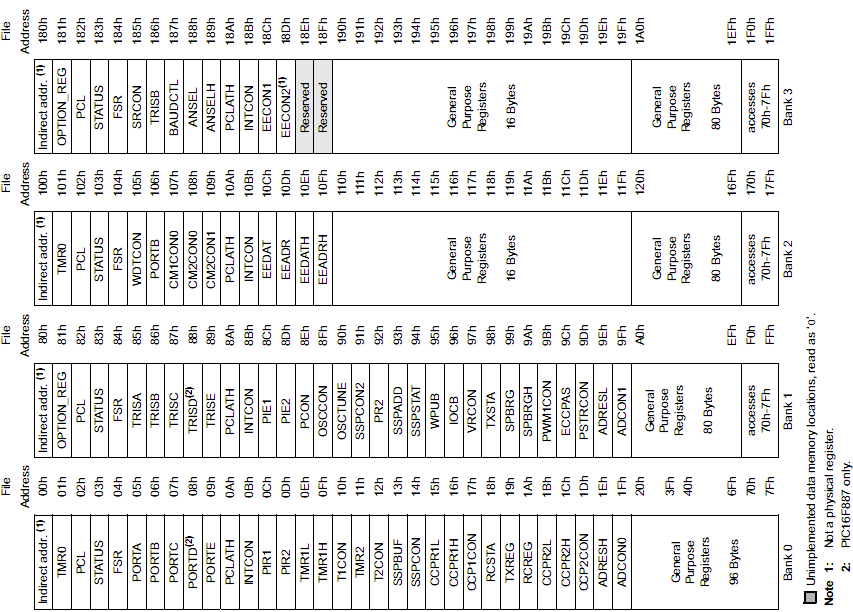
Cấu trúc bộ nhớ của vi điều khiển PIC18F4431 bao gồm bộ nhớ chương trinh(Program memory) và bộ nhớ dữ liệu (Data Memory).

Bộ nhớ chương trình:

Là bộ nhớ flash,lượng 8K word(1 word =14 bit) và được phân chia thành nhiều trang( từ page 0 đến page 3). Như vậy bộ nhớ có khả năng chứa được 8\*1024=8192 lệnh.

**

Bộ nhớ dữ liệu: là bộ nhớ EEPROM được chia làm nhiều bank. Với PIC16F887 bộ nhớ dữ liệu chia làm 4 bank.

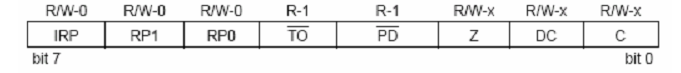
**

THANH GHI CHÖÙC NAÊNG ÑAËC BIEÄT SFR

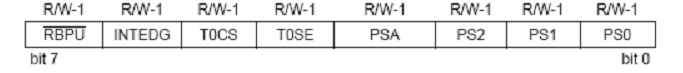
Ñaây laø caùc thanh ghi ñöôïc söû duïng bôûi CPU hoaëc ñöôïc duøng ñeå thieát laäp vaø ñieàu khieån caùc khoái chöùc naêng ñöôïc tích hôïp beân trong vi ñieàu khieån. Coù theå phaân thanh ghi SFR laøm hai loïai: thanh ghi SFR lieân quan ñeán caùc chöùc naêng beân trong (CPU) vaø thanh ghi SRF duøng ñeå thieát laäp vaø ñieàu khieån caùc khoái chöùc naêng beân ngoaøi (ví duï nhö ADC, PWM, …).

Thanh ghi STATUS (03h, 83h, 103h, 183h):thanh ghi chöùa keát quaû thöïc hieän pheùp

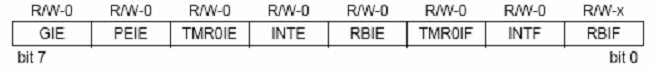
toaùn cuûa khoái ALU, traïng thaùi reset vaø caùc bit choïn bank caàn truy xuaát trong boä nhôù döõ lieäu.



Thanh ghi OPTION\_REG (81h, 181h): thanh ghi naøy cho pheùp ñoïc vaø ghi, cho pheùp ñieàu khieån chöùc naêng pull-up cuûa caùc chaân trong PORTB, xaùc laäp caùc tham soá veà xung taùc ñoäng, caïnh taùc ñoäng cuûa ngaét ngoaïi vi vaø boä ñeám Timer0.



Thanh ghi INTCON (0Bh, 8Bh,10Bh, 18Bh):thanh ghi cho pheùp ñoïc vaø ghi, chöùa caùc bit ñieàu khieån vaø caùc bit côø hieäu khi timer0 bò traøn, ngaét ngoaïi vi RB0/INT vaø ngaét interrput-on-change taïi caùc chaân cuûa PORTB.



Thanh ghi PIE1 (8Ch): chöùa caùc bit ñieàu khieån chi tieát caùc ngaét cuûa caùc khoái chöùc naêng ngoaïi vi.

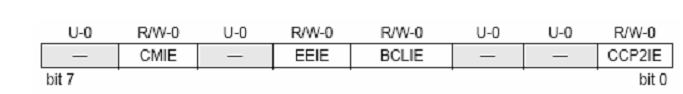


Thanh ghi PIR1 (0Ch) chöùa côø ngaét cuûa caùc khoái chöùc naêng ngoaïi vi, caùc ngaét naøy ñöôïc cho pheùp bôûi caùc bit ñieàu khieån chöùa trong thanh ghi PIE1.

**

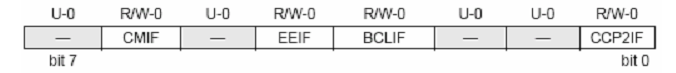
Thanh ghi PIE2 (8Dh): chöùa caùc bit ñieàu khieån caùc ngaét cuûa caùc khoái chöùc naêng

CCP2, SSP bus, ngaét cuûa boä so saùnh vaø ngaét ghi vaøo boä nhôù EEPROM.



Thanh ghi PIR2 (0Dh): chöùa caùc côø ngaét cuûa caùc khoái chöùc naêng ngoaïi vi, caùc ngaét

naøy ñöôïc cho pheùp bôûi caùc bit ñieàu khieån chöùa trong thanh ghi PIE2.



Thanh ghi PCON (8Eh): chöùa caùc côø hieäu cho bieát traïng thaùi caùc cheá ñoä reset cuûa vi ñieàu khieån.



**CAÙC COÅNG XUAÁT NHAÄP CUÛA PIC18F4431**

Coång xuaát nhaäp (I/O port) chính laø phöông tieän maø vi ñieàu khieån duøng ñeå töông taùc vôùi theá giôùi beân ngoaøi. Söï töông taùc naøy raát ña daïng vaø thoâng qua quaù trình töông taùc ñoù, chöùc naêng cuûa vi ñieàu khieån ñöôïc theå hieän moät caùch roõ raøng.

Moät coång xuaát nhaäp cuûa vi ñieàu khieån bao goàm nhieàu chaân (I/O pin), tuøy theo caùch boá trí vaø chöùc naêng cuûa vi ñieàu khieån maø soá löôïng coång xuaát nhaäp vaø soá löôïng chaân trong moãi coång coù theå khaùc nhau. Beân caïnh ñoù, do vi ñieàu khieån ñöôïc tích hôïp saün beân trong caùc ñaëc tính giao tieáp ngoaïi vi neân beân caïnh chöùc naêng laø coång xuaát nhaäp thoâng thöôøng, moät soá chaân xuaát nhaäp coøn coù theâm caùc chöùc naêng khaùc ñeå theå hieän söï taùc ñoäng cuûa caùc ñaëc tính ngoaïi vi neâu treân ñoái vôùi theá giôùi beân ngoaøi. Chöùc naêng cuûa töøng chaân xuaát nhaäp trong moãi coång hoaøn toaøn coù theå ñöôïc xaùc laäp vaø ñieàu khieån ñöôïc thoâng qua caùc thanh ghi SFR lieân quan ñeán chaân xuaát nhaäp ñoù.

Vi ñieàu khieån PIC16F887 coù 5 coång xuaát nhaäp, bao goàm PORTA, PORTB, PORTC, PORTD vaø PORTE.

**Caáu truùc vaø chöùc naêng cuûa töøng coång xuaát nhaäp :**

PORTA

PORTA (RPA) bao goàm 6 I/O pin. Ñaây laø caùc chaân “hai chieàu” (bidirectional pin), nghóa laø coù theå xuaát vaø nhaäp ñöôïc. Chöùc naêng I/O naøy ñöôïc ñieàu khieån bôûi thanh ghi TRISA (ñòa chæ 85h). Muoán xaùc laäp chöùc naêng cuûa moät chaân trong PORTA laø input, ta “set” bit ñieàu khieån töông öùng vôùi chaân ñoù trong thanh ghi TRISA vaø ngöôïc laïi, muoán xaùc laäp chöùc naêng cuûa moät chaân trong PORTA laø output, ta “clear” bit ñieàu khieån töông öùng vôùi chaân ñoù trong thanh ghi TRISA. Thao taùc naøy hoaøn toaøn töông töï ñoái vôùi caùc PORT vaø caùc thanh ghi ñieàu khieån töông öùng TRIS (ñoái vôùi PORTA laø TRISA, ñoái vôùi PORTB laø TRISB, ñoái vôùi PORTC laø TRISC, ñoái vôùi PORTD laø TRISD vaøñoái vôùi PORTE laø TRISE). Beân caïnh ñoù PORTA coøn laø ngoõ ra cuûa boä ADC, boä so saùnh, ngoõ vaøo analog ngoõ vaøo xung clock cuûa Timer0 vaø ngoõ vaøo cuûa boä giao tieáp MSSP (Master Synchronous Serial Port).

Caùc thanh ghi SFR lieân quan ñeán PORTA bao goàm:

PORTA (ñòa chæ 05h) : chöùa giaù trò caùc pin trong PORTA.

TRISA (ñòa chæ 85h) : ñieàu khieån xuaát nhaäp.

CMCON (ñòa chæ 9Ch) : thanh ghi ñieàu khieån boä so saùnh.

CVRCON (ñòa chæ 9Dh) : thanh ghi ñieàu khieån boä so saùnh ñieän aùp.

ADCON1 (ñòa chæ 9Fh) : thanh ghi ñieàu khieån boä ADC.

PORTB

PORTB (RPB) goàm 8 pin I/O. Thanh ghi ñieàu khieån xuaát nhaäp töông öùng laø TRISB. Beân caïnh ñoù moät soá chaân cuûa PORTB coøn ñöôïc söû duïng trong quaù trình naïp chöông trình cho vi ñieàu khieån vôùi caùc cheá ñoä naïp khaùc nhau. PORTB coøn lieân quan ñeán ngaét ngoaïi vi vaø boä Timer0. PORTB coøn ñöôïc tích hôïp chöùc naêng ñieän trôû keùo leân ñöôïc ñieàu khieån bôûi chöông trình.

Caùc thanh ghi SFR lieân quan ñeán PORTB bao goàm:

PORTB (ñòa chæ 06h,106h) : chöùa giaù trò caùc pin trong PORTB

TRISB (ñòa chæ 86h,186h) : ñieàu khieån xuaát nhaäp

OPTION\_REG (ñòa chæ 81h,181h) : ñieàu khieån ngaét ngoaïi vi vaø boä Timer0.

PORTC

PORTC (RPC) goàm 8 pin I/O. Thanh ghi ñieàu khieån xuaát nhaäp töông öùng laø TRISC. Beân caïnh ñoù PORTC coøn chöùa caùc chaân chöùc naêng cuûa boä so saùnh, boä Timer1, boä PWM vaø caùc chuaån giao tieáp noái tieáp I2C, SPI, SSP, USART.

Caùc thanh ghi ñieàu khieån lieân quan ñeán PORTC:

PORTC (ñòa chæ 07h) : chöùa giaù trò caùc pin trong PORTC

TRISC (ñòa chæ 87h) : ñieàu khieån xuaát nhaäp.

PORTD

PORTD (RPD) goàm 8 chaân I/O, thanh ghi ñieàu khieån xuaát nhaäp töông öùng laø TRISD. PORTD coøn laø coång xuaát döõ lieäu cuûa chuaån giao tieáp PSP (Parallel Slave Port).

Caùc thanh ghi lieân quan ñeán PORTD bao goàm:

Thanh ghi PORTD : chöùa giaù trò caùc pin trong PORTD.

Thanh ghi TRISD : ñieàu khieån xuaát nhaäp.

Thanh ghi TRISE : ñieàu khieån xuaát nhaäp PORTE vaø chuaån giao tieáp PSP.

PORTE

PORTE (RPE) goàm 3 chaân I/O. Thanh ghi ñieàu khieån xuaát nhaäp töông öùng laø TRISE. Caùc chaân cuûa PORTE coù ngoõ vaøo analog. Beân caïnh ñoù PORTE coøn laø caùc chaân ñieàu khieån cuûa chuaån giao tieáp PSP.

Caùc thanh ghi lieân quan ñeán PORTE bao goàm:

PORTE : chöùa giaù trò caùc chaân trong PORTE.

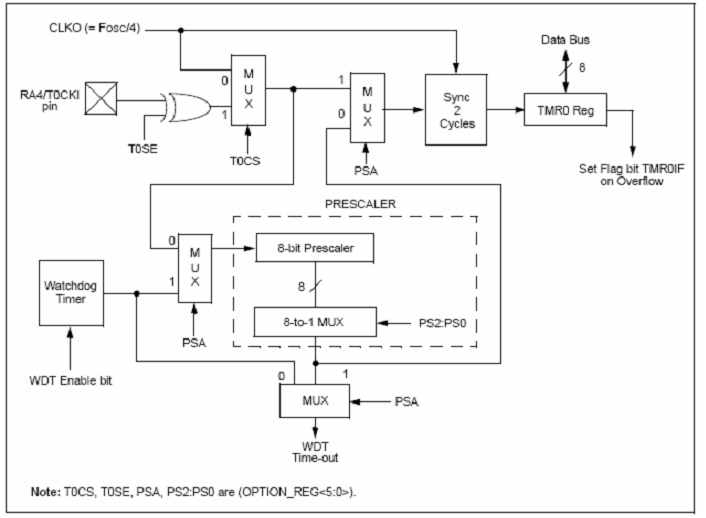
TRISE : ñieàu khieån xuaát nhaäp vaø xaùc laäp caùc thoâng soá cho chuaån giao tieáp PSP.

ADCON1 : thanh ghi ñieàu khieån khoái ADC.

TIMER 0

Ñaây laø moät trong ba boä ñeám hoaëc boä ñònh thôøi cuûa vi ñieàu khieån PIC16F877A. Timer0 laø boä ñeám 8 bit ñöôïc keát noái vôùi boä chia taàn soá (prescaler) 8 bit. Caáu truùc cuûa Timer0 cho pheùp ta löïa choïn xung clock taùc ñoäng vaø caïnh tích cöïc cuûa xung clock. Ngaét Timer0 seõ xuaát hieän khi Timer0 bò traøn. Bit TMR0IE (INTCON<5>) laø bit ñieàu khieån cuûa Timer0. TMR0IE=1 cho pheùp ngaét Timer0 taùc ñoäng, TMR0IF= 0 khoâng cho pheùp ngaét Timer0 taùc ñoäng.

Sô ñoà khoái cuûa Timer0 nhö sau:



Muoán Timer0 hoaït ñoäng ôû cheá ñoä Timer ta clear bit TOSC (OPTION\_REG<5>), khi ñoù giaù trò thanh ghi TMR0 seõ taêng theo töøng chu kì xung ñoàng hoà (taàn soá vaøo Timer0 baèng ¼ taàn soá oscillator). Khi giaù trò thanh ghi TMR0 töø FFh trôû veà 00h, ngaét Timer0 seõ xuaát hieän.

Thanh ghi TMR0 cho pheùp ghi vaø xoùa ñöôïc giuùp ta aán ñònh thôøi ñieåm ngaét Timer0 xuaát hieän moät caùch linh ñoäng.

Muoán Timer0 hoaït ñoäng ôû cheá ñoä counter ta set bit TOSC (OPTION\_REG<5>). Khi ñoù xung taùc ñoäng leân boä ñeám ñöôïc laáy töø chaân RA4/TOCK1. Bit TOSE (OPTION\_REG<4>) cho pheùp löïa choïn caïnh taùc ñoäng vaøo boät ñeám. Caïnh taùc ñoäng seõ laø caïnh leân neáu TOSE=0 vaø caïnh taùc ñoäng seõ laø caïnh xuoáng neáu TOSE=1.

Khi thanh ghi TMR0 bò traøn, bit TMR0IF (INTCON<2>) seõ ñöôïc set. Ñaây chính laø côø ngaét cuûa Timer0. Côø ngaét naøy phaûi ñöôïc xoùa baèng chöông trình tröôùc khi boä ñeám baét ñaàu thöïc hieän laïi quaù trình ñeám. Ngaét Timer0 khoâng theå “ñaùnh thöùc” vi ñieàu khieån töø cheá ñoä sleep.

Boä chia taàn soá (prescaler) ñöôïc chia seû giöõa Timer0 vaø WDT (Watchdog Timer). Ñieàu ñoù coù nghóa laø neáu prescaler ñöôïc söû duïng cho Timer0 thì WDT seõ khoâng coù ñöôïc hoã trôï cuûa prescaler vaø ngöôïc laïi. Prescaler ñöôïc ñieàu khieån bôûi thanh ghi OPTION\_REG. Bit PSA (OPTION\_REG<3>) xaùc ñònh ñoái töôïng taùc ñoäng cuûa prescaler. Caùc bit PS2:PS0 (OPTION\_REG<2:0>) xaùc ñònh tæ soá chia taàn soá cuûa prescaler. Xem laïi thanh ghi OPTION\_REG ñeå xaùc ñònh laïi moät caùch chi tieát veà caùc bit ñieàu khieån treân.

Caùc leänh taùc ñoäng leân giaù trò thanh ghi TMR0 seõ xoùa cheá ñoä hoaït ñoäng cuûa prescaler.

Khi ñoái töôïng taùc ñoäng laø Timer0, taùc ñoäng leân giaù trò thanh ghi TMR0 seõ xoùa prescaler nhöng khoâng laøm thay ñoåi ñoái töôïng taùc ñoäng cuûa prescaler. Khi ñoái töôïng taùc ñoäng laø WDT, leänh CLRWDT seõ xoùa prescaler, ñoàng thôøi prescaler seõ ngöng taùc vuï hoã trôï cho WDT.

Caùc thanh ghi ñieàu khieån lieân quan ñeán Timer0 bao goàm:

TMR0 (ñòa chæ 01h, 101h) : chöùa giaù trò ñeám cuûa Timer0.

INTCON (ñòa chæ 0Bh, 8Bh, 10Bh, 18Bh): cho pheùp ngaét hoaït ñoäng (GIE vaø PEIE).

OPTION\_REG (ñòa chæ 81h, 181h): ñieàu khieån prescaler.

TIMER1

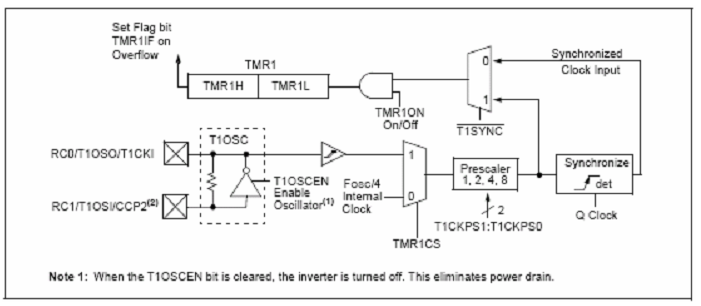
Timer1 laø boä ñònh thôøi 16 bit, giaù trò cuûa Timer1 seõ ñöôïc löu trong hai thanh ghi

(TMR1H:TMR1L). Côø ngaét cuûa Timer1 laø bit TMR1IF (PIR1<0>). Bit ñieàu khieån cuûa

Timer1 seõ laø TMR1IE (PIE<0>).

Töông töï nhö Timer0, Timer1 cuõng coù hai cheá ñoä hoaït ñoäng: cheá ñoä ñònh thôøi (timer) vôùi xung kích laø xung clock cuûa oscillator (taàn soá cuûa timer baèng ¼ taàn soá cuûa oscillator) vaø cheá ñoä ñeám (counter) vôùi xung kích laø xung phaûn aùnh caùc söï kieän caàn ñeám laáy töø beân ngoaøi thoâng qua chaân RC0/T1OSO/T1CKI (caïnh taùc ñoäng laø caïnh leân). Vieäc löïa choïn xung taùc ñoäng (töông öùng vôùi vieäc löïa choïn cheá ñoä hoaït ñoäng laø timer hay counter) ñöôïc ñieàu khieån bôûi bit TMR1CS (T1CON<1>).

Sô ñoà khoái cuûa Timer1:



Ngoaøi ra Timer1 coøn coù chöùc naêng reset input beân trong ñöôïc ñieàu khieån bôûi moät trong hai khoái CCP (Capture/Compare/PWM).

Khi bit T1OSCEN (T1CON<3>) ñöôïc set, Timer1 seõ laáy xung clock töø hai chaân RC1/T1OSI/CCP2 vaø RC0/T1OSO/T1CKI laøm xung ñeám. Timer1 seõ baét ñaàu ñeám sau caïnh xuoáng ñaàu tieân cuûa xung ngoõ vaøo. Khi ñoù PORTC seõ boû qua söï taùc ñoäng cuûa hai bit TRISC<1:0> vaø PORTC<2:1> ñöôïc gaùn giaù trò 0. Khi clear bit T1OSCEN Timer1 seõ laáy xung ñeám töø oscillator hoaëc töø chaân RC0/T1OSO/T1CKI. Timer1 coù hai cheá ñoä ñeám laø ñoàng boä (Synchronous) vaø baát ñoàng boä (Asynchronous).

Cheá ñoä ñeám ñöôïc quyeát ñònh bôûi bit ñieàu khieån (T1CON<2>).

Khi =1 xung ñeám laáy töø beân ngoaøi seõ khoâng ñöôïc ñoàng boä hoùa vôùi xung clock beân trong, Timer1 seõ tieáp tuïc quaù trình ñeám khi vi ñieàu khieån ñang ôû cheá ñoä sleep vaø ngaét do Timer1 taïo ra khi bò traøn coù khaû naêng “ñaùnh thöùc” vi ñieàu khieån. ÔÛ cheá ñoä ñeám baát ñoàng boä, Timer1 khoâng theå ñöôïc söû duïng ñeå laøm nguoàn xung clock cho khoái CCP (Capture/Compare/Pulse width modulation).

Khi =0 xung ñeám vaøo Timer1 seõ ñöôïc ñoàng boä hoùa vôùi xung clock beân trong. ÔÛ

cheá ñoä naøy Timer1 seõ khoâng hoaït ñoäng khi vi ñieàu khieån ñang ôû cheá ñoä sleep.

Caùc thanh ghi lieân quan ñeán Timer1 bao goàm:

INTCON (ñòa chæ 0Bh, 8Bh, 10Bh, 18Bh): cho pheùp ngaét hoaït ñoäng (GIE vaø PEIE).

PIR1 (ñòa chæ 0Ch): chöùa côø ngaét Timer1 (TMR1IF).

PIE1( ñòa chæ 8Ch): cho pheùp ngaét Timer1 (TMR1IE).

TMR1L (ñòa chæ 0Eh): chöùa giaù trò 8 bit thaáp cuûa boä ñeám Timer1.

TMR1H (ñòa chæ 0Eh): chöùa giaù trò 8 bit cao cuûa boä ñeám Timer1.

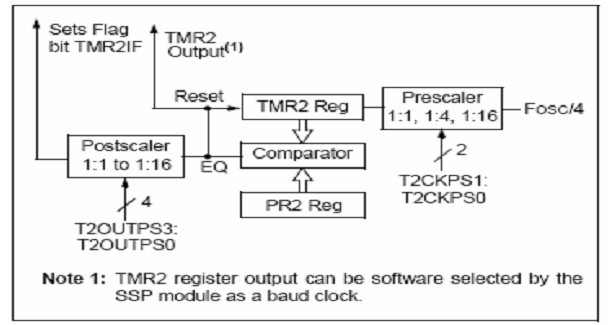
T1CON (ñòa chæ 10h): xaùc laäp caùc thoâng soá cho Timer1.

TIMER2

Timer2 laø boä ñònh thôøi 8 bit vaø ñöôïc hoã trôï bôûi hai boä chia taàn soá prescaler vaø

postscaler. Thanh ghi chöùa giaù trò ñeám cuûa Timer2 laø TMR2. Bit cho pheùp ngaét Timer2 taùc ñoäng laø TMR2ON (T2CON<2>). Côø ngaét cuûa Timer2 laø bit TMR2IF (PIR1<1>). Xung ngoõ vaøo (taàn soá baèng ¼ taàn soá oscillator) ñöôïc ñöa qua boä chia taàn soá prescaler 4 bit (vôùi caùc tæ soá chia taàn soá laø 1:1, 1:4 hoaëc 1:16 vaø ñöôïc ñieàu khieån bôûi caùc bit T2CKPS1:T2CKPS0 (T2CON<1:0>)).

Sô ñoà khoái Timer2:

**

Timer2 coøn ñöôïc hoã trôï bôûi thanh ghi PR2. Giaù trò ñeám trong thanh ghi TMR2 seõ taêng töø 00h ñeán giaù trò chöùa trong thanh ghi PR2, sau ñoù ñöôïc reset veà 00h. Kh I reset thanh ghi PR2 ñöôïc nhaän giaù trò maëc ñònh FFh. Ngoõ ra cuûa Timer2 ñöôïc ñöa qua boä chia taàn soá postscaler vôùi caùc möùc chia töø 1:1 ñeán 1:16. Postscaler ñöôïc ñieàu khieån bôûi 4 bit T2OUTPS3:T2OUTPS0. Ngoõ ra cuûa postscaler ñoùng vai troø quyeát ñònh trong vieäc ñieàu khieån côø ngaét.

Ngoaøi ra ngoõ ra cuûa Timer2 coøn ñöôïc keát noái vôùi khoái SSP, do ñoù Timer2 coøn ñoùng vai troø taïo ra xung clock ñoàng boä cho khoái giao tieáp SSP.

Caùc thanh ghi lieân quan ñeán Timer2 bao goàm:

INTCON (ñòa chæ 0Bh, 8Bh, 10Bh, 18Bh): cho pheùp toaøn boä caùc ngaét (GIE vaø PEIE).

PIR1 (ñòa chæ 0Ch): chöùa côø ngaét Timer2 (TMR2IF).

PIE1 (ñòa chò 8Ch): chöùa bit ñieàu khieån Timer2 (TMR2IE).

TMR2 (ñòa chæ 11h): chöùa giaù trò ñeám cuûa Timer2.

T2CON (ñòa chæ 12h): xaùc laäp caùc thoâng soá cho Timer2. PR2 (ñòa chæ 92h): thanh ghi hoã trôï cho Timer2.

Ta coù moät vaøi nhaän xeùt veà Timer0, Timer1 vaø Timer2 nhö sau:

Timer0 vaø Timer2 laø boä ñeám 8 bit (giaù trò ñeám toái ña laø FFh), trong khi Timer1 laø boä ñeám 16 bit (giaù trò ñeám toái ña laø FFFFh).

Timer0, Timer1 vaø Timer2 ñeàu coù hai cheá ñoä hoaït ñoäng laø timer vaø counter. Xung clock coù taàn soá baèng ¼ taàn soá cuûa oscillator. Xung taùc ñoäng leân Timer0 ñöôïc hoã trôï bôûi prescaler vaø coù theå ñöôïc thieát laäp ôû nhieàu cheá ñoä khaùc nhau (taàn soá taùc ñoäng, caïnh taùc ñoäng) trong khi caùc thoâng soá cuûa xung taùc ñoäng leân Timer1 laø coá ñònh. Timer2 ñöôïc hoã trôï bôûi hai boä chia taàn soá prescaler vaø postcaler ñoäc laäp, tuy nhieân caïnh taùc ñoäng vaãn ñöôïc coá ñònh laø caïnh leân. Timer1 coù quan heä vôùi khoái CCP, trong khi Timer2 ñöôïc keát noái vôùi khoái SSP.

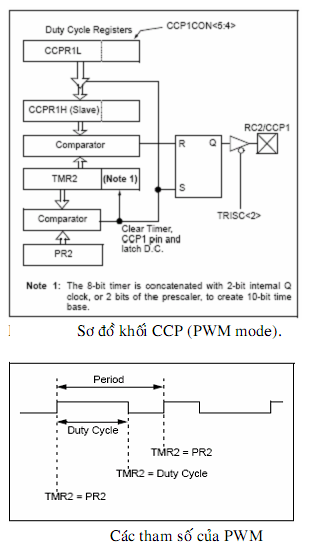
Keânh ñieàu xung CCP :

CCP (Capture/Compare/PWM) bao goàm caùc thao taùc treân caùc xung ñeám cung caáp bôûi caùc boä ñeám Timer1 vaø Timer2. PIC16F887 ñöôïc tích hôïp saün hai khoái CCP : CCP1 vaø CCP2.Moãi CCP coù moät thanh ghi 16 bit (CCPR1H:CCPR1L vaø CCPR2H:CCPR2L), pin ñieàu khieån duøng cho khoái CCPx laø RC2/CCP1 vaø RC1/T1OSI/CCP2. Caùc chöùc naêng cuûa CCP bao goàm:

Capture.

So saùnh (Compare).

Ñieàu cheá ñoä roäng xung PWM (Pulse Width Modulation).

Khi hoaït ñoäng ôû cheá ñoä PWM (Pulse

Width Modulation \_ khoái ñieàu cheá ñoä roäng

xung), tín hieäu sau khi ñieàu cheá seõ ñöôïc

ñöa ra caùc pin cuûa khoái CCP (caàn aán ñònh

caùc pin naøy laø output). Ñeå söû duïng chöùc

naêng ñieàu cheá naøy tröôùc tieân ta caàn tieán

haønh caùc böôùc caøi ñaët sau:

1. Thieát laäp thôøi gian cuûa 1 chu kì

cuûa xung ñieàu cheá cho PWM

(period) baèng caùch ñöa giaù trò

thích hôïp vaøo thanh ghi PR2.

2. Thieát laäp ñoä roäng xung caàn ñieàu

cheá (duty cycle) baèng caùch ñöa

giaù trò vaøo thanh ghi CCPRxL

vaø caùc bit CCP1CON<5:4>.

3. Ñieàu khieån caùc pin cuûa CCP laø

output baèng caùch clear caùc bit

töông öùng trong thanh ghi

TRISC.

4. Thieát laäp giaù trò boä chia taàn soá prescaler cuûa Timer2 vaø cho pheùp Timer2 hoaït ñoäng baèng caùch ñöa giaù trò thích hôïp vaøo thanh ghi T2CON.

5. Cho pheùp CCP hoaït ñoäng ôû cheá ñoä PWM.

Trong ñoù giaù trò 1 chu kì (period) cuûa xung ñieàu cheá ñöôïc tính baèng coâng thöùc:

PWM period = [(PR2)+1]\*4\*TOSC\*(giaù trò boä chia taàn soá cuûa TMR2).

Boä chia taàn soá prescaler cuûa Timer2 chæ coù theå nhaän caùc giaù trò 1,4 hoaëc 16. Khi giaù trò thanh ghi PR2 baèng vôùi giaù trò thanh ghi TMR2 thì quaù trình sau xaûy ra:

Thanh ghi TMR2 töï ñoäng ñöôïc xoùa.

Pin cuûa khoái CCP ñöôïc set.

Giaù trò thanh ghi CCPR1L (chöùa giaù trò aán ñònh ñoä roäng xung ñieàu cheá duty cycle)

ñöôïc ñöa vaøo thanh ghi CCPRxH.

Ñoä roäng cuûa xung ñieàu cheá (duty cycle) ñöôïc tính theo coâng thöùc:

PWM period = [(PR2)+1]\*4\*TOSC\*(giaù trò boä chia taàn soá cuûa TMR2).

PWM duty cycle = (CCPRxL:CCPxCON<5:4>)\*TOSC\*(giaù trò boä chia taàn soá TMR2)

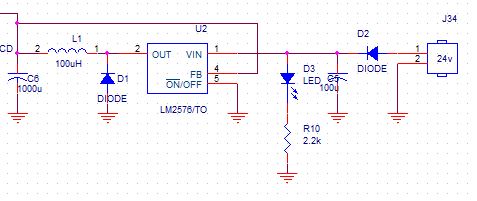
Nhö vaäy 2 bit CCPxCON<5:4> seõ chöùa 2 bit LSB. Thanh ghi CCPRxL chöùa byte cao cuûa giaù trò quyeát ñònh ñoä roäng xung. Thanh ghi CCPRxH ñoùng vai troø laø buffer cho khoái PWM. Khi giaù trò trong thanh ghi CCPRxH baèng vôùi giaù trò trong thanh ghi TMR2 vaø hai bit CCPxCON<5:4> baèng vôùi giaù trò 2 bit cuûa boä chia taàn soá prescaler, pin cuûa khoái CCP laïi ñöôïc ñöa veà möùc thaáp.

Moät soá ñieåm caàn chuù yù khi söû duïng khoái PWM:

Timer2 coù hai boä chia taàn soá prescaler vaø postscaler. Tuy nhieân boä postscaler khoâng ñöôïc söû duïng trong quaù trình ñieàu cheá ñoä roäng xung cuûa khoái PWM.

Neáu thôøi gian duty cycle daøi hôn thôøi gian chu kì xung period thì xung ngoõ ra tieáp tuïc ñöôïc giöõ ôû möùc cao sau khi giaù trò PR2 baèng vôùi giaù trò TMR2.

**3.2.2 Khối nguồn:**



***Hình 3.2 Sơ đồ nguyên lí mạch khối nguồn***

Dùng nguồn 7,5V - 35V vào IC ổn áp LM2576.Nguồn 5v cho vi điều khiển

- Tụ C5 lọc điện áp đầu vào

- Tụ C6 lọc điện áp ra

- IC LM2576 ổn áp cho ra 5v vào vi điều khiển.

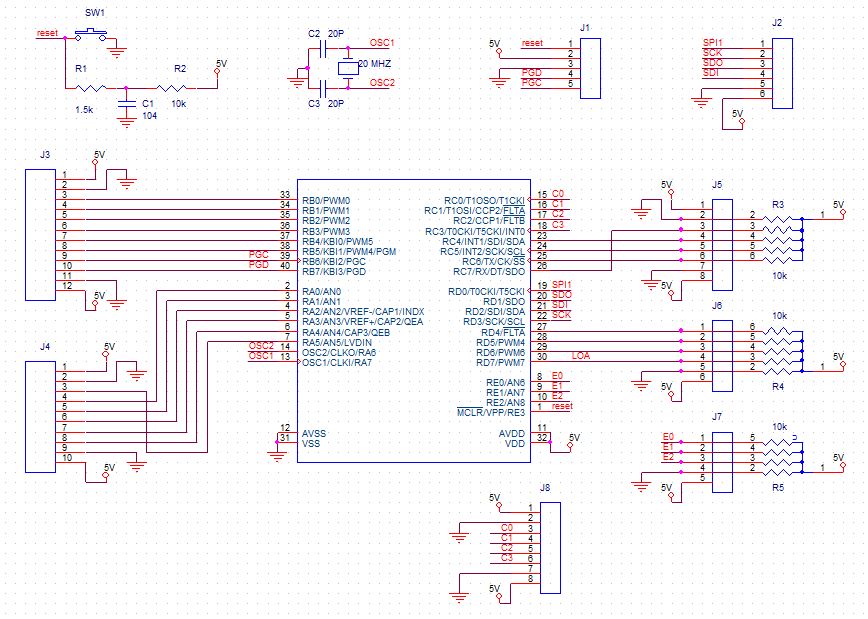
Led báo nguồn vi điều khiển có dòng định mức I = 10mA nên trở của đèn 330ohm

* + 1. **Khối điều khiển:**

Có nhiều phương án để chọn, ở đây chọn VĐK cho phần điều khiển trung tâm ta chọn họ chip VĐK là PIC 18F4431.

**Các tính năng mới của họ PIC:**

* 35 chân vào ra
* Sử dụng thạch anh 20 MHz
* Giao diện SPI đồng bộ.
* Các đường dẫn vào/ra (I/O) lập trình được.
* Các kênh băm xung PWM.
* Các chế độ tiết kiệm năng lượng như sleep, stand by..vv.
* Một bộ định thời Watchdog.
* 1 bộ Timer/Counter 8 bit.
* 3 bộ Timer/Counter 16 bit.
* Bộ nhớ EEPROM.
* Giao tiếp USART..vv.

****PIC 18F4431 có đầy đủ tính năng của họ PIC, về giá thành so với các loại khác thì giá thành là vừa phải khi nghiên cứu và làm các công việc ứng dụng tới vi điều khiển.

***Hình 3.3 Sơ đồ nguyên lí mạch điều khiển Robot con.***

* + 1. **Khối công suất cho hai động cơ quay cánh tay:**

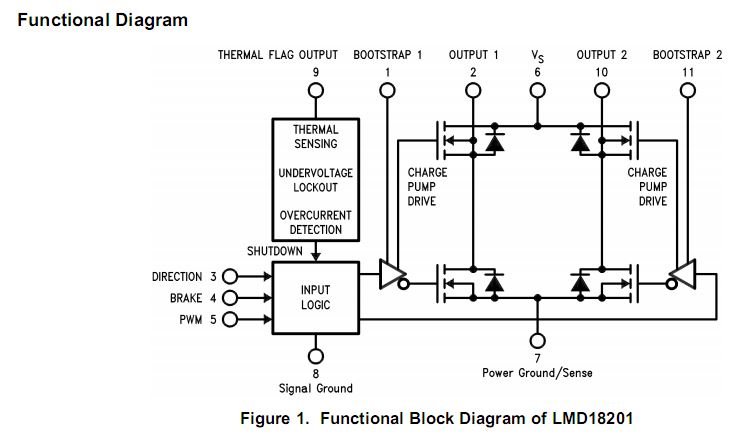
Sử dụng IC cầu H chuyên dụng LMD18201 của Hãng Texas Instruments:

**Các tính năng của IC LMD18201:**

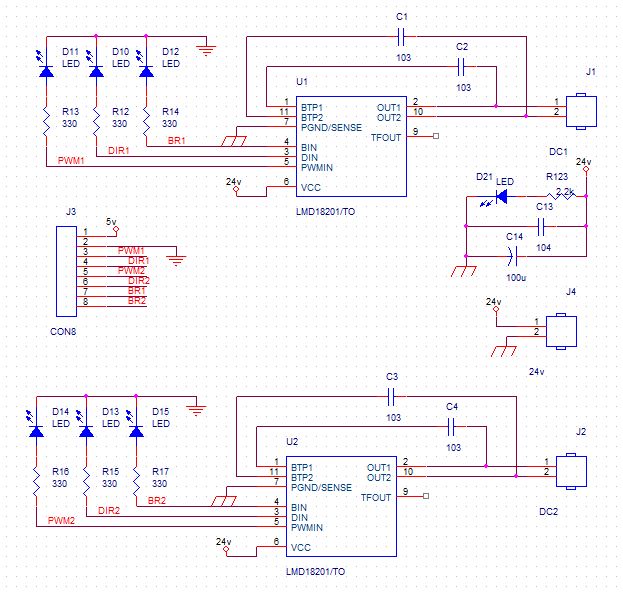
* Dòng max của ic khác lớn 6A
* Kích thước nhỏ gọn
* Mạch điện thiết kế đơn giản
* Dải đáp ứng điện áp rộng từ 12V đến 55V
* Giá thành rẻ

**Nguyên lí hoạt động của IC:**

Đầu tiên ta kéo chân BRAKE của IC xuống 0V để cho phép IC hoạt động, sau đó cấp xung vào chân PWM của IC động cơ sẽ quay và muốn đảo chiều động cơ ta thay đổi mức trên chân DIRECTION điện áp.



***Hình 3.4 Sơ đồ khối IC LMD18201.***



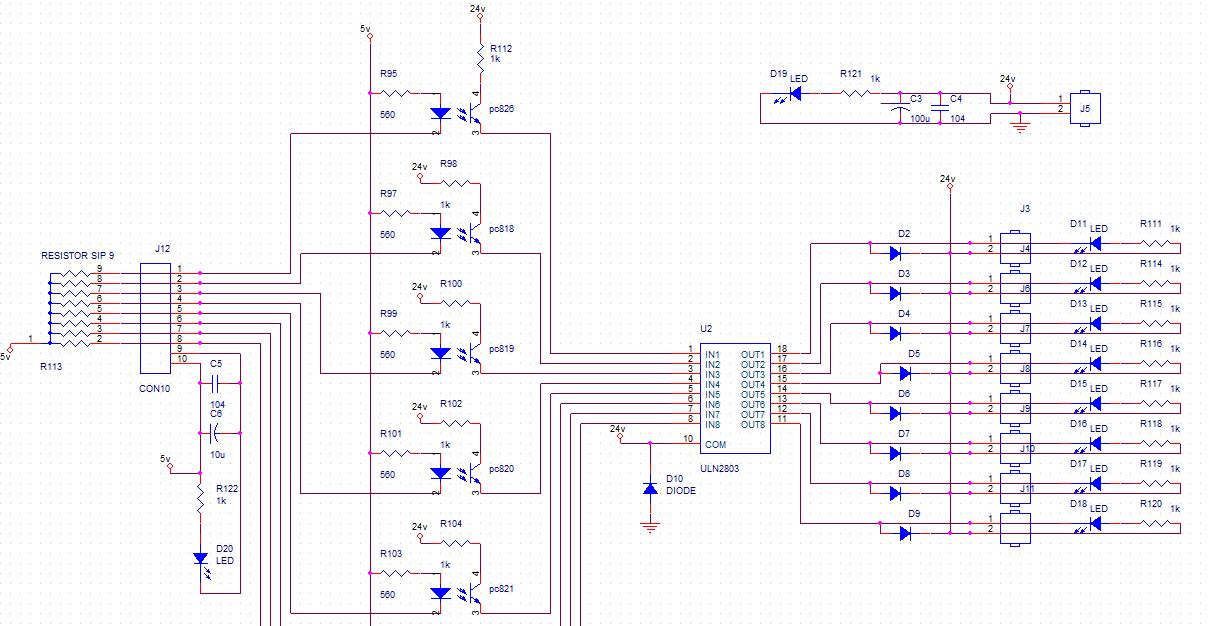
***Hình 3.5 Sơ đồ nguyên lí mạch cầu H dùng IC LMD18201.***

* + 1. **Khối công suất cho hai động cơ leo cầu thang:**



***Hình 3.5 Sơ đồ nguyên lí mạch động lực Fet-Relay.***

* Mạch được thiết cho 2 động cơ đảo chiều.
* Mạch được mắc theo nguyên tắc kéo đẩy của 2 BJT A1015, C1815. Khi có nguồn kích từ chân B của C1815 thì fet dẫn làm cho động cơ hoạt động và khi không kích thì điện áp còn dư của fet sẽ còn và làm gây nóng fet vì vậy A1015 lúc này sẽ bị fet xã hết điện áp và fet hoạt động ổn định
* Diode D23 dùng để bảo vệ vi điều khiển khi kích, mở đúng chiều.
* Trở R330 dùng để hạn dòng kích đúng sườn kích của fet
* Dùng transistor hiệu ứng trường MOSFET IRF540N có khả năng chịu dòng 19A, điều khiển bằng dòng trên ngõ vào G
  + 1. **Khối mạch kích van:**



- Sử dụng opto để cách ly động lực và điều khiển.

- Dùng IC ULN2803 để kích van hoạt động.

# 

# CHƯƠNG 4:

# LƯU ĐỒ THUẬT TOÁN VÀ CHƯƠNG TRÌNH

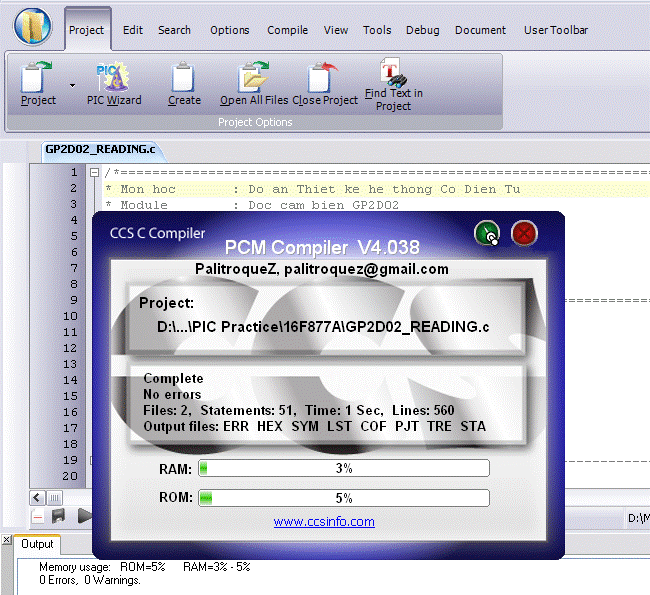
**4.1 **Tổng Quan Về CCS:****

**Để lập trình cho PIC, mọi người có thể chọn cho mình những ngôn ngữ lập trình khác nhau như ASM, CCS C, HT-PIC, pascal, basic,...**

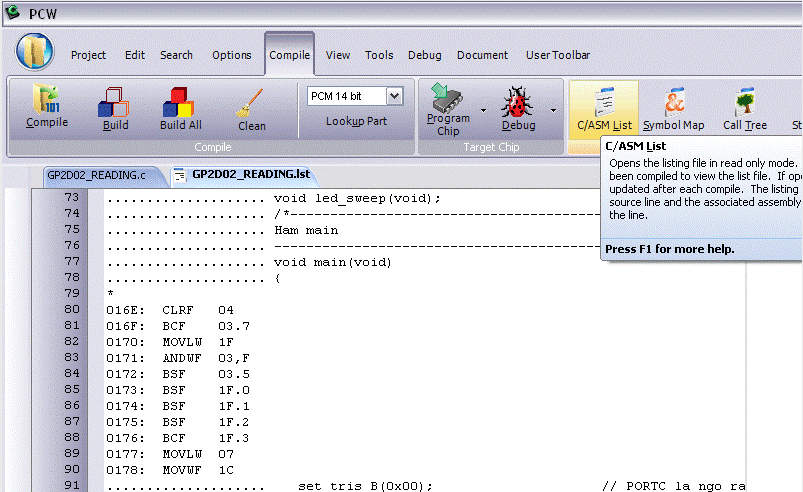
**Với nhiều người, đầu tiên tìm hiểu và viết chương trình cơ bản bằng ASM để hiểu rõ cấu trúc sau đó thì viết bằng CCS C cũng viết lại những cái cơ bản và đi dần lên, tốc độ lúc này nhanh hơn khi viết bằng ASM rất nhiều.**

**Khi viết bằng CCS C thông thường thì dịch ra file.hex có dài hơn so với khi viết bằng ASM. Hai ngôn ngữ CCS C và HT-PIC được ưa chuộng hơn cả, CCS C dễ học,gần gũi với ASM còn HT-PIC là dạng ANSI C.**

**Để lập trình và biên dịch CCS C, dùng chương trình PIC C Complier,sau khi soạn thảo các bạn ấn F9, để dịch,nếu thành công sẽ có thông báo như sau:**



**Ngoài ra, để xem code ASM như thế nào,sau khi dịch bạn chọn mục C/ASM List như hình dưới đây:**



**Vì sao ta sử dung CCS ?**

**Sự ra đời của một loại vi điều khiển đi kèm với việc phát triển phần mềm ứng dụng cho việc lập trình cho con vi điều khiển đó. Vi điều khiển chỉ hiểu và làm việc với hai con số 0 và 1. Ban đầu để việc lập trình cho VĐK là làm việc với dãy các con số 0 và 1. Sau này khi kiến trúc của Vi điều khiển ngày càng phức tạp, số luợng thanh ghi lệnh nhiều lên, việc lập trình với dãy các số 0 và 1 không còn phù hợp nữa, đòi hỏi ra đời một ngôn ngữ mới thay thế. Và ngôn ngữ lập trình Assembly. Ở đây ta không nói nhiều đến Assmebly. Sau này khi ngôn ngữ C ra đời, nhu cầu dùng ngôn ngữ C đề thay cho ASM trong việc mô tả các lệnh lập trình cho Vi điều khiển một cách ngắn gọn và dễ hiểu hơn đã dẫn đến sự ra đời của nhiều chương trình soạn thảo và biên dịch C cho Vi điều khiển : Keil C, HT-PIC, MikroC, CCS…**

**Tôi chọn CCS cho bài giới thiệu này vì CCS là một công cụ lập trình C mạnh cho Vi điều khiển PIC. Những ưu và nhược điểm của CCS sẽ được đề cập đến trong các phần dưới đây.**

**Giới thiệu về CCS ?**

**CCS là trình biên dịch lập trình ngôn ngữ C cho Vi điều khiển PIC của hãng Microchip. Chương trình là sự tích hợp của 3 trình biên dich riêng biết cho 3 dòng PIC khác nhau đó là:**

**- PCB cho dòng PIC 12-bit opcodes**

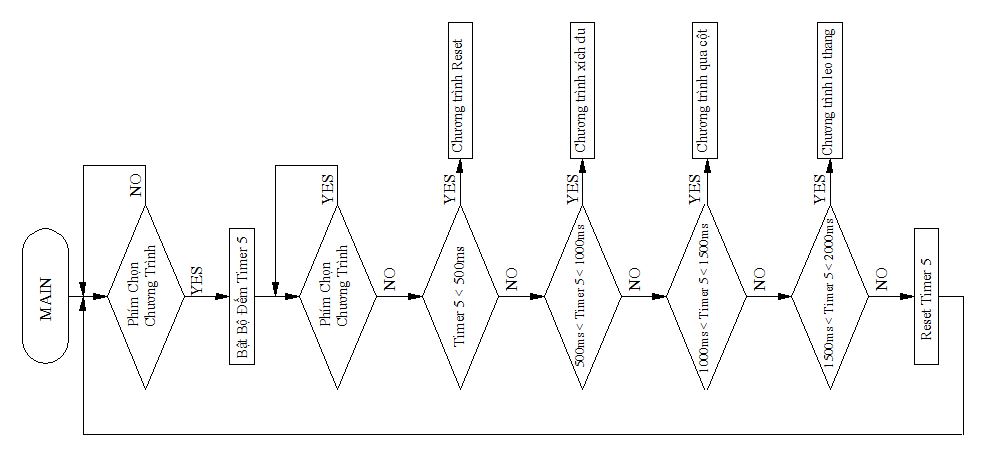
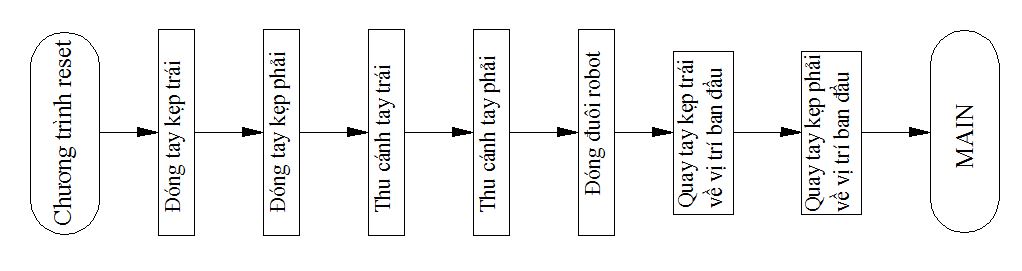
**- PCM cho dòng PIC 14-bit opcodes**

**- PCH cho dòng PIC 16 và 18-bit**

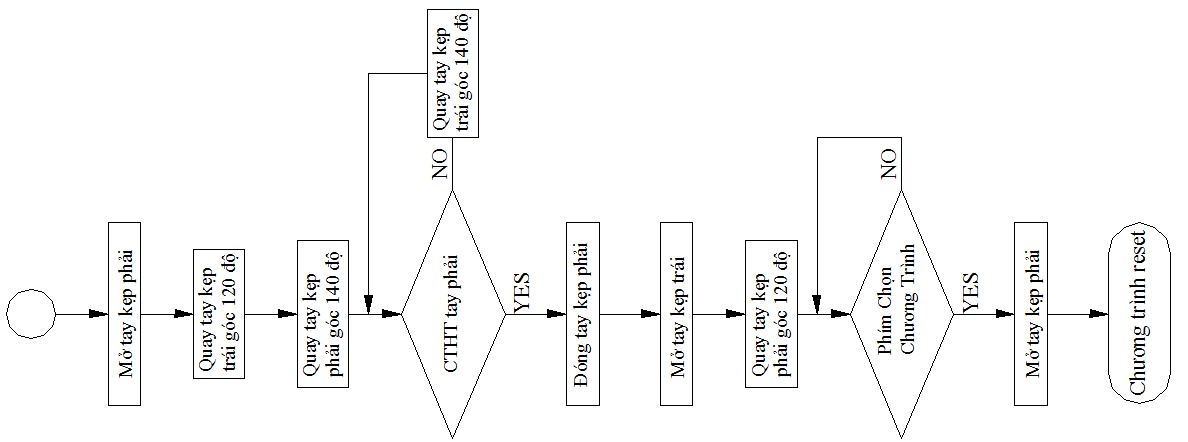
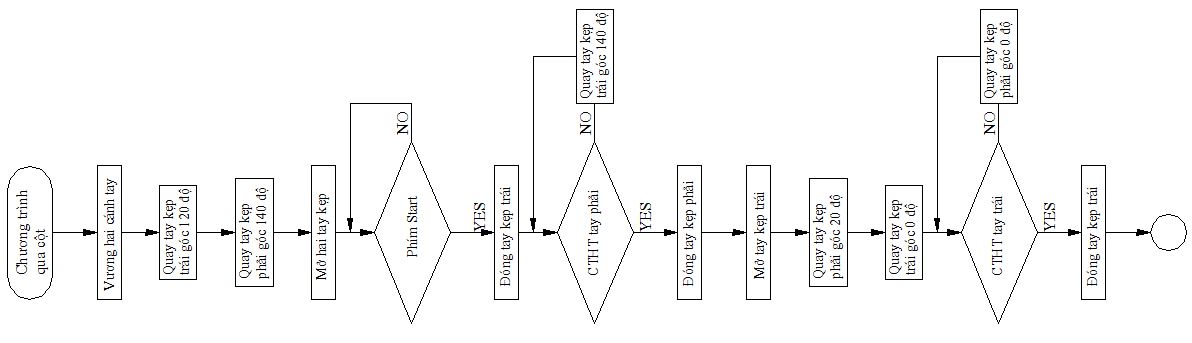
**Tất cả 3 trình biên dich này đuợc tích hợp lại vào trong một chương trình bao gồm cả trình soạn thảo và biên dịch là CCS, phiên bản mới nhất là PCWH Compiler Ver 4.114.**

**Giống như nhiều trình biên dich C khác cho PIC, CCS giúp cho người sử dụng nắm bắt nhanh được vi điều khiển PIC và sử dụng PIC trong các dự án. Các chương trình diều khiển sẽ được thực hiện nhanh chóng và đạt hiệu quả cao thông qua việc sử dụng ngôn ngữ lạp trình cấp cao – Ngôn ngữ C**

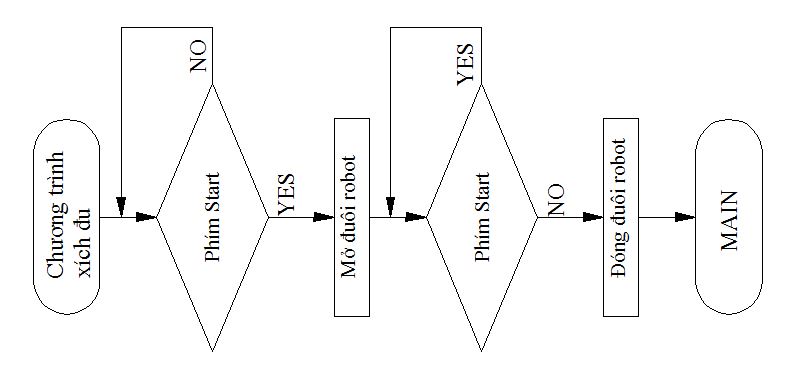
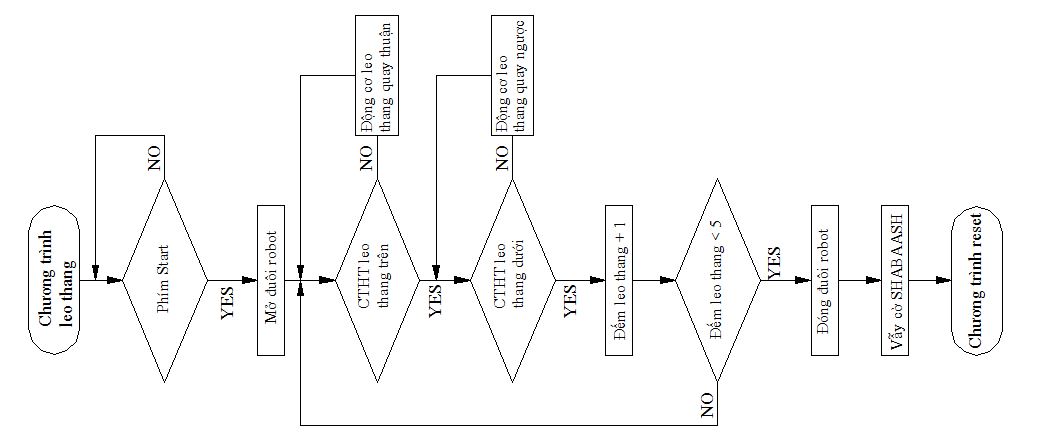
**Tài liệu hướng dẫn sử dụng có rất nhiều, nhưng chi tiết nhất chính là bản Help đi kèm theo phần mềm (tài liệu Tiếng Anh). Trong bản trợ giúp nhà sản xuất đã mô tả rất nhiều về hằng, biến, chỉ thị tiền xủa lý, cấu trúc các câu lệnh trong chương trình, các hàm tạo sẵn cho người sử dụng… Ngoài ra về Tiếng Việt cũng có bản dịch của tác giả Trần Xuân Trường, SV K2001 DH BK HCM. Tài liệu này dịch trên cơ sở bản Help của CCS, tuy rằng chưa đầy đủ nhưng đây là một tài liệu hay, nếu bạn tìm hiểu về PIC và CCS thì nên tìm tài liệu này về đọc.**

**4.2 Lưu đồ thuật toán Robot con:** 

***Hình 4.1 Lưu đồ thuật toán chương trình chính và chương trình reset.***

****

***Hình 4.2 Lưu đồ thuật toán chương trình đi bộ qua cột.***

****

***Hình 4.3 Lưu đồ thuật toán chương trình leo cầu thang và chơi xích đu.***

**4.3 Chương trình:**

#include <lttudong20mhz.h>

#include <khoidong.c>

#include <van-ctht.c>

#include <dongco.c>

#include <qei\_piddc.c>

///// CHUONG TRINH QUET PHIM

int8 demct=0;

///// NGAT TIMER0 LEO THANG, SERVO

volatile CHAR demservo=0, xungleo=100;

volatile INT1 dclep=0 ;

int16 demvanduoi=0;

#INT\_timer0

void ngat\_timer0(VOID)

{

demvanduoi++;

IF (dclep == 1)

{

set\_timer0 (5) ;

demservo++;

IF (demservo > 100)

{

demservo = 0;

pwm3 = 0;

}

IF (demservo >= xungleo) pwm3 = 1;

}

ELSE

{

set\_timer0 (5) ;

demservo++;

IF (demservo >= 200)

{

demservo = 0;

output\_high (pin\_c7) ;

}

IF (demservo >= xungleo) output\_low (pin\_c7);

}

}

///// NGAT TIMER1 PID

volatile INT8 chonpid=4;

#INT\_timer1

void ngat\_timer1(VOID)

{

Ti1++;

Ti2++;

SWITCH (chonpid)

{

case 0 : disable\_interrupts (INT\_timer1); resetdc1 (); resetdc2 ();

BREAK;

CASE 1 : resetdc2 (); piddc1 (setvt1, vt1); break;

CASE 2 : resetdc1 (); piddc2 (setvt2, vt2); break;

CASE 3 : piddc1 (setvt1, vt1); piddc2 (setvt2, vt2); break;

}

set\_timer1 (50286);

}

///// NGAT TIMER5 QUET PHIM

#INT\_timer5

void ngat\_timer5(VOID)

{

set\_timer5 (53035) ;

demct++;

IF (demct == 2) { beep (1, 50); }

IF (demct == 50){ beep (1, 50); }

IF (demct == 100){ beep (1, 50); }

IF (demct == 150){ beep (1, 50); }

if (demct > 200){demct = 0; disable\_interrupts (INT\_timer5); }

}

///// KHAI BAO NHIEM VU

//#include <resetall.c>

#include <resetvan.c>

#include <nhiemvu2.c>

#include <nhiemvu3.c>

#include <nhiemvu3x.c>

#include <nhiemvu4.c>

///// MAIN /////

int8 i=0;

void main()

{

khoidong ();

En2 = 0;

Ti1 = 0;

Ti2 = 0;

setvt1 = 0;

setvt2 = 0;

beep (2, 100);

//ss1 = 1;

WHILE (true)

{

IF (ctsel == 0)

{

enable\_interrupts (INT\_timer5) ;

demct = 0; i = 1; delay\_ms (10) ;

WHILE (ctsel == 0) {}

}

WHILE (i == 1)

{

IF (demct < 50)

{

disable\_interrupts (INT\_timer5) ;

//ss1 = 0; spi\_write (demct); ss1 = 1; delay\_ms (5 ); ss1 = 0; s

pi\_write (demct); ss1 = 1;

i = 0; demct = 0; resetvan ();

}

IF ( (50 < demct)& (demct < 100) )

{

disable\_interrupts (INT\_timer5);

//ss1 = 0; spi\_write (demct); ss1 = 1; delay\_ms (5 ); ss1 = 0; s

pi\_write (demct); ss1 = 1;

i = 0; demct = 0; nhiemvu2 ();

}

IF ( (100 < demct)& (demct < 150) )

{

disable\_interrupts (INT\_timer5);

//ss1 = 0; spi\_write (demct); ss1 = 1; delay\_ms (5 ); ss1 = 0; s

pi\_write (demct); ss1 = 1;

i = 0; demct = 0; nhiemvu3x ();

}

IF ( (150 < demct)& (demct < 200) )

{

disable\_interrupts (INT\_timer5);

//ss1 = 0; spi\_write (demct); ss1 = 1; delay\_ms (5 ); ss1 = 0; s

pi\_write (demct); ss1 = 1;

i = 0; demct = 0; nhiemvu4 ();

}

}

}

}

///// KHOI DONG PIC

void khoidong()

{

trisa = 0b00011000; /// DAU VAO QEI CUNG

porta = 0b11100111;

trisb = 0b11111111; /// DAU VAO CONG TAC HANH TRINH

port\_b\_pullups (0xff) ;

portb = 0xff;

trisc = 0b00110000; /// DAU VAO QEI MEM

portc = 0b00000000;

trisd = 0b00000100; /// DAU VAO SPI 0b00000100

portd = 0b11110100;

trise = 0b00000000;

porte = 0b11111111;

enable\_interrupts (INT\_EXT1) ;

enable\_interrupts (INT\_EXT2) ;

//enable\_interrupts (INT\_timer0) ;

//enable\_interrupts (INT\_timer1) ;

//enable\_interrupts (INT\_timer5) ;

enable\_interrupts (GLOBAL) ;

ext\_INT\_edge (1, L\_to\_H) ;

ext\_INT\_edge (2, L\_to\_H) ;

clear\_interrupt (INT\_EXT1) ;

clear\_interrupt (INT\_EXT2) ;

clear\_interrupt (INT\_timer0) ;

clear\_interrupt (INT\_timer1) ;

clear\_interrupt (INT\_timer5) ;

setup\_qei (QEI\_MODE\_X4\_RESET\_WITH\_INDX, qei\_filter\_enable\_qea|qei\_filter\_enable\_qeb|qei\_filter\_div\_2, 65535) ;

qei\_set\_count (0) ;

//setup\_timer\_0 (RTCC\_INTERNAL|RTCC\_DIV\_1); //6.5 ms overflow, 0.1us

//set\_timer0 (64535) ;

setup\_timer\_0 (RTCC\_INTERNAL|RTCC\_DIV\_2|RTCC\_8\_bit); //102 us overflow, 0.4us

set\_timer0 (5) ;

setup\_timer\_1 (T1\_INTERNAL|T1\_DIV\_BY\_4); //52.4 ms overflow, 0.8us

set\_timer1 (50286) ;

setup\_timer\_5 (T5\_INTERNAL|T5\_DIV\_BY\_4); //52.4 ms overflow, 0.8us

set\_timer5 (53035) ;

//setup\_spi (SPI\_MASTER|SPI\_L\_TO\_H|SPI\_CLK\_DIV\_16) ;

//ss1 = 1;

}

///// DEFINE CHON CT

#bit chon\_nv=portd.2

///// DEFINE VAN

#bit vantay1=porta.5

#bit vantay2=porte.0

#bit vancanh2=porta.0

#bit vancanh1=porta.1

#bit vanduoi=porta.2

///// DEFINE CONG TAC HANH TRINH

#bit ctt2=portb.0

#bit ctc2=portb.1

#bit ctt1=portb.2

#bit ctc1=portb.3

#bit ctld=portb.4

#bit ctlt=portb.5

#bit ctsel=portb.7

#bit cten=portb.6

///// CHUONG TRINH LOA TIN HIEU

#bit trisloa = trisd.7

#bit loa = portd.7

void beep(INT8 solan,int8 time)

{

INT8 i;

FOR (i=0; i<solan; i++)

{

trisloa = 0;

loa = 0;

delay\_ms (20) ;

loa = 1;

delay\_ms (time - 20) ;

}

}

///// DEFINE CAC DONG CO

#bit br1=porte.2

#bit br2=porte.1

#bit trisbr1 = trise.2

#bit trisbr2 = trise.1

#bit dir1=portc.3

#bit dir2=portc.0

#bit trisdir1 = trisc.3

#bit trisdir2 = trisc.0

#bit pwm1=portc.2

#bit pwm2=portc.1

#bit trispwm1 = trisc.2

#bit trispwm2 = trisc.1

void dc1t(INT16 x)

{

trispwm1 = trisdir1 = trisbr1 = 0;

dir1 = 1;

br1 = 0;

setup\_timer\_2 (T2\_DIV\_BY\_4, 250, 1); //401 us overflow, 401 us interrupt

setup\_ccp1 (ccp\_pwm);

set\_pwm1\_duty (x);

}

void dc1n(INT16 x)

{

trispwm1 = trisdir1 = trisbr1 = 0;

dir1 = 0;

br1 = 0;

setup\_timer\_2 (T2\_DIV\_BY\_4, 250, 1); //401 us overflow, 401 us interrupt

setup\_ccp1 (ccp\_pwm);

set\_pwm1\_duty (x);

}

void resetdc1()

{

trispwm1 = trisdir1 = trisbr1 = 0;

setup\_timer\_2 (T2\_DIV\_BY\_4, 250, 1); //401 us overflow, 401 us interrupt

setup\_ccp1 (ccp\_pwm);

set\_pwm1\_duty (0);

br1 = 1;

dir1 = 0;

}

void dc2n(INT16 x)

{

trispwm2 = trisdir2 = trisbr2 = 0;

dir2 = 1;

br2 = 0;

setup\_timer\_2 (T2\_DIV\_BY\_4, 250, 1); //401 us overflow, 401 us interrupt

setup\_ccp2 (ccp\_pwm);

set\_pwm2\_duty (x);

}

void dc2t(INT16 x)

{

trispwm2 = trisdir2 = trisbr2 = 0;

br2 = 0;

dir2 = 0;

setup\_timer\_2 (T2\_DIV\_BY\_4, 250, 1); //401 us overflow, 401 us interrupt

setup\_ccp2 (ccp\_pwm);

set\_pwm2\_duty (x);

}

void resetdc2()

{

trispwm2 = trisdir2 = trisbr2 = 0;

br2 = 1;

dir2 = 0;

setup\_timer\_2 (T2\_DIV\_BY\_4, 250, 1); //401 us overflow, 401 us interrupt

setup\_ccp2 (ccp\_pwm);

set\_pwm2\_duty (0);

}

#bit pwm3 = portd.6

#bit dir3 = portd.5

#bit trispwm3 = trisd.6

#bit trisdir3 = trisd.5

///// QEI MEM + DU LIEU QEI

signed INT16 En2;

#define ChanelA PIN\_C4

#define ChanelB PIN\_C5

#INT\_EXT1

void EXT1\_L2H\_isr(VOID)

{

INTEDG1^=1;

IF (INTEDG1)

{

IF (input (ChanelB)) {En2--; }

ELSE{En2++; }

}

ELSE

{

IF (input (ChanelB)) {En2++; }

ELSE{En2--; }

}

}

#INT\_EXT2

void EXT2\_isr(VOID)

{

INTEDG2^=1;

IF (INTEDG2)

{

IF (input (ChanelA)) {En2++; }

ELSE{En2--; }

}

ELSE

{

IF (input (ChanelA)) {En2--; }

ELSE{En2++; }

}

}

/// PID DONG CO 1 //////////////////

volatile FLOAT kp=2, ki=0.2,kd=0.7, pid1, pid2;

signed INT16 realvt1, errs1=0, err1=0, delta1=0, epre1=0, duty1 ;

signed INT16 realvt2, errs2=0, err2=0, delta2=0, epre2=0, duty2 ;

volatile UNSIGNED int16 Ti1,Ti2;

volatile SIGNED int16 setvt1,setvt2,vt1=900,vt2=900;

void piddc1(SIGNED int16 setvt1, signed int16 vt1)

{

br1 = 0;

realvt1 = qei\_get\_count ();

err1 = setvt1 - realvt1;

IF (Ti1 >= 2)

{

delta1 = err1 - epre1;

Ti1 = 0;

IF (delta1 > - 10) errs1 += err1; //(err1 > - 20)&& (err1 < 20)

IF (errs1 > 200) errs1 = 200;

IF (errs1 < -200) errs1 = - 200;

epre1 = err1;

}

pid1 = kp \* err1 + ki \* errs1 + kd \* delta1;

duty1 = pid1;

IF (duty1 > 0) dir1 = 0;

IF (duty1 < 0) dir1 = 1;

duty1 = abs (duty1);

IF (duty1 > 5) duty1 += 230;

IF (duty1 > vt1) duty1 = vt1;

set\_pwm1\_duty (duty1);

}

/// PID DONG CO 2 //////////////////

void piddc2(SIGNED int16 setvt2, signed int16 vt2)

{

br2 = 0;

realvt2 = En2;

err2 = setvt2 - realvt2;

IF (Ti2 >= 2)

{

delta2 = err2 - epre2;

Ti2 = 0;

IF (delta2 > - 10) errs2 += err2; /// (err2 > - 20)&& (err2<20)

IF (errs2 > 200) errs2 = 200;

IF (errs2 < -200) errs2 = - 200;

epre2 = err2;

}

pid2 = kp \* err2 + ki \* errs2 + kd \* delta2;

duty2 = pid2;

IF (duty2 > 0) dir2 = 1;

IF (duty2 < 0) dir2 = 0;

duty2 = abs (duty2);

IF (duty2 > 5) duty2 += 230;

IF (duty2 > vt2) duty2 = vt2;

set\_pwm2\_duty (duty2);

}

///// RESET TAT CA

int8 ire=0;

void resetvan()

{

disable\_interrupts(INT\_timer1);

disable\_interrupts(INT\_timer0);

delay\_ms(200);

beep(1,100);

FOR (ire=0; ire<50; ire++)

{

IF (ctsel == 0) {vancanh1 = vancanh2 = 0; delay\_ms (1000); vantay1=vantay2 = 0; ire = 100; }

delay\_ms (10) ;

}

delay\_ms(500);

WHILE (ctc1==1) { dc1t(470); }

resetdc1();

WHILE (ctc2==1) { dc2t(550); }

resetdc2();

beep(1,100);

vantay1=vantay2=vancanh1=vancanh2=vanduoi=1;

trispwm3=trisdir3=0;

pwm3=dir3=1;

enable\_interrupts(INT\_timer0);

dclep=1;

WHILE (ctlt==1)

{

dir3 = 1;

xungleo = 30;

}

xungleo = 0;

delay\_ms (300) ;

WHILE (ctlt==0)

{

dir3 = 0;

xungleo = 60;

}

xungleo = 0;

beep (1, 100);

disable\_interrupts(INT\_timer0);

pwm3=1;

dir3=1;

beep(1,100);

qei\_set\_count(0);

En2=0;

Ti1=0;

Ti2=0;

setvt1=0;

setvt2=0;

chonpid=4;

demct=200;

enable\_interrupts(INT\_timer5);

}

///// NHIEM VU 2

void nhiemvu2()

{

delay\_ms(1000);

beep(2,100);

WHILE(cten==0){}

delay\_ms(1000);

vanduoi=0;

delay\_ms(500);

WHILE(cten==1) {}

delay\_ms(1000);

vanduoi=1;

demct=200;

enable\_interrupts(INT\_timer5);

}

///// NHIEM VU 3

int8 inv3=0;

void nhiemvu3()

{

delay\_ms(1000);

beep(3,100);

setup\_timer\_2(T2\_DIV\_BY\_4,250,1); //401 us overflow, 401 us interrupt

setup\_ccp1 (ccp\_pwm);

setup\_ccp2 (ccp\_pwm);

vantay1=vantay2=vancanh1=vancanh2=vanduoi=1;

vancanh1=vancanh2=0;

chonpid=2;

setvt2=280;

enable\_interrupts(INT\_timer1);

WHILE (En2<280) {}

chonpid=1;

setvt1=227;

WHILE (qei\_get\_count() <227) {}

vantay1=vantay2=0;

WHILE (ctsel==1) {}

IF (ctsel==0) { vantay1=1; beep(1,100); delay\_ms(500); }

WHILE (ctsel==0) {}

WHILE (cten==0) {}

delay\_ms(2500);

chonpid=0;

dc1n(600);

FOR ( inv3=0; inv3 < 6 ; inv3++)

{

WHILE (ctt2 == 1) { dc1n (600); } //if (qei\_get\_count () > 285) {resetdc1 (); beep (10, 100); } }

IF (ctt2 == 0) {vantay2 = 1; resetdc1 (); }

delay\_ms (100) ;

IF (ctt2 == 1) {vantay2 = 0; dc1t (300); delay\_ms (400); dc1n (600); }

}

vantay2=1; resetdc1();

delay\_ms(1000);

vantay1=0;

chonpid=2;

setvt2=100;

enable\_interrupts(INT\_timer1);

WHILE ( En2 >250) {}

setvt2=10;

setvt1=0;

chonpid=3;

WHILE ( En2 >10) {}

chonpid=0;

dc2t(600);

FOR (inv3=0; inv3 < 6; inv3++)

{

WHILE (ctt1 == 1) { dc2t (600); } //if (En2 < - 5 ){resetdc2 (); beep (10, 100); } }

IF (ctt1 == 0) {vantay1 = 1; resetdc2 (); }

delay\_ms (100) ;

IF (ctt1 == 1) {vantay1 = 0; dc2n (300); delay\_ms (400); dc2t (600); }

}

vantay1=1;resetdc2();

delay\_ms(800);

vantay2=0;

delay\_ms(200);

chonpid=1;

setvt1=200;

enable\_interrupts(INT\_timer1);

WHILE (qei\_get\_count() < 30) {}

setvt1=270;

setvt2=275;

chonpid=3;

WHILE (qei\_get\_count() < 270) {}

chonpid=0;

dc1n(600);

FOR ( inv3=0; inv3 < 6 ; inv3++)

{

WHILE (ctt2==1) {dc1n(600);} // if ( qei\_get\_count() >285 ) {resetdc1(); beep(10,100);} }

IF (ctt2==0) {vantay2=1; resetdc1(); }

delay\_ms(100);

IF (ctt2==1) {vantay2=0; dc1t(300); delay\_ms(400); resetdc1(); dc1n(600); }

}

vantay2=1; resetdc1();

delay\_ms(1000);

vantay1=0;

chonpid=2;

setvt2=0;

enable\_interrupts(INT\_timer1);

vancanh2=1;

WHILE ( En2 > 240) {}

chonpid=0;

vancanh2=0;

WHILE (cten==1) {}

WHILE (ctsel==1) {}

vantay2=0;

delay\_ms(1000);

WHILE (ctsel==1) {}

resetvan ();

demct=200;

enable\_interrupts(INT\_timer5);

}

///// NHIEM VU 4

void nhiemvu4()

{

INT8 inv4;

delay\_ms(500);

beep(4,100);

trispwm3=trisdir3=0;

pwm3=dir3=1;

WHILE (cten==0) {}

delay\_ms(1000);

dclep=1;

enable\_interrupts(INT\_timer0);

vanduoi=0;

WHILE (ctld==1)

{

dir3 = 0;

xungleo = 80;

}

dir3=1;

xungleo=0;

beep(1,100);

FOR (inv4=0; inv4 < 3; inv4++) /// 4 lan la ok

{

WHILE (ctlt == 1)

{

dir3 = 1;

xungleo = 80;

}

dir3 = 0;

xungleo = 0;

beep (1, 100) ;

WHILE (ctld == 1)

{

dir3 = 0;

xungleo = 85;

}

dir3 = 1;

xungleo = 0;

beep (1, 100) ;

}

//delay\_ms (1000) ;

/////////////////////delay nga robot

WHILE (ctlt == 1)

{

dir3 = 1;

xungleo = 80;

}

dir3 = 0;

xungleo = 0;

beep (1, 100) ;

WHILE (ctld == 1)

{

dir3 = 0;

xungleo = 85;

}

dir3 = 1;

xungleo = 0;

beep (1, 100) ;

WHILE (ctlt==1)

{

dir3 = 1;

xungleo = 80;

}

dir3 = 0;

xungleo = 0;

beep (1, 100) ;

WHILE (ctlt==0)

{

dir3 = 0;

xungleo = 60;

}

xungleo = 0;

beep (1, 100) ;

vanduoi=1;

pwm3=1;

dir3=1;

dclep=0;

xungleo=10;

WHILE (cten==1) { }

disable\_interrupts(INT\_timer0);

demct=200;

enable\_interrupts(INT\_timer5);

}

# CHƯƠNG 5:

# KẾT LUẬN

Sau khi tham gia cuộc thi Robocon 2014 và hoàn thành đề tài tốt nghiệp này. Chúng em rút ra được nhiều kinh nghiệm về thiết kế, chế tạo và xây dựng các quy trình thiết kế cơ khí cũng như hệ thống điều khiển cho Robot. Bên cạnh đó, chúng em cũng đã tìm tòi để phát triển các công nghệ mới giúp điều khiển Robot ổn định hơn…

## Thiết kế và chế tạo cơ khí Robot

Để đạt hiệu quả trong việc thiết kế và chế tạo, ta cần có một quy trình thiết kế và chế tạo hợp lý. Sau đây là các bước chúng em đã thực hiện để thiết kế và chế tạo trong cuộc thi Robocon 2014 vừa qua:

* + Tìm hiểu kỹ và nêu ra các yêu cầu, các ràng buộc kỹ thuật của Robot.
  + Phân tích các yếu tố kỹ thuật để tìm các phương án thiết kế.
  + Chọn các phương án phù hợp nhất ứng với các cơ cấu cơ khí thỏa mãn được phần lớn các yêu cầu đề ra. Bên cạnh đó, người làm mạch và lập trình cũng kết hợp tính toán và chọn lựa các động cơ truyền động phù hợp với các cơ cấu cơ khí.
  + Tính toán và thiết kế trên phần mềm vẽ cơ khí như Solidworks, Autocad, Autodesk Inventor…
  + Tiếp tục phân tích các kết cấu cơ khí trong phần mềm, chọn phương án tối ưu nhất có thể.
  + Chế tạo Robot.

Ngoài các bước thực hiện thì thời gian để hoàn thành chúng cũng cần có một kế hoạch rõ ràng, càng chi tiết thì công việc trong nhóm sẽ logic, hiệu quả.

Người thiết kế cơ khí phải kết hợp chặt chẽ với người lập trình điều khiển, việc này sẽ đảm bảo rằng các kết cấu cơ khí hoạt động được tối ưu trong thực tế.

Để có một Robot tốt, ngoài việc thiết kế tốt thì việc gia công Robot tỉ mỉ, chính xác đóng vai trò then chốt vào sự thi đấu ổn định của Robot.

## Thiết kế mạch điều khiển

Tương tự việc thiết kế cơ khí, thiết kế mạch điều khiển cũng cần có các bước thực hiện để hiệu quả điều khiển là tốt nhất, đơn giản nhất như tính toán tín hiệu vào ra, tính chọn các mạch sao cho phù hợp, độ ổn định cao.

Bên cạnh đó, việc tìm hiểu nhiều công nghệ mới giúp người thiết kế mạch, lập trình có nhiều chọn lựa cho việc điều khiển, làm tăng độ ổn định của Robot trong khi thi đấu. Năm nay, chúng em đã tìm hiểu và đã đạt được một số thành công trong việc phát triển các module mạch và ứng dụng các thuật toán vào điều khiển Robot như:

* + Tìm hiểu và ứng dụng vi điều khiển PIC để điều khiển Robot tốt hơn, ổn định hơn thay vì sử dụng vi điều khiển 8051 đã dùng các năm trước.
  + Tìm hiểu và thực hiện module giao tiếp GamePad PS2 và vi điều khiển giúp tối ưu hơn về thiết kế điều khiển Robot bằng tay.
  + Thiết kế module cảm biến quang thông minh tự học vạch trắng.
  + Tìm hiểu và ứng dụng module điều khiển động cơ vòng kín dùng thuật toán PID.

## Hướng phát triển

Mặc dù đề tài chỉ là tham gia trong một sân chơi, nhưng nó đã giúp chúng em có thể thực hành các kiến thức chuyên môn mà đã được học, thêm các kinh nghiệm về việc giải quyết các bài toán tối ưu về cơ khí, điều khiển.

Chính những kiến thức đã học được và các kinh nghiệm trải qua trong cuộc thi giúp chúng em phát triển các đề tài có tính thực tiễn hơn sau khi ra trường.

Ngoài ra, các kiến thức chúng em tìm hiểu được sẽ được kế thừa bởi các sinh viên năm dưới, nó có thể giúp các bạn ấy có thể tìm hiểu nhanh hơn, sâu hơn để có thể ứng dụng vào các đề tài có ý nghĩa với cuộc sống.

**TÀI LIỆU THAM KHẢO**

**[1].**Chi tiết máy. Nguyễn Trọng Hiệp. Nhà xuất bản giáo dục .

**[2].**Giáo trình kỹ thuật số.Nguyễn Thuý Vân. Nhà xuất bản khoa học kỹ thuật .

**[3].**Giáo trình kỹ thuật xung,kỹ thuật số. Nguyễn Văn Phòng. Đại học bách khoa ĐN .

**[4].**Giáo trình kỹ thuật xung căn bản và nâng cao. Nhà xuất bản trẻ .

**[5].**Sổ tay tra cứu và thay thế linh kiện bán dẫn. Nhà xuất bản trẻ 1999 .

**[6].**Sổ tay sơ đồ linh kiện điện tử. Dương Minh Trí. Nhà xuất bản khoa học kỹ thuật .

**[7].**Simplistic Control of Mecanum Drive của Ian McInerney, FRC Team 2022

**Một số trang web tham khảo:**

www.codientubkdn.com

www.bkpro.inf

www.diendandientu.com

www.dientuvietnam.net

www.sixca.com

[www.Alldatasheet.com](http://www.Alldatasheet.com)