# Competencia de Robótica
# R2-D2 2014

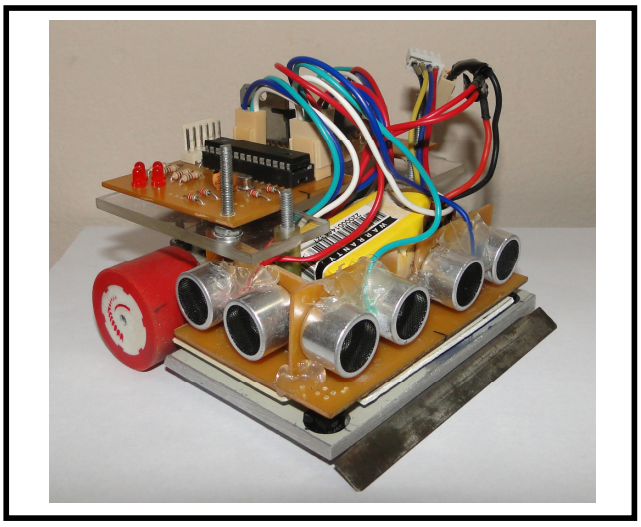
Categoría: MINISUMO

Nombre del Robot: COYOTE

Institución: ITEC RAFAEL DE AGUIAR

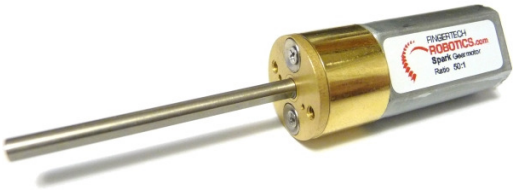
Participantes:

ELIO ANTONIO PAEZ
RAMIRO MARTIN PAEZ
RODRIGO NICOLAS PAEZ



El robot consta de los siguientes elementos que fueron comprados por internet(a un costado de cada imagen se encuentra el link).

Para la tracción dos motores de Fingertech, a los cuales se les corto el eje para poder acoplar las ruedas y estar dentro del reglamento de la categoría. Las ruedas también fueron compradas en el mismo lugar como las baterías de polímero de 360 mAh.




http://www.fingertechrobotics.com/proddetail.php?prod=ft-spark16

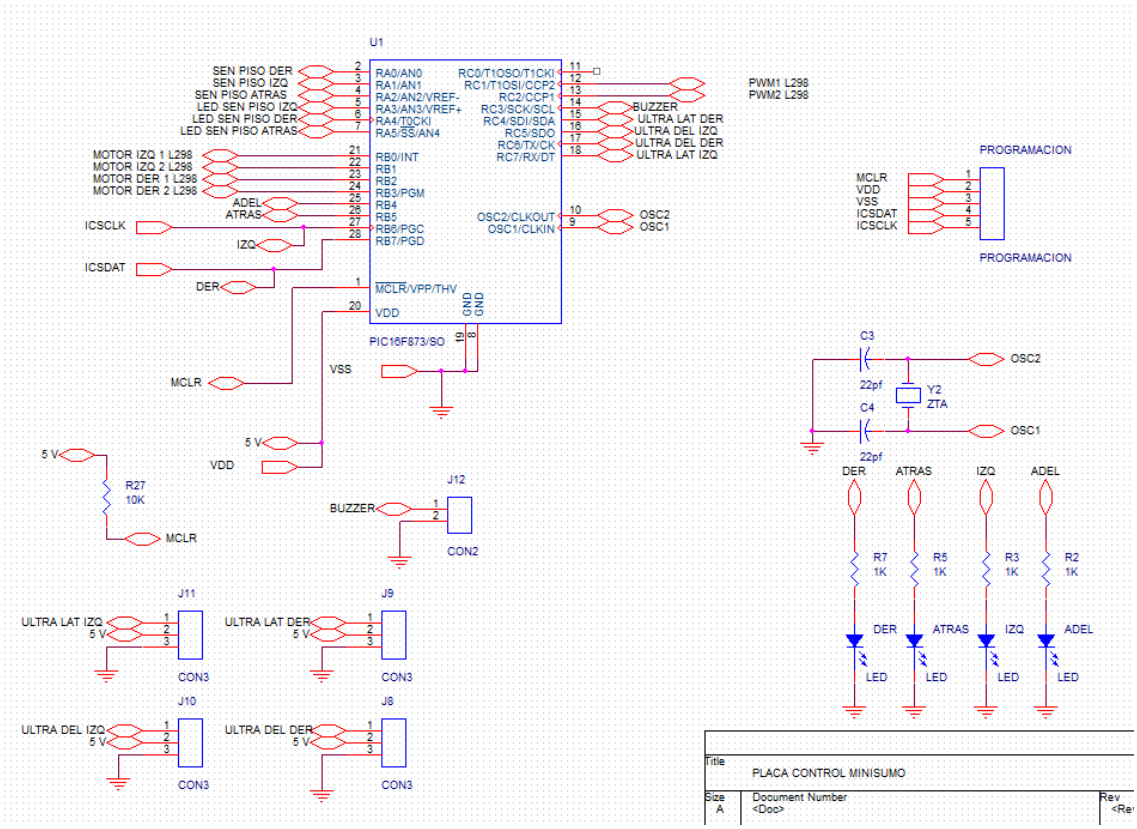
http://www.fingertechrobotics.com/proddetail.php?prod=ft-spark16


http://www.fingertechrobotics.com/proddetail.php?prod=R-3S-mAh

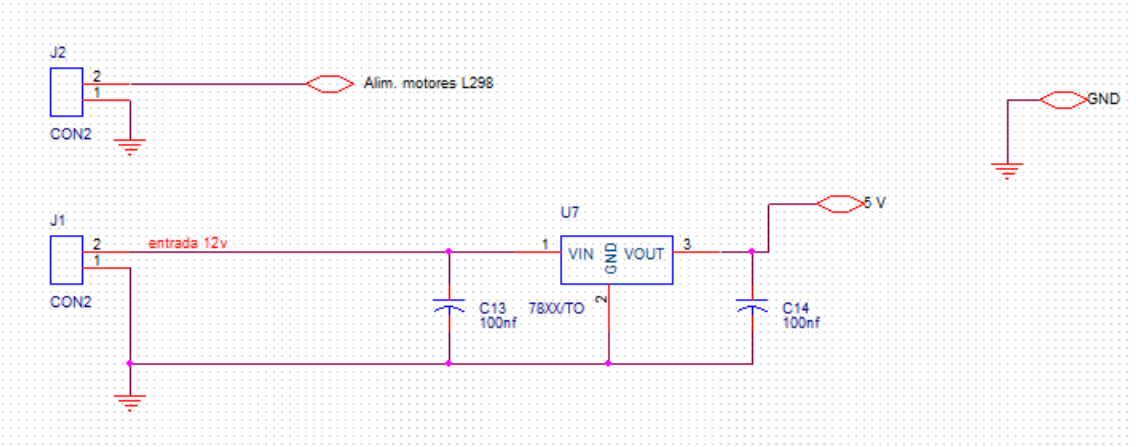
Para este robot se pensó en tartar de bajar la altura pero debido a la batería no se lo pudo hacer tan bajo como se quería. El peso del robot es demasiado bajo y como mejora se pensó en cambiar la base que es de aluminio por otra de hierro.

En lo que respecta a la electrónica utilizamos un pic 16F873A para la lógica y un L298 para el manejo de los motores, a este le agregamos todos los diodos de protección. Para la detección del tatami se utilizaran los CNY70. Y para la detección del rival tres sensores de ultrasonido.
A continuación se agrega los esquematicos de los circuitos utilizados y el programa.

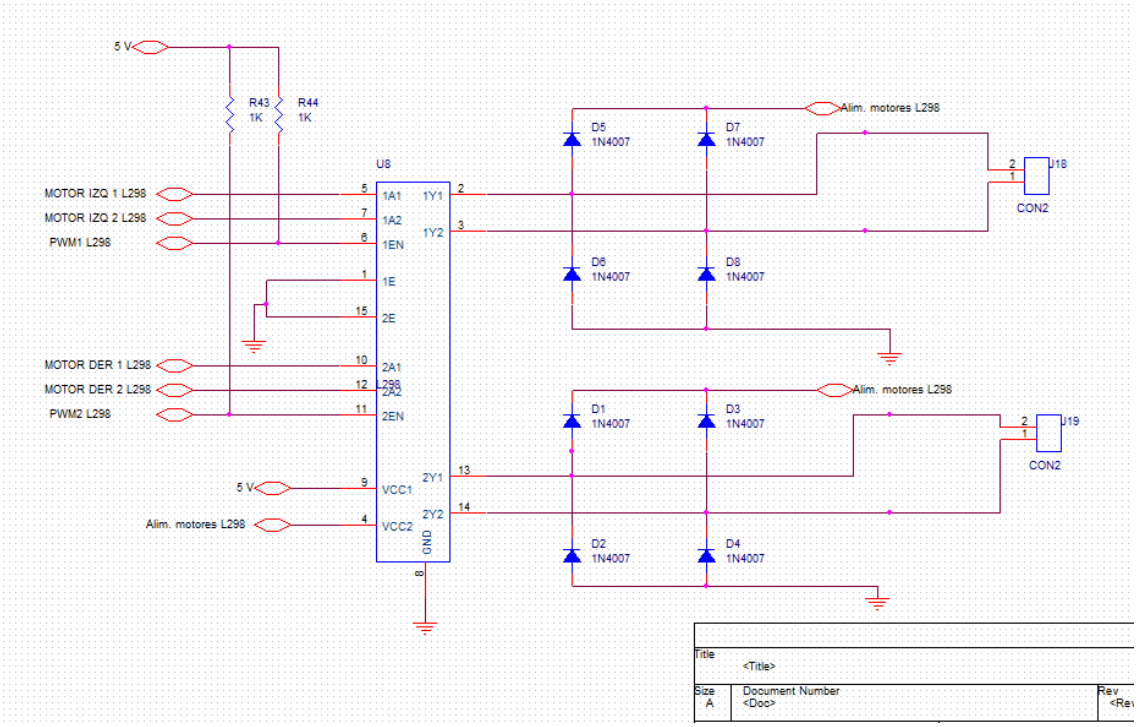
Esta imagen muestra todo lo relacionado a la lógica del microcontrolador, como y donde están conectados los motores y los sensores. Tambien se le incorporo un conector para realizar más fácil la programación del mismo. Como verán también se le agregaron un par de led, para utilizarlos en el debug del robot.




En esta imagen mostramos el regulador utilizado y algo muy importante los capacitores de filtrado.



Aca mostramos el puente H utilizado y sus conecciones.




Para la programación se utilizó MIKROBASIC, y el programa es el siguiente:

```
program FitoMinisumo

' Declarations section

    symbol   LED_ATRAS      = portb.5          ' Led de indicacion de estado de sensores de ultrasonido
    symbol   LED_IZQ        = portb.6
    symbol   LED_ADEL       = portb.4
    symbol   LED_DER        = portb.7
  '' symbol   PULS1         = porta.4          ' Declaracion pulsador de arranque de programa
   '' symbol   PULS2        = porta.5
    symbol   PWMADELMI      = portb.0       ' Habilitacion para que el motor izquierdo vaya hacia adelante
    symbol   PWMATMI        = portb.1       ' Habilitacion para que el motor izquierdo vaya hacia atras
    symbol   PWMADELMD      = portb.2       ' Habilitacion para que el motor derecho vaya hacia adelante
    symbol   PWMATMD        = portb.3       ' Habilitacion para que el motor derecho vaya hacia atras
    symbol   PISOIZQ        = portb.4    ' sensor de piso, izquierda (los utilizare con interrupciones del tipo RBIE)
    symbol   PISODER        = portb.5  ' sensor de piso, derecha (los utilizare con interrupciones del tipo RBIE)
  '' symbol   CNYTRA        = portb.6  ' sensor de piso, derecha (los utilizare con interrupciones del tipo RBIE)

    symbol   BUZZER         = portc.3
    symbol   ULTRALATDER    = portc.4                  ' sensor lateral del lado derecho
    symbol   FRENTEIZQ      = portc.5                  ' sensor delantero del lado izquierdo
    symbol   ULTRALATIZQ    = portc.6                  ' sensor lateral del lado izquierdo
    symbol   FRENTEDER      = portc.7                  ' sensor delantero del lado derecho

    symbol    ADELANTE      = portb    = %00000101
    symbol    ATRAS         = portb    = %00001010
    symbol    GIRODERECHA   = portb    = %00001001
    symbol    GIROIZQUIERDA = portb    = %00000110
```

```
    symbol    derecha=1
    symbol    izquierda=0
    "symbol    GIROIZQARCOAD     = porta     = %0000

    ' Declaración de variables

    dim     aux as byte       ' variable utilizada al comienzo para guardar la lectura inicial del objetivo
  "   dim     intduty1 as byte  ' variable utilizada para guardar el valor generado por la interrupcion de los sensores
de piso
  "   dim     intduty2 as byte
   "  dim     auxint as bit                    ' variable utilizada como bandera de las interrupciones, para evitar el
problema del llamado de un procedimiento dentro de la misma.
      dim     current_duty1   as   byte
      dim     current_duty2   as   byte
      dim     contrutina      as   integer
      dim     contreinicio    as   integer
      dim ultimomov,paso, temp as byte     "derecha=1 izquierda=0

    sub procedure interrupt
        if intcon.RBIF = 1 then
           ATRAS
           TRISC.1=1
           TRISC.2=1
           DELAY_MS(100)
           else
        end if
        intcon.RBIF = 0
        TRISC.1=0
        TRISC.2=0
    end sub

    sub procedure       Configuracion
        adcon1 = $06
        porta = 0
        trisa = %00000000
        portb = 0
        trisb = %01110000
        portc = 0
        trisc = %11110000
        intcon.gie=0
        intcon.RBIE=0
        PWM1_Init(5000)                 ' Initialize PWM2 module at 5KHz
        PWM2_Init(5000)                 ' Initialize PWM2 module at 5KHz
       " auxint=0
        Sound_Init(PORTC, 3)

        PWM1_Set_Duty(0)       ' Set current duty for PWM1
        PWM2_Set_Duty(0)       ' Set current duty for PWM2
        PWM1_Start()                    ' start PWM1
        PWM2_Start()                    ' start PWM2
    end sub
```

```
sub procedure Tone1()
   Sound_Play(659, 250)              ' Frequency = 659Hz, duration = 250ms
end sub

sub procedure Tone2()
   Sound_Play(698, 250)              ' Frequency = 698Hz, duration = 250ms
end sub

sub procedure Tone3()
   Sound_Play(784, 250)              ' Frequency = 784Hz, duration = 250ms
end sub

sub procedure Tone4()
   Sound_Play(824, 250)              ' Frequency = 784Hz, duration = 250ms
end sub

sub procedure Tone5()
   Sound_Play(925, 250)              ' Frequency = 784Hz, duration = 250ms
end sub

sub procedure Melody()               ' Plays the melody "Yellow house"
   Tone1() Tone2() Tone3() Tone3()
   Tone1() Tone2() Tone3() Tone3()
   Tone1() Tone2() Tone3()
   Tone1() Tone2() Tone3() Tone3()
   Tone1() Tone2() Tone3()
   Tone3() Tone3() Tone2() Tone2() Tone1()
end sub



sub procedure Cincosegundos
          Tone1()
          delay_ms (750)        "delay del reglamento

          Tone2()
          delay_ms (750)        "delay del reglamento

          Tone3()
          delay_ms (750)        "delay del reglamento

          Tone4()
          delay_ms (750)        "delay del reglamento

          Tone5()
          delay_ms (750)        "delay del reglamento

          ultimomov   =   0
end sub


sub procedure   valorPWM

    PWM1_Set_Duty(current_duty1)
    PWM2_Set_Duty(current_duty2)


end sub
```

```
sub procedure probarmotores

            ADELANTE
            current_duty1 = 0      'este es el  derecho
            current_duty2 = 255    'este el izquierdo
            valorPWM
            delay_ms (3000)
             ADELANTE
            current_duty1 = 255      'este es el izquierdo
            current_duty2 = 0    'este el derecho
            valorPWM
            delay_ms (3000)
             ATRAS
            current_duty1 = 0      'este es el izquierdo
            current_duty2 = 255    'este el derecho
            valorPWM
            delay_ms (3000)
             ATRAS
            current_duty1 = 255      'este es el izquierdo
            current_duty2 = 0    'este el derecho
            valorPWM
            delay_ms (3000)
end sub

main:
           Configuracion
    Cincosegundos
    intcon.gie=0
    intcon.RBIE=0
 while (1)
           ultimomov = 0
          "if ultimomov = izquierda       then
          "                              ultimomov = izquierda
         '' else
                                  ultimomov = derecha
        "   end if
        '' ultimomov = izquierda
        aux=portb and   %00110000         "00DI0000
          select case aux
                case %00100000        "sensor de piso derecho
                    ATRAS
                    current_duty1 = 255
                    current_duty2 = 255       ''
                    PWM1_Set_Duty(current_duty1)
                    PWM2_Set_Duty(current_duty2)
                    delay_ms (300)

                case %00010000        "sensor de piso izquierdo
                    ATRAS
                    current_duty1 = 255
                    current_duty2 = 255       ''
                    PWM1_Set_Duty(current_duty1)
                    PWM2_Set_Duty(current_duty2)
                    delay_ms (300)

                case %00110000       "ambos
                    ATRAS
                    current_duty1 = 255
```

```
        current_duty2 = 255       ''
        PWM1_Set_Duty(current_duty1)
        PWM2_Set_Duty(current_duty2)
        delay_ms (300)

end select


 aux= portc and   %11110000
 select case aux
     case   %00000000
            ADELANTE
            if ultimomov = derecha then
                    current_duty1 = 0
                    current_duty2 = 200       ''1 gira derecha
                    PWM1_Set_Duty(current_duty1)
                    PWM2_Set_Duty(current_duty2)
            else

                      current_duty1 = 200
                      current_duty2 = 0       '' 0 gira izquierda
                      PWM1_Set_Duty(current_duty1)
                      PWM2_Set_Duty(current_duty2)

            end if

                     ''   if   ultimomov =  derecha then
                      ''         ADELANTE
                        ''       current_duty1 = 255      'este es el derecho
                         ''      current_duty2 = 100    'este el izquierdo
                          ''     valorPWM
                           ''  else

                              ''        ADELANTE
                                 ''     current_duty1 = 100
                                  ''       current_duty2 = 255
                                  ''       valorPWM
                       ''  end if


    case  %01000000            ' sensor frontal izquierda detectando

        ADELANTE
        current_duty1 = 125
        current_duty2 = 0
        ultimomov = izquierda
        PWM1_Set_Duty(current_duty1)
        PWM2_Set_Duty(current_duty2)


    case  %00100000            ' sensor frontal derecho detectando

        ADELANTE
        current_duty1 = 0
        current_duty2 = 125
        ultimomov = derecha
        PWM1_Set_Duty(current_duty1)
        PWM2_Set_Duty(current_duty2)
```

```
        case  %01100000             ' ambos sensores frontales detectando

              ADELANTE
              current_duty1 = 255
              current_duty2 = 255
              PWM1_Set_Duty(current_duty1)
              PWM2_Set_Duty(current_duty2)


       '' case  %00010000            ' sensor lateral izquierdo detectando

         "      ADELANTE               'aca falta que gire rapido!!!!!
           "    current_duty1 = 225    'y 90 grados lo tenemos que calcular
             ''  current_duty2 = 0
             ''  ultimomov = izquierda
              " PWM1_Set_Duty(current_duty1)
              " PWM2_Set_Duty(current_duty2)


        case  %10000000             ' sensor lateral derecho detectando

              ADELANTE              'aca falta que gire rapido!!!!!
              current_duty1 = 0     'y 90 grados lo tenemos que calcular
              current_duty2 = 225
              ultimomov = derecha
              PWM1_Set_Duty(current_duty1)
              PWM2_Set_Duty(current_duty2)


        case  else

                            ' ruido asegurado

      end select

 wend

end.
```

A este programa le falta todavía implementar el circuito de arranque, que esperamos implementar en esta semana.