iRover
Rover controllato da iPhone

GUIDA ALL'USO

ITIS "FAUSER" NOVARA	**CLASSE 5 AI A.S. 2011/2012**	**CLAUDIO CARDINALE**

La pubblicazione è suddivisa in capitoli, il cui titolo è illustrato nella seguente tabella:

NOTA

Per una consultazione rapida e precisa della pubblicazione consultare l'indice generale, al capitolo successivo.

INDICE GENERALE

CAPITOLO/PARAGRAFO . . . . . PAGINA

3-3

C A P I T O L O 1

INTRODUZIONE

1.1 Il progetto

In questo progetto è stato realizzato un rover
Un rover è un piccolo veicolo dotato di almeno 3 ruote in grado di muoversi facilmente in ogni ambiente. Date le sue ridotte dimensioni, spesso, non necessita di una persona a bordo, viene infatti controllato da remoto o, in certi casi, si muove in maniera autonoma.
In questo caso un piccolo veicolo con 4 ruote. La particolarità di questo progetto consiste nel controllare i movimenti del rover attraverso il movimento di uno smartphone(iPhone, iPod, android) o di un tablet (iPad).

1.2 Motivo della scelta del progetto

I motivi principali della scelta di questo progetto sono 2:

1)Si tratta di un progetto che utilizza tecnologie moderne (smatphone) e che quindi avrà maggior successo in un futuro non troppo remoto

2) Utilizza conoscenze di molte materie studiate durante il triennio:

-Informatica: per quanto riguarda la parte di programmazione, quasi tutti i linguaggi di programmazione usati sono stati studiati a scuola

-Sistemi: per quanto riguarda la parte di trasmissione dei dati

-Elettronica: per quanto riguarda l'elettronica base (studiata durante il terzo anno) e per quanto riguarda l'elettronica più complessa (come ad esempio il pilotaggio di motori in corrente continua)

C A P I T O L O 2

DESCRIZIONE TECNICA

2.1 Obiettivi

2.1.1 Assemblaggio

2.1.1.1 assemblaggio dello scheletro del rover (struttura in metallo e motori) acquistato in kit

2.1.1.2 sistemazione interno:

-Sistemazione batterie e test dei motori

-Sistemazione di un modulo PWM(pulse width modulation) per il controllo dei motori (controllo di due coppie di motori – destra e sinistra)

-Stemazione di una scheda I/O ethernet con protocollo MODBUS

2.1.1.3 Sistemazione esterno:

-Sistemazione di un mini computer con processore ARM

-Sistemazione di un router WI-FI

2.1.1.4 Collegamenti: cavi, interruttori e condensatori

2.1.2 Sviluppo software

2.1.2.1 Controllo rover (obbiettivo principale)

-Creazione di un'app multi piattaforma in flash per acquisire i movimenti dello smartphone/tablet

-Creazione di un programma PHP che riceve i dati inviati dallo smartphone/tablet(attraverso il protocollo HTTP) e converte l'inclinazione del cellulare in percentuale di potenza dei motori

-Creazione di un demone in linguaggio C che acquisisce i dati ricevuti dal programma PHP (attraverso una socket) e li invia (formattandoli correttamente) tramite un'interfaccia seriale al modulo PWM

2.1.2.2 “Telemetria” rover (obiettivo secondario)

-Creazione di un demone in linguaggio C che acquisisce i dati della periferica di I/O modubus e li invia ad una pagina PHP attraverso una socket

-Creazione di una pagina PHP che visualizza solo alcuni dati formattando nella pagina di output i vari dati

-Visualizzazione tensione delle batterie

-Visualizzazione dell'inclinazione corrente dello smartphone/tablet (senza utilizzare la periferica MODBUS ma interfacciandosi direttamente con la pagina PHP che riceve)

NOTA

Altre funzioni di telemetria possono facilmente essere implementate (come ad esempio sensori di posizione, di prossimità ad oggetti, ecc) ma in questo progetto l'obiettivo era solo mostrare le potenzialità della periferica MODBUS

2.2 Schema hardware del progetto

2.2.1 sistemazione componenti

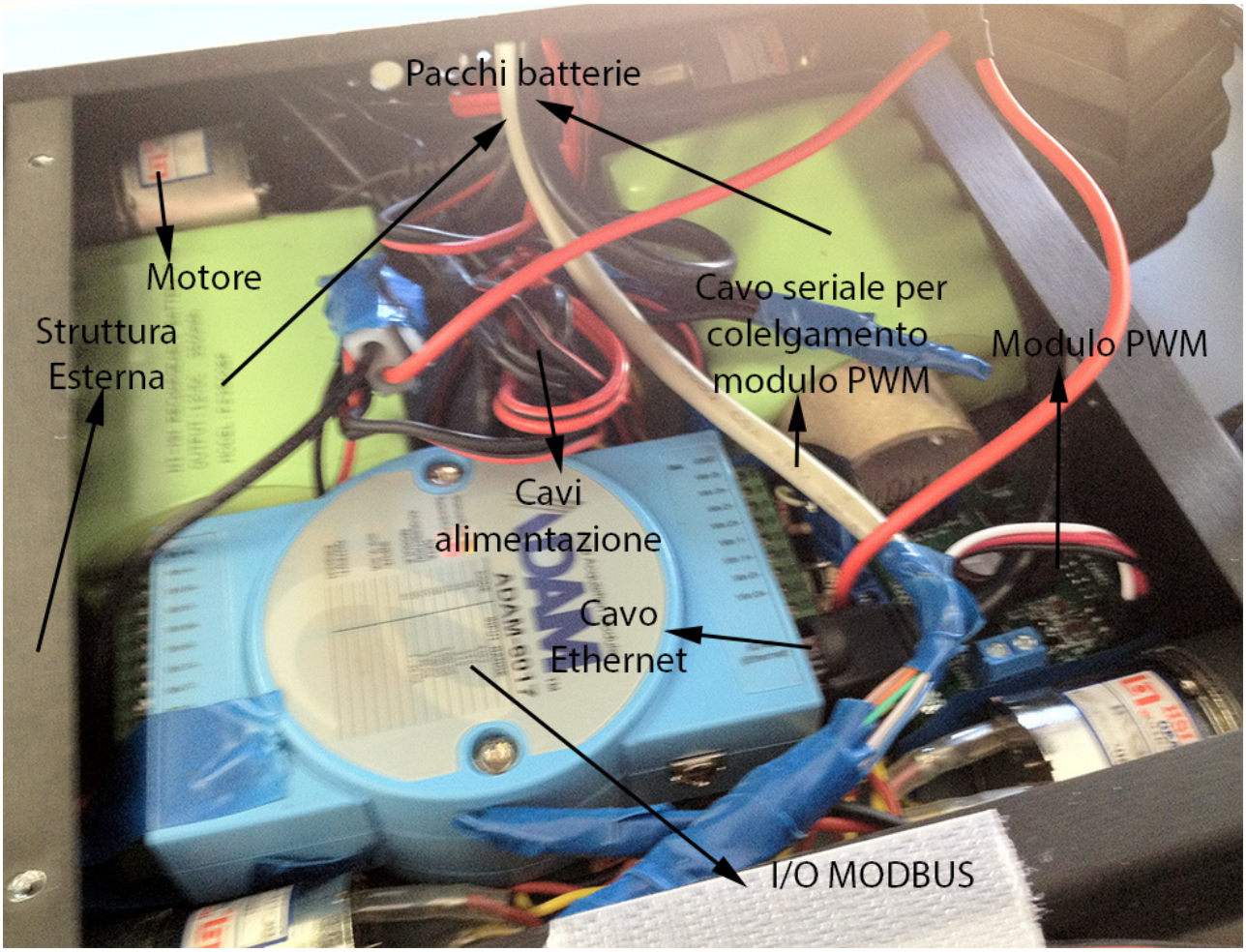

Figura 2-1
Interno rover

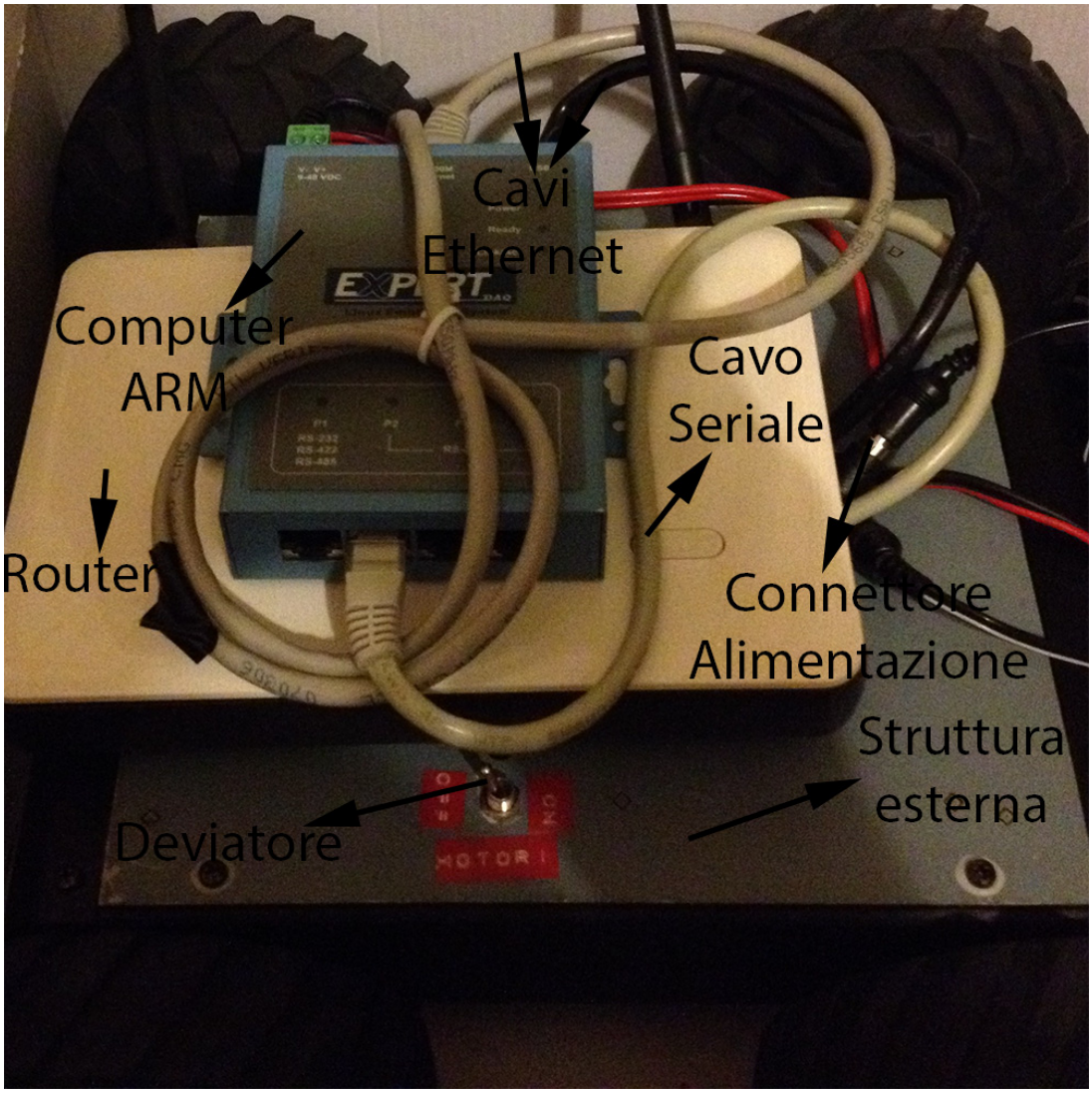

Figura 2-2
Esterno rover

2.2.2 Collegamenti

2.2.2.1 Router

Il router è collegato con l'alimentazione e tramite dei cavi ethernet è connesso al computer ARM e al modulo MODBUS

2.2.2.2 Computer ARM

Il computer ARM è connesso all'alimentazione, al router al modulo PWM(tramite un'interfaccia seriale ma utilizzando un cavo etehrnet standard)

2.2.2.3 Modulo MODBUS

Il modulo MODBUS è connesso all'alimentazione, al router e alle batterie in modo da poter misurare la tensione di esse.

2.2.2.4 Modulo PWM

Il modulo WM è connesso all'alimentazione e al computer ARM

NOTA

Il router in realtà è anche connesso allo smartphone tramite un ponte radio WI-FI

2.3 Schema logico del progetto

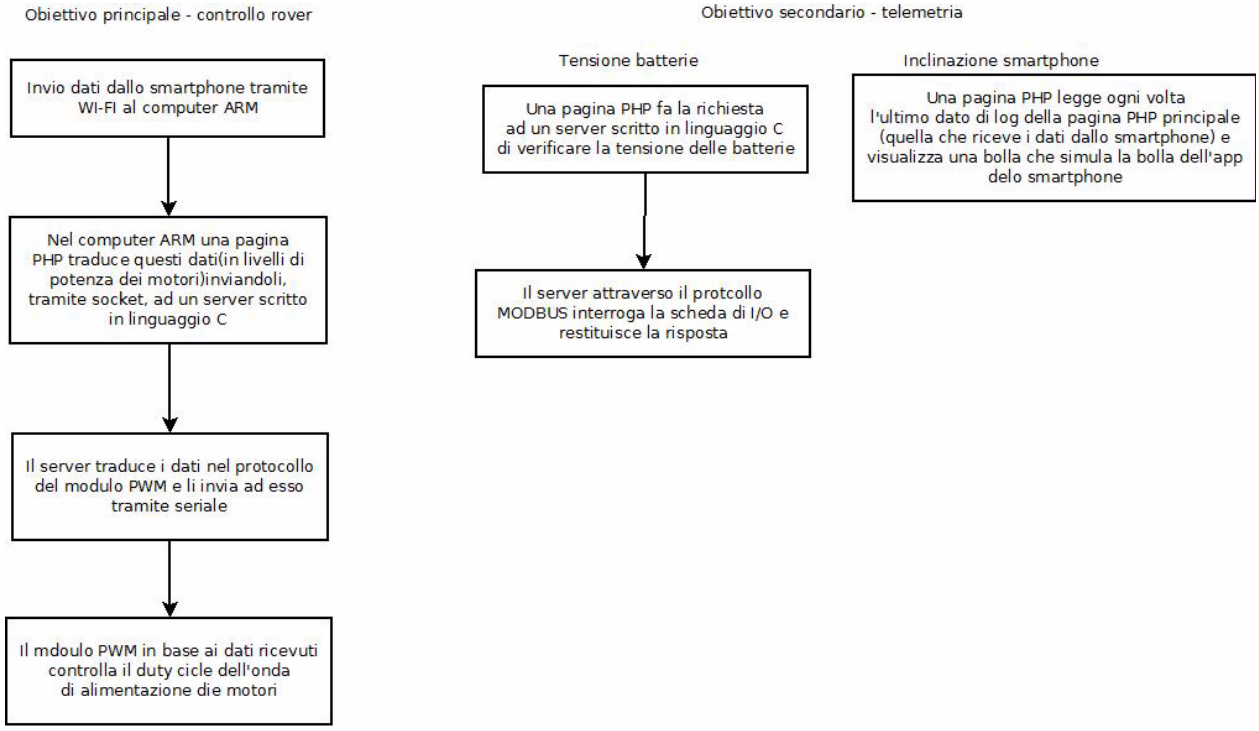

Figura 2-3
Schema logico

C A P I T O L O 3

HARDWARE

3.1 Scheletro rover

Figura 3-1
Scheletro rover

Lo scheletro del rover è stato comprato in kit dato che la sua costruzione non è oggetto del progetto.
Lo scheletro scelto presenta quattro motori indipendenti, due per lato; ma i motori di uno stesso lato vengono controllati insieme, sono stati infatti collegati in parallelo.

3.2 Pacchi batterie ricaricabili

Figura 3-2
Un pacco batterie simile a quello utilizzato nel rover

Tabella 3-1

Tipo	NI-MH (nichel- metallo idruro)
Tensione di utilizzo	12V
Capacità	3500mAh

Due pacchi di batterie, una per i motori(modulo PWM incluso) e una per i componenti elettronici. Così si hanno due impianti indipendenti:

- L'elettronica non risente dell'instabilità della tensione dovuta al consumo dei motori
- Possibilità di sapere se la tensione delle batterie dei motori è bassa prima che sia troppo tardi. Infatti anche se la tensione è bassa i motori riescono a funzionare lo stesso, mentre i componenti elettronici non riescono a funzionare correttamente; così con una batteria separata per essi, i componenti elettronici riescono a funzionare correttamente e riescono a controllare i motori, anche quando la batteria dedicata ai secondi è scarica.

3.3 Modulo PWM

Figura 3-3
Modulo PWM Pololu TReX

Modulo per il controllo dei motori in corrente continua. Permette non solo di regolare la velocità ma permette anche di: effettuare l'inversione di marcia, effettuare un'accelerazione, rilevare la corrente assorbita dai motori istantaneamente, ecc

3.3.1 Controllo PWM con inversione di marcia

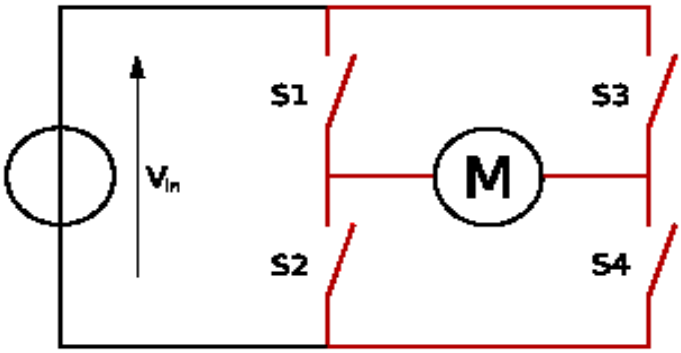

Figura 3-4
Logica PWM

Il controllo in PWM avviene modificando il duty cicle del segnale di uscita che alimenta i motori. Vengono attivati (ripetutamente con una frequenza che va dai 10 ai 20KHz) gli interruttori S1 e S4 o S3 e S2, a seconda del verso di rotazione scelto. Il controllo PWM necessita quindi di una scheda di controllo, questi interruttori nella realtà sono infatti dei transistor, visto che devono essere pilotati da un circuito elettronico.

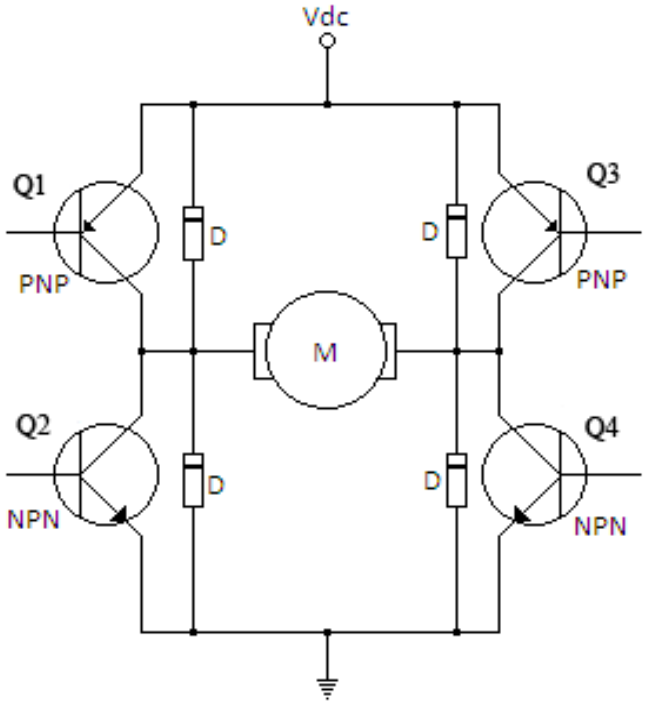

Figura 3-5
Circuito di controllo a logica PWM

Sono presente dei diodi, detti diodi di ricircolo, perché a causa del comportamento induttivo del motore ai suoi capi si verrebbe a creare una tensione altissima, quando si spegne, la quale brucerebbe i transistor.

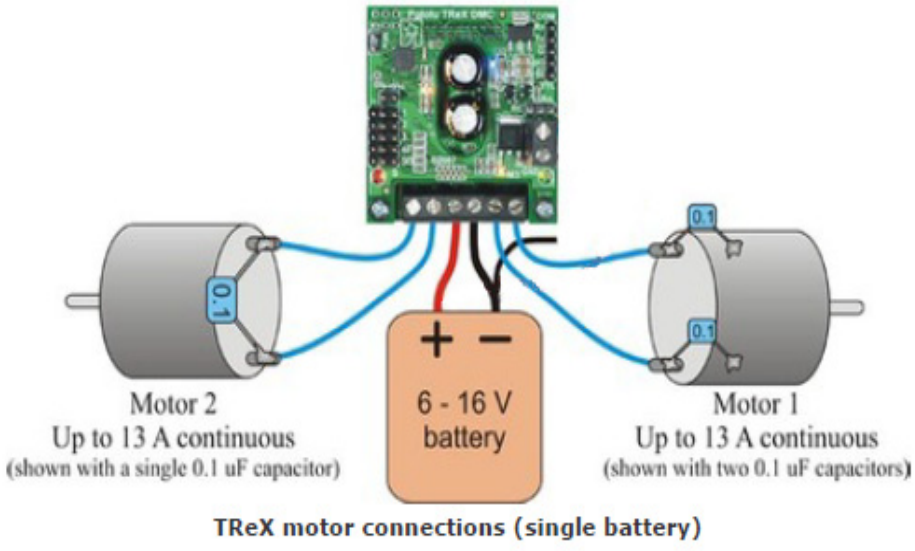

Figura 3-6
Collegamento motori al modulo PWM

3.3.2 Controllo del modulo PWM da PC

Il modulo PWM si interfaccia con due standard principali:

- RC analogico
- Seriale (RS-232 o TTL)

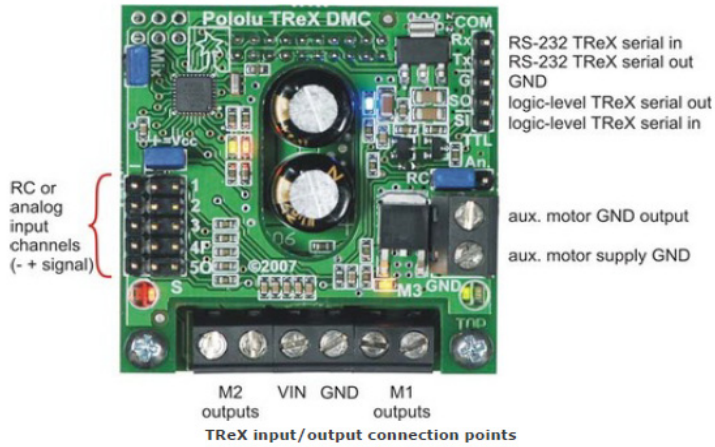

Figura 3-7
Connessioni scheda PWM

Per la connessione con il PC, ovviamente, la scelta migliore è la seriale RS232.

Figura 3-8
Collegamento seriale

Infatti tramite l'uscita seriale del PC si controlla facilmente il modulo PWM attraverso un suo protocollo

3.4 I/O MODBUS

Figura 3-9
ADAM 6017

Si tratta di un dispositivo in grado(a seconda delle versioni) di ricevere input digitali e analogici(sia input di corrente che di tensione) e fornire output digitali e analogici. Il vantaggio di questa periferica è che utilizza un'interfaccia di rete ethernet e un protocollo standard, il MODBUS(un protocollo industriale); è quindi possibile controllarla, facilmente, da diversi pc connessi alla stessa rete.

3.5 Mini computer ARM

Figura 3-10

EX9486-L

Si tratta di un computer con architettura ARM, con sistema operativo GNU/linux. Questa combinazione presenta diversi vantaggi:

- bassi consumi del processore ARM
- vantaggi derivanti dall'utilizzo di un sistema operativo multitasking come GNU/linux9486
- Interfacce a basso livello secondo lo standard POSIX
- Possibilità di utilizzare quasi tutte le librerie scritte in C per un comune sistema GNU /linux su architettura intel

Date le ridotte dimensioni questo computer non presenta un'interfaccia video, l'unico modo per controllarlo è tramite l'interfaccia di rete (attraverso i protocolli telnet e FTP).
Oltre all'interfaccia di rete sono presenti diverse interfacce, come ad esempio delle porte seriali e delle porte USB, le quali permettono di interfacciarsi a basso livello con periferiche esterne, ad esempio la porta seriale RS-232 è stata utilizzata per controllare il modulo PWM.

3.6 Router WI-FI

Figura 3-11
Router WI-FI Sitecom

Si tratta di un normalissimo router con interfaccia ethernet e WI-FI

3.7 Alimentazione

Figura 3-12
Alimentatore simile a quelli utilizzati nel rover

Per alimentare il Rover si utilizzano le batterie (caricate con due caricabatterie separati), quando il rover è in movimento, oppure si può utilizzare un'alimentazione esterna per i componenti elettronici, utile quando si deve programmare il computer

3.8 Smartphone

Figura 3-13
iPhone

Si tratta di un telefono cellulare con funzionalità e interfacce evolute, andando oltre al classico concetto di telefonino con la sola funzione di poter effettuare chiamate. Infatti spesso implementa:

- la possibilità di connettersi facilmente ad internet
- la possibilità di inviare e ricevere email
- la possibilità di poter consultare il web
- la possibilità di poter installare applicazioni di terze parti

3.9 Componenti vari

3.9.1 Cavi

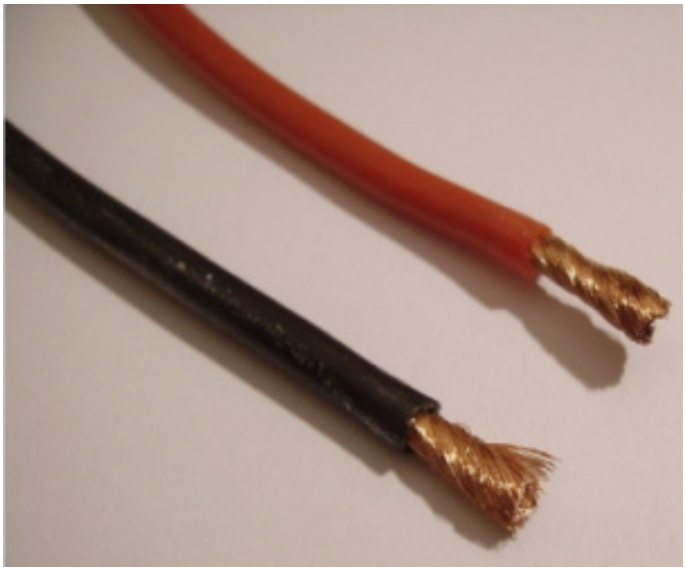

Figura 3-14
Esempio di cavi di alimentazione

Sono stati utilizzati dei normalissimi cavi di alimentazione e dei cavi ethernet (utilizzati anche per l'interfaccia seriale)

3.9.2 Connettori

Figura 3-15
Esempio di connettore utilizzato

Sono stati utilizzati dei connettori per collegare i vari alimentatori alle batterie e collegare esse ai circuiti del rover.

3.9.3 Deviatori

Figura 3-16
Deviatore

Sono stati utilizzati come interruttori di alimentazione per i due circuiti principali del rover e per cambiare la sorgente di alimentazione dei componenti elettronici (batteria o esterna)

3.9.4 Resistori

Figura 3-17
Resistore

I resistori sono stati utilizzati per aumentare il range dei valori misurabili dal dispositivo I/O, infatti quest'ultimo prevede una scala che fa da -10:+10 così utilizzando un partitore di tensione si aumenta il range a -14.5:14.5. Così facendo si rileva sempre un valore di tensione compreso nell'intervallo -10:+10, ma lo si può facilmente convertire nel valore originale con la seguente formula

$$V_{or} = \frac{V_i}{22} \cdot 32$$

Figura 3-18

:

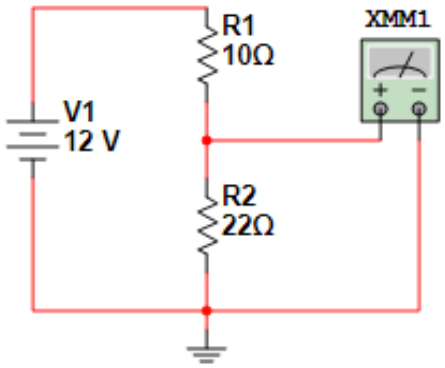

Figura 3-19
Partitore di tensione utilizzato

3.9.5 Condensatori

Figura 3-20
Condensatore

Sono stati utilizzati due condensatori da 2,2 mF collegati in parallelo al circuito di alimentazione dei componenti elettronici per stabilizzare la tensione quando si cambia sorgente di alimentazione

C A P I T O L O 4

INTERFACCIAMENTO HW-SW

4.1 Ethernet

Figura 4-1
Connettore RJ-45

Si tratta di un protocollo del livello 2 della pila ISO/OSI, basato sul protocollo CSM/CD
Permette di collegare facilmente in una LAN diverse apparecchiature tra loro diverse, originariamente le connessioni avvenivano su di un cavo coassiale condiviso tra i vari computer, oggi invece avvengono attraverso una topologia a stella con un switch al "centro di tutto". Per questo motivo sul rover è stato utilizzato un router che ha anche la funzione di switch.
Le comunicazioni avvengono con il protocollo TCP/IP (infatti a bordo del rover è presente anche un server DHCP), quindi attraverso delle socket ad alto livello.

4.2 Protocollo MODBUS

Si tratta di un protocollo, per uso industriale, nato nel 1979 su interfaccia seriale. Serviva infatti a connettere ad un'unità master diverse periferiche(ognuna identificata da un id univoco) collegate ad un unico cavo RS-485. Il protocollo è abbastanza semplice, infatti l'unità master invia semplici comandi o di scrittura o di lettura ad una specifica periferica, identificata dal proprio id.
Oggi questo protocollo è stato adattato alla rete IP(MODBUS over IP), infatti per controllare la scheda di I/O MODBUS il computer utilizza delle librerie apposite che sfruttano il protocollo IP.

NOTA

Il protocollo MODBUS viene utilizzato solo per la lettura dei dati della tensione,l'obiettivo infatti è mostrare le potenzialità di questo protocollo.
Il vantaggio dell'utilizzo di questo protocollo è infatti il poter utilizzare queste periferiche allo stesso modo da qualsiasi unità master.

4.3 WI-FI

Figura 4-2
Esempio di AP

Si tratta di un protocollo del livello 2 della pila ISO/OSI, basato sul protocollo CSM/CA(meno efficiente del CSMA/CD). Questo protocollo utilizza un canale wirless (senza fili) ma è basato su una topologia a stella, infatti le varie apparecchiature si connettono ad un Acces Point(AP), il quale permette di connettersi alla WLAN.

NOTA

Per ovvie ragioni di spazio è stata scelte una soluzione "all in", la quale implementa sia le funzioanlità di server DHCP, che di switch ethernet nonché di AP WI-FI

4.4 Interfaccia seriale

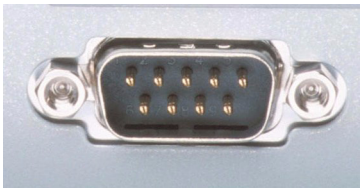

Figura 4-3
Connettore seriale RS-232

La trasmissione seriale è una modalità di comunicazione tra dispositivi digitali nella quale le informazioni sono comunicate una di seguito all'altra e giungono sequenzialmente al ricevente nello stesso ordine in cui le ha trasmesse il mittente. La tipologia di seriale usata è la RS-232.
Per sincronizzare i dati vengono usati i bit di start e di stop. Opera a basso livello e a bassa velocità, ma permette di interfacciarsi facilmente con semplici apparecchiature evitando di utilizzare interfacce più complesse come il USB. Operando a basso livello possono verificarsi degli errori di comunicazione che possono essere risolti con un semplice controllo di checksum(come ad esempio il CRC), ma in questo caso non è necessario eseguire alcun controllo perché i dati vengono trasmessi a breve distanza gli uni dagli altri.

C A P I T O L O 5

SOFTWARE

5.1 Controllo da smartphone

5.1.1 Programma per smartphone

Il programma per lo smartphone è stato scritto con flash per due principali motivi:

- semplicità di creazione di un'interfaccia grafica
- compatibilità tra diversi dispositivi: iPhone e android

Nel programma ogni 100 millisecondi, con un ciclo continuo vengono raccolti nuovi dati riguardo l'inclinazione del cellulare. Ogni volta che vengono raccolti nuovi dati si posiziona una bolla sullo schermo che indica l'inclinazione e si inviano attraverso una POST i dati ad una pagina PHP che si trova a bordo del computer sul rover.

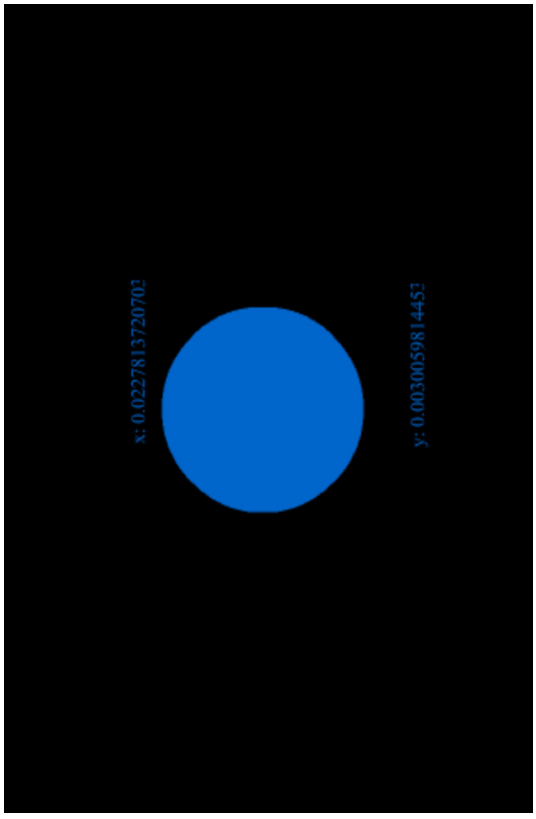

Figura 5-1
Programma per iPhone

il codice action script è il seguente:

```
import flash.sensors.Accelerometer;
import flash.events.AccelerometerEvent;

var my_acc:Accelerometer = new Accelerometer();
my_acc.setRequestedUpdateInterval(100);


my_acc.addEventListener(AccelerometerEvent.UPDATE, onAccUpdate);
```

```
function onAccUpdate(e:AccelerometerEvent):void {
     bolla.x = (-e.accelerationX*160+160);
     bolla.y = (e.accelerationY*240+240);
     testo_x.text = "x: "+String(e.accelerationX);
     testo_y.text = "y: "+String(e.accelerationY);
     if (bolla.x < 0) {
          bolla.x = 0;
     }
     else if (bolla.x > stage.stageWidth) {
          bolla.x = stage.stageWidth;
     }

     if (bolla.y < 0) {
          bolla.y = 0;
     }
     else if (bolla.y > stage.stageHeight) {
          bolla.y = stage.stageHeight;
     }

     //invio dati

     var scriptRequest:URLRequest = new URLRequest("http://192.168.0.102/
robot_flash.php");
    var scriptLoader:URLLoader = new URLLoader();
    var scriptVars:URLVariables = new URLVariables();

    scriptLoader.addEventListener(Event.COMPLETE, handleLoadSuccessful);
    scriptLoader.addEventListener(IOErrorEvent.IO_ERROR, handleLoadError);

    scriptVars.posx = String(e.accelerationX);
     scriptVars.posy = String(e.accelerationY);

    scriptRequest.method = URLRequestMethod.POST;
    scriptRequest.data = scriptVars;

    scriptLoader.load(scriptRequest);

}

    function handleLoadSuccessful($evt:Event):void
    {
        trace("Messaggio inviato.");
    }

    function handleLoadError($evt:IOErrorEvent):void
    {
        trace("Messaggio non inviato.");
    }
```

5.1.2 Programma PHP di ricezione dati

La pagina PHP riceve i dati e tramite l'utilizzo della trigonometria calcola il modulo e la l'inclinazione della "bolla" in un ipotetico piano cartesiano con il centro corrispondente alla posizione della bolla quando lo smartphone non è inclinato.

$$mod = \sqrt{x^2 + y^2}$$
$$inclinazione = \arctan\left(\frac{x}{y}\right)$$

Figura 5-2
Formule modulo e inclinazione

In seguito converte questi valori in livelli di potenza del motore, inserendo però dei range nei quali non convertire proporzionalmente, per esempio intorno al centro c'è un range di 20 gradi di "nullità" in modo da non avere movimenti improvvisi appena si inclina di poco lo smartphone

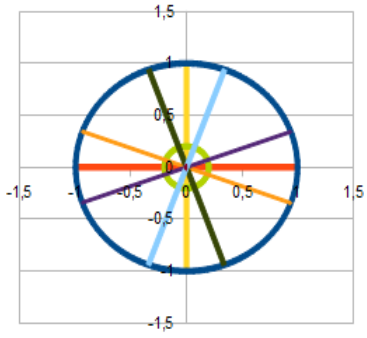

Figura 5-3
Range intorno di "nullità"

Le conversioni sono molto semplici, infatti nel caso in cui il rover debba andare avanti basterà assegnare come potenza dei due motori il modulo.

il codice è il seguente:

```
<?php
header("Cache-Control: no-cache, must-revalidate"); // HTTP/1.1
header("Expires: Mon, 26 Jul 1997 05:00:00 GMT"); // Date in the past
$posx = isset($_POST['posx'])?$_POST['posx']:0;
$posy = isset($_POST['posy'])?$_POST['posy']:0;
$str_file = $_POST['posx']."|".$_POST['posy']."\n";

$service_port = 1745;
$address = "127.0.0.1";
/* Crea un socket TCP/IP. */
```

```
$socket = @socket_create(AF_INET, SOCK_STREAM, 0) or die("socket_create() fallito: motivo: " . socket_strerror($socket) . "\n");

$result = @socket_connect($socket, $address, $service_port) or die("socket_connect() fallito.\nMotivo: ($result) " . socket_strerror($result) . "\n");


$potenza = sqrt($posx*$posx+$posy*$posy);//teorema di pitagora

$potenza = $potenza<0.2?0:($potenza-0.2)*5/4;//range nullo del 20% intorno allo 0

$potenza = $posx<0?(-$potenza):$potenza;//potenza negativa, dato che il teorema di pitagora da sempre un risultato positivo

$angolazione = $posy==0?($posx>0?90:270):(180/3.14*atan($posx/(-$posy)));//conversione in gradi. tan = sin/cos -> tan = posx/posy (sschemro al contrario) -> radianti = arcotan tan

$angolazione = ($posy>0)?(180+$angolazione):(($posx<0 && $angolazione != 270)?(360+$angolazione):$angolazione);//angolazione da 0 a 360, perché se no l'angolazione risulterebbe uguale al valore dell'arco tangente, che nel secondo e quarto quadrante è negativa

$angolazione_m = $angolazione>180?($angolazione-180):$angolazione;// angolazione, due serie,  da 0 a 180

//sinistra range 20°

if ($angolazione_m>160){

     $sinistro = -$potenza;

     $destro = $potenza;

}else

//destra range 20°

if ($angolazione_m<20){

     $sinistro = $potenza;

     $destro = -$potenza;

}else

//diagonale sinistra

if($angolazione_m>110){

     $sinistro = $potenza*(($angolazione_m-110)/50);

     $destro = $potenza;

}else

//diagonale destra

if($angolazione_m<70){

     $sinistro = $potenza;

     $destro = $potenza*((70-$angolazione_m)/50);

}else{

//dritto

     $sinistro = $potenza;

     $destro = $potenza;

}



$in = $sinistro."|".$destro;

$str_file .= $in;
```

```
file_put_contents("pos.txt",$str_file);

@socket_write($socket, $in, strlen ($in)) or die("impossibile scrivere sul socket \n");

@socket_close($socket) or exit();

?>
```

5.1.3 Programma C di invio dati al modulo PWM

Questo programma non fa altro che creare una socket e inizializzare la porta seriale quando si avvia, con l'avvio del PC.in seguito non fa altro che inviare i dati ricevuti dalla socket sulla seriale. In teoria anche il programma PHP stesso poteva scrivere sulla seriale ma così ci sono diversi vantaggi:

- Un'unica connessione sempre attiva
- Possibilità di poter implementare facilmente funzioni di controllo, come ad esempio funzioni di controllo che stoppano automaticamente il rover se questo non riceve più dati per x secondi

Il codice è il seguente:

```
#include <fcntl.h>/* File control definitions */
#include <stdio.h>/* Standard input/output */
#include <string.h>
#include <termio.h>/* POSIX terminal control definitions */
#include <sys/time.h>/* Time structures for select() */
#include <unistd.h>/* POSIX Symbolic Constants */
#include <errno.h>/* Error definitions */
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/ioctl.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include<termios.h>
#include "modbus_tcp.h"
#include <netinet/in.h>
#include <stdlib.h>
#define PERCORSO "/dev/ttyS2"
#define VELOCITA B19200
#define MAX 100

void cambia(int num, int val, int fd);
void inverti(int num, int fd);
int CreaSocket(int Porta);
void ChiudiSocket(int sock);
```

```
int elaborascelta(char *str, int fd);

int main(int argc, char *argv[])
{

    int portFd = 0;

    struct termios term;

      int i;

      for(i=5;i>0;i--){
          printf("Attesa iniziale...(%d)\a\n",i);
          sleep(1);
      }

    //SERIALE
    if((portFd = open(PERCORSO,O_RDWR | O_NOCTTY | O_NDELAY))<0)
         return -1;

    if(tcgetattr(portFd, &term)== -1)
         return -1;

    if(cfsetispeed(&term, VELOCITA) == -1)
         return -1;

    if(cfsetospeed(&term, VELOCITA) == -1)
         return -1;

    cfmakeraw(&term);

    if(tcflush(portFd, TCIOFLUSH) == -1)
         return -1;

    if(tcsetattr(portFd, TCSANOW, &term) == -1)
         return -1;


  char  buffer[512];
  int DescrittoreSocket,NuovoSocket;
  int exitCond=0;
  int Quanti;
```

```
  DescrittoreSocket=CreaSocket(1745);
  printf("Server: Attendo connessioni...\n");
  while (!exitCond)
  {
    //Test sul socket: accept non blocca, ma il ciclo while continua
    //l'esecuzione fino all'arrivo di una connessione.
    if ((NuovoSocket=accept(DescrittoreSocket,0,0))!=-1)
    {
      //Lettura dei dati dal socket (messaggio ricevuto)
      if ((Quanti=read(NuovoSocket,buffer,sizeof(buffer)))<0)
      {
         printf("Impossibile leggere il messaggio.\n");
         ChiudiSocket(NuovoSocket);
      }
      else
      {
         //Aggiusto la lunghezza...
         buffer[Quanti]=0;
         //Elaborazione dati ricevuti
         if (strcmp(buffer,"exit")==0)
              exitCond=1;
         else {
              elaborascelta(buffer, portFd);
              }

      }
      //Chiusura del socket temporaneo
      ChiudiSocket(NuovoSocket);
    }
  }
  //Chiusura del socket
  ChiudiSocket(DescrittoreSocket);
  printf("Server: Terminato.\n");


  printf("uscita in corso....\n");

  return 0;
}
```

```
int CreaSocket(int Porta)
{
  int sock,errore;
  struct sockaddr_in temp;

  //Creazione socket
  sock=socket(AF_INET,SOCK_STREAM,0);
  //Tipo di indirizzo
  temp.sin_family=AF_INET;
  temp.sin_addr.s_addr=INADDR_ANY;
  temp.sin_port=htons(Porta);

  //Il socket deve essere non bloccante
  errore=fcntl(sock,F_SETFL,O_NONBLOCK);

  //Bind del socket
  errore=bind(sock,(struct sockaddr*) &temp,sizeof(temp));

  errore=listen(sock,7);

  return sock;
}

void ChiudiSocket(int sock)
{
  close(sock);
  return;
}


int elaborascelta(char *str, int fd){
    char *p;
    float mots, motd;
    unsigned char buffer[10];

    p = strtok(str, "|");
    mots = atof(p);
    p = strtok(NULL, "|");
    motd = atof(p);
    printf("sinistro: %d, destro: %d\n",(int) abs(mots*127), (int) abs(motd*127));

   buffer[0] = 0xD0|((mots>=0?2:1)*4)|(motd>=0?1:2);
```

```
    buffer[1] = (unsigned char) abs(mots*127);
    buffer[2] = (unsigned char) abs(motd*127);
    write(fd, buffer, 3);
    return 1;
}
```

5.2 Programma di visualizzazione inclinazione cellulare

Questo programma non è altro che un mix di PHP e javascript, la sua unica funzione è quella di leggere i valori da un file e visualizzare una bolla sullo schermo alla posizione indicata da questi valori.

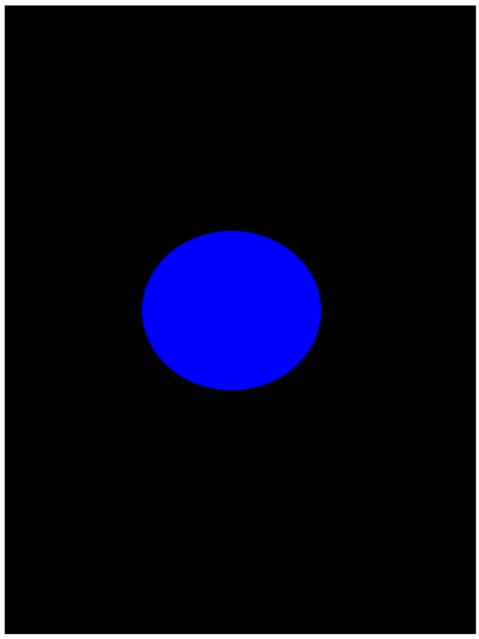

Figura 5-4

Il codice è il seguente:

```
<?php
//bisogna per forza usare il php per via della cache
header("Cache-Control: no-cache, must-revalidate"); // HTTP/1.1
header("Expires: Mon, 26 Jul 1997 05:00:00 GMT"); // Date in the past
if(isset($_GET['dati'])){
     print date("d/m/Y H:s:i",filemtime("pos.txt")).”\n”.file_get_contents("pos.txt");
     die();
}
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/
```

```
xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Bolla</title>
<script>
// funzione per assegnare l'oggetto XMLHttpRequest
// compatibile con i browsers più recenti e diffusi
function assegnaXMLHttpRequest() {

// lista delle variabili locali
var
 // variabile di ritorno, nulla di default
 XHR = null,

 // informazioni sul nome del browser
 browserUtente = navigator.userAgent.toUpperCase();


 // browser standard con supporto nativo
 // non importa il tipo di browser
 if(typeof(XMLHttpRequest) === "function" || typeof(XMLHttpRequest) === "object")
  XHR = new XMLHttpRequest();

 // browser Internet Explorer
 // è necessario filtrare la versione 4
 else if(
  window.ActiveXObject &&
  browserUtente.indexOf("MSIE 4") < 0
 ) {

  // la versione 6 di IE ha un nome differente
  // per il tipo di oggetto ActiveX
  if(browserUtente.indexOf("MSIE 5") < 0)
   XHR = new ActiveXObject("Msxml2.XMLHTTP");

  // le versioni 5 e 5.5 invece sfruttano lo stesso nome
  else
   XHR = new ActiveXObject("Microsoft.XMLHTTP");
 }

 return XHR;
```

```
}

function ajax(){
  var ajax = assegnaXMLHttpRequest();
  if(ajax) {
   ajax.open("get","bolla.php?dati",true);
   ajax.onreadystatechange = function(){
           if(ajax.readyState == 4 && ajax.status == 200){

                var dati = ajax.responseText.split("\n");
                document.getElementById("data").innerHTML = "data ultima ricezione dati: "
+ dati[0];
                document.getElementById("posizione").innerHTML = "posizione bolla: " +
dati[1];
                document.getElementById("potenza").innerHTML = "livelli di potenza del mo-
tore: " + dati[2];
                dati = dati[1];
                dati = dati.split("|");
                document.getElementById("bolla").style.left = (99-dati[0]*160)+"px";
                document.getElementById("bolla").style.top = (179+dati[1]*240)+"px";
                window.setTimeout("ajax()",50);
           }
   }
   ajax.send(null);
 }
}

onload = function() {
     ajax();
}
</script>

<style>
.dati{
     clear:right;
}

#bolla_c{
     position:relative;
     width:320px;
     height:480px;
     background-color:#000;
     overflow:hidden;
```

```
	clear:right;
}

#bolla{
	position:relative;
	left:99px;
	top:179px;
	width:122px;
	height:122px;

}

#lettura{
	position:relative;
	clear:right;
}

#desc{
	font-size:9px;
	position:relative;
	clear:right;
}
</style>
</head>


<body>
<div id="bolla_c"><img id="bolla" src="bolla.png"/></div>
<div id="data" class="dati">xxx</div>
<div id="posizione" class="dati">xxx</div>
<div id="potenza" class="dati">xxx</div>
<div id="desc">N.B. Il segno dell'asse x è invertito</div>
</body>
</html>
```

5.3 Programma di visualizzazione tensione

Anche qui analogamente al programma principale è presente un server C, questo server però all'avvio invece di inizializzare la seriale inzializza la connessione MODBUS.

Per visualizzare i dati analogamente al programma precedente è presente un mix di PHP e javascript.

I codici sono i seguenti:

C:

```
#include <fcntl.h>/* File control definitions */
#include <stdio.h>/* Standard input/output */
#include <string.h>
#include <termio.h>/* POSIX terminal control definitions */
#include <sys/time.h>/* Time structures for select() */
#include <unistd.h>/* POSIX Symbolic Constants */
#include <errno.h>/* Error definitions */
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/ioctl.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include "modbus_tcp.h"
#include <netinet/in.h>
#include <stdlib.h>

int CreaSocket(int Porta);
void ChiudiSocket(int sock);
void SpedisciMessaggio(int sock, char* Messaggio);

int main(int argc, char *argv[])
{
  int fd;
  /* setup communication parameters */
  if(argc<2){
            printf("nessun indirizzo IP passato\n");
            return -1;
  }
  printf("START\n");
  fd = set_up_tcp(argv[1], 502);
  printf("fd: %d\n", fd);
  char buffer2[1024];
  int dest;

   char  buffer[512];
  int DescrittoreSocket,NuovoSocket;
  int exitCond=0;
  int Quanti;
```

```
  DescrittoreSocket=CreaSocket(1746);
  printf("Server: Attendo connessioni...\n");
  while (!exitCond)
  {
    //Test sul socket: accept non blocca, ma il ciclo while continua
    //l'esecuzione fino all'arrivo di una connessione.
    if ((NuovoSocket=accept(DescrittoreSocket,0,0))!=-1)
    {
      //Lettura dei dati dal socket (messaggio ricevuto)
      if ((Quanti=read(NuovoSocket,buffer,sizeof(buffer)))<0)
      {
         printf("Impossibile leggere il messaggio.\n");
         ChiudiSocket(NuovoSocket);
      }
      else
      {
         //Aggiusto la lunghezza...
         buffer[Quanti]=0;
         //Elaborazione dati ricevuti
         if (strcmp(buffer,"exit")==0)
              exitCond=1;
         else {
             read_input_registers_tcp( 1, atoi(buffer), 2, &dest, sizeof(dest), fd );
             sprintf(buffer2,"%f",((((dest-0x7FFF)/((float)0x7FFF))*10)/22)*32);
             SpedisciMessaggio(NuovoSocket, buffer2);
         }
      //Chiusura del socket temporaneo
      ChiudiSocket(NuovoSocket);
    }
  }
}
  //Chiusura del socket
  ChiudiSocket(DescrittoreSocket);
  printf("Server: Terminato.\n");

  return 0;
}

int CreaSocket(int Porta)
{
  int sock,errore;
  struct sockaddr_in temp;
```

```
  //Creazione socket
  sock=socket(AF_INET,SOCK_STREAM,0);
  //Tipo di indirizzo
  temp.sin_family=AF_INET;
  temp.sin_addr.s_addr=INADDR_ANY;
  temp.sin_port=htons(Porta);

  //Il socket deve essere non bloccante
  errore=fcntl(sock,F_SETFL,O_NONBLOCK);

  //Bind del socket
  errore=bind(sock,(struct sockaddr*) &temp,sizeof(temp));
  errore=listen(sock,7);

  return sock;
}

void ChiudiSocket(int sock)
{
  close(sock);
  return;
}

void SpedisciMessaggio(int sock, char* Messaggio)
{
  printf("Client: %s\n",Messaggio);

  if (write(sock,Messaggio,strlen(Messaggio))<0)
  {
    printf("Impossibile mandare il messaggio.\n");
    ChiudiSocket(sock);
    exit(1);
  }
  printf("Messaggio spedito con successo.\n");
  return;
}
```

PHP

```
<?php
header("Cache-Control: no-cache, must-revalidate"); // HTTP/1.1
header("Expires: Mon, 26 Jul 1997 05:00:00 GMT"); // Date in the past
if(isset($_GET['pin'])){
      $service_port = 1746;
      $address = "127.0.0.1";
      $socket = @socket_create(AF_INET, SOCK_STREAM, 0) or die("socket_create() fallito: motivo: " . socket_strerror($socket) . "\n");
      $result = @socket_connect($socket, $address, $service_port) or die("socket_connect() fallito.\nMotivo: ($result) " . socket_strerror($result) . "\n");
      $msg = $_GET['pin'];
      //DIRE SE CI SONO ERRORE CHE FALLISCE LA SOCKET AD ESMEPIO
      @socket_write($socket, $msg, strlen($msg)) or die("impossibile scrivere sul socket \n");
      print socket_read($socket,20);
      @socket_close($socket) or exit();
      die();
}
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Tensione</title>
<script>
// funzione per assegnare l'oggetto XMLHttpRequest
// compatibile con i browsers più recenti e diffusi
function assegnaXMLHttpRequest() {

// lista delle variabili locali
var
 // variabile di ritorno, nulla di default
 XHR = null,

 // informazioni sul nome del browser
 browserUtente = navigator.userAgent.toUpperCase();


 // browser standard con supporto nativo
 // non importa il tipo di browser
 if(typeof(XMLHttpRequest) === "function" || typeof(XMLHttpRequest) === "object")
```

```
  XHR = new XMLHttpRequest();

 // browser Internet Explorer
 // è necessario filtrare la versione 4
 else if(
  window.ActiveXObject &&
  browserUtente.indexOf("MSIE 4") < 0
 ) {

  // la versione 6 di IE ha un nome differente
  // per il tipo di oggetto ActiveX
  if(browserUtente.indexOf("MSIE 5") < 0)
   XHR = new ActiveXObject("Msxml2.XMLHTTP");

  // le versioni 5 e 5.5 invece sfruttano lo stesso nome
  else
   XHR = new ActiveXObject("Microsoft.XMLHTTP");
 }

 return XHR;
}

function ajax(pin, num){
  var ajax = assegnaXMLHttpRequest();
  if(ajax) {
   ajax.open("get","tensione.php?pin="+pin,true);
   ajax.onreadystatechange = function(){
           if(ajax.readyState == 4 && ajax.status == 200){
                document.getElementById("tensione"+num).innerHTML = ajax.responseText;
                window.setTimeout("ajax("+pin+",'"+num+"')",500);
           }
   }
   ajax.send(null);
 }
}

onload = function() {
     ajax(1,'1');
     ajax(4,'2');
}
</script>
</head>
```

```
Batteria elettronica: <span id="tensione1">xxx</span>V<br />
Batteria motori: <span id="tensione2">xxx</span>V

<body>
</body>
</html>
```

C A P I T O L O 6

CONCLUSIONI

6.1 Utilizzi attuali

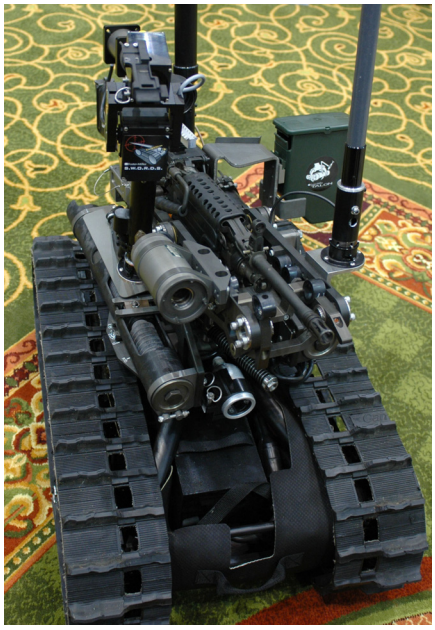

Figura 6-1
Rover da guerra

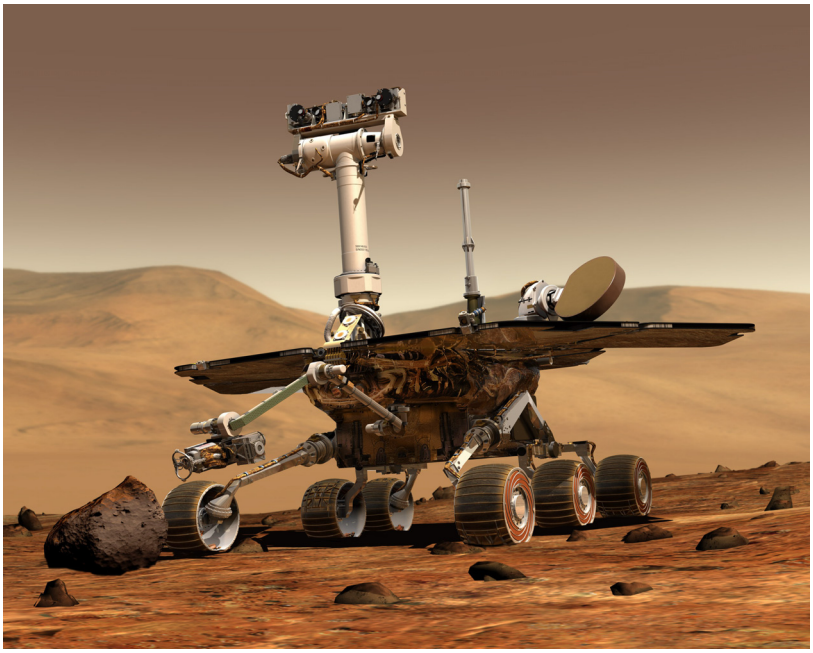

Figura 6-2
Spirit - rover mandato su marte

I rover automatici con ruote non sono stati inventati da molto, basta pensare che il primo rover di questo tipo spedito nello spazio risale solo al 1997.
Il campo dei rover è in continua evoluzione, infatti al momento vengono utilizzati solo per:

- Effettuare esplorazioni su automatiche su marte (analizzando l'ambiente con diversi sensori).

- Permettere ai soldati di non andare in prima linea, vengono infatti usati da essi per diversi ruoli: disinnescare le bombe da remoto, spiare i nemici (tramite dei mini rover), o addirittura in certi casi vengono utilizzati come delle armi portatili (è infatti abbastanza facile montare armi o altri oggetti a bordo di un rover)

6.2 Sviluppi futuri

Il vero sviluppo dei rover deve ancora avvenire, infatti presentano tantissimi vantaggi:

- Possono affrontare molti tipi di terreni e ambienti ostili all'uomo
- Ogni gadget può essere facilmente integrato con essi
- Controllarli è molto facile: infatti basta controllare dei semplici motori in corrente continua
- Presenta molti movimenti possibili: inclusa la rotazione su se stesso (facendo girare una fila di motori in un verso e l'altra nell'altro si ottiene la rotazione su se stesso)

Solo che oggi, purtroppo, i rover dalle persone comuni vengono visti ancora o come un "giocattolo" o come qualcosa di "fantascientifico" (soprattutto riguardo ai rover automatici)

6.3 Considerazioni sul progetto

Gli obiettivi prefissati all'inizio del progetto sono stati tutti raggiunti, non si sono verificati problemi per raggiungerli. gli unici problemi riscontrati riguardano la calibrazione della potenza da dare ai singoli motori, in paritcolare per quanto riguarda il movimento diagonale.

La cosa probabilmente più importante che insegna questo progetto è come generalizzando il più possibile ogni singola funzione si riesca a ottenere sul lungo termine risultati migliori, infatti è grazie a questa generalizzazione che ogni singola parte può essere simulata capendo così come funzione e/o i suoi difetti.

Infine quest'esperienza insegna come da un progetto apparentemente ludico si possa fare tanto, infatti un rover come questo potrebbe avere sensori in più e potrebbe quindi avere diversi campi di utilizzo, è proprio questo il vantaggio dei rover, sono facilmente modificabili per le proprie esigenze. Ovviamente inserire altri sensori (come di prossimità, di temperatura, ecc) a bordo di questo rover non risulterebbe difficile, ma risulterebbe inutile ai fini dell'obiettivi del progetto. L'obbiettivo infatti consiste solo nel mostrare le potenzialità dei rover e dei smartphone, i quali non devono essere necessariamente potenti, tutti i calcoli vengono eseguiti dal rover che fa da server, infatti è proprio in questa direzione che si muove il mercato del software mobile, tante semplici app che rimandano i calcoli complessi ai server.