|  |  |  |  |
| --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | ARDUINO GARAGE AND DOOR OPENER | tipo: http://www.grix.it/immagini/img2006/iconcine/schemi.gif | livello: http://www.grix.it/immagini/livelli/0.gif | |
| **Sistema di apertura remota di portone e garage** |
|  |
|  |
| http://www.grix.it/immagini/spacer.gifhttp://www.grix.it/immagini/spacer.gif    **Arduino Garage & Door 1.1**  In questo piccolo aggiornamento, se così lo possiamo chamare, sono stati aggiunti lo schema elettrico, i commenti nel sorgente e alcune screenshot relative all'utilizzo del sistema.  Dopo aver cercato e ricercato un po' ovunque un sistema veloce e affidabile per poter aprire il portone del garage e la porta attraverso il mio iPhone, ho deciso di realizzare un semplice circuito di controllo comandato da un Arduino combinato con una Ethernet Shield in modo da poterlo collocare in qualsiasi luogo della casa.  **Descrizione**  Il circuito è sostanzialmente un piccolo insieme di relè e transistor NPN PN2222 che fungono da interruttori per il segnale dell'apriporta e di un telecomando per garage lievemente modificato.  http://www.grix.it/UserFiles/icearrow/Image/Garage_bb.png Qesto è lo schema per BreadBoard realizzato con Fritzing.http://www.grix.it/UserFiles/icearrow/Image/Garage%20Scheme_schem.png  Allego anche uno schema elettrico nel quale è mostrata solo la Ethernet Shield, ovviamente quest'ultima deve essere collegata a una scheda Arduino Uno o compatibile.  Nello schema è presente anche un ingresso per l'alimentazione accoppiato a un 7805 per fornire 5V all'Arduino e ai relè.  Dei 3 Morsetti presenti nel circuito, 2 (quelli connessi ai Relè) sono utilizzati per chiudere il circuito dell'apriporta o dell'aprigarage; quello controllato direttamente dall'NPN PN2222 fornisce 12V (11,4V in realtà a causa del diodo di protezione 1N4007) ed è l'alimentazione per il Telecomando del Garage, non è fondamentale ai fini del progetto, ma per evitare di dover cambiare la batteria del telecomando ho preferito integrarlo nel circuito.  http://www.grix.it/UserFiles/icearrow/Image/IMG_0156.JPG Questo è il circuito montato su BreadBoard, rispetto al progetto dell'immagine precedente mancano il 7805 e i contatti per il comando dell'apriporta (devo ancora effettuare il cabalggio).  Vi è anche un ulteriore conatto (il cavo blu) che non è presente nel progetto precedente, tale conatto comunica alla scheda lo stato del circuito (ON o OFF) in modo tale da poter spegnere in caso di alimentazione assente o massa non comune.  http://www.grix.it/UserFiles/icearrow/Image/IMG_0161.JPG  Come potete vedere, l'alimentazione del telecomando è pilotata direttamente dall'NPN in alto a sinistra, mentre i contatto che chiude il circuito per aprire il Garage è collocato in basso a sinsitra.  Il circuito dell'apriporta è composto dai 2 relè di destra: quello in alto è un relè di sicurezza che se spento impedisce anche al relè di controllo (in basso) di chiudersi.  I 4 Led verdi fungono da spie per i vari componenti, non sono fondamentali per il circuito ma, almeno durante la programmazione dell'Arduino, sono molto utili per riconoscere i pin attivi.  **Il Telecomando**  http://www.grix.it/UserFiles/icearrow/Image/IMG_0160.JPG  Per poter attivare il telecomando con i relè basta individuare i conatti dei pulsanti del telecomando, collegarci 2 cavi e connetterli ai conatti del relè.  Meglio utilizzare cavi dello stesso colore per ogni conatto per poterli individuare meglio, non è necessario rispettare alcuna polarità.  L'alimentazione invece deve essere collegata a una fonte come la batteria originale o un alimentatore dello stesso voltaggio della batteria.  Nel mio progetto ho previsto anche l'accensione remota del telecomando attraverso un transistor NPN, in modo da poter spegnere il telecomando quando non è richiesto.  **Il Codice**  Come ogni circuito basato su Arduino, il codice è necessario affinchè il nostro microcontrollore agisca come richiesto.  Lo Sketch che ho utilizzato è sostanzialmente una modifica e un insieme di vari esempi disponibili nell'Arduino PlayGround, con l'aggiunta di alcune variabili, 2 temporizzatori per i relè di controllo, la comuncazione dei pin attualmente in uso, lo stato dei vari componenti e la comunicazione delle variabili.  Nella prima parte dello sketch vengono definiti i pin dei vari componenti e tali valori si applicano lungo tutto il codice, in questo modo non è necessario modificare i vari pin nel resto dello sketch.    /\*  \* Garage Opener via WebServerParsing using GET Request  \* Per impostare il valore di un pin HIGH o LOW  \* aprire un browser web qualsiasi e aprire il sito  \* http://(IP assegnato all'Arduino)/?pinD(numero del pin)=(0 o 1)  \* esempio: http://192.168.1.177/?pinD2=1  \*/  #include <SPI.h>  #include <Ethernet.h>  byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED }; // assegna un indirizzo MAC  byte ip[] = { 192,168,1,177 }; // Indirizzo IP  byte gateway[] = { 192,168,1,1}; // Gateway (o Router in Mac OS X)  EthernetServer server(80); // Porta server utilizzata  const int Garage = 2; // Gruppo di costanti per l'assegnazione  const int Circuit = 4; // dei pin di controllo  const int Door = 9; // La modifica avviene in tutto lo sketch  const int DoorCir = 6;  const int Remote = 8;  int door = 0; // Serie di variabili per il controllo  int ctrl = 1; // dei vari transistor e relè  int cir = 0;  int value = 0;  int rem = 1;  long time = 0;  long interval = 500;  void setup()  {  Serial.begin(9600);  Ethernet.begin(mac, ip);  server.begin();  Serial.println("ready");  pinMode (Garage, OUTPUT);  pinMode (Remote, OUTPUT);  pinMode (Circuit, INPUT);  pinMode (Door, OUTPUT);  pinMode (DoorCir, OUTPUT);  digitalWrite (Remote, rem);  digitalWrite (DoorCir, ctrl);  }  void loop()  {  cir = digitalRead(Circuit);  EthernetClient client = server.available();  if (client) {  while (client.connected()) {  if (cir != 1){  rem = 0;  ctrl = 0;  digitalWrite(Remote, rem);  digitalWrite(DoorCir, ctrl);  }  if (client.available()) {  // indicatori delle varie richieste, sia analogiche PWM che digitali  int digitalRequests = 0;  int analogRequests = 0;  if( client.find("GET /") ) { // ricerca di 'GET' (carattere ? nell'URL)  // ricerca del comando "pin"  while(client.findUntil("pin", "nr")){  char type = client.read(); // D o A  // il primo valore dopo D o A è il pin  int pin = client.parseInt();  int val = client.parseInt(); // valore da assegnare al pin  if( type == 'D') { // pin selezionato digitale  if(cir == HIGH){ // controllo del circuito  Serial.println("Circuit Online"); // comunicazione seriale dello stato del circuito  if(pin == Remote){ //  Serial.print("Remote "); // comunicazione seriale dello stato del telecomando  if(val == 1){ // accensione del telecomando  Serial.println("On"); //  rem = 1; // salvataggio dello stato del telecomando  }  else {  Serial.println("Off");  rem = 0;  }  }  else if (pin == DoorCir){  Serial.print("Door circuit "); // controllo del circuito di sicurezza per la porta  if(val == 1){  Serial.println("On"); // comunicazione seriale  ctrl = 1; // salvataggio dello stato del circuito di sicurezza  }  else{  Serial.println("Off");  ctrl = 0;  }  }  else if(pin == Door){ // sistema di apertura della porta  Serial.print("Door "); // se il circuito di sicurezza (ctrl) è attivo  if(val == 1 && ctrl == 1){ // la porta viene aperta  Serial.println("Open"); // e viene comunicato in sulla porta seriale  door = 1;  }  else {  Serial.println("Nothing Changed"); // circuito di sicurezza spento: la porta non si apre  door = 0;  }  }  else if(pin == Garage){ // sistema di apertura del garage analogo a quello della porta  Serial.print("Garage ");  if(val == 1 && rem == 1){  Serial.println("Opening / Closing");  value = 1;  }  else {  Serial.println("Inactive");  value = 0;  }  }  }  else {  Serial.println("Circuit Offline");  value = 0;  }  Serial.print("Digital pin ");  pinMode(pin, OUTPUT);  digitalWrite(pin, val);  digitalRequests++;  }  else if( type == 'A'){  Serial.print("Analog pin ");  analogWrite(pin, val);  analogRequests++;  }  else {  Serial.print("Unexpected type ");  Serial.print(type);  }  Serial.print(pin);  Serial.print("=");  Serial.println(val);  Serial.print("Value of Circuit (cir) = ");  Serial.println(cir);  Serial.print("Value of Memory (value) = ");  Serial.println(value);  Serial.print("Value of Remote (rem) = ");  Serial.println(rem);  Serial.print("Value of Door Circuit (ctrl) = ");  Serial.println(ctrl);  Serial.print("Value of Door (door) = ");  Serial.println(door);  }  }  Serial.println();  // la funzione findUntil ha rilevato la "fine linea" (a lf followed by cr)  // in questo modo la richiesta è terminata e può iniziare la comunicazione  // invio di un header html standard  client.println("HTTP/1.1 200 OK");  client.println("Content-Type: text/html");  client.println();  // comunicazione sul numero di uscite modificate (pin modificati)  client.print(digitalRequests);  client.print(" digital pin(s) written");  client.println("<br />");  client.print(analogRequests);  client.print(" analog pin(s) written");  client.println("<br />");  client.println("<br />");  // comunicazione del valore di ogni analog input  for (int i = 0; i < 16; i++) {  client.print("analog input ");  client.print(i);  client.print(" is ");  client.print(analogRead(i));  client.println("<br />");  }  // comunicazione dello stato dei pin fondamentali (Garage, circuito, porta ...)  client.println("<br />");  client.print("Garage pin = ");  client.print(Garage);  client.println("<br />");  client.print("Remote pin = ");  client.print(Remote);  client.println("<br />");  client.print("Door pin = ");  client.print(Door);  client.println("<br />");  client.print("Door Circuit pin = ");  client.print(DoorCir);  client.println("<br />");  client.println("<br />");  // Le restanti linee sono semplicemente un reset dei circuiti di controllo  // per non dover comunicare anche lo stato LOW al pin del telecomando e della porta  // Dopo 0,5s vengono spenti  // e viene comunicato lo stato anche via rete  if (cir == HIGH){  client.print("Circuit Online");  client.println("<br />");  if (rem == 1){  client.print("Remote ON");  client.println("<br />");  }  else if (rem != 1){  client.print("Remote OFF");  client.println("<br />");  value = 0;  }  if (ctrl == 1){  client.print("Door Circuit ON");  client.println("<br /r>");  }  else if (ctrl != 1){  client.print("Door Circuit OFF");  client.println("<br /r>");  door = 0;  }  }  else if (cir == LOW){  client.print("Circuit Offline");  client.println("<br />");  digitalWrite(Remote, LOW);  digitalWrite(DoorCir, LOW);  value = 0;  rem = 0;  ctrl = 0;  }  if (value == 1){  client.print("Garage Opening/Closing");  client.println("<br />");  delay(500);  digitalWrite(Garage, LOW);  value = 0;  }  else if (value != 1){  client.print("Garage inactive");  client.println("<br />");  digitalWrite(Garage, LOW);  value = 0;  }  if (door == 1){  client.print("Door Open");  client.println("<br /r>");  delay(500);  digitalWrite(Door, LOW);  door = 0;  }  else if (door != 1){  client.print("Door Closed");  client.println("<br /r>");  digitalWrite(Door, LOW);  door = 0;  }  break;  }  }  delay(1);  client.stop();  }  }  **Utilizzo**    Dopo aver realizzato il progetto e aver caricato il codice sull'Arduino, non ci resta che provare il circuito.    Dopo aver connesso un cavo Ethernet all'Ethernet Shield e aver collegato il sistema a una fonte d'alimentazione, per poter testare la connessione è necessario aprire un qualsiasi browser web e collegarsi al sito http://"IP dell'Arduino"/    A questo punto, se si vuole controllare un pin (0, 1, 11, 12, 13 esclusi) basta aggiungere in fondo all'indirizzo:  ?pinD2=1. In questa formula:  D identifica l'uscita, può essere sia D (digitale) che A (analogica, nei pin PWM)  2  è il numero del pin che si vuole controllare  1 è il valore assunto dalla porta, 1 = HIGH e 0 = LOW    Dopo l'invio del comando, sul browser apparirà anche lo stato dei vari comandi e il numero dei vari pin a essi correllati in modo da potrli individuare più facilmente.    L'invio dei comandi non è riservato ai soli pc o smartphone, qualsiasi dispositivo in grado di inviare richieste http può fungere da controllo.  http://www.grix.it/UserFiles/icearrow/Image/Garage/Garage(1).png  Ecco come ci appare la pagina web dopo l'accesso senza l'invio di comandi...  http://www.grix.it/UserFiles/icearrow/Image/Garage/Garage%20active.png  ... e dopo l'invio di in comando, in questo caso pinD6=0.    Come potete vedere le differenze si notano soprattutto nei vari indicatori di stato a fondo pagina.    Appena sopra di essi si possono vedere anche i valori dei vari pin a cui è collegato il circuito di contollo, e come accennato prima, sono molto utili se non ci si ricorda quale sia il pin da controllare.    Un altro sistema per controllare il circuito è l'utilizzo del terminale, disponibile in ogni disribuzione Linux e in Mac OS X.  Per poter inviare un comando all'Arduino basta utilizzare un semplice comando: curl  Attraverso questo comando si possono inviare anche richieste http, quelle utilizzate nello sketch, e nella finestra del terminale verranno mostrate le stesse informazioni riportate in un browser.  http://www.grix.it/UserFiles/icearrow/Image/Garage/Curl%20Garage(1).png    **Note sulla sicurezza**  Ricordo che nel sorgente non è presente alcun metodo di autenticazione per l'accesso ai comandi, se volete utilizzare il sistema anche da internet (attraverso un DynDNS o un ip statico) vi consiglio di creare un semplice server VPN o almeno cambiare porta di connessione all'inizio del sorgente (Ethernetserver Server (80) ).  http://www.grix.it/UserFiles/icearrow/Image/Garage/Garage.png    **Lista Componenti**  Arduino Uno (o compatibile con almeno 32K di Memoria)  Arduino Ethernet Shield o compatibile  3 Relè da 5V / 6V  4 NPN PN222  4 Led (Qualsiasi colore, opzionali)  4 Resistenze da 220 Ohm (Opzionali, per i led)  4 Resistenze da 150 Ohm  4 Resistenze da 10K Ohm  3 Resistenze da 4,7 Ohm  1 Diodo 1N4007  1 L7805 (Opzionale)  1 Condensatore da 10 uF (Opzionale)  1 Condensatore da 100 uF (Opzionale)  Radiocomando per Garage (Opzionale)  [http://www.grix.it/AD/easyEDA/EasyEDAbannoriz.gif](http://www.grix.it/partners/go.php?act=1&p=vectron&u=easyeda.com/it) |