ISTITUTO NAZIONALE DI FISICA NUCLEARE

Sezione di Genova

INFN-15-02/GE
24th March 2015

Il sistema di controllo periferiche per il portale INFN-ANN dell'esperimento SCINTILLA

R. De Vita[1], E. Fanchini[1], G. Firpo[2], A. Manco[1], G. Ottonello[1], F. Parodi[1], F. Pratolongo[1], V. Vigo[1]

[1]*Istituto Nazionale di Fisica Nucleare Sezione di Genova, Via Dodecaneso 33, I-16146 Genova, Italy*
[2]*Ansaldo Nucleare (ANN), C.so F.M. Perrone 25, I-16152 Genova, Italy*

Abstract

Questa nota descrive il sistema di controllo periferiche utilizzato dal Radiation Portal Monitor (RPM) sviluppato nell'ambito del progetto europeo SCINTILLA per la rivelazione di radiazioni nel monitoraggio di autoveicoli e container da trasporto.

Published by ***SIS–Pubblicazioni***
Laboratori Nazionali di Frascati

1 L'ESPERIMENTO SCINTILLA

SCINTILLA [1] è un progetto europeo di durata triennale finanziato nell'ambito del Seventh Framework Program FP7 (N. 285204) per lo sviluppo di rivelatori di radiazione per applicazioni legate alla sicurezza nucleare.

Lo scopo principale è quello di realizzare sistemi ottimizzati per la rivelazione di sorgenti radiative in diversi ambiti di utilizzo quali il controllo di veicoli, di container, di persone, oppure per il pronto intervento in caso di allarme nucleare. Questi sistemi devono essere in grado di rivelare e/o riconoscere materiali radioattivi per garantire un adeguato livello di sicurezza in ambienti sensibili quali porti commerciali, aeroporti, frontiere, ospedali o durante eventi pubblici.

In questo progetto, l'INFN, in collaborazione con Ansaldo Nucleare (ANN), ha sviluppato un rivelatore per il monitoraggio di veicoli e container. Il sistema, che verrà descritto nel paragrafo successivo, sfrutta l'utilizzo combinato di scintillatori plastici e Gadolinio per la rivelazione e discriminazione di gamma e neutroni emessi da materiale radioattivo con una singola tecnologia.

2 IL PORTALE ED IL SISTEMA DI CONTROLLO DELLE PERIFERICHE

Il sistema sviluppato da INFN ed ANN rientra nella categoria dei portali per la rivelazione di radiazioni (Radiation Portal Monitor RPM) utilizzati nel monitoraggio di autoveicoli e container da trasporto. Questi sistemi sono normalmente dotati di due rivelatori indipendenti, uno sensibile ai neutroni ed uno ai fotoni. L'utilizzo di un rivelatore di neutroni fornisce la garanzia di poter identificare in modo univoco la presenza di materiali fissili, essendo questi gli unici emettitori naturali di neutroni, e quindi di prevenirne il traffico illecito. La rivelazione di fotoni permette invece di verificare la possibile presenza di materiale radioattivo come eventuali sorgenti disperse o radioisotopi per uso in ambito medico.

Il portale sviluppato da INFN ed ANN, mostrato in Figura 1, è formato da due pilastri (pillar), che costituiscono la parte di rivelazione attiva del sistema e sono posti ad una distanza adeguata per agevolare il transito dei veicoli da controllare. I due pilastri sono montati su strutture che ne permettono il riposizionamento. I segnali rilevati dai due pilastri sono digitalizzati, filtrati e inviati ad una workstation per la successiva elaborazione. La workstation utilizzata è un computer desktop su cui il System Control Software (SCS) permette di configurare e controllare il portale e fornisce l'interfaccia grafica per l'utente. I ratei di segnali derivanti da eventi originati da gamma e neutroni vengono correlati con il passaggio dei veicoli, in modo da rivelare un eventuale incremento e dare un segnale di allarme qualora essi superino delle soglie opportunamente definite.

Il RPM deve essere pertanto equipaggiato con sensori di occupazione, per la gestione delle informazioni sul transito dei veicoli, e deve essere fornito di allarmi sonori e luminosi, per comunicare all'utente la rilevazione di materiale radiativo. Nel caso del portale descritto in questa nota, i sensori di occupazione o fotocellule e gli allarmi sono collegati ad un circuito basato su un microcontrollore programmabile, che ne consente la gestione. Questo è parte

integrante di un'unità di gestione delle periferiche (Peripheral Devices Box o PDB), oggetto della presente nota. L'unità di gestione delle periferiche è, a sua volta, collegata alla workstation che analizza i segnali provenienti dai sensori d'occupazione, li correla con i ratei misurati dai rivelatori e definisce i segnali da inviare ai segnalatori di allarme, trasmettendo inoltre i dati elaborati all'interfaccia grafica (GUI) per l'utente.

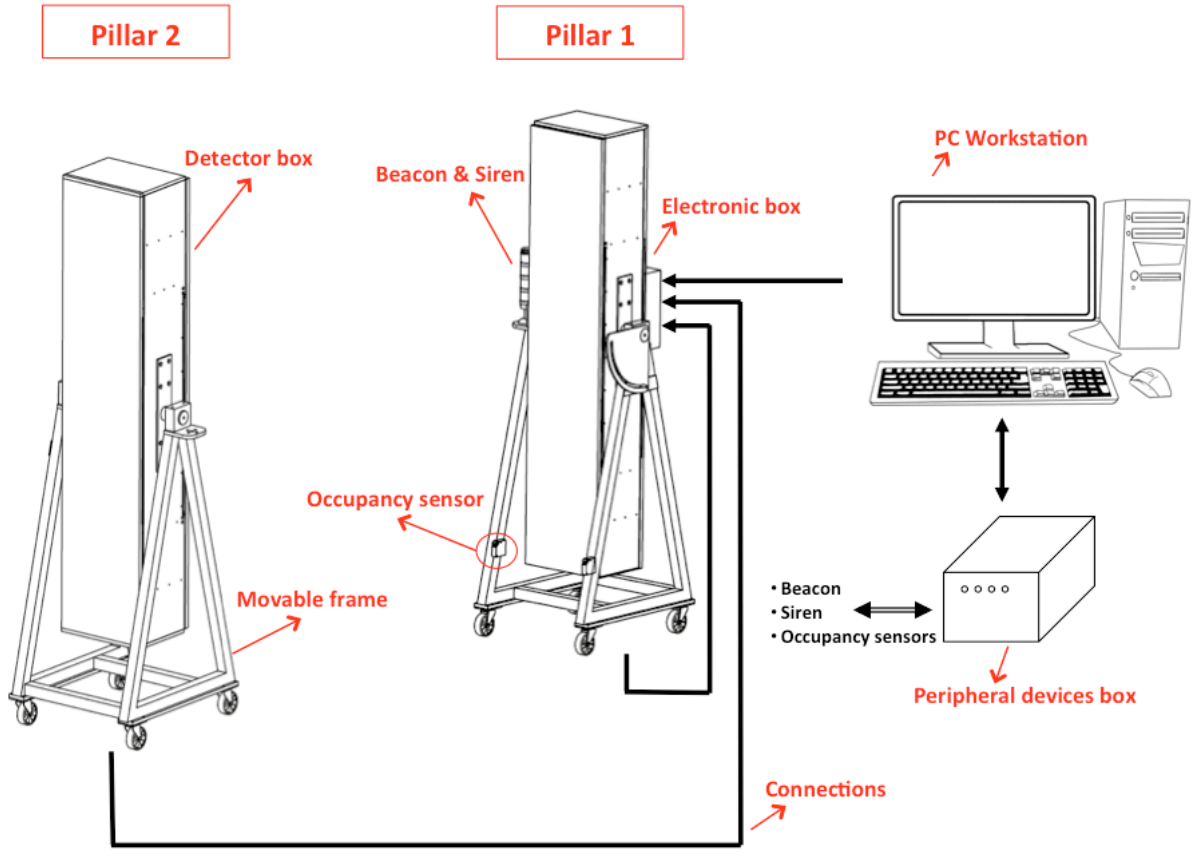

Figura 1: Schema del portale sviluppato da INFN e ANN nell'ambito del progetto SCINTILLA.

3 REQUISITI DEL SISTEMA

Il sistema RPM necessita di riconoscere il transito dei veicoli, di associarvi i ratei misurati dai rivelatori e, nel caso di rivelazione di materiale radioattivo, di segnalare la condizione di allarme.

La necessità di gestire in contemporanea fotocellule, allarmi sonori e luminosi ha portato alla scelta di un sistema di controllo commerciale. La scelta si è basata sulle caratteristiche tecniche necessarie per la gestione in tempo reale dei segnali in ingresso ed in uscita delle periferiche in questione. Questo richiede, in particolare, tempi di risposta paragonabili o inferiori al tempo di misura dei ratei di neutroni e gamma, in modo da poter correlare tali ratei ai segnali delle periferiche senza introdurre ritardi temporali rilevanti. Nel portale in esame, il tempo di misura dei ratei è stato fissato a 100 ms, essendo questo

sufficiente ad accumulare un adeguato numero di conteggi ma allo stesso tempo significativamente inferiore al tempo di transito di un veicolo o un container che secondo gli standard internazionali dovrebbe viaggiare ad una velocità di 2.2 m/s o 8 km/h.

La scelta è pertanto ricaduta su un sistema elettronico basato su un microcontrollore Arduino, mostrato in Figura 2, non solo per il basso costo, versatilità e capacità di soddisfare i requisiti elencati precedentemente, ma anche perché equipaggiato con porte USB facilmente adattabili a computer o workstation con cui può essere direttamente alimentato.

Figura 2: Fotografia del micro-controllore Arduino Leonardo utilizzato per il controllo delle periferiche.

4 IL PROGETTO DELLA SCHEDA DI CONTROLLO

La scheda Arduino Leonardo è dotata di un processore Atmega32u4 con una memoria flash di 32 KB, 14 input/output digitali e 12 ingressi analogici. Le specifiche complete sono riportate in [2].

Il processore Arduino è montato su una scheda di emulazione mostrata in Figura 2. Questa scheda alloggia all'interno dell'unità di controllo PDB, descritta in seguito. Questa configurazione permette la rapida sostituzione del processore in caso di guasto. La connessione dei segnali tra la schedina di emulazione e l'unità di controllo è assicurata da un cavo che esce dalla scheda di Figura 2 e che si inserisce su un connettore montato sul PCB dell'unità di controllo. Il collegamento tra la PDB e il PC con il software di acquisizione (SCS) avviene attraverso la porta USB dell'Arduino.

Come descritto in Sezione 2, il portale possiede un segnalatore acustico, che viene attivato in caso di allarme gamma o neutroni e 4 segnalatori luminosi, con il seguente codice colore:

- verde, per segnalare il normale funzionamento del sistema;
- arancio, per segnalare un malfunzionamento;
- blu, per segnalare un allarme neutroni;
- rosso, per segnalare un allarme gamma;

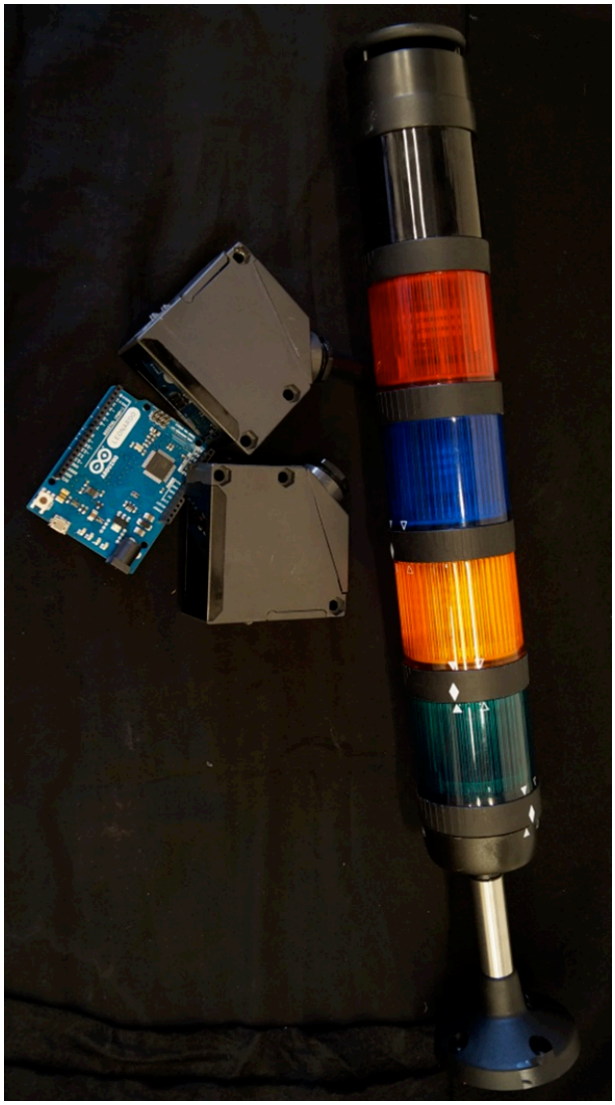

Figura 3: Foto del microcontrollore Arduino a sinistra, dei sensori di occupazione al centro e e dei segnalatori luminosi e sonoro a destra.

Inoltre, il portale deve acquisire i segnali di due sensori di occupazione, posizionati sulla struttura di supporto del Pillar 1 come illustrato in Figura 1. L'utilizzo di due sensori consente non solo di rilevare il transito di un veicolo, ma anche di determinarne direzione e velocità. I sensori e i segnalatori di allarme sono mostrati in Figura 3.

Poiché come segnalatori luminosi sono stati usati delle semplici lampade commerciali a 24 V, mentre i segnali di uscita dell'Arduino sono a 5 V, abbiamo progettato un circuito di pilotaggio con un transistor e un relè MT2 [3] che ha un'eccitazione di 5 V per 15 mA e il

secondario da 60 W e 2 A. Per una semplificazione di progetto abbiamo usato lo stesso relè anche per l'acquisizione dei segnali di occupazione presenza.

In Figura 4 è mostrata la scheda con i 52 componenti già montati e con l'Arduino Leonardo installato. Come illustrato precedentemente, l'Arduino e connesso al PCB tramite un cavetto che si innesta sul connettore J4.

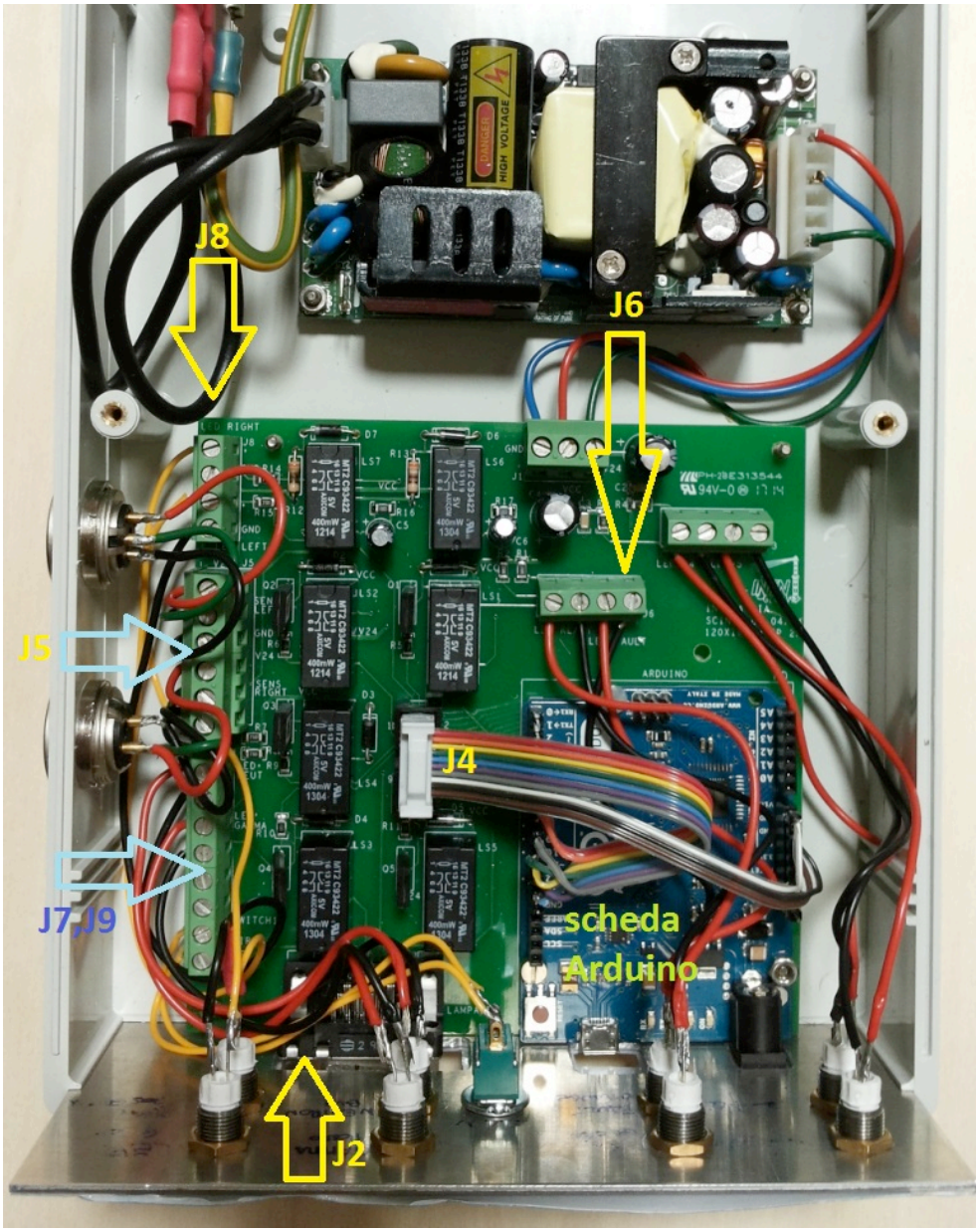

Figura 4: Foto della scheda di controllo con indicazioni della posizione dei principali connettori.

Il connettore J2 per comodità costruttiva è stato montato direttamente sul PCB in basso a sinistra. Esso alimenta le lampade di segnalazione e la sirena per un totale di 5 segnali più la massa. Il connettore è un semplice DB9 in quanto non vi sono particolari esigenze di potenza

elettrica.

Il connettore J5 è sul lato sinistro della scheda e serve per collegare i sensori di presenza del portale al PCB; attraverso una resistenza di caduta l'uscita a 24 V del sensore eccita un relè MT2 che, con un circuito di anti rimbalzo, porta il segnale a 5 V alla porta d'ingresso dell'Arduino.

I connettori J6 (in alto al centro), J7, J8 e J9 (lato sinistro della scheda) portano i segnali corrispondenti allo stato delle fotocellule e dei segnali di allarme ai led posizionati sul pannello frontale della PDB. La scatola per la gestione delle periferiche, che si vede in Figura 5, fornisce così un monitoraggio indipendente e ridondante dello stato del portale per una maggiore sicurezza di funzionamento.

Figura 5: Foto della *Peripheral devices box*, la scatola contenente l'apparato di gestione delle periferiche controllato da Arduino.

Nella scheda di controllo entrano due tensioni di 24 V e 5 V che vengono filtrate e poi distribuite a tutto il PCB. Queste tensioni sono generate da un *switching power supply* che fa la conversione dall'alimentazione di rete.

Per i dettagli del progetto si può fare riferimento allo schema elettrico riportato in ultima pagina (Figura 7).

5 DESCRIZIONE DEL LAYOUT DELLA SCHEDA

Il layout della scheda in oggetto è stato progettato usando il software Allegro PCB Design Editor. Il PCB è un circuito di 2 layers in FR4 con un piano di massa nello strato

inferiore (BOTTOM). Il PCB di 120x120 mm ha uno spessore standard di 1.6 mm ed è stato costruito dalla ditta Phoenix.

Dal punto di vista del layout, le piste hanno una distanza minima di 0.3 mm mentre, in corrispondenza delle piste a 24 V, l'isolamento passa a 0.7 mm con un elevato margine di sicurezza. Lo spessore delle piste è pari a 0.3 mm, eccezion fatta per le piste a 24 V che hanno uno spessore di 0.5 mm. Le piste di alimentazione VCC e V24 hanno uno spessore ancora più grande essendo di 0.8 mm.

Per riuscire a inserire un piano di massa nello strato di BOTTOM e affinché questo piano sia il più continuo possibile, il routing sullo strato superiore (TOP), mostrato in Figura 6, è stato fatto manualmente. Perciò nello strato di BOTTOM c'è il piano di massa ed un numero limitato piste.

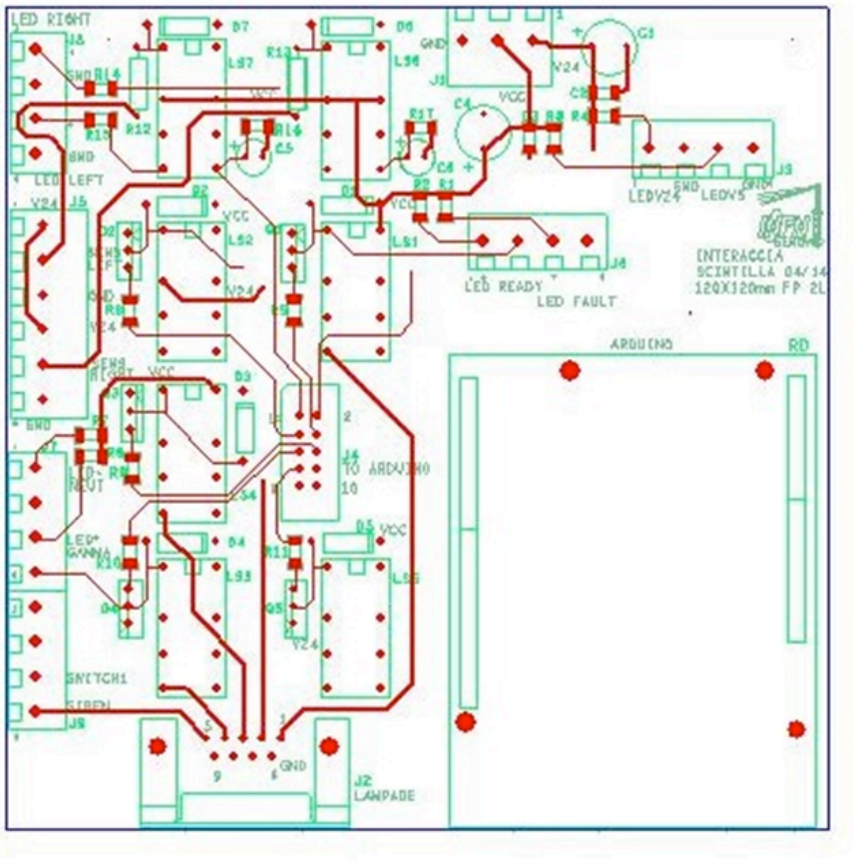

Figura 6: Le piste dello strato TOP della scheda.

Il progetto del layout e il montaggio della scheda sono stati portati a termine presso il laboratorio di elettronica della sezione di Genova. La scheda è stata testata simulando le uscite dall'Arduino e andando a controllare sui sensori il comportamento atteso. Poi si sono simulati i segnali dei sensori di presenza verificando le tensioni di uscita dalla nostra scheda all'Arduino.

6 PROGRAMMAZIONE MICROCONTROLLORE

La programmazione del microcontrollore Arduino è stata effettuata utilizzando il kit Java di sviluppo fornito dal produttore (ARDUINO 1.0.5 - 2013.05.15). Il firmware sviluppato si basa sul codice riportato in Appendice A che esegue le seguenti operazioni:

- configurazione di 5 porte di uscita per la gestione dei 4 segnalatori luminosi e del segnalatore acustico;
- configurazione di 2 porte di ingresso per la lettura dei segnali dei due sensori di occupazione;
- trasferimento periodico dei valori delle variabili di allarme registrati nella memoria del microcontrollore alle porte di uscita per l'aggiornamento dello stato dei segnalatori di allarme;
- lettura periodica dello stato dei sensori di occupazione attraverso le porte di ingresso.

L'aggiornamento in lettura e scrittura della memoria del microcontrollore viene gestito dal SCS del portale come illustrato nella sezione successiva.

7 INTEGRAZIONE NEL SISTEMA CONTROLLO PORTALE

La comunicazione fra il software di controllo del portale o SCS e la memoria del microcontrollore Arduino è stata implementata utilizzando la libreria ModBus [4]. Questa consente di accedere alla memoria del microcontrollore in lettura e scrittura, in modo da leggere le aree di memoria dove è registrato lo stato dei sensori di occupazione e aggiornare le aree di memoria corrispondenti ai valori di allarme. Queste operazioni di lettura e scrittura vengono effettuate con un periodo di 10 ms.

I valori corrispondenti allo stato dei sensori di occupazione vengono elaborati in modo da determinare lo stato di occupazione del portale, identificare quale fra i due sensori ha rilevato per primo il transito del veicolo e quindi la sua velocità di transito e la sua direzione. Per evitare i problemi legati ai segnali multipli indotti, per esempio, dal transito delle ruote di un veicolo di fronte alle fotocellule, il loro segnale, una volta rilevato attivo, è considerato tale per un periodo di 500 ms. Questo consente di determinare correttamente lo stato di occupazione del sistema, indipendentemente dalla struttura del veicolo e si è dimostrato efficace anche nel caso di transito di pedoni. Ogni qual volta il portale risulta occupato il suo stato viene considerato tale per un periodo minimo di 2 s, entro il quale i rate di gamma e neutroni misurati vengono registrati su disco anche in assenza di allarmi.

Il periodo di 10 ms utilizzato per la comunicazione con il microcontrollore consente di aggiornare lo stato dei segnalatori di allarme in tempo reale, rispetto alle misure di rate e di determinare la velocità di passaggio dei veicoli con una precisione tipica del 5-10%.

8 PRESTAZIONI

Il sistema RPM (Radiation Portal Monitor) è stato testato in due campagne di misura presso il Joint Research Centre di Ispra (Va) per verificarne le prestazioni di rivelazione di sorgenti di fotoni e neutroni attraverso test in dinamica.

Le misure effettuate sono ripetizioni del transito di un carrello su cui è alloggiata la sorgente da rivelare. I due pilastri del portale sono posti alla distanza di 2.5 m dalla linea di passaggio della sorgente. Il sistema deve identificare il transito del carrello, allarmare in presenza di sorgenti e generare un file di output per ogni transito con il resoconto della misura. Il file creato contiene le informazioni riguardanti direzione, velocità, data e durata del transito, unite a quelle di rivelazione, come i ratei di fotoni e neutroni, il numero e la tipologia degli allarmi e lo stato del sistema.

I test seguono procedure previste da standard internazionali per la categoria e sono stati effettuati senza alcuna supervisione degli sviluppatori. I risultati mostrano esiti superiori agli standard della categoria e prestazioni confrontabili se non a volte superiori a sistemi commerciali.

9 RINGRAZIAMENTI

La ricerca che ha portato ai risultati di questa nota è stata finanziata nell'ambito dello European Community's Seventh Framework Programme (FP7/2007-2013), Grant Agreement n°285204.

10 BIBLIOGRAFIA

[1] FP7-SCINTILLA, Grant Agreement n°285204: http://www.scintilla-project.eu/ .
[2] "Primi passi con Arduino" speciale di Elettronica In, Dicembre 2011 a cura di Simone Majocchi.
[3] MT2 Relay, Tyco Electronics.
[4] http://libmodbus.org.

APPENDICE A

```
#include <Modbusino.h>
/* Initialize the slave with the ID 1 */
ModbusinoSlave modbusino_slave(1);
/* Allocate a mapping of 10 values */
uint16_t tab_reg[10];
const int outPin_1= 9;
const int outPin_2 = 10;
const int outPin_3 = 11;
const int outPin_4 = 12;
const int outPin_5 = 8;
const int inPin_1 = 5;
const int inPin_2 = 6;
int out_1_state;
int out_2_state;
int out_3_state;
int out_4_state;
int out_5_state;
int in_1_state;
int in_2_state;
int occupancy=0;
int starting_sensor=0;
int sens_1=LOW;
int sens_2=LOW;
int prev_sens_1, prev_sens_2;

void setup() {
   /* The transfer speed is set to 115200 bauds */
   modbusino_slave.setup(115200);
   pinMode(outPin_1, OUTPUT);
   pinMode(outPin_2, OUTPUT);
   pinMode(outPin_3, OUTPUT);
   pinMode(outPin_4, OUTPUT);
   pinMode(outPin_5, OUTPUT);
   pinMode(inPin_1, INPUT);
   pinMode(inPin_2, INPUT);
}

void loop() {

   out_1_state=tab_reg[1];
   out_2_state=tab_reg[2];
   digitalWrite(outPin_1, out_1_state);
   digitalWrite(outPin_2, out_2_state);
   out_3_state=tab_reg[3];
```

```
  out_4_state=tab_reg[4];
  digitalWrite(outPin_3, out_3_state);
  digitalWrite(outPin_4, out_4_state);
  if (out_3_state>0 || out_4_state>0) out_5_state=1;
  else out_5_state=0;
  digitalWrite(outPin_5, out_5_state);


  prev_sens_1=sens_1;
  sens_1=digitalRead(inPin_1);
  prev_sens_2=sens_2;
  sens_2=digitalRead(inPin_2);

  if ((sens_1==HIGH && prev_sens_1==LOW) || (sens_2==HIGH && prev_sens_2==LOW)) {
   occupancy=1;
   if (sens_1==HIGH && prev_sens_1==LOW && starting_sensor!=2)
     starting_sensor=1;
   else if (sens_2==HIGH && prev_sens_2==LOW && starting_sensor!=1)
     starting_sensor=2;
  }
  if (starting_sensor==1 && sens_2==LOW && prev_sens_2==HIGH) {
   occupancy=0;
   starting_sensor=0;
  }
  if (starting_sensor==2 && sens_1==LOW && prev_sens_1==HIGH) {
   occupancy=0;
   starting_sensor=0;
  }


   /* values to read */
  if (sens_1==HIGH)
    tab_reg[5] = 1;
  else
    tab_reg[5] = 0;
  if (sens_2==HIGH)
    tab_reg[6] = 1;
  else
    tab_reg[6] = 0;
  tab_reg[7]=occupancy;
  tab_reg[8]=starting_sensor;
  /* Launch Modbus slave loop with:
     - pointer to the mapping
     - max values of mapping */
  modbusino_slave.loop(tab_reg, 10);
}
```