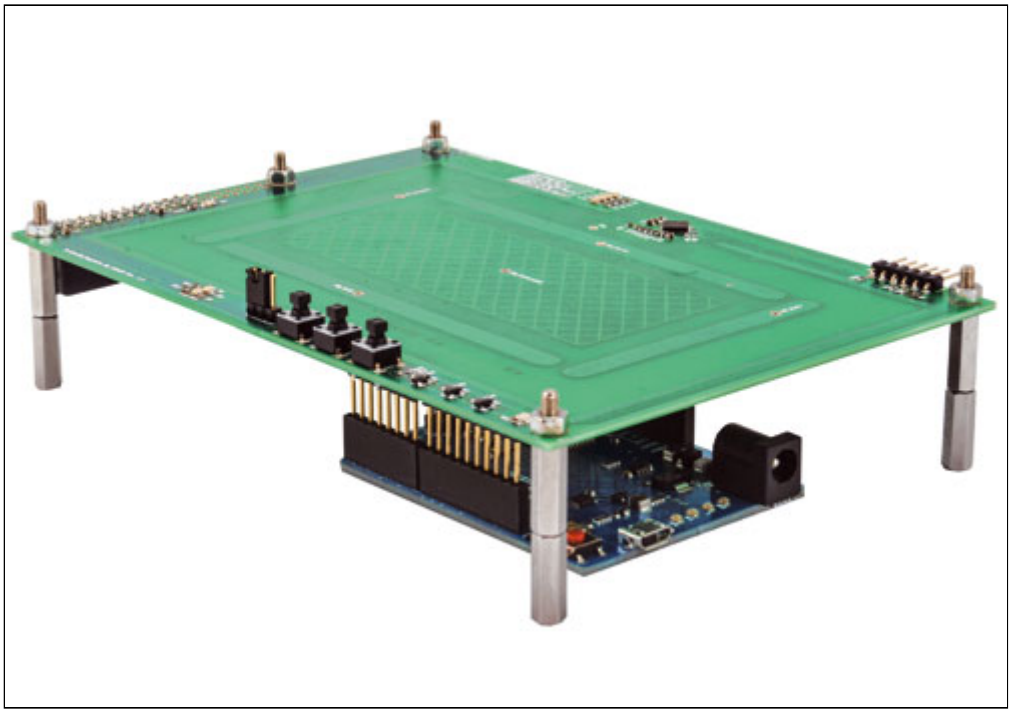
Gesture Raspberry Pi e Arduino - in kit

Prezzo: 47.54 €

Tasse: 10.46 €

Prezzo totale (con tasse): 58.00 €

Basata sul chip MGC3130 della Microchip e abbinata ad Arduino Uno Rev3 (oppure la Arduino Leonardo Rev3) o Raspberry Pi (B+ / 2 / 3B / 3B+), questa scheda (elettrodo) consente di realizzare un sistema per il riconoscimento dei gesti. Dispone di tre pulsanti (P1, P2 e P3) e un jumper, che servono per riprodurre le funzioni implementate nella scheda; un LED di segnalazione per Arduino (LD7) e per Raspberry Pi (LD8), utile durante il riconoscimento delle Gesture o durante la gestione della pressione dei pulsanti P1, P2 e P3; I/O estesi EIO2, EIO3, EIO6 e EIO7 ai quali sono collegati dei LED per segnalare le gesture riconosciute. Ad ogni gesture riconosciuta l'integrato MGC3130 genera un impulso sulla rispettiva uscita. Tuttavia è possibile scegliere e configurare diversamente il comportamento delle uscite rispetto alla gesture riconosciuta. Infatti oltre all'impulso è possibile scegliere: uscita permanentemente alta; uscita permanentemente bassa; toggle. Per studiare meglio la piattaforma Gestic è stato realizzato un software (GesticTester) con cui interfacciarsi, grazie al quale possiamo vedere in tempo reale le gesture riconosciute dall'integrato MGC3130. Il software è suddiviso in due TAB distinti: il primo (Gesture) monitorizza e visualizza tutte le gesture intercettate dall'integrato, mentre il secondo (Firmware) serve per visualizzare la revisione firmware caricata nell'integrato. La comunicazione tra PC e scheda demo avviene tramite interfaccia USB. Dimensioni (mm): 156x108. **Attenzione!** i componenti in SMD sono già saldati mentre gli altri componenti devono essere saldati.



Libreria Arduino MGC3130

Per le schede Arduino Uno Rev.3 e Arduino Leonardo Rev.3 sono disponibili due demo che si appoggiano sulla nostra libreria di gestione dell'integrato MGC3130. La demo scritta per la scheda Arduino Uno Rev.3 viene completata dalla scheda di espansione FT1079K, la quale mette a disposizione 8 ingressi digitali e altrettante uscite a relé. Per la nostra demo useremo solo le otto uscite a relé, tuttavia, per chi lo desiderasse, è possibile gestire anche gli ingressi modificando opportunamente il codice della demo. La gestione degli I/O viene fatta sfruttando l'integrato Microchip MCP23017, il quale viene connesso alle schede Arduino tramite il bus I2C come l'integrato MGC3130. Anche per l'MCP23017 abbiamo scritto una libreria di supporto che andremo a descrivere in breve durante l'articolo. Grazie alle uscite a relé messe a disposizione, possiamo riportare le gesture riconosciute su una delle possibili uscite. Per la nostra demo abbiamo deciso di riportare fino a un massimo di sedici gesture, per un totale di due schede FT1079K. Ovviamente è possibile aggiungere altre FT1079K per riportare più gesture possibili sulle uscite a relé (l'aggiunta di più di due schede FT1079K necessita di apportare modifiche allo sketch da noi scritto e può essere visto come un esercizio didattico interessante). Invece la demo che riguarda la scheda Leonardo Rev.3 ci permette di interagire con il PC e in particolare con un programma di visualizzazione delle immagini. Grazie alle gesture riconosciute dall'integrato, sarà possibile sfogliare le immagini presenti sul PC.

Le gesture riconosciute e messe a disposizione dalla libreria sono: • Gesture Touch South; • Gesture Touch West; • Gesture Touch North; • Gesture Touch East; • Gesture Touch Centre; • Gesture Tap South; • Gesture Tap West; • Gesture Tap North; • Gesture Tap East; • Gesture Tap Centre; • Gesture Double Tap South; • Gesture Double Tap West; • Gesture Double Tap North; • Gesture Double Tap East; • Gesture Double Tap Centre; • Flick West to East; • Flick East to West; • Flick South to North; • Flick North to South; • Edge Flick West to East; • Edge Flick East to West; • Edge Flick South to North; • Edge Flick North to South; • Clock Wise; • Counter Clock Wise.

MGC3130 e la Libreria per Raspberry Pi

Libreria scritta in Python per gestire l'integrato MGC3130, la quale si rifà sulla falsa riga di quella per Arduino. Cominciamo con una puntualizzazione sulla gestione delle strutture dati in Python necessarie a gestire il flusso dati proveniente dall'integrato MGC3130. In Python le strutture dati vengono costruite in modo leggermente differente rispetto al linguaggio di programmazione C, di seguito riportiamo la nuova libreria in Python per Raspberry Pi.

class GestureBit(Structure): _fields_ = [("TouchSouth", c_uint32, 1), ("TouchWest", c_uint32, 1), ("TouchNorth", c_uint32, 1), («TouchEast», c_uint32, 1), («TouchCentre», c_uint32, 1), ("TapSouth", c_uint32, 1), ("TapWest", c_uint32, 1), ("TapNorth", c_uint32, 1), ("TapEast", c_uint32, 1), ("TapCentre", c_uint32, 1), ("DoubleTapSouth", c_uint32, 1), ("DoubleTapWest", c_uint32, 1), ("DoubleTapNorth", c_uint32, 1), ("DoubleTapEast", c_uint32, 1), ("DoubleTapCentre", c_uint32, 1), ("GestWestEast", c_uint32, 1), ("GestEastWest", c_uint32, 1), ("GestSouthNorth", c_uint32, 1), ("GestNorthSouth", c_uint32, 1), ("EdgeGestWestEast", c_uint32, 1), ("EdgeGestEastWest", c_uint32, 1), ("EdgeGestSouthNorth", c_uint32, 1), ("EdgeGestNorthSouth", c_uint32, 1), ("GestClockWise", c_uint32, 1), ("GestCounterClockWise", c_uint32, 1), ("Free", c_uint32, 7)] class GestureByte(Structure): _fields_ = [("Byte0", c_uint8), ("Byte1", c_uint8), ("Byte2", c_uint8), ("Byte3", c_uint8)] class Gesture(Union): _fields_ = [("Gesture32Bit", GestureBit), ("GestureByte", GestureByte), ("GestureLong", c_uint32), ("GestArray", c_ubyte * 4)]

Documentazione e link utili

- **GESTIC_RASPBERRY**
- **GESTIC_ARDUINO**
- **https://github.com/open-electronics/GestIC**