Raspberry Pi, Επικοινωνία με Scratch και Απομακρυσμένος Έλεγχος Ρομποτικού Βραχίονα

Σ. Τσιτσιμπάσης

Καθηγητής Πληροφορικής ΠΕ19 Δευτεροβάθμιας Εκπαίδευσης
stefane@sch.gr

Περίληψη

Ο απομακρυσμένος έλεγχος μέσω του προγραμματιστικού περιβάλλοντος Scratch, ηλεκτρονικών κυκλωμάτων που συνδέονται με τον υπολογιστή Raspberry Pi είναι το αντικείμενο το οποίο πραγματεύεται αυτή η εργασία. Αρχικά, παρουσιάζονται απαραίτητες ρυθμίσεις λογισμικού στο Raspberry Pi, προκειμένου να δοθεί η δυνατότητα ελέγχου ηλεκτρονικών κυκλωμάτων μέσω της εφαρμογής Scratch. Ένα διαμορφωμένο περιβάλλον Scratch σε PC μπορεί να επικοινωνεί και να αλληλεπιδρά με άλλα προγράμματα Scratch ή python που βρίσκονται στον υπολογιστή Raspberry Pi. Μια τέτοια διαμόρφωση παρουσιάζεται στη συνέχεια, ενώ παράλληλα περιγράφεται η υλοποίηση δύο Scratch εργασιών απομακρυσμένου ελέγχου από PC: της λειτουργίας λυχνίας LED και της καθοδήγησης ρομποτικού βραχίονα. Προτείνονται αντίστοιχα διαγράμματα συνδεσμολογίας με το Raspberry Pi και παρατίθενται προγράμματα διαχείρισης σε Scratch και python.

Λέξεις κλειδιά: Raspberry Pi, GPIO, Scratch, σύνοδος Host Mesh, απομακρυσμένος έλεγχος, python, ρομποτικός βραχίονας.

1. Εισαγωγή

Είναι δεδομένος ο εμπλουτισμός των ΤΠΕ στην παιδαγωγική διαδικασία τα τελευταία χρόνια, με την ενσωμάτωση διαδραστικών αντικειμένων από το πραγματικό περιβάλλον. Σε αυτό το πλαίσιο η διασύνδεση του κόσμου των υπολογιστών με το φυσικό κόσμο, δημιούργησε την έννοια του "physical computing", ορίζοντάς την ως δημιουργία φυσικών συστημάτων με τη χρήση λογισμικού και υλικού, που μπορούν να «αισθάνονται» τον πραγματικό κόσμο και να ανταποκρίνονται σε αυτόν (O'Sullivan & Igoe, 2004; Wikipedia, 2014). Ηλεκτρονικές κατασκευές ή σε συνδυασμό με άλλες όπως τηλεχειριζόμενα παιχνίδια, οχήματα, ρομπότ, οι οποίες με κατάλληλο προγραμματισμό μπορούν να αλληλεπιδρούν με το χρήστη, συναρπάζουν τα παιδιά και αυξάνουν την ενασχόλησή τους με αυτές. Γνωστά περιβάλλοντα όπως Arduino, Lego MindStorms, Raspberry Pi, Enchanting, LEGO WeDo, κ.α. που λειτουργούν ως εργαλεία του "physical computing", έχουν κοινό στόχο: να προκαλέσουν την εντονότερη εμπλοκή των

μαθητών με τις ΤΠΕ, να συνεισφέρουν στην καλλιέργεια της δημιουργικής σκέψης, να αναδείξουν την προσέγγιση αλγοριθμικών προβλημάτων με παραστατικό τρόπο.

Το Raspberry Pi (RPi) ως ολοκληρωμένο υπολογιστικό σύστημα χαμηλού κόστους σχεδιάστηκε με στόχο την προώθηση της διδασκαλίας βασικών εννοιών της επιστήμης των υπολογιστών στα σχολεία. Η επικοινωνία του με τον έξω κόσμο επιτυγχάνεται μέσω ακίδων (GPIO pins) που προγραμματίζονται ως ψηφιακοί είσοδοι / έξοδοι. Το Scratch ως εκπαιδευτική εφαρμογή είναι προσανατολισμένο για τη διδασκαλία εννοιών προγραμματισμού σε παιδιά. Η έκδοση 1.4, βρίσκεται προεγκατεστημένη στο λειτουργικό σύστημα του RPi και μπορεί να χρησιμοποιηθεί για τον έλεγχο των GPIO ακίδων του. Αυτό σε συνδυασμό με τη δυνατότητα επικοινωνίας και αλληλοεπίδρασης πολλαπλών προγραμμάτων Scratch μεταξύ τους, ενώ βρίσκονται σε διαφορετικούς υπολογιστές, ανοίγει προοπτικές για χρήση του συστήματος σε πολυάριθμα εκπαιδευτικά projects.

Σκοπός της εργασίας αυτής είναι να προτείνει ένα εναλλακτικό περιβάλλον στον έλεγχο ηλεκτρονικών κατασκευών, απομακρυσμένα μέσω ενός κοινού PC, με τη χρήση του υπολογιστή Raspberry Pi και ενός ήδη γνώριμου και οικείου προγραμματιστικού περιβάλλοντος, του Scratch. Δύο τύποι απομακρυσμένου ελέγχου παρουσιάζονται, μεταξύ PC και Raspberry Pi: ο πρώτος τύπος αφορά συνεργαζόμενα προγράμματα Scratch ενώ ο δεύτερος αφορά συνεργαζόμενα προγράμματα Scratch και python.

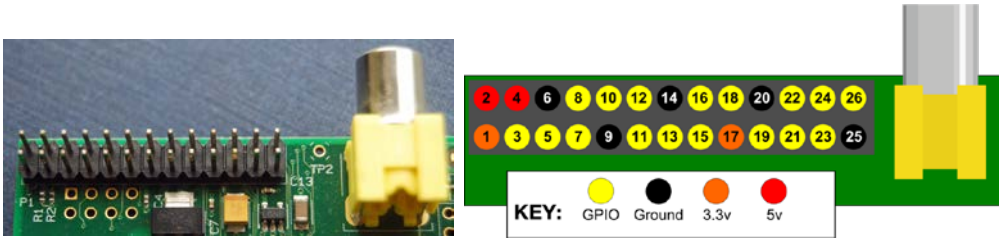
2. Το GPIO

Για τη διαχείριση απλών ηλεκτρονικών κυκλωμάτων στο Raspberry Pi μέσω του προγραμματιστικού περιβάλλοντος Scratch, απαιτείται το πρόσθετο λογισμικό *ScratchGPIO* (Cymplecy, 2014). Πρόκειται για πακέτο με απαραίτητα προγράμματα (python scripts) που διαχειρίζονται την επικοινωνία του Scratch με το GPIO. Η πολύ σύντομη εγκατάσταση προϋποθέτει πως στο Raspberry Pi υπάρχει το λειτουργικό Raspbian (Debian Wheezy), καθώς και πως διαθέτει πρόσβαση στο internet. Σε ένα Terminal παράθυρο στο RPi, θα χρειαστεί απλά να εκτελεστούν οι δύο παρακάτω εντολές:

- sudo wget http://goo.gl/dANpKr -O isgh4.sh
- sudo bash isgh4.sh

Η αλληλοεπίδραση του RPi με το φυσικό κόσμο, γίνεται μέσω των GPIO pins (εικόνα 1). Από τα 26 General Purpose Input Output (GPIO) ports του RPi, 17 μπορούν να προγραμματιστούν είτε ως είσοδοι είτε ως έξοδοι. Οι είσοδοι χρησιμοποιούνται για τον εντοπισμό μεταβολών τάσης σε ηλεκτρονικά κυκλώματα όπως, διακόπτες και αισθητήρες (θερμοκρασίας, απόστασης, φωτός, ήχου, μαγνητισμού, υπέρυθρων κ.λ.π.). Οι έξοδοι χρησιμοποιούνται για έλεγχο ηλεκτρονικών κυκλωμάτων ενεργοποιώντας τα ή απενεργοποιώντας τα, όπως κυκλώματα με λυχνίες LED, μοτέρ, relay διακόπτες, ηχεία, κ.α. Η εγκατάσταση του

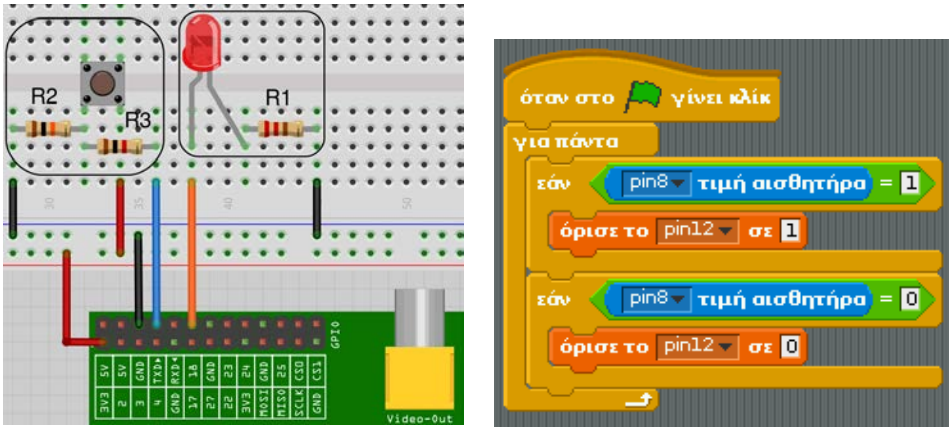
ScratchGPIO παραπάνω, θέτει αρχικά ως προεπιλεγμένα output ports στο GPIO τα pins: 11, 12, 13, 15, 16 και 18 και ως input ports τα pins: 3, 5, 7, 8, 10, 19, 21, 22, 23, 24 και 26. Η προεπιλογή αυτή, μπορεί να αλλάξει ανάλογα με τις απαιτήσεις του κάθε project. Στο περιβάλλον του Scratch, μεταβλητές και αισθητήρες με το χαρακτηριστικό όνομα "*pinX*" (όπου *X* ο αριθμός του input / output pin στο GPIO), είναι "δεσμευμένες" για τον προγραμματισμό των GPIO εισόδων / εξόδων.



Εικόνα 1. *Το GPIO. Φωτογραφία και αντίστοιχο διάγραμμα με την αρίθμηση των 26 pins*

3. Ανίχνευση ρεύματος σε διακόπτη και έλεγχος λυχνίας LED

Ως εφαρμογή των παραπάνω, η κατασκευή δύο βασικών ηλεκτρονικών κυκλωμάτων, θα δώσει την ευκαιρία για τον προγραμματισμό ενός διακόπτη και μιας λυχνίας LED.



Εικόνα 2. *Αριστερά, σχηματικό διάγραμμα συνδεσμολογίας κυκλώματος διακόπτη-αντιστάσεων και LED-αντίστασης με το GPIO. Δεξιά, πρόγραμμα Scratch στο RPi που ανάβει το LED όταν διαρρέει ρεύμα το pin8.*

Τα δύο ανεξάρτητα μεταξύ τους κυκλώματα σημειωμένα στο διάγραμμα της εικόνας 2, συνδέονται κατάλληλα στο GPIO: το κύκλωμα του διακόπτη με το input pin8 (μπλε καλώδιο) για την ανίχνευση ρεύματος, ενώ το κύκλωμα του LED με το output pin12 (πορτοκαλί καλώδιο) για τον έλεγχο του ανάμματος της λυχνίας. Στόχος είναι

το RPi να ενεργοποιεί το LED όταν εντοπίζει το πάτημα του διακόπτη, ενώ να το απενεργοποιεί στην αντίθετη περίπτωση. Οι τιμές των αντιστάσεων είναι R1=220 Ω, R2=10 kΩ, R3=1 kΩ. Οι αντιστάσεις R1 και R3 περιορίζουν πιθανά ισχυρά ρεύματα, ενώ η R2 έχει το ρόλο της σταθεροποίησης των λογικών «0» και «1» (University of Cambridge, 2014).

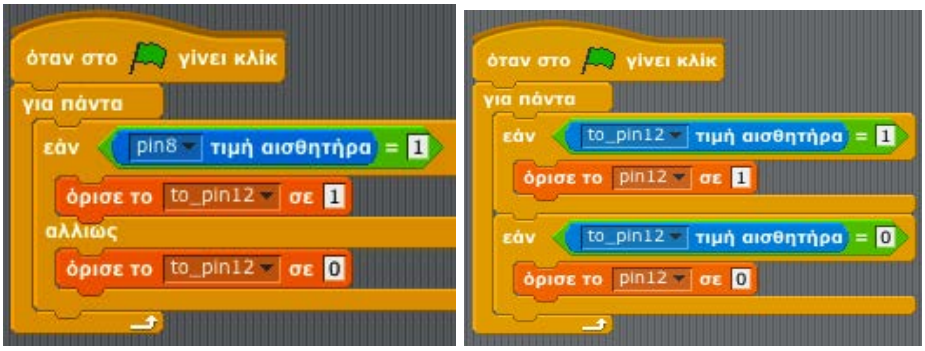
Στο περιβάλλον του Scratch στο RPi, δημιουργώντας τη μεταβλητή "p*in12*" και ορίζοντας την τιμή της σε "1", ενεργοποιούμε το pin12 στο GPIO, κάνοντας το LED να ανάψει, αντίστοιχα θέτοντας την τιμή "0" προκαλούμε το σβήσιμο της λυχνίας. Παράλληλα η τιμή του αισθητήρα *"pin8"* δηλώνει έμμεσα την τρέχουσα κατάσταση του κυκλώματος του διακόπτη: «0» όταν δεν τον διαρρέει ρεύμα και «1» όταν τον διαρρέει. Το πρόγραμμα στην εικόνα 2, ανάβει το LED όταν πατηθεί ο διακόπτης και το σβήνει όταν αφεθεί.

3.1 Απομακρυσμένος έλεγχος από Scratch σε Scratch

Χρησιμοποιώντας τη δυνατότητα του Scratch να επικοινωνεί και να αλληλεπιδρά με άλλα προγράμματα Scratch (Mesh method) ακόμα και όταν αυτά βρίσκονται σε διαφορετικούς υπολογιστές, μπορούμε να ελέγξουμε το άναμμα του LED από οποιοδήποτε PC μέσα στο σχολικό εργαστήριο Η/Υ, έχοντας το Raspberry Pi συνδεδεμένο στο τοπικό δίκτυο. Για μια τέτοια σύνοδο θα χρειαστεί να προστεθεί μία ρύθμιση για τη λειτουργία "Mesh", στους εμπλεκόμενους υπολογιστές. Η ρύθμιση αυτή αποτελείται από 10 απλά βήματα παραμετροποίησης του System Browser της εφαρμογής (Scratch wiki, 2014).

Ενεργοποίηση συνόδου "Host Mesh" και "Join Mesh": Μετά την προσθήκη της ρύθμισης "Mesh" και προκειμένου δύο Scratch projects να μπορούν να "συνομιλούν" μεταξύ τους, από το μενού του Scratch στο Raspberry Pi ενεργοποιείται η επιλογή *Host Mesh* (Shift + "Μοιράσου"), ενώ στην πλευρά του Scratch στο PC με όμοιο τρόπο επιλέγεται *Join Mesh*, δίνοντας τη διεύθυνση IP του Rapsberry Pi στο παράθυρο που θα εμφανιστεί.

Τα δύο προγράμματα της εικόνας 3, συνεργάζονται για να ανάβει το LED (εικόνα 2) όταν πατηθεί ο διακόπτης και να σβήνει όταν αφεθεί. Η δημιουργία μεταβλητής σε μια από τις δύο εφαρμογές Scratch, έχει ως αποτέλεσμα την εμφάνισή της ως αισθητήρας στην άλλη εφαρμογή, ενώ οποτεδήποτε οριστεί νέα τιμή, ενημερώνεται άμεσα και ο αντίστοιχος αισθητήρας. Στο PC η μεταβλητή "*to_pin12*" στο αριστερό μέρος της εικόνας 3, εμφανίζεται ως αισθητήρας στο περιβάλλον του RPi στο δεξιό μέρος, "μεταφέροντας" έτσι την τρέχουσα τιμή της από τον έναν υπολογιστή στον άλλον, κάθε φορά που αυτή μεταβάλλεται. Ο αισθητήρας "*pin8*" στο PC γίνεται αυτόματα διαθέσιμος στην ομάδα εντολών «Αισθητήρες» του Scratch, από τη στιγμή που επιτευχθεί η επικοινωνία με το αντίστοιχο project στο Raspberry Pi.



Εικόνα 3. *Συνεργαζόμενα προγράμματα που ελέγχουν το άναμμα του LED. Αριστερά το πρόγραμμα ελέγχου στο PC και δεξιά το πρόγραμμα στο Raspberry Pi*

3.2 Απομακρυσμένος έλεγχος από Scratch σε Python

Σε αυτόν τον τύπο ελέγχου η εφαρμογή Scratch στο PC επικοινωνεί με πρόγραμμα python στο Raspberry Pi.Το πλεονέκτημα αυτής της σύνδεσης είναι ότι στην πλευρά του RPi, δε χρειάζεται πλέον η χρήση πληκτρολογίου, ποντικιού και οθόνης καθώς το python script δεν απαιτεί γραφικό περιβάλλον για να γραφτεί και να εκτελεστεί. Η απομακρυσμένη σύνδεση (remote login) στο Raspberry Pi, μπορεί να γίνει με κατάλληλο πρόγραμμα από το PC (π.χ. Putty)

Η εγκατάσταση των αναγκαίων εργαλείων python (python-setuptools), καθώς επίσης και πακέτου βιβλιοθηκών python (scratchpy) στο RPi, είναι απαραίτητα για την επικοινωνία Scratch – Python (GitHub, 2014): Σε ένα Terminal παράθυρο του RPi χρειάζεται να εκτελεστούν οι παρακάτω εντολές:

- sudo apt-get install python-setuptools
- sudo easy_install scratchpy

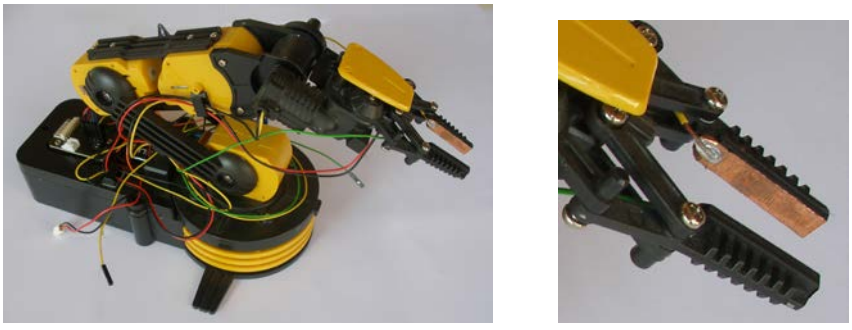
Στην εικόνα 4 φαίνονται τα δύο προγράμματα Scratch και python που αλληλεπιδρούν για τον έλεγχο των κυκλωμάτων διακόπτη - LED της εικόνας 2.
Για την επικοινωνία PC και RPi, θα πρέπει να ενεργοποιηθεί στο PC η επιλογή *Host Mesh* όπως περιγράφεται παραπάνω. Σε αυτό το στάδιο θα εμφανιστεί παράθυρο στο Scratch με την τρέχουσα IP του PC (π.χ. 192.168.1.64). Στο python script που θα εκτελεστεί στο Raspberry Pi, θα πρέπει να δοθεί η IP διεύθυνση του PC, όπως φαίνεται στην εντολή της γραμμής 8 στην εικόνα 4. Το python script, είναι υπεύθυνο για να ενεργοποιεί / απενεργοποιεί τα GPIO pins εξόδου, καθώς και για να ενημερώνει τους αισθητήρες με την κατάσταση των GPIO pins εισόδου. Όπως φαίνεται στην εικόνα 4 αριστερά, η εντολή στη γραμμή 14 ενεργοποιεί το GPIO pin12, ενώ η εντολή στη γραμμή 16 το απενεργοποιεί. Επιπλέον, η εντολή στη γραμμή 23 ενημερώνει τον αισθητήρα *"from_pin8"* για την κατάσταση του pin8.
Όπως μπορεί κανείς να παρατηρήσει, τα προγράμματα Scratch και Python που

```python
from time import sleep
import scratch
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BOARD)            # RPi board numbering
GPIO.setup(12, GPIO.OUT)            # θέσε το pin12 ως OUTPUT PORT
GPIO.setup(8, GPIO.IN)              # θέσε το pin8 ως INPUT PORT
# Εδώ γίνεται η σύνδεση με το απομακρυσμένο PC στην IP '192.168.1.64'
s = scratch.Scratch(host='192.168.1.64',port=42001)

while True:
        msg = s.receive()
        if (msg[1].has_key('to_pin12')):
         if (msg[1]['to_pin12'] == 1):
           GPIO.output(12, True)
         if (msg[1]['to_pin12'] == 0):
           GPIO.output(12, False)


       # ανανεώνει την επικοινωνία με το PC
        s.connect()
        if GPIO.input(8):
          # ανανέωσε την τιμή του αισθητήρα με την τρέχουσα τιμή του pin8 στο GPIO
          s.sensorupdate({'from_pin8' : GPIO.input(8)})
        else :
          s.sensorupdate({'from_pin8' : GPIO.input(8)})
```

Εικόνα 4. *Συνεργαζόμενα προγράμματα που ελέγχουν το άναμμα του LED. Δεξιά το πρόγραμμα ελέγχου στο PC και αριστερά το πρόγραμμα python στο Raspberry Pi*

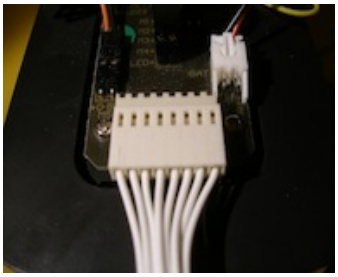
εκτελούνται στο Raspberry Pi και για τους δύο τύπους επικοινωνίας, έχουν ακριβώς τον ίδιο ρόλο: να αντιδρούν σε κάθε μεταβολή τιμής αισθητήρα και ανάλογα να ενεργοποιούν / απενεργοποιούν τα κατάλληλα GPIO pins στο RPi. Μπορεί να θεωρηθεί πως στο RPi εκτελείται το back-end μέρος του προγράμματος, ενώ στο PC το front-end κομμάτι του εκάστοτε project.

4. Έλεγχος ρομποτικού βραχίονα



Εικόνα 5. *Ο ρομποτικός βραχίονας OWI-535, μετά την προσαρμογή φύλλων χαλκού στο εσωτερικό των σιαγόνων της αρπάγης*

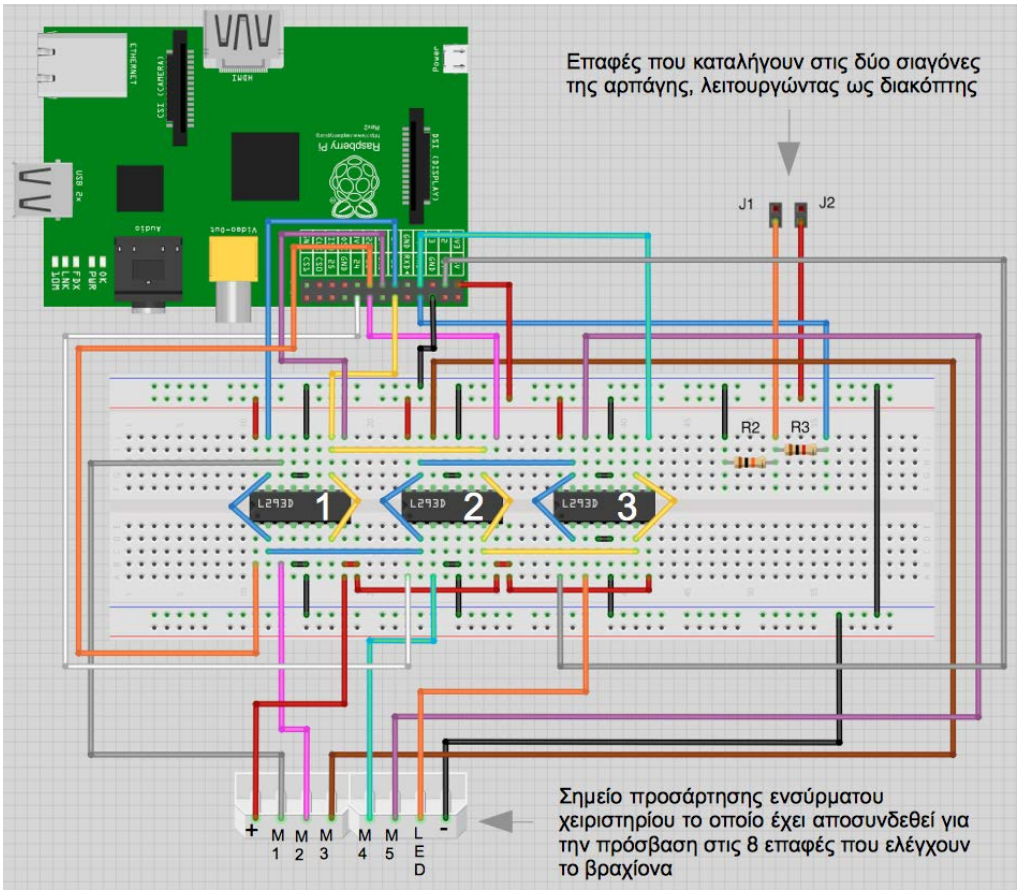
Ο ρομποτικός βραχίονας OWI-535 (εικόνα 5), είναι ένα κιτ συναρμολόγησης που αποτελείται από 5 μοτέρ και 1 LED. Κάθε μοτέρ κινεί διαφορετικό μέρος του βραχίονα (M1:Gripper, M2:Wrist Motion, M3:Elbow Motion, M4:Base Motion, M5:Base Rotation). Η λειτουργία του βραχίονα επιτυγχάνεται μέσω ενσύρματου χειριστηρίου, το οποίο μπορεί να αποσπαστεί αφήνοντας ελεύθερες τις 8 επαφές ελέγχου (εικόνα 6). Στην εργασία αυτή προτείνεται κατάλληλη σύνδεση των επαφών αυτών με το GPIO, ώστε να επιτρέπεται η πλήρης καθοδήγηση του ρομποτικού άκρου από προγραμματιστικά blocks στο Scratch. Επιπλέον, στο εσωτερικό των δύο σιαγόνων της αρπάγης (εικόνα 5), έχουν προσαρμοστεί φύλλα χαλκού κολλημένα με τις άκρες J1 και J2 αντίστοιχα, κυκλώματος αντιστάσεων (εικόνα 7), επιτρέποντας την όλη διάταξη να λειτουργεί ως διακόπτης.



Εικόνα 6. *Η πλακέτα του βραχίονα με την απόληξη των καλωδίων του χειριστηρίου αριστερά και χωρίς τα καλώδια, δεξιά.*

Για τον έλεγχο μικρών κινητήρων χρησιμοποιούνται ολοκληρωμένα κυκλώματα γνωστά ως motor drivers. Για τη συγκεκριμένη συνδεσμολογία, επιλέχτηκε το chip L293D. Το motor driver, προστατεύει το Raspberry Pi όταν η ένταση του ρεύματος ξεπεράσει κάποια όρια. Στο σχηματικό διάγραμμα της εικόνας 7, το πρώτο chip από τα αριστερά ελέγχει τα μοτέρ M1 και M2, το δεύτερο τα M3 και M4 και το τρίτο το μοτέρ M5 και το LED. Για να προγραμματιστεί να λειτουργήσει ένας κινητήρας χρειάζεται να λάβει από το GPIO (μέσω του L293D): α) το σήμα ενεργοποίησης (enable) λογικό “1” και β) τα σήματα εναλλαγής πολικότητας τροφοδοσίας (πίνακας 1) (New York University, 2014). Η πολικότητα της τροφοδοσίας των κινητήρων στο διάγραμμα, ρυθμίζεται από τα pin11 και pin12 του GPIO. Όταν τα pins τεθούν σε λογικά “0” και “1” αντίστοιχα, το μοτέρ περιστρέφεται κατά τη μία φορά, ενώ όταν τεθούν σε “1” και “0”, η περιστροφή αλλάζει φορά. Οποιοσδήποτε άλλος συνδυασμός λογικών τιμών σταματάει τον κινητήρα.

Το κύκλωμα διακόπτη-αντιστάσεων της εικόνας 2 παραπάνω, χρησιμοποιείται και εδώ (εικόνα 7), έχοντας τις σιαγόνες της αρπάγης στο ρόλο του διακόπτη κάθε φορά που έρχονται σε επαφή ή απομακρύνονται. Οι τιμές των αντιστάσεων είναι R2=10 kΩ και R3=1 kΩ, ενώ η ανίχνευση ροής ρεύματος μέσα από τις σιαγόνες γίνεται με τη σύνδεση του κυκλώματος στο input pin8 του GPIO.



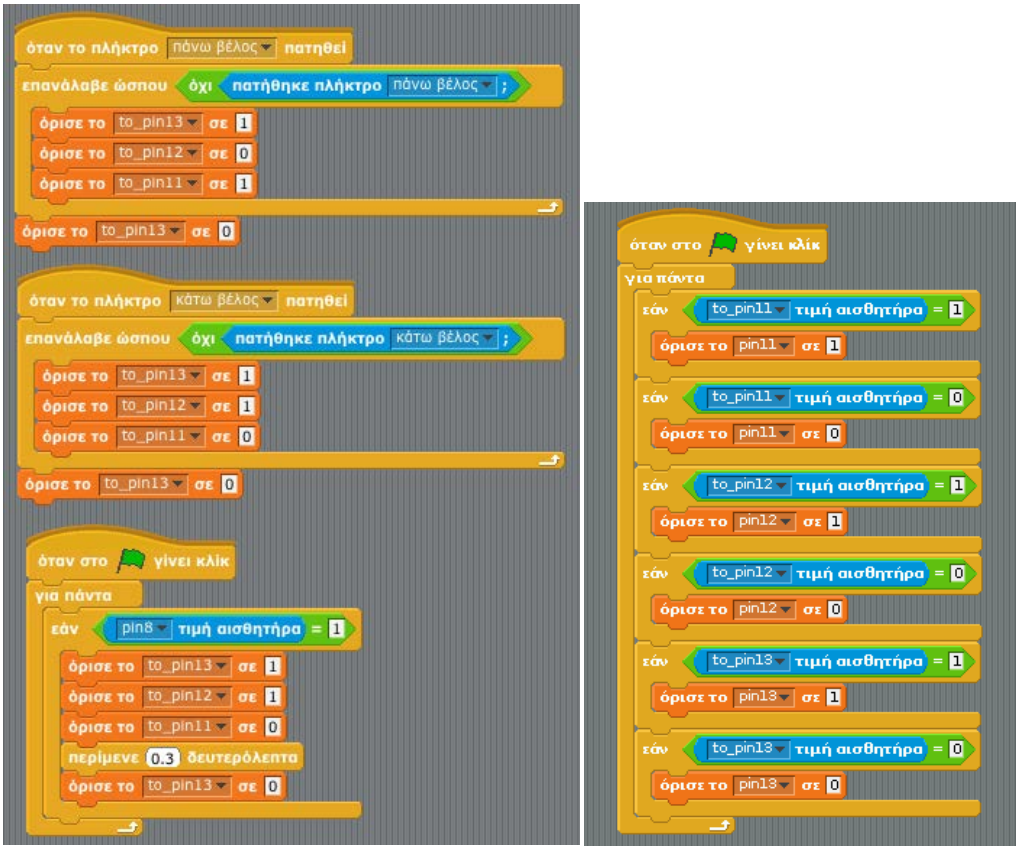
Εικόνα 7. *Σχηματικό διάγραμμα συνδεσμολογίας GPIO, breadboard και επαφών που ελέγχουν το βραχίονα*

Πίνακας 1. *Αντιστοίχιση motor driver, μοτέρ και pin ενεργοποίησης στο GPIO - Συνδυασμός τιμών των pin11 και pin12 που εναλλάσσει τη φορά περιστροφής όλων των κινητήρων, για τη συνδεσμολογία της εικόνας 7*

Motor Driver	**Μοτέρ**		**Enable GPIO pin**		**Περιστροφή**		**Αντίστροφη περιστροφή**	
					pin11	**pin12**	**pin11**	**pin12**
1	M1	M2	13	15	0	1	1	0
2	M3	M4	16	18				
3	M5	LED	7	3				

4.1 Απομακρυσμένος έλεγχος από Scratch σε Scratch

Αρχικά τίθενται σε επικοινωνία οι εφαρμογές Scratch, στο RPi και στο PC (*Host Mesh – Join* Mesh), σύμφωνα με την περιγραφή που δίνεται παραπάνω.



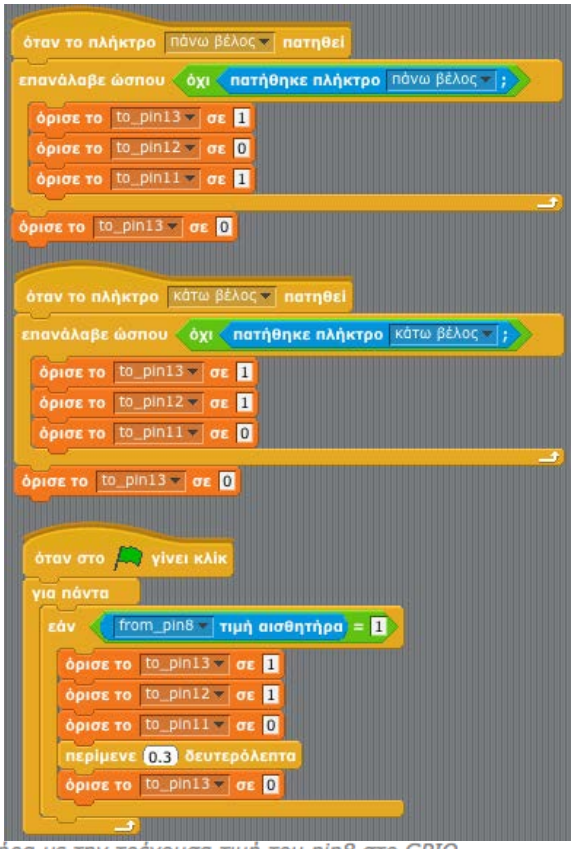
Εικόνα 8. *Συνεργαζόμενα προγράμματα για τον έλεγχο του ρομποτικού μοτέρ M1. Αριστερά το πρόγραμμα ελέγχου στο PC και δεξιά το πρόγραμμα στο Raspberry Pi*

Στο αριστερό μέρος της εικόνας 8, τα δύο πρώτα προγραμματιστικά blocks, εμφανίζουν το κυρίως πρόγραμμα στο PC που ανοίγει και κλείνει την αρπάγη (μοτέρ M1) του ρομποτικού βραχίονα. Οι εμπλεκόμενες μεταβλητές *"pin11"*, *"pin12"* και *"pin13"* προκύπτουν από τον πίνακα 1. Για όσο διάστημα διαρκεί το πάτημα του πλήκτρου "πάνω βέλος", το μοτέρ είναι σε λειτουργία. Σταματάει μόλις αφεθεί το πλήκτρο. Όμοια με το πάτημα του πλήκτρου "κάτω βέλος" το μοτέρ δουλεύει προς την αντίθετη κατεύθυνση. Με ανάλογο τρόπο μπορούν να δημιουργηθούν οι εντολές ελέγχου και των υπόλοιπων κινητήρων σε συνδυασμό με τα αντίστοιχα "Enable GPIO pin" του πίνακα 1. Όταν οι σιαγόνες της αρπάγης έρθουν σε επαφή, το κύκλωμα με τα άκρα J1 και J2 (εικόνα 7), διαρρέεται από ρεύμα το οποίο εντοπίζεται

από το pin8 του GPIO. Τότε ο αισθητήρας *"pin8"* στο τρίτο προγραμματιστικό block της εικόνας 8 γίνεται "1" και η αρπάγη ανοίγει για 0,3 δευτερόλεπτα. Στην άλλη πλευρά, το συνεργαζόμενο πρόγραμμα στο Raspberry Pi (εικόνα 8, δεξιά), είναι όμοιο με εκείνο του ελέγχου της λυχνίας (εικόνα 3, δεξιά), λειτουργώντας ως back-end μέρος του project.

4.2 Απομακρυσμένος έλεγχος από Scratch σε Python

```
from time import sleep
import scratch
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BOARD)
GPIO.setup(11, GPIO.OUT)
GPIO.setup(12, GPIO.OUT)
GPIO.setup(13, GPIO.OUT)
GPIO.setup(8, GPIO.IN)
# Επικοινωνία με το PC (192.168.1.64)
s = scratch.Scratch(host= \
      '192.168.1.64',port=42001)
while True:
      msg = s.receive()
      if (msg[1].has_key('to_pin11')):
       if (msg[1]['to_pin11'] == 1):
         GPIO.output(11, True)
       if (msg[1]['to_pin11'] == 0):
         GPIO.output(11, False)

      if (msg[1].has_key('to_pin12')):
       if (msg[1]['to_pin12'] == 1):
         GPIO.output(12, True)
       if (msg[1]['to_pin12'] == 0):
         GPIO.output(12, False)

      if (msg[1].has_key('to_pin13')):
       if (msg[1]['to_pin13'] == 1):
         GPIO.output(13, True)
       if (msg[1]['to_pin13'] == 0):
         GPIO.output(13, False)

      s.connect()
      if GPIO.input(8):
        # Ανανέωσε την τιμή του αισθητήρα με την τρέχουσα τιμή του pin8 στο GPIO
        s.sensorupdate({'from_pin8' : GPIO.input(8)})
      else :
        s.sensorupdate({'from_pin8' : GPIO.input(8)})
```



Εικόνα 9. *Συνεργαζόμενα προγράμματα για τον έλεγχο του ρομποτικού μοτέρ M1. Δεξιά, το πρόγραμμα ελέγχου στο PC και αριστερά το python script στο Raspberry Pi.*

Σε αυτόν τον τύπο ελέγχου, χρειάζεται πρώτα να ενεργοποιηθεί στο PC η επιλογή *Host Mesh*, ενώ η επικοινωνία συμπληρώνεται στο RPi, με την προσθήκη της

εντολής python που περιέχει την IP του PC (γραμμές 10-11, εικόνα 9).

Στην εικόνα 9, απεικονίζονται τα δύο μέρη της εφαρμογής με την οποία επιτυγχάνεται το άνοιγμα και το κλείσιμο της αρπάγης, καθώς και η αντίδραση των σιαγόνων της αρπάγης στην περίπτωση που αυτές έρθουν σε επαφή μεταξύ τους. Όμοια κι εδώ, έχει διατηρηθεί η ίδια φιλοσοφία στα δύο προγράμματα Scratch και python, με την περίπτωση του ελέγχου της λυχνίας LED (εικόνα 4).

5. Συμπεράσματα

Στο άρθρο αυτό παρουσιάστηκαν δύο έργα ηλεκτρονικών κατασκευών καθώς και η διαχείρισή τους μέσω προγραμματιστικών blocks, αξιοποιώντας τις GPIO εισόδους και εξόδους στο Raspberry Pi. Δόθηκε έμφαση στον απομακρυσμένο έλεγχο των έργων αυτών από PC, με χρήση προγραμμάτων Scratch που αλληλεπιδρούν μεταξύ τους και με συνεργαζόμενα προγράμματα Scratch – python. Παράλληλα δόθηκαν οδηγίες εγκατάστασης απαιτούμενου λογισμικού στο Raspberry Pi και απαραίτητων ρυθμίσεων στο Scratch για την επίτευξη της διαχείρισης από απόσταση των έργων αυτών.

Τα Scratch projects, που θα υιοθετήσουν τον προτεινόμενο απομακρυσμένο έλεγχο, μπορούν να σχεδιαστούν, φροντίζοντας το μέρος του προγράμματος που εκτελείται στο Raspberry Pi να είναι υπεύθυνο μόνο για να ενεργοποιεί / απενεργοποιεί εξόδους στο GPIO, σύμφωνα με την κατάσταση των αισθητήρων που προκύπτει, από τις επιλογές του χρήστη στο PC. Έτσι το πιο σύνθετο (και ενδιαφέρον ίσως), προγραμματιστικό μέρος μπορεί να υλοποιηθεί στο Scratch στο PC. Με αυτόν τον τρόπο ελαχιστοποιείται η παρέμβαση στο Raspberry Pi, σε πιθανές τροποποιήσεις του project.

Στην περίπτωση που εφαρμοστεί η επικοινωνία προγραμμάτων Scratch (PC) με python (RPi), η χρήση οθόνης, πληκτρολογίου και ποντικιού στο Raspberry Pi δεν είναι απαραίτητη. Αρκεί η απομακρυσμένη σύνδεση σε αυτό από το PC δικτυακά, για την εκτέλεση του python script σε γραμμή εντολών Unix. Το πλεονέκτημα αυτού είναι σημαντικό, ιδιαίτερα όταν το Raspberry Pi χρησιμοποιείται ως τμήμα αυτόνομων κινούμενων projects, όπως σε ρομποτικές κατασκευές. Αρκεί μια επαναφορτιζόμενη μπαταρία για τροφοδοσία κι ένας προσαρμογέας usb wifi για πρόσβαση σε δίκτυο. Η απομακρυσμένη σύνδεση από PC θα παρέχει τη δυνατότητα ελέγχου της κατασκευής αλλά και τροποποίησης του κώδικα που εκτελείται στο RPi.

Όταν επιλεγεί η επικοινωνία PC – RPi μέσω Scratch προγραμμάτων, είναι δυνατή η χρήση περισσότερων PCs στο σχολικό εργαστήριο για τον έλεγχο ηλεκτρονικών κυκλωμάτων στο Raspberry Pi, επιτρέποντας έτσι τη συμμετοχή μεγαλύτερου αριθμού μαθητών και υλοποιώντας πιθανώς πολυσκελή projects. Ως εφαρμογή αυτού στην περίπτωση του ρομποτικού βραχίονα, θα μπορούσαν να χρησιμοποιηθούν 5 PCs, ένα για κάθε κινητήρα, με ισάριθμες ομάδες μαθητών να σχεδιάζουν τα

αντίστοιχα προγράμματα ελέγχου.

Αναφορές

Cymplecy (2014). *ScratchGPIO – Introduction for Beginners.* Ανάκτηση από το http://cymplecy.wordpress.com/2013/04/22/scratch-gpio-version-2-introduction-for-beginners/

GitHub (2014). *scratchpy. A python interface to Scratch.* Ανάκτηση από το https://github.com/pilliq/scratchpy

New York University (2014). *DC Motor Control Using an H-Bridge.* ITP Physical Computing. Ανάκτηση από το http://itp.nyu.edu/physcomp/Labs/DCMotorControl

O'Sullivan, D. & Igoe, T. (2004). *Physical Computing. Sensing and Controlling the Physical World with Computers*. Thomson Course Technology

Scratch wiki (2014). *Mesh.* Ανάκτηση από το http://wiki.scratch.mit.edu/wiki/Mesh

University of Cambridge (2014). *Buttons and Switches.* Physical computing with Raspberry Pi. Ανάκτηση από το https://www.cl.cam.ac.uk/projects/raspberrypi/tutorials/robot/buttons_and_switches/

Wikipedia (2014). *Physical computing.* Ανάκτηση από το http://en.wikipedia.org/wiki/Physical_computing

Abstract

The remote control of electronic circuits, connected with the Raspberry Pi computer via the Scratch programming environment, is the subject that this work is dealing. Initially, required software modifications on Raspberry Pi are presented, in order to enable electronic circuit control via the Scratch application. A modified Scratch environment on PC can communicate and interact with other Scratch or python programs, found on Raspberry Pi computer. Such a modification is presented at next, while at the same time, the implementation of two remote control Scratch projects from PC, is described, that of LED operation and that of robotic arm manipulation. There are proposed wiring diagrams with Raspberry Pi and management programs are provided in Scratch and python.

Keywords: Raspberry Pi, GPIO, Scratch, Host Mesh session, remote control, python, robotic arm edge.