ΠΑΝΕΠΙΣΤΗΜΙΟ
ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

Ενσωματωμένα Συστήματα

Ενότητα: ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ: CIRC-05

Δρ. Μηνάς Δασυγένης

mdasyg@ieee.org

Τμήμα Μηχανικών Πληροφορικής και Τηλεπικοινωνιών

Εργαστήριο Ψηφιακών Συστημάτων και Αρχιτεκτονικής Υπολογιστών

http://arch.icte.uowm.gr/mdasyg

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ψηφιακά Μαθήματα του Πανεπιστημίου Δυτικής Μακεδονίας**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Περιεχόμενα

1. Σκοπός της άσκησης

- Χρήση του αναπτυξιακού περιβάλλοντος του Arduino[1] για τη συγγραφή και τη μεταφόρτωση προγραμμάτων στην πλακέτα του Arduino Uno.
- Δημιουργία κυκλώματος ελέγχου 8 LEDs με χρήση του καταχωρητή ολίσθησης 74HC595
- Κατανόηση της λειτουργίας του καταχωρητή ολίσθησης.

2. Παραδοτέα

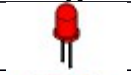
- Ένα αρχείο zip με τα project **CIRC_05**, **CIRC_05_c1, CIRC_05_c2** που θα δημιουργήσετε.
- Ένα screenshot από το arduino IDE που δείχνει ότι η μεταγλώττιση (compile) έγινε με επιτυχία και ταυτόχρονα εμφανίζει το μέγεθος του δυαδικού σχεδίου για κάθε κύκλωμα.
- Ένα video 5-10 sec επίδειξης του κυκλώματος μαζί με ηχητική περιγραφή για κάθε κύκλωμα.

Χρόνος ολοκλήρωσης εργαστηρίου: 45 λεπτά.

Ο καταχωρητής μετατόπισης είναι ένα ολοκληρωμένο κύκλωμα, το οποίο δίνει 8 πρόσθετες εξόδους (για τον έλεγχο ηλεκτρονικών στοιχείων) χρησιμοποιώντας μόνο τρία pins του Arduino. Επίσης μπορούν να συνδεθούν πολλοί καταχωρητές μετατόπισης μαζί ώστε να έχουμε επιπρόσθετες εξόδους χρησιμοποιώντας πάλι τα ίδια pins. Για να το χρησιμοποιήσετε θέτετε το data pin σε HIGH ή LOW, δίνετε έναν παλμό στο ρολόι, και επαναλαμβάνετε μέχρι να έχετε 8 bits δεδομένων. Τότε δίνετε έναν παλμό στο latch και μεταφέρονται τα 8 bits δεδομένων στα αντίστοιχα pins του καταχωρητή μετατόπισης. Για βαθύτερη μελέτη της λειτουργίας του ανατρέξτε στο σχετικό λήμμα της Wikipedia.

3. Κατασκευή του κυκλώματος CIRC-05

Για την εκπόνηση του κυκλώματος CIRC-05 απαιτούνται τα εξής μέρη:

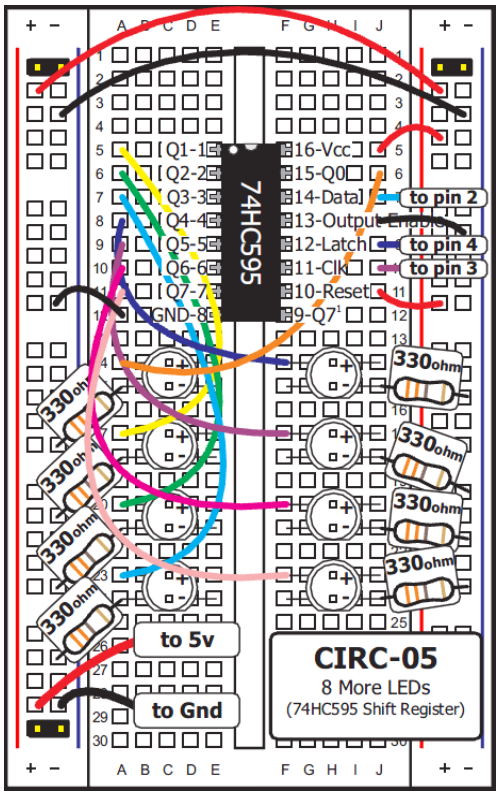
Στοιχείο	Περιγραφή και ποσότητα
	**8x** 5mm κόκκινα LEDs
	**19x** Καλώδια
	**8x** Αντιστάσεις 330 Ohm ***(Πορτοκαλί – Πορτοκαλί – Καφέ)***
	**1x** Καταχωρητής ολίσθησης 74HC595

1 Χρησιμοποιήθηκε υλικό από το SparkFun Inventors Kit for Arduino

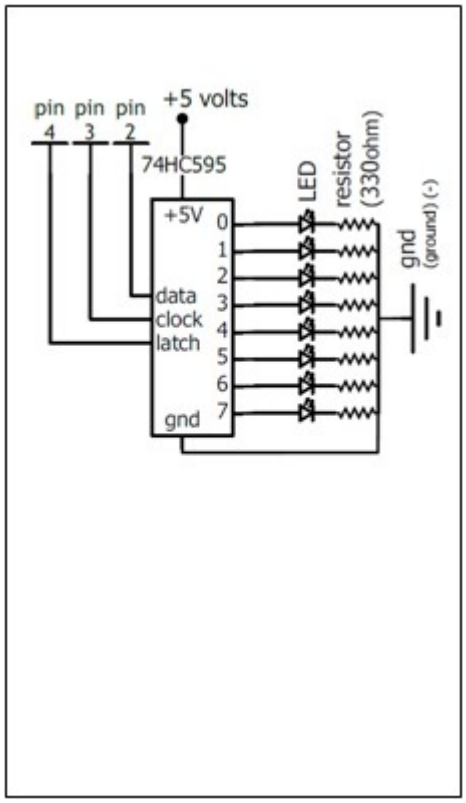
Τα φυλλάδια δεδομένων (datasheets) των υλικών βρίσκονται παρακάτω:

- LED
- Καταχωρητής ολίσθησης 74HC595
- Αντίσταση 330 Ohm

Κατασκευάστε το κύκλωμα, σύμφωνα με τις παρακάτω σχηματικές αναπαραστάσεις:



Σχήμα 1



Σχήμα 2

Τοποθετήστε τον καταχωρητή ολίσθησης στο breadboard. Η **ημικυκλική εγκοπή του**, θα πρέπει να είναι σύμφωνη με τη φορά του Σχήματος 1.

Συνδέστε το **θετικό ακροδέκτη** του κάθε LED στο αναγραφόμενο pin του **καταχωρητή ολίσθησης**, και μια **αντίσταση 330 Ohm** στον κάθε **αρνητικό ακροδέκτη**, που να πηγαίνει στη **γείωση**.

Συνδέστε το **8ο** και το **13ο** pin του καταχωρητή ολίσθησης στη **γείωση** καθώς και το **10ο** και **16ο** στο **+5 V**.

Συνδέστε τα pin **11**, **12** και **14** του καταχωρητή ολίσθησης στα αναγραφόμενα **pins του Arduino**.

***** Προσοχή: Αν συνδέσετε ανάποδα το IC του καταχωρητή ολίσθησης μπορεί να καταστρέψει το ολοκληρωμένο κύκλωμα. Διαβάστε το datasheet και στη συνέχεια ακολουθήστε πιστά τις οδηγίες σύνδεσης. *****

Τέλος συνδέστε μεταξύ τους τις γειώσεις και τα +5 V του breadboard.

Στο σημείο αυτό θα πρέπει να γνωρίζετε κάποιες βασικές πληροφορίες για τον καταχωρητή μετατόπισης (Serial to parallel converter):

Διαβάστε το εξής λήμμα στη Wikipedia.

Στο παρόν κύκλωμα η χρήση του καταχωρητή μετατόπισης θα μας δώσει 8 επιπλέον εξόδους για τον έλεγχο των LEDs, χρησιμοποιώντας μόνο 3 pins του Arduino.

4. Προγραμματισμός του κυκλώματος CIRC-05

Αντιγράψτε τον παρακάτω κώδικα στο προγραμματιστικό περιβάλλον του Arduino.

(Εναλλακτικά κατεβάστε τον από εδώ)

Κώδικας προγραμματισμού του κυκλώματος

```
/*-----------------------------------------------------
 *|  Arduino Experimentation Kit Example Code |
 *|  CIRC-05 .: 8 More LEDs :. (74HC595 Shift Register) |
 *-----------------------------------------------------
 *
 *We have already controlled 8 LEDs however this does it in a slightly
 *different manner. Rather than using 8 pins we will use just three
 *and an additional chip.
 *
 */
//Pin Definitions
//Pin Definitions
//The 74HC595 uses a serial communication
//link which has three pins
int data = 2;
int clock = 3;
int latch = 4;

//Used for single LED manipulation
int ledState = 0;
const int ON = HIGH;
const int OFF = LOW;

/*
 * setup() - this function runs once when you turn your Arduino on
 * We set the three control pins to outputs
 */
void setup()
{
  pinMode(data, OUTPUT);
  pinMode(clock, OUTPUT);
  pinMode(latch, OUTPUT);
}


/*
 * loop() - this function will start after setup finishes and then repeat
 * we set which LEDs we want on then call a routine which sends the states to the 74HC595
 */
void loop()  // run over and over again
{
 int delayTime = 100; //the number of milliseconds to delay between LED updates
 for(int i = 0; i < 256; i++){
  updateLEDs(i);
  delay(delayTime);
 }
```

```
}

/*
 *updateLEDs() - sends the LED states set in ledStates to the 74HC595
 *sequence
 */
void updateLEDs(int value){
 digitalWrite(latch, LOW);  //Pulls the chips latch low
 shiftOut(data, clock, MSBFIRST, value); //Shifts out the 8 bits to the
                                         //shift register
  digitalWrite(latch, HIGH);   //Pulls the latch high displaying the data
}

/*
 * updateLEDsLong() - sends the LED states set in ledStates to the 74HC595
 * sequence. Same as updateLEDs except the shifting out is done in software
 * so you can see what is happening.
 */
void updateLEDsLong(int value){
  digitalWrite(latch, LOW);  //Pulls the chips latch low
  for(int i = 0; i < 8; i++){  //Will repeat 8 times (once for each bit)
  int bit = value & B10000000; //We use a "bitmask" to select only the eighth
          //bit in our number (the one we are addressing this time through
  value = value << 1;  //we move our number up one bit value
                       //so next time bit 7 will be
                      //bit 8 and we will do our math on it
  if(bit == 128){digitalWrite(data, HIGH);}
       //if bit 8 is set then set our data pin high
  else{digitalWrite(data, LOW);} //if bit 8 is unset then set the data pin low
  digitalWrite(clock, HIGH);   //the next three lines pulse the clock pin
  delay(1);
  digitalWrite(clock, LOW);
  }
  digitalWrite(latch, HIGH);
       //pulls the latch high shifting our data into being displayed
}

//These are used in the bitwise math that we use to change individual LEDs
//For more details http://en.wikipedia.org/wiki/Bitwise_operation
int bits[] = {B00000001, B00000010, B00000100, B00001000, B00010000, B00100000,
B01000000, B10000000};
int masks[] = {B11111110, B11111101, B11111011, B11110111, B11101111, B11011111,
B10111111, B01111111};
/*
 * changeLED(int led, int state) - changes an individual LED
 * LEDs are 0 to 7 and state is either 0 - OFF or 1 - ON
 */
void changeLED(int led, int state){
  ledState = ledState & masks[led];  //clears ledState of the bit we are addressing
  if(state == ON){ledState = ledState | bits[led];}
//if the bit is on we will add it to ledState
  updateLEDs(ledState);  //send the new LED state to the shift register
 }
```

Αποθηκεύστε το ως CIRC_05 και στη συνέχεια φορτώστε το πρόγραμμα στην πλακέτα.

Αν το Arduino σβήνει, ελέγξτε μήπως έχετε τοποθετήσει ανάποδα τον καταχωρητή ολίσθησης.

5. Παραμετροποίηση του κυκλώματος CIRC-05

5.1 Υλοποίηση του προγράμματος δίχως τη χρήση της έτοιμης συνάρτησης shiftOut().

Αλλάξτε στη loop() τη γραμμή **updateLEDs(i);** σε **updateLEDsLong(i);** και φορτώστε το πρόγραμμα στην πλακέτα.

Παρατηρήστε ότι ενώ **δεν αλλάζει κάτι στη λειτουργία** του κυκλώματος, ο **κώδικας**, μας επιτρέπει πλέον να "επικοινωνούμε" με το chip **1 bit τη φορά**.

Για περισσότερες λεπτομέρειες **διαβάστε τα σχόλια** του κώδικα της συνάρτησης **updateLEDsLong()** και για μεγαλύτερη εμβάθυνση μπορείτε να ανατρέξτε στο παρακάτω λήμμα της Wikipedia.

5.2 Έλεγχος μεμονωμένων LEDs

Μπορείτε επίσης να ελέγξετε μεμονωμένα LEDs, όπως είχατε κάνει στο CIRC-02, χρησιμοποιώντας την changeLED(<αριθμός LED>,<ON/OFF>);

Για παράδειγμα στη loop(), αντικαταστήστε τον υπάρχοντα κώδικα με τον παρακάτω, αποθηκεύστε το ως CIRC_05_c1 και φορτώστε το στο Arduino:

```
int delayTime = 100; //the number of milliseconds to delay between LED updates
 for(int i = 0; i < 8; i++){
  changeLED(i,ON);
  delay(delayTime);
 }
 for(int i = 0; i < 8; i++){
  changeLED(i,OFF);
  delay(delayTime);
 }
```

5.3 Υλοποίηση ενός δικού σας LED animation

Ορίστε μια δική σας συνάρτηση με **όνομα my_animation()**, η οποία θα υλοποιεί ένα LED animation και καλέστε την, όπως έχει περιγραφεί στο CIRC-02 6.2. **Μπορείτε αν θέλετε να χρησιμοποιήσετε την ίδια συνάρτηση που είχατε υλοποιήσει στο CIRC-02 δ2, τροποποιώντας την κατάλληλα, ώστε να κάνει compile χωρίς σφάλματα** (αλλαγή της digitalWrite() σε changeLED() κτλ).

Αποθηκεύστε το ως CIRC_05_c2 και φορτώστε το στο Arduino.