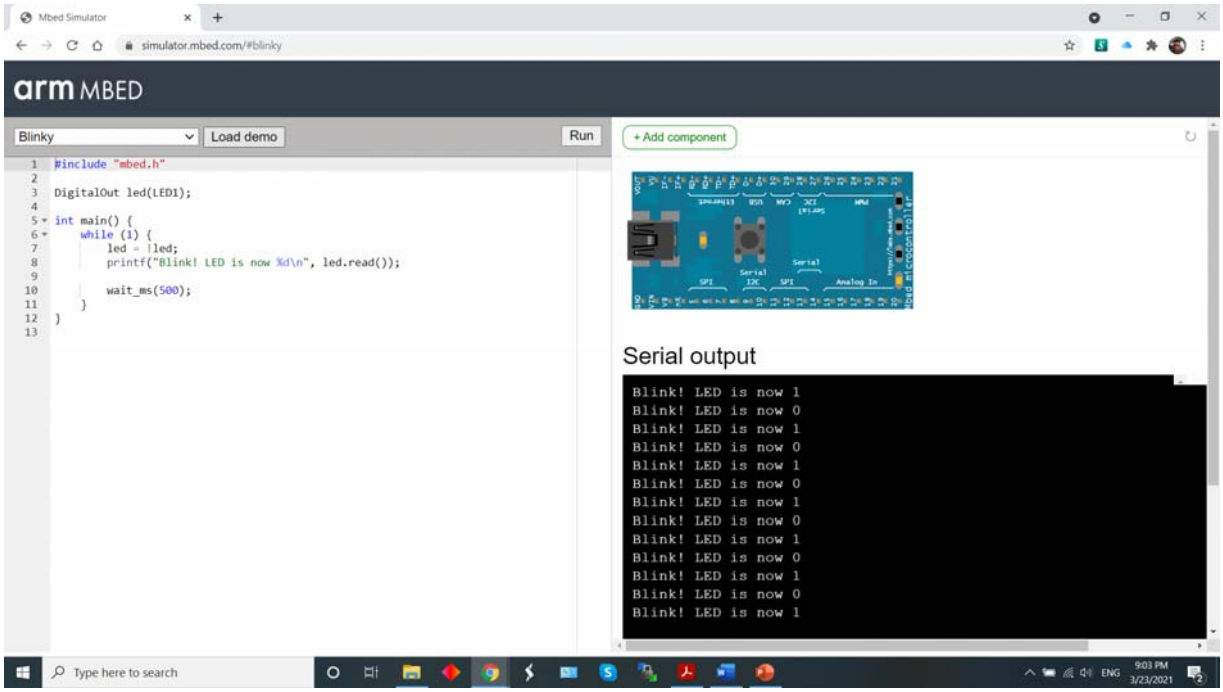
Εργαστήριο 5

Σκοπός του συγκεκριμένου εργαστηρίου είναι η εξομοίωση του αισθητήρα θερμοκρασίας - υγρασίας **Sensor**(**SHT31**) και της τεχνικής ελέγχου αναλογικών κυκλωμάτων **Pulse width modulation** (**PWM**) με το εργαλείο mbed simulator της ARM (https://simulator.mbed.com/) καθώς και η ενσωμάτωση και έλεγχος των παραπάνω στοιχείων από μια πλατφόρμα ανάπτυξης βασισμένη σε ARM επεξεργαστή. Με τον τρόπο αυτό προσομοιώνουμε την αναπτυξιακή πλακέτα που θα είχαμε δια ζώσης στο εργαστήριο και πάνω στην οποία θα συνδέαμε τον αισθητήρα και θα εξετάζαμε και την τεχνική **PWM.**

Ο προσομοιωτής, σας επιτρέπει να εκτελείτε τις εφαρμογές Mbed OS 5 απευθείας στον υπολογιστή σας, ώστε να μπορείτε να ελέγχετε και να επαληθεύετε γρήγορα εφαρμογές χωρίς να τις τρέχετε σε πραγματική πλακέτα. Αυτό είναι ένα πολύτιμο εργαλείο εκμάθησης, καθώς μαθαίνετε γρήγορα πώς λειτουργεί το Mbed. Είναι επίσης πολύ χρήσιμο για την ανάπτυξη σύνθετων εφαρμογών. Στην ARM, έχουν χρησιμοποιήσει τον προσομοιωτή για εφαρμογές βασισμένες στο mbed-http, τη στοίβα Mbed LoRaWAN, το uTensor πολλές άλλες τεχνολογίες.



Ο προσομοιωτής δείχνει τον code editor στα αριστερά. Μπορείτε να αλλάξετε τον κώδικα σε εκείνο το πεδίο και να κάνετε κλικ στο Compile για να τον εκτελέσετε στον προσομοιωτή. Υπάρχει ένα ευρύ φάσμα διαθέσιμων demo-εφαρμογών, από περιφερειακές εφαρμογές (όπως η δημοφιλής οθόνη LCD C12832) έως εφαρμογές δικτύου. Επιλέξτε την εφαρμογή στο αναπτυσσόμενο μενού και κάντε κλικ στο Φόρτωση(Load Demo). Η εφαρμογή φορτώνεται αυτόματα.

Αισθητήρας SHT31

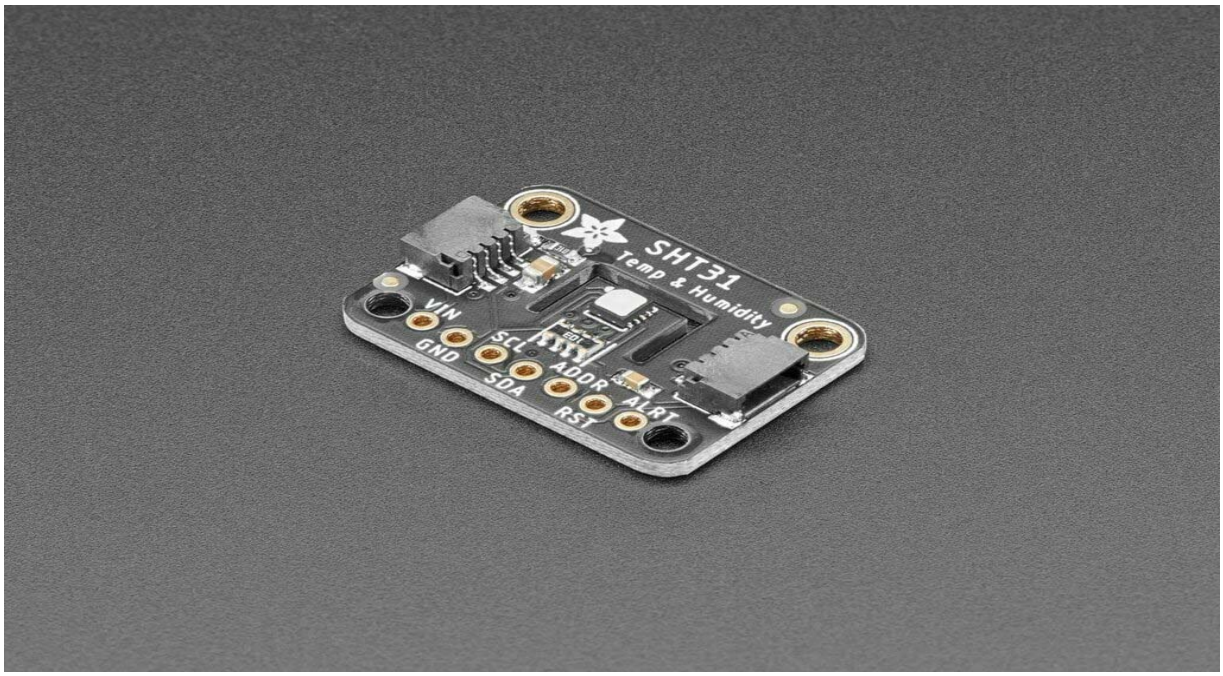
Ο αισθητήρας Grove - Temp & Humi (SHT31) είναι ένας πολύ αξιόπιστος, ακριβής, με γρήγορη απόκριση ενσωματωμένος αισθητήρας θερμοκρασίας και υγρασίας. Ο αισθητήρας (chip) που χρησιμοποιείται στη μονάδα έχει σχεδιαστεί με την τεχνολογία CMOSens® της Sensirion. Το τσιπ είναι καλά βαθμονομημένο, γραμμικό και παρέχει ψηφιακή έξοδο. Υποστηρίζει ταχύτητες επικοινωνίας έως 1MHz. Παρέχει ψηφιακή έξοδο πλήρως βαθμονομημένη, γραμμική με αντισταθμισμένη θερμοκρασία Το SHT31 έχει εκτεταμένο εύρος υγρασίας 0 έως 100% RH (Σχετική υγρασία). Το τυπικό εύρος είναι 20 έως 80% RH. Έχει ακρίβεια ± 2% RH.

Συνοπτικά τα Χαρακτηριστικά του αισθητήρα SHT31

- Πολύ αξιόπιστος, ακριβής και γρήγορος χρόνος απόκρισης
- Grove συμβατό και εύκολο στη χρήση
- Καλά βαθμονομημένο, γραμμικό, αντισταθμισμένο για ψηφιακή έξοδο
- Ιδιαίτερα φιλική προς τον χρήστη βιβλιοθήκη ανάπτυξης
- Διεύθυνση I2C 0x44 - true I2C interface for easy reading

Σε αντίθεση με τους προηγούμενους αισθητήρες SHT, αυτός ο αισθητήρας έχει διασύνδεση I2C. Επίσης διαθέτει δύο επιλογές διεύθυνσης. Αυτό σημαίνει ότι χρησιμοποιεί τα δύο καλώδια δεδομένων / ρολογιού I2C που διατίθενται στους περισσότερους μικροελεγκτές και μπορεί να μοιραστεί αυτές τις ακίδες με άλλους αισθητήρες, εφόσον δεν έχουν ίδια διεύθυνση. Για μελλοντική αναφορά, η προεπιλεγμένη διεύθυνση I2C είναι 0x44 και μπορείτε επίσης να επιλέξετε διεύθυνση 0x45 συνδέοντας τον πείρο ADDR σε σήμα υψηλής τάσης.

Είναι επίσης συμβατό με 3V ή 5V, ώστε να μπορείτε να τροφοδοτείτε και να επικοινωνείτε μαζί του χρησιμοποιώντας σχεδόν οποιονδήποτε μικροελεγκτή ή μικροϋπολογιστή.



Το πρωτόκολλο επικοινωνίας I2C

Το IIC (Inter – Integrated Circuit) είναι ένας δίαυλος επικοινωνίας ενσωματωμένος σε πολλές συσκευές όπως αισθητήρες, RTC και EEPROM. Το IIC αναφέρεται και ως I2C (I^2C) σε πολλά τεχνικά κείμενα.

Το I2C πρωτόκολλο αναπτύχθηκε αρχικά από την εταιρεία Philips, αλλά στις μέρες μας έχει γίνει ένα πλατιά διαδεδομένο πρωτόκολλο υιοθετημένο από πολλές εταιρείες κατασκευής ολοκληρωμένων κυκλωμάτων. Το πρωτόκολλο I2C είναι ιδανικό για τη σύζευξη χαμηλής ταχύτητας περιφερειακών με την μητρική πλακέτα ή οπουδήποτε αλλού που χρειάζεται μία αξιόπιστη επικοινωνία μεταξύ συσκευών σε μικρές αποστάσεις. Το πρωτόκολλο I2C παρέχει κατευθυνόμενη επικοινωνία με επαλήθευση (acknowledge). Οι I2C συσκευές χρησιμοποιούν μόνο δύο ακροδέκτες για τη μεταφορά δεδομένων αντί για οκτώ ή περισσότερους όπως στον παραδοσιακό δίαυλο. Αυτές οι γραμμές ονομάζονται SCL (Serial Clock) που συγχρονίζει τη μεταφορά δεδομένων μεταξύ δύο τσιπς και SDA (Serial Data) για μεταφορά δεδομένων. Αυτός ο μειωμένος αριθμός γραμμών επικοινωνίας έχει σαν αποτέλεσμα το μικρότερο μέγεθος των τσιπς και στη μειωμένη κατανάλωση ρεύματος, κάνοντας το ιδανικό σε συσκευές με απαιτήσεις μικρότερης επιφάνειας. Αυτές οι δύο γραμμές SDA και SCL κάνουν το πρωτόκολλο I2C ένα 2-wire interface. Σε κάποια τεχνικά κείμενα το πρωτόκολλο I2C αναφέρεται και σαν Two-wire Serial Interface (TWI).

Pulse width modulation (PWM) - Διαμόρφωση διάρκειας παλμών

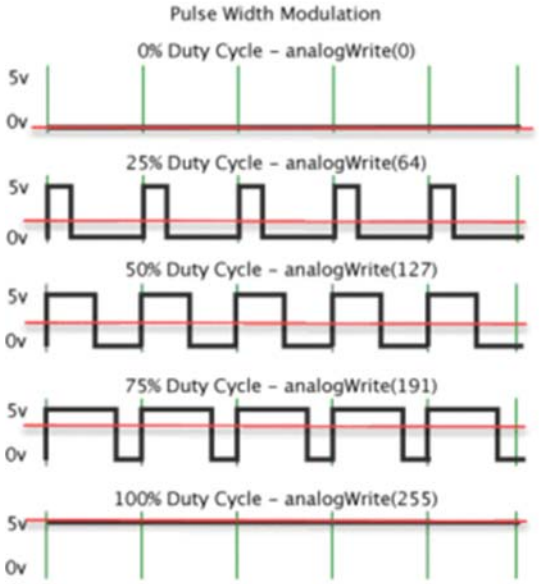
Τα συστήματα υπολογιστών συχνά χρειάζονται διασύνδεση με συσκευές που έχουν αναλογικές εξόδους. Η διαμόρφωση πλάτους παλμού (PWM) είναι μια οικεία μέθοδος για την προσομοίωση αναλογικών σημάτων χρησιμοποιώντας ψηφιακές εξόδους. Με αυτήν την τεχνική, η ποσότητα ισχύος που λαμβάνει ένα σύστημα εξαρτάται από τον κύκλο λειτουργίας του σήματος PWM. Μεταβάλλοντας τον κύκλο λειτουργίας ή την αναλογία χρόνου του σήματος από υψηλή προς χαμηλή, ένα ψηφιακό σύστημα μπορεί να μεταβάλλει συνεχώς την έξοδο ενός δεδομένου συστήματος διατηρώντας παράλληλα μια σταθερή συχνότητα.

Η διαμόρφωση πλάτους παλμού (PWM) είναι μια ισχυρή τεχνική για τον έλεγχο αναλογικών κυκλωμάτων με ψηφιακές εξόδους μικροεπεξεργαστή. Η PWM χρησιμοποιείται σε μια μεγάλη ποικιλία εφαρμογών, από τη μέτρηση και τις επικοινωνίες έως τον έλεγχο ισχύος και τη μετατροπή.

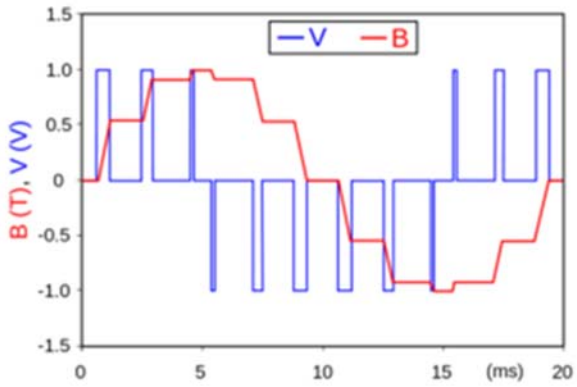
Τα ψηφιακά σήματα έχουν δύο θέσεις: ενεργοποίηση ή απενεργοποίηση, που ερμηνεύονται σε συντομότερο σημείο ως 1 ή 0. Τα αναλογικά σήματα, από την άλλη πλευρά, μπορούν να είναι ενεργοποιημένα, απενεργοποιημένα, μισά, δύο τρίτα μέχρι να ενεργοποιηθούν και έχουν άπειρο αριθμό θέσεων μεταξύ 0 και 1 είτε πλησιάζει το 1 είτε κατεβαίνει στο μηδέν. Αναλογικά και ψηφιακά σήματα αντιμετωπίζονται πολύ διαφορετικά στα ηλεκτρονικά, αλλά πολύ συχνά πρέπει να δουλεύουν μαζί (τότε το ονομάζουμε «μικτά ηλεκτρονικά σήματα»). Μερικές φορές πρέπει να πάρουμε ένα αναλογικό (πραγματικό κόσμο) σήμα εισόδου (π.χ. θερμοκρασία) σε έναν μικροελεγκτή (που καταλαβαίνει μόνο ψηφιακά). Συχνά οι μηχανικοί μεταφράζουν αυτήν την αναλογική είσοδο σε ψηφιακή είσοδο για τον μικροελεγκτή (MCU) χρησιμοποιώντας έναν μετατροπέα αναλογικού σε ψηφιακό. Τι γίνεται όμως με τις εξόδους;

Το PWM είναι ένας τρόπος ελέγχου αναλογικών συσκευών με ψηφιακή έξοδο. Μπορείτε να εξάγετε ένα σήμα διαμόρφωσης από μια ψηφιακή συσκευή, όπως ένα MCU για να οδηγήσετε μια αναλογική συσκευή. Είναι ένα από τα κύρια μέσα με τα οποία οι MCU οδηγούν αναλογικές συσκευές όπως κινητήρες

μεταβλητής ταχύτητας, φώτα με δυνατότητα ρύθμισης, ενεργοποιητές και ηχεία. Ωστόσο, το PWM δεν είναι πραγματική αναλογική έξοδος. Το PWM «πλαστοποιεί» ένα αναλογικό αποτέλεσμα, εφαρμόζοντας ισχύ σε παλμούς ή μικρές εκρήξεις ρυθμιζόμενης τάσης.



Σχήμα 1 : Ένα παράδειγμα σήματος PWM που εμφανίζεται σε διάφορους κύκλους λειτουργίας και υψηλής τάσης 5 βολτ. Η κόκκινη γραμμή είναι η μέση τάση που εφαρμόζεται στην κινούμενη συσκευή (π.χ. ένας κινητήρας).

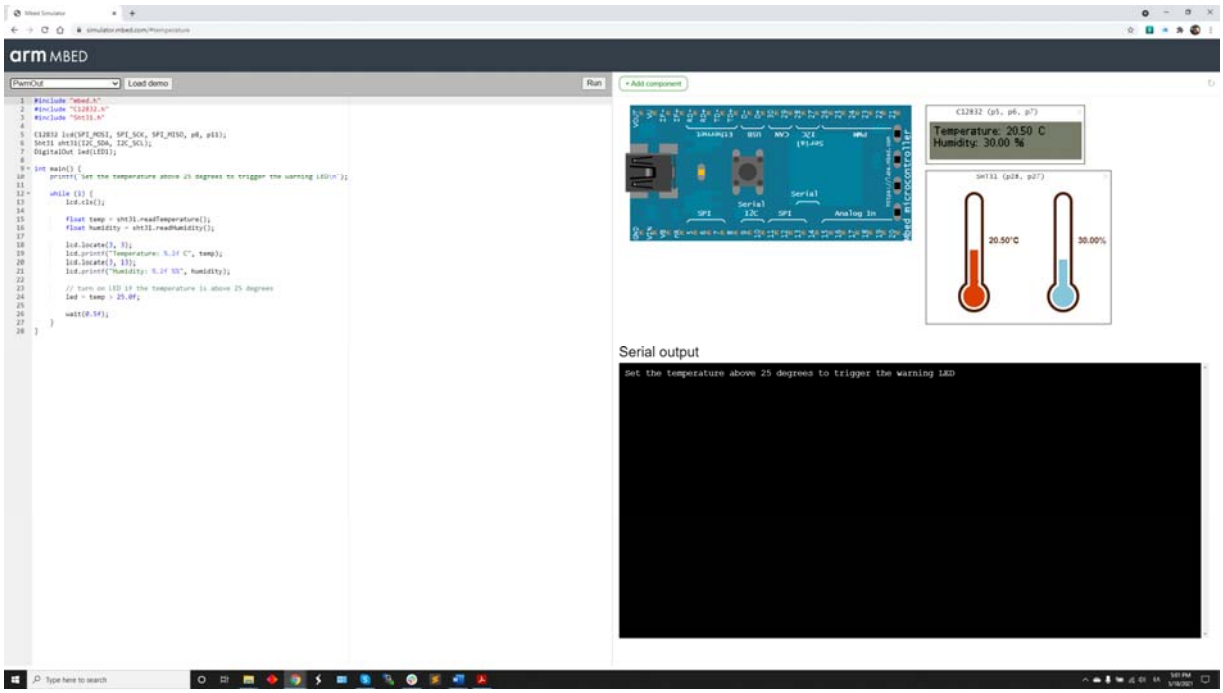


Σχήμα 2 : Οι μπλε γραμμές είναι έξοδος PWM από ένα MCU και η κόκκινη γραμμή είναι η μέση τάση. Σε αυτήν την περίπτωση, το πλάτος παλμού (και ο αντίστοιχος κύκλος λειτουργίας) αλλάζει έτσι

ώστε η μέση τάση να μοιάζει περισσότερο με αναλογική έξοδο που δεν βρίσκεται σε σταθερή κατάσταση, όπως φαίνεται στο σχήμα 1.

Οι ασκήσεις που θα παρουσιαστούν είναι οι Temperature-Humidity, PWM Speaker και η PwmOut οι οποίες είναι προ-εγκατεστημένες στον online simulator. Θα περιηγηθούμε στον βασικό κώδικα των ασκήσεων να δούμε την προκαθορισμένη λειτουργία και θα επέμβουμε με δίκες μας εντολές για να δούμε την διαφορετική συμπεριφορά στην online πλακέτα

Temperature-Humidity



Ας κατανοήσουμε λίγο το κώδικα

Αντικαταστήστε την «led = temp > 25.0f;» με if-then-else δομή

Προσθέστε κώδικα να ενεργοποιεί το αντιδιαμετρικό led όταν η υγρασία πέσει κάτι από 15%

Επεκτείνατε των counter με μετράει μέχρι το 32 μέσω εξωτερικού led

Υποθέσατε συνθήκες υψηλού κινδύνου φωτιάς τις

Θερμοκρασία > 30, Υγρασία < 15

Ας γράψουμε κώδικα όταν ισχύουν οι συνθήκες αυτές

Να αναβοσβήνουν τα ακριανά led

Να αναβοσβήνει μήνυμα στην LCD

```
#include "mbed.h"
#include "C12832.h"
#include "Sht31.h"

C12832 lcd(SPI_MOSI, SPI_SCK, SPI_MISO, p8, p11);
Sht31 sht31(I2C_SDA, I2C_SCL);
/*Με την εντολή αυτή ορίζουμε τους ακροδέκτες στους οποίους συνδέεται το ολοκληρωμένο sht31 για να υλοποιηθεί η επικοινωνία I2C. SCL: Ακροδέκτης για τον συγχρονισμού του ρολογιού και SDA: Ακροδέκτης μέσω του οποίου μεταδίδονται τα πραγματικά δεδομένα των sensors. */
DigitalOut led(LED1);

int main() {
   printf("Set the temperature above 25 degrees to trigger the warning LED\n");

   while (1) {
      lcd.cls();

      float temp = sht31.readTemperature();
```

```
//Μέσω της αντίστοιχη βιβλιοθήκης έχουμε πρόσβαση στα δεδομένα θερμοκρασίας.
      float humidity = sht31.readHumidity();
//Μέσω της αντίστοιχη βιβλιοθήκης έχουμε πρόσβαση στα δεδομένα υγρασίας.
      lcd.locate(3, 3);
      lcd.printf("Temperature: %.2f C", temp);
      lcd.locate(3, 13);
      lcd.printf("Humidity: %.2f %%", humidity);

      // turn on LED if the temperature is above 25 degrees
      led = temp > 25.0f;
  wait(0.5f);
    }
}
```

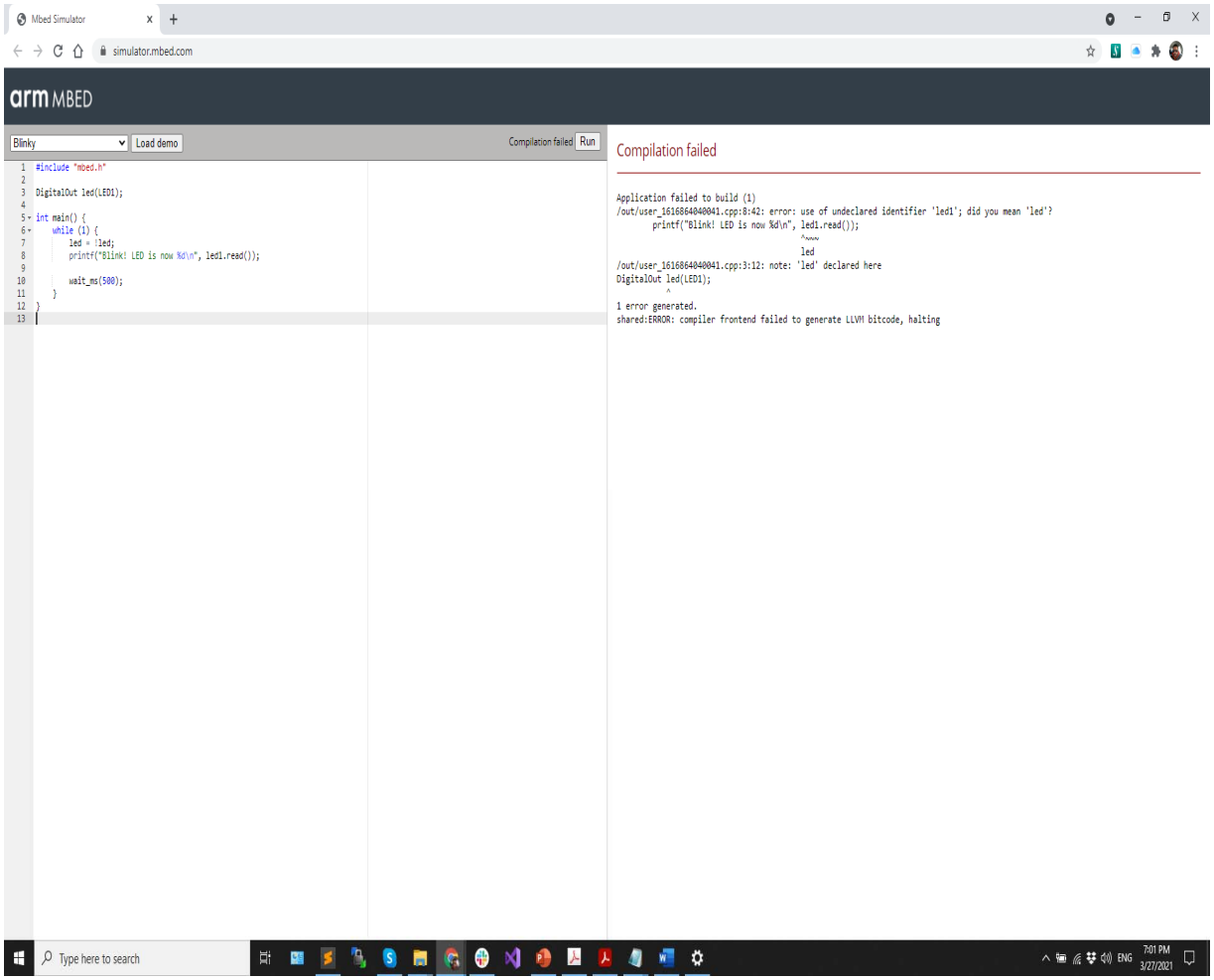
load demo

Με την επιλογή αυτή φορτώνουμε το αντίστοιχο παράδειγμα που θέλουμε να τρέξουμε στην προσωρινή μνήμη.

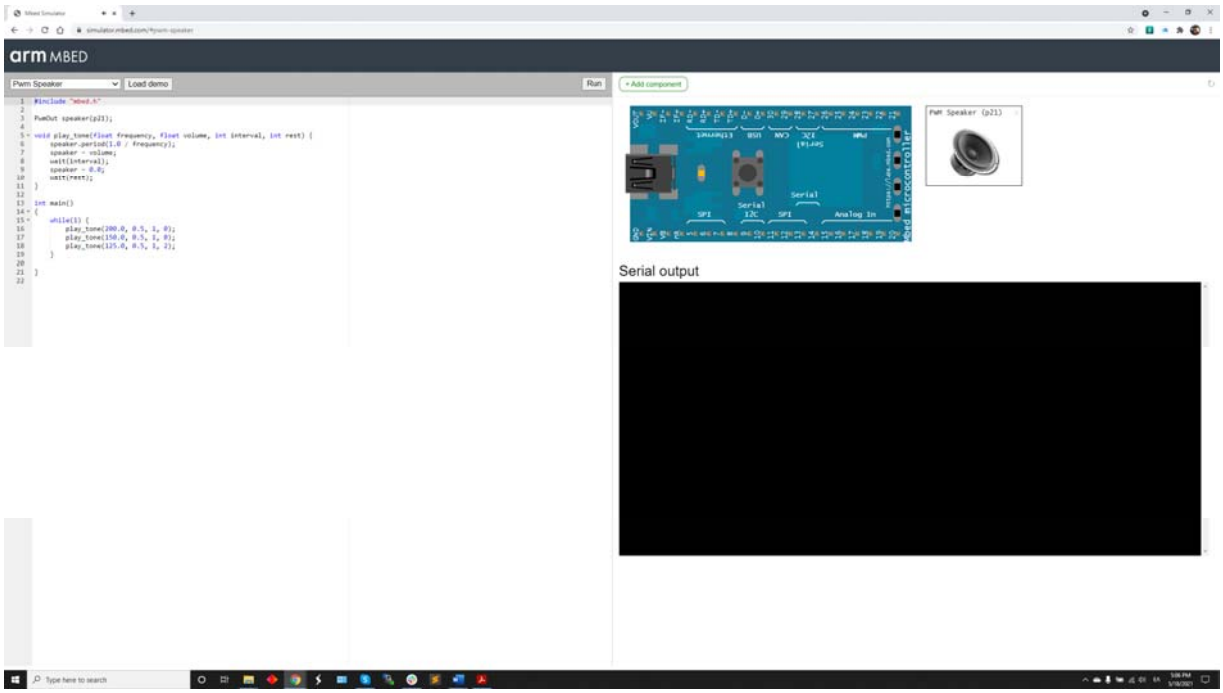
Run

Με την εντολή αυτή γίνεται compile και τρέχει ο κώδικας του παραδείγματος που έχουμε επιλέξει και φορτώσει.

Σε περίπτωσή που ο κώδικας που εισάγουμε η τροποποιήσουμε έχει κάποιο λάθος θα μας εμφανιστεί ένα μήνυμα όπως το παρακάτω.


Mbed Simulator
simulator.mbed.com
arm MBED
Blinky
Load demo
Compilation failed
Run
#include "mbed.h"
DigitalOut led(LED1);
int main() {
while (1) {
led = !led;
printf("Blink! LED is now %d\n", led1.read());
wait_ms(500);
}
}
Compilation failed
Application failed to build (1)
/out/user_1616864040041.cpp:8:42: error: use of undeclared identifier 'led1'; did you mean 'led'?
printf("Blink! LED is now %d\n", led1.read());
led
/out/user_1616864040041.cpp:3:12: note: 'led' declared here
DigitalOut led(LED1);
1 error generated.
shared:ERROR: compiler frontend failed to generate LLVM bitcode, halting
Type here to search
ENG
7:01 PM
3/27/2021

PWM Speaker



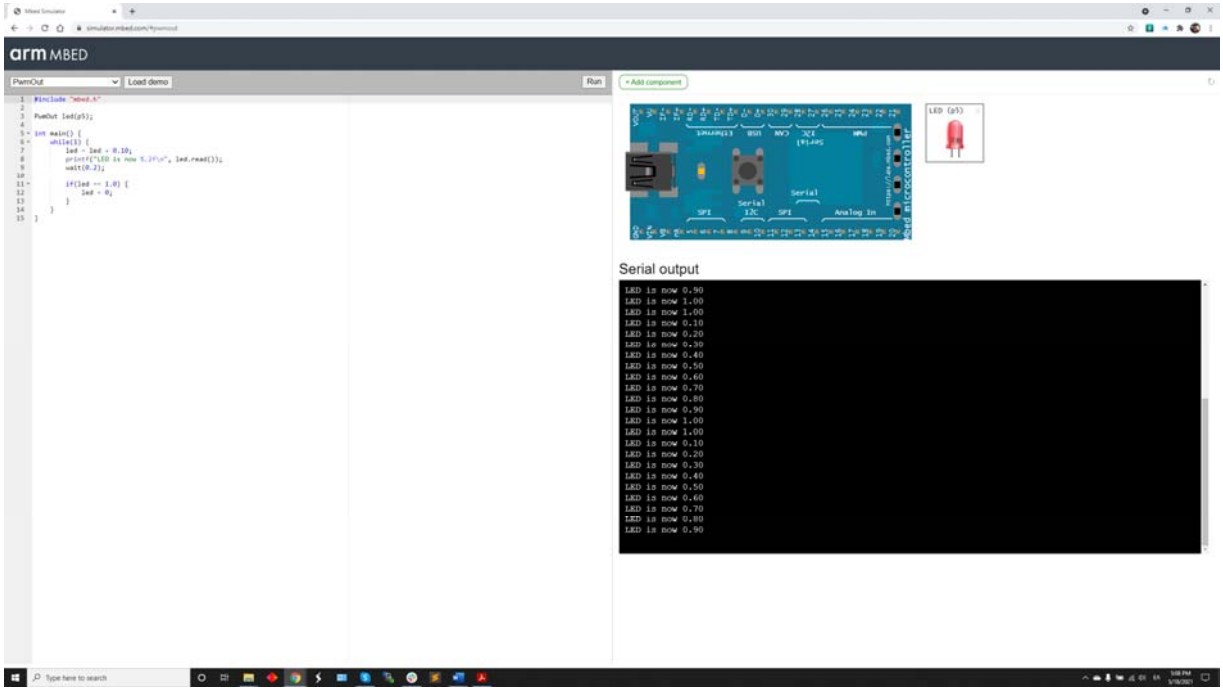
Ας βάλουμε και λίγο ήχο…. Κατανόηση κώδικα

Μπορούμε να συνδυάσουμε το παράδειγμα αυτό με την προηγούμενο κώδικα και ήχος να ενεργοποιείται σε συνθήκες φωτιάς;

```
#include "mbed.h"

PwmOut speaker(p21);

void play_tone(float frequency, float volume, int interval, int rest) {
   speaker.period(1.0 / frequency); //Καθορίζει την συχνότητα του ήχου που θα παραχθεί
   speaker = volume; //Καθορίζει την ένταση του ήχου που θα παραχθεί
   wait(interval); //Καθορίζει την χρονική διάρκεια που θα ακούγεται ο ήχος
   speaker = 0.0; //Ορίζει την ένταση ώστε να μην ακούγεται ήχος
   wait(rest); //Καθορίζει την χρονική διάρκεια που ΔΕΝ θα ακούγεται ήχος
}

int main()
```

```
{
   while(1) {
      play_tone(200.0, 0.5, 1, 0);
      play_tone(150.0, 0.5, 1, 0);
      play_tone(125.0, 0.5, 1, 2);
//Αλλαξτε τις τιμές  frequency και rest και παρατηρήστε τις αλλαγές στην συμπεριφορά του συστήματος
//Μπορείτε να φτιάξετε μια δική σας «play_tone» που να έχει 3 φάσεις λειτουργίας αντί για 2 που έχει αυτή σας δίνεται?
   }
}
```

PwmOut



Κατανόηση κώδικα και λειτουργίας.

Αλλαξτε τιμές σε led και wait και παρατηρηστε τις αλλαγες στην συμπεριφορα στο εξωτερικό λεντάκι π.χ.

```
led = led + 0.05;

wait(0.01);
```

ΑΣΚΗΣΕΙΣ ΓΙΑ ΠΑΡΑΔΟΣΗ

Φτιάξτε μια συνδυαστική άσκηση η οποία θα ικανοποιεί τις εξής απαιτήσεις

1) Σε περίπτωση όπου η θερμοκρασία είναι >35^{o}C ΚΑΙ η υγρασία είναι <15% το σύστημα να θεωρεί ότι υπάρχει φωτιά και να εκπέμπει ενοχλητικό ήχο , να αναβοσβήνει το εξωτερικό κόκκινο LED και η οθόνη να έχει κυλιόμενο μήνυμα “FIRE ALARM!!!”

2) Σε περίπτωση όπου η θερμοκρασία είναι <15 ^{o}C ΚΑΙ η υγρασία είναι <15% το σύστημα να θεωρεί ότι υπάρχει υπερβολική υγρασία και να εκπέμπει ενοχλητικό ήχο (διαφορετικό του ερωτήματος 1) , να αναβοσβήνει το εξωτερικό μπλέ LED και η οθόνη να έχει κυλιόμενο μήνυμα “FIRE ALARM!!!”

3) Σε κάθε άλλη περίπτωση η οθόνη να προβάλει σταθερό μήνυμα “Normal Conditions”

Η αναφορά εκτός από τον κώδικα και σχετικά screen shots πρέπει να έχει και ένα σχετικό flowchart του κώδικα.

https://simulator.mbed.com/