```
#include <Keypad.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include "max6675.h"


//BUG: MEASUREMENTS ARE OFF, TURNS ON NEEDLESSLY


enum states {
  welcome, //Just turned on
  select_temp, //set up for pre-heat
  heating, //is heating the toast
  select_time, //timer for toast
  cooking, //toast is cooking
  done //Done! Enjoy the toast
};

states curr_state = welcome;
bool has_print = false;
double selected_temperature = 0;
double selected_time = 0;
String buffer = "";
char forward = '#';
char backward = '*';
bool heating_start = false;
double cooking_time_reference = 0;
bool just_finished = false;
String enter_time_seconds = "Enter time";
double measurement_var = 0;
unsigned long reading_millis = 0;

unsigned long prev_millis = 0;
const long interval = 1000;
unsigned long target_millis = 0;

LiquidCrystal_I2C lcd(0x27, 16, 2); // set the LCD address to 0x27 for a 16 chars and 2 line
display
```

```
const byte ROWS = 4; //four rows
const byte COLS = 4; //three columns
char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};

byte rowPins[ROWS] = {22, 24, 26, 28}; //connect to the row pinouts of the keypad
byte colPins[COLS] = {30,32,34,36}; //connect to the column pinouts of the keypad

Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );
int ktcSO = 3;
int ktcCS = 13;//doesnt not need squigaily
int ktcCLK = 10;
MAX6675 ktc(ktcCLK, ktcCS, ktcSO);

void setup(){
  lcd.begin(16, 2); //lcd begin
  lcd.init();  //initialize the lcd
  lcd.backlight();
  Serial.begin(9600);
  delay(500);
  pinMode(11, OUTPUT);
  digitalWrite(11, LOW);
  buffer = "";
}

void loop(){
  // basic readout test

   //Serial.print("Deg C = ");
   //Serial.print(ktc.readCelsius());
   if(curr_state == welcome){
    //digitalWrite(11, LOW);
    welcome_subroutine();
   } else if(curr_state == select_temp){
    //digitalWrite(11, LOW);
```

```
    select_temp_subroutine();
   } else if(curr_state == heating){
     heating_subroutine();
   } else if(curr_state == select_time){
    select_time_subroutine();
   } else if(curr_state == cooking){
    cooking_subroutine();
   } else if(curr_state == done){
    done_subroutine();
   }


//   Serial.print("\t Deg F = ");
//   Serial.println(ktc.readFahrenheit());
//   printToScreen(String(ktc.readFahrenheit(), 10), "Deg Fahrenheit");
//   delay(500);


}

void welcome_subroutine(){
  if(!has_print){
    print_to_screen("Welcome! Press", "# to continue!");
    has_print = true;
  }
  char key = keypad.getKey();
  if(key == forward){
    curr_state = select_temp;
    has_print = false;
  }
}

void select_temp_subroutine(){
  if(!has_print){
    print_to_screen("Select pre-heat", "Temp and press #");
    has_print = true;
    buffer = "";
  }
  char key = keypad.getKey();
```

```
  if(key == forward && buffer == ""){
    return;
  }
  if(isDigit(key)){
    has_print = false;
  }
  if(key == backward){
    curr_state = welcome;
    buffer = "";
    has_print = false;
  } else if(key == forward){

    selected_temperature = buffer.toDouble();
    curr_state = heating;
    has_print = false;
  } else {
      buffer += String(key);
      if(!has_print){
        print_to_screen("Temperature:", buffer);
        has_print = true;
      }
  }
}

void heating_subroutine(){
  if(!(selected_temperature >= 100 && selected_temperature <= 500) && (!has_print)){
    print_to_screen("ERROR:", "Bad Temp");
    delay(2000);
    curr_state = select_temp;
    has_print = false;
    buffer = "";
    digitalWrite(11, LOW);
    return;
  }
  if(!has_print){
    print_to_screen("Heating the", "oven...");
    has_print = true;
    prev_millis = millis();
    target_millis = prev_millis + 500;
```

```
  }
  if((double) selected_temperature > measurement_var){
    digitalWrite(11, HIGH);
    if(millis() > target_millis){
      measurement_var = ktc.readFahrenheit();
      String curr_temp_measurement = String(measurement_var);
      Serial.println(measurement_var);
      print_to_screen("Current Temp:", curr_temp_measurement);
      target_millis += 2000;
    }

    char key = keypad.getKey();
    if(key == backward){
      digitalWrite(11, LOW);
      curr_state = select_temp;
      has_print = false;
      buffer = "";
    }

  } else {
    //progress to next phase...
    print_to_screen("", "");
    curr_state = select_time;
    has_print = false;
    heating_start = true;
    digitalWrite(11, LOW);
    buffer = enter_time_seconds;
  }

}

void maintain_temperature(){
  double measure = 0;
  if(millis() > reading_millis){
    measure = ktc.readFahrenheit();
    reading_millis += 2000;
    Serial.println("Measurement: " + String(measure));
    Serial.println("Target: " + String(selected_temperature));
    bool condition = ((int) selected_temperature > (int) measure);
```

```
   Serial.println("Should it be on? " + String(condition));
   //if what I want is hotter than what I have...
   if(condition){
    digitalWrite(11, HIGH);
    //Serial.println("Oven is ON");
   } else {
    digitalWrite(11, LOW);
    //Serial.println("Oven is OFF");
   }
  }

}

void select_time_subroutine(){

  maintain_temperature();

  if(!has_print){
   print_to_screen("Heating done", buffer);
   if(buffer == enter_time_seconds){
    buffer = "";
   }
   has_print = true;
  }
  char key = keypad.getKey();
  if(isDigit(key)){
   has_print = false;
   if(buffer.length() == 5){
    return;
   }
   if(buffer.length() == 2){
    char temp = buffer.charAt(buffer.length() - 1);
    buffer = buffer.substring(0, 1);
    buffer += ":";
    buffer += String(temp);
   } else if(buffer.length() == 4){
    char temp1 = buffer.charAt(buffer.length() - 2);
    char temp2 = buffer.charAt(buffer.length() - 1);
    buffer = buffer.substring(0, 1);
```

```
    buffer += temp1;
    buffer += ":";
    buffer += String(temp2);
   }
   buffer += String(key);
   //if a number, need to store the number and store cook time

  } else if(key == backward){
   has_print = false;
   curr_state = select_temp;
  } else if (key == forward){

   bool minutesFlag = false;
   String minutes = "";
   String seconds = "";
   if(buffer.length() > 2){
    for(int i = 0; i < buffer.length(); i++){
     char currCharChar = buffer.charAt(i);
     String currChar = String(currCharChar);
     if(currChar == ":"){
      minutesFlag = true;
      continue;
     }
     if(!minutesFlag){
      minutes += currChar;
     } else {
      seconds += currChar;
     }
    }
   } else {
    minutes = "0";
    seconds = buffer;
   }

   bool bad_time_flag = false;
   double min_double = minutes.toDouble();
   if(min_double > 59){
    bad_time_flag = true;
   }
```

```
    double minutes_ref = min_double;
    min_double *= 60;
    double sec_double = seconds.toDouble();
    if(sec_double > 59 && minutes_ref == 59){
      bad_time_flag = true;
    }
    if(sec_double > 3599){
      bad_time_flag = true;
    }
    if(bad_time_flag){
      print_to_screen("ERROR:", "Time too high");
      delay(2000);
      has_print = false;
      buffer = enter_time_seconds;
      return;
    }
    sec_double += min_double;
    cooking_time_reference = sec_double;
    buffer = "";
    has_print = false;
    curr_state = cooking;
    Serial.println(curr_state);
    Serial.println("Entering Cooking Mode");
  } else {
      if(!has_print){
        print_to_screen("Time:", buffer);
        has_print = true;
      }
  }
}

void cooking_subroutine(){
  maintain_temperature();
  if(!has_print){
    int seconds = (int) (((int) cooking_time_reference) % 60);
    int minutes = (int) (((int) cooking_time_reference) / 60);
    String time_to_print = String(minutes) + ":";
    if(seconds < 10){
      time_to_print += "0";
```

```
    }
    time_to_print += String(seconds);
    print_to_screen("Time Left:", time_to_print);
    prev_millis = millis();
    target_millis = prev_millis + interval;
    has_print = true;
   }
   prev_millis = millis();
   if(prev_millis > target_millis){
    has_print = false;
    cooking_time_reference--;
   }
   if(cooking_time_reference <= 0){
    print_to_screen("Done!", "Enjoy!");
    curr_state = done;
    just_finished = true;
   }
   char key = keypad.getKey();
   if(key == backward){
    has_print = false;
    curr_state = select_time;
    buffer = enter_time_seconds;
   }

}

void done_subroutine(){
  maintain_temperature();
  if(just_finished){
    prev_millis = millis();
    just_finished = false;
    target_millis = prev_millis + 30000;
    //Serial.println("Target = " + target_millis);
  }
  //Serial.println(millis());
  char key = keypad.getKey();
  //Serial.println(target_millis - millis());
  if(millis() > target_millis){
    curr_state = welcome;
```

```
   print_to_screen("", "");
   has_print = false;
   digitalWrite(11, LOW);
  } else if(key == backward){
   has_print = false;
   curr_state = select_time;
   buffer = enter_time_seconds;
  }


}

//void select_temp_subroutine(){
//  if(!has_print){
//    print_to_screen("Select pre-heat", "Temp and press #");
//    has_print = true;
//  }
//  char key = keypad.getKey();
//  if(isDigit(key)){
//    has_print = false;
//  }
//  if(key == backward){
//    curr_state = welcome;
//    buffer = "";
//    has_print = false;
//  } else if(key == forward){
//
//    selected_temperature = buffer.toDouble();
//    curr_state = heating;
//    has_print = false;
//  } else {
//      buffer += String(key);
//      if(!has_print){
//        print_to_screen("Temperature:", buffer);
//        has_print = true;
//      }
//  }
//}
```

```
void print_to_screen(String str1, String str2){
  lcd.clear();
  lcd.setCursor(0, 0); // set the cursor to column 3, line 0
  lcd.print(str1);  // Print a message to the LCD
  lcd.setCursor(0, 1); // set the cursor to column 2, line 1
  lcd.print(str2);  // Print a message to the LCD.
}



//   pinMode(11, OUTPUT);
//  char key = keypad.getKey();
//    // just print the pressed key
//   if (key){
//    Serial.println(key);
//  }
//
//  // this checkes if 4 is pressed, then do something. Here  we print the text but you can control
something.
//  if (key =='4'){
//    Serial.println("Key 4 is pressed");
//       digitalWrite(11, HIGH);
//  }
//    if (key =='5'){
//    Serial.println("Key 5 is pressed");
//       digitalWrite(11, LOW );
//  }
```