DHT11 (Humidity and Temperature Sensor) to seven-segment display, Lab 3

Embedded System Design - CS3813301
Group 6
Tobias Vera - tobiasvera2011@gmail.com (F10915126),
Gerardo Fisch - gerarfisch@gmail.com (F10915108)
November 9th, 2021

I. INTRODUCTION

The purpose of this lab is to introduce the students to a one-wire protocol device used with embedded systems and for the student to learn about one-wire protocol. The students need to design a one wire (DHT11) to 4-digit seven-segment display.

II. PROCEDURE

- Connect the DHT11 to Arduino using a 10K pull-up (the DHT11 uses open-drain single-wire bi-directional design).
- Read the DHT11 datasheet in order to determine how to activate the DHT11 and decode the 40-bit response. The number data returned from the DHT11 follow a Q8.8 number format.
- Use assembly to decode the DHT11 data and be sure to verify the checksum.
- Run the program on the Arduino board.

III. EXPERIMENTAL DATA

A. Schematic of the Circuit

The schematic of the circuit can be seen on the Figure 1. From the schematic it can be seen that the pins used for this circuit are the pins from 0 to 11 for the 4 digit seven segment display and the pin 13 for the DHT11. From the schematic of the Arduino UNO[1], it can be seen that the only output ports used on the original circuit are the PORT B and PORT D, since the pins from 0 to 7 correspond to the PORT D and the pins from 8 to 11 correspond to the PORT B and the only input/output port used is the pin 13 which is from the PORTB. In addition, the GND and 5V pins were used to power the DHT11.

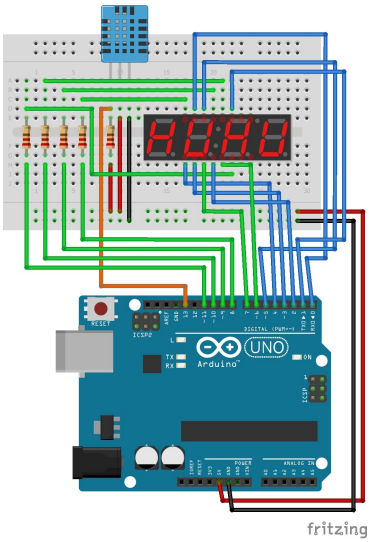

Fig. 1. Schematic of the circuit

B. Photo of the wired-up circuit

Based on the schematic of the circuit, the same circuit was reproduced. The circuit can be seen on Figure 2.

C. Results

The Figures 4, 5, 6 and 7 show the final result.

IV. QUESTIONS

1) What are the pros and cons of one-wire protocol?
Pros:
- Due to use of less wires, the interface is cheaper.
- It is easy to implement the interface.
Cons:
- It is implemented both in the hardware as well as software. The synchronization of data at the receiver has to be taken care in software which is a complex task.
- It supports slower speed of communication.

2) What happens if you sample the DHT11 too quickly?
If the DHT11 if sampled too quickly, the program would take an erroneous result, since not all the bits would be sampled. The result would probably consist of only the first bits sampled multiple times.

3) What rate did you sample the one-wire signal at? What would happen if sampled too slowly?
The signal is sampled at 20us. If the signal is sampled too slowly, then the program would not be able to get any result, because when the sensor receives a low signal but it takes very long to receive a high signal, it stops trying to send any response.

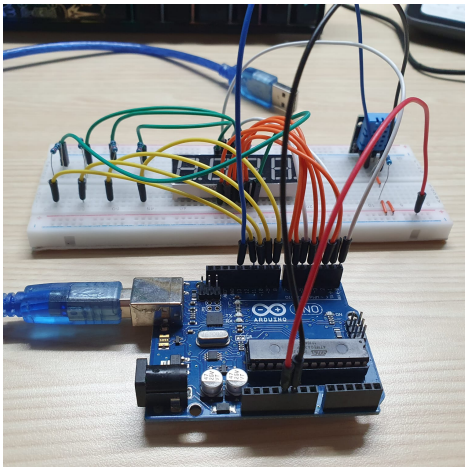

Fig. 2. Wired-up circuit

4) What rate did you sample the temperature sensor at? What would happen if sampled too slowly?
The temperature is sampled at 30us. If the sensor is sampled too slowly, the program would take an erroneous result, since not all the bits would be sampled. The result would probably miss some or most of the bits.

V. DISCUSSION

Two versions of the experiment where tested, the one presented here and a program using the Adafruit library to work with the DHT11 sensor in order to verify if the results were correct. Both of them give the same results, which means that the first program works correctly. The displaying of the values and messages also works correctly and no problems were encountered.

VI. CONCLUSION

The experiment went as expected. The values of the temperature and humidity were detected and displayed correctly. This exercise was also useful for a better learning of the one-wire protocol functionality and for improving the assembly coding abilities.

REFERENCES

[1] https://www.arduino.cc/en/uploads/Main/arduino-uno-schematic.pdf

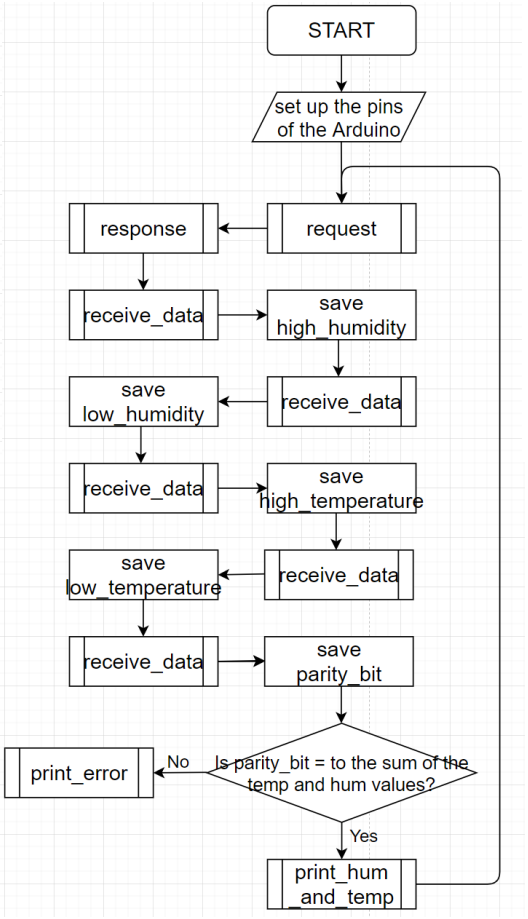

Fig. 3. Flow Diagram of the Program

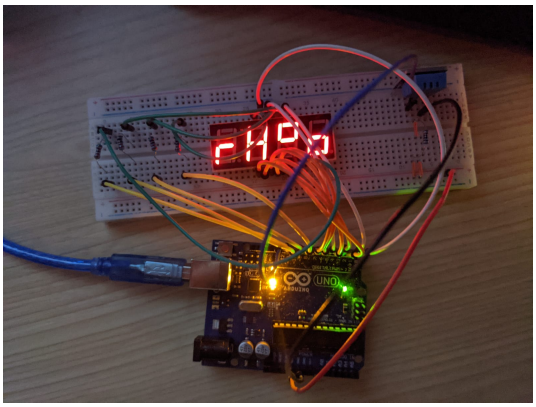

Fig. 4. Example of the circuit showing rH°o.

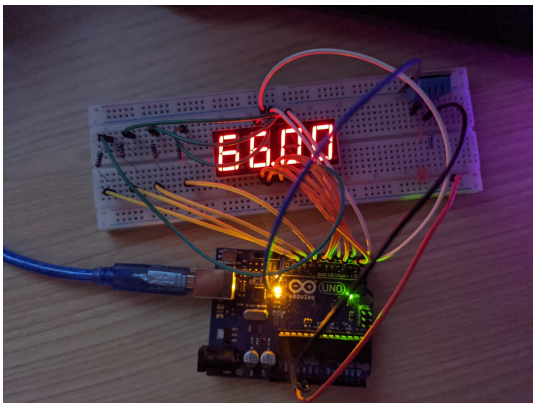

Fig. 5. Example of the circuit showing the humidity.

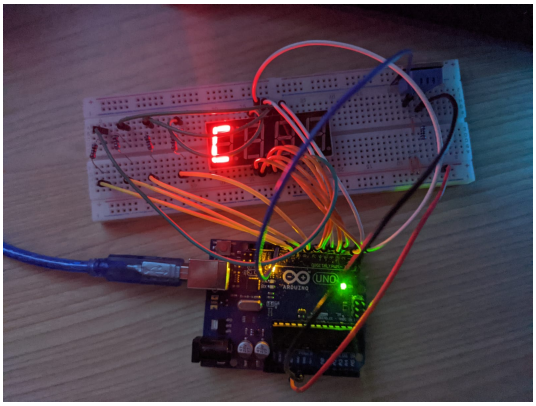

Fig. 6. Example of the circuit showing C.

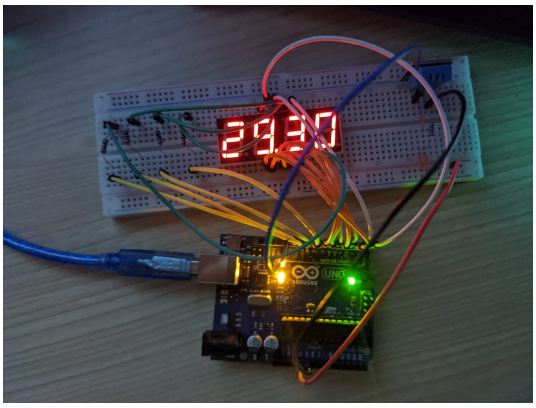

Fig. 7. Example of the circuit showing the temperature.

```
.equ number_1 = 0b00000110
.equ number_2 = 0b01011011
.equ number_3 = 0b01001111
.equ number_4 = 0b01100110
.equ number_5 = 0b01101101
.equ number_6 = 0b01111101
.equ number_7 = 0b00000111
.equ number_8 = 0b01111111
.equ number_9 = 0b01101111
.equ number_0 = 0b00111111
.equ letter_r = 0b01010000
.equ letter_H = 0b01110110
.equ symbol_degree = 0b01100011
.equ letter_o = 0b01011100
.equ letter_C = 0b00111001
.equ letter_E = 0b01111001
.equ off_segments = 0b00000000

.equ digit_4 = 0b00001110
.equ digit_3 = 0b00001101
.equ digit_2 = 0b00001011
.equ digit_1 = 0b00000111

.equ dht11_port = 5 ;(13 - 8 = 5)
.equ point_port = 7

.equ dht11_io = 0b00100000
.equ dec205_binary = 0b11001101

.def high_humidity = r20
.def low_humidity = r21
.def high_temperature = r22
.def low_temperature = r23
.def parity_bit = r24
.def dec_high = r25
.def unit_high = r26
.def dec_low = r27
.def unit_low= r28
.def param1 = r29
.def param2 = r30

; Configure Direction PORT B
ldi r16, 0b00001111 ; digital pins from 8 to 13 of the Arduino
out DDRB, r16
; Configure Direction PORT D
ldi r16, 0b11111111 ; digital pins from 2 to 7 of the Arduino
out DDRD, r16

loop:
  call request             ; a request to the dht11 is made
  call response            ; the response of the dht11 is received
  call receive_data        ; data is received 5 times and saved in 5 diferent registers
  mov high_humidity, r18
  call receive_data
  mov low_humidity, r18
  call receive_data
  mov high_temperature, r18
  call receive_data
  mov low_temperature, r18
  call receive_data
  mov parity_bit, r18

  mov r16, high_humidity
  add r16, low_humidity
  add r16, high_temperature
  add r16, low_temperature
  cp r16, parity_bit       ; if the sum of the first 4 data received is equal to the parity, the data is
                           ; received is correctly
  brne print_error         ; if wrong data is received, the display will show Erro(r)
  call print_hum_and_temp

  rjmp loop

print_error:
```

```
  jmp print_error_long_jump    ; a long jump is required to be able to print Erro(r)

print_hum_and_temp:
  call print_rHo

  mov param1, high_humidity
  mov param2, low_humidity
  call print_value        ; used to print the current humidity

  call print_C

  mov param1, high_temperature
  mov param2, low_temperature
  call print_value        ; used to print the current temperature
  ret

print_value:            ; This function takes 2 parameters (param1 & param2)
  call convert_binary_to_dec

  ldi  r18, 65
  ldi  r19, 240
  ldi  r31, 4
  loop_print_temp_value:
    ldi r16, digit_4
    out PORTB, r16          ; the 4th digit is set
    mov param1, dec_high
    call print_digit

    call reset_display

    ldi r16, digit_3
    out PORTB, r16          ; the 3rd digit is set
    mov param1, unit_high
    call print_digit
    sbi PORTD, point_port   ; used to print the point on the 3rd digit

    call reset_display

    ldi r16, digit_2
    out PORTB, r16          ; the 2nd digit is set
    mov param1, dec_low
    call print_digit

    call reset_display

    ldi r16, digit_1
    out PORTB, r16         ; the 1st digit is set
    mov param1, unit_low
    call print_digit

    call reset_display

    dec  r18
    brne loop_print_temp_value
    dec  r19
    brne loop_print_temp_value
    dec  r31
    brne loop_print_temp_value
  ret

; This function is used to convert the data received from binary to decimal
convert_binary_to_dec:       ; This function takes 2 parameters (param1 & param2) and
  ldi dec_high, 0            ; it returns 4 value (dec_high, unit_high, unit_low, dec_low)
  div_by_10:
    inc dec_high
    subi param1, 10
    brcc div_by_10           ; branch if C is zero

  dec dec_high               ; once too many
  subi param1, 246           ; param1 += 1, add back to get the remainder
  mov unit_high, param1

  mov dec_low, param2
  ldi unit_low, 0
  ret
```

```
reset_display:          ; turn off all the display
  ldi r16, 0x0F
  out PORTB, r16
  ldi r17, 0x00
  out PORTD, r17
  ret

request:
  sbi DDRB, dht11_port     ; set the dht11 as an output port to send the request
  cbi PORTB, dht11_port    ; set the dht11 as low

  delay_20ms:
    ldi r29, 147
    ldi r30, 160
    ldi r31, 2
    dloop_20ms:
      dec r29
      brne dloop_20ms
      dec r30
      brne dloop_20ms
      dec r31
      brne dloop_20ms

  sbi PORTB, dht11_port    ; set the dht11 as high
  ret

response:
  cbi DDRB, dht11_port     ; set the dht11 as an input port to wait for a response

  loop_response_1:         ; while (PIN of the dht11 == 1)
    sbic PINB, dht11_port
    rjmp loop_response_1
  loop_response_2:         ; while (PIN of the dht11 == 0)
    sbis PINB, dht11_port
    rjmp loop_response_2
  loop_response_3:         ; while (PIN of the dht11 == 1)
    sbic PINB, dht11_port
    rjmp loop_response_3
  ret

receive_data:
  ldi r16, 0x00
  ldi r18, 0x00
  rec_data_loop1:          ; loop used to receive the 8 bits
    rec_data_loop1_1:      ; while (PIN of the dht11 == 0)
      sbis PINB, dht11_port
      rjmp rec_data_loop1_1

    call delay_50us

    in r17, PINB
    andi r17, dht11_io
    cpi r17, 0x00
    brne r17_is_not_0
      lsl r18              ; if (PINB == 0)
      rjmp end_if
    r17_is_not_0:
      lsl r18              ; if (PINB == 1)
      ori r18, 0x01
    end_if:

    rec_data_loop1_2:      ; while (PIN of the dht11 == 1)
      sbic PINB, dht11_port
      rjmp rec_data_loop1_2

    inc r16
    cpi r16, 0x08
    brne rec_data_loop1
  ret

delay_50us:
  ldi  r29, 2
    ldi  r30, 9
  L1: dec  r30
```

```
    brne L1
    dec  r29
    brne L1
    ret

print_C:
  ldi  r29, 65
  ldi  r30, 240
  ldi  r31, 52

  ldi r16, digit_4       ; the 4th digit is set
  out PORTB, r16
  ldi r17, letter_C      ; C is printed on the 4th digit
  out PORTD, r17

  loop_print_C:
    dec  r31
    brne loop_print_C
    dec  r30
    brne loop_print_C
    dec  r29
    brne loop_print_C
  ret

print_rHo:
  ldi  r18, 65
  ldi  r19, 240
  ldi  r31, 4
  loop_print_rho:
    call reset_display

    ldi r16, digit_4         ; the 4th digit is set
    out PORTB, r16
    ldi r17, letter_r        ; r is printed on the 3rd digit
    out PORTD, r17

    call reset_display

    ldi r16, digit_3         ; the 3rd digit is set
    out PORTB, r16
    ldi r17, letter_H        ; H is printed on the 3rd digit
    out PORTD, r17

    call reset_display

    ldi r16, digit_2         ; the 2st digit is set
    out PORTB, r16
    ldi r17, symbol_degree   ; degree symbol is printed on the 3rd digit
    out PORTD, r17

    call reset_display

    ldi r16, digit_1         ; the 1st digit is set
    out PORTB, r16
    ldi r17, letter_o        ; o is printed on the 3rd digit
    out PORTD, r17

    dec  r18
    brne loop_print_rho
    dec  r19
    brne loop_print_rho
    dec  r31
    brne loop_print_rho
  ret

print_digit:          ; This function takes 1 parameter (param1)
  cpi param1, 0x00    ; compare param1 with values between 0-9 to decide
  breq print_0        ; which number to show
  cpi param1, 0x01
  breq print_1
  cpi param1, 0x02
  breq print_2
  cpi param1, 0x03
  breq print_3
  cpi param1, 0x04
```

```
  breq print_4
  cpi param1, 0x05
  breq print_5
  cpi param1, 0x06
  breq print_6
  cpi param1, 0x07
  breq print_7
  cpi param1, 0x08
  breq print_8
  cpi param1, 0x09
  breq print_9
  print_0:
    ldi r16, number_0
    out PORTD, r16
    rjmp switch_end
  print_1:
    ldi r16, number_1
    out PORTD, r16
    rjmp switch_end
  print_2:
    ldi r16, number_2
    out PORTD, r16
    rjmp switch_end
  print_3:
    ldi r16, number_3
    out PORTD, r16
    rjmp switch_end
  print_4:
    ldi r16, number_4
    out PORTD, r16
    rjmp switch_end
  print_5:
    ldi r16, number_5
    out PORTD, r16
    rjmp switch_end
  print_6:
    ldi r16, number_6
    out PORTD, r16
    rjmp switch_end
  print_7:
    ldi r16, number_7
    out PORTD, r16
    rjmp switch_end
  print_8:
    ldi r16, number_8
    out PORTD, r16
    rjmp switch_end
  print_9:
    ldi r16, number_9
    out PORTD, r16
  switch_end:
    ret

print_error_long_jump:
  call reset_display

  ldi r16, digit_4       ; the 4th digit is set
  out PORTB, r16
  ldi r17, letter_E      ; E is printed on the 4th digit
  out PORTD, r17

  call reset_display

  ldi r16, digit_3       ; the 3rd digit is set
  out PORTB, r16
  ldi r17, letter_r      ; r is printed on the 3rd digit
  out PORTD, r17

  call reset_display

  ldi r16, digit_2       ; the 2nd digit is set
  out PORTB, r16
  ldi r17, letter_r      ; r is printed on the 3rd digit
  out PORTD, r17
```

```
call reset_display

ldi r16, digit_1        ; the 1st digit is set
out PORTB, r16
ldi r17, letter_o       ; o is printed on the 3rd digit
out PORTD, r17

rjmp print_error_long_jump
```