Universitat de Lleida

Data log and Screen representation (Sprint 4)

MINF UDL 20-21

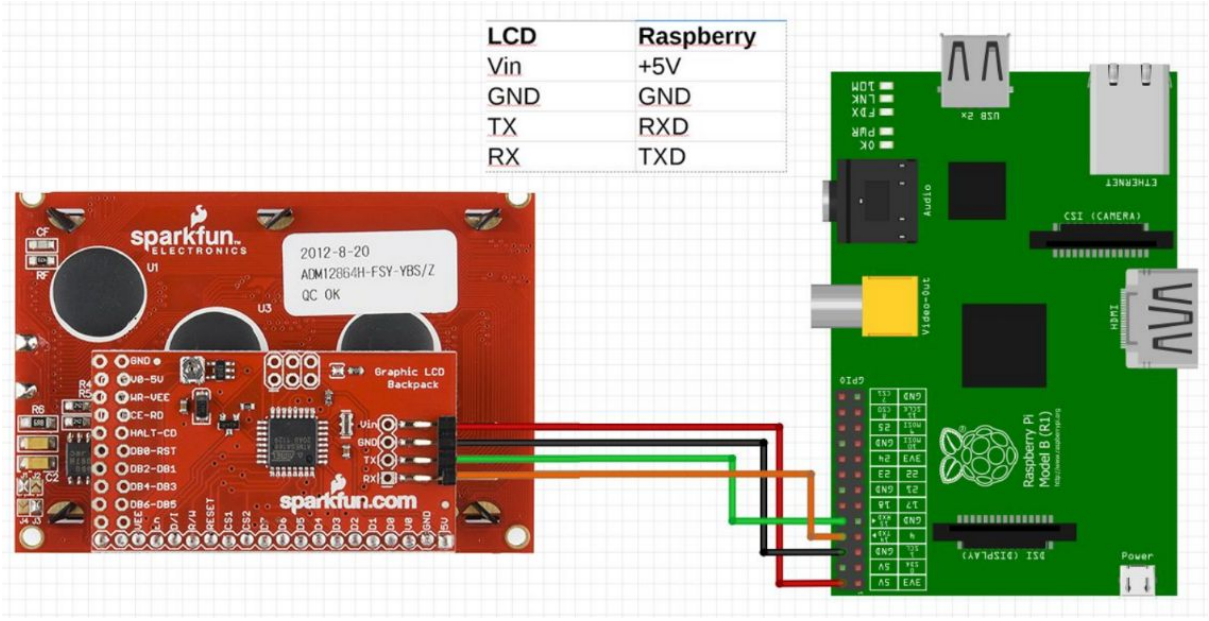
Ubiquitous and embedded systems

Team 1

1. Objective

a. The objective is to make a graphical representation of the obtained temperature and humidity on the LCD Screen, being a relation of value x time (24h), the code will be present in the chibiOS.

2. Wiring



Note: in this case the RXD -> TX cable is not necessary, onyl the TXD <> RX since the LCD is not sending anything to the Raspberry.

3. Code (ChibiOS)

a. Thread

```
static WORKING_AREA(waThread_LCD, 128);
static msg_t Thread_LCD(void *p)
{
  (void)p;
  chRegSetThreadName("SerialPrint");

  //drawStructure();
  // aux variable to control which screen to show
  int screen = 0;

  while (TRUE)
  {
    //  lcdPrintf(32, 32, "Iteration: %u", iteration);
    chBSemWait(&smph);

    drawStructure();

    if (aux_counter == 0)
    {
      stackLineTemp[0][0] = 18;
      stackLineTemp[0][1] = 14 + temperature;
      stackLineTemp[0][2] = 18 + 1;
      stackLineTemp[0][3] = 14 + temperature;

      stackLineHum[0][0] = 18;
      stackLineHum[0][1] = 14 + roundNo(humidity / 2);
      stackLineHum[0][2] = 18 + 1;
      stackLineHum[0][3] = 14 + roundNo(humidity / 2);
    }
    else
    {
      stackHandler(0);

      if (screen == 0)
      {
        drawGraphLineTemp(temperature);
        if (aux_counter % 5 == 0)
        {
          screen = 1;
          clearScreen();
        }
      }
      else if (screen == 1)
      {
        drawGraphLineHum(humidity);
```

```
        if (aux_counter % 10 == 0)
        {
          screen = 0;
          clearScreen();
        }
      }
    }

    chBSemSignal(&smph);
    chThdSleepMilliseconds(2000);
  }
  return 0;
}
```

b. Auxiliar functions

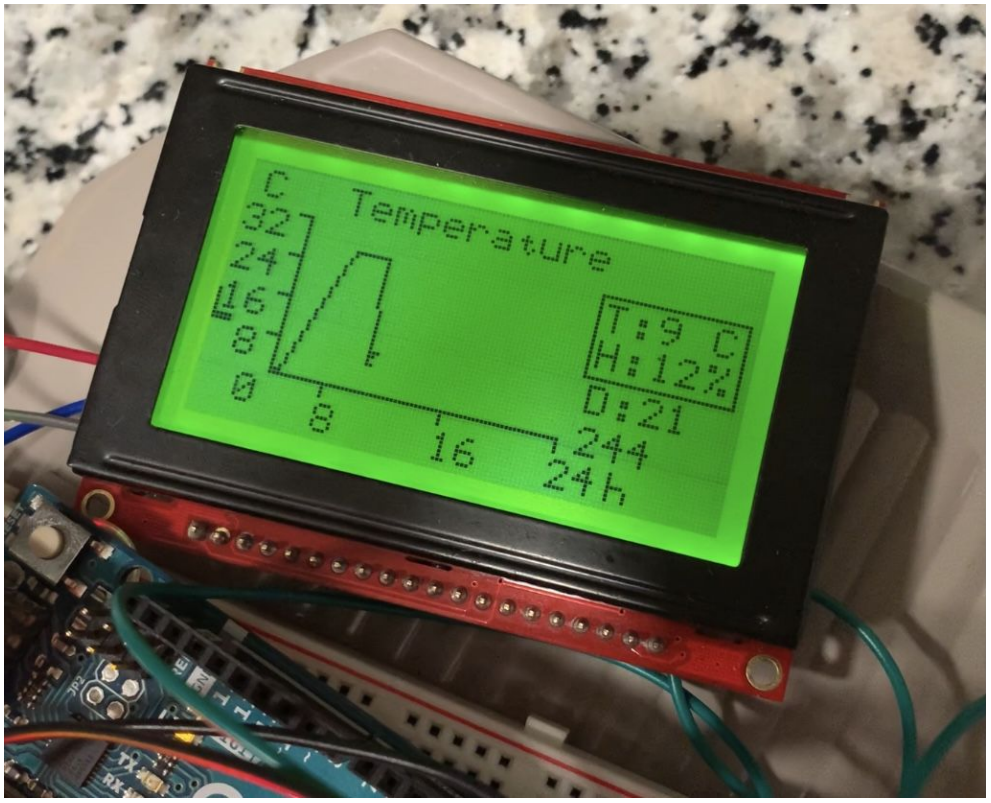
```
void stackHandler(int value)
{
  stackLineTemp[aux_counter][0] = stackLineTemp[aux_counter - 1][2];
  stackLineTemp[aux_counter][1] = stackLineTemp[aux_counter - 1][3];
  stackLineTemp[aux_counter][2] = stackLineTemp[aux_counter - 1][2] + 1;
  if (temperature > 38)
    stackLineTemp[aux_counter][3] = 14 + 38;
  else
    stackLineTemp[aux_counter][3] = 14 + temperature;

  stackLineHum[aux_counter][0] = stackLineHum[aux_counter - 1][2];
  stackLineHum[aux_counter][1] = stackLineHum[aux_counter - 1][3];
  stackLineHum[aux_counter][2] = stackLineHum[aux_counter - 1][2] + 1;
  if (humidity > 76)
    stackLineHum[aux_counter][3] = 14 + 76;
  else
    stackLineHum[aux_counter][3] = 14 + roundNo(humidity / 2);
}

void drawGraphLineTemp(int value)
{
  if (value > 38)
    value = 38;

  // title
  lcdPrintf(25, 61, "Temperature", 0);
  lcdPrintf(4, 61, "C", 0);

  // legend
  lcdPrintf(7, 22, "%u", 8);
  lcdPrintf(1, 32, "%u", 18);
```

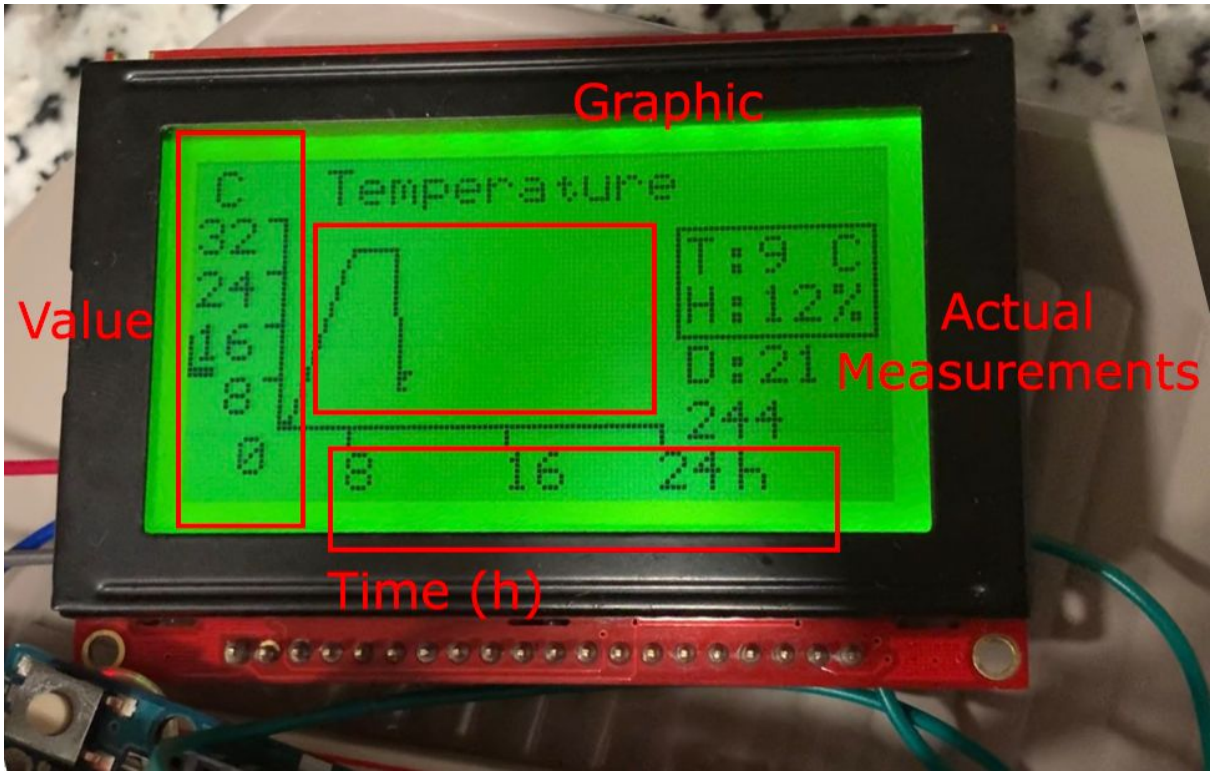
```
  lcdPrintf(1, 42, "%u", 29);
  lcdPrintf(1, 52, "%u", 38);

  // values
  lcdPrintf(105, 47, "%u", temperature);
  lcdPrintf(105, 38, "%u", humidity);
  lcdPrintf(105, 27, "%u", aux_counter);

  // 32/38 pixels alçada, i
  //int startX = 18;
  //int startY = 14;

  int i = 0;
  for (i = 0; i < aux_counter; i++) // draw all previous lines
  {
    drawLine(stackLineTemp[i][0],
             stackLineTemp[i][1],
             stackLineTemp[i][2],
             stackLineTemp[i][3], 0);
  }
  drawLine(stackLineTemp[aux_counter - 1][2],
           stackLineTemp[aux_counter - 1][3],
           stackLineTemp[aux_counter - 1][2] + 1,
           14 + value, 0); // draw current line
}

void drawGraphLineHum(int value)
{
  if (value > 76)
    value = 76;

  // title
  lcdPrintf(25, 61, "Humidity", 0);
  lcdPrintf(4, 61, "%%", 0);

  // legend
  lcdPrintf(7, 22, "%u", 19);
  lcdPrintf(1, 32, "%u", 38);
  lcdPrintf(1, 42, "%u", 57);
  lcdPrintf(1, 52, "%u", 76);

  // values
  lcdPrintf(105, 47, "%u", temperature);
  lcdPrintf(105, 38, "%u", humidity);
  lcdPrintf(105, 27, "%u", aux_counter);

  int i = 0;

  for (i = 0; i < aux_counter; i++)
  {
    drawLine(stackLineHum[i][0],
             stackLineHum[i][1],
```

```
              stackLineHum[i][2],
              stackLineHum[i][3], 0); // draw all previous lines
  }
  drawLine(stackLineHum[aux_counter - 1][2],
           stackLineHum[aux_counter - 1][3],
           stackLineHum[aux_counter - 1][2] + 1,
           14 + roundNo(value / 2), 0); // draw current line
}

void drawStructure()
{
  // info
  lcdPrintf(92, 47, "T:", 0);
  lcdPrintf(92, 38, "H:", 0);
  lcdPrintf(92, 27, "D:", 0);
  //
  lcdPrintf(10, 11, "%u", 0);
  lcdPrintf(118, 47, "C", 0);
  lcdPrintf(118, 38, "%%", 0);
  //mainframe
  drawLine(17, 13, 17, 52, 0);
  drawLine(18, 13, 87, 13, 0);
  //
  drawLine(14, 52, 17, 52, 0);
  drawLine(14, 42, 17, 42, 0);
  drawLine(14, 32, 17, 32, 0);
  drawLine(14, 22, 17, 22, 0);
  //
  drawLine(30, 12, 30, 10, 0);
  drawLine(59, 12, 59, 10, 0);
  drawLine(87, 12, 87, 10, 0);
  //
  drawBox(90, 49, 125, 29, 0);
  // legend
  lcdPrintf(30, 8, "%u", 8);
  lcdPrintf(59, 8, "%u", 16);
  lcdPrintf(87, 8, "%u", 24);
  lcdPrintf(101, 8, "h", 0);
  //
}

//  Prints a text in the LCD screen
void drawLine(int x1, int y1, int x2, int y2, int set)
{
  //draws a line from two given points. You can set and reset just as the pixel
function.
  sdPut(&SD1, (uint8_t)0x7C);
  sdPut(&SD1, (uint8_t)0x0C);
  sdPut(&SD1, (uint8_t)x1);
  sdPut(&SD1, (uint8_t)y1);
  sdPut(&SD1, (uint8_t)x2);
  sdPut(&SD1, (uint8_t)y2);
```

```
  sdPut(&SD1, (uint8_t)0x01);

  chThdSleepMilliseconds(10);
}

void drawBox(int x1, int y1, int x2, int y2, int set)
{
  //draws a box from two given points. You can set and reset just as the pixel function.
  sdPut(&SD1, (uint8_t)0x7C);
  sdPut(&SD1, (uint8_t)0x0F);
  sdPut(&SD1, (uint8_t)x1);
  sdPut(&SD1, (uint8_t)y1);
  sdPut(&SD1, (uint8_t)x2);
  sdPut(&SD1, (uint8_t)y2);
  sdPut(&SD1, (uint8_t)0x01);

  chThdSleepMilliseconds(10);
}

int roundNo(float num)
{
  return num < 0 ? num - 0.5 : num + 0.5;
}

void clearScreen()
{
  //clears the screen, you will use this a lot!
  sdPut(&SD1, (uint8_t)0x7C);
  sdPut(&SD1, (uint8_t)0);

  drawStructure();
}

void lcdPrintf(int x, int y, char text[], int value)
{

  sdPut(&SD1, (uint8_t)0x7C);
  sdPut(&SD1, (uint8_t)0x18);
  sdPut(&SD1, (uint8_t)x);
  chThdSleepMilliseconds(10);

  sdPut(&SD1, (uint8_t)0x7C);
  sdPut(&SD1, (uint8_t)0x19);
  sdPut(&SD1, (uint8_t)y);
  chThdSleepMilliseconds(10);

  chprintf((BaseSequentialStream *)&SD1, text, value);
  chThdSleepMilliseconds(10);
}
```

4. **Final result**
    a. Every 5 iterations, the screen switch between showing the Temperature and the Humidity
    b. All values for the lines are saved on an array of coordinates, so we can keep showing the graphics for 24h.





Reference video: https://www.youtube.com/watch?v=WJ3wlqHZit0