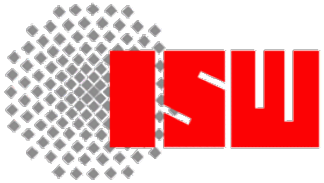


Universität Stuttgart
Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen

MEDIS – Module 2

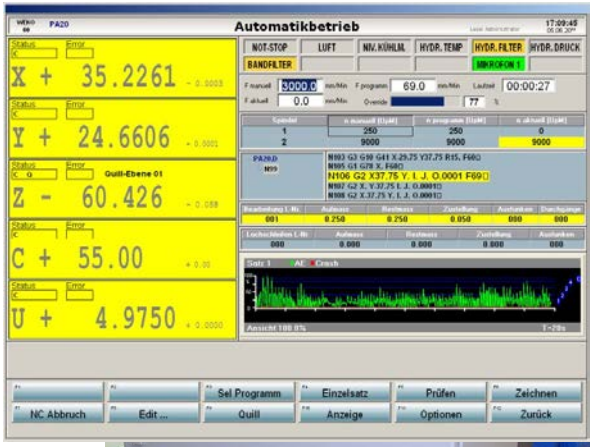
Microcontroller based systems for controlling industrial processes

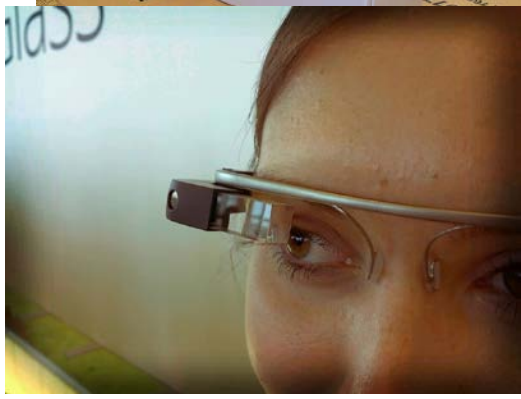
Chapter 5: Graphic system

M. Seyfarth, Version 0.1

User Interface

- The point or action a human interacts with a machine
  - Most simple way: a light and a push button
  - Up to 3D-graphic systems with haptic interaction
- User interface must be adapted to the needs and abilities of the user
  - Ergonomic and user-friendly interface design
  - Usability engineering
  - Research about cognition
- Two main tasks
  - Operate the machine (input to machine)
  - Monitor the machine and the process (output of machine)





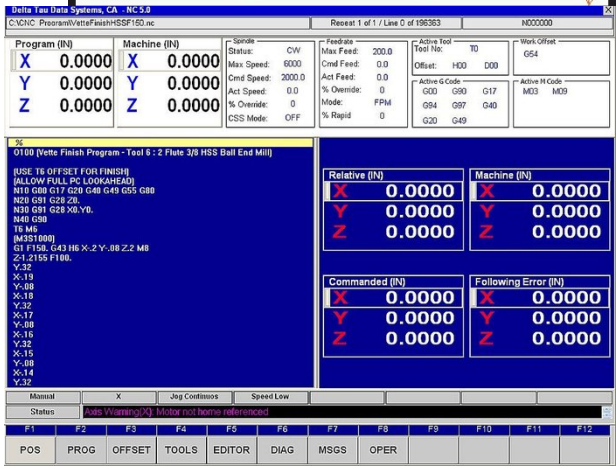
Source: www.adcos.ch; www.wikipedia.de

Universität Stuttgart

Types of User interfaces

- Machine operating panel
  - Lights and LEDs for signaling machine state and process state
  - Push buttons, switches, rotary switches , slider with pictograms or labeling for operating the machine
  - Connection to control mostly by point-to-point wiring

- Graphical user interfaces
  - (Color) display with graphical elements and text elements for signaling machine state and process state
  - Hardware buttons, switches, slider, …
  - Touch display with graphical buttons, roll bars, switches, …
  - Connection to control via bussystem (CAN, USB, Ethernet, ..), serial interface, proprietary connection.



Source: Bosch Rexroth, SPSTiger, Ind-Techno, AllenBradley

Universität Stuttgart

1.1 Types of user interfaces

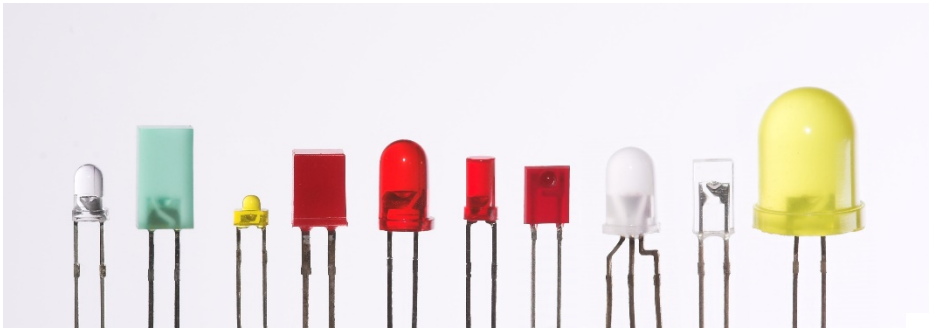
1.2 Types of graphical user interfaces (especially for microcontroller)

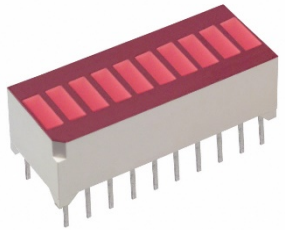
1.3 Programming of graphical user interfaces

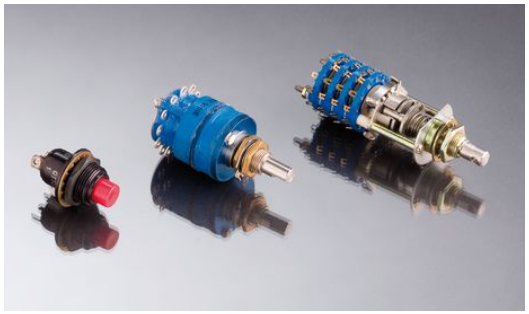
1.4

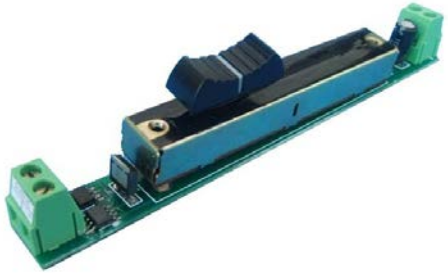
Hardware User Interface

- Buttons, LEDs, Bars, … with pictograms and labeling
  - Direct connection to microcontroller by single wiring
  - Limited functionality
  - Use of many pins of microcontroller



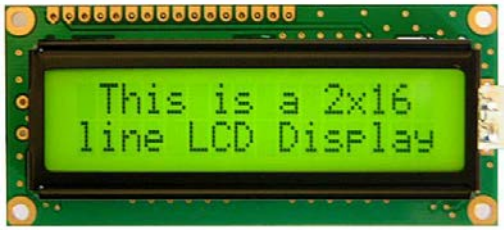


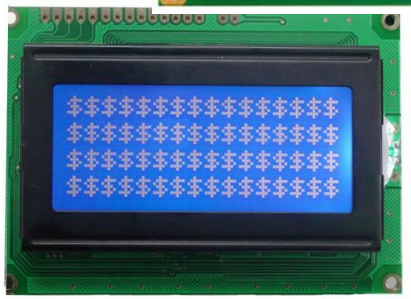




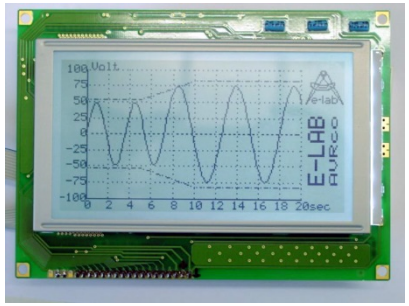
Graphical displays

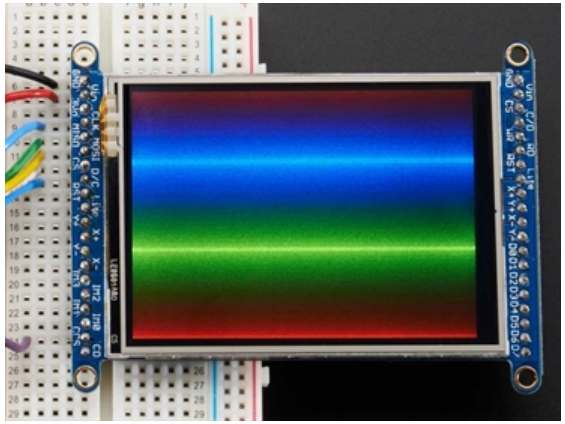
- Text displays
  - 1 to 4 lines
  - Green/blue backlight, monochrome





- Graphic displays
  - 1.5 … 3“ size
  - 160x128 … 320x240 pixel
  - Monochrome or color
  - Own graphic controller (e.g. IL9341) with ram buffer
  - Partly with touch capability

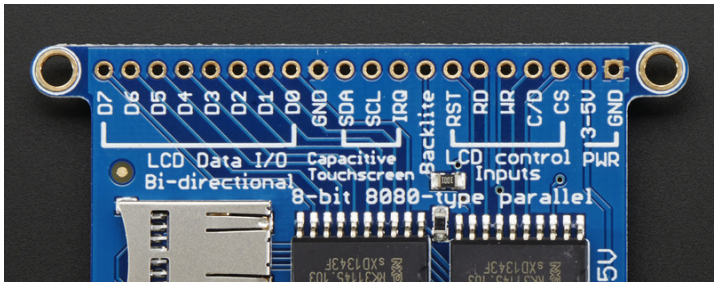




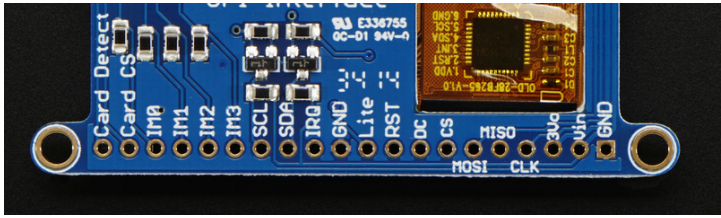
Connection of graphic display to microcontroller

Universität Stuttgart

- 8-bit digital interface
  - Use of 8 pins plus 4 pins (control) → many pins
  - Fast mode
  - 8 bit of data sent parallely



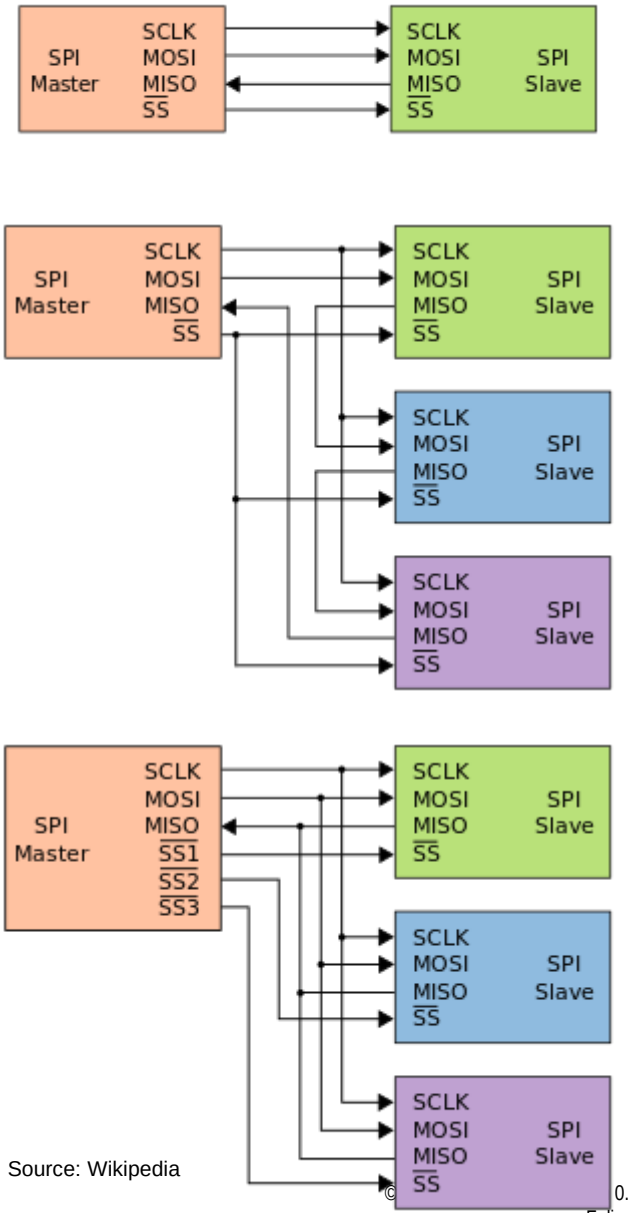
- SPI interface
  - Use of 4 pins
  - Not so fast (2-4-times slower than 8 bit mode)
  - Easy use of microSD card socket on same SPI bus

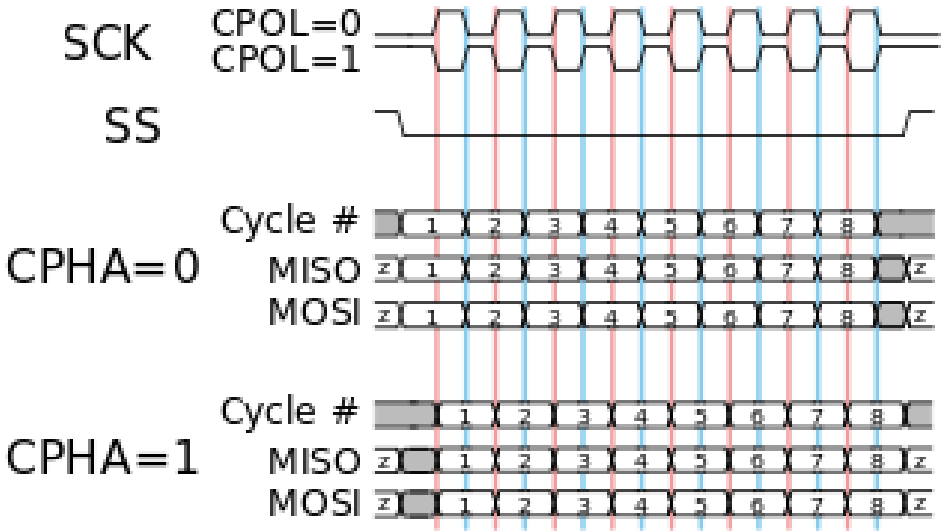


SPI: Serial Peripheral Interface

Serial Peripheral Interface

- Bus system developed by Motorola
- Master-Slave principle
  - SCLK: Serial clock, used by master for synchronisation
  - MOSI: Master Output, Slave Input, serial data out of a unit
  - MISO: Master Input, Slave Input, serial data in of a unit
  - SS (active low): chip select, controlled by master
- Full-duplex capability



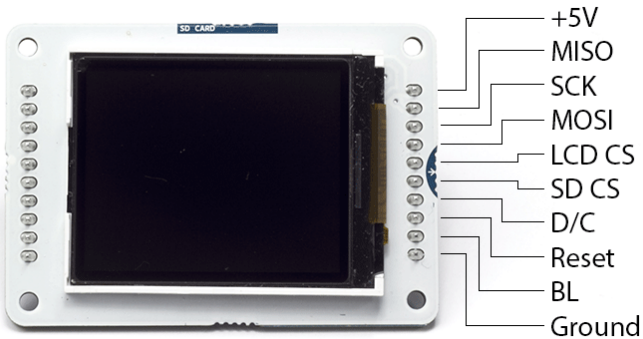


Source: Wikipedia

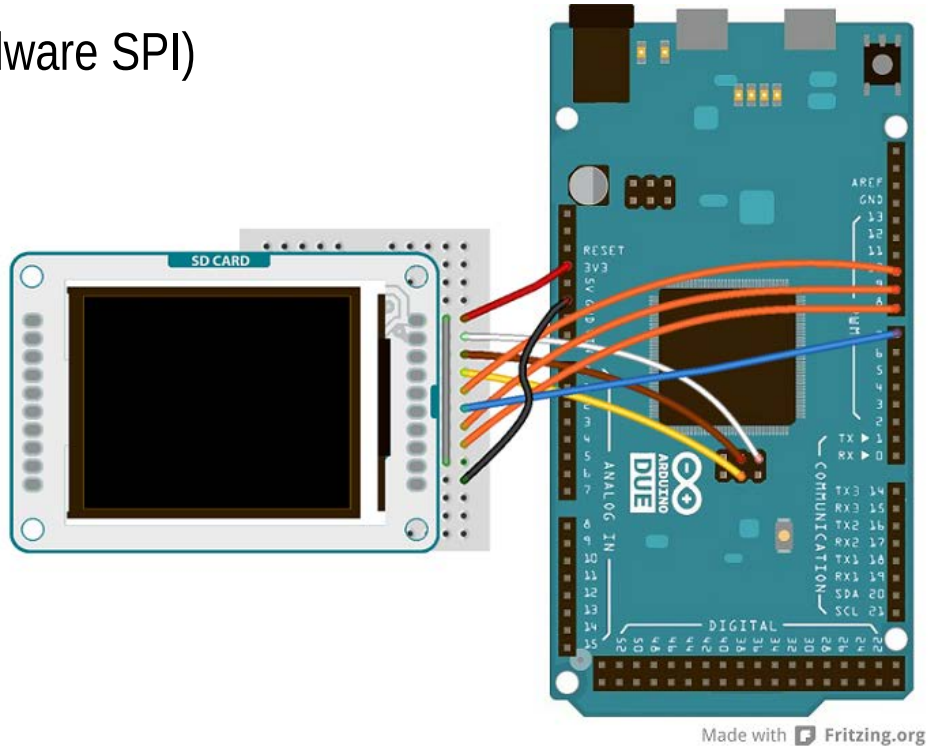
Connection of graphic display

Universität Stuttgart

- In this course the Arduino TFT display is used
  - Backlit LCD screen, with headers → easy to use
  - 1.77“ screen size with 160x128 pixel resolution
  - Runs on 5V
  - Graphic controller ILI9163
  - Backlight dimmable by PWM-signal
  - Supported by TFT library



- Connection to Arduino Due (using hardware SPI)
  - +5V → +3.3V
  - MISO → white wire on SPI
  - SCK → brown wire on SPI
  - MOSI → yellow wire on SPI
  - LCD CS → digital pin, e.g. 10
  - SD CS → digital pin, e.g. 7
  - D/C → digital pin, e.g. 9
  - Reset → digital pin, e.g. 8
  - BL → +3.3V (not dimmable)
  - GND → GND



Programming graphic display

Universität Stuttgart

- Use TFT library
  - Object oriented approach, graphic display is an object of class „TFT“
  - Include necessary libraries
    - `#include <SPI.h> // Arduino SPI library for SPI programming`
    - `#include <SD.h>  // Sdcard library, only needed if SD card is uses`
    - `#include <TFT.h> // Arduino LCD library`
  - Define the pins the display is connected to the Arduino Due
    - `#define sd_cs  7  // only for SDcard`
    - `#define lcd_cs 10`
    - `#define dc     9`
    - `#define reset  8`
  - Create software object for accesing the display with the pin-numbers
    - `TFT MyScreen = TFT(lcd_cs, dc, reset);`
  - Reset the display and initialize it
    - `MyScreen.begin();`

Programming graphic display

Universität Stuttgart

- Use TFT library - continued
  - Set background colour, erases everything of the selected colour
    - `MyScreen.background(255, 255, 255);  // (red, green, blue)=white`
  - Set colour for drawing lines and borders on the display
    - `MyScreen.stroke(red, green, blue); // all int 0..255`
    - `MyScreen.noStroke(); // called if no outline of shapes is desired!`
  - Write text to the display (with new line)
    - `MyScreen.println();  // blank linke`
    - `Myscreen.println(F("First example"));`
  - Write text to a special position
    - `MyScreen.text(text, xpos, ypos); //text is a string („Example")`
  - Set size of the font
    - `MyScreen.setTextSize(size); //int size 1..5 (1=10pix,2=20pix,..)`

Programming graphic display

Universität Stuttgart

- Use TFT library - continued
  - Get size of the display, number of pixels in width and height
    - `x = MyScreen.width();  // int x`
    - `y = MyScreen.heigth(); // int y`
  - Draw a point at Position (xPos, yPos)
    - `MyScreen.point(xPos, yPos); // int xPos,yPos`
  - Draw a line from startpoint (xPosSt, yPosSt) to endpoint (xPosE, yPosE)
    - `MyScreen.line(xPosSt, yPosSt, xPosE, yPosE);  // all integer`
  - Draw a rectangle from startpoint (xPosSt, yPosSt) with size width and height
    - `MyScreen.rect(xPosSt, yPosSt, width, height); // all integer`
  - Draw a circle with center (xPos, yPos) and radius
    - `MyScreen.circle(xPos, yPos, radius); // all integer`
  - Set the filling colour of an object; must be called before the obejct is drawn
    - `MyScreen.fill(red, green, blue); // all integer 0..255`
    - `MyScreen.noFill() // called if no filling of shapes is desired!`