Design and Driving of Embedded Displays Systems

Karlheinz Blankenbach

Pforzheim University, Germany

Prof. Dr. Karlheinz Blankenbach
Pforzheim University
Tiefenbronner Str. 65
D-75175 Pforzheim, Germany

Phone : +49 7231 - 28 - 6658
Fax : +49 7231 - 28 - 6060
Email : kb@displaylabor.de
Web : www.displaylabor.de
Overview

Introduction

Display Measurements (WS)

Embedded Display Systems (SS)

Electives:

Displays (LCD, OLED, E-Paper)

3D, Touch, …

Summary & Outlook

• What are the requirements for embedded systems with a display?
• Which solutions exist?
Overview Embedded Display Systems

• Introduction

• Low resolution displays

• High resolution graphics systems

• Interfaces

Embedded PC, TV, high resolution, interfaces etc. see dedicated lecture
Übersicht zu Embedded Systems mit Displays

- Blockdiagramm eines typischen Embedded System mit Display
- Wie beeinflusst die Displayauflösung das Embedded System?
- Beispiele für Low Resolution und Graphik-Systeme
- Bedeutung von Display-Controllern
- „To do“ um Daten auf dem Display zu anzuzeigen
- Vergleich der Ansätze „Segment-Display“ und „Matrix-Display“
Definition of Embedded Systems in This Lecture

- µC … µP (4 – 32 Bit)
- Display: 8-Segment … WXGA
- Indoor & outdoor use
- An embedded system is in a housing
- Mostly stationary, some mobile systems
High End Embedded System Hardware

Focus of this lecture in terms of an Embedded System
**Systems Design for µC and Display Controller**

A. Segmented Display (4x Segm. 8)  
   Display controller  
   µC (8-32 Bit) with integrated DC

B. “Display module”  
   Display (Character ... 240x128) with integrated Display controller  
   µC (4 - 32 Bit)

C. Display (≥ QVGA)  
   Display controller  
   µC (32 Bit)

D. Display (≥ QVGA)  
   Display controller  
   µC (32 Bit) with integrated DC or FPGA  
   Typically for PCs
   Some ARM, new ATOM

High speed  
Low speed
Tasks of Subassemblies of Embedded System with Display

- \( \mu \text{C} \)
  - Delivers data to be displayed to display controller
  - Interface to display controller, low speed if not video, …
  - No real time if not video

- Display controller
  - Updates display (video) RAM with data to be displayed
  - Real time data to display
  - Interface to display controller and display timing controller
  - Real time

- Display
  - Interface to display controller
  - Display electronics (timing controller TCON, row and column drivers) adapt digital data to electro-optic

Data to display (see GUI lecture) set display resolution and therefore \( \mu \text{C} \), software, interfaces and display controller!
# Systems Design for µC and Display Controller

<table>
<thead>
<tr>
<th></th>
<th>B</th>
<th>C</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Display</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Display controller</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>µC</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Merits
- Data µC to display only when pixel change
- Only one interface
- Freedom of design
- High res displays

## Shortcomings
- Mostly limited to low resolution b/w (some QVGA and e-paper modules available)
- Two high speed interfaces makes PCB design more complex
- Only a few solutions available but market increases
Issues for Driving of Displays (Non-PC)

• **Direct (µC - I/O)**
  + No additional IC needed if µC with display IO
  + Easy
  - Only lowest ‘resolution’
  - SW easy but every segment has to be set

• **Character & Graphics Controller with Built-in RAM**
  + Minor SW because of e.g. character generator
  + No load on µC, ‘no’ timing issues
  - Only up to mid resolution (240 x 128), some QVGA available

• **Graphics Displays (≥ QVGA, color)**
  + High resolution
  + Some high end µC with built-in display support
  - Huge timing issues
  - Real time display driving to avoid failures
  - No support for pixel and graphics (use libs or OS)
Display Controller vs. Resolution Overview

Size, price, ...

Direct drive  MUX  Passive Matrix  Active Matrix

§ “Low resolution displays” (mostly b/w, some with GS)

§ “High resolution displays” (mostly color)
### Resolution - Interface - Display RAM

- **Pixel frequency** = Resolution × frame frequency  
  (limit for parallel interfacing)
- **Data rate** = Pixel frequency × color depth  
  (limit for serial interfacing)
- **Display RAM** = Resolution × color depth  
  (limit for 8-bit microcontroller)

**Remarks:**
- Parallel interfaces: 24 or 48 parallel lines
- Serial interfaces: LVDS (industrial), DVI (PC), HDTV (TV)
- Segment 8 displays: e.g. 4 digit (Clock + icons) 32 pixel ≡ 4 byte

<table>
<thead>
<tr>
<th>Examples</th>
<th>C 128 x 96 (black/white)</th>
<th>320 x 240 (QVGA, 18-bit color)</th>
<th>D 640 x 480 (VGA, 24-bit color)</th>
<th>D 1,366 x 768 (WXGA, 24-bit color)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong># of pixel</strong></td>
<td>12,288</td>
<td>76,800</td>
<td>307,200</td>
<td>1,049,088</td>
</tr>
<tr>
<td><strong>Pixel frequency</strong></td>
<td>0.7 MHz</td>
<td>4.6 MHz</td>
<td>18.5 MHz</td>
<td>63 MHz</td>
</tr>
<tr>
<td><strong>Data rate</strong></td>
<td>0.7 MHz</td>
<td>83 MHz</td>
<td>0.45 GHz</td>
<td>1.5 GHz</td>
</tr>
<tr>
<td><strong>Display RAM</strong></td>
<td>12 kB</td>
<td>169 kB</td>
<td>0.9 MB</td>
<td>3 MB</td>
</tr>
</tbody>
</table>
Character Design vs. Resolution

- **Segment 8**: 10¹ pixels
- **Starburst**: 10² pixels
- **Matrix 5x7**: 10³ pixels
- **QVGA 10x15**: 10⁴ pixels
- **XGA 10x15**: 10⁵ pixels

Driving of Module:
- μC
- Display controller
- 'PC'

Data Rate:
- low
- medium
- high

Software Requirements:
- minor
- high w/o OS or Lib
- minor (OS)
Displays for Mobile Applications as a System

Multimedia System on LCD MDL
（マルチメディアシステム搭載型 LCD MDL）

… only suitable for large volume mass production,
strongly NOT recommended for industrial use!

′Everything′ is in an single IC!
Embedded Display Systems Design (No Embedded PC & E-Signage)

- **Size**: 0.5“ ... ~20“ (>: PC-like, public displays, …)
- **Pixel**: 7 … millions (1x Segment 8 … WXGA)
- **Color**: Monochrome - grey shade - area color - color - full color
- **Lifetime**: 2y … 20 y ; operating time 100 h … 20 y, storage time
- **Environment**: outdoor … indoor, stationary … mobile, …
- **Pieces per year**: 10 … millions
  (e. g. high end measurement device … household appliances)
- **Life cycle**: replacement of subassemblies, supply chain, recycling, …

**Focus on non-ePC systems developed in Europe!**
Low Volume Approach Using “Intelligent Panels”

Issue: Driving displays is often very complex and challenging as very different interfaces, connections, real time requirements for pixel stream, …

Solution for low volumes: Use of panels with built-in microcontroller or PC with text and graphics functionality addressed simple and short commands and serial interface (I²C, SPI)

Drawback: Higher display cost but compensated by less hard- and software

Monochrome graphics – AM LCD graphics – hi res panel PC / touch as option

Suppliers e.g. electronic assembly, Demmel or PanelPC suppliers (display + PC)
Low Volume Approach Using “Intelligent Panels”

Simple SW and IF saves time for development.
Rule of thumb:
Reasonable for volumes up to ~ 1,000 p/y

Source: electronic assembly
Just as Overview: Industrial PC & Computer-On-Module

- **System**: PC-like with operating system and graphics adapter often with Touch Screen (also as Panel PC)

- **Display**: full color, > 10.4”, standard PC interfaces: VGA, DVI, … + LVDS

- **Advantages**: - Standard hardware (often from many companies)
  - Operating System reduced SW effort
  - Easy to integrate (also as COM)
  - Low TCO (total cost of ownership)
  - Faster TTM (time-to-market)
  - Significantly reduced risk (CE-certified iPC or COM)

- **Disadvantages**: Only good for low volumes up to 10,000 pa
Just as Overview: Industrial PC & Computer-On-Module

- iPC

System case: C

- Panel PC with Touch
Just as Overview: **Computer-On-Module**

- **Various standards**
  - PC104 (oldest)
  - ETX
  - DIMM-PC (5 Volt, 68 x 40 mm²)
  - …

**Basic idea:**
- Buy COM
- Make dedicated PCB with peripheral electronics
Just as Overview: **Industrial PC & Computer-On-Module**

Source: ARC

Source: VDC

**Industrial PC Business Worldwide**

- 2010: $379.8
- 2011: $474.4
- 2012: $574.6
- 2013: $674.1
- 2014: $771.6
- 2015: $882.9

- 2011: $52.6
- 2012: $67.6
- 2013: $93.9
- 2014: $125.8
- 2015: $161.5
- 2016: $212.5

**Merchant Carriers**

**COM Modules**
Just as Overview: **Computer-On-Module**

**COM and COM Carrier Shipments by Vertical Market, 2005 & 2007**

- **Communications: Edge**
  - 9%
- **Communications: Core**
  - 7%
- **Industrial Control & Automation**
  - 31%
- **Transportation**
  - 13%
- **Medical**
  - 14%
- **Instrumentation**
  - 11%
- **Military/Aerospace/Defense**
  - 7%
- **Other**
  - 10%

**2004: 3.7 B$**

**Germany:** Kontron, MEN Mikro, DIGITAL-LOGIC, Congatec, F&S, …
**Others:** Advantech, ADLink, Radisys
Other Embedded Systems with Graphic Displays

- **Panel PC**
  - IPC e.g. mounted on display
  - Easy to integrate into systems

  Just for reference because of COTS systems!

- **Serial Interfacing**
  - Easy interfacing via RS232, USB or 8-Bit
  - 8 Bit µC can ‘drive’ up to XGA
  - USB up to QVGA video stream
  - Suitable for low volume and if only low computing power is required
Low Cost, Low Information Content Displays

Characteristics

- 1 ... ~1,000 pixel
- Often icons, numbers, ...
- ‘Cheap’
- Direct drive, multiplex or Passive Matrix
- Mostly for 8 bit µCs
- (rest depends on application)

Technologies

- LCD
- OLED
- LED
- VFD
- E-paper
Embedded Systems with Graphic Displays

• **Embedded Systems (16 - 32 bit)**
  - Lower resolution than PCs
  - Typically digital output, analogue output good for debugging
  - Often with OS
  - Often with graphic controllers

• **Low Resolution b/w**
  - µC needs display controller
  - Limited to about 240 x 128

- µP bus
- B/W STN
- FPD output
- CSTN
- TFT
- Graphics controller
Segmented vs. Graphics Displays

Segmented LCDs
- Low power
- Sunlight readable
- Easy to customize
- Sharp icons
- Fixed screen content

Graphics Display
- Screen content easy to change or adapt
- Low power only and sunlight readable only for monochrome
- High power and limited sunlight performance for color displays
Summary of Introduction

• Data to display should be the starting point for embedded display systems

• Display resolution determines controller

• Low information content LCDs show low power and sunlight readable

• High resolution and low volume systems often base on COTS iPCs

Display are the key component in terms of

• Hardware
• Software
• Price
Overview Embedded Display Systems

• Introduction

• Low resolution displays

• High resolution graphics systems

• Interfaces
Übersicht Low Resolution Displays

- Ausführungen:
  - Einzel-Segmente (Pixel), die direkt angesteuert werden
  - 7-Segment-Displays
  - Character-Displays
  - Monochrome Grafik bis ca. 240 x 128

- Technologien: LCD, LED (meist 7-Segment), VFD, OLED

- Ansteuerung:
  - Einzel-Segmente : I/O-Pin, µC-Controller, etc.
  - 7-Segment-Displays : 7-Segment-Controller, µC-Controller
  - Character-Displays : HD44780 (oder kompatibel)
  - Monochrome Grafik : T6963
Overview: Low Resolution Driving

• Direct (µC - I/O)

A

• Character Controller

B

RAM built in, no load for µC

• Graphics Controller

C

Examples for LCD, other technologies similar
Trend to Small Systems with µC and LCD - Output

- Customized LCD with lowest power consumption and sunlight readability
- Commercial available starting at ~ 1,000 pieces
- Prevents faking

⇒ Simple, cheap and dedicated system. Great for IoT!
Simple LCD 2 Digit Display

- Standard direct drive LCD
- 2 digit Segment 8
- PIN contact:
  - Easy PCB integration
  - No extra fixture of display needed
  - No cable and connector needed
  - Suitable for harsh environments including vibration and shock
- PIN contact fixes
- Price: 1-10 pieces: ~ 5 €
  > 100 pieces: ~ 4 €

DEVICE DESCRIPTION
The LTD202 is a 2-digit, 7-segment LCD. It is intended for use in small counter and indicator panel applications.

QUICK REFERENCE DATA

<table>
<thead>
<tr>
<th>Viewing area dimensions</th>
<th>23.5 x 18.4 mm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall glass dimensions</td>
<td>27.9 x 30.4 mm</td>
</tr>
<tr>
<td>Thickness</td>
<td>2.7 +/- 0.4 mm</td>
</tr>
<tr>
<td>Digit height</td>
<td>12.7 mm</td>
</tr>
<tr>
<td>Preferred viewing direction</td>
<td>6 o'clock</td>
</tr>
<tr>
<td>Driving method</td>
<td>direct drive</td>
</tr>
</tbody>
</table>
Simple LCD 2 Digit Display

Dimensions in mm

<table>
<thead>
<tr>
<th>TYPE</th>
<th>ILLUMINATION Mode</th>
<th>CONNECTION Method</th>
<th>RELIABILITY GRADE</th>
<th>FAMILY CHARACTERISTICS</th>
<th>OPERATING VOLTAGE (V)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LTD202R-12</td>
<td>reflective</td>
<td>with fixed pins</td>
<td>commercial</td>
<td>TR0</td>
<td>3.5 – 6.5</td>
</tr>
<tr>
<td>LTD202R-22</td>
<td>reflective</td>
<td>with fixed pins</td>
<td>extended</td>
<td>TR2</td>
<td>3.5 – 6.5</td>
</tr>
<tr>
<td>LTD202F-22</td>
<td>transflective</td>
<td>with fixed pins</td>
<td>extended</td>
<td>TF2</td>
<td>3.5 – 6.5</td>
</tr>
</tbody>
</table>

Note: (1) drive method = direct drive for all types
(2) see chapter "Family Characteristics" for complete specification
Simple LCD 2 Digit Display

- Display has 18 pins, 17 are connected
- Segment defines “pixel” on display like a, b, c, … with digit code “1” or “2”
- COM can be regarded as common “ground” (see below)

<table>
<thead>
<tr>
<th>PIN NO.</th>
<th>SEGMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>comm</td>
</tr>
<tr>
<td>2</td>
<td>p1</td>
</tr>
<tr>
<td>3</td>
<td>e1</td>
</tr>
<tr>
<td>4</td>
<td>d1</td>
</tr>
<tr>
<td>5</td>
<td>c1</td>
</tr>
<tr>
<td>6</td>
<td>p2</td>
</tr>
<tr>
<td>7</td>
<td>e2</td>
</tr>
<tr>
<td>8</td>
<td>d2</td>
</tr>
<tr>
<td>9</td>
<td>c2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PIN NO.</th>
<th>SEGMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>b2</td>
</tr>
<tr>
<td>11</td>
<td>a2</td>
</tr>
<tr>
<td>12</td>
<td>f2</td>
</tr>
<tr>
<td>13</td>
<td>g2</td>
</tr>
<tr>
<td>14</td>
<td>b1</td>
</tr>
<tr>
<td>15</td>
<td>a1</td>
</tr>
<tr>
<td>16</td>
<td>f1</td>
</tr>
<tr>
<td>17</td>
<td>g1</td>
</tr>
<tr>
<td>18</td>
<td>n.c.</td>
</tr>
</tbody>
</table>
Direct Drive for 8-Segment LCD

Direct drive = every LCD segment is connected to a display controller output pin.
Direct Drive for 8 - Segment LCD

Code example

' LED 8-Seg

define Anzeige BYTEPORT[1]

' Binary BIT 7 6 5 4 3 2 1 0

' Segment h g f e d c b a

Point

All the names mean the same
- Seven Segment (7 “pixel” without point)
- 8-Segment (has 8 segments incl. point)
- Segment 8 (looks like 8)

This segment definition is used for all display technologies for 8-Segment displays
**Direct Drive for 8-Segment LCD with 4 Digits**

- 8 lines per digit
- 4 digits: 32 segments + 1 common

μC with 33 display controller outputs would be required → not reasonable, solution multiplex
Simplified Timing Diagram Direct Drive for 8-Segment LCD

Segment is “ON” if voltage difference $\Delta V$ exist (same for LED)

Remark: Voltage inversion for LCDs not shown
Multiplex Drive for 8 - Segment LCD with 4 Digits

- 2 seg per digit:
- 4 digits: 8 seg

# of µC-IOs is also limiting!

- MUX
- Duty

+ 4 common

12 lines

≈ 35 % of direct drive
Multiplex save outputs
but requires higher
Software effort as segments
are multiplexes
Multiplex Drive Waveforms for 8-Segment LCD (4 Digits)

- Complex waveforms for COMMON and SEGMENT
- Frame inversion (see § LCD)
- 4 voltages

MUX1:4

1 frame 1 frame 1 frame 1 frame

M

Data

COM1

- V1
- V2
- V3
- VSS

COM2

- V1
- V2
- V3
- VSS

COM3

- V1
- V2
- V3
- VSS

COM4

- V1
- V2
- V3
- VSS

SEGn

Data

→ Cannot be done by µC, display controller or LCD-µC needed!
µC with Built-in Display Controller for Segmented LCDs

- Typically up to
  96 segments and 4 commons
  → about 400 ‘pixel’

- Example:
  4 x 8-Segment digits + icons
  for clock and temperature, …

- There are many 8-32 bit-µC with
  built-in display controller for
  segmented LCD output (some LED)

**MICROCHIP PIC16C9XX**
8-Bit CMOS Microcontroller with LCD Driver

**LPC11D14**
NXP
32-bit ARM Cortex-M0 microcontroller; 32 kB flash and 8 kB SRAM; 40 segment x 4 LCD driver
**µC with Built-in LCD Output: NXP LDC11D14**

**LPC11D14**

32-bit ARM Cortex-M0 microcontroller; 32 kB flash and 8 kB SRAM; 40 segment x 4 LCD driver

- LCD driver
  - 40 segments.
  - One to four backplanes.
  - On-chip display RAM with auto-increment addressing.

"Separate" LCD controller built-in

- S = segment = SEG, here 40
- BP = backplane = COM, here 4

Max. number of "pixel":
SEG x COM = 40 x 4
= 160
μC with Built-in LCD Output: NXP LDC11D14

Built-in LCD controller

"Complex LCD voltage supply"

Internal I²C interface
μC with Built-in LCD Output: NXP LDC11D14

Built-in LCD controller

The PCF8576D is a peripheral device which interfaces to almost any Liquid Crystal Display (LCD)\(^1\) with low multiplex rates. It generates the drive signals for a multiplexed LCD containing up to four backplanes and up to 40 segments cascaded for larger LCD applications. The PCF8576D is compatible with microcontrollers and communicates via the two-line bidirectional I\(^2\)C-bus.

“Complex LCD voltage supply”

Table 4. Selection of display configurations

<table>
<thead>
<tr>
<th>Number of Backplanes</th>
<th>Segments</th>
<th>Digits/Characters</th>
<th>Dot matrix/Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>7-segment</td>
<td>14-segment</td>
</tr>
<tr>
<td>4</td>
<td>160</td>
<td>20</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>120</td>
<td>15</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>80</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>40</td>
<td>5</td>
<td>2</td>
</tr>
</tbody>
</table>

Internal I\(^2\)C interface
µC with Built-in LCD Output: NXP LDC11D14

Built-in LCD controller

**Table 4. Selection of display configurations**

<table>
<thead>
<tr>
<th>Number of Backplanes</th>
<th>Segments</th>
<th>Digits/Characters</th>
<th>Dot matrix/Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>7-segment</td>
<td>14-segment</td>
</tr>
<tr>
<td>4</td>
<td>160</td>
<td>20</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>120</td>
<td>15</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>80</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>40</td>
<td>5</td>
<td>2</td>
</tr>
</tbody>
</table>

**Backplane outputs**

The LCD drive section includes four backplane outputs BP0 to BP3 which must be connected directly to the LCD. The backplane output signals are generated in accordance with the selected LCD drive mode. If less than four backplane outputs are required, the unused outputs can be left open-circuit.

**Display RAM**

The display RAM is a static 40 × 4-bit RAM which stores LCD data. There is a one-to-one correspondence between the RAM addresses and the segment outputs, and between the individual bits of a RAM word and the backplane outputs. For details, see Ref. 2.
μC with Built-in LCD Output: NXP LDC11D14

Static (direct) Drive

Waveforms for LCD segments

\[ \Delta V > 0: \text{ON, black} \]

\[ \Delta V = 0: \text{OFF, white} \]
µC with Built-in LCD Output: NXP LDC11D14

**Multiplex 1:2 Drive**

MUX 2 means 2 COMs (backplanes).

The waveforms are even for MUX 1:2 complex and hardly to handle without LCD controller.

---

**Diagram Description:**

- **BP0** and **BP1** represent the multiplexing configurations.
- **Sn** and **Sn+1** denote the segment lines.
- **V_LCD**, **2V_LCD/3**, and **V_SSS** are the voltage levels used in the waveforms.

(a) Waveforms at driver.

---

**Table:**

<table>
<thead>
<tr>
<th>Rows/backplane outputs (BP)</th>
<th>display RAM address &amp; segment outputs (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>n+1</td>
</tr>
<tr>
<td>a</td>
<td>b</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

**Columns:**

- **MSB:**
  - a
  - b
  - g
  - c
- **LSB:**
  - d
  - DP

Blankenbach / Pforzheim Univ. / www.displaylabor.de / Embedded Systems / SS 2015
μC with Built-in LCD Output

- Low power
- Sunlight readable
- Cheap and easy system
µC with Built-in Display Controller for Segmented LCDs

BASIC EXTERNAL CONNECTION DIAGRAM

LCD panel 56 × 32

S1C17704

Typically > 100 pins
μC with Built-in Display Controller for Segmented LCDs

Block diagram

Frees μC load

Display glass

Figure 22.1.1 Configuration of LCD Driver and Drive Power Supply
μC with Built-in Display Controller for Segmented LCDs

Registers of display controller

Table 22.8.1 List of LCD Registers

<table>
<thead>
<tr>
<th>Address</th>
<th>Register name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x50a0</td>
<td>LCD_DCTL (LCD Display Control Register)</td>
<td>Controls the LCD display.</td>
</tr>
<tr>
<td>0x50a1</td>
<td>LCD_CADJ (LCD Contrast Adjust Register)</td>
<td>Controls the contrast.</td>
</tr>
<tr>
<td>0x50a2</td>
<td>LCD_CCTL (LCD Clock Control Register)</td>
<td>Controls the LCD clock duty.</td>
</tr>
<tr>
<td>0x50a3</td>
<td>LCD_VREG (LCD Voltage Regulator Control Register)</td>
<td>Controls the LCD drive voltage regulator.</td>
</tr>
<tr>
<td>0x50a4</td>
<td>LCD_PWR (LCD Power Voltage Booster Control Register)</td>
<td>Controls the LCD voltage booster.</td>
</tr>
<tr>
<td>0x50a5</td>
<td>LCD_IMSK (LCD Interrupt Mask Register)</td>
<td>Enables/disables interrupt.</td>
</tr>
<tr>
<td>0x50a6</td>
<td>LCD_IFLG (LCD Interrupt Flag Register)</td>
<td>Indicates/sets interrupt occurrence status.</td>
</tr>
</tbody>
</table>

*LC[3:0]: LCD Contrast Adjustment Bits in the LCD Contrast Adjust (LCD_CADJ) Register (D[3:0]/0x50a1)

Table 22.6.2.1 LCD Contrast Adjustment

<table>
<thead>
<tr>
<th>LC[3:0]</th>
<th>Contrast</th>
</tr>
</thead>
<tbody>
<tr>
<td>0xf</td>
<td>High (dark)</td>
</tr>
<tr>
<td>0xe</td>
<td>↑</td>
</tr>
<tr>
<td></td>
<td>↓</td>
</tr>
<tr>
<td>0x1</td>
<td>Low (light)</td>
</tr>
<tr>
<td>0x0</td>
<td>(Default: 0x0)</td>
</tr>
</tbody>
</table>
Code Example For µC with LCD - Output

LCD_BASE - base address of LCD module
+__msp430_have_lcdlowr - if LCD controller supports the LCDLOWR bit

*/

00 -24,7 +25,9 00

/* the names of the mode bits are different from the spec */
#define LCDON 0x01
+if defined(__msp430_have_lcdlowr)
#define LCDLOWR 0x02
+endif
#define LCDSON 0x04
#define LCDMXO 0x08
#define LCDMX1 0x10
00 -36,25 +39,26 00
#define LCD2MUX (LCDMXO|LCDSON)
#define LCD3MUX (LCDMX1|LCDSON)
#define LCD4MUX (LCDMX1|LCDMX0|LCDSON)

/* Group select code with Bits 5-7 */

#define LCDSG0 0x00 /* S0 - S1 02 - 029 */
(default) */
#define LCDSG0_1 (LCDP0) /* S0 - S5 06 - 029 */
#define LCDSG0_2 (LCDP1) /* S0 - S9 010 - 029 */

MUX 4 (see VFD)

~ 30 segments
**MUX Drive for 8-Segment VFD**

**Grid (Scan)**

- 1G
- 2G
- 3G
- 4G

**Anode (data)**

- a
- b
- c
- d
- e
- f
- g

Multiplex driving via scan of grids

VFD Driver

Display

1G 2G 3G 4G

4321
**MUX Drive for 8-Segment VFD**

### Grid (Scan)

- **PortNr.:** 16 15 14 13 12 11 10 9
- **BitNr.:** 7 6 5 4 3 2 1 0
- **G1 – 4:** g f e d c b a

...  
Anzeige = 0 ' Space out
GRID = 1
ein = 1
Looktab decoder, ein, Anzeige

### Anode (data)

- Anzeige = 0 ' Space out
- GRID = 2
- zehn = 2
- Looktab decoder, zehn, Anzeige

```
&B00111111 \ 0
&B00000110 \ 1
&B01011011 \ 2
```
LED Matrix Driving

Drives 4 digit 5 x 7 LED

µC

TTL

3.3V
47µF
100nF

4.7kΩ

CSET 26pF
RSET 53.6kΩ

V+
V+
GND
GND
GND

ROW 7 5 1 8 14 13
COL 1 2 3 4 5

MAXIM MAX6952

CLK
DIN
CS
DOUT
BLINK
OSC
ISET

00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 015 016 017 018 019 020 021 022 023
Zusammenfassung  Segment - Displays

• Meist direkt vom µC ansteuerbar
• Niedrige ‚Auflösung‘ erfordert nur wenig RAM
• Einzelpixel quasi nur noch als LEDs
• Segment-Displays ermöglichen große Fonts und ‚scharfe‘ Symbole bei relativ kleiner Größe (im Gegensatz zu Matrix-Displays)
Übersicht Character Displays

• Ausführungen:
  - 1 … 4 Zeilen
  - 8 … 40 Zeichen pro Zeile
  - meist Standard-Font,
    aber auch kyrillisch etc.

• Technologien: LCD, VFD, OLED

• Ansteuerung HD44780 (oder kompatibel)
  - 4- oder 8-Bit Datenbus
  - 3 Steuerleitungen (1 Port bei 4-Bit Daten)
  - Programmierbare Fonts
  - \(\mu C\) muss nur Daten senden wenn sich der Anzeigeinhalt
    ändern soll (also keine Echtzeit oder Permanent-Last)
Character Display Module Specification (Example)

• Standard LCD character module with LED backlight

• 2 lines, 16 characters

• Interface: Usually cable, PIN possible

• Display module must be mounted by screws (costly)

• Price: 1-10 pieces: ~ 10 €
  > 100 pieces: ~ 8 €
# Character Display Module Specification (Example)

## Electrical Interface

### Pin Assignment

<table>
<thead>
<tr>
<th>No.</th>
<th>Symbol</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$V_{ss}$</td>
<td>Gnd, 0V</td>
</tr>
<tr>
<td>2</td>
<td>$V_{dd}$</td>
<td>+5V</td>
</tr>
<tr>
<td>3</td>
<td>$V_0$</td>
<td>LCD Drive</td>
</tr>
<tr>
<td>4</td>
<td>RS</td>
<td>Function Select</td>
</tr>
<tr>
<td>5</td>
<td>R/W</td>
<td>Read/Write</td>
</tr>
<tr>
<td>6</td>
<td>E</td>
<td>Enable Signal</td>
</tr>
<tr>
<td>7-14</td>
<td>DB0-DB7</td>
<td>Data Bus Line</td>
</tr>
<tr>
<td>15</td>
<td>$A^*$</td>
<td>4.2V for LED</td>
</tr>
<tr>
<td>16</td>
<td>K</td>
<td>Power Supply for LED 0V</td>
</tr>
</tbody>
</table>

- **Power for character controller and LCD**
- **Control for character controller**
- **Data for character controller**
- **Power for LED backlight**
Fundamental Character Controller Approaches

- Dedicated to display characters (fonts) and low res graphics
- Two approaches:
  - Hardware (character controller)
  - Software (characters implemented in software)

<table>
<thead>
<tr>
<th>Character Controller</th>
<th>Hardware</th>
<th>Software</th>
</tr>
</thead>
<tbody>
<tr>
<td>Merits</td>
<td>Easy to use, “every µC”, large range of display module with built-in CC</td>
<td>Easy to adapt incl. icons, use of custom display glass</td>
</tr>
<tr>
<td>Shortcomings</td>
<td>Hard to change (internationalisation), higher cost</td>
<td>Needs implementation (more lines of code), only a few µCs available</td>
</tr>
<tr>
<td>Examples</td>
<td>HD 44780, T6963</td>
<td>Bitmap-2-C, SW</td>
</tr>
</tbody>
</table>
Character Controller HD 44780

Display RAM “frees” μC from real time data ⇒ “no” μC load

“Hardware” font referring to ASCII

μC
(typ. 16+ pins)

Simple μC IF

MPU interface

RS
R/W
E
DB 7
DB 0

HD 44780

Instruction Register (IR)

Control logic

Data Register (DR)

Timing generation

Segment & Common driver

Display Data RAM (DDRAM)

Character Generator RAM (CGRAM)

Character Generator ROM (CGROM)

SEG 1
SEG 40

LCD

COM 1
COM 16

Display RAM “frees” μC from real time data ⇒ “no” μC load

“Hardware” font referring to ASCII

Blankenbach / Pforzheim Univ. / www.displaylabor.de / Embedded Systems / SS 2015
**HD 44780 Character Controller Interfacing**

- **µC ↔ Controller**

  \[
  \begin{align*}
    \text{µC} & \quad \begin{array}{c}
      \text{P30 to P37} \\
      \text{P77} \\
      \text{P76} \\
      \text{P75}
    \end{array} \\
    \text{µC} & \quad \begin{array}{c}
      \text{E} \\
      \text{RS} \\
      \text{R/W}
    \end{array} \\
    \text{HD44780U} & \quad \begin{array}{c}
      \text{DB0 to DB7} \\
      \text{COM1 to COM16}
    \end{array} \\
    \text{LCD} & \quad \begin{array}{c}
      \text{Scan (row)} \\
      \text{Data (column)}
    \end{array}
  \end{align*}
  \]
  
- **Controller ↔ Display glass**

  Ansteuerung im Prinzip wie MUX mit 16 COMs (Scan) aber für LCDs relativ komplex
HD 44780 Programming

Character Generator

No need to program fonts

PC : Character generation by OS

Instructions

<table>
<thead>
<tr>
<th>FUNCTION</th>
<th>WRITE</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clear Display</td>
<td>Instruction</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Cursor Home</td>
<td>Instruction</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>N/A</td>
</tr>
<tr>
<td>Entry mode</td>
<td>Instruction</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>I/D</td>
</tr>
<tr>
<td>Display on/off</td>
<td>Instruction</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>D</td>
<td>C</td>
<td>B</td>
</tr>
<tr>
<td>Cursor &amp; display shift</td>
<td>Instruction</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>S/C</td>
<td>R/L</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Function set</td>
<td>Instruction</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>DL</td>
<td>N</td>
<td>F</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Set cursor address</td>
<td>Instruction</td>
<td>1</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>Write data to cursor location</td>
<td>Data</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
</tbody>
</table>
Example for HD 44780 Programming

<table>
<thead>
<tr>
<th>Instruction</th>
<th>RS</th>
<th>RW</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialisation</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8 Bit interface,</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3 x instruction</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2 lines, 5 x 7</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Display on</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Display clear</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Entry mode</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Write character to LCD RAM

<table>
<thead>
<tr>
<th></th>
<th>RS</th>
<th>RW</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>‘F’</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

‘ENABLE’ pulse for valid data required
Example for HD 44780 Programming (I)

' Initialisation of display

' all outputs 0
en = 0  ` Control
rw = 0
rs = 0  ` Instructions
daten = 0  ` Data

daten = 48  ' 0011 0000  ` 8-Bit IF
pulse en

daten = 48  ' 0011 0000
pulse en
daten = 48  ' 0011 0000
pulse en

pause 1

' Function set  0011 NF00  N: lines N=1:2Z, 0:1Z F: Characters 1:5*10Dots, 0:5*8D
daten = &B00111000  ' 2 lines
pulse en

pause 1
Example for HD 44780 Programming (II)

' Display on 0000 1100
daten = 12
pulse en

' Entry Mode set 0000 01 ID S
   ID:1 increment, ID:0 decrement S:1 Display shift
daten = 6  ' 0000 0110
pulse en

' Display clear 0000 0001
daten = 1
pulse en

' Character (4 starting at `F`) writing
rs = 1  ` Data
for BU = 0 to 4
   daten = 46 + BU
   pulse en
next BU
Example for HD 44780 Programming (III)

/**********************************************
 /* Funktion: disp_init */
 /* Initialisierung des HD 44780 */
/****************************************************************************/

void disp_init ( void )
{
    RS = 0; R_W = 0; E = 0;

    DB7 = 0; DB6 = 0;           /* Initialisierungswert 30H an Ports */
    DB5 = 1; DB4 = 1;
    DB3 = 0; DB2 = 0;
    DB1 = 0; DB0 = 0;
    E = 1; _nop_(); E = 0;     /* Schreiben in Display-Controller */
    wait( 500 );              /* warten > 4.1 ms */
    E = 1; _nop_(); E = 0;
    wait( 100 );              /* warten > 100us */
    E = 1; _nop_(); E = 0;
    wait( 50 );               /* warten > 100us */

    put_char( FUNCTION_SET );  /* warten > 4.1 ms */
    wait( 500 );              /* warten > 4.1 ms */
    put_char( DISPLAY_OFF );
    wait( 500 );              /* warten > 4.1 ms */
    put_char( CLS );           /* warten > 4.1 ms */
    wait( 500 );              /* warten > 4.1 ms */
    put_char( ENTRY );
    wait( 500 );              /* warten > 4.1 ms */
}
Example for HD 44780 Programming (IV)

```c
#include <stdio.h>

void sendchar2disp ( char s )
{
    if ( (s != '\0') && (s != '\n') )
    {
        RS = DA; /* An Display schreiben */
        split = s;
        DB7 = SPLIT_7; DB6 = SPLIT_6; DB5 = SPLIT_5; DB4 = SPLIT_4;
        DB3 = SPLIT_3; DB2 = SPLIT_2; DB1 = SPLIT_1; DB0 = SPLIT_0;
        E = 1; wait(100); E = 0; /* Einschreiben in Display-Controller */
        wait(100);
        RS = IN; /* RS = 0 */
    }

    if ( s == '\n' )
    {
        RS = IN; R_W = 0;
        split = LF_ADR; /* Uberabwert in bdata-Bereich legen */
        DB7 = SPLIT_7; DB6 = SPLIT_6; DB5 = SPLIT_5; DB4 = SPLIT_4;
        DB3 = SPLIT_3; DB2 = SPLIT_2; DB1 = SPLIT_1; DB0 = SPLIT_0;
        E = 1; wait(100); E = 0; /* Einschreiben in Display-Controller */
        wait(100);
    }
}
```
Software Character Controller

“hello”

(1) For each character to be displayed

‘h’ - ‘e’ = 3

(2) Get offset into descriptor array by subtracting ‘startChar’ (taken from the generated FONT_INFO structure) from the current letter

(3) Use the ‘offsetIntoBitmap’ member to get offset into the generated charBitmap array – in this case byte #12 - and pump that into your LCD

charBitmaps[] =
{
  // 024 'l' (2 pixels wide)
  0x08, 0x80, //
  0xFF, 0x80, //
  0x04, 0x00, //
  0x04, 0x00, //
  0xFC, 0x00, //
  0xF8, 0x00, //

  // 028 'o' (6 pixels wide)
  0x78, 0x00, //
  0xFC, 0x00, //
  0x84, 0x00, //
  0x84, 0x00, //
  0x84, 0x00, //
  0xFC, 0x00, //
  0x78, 0x00, //

  // 012 'h' (6 pixels wide)
  0xFF, 0x80, //
  0xFF, 0x80, //
  0x04, 0x00, //
  0x04, 0x00, //
  0xFC, 0x00, //
  0xF8, 0x00, //

  // 00 'e' (6 pixels wide)
  0x78, 0x00, //
  0xFC, 0x00, //
  0x94, 0x00, //
  0x94, 0x00, //
  0x9C, 0x00, //
  0x98, 0x00, //

  descriptors[] =
  {
    {6, 0}, // e
    {0, 0}, // f
    {0, 0}, // g
    {6, 12}, // h
    {0, 0}, // i
    {0, 0}, // j
    {0, 0}, // k
    {2, 24}, // l
    {0, 0}, // m
    {0, 0}, // n
    {6, 28}, // o
  };

Zusammenfassung Character-Displays

- Geeignet für viele Low Cost-Anwendungen oder wenn wenig Platz

- Praktisch ausschließlich Character-Controller

- HD44780 ist der meist verbreiteter Character-Controller

- Bei 4-Bit Interface-Mode auch von ‚kleinen‘ (wenig Pins) µCs ansteuerbar

- Viele Codebeispiele für eine Vielzahl von µCs

- Praktisch keine Prozessorlast

- Softcore-Character Controller vereinfachen System (auch Internationalisierung) aber „Echtzeit“-Anforderungen an µC
Case Study for Low Res System (I)

Task: Design temperature capturing & display device with warning on high temperatures

• Microcontroller : …
• Display : …
• Warning : …
• Power supply : …
Case Study for Low Res System (II)

**Task:** Design temperature capturing & display device with warning on high temperatures

<table>
<thead>
<tr>
<th></th>
<th>8-Seg LCD</th>
<th>Character LCD</th>
<th>Low Res Graphics LCD</th>
<th>8-Seg LED</th>
<th>8 Seg VFD</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Warning</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Power</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>consumption</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Price</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Advantage</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Issues</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Case Study for Low Res System (III)

Task: Design temperature capturing & display device with warning on high temperatures

System: …
Übersicht ‘Klein-Grafik’ Displays

- Ausführungen:
  - praktisch nur monochrom (d.h. schwarz – weiß (oder andere Farben), keine Graustufen)
  - Graustufen durch Dithern möglich
  - Auflösung typisch 64 x 32 … 240 x 128

- Technologien: LCD, OLED

- Ansteuerung meist mit T6963 (oder kompatibel)

- µC Interface ähnlich Character-Controller

- Controller haben meist Text- und Grafikmode
Low Resolution Display with Graphics Controller & μC

~ 16 lines flex cable (data & control & power)

8-Bit μC

Column driver

TCON

Row driver
Monochrome Graphics Controller & Drivers

- Column drivers (SEG)
- Row driver (COM)
- Graphics controller
- 64k SRAM
- Module connector
- b/w QVGA LCD
Monochrome Graphics LCD Module

**SP14N001-Z1**

<table>
<thead>
<tr>
<th>Product Category</th>
<th>Display Passive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manufacturer</td>
<td>KOE</td>
</tr>
<tr>
<td>Type</td>
<td>Graphic</td>
</tr>
<tr>
<td>Package</td>
<td>COB</td>
</tr>
<tr>
<td>Integra. Controller</td>
<td>T6963</td>
</tr>
<tr>
<td>Resolution</td>
<td>240 x 128 Pixel</td>
</tr>
<tr>
<td>Size</td>
<td>13.0 cm / 5.1 inch</td>
</tr>
<tr>
<td>Display Type</td>
<td>FSTN</td>
</tr>
<tr>
<td>Display Mode</td>
<td>negative</td>
</tr>
<tr>
<td>Visual Mode</td>
<td>transmissive</td>
</tr>
<tr>
<td>Operating Temp</td>
<td>-10 ... 60 °C</td>
</tr>
<tr>
<td>Outline Dimensions</td>
<td>159.4 x 101.0 x 11.0 mm</td>
</tr>
<tr>
<td>Viewing Area</td>
<td>123.0 x 68.0 mm</td>
</tr>
<tr>
<td>Backlight Color</td>
<td>White</td>
</tr>
<tr>
<td>Viewing Direction</td>
<td>6 o’clock</td>
</tr>
<tr>
<td>Supply Voltage</td>
<td>5 V</td>
</tr>
</tbody>
</table>

Download Datenblatt
- sp14n001-z1

Zubehör
demo KOE: SP14N001-Z1()
Monochrome Graphics LCD Module

5. ELECTRICAL CHARACTERISTICS

5.1 ELECTRICAL CHARACTERISTICS

<table>
<thead>
<tr>
<th>ITEM</th>
<th>SYMBOL</th>
<th>CONDITION</th>
<th>MIN.</th>
<th>TYP.</th>
<th>MAX.</th>
<th>UNIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power Supply Voltage For Logic</td>
<td>VDD-VSS</td>
<td>-</td>
<td>(4.75)</td>
<td>5.0</td>
<td>(5.25)</td>
<td>V</td>
</tr>
<tr>
<td>LC driver Circuit Power Supply Voltage</td>
<td>VEE-VSS</td>
<td>-</td>
<td>-15.5</td>
<td>-15.0</td>
<td>-14.5</td>
<td>V</td>
</tr>
<tr>
<td>Input Voltage</td>
<td>VI</td>
<td>H LEVEL</td>
<td>0.8VDD</td>
<td>-</td>
<td>VDD</td>
<td>V</td>
</tr>
<tr>
<td></td>
<td></td>
<td>L LEVEL</td>
<td>0</td>
<td>-</td>
<td>0.2VDD</td>
<td>V</td>
</tr>
<tr>
<td>Power Supply Current For Logic (Note 1)</td>
<td>IDD</td>
<td>VDD-VSS=5.0V</td>
<td>-</td>
<td>(11.7)</td>
<td>(14.0)</td>
<td>mA</td>
</tr>
<tr>
<td>Power Supply Current For LCD (Note 1)</td>
<td>IEE</td>
<td>VDD-VSS=5.0V</td>
<td>-</td>
<td>(2.5)</td>
<td>(4.0)</td>
<td>mA</td>
</tr>
<tr>
<td>Recommended LC Driving Voltage (Note 2)</td>
<td>VDD-V0</td>
<td>Ta= 0°C , ϕ = 0°</td>
<td>-</td>
<td>(16.9)</td>
<td>-</td>
<td>V</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ta=25°C , ϕ =0°</td>
<td>-</td>
<td>(15.8)</td>
<td>-</td>
<td>V</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ta=50°C , ϕ =0°</td>
<td>-</td>
<td>(15.2)</td>
<td>-</td>
<td>V</td>
</tr>
</tbody>
</table>
Monochrome Graphics LCD Module

8. INTERFACE TIMING

<table>
<thead>
<tr>
<th>ITEM</th>
<th>SYMBOL</th>
<th>MIN.</th>
<th>TYP.</th>
<th>MAX.</th>
<th>UNIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>C / D Setup Time</td>
<td>tCDS</td>
<td>100</td>
<td>-</td>
<td>-</td>
<td>ns</td>
</tr>
<tr>
<td>C / D Hold Time</td>
<td>tCHD</td>
<td>10</td>
<td>-</td>
<td>-</td>
<td>ns</td>
</tr>
<tr>
<td>CE, RD, WR Pulse Width</td>
<td>tCE, tRD, tWR</td>
<td>80</td>
<td>-</td>
<td>-</td>
<td>ns</td>
</tr>
<tr>
<td>Data Setup Time</td>
<td>tDS</td>
<td>80</td>
<td>-</td>
<td>-</td>
<td>ns</td>
</tr>
<tr>
<td>Data Hold Time</td>
<td>tDH</td>
<td>40</td>
<td>-</td>
<td>-</td>
<td>ns</td>
</tr>
<tr>
<td>Access Time</td>
<td>tACC</td>
<td>-</td>
<td>-</td>
<td>150</td>
<td>ns</td>
</tr>
<tr>
<td>Output Hold Time</td>
<td>tOH</td>
<td>10</td>
<td>-</td>
<td>50</td>
<td>ns</td>
</tr>
</tbody>
</table>

Diagram: Diagram showing the timing relationships between different signals (C/D, CE, RD, WR, D0~D7) with notations for timing parameters such as tCDS, tCDH, tCE, tRD, tWR, tDS.
Monochrome Graphics LCD Module

8.3 POWER SUPPLY FOR LCM (EXAMPLE)

V0 = Contrast voltage, optimum is T-dependant

See poti @ copy machine
Monochrome Graphics LCD Module

8.2 TIMING OF POWER SUPPLY AND INTERFACE SIGNAL

The missing pixels may occur when the LCM is driven beyond above power interface timing sequence.
# Monochrome Graphics LCD Module

**CN1**: Pitch 1.0mm 26pins connector  
Suitable connector: Molex: 52207-2690

<table>
<thead>
<tr>
<th>PIN No.</th>
<th>SYMBOL</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>VSS(0V)</td>
<td>Ground</td>
</tr>
<tr>
<td>2</td>
<td>VDD(+5V)</td>
<td>Power supply for logic</td>
</tr>
<tr>
<td>3</td>
<td>V0(Input)</td>
<td>Power supply for LCD drive</td>
</tr>
</tbody>
</table>
| 4       | C/D     | WR="L" : C/D="H" Command write  
          |         | C/D="L" Data write  
          |         | RD="L" : C/D="H" Status read  
          |         | C/D="L" Data read |
| 5       | WR      | Data write (Data write at “L”) |
| 6       | RD      | Data read (Read data at “L”) |
| 7       | DB0     | Data bus |
| 8       | DB1     | Data bus |
| 9       | DB2     | Data bus |
| 10      | DB3     | Data bus |
| 11      | DB4     | Data bus |
| 12      | DB5     | Data bus |
| 13      | DB6     | Data bus |
| 14      | DB7     | Data bus |
| 15      | CE      | Chip enable (CE must be “L”) |
| 16      | RET     | Reset    |
| 17      | VEE     | Power supply for LCD drive |
| 18      | D.OFF   | VDD/Display, GND/Display off |
| 19      | F/S     | Character font select: F/S="H" 6*8Font  
          |         | F/S="L" 8*8Font |
| 20      | P/N     | Display mode reverse. |
| 21      | NC      | No connection |

- **Power for graphics controller and LCD**
- **Control for graphics controller**
- **Data for graphics controller**
- **Power for LCD**
- **Control for graphics controller**
Monochrome Graphics LCD Module

- Matrix display: like matrix in maths
- Organized in rows and columns
- Origin (0/0) usually upper left corner

Rows

Columns

240x128 Dots Matrix

Starting dot for the starting address of display RAM. D0-D7 are 8 bits transmitted data, where D0 is LSB and D7 is MSB.
Low Res Graphics Controller T6963 (I)

FEATURES

- Display format (pin-selectable)
  Columns: 32, 40, 64, 80
  Lines: 2, 4, 6, 8, 10, 12, 14, 16, 20, 24, 28, 32

The combination of number of columns and number of lines must not cause the frequency to exceed 5.5 MHz. (See Fig. 2)

- Character font (pin-selectable)
  Horizontal dots: 5, 6, 7, 8
  Vertical dots: 8 (fixed)

It is necessary to set a character font in Graphic mode just as in Text mode. The oscillation frequency does not change with the font selection.

- Display duty: 1/16 to 1/128 up to 128 rows (lines)

- A 128-word character generator ROM (code 0101) T6963C-0101 is built in as standard.

Monochrome: only black and white ⇒ 1 byte = 8 pixel
Low Res Graphics Controller T6963 (II)
# Low Res Graphics Controller T6963 (II)

<table>
<thead>
<tr>
<th>PIN NAME</th>
<th>I/O</th>
<th>FUNCTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Pins for selection of LCD size</td>
</tr>
<tr>
<td></td>
<td>DUAL</td>
<td>H</td>
</tr>
<tr>
<td></td>
<td>MDS</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>MD1</td>
<td>H</td>
</tr>
<tr>
<td></td>
<td>MD0</td>
<td>H</td>
</tr>
<tr>
<td></td>
<td>LINES</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>V-DOTS</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 SCREEN</td>
</tr>
<tr>
<td></td>
<td>MD2</td>
<td>H</td>
</tr>
<tr>
<td></td>
<td>MD3</td>
<td>H</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Columns</td>
</tr>
<tr>
<td></td>
<td>FS0</td>
<td>H</td>
</tr>
<tr>
<td></td>
<td>FS1</td>
<td>H</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Font</td>
</tr>
<tr>
<td>D0 to D7</td>
<td>I/O</td>
<td>Data I/O pins between CPU and T6963C (D7 is MSB)</td>
</tr>
<tr>
<td>WR</td>
<td>Input</td>
<td>Data Write. Write data into T6963C when WR = L.</td>
</tr>
<tr>
<td>RD</td>
<td>Input</td>
<td>Data Read. Read data from T6963C when RD = L.</td>
</tr>
<tr>
<td>CE</td>
<td>Input</td>
<td>Chip Enable for T6963C. CE must be L when CPU communicates with T6963C.</td>
</tr>
<tr>
<td>R/W</td>
<td>Output</td>
<td>Read/Write signal for display memory</td>
</tr>
<tr>
<td>ED</td>
<td>Output</td>
<td>SDSEL = H : Data output for even columns in both upper and lower areas of LCD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SDSEL = L : Data output for columns in both upper and lower areas of LCD</td>
</tr>
<tr>
<td>HOD</td>
<td>Output</td>
<td>Data output for odd columns in upper area of LCD</td>
</tr>
<tr>
<td>CDATA</td>
<td>Output</td>
<td>Synchronous signal for row driver</td>
</tr>
<tr>
<td>HSCP</td>
<td>Output</td>
<td>Shift clock pulse for column driver of upper area of LCD</td>
</tr>
<tr>
<td>LP</td>
<td>Output</td>
<td>Latch pulse for column driver. Shift clock pulse for row driver</td>
</tr>
<tr>
<td>FR</td>
<td>Output</td>
<td>Frame signal</td>
</tr>
</tbody>
</table>
Low Res Graphics Controller T6963 (IV)

- **Data** block:
  - D0~D7
  - I/O, CE

- **μC**:
  - I/O, WR, RD, A0, A1~A7

- **Control** block:
  - D0~D7

- **DSTN: Dual Scan TN: 2x Column driver**

- **Column drivers**:
  - TC5565
  - T6A39
  - EIO1, EIO2

- **LCD display**:
  - 64x320 dots

- **Row driver**:
  - T6A40
Example for T6963 (I)

//*********************************************************************************************************
// Displayinitialisierung
// Hier werden die Grundeinstellungen des Displays vorgenommen.
//*********************************************************************************************************
// Übergabewerte:   keine
// Rückgabewerte:   keine
//*********************************************************************************************************

void init_Display(void)
{
    write_data2(TEXT_HOME_ADR); // Set Text Home Address
    write_cmd(SET_TEXT_HOME_ADR);
    write_data2(TEXT_AREA);
    write_cmd(SET_TEXT_AREA);
    write_data2(GRAPHIC_HOME_ADR); // Set Graphic Home Address
    write_cmd(SET_GRAPHIC_HOME_ADR);
    write_data2(GRAPHIC_AREA);     // Set Graphic Area
    write_cmd(SET_GRAPHIC_AREA);

    OR_MODE;                       // Set Or Mode
    BOTH_ON;                       // Set Display Mode
    clr_Display();                 
}
Example for T6963 (II)

//*********************************************************************************************************
// Adresspointer setzen
// Setzt den Adresspointer auf die Stelle im RAM an der als nächstes geschrieben werden soll
//*********************************************************************************************************
// Übergabewerte: adresse: unsigned int (Adresse im RAM)
// Rückgabewerte: keine
//*********************************************************************************************************
void set_ADP(unsigned int adresse) // Adress Pointer Setzen
{
    write_data2(adresse);
    write_cmd(SET_ADR_POINTER);
}

//*********************************************************************************************************
// Kommando schreiben
// schickt einen Befehl an den Displaycontroller
//*********************************************************************************************************
// Übergabewerte: command: unsigned char (Controllerbefehl)
// Rückgabewerte: keine
//*********************************************************************************************************
void write_cmd(unsigned char command) 
{
    status();
    cmd = command;
}
Example for T6963 (III)

//*********************************************************************************************************
// Daten schreiben
// schickt 1 Byte Daten an den Displaycontroller
//*********************************************************************************************************
// Übergabewerte:     daten: unsigned char  (1 Byte Daten)
// Rückgabewerte:      keine
//*********************************************************************************************************
void write_data(unsigned char daten)
{
    status();
    dat = daten;
}

Example for T6963 (IV)

void set_char(unsigned char Zeichen) {
    Zeichen -= 0x20; // ASCII to Controller
    if (Zeichen & 128) Zeichen = 0x03; // Zeichen nicht ASCII dann # ausgeben
    write_data(Zeichen);
    write_cmd(DATA_WRITE_INC);
}
Example for T6963 (V)

//*********************************************************************************************************
// Pixel setzen/rücksetzen
// setzt oder löscht ein Pixel
//*********************************************************************************************************

// Übergabewerte: x,y: unsigned char (X-Y-Koordinaten)
// set: unsigned char (0 für rücksetzen, 1 für setzen)
// Rückgabewerte: keine
//*********************************************************************************************************

void pixel(unsigned char x, unsigned char y, unsigned char set)
{
    set_ADP(xy_to_grRAM(x, y));
    if (set==0) //für Reset
    {
        write_cmd(0xF0 | (5-(x%GRAPHIC_LENGTH)) ); // Bit Reset
    }
    else
    {
        write_cmd(0xF8 | (5-(x%GRAPHIC_LENGTH)) ); // Bit Set
    }
}
**Advanced Graphics Controller**

**SOLOMON SSD 1780**
- 104 x 80 x RGB x 4 Bit
- Advanced graphics programming

Built in SW reduces µC load and saves SW development

\[
\begin{align*}
X_1, Y_1 & \\
fillcolor & \\
X_2, Y_2 & \\
\end{align*}
\]
# Case Study for Low Res System

**Task:** Design temperature capturing & display device with warning on high temperatures.

<table>
<thead>
<tr>
<th></th>
<th>Seg-8 LED</th>
<th>Seg-8 LCD</th>
<th>Character LCD</th>
<th>Low Res Graphics LCD</th>
<th>xxx with OLED or VFD</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Warning</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Power consumption</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Price</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Advantage</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Issues</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Zusammenfassung  ‘Klein-Grafik’ Displays

• Ansteuerung praktisch immer über Displaycontroller
  ⇒ somit nur geringe Prozessorlast und keine Echtzeit

• µC-Interface ähnlich Character-Controller

• Graphics Controller beinhaltet meist auch Character-Controller

• Displays nur monochrom mit Auflösung bis 240 x 128

• Ansteuerung meist mit T6963 (oder kompatibel)

• Kein Video, nur „kleine“ (wenige Pixel betreffend) Animationen
Overview Embedded Display Systems

• Introduction
• Low resolution displays
• High resolution graphics systems
• Interfaces
Übersicht ‘Grafik Systeme’

• Auflösung ≥ QVGA, meist farbig

• Graustufen 12 – 24 Bit

• Ansteuerung praktisch immer über Display – Graphics-Controller aber auch µP mit eingebautem Displaycontroller sowie FPGAs

• Problem: Nicht standardisierte Interfaces µC ↔ GC ↔ Display
  Dies betrifft Stecker, Signale, Datenformat und Timing

• Bis XGA TTL-Interface (hier) aber auch LVDS (s.u.)

• Da kein Character-Controller etc. muß jedes Pixel einzeln per Software gesetzt werden oder man verwendet OS oder Bibliothek

• Displays praktisch nur LCD (und einige AMOLEDs)

• To do: Interfacing, Power Sequencing, Software zur Bilddarstellung
High Resolution Displays ($\geq$ QVGA, Non-PC)

- **Graphics Displays ($\geq$ QVGA, color)**
  - High resolution
  - Some high end µC with built-in display support
  - Huge timing issues
  - Real time display driving to avoid failures
  - No support for pixel and graphics (use of libs or OS)

- **Number of pixels for QVGA**: $320 \times 240 = 76,800 > 16$ Bit $\rightarrow$ no 8-Bit µC!
  - Same for color $\approx 140$ KByte (2 Byte color [16 Bit] per pixel)!

- **Data rate for CQVGA**: $320 \times 240 \times 60$ Frames/s $\times$ 2 Byte (color) $\approx 10$ MHz
  - Not achievable with many 16-Bit µC
  - ‘Real time’ data output to display input (row and column data) requires huge effort

New 16 and 32 Bit µC with display output
AM LCD - Panel with Digital RGB - Input

- **EO Transfer Fct.**
- **Gamma Corr.**
- **Digital input signals**
  - R (Red)
  - G (Green)
  - B (Blue)
  - Sync
  - Controls
- **Power Supply**
- **Timing Controller (TCON)**
- **Row Driver Bank**
  - Driver 1
  - Driver 2
  - Driver 3
- **Column Driver Bank**
- **Backlight driver**

Details of LCD module see 'LCDs'
Improper power sequencing will damage display!
Embedded Systems Task Overview

• **Graphics features**
  - Lower resolution as PCs
  - Typically digital output
  - Less standardized as PCs
  - FPGA IP cores

• ’To Do’
  - Interfacing µC or µP to Graphics Controller
  - Interfacing Graphics Controller to Display
  - Software for Graphics IC and/or Operating System
Embedded Systems Task Description & Requirements

- **Interfacing \( \mu \text{C} \) or \( \mu \text{P} \) to Graphics Controller**
  - Transfer of real time data from “calculation” in \( \mu \text{C} \) to GC
  - High speed, high pin count interface
  - Pinning not standardized

- **Interfacing Graphics Controller to Display**
  - Transfer of real time data from GC (with RAM) to display TCON
  - High speed, high pin count interface
  - Pinning not standardized

- **Software for Graphics IC and/or Operating System**
  - Software for generating “content” to be displayed from application software
  - Solutions: Graphics libraries or operating system
Resolution ↔ Pixel Rate

Pixel frequency = Resolution x Frame frequency \{also clock\} \ (limit for parallel interf.)

Data rate = Pixel frequency x RGB x Color depth \ (limit for serial interfacing)
Embedded Systems Task Overview: \( \mu C - DC - Display \)

<table>
<thead>
<tr>
<th>Display ((\geq \text{QVGA}))</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graphics (Display) controller*</td>
</tr>
<tr>
<td>( \mu C ) (32 Bit)</td>
</tr>
</tbody>
</table>

* \( : \) Graphics controller with display RAM

2 interfaces!

Interfacing Graphics Controller to Display

- \( GC \leftrightarrow D \)

Interfacing \( \mu C \) or \( \mu P \) to Graphics Controller

- \( \mu C \leftrightarrow GC \)
- \( SW / OS \)

Software for Graphics IC and/or Operating System
Graphics Display Controller (FUJITSU)

- 2 high speed interfaces to handle: µC ↔ GC & GC ↔ DC
- Fundamental graphics support (font, line, …)
- High pin count, less standardized

Diagram showing the connectivity of the Carmine Graphics Processor with interfaces to µC, GC, and D. The processor has a bus matrix (266MHz), a memory controller (DDR 266bps), and interfaces for CPU I/F (PCI 66MHz), video capture, display controller, geometry engine, and rendering engine.
Display Controller for High End Embedded Applications

Typical features

- Graphics functions (2D, 3D): Point, line, triangle, polygon, BLT and pattern
- Output of analog RGB and digital RGB signals
- Up to 1024 x 768, 8 Bit/Pixel
- Overlaying of layers
- Video input

![Diagram of Display Controller](image)
# Interfacing Embedded Processors to Graphics Controller

## µC ↔ GC

### µC-specific configuration

![Complex!](image.png)

### Generic Interface

<table>
<thead>
<tr>
<th>Generic</th>
<th>Hitachi</th>
<th>MIPS/ISA</th>
<th>Motorola MC68K Bus 1</th>
<th>Motorola MC68K Bus 2</th>
<th>Motorola PowerPC</th>
<th>PC Card</th>
<th>Philips PR31500/PR31700</th>
<th>Toshiba TX3912</th>
</tr>
</thead>
<tbody>
<tr>
<td>A20</td>
<td>A20</td>
<td>LatchA20</td>
<td>A20</td>
<td>A20</td>
<td>A11</td>
<td>A20</td>
<td>ALE</td>
<td>ALE</td>
</tr>
<tr>
<td>A19</td>
<td>A19</td>
<td>SA19</td>
<td>A19</td>
<td>A19</td>
<td>A12</td>
<td>A19</td>
<td>/CARDREG</td>
<td>CARDREG*</td>
</tr>
<tr>
<td>A18</td>
<td>A18</td>
<td>SA18</td>
<td>A18</td>
<td>A18</td>
<td>A13</td>
<td>A18</td>
<td>/CARDIORD</td>
<td>CARDIORD*</td>
</tr>
<tr>
<td>A17</td>
<td>A17</td>
<td>SA17</td>
<td>A17</td>
<td>A17</td>
<td>A14</td>
<td>A17</td>
<td>/CARDIOWR</td>
<td>CARDIOWR*</td>
</tr>
<tr>
<td>[A12:1]</td>
<td>[A12:1]</td>
<td>[SA12:1]</td>
<td>[A12:1]</td>
<td>[A12:1]</td>
<td>[A19:30]</td>
<td>[A12:1]</td>
<td>[A12:1]</td>
<td>[A12:1]</td>
</tr>
<tr>
<td>A0</td>
<td>A0</td>
<td>SA0</td>
<td>LDS#</td>
<td>A0</td>
<td>A31</td>
<td>A0</td>
<td>A0</td>
<td>A0</td>
</tr>
<tr>
<td>WE1#</td>
<td>WE1#</td>
<td>SBHE#</td>
<td>UDS#</td>
<td>DS#</td>
<td>Bi</td>
<td>-CE2</td>
<td>/CARDxCSH</td>
<td>CARDxCSH*</td>
</tr>
</tbody>
</table>

### External Decode

<table>
<thead>
<tr>
<th>Connected to VDD</th>
<th>Connected to VDD</th>
<th>Connected to VDD</th>
</tr>
</thead>
<tbody>
<tr>
<td>BCLK</td>
<td>CKIO</td>
<td>CLK</td>
</tr>
<tr>
<td>Connected to VDD</td>
<td>AS#</td>
<td>CLK</td>
</tr>
<tr>
<td>RD1#</td>
<td>RD/WR#</td>
<td>R/W#</td>
</tr>
<tr>
<td>Connected to VDD</td>
<td>MEMR#</td>
<td>SiZ1</td>
</tr>
<tr>
<td>RD0#</td>
<td>MEMW#</td>
<td>SiZ0</td>
</tr>
<tr>
<td>WE0#</td>
<td>MEM#</td>
<td>SiZ0</td>
</tr>
<tr>
<td>WAIT#</td>
<td>MEM#</td>
<td>SiZ0</td>
</tr>
<tr>
<td>RESET#</td>
<td>MEM#</td>
<td>SiZ0</td>
</tr>
</tbody>
</table>

### TTL Interface

- AB20
- AB19
- AB18
- AB17
- AB[16:13]
- AB[12:1]
- AB0
- DB[15:8]
- DB[7:0]
- WE1#
- M/R#  
- CS#
- BUSCLK
- BS#
- RD/WR#
- RD#
- WE0#
- WAIT#
- RESET#
Embedded Systems Task Overview: \( \mu C + DC - Display \)

Display (\( \geq \) QVGA)

Graphics (Display) controller*

\( \mu C \) (32 Bit) with integrated DC or FPGA

NXP CORTEX with built-in DC

*: Graphics controller with display RAM
μC with Built-in Hi Res Display Graphics Controller

- Basic functions (RTC, timer, key port, UART, …)
- Advanced functions (SD IF, ADC, touch, camera IF, …)
- Serial interfaces (I²C, SPI, USB, …)
- Display controller
- Memory controller
- CPU incl. cache
- Display
- Video RAM
- SRAM
NXP ARM 9 with LCD Output

Color LCD Controller

The LH7A400’s LCD Controller is programmable to support up to $1,024 \times 768$, 16-bit color LCD panels. It interfaces directly to STN, color STN, TFT, AD-TFT, and HR-TFT panels.

Unlike other LCD controllers, the LH7A400’s LCD Controller incorporates the timing conversion logic from TFT to HR- and AD-TFT, allowing a direct interface to these panels and minimizing external chip count.

The Color LCD Controller features support for:

- Up to $1,024 \times 768$ Resolution
- 16-bit Video Bus
- STN, Color STN, AD-TFT, HR-TFT, TFT panels
- Single and Dual Scan STN panels
- Up to 15 Gray Shades
- Up to 64,000 Colors

OS strongly recommended
(simple printf and graphics functions)
NXP ARM 9 with LCD Output

Block Diagram
Example of Hi Res Embedded Display System

ARM 9 with built-in display controller
µP with Built-In Graphics Controller: TOSHIBA TMPA910

1. CPU Data
2. CPU Instruction
3. LCD Controller
4. LCD Accelerator
5. DMA Ctrl. 1
6. DMA Ctrl. 2
7. USB
## μP with Built-In Graphics Controller: TOSHIBA TMPA91

### Resolution vs. CPU load

<table>
<thead>
<tr>
<th>Resolution</th>
<th>Without LCDA</th>
<th>With LCDA</th>
<th>Occupation Ratio</th>
<th>Display Memory (Bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>QVGA</strong></td>
<td>OK</td>
<td>OK</td>
<td>4.32%</td>
<td>153600</td>
</tr>
<tr>
<td>320*240</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16bit/64k Color</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>WQVGA</strong></td>
<td>OK</td>
<td>OK</td>
<td>5.40%</td>
<td>192000</td>
</tr>
<tr>
<td>400*240</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16bit/64k Color</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>WVGA</strong></td>
<td>OK</td>
<td>OK</td>
<td>8.10%</td>
<td>288000</td>
</tr>
<tr>
<td>400*240</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24bit/16M Color</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>VGA</strong></td>
<td>OK</td>
<td>OK</td>
<td>25.92%</td>
<td>921600</td>
</tr>
<tr>
<td>640*480</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24bit/16M Color</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>WVGA</strong></td>
<td>OK</td>
<td>OK</td>
<td>32.40%</td>
<td>1152000</td>
</tr>
<tr>
<td>800*640</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24bit/16M Color</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>XGA</strong></td>
<td>OK</td>
<td>Not Recommended</td>
<td>66.36%</td>
<td>2359296</td>
</tr>
<tr>
<td>1024*768</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24bit/16M Color</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Condition: 32Bit SDRAM, LCD refresh rate: 60 Hz
**µP with Built-In Graphics Controller: TOSHIBA TMPA910**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Scaler function</strong></td>
<td>Scale up: Can scale up to the magnification of 256/n: (n = 1 to 255). Can scale up independently in horizontal/vertical directions. Filtering by Bi-Cubic method is possible in scaled up images.</td>
</tr>
<tr>
<td></td>
<td>Scale down: Can scale down to the magnification of 256/(n x m): (n = 1 to 255, m = 1 to 16). Can scale down independently in horizontal/vertical directions. Filtering by Bi-Cubic method is possible in scaled down images.</td>
</tr>
<tr>
<td><strong>Image rotation function</strong></td>
<td>90° / 180° / 270° / horizontal mirror reversal / vertical mirror reversal possible</td>
</tr>
<tr>
<td><strong>Image Blend function</strong></td>
<td>Function of superimposing two images (Picture in Picture)</td>
</tr>
<tr>
<td></td>
<td>Superimposing (α-Blend) possible adjusting the gray level of two images</td>
</tr>
<tr>
<td></td>
<td>Font Draw function for Font Data represented in binary (monochrome)</td>
</tr>
</tbody>
</table>

Only a few function available depending on display controller
TOSHIBA TMPA910 Software Support

Blend + Font mix function

Original Picture

Character = 1 bit
Monochrome will be converted up to 24 bit

Delicious!

Picture mix

Brightness control Process

Delicious!

Color changing

Delicious!
TOSHIBA TMPA910 Software Support
μP with Built-In Graphics Controller: TOSHIBA TMPA91

**Touch screen I/F**

- An interface for 4-terminal resistor network touch-screen is built in.
- Touch data via TSI control register and using an internal AD converter.
Digital (TTL) Interfacing

- **Signals**:
  - **CK**: Pixel clock
  - **Hsync**: Line clock
  - **Vsync**: Frame clock
  - **Data**: 12 - 24 bpp

- Many lines (length < 0.5 m)
  → display nearby controller
  long distance via Differential signalling (LVDS, DVI, USB; see dedicated paragraph)

- Panel specific timing
- High frequencies → EMI

3 x 6 = 18 bpp
Not standardized!
Digital (TTL) Interfacing

- Pins are labelled different for Graphics Controller and display input

- Line (wire) increase with color depth (1 line per color bit)

- Pixel frequency
  \[ \text{Pixel frequency} = \text{Resolution} \times \text{Frame frequency} \]

- Examples:
  - QVGA : 4.6 MHz
  - VGA : 18.4 MHz (maximum useful res.)
  - WXGA : 62.9 MHz (LVDS)

- Reduction of pixel frequency by doubling pixels (lower and upper) hence doubling lines (Dual Data interface)
Interfacing Data to Display Timing Controller
(clocks omitted)

<table>
<thead>
<tr>
<th>LCD Type</th>
<th>18-bit Sharp HR-TFT¹</th>
</tr>
</thead>
<tbody>
<tr>
<td>GC Output</td>
<td>D ↔ GC</td>
</tr>
</tbody>
</table>

### Monochrome Passive Panel

<table>
<thead>
<tr>
<th></th>
<th>4-bit</th>
<th>8-bit (format stripe)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LFRAME</td>
<td>LFRAME</td>
<td></td>
</tr>
<tr>
<td>LLINE</td>
<td>LLINE</td>
<td></td>
</tr>
<tr>
<td>LSHIFT</td>
<td>LSHIFT</td>
<td></td>
</tr>
</tbody>
</table>

### Color Passive Panel

<table>
<thead>
<tr>
<th></th>
<th>9-bit</th>
<th>12-bit</th>
<th>18-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDATA0</td>
<td>D0</td>
<td>D0(G3)²</td>
<td>R2</td>
</tr>
<tr>
<td>LDATA1</td>
<td>D1</td>
<td>D1(R3)²</td>
<td>R1</td>
</tr>
<tr>
<td>LDATA2</td>
<td>D2</td>
<td>D2(B2)²</td>
<td>R0</td>
</tr>
<tr>
<td>LDATA3</td>
<td>D3</td>
<td>D3(G2)²</td>
<td>G0</td>
</tr>
<tr>
<td>LDATA4</td>
<td>D4</td>
<td>D4(R2)²</td>
<td>G1</td>
</tr>
<tr>
<td>LDATA5</td>
<td>D5</td>
<td>D5(B1)²</td>
<td>G2</td>
</tr>
<tr>
<td>LDATA6</td>
<td>D6</td>
<td>D6(G1)²</td>
<td>B2</td>
</tr>
<tr>
<td>LDATA7</td>
<td>D7</td>
<td>D7(R1)²</td>
<td>B1</td>
</tr>
<tr>
<td>LDATA8</td>
<td>Drive 0</td>
<td>Drive 0</td>
<td>B0</td>
</tr>
<tr>
<td>LDATA9</td>
<td>Drive 0</td>
<td>Drive 0</td>
<td>B1</td>
</tr>
<tr>
<td>LDATA10</td>
<td>Drive 0</td>
<td>Drive 0</td>
<td>B2</td>
</tr>
<tr>
<td>LDATA11</td>
<td>Drive 0</td>
<td>Drive 0</td>
<td>B3</td>
</tr>
<tr>
<td>LDATA12</td>
<td>Drive 0</td>
<td>Drive 0</td>
<td>B4</td>
</tr>
<tr>
<td>LDATA13</td>
<td>Drive 0</td>
<td>Drive 0</td>
<td>B5</td>
</tr>
<tr>
<td>LDATA14</td>
<td>Drive 0</td>
<td>Drive 0</td>
<td>B6</td>
</tr>
<tr>
<td>LDATA15</td>
<td>Drive 0</td>
<td>Drive 0</td>
<td>B7</td>
</tr>
<tr>
<td>LDATA16</td>
<td>Drive 0</td>
<td>Drive 0</td>
<td>B8</td>
</tr>
<tr>
<td>LDATA17</td>
<td>Drive 0</td>
<td>Drive 0</td>
<td>B9</td>
</tr>
</tbody>
</table>

### Color TFT Panel

<table>
<thead>
<tr>
<th></th>
<th>9-bit</th>
<th>12-bit</th>
<th>18-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDEN</td>
<td>MOD</td>
<td>LDEN</td>
<td>Drive 0</td>
</tr>
<tr>
<td>LDATA0</td>
<td>Drive 0</td>
<td>R2</td>
<td>R5</td>
</tr>
<tr>
<td>LDATA1</td>
<td>Drive 0</td>
<td>R1</td>
<td>R4</td>
</tr>
<tr>
<td>LDATA2</td>
<td>Drive 0</td>
<td>R0</td>
<td>R3</td>
</tr>
<tr>
<td>LDATA3</td>
<td>Drive 0</td>
<td>G2</td>
<td>G5</td>
</tr>
<tr>
<td>LDATA4</td>
<td>Drive 0</td>
<td>G1</td>
<td>G4</td>
</tr>
<tr>
<td>LDATA5</td>
<td>Drive 0</td>
<td>G0</td>
<td>G3</td>
</tr>
<tr>
<td>LDATA6</td>
<td>Drive 0</td>
<td>B2</td>
<td>B5</td>
</tr>
<tr>
<td>LDATA7</td>
<td>Drive 0</td>
<td>B1</td>
<td>B4</td>
</tr>
<tr>
<td>LDATA8</td>
<td>Drive 0</td>
<td>B0</td>
<td>B3</td>
</tr>
<tr>
<td>LDATA9</td>
<td>Drive 0</td>
<td>B1</td>
<td>B2</td>
</tr>
<tr>
<td>LDATA10</td>
<td>Drive 0</td>
<td>B2</td>
<td>B1</td>
</tr>
<tr>
<td>LDATA11</td>
<td>Drive 0</td>
<td>B3</td>
<td>B0</td>
</tr>
</tbody>
</table>

Note: SPS, LP, CLK, TTL indicate different types of signals or interface configurations.
**Digital Input Timing Diagram**

Example for XGA

- $V_{\text{sync}}$
- $V_{\text{porch}}$
- $H_{\text{porch}}$
- $H_{\text{sync}}$
- $\text{Data}$

16.7 ms

21.5 µs

# of data per line: 1024 x RGB x bpp

(XGA: 768 rows)

Clocks and other controls not shown
Adaptor Interface Board for 3.5” AMLCD

- Standard connector to customer board incl. single voltage supply
- Power supply for LCD module
- Power sequencing

GC ↔ D
Block Diagram of 3.5” AMLCD

TTL input

- Data / Clock
- Timing Signals
- Power Supply
- LED Control Signal
- Touch Panel Signals

I/F(CN1) → Timing Controller → Gate Driver

Power Circuit → TFT-LCD

LED Driving Circuit → LED B/L

Source Driver

GC ↔ D
## Adaptation of LCD to Embedded System

<table>
<thead>
<tr>
<th>Pin of panel</th>
<th>Pin of graphics controller</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin</td>
<td>Meaning</td>
</tr>
<tr>
<td>1</td>
<td>GND</td>
</tr>
<tr>
<td>2</td>
<td>CLK</td>
</tr>
<tr>
<td>3</td>
<td>$H_{SYNC}$</td>
</tr>
<tr>
<td>4</td>
<td>$V_{SYNC}$</td>
</tr>
<tr>
<td>5</td>
<td>GND</td>
</tr>
<tr>
<td>6</td>
<td>R0</td>
</tr>
<tr>
<td>7</td>
<td>R1</td>
</tr>
<tr>
<td>8</td>
<td>R2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pin</th>
<th>Pin of graphics controller</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>B5</td>
</tr>
<tr>
<td>26</td>
<td>GND</td>
</tr>
<tr>
<td>27</td>
<td>DE</td>
</tr>
<tr>
<td>28</td>
<td>$V_{CC}$ (*)</td>
</tr>
<tr>
<td>29</td>
<td>$V_{CC}$ (*)</td>
</tr>
<tr>
<td>30</td>
<td>DPSH (*)</td>
</tr>
<tr>
<td>31</td>
<td>DPSV (*)</td>
</tr>
<tr>
<td>32</td>
<td>PNS (*)</td>
</tr>
<tr>
<td>33</td>
<td>GND</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pin</th>
<th>Pin of graphics controller</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>B5</td>
</tr>
<tr>
<td>24</td>
<td>GND</td>
</tr>
<tr>
<td>21</td>
<td>M-Signal</td>
</tr>
<tr>
<td>25</td>
<td>$V_{LCD}$ (3.3/5V)</td>
</tr>
<tr>
<td>25</td>
<td>$V_{LCD}$ (3.3/5V)</td>
</tr>
<tr>
<td>25/28</td>
<td>$V_{CC}$ /GND</td>
</tr>
<tr>
<td>25/28</td>
<td>$V_{CC}$ /GND</td>
</tr>
<tr>
<td>25/28</td>
<td>$V_{CC}$ /GND</td>
</tr>
<tr>
<td>28</td>
<td>GND</td>
</tr>
</tbody>
</table>
### Interface Timing of 3.5” AMLCD

<table>
<thead>
<tr>
<th></th>
<th>MIN.</th>
<th>TYP.</th>
<th>MAX.</th>
<th>UNIT</th>
<th>SYMBOL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vertical Total</td>
<td>-</td>
<td>327</td>
<td>-</td>
<td>Line</td>
<td>T0</td>
</tr>
<tr>
<td>Vertical Sync Width</td>
<td>1</td>
<td>1</td>
<td>-</td>
<td>Line</td>
<td>T1</td>
</tr>
<tr>
<td>Vertical Sync Start</td>
<td>-</td>
<td>322</td>
<td>-</td>
<td>Line</td>
<td>T2</td>
</tr>
<tr>
<td>Vertical Sync End</td>
<td>-</td>
<td>323</td>
<td>-</td>
<td>Line</td>
<td>T3</td>
</tr>
<tr>
<td>Vertical Blank Time</td>
<td>5</td>
<td>7</td>
<td>-</td>
<td>Line</td>
<td>T4</td>
</tr>
<tr>
<td>Vertical Display End</td>
<td>-</td>
<td>320</td>
<td>-</td>
<td>Line</td>
<td>T5</td>
</tr>
<tr>
<td>Horizontal Total</td>
<td>258</td>
<td>273</td>
<td>509</td>
<td>Pixel Clock</td>
<td>T6</td>
</tr>
<tr>
<td>Horizontal Sync Width</td>
<td>4</td>
<td>5</td>
<td>10</td>
<td>Pixel Clock</td>
<td>T7</td>
</tr>
<tr>
<td>Horizontal Sync Start</td>
<td>246</td>
<td>251</td>
<td>307</td>
<td>Pixel Clock</td>
<td>T8</td>
</tr>
<tr>
<td>Horizontal Sync End</td>
<td>250</td>
<td>256</td>
<td>317</td>
<td>Pixel Clock</td>
<td>T9</td>
</tr>
<tr>
<td>Horizontal Blank Time</td>
<td>18</td>
<td>33</td>
<td>269</td>
<td>Pixel Clock</td>
<td>T10</td>
</tr>
<tr>
<td>Horizontal Display End</td>
<td>-</td>
<td>240</td>
<td>-</td>
<td>Pixel Clock</td>
<td>T11</td>
</tr>
</tbody>
</table>

- Note: Vertical Total should be set to odd.

- Portrait orientation

- There is practically no timing standard! → Individual adaptation needed
Interface Timing of 3.5” AMLCD

Fig. (a) Horizontal timing chart
Interface Timing of 3.5” AMLCD

Fig.(b) Vertical timing chart
## Display Color vs. Input Signal

<table>
<thead>
<tr>
<th>COLOR</th>
<th>INPUT DATA</th>
<th>R Data</th>
<th>G Data</th>
<th>R Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLACK</td>
<td>R↑ R R R R R R</td>
<td>R R R R R R</td>
<td>R R R R R R</td>
<td>R R R R R R</td>
</tr>
<tr>
<td>RED(255)</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
</tr>
<tr>
<td>GREEN(255)</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
</tr>
<tr>
<td>BLUE(255)</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
</tr>
<tr>
<td>CYAN</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
</tr>
<tr>
<td>MAGENTA</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
</tr>
<tr>
<td>YELLOW</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
</tr>
<tr>
<td>WHITE</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
</tr>
<tr>
<td>RED(0)</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
</tr>
<tr>
<td>RED(1)</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
<td>R R R R R R R</td>
</tr>
</tbody>
</table>
Display Pixel Co-ordinates

Row (line), column (data)

Example for XGA

1,1 1,2

2,1 2,2

768,1 768,1024

1,1024 2,1024
# NXP ARM 9 with LCD Output

## ARM

<table>
<thead>
<tr>
<th>BGA PIN</th>
<th>LFBGA PIN</th>
<th>LCD DATA SIGNAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>K1</td>
<td>K2</td>
<td>LCDVD17</td>
</tr>
<tr>
<td>J5</td>
<td>K1</td>
<td>LCDVD16</td>
</tr>
<tr>
<td>R10</td>
<td>T13</td>
<td>LCDVD15</td>
</tr>
<tr>
<td>P10</td>
<td>R12</td>
<td>LCDVD14</td>
</tr>
<tr>
<td>T9</td>
<td>R11</td>
<td>LCDVD13</td>
</tr>
<tr>
<td>R9</td>
<td>T12</td>
<td>LCDVD12</td>
</tr>
<tr>
<td>N11</td>
<td>T11</td>
<td>LCDVD11</td>
</tr>
<tr>
<td>K8</td>
<td>P10</td>
<td>LCDVD10</td>
</tr>
<tr>
<td>L11</td>
<td>K10</td>
<td>LCDVD9</td>
</tr>
<tr>
<td>M11</td>
<td>M9</td>
<td>LCDVD8</td>
</tr>
<tr>
<td>M10</td>
<td>R10</td>
<td>LCDVD7</td>
</tr>
<tr>
<td>M9</td>
<td>T10</td>
<td>LCDVD6</td>
</tr>
<tr>
<td>N10</td>
<td>K9</td>
<td>LCDVD5</td>
</tr>
<tr>
<td>L10</td>
<td>T9</td>
<td>LCDVD4</td>
</tr>
<tr>
<td>N8</td>
<td>T8</td>
<td>LCDVD3</td>
</tr>
<tr>
<td>T7</td>
<td>R8</td>
<td>LCDVD2</td>
</tr>
<tr>
<td>R7</td>
<td>P8</td>
<td>LCDVD1</td>
</tr>
<tr>
<td>P7</td>
<td>M8</td>
<td>LCDVD0</td>
</tr>
</tbody>
</table>

## Display

<table>
<thead>
<tr>
<th>LCD DATA SIGNAL</th>
<th>STN MONO 4-BIT</th>
<th>STN MONO 8-BIT</th>
<th>STN COLOR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SINGLE PANEL</td>
<td>DUAL PANEL</td>
<td>SINGLE PANEL</td>
</tr>
<tr>
<td>LCDVD17</td>
<td></td>
<td></td>
<td>MLSTN7</td>
</tr>
<tr>
<td>LCDVD16</td>
<td></td>
<td></td>
<td>MLSTN6</td>
</tr>
<tr>
<td>LCDVD15</td>
<td>MLSTN5</td>
<td>CLSTN5</td>
<td>INTENSITY</td>
</tr>
<tr>
<td>LCDVD14</td>
<td>MLSTN4</td>
<td>CLSTN4</td>
<td>BLUE3</td>
</tr>
<tr>
<td>LCDVD13</td>
<td>MLSTN3</td>
<td>CLSTN3</td>
<td>BLUE2</td>
</tr>
<tr>
<td>LCDVD12</td>
<td>MLSTN2</td>
<td>CLSTN2</td>
<td>BLUE1</td>
</tr>
<tr>
<td>LCDVD11</td>
<td>MLSTN1</td>
<td>CLSTN1</td>
<td>GREEN3</td>
</tr>
<tr>
<td>LCDVD10</td>
<td>MLSTN0</td>
<td>CLSTN0</td>
<td>GREEN2</td>
</tr>
<tr>
<td>LCDVD9</td>
<td></td>
<td></td>
<td>GREEN1</td>
</tr>
<tr>
<td>LCDVD8</td>
<td>MLSTN3</td>
<td>MUSTN7</td>
<td>MUSTN7</td>
</tr>
<tr>
<td>LCDVD7</td>
<td></td>
<td>MUSTN6</td>
<td>MUSTN6</td>
</tr>
<tr>
<td>LCDVD6</td>
<td>MLSTN2</td>
<td>MUSTN5</td>
<td>MUSTN5</td>
</tr>
<tr>
<td>LCDVD5</td>
<td>MLSTN1</td>
<td>MUSTN4</td>
<td>MUSTN4</td>
</tr>
<tr>
<td>LCDVD4</td>
<td></td>
<td>MUSTN3</td>
<td>MUSTN3</td>
</tr>
<tr>
<td>LCDVD3</td>
<td>MUSTN0</td>
<td>MUSTN2</td>
<td>MUSTN2</td>
</tr>
<tr>
<td>LCDVD2</td>
<td></td>
<td>MUSTN1</td>
<td>MUSTN1</td>
</tr>
<tr>
<td>LCDVD1</td>
<td>MUSTN0</td>
<td>MUSTN0</td>
<td>MUSTN0</td>
</tr>
<tr>
<td>LCDVD0</td>
<td></td>
<td></td>
<td>CUSTN0</td>
</tr>
</tbody>
</table>
NXP ARM 9 with LCD Output

Vertical Timing (rows)
NXP ARM 9 with LCD Output

Horizontal Timing (columns)

Grey level & color data
Adaptation of Embedded OS (here WIN CE) (I)

The registry key for the driver is:

```
HKLM\Drivers\Display\SMIVGX
```

The following configurations are predefined in the kernel:

<table>
<thead>
<tr>
<th>Mode</th>
<th>Name</th>
<th>XxY</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>TFT, 60 Hz, 16Bpp</td>
<td>640x480</td>
<td>Active</td>
</tr>
<tr>
<td>1</td>
<td>TFT, 16Bpp</td>
<td>800x600</td>
<td>Active</td>
</tr>
<tr>
<td>2</td>
<td>TFT, 16Bpp</td>
<td>1024x768</td>
<td>Active</td>
</tr>
<tr>
<td>3 *</td>
<td>TFT, 16Bpp</td>
<td>1280x1024</td>
<td>Active</td>
</tr>
<tr>
<td>4</td>
<td>TFT, 16Bpp</td>
<td>320x240</td>
<td>Active</td>
</tr>
</tbody>
</table>

Use of standard types reduce risk but only certain panels defined!

Here: LQ104V1LG61
## Adaptation of Embedded OS (here WIN CE) (II)

<table>
<thead>
<tr>
<th>Key</th>
<th>Data-Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>sz:</td>
<td>QVGA standard display</td>
</tr>
<tr>
<td>Type</td>
<td>Dword:</td>
<td>6</td>
</tr>
<tr>
<td>Config</td>
<td>Dword:</td>
<td>0x00300000</td>
</tr>
<tr>
<td>Columns</td>
<td>Dword:</td>
<td>320</td>
</tr>
<tr>
<td>BLW</td>
<td>Dword:</td>
<td>52</td>
</tr>
<tr>
<td>HSW</td>
<td>Dword:</td>
<td>2</td>
</tr>
<tr>
<td>ELW</td>
<td>Dword:</td>
<td>4</td>
</tr>
<tr>
<td>PPL</td>
<td>Dword:</td>
<td>320</td>
</tr>
<tr>
<td>Rows</td>
<td>Dword:</td>
<td>240</td>
</tr>
<tr>
<td>BFW</td>
<td>Dword:</td>
<td>5</td>
</tr>
<tr>
<td>VSW</td>
<td>Dword:</td>
<td>3</td>
</tr>
<tr>
<td>LPP</td>
<td>Dword:</td>
<td>240</td>
</tr>
<tr>
<td>EFW</td>
<td>Dword:</td>
<td>5</td>
</tr>
<tr>
<td>Width</td>
<td>Dword:</td>
<td>115</td>
</tr>
<tr>
<td>Height</td>
<td>Dword:</td>
<td>86</td>
</tr>
<tr>
<td>Bpp</td>
<td>Dword:</td>
<td>16</td>
</tr>
<tr>
<td>Voltage</td>
<td>Dword:</td>
<td>33</td>
</tr>
<tr>
<td>ContrastEnable</td>
<td>Dword:</td>
<td>9</td>
</tr>
<tr>
<td>ContrastValue</td>
<td>Dword:</td>
<td>0</td>
</tr>
<tr>
<td>LCDClk</td>
<td>Dword:</td>
<td>6300000</td>
</tr>
<tr>
<td>Msignal</td>
<td>Dword:</td>
<td>0</td>
</tr>
<tr>
<td>EnableCursor</td>
<td>Dword:</td>
<td>1</td>
</tr>
<tr>
<td>PhysFrameBuffDraw</td>
<td>Dword:</td>
<td>0</td>
</tr>
</tbody>
</table>

### Examples

#### Beginning-of-line-wait:

Specifies the number of pixel clock periods to add to the beginning of a line transmission before the first set of pixels is sent to the display.

#### End-of-frame line clock wait count:

Specifies the number of line clock periods to add to the end of each frame.

If not predefined!

Values from panel spec!
Hi Res Display Software Approaches

<table>
<thead>
<tr>
<th>µC with Display Controller</th>
<th>GUI-Software</th>
<th>OS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application software</td>
<td>Application software</td>
<td>Application software</td>
</tr>
<tr>
<td>Various device drivers</td>
<td>Various device drivers</td>
<td>Operating system</td>
</tr>
<tr>
<td>Display controller</td>
<td>Graphics library</td>
<td></td>
</tr>
<tr>
<td>Hardware</td>
<td>Display driver</td>
<td>Hardware</td>
</tr>
</tbody>
</table>

SW / OS
Hi Res Display Software Approaches

- Dedicated to display graphics, characters and special features
- Three approaches:
  - Hardware (implemented in display controller as “software”)
  - Software (as special GUI software or operating system)

<table>
<thead>
<tr>
<th>Hi Res Software</th>
<th>μC with Display Controller</th>
<th>GUI-Software</th>
<th>OS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Merits</strong></td>
<td>Easy to use, GUI SW available</td>
<td>Fast, professional features</td>
<td>Easy to use</td>
</tr>
<tr>
<td><strong>Shortcomings</strong></td>
<td>μC with DC: Huge software effort, limited functionality</td>
<td>Proprietary</td>
<td>Real-time issues, computing power &amp; storage, cost</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>FUJITSU, TOSHIBA</td>
<td>SEGGER</td>
<td>WINDOWS, LINUX</td>
</tr>
</tbody>
</table>
Programming of Graphics Controller

C - Code

```
// SED 1374 - SHARP CQVGA STN 4bpp
...
// Draw 100x100 rectangle starting (0,0), color : red
for (y = 0; y < 100; y++)
{
    pMem = (LPBYTE)MEM_OFFSET + (y * 320 * BitsPerPixel / 8) + 0;
    for (x = 0; x < 100; x+=2)
    {
        pMem++;  // draws 2 pixel
    }
}
...
```

Each pixel of a character and for graphics must be set individually

ANSI-C Library

```
Graphics Application Program

C language interface

Initializing command

Drawing command

Driver command

MB86290 Series Graphics Controller

Display List

MB86290 Series Graphics Controller

→ Use subroutines, libraries or operating system (OS)
Display Controller Layer Technology

... for simplified implementation of various data including video input

- **C (console) Layer**
  - 8 or 16 Bits/Pixel
  - Used for Continuous Switch Display, Etc.

- **W (window) Layer**
  - 8 or 16 Bits/Pixel (Y:U:V = 2:1:1)
  - Used for Display of External Input Image (can also be used as normal layer)

- **M (middle) Layer**
  - 8 or 16 Bits/Pixel
  - Additional Base Screen
  - Split into Separate Segments

- **B (base) Layer**
  - 8 or 16 Bits/Pixel
  - Base Display Screen
  - Split into Separate Segments
Example of Display SW C – Library: SEGGER

Graphics software and GUI emWin Features

- Any CPU, any LCD, any LCD controller
- ANSI “C” no C++ required
- Simulation included, develop prior availability of target hardware
- Multiple layer / multi display support
- Small footprint, no C++ required
- Customizable Widgets
- Touch screen support
- Child windows
- Alpha blending
- Support for transparent windows
- JPEG support
- Font converter available
- VNC Server available, ...

More infos & free trial version: www.segger.com
Example of Display SW C – Library: SEGGER

Graphics software and GUI emWin Features
Example of Display SW C – Library: SEGGER

```c
#include "BSP.h"
#include "GUI.h"
#include "Rtos.h"
#include <string.h>

/*
 * main()
 */
void MainTask(void);
void MainTask(void) {
    BSP_GUI_Init();
    GUI_DispString("Hello Till!");
    while(1) {
        OS_Delay(500);
    }
}

/***** End of File ********************/
```
Example of Display SW C – Library: SEGGER

```c
43  /**************************************************************************/
44  *
45  */
46  void MainTask(void);
47  void MainTask(void) {
48     BSP_GUI_Init();
49     GUI_SetColor(GUI_RED);
50     GUI_SetBkColor(GUI_WHITE);
51     GUI_Clear();
52     GUI_DispString("Hello World!");
53     while(1) {
54         OS_Delay(500);
55     }
56 }
57
58  /**************************************************************************/
```

advanced
Summary of High Resolution

• Three approaches
  - Graphics controller
  - GUI software
  - Operating System

• Microcontroller with built-in display controller has typically > 100 pins and limited features support

• GUI software is proprietary but easy to handle

• Operating system (not mentioned here) is most easy for “ready to go” systems like embedded PCs
Integration of Graphics Controller into FPGA

Enables LCD module with only gate and source drivers
Integration of Graphics Controller into FPGA

Analogue input + 32-bit µC + TCON
Integration of Graphics Controller into FPGA

Customizable Display Controller IP-Core for Embedded Systems

Thesis by C. Bayer & S. Reiser @ Display Lab
Integration of Graphics Controller into FPGA

- **μP Interface**
  - Data
  - Instruction

- **Control Block**
  - ASCII Data
  - Graphic Data
  - Character Data

- **Character Generator**

- **Page Generator**

- **Display**
  - Display Timing
  - Logic
    - Text
    - Graphics

- **Memory Controller Interface**
  - Text Page
  - Graphic Page

- **Character Memory**
Integration of Graphics Controller into FPGA

Logic → Display Timing Interface

aclr
clk (100 MHz)
rdreq
empty
q

LOAD
CP
D0 ~ D3
FRAME
LOAD
FRAME
240x\(T\)

52.1 \(\mu\)s \(\leq T \leq 59.5 \mu\)s
Integration of Graphics Controller into FPGA

Source: XILINX LOGICBRICKS
Zusammenfassung  ‘Grafik Systeme

- 3 Hauptaufgaben zu lösen:
  - Microcontroller ↔ Graphics Controller
  - Graphics Controller ↔ Display
  - Software zur Bilddarstellung

- Interface bis VGA: TTL,
höhere Auflösung bzw. abgesetztes Display: Serielles IF (s.u.)

- Mehr und mehr 32-Bit Prozessoren sind auch mit eingebauten
  Displaycontroller erhältlich (seit 2010: auch INTEL ATOM)

- FPGAs: IP Cores für Prozessor, Graphics Controller und Interface
  von zahlreichen Anbietern erhältlich

- Probleme: keine oder nur wenige Standards beim Interface
Advanced Mobile Phone Colour LCD Graphics Controller

- One chip solution with gate (row, scan) and source (column, data) drivers
- Resolution: 176 x RGB x 220
- Up to 3 x 6 Bit colour
- Build-in RAM
- Power save mode: partial display, 8 colour
- Multiple µC interface

Not recommended because of CE!
Overview Embedded Display Systems

- Introduction
- Low resolution displays
- Graphics systems
- Interfaces
Übersicht ‘Interfaces’

• TTL-Interface ist in Auflösung (bis VGA) und Kabellänge limitiert und hat 30 – 60 Leitungen (‘großes’ Kabel) (siehe vorangegangenes Kapitel)

• Ausweg: Serielle Interfaces mit 2-6 Twisted Pairs (hier)

• Vorteile seriell: längere Leitungen, weniger EMI-sensitiv

• Nachteile: zusätzlicher Serializer und Deserialízer (falls nicht im GC und Display vorhanden)

• Meist verwendet (analoges IF praktisch nur noch PC-Welt und SDTV):
  - LVDS in der Industrie, automotive
  - DVI: PC, E-Signage, TV, …
  - HDMI: DVI mit DRM für HDTV
  - DISPLAYPORT: zunehmende Verbreitung, auch Embedded-Bereich
  - spezielle IFs wie APIX
### Data Date vs. Display Resolution

#### Bandwidth [Gbps]

- **DP v1.2**
  - 17.28 Gbps
- **DP v1.1a**
  - 8.64 Gbps
- **HDMI/340MHz**
  - 8.16 Gbps
- **DL-DVI**
  - 7.92 Gbps
- **HDMI/225MHz**
  - 5.4 Gbps
- **SL-DVI**
  - 3.96 Gbps

#### Display Format Resolution

- **WSXGA**
  - 1680x1050
- **1080p**
  - 1920x1080
- **WQXGA**
  - 2560x1600
- **4k x 2k**
  - 4096x2160

#### Net data rate (without protocol overhead)

- 60Hz 24bpp
- 60Hz 30bpp
- 60Hz 36bpp
- 120Hz 24bpp
- 120Hz 30bpp
- 120Hz 36bpp
- 120Hz 30bpp
- 120Hz 24bpp
- 120Hz 18kHz

#### Data Date vs. Display Resolution

<table>
<thead>
<tr>
<th>Data Date</th>
<th>Display Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>60Hz</td>
<td>WSXGA 1680x1050</td>
</tr>
<tr>
<td>60Hz</td>
<td>1080p 1920x1080</td>
</tr>
<tr>
<td>60Hz</td>
<td>WQXGA 2560x1600</td>
</tr>
<tr>
<td>60Hz</td>
<td>4k x 2k 4096x2160</td>
</tr>
<tr>
<td>120Hz</td>
<td>WSXGA 1680x1050</td>
</tr>
<tr>
<td>120Hz</td>
<td>1080p 1920x1080</td>
</tr>
<tr>
<td>120Hz</td>
<td>WQXGA 2560x1600</td>
</tr>
<tr>
<td>120Hz</td>
<td>4k x 2k 4096x2160</td>
</tr>
<tr>
<td>120Hz</td>
<td>WSXGA 1680x1050</td>
</tr>
<tr>
<td>120Hz</td>
<td>1080p 1920x1080</td>
</tr>
<tr>
<td>120Hz</td>
<td>WQXGA 2560x1600</td>
</tr>
<tr>
<td>120Hz</td>
<td>4k x 2k 4096x2160</td>
</tr>
<tr>
<td>120Hz</td>
<td>WSXGA 1680x1050</td>
</tr>
<tr>
<td>120Hz</td>
<td>1080p 1920x1080</td>
</tr>
<tr>
<td>120Hz</td>
<td>WQXGA 2560x1600</td>
</tr>
<tr>
<td>120Hz</td>
<td>4k x 2k 4096x2160</td>
</tr>
<tr>
<td>120Hz</td>
<td>WSXGA 1680x1050</td>
</tr>
<tr>
<td>120Hz</td>
<td>1080p 1920x1080</td>
</tr>
<tr>
<td>120Hz</td>
<td>WQXGA 2560x1600</td>
</tr>
<tr>
<td>120Hz</td>
<td>4k x 2k 4096x2160</td>
</tr>
<tr>
<td>120Hz</td>
<td>WSXGA 1680x1050</td>
</tr>
<tr>
<td>120Hz</td>
<td>1080p 1920x1080</td>
</tr>
<tr>
<td>120Hz</td>
<td>WQXGA 2560x1600</td>
</tr>
<tr>
<td>120Hz</td>
<td>4k x 2k 4096x2160</td>
</tr>
<tr>
<td>120Hz</td>
<td>WSXGA 1680x1050</td>
</tr>
<tr>
<td>120Hz</td>
<td>1080p 1920x1080</td>
</tr>
<tr>
<td>120Hz</td>
<td>WQXGA 2560x1600</td>
</tr>
<tr>
<td>120Hz</td>
<td>4k x 2k 4096x2160</td>
</tr>
<tr>
<td>120Hz</td>
<td>WSXGA 1680x1050</td>
</tr>
<tr>
<td>120Hz</td>
<td>1080p 1920x1080</td>
</tr>
<tr>
<td>120Hz</td>
<td>WQXGA 2560x1600</td>
</tr>
<tr>
<td>120Hz</td>
<td>4k x 2k 4096x2160</td>
</tr>
</tbody>
</table>
Parallel Digital TTL vs. LVDS

**Parallel Digital TTL RGB**
- TX $\rightarrow$ RX
- $V = 3.3V$
- $I = 40 - 120mA$
- 18 lines

**LVDS (see below)**
- TX $\rightarrow$ RX
- $V = 0.45V$
- $I = 3 - 9.0mA$
- 3 pairs (6 lines)
- 1 pair (2 lines)
Serial Interfacing: LVDS, TMDS (DVI)

- Reduction of lines by using twisted pair (serial data)
- Small and thin cable (a must for laptops)
- ‘L’ and ‘H’ are defined as voltage difference instead of voltage level
  → low sensitivity to EMI
- LVDS is also used for digital image processing cameras
- Avoid static voltage generation by large numbers of ’L’ or ’H’ → Bit inversion
- Tx: transmitter ; Rx: receiver
Low Voltage Differential Signalling (RS-644)

Transition Minimised Differential Signalling

- Multiplexing of 7 signals (7 Bit) on 1 line
- Both use voltage difference (~ 0.1 V)
- TMDS has current loop (disadvantage for EMI)
- LVDS by NATIONAL SEMICONDUCTOR and TEXAS INSTRUMENTS mainly used in industry, laptops and digital industrial cameras
- TMDS by SILICON IMAGE, also named as PANEL-LINK and Digital Visual Interface (DVI)

- PC standard: Digital Visual (Video) Interface
- TV standard: High Definition Multimedia Interface (DVI with DRM)
Digital Low Voltage Differential Signaling

Standard for ≥ VGA panels, differential signaling like LVDS, also some solution using USB, Ethernet, …
Digital Low Voltage Differential Signaling

Standard for VGA+ panels, differential signaling like USB, Ethernet, …

Transmitter

Receiver
**Block Diagram LVDS & TMDS**

<table>
<thead>
<tr>
<th>Transmitter</th>
<th>Receiver</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data 24/48</strong></td>
<td><strong>Data 24/48</strong></td>
</tr>
<tr>
<td><strong>DE</strong></td>
<td><strong>DE</strong></td>
</tr>
<tr>
<td><strong>Controls (5)</strong></td>
<td><strong>Controls (5)</strong></td>
</tr>
<tr>
<td><strong>Clock</strong></td>
<td><strong>Clock</strong></td>
</tr>
</tbody>
</table>

**Transmitter:**
- **Channel 0**: RED
- **Channel 1**: GREEN
- **Channel 2**: BLUE
- **Jitter Filter**
- **PLL**
- **CLK**

**Voltage Swing Control**

**3 Data Pairs**

**Receiver:**
- **RED**
- **GREEN**
- **BLUE**
- **CLK**
- **PLL**

**Reflection Minimizer**

**Sync Detect**

**Output Panel Interface Logic**

**SVGA ↔ D**

**LVDS**: 4 data pairs
**TMDS**: 3 data pairs

*(instead of 30 ... 50 for TTL)*
Principle of Serialising for LVDS

LVTTTL  |  DS90CR217  |  LVDS

1 2 3 7
8 9 10
14
15 16 17
21

1 2 3 4 5 6 7
14 13 12 11 10 9 8
21 20 19 18 17 16 15

7:1 Mux
Input Latch
PLL

GC ↔ D
Principle of Serialising for LVDS

Transmission:
- TxCLK out/
- RxCLK in
- TxOUT3/
- RxIN3
- TxOUT2/
- RxIN2
- TxOUT1/
- RxIN1
- TxOUT 0/
- RxIN0

Previous Cycle
- RxIN0
- RxIN1
- RxIN2
- RxIN3

Next Cycle
- TxIN0
- TxIN1
- TxIN2
- TxIN3

Serialization:
- RES
- B7
- B6
- G7
- G6
- R7
- R6
- B5
- B4
- B3
- B2
- G5
- G4
- G3
- G2
- G1
- R5
- R4
- R3
- R2
- R1
- R0

Next Cycle
- GC ↔ D
Bandwidth of PANEL-Link (TMDS)

- Single channel bandwidth [Gbs]
  - Copper barrier
  - Selective refresh
  - One TMDS link (3 channels)
  - Two links (6 channels)

**Refresh rates**
- 60 Hz LCD
- 5% Blanking
- 60 Hz CRT
- GTF Blanking
- 75 Hz CRT
- GTF Blanking
- 85 Hz CRT
- GTF Blanking

**Legend:**
- VGA (640x480)
- SXGA (1280x1024)
- QXGA (2048x1536)
- SVGA (800x600)
- UXGA (1600x1200)
- XGA (1024x768)
- HDTV (1920x1080)
DVI Connector

**Digital Visual Interface**

- Only digital

![DVI-D Receptacle Connector](image)

- Digital & analog

![DVI-I Receptacle Connector](image)

---

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
<th>Pin</th>
<th>Signal</th>
<th>Pin</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TMDS Data 2-</td>
<td>11</td>
<td>TMDS Data 1/3 Shield</td>
<td>21</td>
<td>TMDS Data 5+</td>
</tr>
<tr>
<td>2</td>
<td>TMDS Data2+</td>
<td>12</td>
<td>TMDS Data 3-</td>
<td>22</td>
<td>TMDS Clock Shield</td>
</tr>
<tr>
<td>3</td>
<td>TMDS Data 2/4 Shield</td>
<td>13</td>
<td>TMDS Data 3+</td>
<td>23</td>
<td>TMDS Clock+</td>
</tr>
<tr>
<td>4</td>
<td>TMDS Data 4-</td>
<td>14</td>
<td>+5 V Power</td>
<td>24</td>
<td>TMDS Clock-</td>
</tr>
<tr>
<td>5</td>
<td>TMDS Data 4+</td>
<td>15</td>
<td>Ground (+5 V, Analog H/V Sync)</td>
<td>C1</td>
<td>Analog Red Video Out</td>
</tr>
<tr>
<td>6</td>
<td>DDC Clock</td>
<td>16</td>
<td>Hot Plug Detect</td>
<td>C2</td>
<td>Analog Green Video Out</td>
</tr>
<tr>
<td>7</td>
<td>DDC Data</td>
<td>17</td>
<td>TMDS Data 0-</td>
<td>C3</td>
<td>Analog Blue Video Out</td>
</tr>
<tr>
<td>8</td>
<td>Analog Vertical Sync</td>
<td>18</td>
<td>TMDS Data 0+</td>
<td>C4</td>
<td>Analog Horizontal Sync</td>
</tr>
<tr>
<td>9</td>
<td>TMDS Data 1-</td>
<td>19</td>
<td>TMDS Data 0/5 Shield</td>
<td>C5</td>
<td>Analog Common Ground Return (R,</td>
</tr>
<tr>
<td>10</td>
<td>TMDS Data 1+</td>
<td>20</td>
<td>TMDS Data 5-</td>
<td></td>
<td>G, B Video Out)</td>
</tr>
</tbody>
</table>
Single Twisted Pair Serializer Deserializer Interface

- TTL (graphics IC out) → serial → TTL (display panel in)
- Single twisted pair for differential signalling
- Up to XGA support (65 MHz clock = 1.56 Gbit/sec)
- Some implementations with USB, audio, …
Single Twisted Pair Serializer Deserializer Interface

„FPD Link II“ (National Semiconductor)
Single Twisted Pair Serializer Deserializer Interface

„FPD Link II“ (National Semiconductor)

- FPD Link

- FPD Link II
Serializer Deserializer Interface

- Cable have to be adapted for Bit-Error-Rate free transmission

640 Mbps

No Pre-Emphasis 100% Pre-Emphasis
Automotive Multimedia System
Comparison of Major Automotive Bus Systems

<table>
<thead>
<tr>
<th>Application</th>
<th>LIN: Low-level communication systems</th>
<th>CAN: Soft real-time systems</th>
<th>FlexRay: Hard real-time systems (X-by-wire)</th>
<th>MOST: Multimedia, telemetrics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control</td>
<td>Single-master</td>
<td>Multi-master</td>
<td>Multi-master</td>
<td>Timing-master</td>
</tr>
<tr>
<td>Bus Access</td>
<td>Polling</td>
<td>CSMA/CA</td>
<td>TDMA/FTDMA</td>
<td>TDM/CSMA</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>19.6 kBit/s</td>
<td>500 kBit/s</td>
<td>10 Mbit/s</td>
<td>24.8 mbit/s</td>
</tr>
<tr>
<td>Data Bytes per Frame</td>
<td>0 to 8</td>
<td>0 to 8</td>
<td>0 to 254</td>
<td>0 to 60</td>
</tr>
<tr>
<td>Redundant Channel</td>
<td>Not supported</td>
<td>Not supported</td>
<td>Two channels</td>
<td>Not supported</td>
</tr>
<tr>
<td>Physical Layer</td>
<td>Electrical (single wire)</td>
<td>Electrical (twisted pair)</td>
<td>Optical, electrical</td>
<td>Mainly optical</td>
</tr>
</tbody>
</table>

**Example:** Data rate = $800 \times 480 \times 60 \text{ Hz} \times 3 \times 8 \text{ bit} \times 1.1 \approx 550 \text{ Mbit/s}$

⇒ Compression or high speed bus needed
## Comparison of Major Automotive Bus Systems

<table>
<thead>
<tr>
<th></th>
<th>Ethernet</th>
<th>MOST®</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Applications</strong></td>
<td>VolP, STB, PCs &amp; servers, home, office &amp; factory networks, Industrial Control...</td>
<td>Automotive only</td>
</tr>
<tr>
<td><strong>Standards</strong></td>
<td>Open (IEEE), field proven, independent compliance &amp; interop test houses</td>
<td>'Proprietary'</td>
</tr>
<tr>
<td><strong>Media</strong></td>
<td>Copper and Fiber (inc. POF)</td>
<td>POF</td>
</tr>
<tr>
<td><strong>Topology</strong></td>
<td>Star, Ring or hybrid</td>
<td>Ring</td>
</tr>
<tr>
<td><strong>Speed</strong></td>
<td>10Mbps to 10Gbps &amp; Interoperable</td>
<td>25Mbps to 150Mbps but no interoperability</td>
</tr>
<tr>
<td><strong>Bandwidth</strong></td>
<td>Full bandwidth available per node ie. 100Mbps, 1Gbps</td>
<td>Shared between devices on ring eg. MOST-150 ring of 5 nodes = average 30Mbps per node</td>
</tr>
<tr>
<td><strong>Volume production</strong></td>
<td>Billions ports operating in the field to date. Hundreds of Million chips per year</td>
<td>A few Million chips per year</td>
</tr>
<tr>
<td><strong>Suppliers</strong></td>
<td>Multiple</td>
<td>Single (SMSC)</td>
</tr>
<tr>
<td><strong>Software / Equipment Costs</strong></td>
<td>Low - TCIP/IP common, tools &amp; s/w freely available and often open source</td>
<td>High - few experts in MOST s/w, tools not common and hence expensive</td>
</tr>
<tr>
<td><strong>Silicon Costs</strong></td>
<td>Low</td>
<td>Higher</td>
</tr>
</tbody>
</table>

**Source:** Micrel
Single Twisted Pair Automotive Pixel Link (APIX)

- ‘Cheap’ line from head unit to display @ high data rate
- Separation of display and controller saves cost
- APIX by INOVA, Munich
- Thesis e.g. by D. Lebherz
Single Twisted Pair Serializer Deserializer Interface

„APIX“ (Inova Semiconductor, Automotive Pixel Link)

Video 18 bit @ 42 MHz
24 bit @ 32 MHz

and

I2C or CAN

or

RS232 (Tx / Rx)

or

Dig.Audio SPDIF / I2S

APIX TX

VIDEO DATA
850 MBit/s

SIDEBAND DATA
2 x 9 MBit/s

APIX RX

Video 18 bit @ 42 MHz
24 bit @ 32 MHz

and

I2C or CAN

or

RS232 (Tx / Rx)

or

Dig.Audio SPDIF / I2S

Full Duplex Control Data Channels
with guaranteed bandwidth
fully pixel clock independent
High Definition Multimedia Interface (HDMI)

- Evolved from DVI / Panel Link (Display centric)
- Focused on Consumer Electronics (Set Top Box, DVD Player, Game Console, PC, HDTV, Projectors, Cameras, Cell Phones, …)
- Interface includes:
  TMDS Channels (3) for Video / Audio / Control plus Clock
  - unique 8b/10b scheme,
    3 data channels & 1 clock (3D+C)
  - DDC (Display Data Channel) for Configuration & HDCP
  - CEC (Consumer Electronic Control) for device control
  - HPD (Hot Plug Detect)
  - HDCP – High Definition Copy Protection
High Definition Multimedia Interface (HDMI)
High Definition Multimedia Interface (HDMI)

- HDTV resolution: \(1,920 \times 1,080 = 2 \text{ Mpixel}\)
- Pixel frequency or pixel clock: \(1,920 \times 1,080p \times 60 \times 1.2 \rightarrow 150 \text{ MHz}\)
  (1.2 is the blanking overhead (20%), 60 Hz frame frequency)

### HDMI standard

<table>
<thead>
<tr>
<th>1.2</th>
<th>P Clock</th>
<th>Line Speed</th>
<th>Total Throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td>165 MHz</td>
<td>x10 bits</td>
<td>1.65 Gbps</td>
<td>4.95 Gbps</td>
</tr>
<tr>
<td>225 MHz</td>
<td>x10 bits</td>
<td>2.25 Gbps</td>
<td>6.75 Gbps</td>
</tr>
<tr>
<td>1.3</td>
<td>340 MHz</td>
<td>x10 bits</td>
<td>10.2 Gbps</td>
</tr>
</tbody>
</table>

- Deeper Color
- Higher Refresh
- Higher Resolutions
High Definition Multimedia Interface (HDMI)

Serial IFs must be adjusted (emphasized) to cable used:

- **Before Equalization**
  - 5m 28AWG HDMI Cable
  - 2.25 Gbps
  - $1V \text{VINp-p}$

- **After Equalization**
  - Less than 0.2UI Jitter

- **Before Equalization**
  - 20m 28AWG HDMI Cable
  - 2.25 Gbps
  - $1V \text{VINp-p}$

- **After Equalization**
  - Less than 0.2UI Jitter
Display Port (DP)

• Digital audio/video interface standard developed by Video Electronics Standard Association (VESA)

• Up to WQXGA+ resolution support

• Focused on both internal (notebook) and external interfaces (monitors), embedded systems, …

• Interface:
  - 1, 2 or 4 pairs in Main Link (CML signaling, current mode logic)
  - AUX channel for two way transfer contains DDC, device control, …
  - Hot Plug Detect
  - Audio support
  - 20-pin USB sized connector (optional latching connector)
  - Mini Display Port (since 2009)
Display Port (DP)

AUX channel can also be used for touch screens, …
**Display Port (DP) Performance**

- 1 monitor up to 4096 x 2304 resolution (4K x 2K @60Hz, 24 bit color)

- Using Multistream Technology (MST):
  - 2 monitors at 2560 x 1600 resolution (WQXGA @60Hz, 24 bit color)
  - 4 monitors at 1920 x 1200 resolution
Display Port & Other Interfaces
Connectors and Receptacles

Display Port Plugs

Cable with DP plug

Slot bracket of a graphics card with 1x DVI and 2x DP receptacle

Mini DP plug

Slot bracket of a graphics card with 2x DVI and 1x mDP receptacle
**Embedded Display Port (eDP)**

Embedded DisplayPort (eDP) will replace the aging LVDS panel interface standard to increase performance, enhance integration, and reduce power.

- Up to 4096 x 2304 resolution
  (4K x 2K @60Hz, 24 bit color)
- Can service many separate functions of highly integrated panel over one IF

---

![Diagram of eDP Interface](image-url)
Embedded Display Port (eDP)

Diagram showing the connection between a Notebook PC Motherboard or Video Subsystem, eDP Interface, Display Panel, and related components such as Video/Graphics Processing Unit (GPU), Display EDID Memory, TCON, and LED Backlight.
## Display Port (DP)

<table>
<thead>
<tr>
<th>Specification</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>No. of high-speed differential pairs</strong></td>
<td>1, 2, or 4 pairs (No clock pairs)</td>
</tr>
<tr>
<td>For 1680x1050 @18bpp</td>
<td>1 pair</td>
</tr>
<tr>
<td>For 1600x1200 @30bpp</td>
<td>2 pairs</td>
</tr>
<tr>
<td>For 2048x1536 @36bpp</td>
<td>4 pairs</td>
</tr>
<tr>
<td><strong>Bit rate, per pair</strong></td>
<td>2.7Gbits/sec, fixed rate (1.62Gbps option available)</td>
</tr>
<tr>
<td><strong>Total raw capacity per 4-differential pairs</strong></td>
<td>10.8Gbits/sec</td>
</tr>
<tr>
<td><strong>AC-coupled for process migration</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Audio support</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Aux. channels</strong></td>
<td>1Mbps AUX CH, 500us max. latency</td>
</tr>
<tr>
<td><strong>Channel Coding</strong></td>
<td>ANSI8B/10B (open)</td>
</tr>
<tr>
<td><strong>Content protection</strong></td>
<td>Philips’s DPCP optional</td>
</tr>
<tr>
<td><strong>Protocol</strong></td>
<td>Micro-Packet-based; extensible in the future to add features.</td>
</tr>
</tbody>
</table>
## Comparison

<table>
<thead>
<tr>
<th></th>
<th>DISPLAYPORT 1.1A</th>
<th>HDMI 1.3</th>
<th>DVI</th>
<th>VGA</th>
<th>LVDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Digital or analog</td>
<td>Digital</td>
<td>Digital</td>
<td>Digital</td>
<td>Analog</td>
<td>Digital</td>
</tr>
<tr>
<td>Data pairs or lanes</td>
<td>One, two, or four</td>
<td>Three (must use all three)</td>
<td>Three (must use all three)</td>
<td>Three (RGB)</td>
<td>Eight (for dual channel)</td>
</tr>
<tr>
<td>Clock pins</td>
<td>None</td>
<td>Two (one pair)</td>
<td>Two (one pair)</td>
<td>Two: horizontal synchronous and vertical synchronous</td>
<td>Four (two pairs for dual channel)</td>
</tr>
<tr>
<td>Clock architecture</td>
<td>Embedded clock</td>
<td>Separate clock</td>
<td>Separate clock</td>
<td>Separate clock</td>
<td>Separate clock</td>
</tr>
<tr>
<td>Clock speed</td>
<td>162 or 270 MHz</td>
<td>340 MHz maximum</td>
<td>165 MHz maximum</td>
<td>DAC-dependent</td>
<td>25 to 135 MHz</td>
</tr>
<tr>
<td>Bandwidth/pair or lane</td>
<td>1.8 or 2.7 Gbps</td>
<td>3.4 Gbps maximum</td>
<td>1.65 Gbps maximum</td>
<td>DAC-dependent</td>
<td>175 to 945 Mbps</td>
</tr>
<tr>
<td>Total bandwidth</td>
<td>1.6 to 10.8 Gbps, depending on number of lanes/data pairs</td>
<td>10 Gbps</td>
<td>4.95 Gbps</td>
<td>DAC-dependent</td>
<td>1.74 to 7.56 Gbps</td>
</tr>
<tr>
<td>Carries audio</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Bidirectional auxiliary channel</td>
<td>Yes, 1 Mbps</td>
<td>Yes, DDC</td>
<td>Yes, DDC</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Protocol</td>
<td>Packet-based (8B10B)</td>
<td>Serial-data stream (TMDS)</td>
<td>Serial-data stream (TMDS)</td>
<td>Analog</td>
<td>Sequential-data stream</td>
</tr>
<tr>
<td>DRM support</td>
<td>DPCP, based on HDCP 1.3</td>
<td>HDCP 1.3</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>External-bus standard</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Internal-bus standard</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Summary

• Display interfaces and driving have many aspects
• Individual software is necessary for many systems
• Operating systems lower individual SW effort significantly
• Low resolution (... QVGA) is possible with parallel TTL IF
• High resolution (> VGA) use mostly serial IF like LVDS (industrial)
• Non-PC systems are more individual
• Many modern industrial systems base on IPCs which lowers R&D effort significantly. Second source etc. are other benefits
• IPCs, Computer on Module (COM) and µC-kits as ‘BUY’ component become more and more widespread shifting away from hardware development (mainly SW has to be done)
Questions

• Why are Hi Res panels so difficult to integrate into μC systems?

• LVDS is used for Embedded System panels, DVI for PC-Systems
  New trend: DISPLAYPORT – search web for benefits and issues

• What are the benefits of high end display controllers?

• What are the benefits and limits of single twisted pair interfaces?
**Summary “Driving”**

- **Pixel frequency**
  \[ \text{Pixel frequency} = \text{Resolution} \times \text{frame frequency} \]

- The higher the resolution, the more complex is display driving.

- If no video required, displays with built-in RAM reduce \( \mu \)C load.

- Segmented displays are driven mostly by \( \mu \)C with built-in DC

- Character and graphics modules are easy to interface with low software effort

- High resolution displays is “easy” with PC-like hard- and software

- „Own“ solutions might be complex, FPGA is “easy“.

\[ \mu \text{C} \leftrightarrow \text{GC} \]

\[ \text{GC} \leftrightarrow \text{D} \]

\[ \text{SW} / \text{OS} \]