Contents

1 Overview 2
2 Features 2
3 Installation with Arduino 3
4 Operation 4
  4.1 ANSI escape codes 5
  4.2 High-resolution modes 6
5 Hardware 9
6 Specifications 10
  6.1 DC characteristics 10
  6.2 AC characteristics 10
7 Support information 10

List of Figures

1 sample output at 80x25 4
2 Counter example output 5
3 mode 1 (128x48) 7
4 mode 2 (96x64) 8
5 Arduino Uno 9
1 Overview

TermDriver listens on a serial line and emulates a text terminal, producing output on a standard VGA connector. It gives embedded microcontrollers a real console.

2 Features

- serial input at 115200 bps
- connects to the Arduino serial output pin, no libraries required
- supports standard ANSI terminal control codes
- VGA 16-color output at 1024x768
- standard 80x25 and high-resolution 128x48 modes
- rotated-screen 96x64 mode
- screen-saver under CPU control
- input signals 3.3V and 5V compatible
3 Installation with Arduino

1. Disconnect power from the Arduino
2. Attach the TermDriver to the Arduino
3. Connect the VGA plug to the TermDriver and turn on the monitor
4. Apply power to the Arduino. You should see a blank screen with a blinking cursor at top-left
5. Load a sketch on the Arduino that prints text at 115200 baud, like the one below

```cpp
void setup()
{
    Serial.begin(115200);
}

void loop()
{
    Serial.println("Hello world");
}
```
4 Operation

TermDriver monitors the serial line at 115200 baud, and draws any text on the VGA. There's nothing to set up or load. For example this Arduino sketch

```c
void setup() {
    Serial.begin(115200);
}

int counter;
void loop() {
    Serial.print("Counter is ");
    Serial.println(counter++);
}
```

Or this code in plain C

```c
for (;;) {
    printf("Counter is %d
", counter++);
}
```
Gives this output on the VGA:

Figure 2: Counter example output

4.1 ANSI escape codes

The following standard CSI codes are supported:

<table>
<thead>
<tr>
<th>Code</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESC [ n A</td>
<td>Cursor up</td>
</tr>
<tr>
<td>ESC [ n B</td>
<td>Cursor down</td>
</tr>
<tr>
<td>ESC [ n C</td>
<td>Cursor forward</td>
</tr>
<tr>
<td>ESC [ n D</td>
<td>Cursor back</td>
</tr>
<tr>
<td>ESC [ r; c H</td>
<td>Cursor position</td>
</tr>
<tr>
<td>ESC [ n J</td>
<td>Erase display</td>
</tr>
<tr>
<td>ESC [ n m</td>
<td>Select graphic rendition</td>
</tr>
<tr>
<td>ESC [ s</td>
<td>Save cursor position</td>
</tr>
<tr>
<td>ESC [ u</td>
<td>Restore cursor position</td>
</tr>
</tbody>
</table>

In addition the following sequences are specific to TermDriver:

<table>
<thead>
<tr>
<th>Code</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESC [ n h</td>
<td>Set display mode. 0 is 80x25, 1 is 128x48, 2 is 96x64 (rotated)</td>
</tr>
<tr>
<td>ESC [ n S</td>
<td>Screen-saver. 0 stops video output, 1 restarts video output</td>
</tr>
</tbody>
</table>

For example this C program displays all available foreground and background...
colors.

```c
#include <stdio.h>

int main()
{
    int attr, fg, bg;

    printf("TermDriver listens on a serial line and emulates a "
           "terminal, generating output on a standard VGA connector."
           "It gives embedded microcontrollers a real console."
           "\n\n");
    for (attr = 0; attr <= 1; attr++) {
        for (fg = 30; fg <= 37; fg++) {
            for (bg = 40; bg <= 47; bg++) {
                printf("\e[1;%d;%d;%dm%d;%d;%d\e[m",
                        attr, bg, fg, attr, bg, fg);
            }
            printf("\n");
        }
    }
    return 0;
}
```

### 4.2 High-resolution modes

In addition to standard 80x25 text mode, TermDriver supports a higher density 128x48 mode, and a portrait orientation 96x64 mode. Both are very readable because they match TermDriver’s native 1024x768 @ 60 Hz VGA output.
Figure 3: mode 1 (128x48)
Figure 4: mode 2 (96x64)
5 Hardware

Figure 5: Arduino Uno

TermDriver connects directly to any Arduino or Arduino-compatible. It requires four connections:

- GND
- 5V
- RESET
- TX

To use another MCU, make the above four connections. Note that RESET is active-low. The serial protocol on TX is 115200 bps, 8 bits, no parity, 1 stop bit. This is frequently described as 115200-8N1. All signaling is 3.3V, but 5V tolerant.
6 Specifications

6.1 DC characteristics

<table>
<thead>
<tr>
<th></th>
<th>min</th>
<th>typ</th>
<th>max</th>
<th>units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supply voltage</td>
<td>4.0</td>
<td>5.0</td>
<td>9.0</td>
<td>V</td>
</tr>
<tr>
<td>Supply current</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>operation</td>
<td>25</td>
<td></td>
<td></td>
<td>mA</td>
</tr>
<tr>
<td>screen saver</td>
<td>10</td>
<td></td>
<td></td>
<td>mA</td>
</tr>
<tr>
<td>RESET,RX low voltage</td>
<td>0.6</td>
<td></td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>RESET,RX high voltage</td>
<td>2.7</td>
<td>5.8</td>
<td></td>
<td>V</td>
</tr>
</tbody>
</table>

6.2 AC characteristics

<table>
<thead>
<tr>
<th></th>
<th>min</th>
<th>typ</th>
<th>max</th>
<th>units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serial input line speed</td>
<td>115200</td>
<td></td>
<td></td>
<td>bps</td>
</tr>
<tr>
<td>VGA</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>resolution</td>
<td>1024x768</td>
<td></td>
<td></td>
<td>pixels</td>
</tr>
<tr>
<td>vertical sync</td>
<td>60.004</td>
<td></td>
<td></td>
<td>Hz</td>
</tr>
<tr>
<td>horizontal sync</td>
<td>48.363</td>
<td></td>
<td></td>
<td>kHz</td>
</tr>
<tr>
<td>pixel clock</td>
<td>65.000</td>
<td></td>
<td></td>
<td>MHz</td>
</tr>
<tr>
<td>Cursor blink rate</td>
<td>1.875</td>
<td></td>
<td></td>
<td>Hz</td>
</tr>
<tr>
<td>Startup time</td>
<td>300</td>
<td></td>
<td></td>
<td>ms</td>
</tr>
</tbody>
</table>

7 Support information

Technical and product support is available at support@excamera.com