Simple VGA/Video Adapter Using Atmel AVR

WITH COMMONLY AVAILABLE MICROCONTROLLERS LIKE THE MEGA8, MEGA16 AND SIMILAR, AND WITH A MINIMUM OF EXTERNAL COMPONENTS, I WANTED A DESIGN THAT WOULD BE CAPABLE OF DISPLAYING AT LEAST 15X15 CHARACTERS ON A VGA MONITOR USING STANDARD VGA FREQUENCIES.

By: Ibragimov Maksim

Background of the project:

Several months ago I tried to connect a microcontroller system to a VGA monitor to output data in the form of text. I was surprised to find little on this subject on the internet, to assist me in achieving this goal. Certainly nothing simple a beginner could find useful.

There are examples out there that utilize standards such as PC-104 or complex FGPA implementations found at www.opencores.org. Other solutions include graphic controllers from Fujitsu or even one local Russian person who was offering for sale a project for $5000 on ACEX. These are fine but are little help to most hobbyists, etc. out there who wish to display text on a VGA or similar screen.

What I desired was a “quick and dirty” solution that did not cost too much.

Initial calculations showed that the the AVR 8-bit microcontroller from Atmel, with its 16 Mhz clock speed providing approximately 16 MIPS was a good candidate for further research. Also note that newer AVRs such as the Mega48, Mega88 and Mega168 will officially support clock rates up to 20 MHz. Therefore I concluded that with a clock of 16 MHz I could achieve something in the order of 8 MHz speed of data being transferred out of a port. I also chose the AVR as I had already built up quite a body of experience with it and so I began work of the project.

After approximately two to three months of research, I present you the fruits of my labor!

The goal of the project:

The goal of the project is simple enough to enumerate: With commonly available microcontrollers like the Mega8, Mega16 and similar, and with a minimum of external components I wanted a design that would be capable of displaying at least 15x15 characters on a VGA monitor using standard VGA frequencies. The data itself is to be received by the microcontroller via its USART port. All using a 16 MHz clock for the AVR.

The given problem is solved successfully, and the initial goal has been achieved. The project has expanded to include Monochomatic Video signal (PAL/SECAM). In my test set up a mere jumper determines whether the output is VGA or Composite Video.

Characteristics of the project:

VGA-terminal:

Quantity of symbols: 20 lines by 20 characters.
The resolution of a character matrix: 8x12 points
Supported code page: WIN 1251
Formed signal: VGA
The resolution: 640x480
Frequency of vertical synchronization: 60 Hz
Speed of exchange UART 19200 bps

Video terminal:

Quantity of symbols: 20 lines by 38 characters.
The resolution of an individual character matrix: 8x12 points
Supported code page: WIN 1251
Formed signal: Composite Video (PAL/SECAM)
Resolution: 625 lines (interlaced)
Frequency of vertical synchronization: 50 Hz
Speed of exchange UART 19200 bps

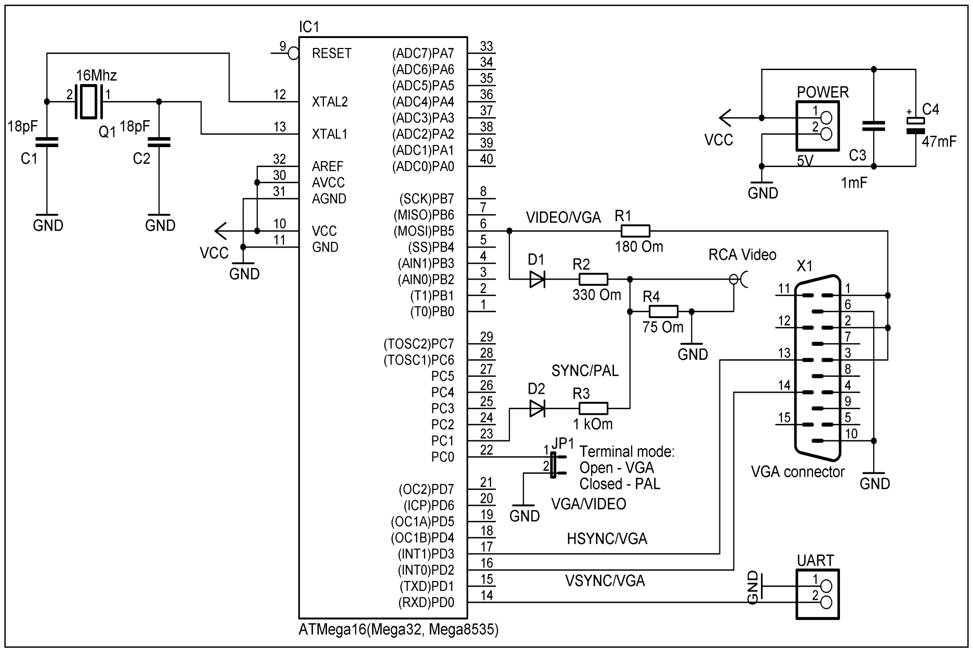


Figure 1: Basic scheme of a simple VGA/Video terminal.

Microcontroller: Mega8, Mega16, Mega32, Mega8535, etc.
Clock frequency of the microcontroller standard - 16 MHz.

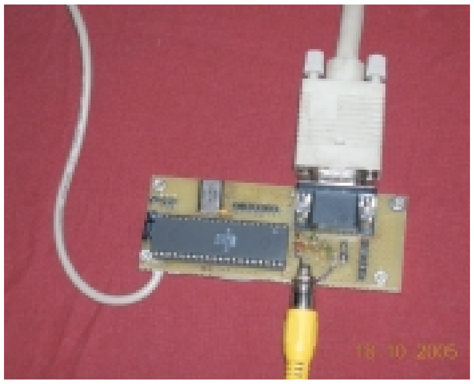
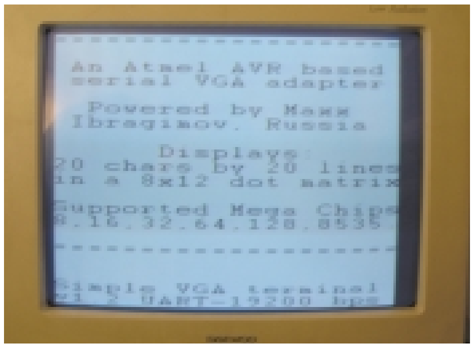
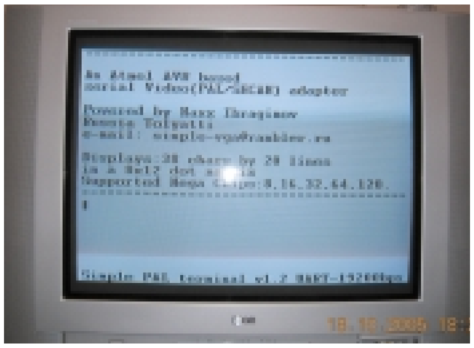


Figure 2: This is what it looks like in the flesh after an hour of assembling



```
-------------------------
An Atmel AVR based
serial VGA adapter

 Powered by Hazz
Ibragimov, Russia

     Displays
20 chars by 20 lines
in a 8x12 dot matrix

Supported Mega Chips
8.16.32.64.128.8535
-------------------------

Simple VGA terminal
v1.2 UART-19200 bps
```

Figure 3: And this is what it looks like on a VGA monitor



```
An Atmel AVR based
serial Video(PAL/SECAM) adapter

Powered by Hazz Ibragimov
Russia Tolyatti
e-mail: simple-vga@rambler.ru

Displays:38 chars by 28 lines
in a 8x12 dot matrix
Supported Mega Chips:8.16.32.64.128.
------------------------------------
_

Simple PAL terminal v1.2 UART-19200bps
```

Figure 4: And on the TV

Notes:
1. To avoid distortion of the image, when receiving data through the UART, for VGA, it is recommended to make the data exchange with the terminal in approximately 300-600 us after a signal of vertical synchronization (VSYNC).

2. The available internal RAM of the Mega8535 (only 512 bytes) is not enough for the formation of a video signal with a resolution of 38x20 symbols.

Explanations on work of the program

The algorithm of rendering the image is traditional enough, the main know-how of the project is the bit-by-bit shifting of the image, utilizing the SPI shift register SPDR via the MOSI pin. Thus two jobs are performed at the same time, when the subsequent byte for rendering is sent, the previous byte is shifted out through the shift register (SPI SPDR MOSI). The differences between Figures 6 and 7 demonstrate this.

Conclusion:

Given the project was written with WinAVR (GCC), it is

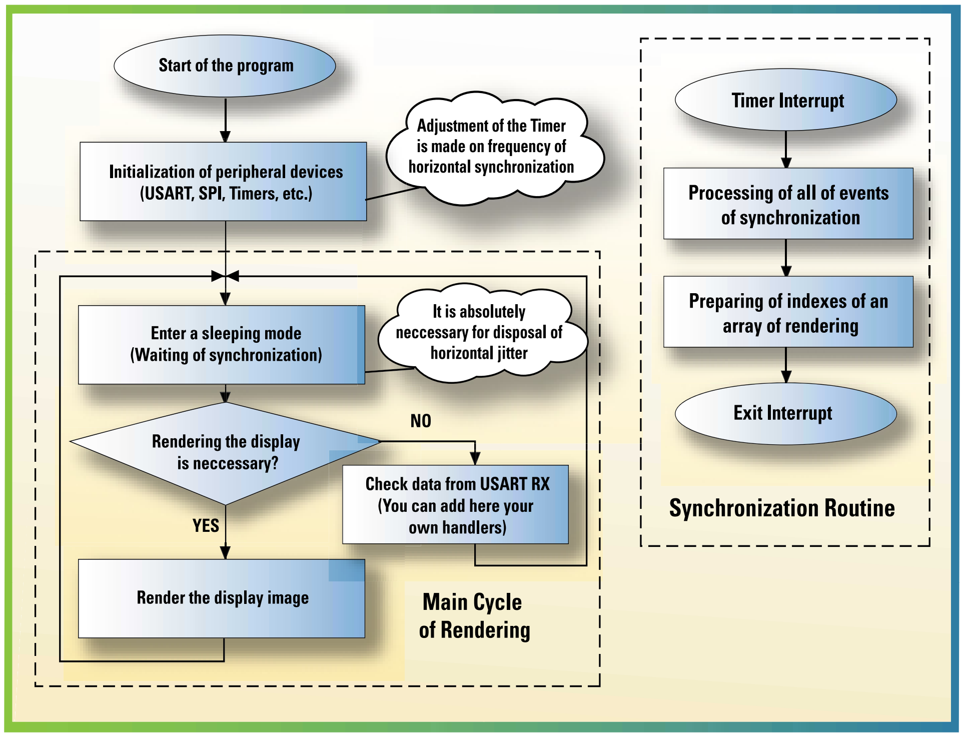

Start of the program
Initialization of peripheral devices (USART, SPI, Timers, etc.)
Adjustment of the Timer is made on frequency of horizontal synchronization
Enter a sleeping mode (Waiting of synchronization)
It is absolutely neccessary for disposal of horizontal jitter
Rendering the display is neccessary?
NO
YES
Check data from USART RX (You can add here your own handlers)
Render the display image
Main Cycle of Rendering
Timer Interrupt
Processing of all of events of synchronization
Preparing of indexes of an array of rendering
Exit Interrupt
Synchronization Routine

Figure 5: Base structure of the Simple VGA terminal program

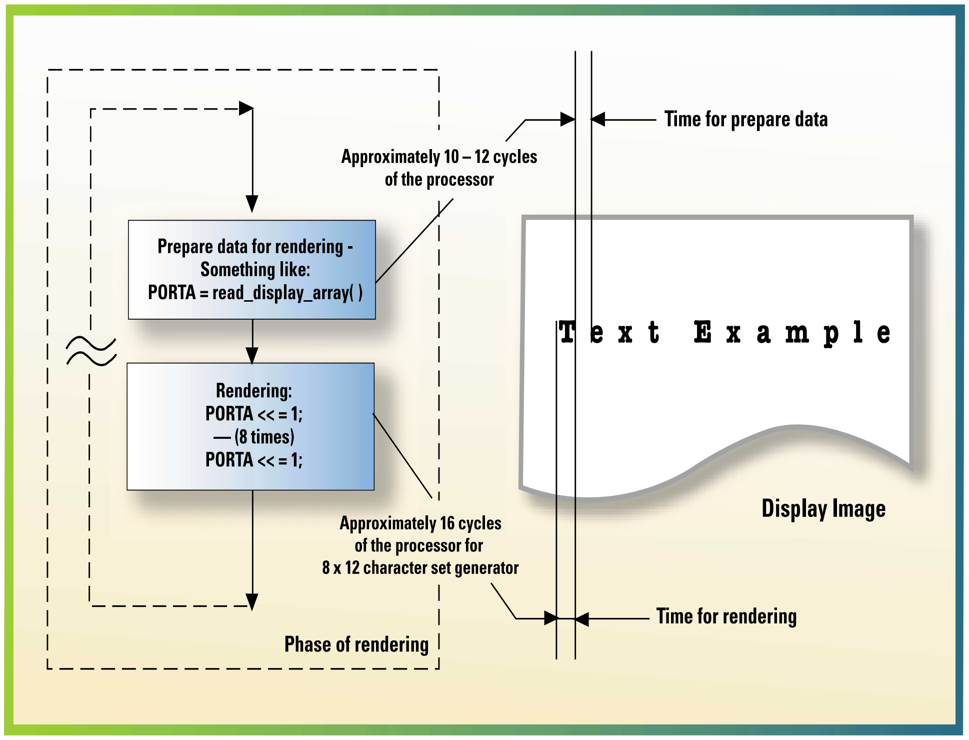

Time for prepare data
Approximately 10 – 12 cycles of the processor
Prepare data for rendering - Something like: PORTA = read_display_array( )
Text Example
Rendering: PORTA << = 1; — (8 times) PORTA << = 1;
Approximately 16 cycles of the processor for 8 x 12 character set generator
Display Image
Time for rendering
Phase of rendering

Figure 6: Traditional techniques of rendering

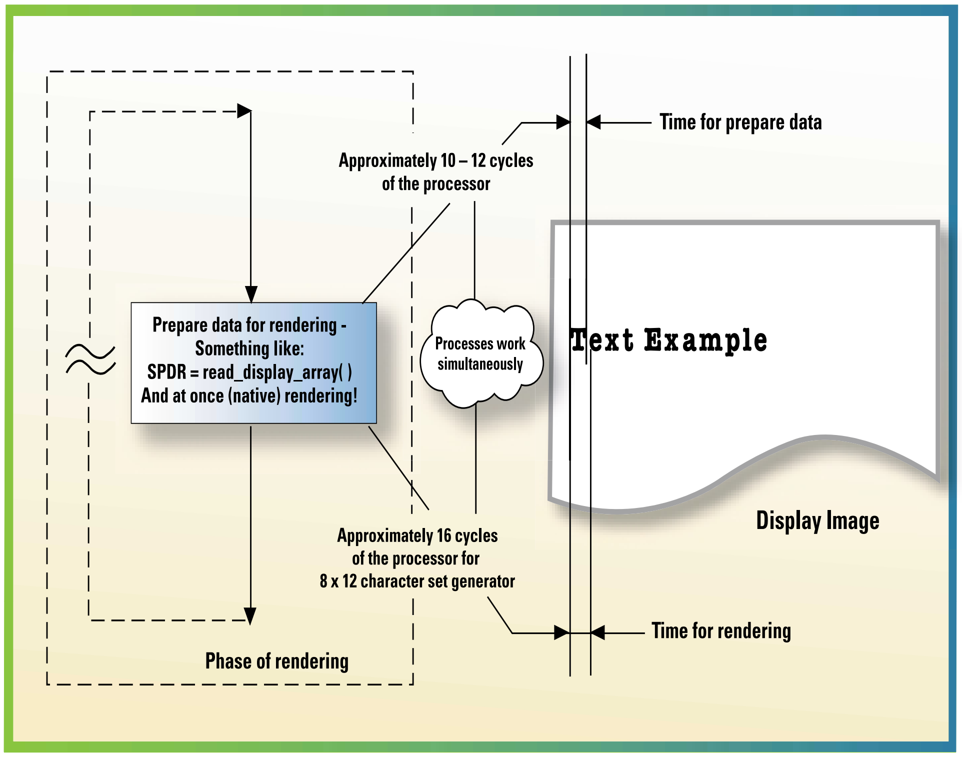

Time for prepare data
Approximately 10 – 12 cycles of the processor
Prepare data for rendering - Something like: SPDR = read_display_array( ) And at once (native) rendering!
Processes work simultaneously
Text Example
Display Image
Approximately 16 cycles of the processor for 8 x 12 character set generator
Time for rendering
Phase of rendering

Figure 6: Traditional technics of rendering

relatively easy to increase the resolution and/or frequencies used in creating the display images. With the forthcoming availability of AVR microcontrollers such as the Mega48, Mega88 and Mega168 officially supporting clock speeds of 20 MHz it is possible to achieve resolutions of 20 lines by 25-30 characters. This is possible using exactly the same circuitry.

Applications of the adapter:

Applications of the project are not limited to only one terminal variant (look demo in a folder examples) - despite on serious congestion of the processor regeneration of the display the remained capacity has enough for the organization of processing for example several digital and analog signals and reaction to them, and also delivery of results of their measurements on the display in real-time (Security systems, Industrial automatics etc.). The author has the improved variants of similar systems with the resolution of the symbolical display 40x24 symbols in mode VGA, working in commercial products.

The further development of the project watch on http://www.vga-avr.narod.ru/
Contact with author at the e-mail:
simple-vga@rambler.ru

IMAGEcraft

ANSI C development tools for Atmel AVR & TinyAVR

- Full featured IDE
- Code Compressor™
- Inline assembly, interrupt handlers in C
- Full support for AVR Studio
- Application Builder
- ISP support
- Code Browser
- Prices: ICCAVR V6 STD $199, PRO $499 ICCtiny V6 $129

high powered development, garage powered prices!

VISA
MC
AMEX
Check
MO

Full-featured 30-day demos at:
www.imagecraft.com
info@imagecraft.com
706 Colorado Ave. Palo Alto, CA 94303
(650) 493-9326 • FAX (650) 493-9329

...say 'NO' to JURASSIC PRICES! tm