floresta
Western New York, USA
Offline
Faraday Member
Posts: 2508

Re: arduino for vfd
« **Reply #18 on:** June 02, 2012, 01:00:07 AM »

Bigger Smaller Reset Quote

Have you substituted an LCD for your VFD? You have posted two broken links that contain timing diagrams that are essentially identical to one another and to those for all other LCDs. Why are you not using the timing diagram for your VFD? The timing diagrams for your VFD are very similar to those of LCDs in terms of the signal sequence but the times are significantly different. At least one of the parameters specified on the VFD diagram does is not show up on the LCD diagrams.

In any case your basic problem is that you still do not understand the relationship between whichever Timing Diagram you choose to follow and the code that you are attempting to use to implement that diagram. Specifically, you do not understand how the delays in the program relate to the information on the Timing diagram. This is evident by the values used in and the comments accompanying these two statements:

Code:
```
delay_us(41000);                        // Enable high for 41 ms for LCD module to run a
command.
...
delay_us(100);                          // Delay 100us between commands sent.
```

Take a look at the comments (not the values) in the corresponding statement in the the 'After' code shown in Reply #14. You must compare those comments to the Timing Diagram that I used, the one for the VFD, for them to make any sense. After you do that you should be able to fix the errors that still exist and find out what is missing.

.....

Your reply #17 above shows two functions that *should be identical* except for one statement (well, two statements the way you have implemented things). The timing for the 'write' operation is identical for instructions (the correct name for what you are calling commands) and for data.

Quiz: Why does the timing diagram show specific levels for E and R/W while the same diagram shows simultaneous high and low levels for RS?

.....

You obviously haven't used the 'code' tags correctly. Here's how:
1. Enter your text.
2. Copy and paste your code in the appropriate location.
3. Highlight the code portion.
4. Click on the 'code' button which looks like '#'.

You also obviously didn't check your links before submitting your post. You should do this while you are previewing your post so you can correct any problems before you submit the post.

Don

« *Last Edit: June 02, 2012, 01:03:44 AM by floresta* »

Report to moderator Logged

miragabe
Offline
Newbie
Posts: 14

Re: arduino for vfd
« **Reply #19 on:** June 05, 2012, 02:35:23 PM »

Bigger Smaller Reset Quote Modify Remove

Hello Don,
1)
Quote
Have you substituted an LCD for your VFD?
Don't have one; sent for 2 online yesterday 16x2 and 20x4, 5x7 dot matrix, arduino compatible.

2)
Quote
The timing diagrams for your VFD are very similar to those of LCDs in terms of the signal sequence but the times are significantly different. At least one of the parameters specified on the VFD diagram does is not show up on the LCD diagrams.
I honestly don't see which one it is; I've compared each diagram and what I do see is that the LCD diagram gives the tEr and tEf (enable rise and fall times). I also see, as mentioned before, the tAS (address set up time) for the LCD is 40ns as compared to the 20ns, which I thought was the difference you were referring to.
Based strictly on the timing diagram for the vfd I have revised the two functions as follows, uploaded and nothing yet:

Code:

```
void lcdCommand( unsigned char cmnd )
{
  LCD_DPRT = cmnd;                      // ready data lines
  LCD_CPRT &= ~ (1<<LCD_RS);          // RS low (ref.quiz:lower line on the timing diagram)to
select the Instruction register.
  LCD_CPRT &= ~ (1<<LCD_RW);        // RW low to write instructions.
  LCD_CPRT |= (1<<LCD_EN);            // Enable pin high to latch data on the falling edge.
  delay_us(.23);                              // Enable pin high for 230 ns to enable latching of
instruction set.
  LCD_CPRT &= ~ (1<<LCD_EN);        // LCD_EN pin of PortB low after sending an instruction.
```

```
  delay_us(.23);                              // Delay 230 ns between instructions sent.
}

//*****************************************************
void lcdData( unsigned char data )
{
  LCD_DPRT = data;                  // ready data lines
  LCD_CPRT |= (1<<LCD_RS);       // RS pin high (ref.quiz:upper line on the timing diagram)to
select the Data register.
  LCD_CPRT &= ~ (1<<LCD_RW);  // RW low to write instructions.
  LCD_CPRT |= (1<<LCD_EN);      // Enable pin set to latch data on the falling edge.
  delay_us(.23);                         // Enable pin high for 230 ns to enable latching of data
set.
  LCD_CPRT &= ~ (1<<LCD_EN);  // LCD_EN pin of PortB low after sending data byte.
  delay_us(.23);                        // Delay 230 ns between instructions sent.
}
```

3)
Quote
In any case your basic problem is that you still do not understand the relationship between whichever Timing Diagram you choose to follow and the code that you are attempting to use to implement that diagram. Specifically, you do not understand how the delays in the program relate to the information on the Timing diagram.
You are correct; I do not understand how exactly to implement the timing diagram for the device in terms of the delays required in the program. That's the reason why I came onto this forum for help in understanding how this is done in general so that I may simply do it on my own for any device when given the specifications sheet and it's timing diagrams, in addition to learning as much as can be garnered about every other aspect of successfully interfacing any microcontroller to any peripheral. I was given this task to do for quite some time now, and have tons of code variations for it, first with the PIC18F458, and when that fell through because of outmoded connection issues, now with the arduino uno.

4)
Quote
You must compare those comments to the Timing Diagram that I used, the one for the VFD, for them to make any sense. After you do that you should be able to fix the errors that still exist and find out what is missing.
If I'm not mistaken, what could be missing (and which would replace the second delay), is with reference to these links:
1) http://www.8051projects.net/lcd-interfacing/busyflag.php):

Quote
►BF - the Busy Flag

The Busy Flag is a status indicator flag for LCD. When we send a command or data to the LCD for processing, this flag is set (i.e BF =1) and as soon as the instruction is executed successfully this flag is cleared (BF = 0). This is helpful in producing and exact ammount of delay. for the LCD processing.

To read Busy Flag, the condition RS = 0 and R/W = 1 must be met and The MSB of the LCD data bus (D7) act as busy flag. When BF = 1 means LCD is busy and will not accept next command or data and BF = 0 means LCD is ready for the next command or data to process

2) http://www.8051projects.net/lcd-interfacing/busyflag.php:

Code:

```
#define F_CPU 16000000UL

#include <avr/io.h>
#include <util/delay.h>

#define   LCD_DPRT  PORTD         // configuring PortD for data
#define   LCD_DDDR  DDRD
#define   LCD_DPIN  PIND
#define   LCD_CPRT  PORTB           // utilizing PortB pins for the control.
#define   LCD_CDDR  DDRB
#define   LCD_CPIN  PINB
#define   LCD_RS  0                                 // control pin assignments.
#define   LCD_RW  1
#define   LCD_EN  2
#define      LCD_D7  3
//*****************************************************
void delay_us(unsigned int d)
{
  _delay_us(d);
}
//*****************************************************
void LCD_busy()
{
     DDRD = DDRD &= ~ (1<<LCD_D7); //Make D7th bit of LCD as input
     DDRB = DDRB |= (1<<LCD_EN);    //Make port pin as output
     LCD_CPRT &= ~ (1<<LCD_RS);    //Selected instruction register
     LCD_CPRT   |= (1<<LCD_RW);    //We are reading
     while(LCD_D7){                        //read busy flag again and again till it becomes 0
       LCD_CPRT &= ~ (1<<LCD_EN);  //Enable H->L
       LCD_CPRT |= (1<<LCD_EN);
     }
```

```
}
//*******************************************************
void lcdCommand( unsigned char cmnd )
{
  LCD_DPRT = cmnd;                  // ready data lines
  LCD_CPRT &= ~ (1<<LCD_RS);     // RS low (lower line on the timing diagram)to select the
Instruction register.
  LCD_CPRT &= ~ (1<<LCD_RW);    // RW low to write instructions.
  LCD_CPRT |= (1<<LCD_EN);       // Enable pin high to latch data on the falling edge.
  delay_us(.23);                          // Enable pin high for 230 ns to enable latching of
instruction set.
  LCD_CPRT &= ~ (1<<LCD_EN);    // LCD_EN pin of PortB low after sending an instruction.
  LCD_busy();                             // Wait for LCD to process the instruction.
}

//*******************************************************
void lcdData( unsigned char data )
{
  LCD_DPRT = data;
  LCD_CPRT |= (1<<LCD_RS);       // RS pin high (upper line on the timing diagram)to select the
Data register.
  LCD_CPRT &= ~ (1<<LCD_RW); // RW low to write instructions.
  LCD_CPRT |= (1<<LCD_EN);      // Enable pin set to latch data on the falling edge.
  delay_us(.23);                        // Enable pin high for 230 ns to enable latching of data
set.
  LCD_CPRT &= ~ (1<<LCD_EN); // LCD_EN pin of PortB low after sending data byte.
  LCD_busy();                           // Wait for LCD to process the instruction.
}
```

This compiles but when uploaded, still nothing.
Dave.

Report to moderator 69.92.82.217

miragabe
Offline
Newbie
Posts: 14

Re: arduino for vfd
« **Reply #20 on:** June 05, 2012, 02:43:23 PM »

Bigger Smaller Reset Quote Modify Remove

...I know you said don't post anymore pdf's but this is what I've been looking at for comparison.

timing diags lcd-vfd.pdf (106.68 KB - downloaded 3 times.)

Report to moderator 69.92.82.217

floresta
Western New York, USA
Offline
Faraday Member
Posts: 2508

Re: arduino for vfd
« **Reply #21 on:** June 05, 2012, 08:35:11 PM »

Bigger Smaller Reset Quote

Quote
Quote
Have you substituted an LCD for your VFD?
Don't have one; sent for 2 online yesterday 16x2 and 20x4, 5x7 dot matrix, arduino compatible.
That is what I thought. If you do not have an LCD and you do have a VFD then why were you using the timing diagrams for an LCD?

Quote
I also see, as mentioned before, the tAS (address set up time) for the LCD ...
Please tell me how this delay is accounted for in your code. It should be somewhere in here:
Code:

```
{
  LCD_DPRT = cmnd;                    // ready data lines
  LCD_CPRT &= ~ (1<<LCD_RS);        // RS low (ref.quiz:lower line on the timing diagram)to
select the Instruction register.
  LCD_CPRT &= ~ (1<<LCD_RW);      // RW low to write instructions.
  LCD_CPRT |= (1<<LCD_EN);          // Enable pin high to latch data on the falling edge.
  delay_us(.23);                              // Enable pin high for 230 ns to enable latching of
instruction set.
  LCD_CPRT &= ~ (1<<LCD_EN);      // LCD_EN pin of PortB low after sending an instruction.
  delay_us(.23);                              // Delay 230 ns between instructions sent.
}
```

Believe me, you do not want to attempt to deal with the busy flag until you can get the device functioning with simple time delays between instructions.

Don

Report to moderator Logged

miragabe
Offline
Newbie
Posts: 14

Re: arduino for vfd
« **Reply #22 on: Today** at 05:56:18 PM »

Bigger Smaller Reset Quote Modify Remove

I tried this and these connections...still midnight.

Code:
```
void lcdCommand( unsigned char cmnd )
{
  LCD_DPRT = cmnd;         // ready data lines
  LCD_CPRT &= ~ (1<<LCD_RS);   // RS low (lower line on the timing diagram)to select the
Instruction register.
  LCD_CPRT &= ~ (1<<LCD_RW);   // RW low to write instructions.
  delay_us(.020);              // Delay 20 ns for address set-up time.
  LCD_CPRT |= (1<<LCD_EN);     // Enable pin high to latch data on the falling edge.
  delay_us(.23);           // Enable pin high for 230 ns to enable latching of instruction set.
  LCD_CPRT &= ~ (1<<LCD_EN);   // LCD_EN pin of PortB low after sending an instruction.
  delay_us(.23);           // Enable pin low for 230 ns after sending instruction set.
}

//*****************************************************
void lcdData( unsigned char data )
{
  LCD_DPRT = data;
  LCD_CPRT |= (1<<LCD_RS);     // RS pin high (upper line on the timing diagram)to select the
Data register.
  LCD_CPRT &= ~ (1<<LCD_RW);   // RW low to write instructions.
  delay_us(.020);              // Delay 20 ns for address set-up time.
  LCD_CPRT |= (1<<LCD_EN);     // Enable pin set to latch data on the falling edge.
  delay_us(.23);               // Enable pin high for 230 ns to enable latching of data set.
  LCD_CPRT &= ~ (1<<LCD_EN);   // LCD_EN pin of PortB low after sending data byte.
  delay_us(.23);           // Delay 230 ns between data sent.
}
```
Dave.

PICT. 1 .JPG (281.18 KB, 1632x1224 - viewed 2 times.)

PICT.2.JPG (231.71 KB, 1632x1224 - viewed 3 times.)

PICT. 3.JPG (203.52 KB, 1632x1224 - viewed 2 times.)

Report to moderator 69.92.82.217

Re: arduino for vfd
« **Reply #23 on: Today** at 08:49:22 PM »

Bigger Smaller Reset Quote

Quote
I tried this and these connections...still midnight.
What you now have completed are the routines that you need to send Instructions and Data to the LCD controller. Next you have to fix up the initialization routine. As I mentioned in reply #8 your data sheet makes a reference to 'initialization by commands' but it does not seem to specify what those commands should be. The sequence of instructions that you have in the program that you attached to reply #13 may work, but you have to make sure that you do not send them to the LCD controller too quickly. This is where the time it takes the LCD controller to deal with an instruction has to be implemented and you get those times from the VFD Instruction table on pages 3 and 4.

When you get that fixed up I suggest that you first try to write a single character to the display and then put the processor in an endless loop. You want to make sure that a correctly written character is not overwritten by something else if your string writing routine is not working as you expect.

Next I would use a loop to send 80 different displayable characters to the display and again end by putting the processor in an endless loop. I usually start with the ASCII code for a '/' and increment it each time around. If you put a 200mS delay in the loop you can see what is going on. This is particularly interesting with LCDs, especially those with less than 80 characters.

In your 'gotoxy' routine you have confused the memory addresses in the LCD controller with the instruction that is used to set those addresses. The addresses are 7-bit values so 0x80, 0xC0, 0x84, and 0xD4 are not valid addresses and you will not find those addresses mentioned anywhere in most (there are exceptions) LCD data sheets. You really should identify where any magic numbers come from.

As far as your photographs are concerned the quality is good but the wires look like red spaghetti. It is impossible to follow any specific wire to see where it starts and where it ends.

Don