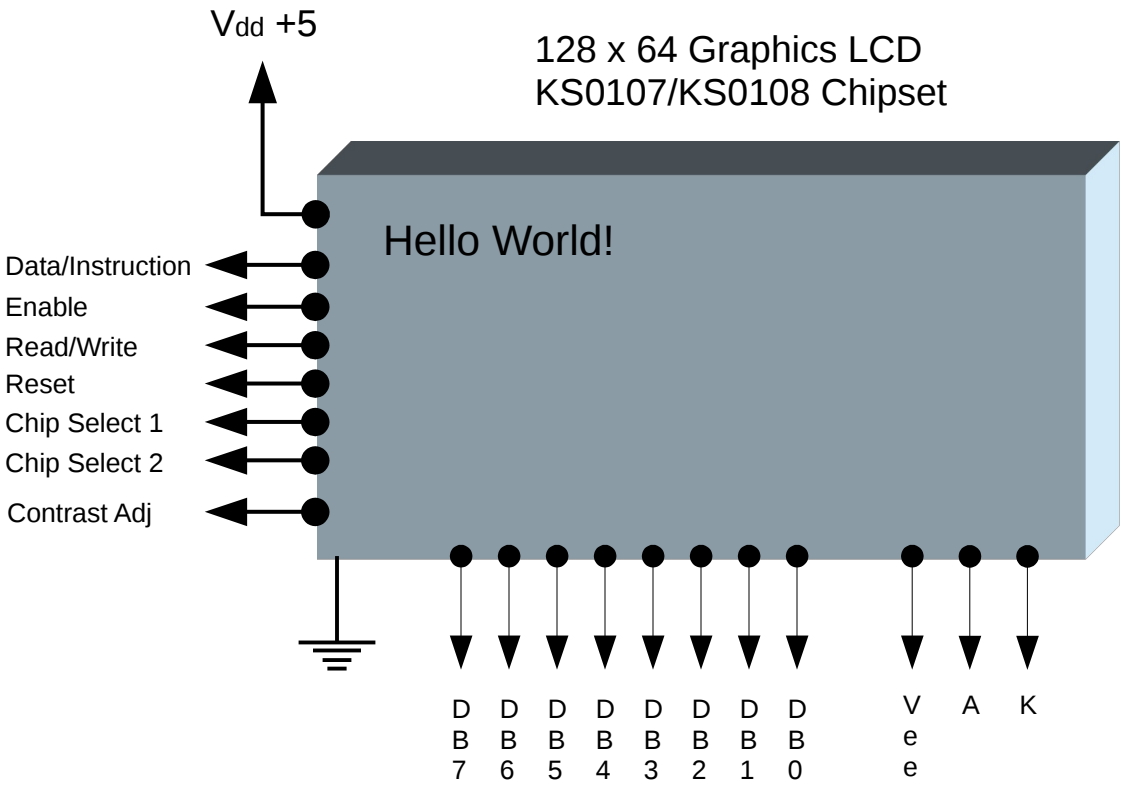
Graphics LCD Display C Library for the PIC18F4550 MCU (or similar advanced microcontrollers)

Version 1.00 – 5/2013 **www.muniac.com**



SCHEMATICS

Introduction:

This document includes a collection of schematics that suggest various methods of wiring and interfacing a KS0107/KS0108 based 128x64 graphics LCD to an MCU for driving it with the GLCD library functions. The schematics included herein have been tested and all work properly. That said, many other workable circuits are possible. The best one(s) depend on specific MCUs and the applications being designed. It is hoped that the circuits and wiring presented here will assist anyone driving a GLCD from an MCU application. If nothing else, the circuits will serve as a good basic starting point. **MAKE SURE TO CHECK THE PIN OUT FUNCTIONS OF THE GLCD BEING USED AND MAKE ANY MODIFICATIONS NECESSARY IF THEY DIFFER FROM THE SCHEMATICS SHOWN. FAILURE TO DO SO MAY RESULT IN DAMAGING THE GLCD AND/OR THE MCU.**

GLCD Default Circuit:

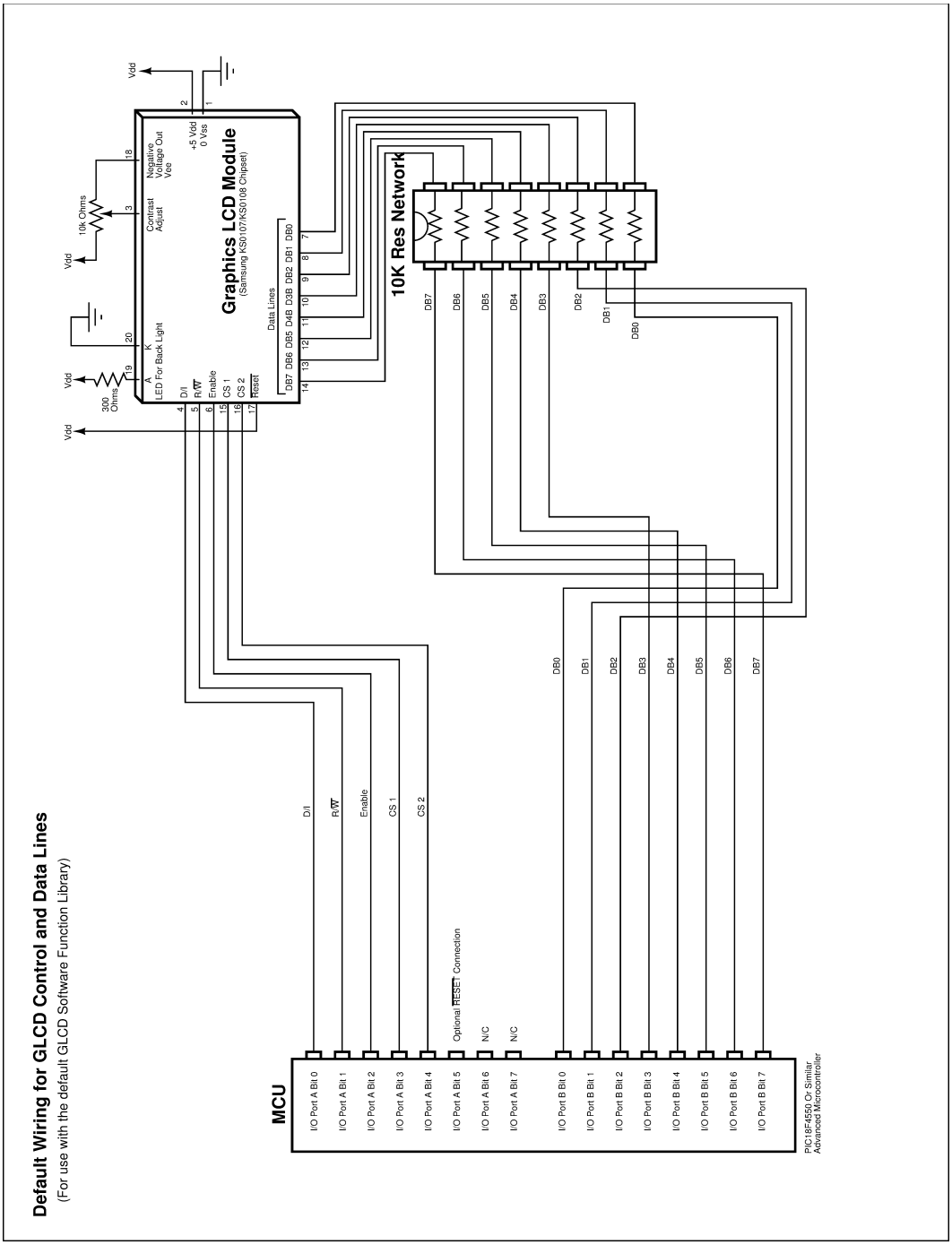
The easiest circuit to use is the default GLCD wiring. It uses the least external components and is recommended as a good starting point. Once the GLCD is up and running, modifications can be made if and as required. A resistor network has been included on the data lines to limit current. If the MCU port initializes as an output in an undefined state this could cause excessive currents to flow. This would occur if the GLCD data pins power up as an output in an undefined state. Such a condition may possibly result in two outputs competing for source and sink. The resistor network limits current in this case and protects the output drivers from an overload. If such a condition is known to not occur then the resistor network can be eliminated. This saves on a component and extra wiring.

To save an I/O port pin, RESET has been tied to Vdd in the default circuit. The hardware reset functions will thus be handled in software with only a very small performance penalty. With RESET tied to Vdd only 5 control lines need to be managed. By default port A is used for the control lines and port B is used for the data lines. This can be changed in HWSpec.h as required to suit specific applications. If possible, use with default wiring initially to get the GLCD working. The glcd.lib has been compiled for the default circuit and for those using MPLAB IDE it may only be necessary to include the library in the linker script. Note control and data lines have simply been wired in ascending order by pin number. To communicate with the GLCD, port A needs to be output only and 5 bits wide. Port B needs to be input/output and 8 bits wide.

Attention to the contrast circuit is important. If it isn't functioning properly the GLCD won't display anything. Make sure the GLCD being used has proper contrast control. The display shown in the circuit includes LEDs for back lighting. A current limiting resistor has been included to limit current to 16.6ma (5V/300 Ohms). Change this value as required. The 300 ohm resistance provided adequate back lighting for the GLCD used in the development effort.

Good wiring practice includes double checking all connections, testing polarity and making sure everything is correct before powering the circuit up. Also make sure the power supply is providing clean DC at the desired voltage and grounding is good. Double check all power and ground connections as well as all digital signal lines. A single wire out of place can cause damage and/or operational failures.

Default GLCD Wiring For Use With GLCD Function Library (Reset tied to Vdd)

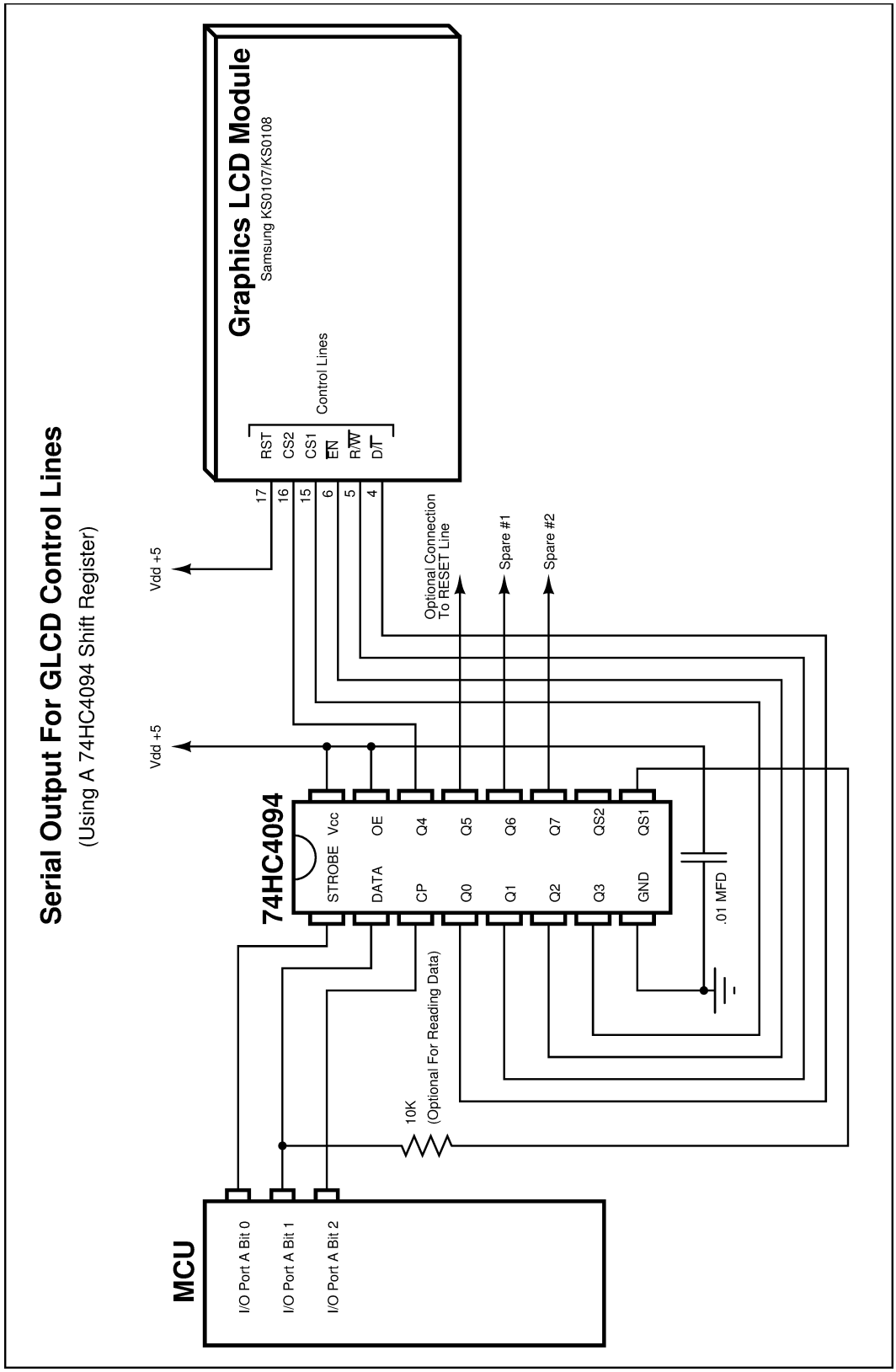


Serial GLCD Control Lines:

With direct connection, 5 or 6 MCU I/O port pins will be required to manage the control lines. With some added circuitry utilizing a serial in parallel out shift register like the 74HC4094 it is possible to use only 3 I/O port pins to control up to 8 single bit signals. The circuit below was created and tested specifically for this purpose. It may allow moving the control signals to a less used narrower port like PORT E on the 40 pin PIC18F4550, for example. The GLCD function library includes the code to manage the 74HC4094 circuit presented herein. Several parameters in HWSpec.h need only be changed to enable serial output to GLCD control lines. 2 or 3 extra outputs (spare utility) can be used for other functions requiring a digital output. It is left to the system designer to determine whether or not a serial approach is advantageous.

The shift register circuit simply allows bits from a control byte (CByte) to be clocked into the 74HC4094. The shift length will range from 5 to 8 depending on whether or not RESET has been tied to Vdd and if the extra spare utility bits are being used. The 74HC4094 has output latches which avoid the turbulence of shifting in bits from appearing on its Q outputs. When the entire string of bits has been shifted in, a strobe pulse latches the shift register and updates the values on the Q outputs. The circuit was tested without the strobe which proved unsuccessful owing to the GLCD not being able to handle the instability on its control lines. The serial control line approach reduces performance only very slightly. The serial circuit uses the normal delay required in the direct connection to accomplish the shifting. The shifting process is only slightly longer than the required delay. Hence only a slight bit of performance loss occurs when serial output replaces the delay. Obviously if the shift length increases to 6, 7 or 8 more performance would be lost. With a shift length of 5, performance loss is barely noticeable. The fastest shift operation would require that RESET be tied to Vdd and no spare utility bits being used.

An optional resistance connecting the QS1 to the DATA pin allows reading back each bit shifted out of the 74HC4094 should this be necessary. In the example circuit, bit 1 of the MCU's port would need to be configured as input to retrieve the value. Successive reads would be required to read the entire string of bits. This optional resistance is shown in the circuit for completeness. The resistance can be omitted if reading back data isn't required. The GLCD library does not provide for reading the QS1 output. Refer to the 74HC4094 data sheet for detailed information about how this chip functions and a truth table. This circuit may come in handy for some applications that can't afford dedicating a wider port to control line management.


Serial Output For GLCD Control Lines
(Using A 74HC4094 Shift Register)
Graphics LCD Module
Samsung KS0107/KS0108
Control Lines
RST
CS2
CS1
EN
R/W
D/I
17
16
15
6
5
4
Vdd +5
Vdd +5
Optional Connection To RESET Line
Spare #1
Spare #2
74HC4094
STROBE
DATA
Vcc
OE
CP
Q4
Q5
Q6
Q7
QS2
QS1
Q0
Q1
Q2
Q3
GND
.01 MFD
10K
(Optional For Reading Data)
MCU
I/O Port A Bit 0
I/O Port A Bit 1
I/O Port A Bit 2

Conclusions:

Many options exist for creating a working interface between an MCU and GLCD. With the cost of MCUs so low, one could be dedicated entirely to managing a GLCD. In this case, conserving I/O port pins would be less of a problem and perhaps not a problem at all. Using the GLCD library, additional software could be written to allow commands to be sent to a slave GLCD/MCU combination via RS232, RS485, SPI or another popular communication protocol. A simple command structure would need to be developed allowing access to the required text and graphics functions. If an application has 11 I/O port pins available in 2 ports, then a standalone system can be easily created which includes the GLCD. In this case, the application code would simply make calls to the GLCD library functions as needed.

MUNIAC, LLC
scott@muniac.com