LABORÜBUNGEN MIKROCONTROLLER – ÜBUNG 2

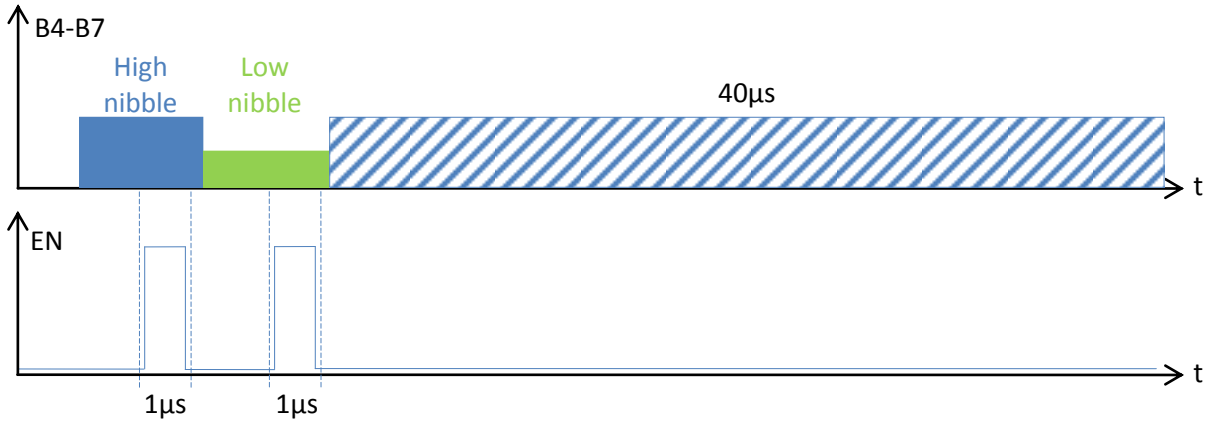
BEISPIEL 4 – LCD-ANSTEUERUNG

Das im Labor verwendete LCD ist mit insgesamt 6 Leitungen an das Entwicklungsboard angebunden und wird im 4-Bit-Modus betrieben. Neben den 4 Datenleitungen (B4-B7) gibt es 2 Steuerleitungen (RS an B2 und EN an B3), die der LCD-Ansteuerung dienen.

Erstellen Sie eine neue Datei inkl. Headerdatei zur LCD-Ansteuerung (Name z.B. LCD.c und LCD.h) und definieren Sie die Methoden *SendCommand(unsigned char)* und *SendData(unsigned char)*.

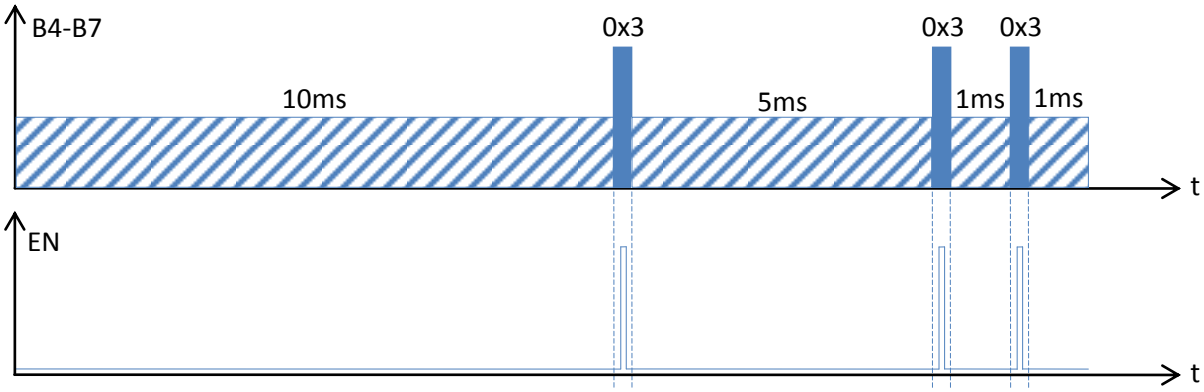
Das LCD unterscheidet zwischen Steuerbefehlen und anzuzeigenden Daten („Datenbefehlen“), wobei die Unterscheidung mittels der RS-Steuerleitung erfolgt. Soll ein Steuerbefehl übertragen werden, muss RS Low sein, bei Daten High.

Da das LCD wie oben beschrieben im 4-Bit-Modus betrieben wird, müssen Bytes auf 2x4 Bit aufgeteilt und getrennt übertragen werden – zuerst die oberen 4 Bit (das so genannte High Nibble), dann die unteren 4 Bit (das so genannte Low Nibble). Um zwischen den beiden Nibbles unterscheiden zu können, muss die EN-Leitung während jedem angelegten Nibble für 1 µs auf High gesetzt werden (vgl. Abbildung unten). Den Rest der Zeit muss die EN-Leitung Low sein. Nachdem ein Byte übertragen wurde, muss 40 µs gewartet werden, bis das LCD die Steuerinformation bzw. die Daten verarbeitet hat. Implementieren Sie auf Basis dieser Informationen die beiden o.g. Funktionen.



Um das LCD ansteuern zu können, muss es aufgeweckt und initialisiert werden. Vor diesen beiden Schritten muss die Datenrichtung (Lesen/Schreiben) für den Port, an den das LCD angeschlossen ist, richtig gesetzt werden, damit EN, RS und die 4 Datenleitungen beschrieben werden können. Implementieren Sie das in einer Routine *InitLCD()*, die das LCD aufweckt und in den 4-Bit-Modus versetzt.

Der Aufweckprozess ist in der nachfolgenden Abbildung illustriert. Nachdem 10 ms gewartet wird, bis das LCD eingeschaltet ist, wird das Wakeup-Signal (0x3 hexadezimal) an die Datenleitungen angelegt. Beachten Sie, dass das Wakeup-Signal ein Steuerbefehl ist und daher die RS-Leitung beim Setzen der Datenleitungen Low sein muss. Beachten Sie außerdem, dass die EN-Leitung nach dem Übertragen der 4 Bit wie oben 1 µs lang High sein muss (der Übersichtlichkeit halber ist diese Zeit nicht abgebildet). Anschließend wird 5ms gewartet, das Wakeup-Signal erneut angelegt und 1 ms gewartet. Dieser Schritt wird ein weiteres Mal wiederholt – dann ist das Display betriebsbereit.



Nachstehend ist die Wakeup-Sequenz tabellarisch zusammengefasst (ebenfalls ohne Berücksichtigung der EN-Leitung, deren Verwendung oben beschrieben ist):

Bootzeit des LCDs abwarten	10ms
Wakeup-Steuerbefehl 0x03 senden	min. 1µs
Wartezeit	5ms
Wakeup-Steuerbefehl 0x03 senden	min. 1µs
Wartezeit	1ms
Wakeup-Steuerbefehl 0x03 senden	min. 1µs
Wartezeit	1ms

Nach dem Aufwecken des LCDs muss in den 4-Bit-Modus gewechselt werden, indem ein 1-Byte-Steuerbefehl übertragen wird, in dem nur das Bit mit der Nummer 1 (bei 0 zu zählen beginnend) gesetzt ist. Danach muss 40 µs gewartet werden, bis das Display für weitere Befehle bereit ist.

Anschließend kann das Display konfiguriert werden. Konsultieren Sie dazu den nachfolgenden Datenblattauszug und senden Sie zumindest den Befehl zum Einschalten des Displays und den zum Wechsel in den 2-Zeilen-Modus. Die Verwendung anderer Cursor-Modi (wie z.B. Auto-Inkrement) ist zwar empfohlen, aber nicht verpflichtend.

Um Daten auf das Display schreiben zu können, werden diese zeichenweise (als *unsigned char* im ASCII-Zeichensatz) als Datenbefehl ans Display übertragen. Zwischen zwei übertragenen Zeichen muss eine Wartezeit von 40 µs eingehalten werden. Befindet sich der Cursor im Auto-Inkrement-Modus, wird automatisch an die nächste Stelle am Display weitergeschaltet. Beachten Sie, dass das Display einen internen Puffer verwendet, in dem die Zeilen länger sind als auf dem Display physikalisch sichtbar. Verwenden Sie daher bei Bedarf die unten beschriebenen Befehle zur manuellen Positionierung des Cursors.

Geben Sie auf Basis dieser Informationen zuerst „Hello world“ auf dem LCD aus und erweitern Sie anschließend Ihr Projekt so, dass im Abstand von ca. 500ms eine (beliebige) Zahl auf dem Display ausgegeben wird. Verwenden Sie zur Implementierung dieser Funktionalität die Funktion sprintf und erstellen Sie eine Funktion *WriteLCDString(char *str)*, der ein auf dem Display auszugebender String übergeben wird.

Steuerbefehle (Auszug)

Instruction	Code									Function	Execution time (max)
	RS	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
Display Clear	0	0	0	0	0	0	0	0	1	Clear entire display area, Restore display from shift, and load address counter with DD RAM address 00H.	1.64ms
Display / Cursor Home	0	0	0	0	0	0	0	1	*	Restore display from shift and load address counter with DD RAM address 00H.	1.64ms
Entry Mode Set	0	0	0	0	0	0	1	I/D	S	Specify direction of cursor movement and display shift mode. This operation takes place after each data transfer (read/write).	40µs
Display ON/OFF	0	0	0	0	0	1	D	C	B	Specify activation of display (D) cursor (C) and blinking of character at cursor position (B).	40µs
Display/ Cursor Shift	0	0	0	0	1	S/C	R/L	*	*	Shift display or move cursor.	40µs
Function Set	0	0	0	1	DL	N	F	*	*	Set interface data length (DL), number of display line (N), and character font (F).	40µs
RAM Address Set	0	0	1	ACG						Load the address counter with a CG RAM address. Subsequent data access is for CG RAM data.	40µs
DD RAM Address Set	0	1	ADD							Load the address counter with a DD RAM address. Subsequent data access is for DD RAM data.	40µs
CG RAM/ DD RAM Data Write	1	Write data								Write data to CG RAM or DD RAM.	40µs
	I/D = 1 : Increment S = 1 : Display Shift On D = 1 : Display On C = 1 : Cursor Display On B = 1 : Cursor Blink On S/C = 1 : Shift Display R/L = 1 : Shift Right DL = 1 : 8-Bit N = 1 : Dual Line F = 1 : 5x10 dots									I/D = 0 : Decrement S/C = 0 : Move Cursor R/L = 0 : Shift Left DL = 0 : 4-Bit N = 0 : Signal Line F = 0 : 5x8 dots	DD RAM : Display Data RAM CG RAM : Character Generator RAM ACG : Character Generator RAM Address ADD : Display Data RAM AC : Address Counter	

Note 1: Symbol "*" signifies an insignificant bit (disregard).
Note 2: Correct input value for "N" is predetermined for each model.