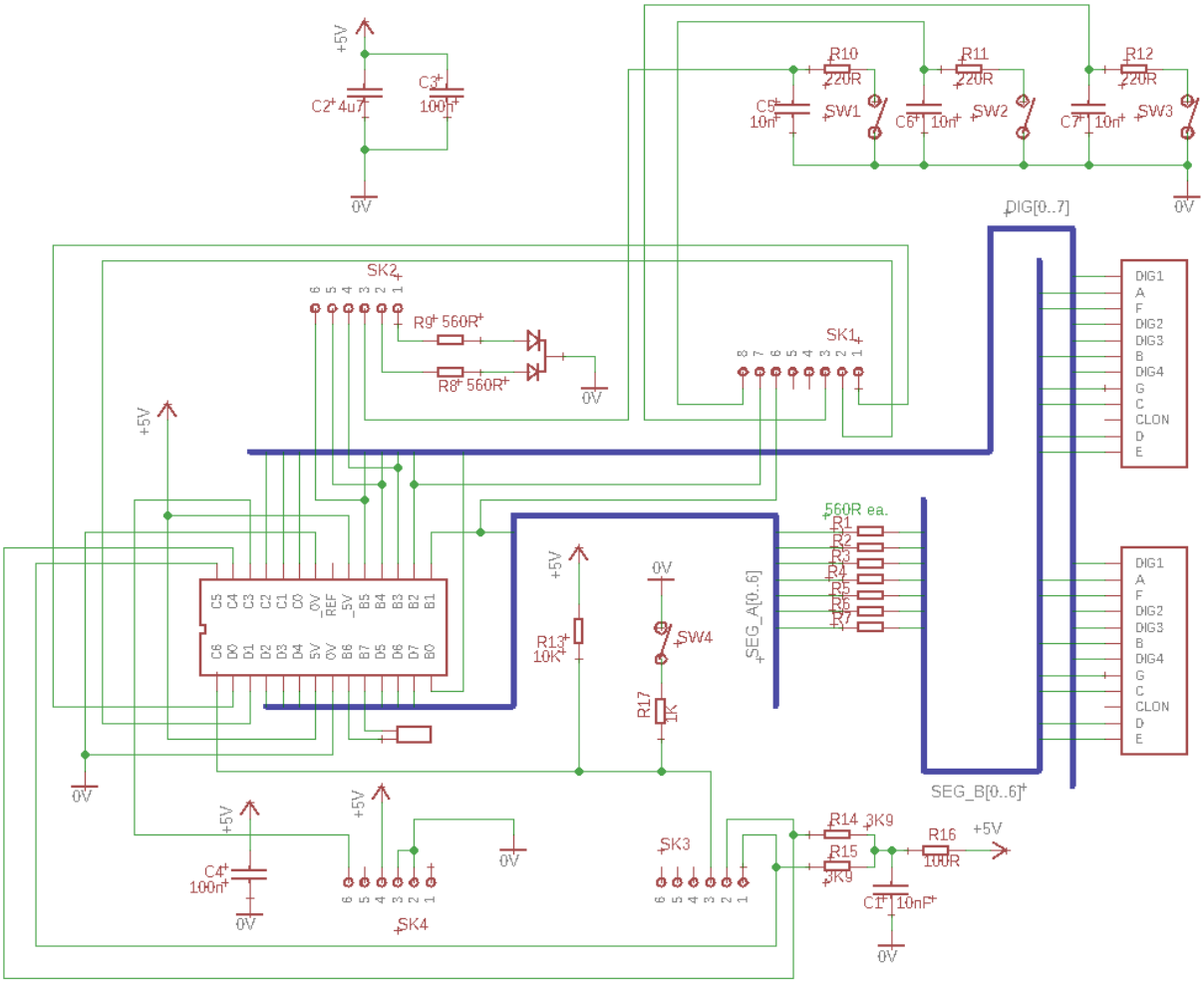
PCB111000_UNO

What is it

It is a simplified version of PCB111000 designed to plug onto an Arduino UNO pcb. It contains the display, the OS device (an Atmega 328 loaded with the mini-OS), user switches and the dual led. Note: The mini-OS consists of the display driver and PCB bootloader.

The UNO supplies the USB bridge, power and sockets and the user device (an Atmega 328 that hosts an Atmega bootloader and the user projects).

Circuit diagram (note: A watch crystal is connected between B6 and B7)



ARDUINO UNO connections

SK1		SK2		SK3		SK4	
1	D0 RXD	1	B0	1	C5 SCL	1	
2	D1 TXD	2	B1	2	C4 SDA	2	0V
3	D2	3	B2	3	C3	3	0V
4	D3	4	B3 MOSI	4	C2	4	5V
5	D4	5	B4 MISO	5	C1	5	3V3
6	D5	6	B5 SCK	6	C0	6	C6 RESET
7	D6						
8	D7						

PCB111000/Arduino interconnections

Note:
The Arduino_UNO (to be k/a the UNO) contains the user IC, the one that hosts user projects and Atmega bootloader. PCB111000_UNO (to be k/a PCB111000_1 or just pcb_A) contains the IC that hosts the mini-OS.

	PCB111000_1	UNO	
	SW1	B2	User switches
	SW2	D7	
	SW3	D2	
	SW4 C6 RESET	C3	Resets mini-OS/programmer IC
	R8	B0	Dual LED driver
	R9	B1	
	R14	C4 SDA	I2C bus
	R15	C5 SCL	
	B3	B3 MOSI	Bi directional programming interface
	B4	B4 MISO	
	B5	B5 SCK	
	C3	C6 Reset	Resets the user IC
	B1	D5	Reset SW control for a CA display
	B2	D6	Reset SW control for a CC display

PCB111000_1 Display connections

	Segments	Port	Digits	Port
	A	B1	1	B5
	B	D2	2	C0
	C	D3	3	C1
	D	D4	4	C2
	E	D5	5	B0
	F	D6	6	B2
	G	D7	7	B3
			8	B4

Project system applications

1. Hex/text bootloader. This resides in the boot partition of the user device and programmes its flash with hex and text files (it replaces the standard UNO bootloader).

2. Atmega 328 system programmer. This is designed to be hosted by the user device and be used to program the flash of the pcb_A device with the mini-OS and the EEPROM with text and numbers.

3. Mini_OS. The display controller and I2C master reside in the application partition of the pcb_A device. The PCB-bootloader resides in the boot partition of the OS device. Its main function is to overwrite the original UNO bootloader with the Hex/text bootloader. After that it should only rarely be required.

Reset Options: Resetting the UNO device with the pcb reset switch

Reset switches are provided on both the UNO and pcb_A.
Once the mini_OS and hex/text programmer have both been loaded the operation of the reset switches is as follows:

For the UNO
Reset takes program control to the “Hex/text bootloader” which generates the following user prompt

h/t/r h/t/r h/t/r h/t/r.....

The bootloader responds to keypresses as follows:

'h' A hex file is requested.
't' A text file is requested
'r' The hex file is allowed to run

For pcb_A
A single click of the reset switch takes program control to the display controller and the user application on the UNO device. (If no user application has been loaded the UNO must be reset.)

A double click of the rest switch results in the h/t/r h/t/r.....user prompt. This time however an additional keypress is available:

'p' If this is selected the pcb bootloader is launched.
This generates the following user prompt p/r p/r p/r.....
which responds to keypresses as follows:
'p' The UNO device can be reprogrammed after which control passes back to the hex/text bootloader
'r' Control passes straight back to the user application.

Setting up the UNO and pcb_A

As delivered the UNO already has a bootloader.
This is used to load the 328 system programmer.
This in turn loads the mini-OS
The pcb bootloader is then launched and used to replace the UNO bootloader with the hex/text bootloader. (There will be no user prompt but a 'p' keypress will launch the pcb bootloader.)
Use applications can now be launched.

Issues
Three issues are noted as a result of which the system can latch up or one of the devices refuse to work as designed.

The main issue is that:
The TX port of the UNO and pcb_A are permanently connected together. (For PCB 111000 a DPDT switch was used to select with one would be connected to the UDSB bridge.) This means that care must be taken to ensure that the UART of only one device is ever active at any one time.

When user applications are running this is not a problem because the display driver does not use its UART.
When the 328 system programmer is running it can be a problem however especially if inappropriate code is loaded onto the pcb_A device. This is largely resolve by using a “weak reset”.

The user can hold the device in reset but the programmer can overcome this reset and remove the inappropriate code.

The second issue is that both UNO and pcb_A employ Atmega 328 devices. (PCB 111000 employed both 328 and 168 devices). It is therefore easy to get confused about which device is running especially at system setup time. However the UNO uses an external 16Mhz resonator and pcb_A the internal 8Mhz RC clock.

This fact is used to distinguish between them. The system programmer can be loaded onto either device. However it will check the configuration bytes of the device it is requested to program and refuse if it is the UNO.

Finally, when using the system programmer or pcb bootloader, having pressed 'p' the target device is immediately erased. A modification ensures that it is not erased until the first character of the new hex file has been received.

As the result of these steps it is hoped that unintended operation, or corruption or loss of system code can be avoided. However in the last resort it will always be possible to reload the original Arduino code.

Why interfere with the Arduino code

The UNO comes with a very good bootloader. The idea here however is to combine the UNO with PCB 111000_UNO to create a new product that will replace the original PCB 111000 and be very easy to assemble.

The benefit of having access to programmer code is that each use of the programmer is tailored to an individual requirement. Hopefully this should enables the UNO plus pcb_A to perform as a single product rather than the combination of two quite different products.