Floating point display for the UNO

A floating point display for the UNO has been developed. A system diagram is shown over the page and a photograph below. Two ATtiny devices are used to drive the display and the I2C bus is used to connect them to the UNO.

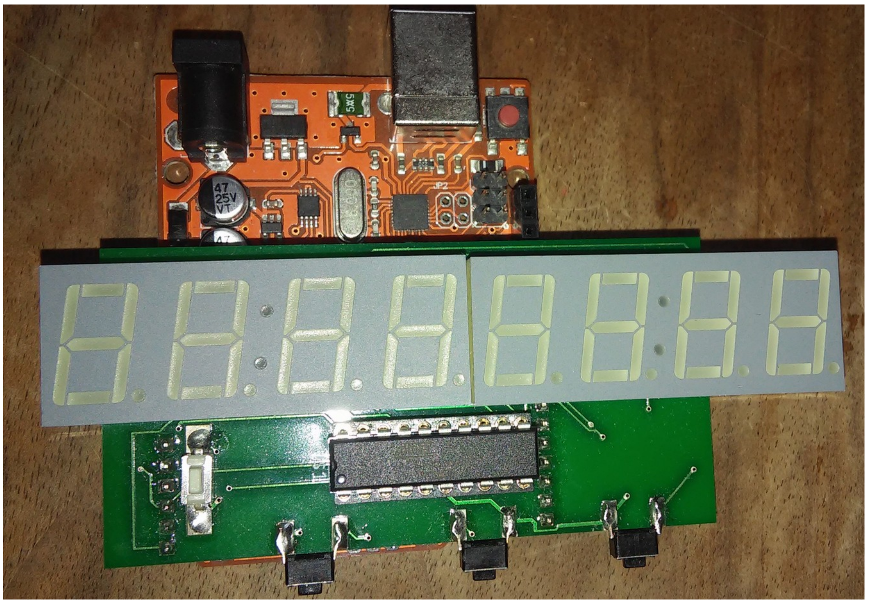
Whenever the UNO has new data to display it connects itself to the TWI/I2C bus. The master interface which polls the UNO every mS:

- Takes the data
- Converts it to a string if necessary
- Displays the lower four digits
- Forwards the upper digits to the other device

Four data types are recognised:

- Floating point strings — Integer strings
- Floating point data — Integer data

When the master device detects a string it converts it to a number and returns the result to the UNO. When it detects a number it simply displays it.



Photograph of the display plugged into a UNO.

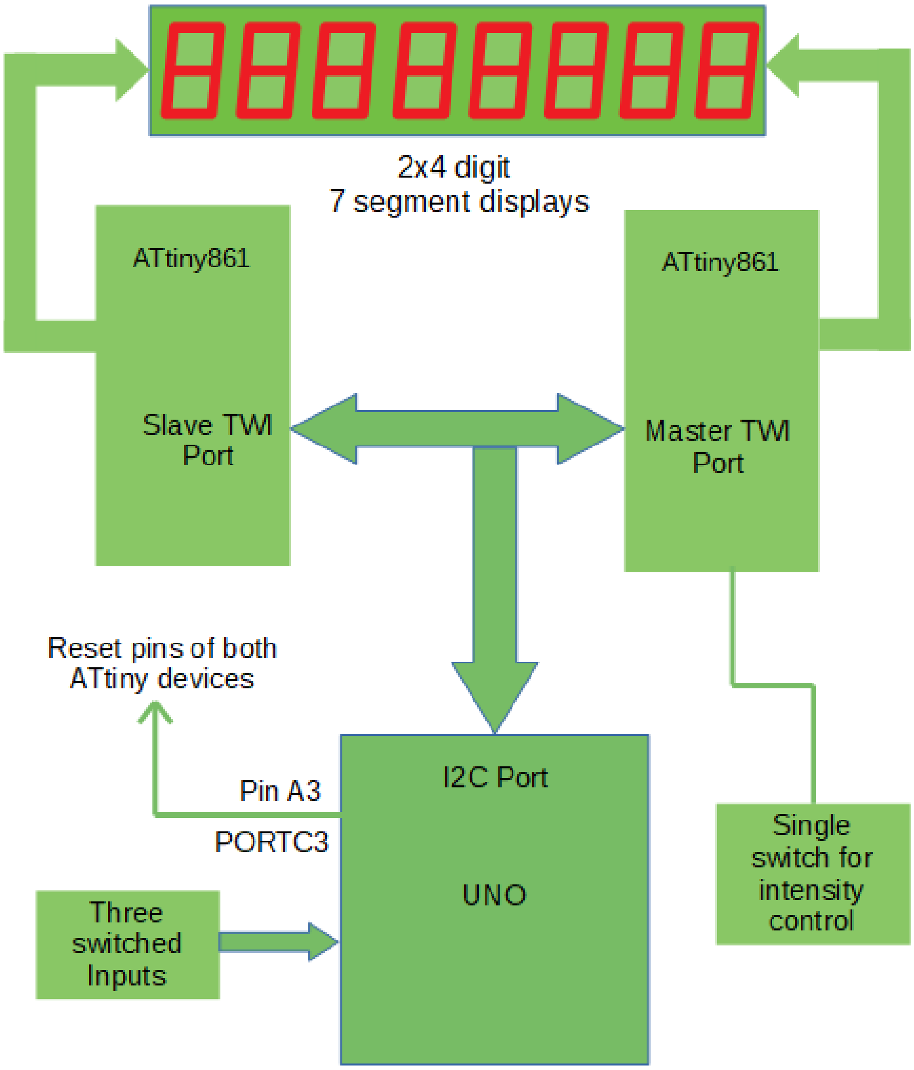
The vertical switch is used to select one of two intensity levels and control the display of the exponent.

The row of three IO switches (SW1 to SW3 from left to right) is for entering data.

A UNO template is provided which has been developed to test the display. It accepts numbers, does some simple arithmetic and includes the following resources:

- Askii_subroutines.c file — UNO_TWI_subroutines.c file
- PCI_subroutines.c file (pin change interrupt) to read the IO switches
- A header file contains a "setup_328_HW" macro which
  - Initialises the IO — Sets up the watchdog — Resets the display devices
  - Sets the UNO address — Checks for overflow

It is anticipated that users will copy and paste from the template for their applications as required.



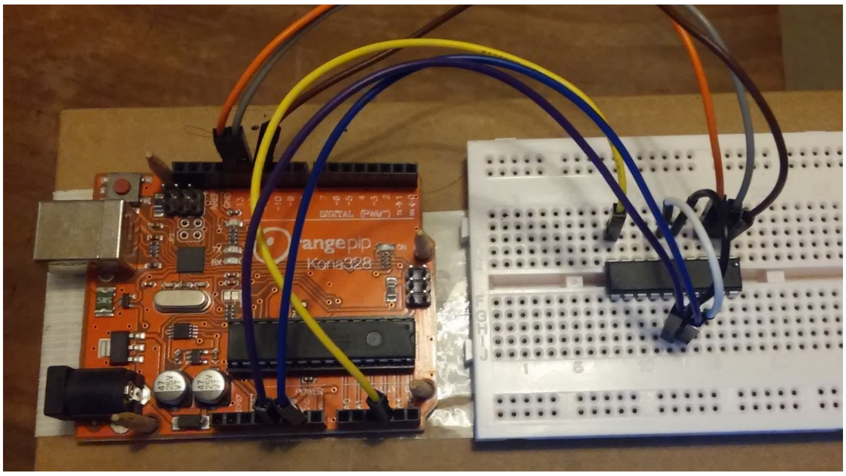
Block diagram of UNO floating point display
Connected to the UNO

The three switched inputs are for entering
numeric data or for general purpose use.

Note the slave device (the one on top of the pcb) can be either an ATtiny 861 or 461.

Programming the ATtiny devices

Code for these devices has been developed using Atmel Studio 7 and uploaded to them using UNO_AVR_Programmer_V2 posted on AVR_freaks projects on Wednesday, 21 October 2020. See also UNO_AVR_Programmer posted on the 9th September 2020 for an introduction to the programmer

<table>
<tr><td colspan="2">
</td><td>Programming the ATtiny devices.

Load Arduino sketch UNO_AVR_Programmer_V2 onto the UNO.
Connect the target device as given below.

Open the terminal program.
Click connect then press ‘s’ followed by ‘P’ and upload the hex file.</td></tr>
<tr><td>UNO pin</td><td></td><td>ATtiny 861/461 pin</td></tr>
<tr><td>5V</td><td>Brown wire</td><td>5, 15</td></tr>
<tr><td>0V</td><td>Blue wire</td><td>6, 16</td></tr>
<tr><td>A3</td><td>Yellow wire</td><td>10 Reset</td></tr>
<tr><td>Digital 11</td><td>Brown wire</td><td>1 MOSI</td></tr>
<tr><td>Digital 12</td><td>Grey wire</td><td>2 MISO</td></tr>
<tr><td>Digital 13</td><td>Orange wire</td><td>3 SCK</td></tr>
</table>

Note:
I recommend Br@y++ version 20130820, for the terminal program. It can be downloaded from https://sites.google.com/site/terminalbpp/. Settings for the terminal program are: Baud rate: 38400, 8 data bits, no parity, 1 stop bit and no handshaking.

The UNO_AVR programmer should automatically detected the ATtiny device.

When uploading the hex files press ‘P’ rather than ‘p’ to ensure that the configuration bytes are also programmed. These are 0xFF, 0xD7, 0E2 and 0xFF for extended, high and low and lock fuse bytes and are default values for the programmer. Important parameters are WDT under program control and internal RC clock.

The ATtiny devices use their internal RC clocks. The UNO_AVR programmer can be used to calibrated these clocks. However the devices will not be talking directly to a PC. They communicate with each other and the UNO via the I2C bus. Calibration should not therefore be necessary. During tests the clocks were not calibrated.

Both devices should be programmed prior to assembly of the display pcb.

The hex files: The display pcb contains two devices, a slave I2C device on the top face and a master I2C device on the bottom face. Both devices can be ATtiny861's but the slave device can also be the smaller ATtiny 461.
Six hex files are provided, 3 for use with common cathode displays and three for use with common anode displays. Two files are provided for use with the slave device, one assuming use of an ATtiny861 and one for the ATtiny461.

Operating Instructions These instructions apply to the UNO template that is supplied as part of this posting. The purpose of the template is to:

demonstrate the operation of the display PCB
provide a starting point for user applications

Getting started Open the UNO template, upload it to a UNO pcb, then plug-in the display pcb. Open the terminal program set the baud rate to 38400 and click on the "Rescan" and "Connect" buttons. User prompt F/I should be generated.

Entering data from the keyboard Press F to enter data from the keyboard and follow instructions. Use carriage return to terminate number entry before entering +,- * etc. to do the arithmetic. If overflow occurs the display will flash. Press SW3 to stabilise it and the user prompt will be generated. Press F either before or after releasing SW3 to restore the display with one of the two most recent results. Press I at the user prompt to enter integer numbers, then press any key to do some simple arithmetic.

Entering data from the user switches (IO data entry)

A floating point number
After a reset press and hold SW1 down until zero is displayed, then release it.
Press SW1 again and the display will scroll through 0 to 9 and -.
Release SW1 when the required symbol is being displayed
Pulse SW2 to shift the display one place.
Press SW1 again to select the next digit.
If a decimal point is required press SW3 while SW1 is being pressed.
Release SW1 before SW3.
When data entry is complete pulse SW3.

An arithmetic operation
Press SW3 and then pulse SW1
once for addition, twice for subtraction
three times for multiplication, and four for division
five for power and six to exit
Release SW3 before SW1 is pulsed for the last time
Zero should now be displayed
Enter the second number and the result should be displayed

Number entry can be cleared if necessary by pressing SW3 while SW2 is down.
A flashing display due to overflow can be stabilised by pressing SW3.
Then press SW1 before or after releasing SW3 to continue doing arithmetic

Swapping between IO and keyboard data entry

The template saves the two most recent arithmetic results to the EEPROM of UNO device.
Resetting the UNO while SW3 is held down recovers the latest and generates the user prompt. SW3 can be released either before or after users continue with IO or keyboard data entry depending on which result is required.
Note:
Rounding is applied on the display pcb however it is not applied in the UNO template.
Underflow is dealt with by displaying 0.0 and resetting the floating point number to 0x1 (~1e-45).
The software does not deal with an illegal request such as the square root of -10 (negative numbers can only be raised to an integer power). The answer will be garbage and the EEPROM will also be corrupted.
To enter integers press SW2 following a reset.

Setting the intensity: There are two intensity settings.
To change the intensity reset the UNO while pressing the vertical switch.

Swapping between devices

When recompiling the TWI master and slave files there are two options for the display and two for the slave device.
All changes are handled in the “Resources/Display_header.h” file.
Each file contains two sections one for CC and one for CA devices
The slave file also contains two lines, one for the 861 and one for the 461 device.

Links and downloads

The following on-line resources have been found invaluable:

For developing the master TWI subroutines:
http://edge.rit.edu/content/P14254/public/Design/Electronics/Controller/Code/Atmel %20Application%20Notes
Click on I2C AVR310 GCC AVR310 Using the USI module as a TWI master

For developing the slave TWI subroutines:
https://www.microchip.com/wwwAppNotes/AppNotes.aspx?appnote=en591197

For the Floating point to ASKII conversion:
https://www.geeksforgeeks.org/convert-floating-point-number-string/

For developing C files
https://www.pnotepad.org/download/ The portable edition of programmers notepad is ideal for opening the C-files required by the UNO template.

For use as a terminal program
https://sites.google.com/site/terminalbpp/ For the excellent terminal program Br@y++ version 20130820.

Generating the Gerber files

The Eagle pcb design tool must first be downloaded. A free version is available from Autodesk which can be used to process two layer boards of a limited size.

Place the Eagle board and schematic files in any suitable folder (to be referred to as the Project directory).
Open the Eagle control panel
Click on CAM Jobs then examples then example_2_layer.cam
Tick the Export to Project Directory box
Click on Select board
Navigate to the project directory and open the board file
Click on Process Job the Open folder
A CAMOutputs folder should have been added to the project directory. This will contain the Gerber and drill files needed to get the pcb manufactured